ME 02.74/740: Bio-inspired Robotics

HW 04: Operational Space Control and Impact Based Constraint

Due Online, Tuesday, Oct. 9th, 11:59 PM

---

1 Operational Space Control

In HW3 you designed a control law for the leg shown in Figure 1. Given a desired position of the foot $\mathbf{r}_E^d = [x_d,\, y_d]^\top$, your control law took the form:

$$\tau = J^T \left[K(\mathbf{r}_E^d - \mathbf{r}_E) + D(\dot{\mathbf{r}}_E^d - \dot{\mathbf{r}}_E) \right] . \tag{1}$$

As we experienced in lab, this method effectively controlled the position of the foot using a virtual spring-damper in task-space. However, the Jacobian $J = \frac{\partial \mathbf{r}_E}{\partial q}$ was the only part of this control law that required information about the model! This homework will explore methods to improve the performance of the controller using more information about the model. Specifically, we will use information about its dynamics.

As we have seen in class, the dynamics of the leg are given by the **configuration-space** equations of motion:

$$M(q)\ddot{q} + V(q,\dot{q}) + G(q) = \tau \tag{2}$$

A_leg Corr_leg Grav_leg

After a bit of algebraic manipulation, these equations can be rearranged into the **operational-space** equations of motion to describe the dynamics of the foot:

$$\Lambda(q)\,\ddot{\mathbf{r}}_E + \mu(q,\dot{q}) + \rho(q) = F \tag{3}$$

Lambda

where $\Lambda(q)$ is the effective mass felt at the foot, $\mu(q,\dot{q})$ gives the coriolis and centripetal forces on the foot, and $\rho(q)$ gives the gravity force felt on the foot. Formula for these quantities are given below:

$$\Lambda(q) = (JM^{-1}J^T)^{-1} \tag{4}$$

$$\mu(q,\dot{q}) = \Lambda J M^{-1} V - \Lambda \dot{J}\dot{q} \tag{5}$$

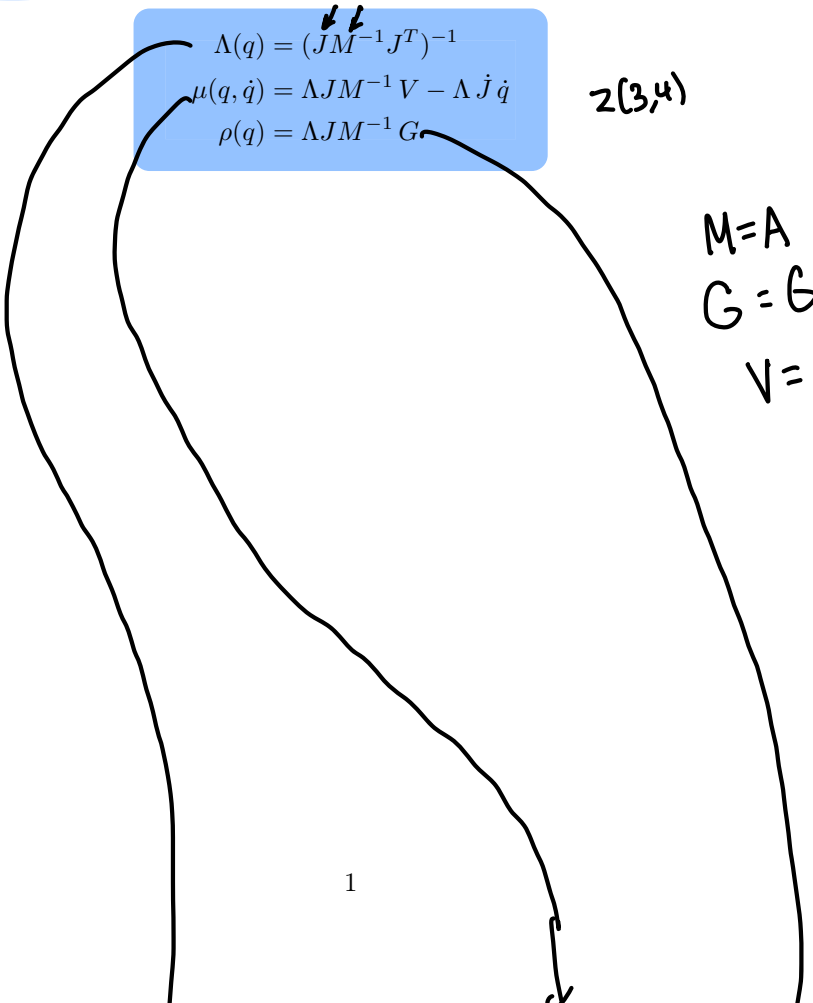
$$\rho(q) = \Lambda J M^{-1} G \tag{6}$$

