MECH467 Lab 3 Report

Ratthamnoon Prakitpong
#63205165

Abstract:

Continuing from Lab 1 and 2, where we learned to control and design controllers for a linear actuator, we are learning to use those actuators and controllers in simulating a toolpath, comparing simulated paths to experimental paths, and finding errors for those paths in this lab. These tasks will wrap up the labs' purposes: designing a two-axis machine from beginning to end.

Introduction:

A popular way to use linear actuators is in XY tables, each axis driven by a linear actuator. May types of machines in mechanical engineering uses this, including milling machine, CNC, and laser cutter. This lab lays out how to design such machines, covering topics such as toolpath simulation and comparison to experimental toolpaths. We need to generate a trajectory, design an appropriate two-axis controller for toolpath to be executed on, and simulate its performance. After the toolpath is run on a machine, we can compare simulated and experimental toolpaths, and observe the effects of parameters on performance, such as bandwidth and maximum feedrate. A way to quantify comparisons is to compare their tracking and contouring errors. These analysis are laid out in the lab report in this approximate order, to show a real progression on how to design a two-axis machine.

E1.

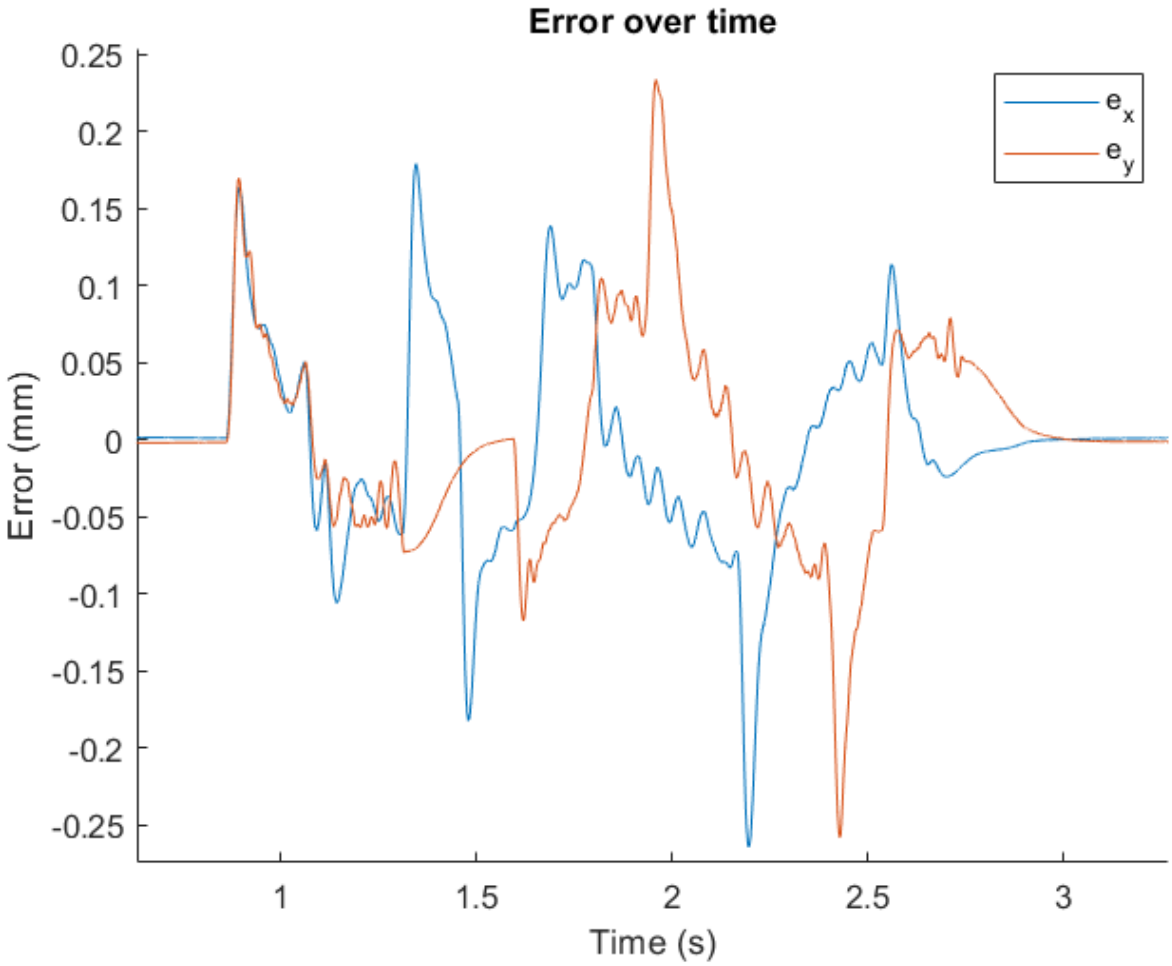


Fig E1: Error over time

E2.

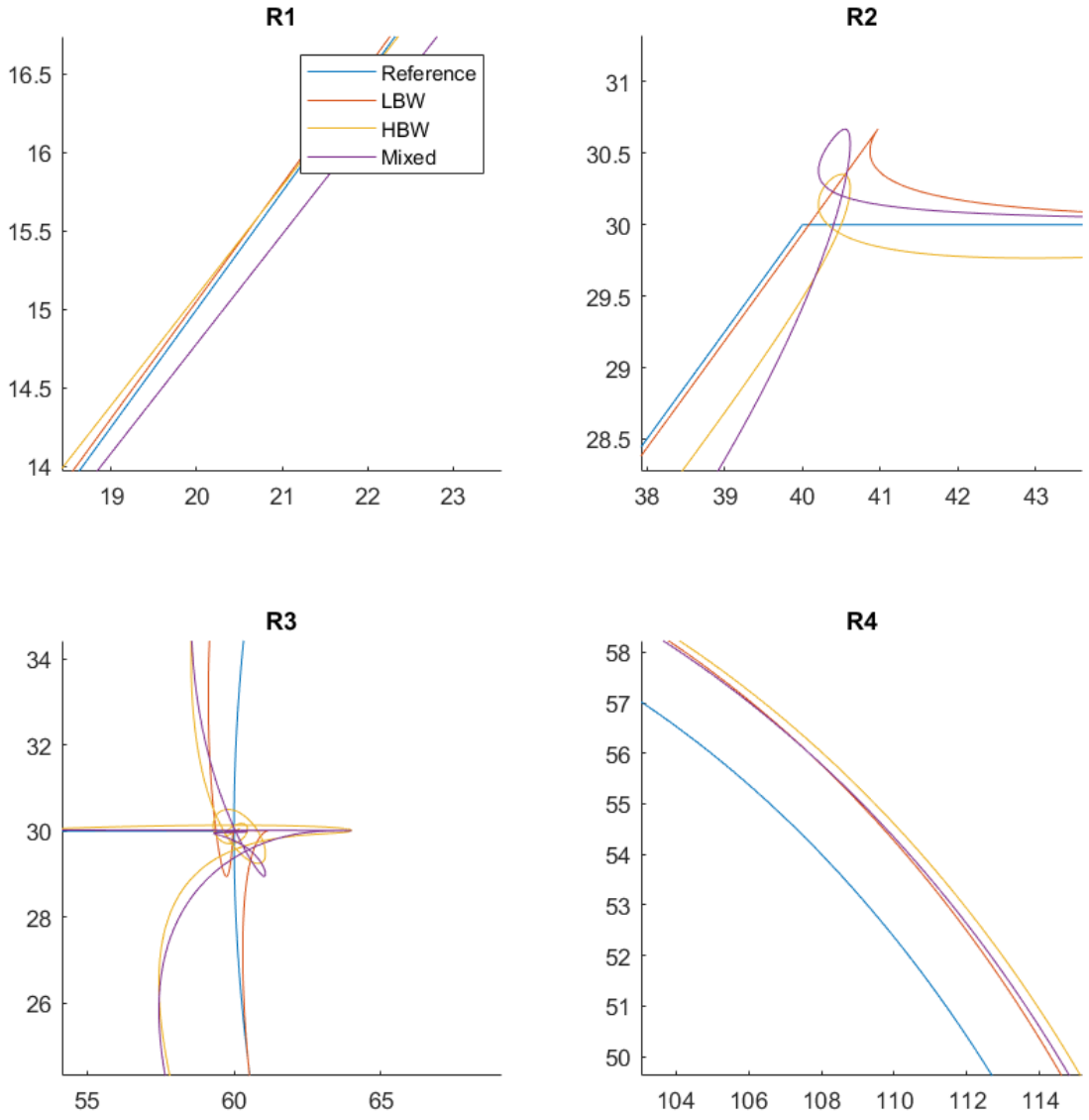


Fig E2: Toolpaths at critical regions

Controller	Location	Reference	Path	Error
LBW	R2	(40, 30)	(40.9694, 30.6684)	1.1775
HBW	R3	(60, 30)	(64.0108, 30.0134)	4.0108
Mixed	R3	(60, 30)	(64.0109, 30.0085)	4.0109

Table E3: Maximum approximate toolpath contour error

Increasing bandwidth increases contour error. Low bandwidth case (20 rad/s) has about a bit less than half the maximum error of high bandwidth case (40 rad/s).

E3.

Time (s)	x_ref	y_ref	x_sim	y_sim	e_x	e_y	e_contour
1.0613	15.984	11.988	14.9992	11.3087	0.9848	0.6793	1.1963
1.0614	16	12	15.0152	11.3207	0.9848	0.6793	1.1963
1.0615	16.016	12.012	15.0312	11.3327	0.9848	0.6793	1.1964
1.0616	16.032	12.024	15.0472	11.3447	0.9848	0.6793	1.1964
1.0617	16.048	12.036	15.0632	11.3567	0.9848	0.6793	1.1964
1.0618	16.064	12.048	15.0792	11.3687	0.9848	0.6793	1.1964
1.0619	16.08	12.06	15.0952	11.3807	0.9848	0.6793	1.1964
1.062	16.096	12.072	15.1112	11.3927	0.9848	0.6793	1.1963
1.0621	16.112	12.084	15.1273	11.4047	0.9847	0.6793	1.1963
1.0622	16.128	12.096	15.1433	11.4167	0.9847	0.6793	1.1963
1.0623	16.144	12.108	15.1593	11.4288	0.9847	0.6792	1.1962
1.0624	16.16	12.12	15.1754	11.4408	0.9846	0.6792	1.1961

Table E3: Sample section of calculated errors

Maximum contour error calculated is 1.1964 mm, which is also shown in the table above.

Calculating the contouring error analytically is not always possible. It is possible in this case because the contour is very simple (a straight line), so contour error can be approximated from tracking error (pythagorean theorem). When the contour is complicated, it may be difficult to write code to analytically judge which point on the reference contour is the simulated contour supposed to be referenced from when calculating for error.

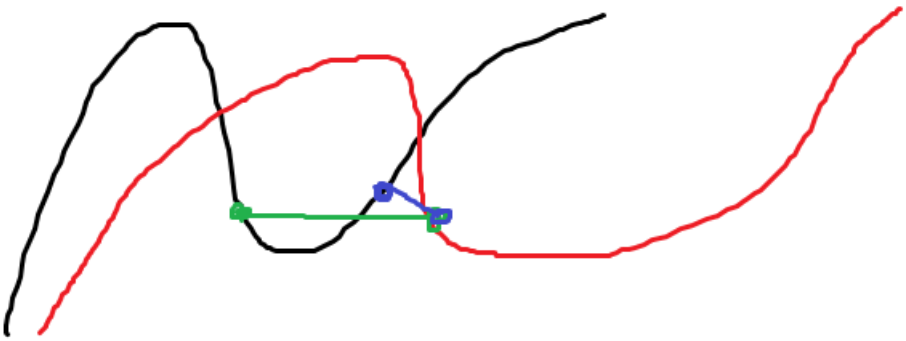


Fig E3: Complex toolpath error calculation

See this figure for an illustration of the previous point made. Done analytically and without human judgement, computer would've drawn the blue line as the contour error (ie shortest path between reference and actual), whereas the actual contour error is the green line.

E4.

Contour error of mismatched dynamics toolpath is similar to maximum error between LBW and HBW controllers. It's not ideal to have mismatched axis dynamics because creating controllers is more difficult. Additionally, it just takes the error that is the maximum between the two bandwidths. Time may be better spent finding ways to design better controllers for a case of matched bandwidth dynamics.

F1.

