Knowledge sharing for Optimal path planning of Multi Robot systems in Dynamic maps

Srujan Panuganti
MEng of Robotics 2018-20
University of Maryland
College Park, USA
psrujan@terpmail.umd.edu

Varsha Eranki
MEng of Robotics 2018-20
University of Maryland
College Park, USA
evarsha@terpmail.umd.edu

Akshitha Pothamshetty
MEng of Robotics 2018-20
University of Maryland
College Park, USA
apothams@terpmail.umd.edu

***Abstract*—Multi-mobile robotic systems are a boon to large scale industries in implementing an efficient work flow or for mastering dynamic path planning when time is a constraint. They need to plan collision-free paths in physical environments and also consider various dynamic obstacles at remote areas, apart from the conventional set up of stationary obstacles. However, Robot perception limits the data to sensor range and robots are only aware of the obstacles within the range of its sensor. Also, the knowledge of interacting with new obstacles or dynamic obstacles observed by each robot is kept local to itself. In our project, we developed a data sharing architecture in which the robots share the information of new obstacles and blocked paths with each other under the same dynamic map implementing the SLAM(Simultaneous localization and Mapping) technique. This enables the multi-robots to update their map with the new and remote obstacles at different instincts, and re-plan efficient and optimal paths. Moreover, this eliminates the need for re-planning of paths in conventional navigation methods. Experimental results show that the proposed method is also productive for multi-robot navigation in large and dynamic maps.**

***Index Terms*—Multi-mobile robotic systems, path planning, collision-free paths, dynamic obstacles, Robot perception, data sharing architecture, dynamic maps.**

I. Introduction

Multi-robot systems have found their prominence in a wide domain of work flow management in large scale industries, warehouses and other navigational operations. Their application is mainly due to their low fault tolerances and parallel task management in a diverse environment where multiple operations are being simultaneously navigated to their respective destinations. All these specifications decrease the final output time by providing well coordinated pathways.

However, traditionally, in Multi-mobile robotic systems, each robot has to keep its own predefined map of the environment to use it for planning the path by SLAM technique. But in large and dynamic environments, planning the path to remote locations becomes challenging as the range of obstacle detection by the sensor is constrained to its respective robot. Moreover, robots may also consider each other to be obstacles while traversing their respective paths and delay the time period.

To overcome this issue, in this project we worked on a obstacle data sharing architecture, where each robot shares information about the new obstacles that are added into their path with the other robots. It broadcasts this data about the new obstacles and dynamic obstacles at their respective interaction time under one dynamic map. This shared knowledge about the new obstacles is then used in re-planning the path to the destination by each robot. This capability of sharing information improves the abilities of the system as there is no time delay for each robot to relay on its sensor data independently, but to relay on the complete obstacle data of all the robots collectively. This proposed scheme can bring significant performance gains in large and dynamic environments employing multiple service robots and can be well implemented in various labor intensive environments.

II. Previous work

Path planning has two main aspects to be considered, one is a collision free path and the other is the shortest path between the initial and final positions. Rapidly exploring Random tree star(RRT*) has been compared to A* and D* algorithms on the basis of its optimality and application [1]. However, we designed our architecture on the basis of implementing any path planning algorithm. Another approach of firstly determining the data structure of the motion of multi-robot systems by firstly constructing individual maps, then implementing the probabilistic path planner for feasibility conditions and then combining these maps [2], [12] to form a composite robot seemed very intricate and not flexible. In implementing automated guided vehicles for warehouse transportation [3], the solution was simulated using greedy algorithm like the Dijkstra and the heuristic was Manhattan distance. But this methodology was not optimal and showed lesser reduction in output time. The path planning algorithms and their respective pseudo codes [5] were deliberated and designed for their application on the proposed project and hence it was made flexible to take the input in the form of any path planning algorithm. In this project, a knowledge sharing architecture is implemented by applying the concept of a unique dynamic map among all the robots of the Multi-robot system [7]. In this architecture, when a robot encounters a new obstacle which is not in it's predefined map, it sends out a message to other robots, so that they update their maps with the new obstacle. This saves a considerable amount of planning and navigation time of the robots. The pose estimation of each obstacle

detected by the robot was executed using the extended Kalman filter [4]. In this work the probabilistic method, Extended Kalman Filter is used to estimate the current location of the robot along with the uncertainty matrix associated with the newly identified obstacle. Such architectures can be effectively implemented in applications like, ware-house [3] fork-lifts and in distributed robotics. Below figure 1 depicts how their architecture works in action. This architecture can be used along with any search algorithms like A*, D* [10], Rapidly exploring random trees [1] and Probabilistic road maps [9] to find the optimal path to the goal location. The bridging of Python script file to ROS for implementation and simulation [8], [9], [13] on Turtlebots was developed because of its ease and accuracy. As most of the previous developments and techniques talk mostly about path planning strategies to avoid collisions thereby finding an optimal path, the proposed work revolves around data sharing among the robots and the dynamic changes in the obstacle space for faster path planning.

