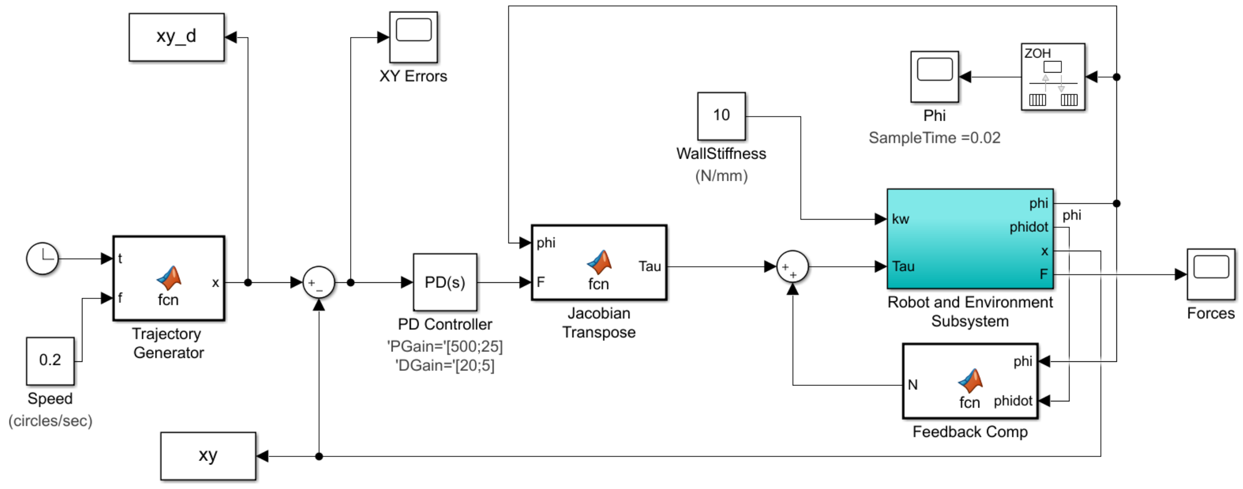
ME EN 6230
Problem Set 9
Ryan Dalby

Note: All simulations were run with a trajectory speed of 0.2 circle/s with a stop time of 10 seconds.

1. Stiffness/Damping Control

1.1. Model



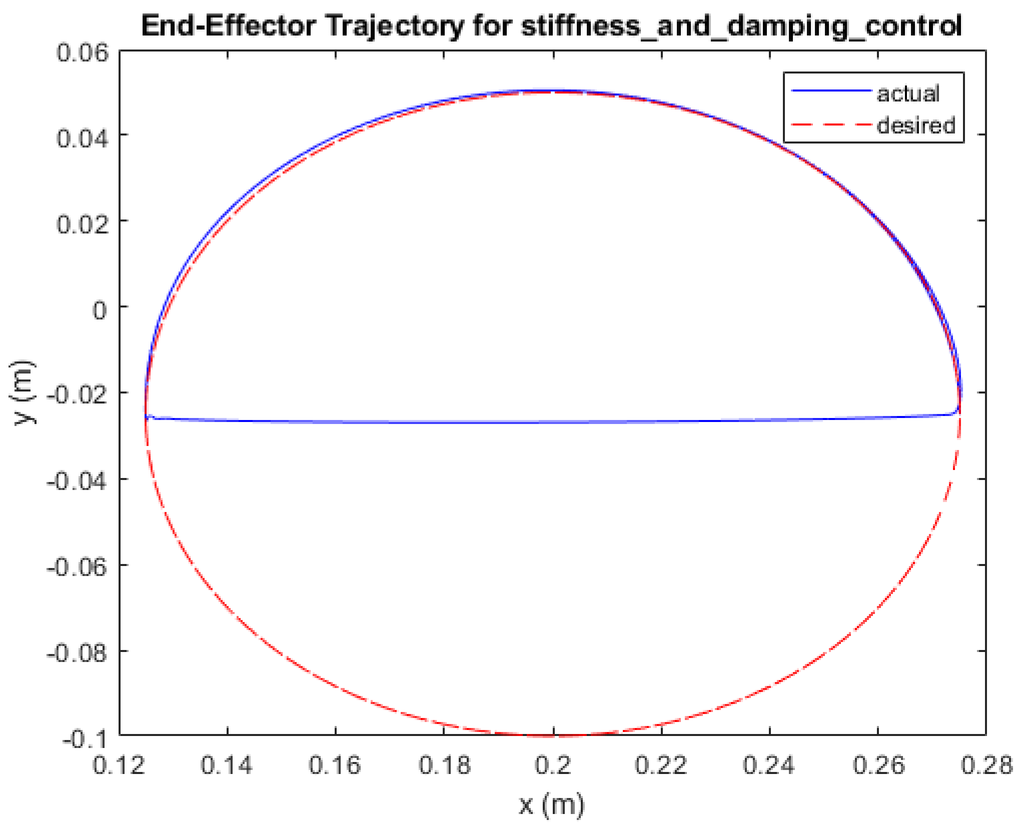
Parameter values: K_x=500, B_x=20, K_y=25, B_y=5

1.2. Code

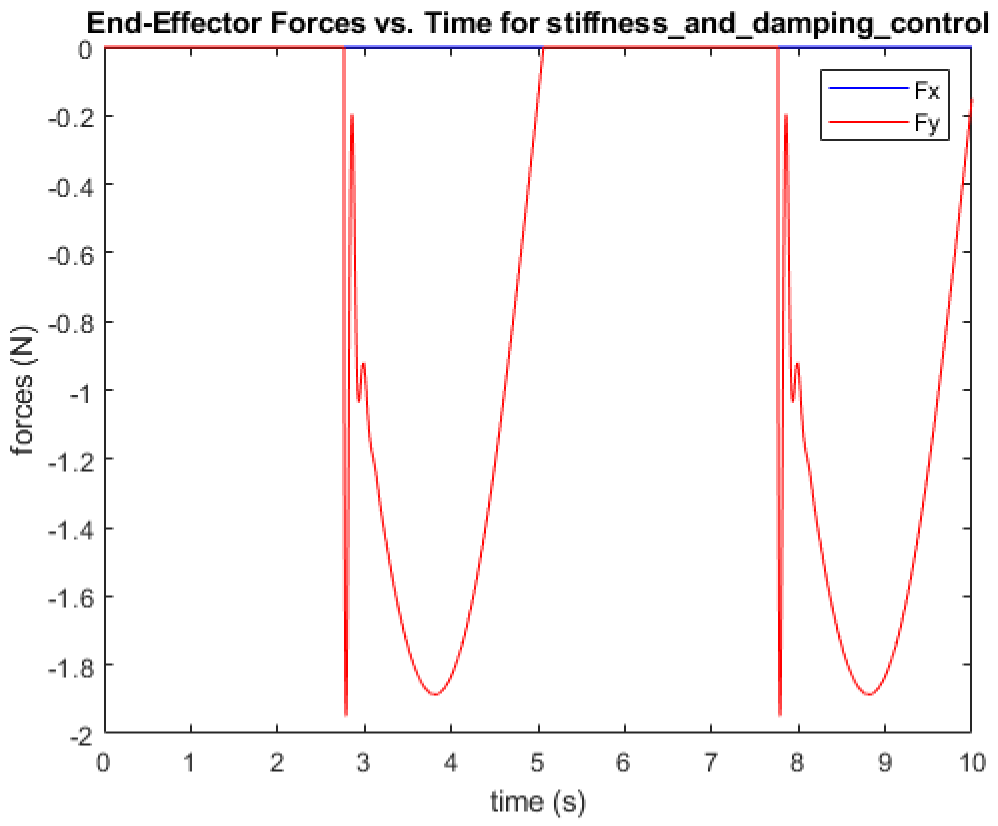
No additional MATLAB function blocks were needed for this controller that were not already described by the template or above.

