BILKENT UNIVERSITY

MECHANICAL ENGINEERING DEPARTMENT

ME 574: MOBILE ROBOTS

TERM PROJECT REPORT

Cengizhan Demirbaş

Gürkan Kılıçaslan

Adil Aytar

Course Instructor: Dr. Onur Özcan

Teaching Assistant: Amirali Abazari

Submission Date: 27.05.2022

INTRODUCTION

The goal of this project is to create a self-navigating transportation system that can go inside a house (or workplace) with rooms that are not designed for accessibility. The aim is to provide assistance to elderly people.

To achieve autonomous navigation, the project is divided into several steps. First, kinematics for two different designs are derived and analyzed. After the analysis, it is decided to continue with the differential drive robot because of its convenience and abundance of resources. To perform safe navigation around a place that is not designed for accessibility, the final step of the project is determined as path planning and obstacle avoidance. Required modules such as feedback control, mapping, and localization are developed to reach the final step of path planning and obstacle avoidance. Throughout the simulations, probabilistic errors are implemented to simulate real-world conditions.

Since the A* algorithm provides efficiency with completeness, it is preferred over different algorithms such as Djikstra's algorithm (1), D* algorithm (2), and RRT (3). Markov Localization is used instead of Kalman Localization because Markov addresses the kidnapped robot problem, which is highly possible to encounter with the wheelchair in case of shutdowns. Chosen algorithms are tested with various scenarios to ensure their reliability and robustness to different conditions.

In addition to the development of an autonomous robot, the safety and comfort of the users are essential. To achieve this goal, the autonomous driving system is expected to be similar to a regular, user-driven wheelchair. Onyango et. al. identifies human-like steering in [4] with the variables time, the curvature of the path, desired velocity, and risk factors. For the convenience of the user, these factors should also be considered in developing the control system.

Kinematics

-Differential Drive with Free Castor Wheel

The free castor wheel does not impose any constraint on the wheelchair's kinematics, therefore, the wheelchair kinematics are the same as the kinematics of the differential drive robot. Kinematics of the differential drive is a well-studied topic, hence, it is believed that calculations do not require further explanation [5].

$$\dot{x}_R = \frac{r_1\dot{\varphi}_1 + r_2 * \dot{\varphi}_2}{2} \tag{1}$$

$$\dot{y}_R = 0 \tag{2}$$

$$\dot{\theta}_R = \frac{r_1\dot{\varphi}_1 - r_2 * \dot{\varphi}_2}{2 * l} \tag{3}$$

$$\zeta_I = R(\theta)^{-1} \begin{bmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta}_R \end{bmatrix} \tag{4}$$

-Castor wheel steering and rear wheel drive(rear wheels are connected with axle)

In this design, the wheelchair has two standard wheels connected with an axle and a castor wheel that is responsible for the steering. Kinematics is derived from the position of the instantaneous center of rotation (ICR) and the expected movement of the wheelchair. This configuration and definition of the parameters can be seen in Figure 1. Kinematics of this configuration is expected to be the same as tricycle kinematics since tricycle has one steering wheel and two standard wheels similar to the wheelchair's configuration. [6]