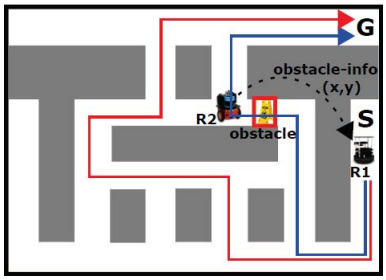

Fig. 1. Robot R2 finds a path blocked and shares this information with R1 enabling it to plan an optimal path shown in red color. Traditional method plan a path shown in blue color which requires re-planning.

III. Methodology

The concept of multi robot-systems comes into picture when multiple tasks are being carried out by multiple robots. These multiple robots to carry out the tasks need a road map of the environment around it to execute robot motion, avoid collisions and plan its path iteratively.

Since the range of the sensors such as laser sensors, ultrasonic sensors or tactile sensors is limited to individual robots, these robots cannot picture the path way ahead of them. They also cannot re-plan their individual paths unless they independently interact with the obstacle. The knowledge of obstacles and the map is kept local to the robot only. Hence, there is time loss in re-planning the paths only after each individual robot interacts with the obstacle even though the same obstacle was previously detected by another robot.

In our project we implemented a data sharing platform for multi-robots under one dynamic map. This centralized approach of robot path planning is beneficial for large maps with dynamic obstacles as well. When an individual robot interacts with an obstacle, the sensor detects the obstacle and broadcasts a change in the dynamic map shared among all the other robots of the system. This dynamic change added to the obstacle space is in the form of a six parameter message to the dynamic map giving timely information of the obstacle interaction for collision avoidance and efficient path planning for the other robots. The important assumptions to be made for this system are that all the robots are in the same network. A grid based dynamic map is made available for all the robots in the system. It is also simpler and convenient when all the robots are equipped with the same dynamic map. However, this did land us into a problem of different map scaling ratios and orientations observed by individual robots.

The message readings that are being shared from the sensors can be denoted by the following six parameters as:

$$M = \{1/0, x, y, \xi, w, b, t\}$$

where x and y represent the center of the obstacle, w and b are the width and breadth of the obstacle, t is the time at which the obstacle has been identified. The parameter ξ is uncertainty associated with the pose of the obstacle. The project has been worked using the laser sensor which determines the width and breadth of the obstacle along with its centre (x, y). Semi-algebraic Hough planes [5] are used to define the map of the environment. The state or pose of the robot with respect to the dynamic map can be estimated using any probabilistic filters like Practical filter or extended Kalman filter. The filter we have considered is the extended Kalman filter. It is always important to determine the pose uncertainty as the sensor range for defining the obstacle space is involuntarily prone to errors.

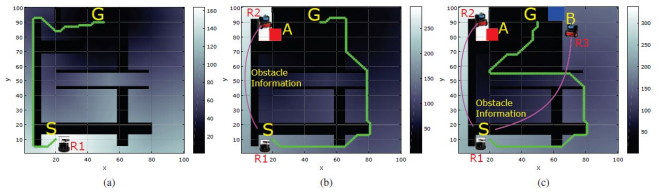

Fig. 2. D* planned paths from S to G. (a) Path without obstacles. (b) Path when knowledge of obstacle A is informed to robot. (c) Path when obstacle A and B are informed. The darker shades in the color map represents proximity to goal (G).

When determining the new obstacle update, uncertainty is thus an important aspect that should be taken into consideration to be modeled precisely.

For consideration and computation purposes, the i^{th} obstacle possesses an absolute location of (x^i_{obs}, y^i_{obs}) with uncertainty ξ^i_{obs}, also each robot has a certain identification number say R_i. The respective notations are also determined for the breadth and width of the obstacle to generate the message(M).

Another assumption taken into consideration while programming the path planning algorithm was all the dynamic obstacles are constrained to be in rectangular in shape. But however, when determined physically, the sensor predicts the exact dimensions of the obstacle in interaction. For sensors that are based on distance, it is difficult to estimate the accurate value of the dimensions, but the width is sufficient to determine if a path is traversable or blocked.

