Automated driving toolbox tutorial

The intention of this tutorial is to present the most important functions of the automated driving toolbox relevant to the TP. Note that the main objective of this toolbox is to create a driving scenario where vehicles move along some pathway in which there is an access point. The output of such a driving scenario is the positions of all actors (vehicles) at each time step of the simulation.

Part 1: Getting started with Automated Driving Toolbox of MATLAB

First, let's load the scenario provided. To do so, download all given files available for this TP at the EDUNAO platform. With MATLAB 2021a open, browse to the same folder where you just downloaded the files. One way of doing it is to click "*browse for folder*" and select the folder with visual support.

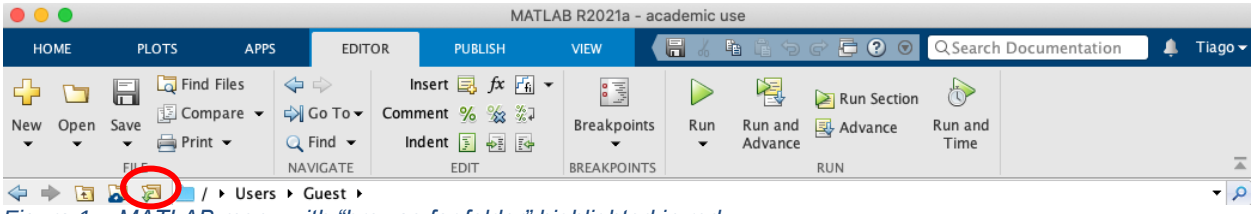

Figure 1 – MATLAB menu with "browse for folder" highlighted in red.

Once you are there, you can open the 'scenario_5vehicles.mat' with the following command.

```
drivingScenarioDesigner('scenario_5vehicles.mat');
```

The following screen should appear. Otherwise, make sure you select "roads", "scenario canvas" and "ego-centric view" as shown below.

