Impact Forces

Lecture #15

ECE 761: Robotics

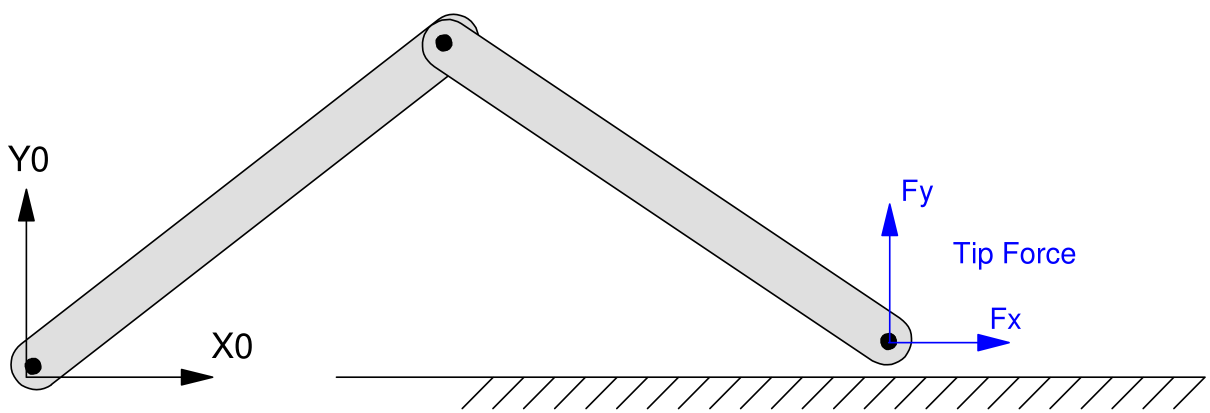
Class taught at North Dakota State University
Department of Electrical and Computer Engineering

Please visit www.BisonAcademy.com for corresponding lecture notes, homework sets, and solutions.

Impact Forces

- Control in free space is easy (2 d.o.f. robotic arm)
- Control when stationary in contact with a surface is easy (1 d.o.f. robotic arm)

What happens if you go back and forth from free space to contact with a surface?

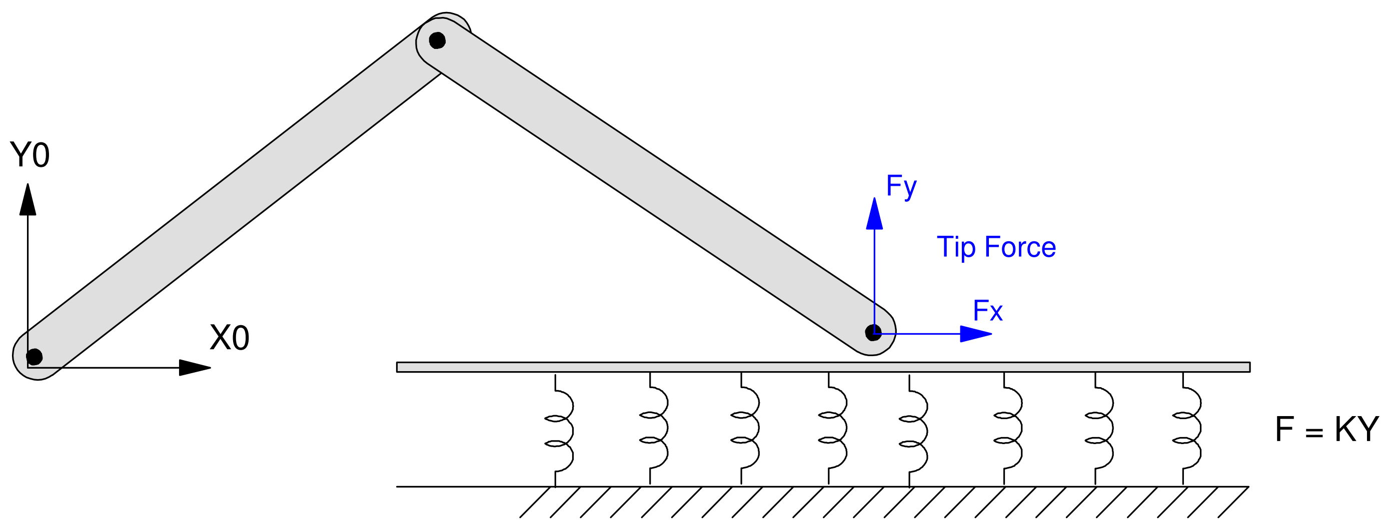


Tip Constraint: The robot motion is constrained by the floor at y = 0

Solution:

- Model the floor as a spring
- Add tip forces if the spring is compressed
- Remove the constrant
- Tip Forces translate to torques as

$$T = J^T F_{tip}$$



Example: Move back and forth from X2 = 0.5 to X2 = 1.5 with a force of 10N.

Solution: Specify the motion of the robot to be

$$x_{2ref} = \left(\frac{1-\cos(\pi t)}{2}\right) + 0.5$$

$$y_{2ref} = 0$$

- Using the previous Cartesian control
- Add a spring constant to the floor (100 N/m)

$$F_{floor} = \min(0, 100x_2 + 20\dot{x}_2)$$

X direction: PD control law

$$F_x = 16\left(x_{2ref} - x_2\right) + 6\left(\dot{x}_{2ref} - \dot{x}_2\right)$$

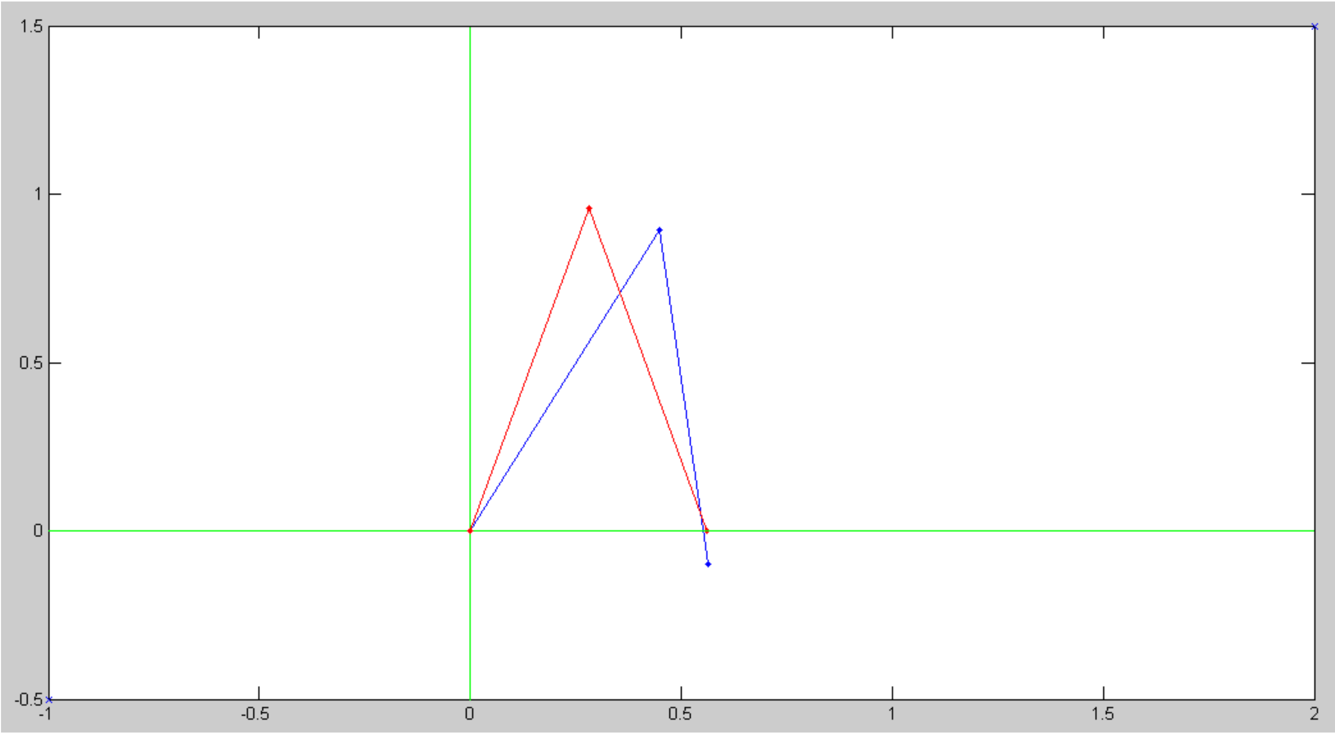
Y direction: I control

$$F_y = 100\int\left(10 - F_{floor}\right)dt$$

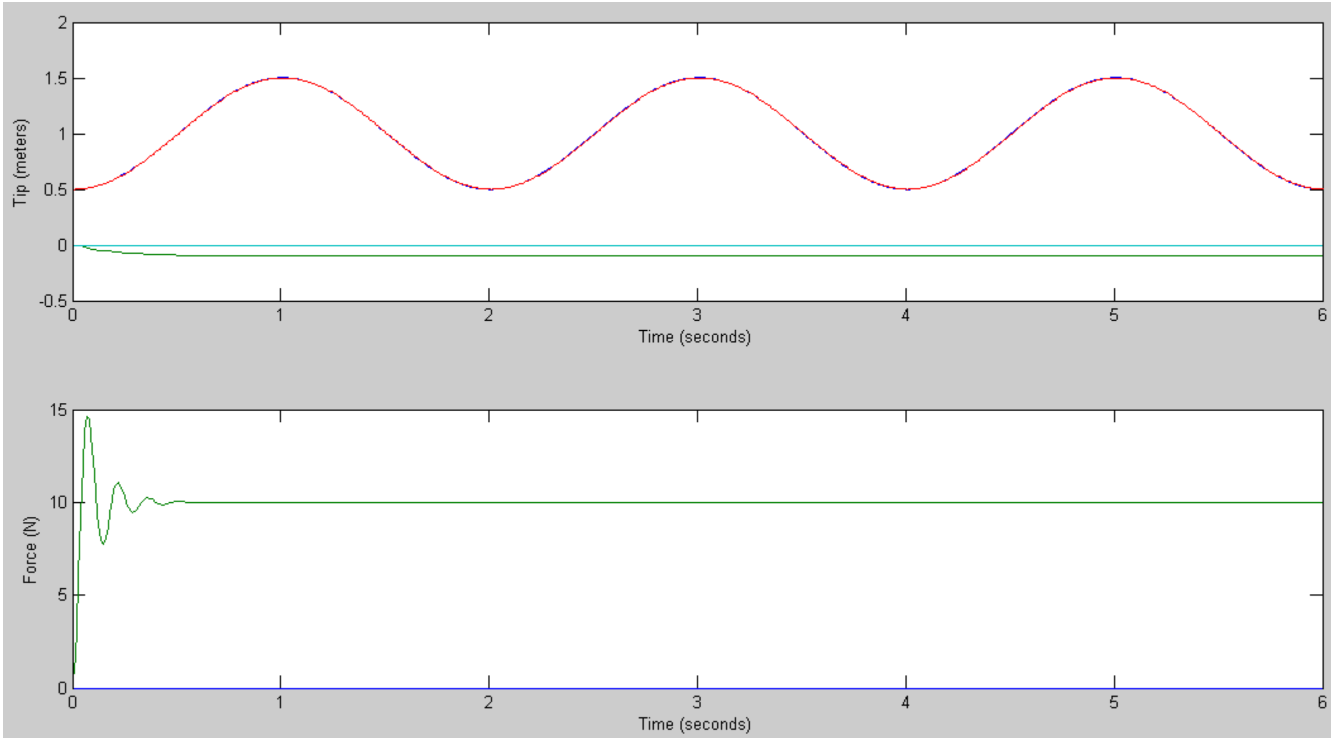
The joint torques are then computed using the Jacobian

$$T = J^T\begin{bmatrix} F_x \\ F_y \end{bmatrix}$$

Simulation Results (RR_XY_Floor_Control.txt)



Desired Motion of the Robot (red) and Actual (Blue). The robot sinks into the floor so that 10N of force is applied



Resulting tip position and tip force. The tip force is regulated at 10N as desired.

