Programming Assignment: Final Project: Self-Driving Vehicle Control

Passed · 1/1 points

Deadline The assignment was due on Nov 16, 1:59 AM EST
You can still pass this assignment before the course ends.

Instructions

My submission

Discussions

In this project, you will write and implement a controller for the CARLA simulator. Your goal is to control the vehicle to follow a race track by navigating through preset waypoints. The vehicle needs to reach these waypoints at certain desired speeds, so both longitudinal and lateral control will be required.

To do this assignment, the CARLA simulator along with the assignment code needs to be installed. Please follow these instructions:

1) Follow the CARLA Installation guide from a previous reading to install the CARLA simulator

2) Download the following "Course1FinalProject.zip" file and unpack into the subfolder folder "PythonClient" inside the "CarlaSimulator" (root) folder. This will create a subfolder "Course1FinalProject" under "PythonClient" which contains the assignment files.

Course1FinalProject.zip

It is very important to have the contents of "Course1FinalProject.zip" be under the folder "PythonClient\Course1FinalProject" (for Windows) or "PythonClient/Course1FinalProject" (for Ubuntu). Installing it into another directory might cause runtime issues.

After successfully downloading CARLA and the assessment script, you can now begin the assignment.

3) Edit the "controller2d.py" class file (found inside the "Course1FinalProject" folder, under "PythonClient"). This is where your controller will be implemented.

The "controller2d.py" file contains a controller object. You will implement the controller in the update_controls method labeled by the comment blocks (find all comment blocks that contain "MODULE 7" - these serves as anchor points for you to implement your code below them).

The controller provides you with the following relevant information required for its implementation. All units are in **SI** (meters, seconds, radians), and CARLA works in the **left-handed coordinate system** (due to the Unreal Engine adopting the left-handed coordinate system**)**. This generally shouldn't be an issue for this assignment since the controller operates in the x, y and yaw space.

The waypoints variable is a Python list of waypoints to track where each row denotes a waypoint of format [x, y, v], which are the x and y positions as well as the desired speed at that position, respectively. An example of accessing the third waypoint's y position is:

```
1  waypoints[2][1] # Remember that Python's indexing begins at 0
```

More details on the structure of the waypoint variable is written in the comments of the controller2d.py file.

Along with the other variables, the waypoints will update on each simulation step - so please do not assume the waypoint variable never changes. Here the waypoints are a **linearly interpolated (for location and speed) subset of the entire set of waypoints** (from racetrack_waypoints.txt). In other words the waypoints variable is an enhanced (finer resolution) portion of the entire set of waypoints that is near the vehicle. This is done to reduce the computation time and the performance of the controller.

The desired speed is computed to be the waypoint speed at the closest waypoint to the vehicle.

Description	Variable Name	Units
Vehicle Location	[x, y]	[meters, meters]
Vehicle Orientation	yaw	radians
Vehicle Speed	v	meters per second
In-Game Time	t	seconds
Desired Speed	v_desired	meters per second
Waypoints to track (x, y, v)	waypoints	[meters, meters, meters per second]

Using this information for your controller, you will output vehicle throttle, steer, and brake. Details about these outputs in "controller2d.py" are as follows.

Description	Variable Name	Limits
Throttle	throttle_output	0 to 1 (in percentage)
Steering	steer_output	-1.22 to 1.22 (in radians, from left to right)

Description	Variable Name	Limits
Brake	brake_output	0 to 1 (in percentage)

You may also treat all measurements from CARLA to be with respect to the center position of the vehicle. If required, the distance between the center position to the front axle of the vehicle is 1.5 meters.

4) In one terminal, start the CARLA simulator at a 30hz fixed time-step:

Ubuntu:

```
./CarlaUE4.sh /Game/Maps/RaceTrack -windowed -carla-server -benchmark -fps
    =30
```

Windows:

```
CarlaUE4.exe /Game/Maps/RaceTrack -windowed -carla-server -benchmark -fps
    =30
```

5) In another terminal, change the directory to go into the "Course1FinalProject" folder, under the "PythonClient" folder.

