EECS 461, Winter 2023, Problem Set 8: SOLUTIONS[1]

1. Consider the problem of PD control for a double integrator, as depicted in Figure 1, where

$$K_{PD}(s) = K_P + K_D s. \tag{1}$$

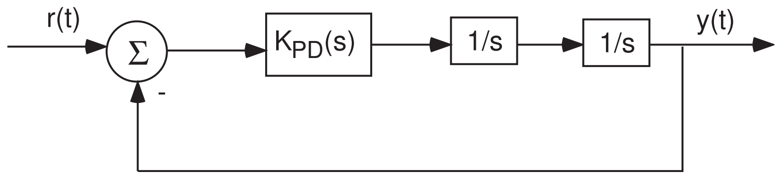

Figure 1: PD control of a double integrator.

We saw in class that the transfer function from the command $r(t)$ to the output $y(t)$ is given by

$$\begin{aligned} Y(s) &= \frac{(1/s^2)(K_P + K_D s)}{1 + (1/s^2)(K_P + K_D s}R(s) \\ &= \frac{K_P + K_D s}{s^2 + K_D s + K_P}R(s). \end{aligned}$$

and thus that the characteristic polynomial is given by $s^2 + K_D s + K_P$. Hence, if we wish that the characteristic roots have natural frequency ω_n and damping ζ, then we can solve for K_P and K_D by equating coefficients:

$$s^2 + K_D s + K_P = s^2 + 2\zeta\omega_n s + \omega_n^2. \tag{2}$$

An alternate implementation of PD control is depicted in Figure 2, where

$$K'_D = K_D/K_P. \tag{3}$$

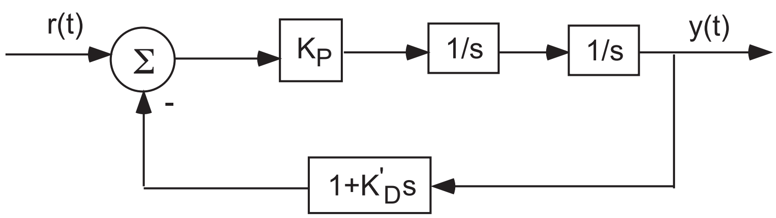

Figure 2: PD control of a double integrator, with differentiator in the feedback loop.

The transfer function from $r(t)$ to $y(t)$ is given by

$$\begin{aligned} Y(s) &= \frac{(K_P/s^2)}{1 + (K_P/s^2)(1 + K'_D s)}R(s) \\ &= \frac{K_P}{s^2 + K_P K'_D s + K_P}R(s). \end{aligned}$$

As in the original implementation of PD control, one may choose K_P and K'_D to yield desired characteristic roots.

[1] Revised April 5, 2023.

(a) Suppose we wish the characteristic roots to have $\omega_n = 1$ and $\zeta = 0.7$, corresponding to a rise time of $t_r = 1.8$ seconds, and an overshoot of 5%. What values should K_P and K_D have for the PD controller implementation in Figure 1? Plot the step response. Do the actual rise time and overshoot agree with the desired values?

(b) Repeat the previous part by choosing K_P and K_D' in the alternate implementation of Figure 2 to yield the same characteristic roots. Plot the step response. Do the actual rise time and overshoot agree with the desired values?

SOLUTION to Problem 1:

(a) We find the values of ζ and ω_n from the formulas

$$K_P = \omega_n^2$$
$$K_D = 2\zeta\omega_n,$$

so that if we want $\omega_n = 1$ and $\zeta = 0.7$, we should set $K_P = 1$ and $K_D = 1.4$.

(b) We need to set $K'_D = K_D/K_P$, and thus $K'_D = 1.4$.

The step responses, together with the controller parameters, are depicted in Figure 21. Note that only the alternate implementation achieves the desired overshoot and rise time.

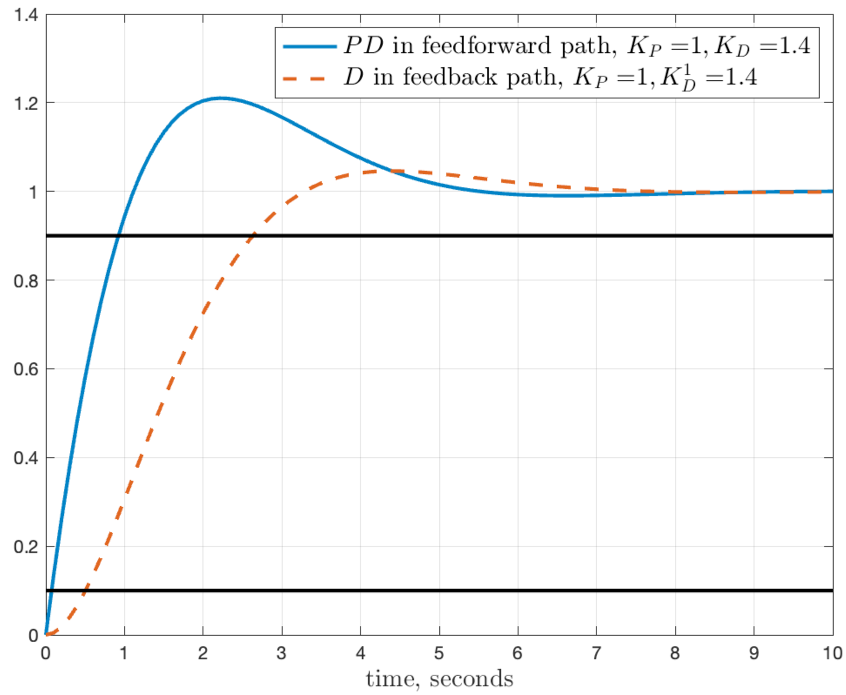

Figure 3: Comparison of step responses.

2. Now consider the use of PD to control the speed of a DC motor, as depicted in Figure 4. The transfer

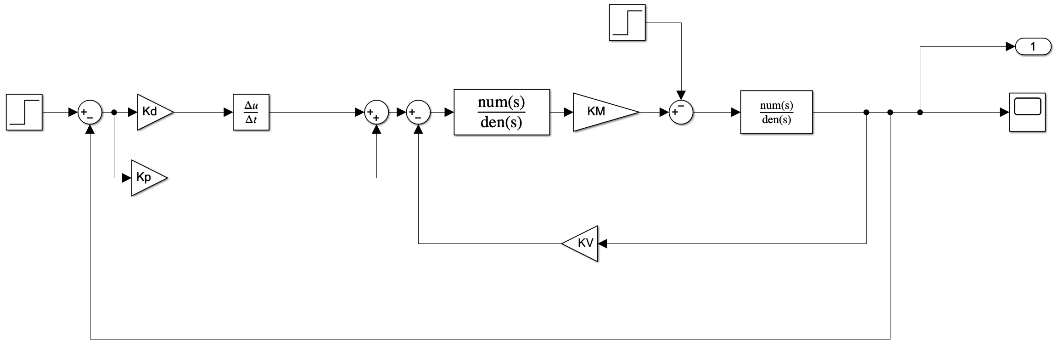

Figure 4: PD speed control of a DC motor.

function from input voltage to shaft speed (with load torque $T_L = 0$) is given by $\Omega(s) = G(s)V(s)$, where

$$\begin{aligned} G(s) &= \frac{\frac{K_M}{(sL+R)(sJ+B)}}{1 + \frac{K_M K_V}{(sL+R)(sJ+B)}} \\ &= \frac{K_M}{LJs^2 + (LB + RJ)s + K_M K_V + RB} \end{aligned}$$

Denote the transfer function of the PD controller by

$$K_{PD}(s) = K_P + K_D s.$$

With PD feedback control, the transfer function from input voltage to shaft speed is given by $\Omega(s) = T(s)V(s)$, where

$$T(s) = \frac{K_{PD}(s)G(s)}{1 + K_{PID}(s)G(s)} \tag{4}$$

$$= \frac{K_M(K_D s + K_P)}{LJs^2 + (LB + RJ + K_D K_M)s + (K_M K_V + K_P K_M + RB)} \tag{5}$$

(a) As in problem 1, choose $\omega_n = 10$, $\zeta = 0.7$ and equate coefficients of the characteristic equation to find K_P and K_D. Suppose that J = 0.0113 N-m/(rad/msec^2), B = 0.028 N-m/(rad/msec), L = 0.1 H, R=0.45 ohms, $K_M = 0.067$ N-m/amp, $K_V = 0.067$ V/(rad/sec). What values of K_P and K_D achieve these specifications? Use `ps8_problem2a.m` and `PD_motor_speed_control.slx` to plot the speed response, Ω to a commanded speed $\Omega^* = 1$ and step disturbance torque $T_L = 0.125$.