2(3,4)

M=A

G=Grav

V=Grav



In this assignment, you will explore the use of an extended version of Eq. 1 given by:

$$\tau = J^\top \left[\Lambda \left(\ddot{\mathbf{r}}_E^d + K(\mathbf{r}_E^d - \mathbf{r}_E) + D(\dot{\mathbf{r}}_E^d - \dot{\mathbf{r}}_E)\right) + \mu + \rho\right] . \tag{7}$$

Under mild assumptions, it can be show that the foot will converge to the desired trajectory with this control law.

1. Download the updated `derive_leg_HW4.m` code from Canvas. Using the `simulate_leg_HW4.m` code from Canvas as a starting point, implement the control law in Eq. 7. The script `derive_leg_HW4.m` creates a number of useful functions what will help in implementing the controller.
2. With this new controller, rerun the circular trajectory tracking simulation from HW3. Do not simulate contact with the ground. You should use parameters:

$$\mathbf{r}_E^d(t) = \begin{bmatrix} 0.025\cos(\omega t) \\ -0.125 + 0.025\sin(\omega t) \end{bmatrix} \tag{8}$$

$$\omega = 30 \text{ rad/s} \tag{9}$$

$$K = \begin{bmatrix} 150 & 0 \\ 0 & 150 \end{bmatrix} \tag{10}$$

$$D = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \tag{11}$$

$$q|_{t=0} = \begin{bmatrix} -\pi/4 \\ \pi/2 \end{bmatrix} \tag{12}$$

$$\dot{q}|_{t=0} = [0, 0]^T \tag{13}$$

   Turn in a plot of of x, y, x_d, and y_d versus time. Choose an appropriate time scale for the plot.
3. Repeat step 2. using modified control laws:
   - $\tau = J^T \left[\Lambda \left(\ddot{\mathbf{r}}_E^d + K(\mathbf{r}_E^d - \mathbf{r}_E) + D(\dot{\mathbf{r}}_E^d - \dot{\mathbf{r}}_E)\right) + \mu\right]$
   - $\tau = J^T \left[\Lambda \left(\ddot{\mathbf{r}}_E^d + K(\mathbf{r}_E^d - \mathbf{r}_E) + D(\dot{\mathbf{r}}_E^d - \dot{\mathbf{r}}_E)\right) + \rho\right]$
   - $\tau = J^T \left[\Lambda \left(K(\mathbf{r}_E^d - \mathbf{r}_E) + D(\dot{\mathbf{r}}_E^d - \dot{\mathbf{r}}_E)\right) + \mu + \rho\right]$