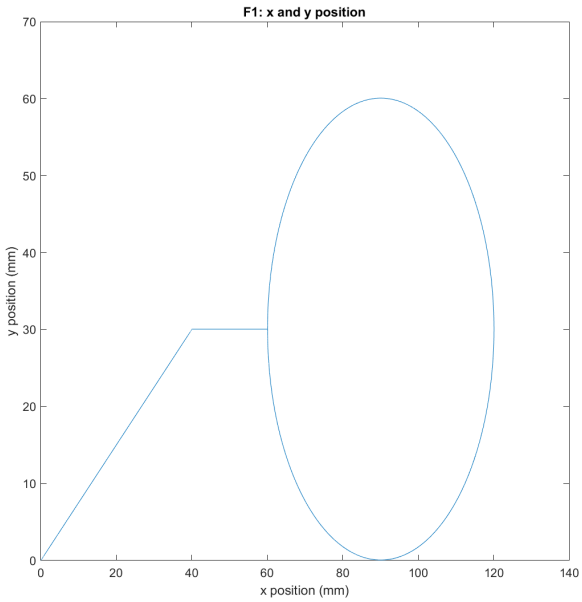


Fig F1.1: X and Y position of toolpath

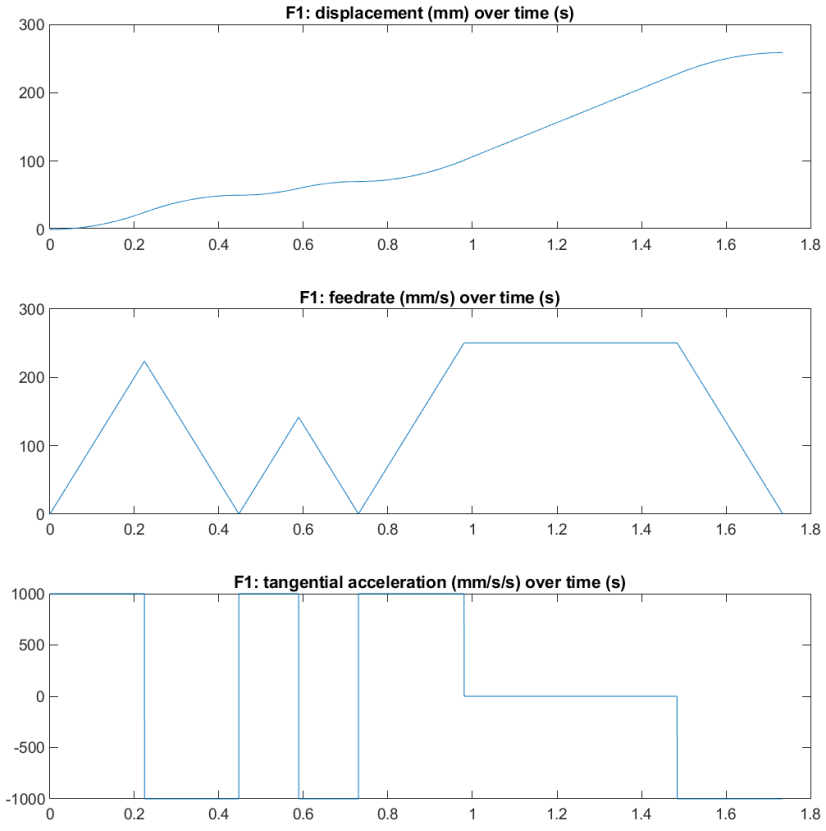


Fig F1.2: Displacement, feedrate, and tangential acceleration of sample trajectory

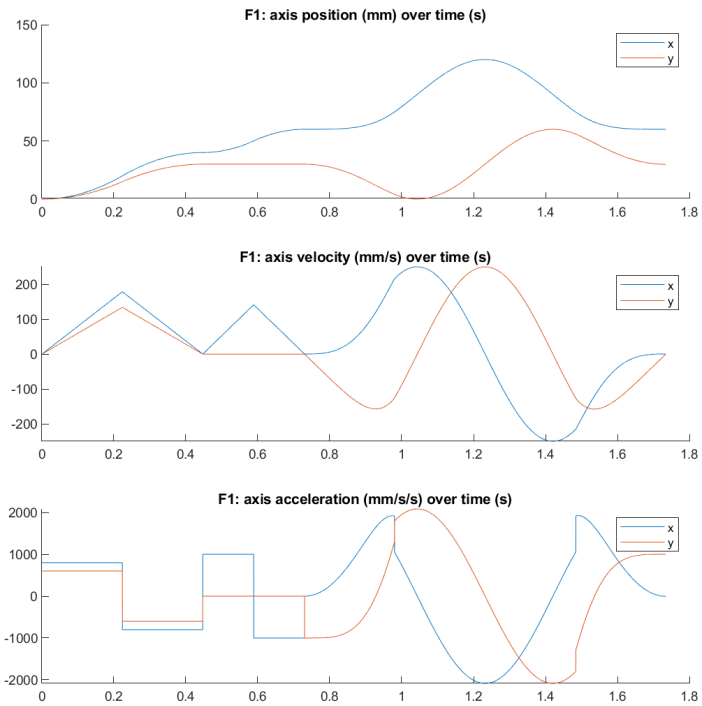


Fig F1.3: Axis position, velocity, and acceleration of sample trajectory

F2.

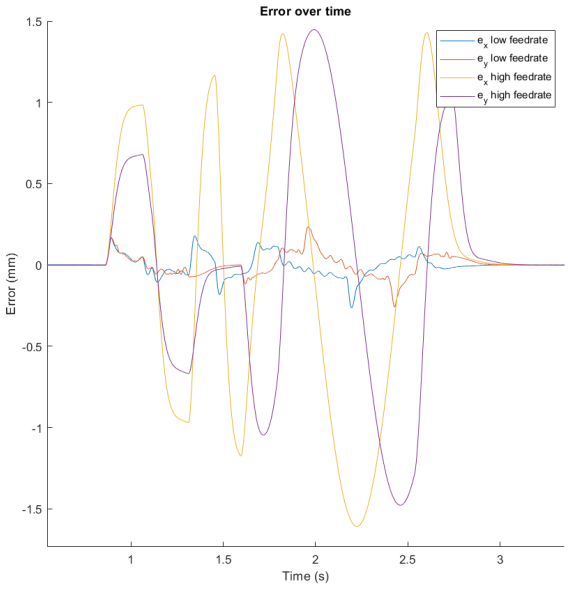


Fig F2: Error over time of low and high feedrate toolpaths

Increasing feedrate likely increases tracking error. However, the difference in error between feedrate changes is too large, which leads me to believe that something may be erroneous. I used my controller and my part-F1 reference toolpath to generate high feedrate toolpath, and given toolpath for low feedrate toolpath.

F3.
For comparison's sake, I'll be comparing my own simulated toolpaths for both low and high feedrate, ignoring given toolpaths. This way, discrepancies are consistent between two toolpaths, and differences that came from differing feedrate can be highlighted. This solves the issue seen in F2.

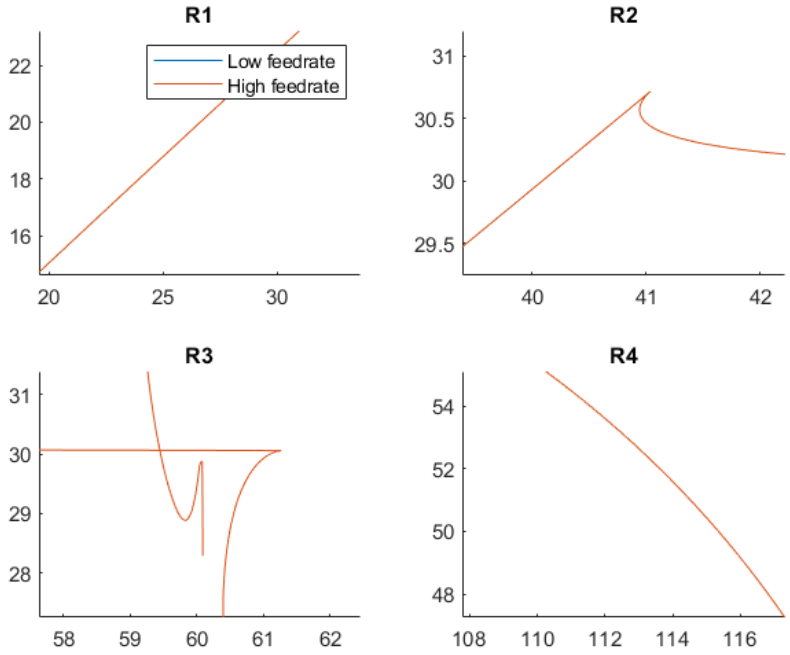


Fig F3: Simulated toolpaths of differing feedrate

While there may be more error when calculated, the toolpaths are not visibly different in terms of contouring accuracy. This makes sense since the increase in feedrate alone may not be enough to noticeably influence error. For example, path from P1 to P3 isn't long enough for tool head to accelerate to maximum feedrate, resulting the same toolpath even though the feedrate is different. Even in sections where it could go faster than earlier specification, the time difference is small.

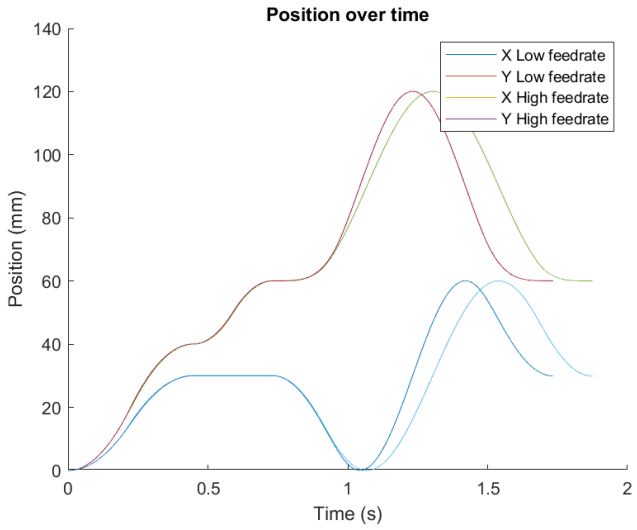


Fig F3.2: Position over time of toolpaths of differing feedrate

G1.

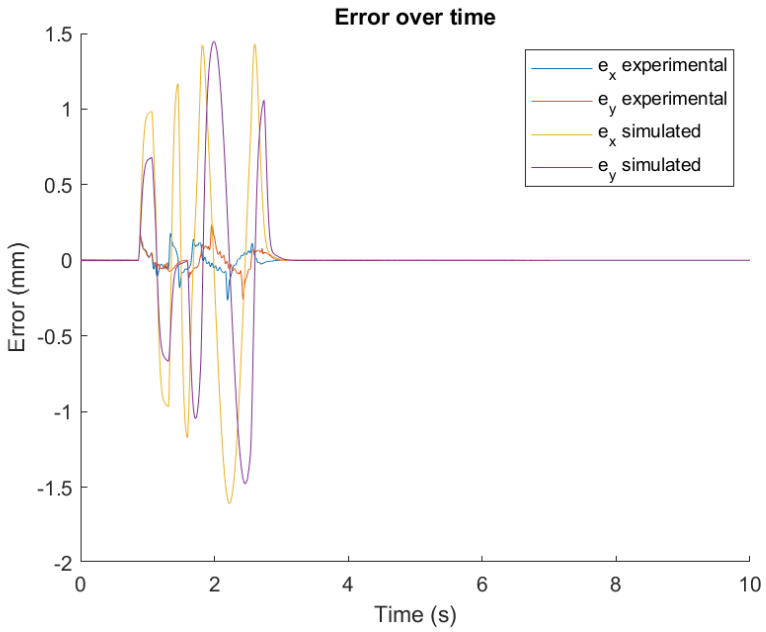


Fig G1.1: Error over time of experimental and simulated toolpaths

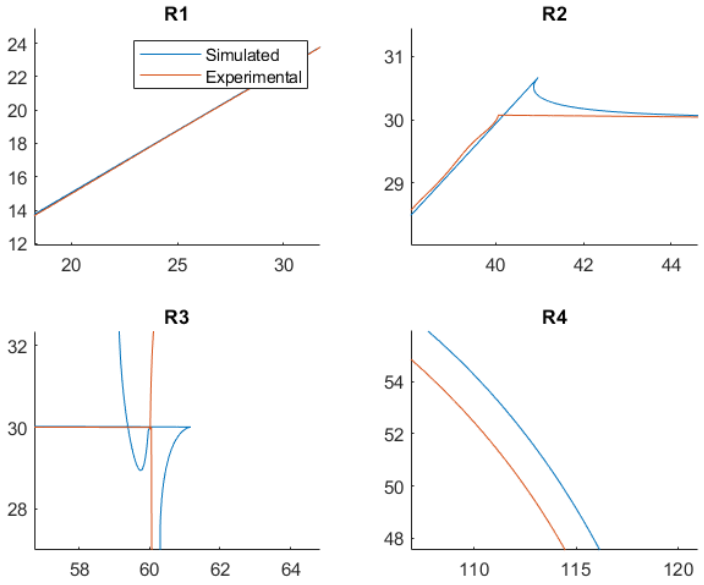


Fig G1.2: Experimental and simulated toolpaths

Simulated and experimental toolpaths matched up well in the sections with straight lines. However, simulated toolpaths tend to overshoot from its intended stop point (see R2 and R3), which causes circular path to be erroneous. This may be due to control time interval not being small enough, among other reasons. This ultimately results in much larger error in simulation than in experimental toolpath.

