MEAM 520
Lecture 24: Control

Cynthia Sung, Ph.D.

Mechanical Engineering & Applied Mechanics

University of Pennsylvania

Final Projects

Final Project

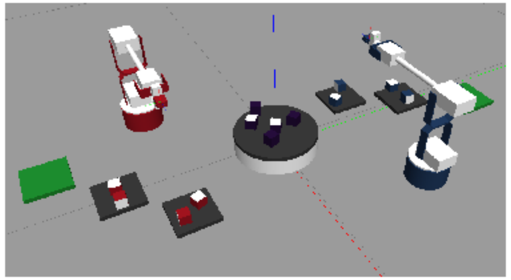
MEAM 520, University of Pennsylvania

November 13, 2020

Teams will use the concepts learned during the semester to control their simulated Lynx robot in a head-to-head competition with their opponents' robot. The robots will manipulate objects in the simulated environment to score points, culminating in a class-wide tournament!

Instructions: Just as in labs, this final project is an opportunity for you to explore the concepts we learned in class in a more complicated environment. Expand on previous labs, pull techniques from the literature, or try some experimentation of your own. You should document your approach through a report similar to the reports you have written throughout the semester.

The final project is worth 70 pts. Bonus points will be awarded to teams who perform particularly well during the tournament: 5 pts to 1st place, 3 pts to 2nd place, 1 pt to 3rd place.



1 Competition Rules

1.1 Ground Rules

1. **Students are required to work in teams of four.** If you would like, you may also be randomly matched with other students by the teaching staff **Regardless, you must fill out the form on Piazza to either register your team or ask to be matched by November 20 @ 12 noon.** Any student who does not register their team will be automatically assigned to a team. A few students will likely end up in teams of 3 but this will be sorted out by the teaching staff.
2. **Teams will submit their code through Gradescope before the competition.** During the competition, TA's will run the game on a physical Ubuntu machine (not the provided Virtual Machine) while streaming the simulation live on Zoom.

1

poll @483

24 views

Actions

Final Project Timing

We've gotten some feedback that the timing of the final project is a bit tight. We'd like to get your thoughts on the best way to schedule the final tournament. Part of the squeeze is due to 12/10 (a Thursday) being a Monday schedule due to the strange academic calendar this semester, so we do not officially have class that day.

Which option do you prefer?

[Option A] *Round Robin:* 12/3 in class, *Top 8 Final Round:* 12/8 in class (as written in handout)
[Option B] *Round Robin:* 12/8 in class, *Top 8 Final Round:* 12/10 12-1:30 (what would have been class time)
[Option C] *Round Robin:* 12/8 in class, *Top 8 Final Round:* **some time during reading days**

Please answer this poll within the next two days to have your input considered! Thanks!

- Option A (stick to the plan)
- Option B
- Option C

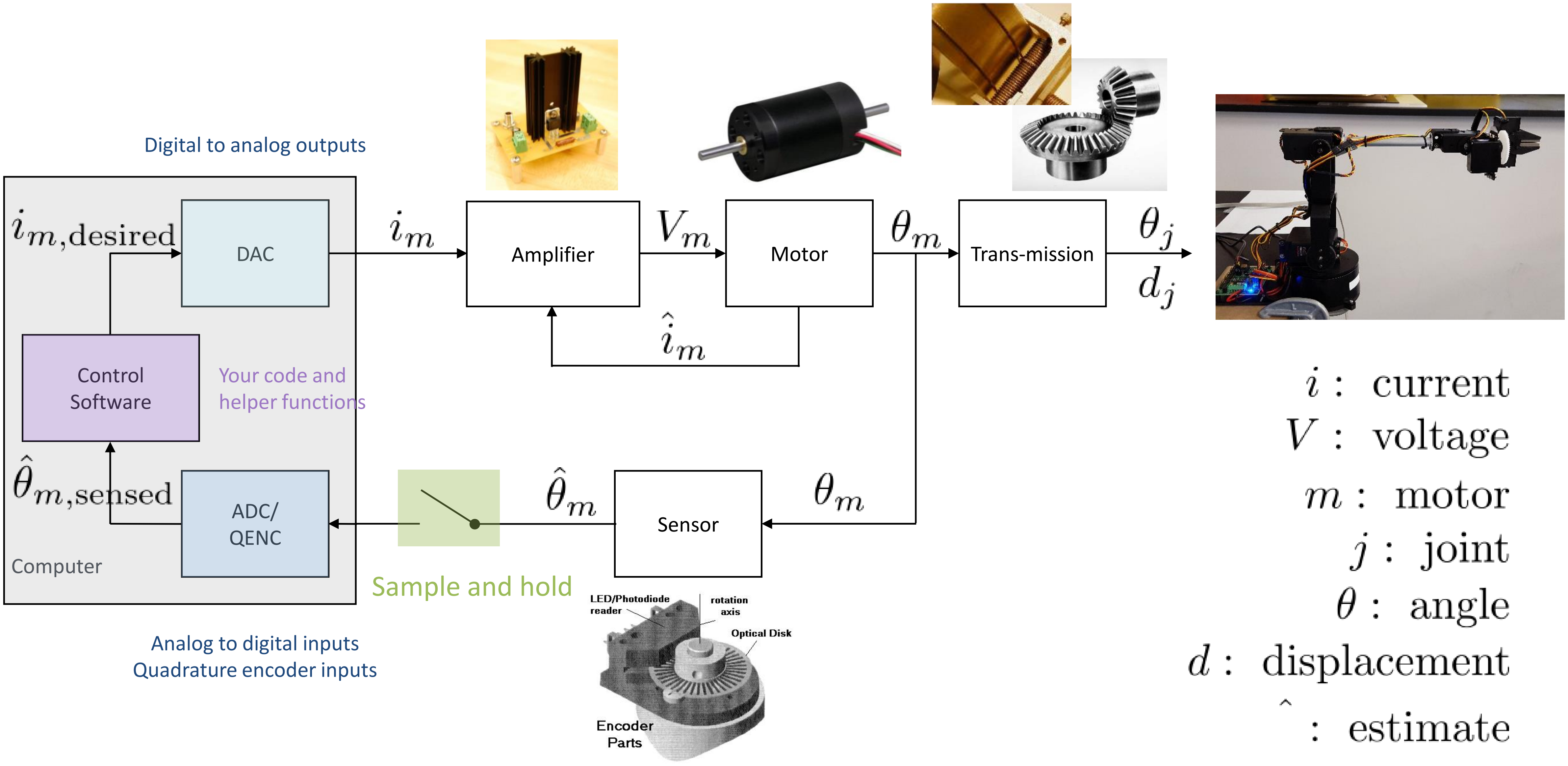
Submit

11/16	Actuation and Control *Read: AKKK 7.2-7.3, SHV 6.intro-6.3*	PID Control *Read: SHV 6.3*	Lab 5 due 11/20
11/23	Sensing and State Estimation *Read: AKKK 8.1-8.3*	Thanksgiving – No class	
11/30	Multi-agent Planning *Paper reading*	Final project: Round Robin	Mid-project update due 12/2
12/	Final project: Bracket		Final project due 12/10 (no penalty deadline: 12/14)

The purpose of the tournament is to help you in your evaluation. Identify strengths of limitations throughout the course of the tournament. Feel free to edit your code between this date and the final submission.

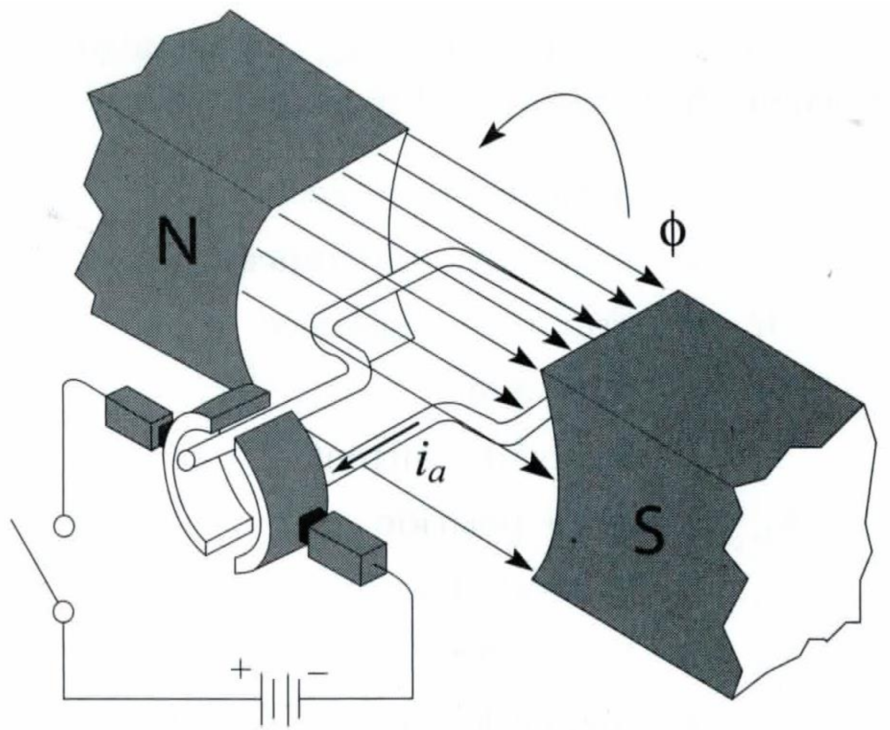
Register your team at:
https://forms.gle/wnpzXc44BbVyg1Dt9

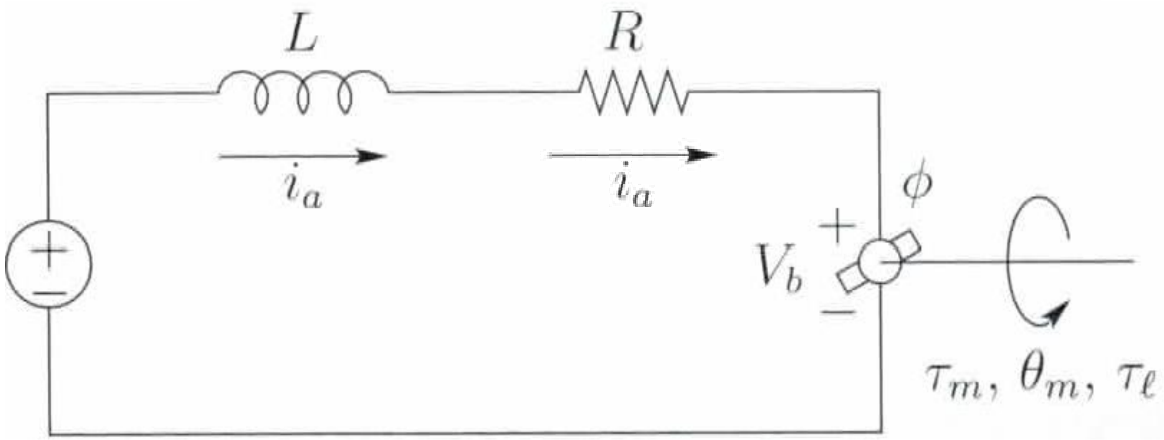
Previously: How most real robots work



(SHV Section 6.1)