Impact Forces

Develop a control law that

- Starts out above the floor P0 = (0.5, 0.5)
- Moves to the floor P1 = (0.5, 0)
- Moves 1m to P2 = (1.5, 0)
- with a force of 10N
- Lifts up from the floor to P3 = (1.5, 0.5)
- Moves back to P0,

and repeat traces out a square with the corners at

Method #1: You don't really have to change anything at this point. Just specify the points on the floor as being below the surface and you'll get a force. The distance into the floor you need to go depends upon the net spring constant

$$K_{net} = K_{floor} || K_{pid}$$

where

Kpid is the P gain for your PID control law.

For example, let

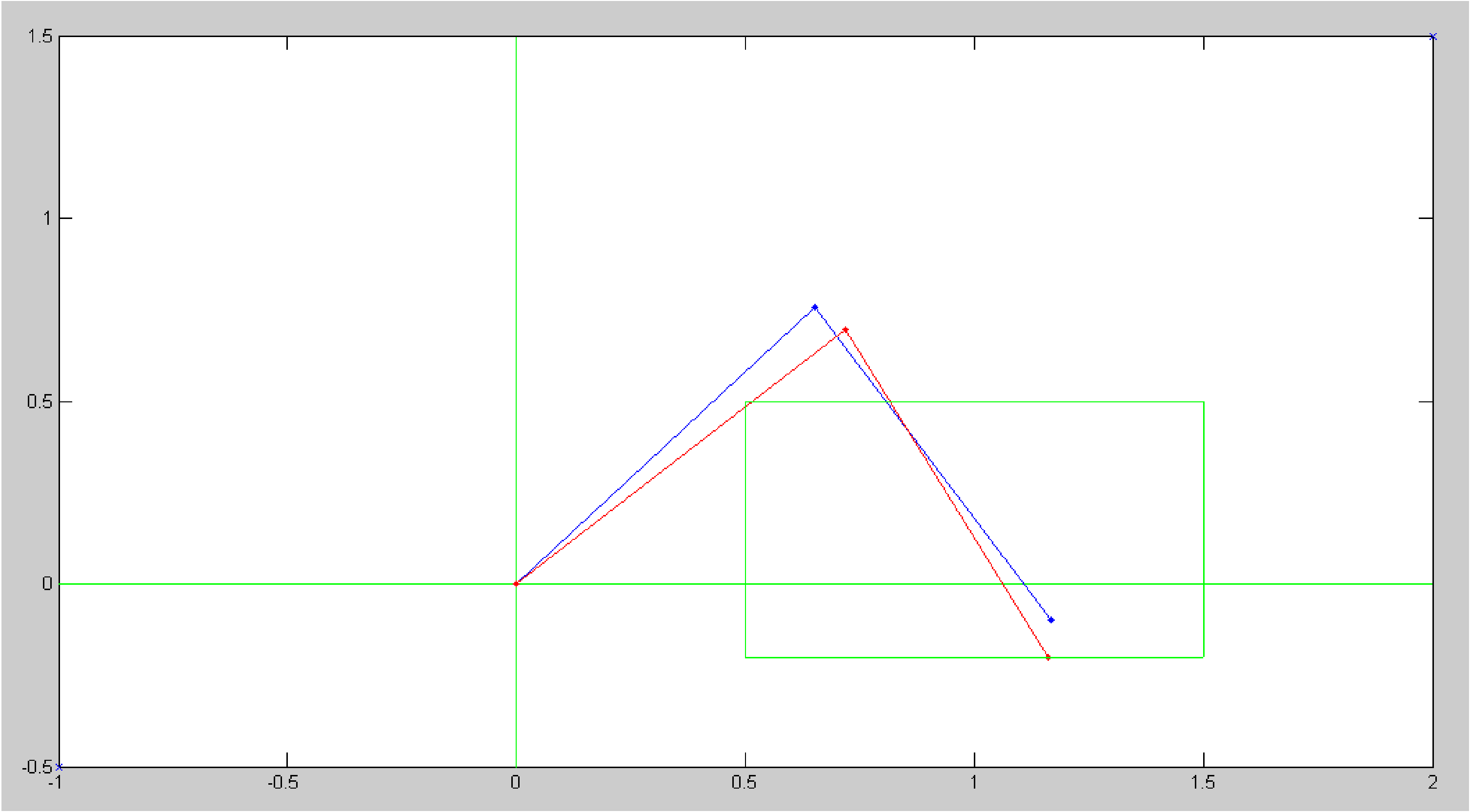
$$K_{floor} = 100$$

$$PID = 100(y_{2ref} - y_2) + 20(\dot{y}_{2ref} - \dot{y}_2)$$

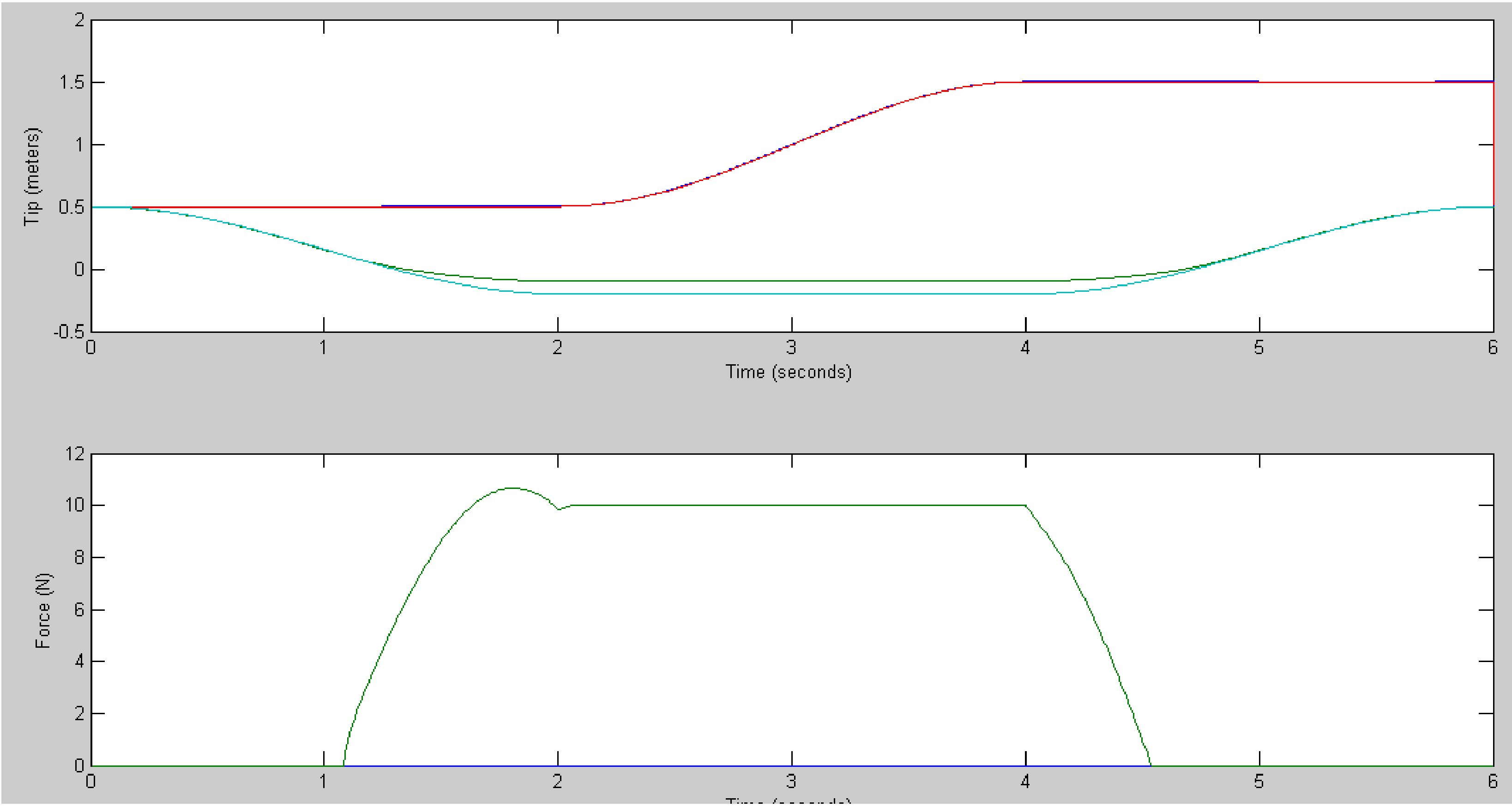
$$K_{net} = 100 || 100 = 50$$

for 10N of force, you need to try to move 0.2m into the floor

Simulation Results (XY_Floor_Control_v2.txt)



For a force of 10N, the desired position is 0.2m into the floor