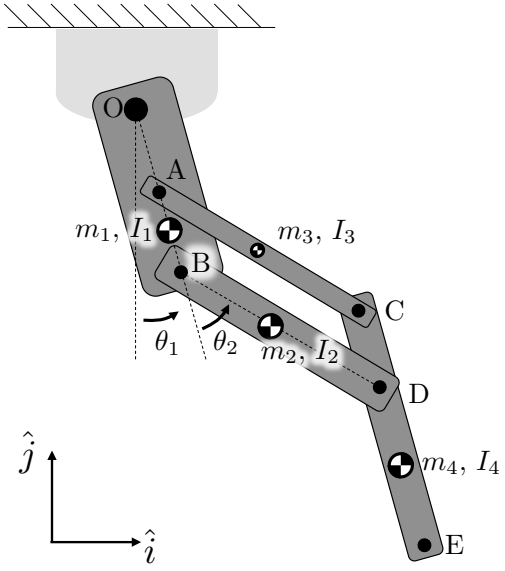
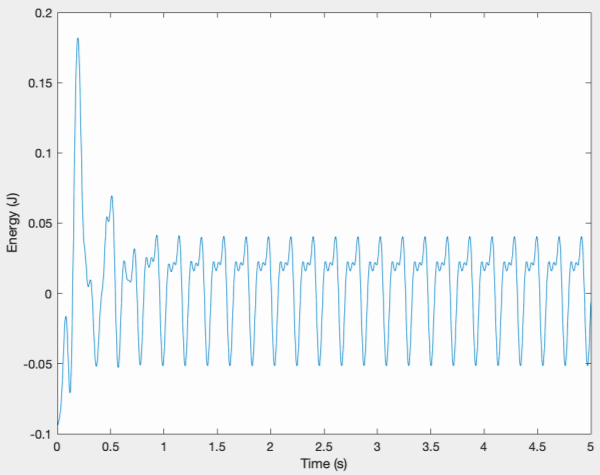
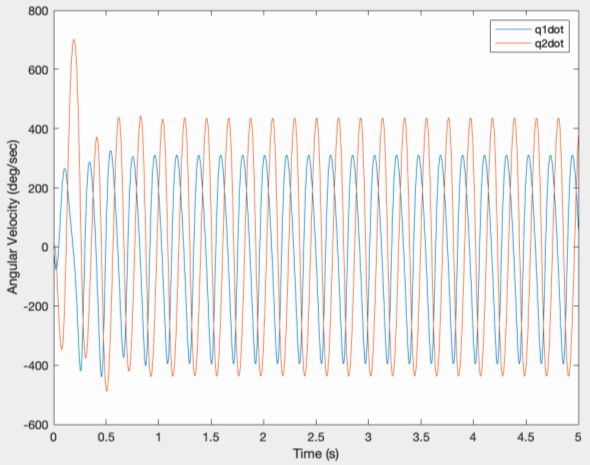
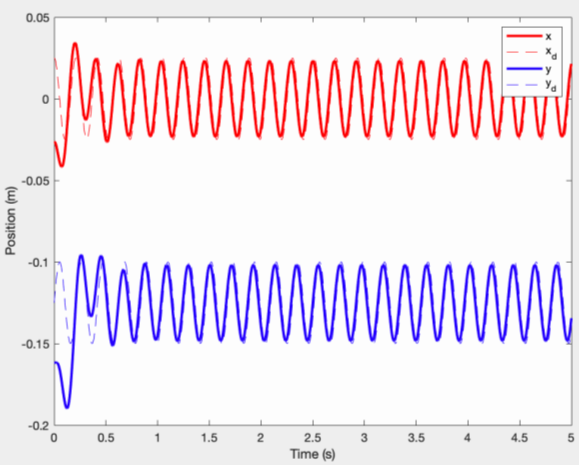


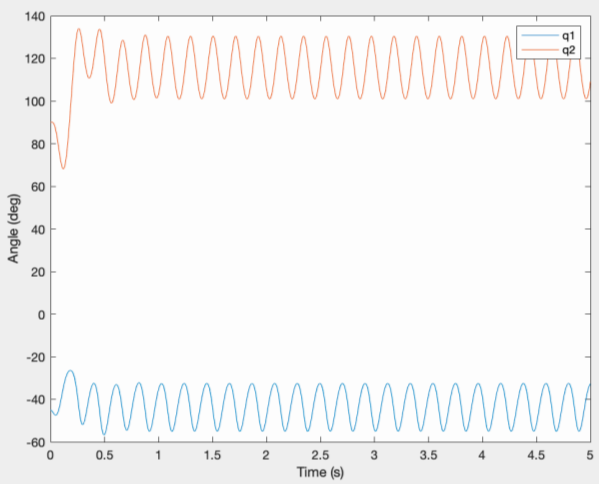
Figure 1: Leg model.

