Robótica Móvel

Assignment 2

98430: Paulo Pereira

Topics

1 Objective

2 Solution

3 Conclusion

Objective

Develop a robotic agent that implements a set of robotic tasks

- **Localization->** Agent needs to localize itself in a maze using a movement model, a compass and the linesensor.
- **Mapping->** Needs to extract its map and identify beacons in the map
- **Planning->** Compute a closed path with minimal cost that visits all beacons and starts/ends at starting spot

Solution

build.sh

Before we started implementing our robot we were asked to create two bash scripts, one would build the agent and one that would in fact run the agent for the C4 challenge.

The build.sh file is merely a script that creates a environment and installs the libraries the robot will need in the future.

Solution

Run.sh

The second script consisted of creating a bash script that would run the agent itself with customizable arguments.

- c -> The challenge number
- p-> starting position of the robot
- r-> the robot's name
- h-> its host
- f-> the file name of the .map and .path files

Solution

Run10.sh

As a bonus i also created a script that runs the scenario 10 times and returns the average of the mapping and scoring as a way to track my progress

Solution

mainRob

This is the main robot's file. We firstly initialize the robot with multiple variables that will be used in the future, most notably the position,connections, and the seenBeacons.

The run method is mostly untouched and it simply tells the robot when to wander (the main component)

Solution

mainRob->wander

The wander function acts as the main force behind the robot as it is the function responsible for calculating how to move and what to do.

It starts by reading and printing to the console multiple sensor data, as well as parsing the line, this parsing is made by recognizing how the line works.

A line is a 7 item list where each item is the char 0 or 1 where 1 means the line is present, from the central point the items to the right are the reads from the right sensor and the ones to the left from the left sensor. This means if one side has more 1’s there is a path to that side. This would later be noticed as problematic.

Solution

mainRob->wander

After parsing the line there is a small fix to a recurrent problem that is getting stuck on corners, sometimes due to noise the robot would turn a bit more than expected and would get stuck on a turn right turn left cycle.

By introducing a grace period that after a while would just take the robot a bit forward was enough to mostly fix this issue.

Solution

mainRob->wander

The robot’s decision to turn to a side, as mentioned before, is defined by how the line is.
For example: [1,1,1,1,0,0,0] would be a left turn

But there are also cases where the robot hits a T junction, in this case the robot has a decision to make since all the values in the line sensors say true.

To know which direction to move I implemented the rule to always turn right, unless the right turn at that junction was already made, this proved to be a good idea since the map got mapped better by this.

Solution

mainRob->wander

Each time the robot wants to move it makes a call to the driveMotors function by giving the wheel speed of each wheel.

After each driveMotors it will update the pose based on the movement model as defined in the assignment.

Solution

mainRob->wander

Finally I identify the beacons by reading the ground sensor (it changes from -1 to the value of the beacon when passing through it)

After that two more functions were defined.

DrawMap and CreateSmartPath.

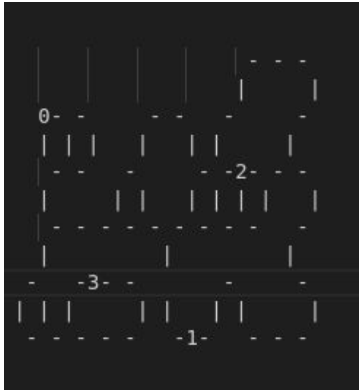
These functions would activate after the 5000 timeout

Solution

drawmap

Firstly we convert from the simpler reads of the sensors and position calculations to the final file format that was defined in the assignment.

Then we create a map structure that looks something like the following



```
                   - - -
                  |     |
  0- -     - -   -     -
  | | |   |   | |     |
   - -   -     - -2- - -
  |     | |   | | | |   |
   - - - - - - - - -   -
  |         |         |
 -   -3- -       -     -
| | |     | |   | |     |
 - - - - -   -1-   - - -
```

Solution

CreateSmartPath

This function will create a graph and then displays it, this graph represents the map in a more visually pleasant way, then since we have a small amount of overall points and beacons we can calculate the shortest path between each beacon, then use the permutations of beacons to find the shortest paths.

Then the function simply combines the shortest paths between each set of beacons and writes the full path to a file.

Problems

There were some noted problems in the robot, despite the initial fix sometimes it still gets stuck on corners.

Due to the fact we are using mostly the line sensor for moving the robot will almost always turn right or left if that's a possibility, this creates loops inside the map and sometimes it can't break the loop fast enough to explore the whole map.

If not all beacons are connected the agent will crash because it can't create a path to the isolated beacon (this happens because sometimes it fails to register new connections)

Changes

From the start I noticed there were times that the robot would simply not map correctly, this was happening due to the fact that previously I wasn’t updating the old column and row values and there were cases (specially around the rectangle below the second beacon) where there would be extra connections.

This was fixed and now there are very rare cases where the robot will simply fail completely at mapping and I am still uncertain what would cause this issue.

Changes

Another problem that was happening previously was the fact that the scoring script was always returning an error or simply 0, this happened because the path that was being written to the file was considering the 4th quadrant of the 2D euclidean space, instead of the 1st.

This was changed and the robot is almost always being scored.

Changes

The robot also stopped being stuck on corners this happened due to the fact i changed the threshold values for rounding as well as fixing the mapping issue.

Changes

Now if the robot only detects 1 beacon found (the starting one) it will not crash and produce an empty path file.

Problems -summarized

There were some noted problems in the robot, despite the initial fix sometimes it still gets stuck on corners. **(FIXED)**

Due to the fact we are using mostly the line sensor for moving the robot will almost always turn right or left if that’s a possibility, this creates loops inside the map and sometimes it can’t break the loop fast enough to explore the whole map.

If not all beacons are connected the agent will crash because it can’t create a path to the isolated beacon (this happens because sometimes it fails to register new connections). **(FIXED)**

Persistent Problems

Due to the fact we are using mostly the line sensor for moving the robot will almost always turn right or left if that’s a possibility, this creates loops inside the map and sometimes it can’t break the loop fast enough to explore the whole map.

Despite a lot of attempts I couldn’t figure out how to solve this problem.

I tried to detect corners using the line sensor but that proved ineffective due to the fact that the line sensor returned a lot of values (specially during curves) that would become corners even when they weren’t for example if the robot came out of a corner a bit sideways and was correcting itself it would register a corner again.

Persistent Problems

To also solve the previous issue i tried to make the robot rotate in place every corner.

Since corner detection was very hard using a line this way the robot would rotate at random places.

Another solution was to use the changes in orientation to find corners, this would prove ineffective because due to the pose not being super precise the rotation wouldn’t detect other paths.

The rotation itself which seemed like a great idea was indeed inherently flawed because rotating with the motor noise would cause disturbance in pose calculation and therefore the mapping would severely suffer.

Currently lacking

To summarize what the agent can’t do:

- Return to start after exploring
- Correctly explore the whole map
- Understand map logic and consciously take different routes

Conclusion

To sum up the project, despite not being able to fully make the robot “smart” about it’s mapping it is quite good at it.

There will also be provided the new scripts.

The results I got were quite alright with 0.3 average pathing score and 560 average mapping score

All files provided in the agent folder should be inside of pClient folder (in ciber rato environmen)