Resulting tip positionand tip force. When in contact with the floor, the force is 10N as desired.

Force Control (take 2)

While the former method works, it requires you to specify the path depending upon the choses spring constant for the floor and the chosen P gain in the PID controller.

Using an integrator for force control works really well, however.

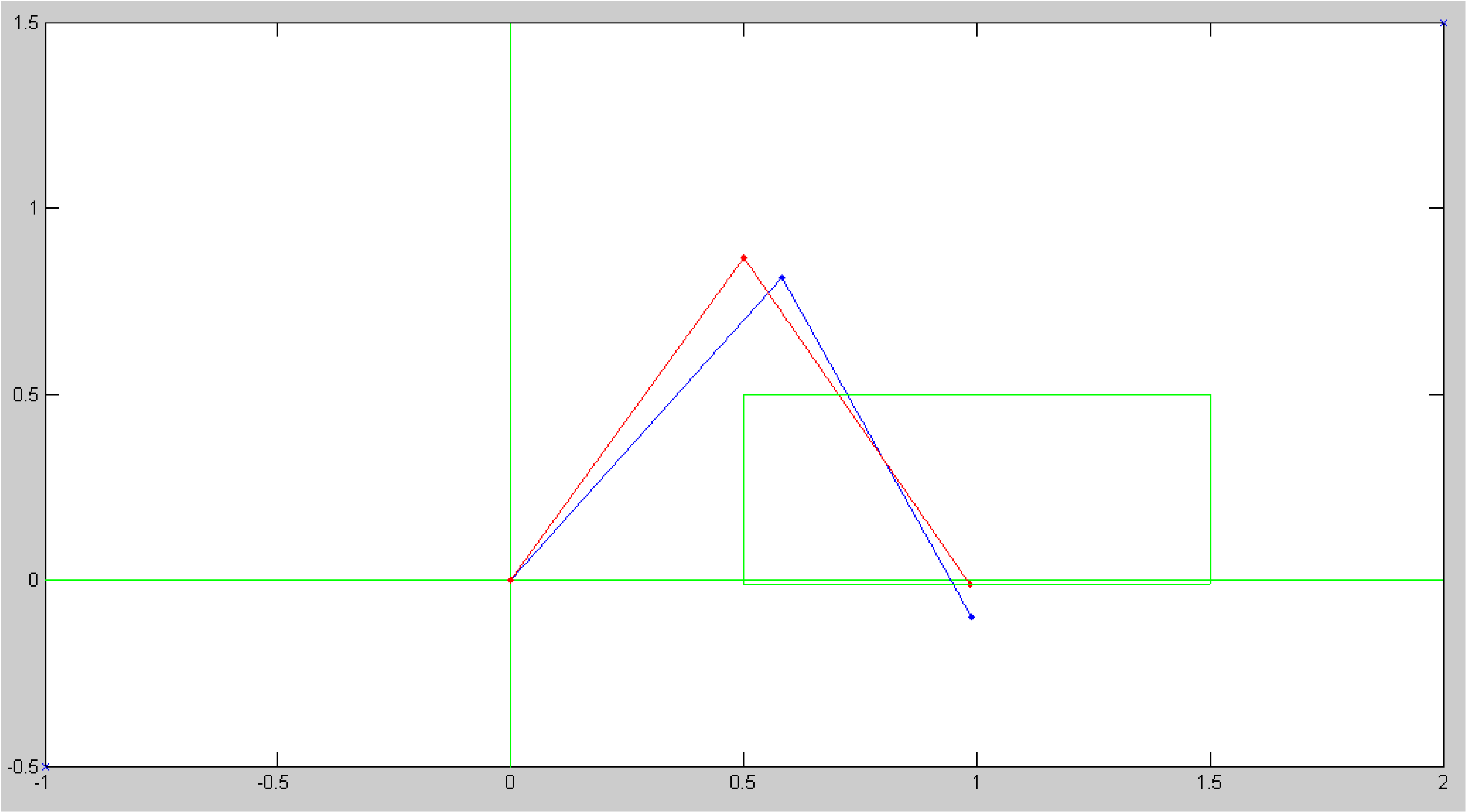
To get the best of both worlds, change the control law on the fly:

- When not in contact with the floor, use the PID position control law
- When you *are* in contact with the floor, use

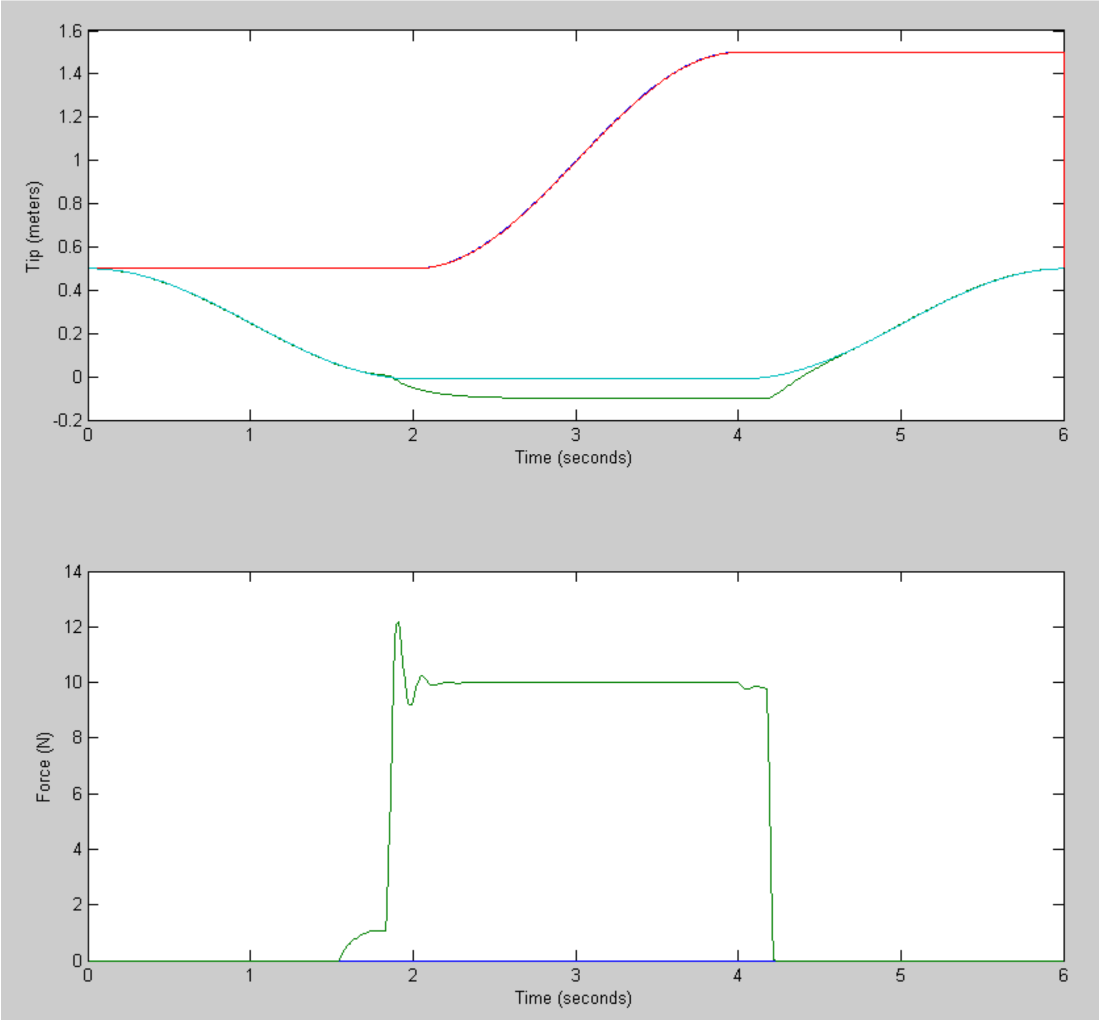
  PID position control in the X direction