(b) We see from equation 5 that our motor with PD control is a type 0 system, so we expect that there will be a steady state error to a step input. Use the rules for block diagram reduction to develop equations for the steady state value of Ω in response to the command and disturbance inputs and verify that the steady state values seen in part (a) are correct. Recall that the steady state value for a step input is the transfer function evaluated at $s = 0$. Further recall that to compute the transfer function from load torque to speed, simply set the command input, $\Omega^* = 0$ and solve for Ω, see problem set 3.

SOLUTION to Problem 2:

(a) Equating coefficients gives us:

$$K_D = \frac{2\zeta\omega_n LJ - LB - RJ}{K_M}$$

$$K_P = \frac{\omega_n^2 LJ - RB - K_M K_V}{K_M}$$

For the values given in the problem, $K_D = 0.1184$ and $K_P = 1.4315$. The step response is shown in Figure 5.

(b) The DC gain from Ω^* to Ω with $T_L = 0$ is $H_{\Omega\Omega^*}(0) = \frac{K_M K_P}{K_M K_P + RB + K_M K_V}$.

The DC gain from T_L to Ω with $\Omega^* = 0$ is $H_{\Omega T_L}(0) = \frac{-R}{K_M K_P + RB + K_M K_V}$.

Evaluating the DC gains yields a steady state value for speed $= 1 \times 0.8488 = 0.8488$ for the step command and $-0.125 \times 3.982 = -0.498$ for the step disturbance. Thus the steady state speed (with $T_L = 0$) is 0.8488, the steady state speed (with $\Omega^* = 0$) is $-.498$, and for the combined inputs is $0.8488 - 0.498 = 0.35$, which is consistent with the simulation.

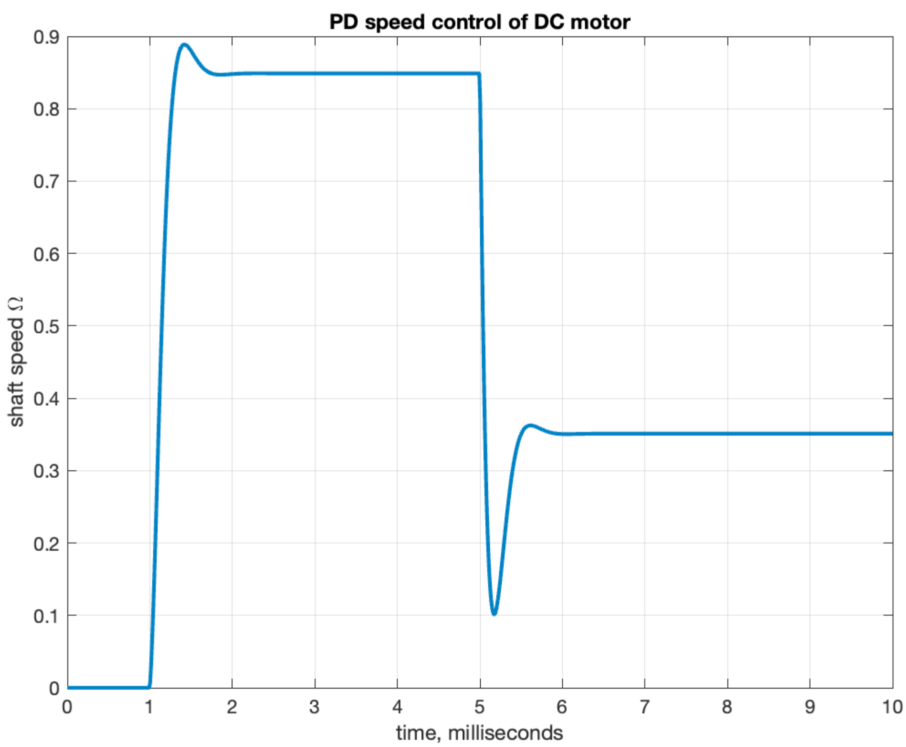

Figure 5: Step response of PD control motor.

3. Now consider the use of PID to control the speed of a DC motor, as depicted in Figure 6. The transfer

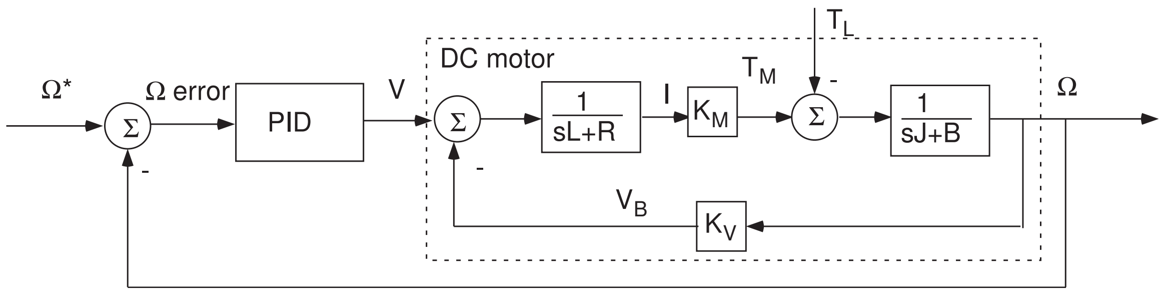

Figure 6: PID speed control of a DC motor.

function from input voltage to shaft speed is the same as in problem 2. Now denote the transfer function of the PID controller by

$$K_{PID}(s) = K_P + \frac{K_I}{s} + K_D s.$$

With PID feedback control, the transfer function from input voltage to shaft speed is given by $\Omega(s) = T(s)V(s)$, where

$$\begin{aligned} T(s) &= \frac{K_{PID}(s)G(s)}{1 + K_{PID}(s)G(s)} \\ &= \frac{K_M(K_D s^2 + K_P s + K_I)}{LJs^3 + (LB + RJ + K_D K_M)s^2 + (K_M K_V + K_P K_M + RB)s + K_I K_M} \end{aligned}$$

We wish that our system be predominantly second order so that we may apply the rules regarding rise time, overshoot, etc. to shape the dynamic response of the system. We know that poles far away from the imaginary axis are associated with fast dynamics. Thus we wish the feedback system to have two characteristic roots with natural frequency ω_n and damping ζ, and a third that is real and much faster. Then the desired characteristic equation has the form

$$(s+p)(s^2 + 2\zeta\omega_n s + \omega_n^2) = s^3 + (2\zeta\omega_n + p)s^2 + (\omega_n^2 + 2\zeta\omega_n p)s + p\omega_n^2$$

Equating the characteristic polynomial of the DC motor with speed control to the desired characteristic polynomial yields

$$\begin{aligned} & s^3 + (2\zeta\omega_n + p)s^2 + (\omega_n^2 + 2\zeta\omega_n p)s + p\omega_n^2 = \\ & \qquad s^3 + s^2\left(\frac{LB + RJ + K_D K_M}{LJ}\right) + s\left(\frac{K_M K_V + K_P K_M + RB}{LJ}\right) + \frac{K_I K_M}{LJ} \end{aligned} \tag{6}$$

Again suppose we are given the values of the motor parameters B, R, L, J, K_M, and K_V, and the desired values of the characteristic root parameters ζ, ω_n, and p. Then it follows from (6) that we may solve for PID parameters K_P, K_I, and K_D so that the feedback system has the desired characteristic roots.

