Mini Project: Matlab Documentation

EENG 350 February 26, 2021

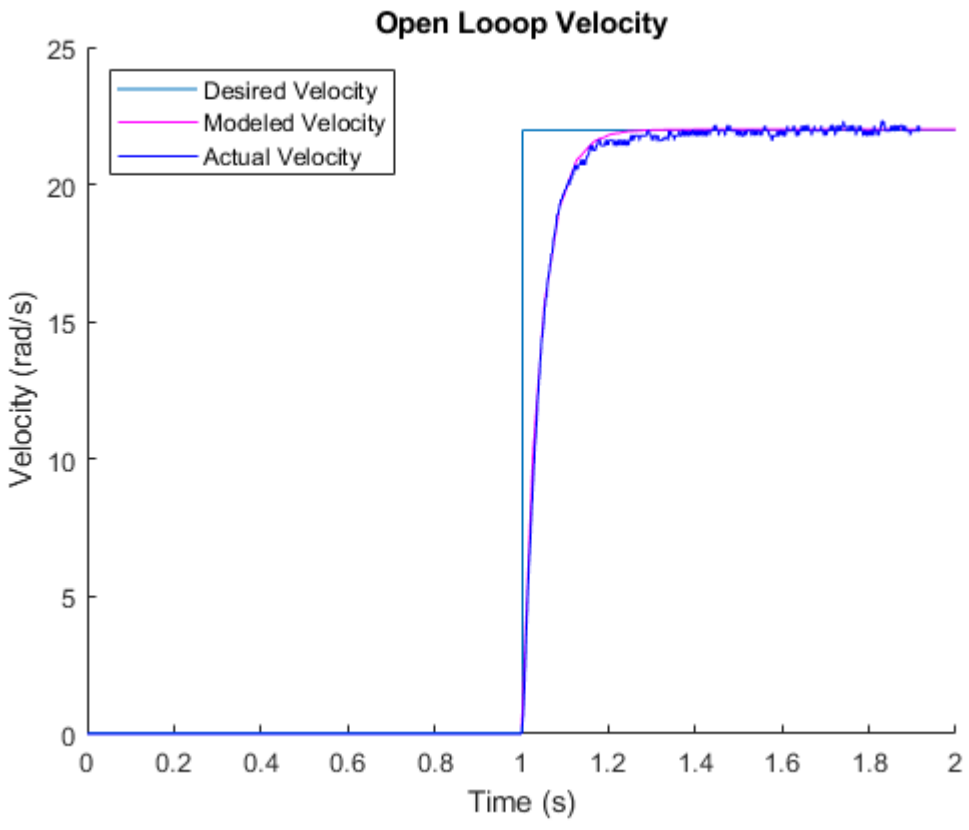
Luis Cisneros

Contents

Mini Project 4.6: Simulate and Tune the Model to Match the 'actual' Data

The controller was designed using MATLAB to guess and check sigma and K values that could be used to simplify the transfer function for the step response recorded from the motor step response data. This is done graphically by overlaying the two graphs to determine the sigma and K values that best match the recorded motor performance. NOTE: The format of the 'actual' matrix should be [0,dataColumn]

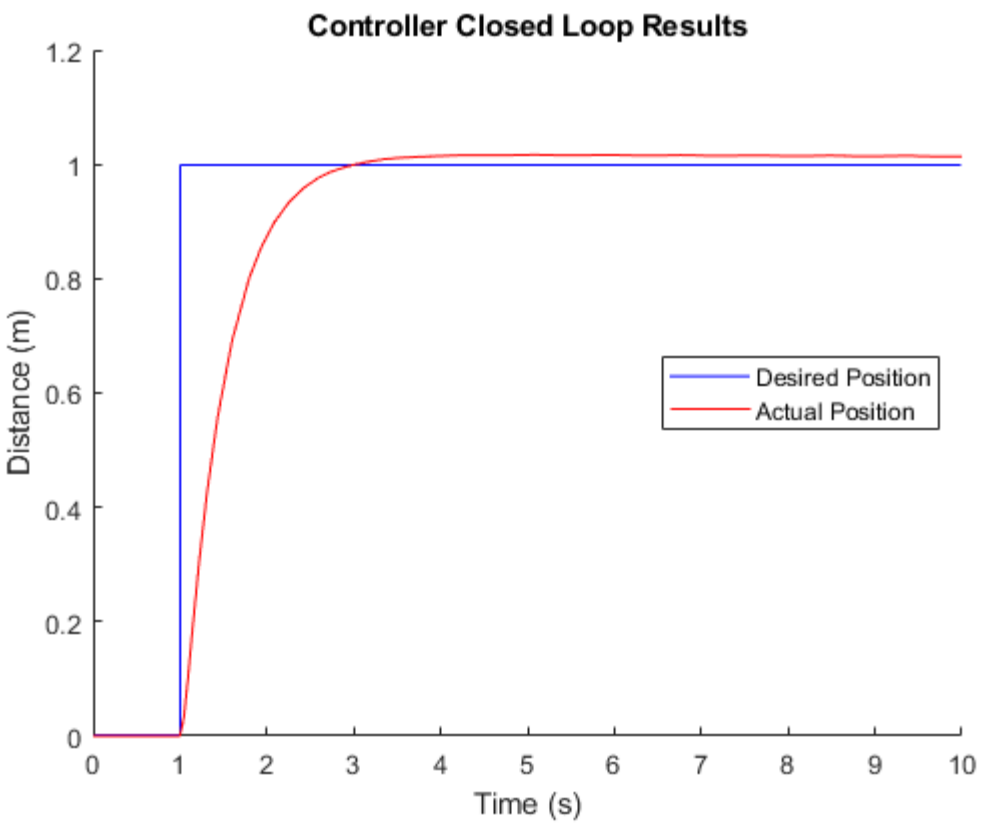
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% User variables %%%%%%%%%%%%%%%%%%%%%%%%%%%%
sampleRate = 1; % Number of MILLISECONDS between samples

