Lab 6

Path Planning (Occupancy Grid and Topological Map)

Reading: *Ch. 4 of the text*

(Demonstration due in class on **Thursday**)

(Code and Memo due in Moodle drop box by midnight on **Sunday at midnight**)

Read this entire lab procedure before coming to lab.

****

Purpose: The purpose of this lab is to use topological and metric navigation to move the mobile robot from a start point to a goal point in the world.

Objectives: At the conclusion of this lab, the student should be able to:

- Implement path planning on a mobile robot to move the robot from a start point to a goal point given an a priori map of the world
- program wavefront or grassfire expansion to create a path planning algorithm

Equipment: base Robot

range and contact sensors

Theory:

Topological Path Planning and Execution

Topological path planning is based upon landmarks in the world. A distinctive place is a landmark where the robot can make a navigation decision. Typically the robot will use one behavior to move in the world and then change when it gets close or in the neighborhood of the distinctive place. For the purpose of this lab the distinctive places in the world will be corners and t-junctions. The student team will design behaviors and perceptual schema for identifying gateways in an artificial environment. The robot will be given a list of navigation commands based upon the world's topology and use a parsing routine and a sequencer to move the robot from a start point to a goal point. For example, if the robot is given "SLRT" (S = **S**tart, L = go **L**eft, R = go **R**ight, T = **T**erminate) then it would start, follow hallway, turn left at the next gateway, turn right at the next gateway, then stop at the last gateway. The robot should continue to move forward until a gateway is encountered or until the stop command is encountered. In order to extend the robot's topological path planning capabilities, the navigation commands could be modified to include numerical information such as "3L" (make the third left") or "1L" (make the first available left). In other words, if the robot has a left command next but the first available gateway is on the right then it should continue following the hallway. This would mean that the robot must keep track of the type of and number of gateways encountered. Figure 1 is an example of topological navigation using the command "SRLT".

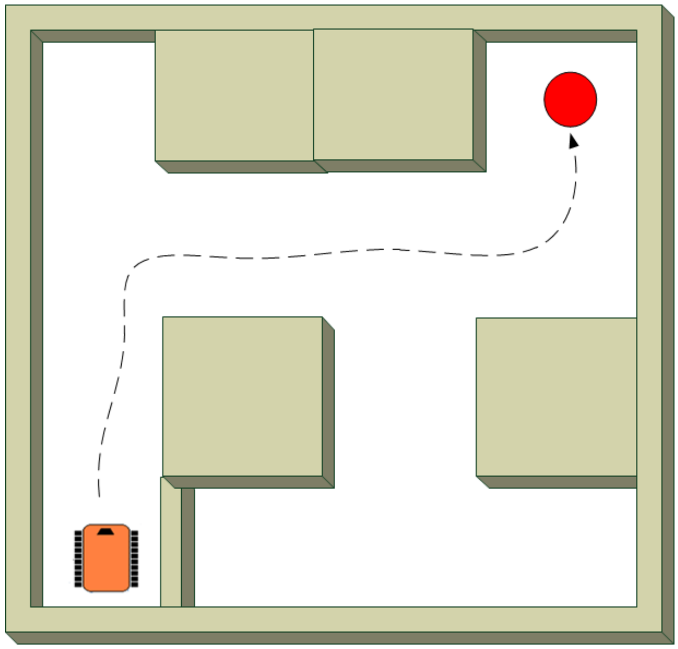
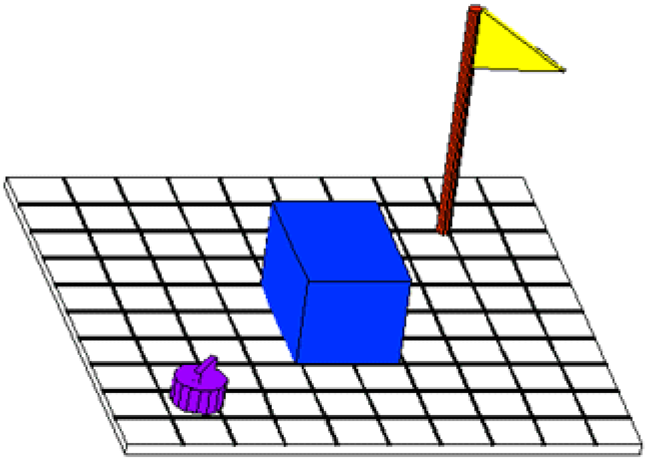


