Intelligent Mobile Robotics Mapping Technical Report

Author. Ibrahim Ahmethan, Supervisor Dr. Aboozar Taherkhani. De Montfort University.

I. INTRODUCTION

One of the fundamental core concepts in mobile robotics is mapping, mapping allows the robot to discover and translate the dimnesions, shapes and environment of the robot when exposed to a new environment for the first time. Map construction or simply mapping goes throgh many stages; starting from the data obtained from sensors, software development, visualizing and simulating are the main concepts to bulid a 2-D map for new enviromnet. In order to achieve this assignment tools such as softwares, similators and visuliazitions used to implement the task of robot mapping, mainly ROS (linux bsed robotic operating system), Python (Programing software), (RVIZ) for Visulization and Gazebo simlation platform are used to build, test, display and improve the performance of the system in order to construct a reliable map. Turtle bot burger is the type robot used in this assigment for mapping [2].

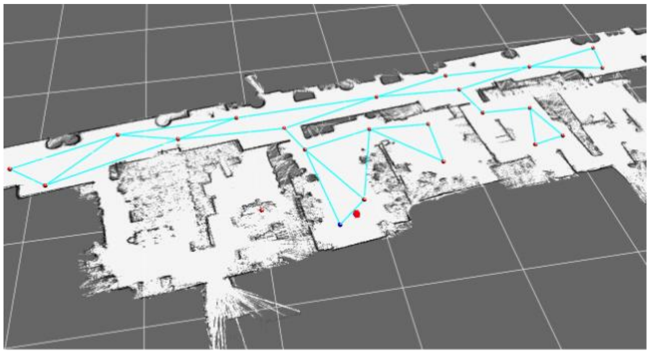

Figure 1. 2-D Mapping of a floor plan

II. MAP CONSTRUCTION THECNIQUES

The occupancy grid mapping algorithm technique is utilized in this assignment to perform map construction of unknown environment. Occupancy grid is a probabilistic approach which generates a map from uncertain sensor measurements and noisy data by taking into consideration that the position and heading of the robot is known, the grid is a sinlge point in a 2-D co-ordinate system that can be represented in the form of (x, y) values, each grid in the co-ordinated system initially is set to default value of 0.5 "50%" this value corresponds to the probability of the occupancy of each single grid, at the begining of map consturction since we do not have an idea about the occupnacy of each grid in the 2-D co-ordinate system; that is why we assgin the probability of unkown gird to 0.5, as the robot starts moving in the environment the probability of each single grid then will be updated according to the sensor readings of a robot[3], higher probability > 0.5 corresponds to a higher chance of a grid is being occupied, respectively lower probability < 0.5 corresponds to a lower chance of a grid being occupied. In summary the purpose of occupancy grid algorithm is to assign a probability between 0 – 0.9 "0% - 99%" for each single grid in the 2-D co-oredinate system. In the following figure 2 shown simple representations of grids in 2-dimensional,
As the color of each grid gets darker the probability of occupancy will be higer and vice versa.[4]

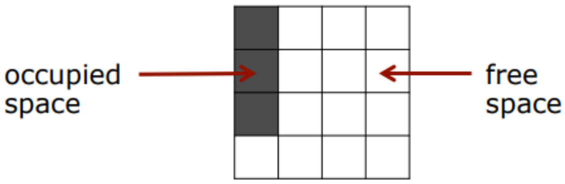

Figure 2. Representation of 2-D grids

A. *Bayes' Theorem*

In order to achieve occupancy grid algorithm to calculate the probability of 2-D co-oredinate system we need to have some calculation method. As an electrical engineer I can say that Bayes' theorem is one of the most widely used and greatest theorem in scince, moreover it has countless contributions in science. [5]

Bayes' theorem basically helps us to calculate the probability of an event under different circumstances, and the formula of the Bayes' as the following:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

A is the hypothesis that we need to test how much is true event and B is the available evidence or (prior probability). Thus in this assignment Recursive Bayes' Theorem is applied to update the occupancy grid for each single sensor reading with respect to the prior probability, the implementation of Bayes' theorm "conditional probability" will be shown in section III Software implementation section.[6]

