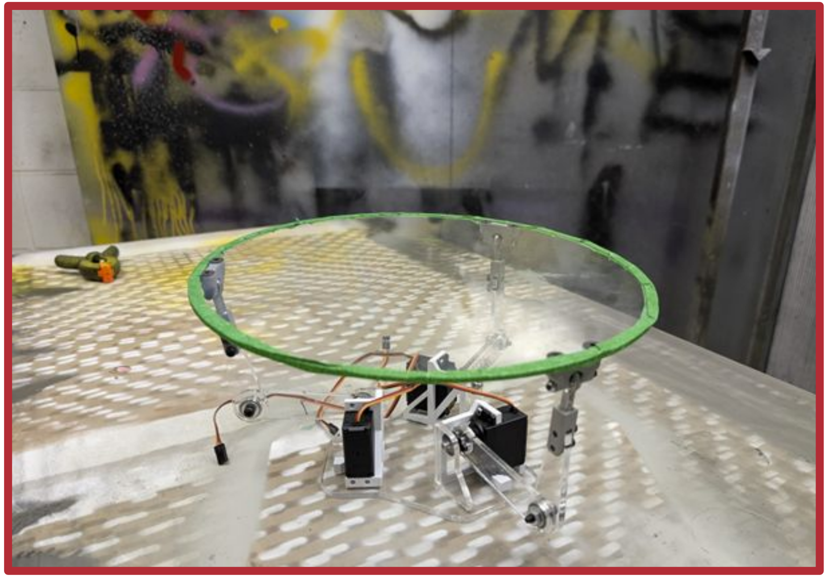
MTE 380 Stewart Platform

Group 9

The Design Problem

- Design a 3 DOF Stewart Platform that accurately controls the position of a ball
- **Goals**
  - Stable control of ball
  - Consistency in positioning
  - Gain experience in the design process
- **Challenge points**
  - Working with hardware and software simultaneously
  - Camera lag in ball tracking
  - PID control in real time



Our System Description

Existing System:

- Three MG995 Servos to control platform position
- Arduino and a PCA9685 motor shield to drive servos and control signals
- Plastic Linkages (Laser Cut Acrylic)
- Logitech C270 HD Webcam for vision

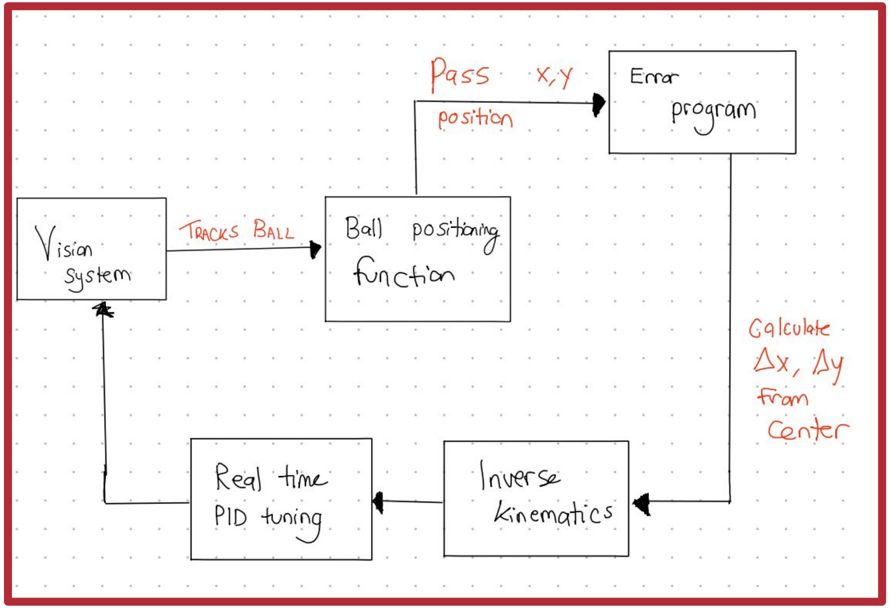
Modifications to existing mechanical system:

- Green rim tape to improve rim accuracy detection
- Increased height to get complete stewart platform in camera view

