Final Project Report

Cole Sturza: cost5824@colorado.edu
Alex Book: albo6624@colorado.edu
Justin Vuong: juvu0180@colorado.edu
Jacob Christiansen: jach7037@colorado.edu

Abstract

We will be creating a robot that maps and navigates around a maze, as well as completing three search and rescue missions while in the maze. The maze will have three different color cubes, and matching color buckets to hold those cubes. The robot should find all 6 buckets/cubes and place the appropriate cube in the corresponding bucket.

Raised Alarms and How We're Addressing Them:
After a cursory search/attempt to implement arm kinematics, we will very likely adjust our project to be an e-puck that performs a similar task as above. The e-puck will map the maze, finding 3-5 colored spaces, then attempt to traverse the maze to each color in a provided sequence.

Equipment

The robot we will use is the e-puck that we have used for the labs thus far. This will allow cameras to see walls, as well as light sensors to see colors on the floor tiles.

Possible Raised Alarms and How We will Address Them:
If updating the odometry becomes an issue after traversing the maze for a long period of time, we will add a gps and compass to the e puck. This will allow us to have an accurate idea of where the e puck is located and limit the error introduced from the odometry.

Other possible solutions could be to look into more advanced odometry calculations.

Deliverables and Implementation Plan

- Create maze and colored tiles within it - Jacob (12/2)
    - Create physical "board"
    - Create walls that form a predefined maze
    - Create 3-5 differently-colored tiles that serve as the destinations to be traversed through
- Create a CV algorithm to determine the colors of the different tiles.- Cole (12/3)
    - Research algorithms/techniques to accomplish this task.
    - Figure out how to use the Webots camera.
    - Test that it can appropriately label a tile.

    - Test that it can differentiate between the colors (red, green, blue, etc.).
- Create mapping system - Justin (12/2)
    - Use something along the lines of this video: https://www.youtube.com/watch?v=O7pMHx09Z_w
    - Map the locations of each tile using the CV algorithm to identify their locations.
    - Test that the robot can discover the entire environment.
    - The robot will keep track of the walls it has discovered, the locations of all the colored tiles, and the open tiles.
- Create decision-making system to determine shortest path to a given tile - Alex (12/2)
    - Use Dijkstra's algorithm to determine the shortest path from one tile to another.
    - Test that if given a map that the robot can navigate from one starting tile to a destination tile.
- Create logic for order of operations - Cole (12/4)
    - Test that the robot maps the maze first then completes the navigation task.
- Research more advanced odometry calculations and implement one - Jacob (12/2)
    - Look into the possibility of using a better mechanism for updating the odometry of the robot's location.

Demo

For the demo we will have three worlds with different layouts to demonstrate that the e puck is not hardcoded to one specific environment. We will show a graphical representation of the map once the robot has mapped its environment. We will then show how the system determines the shortest path between two vertices. We will then show how the robot drives between the goal locations in the proper order.

Deliverables

- Create maze and colored tiles within it - Jacob (12/2)
    - Completed.
- Create a CV algorithm to determine the colors of the different tiles - Cole (12/3)
    - Works
    - Uses code from HW4 to determine if a picture from the epucks camera is either red, green, blue, or yellow.
        - Images are only captured if a wall is currently present in front of the e-puck.
- Create mapping system - Justin/Jacob (12/2)
    - Works. Could be improved on.
    - Looks for possible paths from the current position and weights them based on if they have been visited. Unvisited tiles are weighted more heavily than visited tiles. The target tile is selected at random using the weights.
    - The e-puck uses a lidar sensor with a 180 field of view. The sensor checks if there is a wall to the left, right and center of the e-puck. Mapping is only

performed at the center of tiles to avoid error introduced by the corners of the maze.
    - If a wall is in front of the e-puck the CV algorithm is called to determine its color.
    - After all tiles have been visited the e-puck returns to the starting location and begins the color traversal.
- Create decision-making system to determine shortest path to a given tile - Alex (12/2)
    - Works.
    - Simple Dijkstra's algorithm to determine the shortest path from one tile to another.
    - Uses a graph representation of the maze.
- Create logic for order of operations - Cole (12/4)
    - Works.
    - The e-puck first maps the maze, then returns to the starting location, then traverses the maze to the colors in the correct order.
- E-puck movement - Cole
    - Works. Could be better.
    - We used a GPS and Compass to retrieve the e-pucks position and facing. Then used the simple bearing error and distance error approach from lab 3. The e-puck first corrects for bearing and then for distance. It moves from the center of a tile to the center of another tile.
    - If we had more time it would have been cool to use a finer map for the maze and more advanced movement mechanics.
    - The robot sometimes rotates in the wrong direction and ends up spinning longer than it needs to (i.e. counter clockwise vs clockwise).

Overall, we accomplished the objective to have a robot map a maze, find goal locations using CV, and traverse the maze to those goal locations in a provided order.

If you run the code, you need to turn shadows off. The robot has trouble detecting the colored walls with shadows.