Assignment 3

Dr. Bastani

Scenario

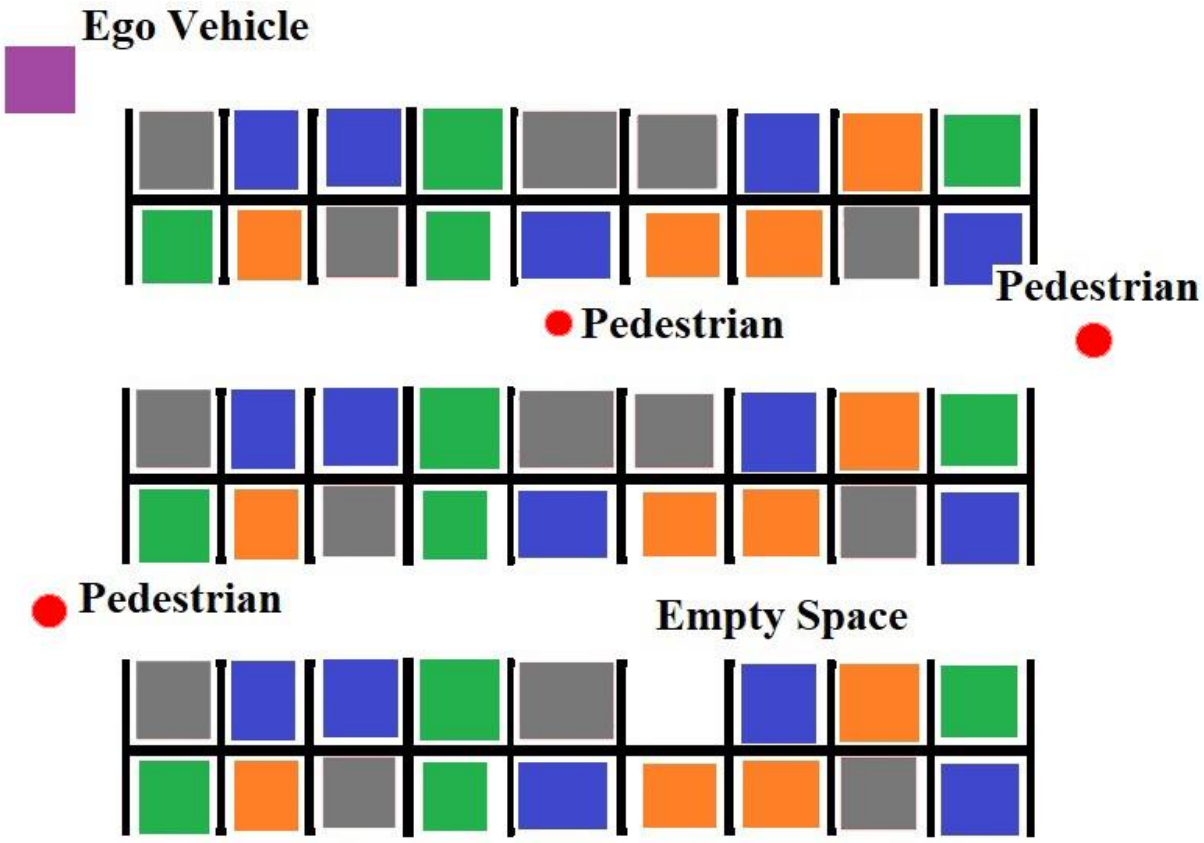

Figure 1

- **Goal**: Move the Ego vehicle through the parking lot and into the empty space.
- **Constraints:**
  - Use only the sensor data which is given through the RTOS.
  - Use only the actuators that are given by the RTOS.
  - Stay in the parking lot
  - Don't hit the parked cars
  - Don't hit pedestrians

Sensors

- DeviceRegsitry.pixels : Readings from a 5 pixel by 5 pixel color camera
- DeviceRegistry.lidar : Readings from the 16 lidar sensors placed in a circular ring around the Ego vehicle
- DeviceRegistry.compass : Readings from a compass that tells you how different your current orientation is from North.
- DeviceRegistry.speedometer : Readings from a speedometer that tells you how fast the vehicle is moving in m/s.
- DeviceRegistry.gps : Readings from a GPS device that gives the latitude and longitude of the Ego vehicle. Uses meters instead of degrees.