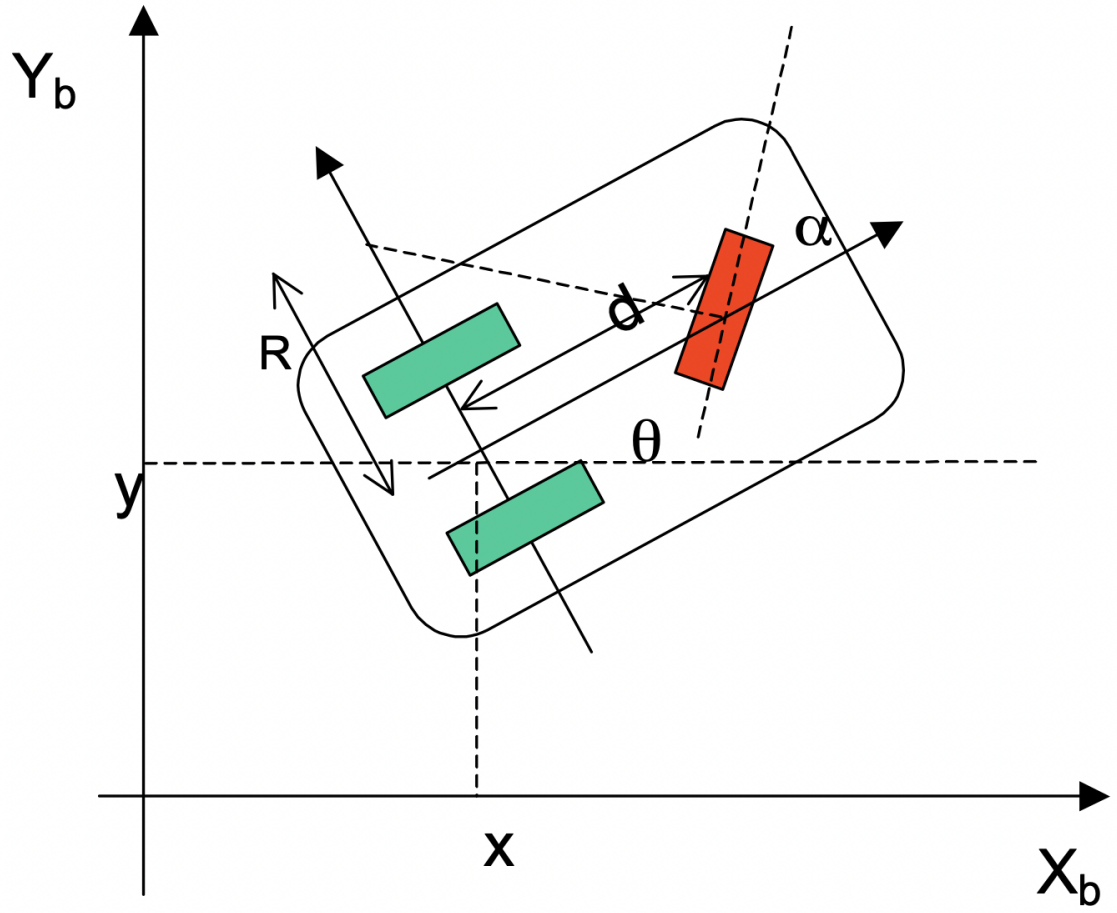

Figure 1: Scheme of the wheelchair with steerable castor wheel[6]

The linear velocity of the wheelchair is given by multiplying the rotational speed of the steering wheel with its radius.

$$v_s = \varphi_s * r_s \tag{5}$$

Distance from the center to the ICR (R(t)) is calculated from the geometry.

$$R(t) = d * tan(\frac{\pi}{2} - \alpha) \tag{6}$$

$$\dot{x}_R = v_s * cos(\alpha) \tag{7}$$

$$\dot{y}_R = 0 \tag{8}$$

The wheelchair rotates around the ICR with a radius calculated from the geometry.

$$\varphi_R = \frac{\varphi_s * r}{\sqrt{d^2 + R(t)^2}} \tag{9}$$

$$\varphi_R = \frac{v_s(t))}{d} * sin(\alpha(t)) \tag{10}$$

$$\zeta_I = R(\theta)^{-1} \begin{bmatrix} \dot{x}_R \\ \dot{y}_R \\ \dot{\theta}_R \end{bmatrix} \tag{11}$$

Since both configurations are capable of move the chair along any trajectory, decision on the wheelchair wheel configuration is made considering the abundance of the resources on differential drive robots.

FEEDBACK CONTROL

To perform feedback control, the path planning algorithm is split into the intermediate states. Polar coordinates are used to define the error between the robot position and the goal position. As explained in [5] finding control parameters $k_\alpha, k_\beta, k_\rho$ to control the linear and rotational velocity of the robot according to the equations below drives error to zero, which means reaching the goal position. The definition of the polar coordinates can be seen in the figure below.