The first number value of the message denotes the adding and elimination of an obstacle in the workspace. (1 is when an obstacle is added to the map). This binary format is done for easier data handling of the obstacles from the workspace. The information about the uncertainty(ξ) is incorporated by inflating the centre of the obstacle with (ξ). The eigenvalue - eigenvector decomposition of the uncertainty matrix (ξ) gives the direction and scale of variance for a given obstacle detected.

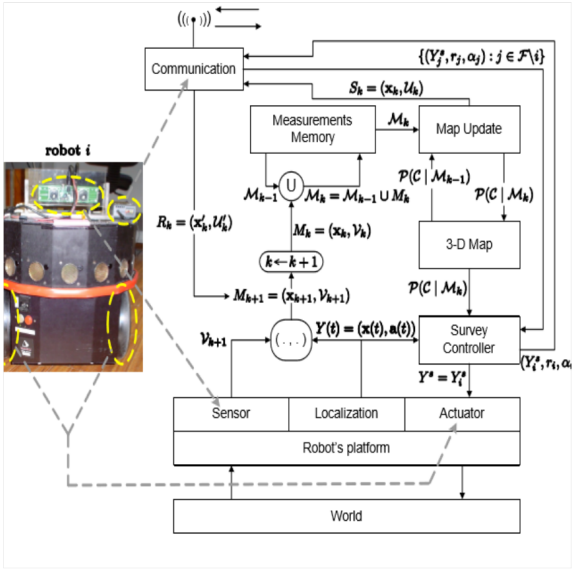

Fig. 3. The flowchart of the Architecture

The implementation of the data sharing architecture was done using the Robot Operating System (ROS) and the simulation environment is Gazebo. The search algorithm and workspace design are done in Python3 programming language. ROS packages are used to make the multiple robots [6] work together. In our approach, we initialized each robot as a node [8] , given it's own map and search algorithm. We also created a ROS topic on to which these nodes will be publishing messages about changes in obstacles. The section below lists the details of the high-level pseudo code of our approach.

A. Pseudo Code

1) main()
2) import classes
3) initialize Robots R_1, R_2, R_3, R_4, R_5 with map M, Initialize the search algorithm for each robot respectively as S_1, S_2, S_3, S_4, S_5
4) Create nodes N_1, N_2, N_3, N_4, N_5 for robots R_1, R_2, R_3, R_4, R_5
5) Create a ROS topic update_message with format

$$M = \{1, x, y, \xi, w, b, t\}$$

6) Create a subscriber and publisher to update_message for each node with *'/mobile_base/commands/velocity'* to publish velocities.
7) Function mover( ): #for each node
   for velocity in velocity_list:
   while rospy.time.now( ) $\leq$ end_time:
   publish(M)
   publish(velocities)
   if subscriber.message[0] == 1:
   map.update_map(M[1],M[2],M[3],M[4])
   search.replan()
   else if subscriber.message[0]==0:
   map.update_map_remove(M[1],M[2],M[3],M[4])
   search.replan()
   else:
   return nothing

B. Programming using Python

Since we wish the multi-robot system to work under any path planning algorithm, we implemented A* and Dijkstra for the present scenario. For easier computation purposes, we implemented four classes for code as follows:

1) Map Class: Takes the input of map_size, robot_radius, clearance, resolution.

2) Robot Class: Robot class is initialized with the Map, resolution, robot_radius, rpms, length, radius of the wheels. This class has the function to create the actions set for succeeding nodes.

3) Search Class: This class is initialized with all the lists and dictionaries along with several functions used to find the path to goal. This class is initialized with the start_position, goal_position, map, robot, initial_theta. This class has the function A_star( ) which return the path and the velocities for the supplied robot.

4) Visualize Class: This class is created to visualize the process. This is initialized with the start_position, goal_position and map_size. This class has a function display_map( ), which takes the node_path, visited_nodes and obstacle_set to visualize the whole process.

C. Simulation using ROS

In the main function ($main()$) we declare all the robots with their respective map and search algorithm. We declared each robot as a node.

