BONN-RHEIN-SIEG UNIVERSITY OF APPLIED SCIENCES

Scientific Experimentation and Evaluation
- Task 5 : Motion Model -

Mazin Eltayeb, Bastian Lang

June 13, 2016

CONTENTS

Table A.1: Motor Commands

Log no.	Right wheel turns	Right wheel power	Left Wheel turns	Left wheel power
1	200	100	100	50
2	200	100	50	25
3	100	50	200	100
4	50	25	200	100

Table B.1: Obtained Values From Each Log File

Log No	V in m/s	Omega in radiant/s	Sigma1	Sigma2	Sigma3
1	0.0582	2.2015e-04	0.2900	8.4868e-04	0.0101
2	-0.2020	-0.0011	0.1514	0.0009	0.0125
3	-0.1432	-0.0004	0.2666	0.0008	0.0109
4	-0.2472	-0.0030	0.0630	0.0012	0.0131

1. Abstract

This report describes how we solved the task of determining the parameters of the motion model for our LEGO NXT robot. We will describe the experiments we performed, the preprocessing of the data and the way we computed the parameters. The results and the code can be found in the Appendix.

2. Experiments

In our experiments we commanded the robot to do two different sized left circles and two different sized right circles. All movements have been performed with different velocities. The values can be seen in table A.1. The motion trajectories can be seen in the figures in Appendix C.

3. Obtaining Alphas

To calculate the alphas we first needed to pre-process our obtained log data. The entries needed to be cropped by the first and last 450 entries because those weren't part of the actual motion. For each different kind of motion we applied Thrun's algorithm to calculate $\hat{v}$, $\hat{w}$ and $\hat{\gamma}$ and then computed their means and standard deviations. Using pseudo inverse we then used to acquire the alpha values.

A. Commands

B. Results

Table B.2: Obtained Alphas

Parameter	Value
Alpha1	2.1242
Alpha2	-158.9048
Alpha3	0.0063
Alpha4	-0.1271
Alpha5	0.0864
Alpha6	-2.8262