G2.

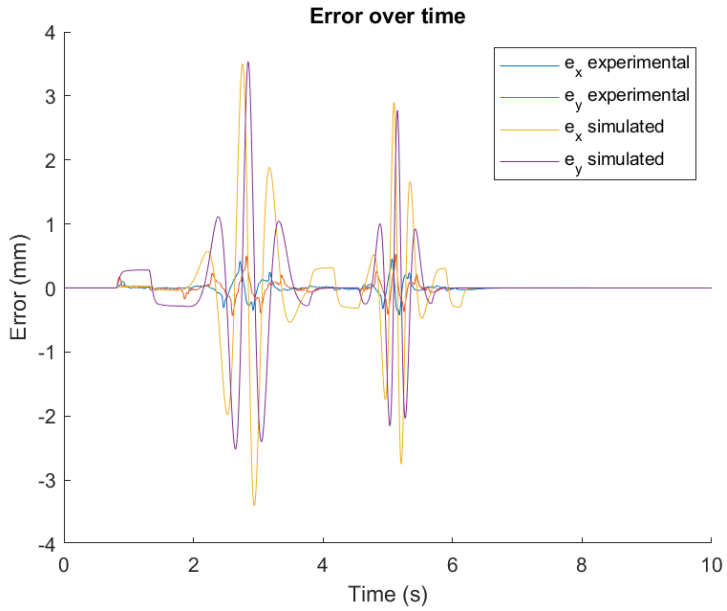


Fig G2.1: Error over time of experimental and simulated toolpaths

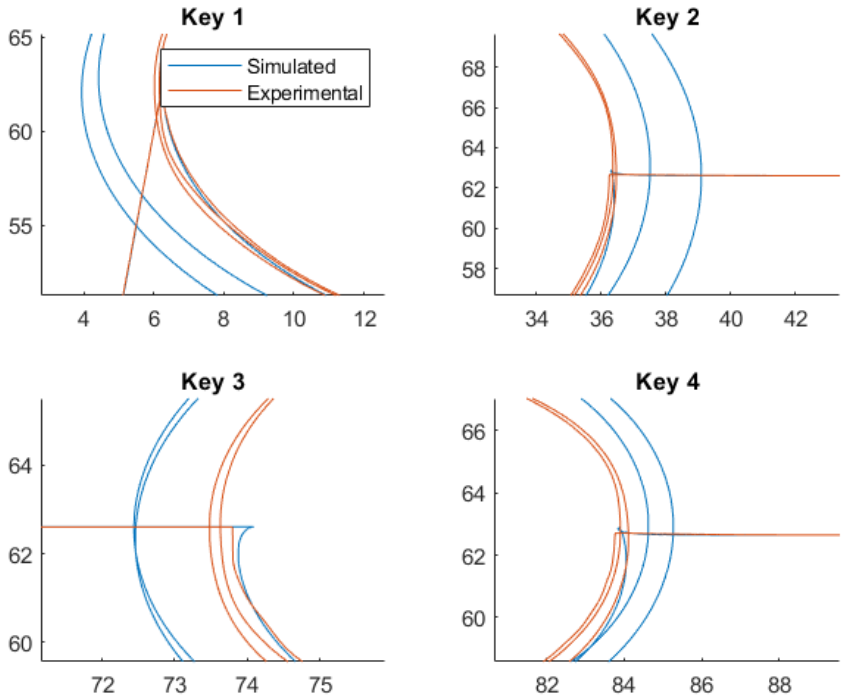


Fig G2.2: Experimental and simulated toolpaths

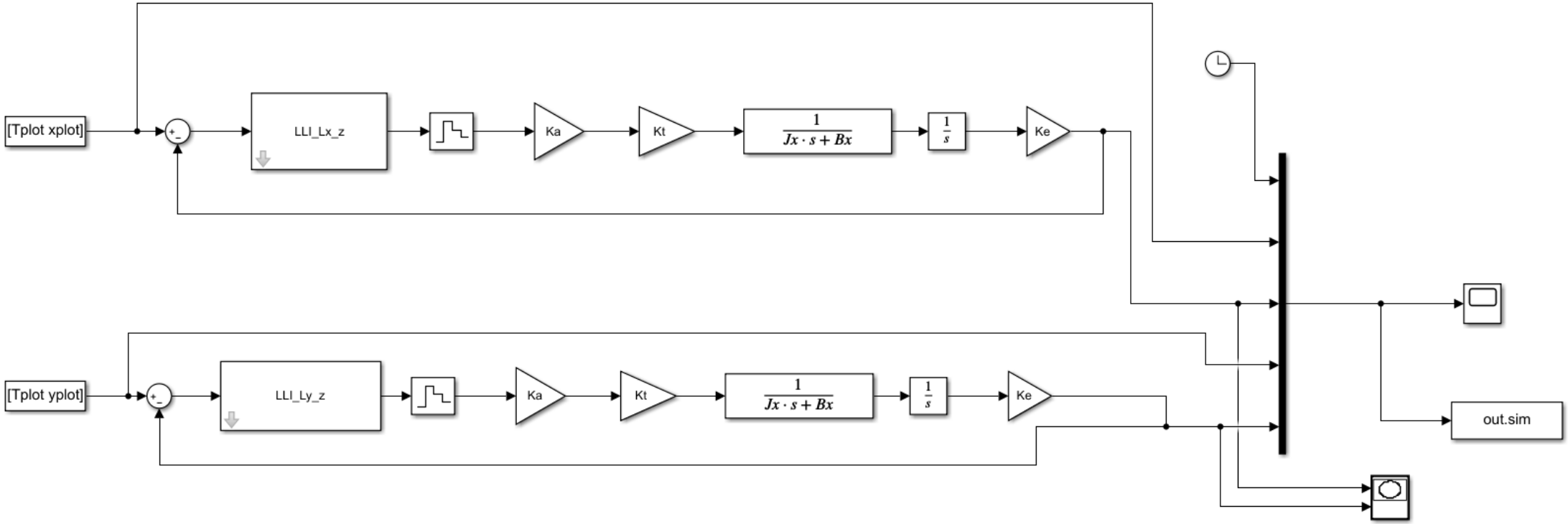
Once again, straight line sections matched well, but its overshoot may have caused circular sections to be significantly different. This ends up causes large differences in contouring error as well, even when the trend of errors are the same (error spike ups and spike downs are at the same location for simulated and experimental errors).

Conclusion:

In this lab I’ve learned to generate toolpath and simulate it based on conditions of the machine it’s being operated on. Bandwidth is one of the operating conditions, where its increases increase contour error, and mismatching bandwidths between the two axis can complicate design. Feedrate is another conditions; increasing maximum feedrate will only affect the toolpath if sections are long enough for toolhead to accelerate to that feedrate. While simulation is a useful tool for predicting the actual toolpath based on reference toolpath, simulated and experimental toolpaths may vary, depending on design of controllers used for simulation and experiment.

Appendix:

1. Simulink Model
2. E1 Matlab code
3. E2 data generation Matlab code
4. E2 data plotting Matlab code
5. E3 Matlab code
6. F1 Matlab code
7. F2 Matlab code
8. F3 Matlab code
9. G1 Matlab code
10. G2 Matlab code


[Tplot xplot]
LLI_Lx_z
Ka
Kt
$\frac{1}{Jx \cdot s + Bx}$
$\frac{1}{s}$
Ke
[Tplot yplot]
LLI_Ly_z
Ka
Kt
$\frac{1}{Jx \cdot s + Bx}$
$\frac{1}{s}$
Ke
out.sim

Contents

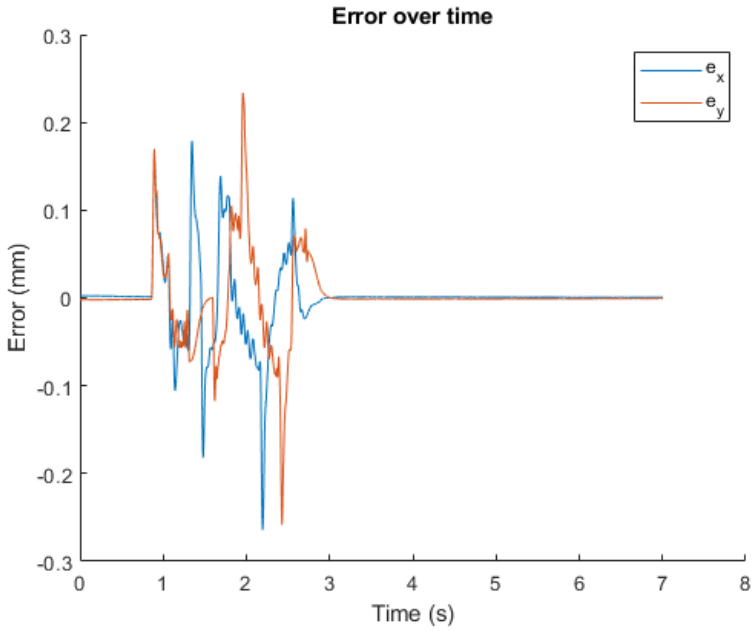

```
%{
The data sets for LAB 3 is now uploaded in the comment section of your prelab submission.

As mentioned today, each one of you have received 2 files. The file named as "Line_Cricle" is the default trajectory of this project and the s

Each of the mentioned files include two sets of measurements: "LLI.mat" and "PP.mat" which refer to Lead-Lag-Integrator and Pole Placement con

The students who did not submit their custom trajectory by the deadline today only received the Line_Circle file, and they will loose the poin



Instruction on the format of provided data:

Each file is a structure with 5 structure fields. The data that you need for your lab report can be accessed as:

Time: lli.X.Data

Actual (measurement) x position: lli.Y(1).Data

Actual (measurement) y position: lli.Y(2).Data

Reference (command) x position: lli.Y(3).Data

Reference (command) y position: lli.Y(4).Data
%}
```

set up

```
name = 'lli.mat';
lli = load(name);
time = lli.lli.X.Data;
x_act = lli.lli.Y(1).Data;
y_act = lli.lli.Y(2).Data;
x_ref = lli.lli.Y(3).Data;
y_ref = lli.lli.Y(4).Data;
```

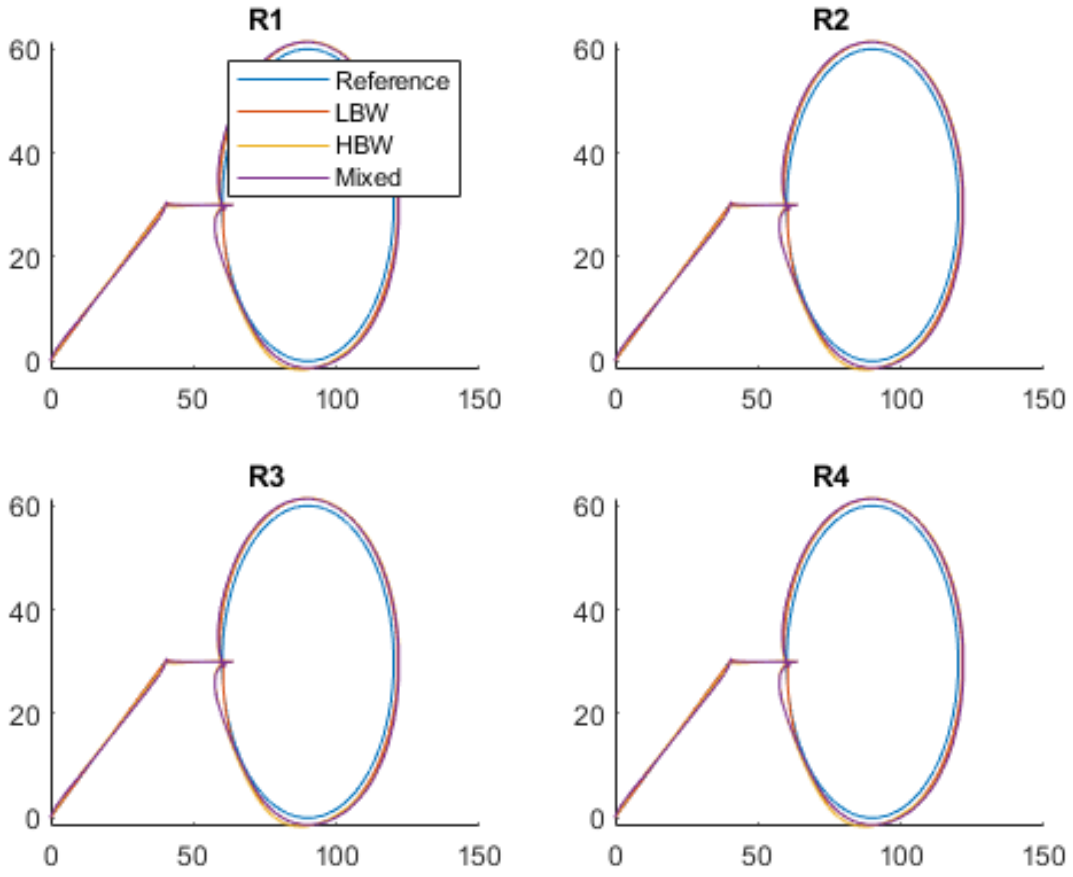
E1

```
e_x = x_ref - x_act;
e_y = y_ref - y_act;

clf();
hold on;
plot(time, e_x);
plot(time, e_y);
legend('e_x', 'e_y');
title('Error over time');
xlabel('Time (s)');
ylabel('Error (mm)');
saveas(gcf, 'E1.png');
```



