Implementation of Alg1 and Alg2 Path Planning Algorithms for Mobile Robots Using ROS Noetic

Anastasia Yankova
HSE University, Moscow, Russian Federation

Timur Gamberov
Intelligent Robotics Department, Kazan Federal University, 420008, Kazan, Russian Federation

Tatyana Tsoy
Intelligent Robotics Department, Kazan Federal University, 420008, Kazan, Russian Federation
E-mail: aayankova@edu.hse.ru, TRGamberov@stud.kpfu.ru, tt@it.kfu.ru

Abstract

Two standard approaches for a robot path planning include a global and a local navigation. The later does not require to store an environment model in a robot memory. This paper presents implementations of two local navigation algorithms, Alg1 and Alg2, with a robot having no prior information about an environment and obstacles. It calculates a path in a real time, continuously changing its states depending on correspondent conditions. The algorithms were implemented for an existing differential drive robot Turtlebot3 Burger using Robot Operation System (ROS). Virtual experiments were performed in the Gazebo simulator employing a simple 3D environment with only convex obstacles and a small 3D maze.

Keywords: path planning, mobile robots, local navigation, Alg1, Alg2, Gazebo, ROS Noetic

1. Introduction

Mobile robots and autonomous navigation are gradually integrated in various aspects of human activities, from occasional operations in dangerous scenarios to daily social interactions. The former include firefighting services [1], [2], urban search and rescue operations [3], [4], and special military operations [5], while the later improve processes in education [6], [7], manufacturing [8], [9], medicine [10], [11], agriculture [12], [13], rehabilitation [14] and service tasks [15], [16]. These tasks require advanced sensory-based autonomous navigation in various environments [17], [18]. Autonomous navigation allows a robot to decide on its motion and actions, based on onboard sensory data about its environmental and current location [19].

There are two types of path-planning: a global navigation and a local navigation [20]. In the global planning approach, a mobile robot has a well-defined map of an environment in which the robot can build its path. In more realistic settings, a robot deals with uncertain maps and relies on its sensors to plan a path [21]. which is called the local navigation. In the local planning approach a robot is being placed at a starting position and must reach a target or report if it cannot be reached while no other information is known to the robot in advance [22]. In this case, the robot uses local sensory data [20] to detect obstacles within its radius of vision [23], which allows the robot to encounter an obstacle only when it hits the obstacle in most algorithms [24].

Boundary-following and Ultimate Goal (BUG) family algorithms were designed to solve the local navigation problem without generating a full map of an environment [22]. Following a BUG family algorithm, a robot could operate in a broad variety of environments [25] and (by an algorithm design) attempts to construct a shortest path toward its destination [26]. BUG algorithms have two modes of motion: moving towards a target and following an obstacle boundary. A robot goes towards the target until it hits any obstacle. Then it starts to follow the boundary until a straight path towards the target becomes clear again [27]. A condition that defines if the path is clear differs depending on a particular algorithm.

In BUG model a robot is considered as a point object [22] in a 2D-map. In this paper a path-planning sensory-based navigation was simulated in a 3D-environment with a real mobile robot model, which employs Alg1 [28] and Alg2 [29] algorithms. The algorithms were implemented using robot operating system (ROS) [30] and evaluated using Gazebo simulator [31].

2. Brief description of the algorithms

Alg1 and Alg2 algorithms belong to BUG family. Two states of a robot under BUG strategy are: 1) go to the target; 2) follow the boundary. Most algorithms employ a line from start point to target, which is called *M-line*. The robot switches from state 1 to state 2 when it hits an obstacle at a point that is defined as a *Hit-point* (*H-point*). The switch from state 1 to state 2 occurs when the robot decides to abandon the current obstacle at a point that is defined as a *Leave-point* (*L-point*). Both H-point and L-point are defined differently by particular algorithms.