1) ROS node: Each node is a declared with a robot which subscribes and publish on to the topic that is explained below.
2) ROS topic update_message:
   a) We create a ROS topic called update_message which is of the format $M = 1, x, y, \xi, w, b, t$. Here the first element 1 indicates that the obstacle needs

to be added as stated above. ξ is the uncertainty matrix with covariance and eigen-values.

b) Every node which is initialized with the robot and its map subscribes to this topic and publish messages regularly, whenever obstacles obstacles which are not in their map are sensed by the laser_scan sensor, the robot publishes on to the update_message topic.

3) Every robot which subscribes to the topic, receives the message, if the new obstacle points are not in their map, it updates its map and re-plans its path which are declared in the first step.
4) ROS simulation: We initialized the map spawning all the robots(turtle_bots) initialized as nodes. All the robots share the same map and communicate each other to share knowledge about new obstacles and find the optimal path with efficient re-planning compared to traditional methods.

IV. Results

To determine the results we first tested our multi-robot system considering the robots individually. We checked for their respective traversable paths, and then proceeded to combine them for their collective operation. The A* search algorithm for the multiple robots was successfully implemented. Depending on the start goal and end goal given to each robot, the average time the search algorithm consumes to find the path for each robot was approximately 3 seconds.

The following plots have been determined for the robots to generate the paths in the workspace of $10x11m^2$.

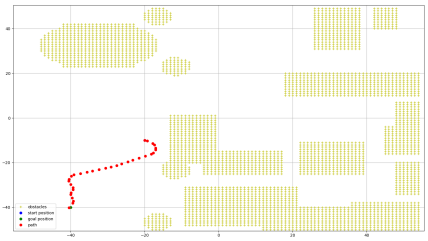

Fig. 4. The path generated by Robot-1 from initial_node(-20,-10) to goal_node(40,-40)

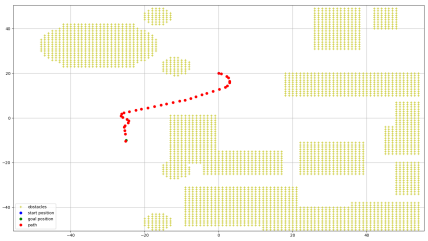

Fig. 5. The path generated by Robot-2 from initial_node(0,20) to goal_node(-25,-10)

From the above two plots for the first two robots, we can depict that the path traversed by each of the robot is not the same as the other. Moreover, the system does not commute the same point at the same time for the robots thereby eliminating any chances of collision. Moreover, the paths generated are traced by determining the velocities of the wheels of the robot on either side. So the given plots are calculated for speeds of two speeds of $(4rpm, 5rpm)$. So the final path obtained is based on the combinations of these two velocities to generate the path and the orientation is determined iteratively for path nodes generated.

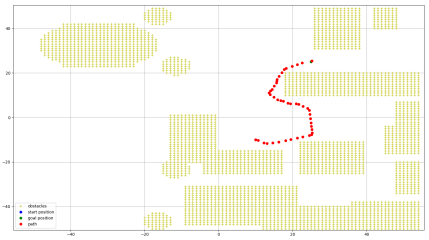

Fig. 6. The path generated by Robot-3 from initial_node(10,-10) to goal_node(25,25)

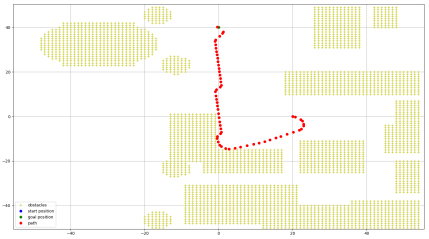

Fig. 7. The path generated by Robot-4 from initial_node(20,0) to goal_node(0,-40)

The below robots (R_3, R_4) calculate their path without collision as well. These paths are calculated in the same way to eliminate any chances of collisions between the robots. One main assumption that is considered is that the robots are taken to be rigid robots with clearance $0.1m$ summed up to the robot radius (In this context since the multi-robot system consists of turtlebots, the radius of turtle and the given clearance is added to the obstacle space to define the true traverable dimensions).

The paths below also have many nodes in common but the path was generated in accordance to eliminate collisions, so the paths generated by traced by R_3 and R_4 are considered to be optimal most probably but will be different if the sharing architecture comes into picture.