Previously: DC Motor





$$\tau_m = K_1\,\phi\,i_a = k_t\,i_a$$

- τ_m: generated torque (N•m)
- K_1: physical constant
- ϕ: magnetic flux (webers)
- i_a: armature current (A)
- k_t: torque constant (N•m/A)
- i_a: armature current (A)

$$k_t = k_v$$

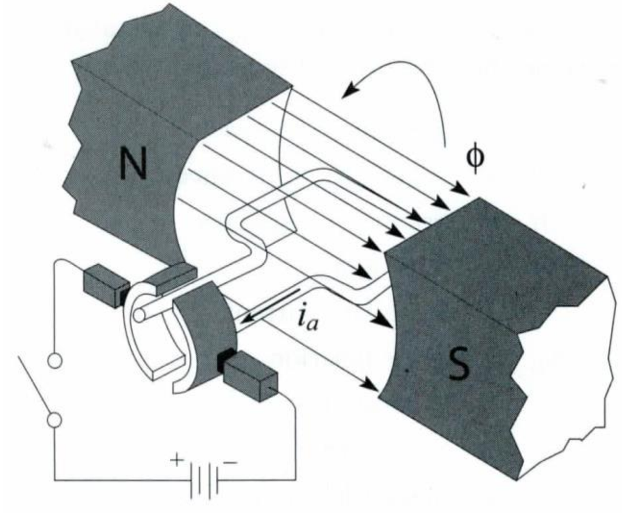
if using meters, kilograms and seconds

$$V_b = K_2\,\phi\,\omega_m = k_v\,\omega_m$$

- V_b: back emf (V)
- K_2: physical constant
- ϕ: magnetic flux (webers)
- ω_m: motor velocity (rad/s)
- k_v: back-emf constant (V•s)
- ω_m: motor velocity (rad/s)

(SHV Section 6.1)

Previously: DC Motor



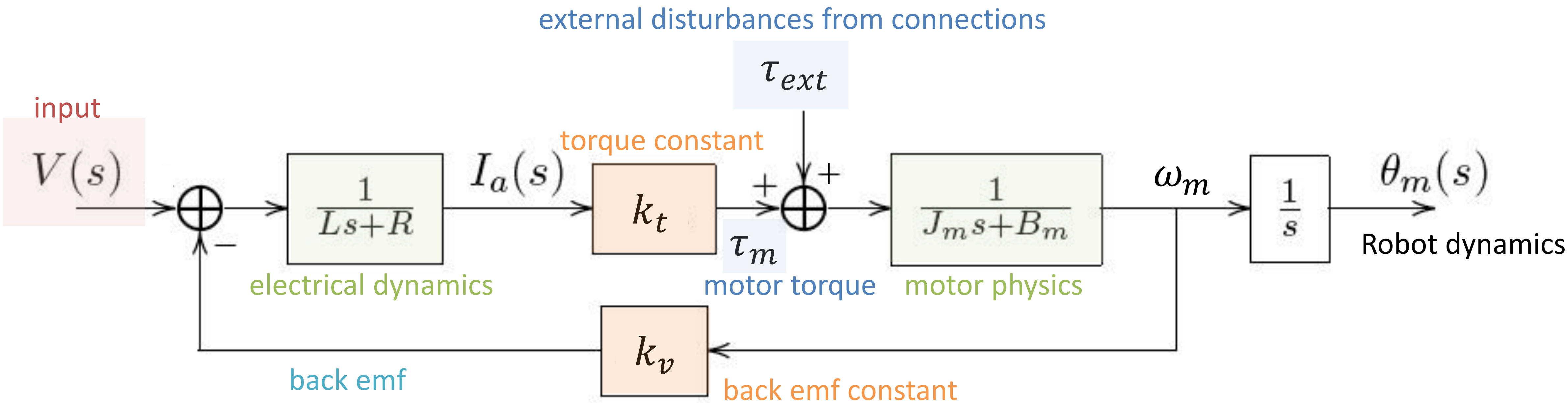
Electrical Dynamics

$$V(t) = L\frac{di_a}{dt} + Ri_a + k_v\frac{d\theta_m}{dt}$$

Physical Dynamics

SHV shows the load torque in the wrong direction and confusingly calls gear ratio “r”

$$J_m\frac{d^2\theta_m}{dt^2} + B_m\frac{d\theta_m}{dt} = \tau_m + \tau_{ext} = k_t i_a + \tau_{ext}$$



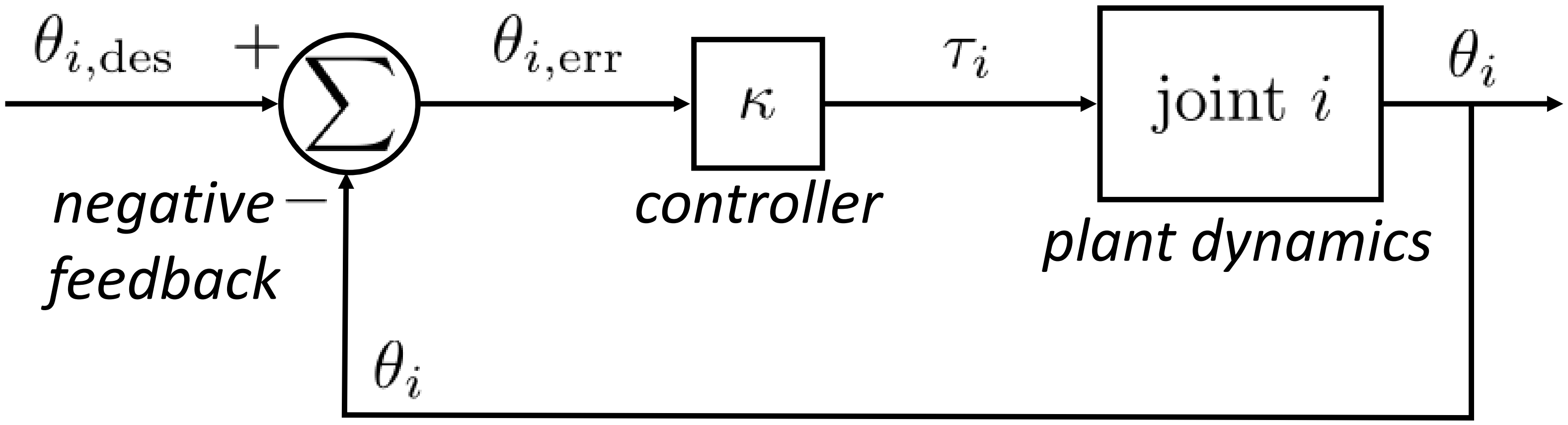
Desired Joint Angles

$\theta_{1,\text{des}}, \theta_{2,\text{des}}, \theta_{3,\text{des}}$ ...

Actual Joint Angles

$\theta_1, \theta_2, \theta_3$ ...

Proportional Feedback Controller



Desired Joint Angles

$\theta_{1,\mathrm{des}}, \theta_{2,\mathrm{des}}, \theta_{3,\mathrm{des}}$ ...

Actual Joint Angles

$\theta_1, \theta_2, \theta_3$ ...

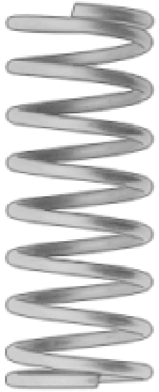
Proportional Feedback Controller

joint torques

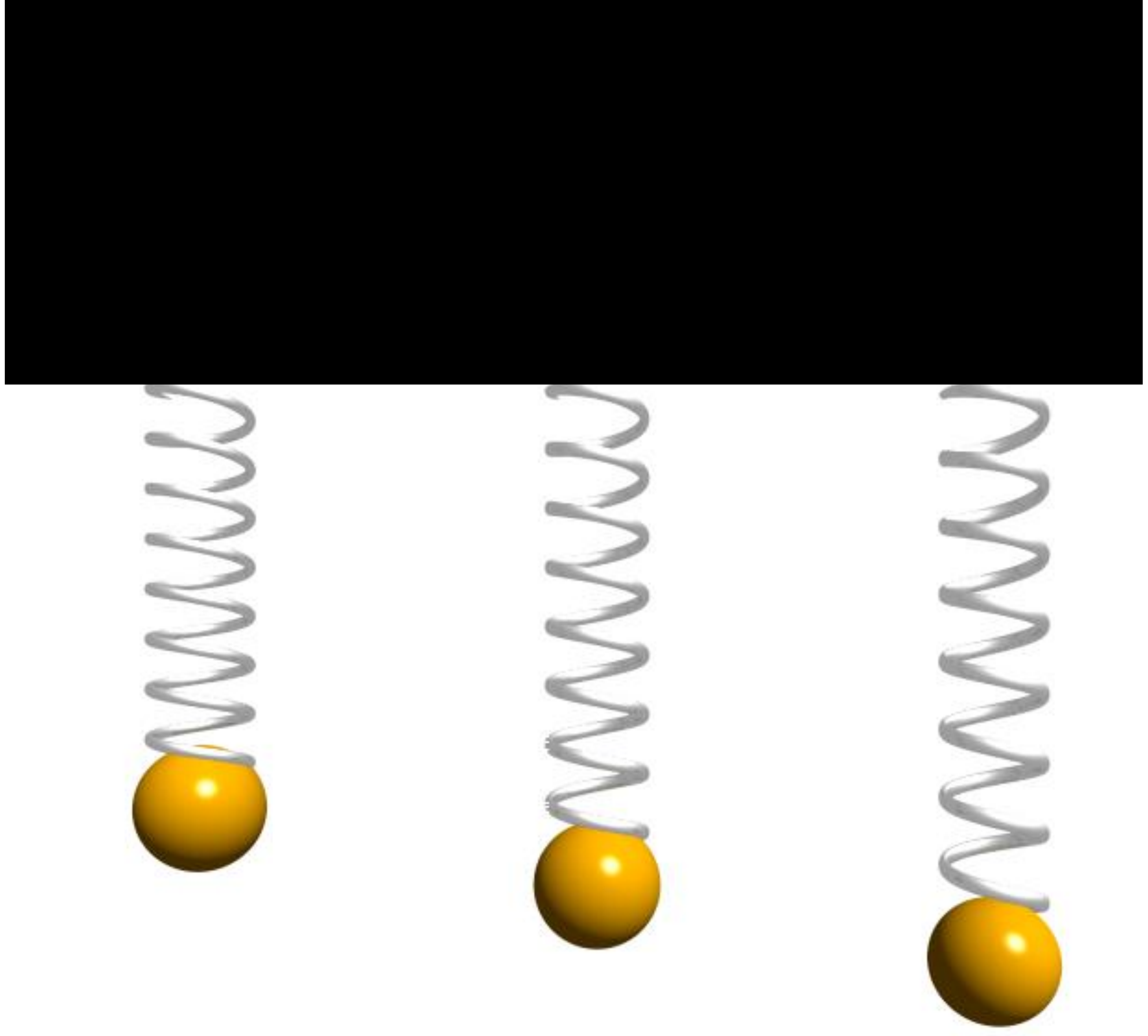
$$\tau_1 = \kappa(\theta_{1,\mathrm{des}} - \theta_1)$$
$$\tau_2 = \kappa(\theta_{2,\mathrm{des}} - \theta_2)$$
$$\tau_3 = \kappa(\theta_{3,\mathrm{des}} - \theta_3)$$

joint angle errors

proportional gain in Nm / rad



Proportional feedback acts like a torsional spring with linear stiffness, pulling each joint to the desired angle.



