

Assignment 2

Dr. Bastani

Scenarios

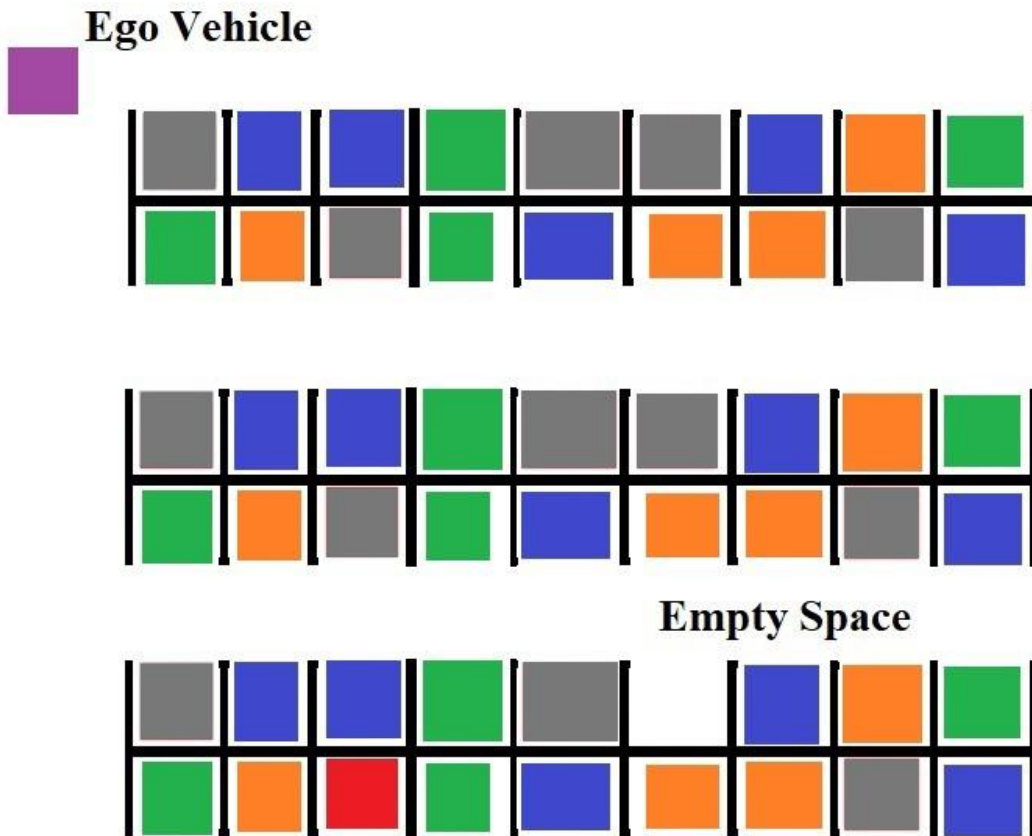


Figure 1

- **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

- DeviceRegistry.pixels : Readings from a 7 pixel tall by 15 pixel wide 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.

Color Meanings

Each pixel of the camera contains an RGB color that was detected by a raycast sent out by that pixel. The objects in the scene all have specific colors assigned to them, although not all colors are unique.

<u>Color (RGB)</u>	<u>Objects</u>
 (0, 0, 0)	Empty Space (No object detected)
 (24, 24, 24)	Parking Lot Ground
 (255, 255, 255)	Parking Lot Lines
 (255, 244, 0)	Walls
 (0, 124, 255)	Obstacle Car
 (0, 255, 27)	Obstacle Car
 (172, 0, 255)	Obstacle Car
 (192, 192, 192)	Obstacle Car
 (101, 33, 0)	Obstacle Car

Obstacle Cars

The parking lot is filled with obstacle cars. These cars do not move, and they functionally serve as the walls of a maze. Don't let the Ego vehicle touch them however, or the owners may sue you!

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 (45 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.

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

Remember though – you are free to follow whatever strategy you wish.

Problem 2: Parking Procedure (45 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 must still avoid hitting the other vehicles on either side of the parking space.

You are free to use whatever parking strategy you wish.

Problem 3: 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.

Submission Requirements

To receive credit, you must submit the following items within a ZIP file:

- All .cs files that define the classes of your custom tasks.
- The updated TaskList.cs file that adds your custom classes to the Task List.
- Your documentation.
- A text file that contains the NET IDs of all group members.

To be safe, all group members should submit the same ZIP file. If you have an issue with a group member who does not contribute, contact the TA beforehand.