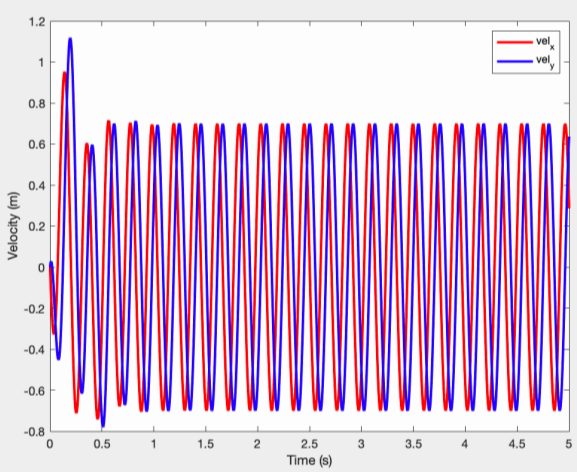
1.2

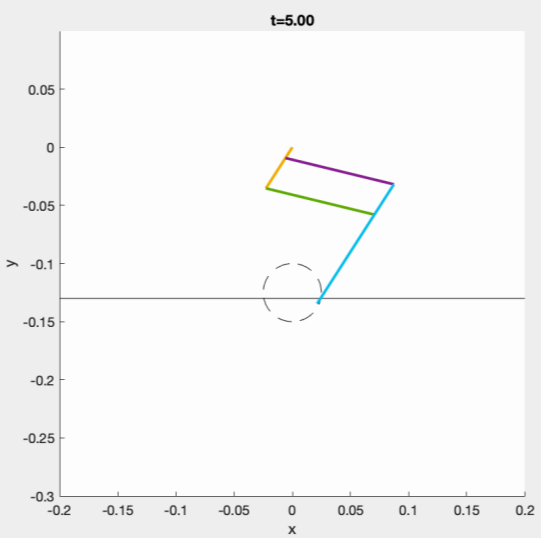

Energy (J)
Time (s)


q1dot
q2dot
Angular Velocity (deg/sec)
Time (s)


x
x_d
y
y_d
Position (m)
Time (s)

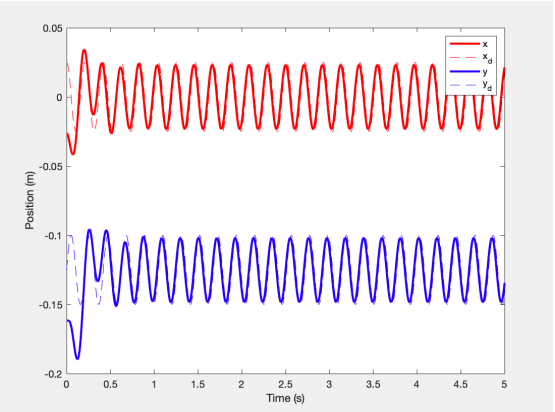

q1
q2
Angle (deg)
Time (s)


vel_x
vel_y
Velocity (m)
Time (s)


t=5.00
y
x

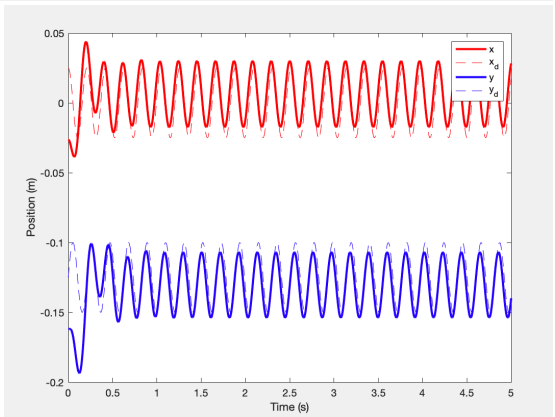
1.3

All:



No ρ:

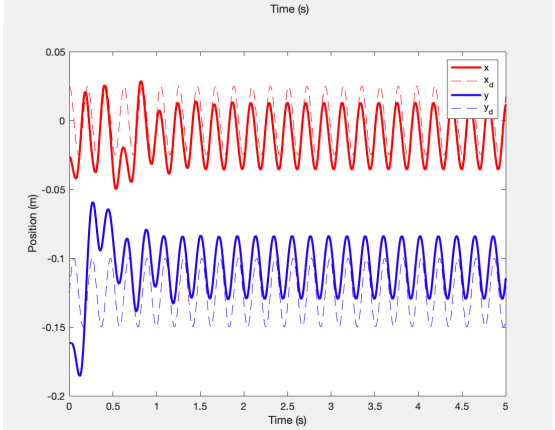
(gravity force)



a little off of circular trajectory

No μ:

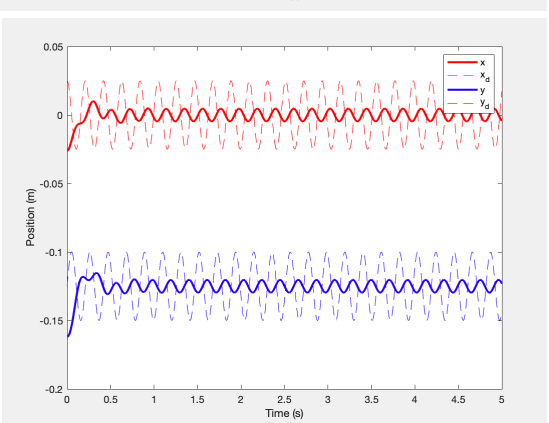
(coriolis/centripetal force)



only follows top half of circular trajectory

No $\ddot{r}_E^d$:

(desired acceleration)



small circle trajectory (much smaller than desired)

Turn in a plot of of x, y, x_d, and y_d versus time for each case. Also, include a short description of what is neglected in each controller, and how that relates with the observed performance.

4. (Grad students only) The control in the previous section included perfect information about the model. Which of our model **parameters** would be most prone to improper estimation? Extend the simulator to use a different set of model parameters for control vs. simulation. Turn a plot and description of the modified simulation with imperfect model information.

2 Discrete Impact Based Contact Simulation

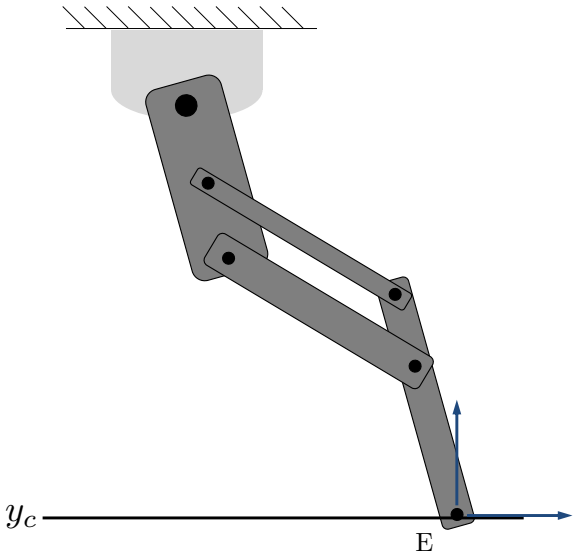


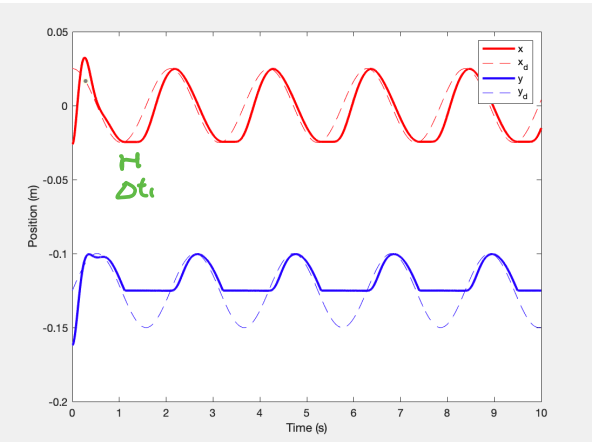
Figure 2: Leg model with contact.

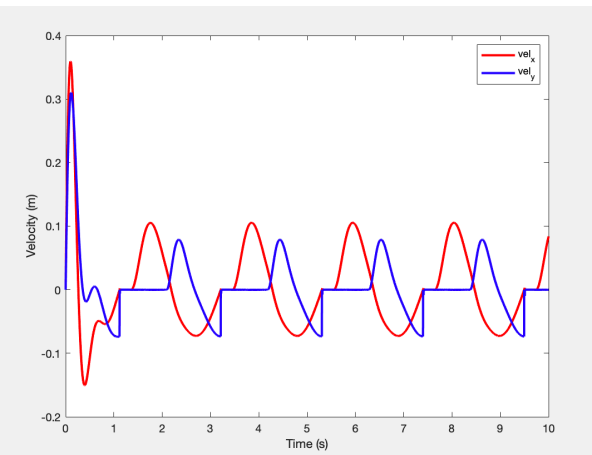
In this problem, you will extend your previous simulation to include constraints. In this homework, we will add the slippery surface, which makes a friction force depending on the friction coefficient and the normal force as shown in Figure 2.

