Automated Robo Taxi

Michael Annor and Agatha Maison

Introduction

-This project models an automated taxi system that efficiently finds the quickest route for passenger pickups and drop-offs.

-The project was implemented using Lego Mindstorm kits. The system includes a base station that receives and processes requests for rides from passengers.

-Processing of these inputs is done using the A* path planning algorithm to determine the closest taxi to deploy to the requesting passenger.

Introduction

-The taxis would then determine the shortest routes to the passenger and from the passenger to the point of destination (also using the A* path planning algorithm).

-The taxis and base station involved communicate via Bluetooth and the base station receives passenger travel details using the Mindstorm numeric keypad.

Background

- Considering the advent of driverless cars and unmanned vehicles, this was a foundation for understanding this new field.

- Inspired by a paper review that was done earlier in the semester (Kumar and Kumar, Automated Taxi/Cab System Using A* Algorithm)

- Taking a robotics class that gave us insights on intelligent systems

The System

Automated Taxi

Base Station

Approach

-Decide on the planning algorithm

-Implement the A* algorithm

-Modelled in Microsoft Excel

-Explore peripherals

-Explore connection options

-Develop modules

Result

- Bluetooth connection successful for inter-robot communication

- A star pathfinding algorithm successful

- Mindsensor keypad successfully used to input data

Future Work

-Additional functionality for real-time multi-taxi navigation

-Explore other algorithm options like the D* Lite algorithm

-Explore other robot platforms like the turtlebot

A* Algorithm

```
OPEN
CLOSED

add the start node to OPEN

loop
        current = node in OPEN with the lowest f_cost
        remove current from OPEN
        add current to CLOSED

        if current is the target node
                return

        foreach neighbour of the current node
                if neighbour is not traversable or neighbour is
in CLOSED
                        skip to the next neighbour

                if new path to neighbour is shorter OR
neighbour is not in OPEN
                        set f_cost of neighbour
                        set parent of neighbour to current
                        if neighbour is not in OPEN
                                add neighbour to OPEN
```