Figure 1: Topological Navigation ("SRLT")

Metric Map Path Planning and Execution

In this lab, you will use a wavefront algorithm on an a priori map to create a path from the robot's start position to goal location. Use the obstacle avoidance and move to goal behaviors to move through the list of goal points until the robot arrives at the final destination. Assume that the algorithm uses an eight-neighborhood so that the robot can move diagonally. The configuration space will be an occupancy grid divided into 18" x 18" squares, where free space is represented by 0's and occupied space by 99's. You should devise a scheme to represent the robot's start position and goal position. Your code should be flexible such that these values can be specified at run time. Figure 2 is an example of the world representation.

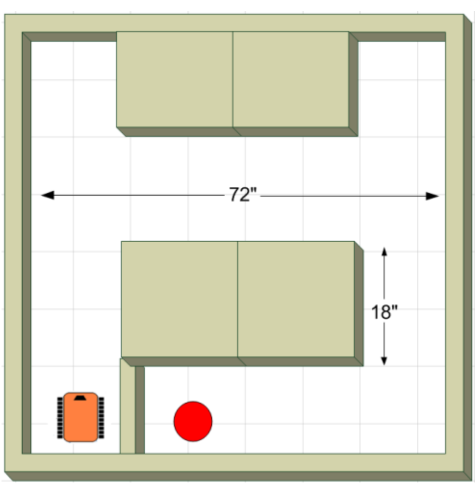


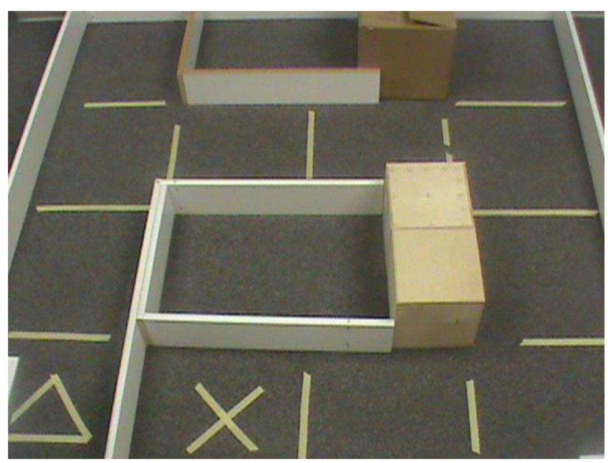
0	0	0	0	0	0	0	0
0	0	0	0	0	0	G	0
0	0	0	0	0	0	0	0
0	0	0	99	99	99	0	0
0	0	0	99	99	99	0	0
0	0	0	99	99	99	0	0
0	0	0	0	0	0	0	0
0	S	0	0	0	0	0	0

a. Real world b. Configuration Space (8 x 8 matrix)

Figure 2: World Representation

The test arena for the lab demonstration will be 6 ft x 6 ft with 18” x 18” obstacles. This artificial world will be a 4 x 4 grid where the robot’s start point is denoted by a ‘Δ’ and the goal point is marked by an “X”. Figure 3 shows a sample test arena.





a. Artificial world b. Real world

Figure 3: Test Arena

The wavefront is created by starting at the destination and creating eight connected neighbors back to the start point. The numbers represent the number of steps to the goal point. The robot would then follow the numbers in the reverse order to arrive at the goal point (see Figure 4). The goal is for the robot to always move such that the steps to the goal position are reduced. This path plan has 9 steps from start to finish. Note that you may need to grow the obstacles by the robot’s width or radius to avoid clipping them.

