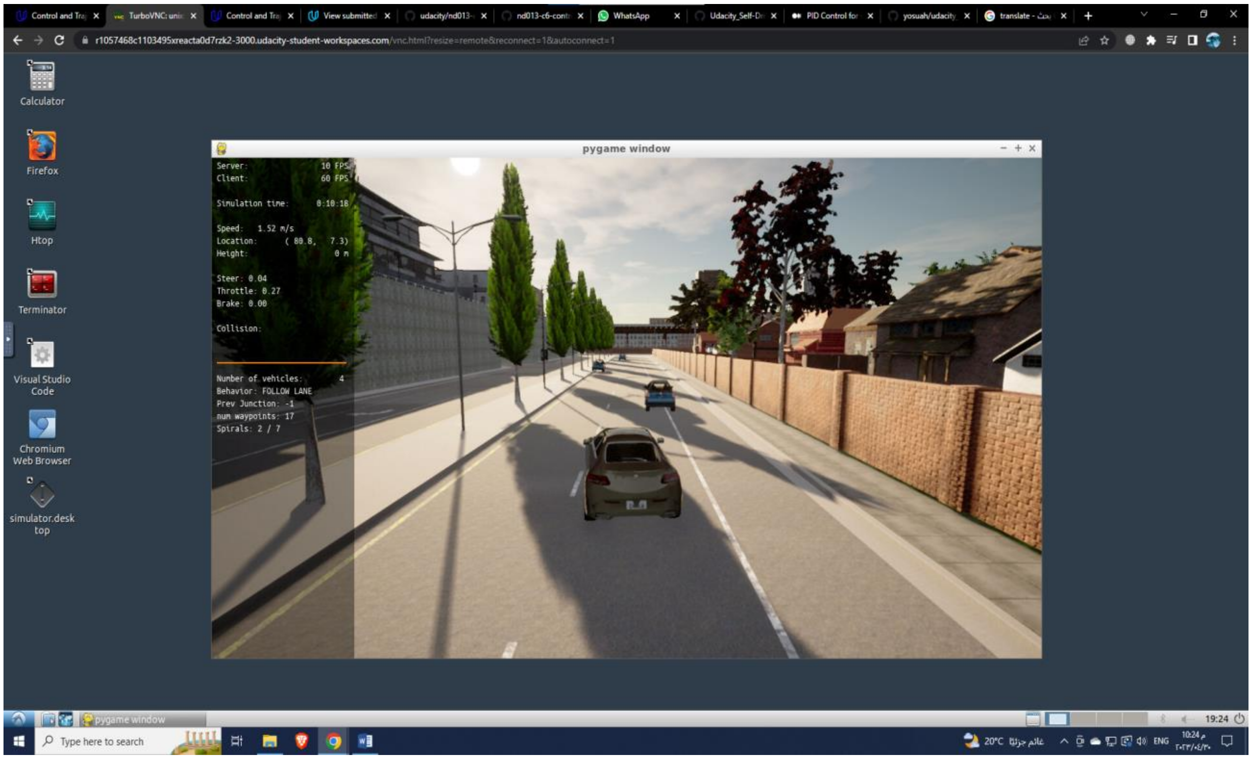
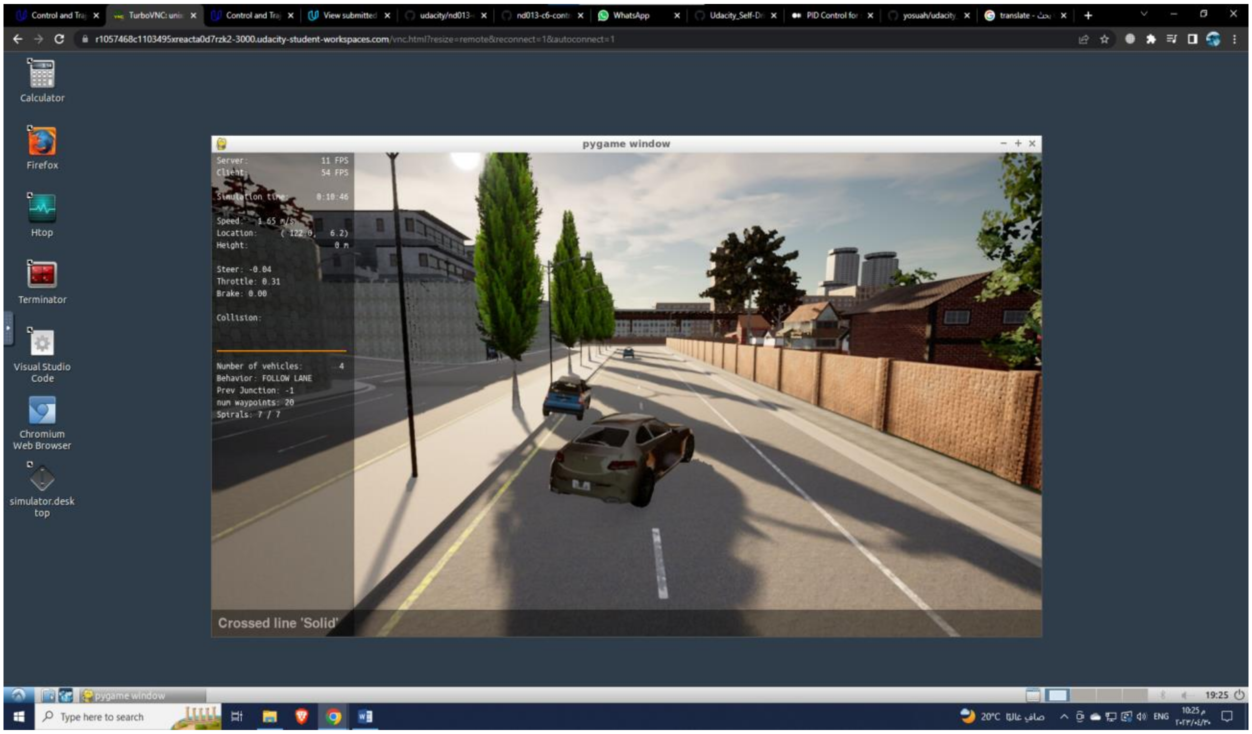
Control and Trajectory Tracking for Autonomous Vehicles