$$\rho = \sqrt{(\Delta x)^2 + (\Delta y)^2} \tag{12}$$

$$\alpha = -\theta + atan2(\Delta y, \Delta x) \tag{13}$$

$$\beta = -\theta - \alpha \tag{14}$$

$$v = k_\rho * \rho \tag{15}$$

$$\varphi = k_\alpha * \alpha + k_\beta * \beta \tag{16}$$

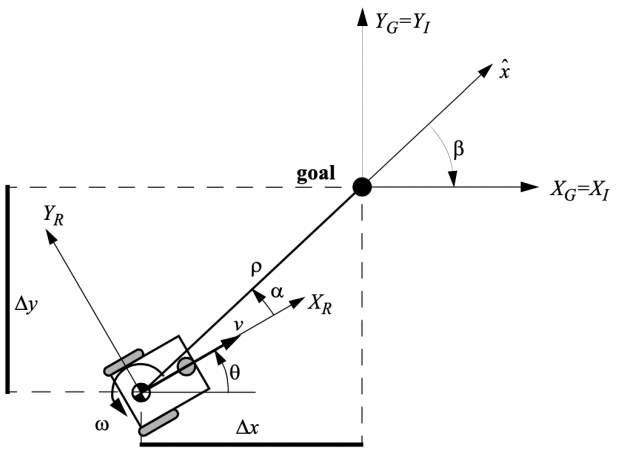

Figure 2: Scheme for the definition of the variables used in the feedback control [5]

MAP

According to the project definition, the map had to be a house or an office where the rooms are not designed for accessibility. Therefore, we researched for a house with these criteria. We found the following drawing, and used that as a reference with some modifications for our map, given below it:

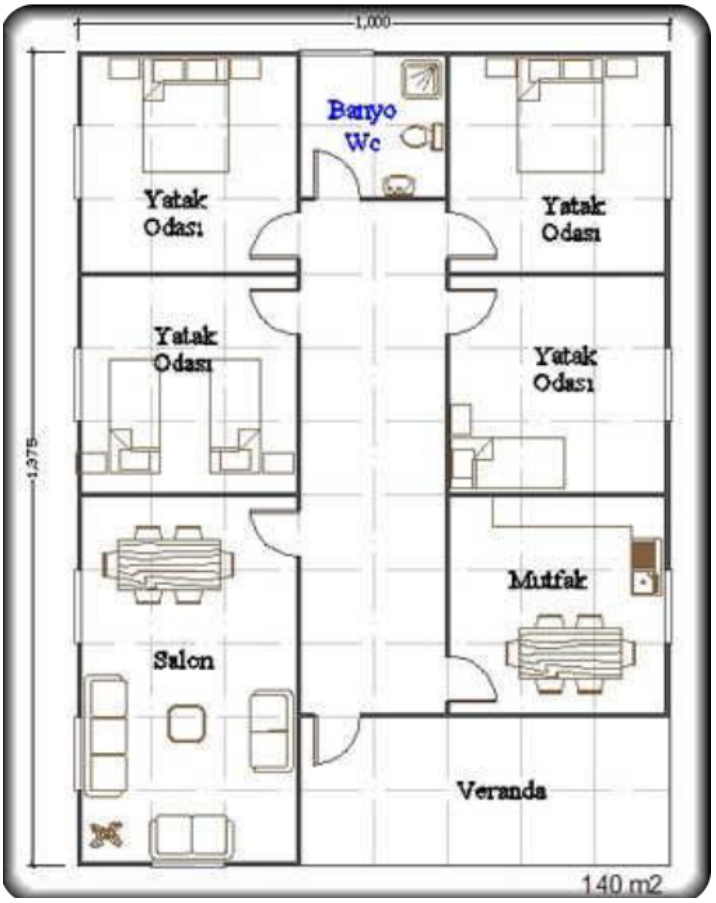

Fig 3: The map we imitated (above) and the binary map we created (7)

The dark parts in the map represent obstacles and the white parts represent empty spaces. Some of the obstacles are drawn similar to furniture. We did further modifications to the map as we progressed along with the project. We mainly tried to make the navigation harder by adding more obstacles. The figure below is the second iteration of the map:

Fig 4: Second iteration of the map

This version includes more furniture, along with pot plants on the balcony. The final version was made with the following scenario in mind: an elderly person in a wheelchair walks out of the bathroom and wants to go to the living room. However, his grandchildren made a mess of the hallway, and now the robot needs to navigate through all the toys and pillow castles on the following map:

Fig 5: Final iteration of the map

All these three maps are used during the project. The following are the binary images of these maps:

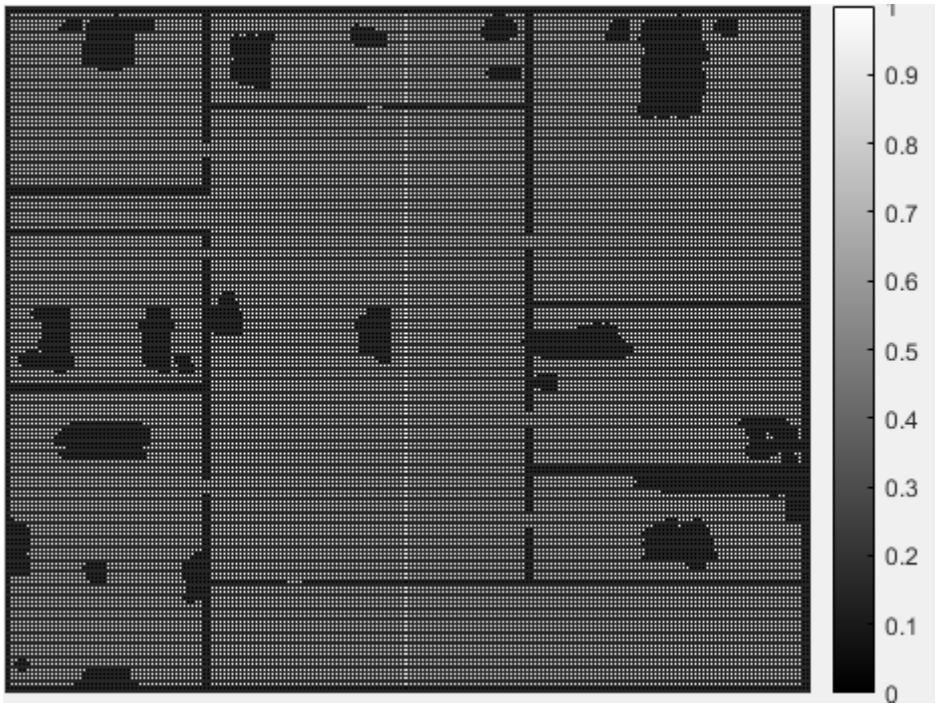

Fig 6: First iteration of the map in the heatmap

SENSOR

The environment includes obstacles over the maps used. The robot has to learn or at least estimate where they are. For this purpose, a lidar range sensor is needed for our project. A typical lidar sensor emits pulsed light waves into the surrounding environment. These pulses bounce off surrounding objects and return to the sensor. The sensor uses the time it took for each pulse to return to the sensor to calculate the distance it traveled [8]. Developed lidar for our project can see a maximum range of 10 cells. It is not able to see anything beyond obstacles. The lidar sees an obstacle at its proper location with a 40% probability whereas it sees an obstacle in an adjacent cell has a probability of 7.5%. Lidar perception affects the movement of the robot. Estimating the wall's position in the wrong cell makes it move unnecessarily, extending localization time. The probabilistic interpretation of the lidar reading is as follows:

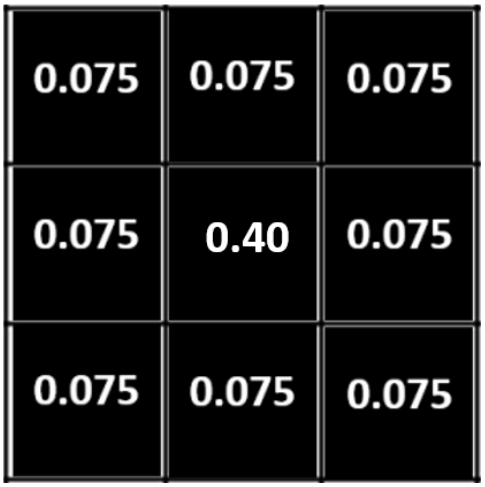

Fig 7: The probabilities that the robot may see where an obstacle cell is. The middle cell is the actual cell.

