Nathan Donaldson/U0632803

ECE5780/Lab 8 – Motor Control Implementation

1. What gain parameters did you end up using for your PI controller?
    - I ended up using Ki = 6 and Kp = 7
    - Error increases as it tries to fight resistance, that is because Ki and Kp are fighting to get back at the desired speed and they end up overshooting the desired values to get back. After resistance has been fought the error starts to level out.
2. Describe what the derivative control parameter does.
    - It detects the rate that error develops and then provides corrective action in response.
    - It helps reduce overshoots or oscillations.
    - If there tends to fluctuations in your system, like with flowrates, then the small but rabid fluctuations can be magnified by the derivative control parameter and lead to further oscillations in your signal.
3. List four basic issues/problems that are often encountered in a basic PID algorithm.
    - It is designed to be called irregularly, which causes two issues
        1. You don't get consistent behavior from the PID, since it is called irregularly.
        2. You need to do extra math computing the derivative and integral, since they're both dependent on the change in time.

      The way to fix these two issues is to ensure that the PID is called at a regular interval. You can do this by computing every cycle.
    - Derivative kick is another issue, this is caused by the setpoint being changed and causing an instantaneous change in error, causing spikes in the output. To fix this, instead of adding (Kd * derivative of Error), we subtract (Kd * derivative of input). This is known as using "Derivative on Measurement".
    - Another problem is the ability to change tuning parameters while the system is running. The PID acts crazy if you try to change the tunings while it's running. One way of fixing this is to rescale errSum. If Ki is doubled, curt errSum in half. That keeps the output from bumping and it works.