1.3. Soft Wall (k_w=1N/mm)

1.3.1. End-Effector X-Y Trajectory

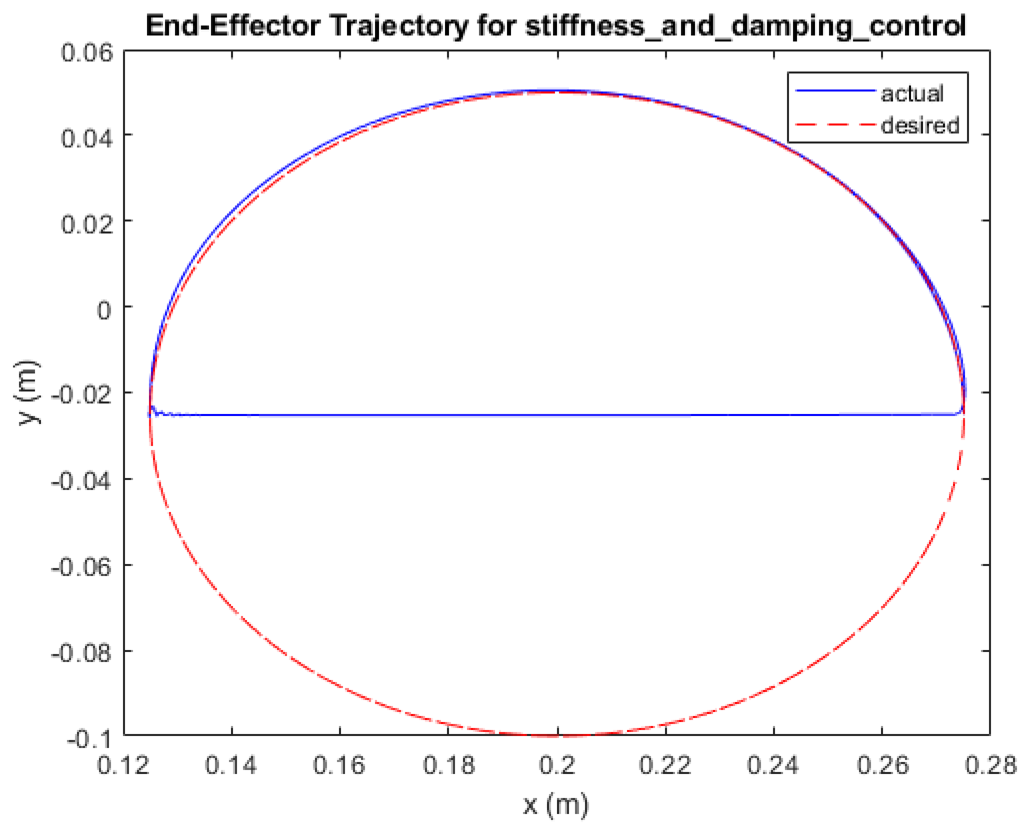


1.3.2. End-Effector Forces

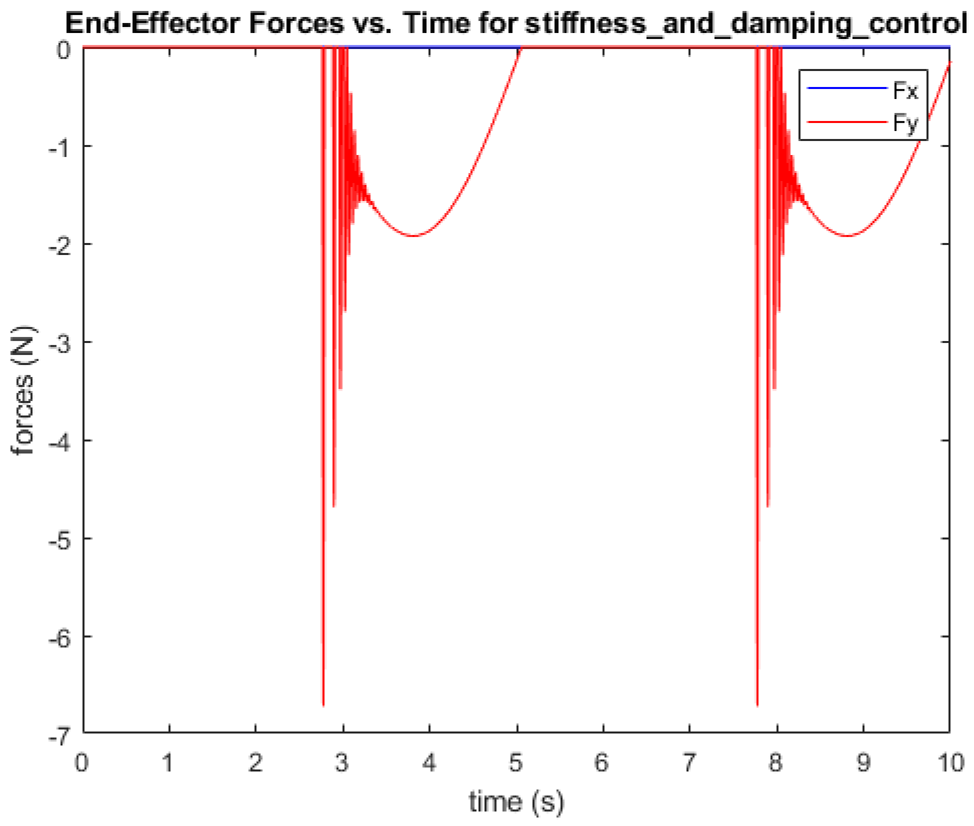


1.4. Hard Wall (k_w=10N/mm)

1.4.1. End-Effector X-Y Trajectory



1.4.2. End-Effector Forces



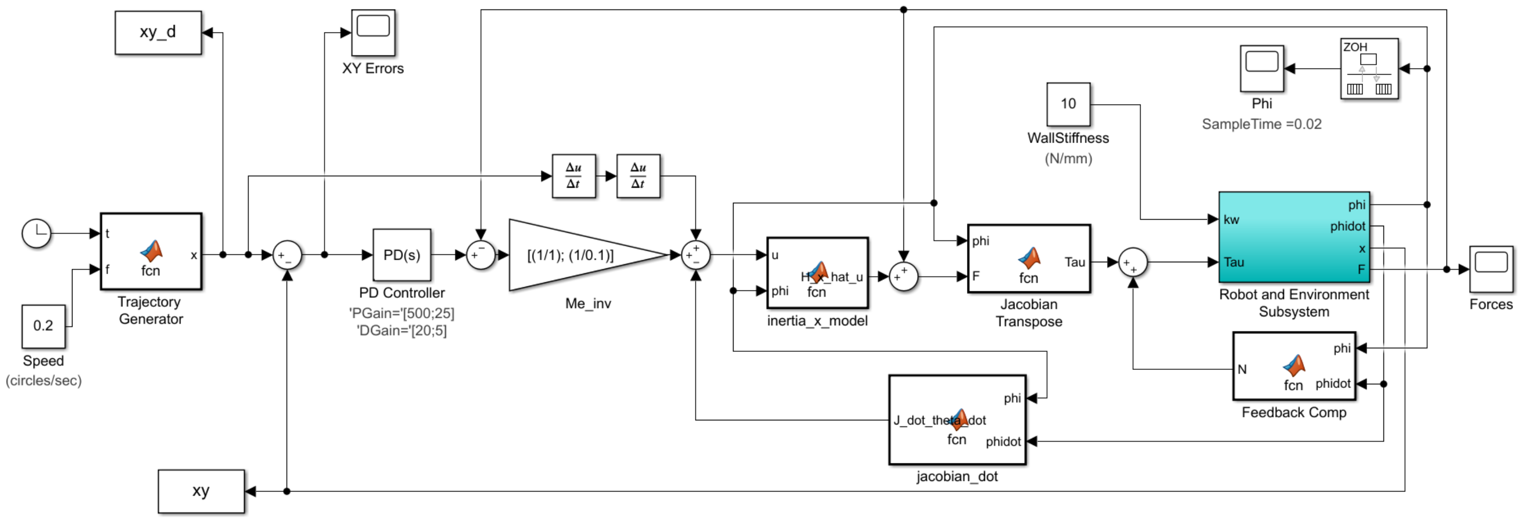
1.5. Analysis

For stiffness and damping control alone it was possible to reduce the maximum external force to under 2N for a soft wall and still have acceptable tracking. For a hard wall, parameter values that reduce the maximum external force under 3N and still maintain acceptable tracking performance could not be found. Both soft and hard walls exhibit similar steady state force behavior at under 2N. Note that lowering the K_y gain from the chosen values began to degrade tracking performance dramatically.