Actuators

- DeviceRegistry.speedControl : The control bus for the acceleration subsystem that controls the speed of the Ego vehicle.
- DeviceRegistry.brakeControl : The control bus for the brakes that can slow down the Ego vehicle.
- DeviceRegistry.steeringControl : The control bus for the steering controller that changes the direction the Ego vehicle is facing.

Additional Environment Information

There are some other entities and objects in this scenario that you must search for or otherwise avoid. Details about them will be documented here.

Moving Cars

The parking lot is filled with obstacle cars. Every few seconds a random car will slowly reverse out of its space, and then it will move back to its original position. Don't let the Ego vehicle touch them however, or the owners may sue you!

If the Ego vehicle is behind a car when it begins to reverse, the car will cancel its movement and return forward to its original position. Thus no car should ever accidentally back up into the Ego vehicle (if this does happen however, its not your fault, so you will lose no points). The main point is to avoid driving the Ego vehicle into an obstacle car.

Pedestrians

There are several pedestrians that are randomly placed throughout the parking lot. These pedestrians will sporadically move towards random locations in the parking lot, and once they reach their target destination they will choose a new target destination and begin moving towards it as well. You must avoid hitting these pedestrians – you won't just be sued for money, you will likely also be arrested and charged with a crime. Since you are the person who designed the Ego vehicle, it is only fair that you face the consequences for the behavior that you programmed into it!

If a pedestrian is walking towards the Ego vehicle, then it will stop moving once it is within 4.1 meters of the Ego vehicle. The pedestrian will spend a few seconds to admire the Ego vehicle, and then the pedestrian will turn around and walk in the opposite direction. This will prevent a pedestrian from running into the Ego vehicle from behind or from the side – thus you only need to worry about hitting a pedestrian from the front.

Empty Parking Space

At the beginning of the simulation, a random obstacle car is deleted, thus leaving a single empty space. It is the goal of the scenario for the Ego vehicle to successfully navigate to this empty space and occupy it. The empty space can occur in all positions, so you will need to be able to account for all the possible sensor readings.

Problem 1: Parking Search (20 points)

The Ego vehicle will need to search the parking lot for an empty parking space. The location of the empty space is randomized, and so is the starting position and orientation of the Ego vehicle.

For now, focus on creating the tasks necessary solely for avoiding parked cars while searching – you might find it helpful to simply ignore the pedestrian-avoidance constraint for now. You are allowed to reuse your old code from Project 2.

You are free to employ whatever search strategy you wish. A simple approach for solving these kinds of problems is known as the "wall follower strategy", although some people call it "the right hand rule". The idea behind the strategy is simple:

- Place your hand on the right wall of a maze
- Move forward along the wall of the maze, and never remove your right hand from the wall.
- Continue moving until you reach the goal position.

This strategy will likely require some modifications to deal with the fact that the “walls” of the maze are non-contiguous. You can also review this Wikipedia article to learn about other strategies for solving mazes:

- https://en.wikipedia.org/wiki/Maze-solving_algorithm

Problem 2: Obstacle Avoidance (35 points)

There will be numerous pedestrians which are walking around the parking lot. Also, some vehicles will randomly reverse out of their parking space before driving forward back into them.

Pedestrians are randomly placed, and they will follow random paths. If a pedestrian or another car is hit by your Ego vehicle, they will likely sue both you and the company. You will be fired, and you might even potentially face jail if a judge decides that you engaged in gross negligence while developing your control tasks.

Thus, it is very important not to hit any of these obstacles. If a pedestrian is walking towards to your vehicle, they will turn away once they get too close. The other vehicles will only reverse out of their parking space before immediately moving back into it, and they will not drive around the parking lot. You may employ any avoidance strategy you wish, so long as you avoid hitting an obstacle.

Problem 3: Parking Procedure (35 points)

Once the empty parking space has been found, the Ego vehicle will begin to maneuver itself into the empty space. While doing so, the Ego vehicle was still avoid hitting pedestrians or the other vehicles on either side of the parking space.

You are free to use whatever parking strategy you wish.

Problem 4: Documentation (10 points)

You must document and describe your implementation in a final report. This documentation must include:

- Information about the tasks which you defined to control the Ego vehicle
    - Task name
    - Task description
    - Dependencies on other tasks and/or sensor data
- A graph showing the task schedule for at least two cycles of the simulation.
    - Describe the graph. Point out which tasks are being executed, and explain at what point in the simulation the task is occurring at.

Please note that while bad documentation may lose you at most 10 points, a complete lack of documentation will result in an automatic -50 points penalty.