  I force control in the Y direction

Simulation Results (RR_XY_Floor_Control_v3.txt)



Tracking Motion: Y2ref < 0 so the robot is using force control



Resulting tip position and tip force. Note that while in contact the tip force is held at 10N as desired.

```
P0 = [0.5; 0.5];
P1 = [0.5; -0.01];
P2 = [1.5; -0.01];
P3 = [1.5; 0.5];
P4 = P0;

t = [0:0.01:1.99];
a = t/2;
a = (1 - cos(pi*a))/2;

% for a step change in poistion, add the following line
%a = 1*(a>0);

X1 = P0*(1-a) + P1*a;
X2 = P1*(1-a) + P2*a;
X3 = P2*(1-a) + P3*a;
X4 = P3*(1-a) + P4*a;
Xr = [X1, X2, X3, X4];
TIP = Xr;



% tip velocity ( used for feedforward control )
dXr = 0*Xr;
for i=2:length(Xr)-1
   dXr(:,i) = ( Xr(:,i+1) - Xr(:,i-1) ) / 0.02;
   end

% tip acceleration (used for feedforward control )
ddXr = 0*Xr;
for i=2:length(Xr)-1
   ddXr(:,i) = ( dXr(:,i+1) - dXr(:,i-1) ) / 0.02;
   end

Q = InverseRR( Xr(:,1) ) ;
dQ = [0; 0];
t = 0;
dt = 0.001;
Fy = 0;

% Start the simulation (dt = 0.001 for stability concerns)
Xq = [];
Tq = [];
Ff = [];

for i=1:length(Xr)
   Qr = InverseRR(Xr(:,i));
```

```
    for j=1:10
       X = [ cos(Q(1)) + cos(Q(1)+Q(2));
             sin(Q(1)) + sin(Q(1)+Q(2)) ];
       J = [ -(sin(Q(1)) + sin(Q(1)+Q(2))), -sin(Q(1)+Q(2)) ;
              (cos(Q(1)) + cos(Q(1)+Q(2))), cos(Q(1)+Q(2)) ];
       dX = J * dQ;

% Control Law and Feedforward Terms
       Facc = ddXr(:,i);
       Fpid = 100*(Xr(:,i) - X) + 20*(dXr(:,i) - dX );

% Floor ( spring with k = 100 )
       Ffloor = [ 0 ; 100*(0 - X(2))  +  20*(0 - dX(2)) ];
       Ffloor = max(0, Ffloor);

% Control the force to 10N if Y2r < 0
       if (Xr(2,i) < 0)
          dFy = 100*(10 - Ffloor(2));
          Fy = Fy + dFy*dt;
       else
          Fy = 0;
          end


 % gravity
       Tg = -9.8 * [ 2*cos(Q(1)) + cos(Q(1) + Q(2));  cos(Q(1) + Q(2)) ];

       T = (J') * ( Fpid + Facc + Ffloor - [ 0 ; Fy ] ) - Tg;

       ddQ = RRDynamics(Q, dQ, T);
       dQ = dQ + ddQ * dt;
       Q = Q + dQ*dt;
       t = t + dt;
       end

    RR(Q, Qr, TIP);

    Xq = [Xq, X];
    Tq = [Tq, T];
    Ff = [Ff, Ffloor];
    end

 t = [1:length(Xr)] * 0.01;
 t = min(t,6);
 clf
 subplot(211)
 plot(t,Xq,t,Xr);
```

```
xlabel('Time (seconds)');
ylabel('Tip (meters)');
subplot(212)
plot(t,Ff);
xlabel('Time (seconds)');
ylabel('Force (N)');
```