Position Servo Control

Model-Reference Adaptive System (MRAS) Design and Implementation

Setup:

DC motor dynamics:

$$\frac{dv}{dt} = -\underbrace{a}v + \underbrace{b}u$$

Unknown motor constants

- Input is terminal voltage
- Output is angular position

Control Law:

$$\mathrm{u} = \theta_1(u_c - y) - \theta_2 v$$

- θ_1 / θ_2 are the control parameters
- u_c is the reference input

Model system:

$$G_m(s) = \frac{\omega^2}{s^2 + 2\zeta\omega s + \omega^2}$$

- Standard 2nd order model
- ζ and ω define the intended system response

Choose parameters?
NO - adaptively update them based on error $(y - y_m)$!

Loss function:

$$J(\theta) = \frac{1}{2}e^2$$

MIT rule:

$$\frac{d\theta}{dt} = -\gamma \frac{\partial J}{\partial \theta}$$

Parameter updates:

$$\frac{d\theta}{dt} = -\gamma \frac{\partial e}{\partial \theta} e$$

Action:

Implement control law:

$$\frac{y}{u_c} = \frac{b\theta_1}{p^2 + (a + b\theta_2)p + b\theta_1}$$

Equate coefficients:

$$\omega^2 = b\theta_1,\ 2\zeta\omega = a + b\theta_2$$

Sensitivity partials:

$$\frac{\partial e}{\partial \theta_1}, \frac{\partial e}{\partial \theta_1}\ where\ e = y - y_m$$

$$\left\{\begin{aligned}
&\frac{\partial e}{\partial \theta_1} = \frac{b(p^2 + (a + b\theta_2)p + b\theta_1) - b^2\theta_1}{(p^2 + (a + b\theta_2)p + b\theta_1)^2} u_c \\
&\frac{d\theta_1}{dt} = -\gamma' \frac{\omega^2}{p^2 + 2\zeta\omega p + \omega^2}(u_c - y)e \\
&\frac{\partial e}{\partial \theta_2} = \frac{-b^2\theta_1 p}{(p^2 + (a + b\theta_2)p + b\theta_1)^2} u_c \\
&\frac{d\theta_2}{dt} = -\gamma' \frac{-\omega^2 p}{p^2 + 2\zeta\omega p + \omega^2} y e
\end{aligned}\right.$$

Software Implementation:

Parameters:

$\omega = 5, \zeta = 0.5, \gamma' = 2$

Constants:

$a = 1.234, b = 5.678$

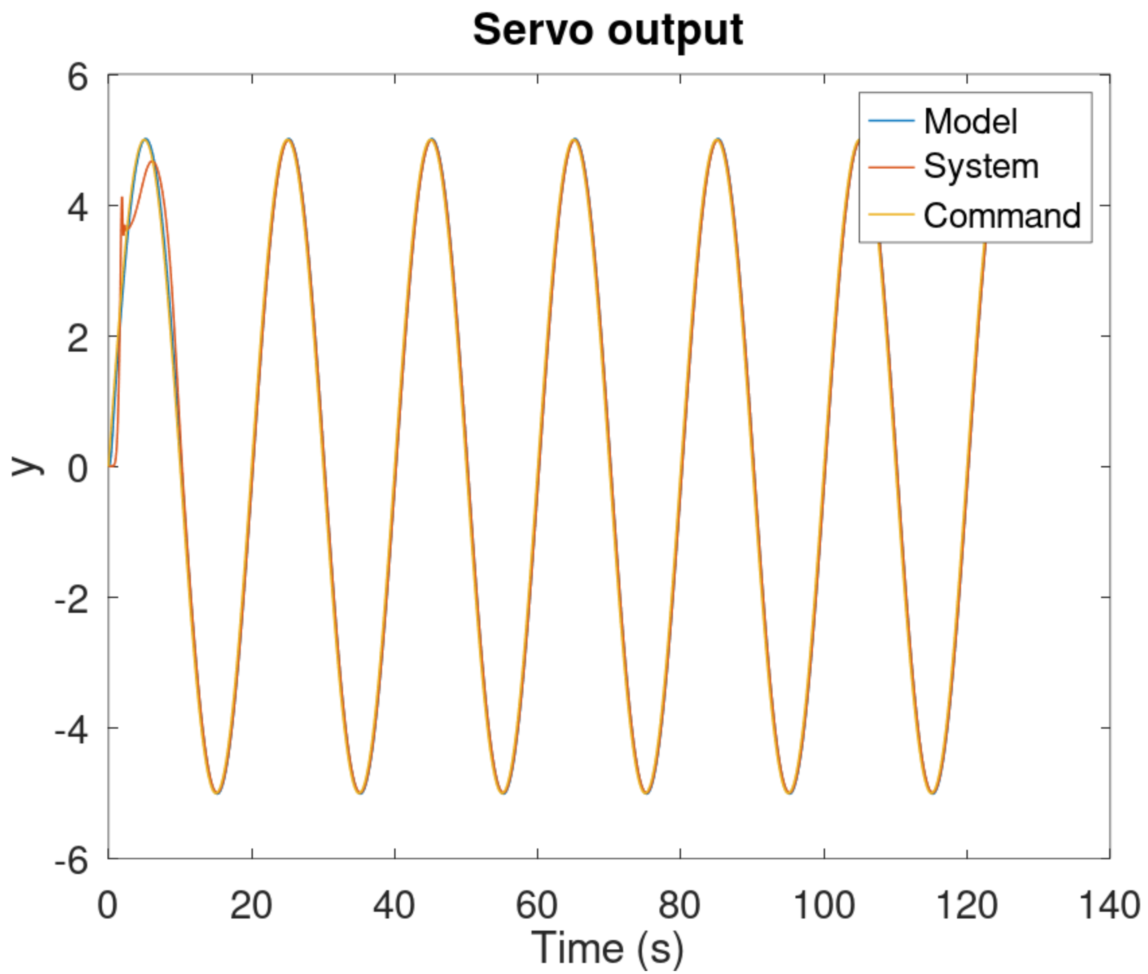
Input:

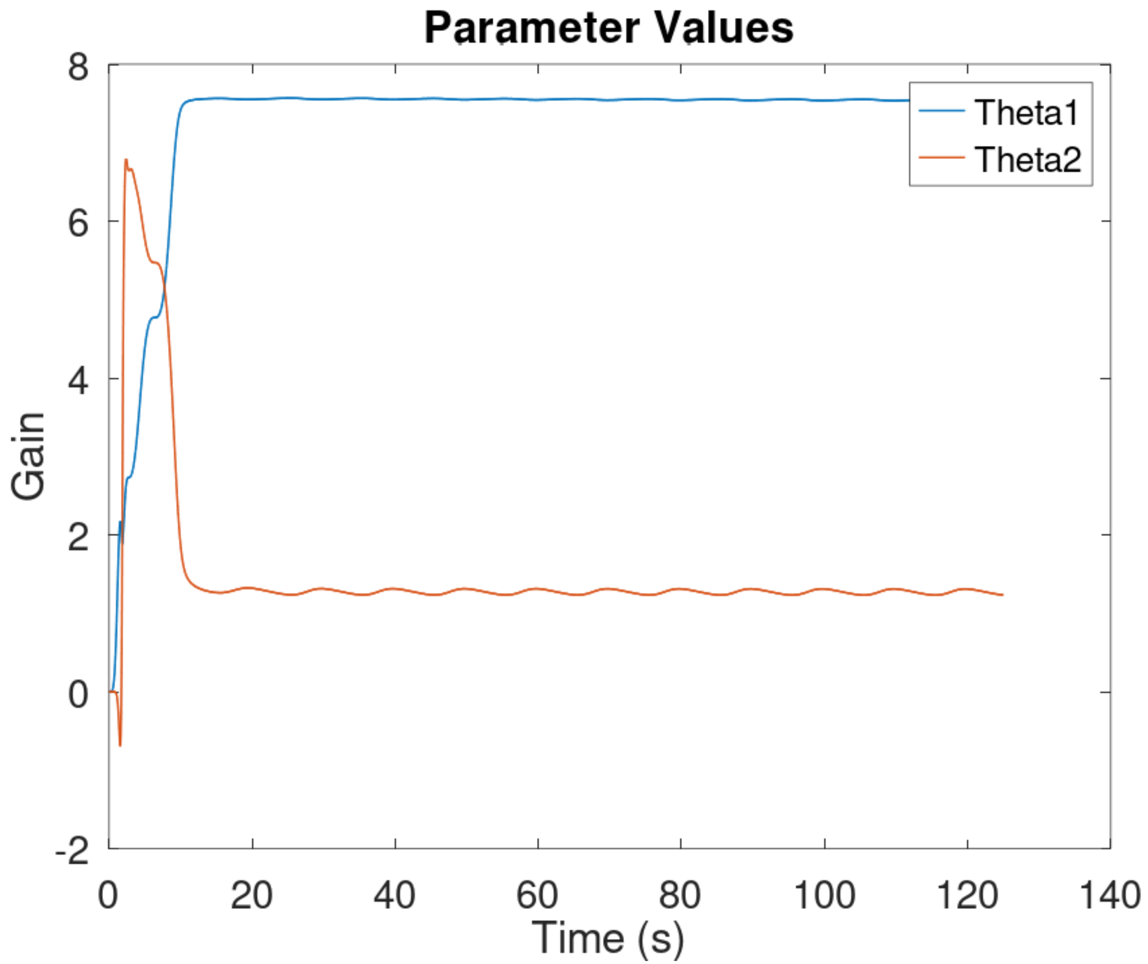
$u = 5\sin({}^{2\pi}/_{20}\, t)$

States:

```
% model reference: x(1) = ym | x(2) = vm | input = uc
px([1;2]) = Model.a*x([1;2]) + Model.b*uc;

% plant: x(3) = y | x(4) = v | input = u
px([3;4]) = Plant.a*x([3;4]) + Plant.b*u;

% sensitivity partial for theta1 (spt1): x(5) = spt1 | x(6) = pspt1 | input = uc - y
px([5;6]) = Spt1.a*x([5;6]) + Spt1.b*(uc-x(3));

% sensitivity partial for theta2 (spt2): x(7) = spt2 | x(8) = pspt1 | input = vfiltered
px([7;8]) = Spt2.a*x([7;8]) + Spt2.b*x(11);

% theta updates with MIT: x(8) = theta1 | x(9) = theta2
px(9) = -gamma*x(5)*e;
px(10) = -gamma*x(7)*e;

% velocity low-pass filter (vlpf): x(11) = vfiltered | input = v
px(11) = Vlpf.a*x(11)+Vlpf.b*x(4);
```

