University of Massachusetts Lowell

Department of Mechanical Engineering

Dynamic Systems and Controls, MECH.5540 Fall 2020

Project Title

Controller design for Pioneer ROS in Simulink

Date of submission

11/29/2020

SUBMITTED BY

Srivar Kashyap

Part 1

Open-Loop Behavior of Robot

1. The Pure Pursuit block was removed as shown in the instructions.

2. Replacement parameters for the Pure Pursuit block were implemented as follows:

    a. The waypoints were set at (0,0) and (100, 0) to drive the robot in a straight line.

    b. A constant block of value 0 was placed for omega further assisting in a straight line trajectory

    c. An input step function was placed as a robot's linear velocity reference signal

The figure below shows the new configuration.

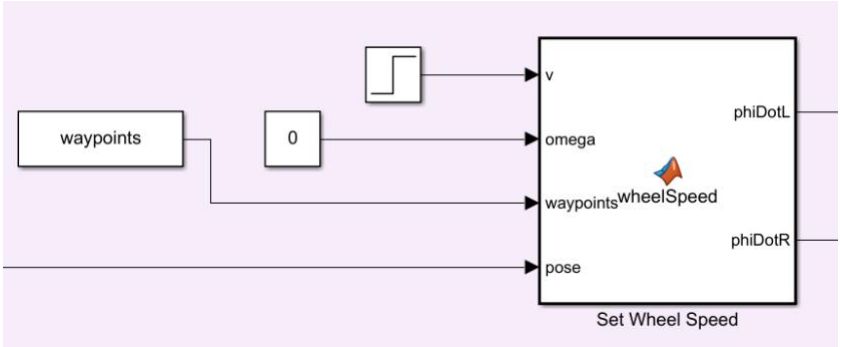


Figure 1. Basic Robot Reference Blocks

3. Running the simulation

a. A calculation was added to determine the actual forward speed of the robot from the individual wheel speeds, where

$$v = \frac{1}{2}\left(v_L + v_R\right)$$

i. The following is a Simulink diagram of the calculation and the resultant linear speed (v) data extraction

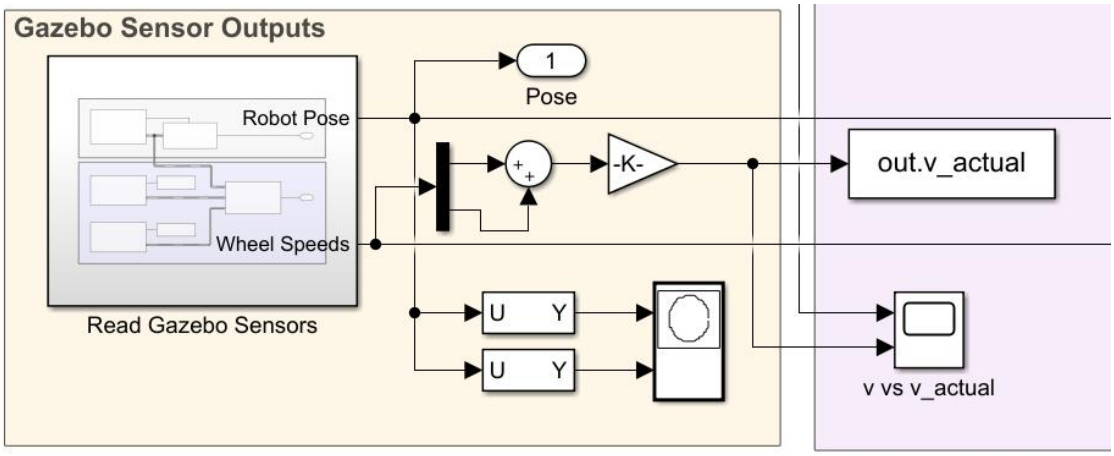


Figure 2. Simulink Diagram of Linear Wheel Speed Sensor

b. The following plot displays the response of the uncontrolled simulation to a unit step input.