C. Plots

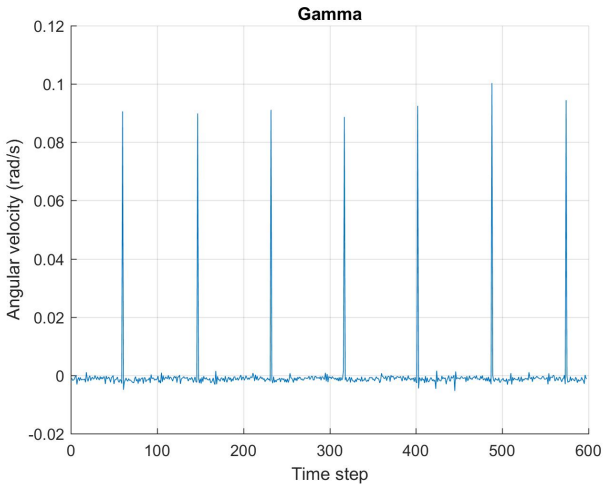

Figure C.1: log1:Gamma vs Time

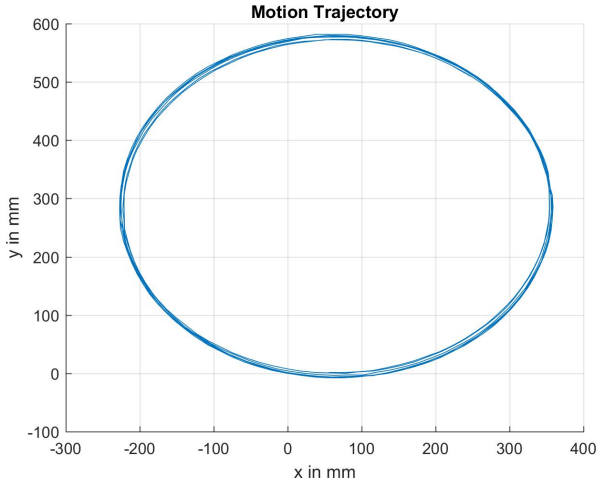

Figure C.2: log1:Motion Trajectory

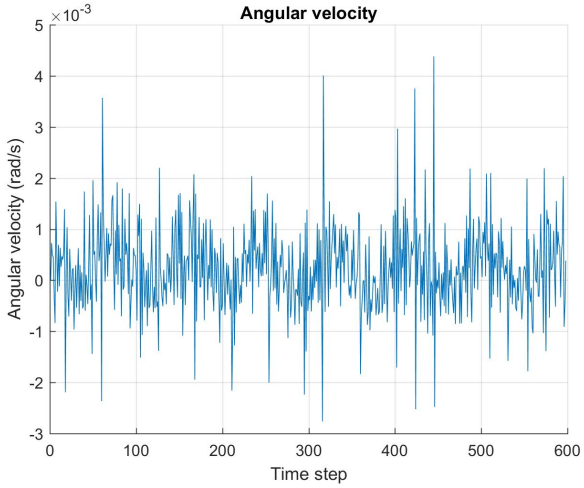

Figure C.3: log1:Omega vs Time

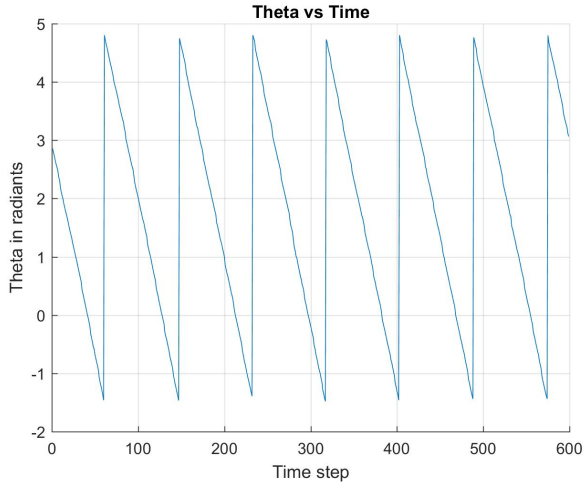

Figure C.4: log1:Theta vs Time