Simulation:


Servo output
y
Time (s)
Model
System
Command
6
4
2
0
-2
-4
-6
0
20
40
60
80
100
120
140


Parameter Values
Gain
Time (s)
Theta1
Theta2
8
6
4
2
0
-2
0
20
40
60
80
100
120
140

Hardware Implementation:

Setup:

HW:
- MSP430FR2355
- 12V 150RPM geared motor
- 28 PPR quadrature encoder (pre-gearbox)
- LM298N H-Bridge motor driver
- POT for command

SW:
- Hardware interrupts for encoder
- Hardware PWM for motor driver
- UART for data transmission
- Zero-order hold discretization at 16 Hz
- Low-pass filter on velocity
- Floats used

Tests:

Sinusoidal input:
- 0-270° at 100 mHz
- Unstable at higher frequencies

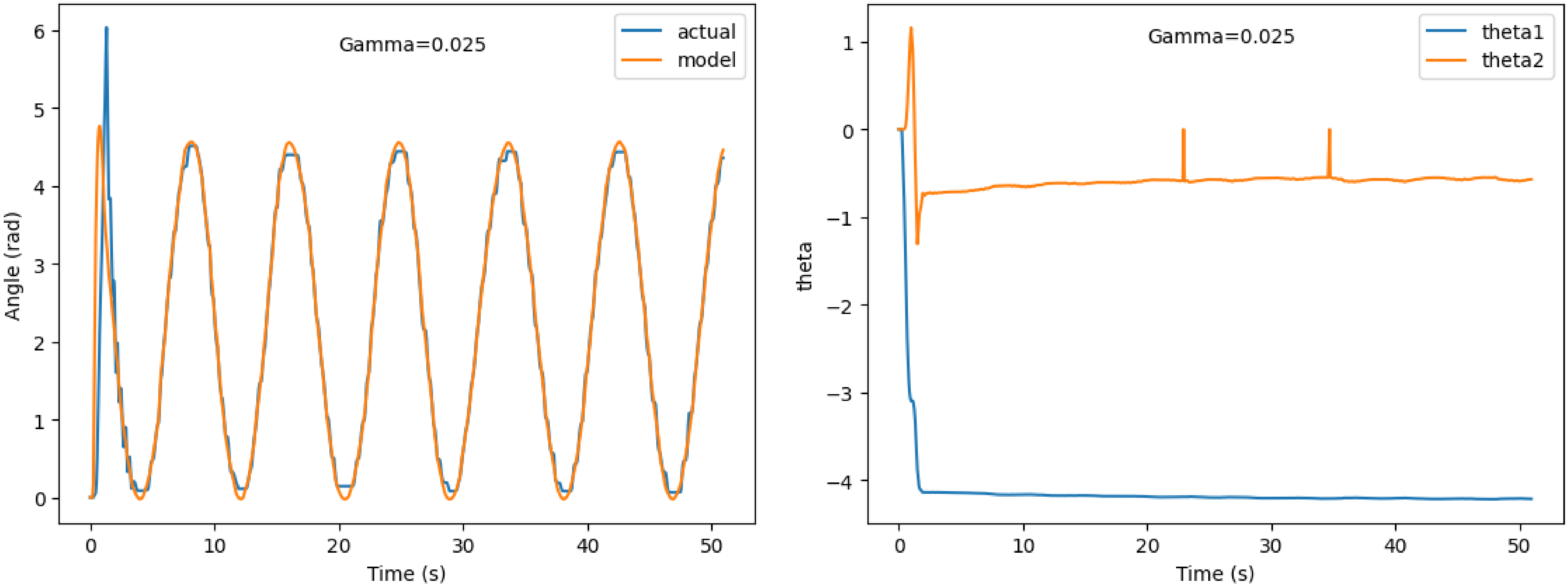
Command following:
- Turn POT to hearts content
- Unstable for very quick adjustments