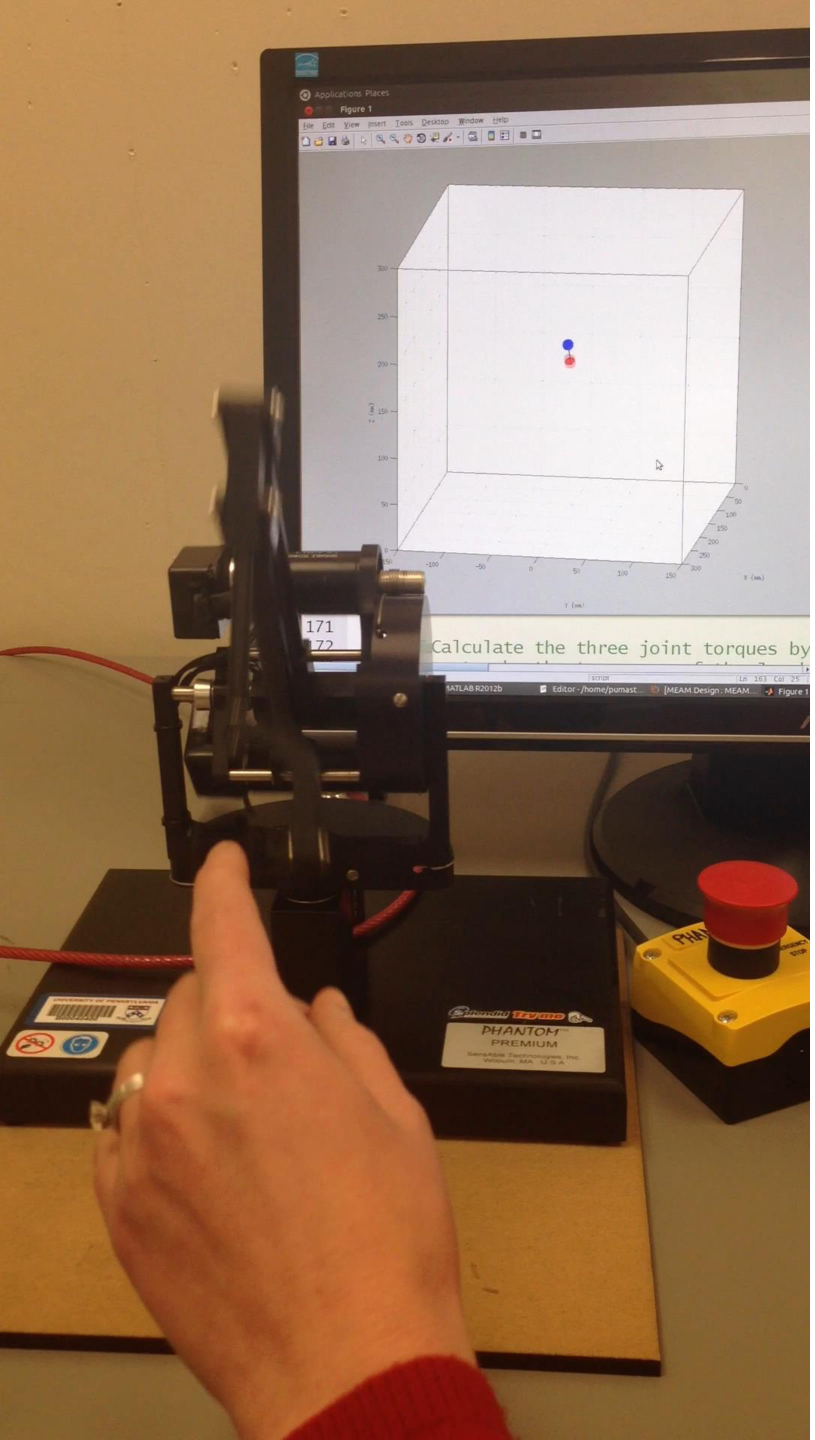
Mass on a spring: simple harmonic oscillator

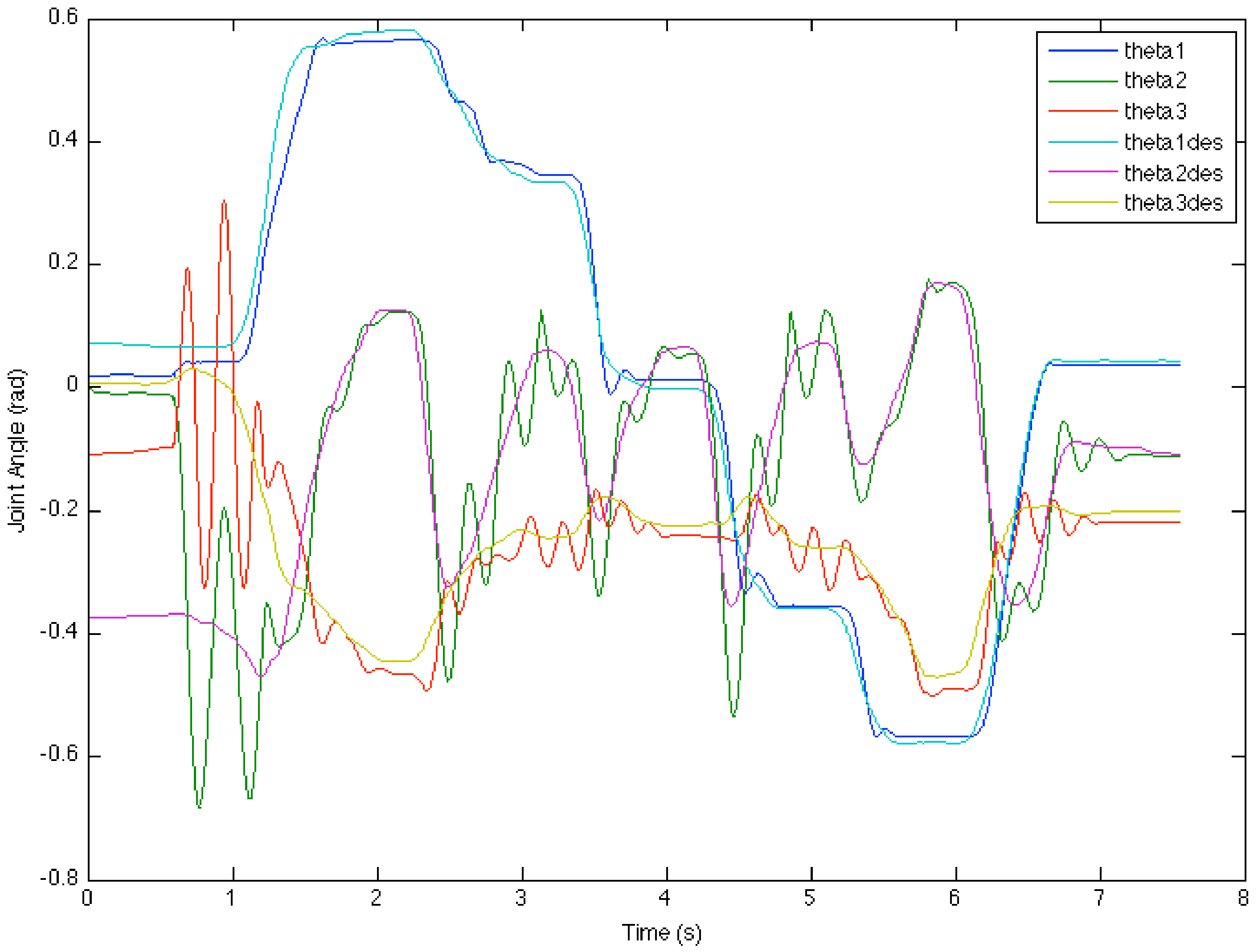
$$\tau_i = \kappa\left(\theta_{i,des} - \theta_i\right)$$

$$f_1(q)\ddot{q} + f_2(q,\dot{q}) = \kappa\left(\theta_{i,des} - \theta_i\right)$$

$$f_1(q)\ddot{q} = \left[\kappa\left(\theta_{i,des} - \theta_i\right) - f_2(q,\dot{q})\right]$$

Image from
http://lifshitz.ucdavis.edu/~dmartin/phy7/7C.html





It's pretty oscillatory.

How can we improve the controller's tracking?

Add derivative feedback – virtual damping.

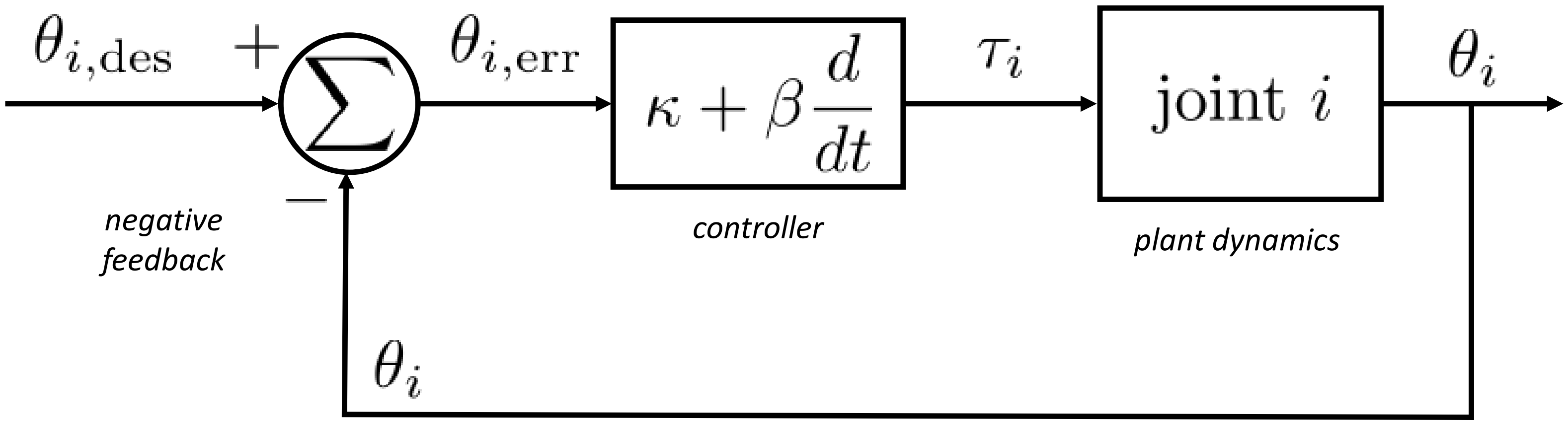
Desired Joint Angles

$\theta_{1,\mathrm{des}}, \theta_{2,\mathrm{des}}, \theta_{3,\mathrm{des}}$ ...