Our System Description

Software/Control System:

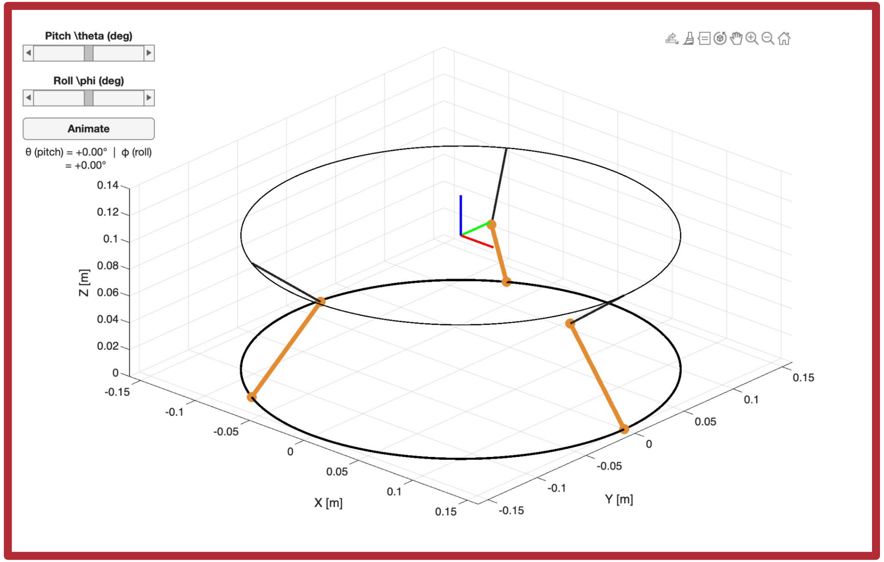
- As seen in diagram, the vision system localizes the ball
- Passes to first function which determines (x,y) position relative to center (0,0)
- Calculate error from center and roll/pitch from PID controllers in x & y
- Use that in the inverse kinematics scripts that results in each motor’s angle to minimize that error
- Tune the PID controller to ensure control of system



Inverse Kinematics

Software/Control System:

- A transform is applied to the output of the PID controllers to return a normal vector from the roll & pitch angles
- Forward kinematics as well as inverse kinematics is then applied to retrieve 3 servo angles
- Verified in simulation



Question 1

As you move the ball to the center it overshoots the destination point. How to eliminate this overshoot. Discuss with analysis, simulation, plots and implementation.

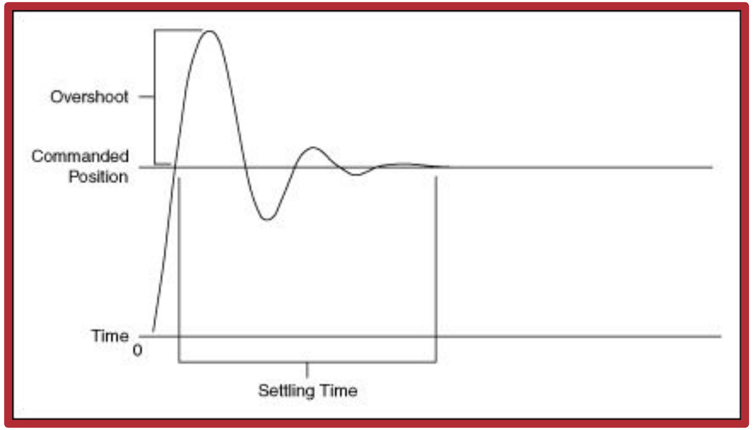
Theoretical Approach

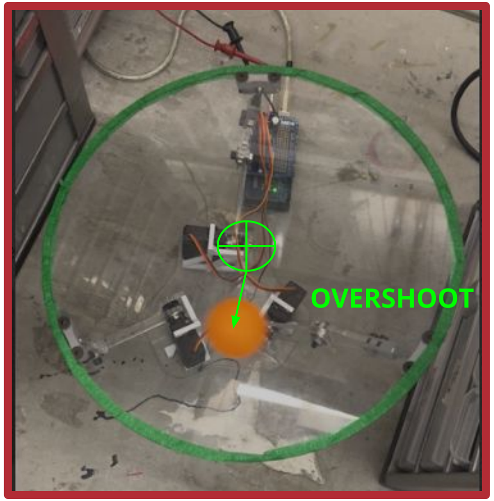
Our approach in limiting the overshoot consists:

- Separate controller for each motor
- Potential separate fine-tuning for each motor if there are intrinsic differences in each

Initial Tuning Approach:

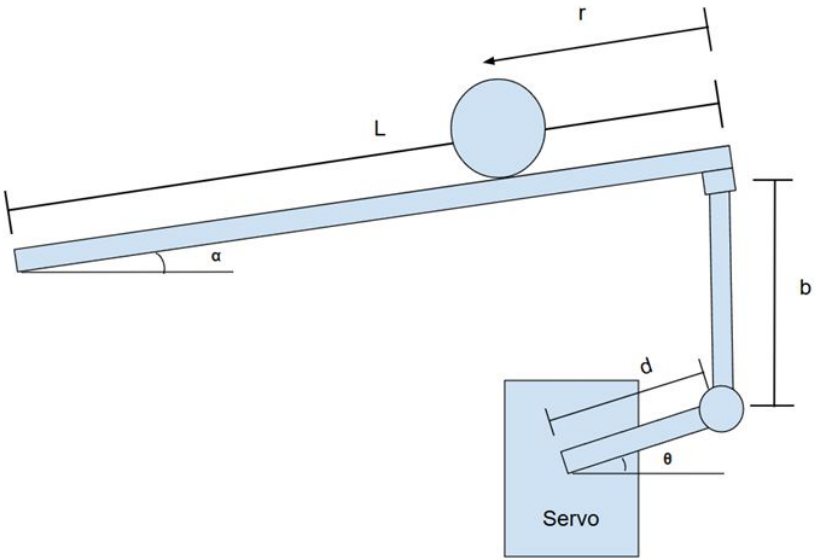
- Start with P gain, set I and D to zero
- Increase P until system oscillates
- Halve the value of P when system oscillates
- Increase derivative (D) gain to reduce overshoot until just before system jitters
- Tune I gain until system is settling
- Another approach: Ziegler-Nichols method





Deriving Transfer Function for Simulation

Ball dynamics in one axis can be modeled using motor angle as input and ball position as output.



Equation of motion transfer function

$$P(s) = \frac{R(s)}{\Theta(s)} = -\frac{mgd}{L\left(\frac{J}{R^2} + m\right)} \cdot \frac{1}{s^2}$$

