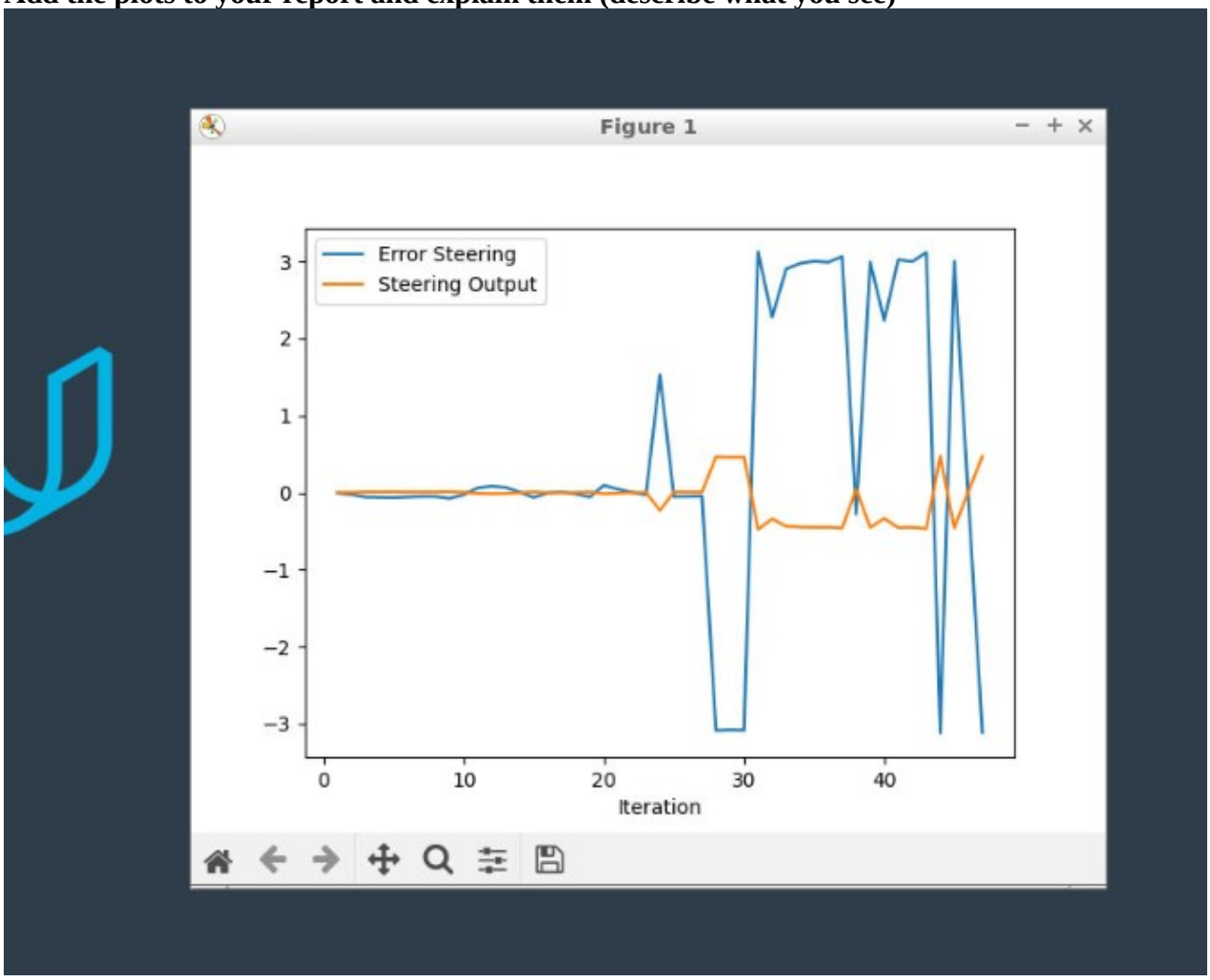