2.1. Alg1 algorithm

Alg1 algorithm improves basic Bug2 algorithm [24] in a sense of excluding multiple traces of long segments of a path. It collects H-points (H_i) and L-points (L_j) of previous iterations and uses this information to generate shorter paths by changing a local direction to the opposite. The algorithm works as follows [32]:

0) Initialize iteration i to 0, define *M-line* as line connecting start S and target T points.
1) Increment i and follow *M-line* toward T until either:
   - T is reached. Stop.
   - An obstacle is hit. Define this point as H_i. Go to step 2.
2) Keeping the obstacle on the right, follow the obstacle boundary. Do this until one of the following occurs:
   - T is reached. Stop.
   - A point y is found such that:
     - it is on *M-line* and
     - $d(y, T) < d(x, T)$ for all x ever visited by the robot along *M-line* and
     - The robot can move towards T at y.

   Define this point as L_i and go to step 1.
   - A previously defined point H_j or L_j is encountered such that $j<i$. Change the local direction one and return to H_i. When H_i is reached, follow the obstacle boundary keeping the wall on the left. This rule cannot be applied again until L_i is defined.
   - The robot returns to H_i. T is unreachable. Stop.

2.2. Alg2 algorithm

Alg2 algorithm upgrades Alg1 algorithm by abandoning *M-line* concept. The leaving condition is that the robot became closer to T than before. The algorithm operates as follows [30]:

0) Initialize iteration i to 0 and $Q = d(S, T)$.
1) Increment i and proceed in the direction of T whilst continuously updating Q to $d(x, T)$ *if* $Q < d(x, T)$, where x is a current position. Q should now represent the closest to T point where the robot has ever been. Repeat this until one of the following occurs:
   - T is reached. Stop.
   - An obstacle is hit. Define this point H_i. Go to step 2.
2) Keeping the obstacle on the right, follow the obstacle boundary continuously updating Q to $d(x, T)$ if $Q < d(x, T)$. Do this until one of the following occurs:
   - T is reached. Stop.
   - A point y is found such that:
     - $d(y, T) < Q$ and
     - The robot can move towards T at y.

   Define this point as L_i and go to step 1.
   - A previously defined point H_j or L_j is encountered such that $j<i$. Change the local direction and return to H_i. When H_i is reached, follow the obstacle boundary keeping the wall on the left. This rule cannot be applied again until L_i is defined.
   - The robot returns to H_i. T is unreachable. Stop.

3. Implementation details

Ubuntu operating system and ROS Noetic Ninjemys were used. Gazebo simulator was employed for debugging and verifying algorithms' implementation with Turtlebot3 Burger robot model (Fig. 1) from an open source software kit [33].

Alg1 and Alg2 were implemented in Python3 programming language and arranged as a package with two files, containing Alg1 and Alg2 respectively. A main service uses services for going to a point and a wall following in a clockwise and counter clock-wise order.

The following libraries and modules were used for all the services:

- *rospy* – a pure Python client library for ROS;
- *geometry_msgs* module used for generating and sending messages for setting robot's position;
- *sensor_msgs* module to register laser range finder's measurements;
- *gazebo_msgs* module to define and set a robot's current state;
- *tf* module for angles' operations;
- *nav_msgs* module for odometry;
- *std_srv* module for *ros services*.

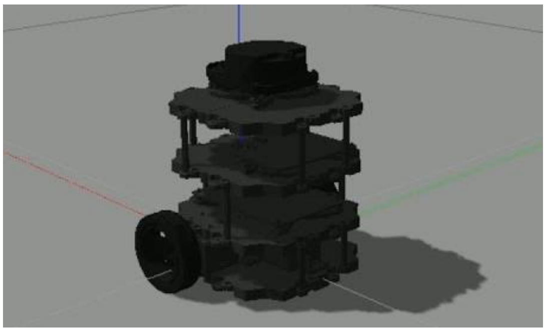


Fig. 1. Turtlebot3 Burger in the Gazebo environment

A main difficulty that was encountered while implementing the algorithms was a definition of suitable constants that allow comparing distance measurements. For example, one of the problems was to define a threshold ε>0 that could be employed to define two distinct robot positions. This way we define that if a Euclidean distance *Dist* between two points p_k and p_m is less than ε, it means that the two points coincide with each other:

$$Dist(p_k,p_m) \leq \varepsilon \iff p_k = p_m \quad (1)$$

A value of ε was chosen empirically to fit all types of environments.