7	6	5	4	3	2	1	1	1
7	6	5	4	3	2	1	**0**	1
7	6	5	4	3	2	1	1	1
7	6	5	**99**	**99**	**99**	2	2	2
7	6	6	**99**	**99**	**99**	3	3	3
7	7	7	**99**	**99**	**99**	4	4	4
8	8	8	7	6	5	5	5	5
9	**9**	8	7	6	6	6	6	6
10	9	8	7	7	7	7	7	7

Figure 4: Test Arena Wavefront

****
LAB PROCEDURE

****

Part I - Topological Navigation

1. The student team should place the robot in several corners and intersections of hallways in the artificial environment and determine the perceptual schema to identify these gateways and distinctive places.
2. It would be advisable to use a combination of the IR and contact for sensor redundancy to identify these locations in the world. There will be some sensor error, this is a standard problem in mobile robot navigation and the program should be designed in such a way to minimize the effect of the error. Figure 5 provides descriptions of the possible world landmarks.

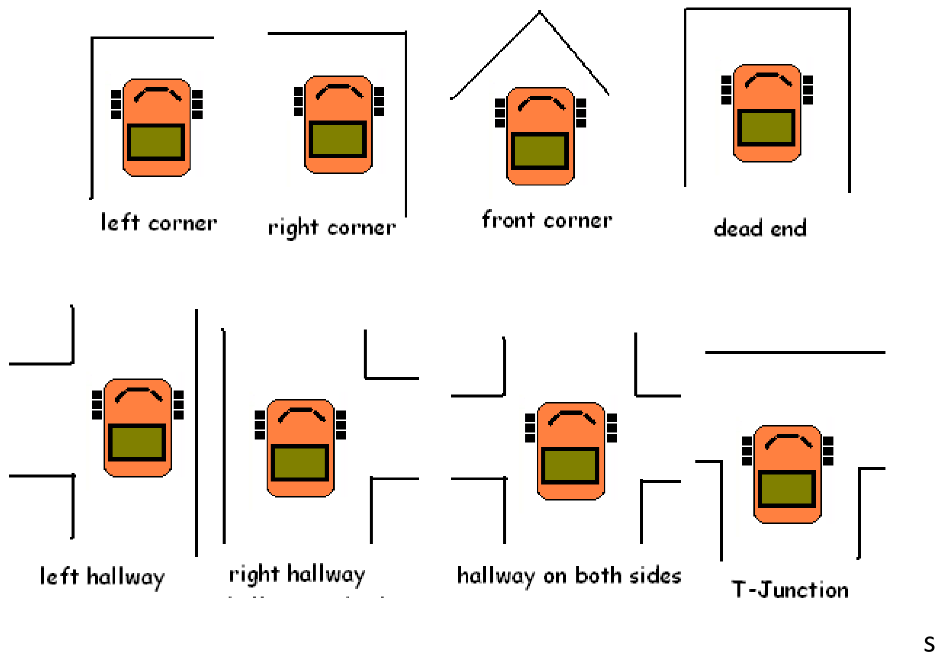


Figure 5: Distinctive Places

3. Include a table similar to Table 1 that includes the test data and perceptual schema for each of the landmarks.
4. Program the robot to identify the gateways and make navigation decisions such as follow hallway, turn left or turn right based upon a topological path plan.
5. Finally, use the robot push buttons to give the robot a command such as "SRLT" and place it in the world so that it can execute the path.

Table 1: Perceptual Schema Data Table

Landmark	Front IR	Left IR	Right IR	Right Contact	Left Contact
Corner in front					
Corner on left					
Corner on right					
Hallway on both sides					
Hallway on left					
Hallway on					

right					
T-junction					
Dead End					

Part II - Metric Path Planning and Execution

1. Download the world map from the Lab 6 folder in Moodle. The world map is a text file that includes an 4 x 4 array of 0's and 99's that represents free space and obstacles. You must devise a method to code the a priori map into your program. You can also display the map on the LCD to confirm that it matches the real world. Download the "PixelPrint.c" program from the Moodle folder to see examples of how to print pixels on the LCDE screen.

2. At the beginning of the demonstration, you will be given the robot's start position and goal point. Create a technique to use a pushbutton sequence to input the start position and goal position to the robot to the robot.

3. The robot should plan a path from the start point to goal position by using the wavefront algorithm. The LCD should display the path planned by the robot. One suggestion for doing this is to show a list of cells that the robot will traverse from the start point to goal point.