PID controller transfer function

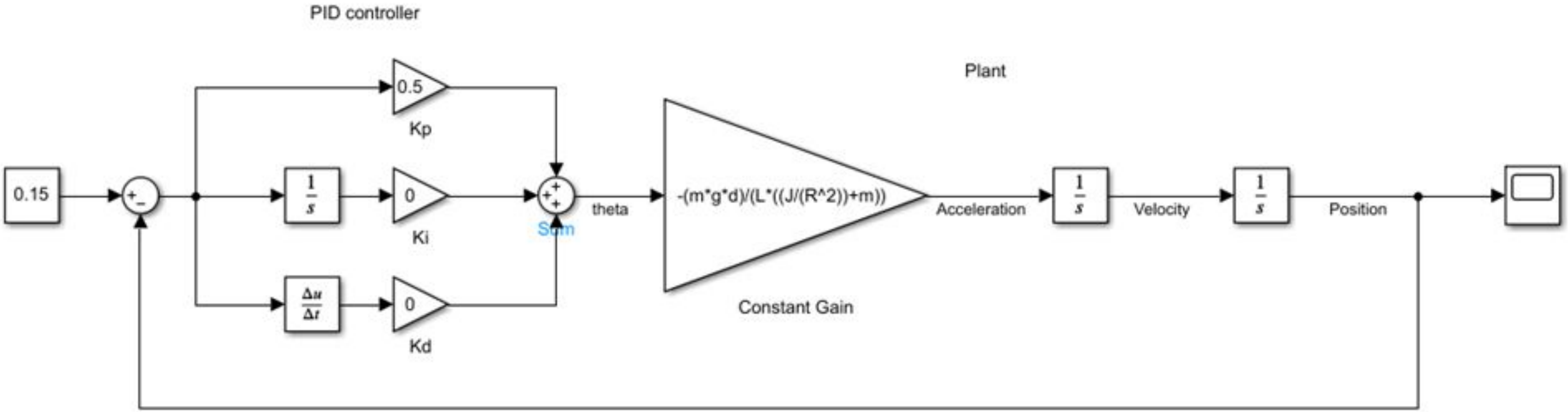
$$C(s) = K_p + \frac{K_i}{s} + K_d s$$

Equation of motion and transfer function from "Control Tutorials for Matlab & Simulink"

Simulink Block Diagram

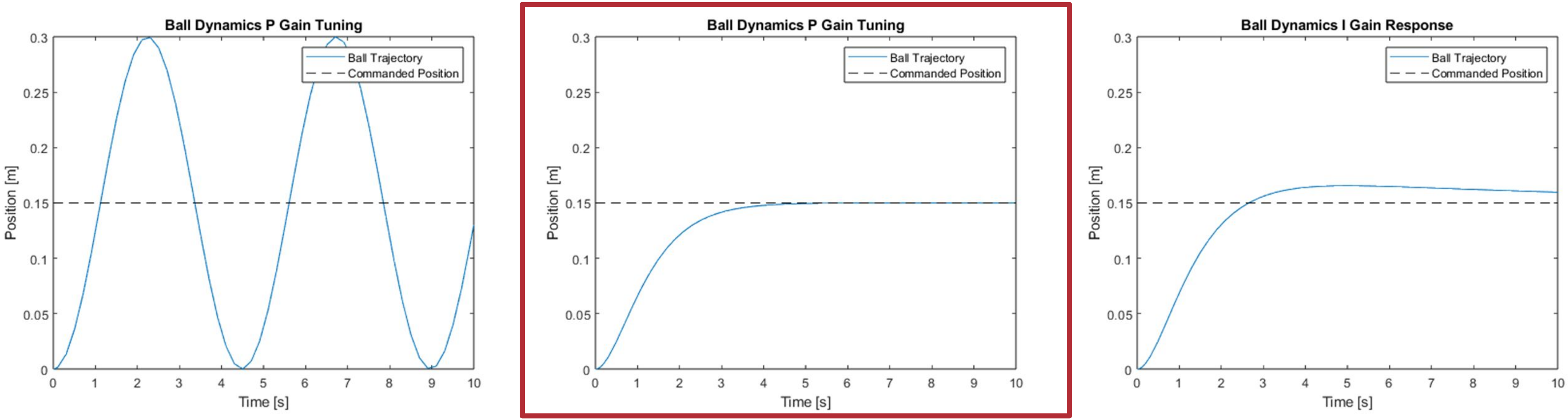
$$C(s) = K_p + \frac{K_i}{s} + K_d s$$

$$P(s) = \frac{R(s)}{\Theta(s)} = -\frac{mgd}{L\left(\frac{J}{R^2} + m\right)} \cdot \frac{1}{s^2}$$