Published with MATLAB® R2020b

```
figure;
clf();
subplot(2,2,1);
title('R1');
plotter();
legend('Reference', 'LBW', 'HBW', 'Mixed');
subplot(2,2,2);
title('R2');
plotter();
subplot(2,2,3);
title('R3');
plotter();
subplot(2,2,4);
title('R4');
plotter();

function plotter()
    hold on;

    name = 'lli.mat';
    lli = load(name);
    time = lli.lli.X.Data;
    x_act = lli.lli.Y(1).Data;
    y_act = lli.lli.Y(2).Data;
    x_ref = lli.lli.Y(3).Data;
    y_ref = lli.lli.Y(4).Data;

    plot(x_ref, y_ref);

    names = {'lbw.mat', 'hbw.mat', 'mix.mat'};
    for i = 1:3
        data = load(names{i});
        x_sim = data.output.Data(:,3);
        y_sim = data.output.Data(:,5);
        plot(x_sim, y_sim);
    end
end
```

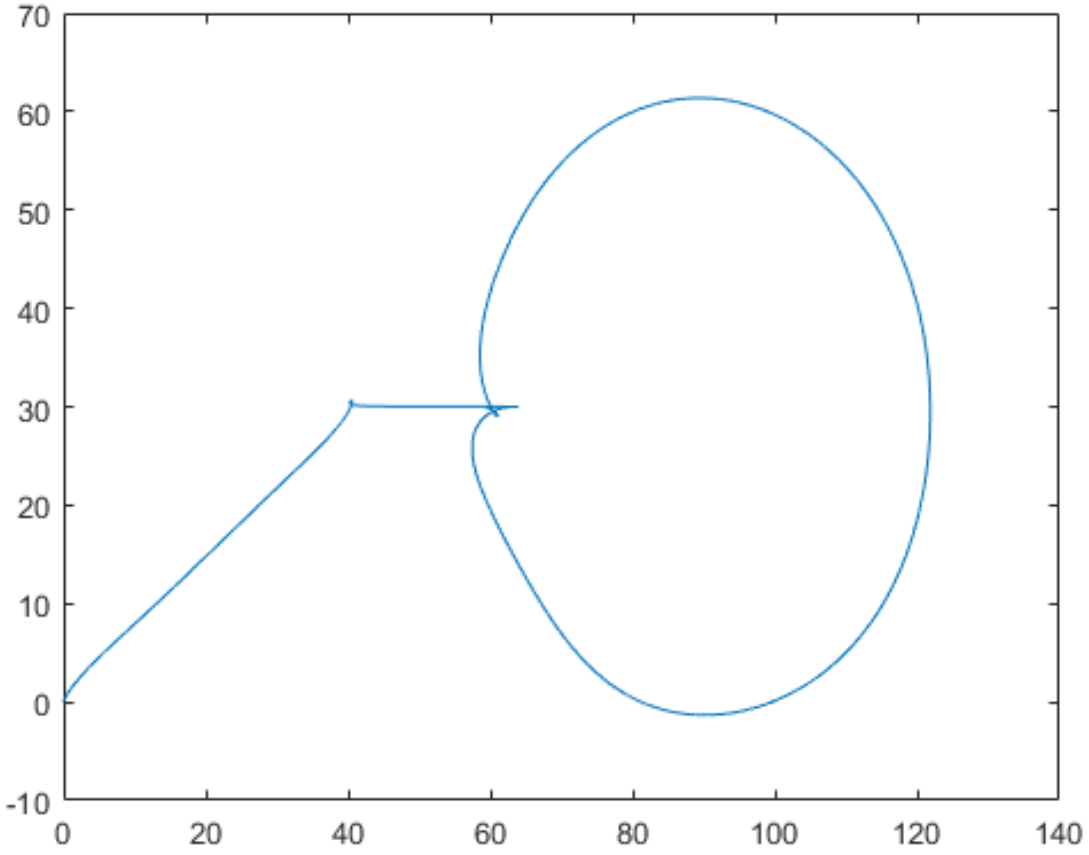


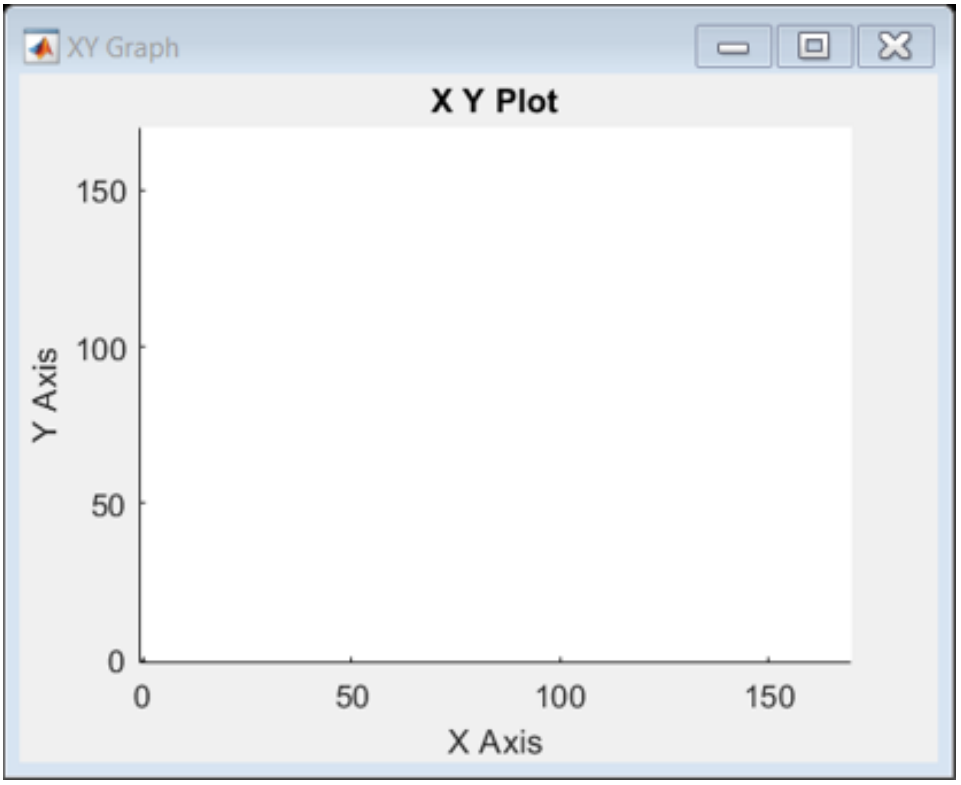
Published with MATLAB® R2020b

s

```
name = 'lli.mat';
lli = load(name);
time = lli.lli.X.Data;
x_act = lli.lli.Y(1).Data;
y_act = lli.lli.Y(2).Data;
x_ref = lli.lli.Y(3).Data;
y_ref = lli.lli.Y(4).Data;

name_ = {'lbw.mat', 'hbw.mat', 'mix.mat'};

ax_ = [13.9282, 13.9282, 13.9282]; %lbw, hbw, hbw
Tx_ = [0.0021323, 0.0010661, 0.0010661];
Kx_ = [0.75858, 0.33497, 0.33497];
Kix_ = [12.5664, 25.1327, 25.1327];

ay_ = [13.9282, 13.9282, 13.9282]; %lbw, hbw, lbw
Ty_ = [0.0021323, 0.0010661, 0.0021323];
Ky_ = [0.82224, 0.47315, 0.82224];
Kiy_ = [12.5664, 25.1327, 12.5664];
for i = 1:3
    % init variables to be loaded into simulink model
    T = 0.0001;
    Ka = 1;
    Kt = 0.49;
    Ke = 1.59;
    Jx = 0.000436;
    Bx = 0.0094;
    Jy = 0.0003;
    By = 0.0091;

    % use LLI controller
    ax = ax_(1, i);
    TTx = Tx_(1, i);
    Kx = Kx_(1, i);
    Kix = Kix_(1, i);

    LLx = tf([ax*TTx 1],[TTx 1]);
    Ix = tf([1 Kix],[1 0]);
    LLI_Lx_z = Kx*c2d(LLx*Ix, T, 'tustin');

    ay = ay_(1, i);
    TTy = Ty_(1, i);
    Ky = Ky_(1, i);
    Kiy = Kiy_(1, i);

    LLy = tf([ay*TTy 1],[TTy 1]);
    Iy = tf([1 Kiy],[1 0]);
    LLI_Ly_z = Ky*c2d(LLy*Iy, T, 'tustin');
```

```
    Tplot = time';
    xplot = x_ref';
    yplot = y_ref';

    sim('e2_sim');

    output = ans.sim;
    outputName = name_{i};
    save(sprintf(outputName), 'output');
    x_sim = ans.sim.Data(:,3);
    y_sim = ans.sim.Data(:,5);
    plot(x_sim, y_sim);
end
```


70
60
50
40
30
20
10
0
-10
0
20
40
60
80
100
120
140


XY Graph
X Y Plot
150
100
50
0
Y Axis
0
50
100
150
X Axis

Published with MATLAB® R2020b

```
name = 'lbw.mat';
data = load(name);
time_ = data.output.Data(:,1);
x_ref_ = data.output.Data(:,2);
y_ref_ = data.output.Data(:,4);
x_sim_ = data.output.Data(:,3);
y_sim_ = data.output.Data(:,5);

toPrint = [];
flag = true;
i = 1;
while flag
    time = time_(i, 1);
    x_ref = x_ref_(i, 1);
    y_ref = y_ref_(i, 1);
    x_sim = x_sim_(i, 1);
    y_sim = y_sim_(i, 1);
    if x_ref >= 40 && y_ref >= 30
        flag = false;
    else
        if time >= 0.8615
            e_x = x_ref - x_sim;
            e_y = y_ref - y_sim;
            e_c = sqrt(e_x*e_x + e_y*e_y);
            toPrint = [toPrint; time, x_ref, y_ref, x_sim, y_sim, e_x, e_y, e_c];
        end
        i = i + 1;
    end
end

%disp(toPrint);
disp(max(toPrint(:,8)));
```

```
    1.1964
```

Published with MATLAB® R2020b

Contents

init vars

```
clf();

A = 1000; % mm/s^2
fc = 250; % mm/s
T = 0.1; % ms
T = T * 0.001; % s

P1 = [0 0];
P2 = [40 30];
P3 = [60 30];
P4 = [90 30];
```

P1 -> P2

```
[T1, T2, T3] = calc(50); % sqrt(30*30+40*40)=50
T_total = T1+T2+T3;

x_ratio = 4/5;
y_ratio = 3/5;

t = 0;
s = 0;
sdot = 0;
sdotdot = A;
xr = 0;
yr = 0;
vxr = 0;
vyr = 0;
axr = A*x_ratio;
ayr = A*y_ratio;

count = 1;
data1 = zeros(T_total/T, 10);
while t <= T_total-T
    data1(count, 1) = t;
    data1(count, 2) = s;
    data1(count, 3) = sdot;
```

```
    data1(count, 4) = sdotdot;
    data1(count, 5) = xr;
    data1(count, 6) = yr;
    data1(count, 7) = vxr;
    data1(count, 8) = vyr;
    data1(count, 9) = axr;
    data1(count, 10) = ayr;

    if t < T1
        sdotdot = A;
    elseif t >= T1 && t < T1+T2
        sdotdot = 0;
    else
        sdotdot = -A;
    end

    sdot = sdot + T*sdotdot;
    s = s + sdot*T;

    xr = s*x_ratio;
    yr = s*y_ratio;
    vxr = sdot*x_ratio;
    vyr = sdot*y_ratio;
    axr = sdotdot*x_ratio;
    ayr = sdotdot*y_ratio;

    t = t + T;
    count = count + 1;
end
```

