Rose-Hulman Institute of Technology

ECE 425 - Mobile Robotics

Devon Adair & Hunter LaMantia

Robot: Moravec

Navigation Competencies & Wireless Communication

2/17/2019

**Abstract**

**Table of Contents**

Abstract……………………………………………………………………………………………..

I. Objective………………………………………………………………………………………….

II. Theory…………………………………………………………………………………………...

III. Methods…………………………………………………………………………………………

IV. Results…………………………………………………………………………………………..

V. Conclusion………………………………………………………………………………………

VI. Appendix………………………………………………………………………………………..

**I. Objective**

The purpose of this project is to develop programs that allow Moravec to travel a four square by 4 square environment using various methods. These methods include path planning, localization, and mapping. Moravec can be driven automatically or with manual wireless inputs. In both cases, Moravec wirelessly sends map data back to a transceiver connected to a laptop, where the map data is displayed. The main movement algorithm utilized by the robot functions by using wall following code from a previous lab. The robot is also given a series of directions to go in at T junctions, allowing it to make turns.

**II. Theory**

<https://www.sciencedirect.com/science/article/pii/S092188901300167X>

<https://books.google.com/books?hl=en&lr=&id=3c9w6XEUxIMC&oi=fnd&pg=PA1&dq=robotic+mapping&ots=ZDXduCj_DK&sig=wTDgoL34OlkkdDIaSWvIEZsTiyE#v=onepage&q=robotic%20mapping&f=false>

<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4767907>

**III. Methods**

The first algorithm made was a simple movement algorithm meant to supplement the eventual path planning algorithm. The movement algorithm is passed a string containing the letters S, T, L, R, and F. The robot increments through the string and its behavior changes based on what letter it is currently on. S tells the robot to start its hallway following, and always appears at the beginning of the passed string. T tells the robot to stop moving once it reaches its goal and is always placed at the end of the string. L, R, and F are all behaviors for when the robot senses a place where it could potentially turn. L tells the robot to turn left, R tells it to turn right, and F tells it to keep moving forward.

We opted to give the path planning algorithm a second helper function to make maps easier to parse. Instead of writing path planning functions for the robot to navigate 4 by 4 occupancy grids and four by four topological maps, we converted both into nine by nine occupancy grids, where the 5 extra rows and columns are represent the spaces between cells of the real grid, and are counted as full if walls exist there. This allowed us to make only one path planning function for both types of input.

The path planning algorithm itself takes the starting coordinates and target coordinates of the robot as inputs, and also accesses the nine by nine occupancy grid created by the aforementioned helper function. The algorithm assigns a value of zero to the location of the target coordinates, and increments the value up for the surrounding map squares. This results in each square having a number equal to its distance from the target square. The algorithm also takes walls into account, making the numbers snake around them. The algorithm then creates a list of instructions in the form of a string that the movement algorithm and read and passes it.

The localization algorithm is similar to the path planning algorithm in that it uses the nine by nine occupancy grid to navigate, except it does not know where on the map it begins. We set up a basic movement algorithm for localization that uses the sensors to determine which adjacent squares are empty, and then it chooses one to travel to. Every time the robot travels to a new square, it updates how far from its starting position it has gone and uses its sensors to scan the area around it. After scanning the area, it interprets its current location as a topological map square and adds it to an array. It then checks the map for possible combinations of squares it could have traveled based on the topological values. Once it has been narrowed down to one path, the robot knows where it is and can back calculate where it began based on its tracked displacement. The robot then sends its current coordinates to the path planning algorithm and it travels to its target. This process can also be done by driving the robot manual with commands from the laptop and transceiver. In either case, the robot also sends its current, beginning, and ending coordinates to the transceiver using bi-directional communication.

The mapping algorithm, much like the localization algorithm, works with both manual driving and automatic driving. It is essentially the opposite of the localization algorithm, in that the robot knows its starting position, but does not know anything about the world around it. We set up another movement algorithm similar to the one made for localization, except we make sure that the robot eventually enters and scans every map square that it detects as open. It starts out with an occupancy grid marked as completely full, and whenever it enters a new map square and scans it, it changes the relevant squares to be empty. Once the map is completely explored, all empty map squares have been marked as empty. Once again, the completed data is then used by the path planning algorithm to make the robot go to its target location.

*1. Were there any issues with the wireless communication? How could you resolve them? If at all.*

We had a lot of problems where the robot would fail to send maps to the transceiver, or it would only send brief parts of maps. We eventually found that us sending commands to the robot was interrupting the robot sending us maps, so we resolved the problem by only having the robot send one row of the map at a time and adding delays to the sending/receiving code.

*2. What does the state machine, subsumption architecture, flowchart, or pseudocode look like for the path planning, localization, and mapping? (It should be in the appendix of the report).*

*3. How would you implement SLAM on the CEENBot given what you have learned about navigation competencies after completing the final project? If you research solutions, make sure you cite and list references in APA or MLA format.*

SLAM would be simple to implement using our current code. The robot

*4. What was the strategy for implementing the wavefront algorithm?*

*5. Were there any points during the navigation when the robot got stuck? If so, how did you extract the robot from that situation?*

*6. How long did it take for the robot to move from the start position to the goal?*

*7. What type of algorithm did you use to selection the most optimal or efficient path?*

*8. How did you represent the robot’s start and goal position at run time?*

*9. Do you have any recommendations for improving that robot’s navigation or wavefront algorithm?*

*10. How did you use the serial monitor and bi-directional wireless communication to represent the map?*

*11. What type of map did you create and why?*

*12. What was key in the integration of the localization, mapping, and path planning?*