4. You should then place your robot at the start position and press start and it should move to the goal point. You will be graded on how well your algorithm works; the efficiency of the path chosen by the robot, the ability of the robot to reach the goal point while also avoiding obstacles. *(Note that the robot's center of rotation is between its wheels not the center of the chassis so you should offset the robot in the starting cell so that the robot's center of rotation is at the center of the cell. Another technique to prevent the robot from hitting walls and obstacles is to select the path that maximizes the distance between walls and obstacles.)*

5. **Your algorithm should include a combination of odometry and reactive behaviors to avoid hitting the walls while executing the planned path.**

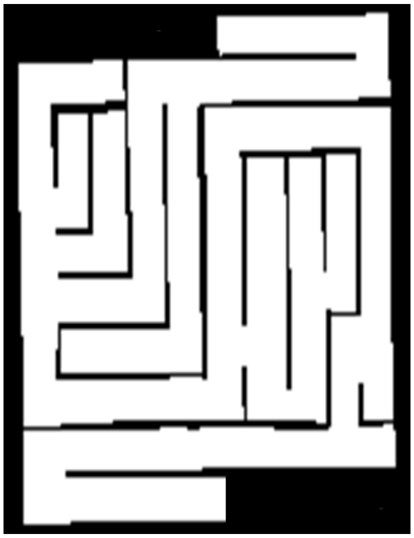
Part III - Topological Map Path Planning and Execution

1. In this exercise, you will use an a priori topological map to plan a path from the robot start location to a goal position using a wavefront algorithm. Instead of representing the world map as an occupancy grid as in last week's lab, it will be based upon the topology of the space. The salient features of the space are walls, hallways, corners and junctions. Each square will be represented by an integer between 0 and 15, dependent upon where walls are present around the square. The north (0001), east (0010), south (0100) and west (1000) walls represent one bit of that integer (see Table 2).

Table 2: Topological map coding

Integer	Binary	Hexadecimal	Direction	Wall Location
0	0000	0		
1	0001	1	North	
2	0010	2	East	
3	0011	3		
4	0100	4	South	
5	0101	5		
6	0110	6		
7	0111	7		
8	1000	8	West	
9	1001	9		
10	1010	a		
11	1011	b		
12	1100	c		
13	1101	d		
14	1110	e		
15	1111	f		

Using the coding in Table 2, the maze shown in Figure 6 is represented by an 11 x 10 matrix of integers.

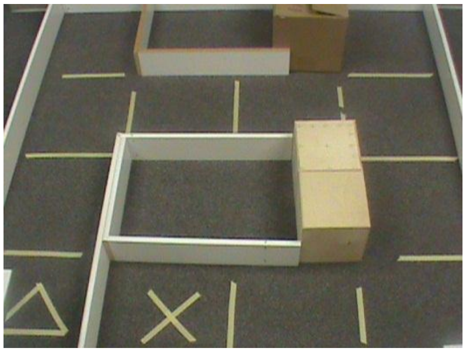


15	15	15	15	15	13	5	5	5	3
9	5	7	9	1	5	5	5	5	6
10	11	11	10	10	9	5	5	5	3
10	10	10	10	10	10	11	11	11	10
8	6	10	10	10	10	10	10	10	10
8	5	6	10	10	10	10	10	10	10
8	5	5	6	10	10	10	10	6	10
10	13	5	5	6	8	2	10	9	2
12	5	5	5	5	6	12	6	10	14
9	5	5	5	5	5	5	5	4	7
12	5	5	5	5	15	15	15	15	15

Figure 2: Maze Topological Map

2. To use the topological map to plan a path from a robot start location to a goal point it is possible to use the wavefront algorithm again. However, instead of the robot moving to cells on the occupancy grid, the robot will use behaviors and rules such as move forward, turn left, follow wall, follow hallway, avoid obstacle, etc. The navigation involves taking the list of actions and executing them.