P2 -> P3

```
[T1, T2, T3] = calc(20); % 60-40=20
T_total = T1+T2+T3;

x_ratio = 1;
y_ratio = 0;

t = 0;
s = 0;
sdot = 0;
sdotdot = A;
xr = 0;
yr = 0;
vxr = 0;
vyr = 0;
axr = A*x_ratio;
ayr = A*y_ratio;

count = 1;
data2 = zeros(T_total/T, 10);
```

```
while t <= T_total-T
    data2(count, 1) = t+data1(end, 1);
    data2(count, 2) = s+data1(end, 2);
    data2(count, 3) = sdot+data1(end, 3);
    data2(count, 4) = sdotdot;
    data2(count, 5) = xr+data1(end, 5);
    data2(count, 6) = yr+data1(end, 6);
    data2(count, 7) = vxr+data1(end, 7);
    data2(count, 8) = vyr+data1(end, 8);
    data2(count, 9) = axr;
    data2(count, 10) = ayr;

    if t < T1
        sdotdot = A;
    elseif t >= T1 && t < T1+T2
        sdotdot = 0;
    else
        sdotdot = -A;
    end

    sdot = sdot + T*sdotdot;
    s = s + sdot*T;

    xr = s*x_ratio;
    yr = s*y_ratio;
    vxr = sdot*x_ratio;
    vyr = sdot*y_ratio;
    axr = sdotdot*x_ratio;
    ayr = sdotdot*y_ratio;

    t = t + T;
    count = count + 1;
end
```

P3 circle

```
L = 2*3.1415*30;
[T1, T2, T3] = calc(L); % 90-60=30
T_total = T1+T2+T3;

t = 0;
s = 0;
sdot = 0;
sdotdot = A;
xr = 0;
yr = 0;
vxr = 0;
vyr = 0;
axr = A*x_ratio;
ayr = A*y_ratio;
```

```
count = 1;
data3 = zeros(T_total/T, 10);
while t <= T_total-T
    if t < T1
        sdotdot = A;
    elseif t >= T1 && t < T1+T2
        sdotdot = 0;
    else
        sdotdot = -A;
    end

    sdot = sdot + T*sdotdot;
    s = s + sdot*T;

    [x_ratio, y_ratio] = getRatios(s/L);

    vxr_prev = vxr;
    vyr_prev = vyr;
    vxr = sdot*x_ratio;
    vyr = sdot*y_ratio;

    R = 30;
    xr = xr + vxr*T;
    yr = yr + vyr*T;

    axr = (vxr - vxr_prev)/T;
    ayr = (vyr - vyr_prev)/T;

    data3(count, 1) = t+data2(end, 1);
    data3(count, 2) = s+data2(end, 2);
    data3(count, 3) = sdot+data2(end, 3);
    data3(count, 4) = sdotdot;
    data3(count, 5) = xr+data2(end, 5);
    data3(count, 6) = yr+data2(end, 6);
    data3(count, 7) = vxr+data2(end, 7);
    data3(count, 8) = vyr+data2(end, 8);
    data3(count, 9) = axr;
    data3(count, 10) = ayr;

    t = t + T;
    count = count + 1;
end
```

data for plot

```
data = [data1; data2; data3];
```

plot

```
plot(data(:, 5), data(:, 6));
title('F1: x and y position');
xlabel('x position (mm)');
ylabel('y position (mm)');
saveas(gcf, 'qF1-1.png');
clf;
txy.t = data(:, 1);
txy.x = data(:, 5);
txy.y = data(:, 6);
save f1 txy

subplot(3,1,1);
plot(data(:,1), data(:,2));
title('F1: displacement (mm) over time (s)');
subplot(3,1,2);
plot(data(:,1), data(:,3));
title('F1: feedrate (mm/s) over time (s)');
subplot(3,1,3);
plot(data(:,1), data(:,4));
title('F1: tangential acceleration (mm/s/s) over time (s)');
saveas(gcf, 'qF1-2.png');
clf;

subplot(3,1,1);
hold on;
plot(data(:,1), data(:,5));
plot(data(:,1), data(:,6));
title('F1: axis position (mm) over time (s)');
legend('x', 'y');
subplot(3,1,2);
hold on;
plot(data(:,1), data(:,7));
plot(data(:,1), data(:,8));
title('F1: axis velocity (mm/s) over time (s)');
legend('x', 'y');
subplot(3,1,3);
hold on;
plot(data(:,1), data(:,9));
plot(data(:,1), data(:,10));
title('F1: axis acceleration (mm/s/s) over time (s)');
legend('x', 'y');
saveas(gcf, 'qF1-3.png');
clf;
```

helper

```
function [x_r, y_r] = getRatios(r)
    x_r = 0;
    y_r = 0;
    if r < .25
        theta = (pi/2)*(r/.25);
        x_r = sin(theta);
        y_r = -cos(theta);
    elseif r >= .25 && r < 0.5
        theta = (pi/2)*((r-0.25)/.25);
        x_r = cos(theta);
        y_r = sin(theta);
    elseif r >= 0.5 && r < 0.75
        theta = (pi/2)*((r-0.5)/.25);
        x_r = -sin(theta);
        y_r = cos(theta);
    else
        theta = (pi/2)*((r-0.75)/.25);
        x_r = -cos(theta);
        y_r = -sin(theta);
    end
end

function [T1, T2, T3] = calc(L)
```

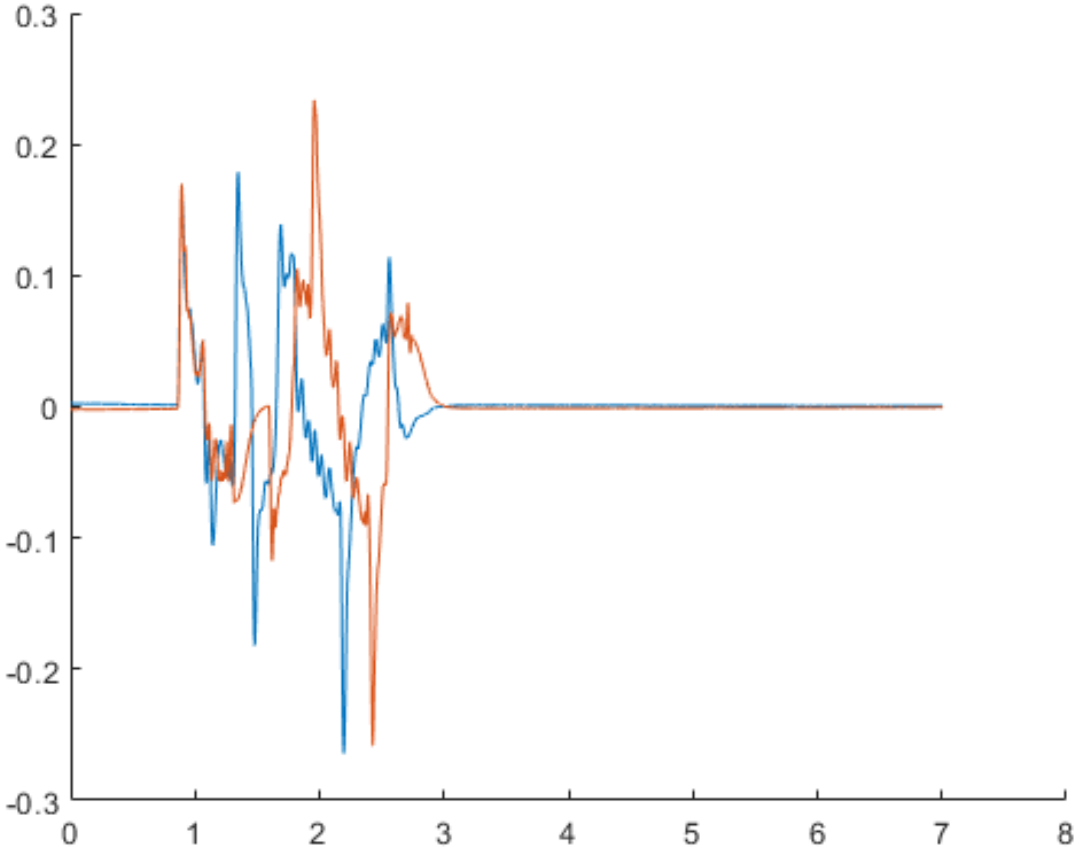
```
    A = 1000; % mm/s^2
    fc = 250; % mm/s
    T = 0.1; % ms
    T = T * 0.001; % s

    T1 = fc/A;
    T3 = T1;
    s_init = A*T1*T1/2; % this is 20
    T2 = (L-2*s_init)/fc;
    T2 = ceil(T2/T)*T;
    if T2 < 0
        T2 = 0;
        s_init = L/2;
        T1 = sqrt(2*s_init/A);
        T1 = ceil(T1/T)*T;
        T3 = T1;
    end
end
```

Published with MATLAB® R2020b

Contents

low feedrate

```
name = 'lli.mat';
lli = load(name);
time = lli.lli.X.Data;
x_act = lli.lli.Y(1).Data;
y_act = lli.lli.Y(2).Data;
x_ref = lli.lli.Y(3).Data;
y_ref = lli.lli.Y(4).Data;

e_x_lowFR = x_ref - x_act;
e_y_lowFR = y_ref - y_act;

clf();
hold on;
plot(time, e_x_lowFR);
plot(time, e_y_lowFR);
```