Another problem relates to the construction of the robot that causes stuck because the robot sensor does not consider the robot's wheels, which poke out from the mobile base. In some cases, the robot does not register an obstacle when its wheels already have hit a convex corner. A possible solution is an increase of a threshold δ that helps defining a *H-point* as:

$$Dist(p_k,p_{obs}) \leq \delta \Rightarrow p_{obs} = H_i \quad (2)$$

where p_{obs} is a point on a boundary of a currently hit obstacle, p_k is a current position of the robot, H_i is a newly defined *H*-point. Yet, this may cause the algorithms' failure since the robot may miss a *H-point* and thus skips a switch of its state into the boundary following mode.

Finally, one more difficulty appeared only for Alg2 algorithm at step 1 when it checks whether the robot became closer to *T* than before ($Q < d(x, T)$). While in a mathematical sense this comparison is performed constantly, in practice a particular small time step Δt between checking a new value of *x* should be selected. The value of Δt was found empirically so that it successfully works for both employed types of maps: separate convex obstacles and mazes.

4. Validation

To validate the implementation of Alg1 and Alg2 algorithms 45 tests within two different types of environments were conducted. While there exist several popular tools for environment construction for Gazebo worlds, ranging from semiautonomous generators of single environments from 2D images [18],[34] to autonomous generators of multiple environments, including maze-like environments [35] and random step environment generators [36] using ergonomic graphical user interfaces, it was decided to construct two environments manually in order to ensure interesting cases for algorithms' testing. The first constructed environment was bounded by an external non-traversable black wall with five green towers and contained separate convex obstacles in a form of nine identical columns. The second environment was a maze. 25 and 20 test cases within the convex environment and the maze were conducted, correspondingly. For each test cases four runs were conducted. In total, 90,5% of the tests were successful while seven tests in the convex environment and ten tests in the maze failed due to the problems that were stated in Section 3. Additional tests for the target reachability were performed successfully in both environment. Table 1 presents the system configuration.

Table 1. System configuration

Parameter	*Characteristics*
Operation System details	Ubuntu 20.0.4
Memory	8.00 GB
Processor	Intel(R) Core (TM) i7-4510U CPU @ 2.00GHz 2.60 GHz

Fig. 2 presents a path constructed by Alg1 algorithm in the convex environment. Due to the environment's simplicity, a path of Alg2 algorithm was almost identical. Fig. 3 and Fig. 4 demonstrate paths with the same *S*, *T* locations that were constructed in the maze environment by Alg1 and Alg2 respectively. In this particular case Alg2 outperformed Alg1; it is wrong to state that Alg2 always outperforms Alg1 in every case as a result depends on the *S*, *T* and the environment selection.

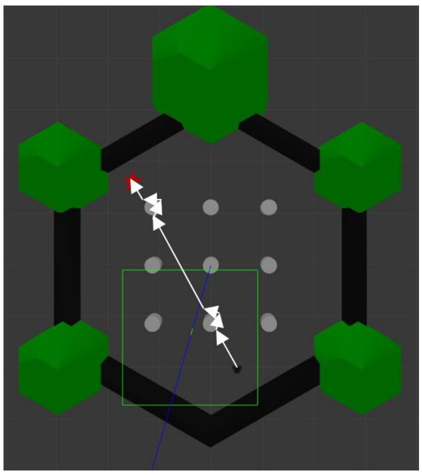


Fig. 2. A path of Alg1 in the convex environment is shown by white lines. The arrows depict the direction of motion, the red star marks the target.