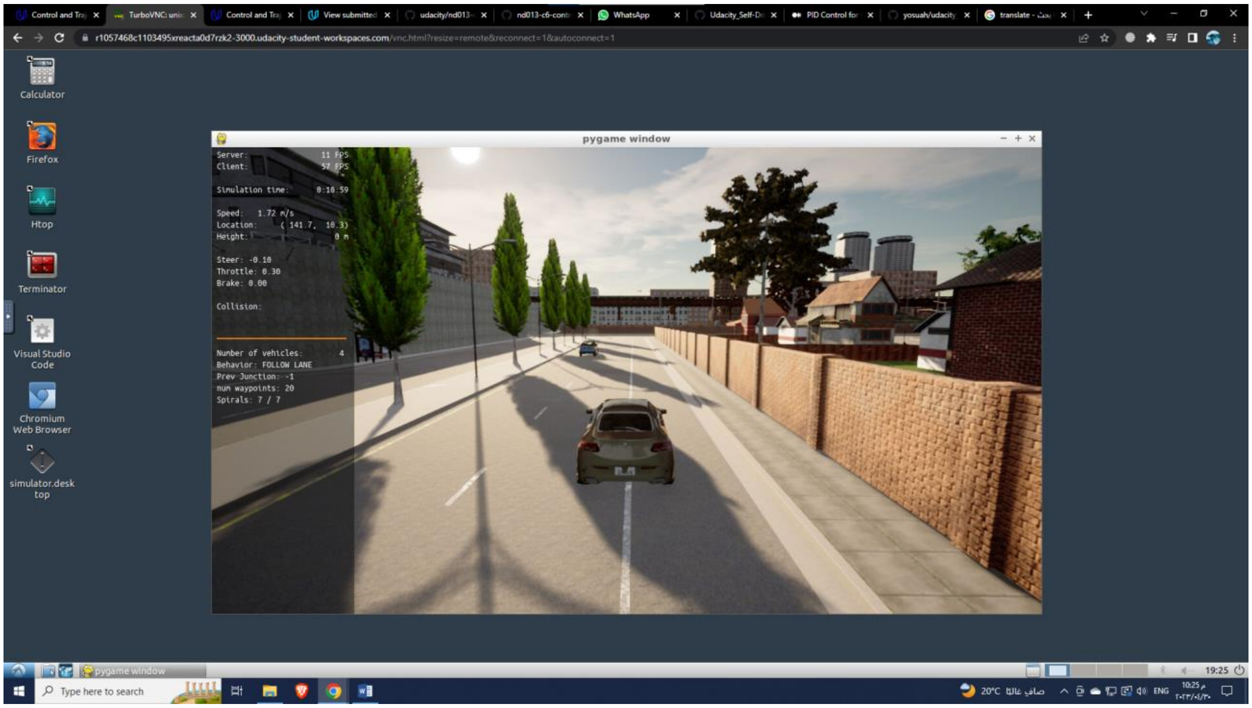


```
Server:                 10 FPS
Client:                 60 FPS

Simulation time:       0:10:18

Speed:   1.52 m/s
Location:       ( 80.8,   7.3)
Height:                    0 m

Steer: 0.04
Throttle: 0.27
Brake: 0.00

Collision:

Number of vehicles:         4
Behavior: FOLLOW LANE
Prev Junction: -1
num waypoints: 17
Spirals: 2 / 7
```