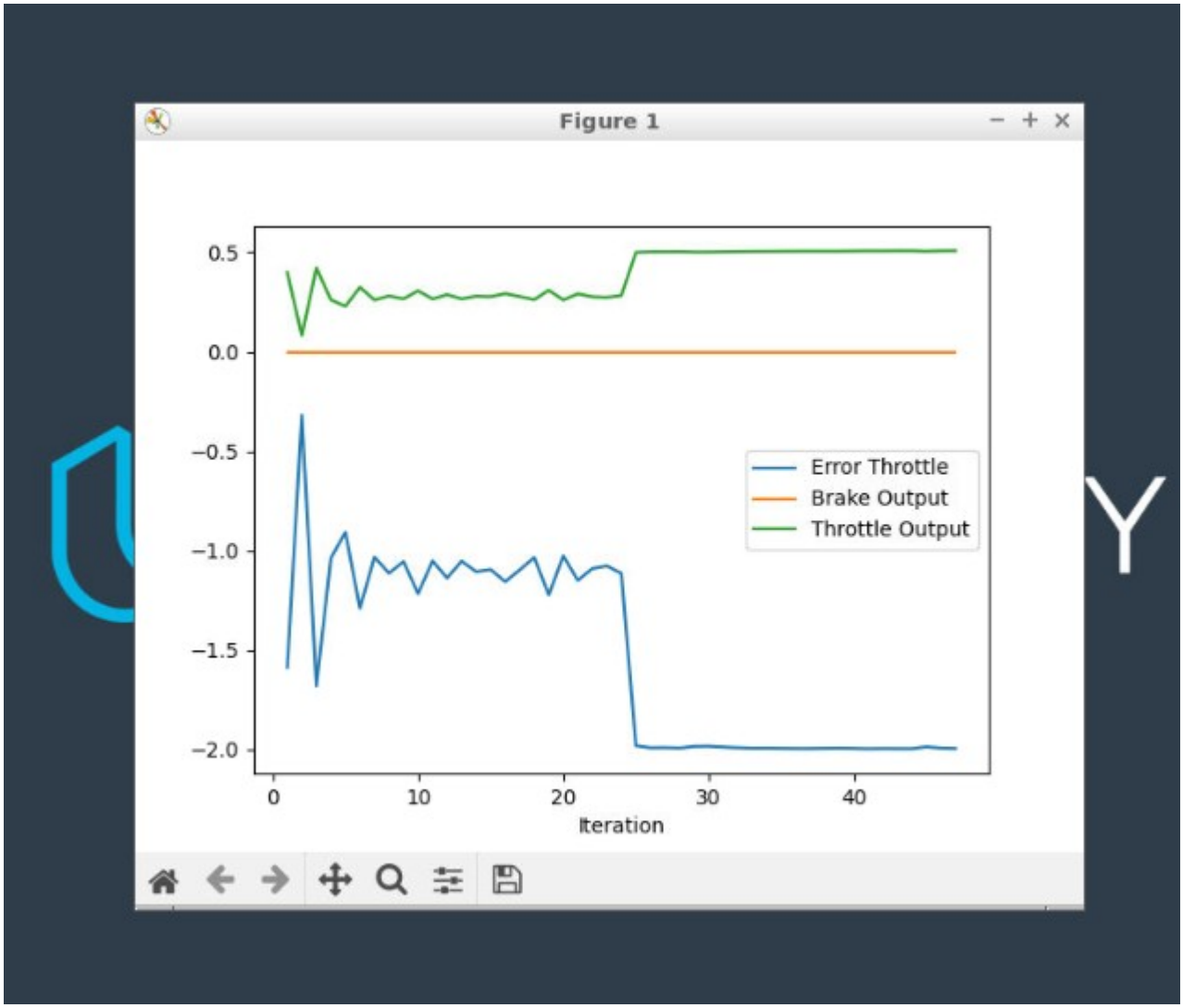
// Answer the following questions:

// Add the plots to your report and explain them (describe what you see)
// What is the effect of the PID according to the plots, how each part of the PID affects the control command?
// How would you design a way to automatically tune the PID parameters?
// 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?
// (Optional) What would you do to improve the PID controller?

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



Initially, it seems balance in steering angle between actual steer and desired one. However, after 20 iteration error for steering is high ,that is my desire steering and actual steering are not approximate phase of oscillation(blue line has lots of high peak on y axis over iteration).



Here, throttle error initial shows good at converging towards reference and brake output but after 20 iteration ,goes parallel to reference.

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

With the selected Kp, Ki, Kd values for steering and speed control, i was able to guide little ahead the car through the track without the car leaving the track but later making collide with car which need more attempt to tune PID parameter. The reason i used manual tuning is to understand the inside working of PID controller and how the values influence the vehicle behavior.

The Following are my conclusions :

Kp value had significant influence on vehicle behaviour since it was related directly to the error. Very high value of Kp will make the car swerve wildly at curves. Lower Kp values lead to smoother steering corrections. However ,in my case, low value of these parameter doesnot show any good results.

Kd value is required to dampen these oscillations. A wide range of Kd values are suitable since its influence is dependent on error gradient and not the error itself. Whenever the vehicle navigates a sharp turn, the error increases but in my case ,there is flat line arise after certain iteration. A combination of good Kp and Kd values reduces the error and dampen oscillations for gaining control back. If steady state error is present, then Ki value is required.

How Would You Design A Way To Automatically Tune The Pid Parameters?

The main design effort is to carefully tune these three coefficients to get the best possible performance i.e proportional, integral and differential control in terms of Kp, KI and Kd coefficients respectively and the main goal of the controller is to minimize the aggregated CTE.

Tuning these values manually is quite challenging. Some values selected performed well in certain areas of the track but failed in other areas. Automated tuning algorithms like TWIDDLE,ZIEGLER–NICHOLS which tunes by varying all the three hyperparameters in unison would be a good alternative.

reference
link[https://www.researchgate.net/publication/256423413_Model_based_Tuning_of_PID_Controller].

Besides these algorithm, there is also simulink control design section integrated in MATLAB software which can readily design PID control system.
Model free controller examles videos link
[https://ch.mathworks.com/support/search.html/videos/how-to-automatically-tune-pid-controllers-1600850427273.html?fq[]=asset_type_name:video&fq[]=category:simulink/index&page=1] .

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?

pros:
1.Its feasibility and easy to be implemented i.e the P Controller stabilizes the gain but produces a constant steady-state error,the I Controller reduces or eliminates the steady-state error,the D Controller reduces the rate of change of error.

cons:
1.The only disadvantage is the tuning methodology.

reference link
[https://www.researchgate.net/post/Can_you_explain_the_advantages_and_disadvantages_of_PID_control_in_PLCs].
[https://hal.inria.fr/inria-00563941/document]

Find at least 2 pros and cons for model free versus model based.

Model free controller:

pros:

1.can be use it for any kind of control tasks that require PID controller such as DC motor speed, Oven temperature (recently added)

2.it reduces the time constant of the system and thus making faster

cons:

1.poor implementation and/or tuning and not the algorithm itself i.eit is a feedback algorithm so cannot take any control action until it sees a disturbance and an error is produced.

2.Low performance to handle strong non-linearities,difficulty to handle multiple variables with strong interaction;

Model based controller:

pros:

1.Model-based design provides a common design environment, which facilitates general communication, data analysis, and system verification between various (development) groups.

2.Engineers can locate and correct errors early in system design, when the time and financial impact of system modification are minimized.

cons:

1.One major disadvantage is that the approach taken is a blanket or coverall approach to standard embedded and systems development. Often the time it takes to port between processors and ecosystems can outweigh the temporal value it offers in the simpler lab based implementations.

2.Much of the compilation tool chain is closed source, and prone to fence post errors, and other such common compilation errors that are easily corrected in traditional systems engineering.