As compared to the other controllers which use force sensors in some capacity, stiffness control requires no force feedback and thus can be implemented cheaply. Its performance is acceptable for soft walls but suffers with hard walls because of its inherent inability to anticipate the force interaction like the other controllers can. Tracking performance is acceptable but this controller will suffer with disturbances because of the low PD gain values used (i.e. K_y and B_y).

2. Full Impedance Control with Stiffness/Damping/Inertia

2.1. Model



Parameter values: $K_x=500$, $B_x=20$, $M_x=1$, $K_y=25$, $B_y=5$, $M_y=0.1$

2.2. Code

inertia_x_model block:

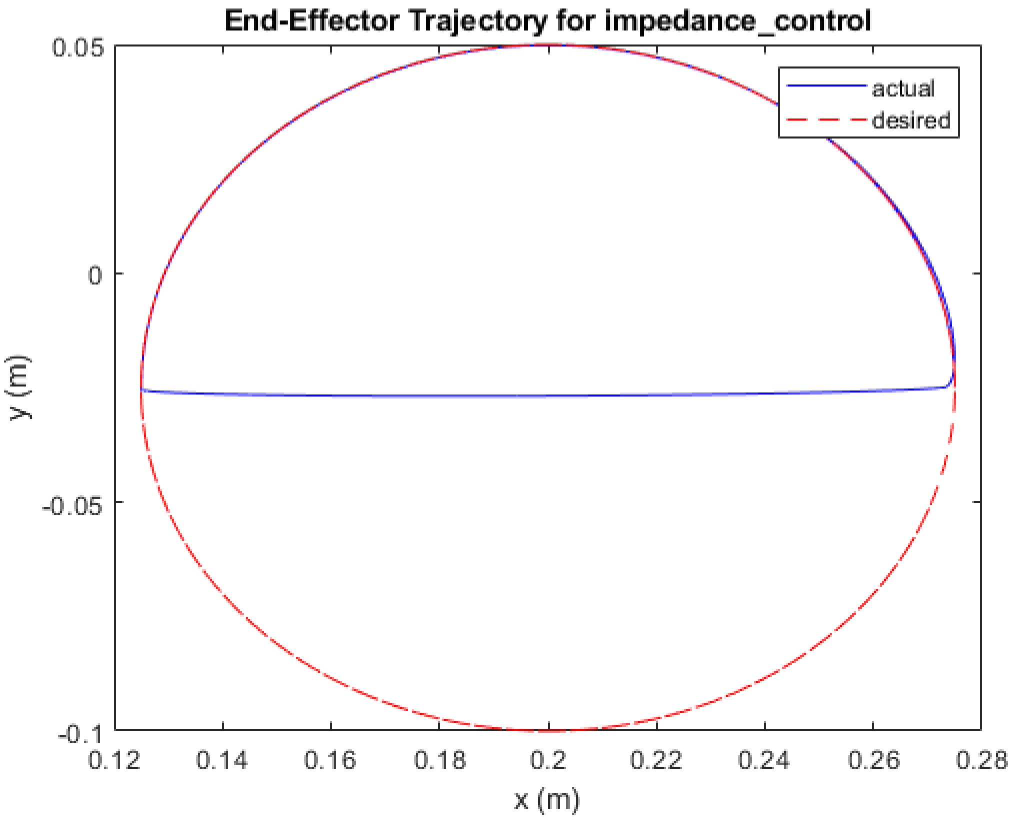
```
function H_x_hat_u = fcn(u,phi)
% This block supports an embeddable subset of the MATLAB language.
% See the help menu for details.

a1 = 0.15;  % link 1 length
a2 = 0.15;  % link 2 length
m1 = 0.092;  % link 1 mass
m2 = 0.077; % link 2 mas
r01 = 0.062; % link 1 center of mass
r12 = 0.036; % link 2 COM
I1 = 0.64e-3;  % link 1 inertia
I2 = 0.30e-3;  % link 2 inertia
Jm1 = 0.65e-6; % motor inertias
Jm2 = 0.65e-6;
b1 = 3.1e-6; % viscous damping constants
b2 = 3.1e-6;
c1 = 0.0001; % coulomb friction constants
c2 = 0.0001;
g = 9.8; % gravitational constant
N1 = 70; % gear ratios
N2 = 70;

a1 = 0.15;  % link 1 length
a2 = 0.15;  % link 2 length
% Describe combined relationship of jacobian from phi to x using jacobian
% (Combines transmission jacobian from phi to theta and manipulator
% jacobian from theta to x)
J11 = -a1*sin(phi(1));
J12 = -a2*sin(phi(2));
J21 = a1*cos(phi(1));
J22 = a2*cos(phi(2));
J = [J11 J12; J21 J22];

H11 = N1^2*Jm1 + I1 + m2*a1^2;
H12 = a1*r12*m2*cos(phi(2)-phi(1));
H21 = H12;
H22 = N2^2*Jm2 + I2;

H_hat = [H11 H12; H21 H22]; % inertia matrix
H_x_hat = inv(transpose(J)) * H_hat * inv(J);
H_x_hat_u = H_x_hat * u;
```

Jacobian_dot block:

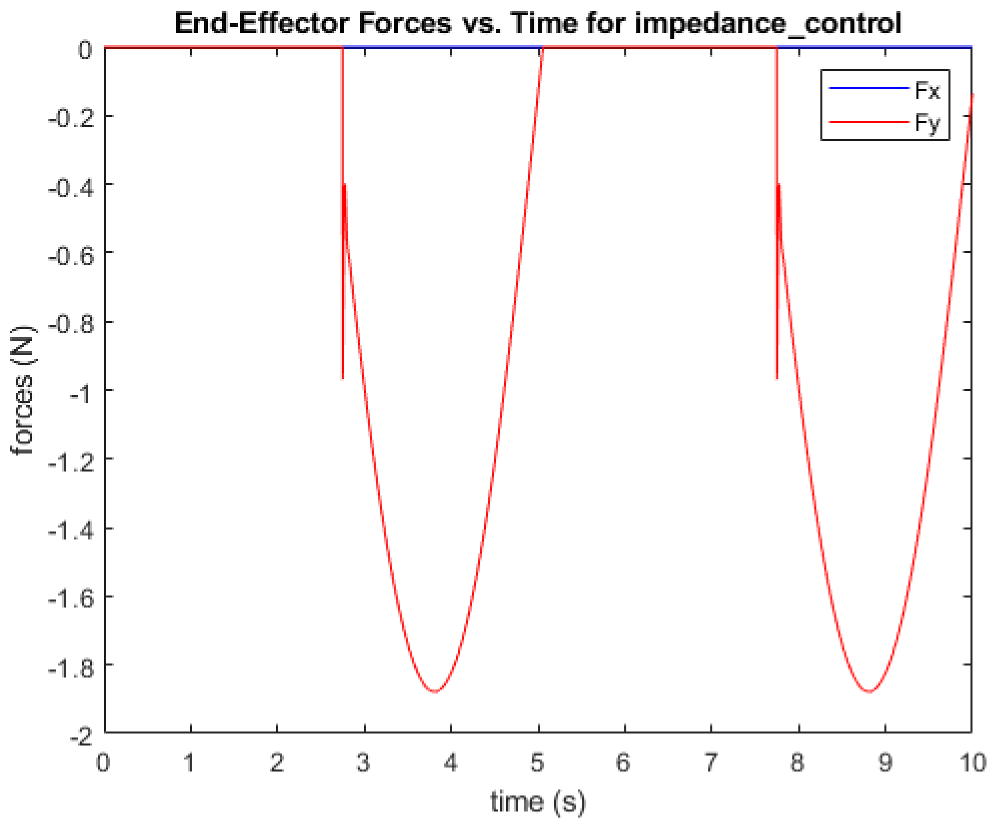
```
function J_dot_theta_dot = fcn(phi, phidot)
% This block supports an embeddable subset of the MATLAB language.
% See the help menu for details.

a1 = 0.15;  % link 1 length
a2 = 0.15;  % link 2 length

J11_dot = -a1*cos(phi(1))*phidot(1);
J12_dot = -a2*cos(phi(2))*phidot(2);
J21_dot = -a1*sin(phi(1))*phidot(1);
J22_dot = -a2*sin(phi(2))*phidot(2);

J_dot = [J11_dot J12_dot; J21_dot J22_dot];

J_dot_theta_dot = J_dot * phidot;
```