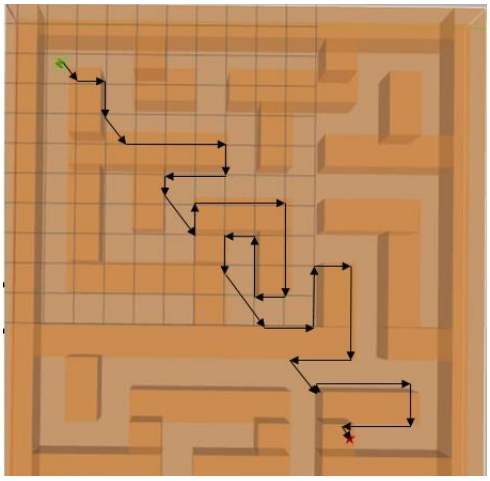


Fig. 3. A path of Alg1 in the maze

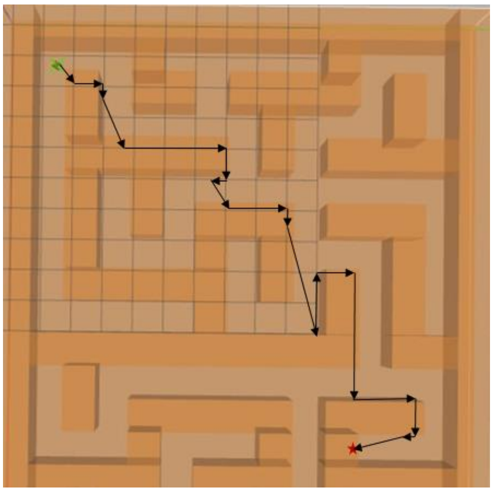


Fig. 4. A path of Alg2 in the maze

5. Conclusions

The paper presented the implementation of Alg1 and Alg2 algorithms in Python programming language for Turtlebot3 Burger robot model using ROS Noetic. The conducted tests successfully validated the implemented algorithms in the simple convex environment and in the maze. A number of difficulties that were encountered while implementing the algorithms are discussed.

Acknowledgment

This paper has been supported by the Kazan Federal University Strategic Academic Leadership Program (“PRIORITY-2030”).