(a) Assume the parameters of the system are the same as those given in problem 2, that is, J = 0.0113 N-m/(rad/msec^2), B = 0.028 N-m/(rad/msec), L = 0.1 H, R=0.45 ohms, K_M = 0.067 N-m/amp, K_V = 0.067 V/(rad/sec). Suppose also that we want the feedback system to have one real characteristic root at -100, and complex roots with $\omega_n = 10$ and $\zeta = 0.7$. What values of K_P, K_I, and K_D achieve these specifications? Verify that the resulting feedback system has characteristic roots in the desired locations. Simulate the response to a step command and a step disturbance, $T_L = 0.3$. Use `ps8_problem3a.m`. Replace PD control with PID in `PD_motor_speed_control.slx` and rename the file `PID_motor_speed_control.slx`. You can use either the Matlab `roots` or `pole` command to find the characteristic roots of the system.

(b) Approximate the differentiator by $s/(0.0001s+1)$. Note that the resulting system is 4th order instead of 3rd order. Where are the characteristic roots? Does the location of the dominant roots change significantly? Modify `PID_motor_speed_control.slx` and simulate the response to a step command and disturbance.

(c) Comparing the PD control of problem 2 and the PID of problem 3, you should see that PID control eliminates the steady state error to a step input (the system is type 1, as can be seen from the characteristic equation). If, however, you examine the control output voltage command to the motor, you will see that the commanded voltage is very large (albeit for a very short period of time). Assume the motor has a 10 volt supply, -5 to 5 volts, and we wish to speed limit the output to 75% of its maximum. Modify `PD_motor_speed_control.slx` as shown in Figure 7, with saturation block limits -3.5 to $+3.5$ volts. Simulate the response to a step command and disturbance.

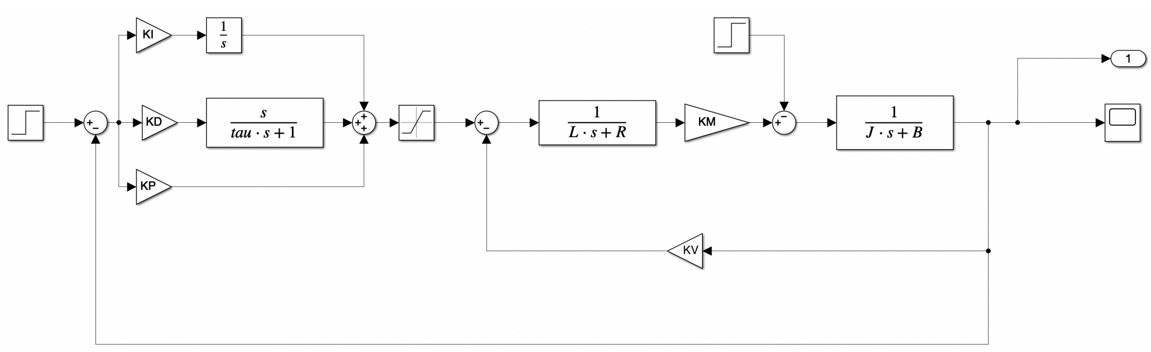

Figure 7: PID speed control of a DC motor with supply voltage limited to -3.5 to 3.5 Volts.

(d) Modify `PD_motor_speed_control.slx` to add anti-windup as shown in Figure Figure 8 and simulate the response to a step command and disturbance.

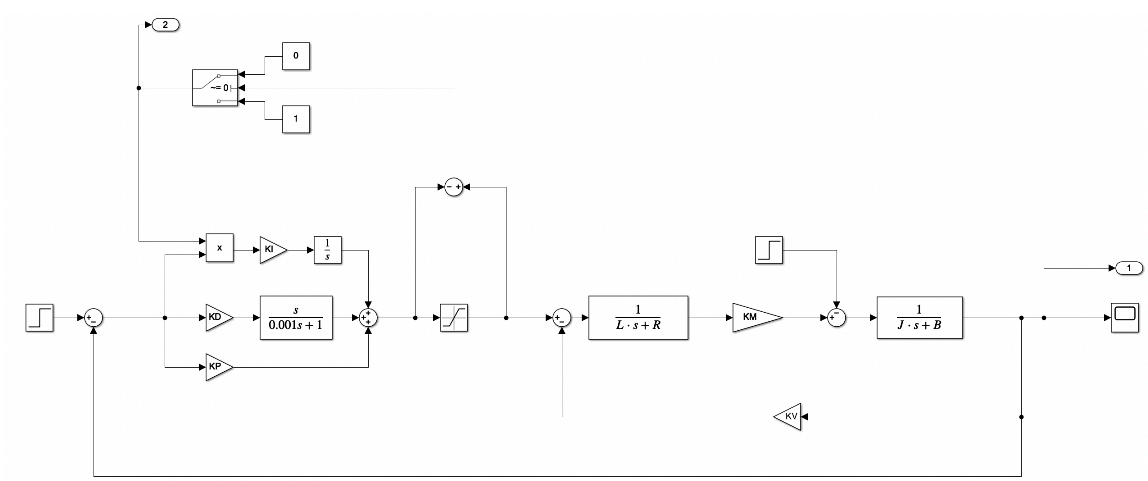

Figure 8: PID speed control of a DC motor with supply voltage limited to 0–12 Volts and reset antiwindup.

(e) Return to the original PID controller you designed in problem 3 (a). Using a sample time $T = 0.001$ seconds, design a discrete-time approximation to the analog PID controller, and simulate its response. Hand in a plot with both the analog and discrete-time responses.

SOLUTION to Problem 3:

(a) By matching coefficients, one may show that

$$K_I = (1/K_M)\omega_n^2 pLJ$$
$$K_P = (1/K_M)((\omega_n^2 + 2\zeta\omega_n p)LJ - K_M K_V - RB)$$
$$K_D = (1/K_M)((2\zeta\omega_n + p)LJ - LB - RJ).$$

This yields the numerical values $K_I = 169$ 1/msec, $K_P = 25.0$, $K_D = 1.81$ msec. The resulting step response is in Figure 9.

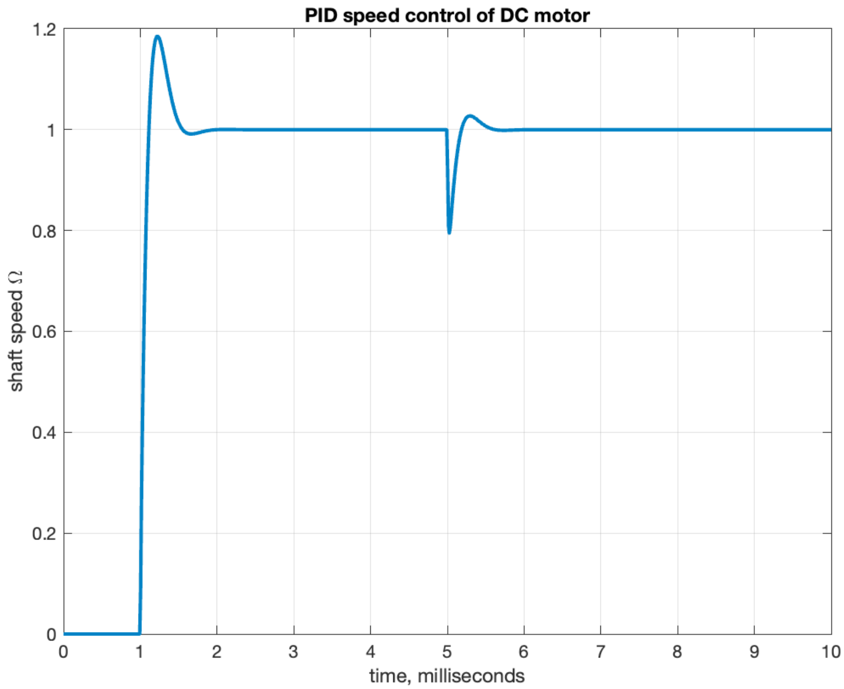

Figure 9: PID speed control for DC motor.