Simulation Results

First tuning approach



Simulation Results

Ziegler-Nichols method

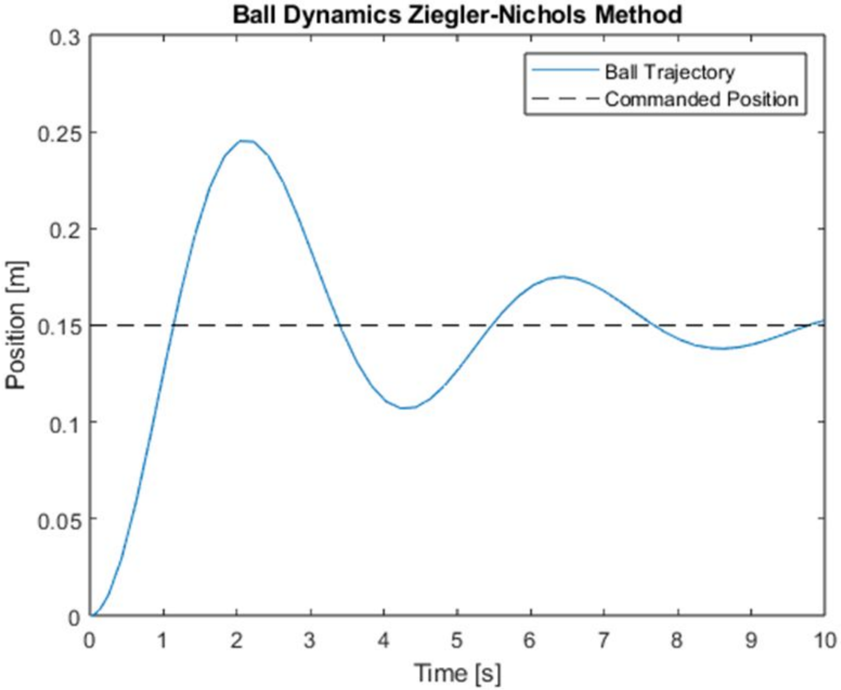
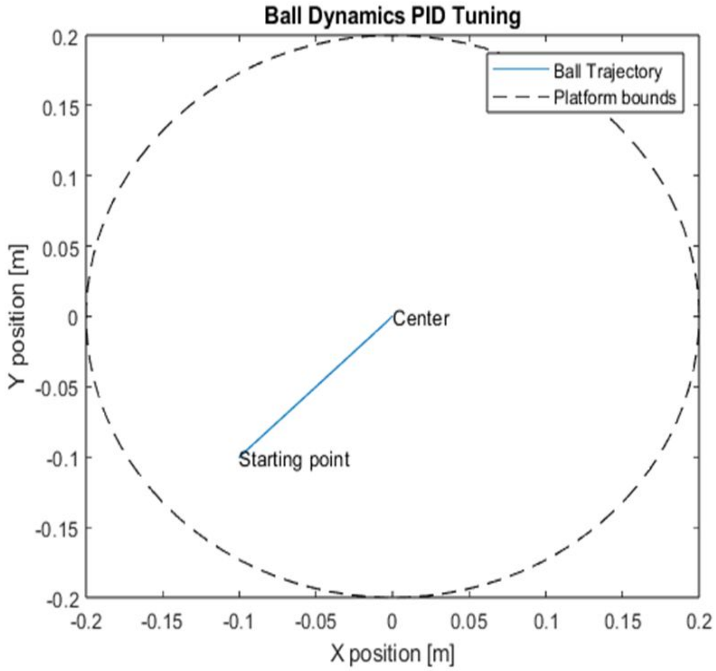


Table of values

Gain	First Approach	Ziegler-Nichols
Kp	1	1.2
Ki	0	0.4878
Kd	1.3	0.5125

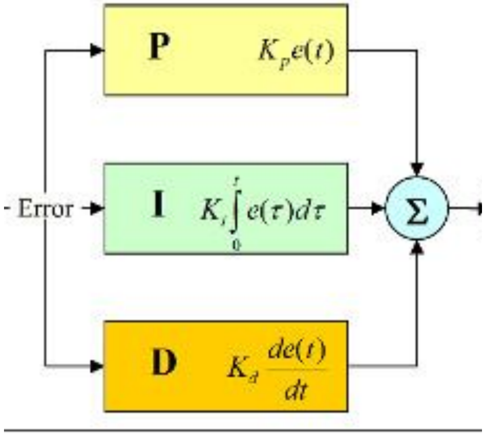
Analysis

- Ideal response when combining all axes system responses is a straight line trajectory from the initial position to the center
- This can be obtained when combining the tuned responses from first approach