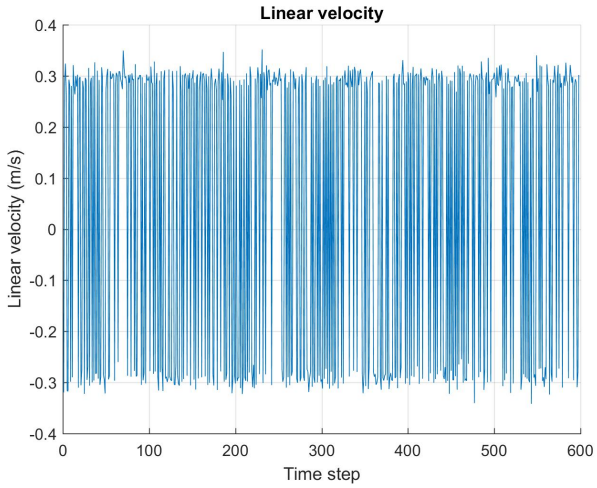

Figure C.5: log1:V vs Time

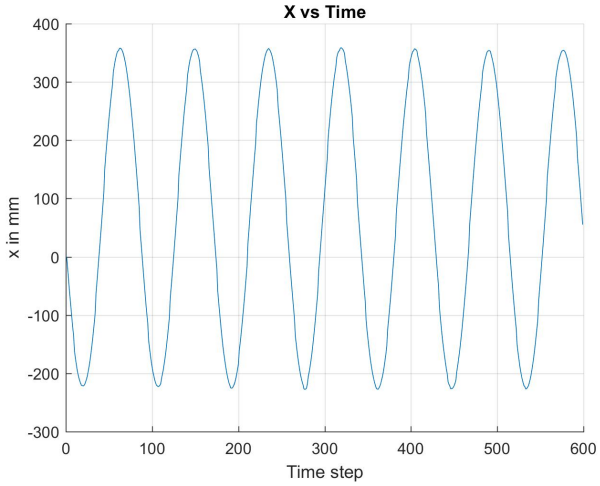

Figure C.6: log1:X vs Time

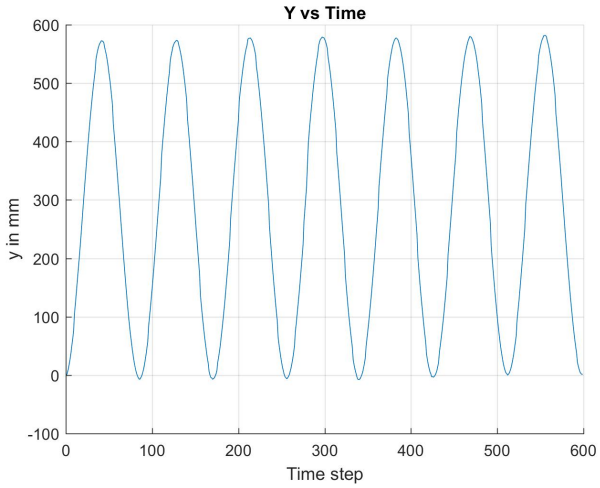

Figure C.7: log1:Y vs Time

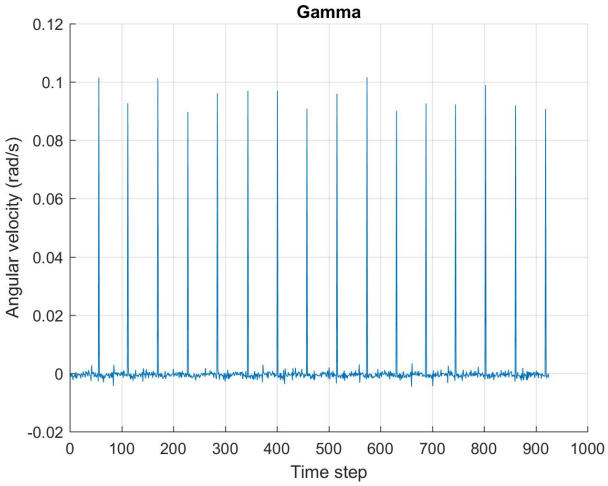

Figure C.8: log2:Gamma vs Time

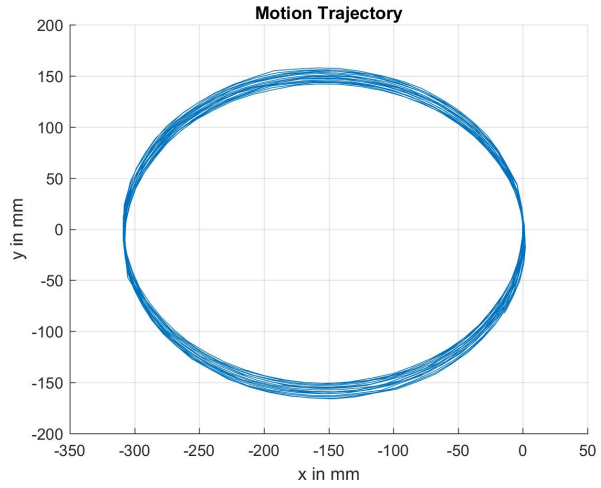