(b) The characteristic roots of the original third order PID motor control system are:

$$-100, \qquad -7.0 \pm 7.1j. \tag{7}$$

The characteristic roots with the approximate differentiator are given by

$$-9891, \qquad -101, \qquad -7.0 \pm 7.1j, \tag{8}$$

so that the dominant roots at $-7 \pm 7j$ do not change significantly. A plot of the step response in Figure 10 shows that the only significant change is in the initial response to the step command input, which is now smaller since we are no longer trying to differentiate a discontinuous function!

(c) The responses of the saturated and the analog PID controller without saturation are shown in Figure 11. The volltage saturation and resulting integrator windup result in greater overshoot and oscillation in response to a step speed command.

(d) Simulations with antiwindup are illustrated in Figures 12 and 13. Antiwindup reduces overshoot, but increases response time.

(e) The responses with both a discrete time PID controller and the analog PID controller with an approximate differentiator are plotted in Figure 14. Note that they agree very well, and thus that $T = 0.001$ seconds is a reasonable choice of sample time.

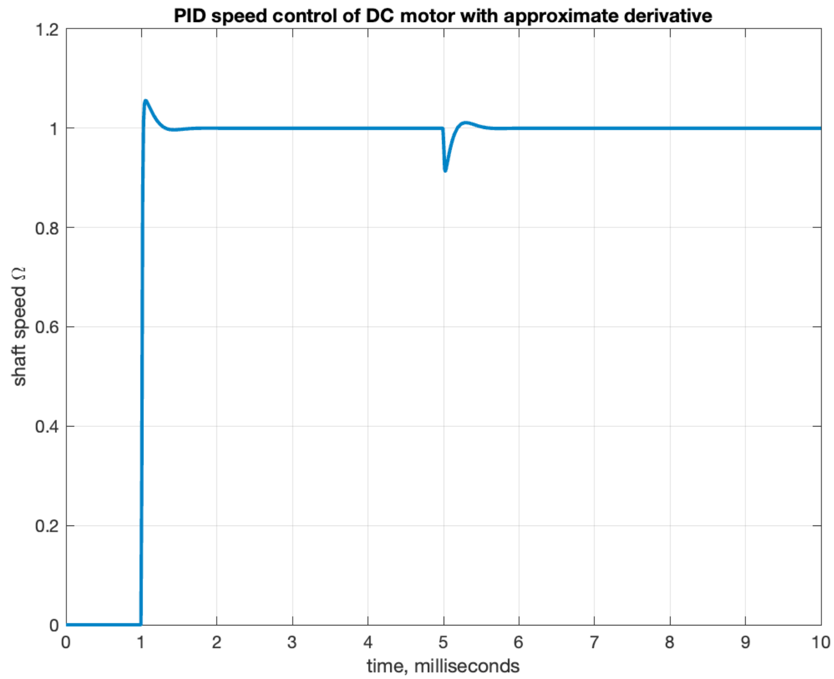

Figure 10: PID speed control for DC motor with approximate differentiator.

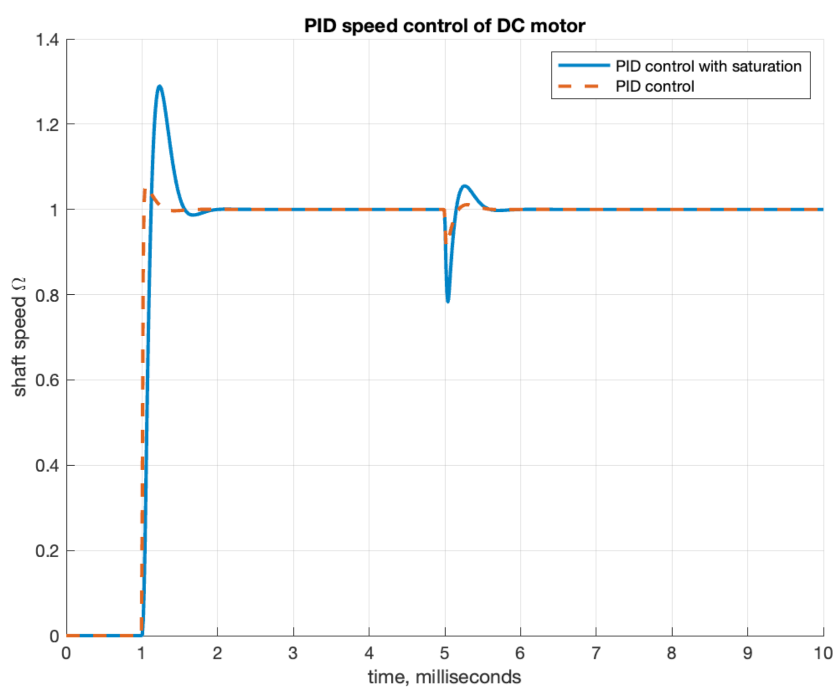

Figure 11: PID control with and without voltage saturation and integrator windup.

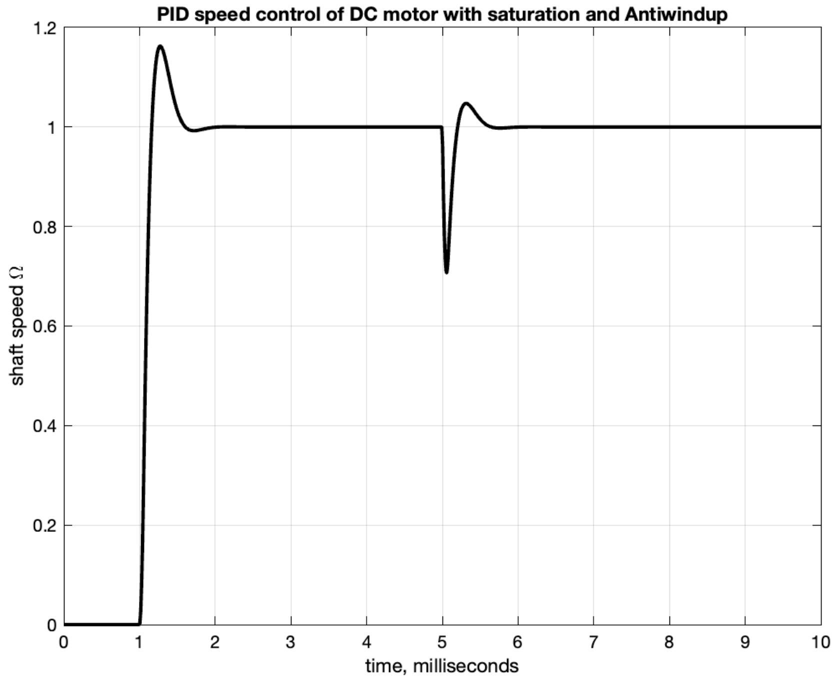

Figure 12: PID control with voltage saturation and integrator antiwindup.