Analysis - Adjustments to first tuning approach

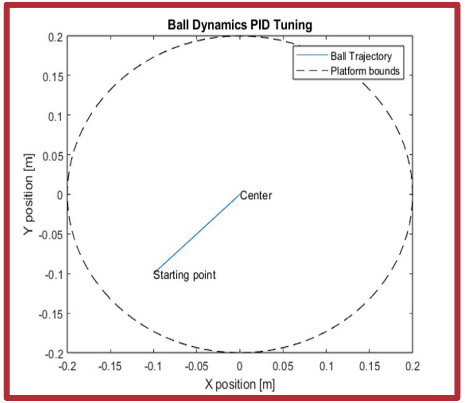
- Choose Kp → oscillates quickly
- Choose Ki → reduces steady state error, induces overshoot
- Kd must be tuned again afterwards, then iterate
- **Ziegler Nichols** method provided a quick process to get tuning parameters
  - However, it did not eliminate overshoot in simulation
  - Does get system into slightly underdamped state



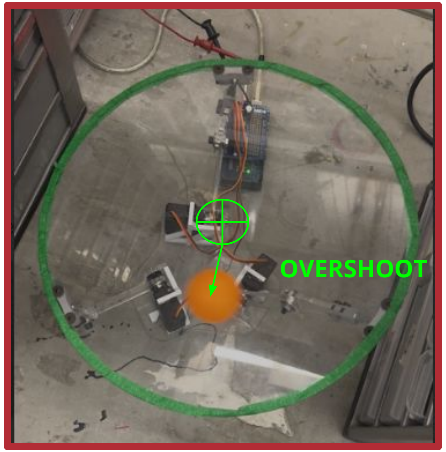
Analysis - Summary

- Tuning does reduce/eliminate overshoot, while getting the system to a stable state
- Important to Consider
  - Simulation model does not take into account friction, thus likely difference with physical model
- As such, both methods (initial tuning + Ziegler) were tested in the tuning of the physical system

SIM

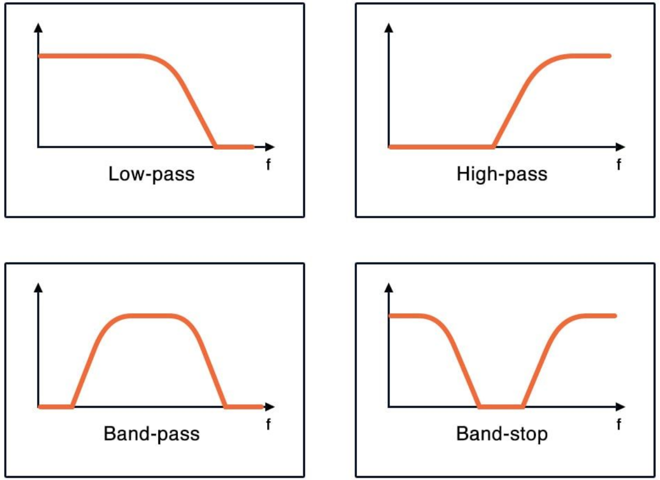


REAL



Analysis - Next Steps

- Instead of additional tuning we can recalibrate the motors, or change the material of the platform
- Strategies such as filtering to help with noisy data can help with smoother motor motion and trajectory planning



Question 2

Which type of motor is preferred for high accuracy ball positioning system. Discuss with theoretical, implementational and experimental details.

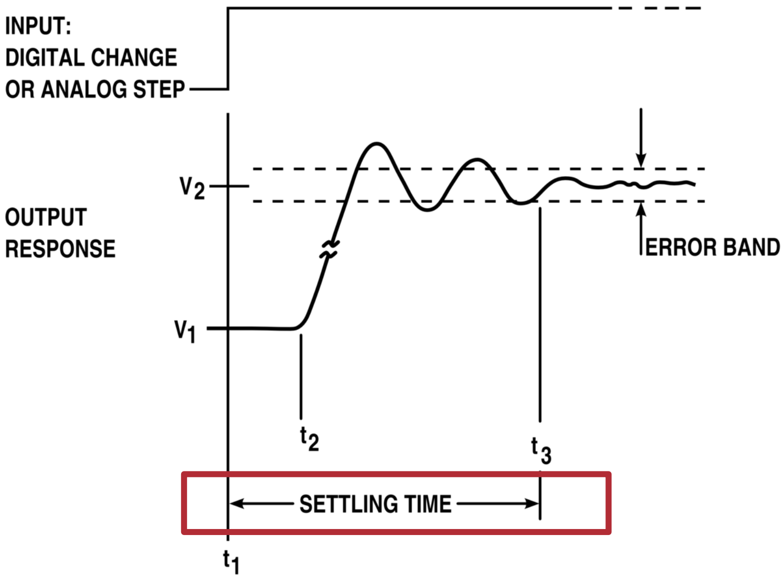
Theoretical Approach

Motors under comparison:

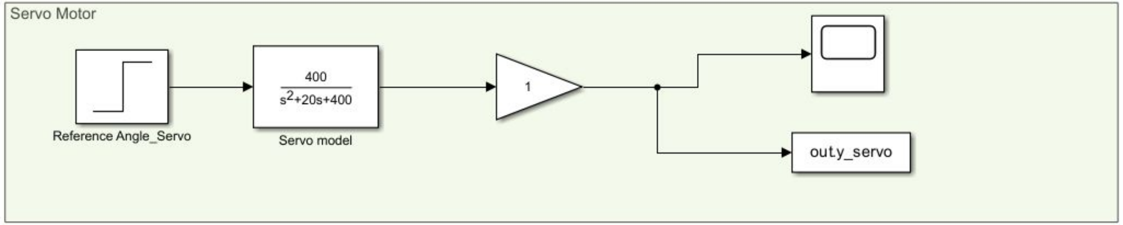
- Servo
  - MG995 Servos
- Stepper
  - NEMA 17 Stepper with TB6600 Driver
- DC Motor
  - Adafruit Generic DC Motor

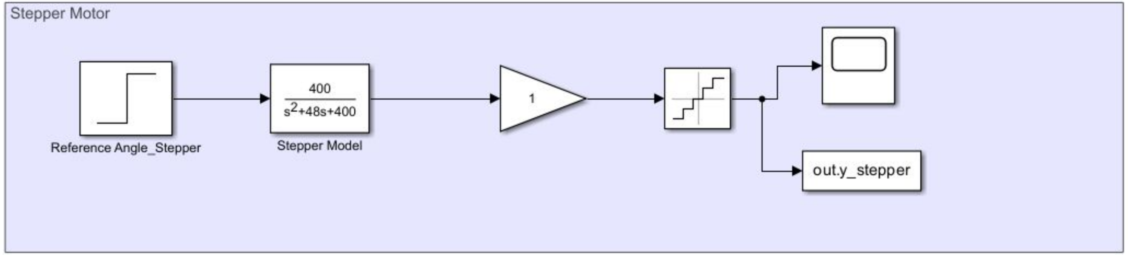
Analysis Approach

- Position accuracy (Ex: overshoot)
- Repeatability
- Smoothness of motion
- Ease of use (code focus such as drivers, libraries, wiring, etc)
- Response time (Ex: settling time)



Simulation





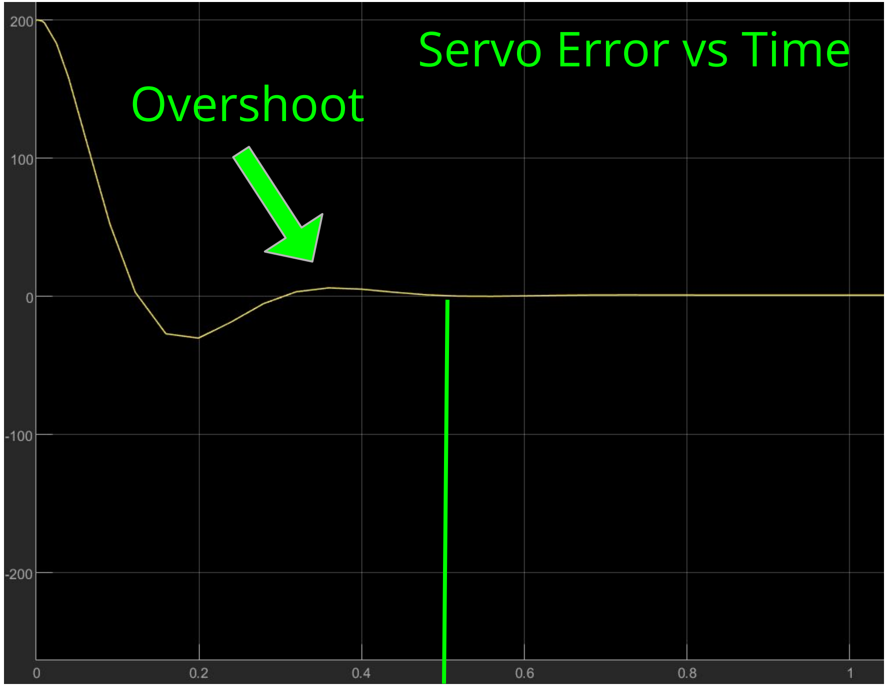
Modelled both the servo and stepper motors error, as **second order systems**, with different damping ratios to reflect typical motor characteristics