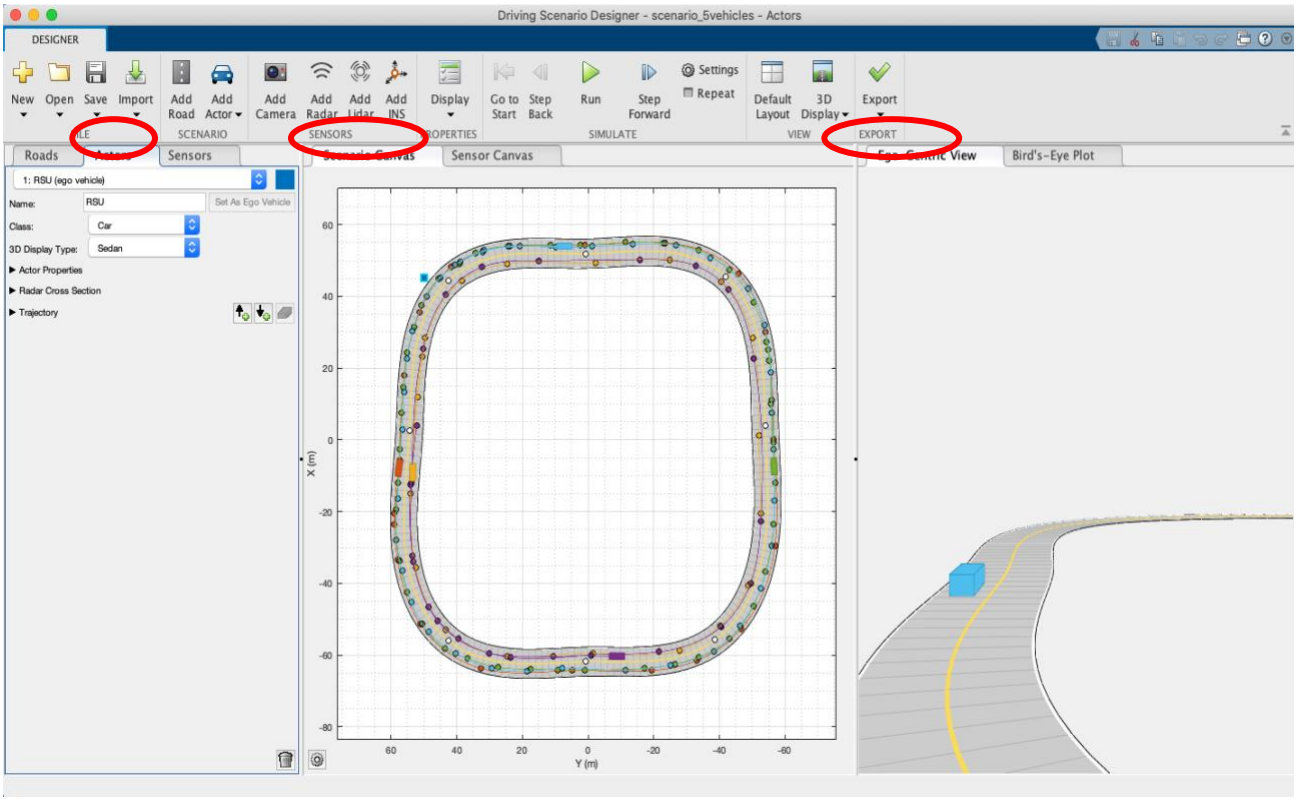

Figure 2 - Automated driving toolbox main screen.

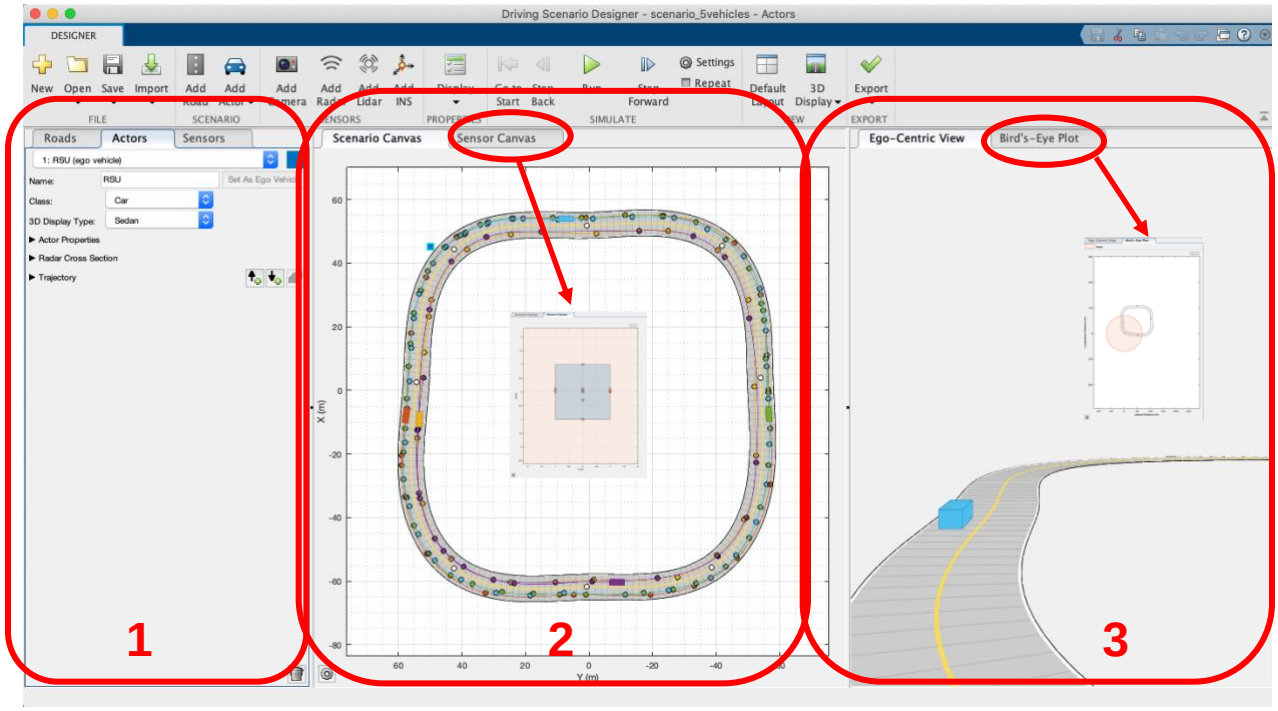

Figure 3 – Different sections of the driving scenario.