sigma = 23.5; % Tune to match 'actual' plot
K = 22; % Gain. AKA the final velocity
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Add the time column to the 'actual' matrix (uncomment as needed)
load('actual.mat'); % Load the measured velocity values
actual(:,1) = (0 : 1 / (sampleRate * 1000) : (size(actual,1) - 1) / 1000); % Use the sampling rate to create the time column
out = sim('Mini_motorsim4_1'); % Run the simulation and make the results part of the out. object

% Plot the results
clf; figure(1); hold on;
plot(out.step * K); % Plot the step input
plot(out.newv,'m-'); % Plot the modeled step response
plot(actual(:,1),actual(:,2), 'b-'); ylabel('Velocity (rad/s)'); xlabel('Time (s)'); % Plot the actual step response
title('Open Looop Velocity'); legend({'Desired Velocity','Modeled Velocity','Actual Velocity'},'Location','northwest'); hold off;
```



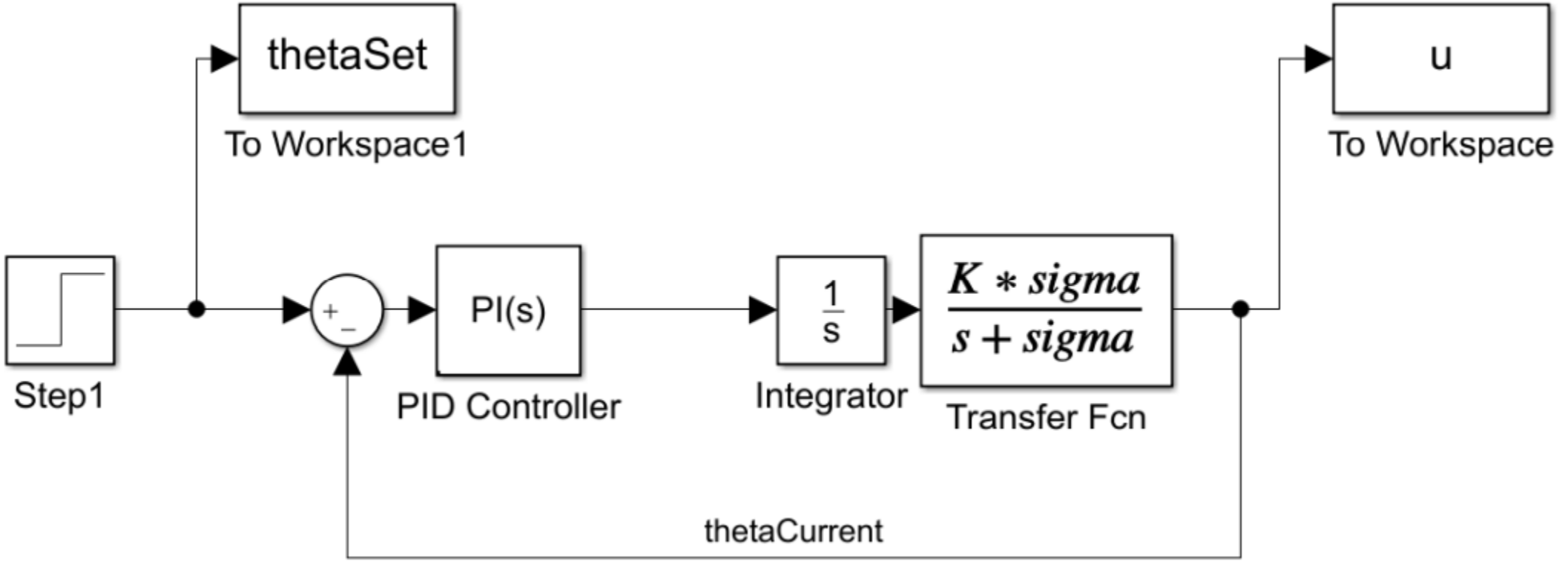
Mini Project 4.7: Design a Controller for Your Motor