Figure C.9: log2:Motion Trajectory

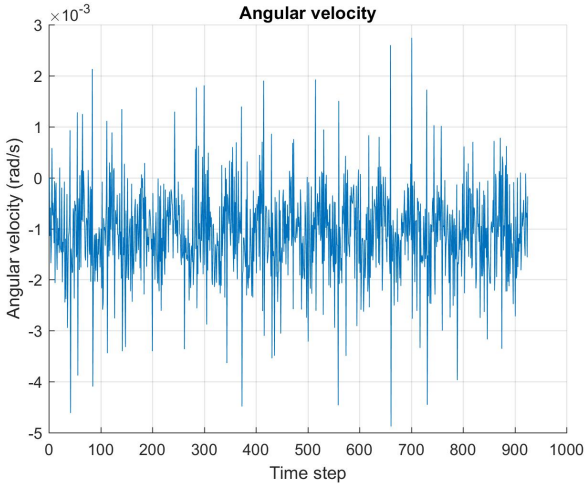

Figure C.10: log2:Omega vs Time

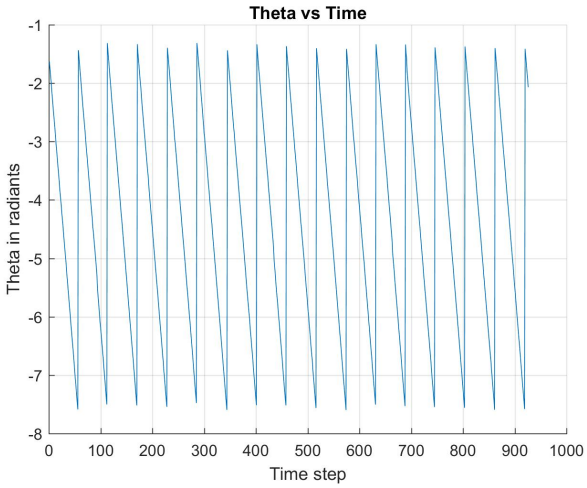

Figure C.11: log2:Theta vs Time

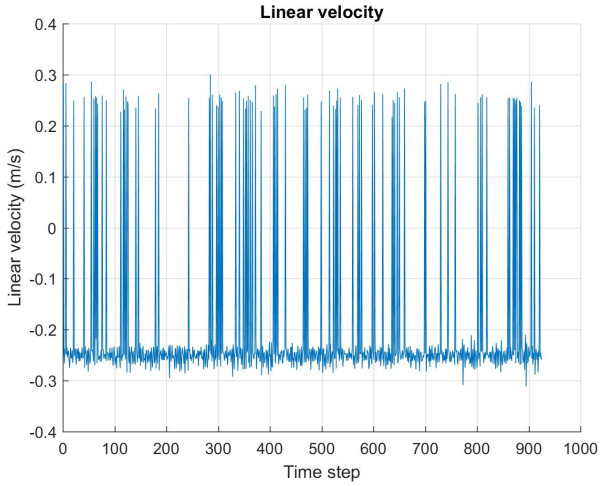

Figure C.12: log2:V vs Time

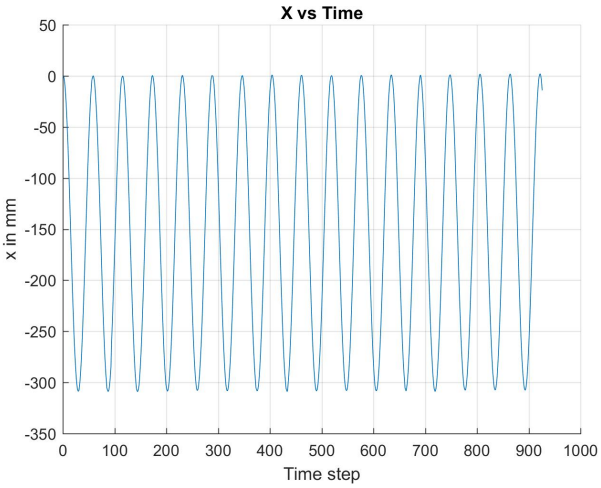