2.3. Soft Wall (k_w=1N/mm)

2.3.1. End-Effector X-Y Trajectory

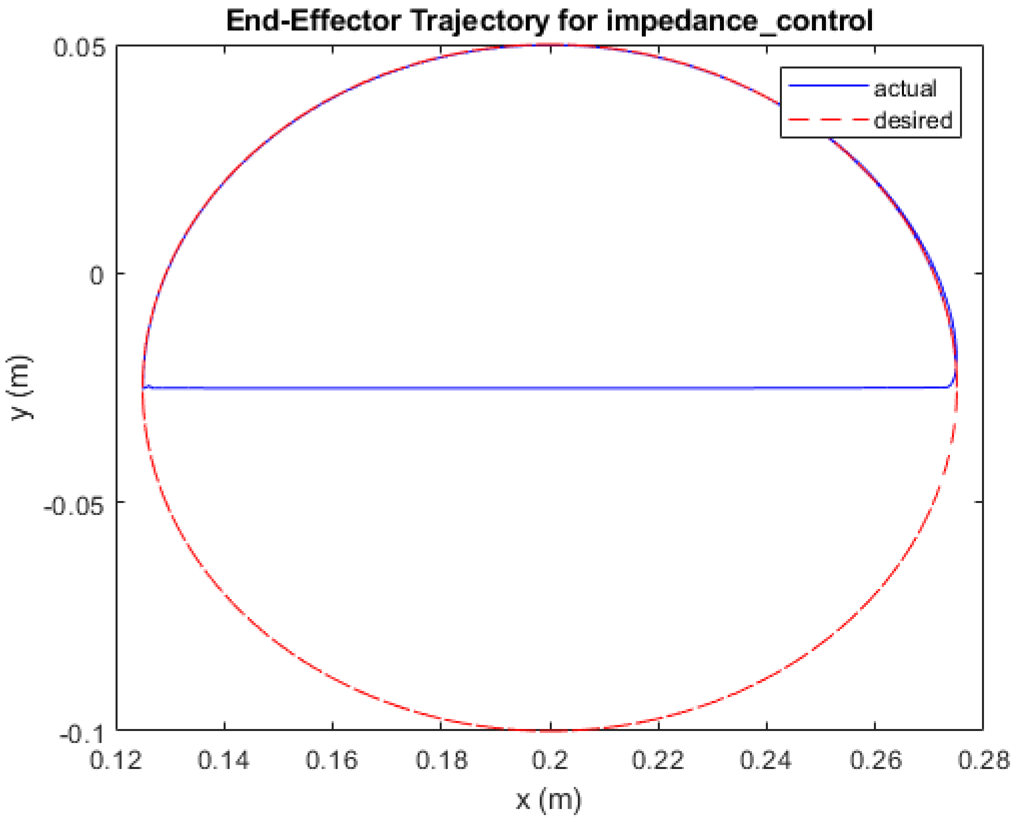


2.3.2. End-Effector Forces

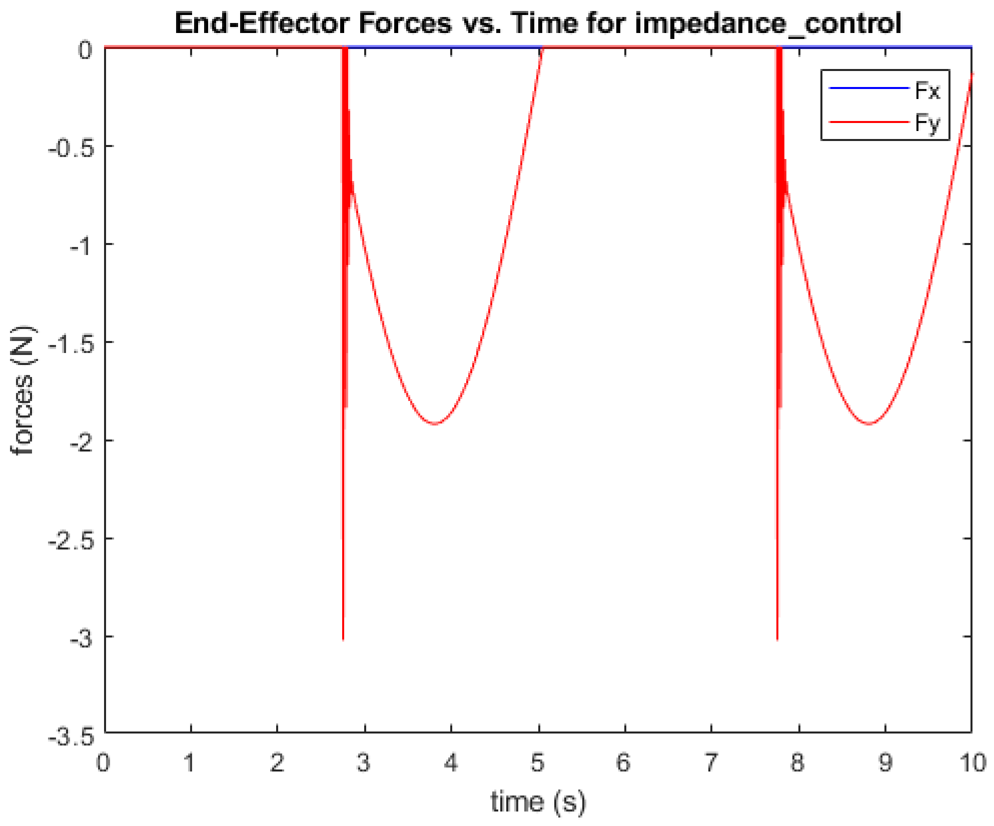


2.4. Hard Wall (k_w=10N/mm)

2.4.1. End-Effector X-Y Trajectory



2.4.2. End-Effector Forces



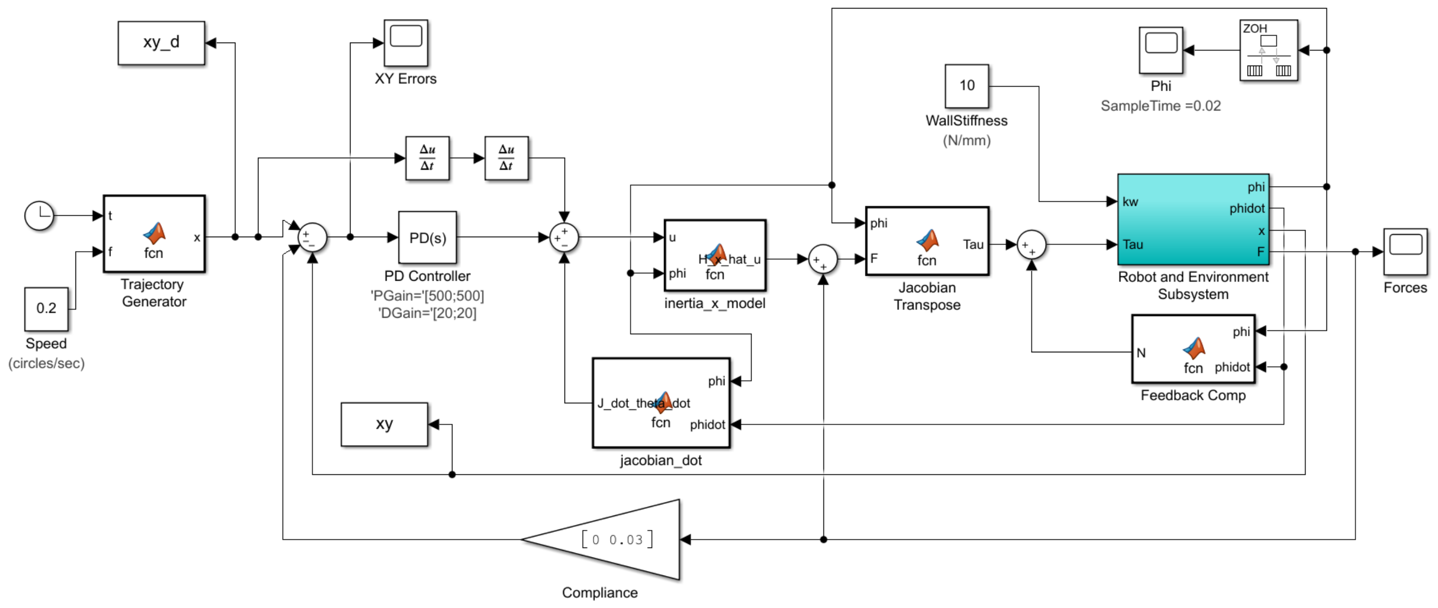
2.5. Analysis

For impedance control it was possible to get controllers which reduce the maximum external force to 3N or under for both a soft and hard wall while still maintaining a good trajectory. The steady state forces are also reasonable for both the hard and soft walls at under 2N. The ability to reduce the effective mass of the robot helps combat the impact force of a hard wall.

As compared to stiffness control impedance control clearly provides better external force protection while still maintaining tracking as good as stiffness control. The primary tradeoff with impedance control is the necessity of a force sensor