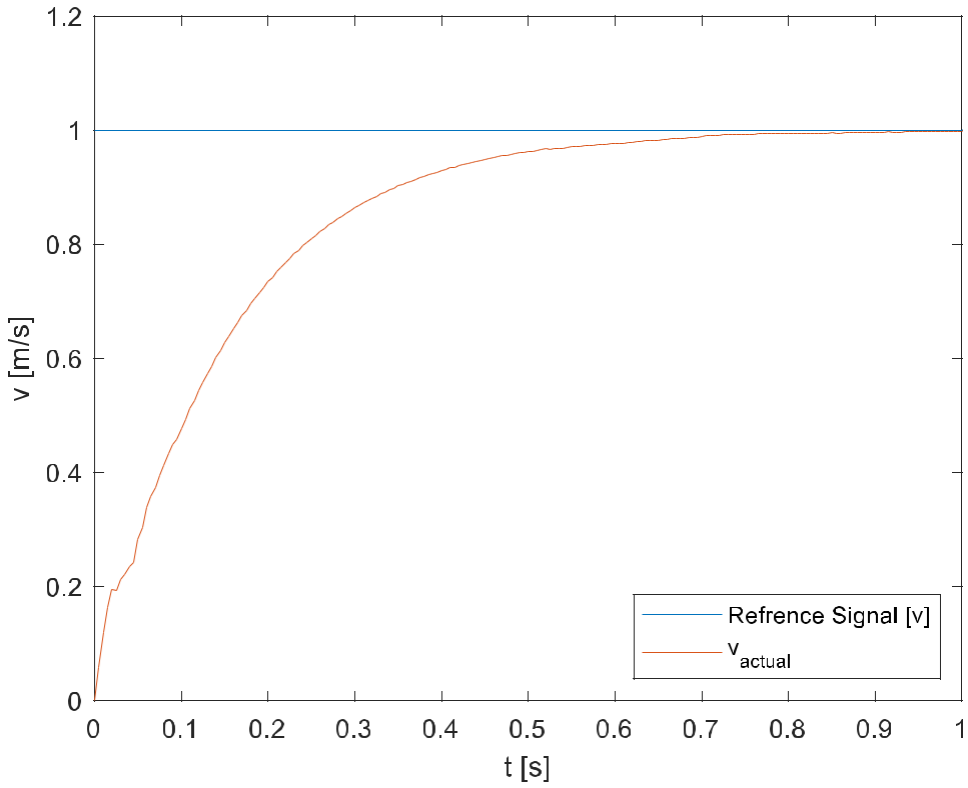


Figure 3. Response of Gazebo Differential Drive Robot to a Unit Step Reference

c. The simplest possible transfer function G(s) for this system would be a first-order system of the form

$$G(s) = \frac{K}{\tau s + 1}$$

Where τ is the time constant of the system, and K is the steady state gain.

d. To find the transfer function G(s) to model this system, the step response of the robot was observed for the value of the time constant:

$$\tau = 0.156$$

Therefore, the transfer function is,

$$G(s) = \frac{1}{0.156 s + 1}$$

i. The following is the plot of the step response of the determined transfer function G(s) compared to the step response of the system:

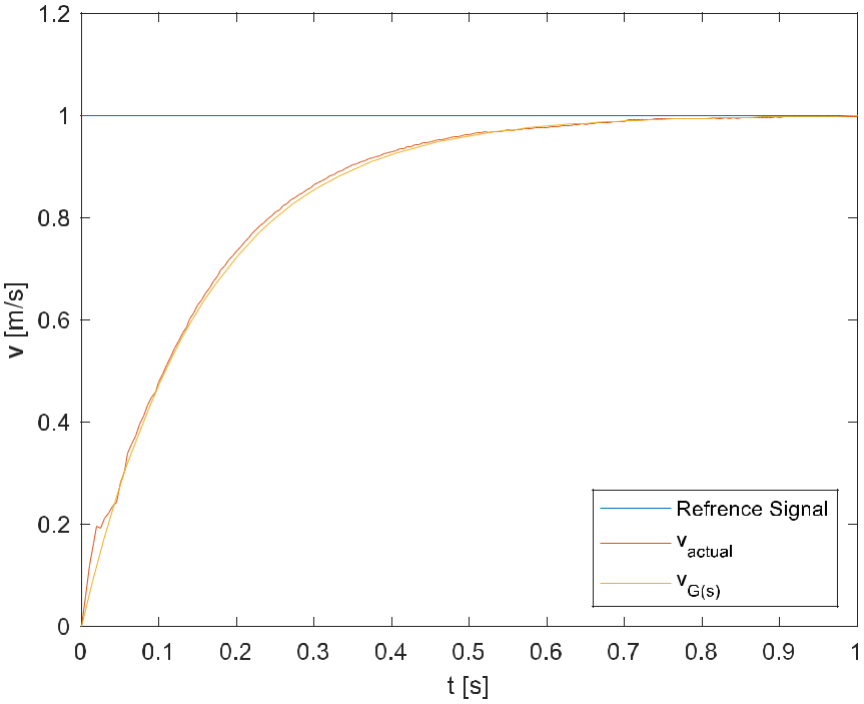


Fig4. Comparison of Linear Velocity Responses and References

ii. The simulated open-loop response G(s) is very close to the original response of the system. Some inaccuracies may arise from the precision when determining the time constant τ, which can account for any slight differences between simulated and actual response.

