TÉCNICO LISBOA

MSc Electrical and Computer Engineering

ROBOTICS

Lab 2 - Autonomous Cars
Script

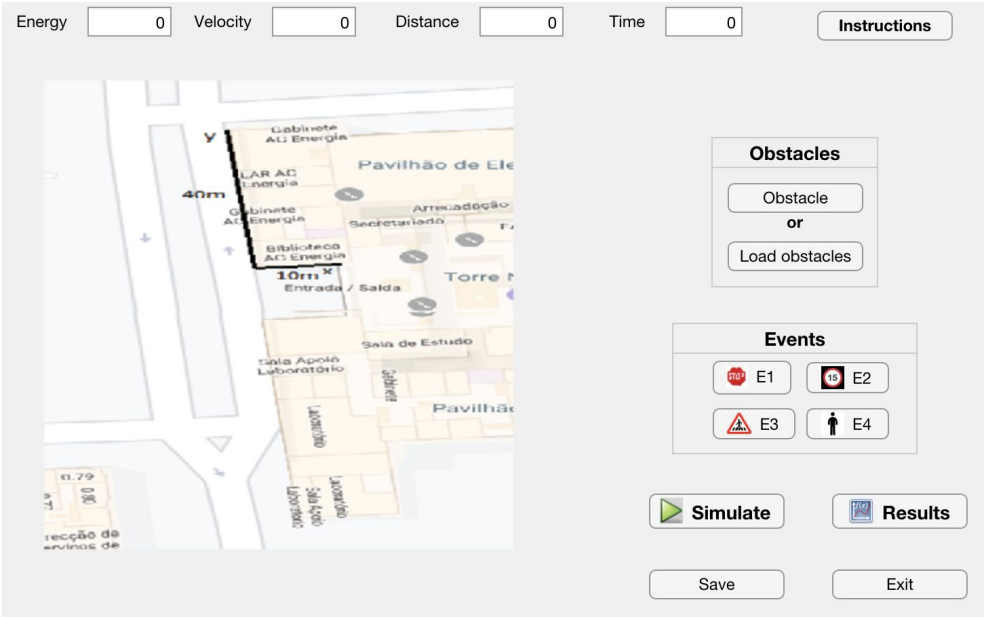
Grupo:
Carlos Silva, 81323
Gonçalo Pereira, 81602
Ivan Andrushka, 86291

Professor:
Prof. João Sequeira

22nd May, 2020

Instructions

- Once you open the app the following screen will appear:



This figure will be used as a reference until the rest of the instructions.

1. A summarized version of the instructions can be found by clicking in the button in the top-right corner of the app window. A .txt file will be opened in the built-in MATLAB text editor.

2. If this is the first time using the app, we provide a file ‘map.obstacles.mat’ containing pre-selected obstacles. This is because the selection of the obstacles may take some time and thus you can start the simulation right away. To use these obstacles, press the “Load obstacles” button.
    - If you wish to select your own obstacles, start by selecting the corners of each object (one at a time). Once you’re done with one object, press any key on the keyboard or right click to continue the selection of objects.
    - Finally, if you want to save this selection in a ".mat" file, press the "Save" button. A file with the name "map_obstacles.mat" will be created. This will overwrite any files existing before.

3. Select the desired location of each event. You have four events possible:
    - E1 Stop traffic sign, at a fixed location.
    - E2 Speed limit traffic sign, at a fixed location.
    - E3 Pedestrian crossing traffic sign, at a fixed location.
    - E4 Pedestrian crossing at a pre-specified location and time; the duration of the crossing is also pre-specified.

    After placing events press any key on the keyboard or right click to confirm.

4. You are now in conditions to start the simulation. Press the "Simulate" button and indicate the amount of energy available to the car as well as the value of the P0 constant.
    - A pop up figure will show up (it may take a while before it appears) and now you must select multiple points through which you want the car to pass using the left mouse button. Press any key to continue to the simulation.