Actual Joint Angles

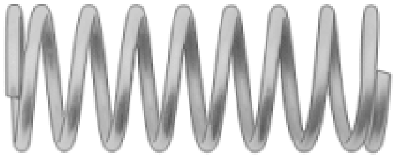
$\theta_1, \theta_2, \theta_3$ ...

Proportional Derivative Feedback Controller

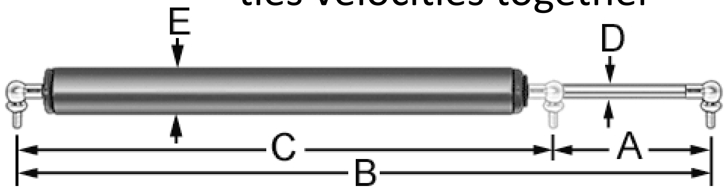


Add a derivative term to our position feedback controller, making it a Proportional Derivative (PD) controller.

virtual spring,
ties positions together



virtual damper,
ties velocities together



$$\tau_1 = \kappa(\theta_{1,\text{des}} - \theta_1) + \beta(\omega_{1,\text{des}} - \omega_1)$$

$$\tau_2 = \kappa(\theta_{2,\text{des}} - \theta_2) + \beta(\omega_{2,\text{des}} - \omega_2)$$

$$\tau_3 = \kappa(\theta_{3,\text{des}} - \theta_3) + \beta(\omega_{3,\text{des}} - \omega_3)$$

proportional gain in Nm / rad

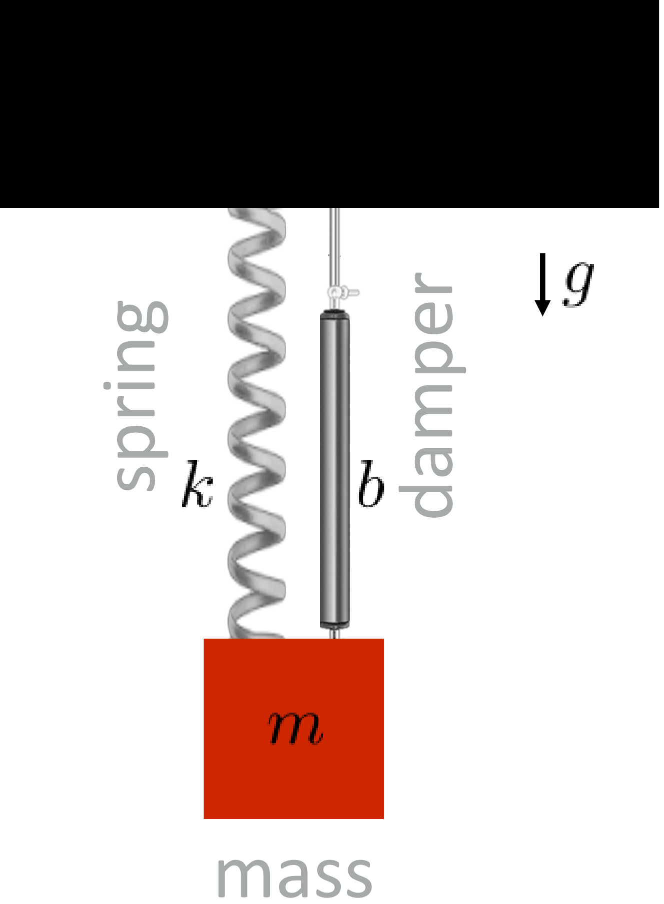
derivative gain in Nm / (rad/s)

The gains are typically tuned separately for each joint.

$$\theta_{i,\text{err}} = \theta_{i,\text{des}} - \theta_i$$

$$\dot{\theta}_{i,\text{err}} = \omega_{i,\text{des}} - \omega_i$$

$$\tau_i = \kappa\, \theta_{i,\text{err}} + \beta\, \dot{\theta}_{i,\text{err}}$$



$$\Sigma F_y = m\ddot{y}$$

$$-mg - ky - b\dot{y} = m\ddot{y}$$

$$-mg = m\ddot{y} + b\dot{y} + ky$$

$$-g = \ddot{y} + \frac{b}{m}\dot{y} + \frac{k}{m}y$$

Second-order system

$$\frac{k}{m} = \omega_n^2$$

natural frequency

$$-g = \ddot{y} + 2\zeta\omega_n\dot{y} + \omega_n^2 y$$

$$\frac{b}{m} = 2\zeta\omega_n$$

$$k_{\text{controller}} = m\omega_{n,\text{desired}}^2$$

$$b_{\text{controller}} = 2m\zeta_{\text{desired}}\omega_n - b_{\text{robot}}$$

usually, $\zeta_{\text{desired}} = 1$ *damping ratio*

What are the effects of the gains?

- The system goes unstable if either k_p or k_d are negative
- The system is critically damped if $\zeta = \frac{b}{2m\omega_n} = 1$
- For a fast response, k_p should be as high as possible, subject to saturation, chattering, etc.
- With a constant disturbance D (e.g., gravity), the constant offset with PD control is $-\frac{D}{k_p}$

https://www.youtube.com/watch?v=C3jDRbwDzsM

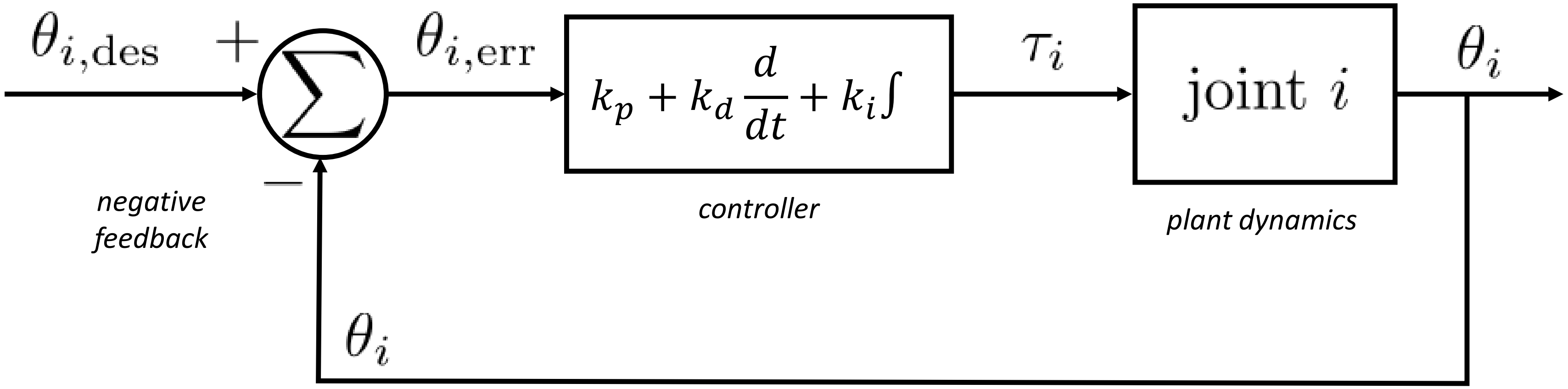
Desired Joint Angles

$$\theta_{1,\mathrm{des}}, \theta_{2,\mathrm{des}}, \theta_{3,\mathrm{des}} \ \ldots$$

