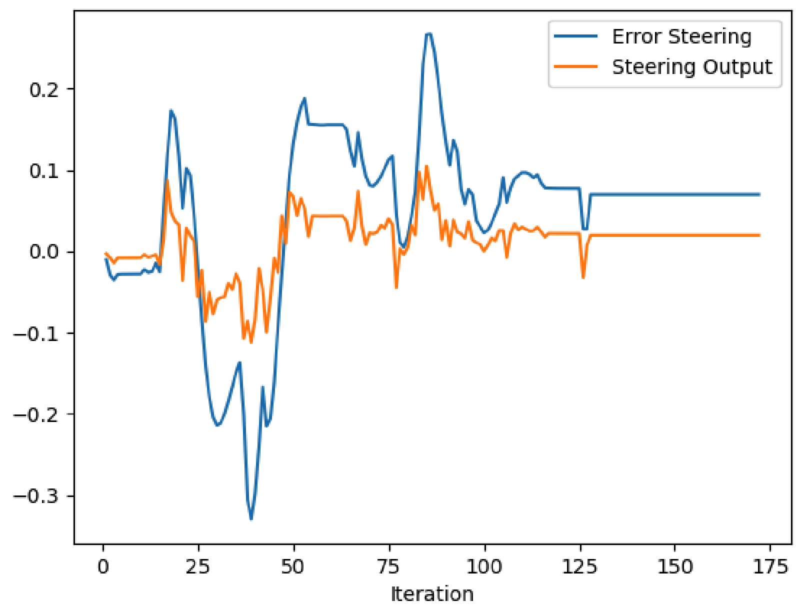
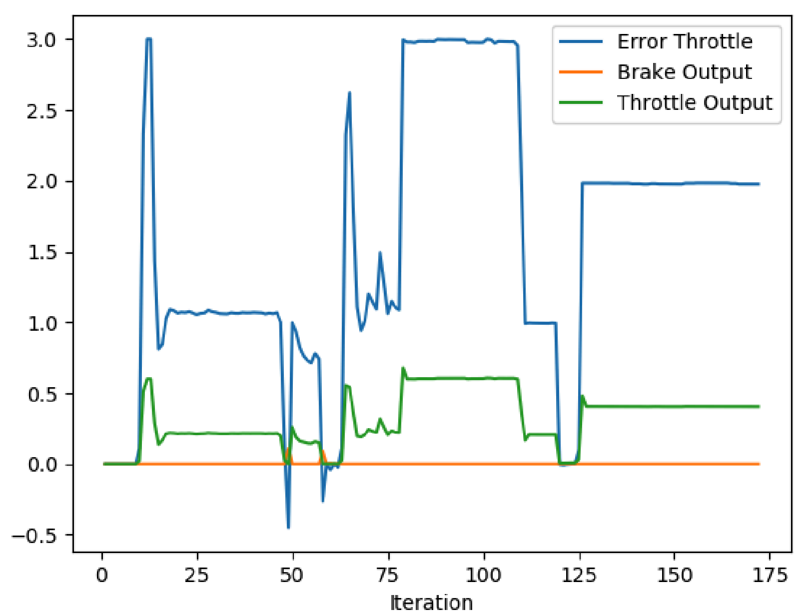
Report

1. Plot the results of the simulation.

- With the below parameter, the result plot is as follow:
    - pid_steer
        - k_p = 0.28, k_i = 0.0001, k_d = 0.8, max = 1.2, min = -1.2
    - pid_throttle
        - k_p = 0.2, k_i = 0.0001, k_d = 0.04, max = 1.0, min = -1.0





2. Analyzing the plot.

- First Plot: Steering
  - Variable
    - X Axis: Iteration
    - Y Axis: Error Steering, Steering Output
  - Phenomenon
    - The error oscillates around 0.0.
    - The absolute value of the raw error is consistently larger than the output, which is obtained by multiplying the error with the coefficients K_p, K_i, K_d.
    - The output is limited to a certain value, resulting in a range smaller than that of the raw error.
- Second Plot: Throttle
  - Variable
    - X Axis: Iteration
    - Y Axis: Error Throttle, Brake Output, Throttle Output
  - Phenomenon
    - The absolute value of the raw error is consistently larger than the output, which is obtained by multiplying the error with the coefficients K_p, K_i, K_d.
    - The output is limited to a certain value, resulting in a range smaller than that of the raw error.

3. Recognize the action of each part of the PID controller.

- What is the effect of the PID according to the plots, how each part of the PID affects the control command?
- Proportional (P):
  - The P component is directly proportional to the Cross Track Error (CTE). It adjusts the steering angle based on a factor multiplied by the CTE. However, this adjustment can lead to overshooting, achieving only a marginally stable status. Additionally, higher factors result in larger oscillations.
- Derivative (D):
  - The D component aims to reduce oscillations by considering the differential of the current and previous CTE values. The larger the difference between these values, the greater the adjustment in the steering angle.

- Integral (I):
    - The I component addresses systematic bias by incorporating all errors encountered thus far. It adjusts the steering angle based on the cumulative error. Consequently, a larger cumulative error leads to a greater adjustment in the steering angle, aiding in the correction of systematic biases.

4. Elaborate a critical analysis of the controller.

- How would you design a way to automatically tune the PID parameters?
    - Twiddle algorithm, which we introduced in class.By iteratively exploring the parameter space and evaluating the controller's response, Twiddle aims to converge towards an optimal set of parameters that minimize error and enhance performance.
    - Reference Code from Class Exercise:

```
def twiddle(tol=0.2):
    # Don't forget to call `make_robot` before every call of `run`!
    p = [0, 0, 0]
    dp = [1, 1, 1]
    robot = make_robot()
    x_trajectory, y_trajectory, best_err = run(robot, p)

    while sum(dp) > tol:
        for i in range(3):

            print ("Twiddle #", i, p, best_err)

            # Move up dp[i], if error becomes smaller, make dp[i] larger, keep p
            p[i] += dp[i]

            curr_robot = make_robot()
            _, _, curr_err = run(curr_robot, p)

            if curr_err < best_err:
                best_err = curr_err
                dp[i] *= 1.1
            else:

                # Move down dp[i], if error becomes smaller, make dp[i] larger, keep p
                p[i] -= 2 * dp[i]

                curr_robot = make_robot()
                _, _, curr_err = run(curr_robot, p)
                if curr_err < best_err:
                    best_err = curr_err
                    dp[i] *= 1.1

                # If error does not become smaller, make dp[i] smaller, get back original p
                else:
                    p[i] += dp[i]
                    dp[i] *= 0.9


    return p, best_err
```

- 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? Find at least 2 pros and cons for model free versus model based.
    - Pros
        - Universal Applicability: Model-free controllers like PID can be deployed across various vehicles without the need for vehicle-specific models.
        - Accessibility: More individuals can contribute to the development and implementation of PID controllers. This accessibility broadens the pool of potential contributors and fosters innovation in control system design.
        -
    - Cons
        - Limited Optimization: Since model-free controllers like PID do not rely on specific vehicle models, they may not be optimized for the unique hardware or specifications of a particular car.
        - Generic Representation: Model-free controllers often necessitate generic representations of vehicle dynamics. For instance, using simple geometric shapes like circles to represent vehicles may not accurately capture the non-standard vehicles, such as long trucks or vehicles with unconventional designs.