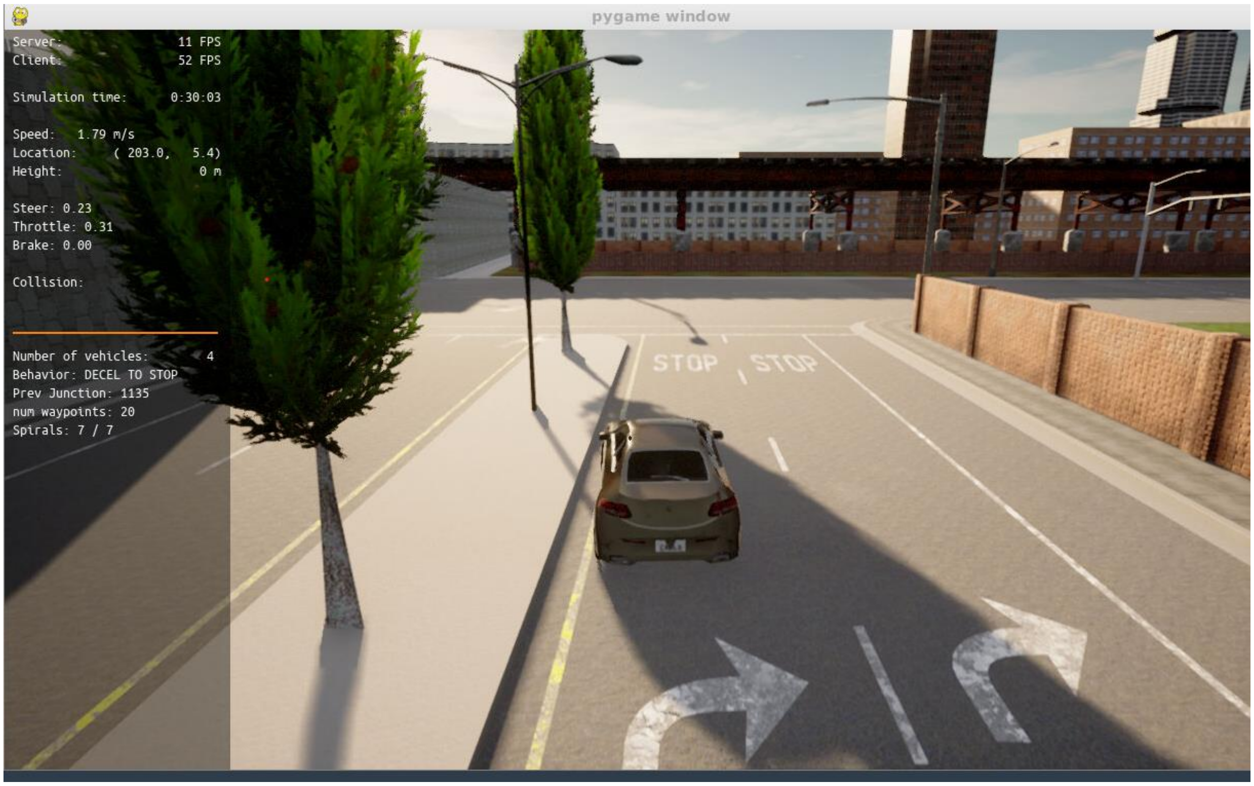


```
Server:                 11 FPS
Client:                 54 FPS

Simulation time:       0:10:46

Speed:   1.65 m/s
Location:       ( 122.0,   6.2)
Height:                    0 m

Steer: -0.04
Throttle: 0.31
Brake: 0.00

Collision:

Number of vehicles:         4
Behavior: FOLLOW LANE
Prev Junction: -1
num waypoints: 20
Spirals: 7 / 7
```