B. *Odometry and Robot Position*

To construct a map first of all we have to know the position of the robot, in the turtle bot robot we have 360 Laser sensor (360° field view) each sensor covering degree 1° of angle with min and max range finder, thus the range finder gives us the angle and the distance to the closest obstacle, all of these sensors are publishing their readings continuously and simultaneously to the /odom topic and this odom topic has a massage type of nav_msgs/Odometry which provide us with all the necessary information about the location or "position" of our robot in the 2-D co-ordinate system as the following

- x – position data type float
- y – position data type float
- Ɵ – heading of the robot data type float

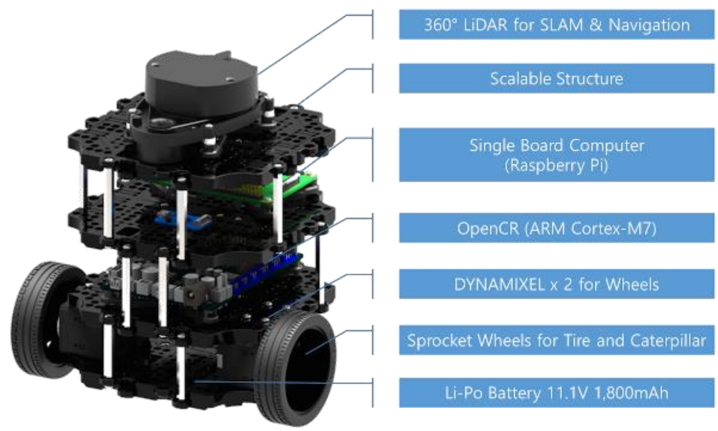

Figure 3. Turtle bot burger robot

C. *Global and Local Maps*

Each sensor finds the nearest obstacle to itself, thus moving robot will have a local co-ordinates that changes with respect to the robot movment (local maps), and also there is global co-ordinate system(global map) fixed, so translation and rotation operations performed to see the obstacles location in global positions to create a map of the environmnet. Thus we will have 3 steps to do assuming we know the location of the robot:

1- Finding the x, y coordinates of the obstacle

$$x = \cos(\theta_s) \text{ X } (r + \text{robot radius})$$
$$y = \sin(\theta_s) \text{ X } (r + \text{robot radius})$$

2- Rotate x, y to global co-ordinate system

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} \cos(\theta_r) & -\sin(\theta_r) \\ \sin(\theta_r) & \cos(\theta_r) \end{bmatrix}$$

3- Translate x' and y' to global co-ordinate

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x' + x_r \\ y' + y_r \end{bmatrix}$$

Further information about the implementation will be shown in section III Software impelemntation.

III. Sofware

Python programming language used to develop codes for Bays' theorem and converting the local map to global map in order to construct a map.

A. Recursive Bays' theroem

$$P(H|s_n) = \frac{P(s_n|H)P(H|s_{n-1})}{P(s_n|H)P(H|s_{n-1}) + P(s_n|\neg H)P(\neg H|s_{n-1})}$$

This theorem coded in python under def bays() function in order to perform mapping [7]

```
bayes(self, pos_x, pos_y, sensor_reading):
# R.Bayes theorem
if pos_y >= 0 and pos_y < self._map.height: # if the position inside the boundari
    if pos_x >= 0 and pos_x < self._map.width:
        previous_occupied_probability = self._map.grid[pos_x, pos_y] # to save th
        denominator_of_bayas = previous_occupied_probability * sensor_reading + .
        (1 - sensor_reading) * (previous_occupied_probability)
        numerator_of_bayas = previous_occupied_probability * sensor_reading
        bayes_theorem = numerator_of_bayas/denominator_of_bayas
```

Figure 4. Recursive Bays' theorem

B. translation and rotation to global map

```
radius_of_robot = 0.00001 #radius of turtlebot3
# building a scattergram map
for i in range(len(scan.ranges)):
    #angle_in_rad converting from degree to radians
    scan_range = scan.ranges[i]
    angle_in_rad = math.radians(i)

    if (scan.ranges[i]) != np.inf:
        if scan.ranges[i] > scan.range_min:
            if scan.ranges[i] < scan.range_max:
                # x and y obstacle positions
                x_pos = math.cos(angle_in_rad) * (radius_of_robot + scan_range)
                y_pos = math.sin(angle_in_rad) * (radius_of_robot + scan_range)
                # rotating x and y
                x_prime = (x_pos * math.cos(yaw)) + (y_pos* -math.sin(yaw))
                y_prime = (x_pos * math.sin(yaw)) + (y_pos* math.cos(yaw))
```