1. Given a ground height $y_c = -0.125$ m, complete skeleton the code for the function `discrete_impact_contact` in `simulate_leg_HW4.m`. This function should:
   - Compute the height of the foot relative to the ground $C_y = y - y_c$.
   - Compute the $\dot{C}_y = \dot{y}$.
   - If the constraints are not violated (i.e. $C_y > 0$ or $\dot{C}_y > 0$), then your function should not update $\dot{q}$.
   - If the constraints are violated (i.e. $C_y < 0$ and $\dot{C}_y < 0$), compute the vertical impulse force, $\hat{F}_{c,y} = \Lambda_{c,y}(-\gamma \dot{C}_y - J_{c,y}\dot{q})$, where $\Lambda_{c,y}$ is the vertical directional operational space mass, γ is the coefficient of restitution, and $J_{c,y}$ is the vertical directional Jacobian at the contact point.
   - Update $\dot{q}$ using the equation, $\dot{q} = \dot{q} + M^{-1} J_{c,y}^\top \hat{F}_{c,y}$.
   - Using the same procedure, update $\dot{q}$ by applying tragential impulse force to satisfy a friction cone constraint. Compute the tangential impulse force, $\hat{F}_{c,x} = \Lambda_{c,x}(0 - J_{c,x}\dot{q})$.
   - Truncate $\hat{F}_{c,x}$ if it is outside of friction cone, $> \mu \hat{F}_{c,y}$.
   - Update $\dot{q}$ using the equation, $\dot{q} = \dot{q} + M^{-1} J_{c,x}^\top \hat{F}_{c,x}$.

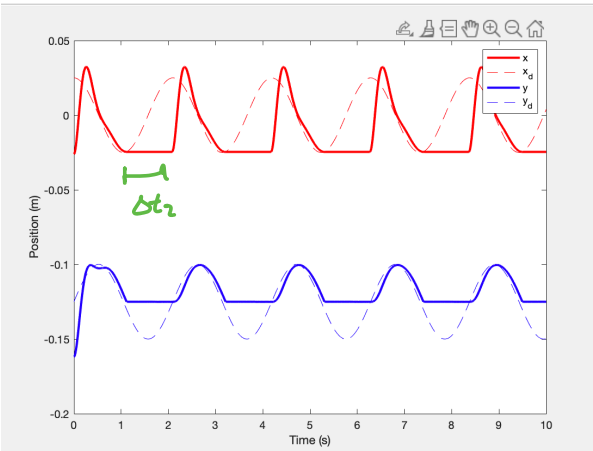
2. Find a proper place to put the update function in the simulation. Does it need to be in `dynamics` function? Or in the middle of numeric integration? NO, it's just that one is instantaneous & the other is over a timestep
3. Use the coefficient of resitution $\gamma = 0$ and a friction coefficient $\mu = 0.3$ to simulate trajectory tracking for the circle task with $\omega = 3$ rad/s. Provide position and velocity plots of x and y over the interval $t = [0\text{s}, 10\text{s}]$.
4. Increase μ to 10 and perform another simulation. Describe what changes in this simulation.
5. (Graduate students) Note that this method for enforcing constraints is general to cases beyond contact with the ground. For instance, in reality, this mechanism has a kinematic joint limit that constrains q_1. In your code, implement a joint limit constraint to enforce $q_1 > -50°$.

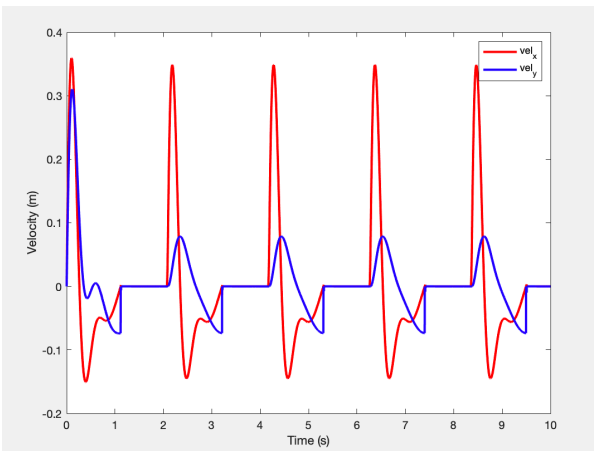
$\mu = 0.3$





$\mu = 10$





takes longer to kick upon initial contact w/ the floor

$\Delta t_2 > \Delta t_1$

also does not follow desired trajectory as well (especially in the x direction)