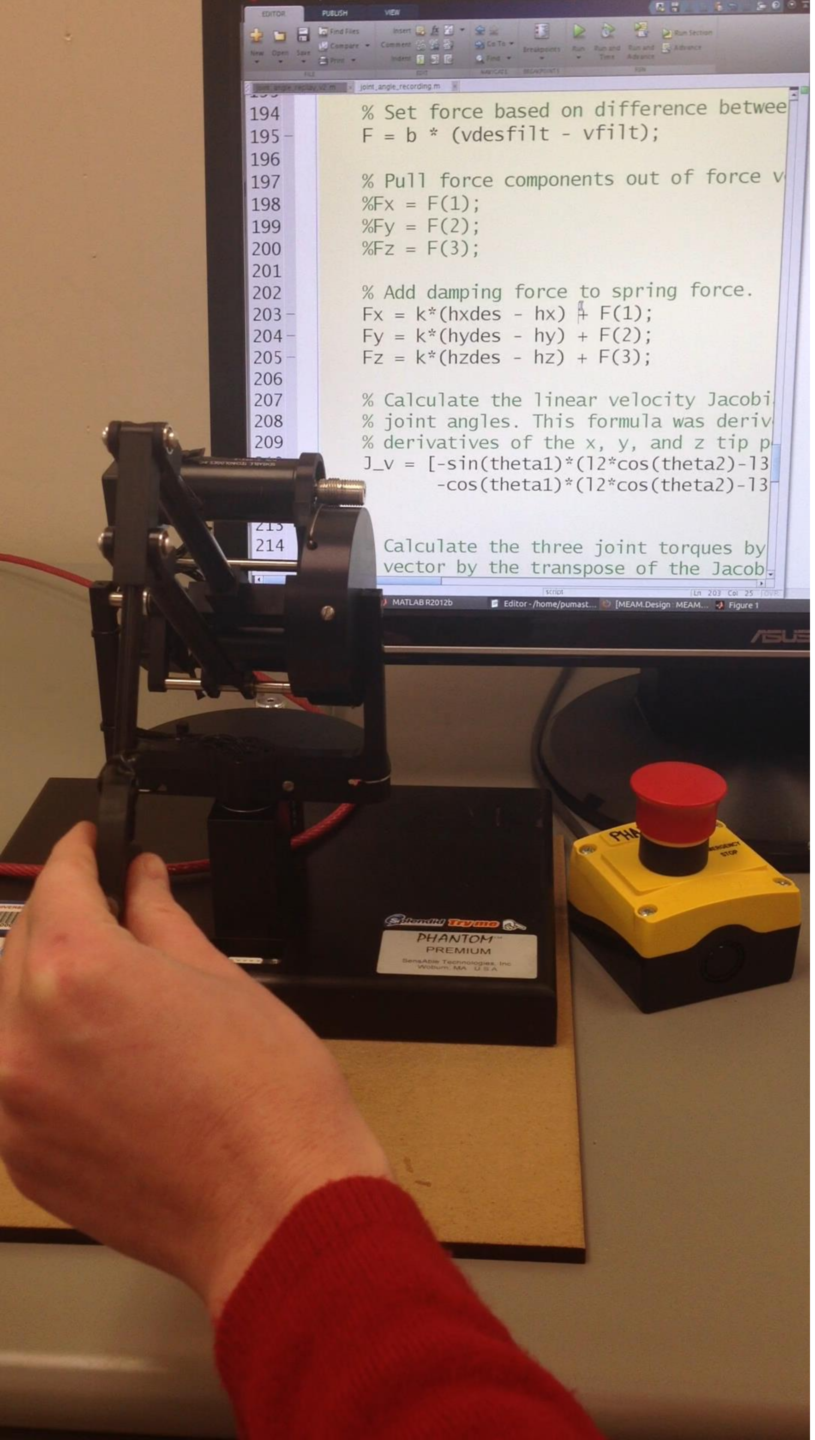
Actual Joint Angles

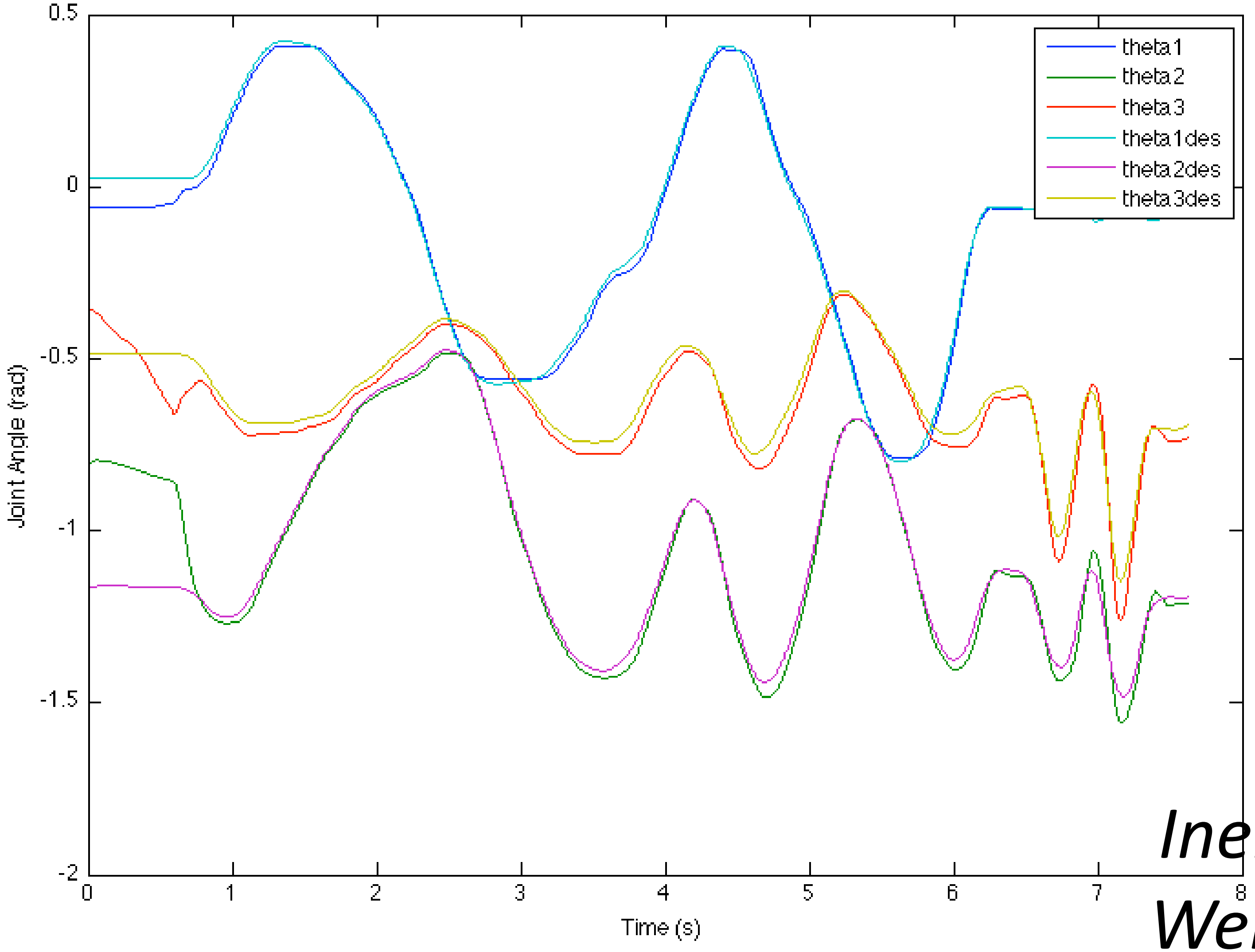
$$\theta_1, \theta_2, \theta_3 \ \ldots$$

Proportional Integral Derivative Feedback Controller



$$\tau_i = k_p(\theta_{i,des} - \theta_i) + k_d(\omega_{i,des} - \omega_i) + k_i \int \theta_{i,des} - \theta_i$$





Inertia
Weight
Friction

Why isn’t it perfect?

The robot’s dynamics interfere with tracking.

The inertia, weight, and friction of the robot all interfere with tracking.

Robot designers generally try to minimize the **inertia (mass & mass distribution)** of the robot so it can **accelerate** more quickly and be less affected by gravity.

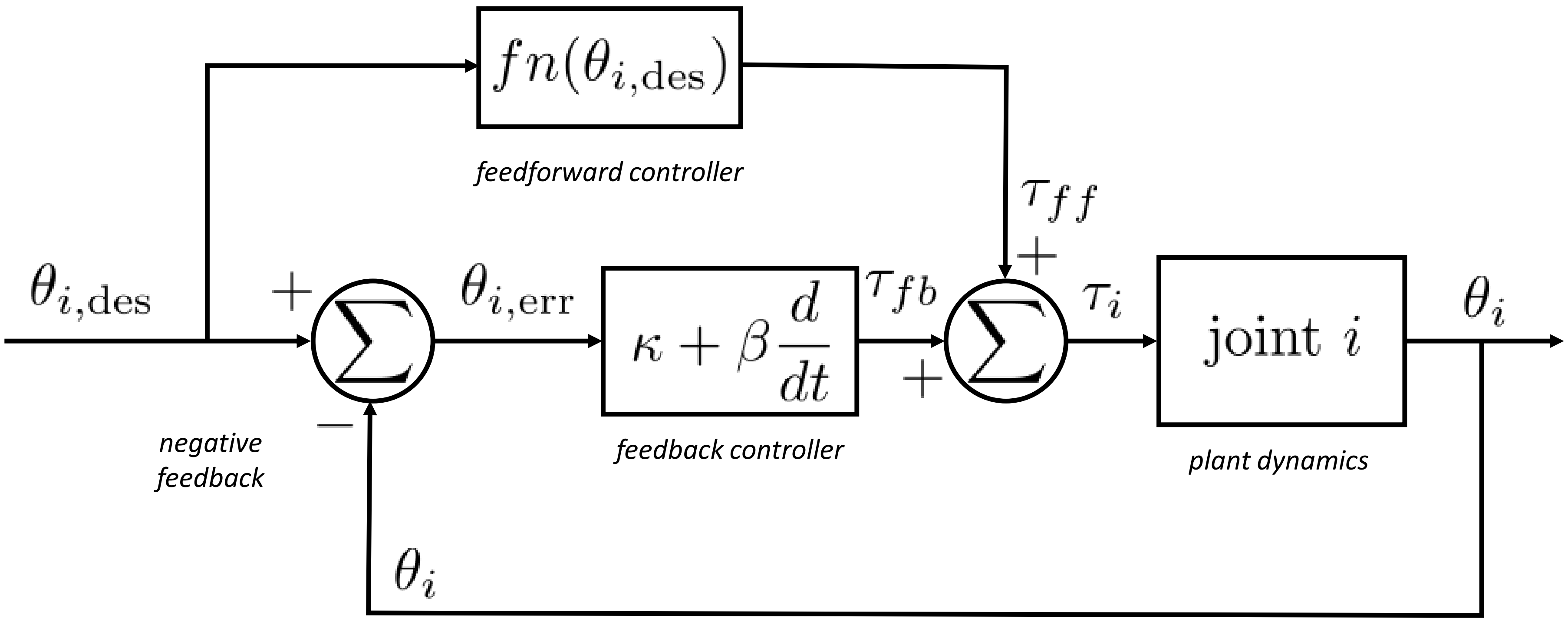
Similarly, robot designers try to minimize the **friction** of the robot so that the start of motion is **smooth** and sustained motion doesn’t require much torque.

How can we improve the controller’s tracking?

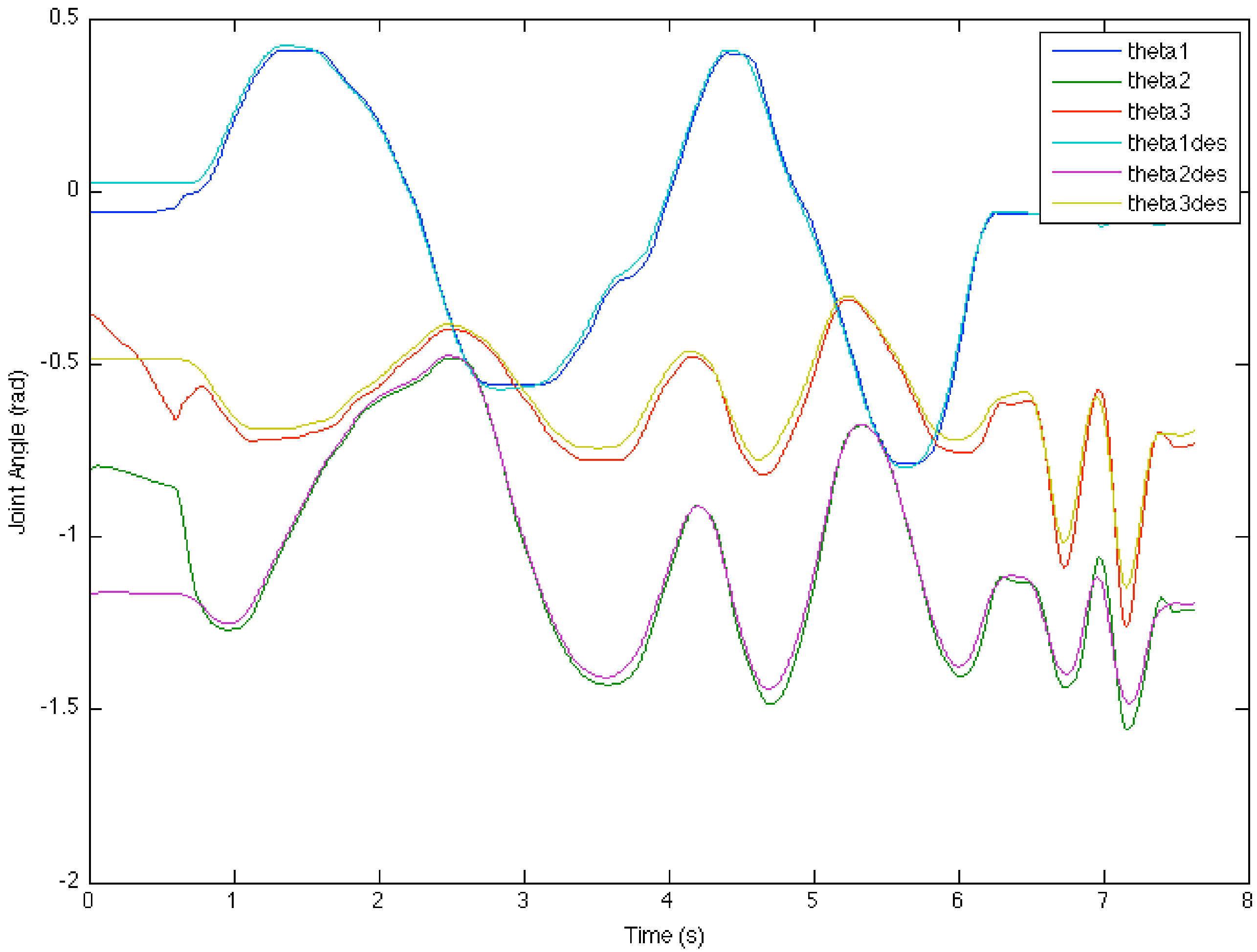
Try to compensate for the robot's dynamics in advance, instead of just reacting to errors when they occur.