in order to get an accurate measurement of the acceleration and provide the ability to “program” the robot mass. Another tradeoff is tracking performance which is acceptable but will suffer with disturbances because of the low PD gain values used (i.e. K_y and B_y). This controller may also suffer because of the causality of the control law which will not provide a reaction until friction is overcome and a change in position is registered. The flip-side advantage to this is that there is more of a guarantee of stability than compliance and admittance control because of the control law.

3. Compliance Control with Inverse Dynamics Control

3.1. Model



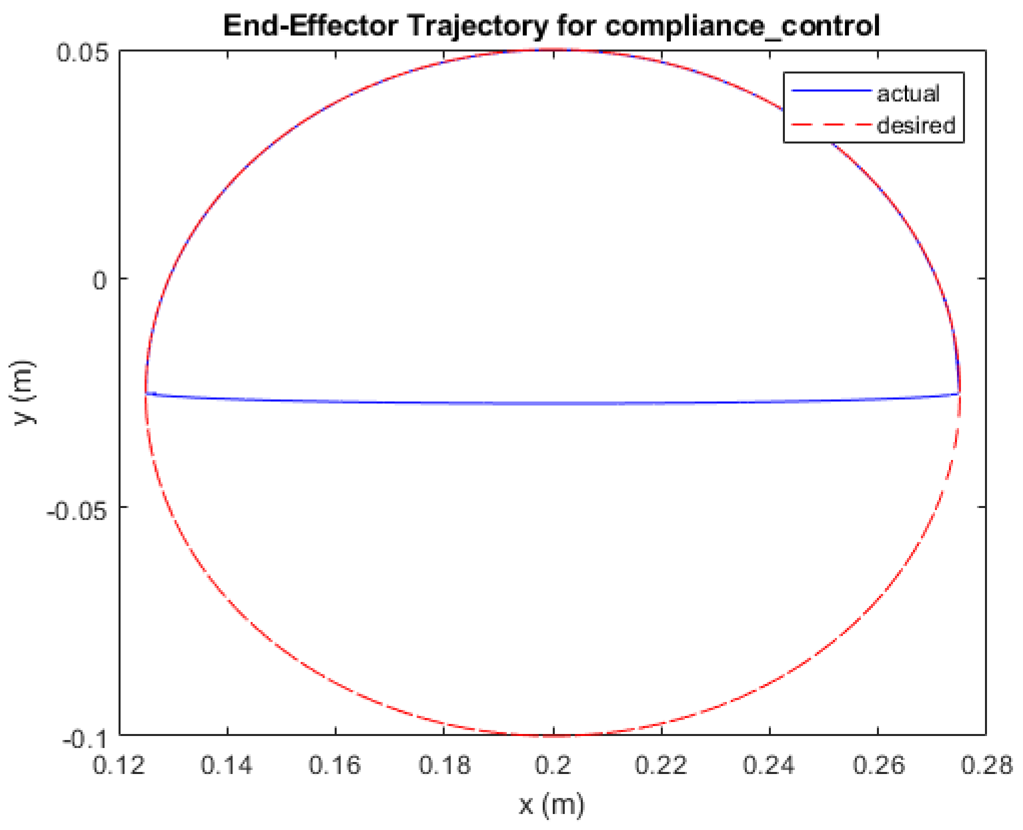
Parameter values: K_x=500, B_x=20, C_x=0, K_y=500, B_y=20, C_y=0.03

3.2. Code

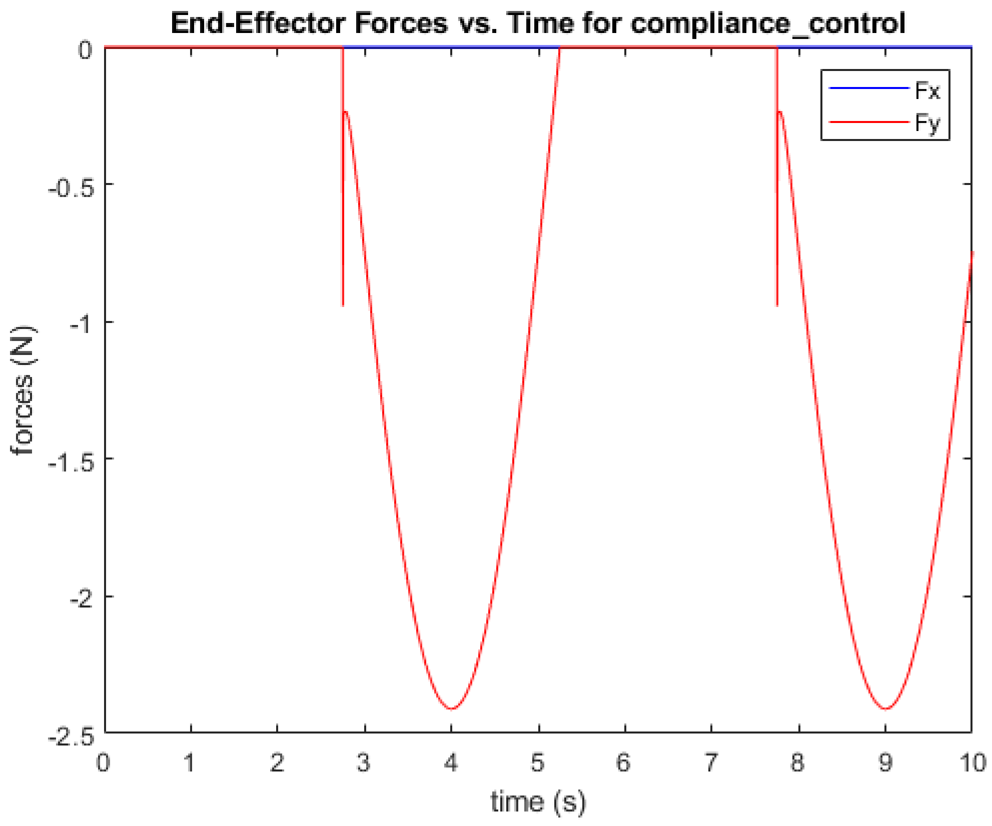
No additional MATLAB function blocks were needed for this controller that were not already described by the template or above.