Servo: Lower damping (more overshoot, rough movement)

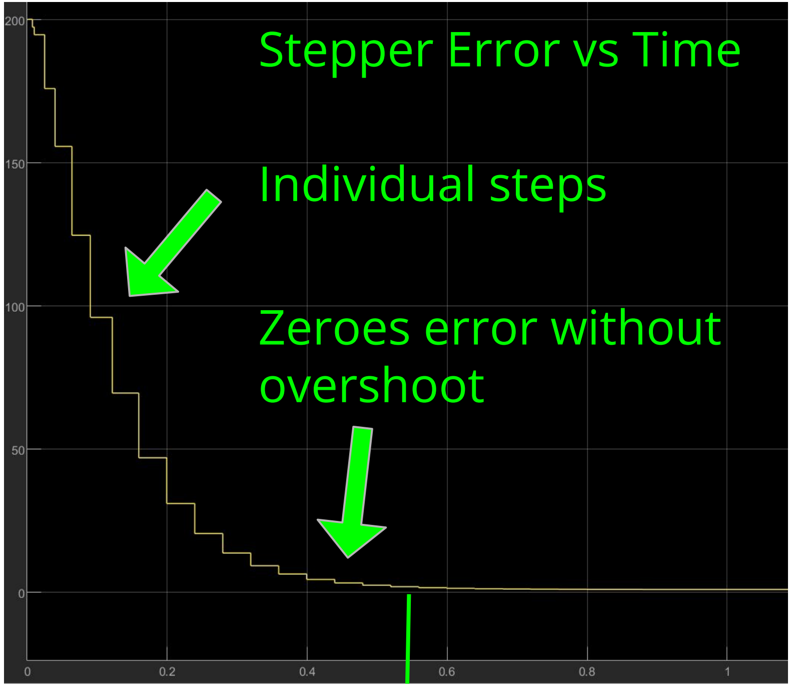
Stepper: Higher damping (less overshoot, smoother convergence)

Gave both same step input to model their behaviour

Simulation



Similar settling time



Similar settling time

Experimental Approach + Results

Evaluated Trait	MG995 Servo	Nema 17 Stepper with TB6600 Driver	DC Motor from Adafruit
Position Accuracy	2	3	0
Repeatability	1	3	0
Smoothness of Motion	2	3	1
Ease of Use	3	1	1
Response Time	2	3	1
Overall Score (/15)	10	13	3

0 = Poor Selection, 1 = Usable Option, 2 = Good Option 3= Excellent Choice

Justification

Motor Types		
Servo	Stepper	DC Motor
Ease of Use is Simple (Arduino, Motor Shield, Simple Wiring and Power Supply) Not Easily Repeatable (Needs to be Tuned and Calibrated Each Time) Great Option for Position Accuracy, Smoothness of Motion and Response Time	Easily Repeatable as the Steppers are a Linear Motion Profile, which in turn allows for increased precision as less calibration is required Steppers don't have gearbox, increased smoothness in operation Increased overhead to setup system	Not usable for a Stewart platform Lacks an encoder/feedback system to determine angle/speed Requires a Motor Shield

Questions