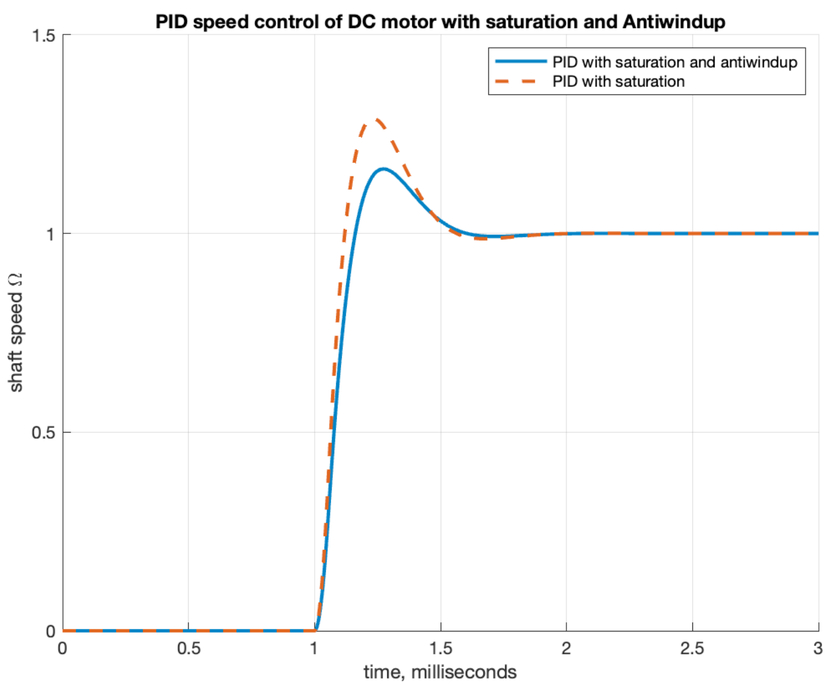

Figure 13: PID control with and without antiwindupwindup.

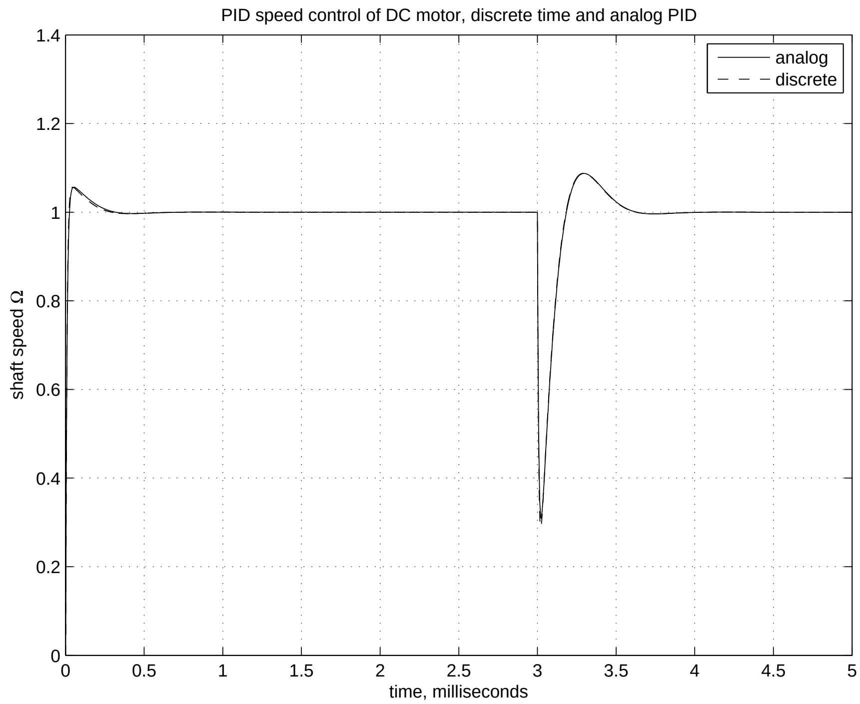

Figure 14: PID speed control for DC motor with discrete and analog controllers.

4. In problems 4 through 6 you will design the automatic steering controller that you will implement in the EECS 461 laboratory to autonomously drive your virtual vehicle on the driving simulator. You may modify the file `LinearVehicleGains.m`. As discussed in class, the control architecture will consist of two cascaded loops: an inner loop to control the haptic wheel position, and an outer loop to control the position of the vehicle with respect to the road centerline. Proportional plus derivative (PD) control will be implemented for both control loops, see Figure 15. Similar to the technique applied in problem 2, we will rely on frequency separation between the two loops to allow us to design the two controllers independently. That is, we will first design the inner loop around the haptic wheel. Then, if the haptic controller is sufficiently "fast", we can neglect its dynamics in the design of the outer loop control of vehicle position.

   For the EECS 461 project, the haptic wheel position is the steering angle input to the vehicle model to guide the vehicle along the virtual road. Thus the output of the outer loop controller is a *target* steering angle applied to the haptic wheel. We must design a controller for the haptic wheel that controls the haptic wheel angle to the desired target steering angle. We will implement PD control as in Figure 16, where the PD controller is implemented as:

$$K_{PD}(s) = K_{pm} + K_{dm}s.$$

   Note that our model includes the self-aligning torque and a gain between the haptic wheel angle and the vehicle steering angle.

   The transfer function of the haptic wheel from torque, $T(t)$, (Nmm), to wheel angle, $\delta(t)$, (radians) has been identified [2] as:

$$\delta(s) = \frac{1/J}{s^2 + (b/J)s + k/J} T(s),$$

   where J=0.683 Nmm-s^2/rad, $b = 1.2$ Nmm/rad/s and $k = 0.0097$ Nmm/rad.

[2] See Bo Yu, "Haptic_Wheel_Model.pdf," May 2014 on Canvas.

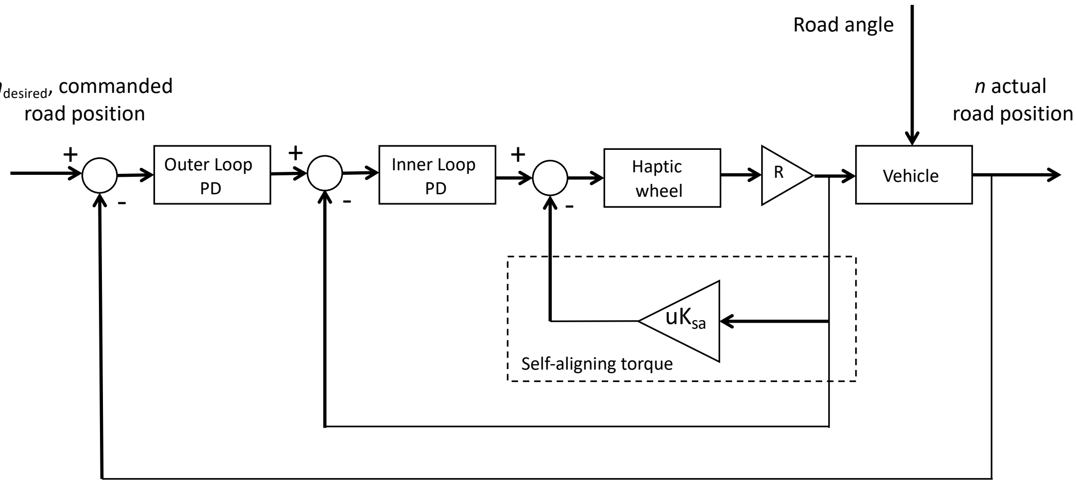

Figure 15: EECS 461 Automatic Steering Project Vehicle plus Controllers.