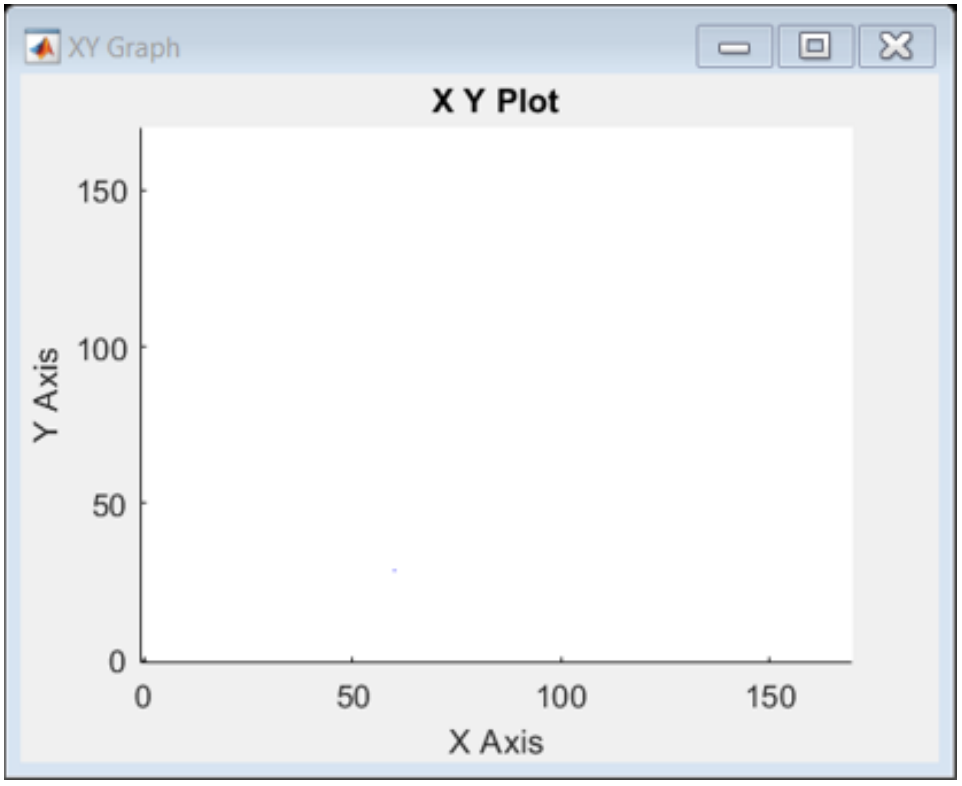


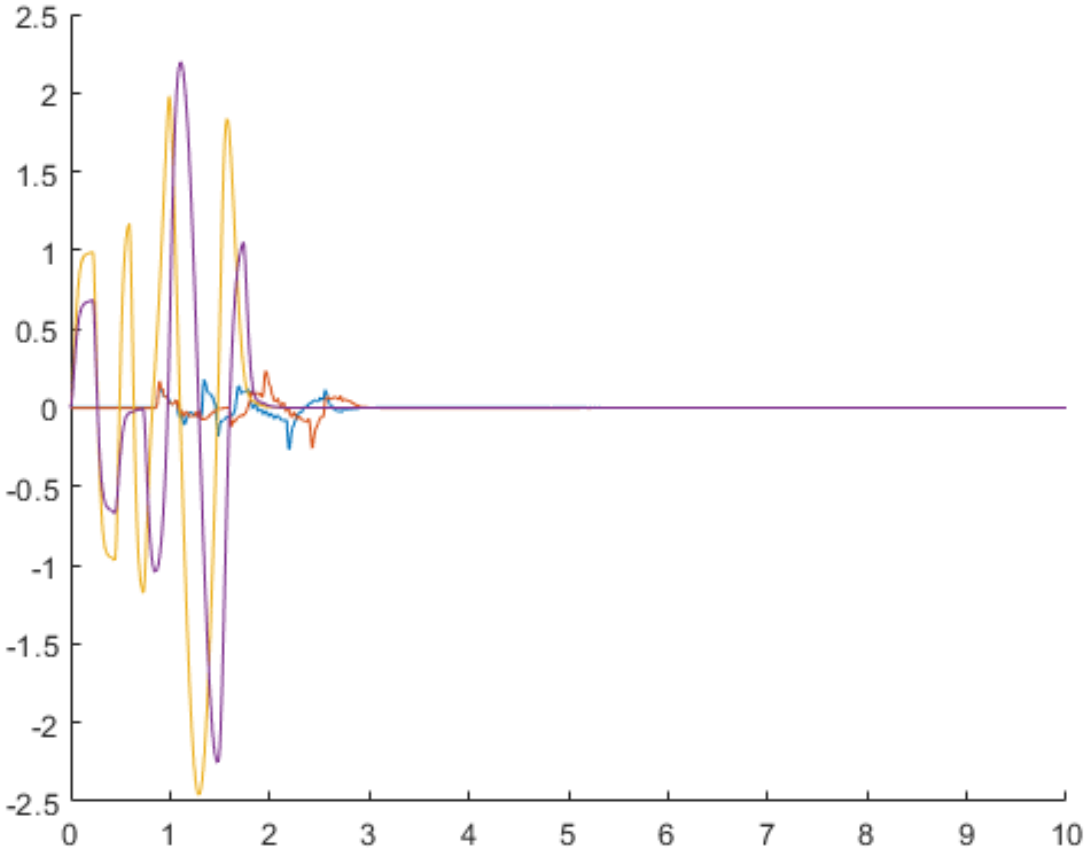
high feedrate

init variables to be loaded into simulink model

```
T = 0.0001;
Ka = 1;
Kt = 0.49;
Ke = 1.59;
Jx = 0.000436;
Bx = 0.0094;
Jy = 0.0003;
By = 0.0091;

% use LBW LLI controller
a = 13.9282;
T_ = 0.0021323;
Kx = 0.75858;
Ky = 0.82224;
Ki = 12.5664;

LL = tf([a*T_ 1],[T_ 1]);
I = tf([1 Ki],[1 0]);
LLI_Lx_z = Kx*c2d(LL*I, T, 'tustin');
LLI_Ly_z = Ky*c2d(LL*I, T, 'tustin');
```

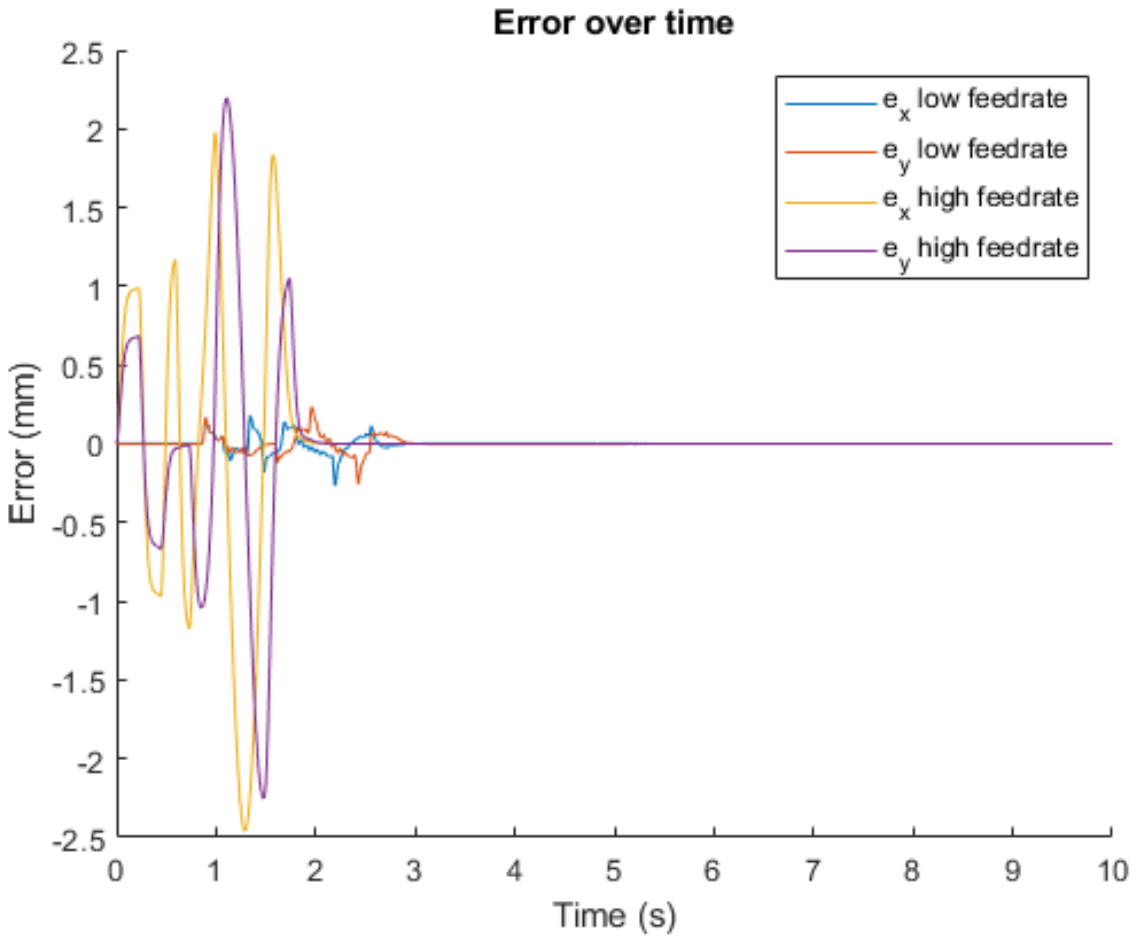
```
name = 'f1.mat';
data = load(name);
Tplot = data.txy.t;
xplot = data.txy.x;
yplot = data.txy.y;

sim('e2_sim.slx');

t = ans.sim.Data(:,1);
e_x_hiFR = ans.sim.Data(:,2) - ans.sim.Data(:,3);
e_y_hiFR = ans.sim.Data(:,4) - ans.sim.Data(:,5);
plot(t, e_x_hiFR);
plot(t, e_y_hiFR);
```





plot formatting

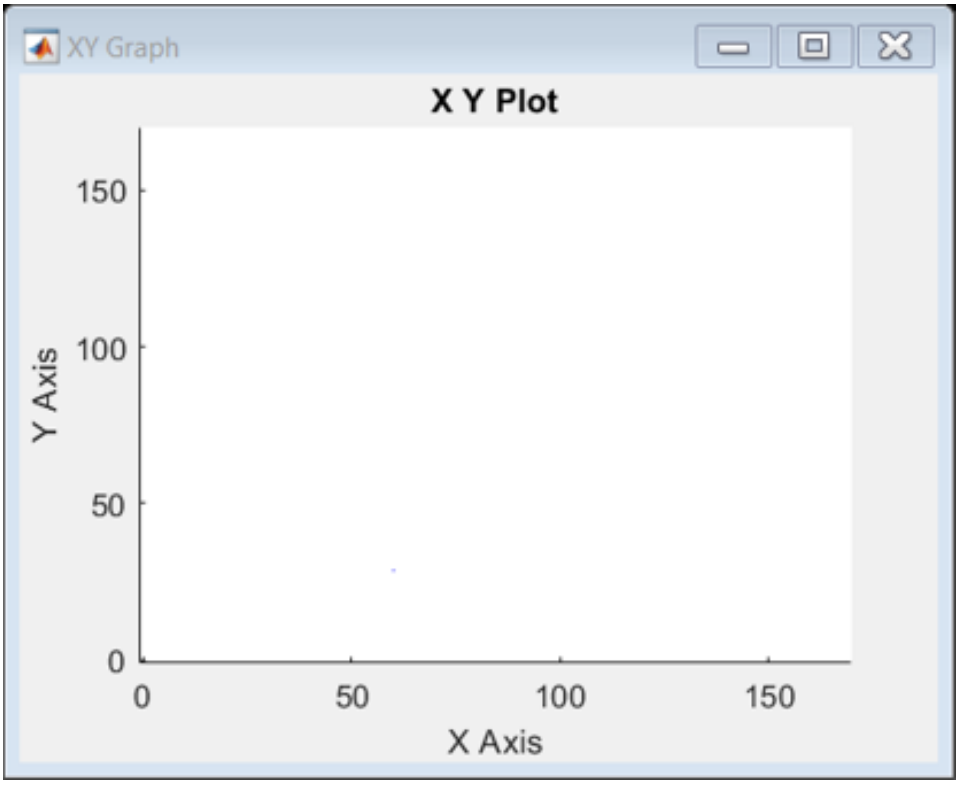
```
legend('e_x low feedrate', 'e_y low feedrate', 'e_x high feedrate', 'e_y high feedrate');
title('Error over time');
xlabel('Time (s)');
saveas(gcf, 'F2.png');
ylabel('Error (mm)');
```