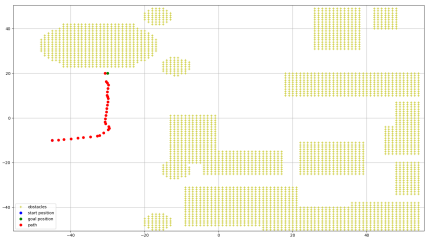

Fig. 8. The path generated by Robot-5 from initial_node(-45,-10) to goal_node(-30,20)

The Robot paths are by taking into account of the differential drive constraints of the robot so as to generate the left and right wheel velocities and angle of orientation of the robot in accordance towards the direction of the destination node. The code was programmed by using classes and lists to store the data for the individual robots. One problem we faced while simulating the given python script in ROS Gazebo was, even though the output speed nodes of left and right wheels were well determined for multi-robot system in Python, the ROS connection could not be well established for multiple robots at the same instinct.

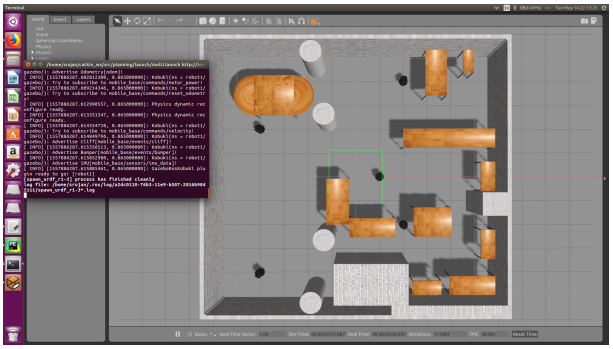

Fig. 9. The Multi-Robot spawning simulation in ROS Gazebo

Another set of readings and paths were generated as follows for the same set of speeds as mentioned above.

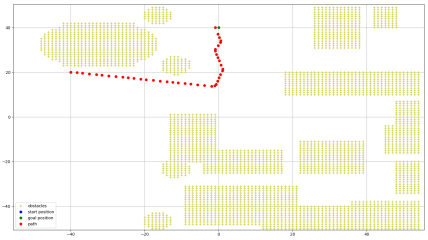

Fig. 10. Second path generated by Robot-1 from initial_node(-40,20) to goal_node(0,40)

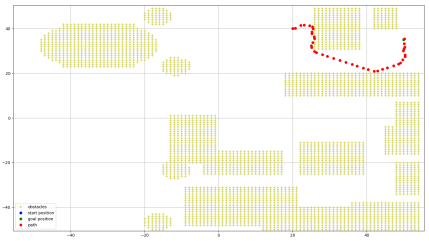

Fig. 11. Second path generated by Robot-2 from initial_node(20,40) to goal_node(50,35)

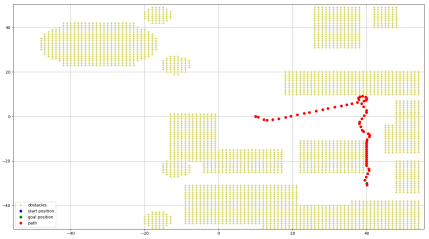

Fig. 12. Second path generated by Robot-3 from initial_node(10,0) to goal_node(40,-30)

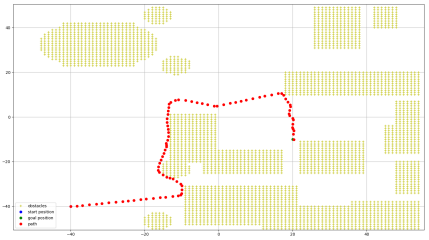

Fig. 13. Second path generated by Robot-4 from initial_node(-40,-40) to goal_node(20,-10)

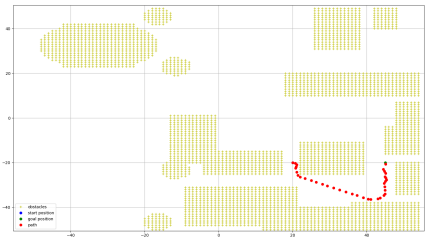

Fig. 14. Second path generated by Robot-5 from initial_node(20,-20) to goal_node(45,-20)

V. Conclusion

The above structure of data sharing is robust and can be applied to any system of multi-robots. In our project, we implemented this multi-robot system on five robots of Turtlebots. The feasibility of implementing any path planning algorithm for a multi-robot system makes it very adaptable to any programmed system and also easy to execute the operations that are needed for a certain environment. This project uses A* search algorithm in order to take advantage of the heuristic cost function which comes into play making it traverse through the path by exploring only the nodes that are closer to the destination node. We have come to a conclusion that even though it was easier to spawn a set of five Turtlebots at the same time, the concept of establishing the connection for the message readings from the sensor for localization and mapping in the dynamic map could not be well estimated and simulated for carrying out the sensor readings. So we tried to implement the same programmed script for the five robots at the same time from different locations to different locations in the map by eliminating collision between the robots. This collision elimination was done by generating different optimal paths when there is an exploration of mutually similar nodes. Moreover, since the multi-robot system is still an active field, there is work that needs to be resolved in our project regarding path generation using the mentioned sensor readings in determining the path planning in the dynamic map. It also saves time, distance travelled which in turn saves the memory allocation of nodes and battery power.