Figure 6. Finding the postion of single x and y readings and rotating

```
                x_double_prime = x_prime + pos.x
                y_double_prime = y_prime + pos.y
                (global_cor_x, global_cor_y ) = self.get_indix(x_double_prime,y_double_prime)
                calculated_probabilty = self.bayes(global_cor_x,global_cor_y, 1)
                self._map.grid[global_cor_x, global_cor_y] = calculated_probabilty
                print(calculated_probabilty)
else:
    (global_cor_x, global_cor_y ) = self.get_indix(x_double_prime,y_double_prime)
    self._map.grid[global_cor_x, global_cor_y] = 0.1
```

Figure 7. Translating to global map

[8]

IV. Testing and Results

Testing done with different resolutions and robot radius values to examine and record the changes in the mapping performance whether performance improving or not.

A. High raduis high resolution values

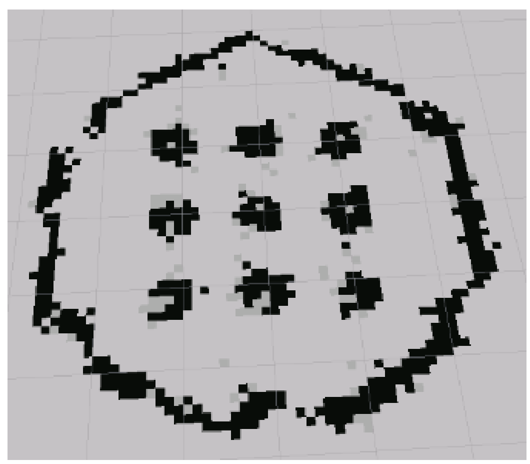

Figure 5. mapping with high radius high resolution

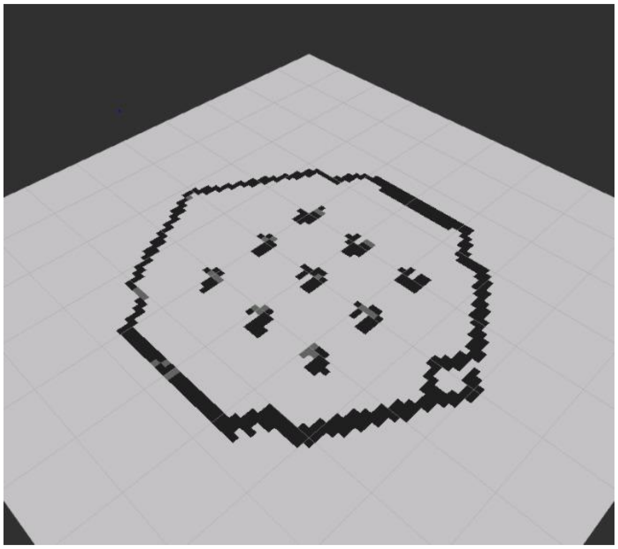

Figure 8. High resolution low radius

Balancing between resolution and radius of robot is important to obtain a map with better resolution and allow the robot to work in the specified boundaries in the map.

V. Conclusion

In conclusion, there are different mapping algorithms, vizulizations and simulations techniques, based on the given options in the assigment paper I prerfered to choose occupmany grid as technique, riviz as a visulization and gazebo as simulation method. During my studies for this assigment I aquired knowledge in many different fields such as operations in robotic systems, Python programming and develop a critical thinking skills to approach and tackle problems related to robotics.

the results of the mapping shown in the Testing and result section obtained after many attempts of trail and error to bulid the map, I have been through many challenges to bulit it, from my prespecitve even the mapping has not perfectlly achieve the grey scale (unknown area with probability 0.5 “50%”), however the obtained result is satisfactory.

Many thanks to the course teacher Dr. Aboozar tahirkani for his efforts in the course.

References

[2] https://en.wikipedia.org/wiki/Robot_Operating_System

[3] Week 7 – Occupancy grid slides – DMU black board

[4] https://en.wikipedia.org/wiki/Occupancy_grid_mapping

[5] https://betterexplained.com/articles/an-intuitive-and-short-explanation-of-bayes-theorem/

[6] http://ais.informatik.uni-freiburg.de/teaching/ws12/mapping/pdf/slam11-gridmaps-4.pdf

[7] Week 4 – Building map slides – DMU black board

- All the studying materials and lessons in BB – DMU has been used.