This approach is called **feedforward control**, and it's very powerful for tracking time-varying trajectories.

Adding a Feedforward Term to the PD Controller



Which aspect of the robot dynamics can we feedforward?



Inertia

Weight

Friction

We could try any of them, but robot weight is generally the easiest and most useful.

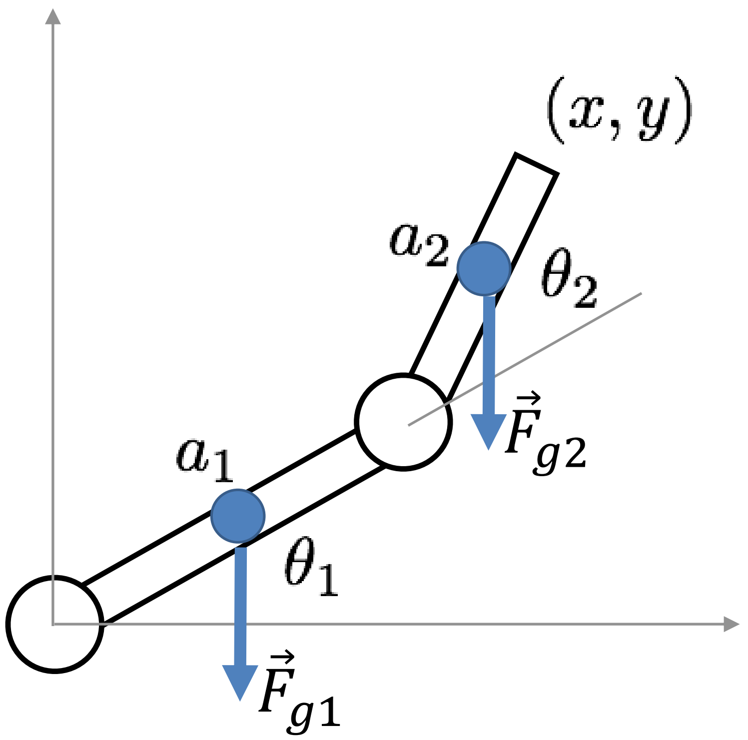
Previously: Gravitational Force/Torque

$$\vec{\tau}^{\top} d\vec{q} = \vec{F}^{\top} d\vec{x}$$

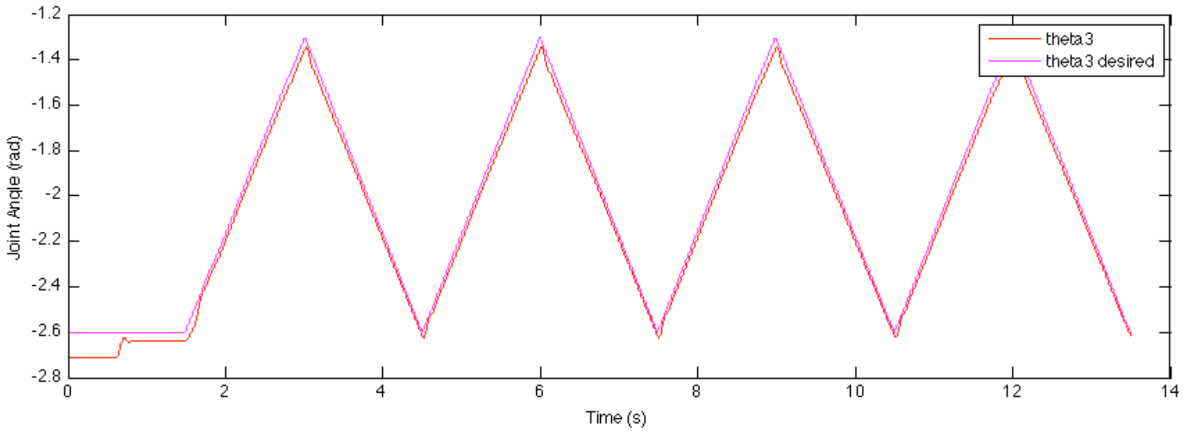
$$\vec{\tau}^{\top} d\vec{q} = \sum_{i=1}^{n} \vec{F}_{gi}^{\top} d\vec{x}_i$$

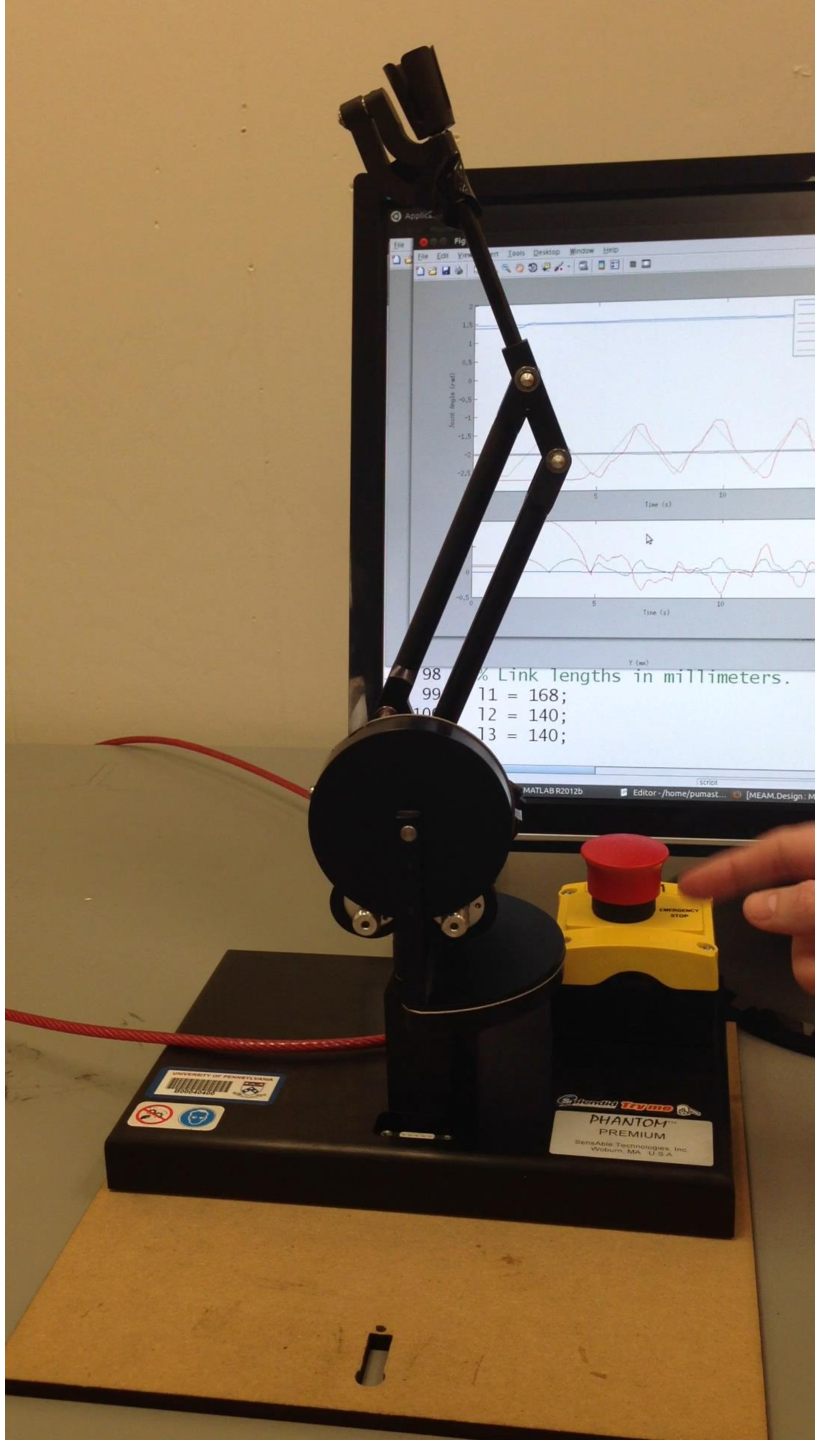
$$\vec{\tau}^{\top} d\vec{q} = \sum_{i=1}^{n} \vec{F}_{gi}^{\top} J_i d\vec{q}$$

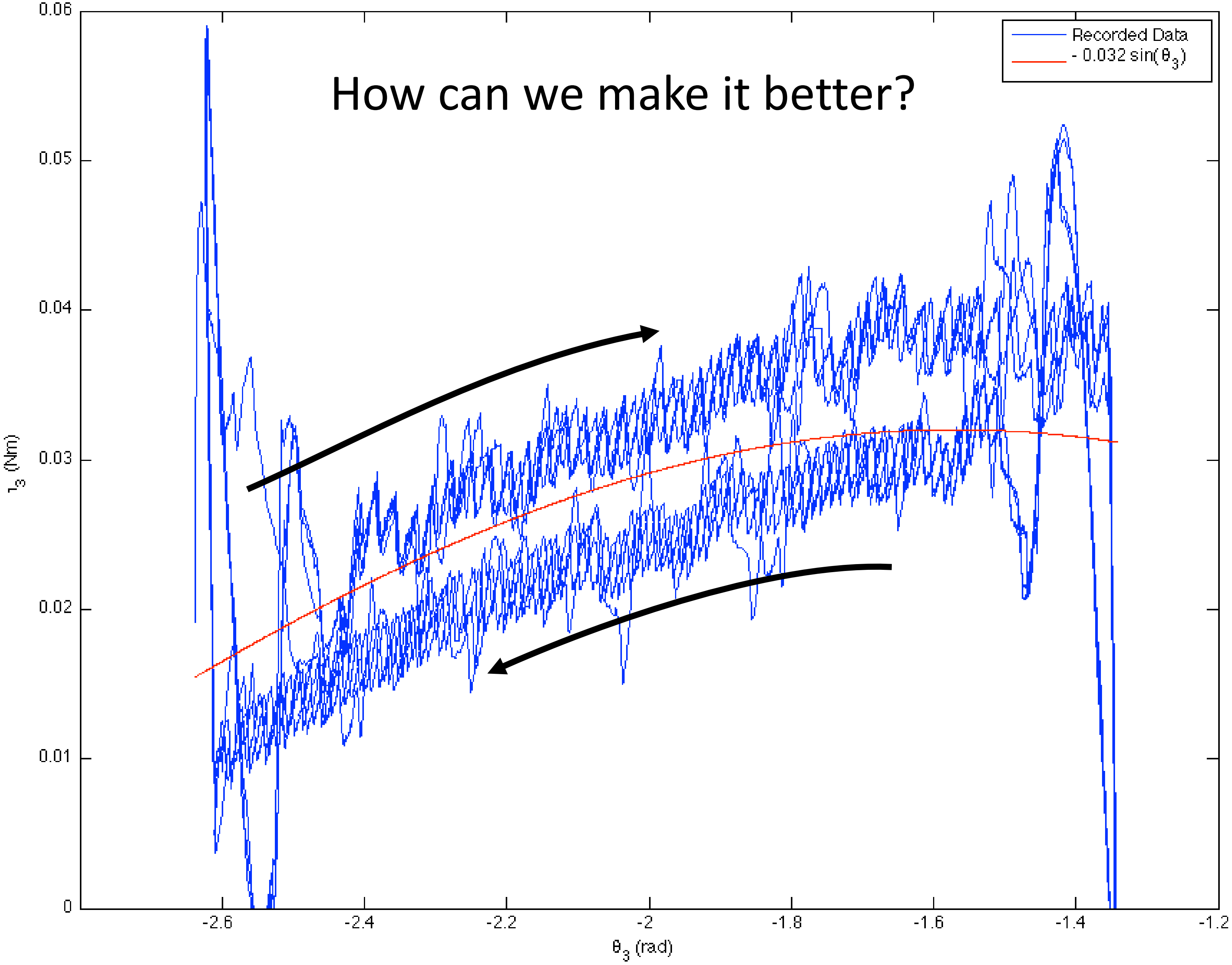
$$\vec{\tau} = \sum_{i=1}^{n} J_i^{\top} \vec{F}_{gi}$$