3.3. Soft Wall (k_w=1N/mm)

3.3.1. End-Effector X-Y Trajectory

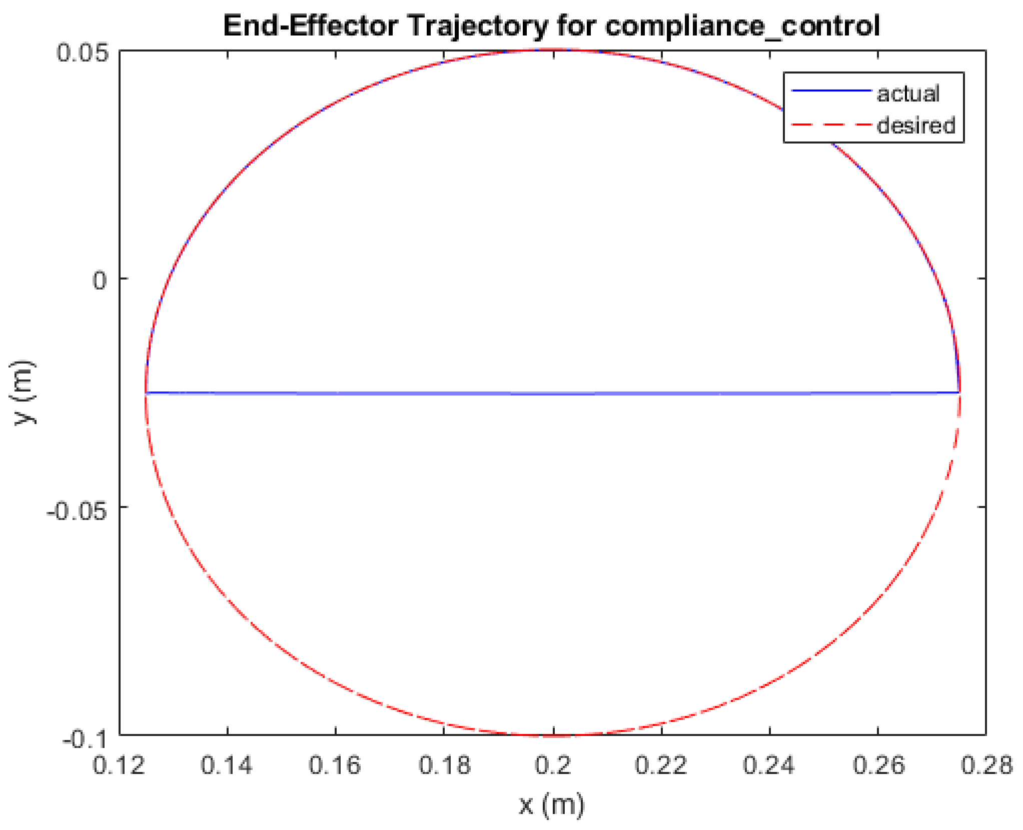


3.3.2. End-Effector Forces

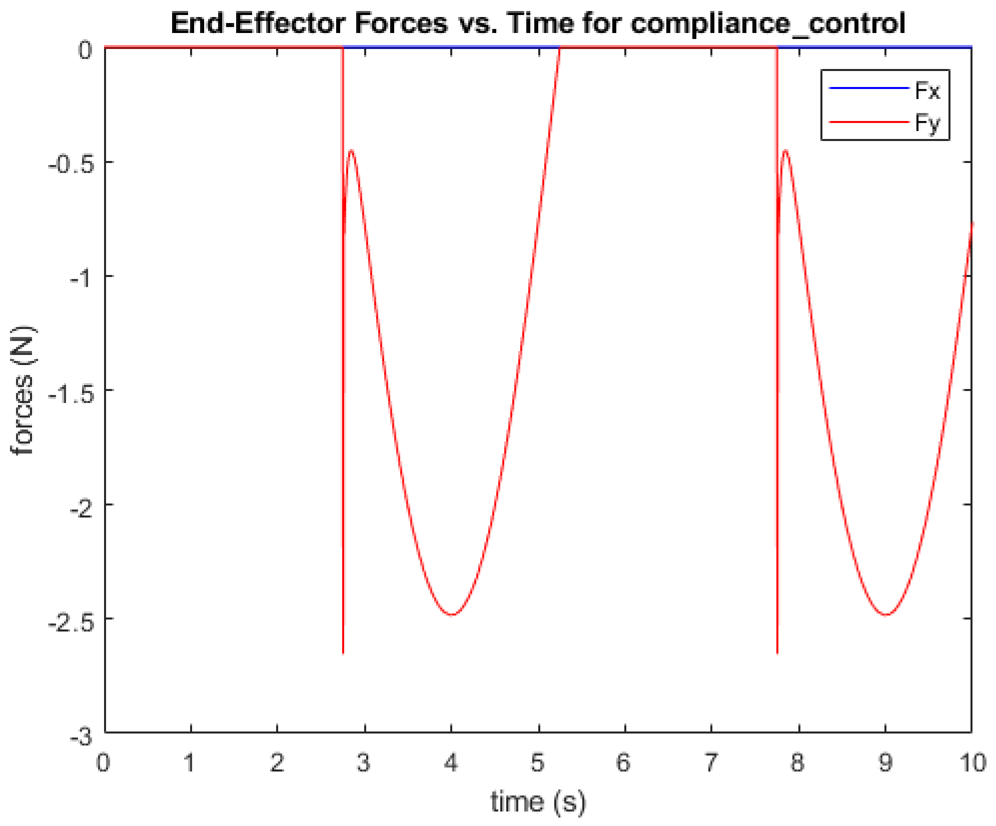


3.4. Hard Wall (k_w=10N/mm)

3.4.1. End-Effector X-Y Trajectory



3.4.2. End-Effector Forces



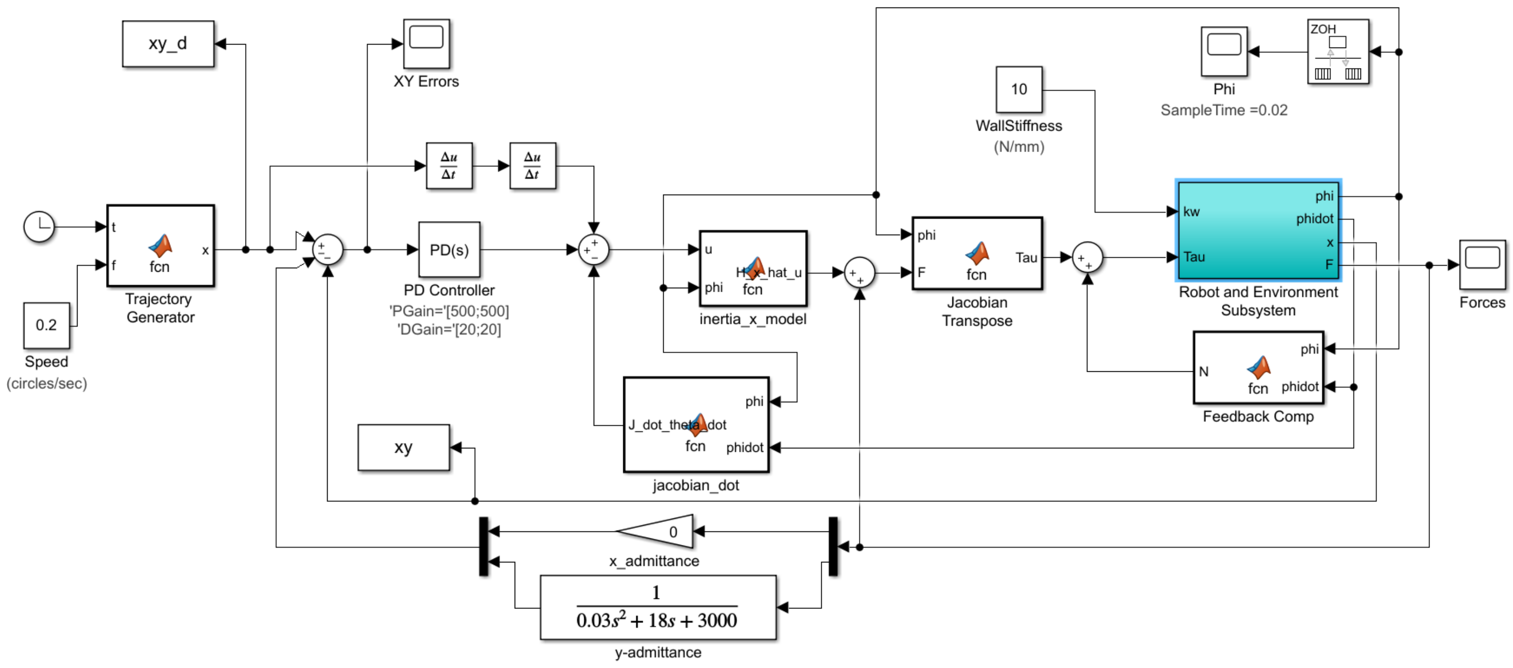
3.5. Analysis

For compliance control it was possible to find parameter values which limited the maximum external force on the robot to under 3N for both hard and soft walls. Compliance control also attained very effective trajectory tracking, especially for a hard wall.

Comparing compliance control to the other controllers it is clear that the high PD gains (i.e. K_y and B_y) resulted in much better tracking performance than stiffness and impedance control which had soft PD gains as a result of using the gains to program stiffness and damping. A clear disadvantage of this controller