Notice that we have 3 main sections,

1. The first section is responsible for modifying the road, actors (vehicles), and sensors' parameters.
    a. Roads
    b. Actors (vehicles)
    c. Sensors
2. The second section is divided in:
    a. Scenario canvas: used to inspect the scenario and the vehicles dynamics in a global perspective.
    b. Sensor canvas: shows where the sensors are placed in respect to the vehicle.
3. The last part is responsible to inspect in a more detailed view the scenario.
    a. Ego-centric view: shows a more detailed view in respect of the ego vehicle (defined in Actors by 'Set as ego vehicle').
    b. Bird's-eye plot: it shows the range of the sensor measurements in respect of the ego vehicle.

In order to add a vehicle in the scenario, you must: Add actor => Car as in (1). Then choose the coordinates to place the vehicle. Next, click in “add new *forward* waypoints” as shown in (2) and define trajectory as in (3) in the figure below. Remark that for simplicity we have added the vehicle in the middle of the scenario, make sure to place it properly on the highway.

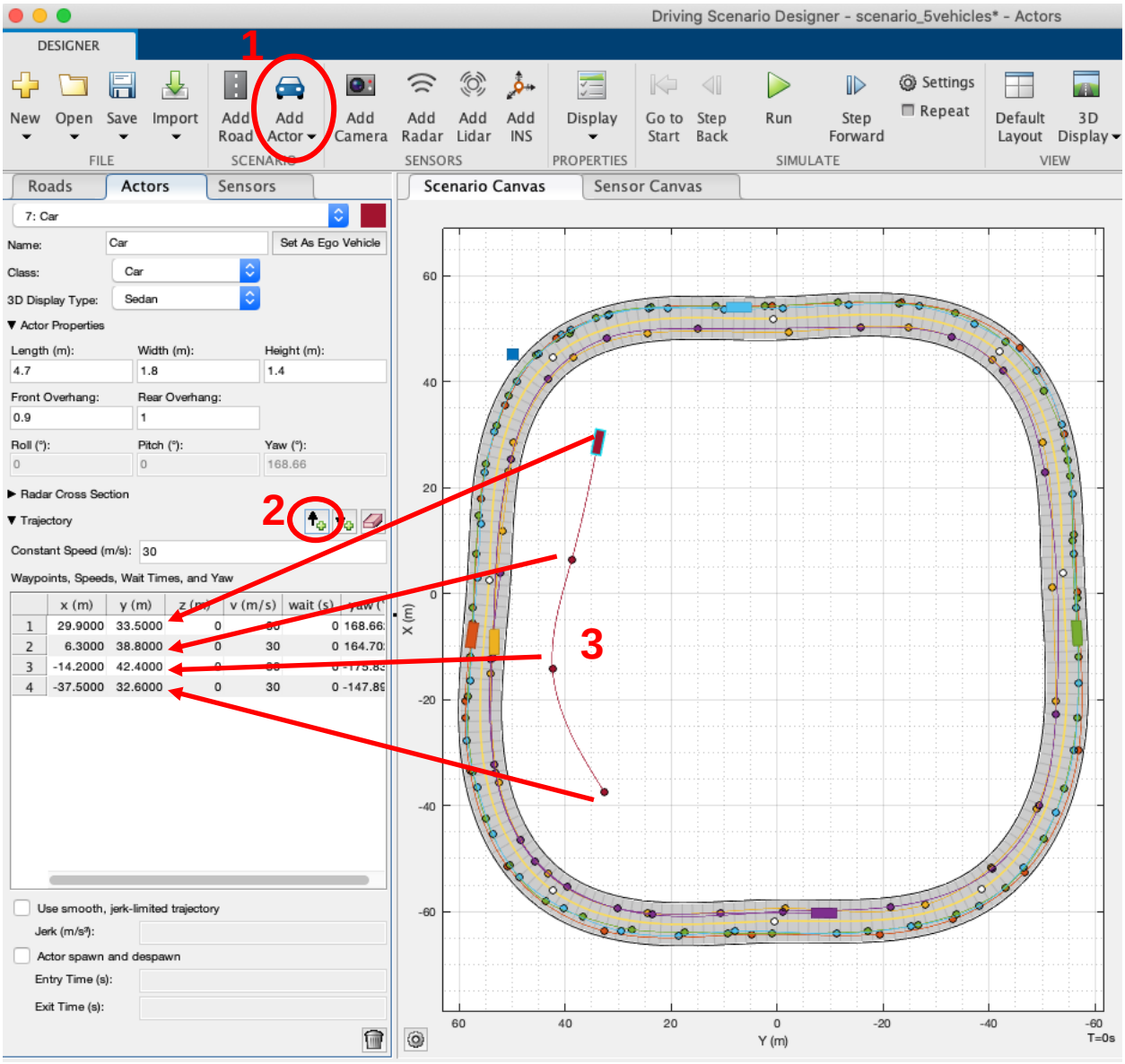

Figure 4 – Adding a vehicle and its waypoints in the scenario.