Results:

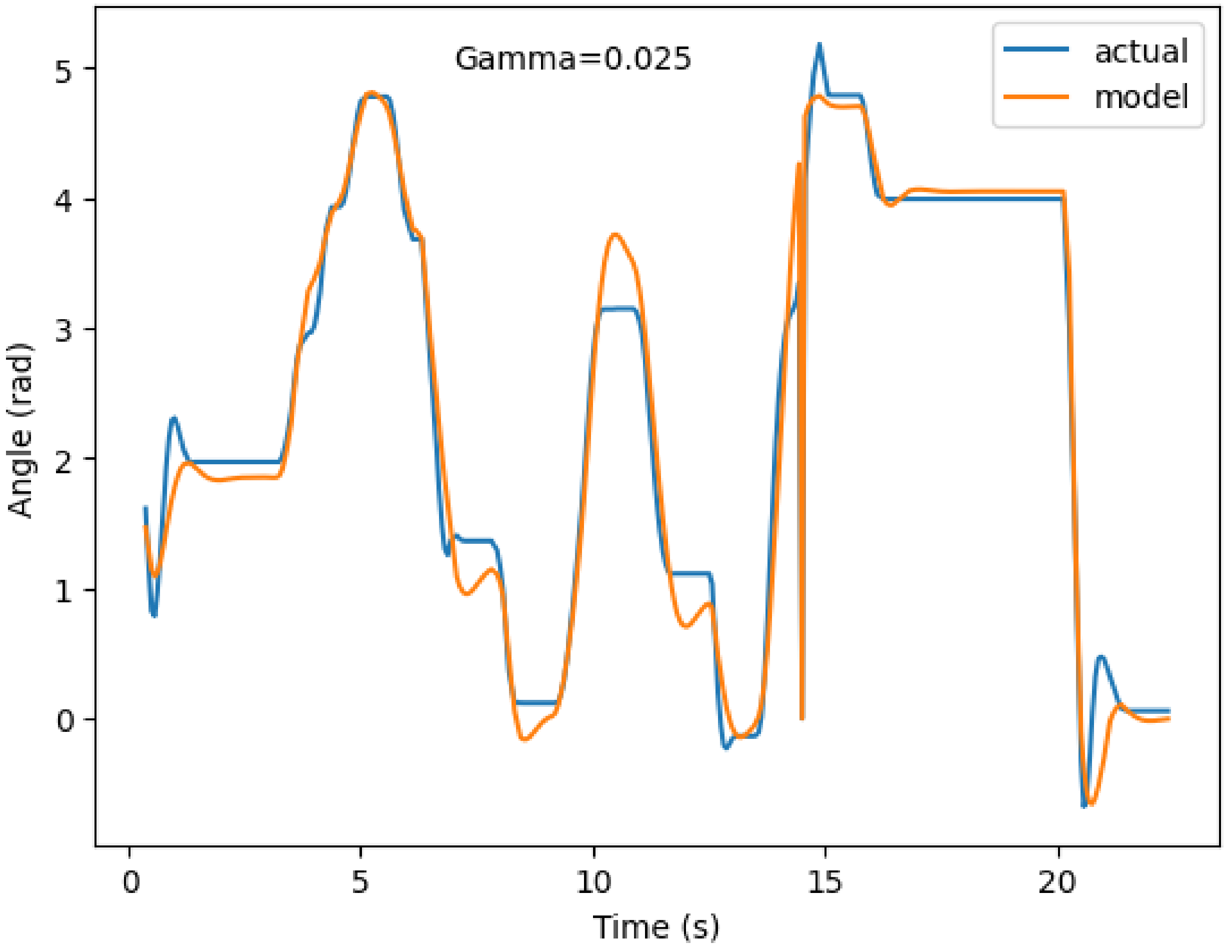
Sinusoidal input:
- Pretty cool!
- Backlash/stall torque?