is the necessity of a force controller which is directly used to measure the force and augment it into a change of position which is fed back to adjust the trajectory. Another disadvantage is that compliance control does not give control over apparent damping and inertia like compliance control and admittance control does. Another possible disadvantage is that changing the compliance values just slightly can cause the robot to go unstable. This is easier than it is to get stiffness and impedance control to go unstable.

4. Full Admittance Control with Stiffness/Damping/Inertia

4.1. Model



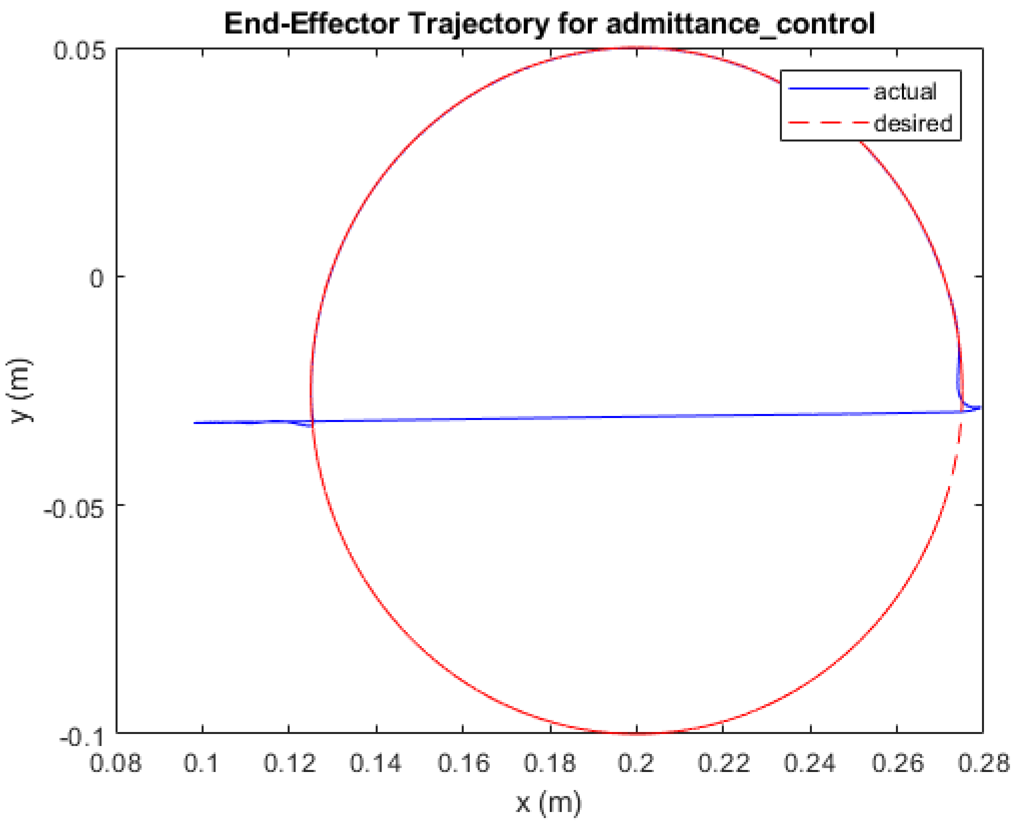
Parameter values: K_x=500, B_x=20, A_x(s)=0, K_y=500, B_y=20, A_y(s)=1/(0.03s^2+18s+3000)

4.2. Code

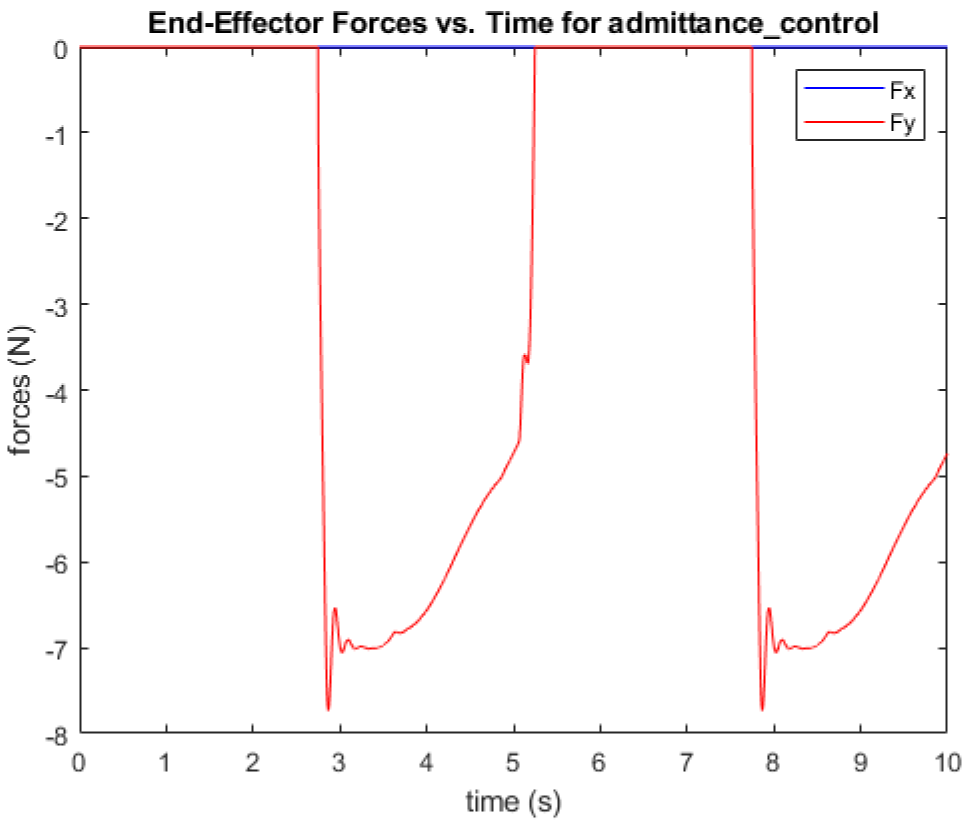
No additional MATLAB function blocks were needed for this controller that were not already described by the template or above.