Crossed line 'Solid'

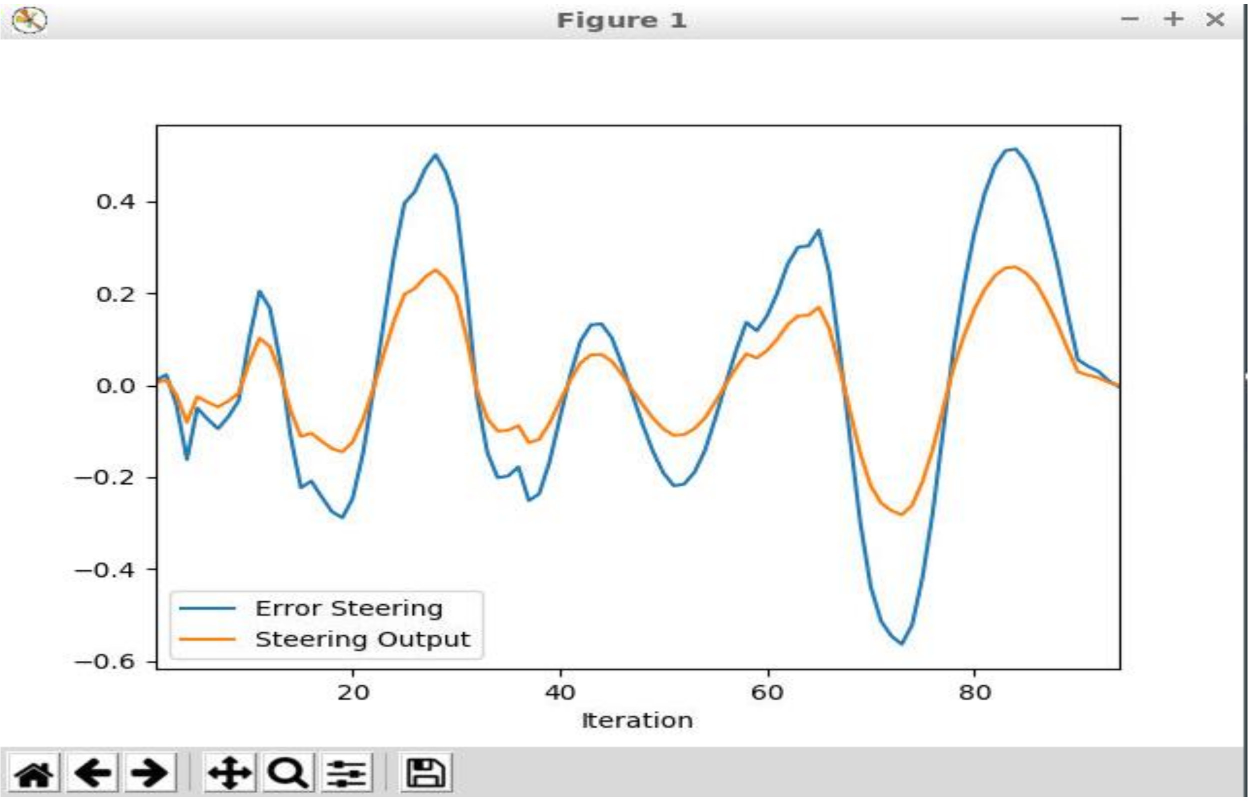


```
Server:              11 FPS
Client:              57 FPS

Simulation time:    0:10:59

Speed:   1.72 m/s
Location:      ( 141.7,  10.3)
Height:                  0 m

Steer: -0.10
Throttle: 0.30
Brake: 0.00

Collision:

Number of vehicles:       4
Behavior: FOLLOW LANE
Prev Junction: -1
num waypoints: 20
Spirals: 7 / 7
```