LOCALIZATION

To proceed with the path planning and moving on the path with the feedback controller output, the robot needs to be where it is. For this purpose, we used Markov Localization. Markov Localization addresses the problem of state estimation from sensor data which is lidar in our case. Markov localization is a probabilistic algorithm: Instead of maintaining a single hypothesis as to where in the world a robot might be, Markov localization maintains a probability distribution over the space of all such hypotheses. The probabilistic representation allows it to weigh these different hypotheses in a mathematically sound way [9].

We defined the robot movement function as follows:

- It can choose to go up, down, left, or right with the probability of 0.25 for each.
- It can move 1,2,3,4 or 5 cells at each cycle.
- The robot is able to move in any direction it wants, but when it moves in a direction, with a 5% chance it moves 1 or 5 cells, with a 15% chance it moves 2 or 4 cells, and with a 60% chance it moves 3 cells.

These conditions yield the following movement capability:

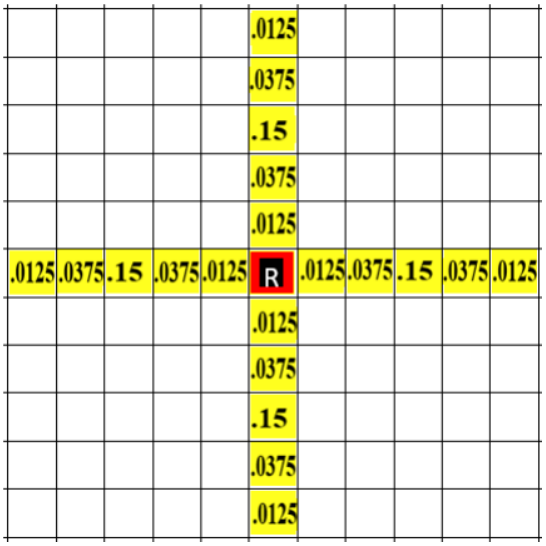

Fig 8: Probabilities of the cells that the robot may be inside in the next position

If the maps are relatively empty(less number of obstacles), the robot's belief is hardly changing because there is less opportunity to know where it is because the map is hardly changing from the robot’s perspective. Including more obstacles on the map makes robots learn where it is faster. In the case of helping elderly people, we expect the rooms are messy, meaning more obstacles.

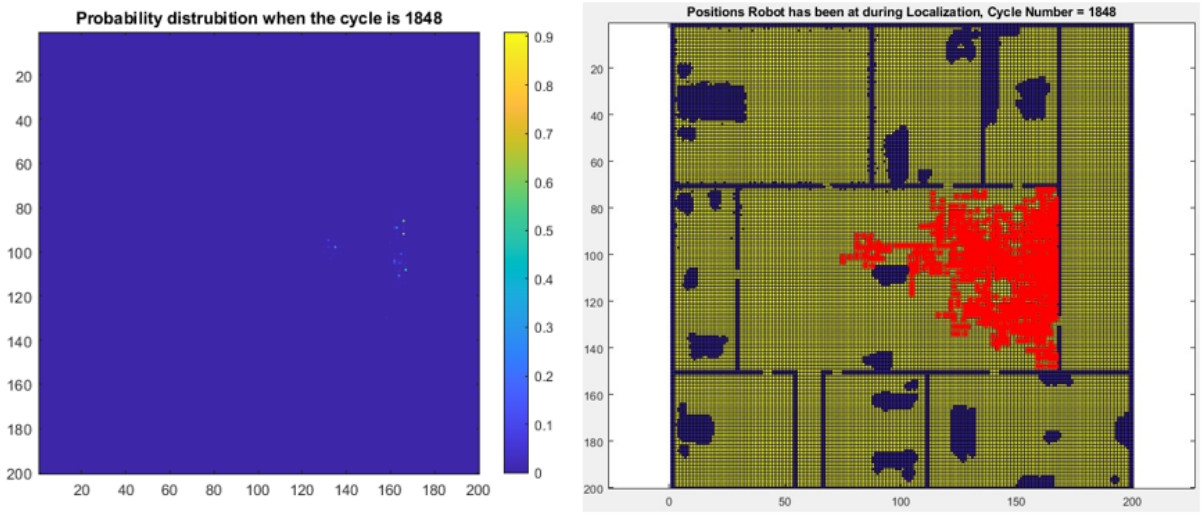

Fig 9: Robot’s belief and positions it has been. The map is empty, so robot makes a lot of move to learn.