Those are waypoints that are coordinates in the (x,y,z) axis that is used to define the vehicle position at each point of the simulation. The toolbox is responsible for interpolating and finding the corresponding midpoints. Make sure to add enough points so the vehicles can perform curve trajectories properly.

After **running the entire simulation**, we must export the sensor data to the MATLAB workspace. Click Export > Export Sensor Data, enter a workspace variable name (e.g. “sensor_data”), and click OK.

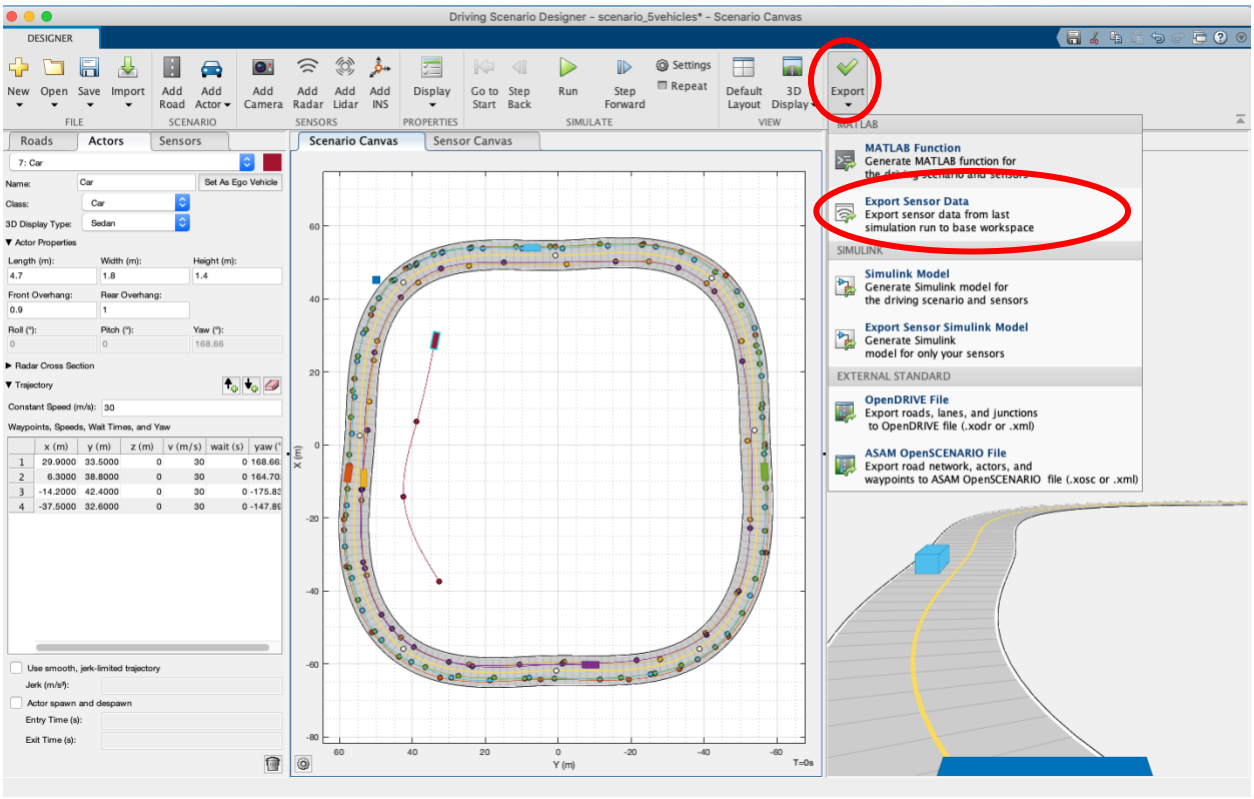

Figure 5 – Exporting sensor data of the driving scenario.