VI. Future Work

This project mainly eliminated the idea of limiting the robot perception to an individual robot, and gathers information collectively for faster decisions for the multi-robot system. However, there are certain obstacles in the proposed project that need to be overcome. The most important job is to implement this on a real physical environment and test its efficiency based on it. Although simulation does estimate the working and efficiency, it is still not accurate unless tested in real life scenarios. The path planning for the multi-robot system by estimating the path from the sensor readings and extended Kalman filter readings for pose estimation should be simulated so as to obtain the path generation for the robots.

Another limitation is to remove the dynamic obstacles tentatively in the dynamic map. While path planning, most robots avoid a path that is blocked, but as the dynamic obstacle vanishes overtime a robot may not efficiently judge the elimination of the obstacle from the dynamic map. Hence, the transience of dynamic obstacles needs to be well modeled. We look forward to overcome this situation overtime.

There is a possibility of minute differences that may arise while considering different types of robots in the multi-robot system. Not all robots have the same scaling factor and orientation when they picture the maps from two different locations. This situation needs to be addressed in the future as well. This data sharing architecture can be well implemented on multi-robot systems for warehouse transportation, restaurant and hospital transportation for goods or maintenance purposes.

VII. Acknowledgment

We express our gratitude to Professor Reza Monfaredi for the well designed and enlightening coursework on path planning algorithms for autonomous robots. These aided us in building up our technical knowledge and made us though provocative towards one of the main aspect in programming robots. We are also grateful to the teaching assistants, Ashwin Goyal and Utsav Patel for their help whenever needed.

References

[1] Iram Noreen, Amna Khan, Zulfiqar Habib, "Optimal Path Planning using RRT* based Approaches: A Survey and Future Directions", *International Journal of Advanced Computer Science and Applications*, Vol. 7, No. 11, 2016.

[2] P. Svestka, M. H. Overmars, "Coordinated path planning for multiple robots", *Department of Information and Computing Sciences, Utrecht University*, Tech. Rep. UU-CS-1996-43, 1996.

[3] N. Pinkam, F. Bonnet, N. Y. Chong, "Robot collaboration in warehouse", *Sixteenth International Conference on Control, Automation and Systems (ICCAS)*, Oct, 2019 pp. 269 - 272.

[4] Abu Bakar Sayuti H M Saman, Ahmed Hesham Lotfy, "An implementation of SLAM with extended Kalman filter", *2016 6^{th} International Conference on Intelligent and Advanced Systems (ICIAS)*, Aug. 2016, pp. 15 - 17

[5] S. M. LaValle, "Planning algorithms", *Cambridge University Press*, 2006, http://planning.cs.uiuc.edu/

[6] I. Chabini, Shan Lan, "Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks", *IEEE Transactions on Intelligent Transportation Systems*, March 2002.

[7] Abhijeet Ravankar, Ankit A. Ravankar, Yukinori Kobayashi, Takanori Emaru, "Can robots help each other to plan optimal paths in dynamic maps?", *56^{th} Annual Conference of the Society of Instrument and Control Engineers of Japan*, November, 2017.

[8] Jason M. O'Kane, "A Gentle Introduction to ROS", version 2.1.6, 2014.

[9] Morgan Quigley, Brian Gerkey, William D. Smart, "Programming Robots with ROS: A Practical Introduction to the Robot Operating System"

[10] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", *Robotics and Automation, IEEE Transactions*, vol. 12, no. 4, pp. 566 - 580, Aug 1996.

[11] A. Stentz, The focussed d* algorithm for real-time replanning, *In Proceedings of the International Joint Conference on Artificial Intelligence*, 1995, pp. 1652 - 1659.

[12] Wanli Tian, "The research into methods of map building and path planning on mobile robots", *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 08 February 2018.

[13] Mustafa Alberri, Sherif Hegazy, Mohamed Badra, Mohamed Nasr, Omar M. Shehata, Elsayed I. Morgan, "Generic ROS-based Architecture for Heterogeneous Multi-Autonomous Systems Development", *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 05 November 2018.