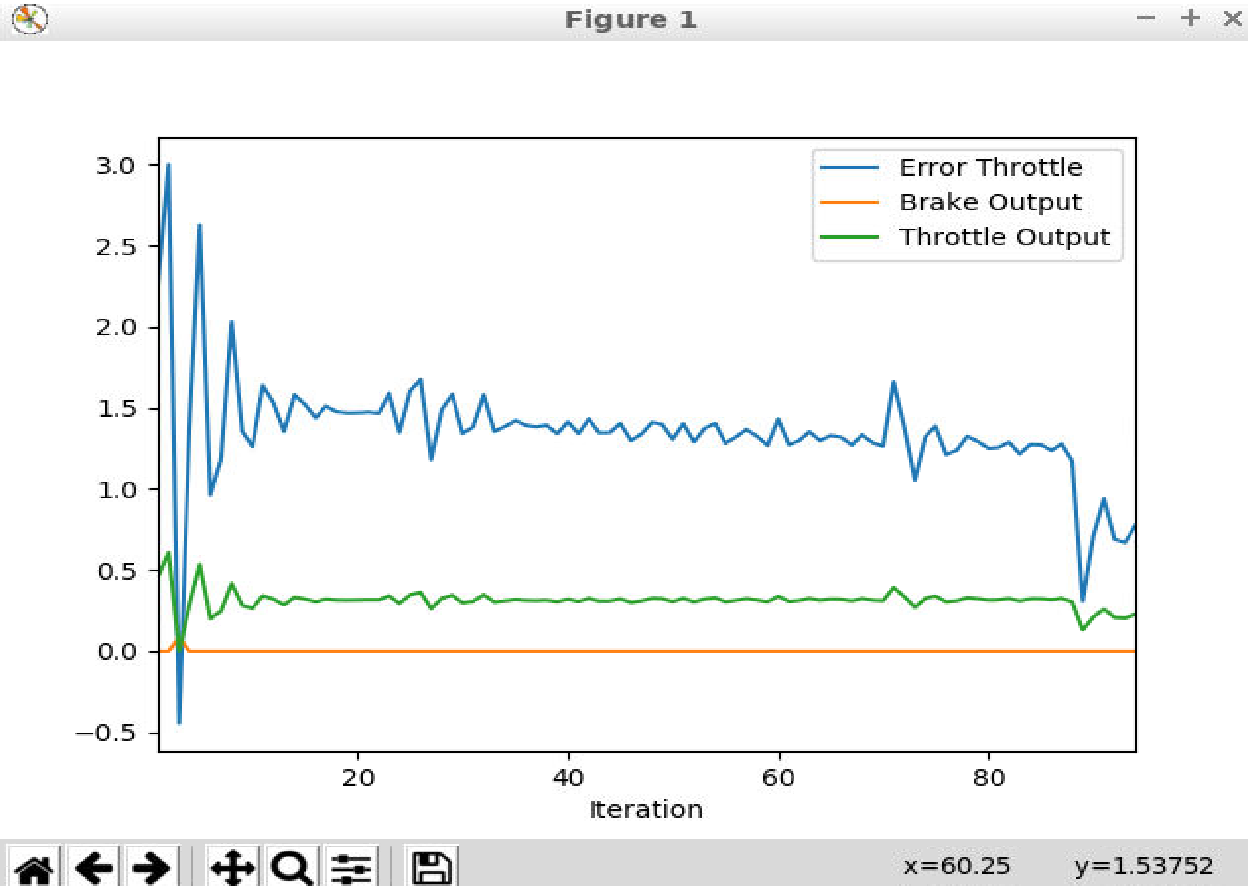


```
Server:              11 FPS
Client:              52 FPS

Simulation time:    0:30:03

Speed:   1.79 m/s
Location:      ( 203.0,   5.4)
Height:                  0 m

Steer: 0.23
Throttle: 0.31
Brake: 0.00

Collision:

Number of vehicles:       4
Behavior: DECEL TO STOP
Prev Junction: 1135
num waypoints: 20
Spirals: 7 / 7
```