Another Option: Record torque on robot through a trajectory







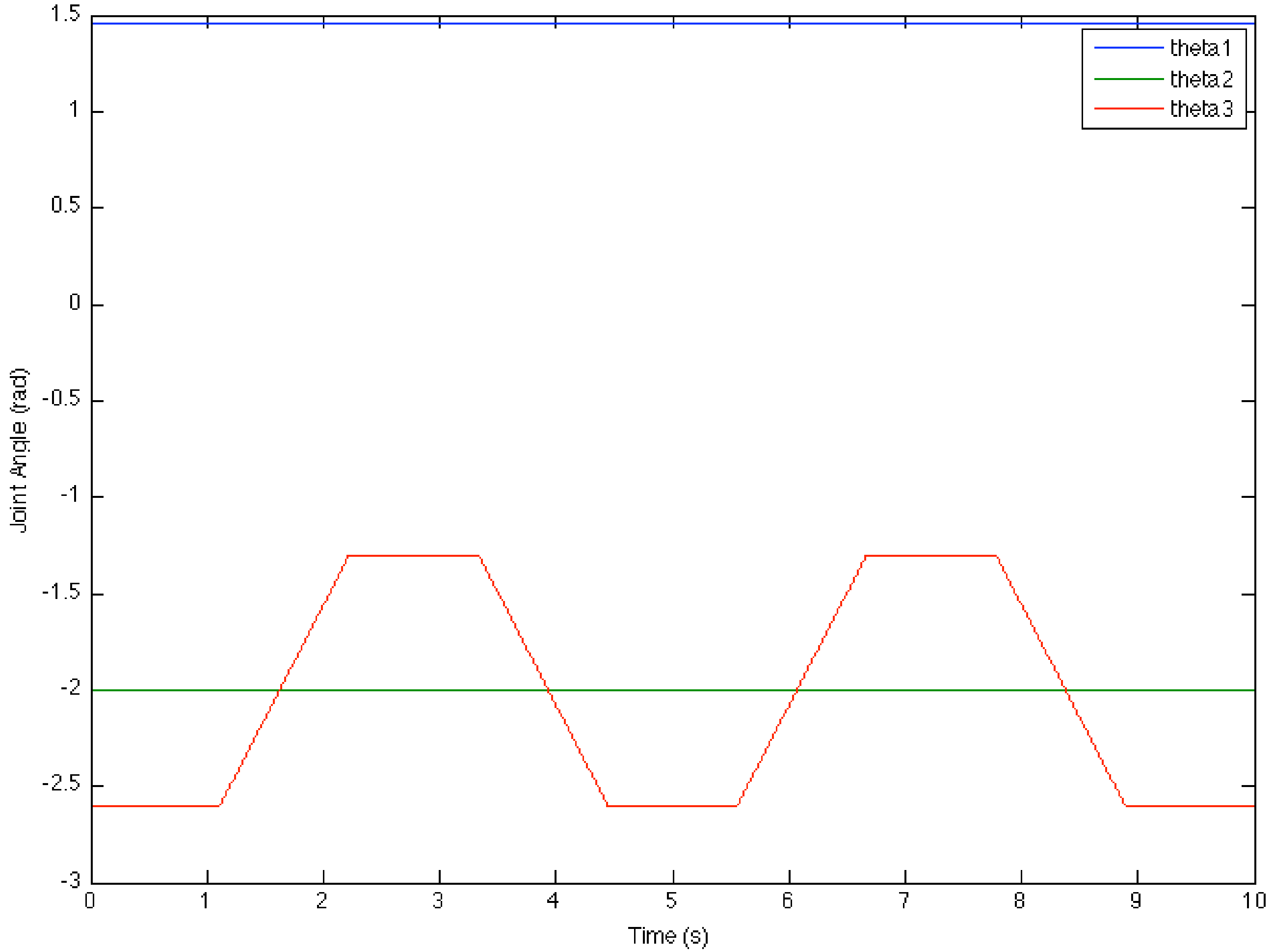
Inertia

Weight

Friction

How does each of these dynamic properties affect this test?

Flatten the sharp corners



Trajectory Generation

start

end

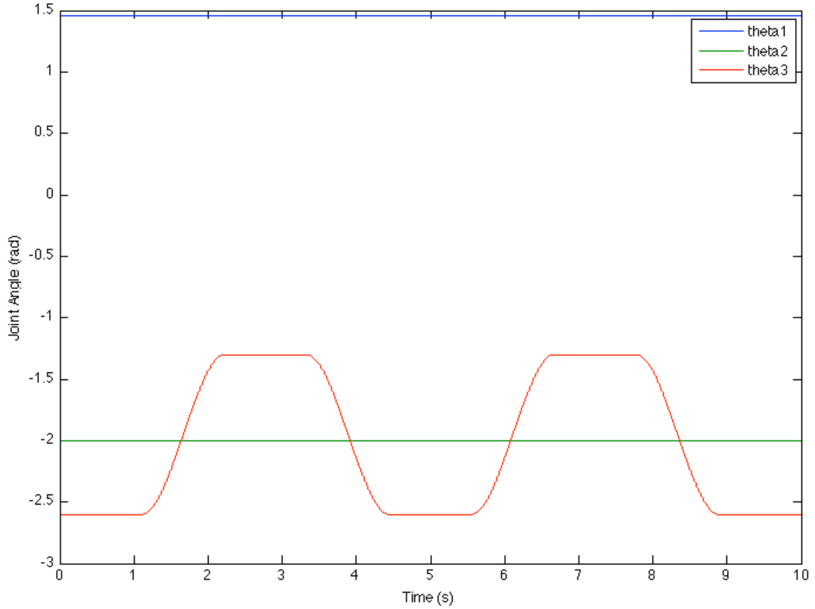
$$q(t_0) = q_0 \longrightarrow q(t_f) = q_f$$