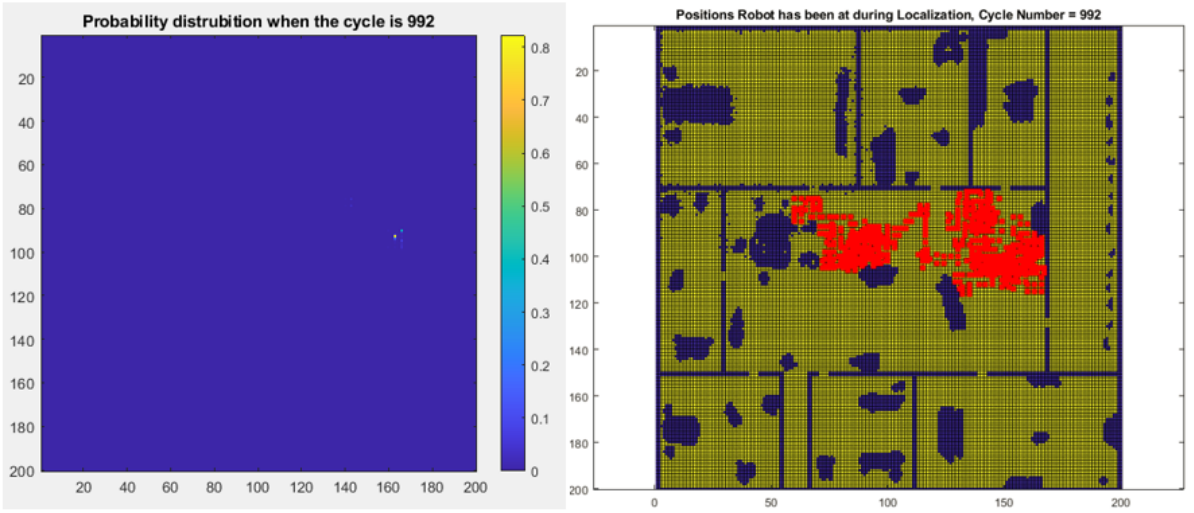

Fig 10: Robot’s beliefs and positions it has been on a new map. The map has more obstacles, so the robot makes fewer moves to learn.

PATH PLANNING AND OBSTACLE AVOIDANCE

There are many algorithms used in robotics applications, such as Djikstra's algorithm (1), D* algorithm (2), and RRT (3). During this project, we used the A* algorithm, which is a graph traversal and path search algorithm which is often used due to its completeness, optimality, and optimal efficiency (10).

The algorithm uses its current position, goal position, and its neighboring cells to determine a path. However, in our code, we did not include diagonal neighboring cells for simplicity. The code starts with adding heuristic cost to every cell depending on their distance to the goal position. The cells with obstacles are assigned a large value so that they are never chosen during path planning. After assigning a heuristic cost to every cell, we focus on the initial cell. We take a note of the g cost, which is the distance from the initial cell, and add a heuristic to find the f cost, which is the most important cost parameter we consider for cells. Then, we look at the neighboring cells that are within the limits of the map, are not the cells we visited before, and are not obstacles. We calculate the costs of these neighboring cells and note the direction the robot takes to reach them, which will be later used for finding the path. After every neighbor is inspected, the neighbor with the lowest f score is selected, and the same process is done for its neighbors. If two neighbors have the same f score, the one with the lowest g score is chosen. Note that unchosen neighbors are not eliminated, and they are chosen if the other neighbor led to neighbors with higher costs. This is done iteratively until the goal position is reached.
We performed the path planning task for three maps given before. This is the path planning for the first iteration:

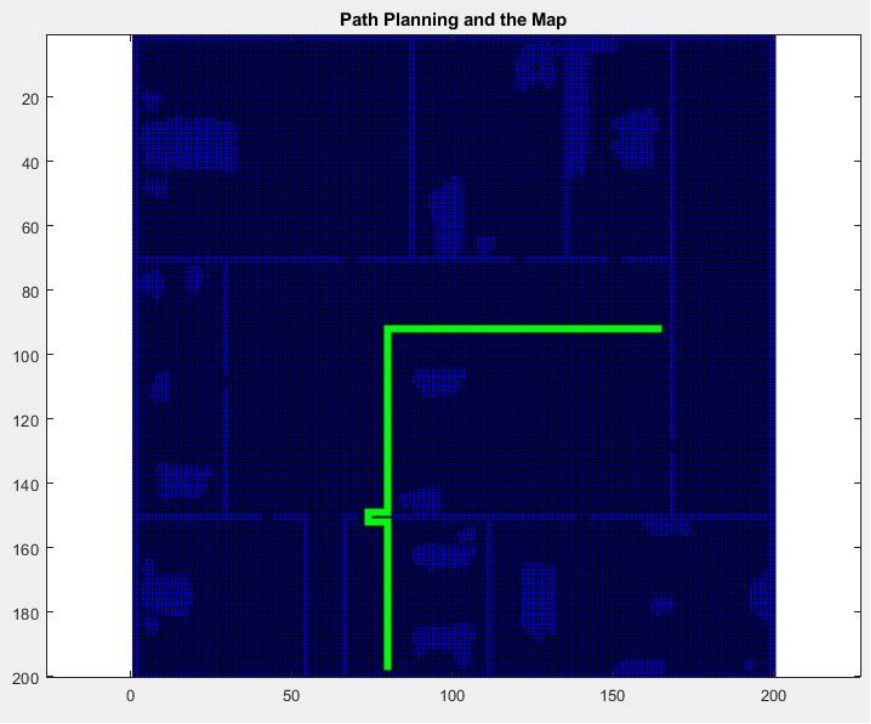

Fig 11: Path planning in the first iteration of the map

The path planning given for the second iteration of the map is as below:

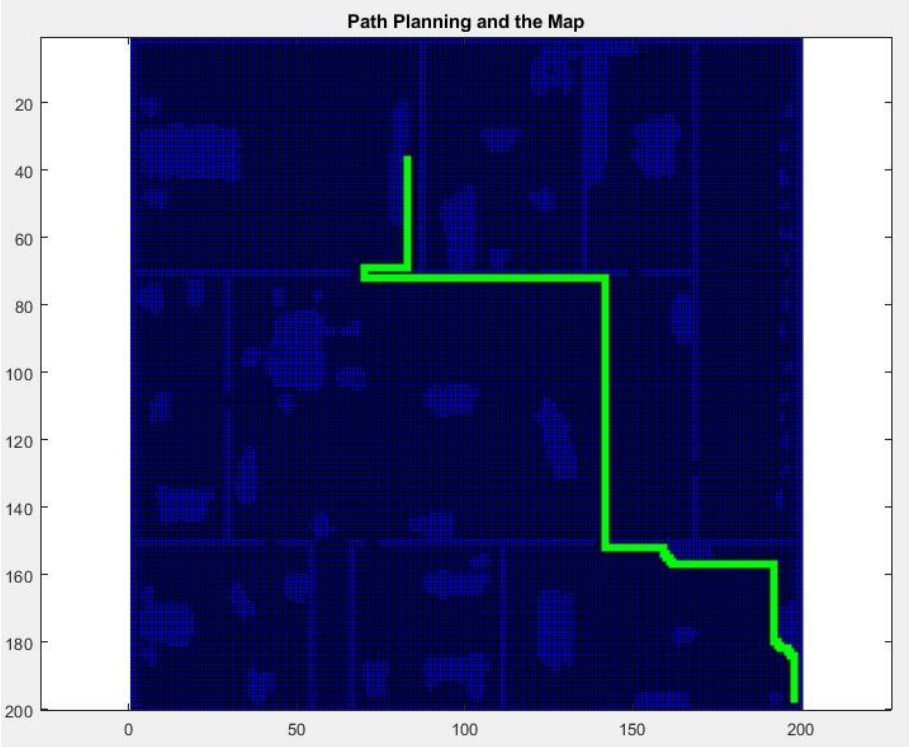

Fig 12: Path planning in the second iteration of the map

Finally, path planning and obstacle avoidance is done on the hardest map among the three:

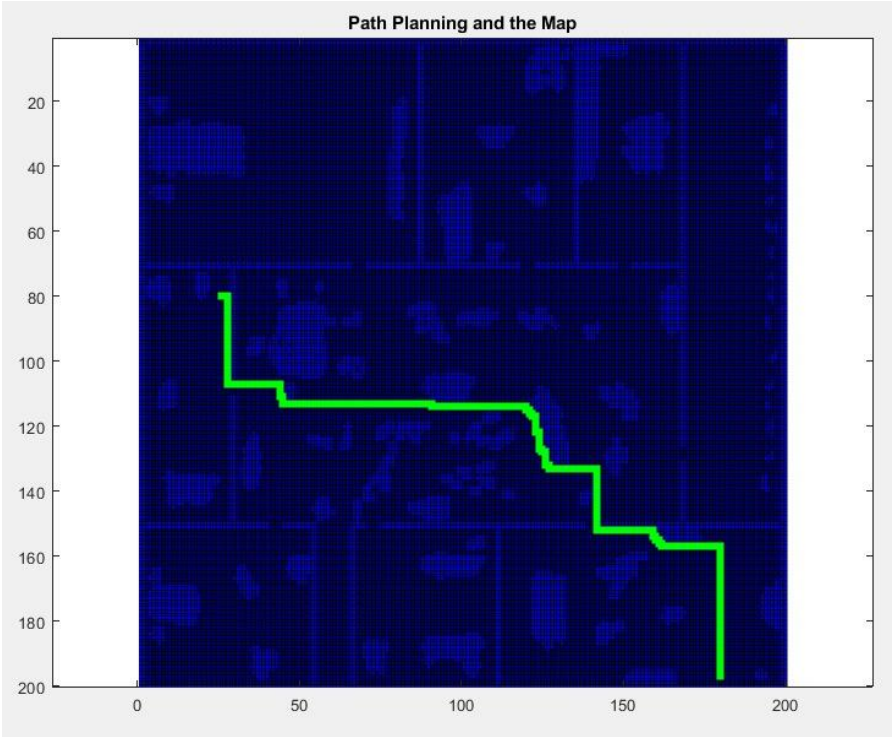

Fig 13: Path planning in the last iteration of the map

As one can see, the path generated in all cases avoids obstacles. However, it is not the shortest path algorithm, because the diagonal cells are discounted.

CONCLUSION

As aimed, the design and simulation of an autonomous wheelchair with obstacle avoidance and safe navigation around non-accessbility-friendly places are completed. The autonomous wheelchair uses the A* path planning algorithm alongside the Markov Localization algorithm. Simulations are run for several cases under different conditions. We successfully drive the wheelchair autonomously thanks to all algorithms used. We expect elderly people to be slow moving and therefore they find it difficult to organize the space in which they live. The robot handles this problem as well. It is not important how messy the rooms are, the robot doesn't get affected by mess and get confused. In fact, obstacles make localization easier. It is a fact that a huge part of elderly people is not well capable of self-moving. We believe our design of localization, path planning, and controlling the robot will help those who lack movement capability.

REFERENCES

1: Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271.

2: A. Stentz, "Optimal and efficient path planning for partially known environments," *Intelligent Unmanned Ground Vehicles*, pp. 203–220, 1997.

3: "Rapidly-exploring random trees: Progress and prospects," *Algorithmic and Computational Robotics*, pp. 303–307, 2001.

4: S. O. Onyango, Y. Hamam, K. Djouani, and B. Daachi, "Identification of wheelchair user steering behaviour within indoor environments," *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2015.

5: R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. Cambridge, MA: MIT, 2011.

6: M.I.Riberio, P. Lima, "*Kinematics models of mobile robots*," Lisboa, PT: Instituto Superior Técnico,2002.

7: "Prefabrik EV planı 2: EV planı, EV Planları, evler," *Pinterest*, 06-Oct-2015. [Online]. Available: https://tr.pinterest.com/pin/298152437809694426. [Accessed: 27-May-2022].

8: *What is LIDAR? Learn How Lidar Works*. Velodyne Lidar. (2022, May 20). Retrieved May 27, 2022, from https://velodynelidar.com/what-is-lidar/

9: Markov localization. (n.d.). Retrieved May 27, 2022, from https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume11/fox99a-html/node2.html

10: N. J. Nilsson, “Artificial Intelligence: A modern approach,” *Artificial Intelligence*, vol. 82, no. 1-2, pp. 369–380, 1996.