3. During the demonstration, you will be given the map as an 4 x 4 array of integers or in a .txt file that represents the world's salient features. Figure 7 provides an example of the world representation. You will be given the robot's start position and goal position at the beginning of the demonstration. Your program should read the world map as an array, and use an algorithm to plan the path to move the robot from the start position to the goal. You should then place your robot at the start position and press start and it should move to the goal point. You will be graded on how well your algorithm works; the efficiency of the path chosen by the robot, the ability of the robot to reach the goal point while also avoiding obstacles. *(Note that the robot's center of rotation is between its wheels not the center of the chassis so you should offset the robot in the starting cell so that the robot's center of rotation is at the center of the cell. Another technique to prevent the robot from hitting walls and obstacles is to select the path that maximizes the distance between walls and obstacles or the center line (see Voronoi diagram).)*



c. Artificial world

9	3	15	15	15	15	9	3
8	2	15	15	15	15	8	2
8	0	1	1	1	1	0	2
8	0	4	4	4	4	0	2
8	2	15	15	15	15	8	2
8	2	15	15	15	15	8	2
8	2	9	1	1	1	0	2
12	6	12	4	4	4	4	6

d. Representation

Figure 7: Topological Map

Demonstration:

The demonstration of the program for Lab 6 will include three phases. In the first phase, the robot will use topological navigation to move from a start point to goal position. In phase 2, the robot will navigate from a start point to a goal point using wavefront expansion on a metric map (occupancy grid). For the third phase of the demonstration the robot will navigate from a start point to a goal point using wavefront expansion on a topological map of the world's salient features. The list of robot commands and/or generated wavefront should be shown on the LCD to make it evident the state that the robot is in or what it plans to do.

Bring your robot fully charged to class on Thursday for the demonstration. Note that you always must re-flash the factory firmware and plug in the AC adapter in order for the robot to charge. Alternately, you can put the robot battery in the RC car battery charger. Note that this is a fast charger and will not last as long as the outlet charge.

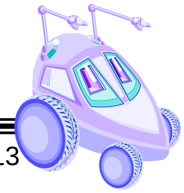
Program:

The program should be properly commented and modular with each new behavior representing a new function call. The design of the architecture should be evident from the program layout. You should use the GUI, keypad, LCD and speech module as needed to illustrate robot state, input and output data.

Memo:

The following list provides the basic guidelines for writing a technical memorandum.

- ✓ Format
  - Begins with Date, To , From, Subject
  - Font no larger than 12 point font
  - Spacing no larger than double space



  - Written as a paragraph not bulleted list
  - No longer than three pages of text
- ✓ Writing
  - Memo is organized in a logical order
  - Writing is direct, concise and to the point
  - Written in first person from lab partners
  - Correct grammar, no spelling errors
- ✓ Content
  - Starts with a statement of purpose
  - Discusses the strategy or pseudocode for implementing the robot paths (may include a flow chart)
  - Discusses the tests and methods performed
  - States the results including error analysis
  - Shows data tables with error analysis and required plots or graphs
  - Answers all questions posed in the lab procedure
  - Clear statement of conclusions

Questions to Answer in the Memo:

1. What was the strategy for implementing the wavefront algorithm?
2. Were there any points during the navigation when the robot got stuck? If so, how did you extract the robot from that situation?
3. How long did it take for the robot to move from the start position to the goal?
4. What type of algorithm did you use to selection the most optimal or efficient path?
5. How did you represent the robot's start and goal position at run time?
6. Do you have any recommendations for improving that robot's navigation or wavefront algorithm?

Grading Rubric:

The lab is worth a total of 30 points and is graded by the following rubric.

Points	**Demonstration**	**Code**	**Memo**

10	Excellent work, the robot performs exactly as required	Properly commented, easy to follow with modular components	Follows all guidelines and answers all questions posed
7.5	Performs most of the functionality with minor failures	Partial comments and/or not modular with objects	Does not answer some questions and/or has spelling, grammatical, content errors
5	Performs some of the functionality but with major failures or parts missing	No comments, not modular, not easy to follow	Multiple grammatical, format, content, spelling errors, questions not answered
0	Meets none of the design specifications or not submitted	Not submitted	Not submitted

Submission Requirements:

You must submit you properly commented code as a zipped folder of the C file and the lab memo in a zipped folder by **11:59 pm on Sunday** to the Moodle Course Drop box. Your code should be modular with functions and classes in order to make it more readable. You should use the push buttons, buzzers and LCD to command the robot and indicate the robot state during program execution.