CS 01: Term Project. Fall 2009

Robot Maze Navigation

This project will involve working in your team to build an autonomous Robot that can navigate a maze, and "learns" the maze to find the fastest way to go back to starting point.

The objectives of this exercise are:

- learn to work in teams
- design and build a fully autonomous Robot by integrating subtask modules you developed earlier,
  - integration of information from multiple sources (i.e., use multiple sensors)
  - developing and analyzing alternate solutions to a problem.

The Maze Navigation Problem

The problem to be solved is a maze navigation problem -- you will be given a maze and a starting point and a final destination point on the maze. Your job is to come up with a strategy, and implement the corresponding "algorithm", so that your Robot will go from the starting point to the goal/destination and then make its way back without making a wrong decision. You can build your Robot using any of the available sensors.

You should first think of implementing navigation from the start to the goal. Next, you will refine your solution so that the Bot can find its way back to the starting point. Ideally, the Bot should return to the starting point without making any wrong turn on the maze. Finally, you can think of "optimizing" your solution so that it will return to the starting line in the fastest time.

Options: Once you figure out a solution to solve the given problem, you can explore other "options". These options are based on using more types of sensors, and are described later in this document.

Hint: Keep it simple!

Specification of the Maze:

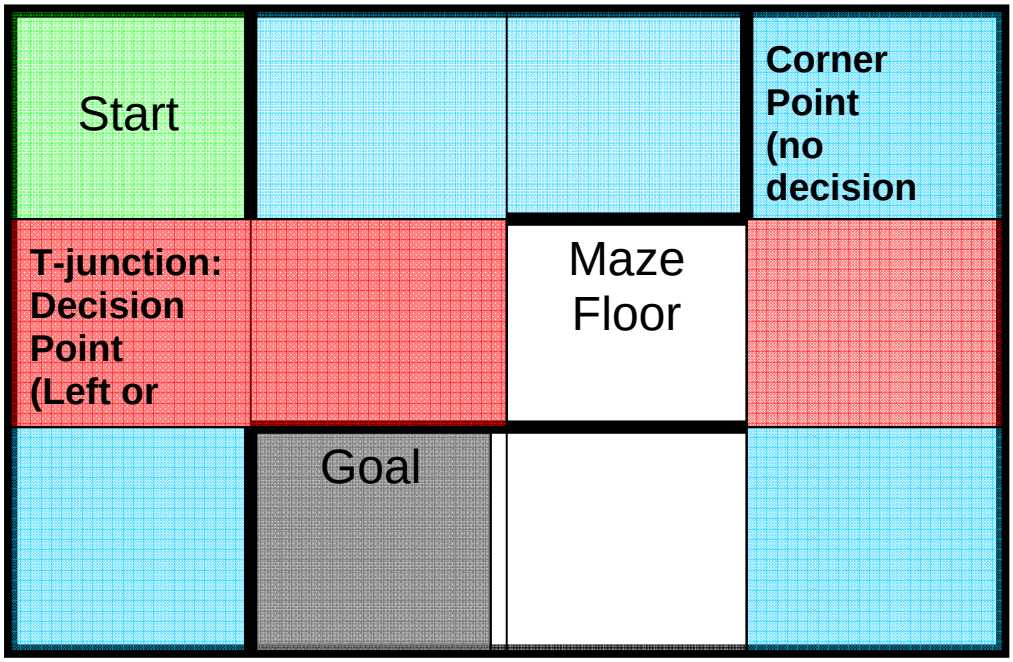
To simplify the navigation problem, you should use the following specifications for the maze. Make sure you design your solution to meet specifications.

- The maze can be described as a *M by N* grid of cells where each cell is an *X inch square*, i.e., it contains *M* rows and *N* columns and each cell is *X* inches by *X*

inches. For example we can have a 4 by 3 grid Maze, with cell measuring 11 inches by 11 inches (i.e., *M=4*, *N=3*, and *X=11*).

- There are five types of cells, and the type will be specified by the color of the cell.
  - The floor is Color A. For example, the color A could be Whie.
  - The starting point is Color B. For example color B = Green.
  - The goal (end point or destination) is Color C. For example, color C could be Black.
  - Corners are Color D. For example, these could be blue. Corners indicate that a turn needs to be made.
  - T-junctions are Color E. For example, this could be Red. T-junctions indicate a decision point where you have a choice of next steps.
    - By elimination, you could therefore assume that a cell with the base color A implies the Robot could keep moving straight (i.e., only one direction to keep moving).
- Your Robot will begin in the center of the starting cell.
- Once your bot has reached the goal cell, it should acknowledge this by playing a simple song. Once the song is finished, the timer will start to record how long it takes to return to the start cell.
- There is no limit to the number and placement of T junctions and corners. There may also be dead-ends – these are special cases of corners.

An example 3 by 4 maze is shown in the Figure below. The brown lines indicate walls, and the color assignment is Color A (floor) = White, Color B (start)= Green, Color C (Goal)= Black, Color D = Blue, Color E (T-junction) = Red. This is only an example – **the exact specification for your project maze will be provided to you (i.e., the colors and the size of the cells)**.



Using Sensors:

Your kit contains a number of types of sensors, and you have been given an additional tactile sensor. You have to figure out how many sensors and what types you will need to solve the problem – note that there are many ways to solve it, so don't worry about searching for the "one" solution. When you consider the specification of the maze, you will realize that you must use the optical sensor. However, to figure out walls and corners you can use tactile or ultrasonic sensors.

Optional Features/Scenarios:

Here are some options for your project. Think of these after you have figured out a solution strategy for the basic problem.

- Sending "correction" sound commands: Can you use the acoustic sensor to send commands to the robot to either navigate it or preferably to correct it ? For example, if you see it going in the wrong direction then you can say clap your hands to tell it to correct its decision.
- Remote commands using Bluetooth: Can you use the Bluetooth capability to remotely send commands (or even to control it entirely) to your Robot ? You cannot assume that the remote controller can see the full maze.
- Communication between Robots: Can you use the Bluetooth capability to get two Robots to communicate ? If you can figure this out, then think of sending commands from one Robot to another to help the second Robot navigate from Start to Goal without making any wrong turns/decisions.

Final Presentation Requirements:

The final presentation is scheduled during the last class (November 13th). For your final presentation, each team is required to:

(1) give a presentation describing how you solved the problem and lessons learnt, and
(2) demo your robot navigating from the start to the finish line, and back to the start.

This is not a trivial assignment, you are strongly encouraged to begin working on this project as soon as you finish Lab this week. There will be discussions of the project during the class on October 30 and Nov.6th. You are required to come to the October 30th class with a draft "solution", i.e., strategy for how you will solve the problem. Specifically, think of the sub-problems you need to solve, and any assumptions you need to make about the model. Make sure you meet with your team members at least once before the next class – each team must inform the instructor of their meeting time.

Check the links (on the course webpage) to the previous years project demos to see some working projects.

Your Next Steps:

- Set up meeting times for your team
- Read the project description carefully and formulate solution strategy
- Design and build the basic modules (methods) that you will use
- Test your code, and then test your code some more
- Test your solution on the trial maze