References

1. J. Zhu, W. Li, D. Lin, H. Cheng and G. Zhao, Intelligent fire monitor for fire robot based on infrared image feedback control. Fire Technology, vol. 56, no. 5, 2020, pp. 2089–2109.
2. R. Bogue, The role of robots in firefighting. Industrial Robot: the international journal of robotics research and application, 48(2), 2021, pp.174-178.
3. E. Magid, K. Ozawa, T. Tsubouchi, E. Koyanagi and T. Yoshida, Rescue Robot Navigation: Static Stability Estimation in Random Step Environment. Lecture Notes in Computer Science, vol. 5325, 2008, p. 305-316.
4. E. Magid, Simulation Tools for Urban Search and Rescue Robotics. International Conference on Artificial Life and Robotics (ICAROB 2023), 2023, pp. 4-11.
5. C. Chen, S. Wang, L. Li, S. Ke, C. Wang and X. Bu, Intelligent covert satellite communication for military robot swarm. IEEE Access, vol. 8, 2019, pp. 5363–5382.
6. T. Tsoy, L. Sabirova, M. Abramsky and E. Magid, Establishing effective teaching for robotics: a comparison study of bachelor students participated in introduction to robotics course. In Int. Conf. on Artificial ALife and Robotics, 2018, pp. 212–215.
7. E. Chebotareva and M. Mustafin, Android Based Educational Mobile Robot Design and Pilot Evaluations. International Conference on Artificial Life and Robotics (ICAROB 2023), 2023, pp. 146-149.
8. A. Dobrokvashina, S. Sulaiman, A. Zagirov, E. Chebotareva, K.-H. Hsia and E. Magid, Human robot interaction in collaborative manufacturing scenarios: Prospective cases. Siberian Conference on Control and Communications, (SIBCON 2022), November 2022, p. 1-6.
9. R. Galin and R. Meshcheryakov, Automation and robotics in the context of Industry 4.0: the shift to collaborative robots. In IOP Conference Series: Materials Science and Engineering, vol. 537, no. 3, May 2019, p. 032073.
10. E. Magid, A. Zakiev, T. Tsoy, R. Lavrenov and A. Rizvanov, Automating pandemic mitigation. Advanced Robotics, vol. 35, no. 9, 2021, pp. 572–589.
11. R. Safin, R. Lavrenov, K.-H. Hsia, E. Maslak, N. Schiefermeier-Mach and E. Magid, Modelling a TurtleBot3 Based Delivery System for a Smart Hospital in Gazebo. The 15th Siberian Conference on Control and Communications (SIBCON 2021), 2021, № 9438875.
12. J. Samaniego, E. López-González, E. Magid, and E. A. Martínez-García, Path-Tracking in Double Spiraliform Sowing Fields with Agricultural Robotics. 2023, Available at SSRN 4589037.
13. M.S.A. Mahmud, M.S.Z. Abidin, A.A. Emmanuel and H.S. Hasan, Robotics and automation in agriculture: present and future applications. Applications of Modelling and Simulation, vol. 4, 2020, pp.130-140.
14. C. Eicher, M. Haesner, M. Spranger, O. Kuzmicheva, A. Gräser and E. Steinhagen-Thiessen, Usability and acceptability by a younger and older user group regarding a mobile robot-supported gait rehabilitation system. Assistive technology, 31(1), 2019, p. 25-33.
15. D. Ryumin, I. Kagirov, A. Axyonov, N. Pavlyuk, A. Saveliev, I. Kipyatkova, M. Zelezny, I. Mporas and A. Karpov, A multimodal user interface for an assistive robotic shopping cart. Electronics, vol. 9, no. 12, 2020, p. 2093.
16. E. Chebotareva, R. Safn, K.-H. Hsia, A. Carballo and E. Magid, Person-following algorithm based on laser range finder and monocular camera data fusion for a wheeled autonomous mobile robot, in International Conference on Interactive Collaborative Robotics (ICR 2020). Lecture Notes in Computer Science, vol 12336, 2020.
17. A. Eryomin, R. Safin, T. Tsoy, R. Lavrenov, and E. Magid, Optical Sensors Fusion Approaches for Map Construction: A Review of Recent Studies. In Journal of Robotics,

Networking and Artificial Life, vol. 10(2), 2023, pp. 127-130.