Simulink was then used to tune a PI controller to the deisred specifications of a rise time of 1 second and a percent overshoot of less than or equal to 12% while having zero steady state error. The PI controller that was selected had a rise time of exactly 1 second while having a percent overshoot of 1.58% as seen in the screenshot below.

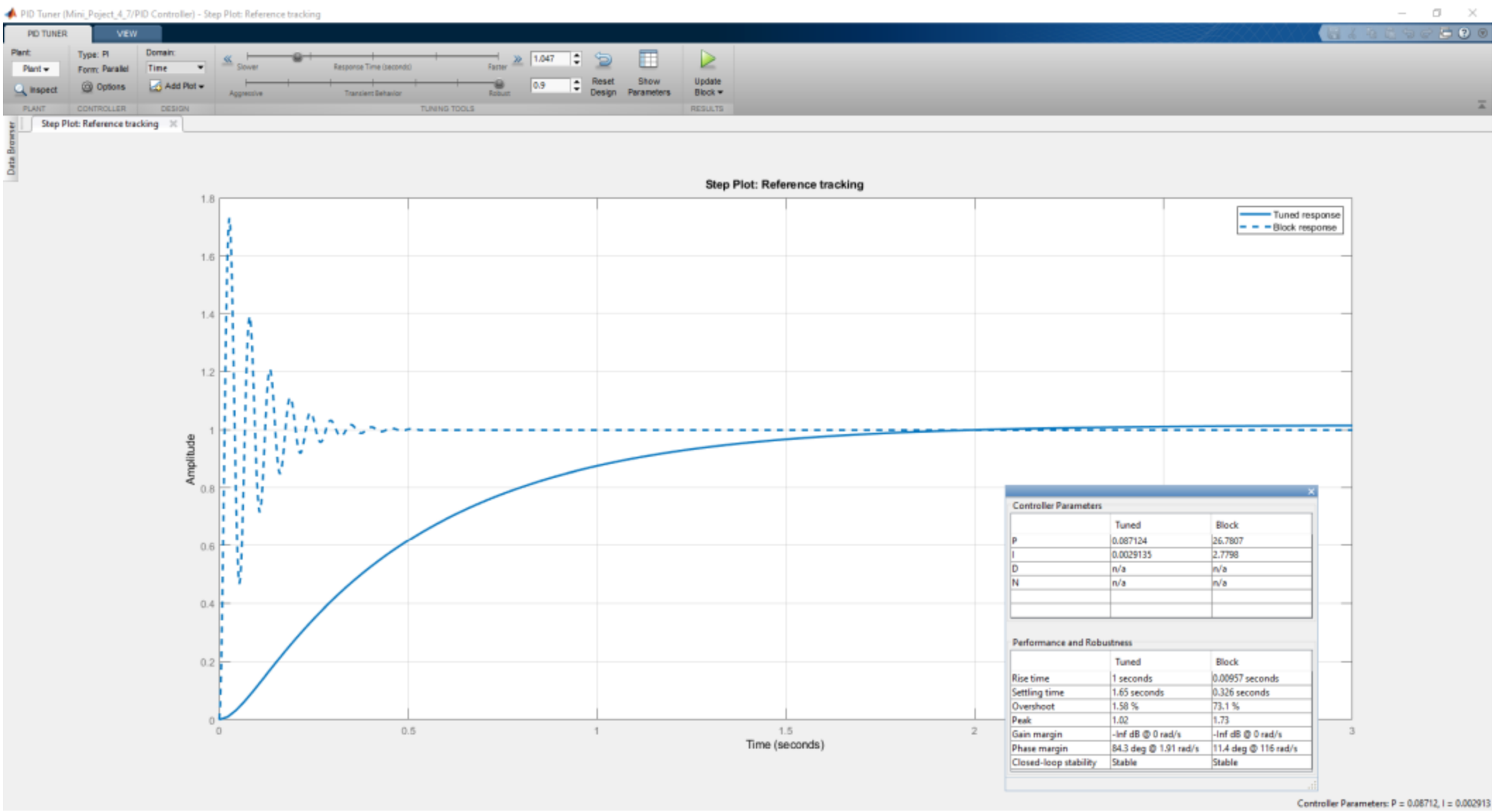
```
%%%%%%%%%%%%%%%%%%%%%%%%%%% User variables %%%%%%%%%%%%%%%%%%%%%%%%%%%
sigma = 23.5;
K = 22;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

out = sim('Mini_Poject_4_7'); % run the simulation

% Plot the results
clf; figure(2); hold on;
plot(out.thetaSet, 'b-'); ylabel('Distance (m)'); % Plot the desired position
plot(out.u, 'r-'); ylabel('Distance (m)'); xlabel('Time (s)'); % Plot the actual position
title('Controller Closed Loop Results'); legend({'Desired Position','Actual Position'},'Location','east'); hold off;
```



Simulink Block Diagram



Simulink Tuned PI Performance



Published with MATLAB® R2019a