The transfer function from Auto-steering torque (the torque applied to the haptic wheel, Nmm) to vehicle steering angle (radians) including the self-aligning torque and the steering angle gain becomes:

$$\delta(s) = \frac{R/J}{s^2 + (b/J)s + (k + RuK_{sa})/J} T_{auto}(s),$$

where u is vehicle speed, R is the gain between haptic wheel angle and vehicle steering angle, and K_{sa} is the self-aligning torque constant in appropriate units to provide self-aligning torque in Nmm at the input to the haptic wheel. Note that the natural frequency of the system is dominated by the self-aligning torque. For $u = 16$ m/s and $K_{sa} = 100$ Nmm-s/rad-m, the characteristic roots are $-0.8785 \pm 21.6278j$, indicating that the system is almost a harmonic oscillator with a natural frequency of 21.6 rad/s (3.4 Hz).

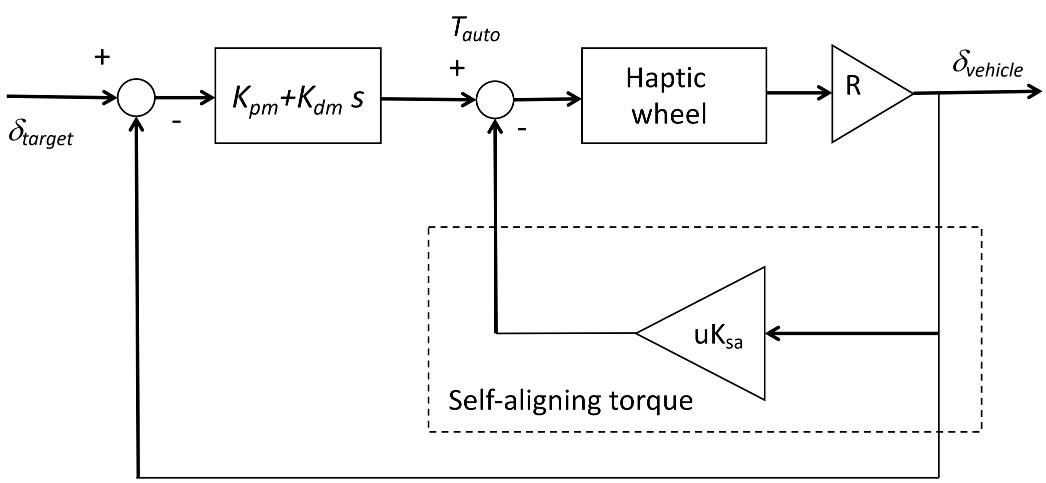

Figure 16: Haptic wheel PD position control.

Applying PD control results in transfer function from the target steering angle to vehicle steering angle:

$$\delta_{vehicle}(s) = \frac{R(K_{pm} + K_{dm}s)/J}{s^2 + ((b + RK_{dm})/J)s + (k + R(K_{pm} + uK_{sa}))/J}\delta_{target}(s),$$

(a) For $u = 16$ m/s and $K_{sa} = 100$ Nmm-s/rad-m, find gains K_{pm} and K_{dm} such that the characteristic roots of the system shown in Figure 16 have $\omega_n = 30$ and $\zeta = 0.7$ (that is, a 2% settling time, $4/(\zeta\omega_n)$, of about 0.2 seconds). Use the Simulink model `HapticBox_PD.slx` and plot the step response. For simulation purposes, the derivative is approximated as $s/(0.0001s + 1)$. Note that the cascade PD controller puts a zero at $s = -7.3$, so you should expect a significant amount of overshoot. Also note that since our system with PD control is type 0, there is a substantial steady state error. Although we could design a PID controller for the haptic wheel, we will instead design a simpler PD control and rely on the outer loop control system to correct the offset error.

(b) An alternative PD implementation for the haptic wheel is illustrated in Figure 17, resulting in the transfer function from the target steering angle to vehicle steering angle:

$$\delta_{vehicle}(s) = \frac{RK_{pm}/J}{s^2 + ((b + RK_{pm}K_{dm_{Alt}})/J)s + (k + R(K_{pm} + uK_{sa}))/J}\delta_{target}(s).$$

Again assume $u = 16$ m/s and $K_{sa} = 100$ Nmm-s/rad-m and find gains K_{pm} and K_{dm} such that the characteristic roots of the system with the alternative PD implementation have $\omega_n = 30$ and $\zeta = 0.7$. Modify Simulink model `HapticBox_PD.slx`. Call your new model `HapticBox_AltPD.slx` and plot the step response.

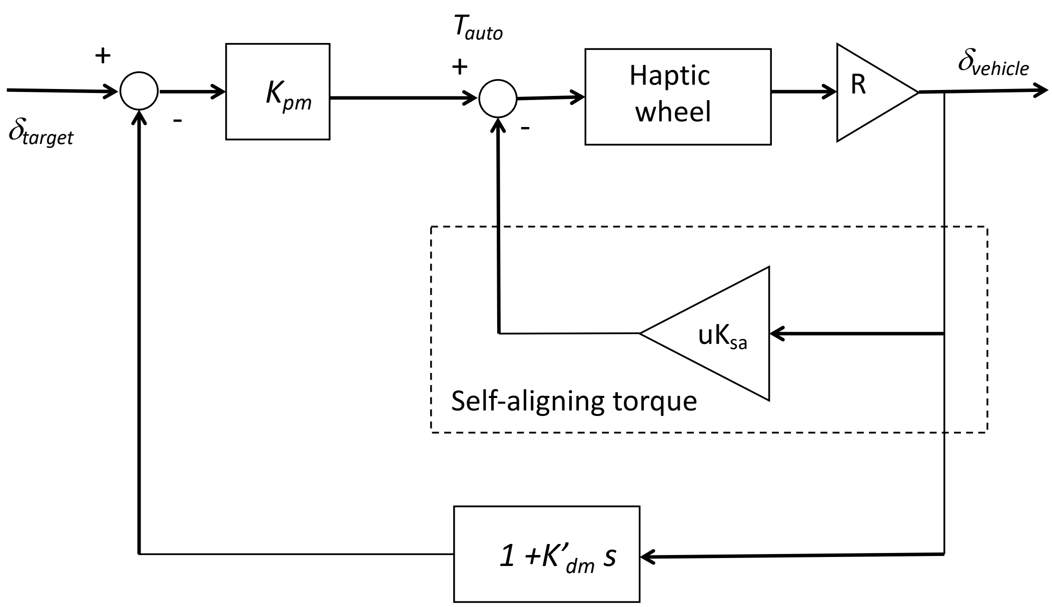

Figure 17: Haptic wheel position control with alternative PD implementation.

SOLUTION to Problem 4:

(a) For the cascade PD implementation find the values of K_p and K_d from the formulas

$$K_{pm} = ((\omega_n^2)J - k - (RuK_{sa}))/R$$
$$K_{dm} = (J2\zeta\omega_n - b)/R$$

so that if we want $\omega_n = 30$ and $\zeta = 0.7$, we should set $K_p = 1473.5$ and $K_d = 137.4$.

(b) For the alternative PD implementation find K_p and K_d from the formulas

$$K_{pm} = ((\omega_n^2)J - k - (RuK_{sa}))/R$$
$$K_{dmAlt} = (J2\zeta\omega_n - b)/(RK_{pm})$$

so that if we want $\omega_n = 30$ and $\zeta = 0.7$, we should set $K_p = 1473.5$ and $K_{dmAlt} = 0.0933$.

The step responses are depicted in Figure 18. You may wish to adjust the design parameters of the alternative PD implementation to improve the speed of response at the cost of some overshoot.