Part 2

Controller design for the Pioneer robot

1. The time taken for the robot to reach T1 and T2 were determined from the plot of !"#$!% a. T1 = 63.2% steady state value

$$1 = 0.156$$

b. T2 = 90% steady state value

$$2 = 0.351$$

2. The rise time for the closed-loop system is equal to half of T2

$$\frac{2}{} \quad \frac{0.351}{}$$

$$(= \quad 2 \quad = \quad 2 \quad = 0.175$$

3. Determining the system type
   a. In order for the system to have a zero steady state error with respect to a step reference it has to be at least a type 1 system.
   b. In order to create a type 1 system, the system should have one pole at the origin when expressed in the form,

$$'.^{-} = ( ) ( )$$

Given the plant has no poles at the origin the controller should be more than just a proportional controller. The controller chosen for this project was a PI controller, for simplicity and also to avoid any noise being magnified by the derivative controller of the PID. This adds one pole at the origin, making the system a type 1 system.

4. A third design requirement is to limit any potential overshoot (M_p) to be less than 10%. From the rise time criteria, the damping coefficient (ζ) was determined:

$$/ \leq 10\% \quad / = \exp @ \qquad \frac{- \quad {}_0\mathrm{E} \rightarrow}{\mathrm{D1} -} \quad = \mathrm{G} \quad \frac{0 \ln(/)_0 \; {}_0 = 0.591}{+ \ln(/)}$$

From the rise time found above, the natural frequency was found as follows:

$$\frac{1.8}{} \qquad \frac{1.8}{} \quad \frac{1.8}{}$$

$$(=0.175 \quad (= \qquad . \rightarrow . = \qquad (= 0.175 = 10.3$$

The PI controller K(s) has the structure:

$$\frac{1}{} \qquad \frac{/+\ 1}{\qquad\qquad}$$

$$( ) = \ / + \qquad =$$

The addition of the PI controller results in the total system taking the form of a 2nd order dynamic system. The controller gains can then be determined by comparing the numerator of systems characteristic equation to the standard characteristic equation of a 2nd order system as follows:

$$^{0}+^{1}\mathrm{L}\ _{2}+1\mathrm{M}\ +\ ^{1}=\ ^{0}+2\ .\ +.^{0}$$

This results in following initial controller gains:

$$_{/3)} = 1.010$$
$$_{13)} = 16.410$$

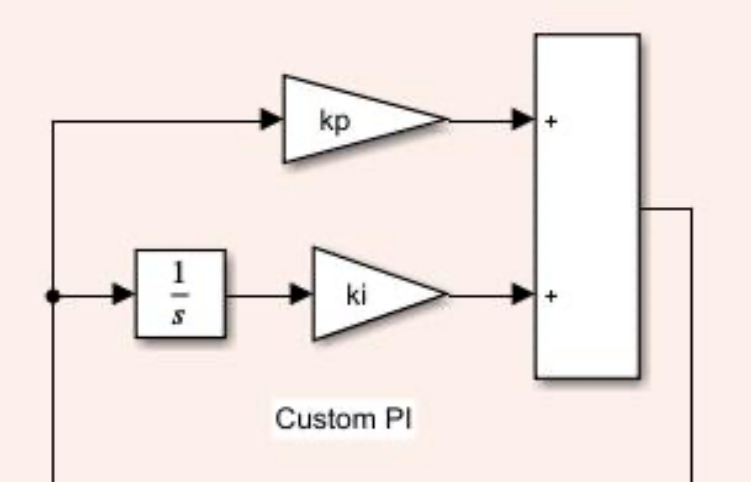
The initial gain values did not meet the design requirements and were changed in increments of 0.005 until the controller performed to the required parameters. The resulting final gains are:

$$_{/} = 1.210$$
$$_{1} = 16.560$$

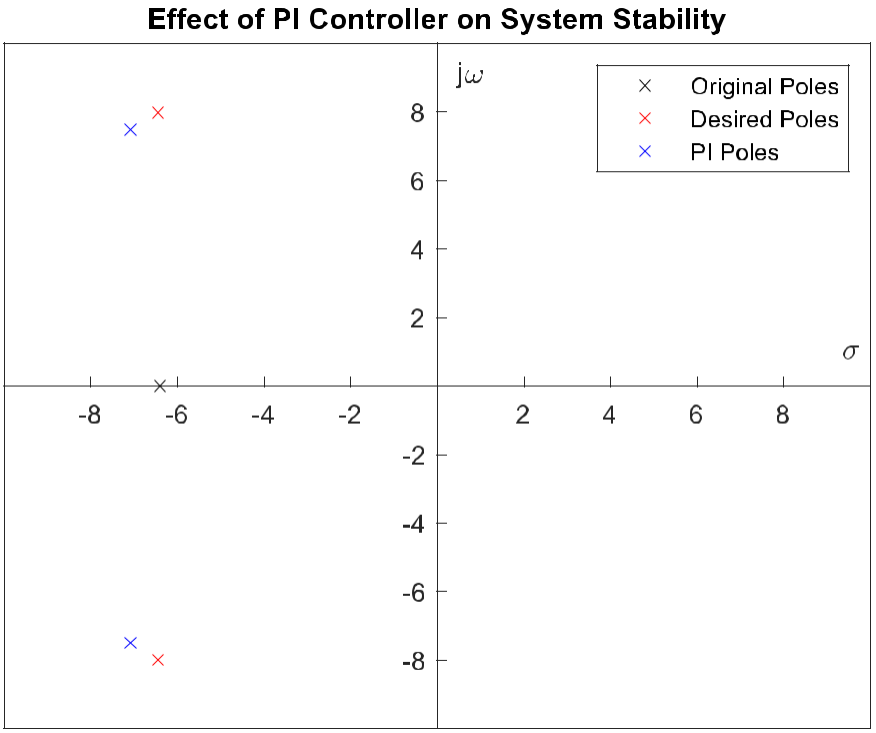
The resulting K(s) controller is:

$$_{(\ )=1.210+\ 16.560} \qquad \frac{\qquad\qquad}{\qquad\qquad}$$

5. Implement K(s)
   a. Simulink diagram



b. Stability test



6. The system was given a step reference to test the response of the controlled, closed-loop system.
   a. The plot of the controlled system response to a step input can be seen below:

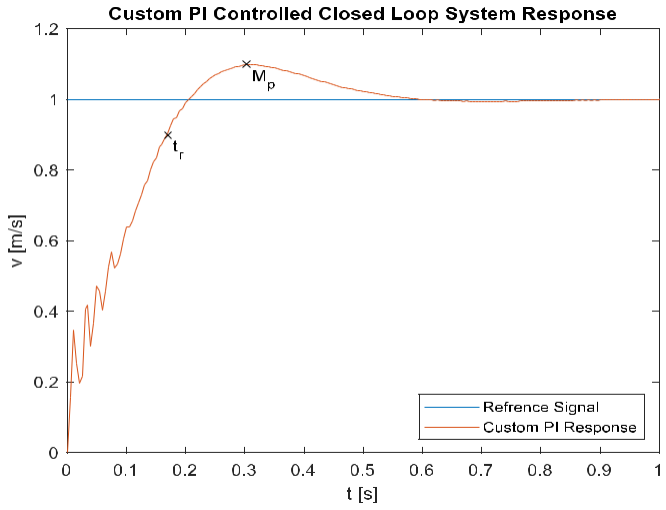


Figure X. Plot of the step response of the manual PI controlled system

The rise time and peak overshoot can be seen in the plot above. The resultant rise time was 0.17 seconds, and the max overshoot was 9.9%. This response satisfied the specifications of the controller design, where the rise time needed to be less than 0.175 seconds and the overshoot less than 10%.

7. Using the built-in PID controller

The control gains calculated in the previous section were entered into the built-in PID Simulink block, with the derivative component set to zero. The result was plotted compared to the manual PID.