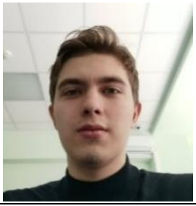
18. R. Lavrenov, A. Zakiev and E. Magid, Automatic mapping and filtering tool: From a sensor-based occupancy grid to a 3D Gazebo octomap. International Conference on Mechanical, System and Control Engineering (ICMSC), 2017, pp. 190-195.
19. M. Mustafin, T. Tsoy, E.A. Martınez-Garcıa, R. Meshcheryakov and E. Magid, Modelling mobile robot navigation in 3D environments: camera-based stairs recognition in Gazebo. Moscow Workshop on Electronic and Networking Technologies (MWENT-2022), 2022, pp. 1-6.
20. E. Magid and E. Rivlin, CAUTIOUSBUG: A Competitive Algorithm for Sensory-Based Robot Navigation. Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003, pp. 2757-2762.
21. I. Kamon and E. Rivlin, Sensory-based motion planning with global proofs. IEEE Transactions on Robotics and Automation, vol.13, 1997, pp. 814-822.
22. N. James and T. Bräunl, Performance comparison of bug navigation algorithms. Journal of Intelligent and Robotic Systems 50, no. 1, 2007, pp. 73- 84.
23. V. Lumelsky and T. Skewis, Incorporating range sensing in the robot navigation function. IEEE Transactions on Systems, Man, and Cybernetics, 1990, pp. 1058-1069.
24. V. Lumelsky and A. Stepanov, Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. In Autonomous robot vehicles, 1990, pp. 363-390.
25. H. Noborio, Evaluation of Path Length Made in SensorBased Path-Planning with the Alternative Following. Proceedings of the 2001 IEEE International Conference on Robotics 8 Automation, 2001, pp. 1728-1735.
26. H. Noborio, Several Path-Planning Algorithms of a Mobile Robot for an Uncertain Workspace and their Evaluation. Proc. of the IEEE Intelligent Motion Contro, 1990, pp. 289-294.
27. A. Yufka and O. Parlaktuna Performance comparison of bug algorithms for mobile robots. In Proceedings of the 5th international advanced technologies symposium, 2009, pp. 13-15.
28. A. Sankaranarayanan and M. Vidyasagar, A New Path Planning Algorithm For Moving A Point Object Amidst Unknown Obstacles In A Plane. IEEE Conference on Robotics and Automation, 1990, pp. 1930-1936.
29. A. Sankaranarayanan and M. Vidyasagar, Path Planning For Moving A Point Object Amidst Unknown Obstacles In A Plane: A New Algorithm And A General Theory For Algorithm Development. Proc. of the 29th Conference on Decision and Control, 1991, pp. 1111-1119.
30. A. Apurin, B. Abbyasov, E.A.Martínez-García and E. Magid Comparison of ROS Local Planners for a Holonomic Robot in Gazebo Simulator. In International Conference on Interactive Collaborative Robotics, Cham: Springer Nature Switzerland, 2023, pp. 116-126.
31. A. Dobrokvashina, R. Lavrenov, E. Magid, Y. Bai and M. Svinin. How to Create a New Model of a Mobile Robot in ROS/Gazebo Environment: An Extended Tutorial. International Journal of Mechanical Engineering and Robotics Research, 12(4), 2023, pp. 192-199.
32. N. James and T. Bräunl, A Practical Comparison of Robot Path Planning Algorithms given only Local Information. Centre for Intelligent Information Processing Systems School of Electrical and Electronic Engineering The University of Western Australia, 2005.
33. F. U. Pereira, P. M. de A. Brasil, M. A. De S. L. Cuadros and A. R. Cukla, Analysis of Local Trajectory Planners for Mobile Robot with Robot Operating System. IEEE Latin America Transactions, vol. 20(1), 2022, pp. 92-99.
34. B. Abbyasov, R. Lavrenov, A. Zakiev, K. Yakovlev, M. Svinin and E. Magid, Automatic Tool for Gazebo World Construction: From a Grayscale Image to a 3D Solid Model. International Conference on Robotics and Automation (ICRA), 2020, p. 7226-7232.
35. A. Iskhakova, B. Abbyasov, T. Tsoy, T. Mironchuk, M. Svinin and E. Magid, LIRS-Mazegen: An Easy-to-Use Blender Extension for Modeling Maze-Like Environments for Gazebo Simulator. Frontiers in Robotics and Electromechanics, vol. 329, 2023, pp. 147-161.
36. R. Gabdrahmanov, T. Tsoy, Y. Bai, M. Svinin and E. Magid, Automatic Generation of Random Step Environment Models for Gazebo Simulator. Lecture Notes in Networks and Systems, vol. 324, 2021, p. 408-420.

Authors Introduction

Ms. Anastasia Yankova

Anastasia Yankova is a master student in System analysis and mathematical technologies at HSE University, Russia. Previously she obtained a bachelor degree in information systems at ITMO University, Russia.

Mr. Timur Gamberov



Timur Gamberov is a bachelor student at Institute of Information Technology and Intelligent Systems, Kazan Federal University, Russia.

Ms. Tatyana Tsoy

In 2012 she graduated from International Area Studies Master's Degree Program at the University of Tsukuba. In 2023 completed a PhD in Computer science and information processes at ITIS, KFU, Russia.