Run your controller, execute the following command while CARLA is open:

Ubuntu (use alternative python commands if the command below does not work, as described in the CARLA install guide)**:**

```
python3 module_7.py
```

Windows (use alternative python commands if the command below does not work, as described in the CARLA install guide)**:**

```
python module_7.py
```

The simulator will begin to run if the module_7 client connects to the server properly. It will open two new feedback windows (unless live_plotting is disabled - see the editing of **options.cfg** below for more details), one of which shows the trajectory and the other which shows the controls feedback.

The trajectory feedback will contain the car, start and end positions, entire path/path traveled and a small shaded region which denotes the subset of interpolated points to be sent into the controller for control updates. Linear interpolation is used between waypoints to provide a finer resolution path/speed requests for the controller. The X and Y axes are in meters.

The controls feedback shows the throttle, steering and brake outputs, as well as the speed response for the simulation (desired speed and current speed in the single plot). This is a general feedback for viewing what the client is sending to the CARLA server in terms of control commands. The desired speed is set to the closest interpolated speed point to the current position of the car. The speeds are in meters per second and the throttle (0 to 1), brake (0 to 1) and steering (-1 to 1, or left to right turn) are unitless. Note that the steering command output inside controller2d.py is automatically converted from radians (-1.22 to 1.22 rad) to a percentage (-1 to 1) before the command is sent to the CARLA server. The X axis for all four plots in the controls feedback is the in-game time, in seconds.

If the simulation runs slowly, you can try increasing the period at which the live plotter refreshes the plots, or disabling the live plotting altogether. Disabling the live plotting does not affect the plot outputs at the end of the simulation.

To do this, edit the **options.cfg** file found in the "Course1FinalProject" folder for the relevant parameters. The following table below explains each option:

Parameter	Description	Value
live_plotting	Enable or disable live plotting.	true/false
live_plotting_period	Period (in seconds) which the live plot will refresh on screen. Set to "0" for an update every simulation timestep.	[seconds]

6) The client will close once you reach the final waypoint, or after around 200 to 250 in-game seconds have passed. After the simulation completes, a text file containing the trajectory generated by your controller is saved. This file is called "trajectory.txt" and it is located inside the "controller_output" folder under the "Course1FinalProject" folder. The plots for speed, throttle, brake, steering and the executed 2D trajectory are also saved into this folder.

A grading script (provided below) compares your trajectory to the waypoints and scores its performance. Extract this grader into the "Course1FinalProject" folder.

grade_c1m7.py

To run the grading script, execute the following command from the "Course1FinalProject" directory in a new terminal:

Ubuntu (use alternative python commands if the command below does not work, as described in the CARLA install guide)**:**

```
python3 grade_c1m7.py racetrack_waypoints.txt /controller_output/trajectory
    .txt
```

Windows (use alternative python commands if the command below does not work, as described in the CARLA install guide)**:**

```
python grade_c1m7.py racetrack_waypoints.txt /controller_output/trajectory
    .txt
```

The grading script plots your trajectory along with the velocity at each waypoint. The distance and velocity error bounds are also shown on the plots. Your trajectory passes the assignment if it successfully reaches 50% or more waypoints.

7) Once your trajectory passes the grading script, you may upload its file (trajectory.txt) to Coursera for official evaluation.

8) Finally, for an even greater challenge, try modifying the reference speeds or path to see how quickly can you complete the track. No points or grading for this exercise, but feel free to post your results on the forums!

To modify the reference path and speed, edit the "racetrack_waypoints.txt" file. This file will contain the path and reference speed for each waypoint in the path.

The rows of the txt file is each waypoint in sequence, and the columns are in the following format: "X position (meters), Y position (meters), reference speed (meters per second". You may modify the reference speed by changing the third column of the file, and the path by modifying the path by changing the first and second columns.