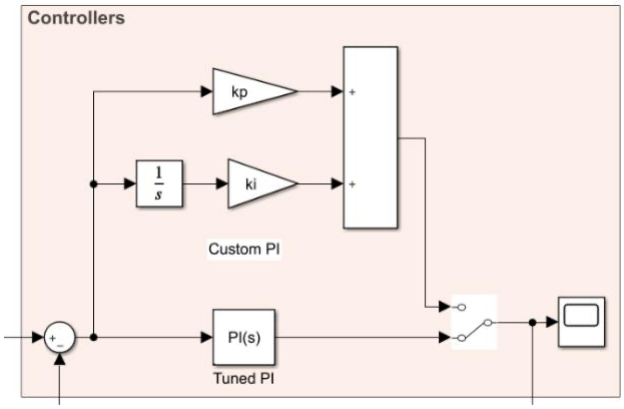


Figure X. Diagram of the PID controller block placement.

a. Plotting the response of both controllers

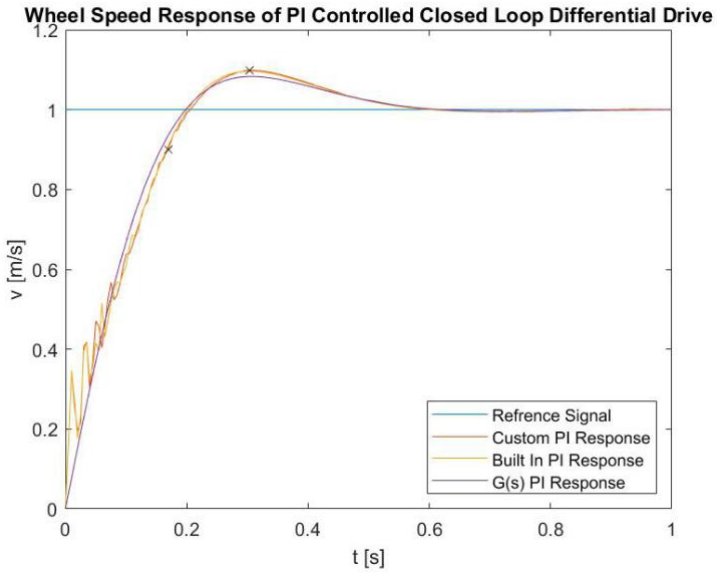


Figure X. Plot of the step response of the manual and built-in PID controller

The response of the system with the built-in PID controller is very similar to the response of the system with the manual controller. The rise time and max overshoot are the same.

8. Pulse reference trajectory tracking
    a. The system response was tested for a new input reference, a combination of a constant and pulse signal.

The new signal is a combination of two signals:

- A constant signal of magnitude 0.5
- A pulse signal

The response of the system to this reference is plotted below.