4.3. Soft Wall (k_w=1N/mm)

4.3.1. End-Effector X-Y Trajectory

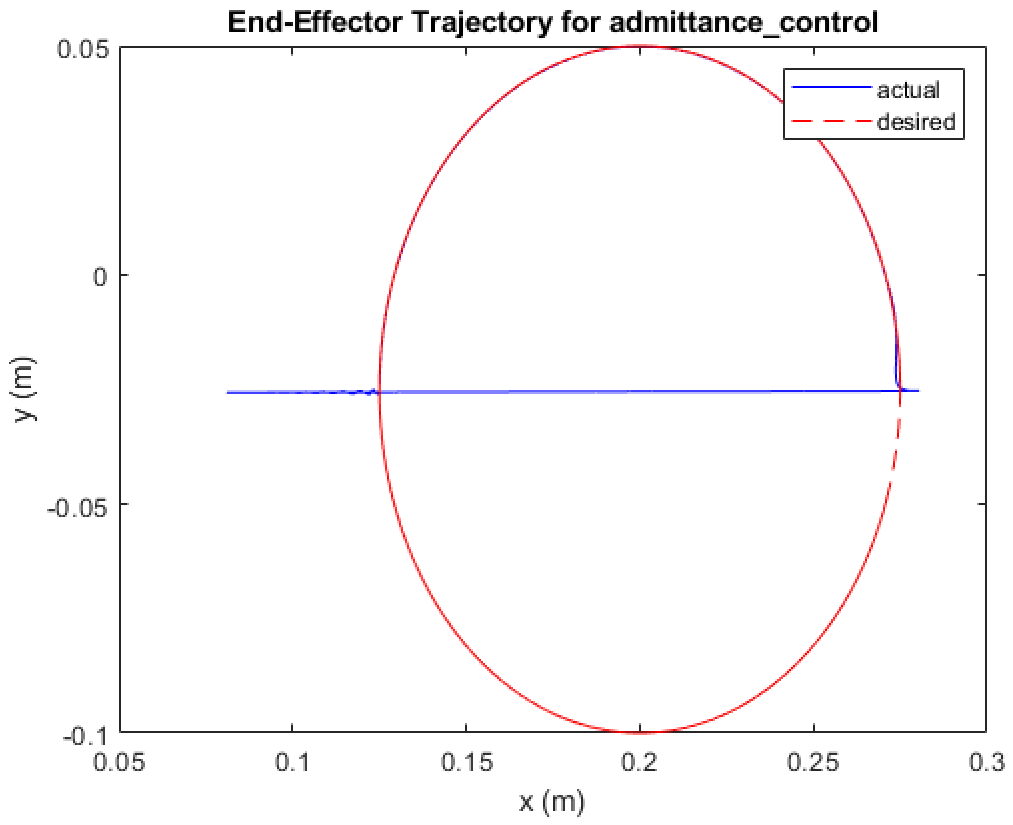


4.3.2. End-Effector Forces

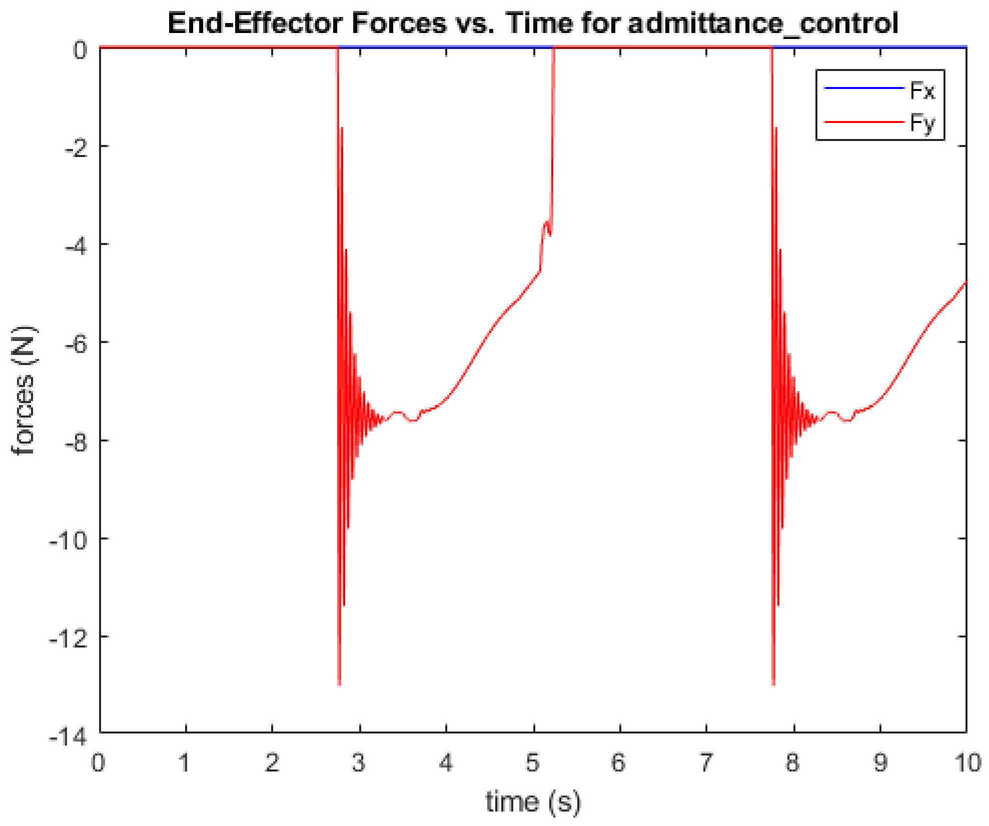


4.4. Hard Wall (k_w=10N/mm)

4.4.1. End-Effector X-Y Trajectory



4.4.2. End-Effector Forces



4.5. Analysis

Note: Technically limiting the maximum external force to under 3N was possible for the soft wall using $A_y(s)=1/(0.0001s^2+0.033s+33)$ but the solution entirely relies on consistent saturation of the amplifier and is unstable without the amplifier saturation. Thus these parameter values were disregarded.

For admittance control it was not possible to find parameter values that limited the maximum external force on the robot to under 3N for either the soft or hard wall. An acceptable result for this controller was given in the model description. Trajectory tracking was good when not in contact with the wall because of high PD gains (i.e. K_y and B_y). Overall, admittance control was very hard to tune and get stable tracking.

Comparing admittance control to the other controllers it is clear that the high PD gains (i.e. K_y and B_y) resulted in much better tracking performance than stiffness and impedance control when not in contact with the wall. A disadvantage of this controller is the necessity of a force controller which is directly used to measure the force and augment it into a change of position which is fed back to adjust the trajectory. The main disadvantage of admittance control is how unstable the controller generally is and how wall stiffness can directly affect stability. In the end, the results for admittance control were not great but the controller theoretically could give control over inertia, dampening, and stiffness while giving good tracking and disturbance rejection in a different application.

Appendix- Plotting Code (Used to make simulation plots)

```
%% ME EN 6230 Problem Set 9 Ryan Dalby
% close all;
set(groot, 'DefaultTextInterpreter', 'none') % Prevents underscore from
becoming subscript

% Extract necessary data, will error if the data does not exist
time = xy_errors.time; % s
model_title = extractBefore(xy_errors.blockName, "/");
actual_trajectory = xy; % m
desired_trajectory = xy_d; % m
forces = F.signals.values; % N

% Plot End-Effector Forces vs Time
figure;
plot(time, forces(:,1), 'b-');
hold on;
plot(time, forces(:,2), 'r-');
title(strcat("End-Effector Forces vs. Time for ", model_title));
xlabel("time (s)");
ylabel("forces (N)");
legend("Fx", "Fy");

% Plot End-Effector X-Y Trajectory
figure;
plot(xy(:,1), xy(:,2), 'b-');
hold on;
plot(xy_d(:,1), xy_d(:,2), 'r--');
title(strcat("End-Effector Trajectory for ", model_title));
xlabel("x (m)");
ylabel("y (m)");
legend("actual", "desired");
```