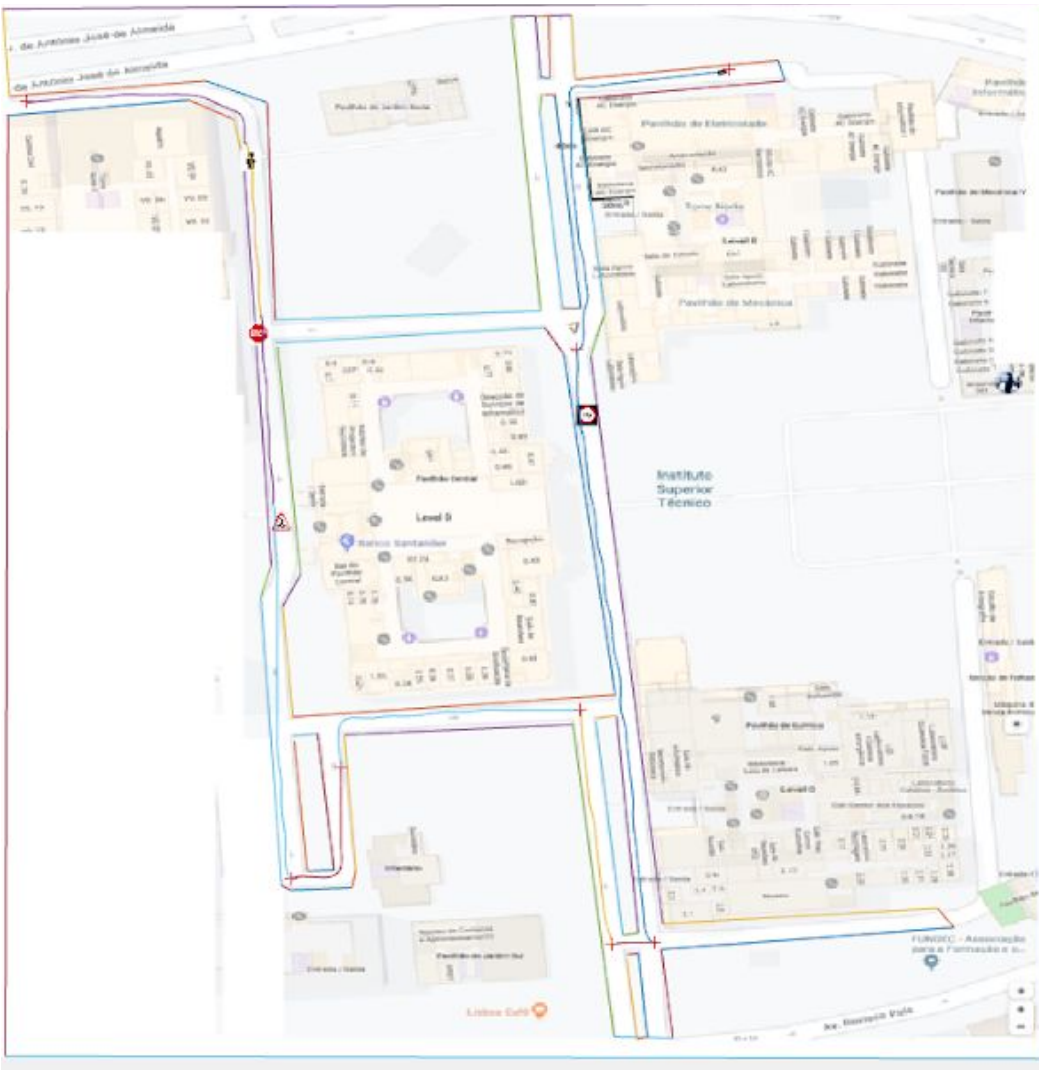
5. The simulation will now run in the UI window. Several performance metrics are available, such as the amount of energy available, the linear velocity (scaled) that is fed to the car, the distance covered between any 2 configurations and finally the time elapsed.

6. If the car runs out of energy, a message box will show up and the simulation comes to an early end.

7. Once the simulation is over, you can check the plots of the controller outputs (v, ω_s) and the look ahead errors.

8. At any moment, you can leave the simulation by pressing the “Exit” push button.

Tests



The car follows a path created by the selected points (represented by a +).
When the car reaches the “stop sign” event it stops for a fixed amount of time before proceeding.
When the car reaches the “pedestrian crossing sign” event it limits its velocity to 20km/h for a set amount of time before proceeding with its regular velocity.
When the car reaches the “speed limit” event, the velocity is capped at the amount defined by the user. The maximum velocity was predefined by us as 50km/h.
When the car encounters the “pedestrian crossing” event at the specified time, it stops for the amount of time set by the user before continuing with its usual behaviour.

The path generation module works by applying the quadtree decomposition algorithm on a grayscale map. On the resulting graph, the optimal path is found with Djikstra.

In terms of the low level controller of the car, i.e, the actual movement of the car, it follows the trajectory relatively well, aside from some oscillations. We used a non-linear controller and since we had to manually select the gains the optimal movement of the vehicle is not achieved.