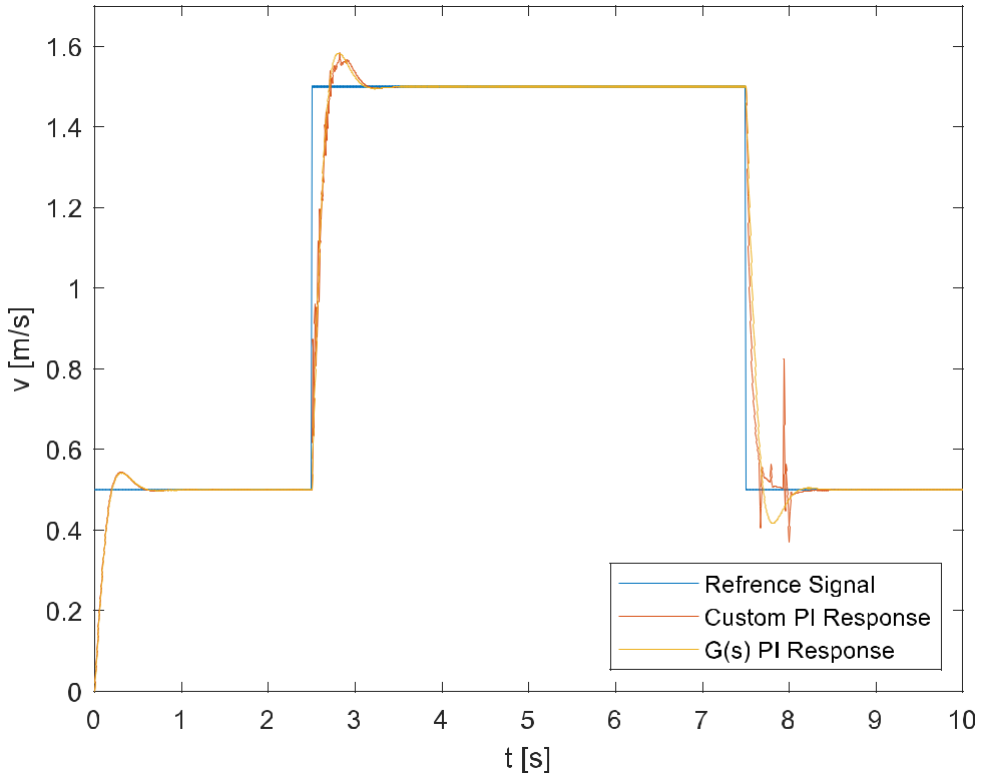
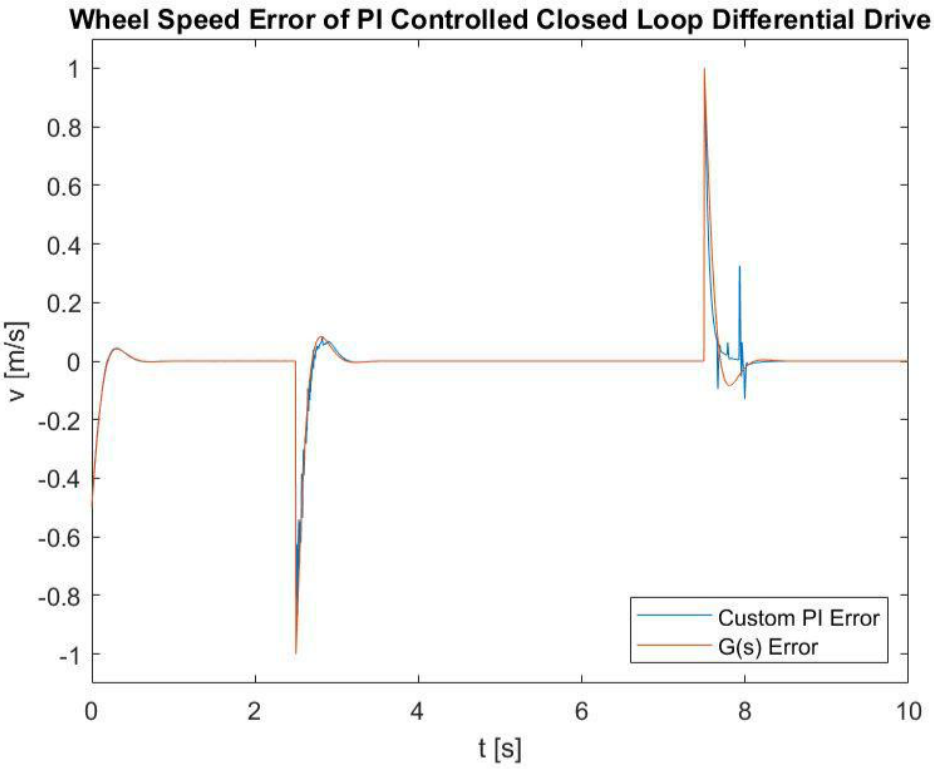


Figure X. System response to a pulse input signal

9. Calculate error Demonstrate plots



The error signal was calculated by subtracting the actual system response from the ideal system response. The error spikes at the beginning and at the end of the pulse signal.