1. Add the plots to your report and explain them (describe what you see)



In case of an obstacle-filled scene, the final figure displays the steering controller error and output. There are three points in the timeline where the inaccuracy grew significantly; these three instances occur while the car was passing objects and had to change lanes. The error curve shows that oscillation also affects the steering controller, hence further adjusting the derivative value is probably required. The root cause of this oscillation, for instance, may be that the controller plans too far in advance (gets the last waypoint), but the update is called more frequently, making the error always seem larger than it is. This oscillation can also be caused by delay between the controller and the simulation, or by other mismatches.



The following plots show the error calculated and he control output for both controllers. The x axis shows iterations, which are 20ms each.

This illustration depicts the throttle controller with a clear, obstacle-free background and a constant reference speed of 3 m/s. Since the controller is damped, it is clear that it takes a while to reach the desired speed, but oscillation is still clearly discernible. The differential term in the controller lowers but does not eliminate oscillation.

2. *What is the effect of the PID according to the plots, how each part of the PID affects the control command?*

- Proportional gain P: This gain directly affects the output since it is multiplied with error term so the larger the P gain, the more the output changes similar to the error.
- Integral gain I: This gain is trying to decrease the accumulated error of the entire driving, which means that this gain has an ability to control the error caused by systematic error.
- Deferential gain D: This gain is applied to the deferential term of error. Accordingly, it can make the actual trajectory of the ego vehicle smoother and avoid overshooting.

A regular PID controller is implemented and used for both throttle and steering control.

Given the distance, the needed acceleration from the present velocity to the trajectory's end velocity is computed for throttle control. Ideally, in my opinion, we should control from the present state to the next one rather than the previous one, however owing to internal inconsistencies, this strategy appeared to be more effective. I also attempted to control only solely on the needed velocity change, but in that situation I was unable to eliminate oscillation. However, utilizing the required acceleration as the input provided a reasonably smooth control signal.

The distance between the most recent waypoint and the current direction of travel is estimated for steering control. It is a very crude approximation to determine the direction of travel as a straight line based on the present yaw without accounting for the direction of the front wheel. Although it can be seen that this produces very large error values even for moderate steering, it works in the current simulation.

3. *How would you design a way to automatically tune the PID parameters?*

I learned about "Twiddle" during the presentation, which iteratively repeats the control algorithm to discover the optimal PID parameters that produce the least amount of error beginning from initial values and may be used to automatically tweak the PID parameters.

4. *PID controller is a model free controller, i.e. it does not use a model of the car. Could you explain the pros and cons of this type of controller?*

- One benefit of the PID controller, which is a model-free controller, is that it is simple to build and compute the error and the output.
- However, in practice, the dynamics of the vehicle must be taken into account because it may occasionally be impossible to drive while adhering to controller output.

5. (Optional) What would you do to improve the PID controller?

The 'Twiddle' programmer may be used to enhance the PID controller and find the optimal parameters for driving.