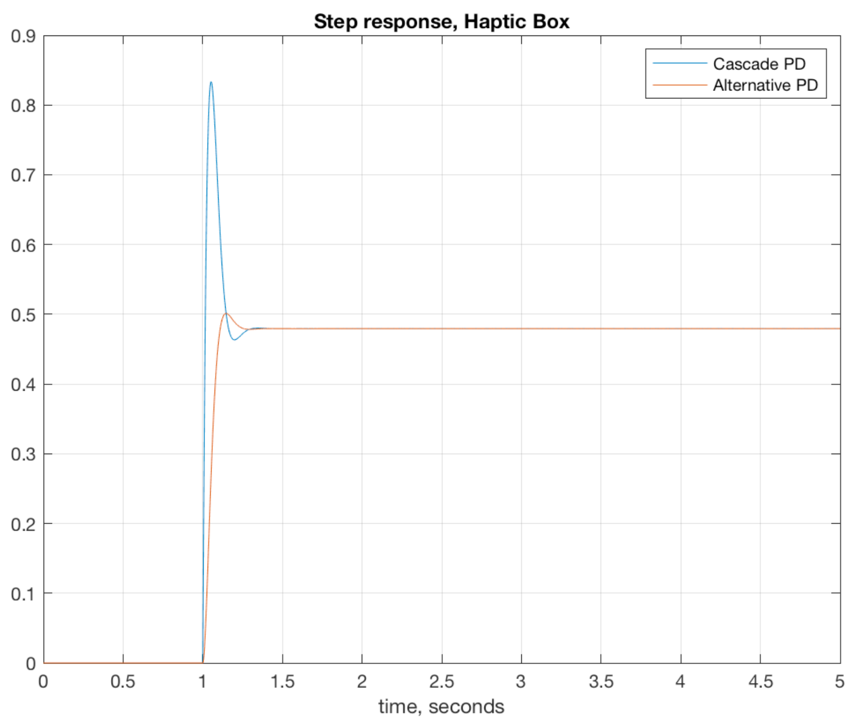

Figure 18: Haptic Wheel position control, comparison of step responses.

5. Having designed the inner loop control, we now turn our attention to the outer loop, vehicle position control. We have seen that the EECS 461 project vehicle model may be linearized as shown in Figure 19, where δ is the steering angle, ψ is the yaw angle and ϕ is the road angle, all in radians; n is the distance across the road and L is the vehicle wheelbase, both in meters, and u is the vehicle speed in meters per second.

   We will first implement automatic steering by applying cascade configuration PD control, resulting in the block diagram of Figure 20, where the PD controller is implemented as:

$$K_{PD}(s) = K_p + K_d s.$$

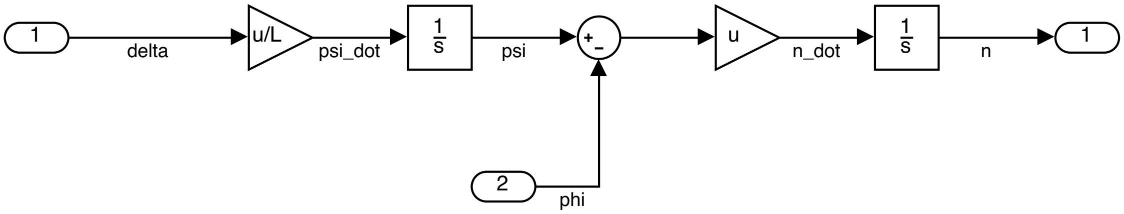

Figure 19: Linearized Vehicle Model.

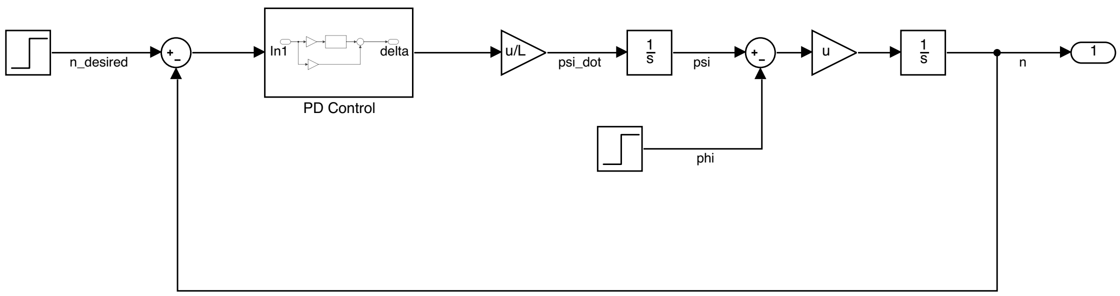

Figure 20: Linearized Vehicle with PD controller.

We saw in class that the transfer function from the command $n_{des}(t)$ to the output $n(t)$ (assuming the road input $\phi = 0$ is given by

$$N(s) = \frac{(1/s^2)(u^2/L)(K_p + K_d s)}{1 + (1/s^2)(K_p + K_d s)} N_{des}(s)$$
$$= \frac{(u^2/L)(K_p + K_d s)}{s^2 + (u^2/L)K_d s + (u^2/L)K_p} N_{des}(s).$$

and thus the characteristic polynomial is given by $s^2 + (u^2/L)K_d s + (u^2/L)K_p$. Hence, if we wish that the characteristic roots have natural frequency ω_n and damping ζ, then we can solve for K_p and K_d by equating coefficients:

$$s^2 + (u^2/L)K_d s + (u^2/L)K_p = s^2 + 2\zeta\omega_n s + \omega_n^2.$$

If we implement the alternative derivative as we did for the haptic controller in problem 3, the transfer function from $n_{des}(t)$ to $n(t)$ is given by

$$N(s) = \frac{((u^2/L)K_p/s^2)}{1 + ((u^2/L)K_p/s^2)(1 + K_d s)} N_{des}(s)$$
$$= \frac{(u^2/L)K_p}{s^2 + (u^2/L)K_p K_d s + (u^2/L)K_p} N_{des}(s).$$

Again, one may choose K_p and K_d to yield desired characteristic roots.

(a) Suppose we wish the characteristic roots to have $\omega_n = 10$ and $\zeta = 0.7$, corresponding to a rise time of $t_r = 0.18$ seconds, an overshoot of 7% and a settling time of 0.6 seconds. If the vehicle speed is 16 meters/second (about 35 miles per hour) and the wheelbase is 1.6 meters, what values should K_p and K_d have for the PD controller implementation in Figure 20? Use the Simulink model `LinearVehicle_PD.slx` and plot the step response (again, the controller is implemented in the model using an approximate derivative with time constant, $\tau = 0.0001$ seconds).

(b) Repeat the previous part by choosing K_p and K_d in the alternative implementation to yield the same characteristic roots. Modify `LinearVehicle_PD.slx` to place the derivative gain in the feedback path. Call your new model `LinearVehicle_AltPD.slx` and plot the step response.

SOLUTION to Problem 5:

(a) For the cascade PD implementation find the values of K_p and K_d from the formulas

$$Kp = (\omega_n)^2(L/u^2)$$
$$Kd = 2\zeta\omega_n(L/u^2)$$

so that if we want $\omega_n = 10$ and $\zeta = 0.7$, we should set $K_{\mathrm{p}} = 0.25$ and $K_{\mathrm{d}} = 0.0358$. The transfer function from n_{desired} to n is

$$\frac{n(s)}{n_{\text{desired}}} = \frac{\frac{u^2 K_{\mathrm{d}}}{L}}{s^2 + \left(\frac{u^2 K_{\mathrm{d}}}{L}\right) s + \frac{u^2 K_{\mathrm{p}}}{L}}$$

The step responses are depicted in Figure 21. Note that only the alternative implementation achieves the desired overshoot and rise time.