Appendix A: MATLAB Code

```
close all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Simulation Parameters:
    tFin = 10; %[s]
    tSamp = 0.005; %[s]
    waypoints = [0 0; 100 0]; %[x1 y1 x2 y2] [m]
    fig = 2; %Initial Figure
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Save|Import Simulation Results
    %Retrieve Data
        %tSim = out.tout';

        %stepRef = out.refrenceSig';
        %pulseRef = out.refrenceSig';

        %stepResp_OL = out.v_actual';
        %stepResp_CL_Cust = out.v_actual';
        %stepResp_CL_BuiltIn = out.v_actual';
        %pulseResponse_CL_Cust = out.v_actual';

    %Save Data
        %{
    save('GazeboResponse.mat','tSim',... %Time
                              'stepRef', 'pulseRef',... %Refrence
                              'stepResp_OL','stepResp_CL_Cust','stepResp_CL_BuiltIn',...
                              %Step 'pulseResponse_CL_Cust'... %Pulse );

    %}

    %Load Data
    load('GazeboResponse.mat');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Open Loop Model
    T1 = 0.156; %[s]
    T2 = 0.351; %[s]
    tSettle = 0.714; %[s]

    G = tf(1,[T1 1])

    fprintf("\nController Requirements: \"...
          + "nRise Time: < %.3f s \nOvershoot: < 10.000 %%\n",...
             T2/2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Controller Design: PI
    %Design Requirements
        nFreqCont = (2*1.8)/T2;
        dampCont = 4.6/(nFreqCont*tSettle);

    %Determine Controller Gains
        charPolyResponse = [1 (2*nFreqCont*dampCont) (nFreqCont^2)];

        kpOriginal = (charPolyResponse(2)*T1) - 1;
        kiOriginal = charPolyResponse(3)*T1;

        kp = kpOriginal + 0.2; %Gain tweaks for final tuning
        ki = kiOriginal + 0.15;
```

```
        charPolyControl  = [1 (kp+1)/T1 ki/T1];

    %Confirm System Stability
        figure(fig);
        fig = fig + 1;

        stabilityG = roots([T1 1]);
        stabilityReponse = roots(charPolyResponse);
        stabilityControl = roots(charPolyControl);

        plot(real(stabilityG),imag(stabilityG),'kx');
        hold on;
        plot(real(stabilityReponse),imag(stabilityReponse),'rx');
        hold on;
        plot(real(stabilityControl),imag(stabilityControl),'bx');
        ax = gca;
        ax.XAxisLocation = 'origin';
        ax.YAxisLocation = 'origin';
        axis([-10 10 -10 10]);
        title('Effect of PI Controller on System Stability');
        xlabel('\sigma');
        ylabel('j\omega');
        legend('Original Poles','Desired Poles', 'PI Poles','location','northeast');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Closed Loop Model
    K = pid(kp,ki,0)
    TF = feedback(K*G,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Step Response
    %Simulate Step Response
        [vG, tG] = step(G,tFin);
        [vMATLAB, tMATLAB] = step(TF,tFin);

    %Plot Step Response Plots
        % Uncontrolled Response
            figure(fig);
            fig = fig + 1;

            plot(tSim,stepRef,tSim,stepResp_OL);
            title('Actual Wheel Speed Response Compared to Input Velocity');
            xlabel('t [s]');
            ylabel('v [m/s]');
            legend('Reference Signal [v]', 'v_{actual}', 'location','southeast');
            axis([0 1 0 1.2]);

        %Uncontrolled Response w/ G(s)
            figure(fig);
            fig = fig + 1;

            plot(tSim,stepRef,tSim,stepResp_OL,tG,vG);
            title('Wheel Speed Response of Uncontrolled Open Loop Differential Drive');
            xlabel('t [s]');
            ylabel('v [m/s]');
            legend('Refrence Signal', 'v_{actual}', 'v_{G(s)}','location','southeast');
            axis([0 1 0 1.2]);


        %PI Response
            tr_Gazebo = 0.170; %[s]
            Mp_Gazebo = 0.099; %[m/s]
            tp_Gazebo = 0.303; %[s]
```

```
            tlabel = [tr_Gazebo tp_Gazebo]; %[s]
            vlabel = [0.9 Mp_Gazebo+1]; %[m/s]

            %Custom PI Only
                figure(fig);
                fig = fig + 1;

                plot(tSim,stepRef,tSim,stepResp_CL_Cust); %Plot
                Data hold on;
                plot([tr_Gazebo tp_Gazebo], [0.9 Mp_Gazebo+1], 'kx'); %Plot tr and
                 Mp text(tlabel, vlabel,{' t_{r}', ' M_{p}'},...
                     'VerticalAlignment','top','HorizontalAlignment','left' );
                title('Custom PI Controlled Closed Loop System Response');
                xlabel('t [s]');
                ylabel('v [m/s]');
                legend('Refrence Signal','Custom PI Response',...
                       'location','southeast');
                axis([0 1 0 1.2]);

            %All PI Responses
                figure(fig);
                fig = fig + 1;

                fprintf("\nResponse Charachteristics of Custom PI Controller:"...
                      + "\nRise Time: %.3f s \nOvershoot: %.3f %%\n",...
                        tr_Gazebo, Mp_Gazebo*100);


plot(tSim,stepRef,tSim,stepResp_CL _Cust,tSim,stepResp_CL_BuiltIn,tMATLAB,vMATLAB); %Plot
                Data title('Wheel Speed Response of PI Controlled Closed Loop Differential
                Drive'); xlabel('t [s]');
                ylabel('v [m/s]');
                legend('Refrence Signal','Custom PI Response', 'Built In PI Response', 'G(s) PI
Response',...
                       'location','southeast');
                axis([0 1 0 1.2]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Pulse Response
    %Synthesize Pulse Wave
        %Parameters
            c = 0.5; %[m/s]
            A = 1; %[m/s]
            T = 10; %[s]

    %Simulate Pulse Response
        [vMATLAB, tMATLAB] = lsim(TF,pulseRef,tSim);

    %Plot Pulse Response
        figure(fig);
        fig = fig + 1;

        plot(tSim,pulseRef,tSim,pulseResponse_CL_Cust,tMATLAB,vMATLAB);
        title('Wheel Speed Response of PI Controlled Closed Loop Differential Drive');
        xlabel('t [s]');
        ylabel('v [m/s]');
        legend('Refrence Signal', 'Custom PI Response','G(s) PI Response',...
               'location','southeast');
        axis([0 10 0 1.7]);

    %Determine Error Signal
        E_MATLAB = vMATLAB-pulseRef';
        E _Gazebo = pulseResponse_CL_Cust - pulseRef;

        figure(fig);
        fig = fig + 1;
```

```
        plot(tSim,E_Gazebo,tSim,E_MATLAB);
        title('Wheel Speed Error of PI Controlled Closed Loop Differential Drive');
        xlabel('t [s]');
        ylabel('v [m/s]');
        legend('Custom PI Error','G(s) Error','location','southeast');
        axis([0 10 -1.1 1.1]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fprintf('\n');
```