Part 2: Evaluating the simulation data

Now, we must treat the simulation data to obtain useful information to resolve the TP. First, let’s check how it looks like the exported file which contains the simulation data. To do so, double-click the file “sensor_data” in the workspace in Matlab as (1) in the Figure below.

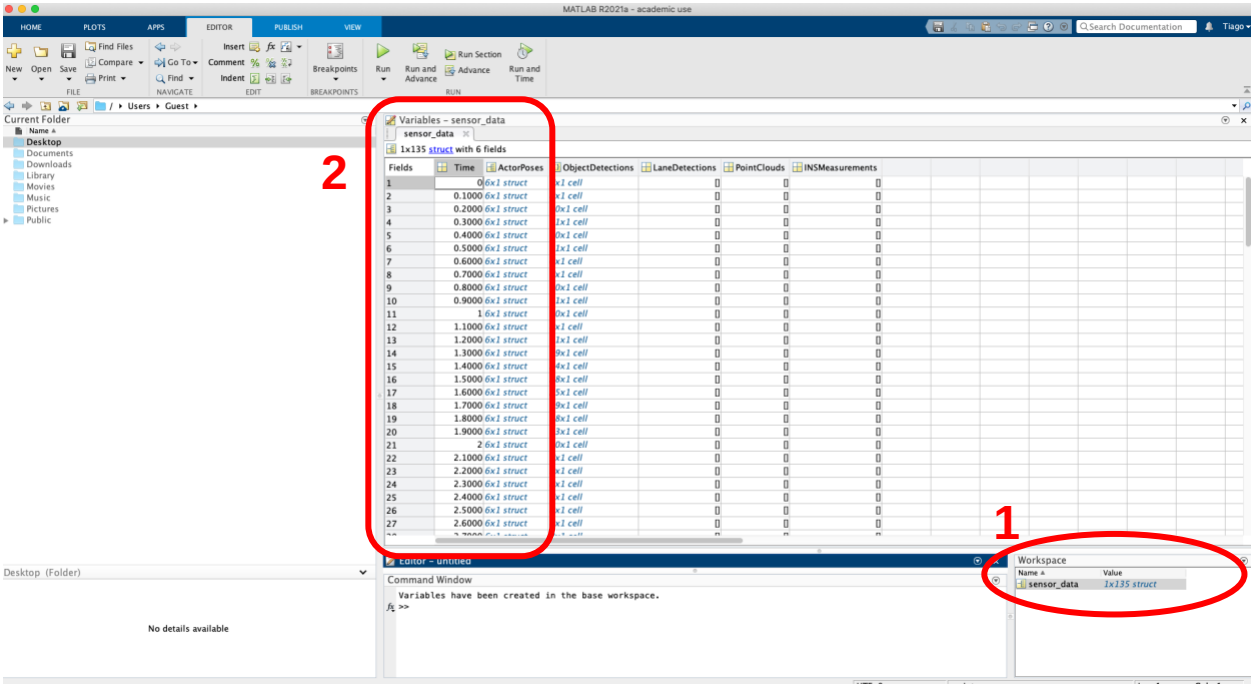

Figure 6 – Evaluation of the exported variable.

We can see there are many fields (time, actorposes, objectdectations, etc), but, for now, we just need the ones highlighted in (2).

Now you have two objectives: (1) to compute the relative distance and (2) the number of vehicles in the range of the AP at each time-step from all vehicles to the RSU. To facilitate, we have provided the function `vehicle_position.m` as shown next. Remark that a scale is needed to cope with realistic distances.

```
function [counter,relative_position_AP] = vehicle_position(strData)
% Input : 'exported_file.mat' file (struct generated by Driving Scenario
Designer after simulation -> Export -> Export Sensor Data)
% Output : counter = counter array the number of vehicles in the range of the
AP indexed by time
%           relative_position_AP = relative distance from each vehicle from
the AP indexed by time

%initialization of scenario
data = load(strData); %loads the scenario file into the variable data
fild_name=fieldnames(data);
simulation = data.(fild_name{1}); %simulation variables indexed by time

%paramters initialization
K = length(simulation); %number of simulation points
position_RSU = simulation(1).ActorPoses(1).Position; %take the fixed position
of the AP

for i = 1 : K

    %Compute the relative distance from each vehicle from the AP


    %Compute the number of vehicles in the range of the AP

end

end
```