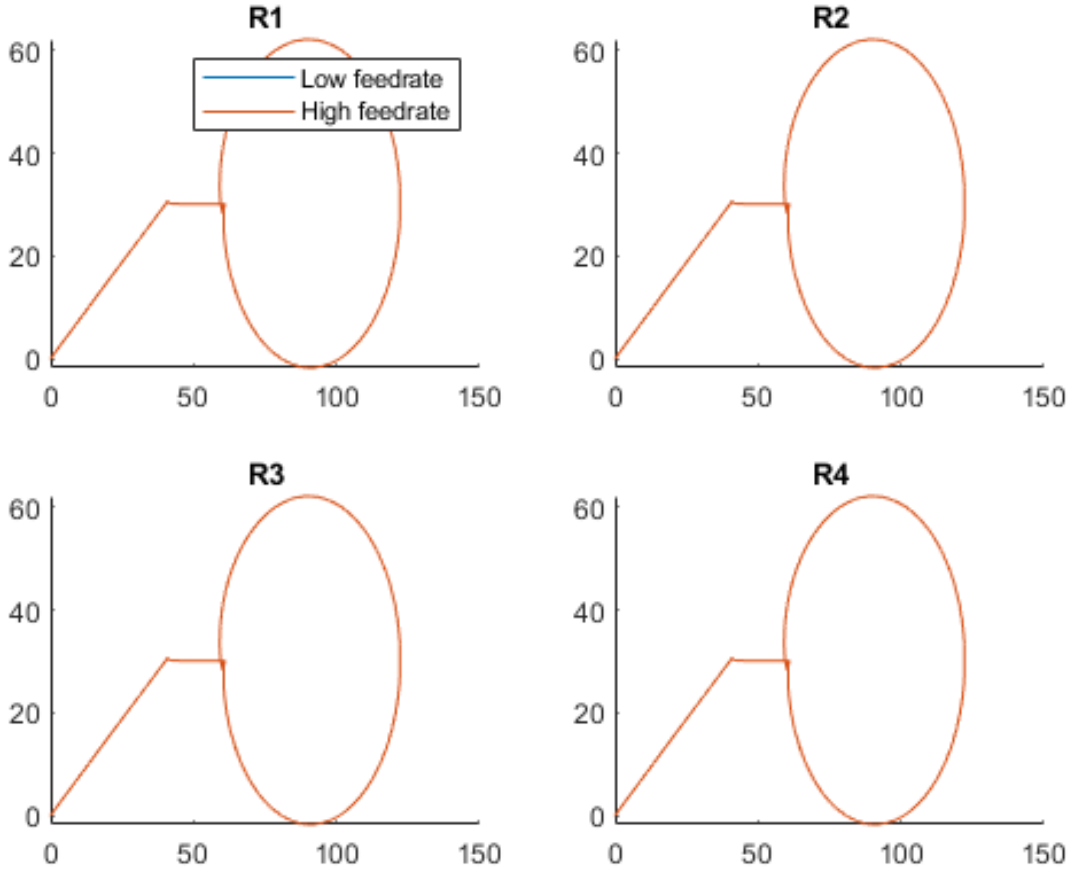


Published with MATLAB® R2020b

```
%{
names = {'f3_200.mat', 'f3_250.mat'};
hold on;
for i = 1:2
    name = names{i};
    data = load(name);
    Tplot = data.txy.t;
    xplot = data.txy.x;
    yplot = data.txy.y;
    plot(Tplot, xplot);
    plot(Tplot, yplot);
end
title('Position over time');
ylabel('Position (mm)');
xlabel('Time (s)');
legend('X Low feedrate','Y Low feedrate','X High feedrate','Y High feedrate');
saveas(gcf, 'F3-additional.png');
%}

figure;
clf();
subplot(2,2,1);
title('R1');
plotter();
legend('Low feedrate', 'High feedrate');
subplot(2,2,2);
title('R2');
plotter();
subplot(2,2,3);
title('R3');
plotter();
subplot(2,2,4);
title('R4');
plotter();

function plotter()
    hold on;

    names = {'f3_200.mat', 'f3_250.mat'};
    for i = 1:2
        % init variables to be loaded into simulink model
        T = 0.0001;
        Ka = 1;
        Kt = 0.49;
        Ke = 1.59;
        Jx = 0.000436;
        Bx = 0.0094;
        Jy = 0.0003;
        By = 0.0091;

        % use LBW LLI controller
```

```
        a = 13.9282;
        T_ = 0.0021323;
        Kx = 0.75858;
        Ky = 0.82224;
        Ki = 12.5664;

        LL = tf([a*T_ 1],[T_ 1]);
        I = tf([1 Ki],[1 0]);
        LLI_Lx_z = Kx*c2d(LL*I, T, 'tustin');
        LLI_Ly_z = Ky*c2d(LL*I, T, 'tustin');

        name = names{i};
        data = load(name);
        Tplot = data.txy.t;
        xplot = data.txy.x;
        yplot = data.txy.y;

        sim('e2_sim.slx');

        x = ans.sim.Data(:,3);
        y = ans.sim.Data(:,5);
        plot(x, y);
    end
end
```


XY Graph
X Y Plot
Y Axis
X Axis
0
50
100
150

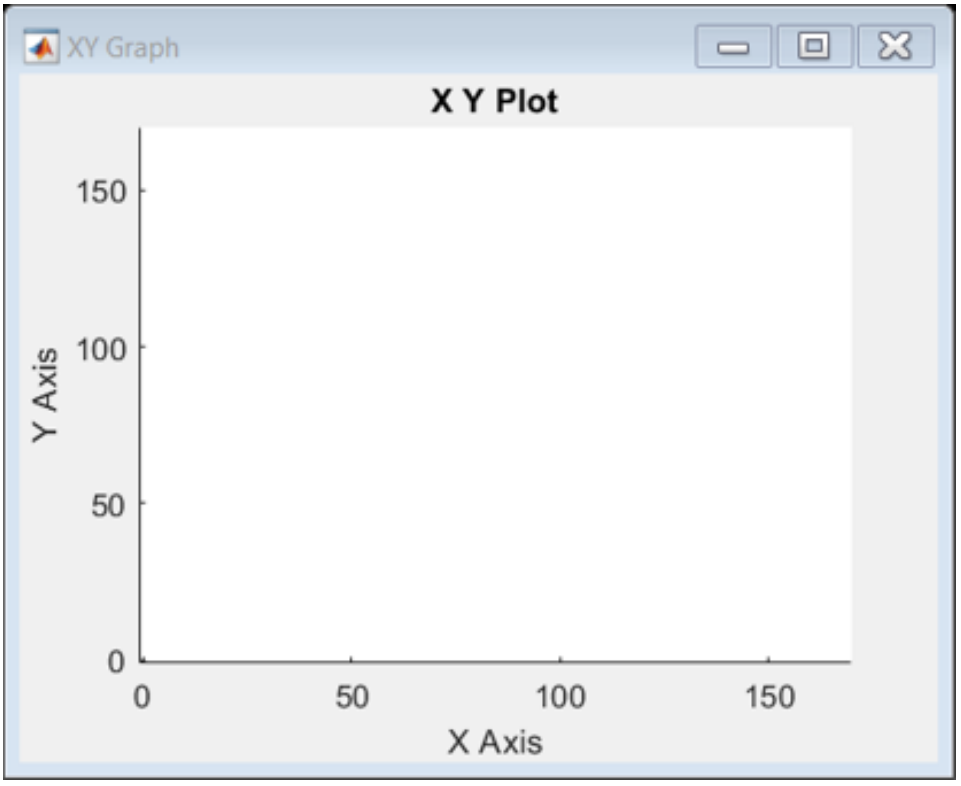

R1
R2
R3
R4
Low feedrate
High feedrate
0
20
40
60
0
50
100
150

Published with MATLAB® R2020b

Contents

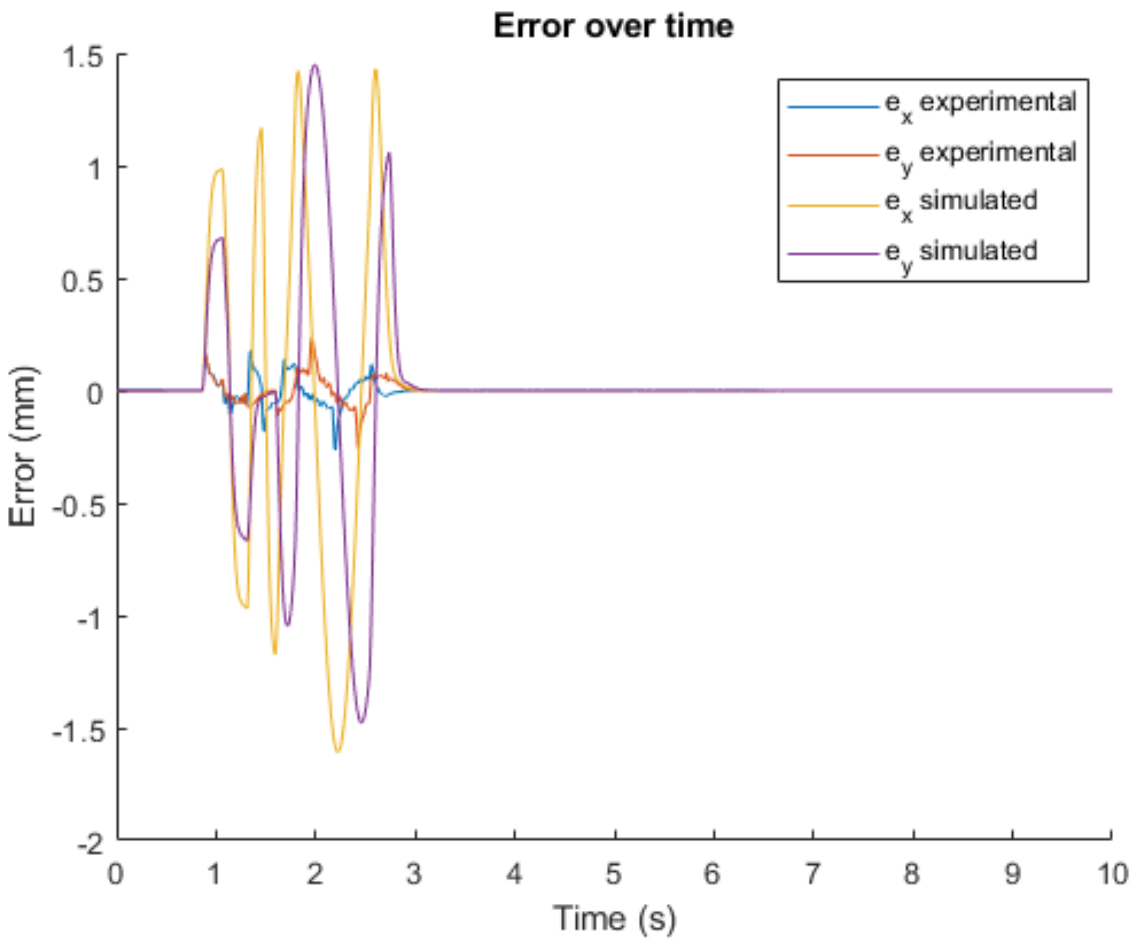
set up

```
name = 'lli.mat';
lli = load(name);
time = lli.lli.X.Data;
x_act = lli.lli.Y(1).Data;
y_act = lli.lli.Y(2).Data;
x_ref = lli.lli.Y(3).Data;
y_ref = lli.lli.Y(4).Data;

% init variables to be loaded into simulink model
T = 0.0001;
Ka = 1;
Kt = 0.49;
Ke = 1.59;
Jx = 0.000436;
Bx = 0.0094;
Jy = 0.0003;
By = 0.0091;

% use LBW LLI controller
a = 13.9282;
T_ = 0.0021323;
Kx = 0.75858;
Ky = 0.82224;
Ki = 12.5664;

LL = tf([a*T_ 1],[T_ 1]);
I = tf([1 Ki],[1 0]);
LLI_Lx_z = Kx*c2d(LL*I, T, 'tustin');
LLI_Ly_z = Ky*c2d(LL*I, T, 'tustin');

Tplot = time';
xplot = x_ref';
yplot = y_ref';

sim('e2_sim.slx');

t_sim = ans.sim.Data(:,1);
x_sim_ref = ans.sim.Data(:,2);
x_sim = ans.sim.Data(:,3);
y_sim_ref = ans.sim.Data(:,4);
y_sim = ans.sim.Data(:,5);
```



plot error

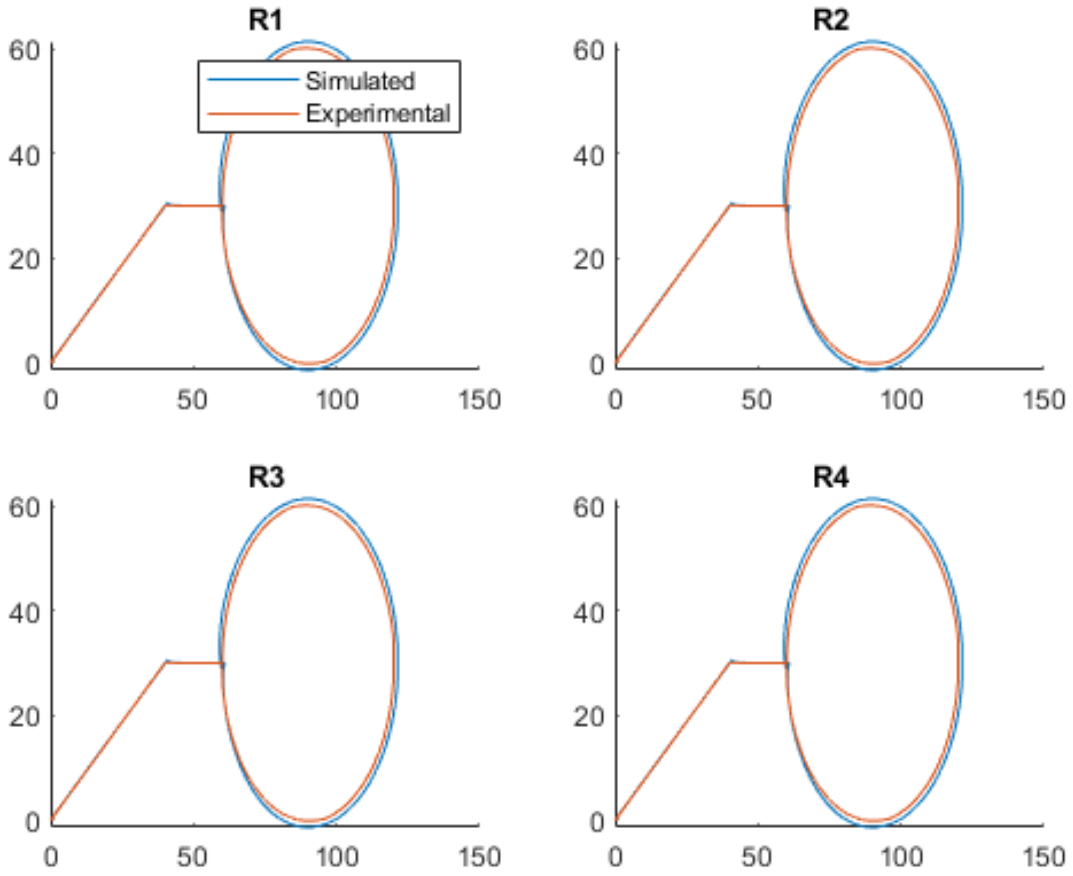
```
e_x_sim = x_sim_ref - x_sim;
e_y_sim = y_sim_ref - y_sim;
e_x_exp = x_ref - x_act;
e_y_esp = y_ref - y_act;

clf();
hold on;
plot(time, e_x_exp);
plot(time, e_y_esp);
plot(t_sim, e_x_sim);
plot(t_sim, e_y_sim);
legend('e_x experimental', 'e_y experimental', 'e_x simulated', 'e_y simulated');
title('Error over time');
xlabel('Time (s)');
ylabel('Error (mm)');
saveas(gcf, 'G1-1.png');
```



plot toolpath

```
x_exp = x_act;
y_exp = y_act;
figure;
clf();
subplot(2,2,1);
title('R1');
hold on;
plot(x_sim, y_sim);
plot(x_exp, y_exp);
legend('Simulated', 'Experimental');
subplot(2,2,2);
title('R2');
hold on;
plot(x_sim, y_sim);
plot(x_exp, y_exp);
subplot(2,2,3);
title('R3');
hold on;
plot(x_sim, y_sim);
plot(x_exp, y_exp);
subplot(2,2,4);
title('R4');
hold on;
```

```
plot(x_sim, y_sim);
plot(x_exp, y_exp);
```