Appendix B: Simulink Diagram

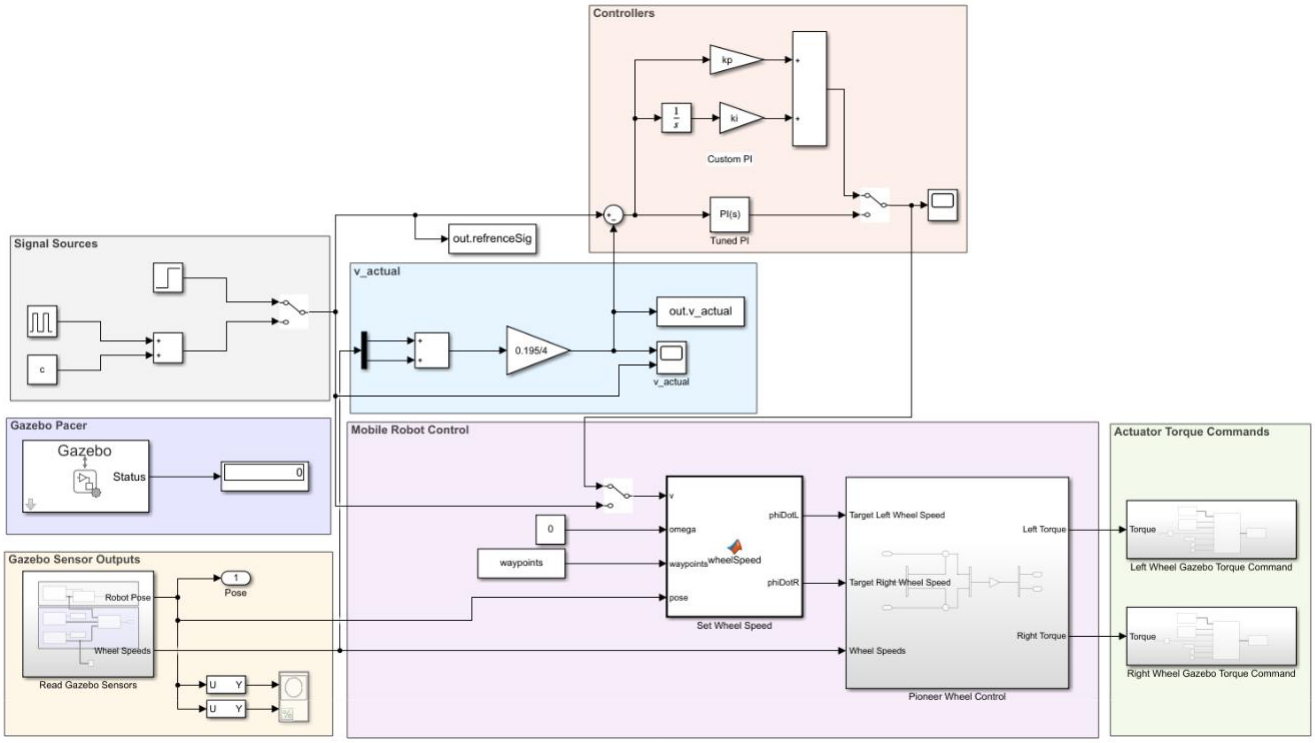


Figure 1B. Complete Simulink Diagram of a PI Controlled Differential Drive Robot