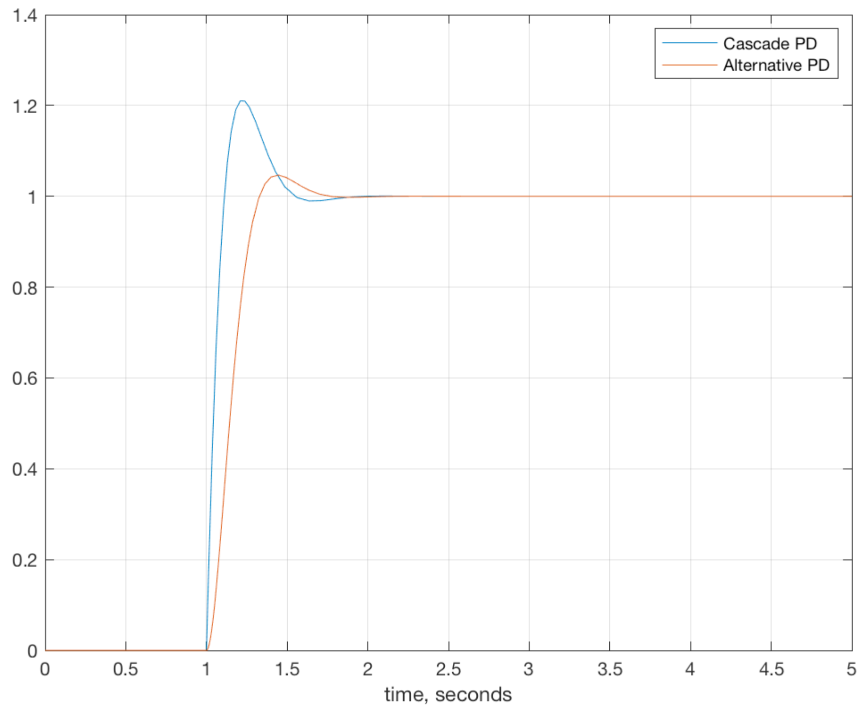

Figure 21: Linear Vehicle position control, comparison of step responses.

6. Students have been successful implementing automatic steering in the EECS 461 lab using the simple cascade configuration illustrated in Figure 15 which incorporates the linear vehicle model and two control loops: inner loop control of haptic wheel position and outer loop control of the distance from the road centerline.

   (a) Build a Simulink model called `LinearVehiclePlusHaptic.slx` of the system shown in Figure 15. Use the cascade PD controllers you designed in problems 4 and 5 above plus the linear vehicle and haptic wheel models. Set $n_{desired}$ to 0 and plot the response to a unit step input in road angle. Your Simulink model should look similar to Figure 22.

   (b) Using the project sample time $T = 0.01$ seconds, design discrete-time approximations to the analog PD controllers, and simulate the response as in part (a). Hand in a plot with both the analog and discrete-time responses. Note that the Simulink discrete derivative does not work in triggered subsystems. Implement your discrete PD controllers as shown in Figure 23, remembering to put rate transition blocks around the controller and zero-order hold blocks to sample the continuous feedback signals (for simulation only, you won't require them for the project). In the Simulink Configuration Parameters/Solver menu, choose fixed step solver ODE8 and set the sample time to $dT = 0.01$.

   (c) Record the gains you used in your model. In the final project report, state the controller gains that you established in this problem set and the final gains you implemented for your project controller and, if there are significant differences, explain why. In the lab, you will be asked to demonstrate your vehicle driving autonomously at a higher speed than the design speed (usually about 20 m/s). Your best approach to doing this is to recalculate all the gains you determined in the previous problems rather than trying to "guess and test" in the lab. Once you've established the appropriate gains at several speeds, you may wish to gain schedule your controller to provide stable operation over the speed range.

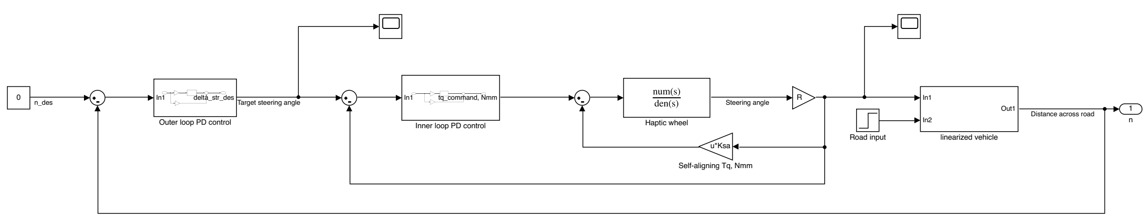

Figure 22: EECS 461 Automatic Steering Project Vehicle plus Controllers.

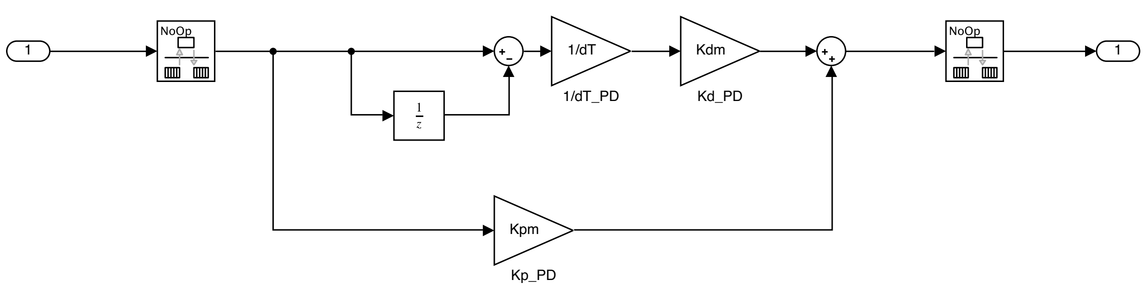

Figure 23: Simulink implementation, discrete PD.

SOLUTION to Problem 6:

In past semesters, students were given gains $K_{dm} = 100, K_{pm} = 1000, K_d = 0.05$ and $K_p = 0.45$ as a starting point and asked to tune the gains *ad hoc* so that the vehicle would stably traverse the virtual road at 35 miles/hour with minimal overshoot. Figure 24 compares the vehicle response to a step input in road angle for the recommended starting point gains and the gains calculated in problems 4 and 5.

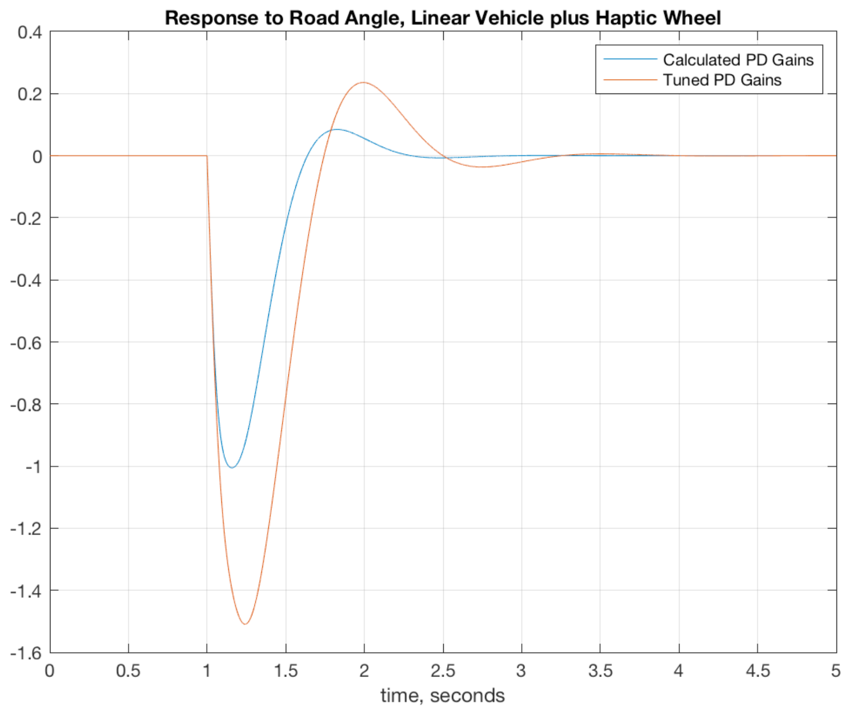

Figure 24: Response to Road Angle Step Input.