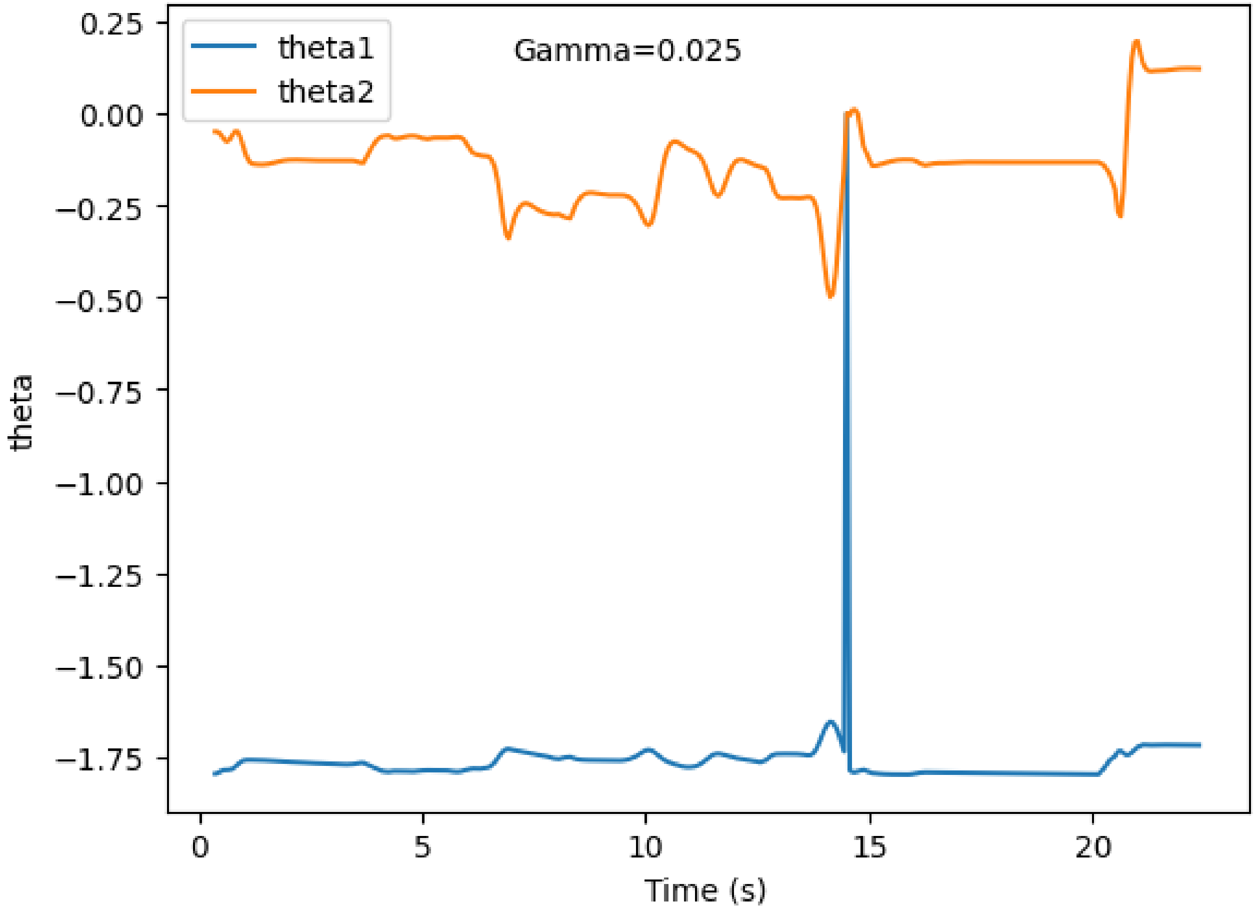
Command following:
- Acceptable.
- Again, backlash/stall torque?

Sinusoidal Results:



Command Following Results:





MONTANA STATE UNIVERSITY
NORM ASBJORNSON College of ENGINEERING