$$\dot{q}(t_0) = v_0 \longrightarrow \dot{q}(t_f) = v_f$$

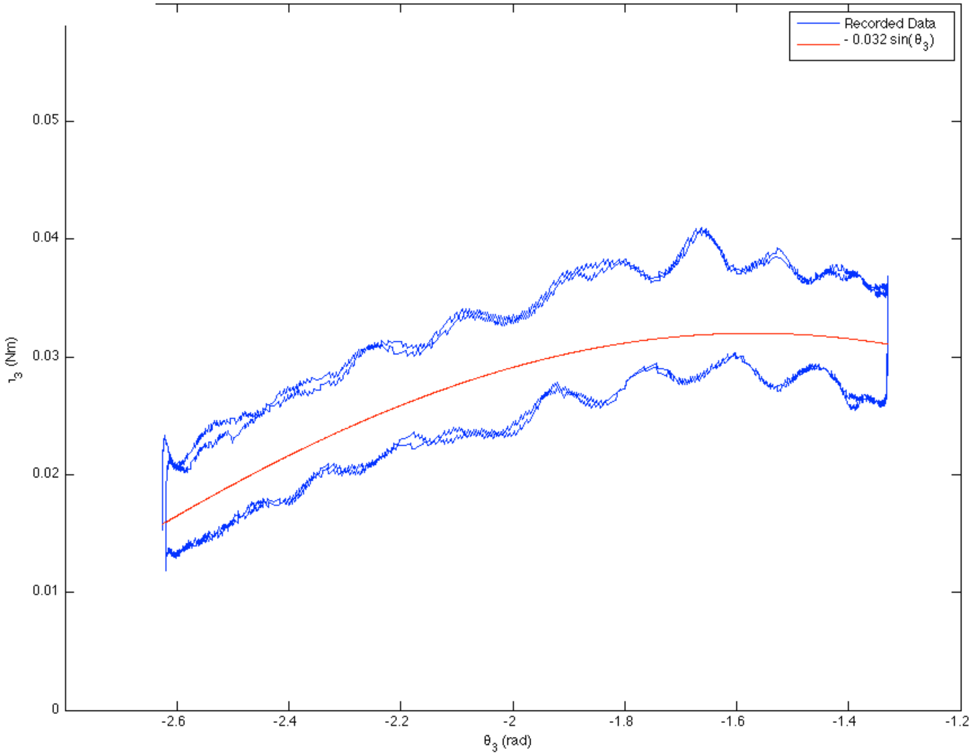
cubic polynomial

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

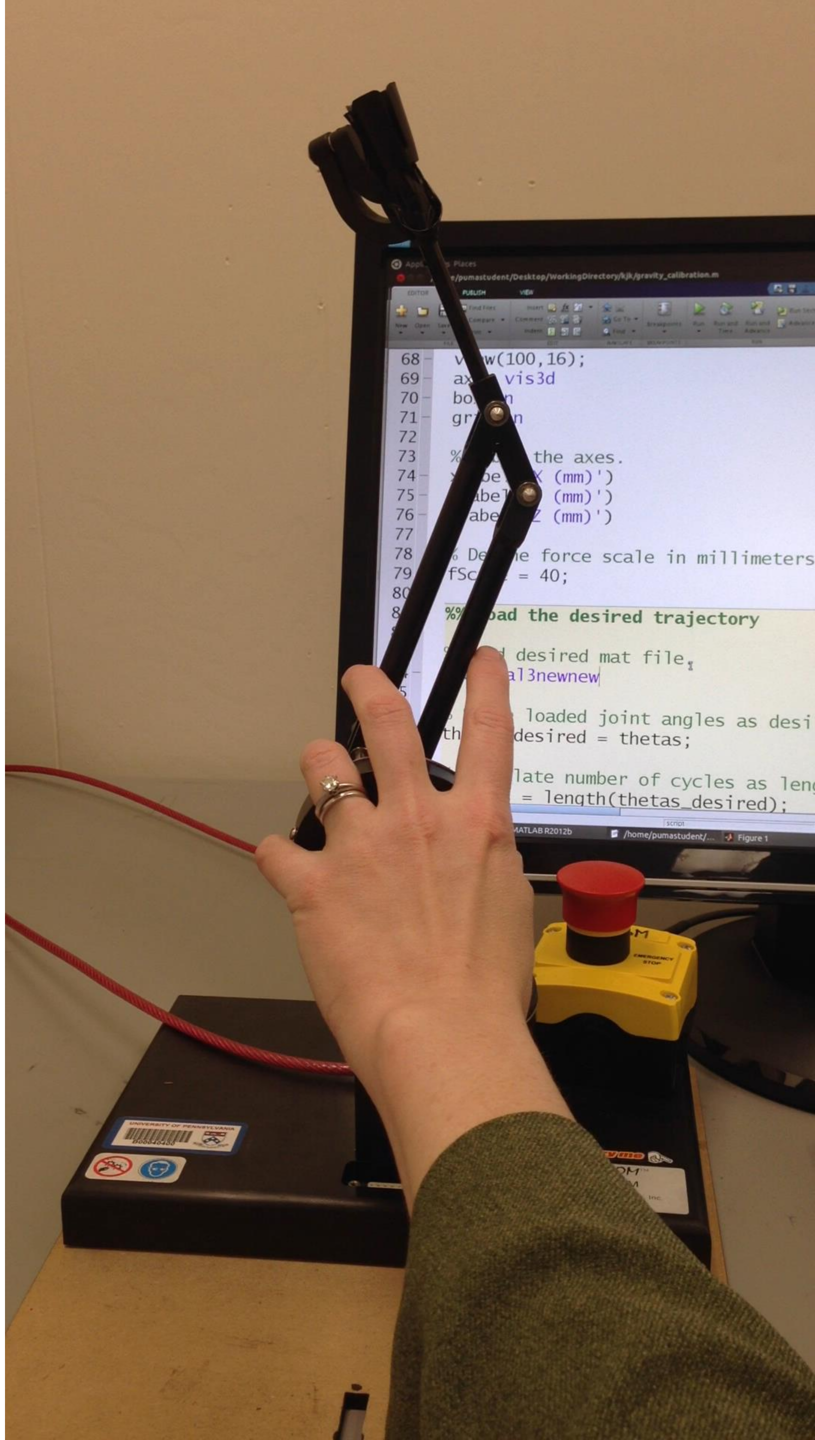
$$\begin{bmatrix} q_0 \\ v_0 \\ q_f \\ v_f \end{bmatrix} = \begin{bmatrix} 1 & t_0 & {t_0}^2 & {t_0}^3 \\ 0 & 1 & 2t_0 & 3{t_0}^2 \\ 1 & t_f & {t_f}^2 & {t_f}^3 \\ 0 & 1 & 2t_f & 3{t_f}^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$



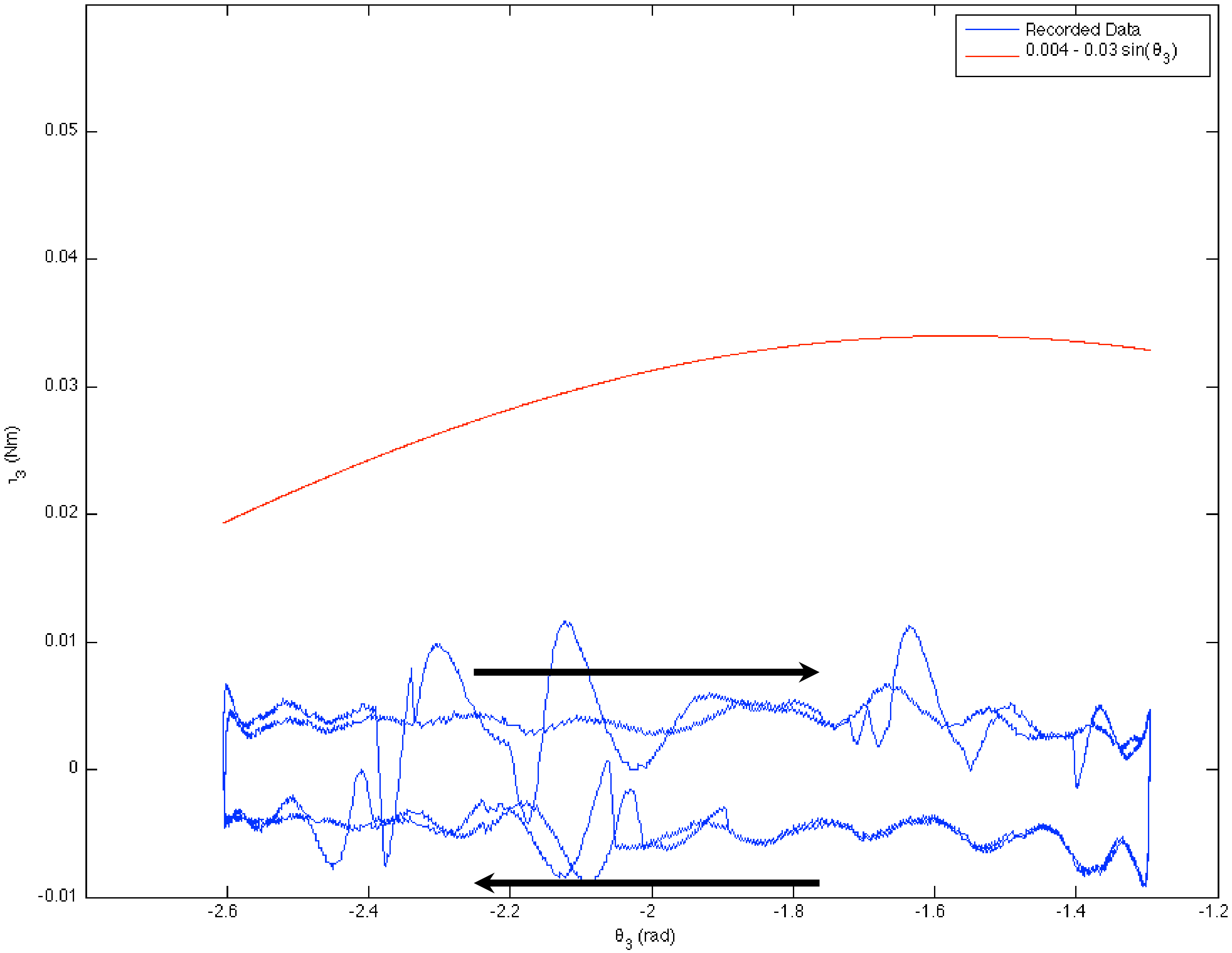
theta1
theta2
theta3
Joint Angle (rad)
Time (s)
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
0 1 2 3 4 5 6 7 8 9 10

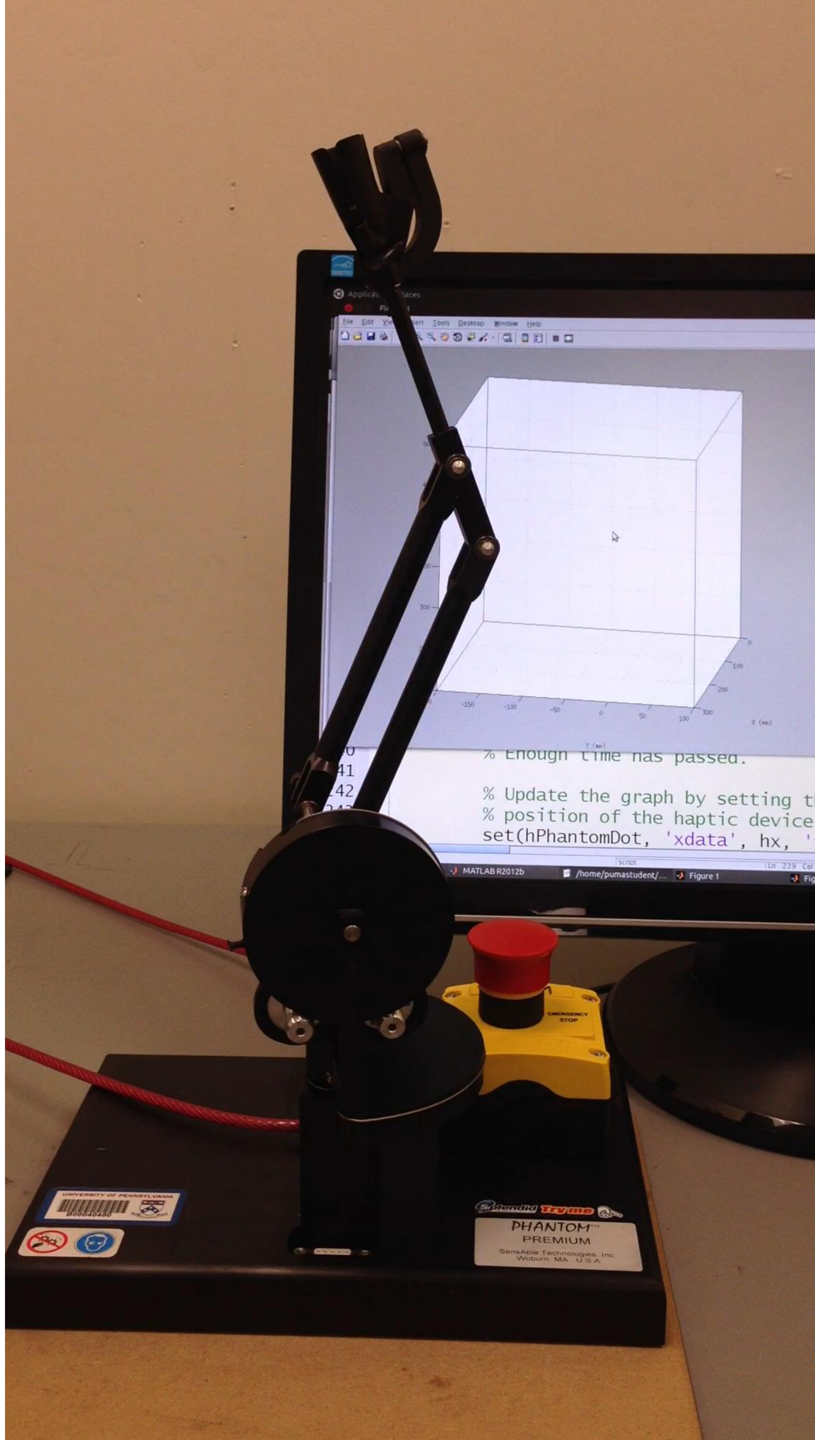


Recorded Data
- 0.032 sin(θ_3)
τ_3 (Nm)
θ_3 (rad)
0.05
0.04
0.03
0.02
0.01
0
-2.6 -2.4 -2.2 -2 -1.8 -1.6 -1.4 -1.2



You can also use this strategy to compensate for friction.





Upcoming: Sensing, State Estimation

Read

- AKKK: 8.1 – 8.3

Updated Schedule

- 11/24: Sensing and State Estimation
- 12/1: Paper reading: Multi-robot coordination
- 12/3-12/10: Final Project