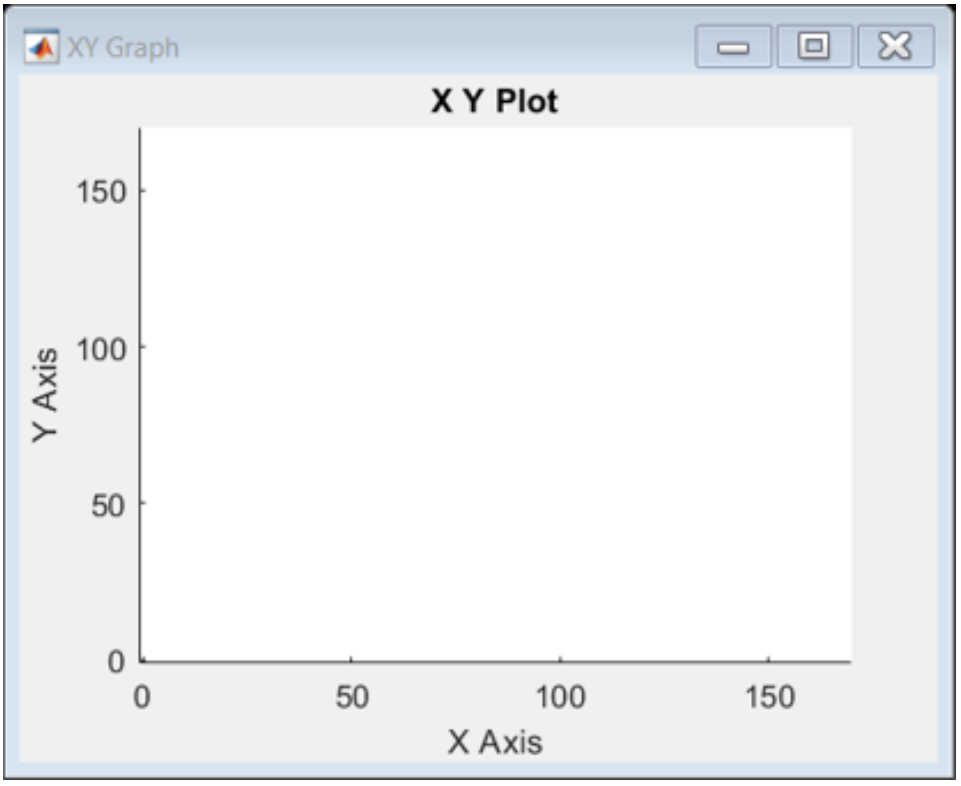


Published with MATLAB® R2020b

Contents

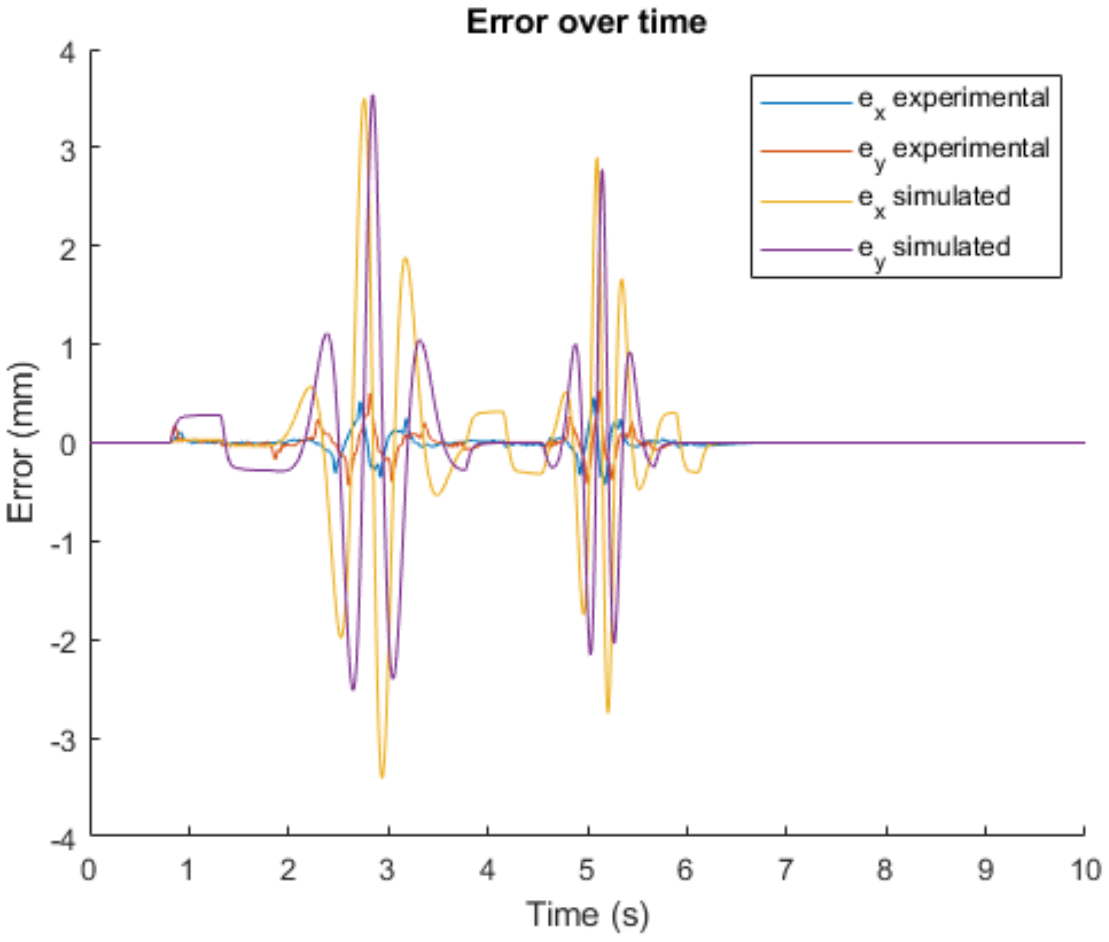
set up

```
name = 'lli.mat';
lli = load(name);
time = lli.lli.X.Data;
x_act = lli.lli.Y(1).Data;
y_act = lli.lli.Y(2).Data;
x_ref = lli.lli.Y(3).Data;
y_ref = lli.lli.Y(4).Data;

% init variables to be loaded into simulink model
T = 0.0001;
Ka = 1;
Kt = 0.49;
Ke = 1.59;
Jx = 0.000436;
Bx = 0.0094;
Jy = 0.0003;
By = 0.0091;

% use LBW LLI controller
a = 13.9282;
T_ = 0.0021323;
Kx = 0.75858;
Ky = 0.82224;
Ki = 12.5664;

LL = tf([a*T_ 1],[T_ 1]);
I = tf([1 Ki],[1 0]);
LLI_Lx_z = Kx*c2d(LL*I, T, 'tustin');
LLI_Ly_z = Ky*c2d(LL*I, T, 'tustin');

Tplot = time';
xplot = x_ref';
yplot = y_ref';

sim('g2_sim.slx');

t_sim = ans.sim.Data(:,1);
x_sim_ref = ans.sim.Data(:,2);
x_sim = ans.sim.Data(:,3);
y_sim_ref = ans.sim.Data(:,4);
y_sim = ans.sim.Data(:,5);
```



plot error

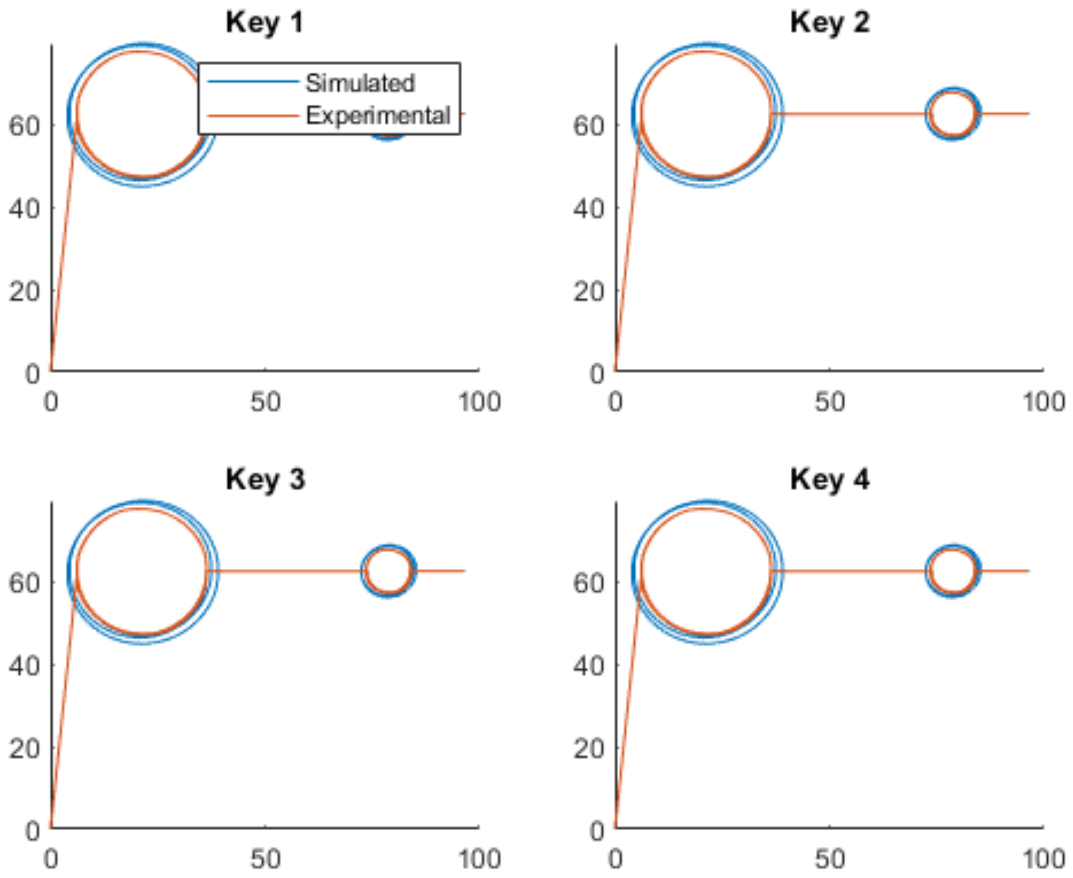
```
e_x_sim = x_sim_ref - x_sim;
e_y_sim = y_sim_ref - y_sim;
e_x_exp = x_ref - x_act;
e_y_esp = y_ref - y_act;

clf();
hold on;
plot(time, e_x_exp);
plot(time, e_y_esp);
plot(t_sim, e_x_sim);
plot(t_sim, e_y_sim);
legend('e_x experimental', 'e_y experimental', 'e_x simulated', 'e_y simulated');
title('Error over time');
xlabel('Time (s)');
ylabel('Error (mm)');
saveas(gcf, 'G2-1.png');
```



plot toolpath

```
x_exp = x_act;
y_exp = y_act;
figure;
clf();
subplot(2,2,1);
title('Key 1');
hold on;
plot(x_sim, y_sim);
plot(x_exp, y_exp);
legend('Simulated', 'Experimental');
subplot(2,2,2);
title('Key 2');
hold on;
plot(x_sim, y_sim);
plot(x_exp, y_exp);
subplot(2,2,3);
title('Key 3');
hold on;
plot(x_sim, y_sim);
plot(x_exp, y_exp);
subplot(2,2,4);
title('Key 4');
hold on;
```

```
plot(x_sim, y_sim);
plot(x_exp, y_exp);
```



Published with MATLAB® R2020b