Figure C.13: log2:X vs Time

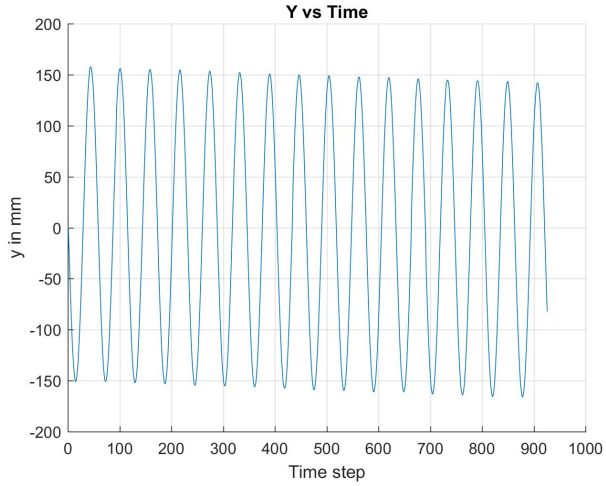

Figure C.14: log2:Y vs Time

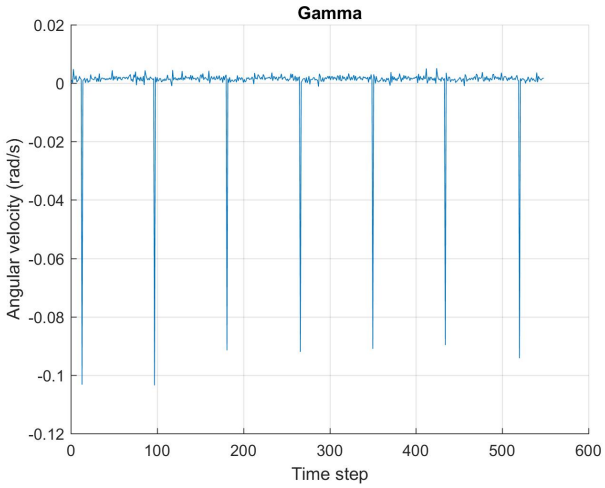

Figure C.15: log3:Gamma vs Time

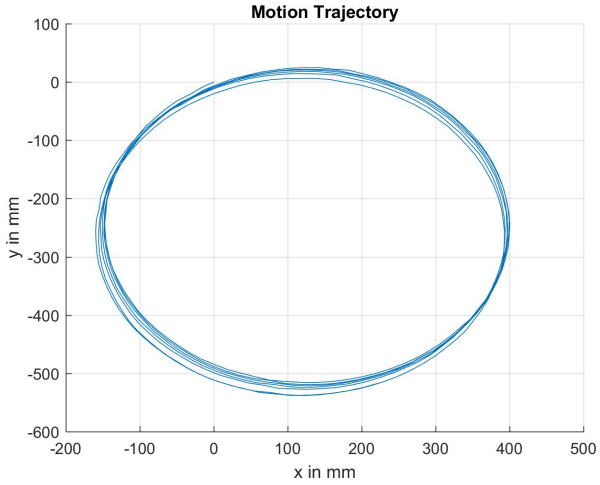

Figure C.16: log3:Motion Trajectory

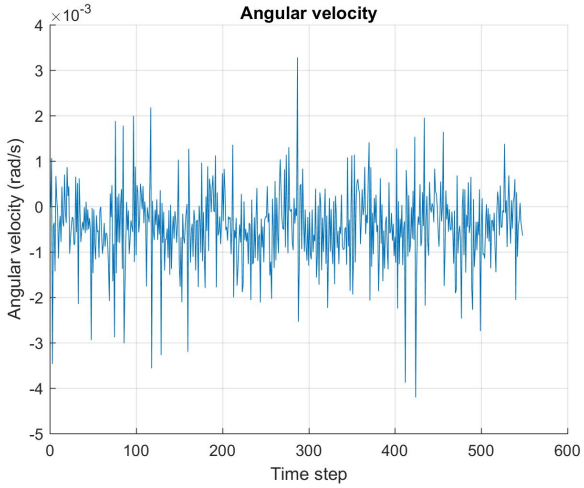

Figure C.17: log3:Omega vs Time

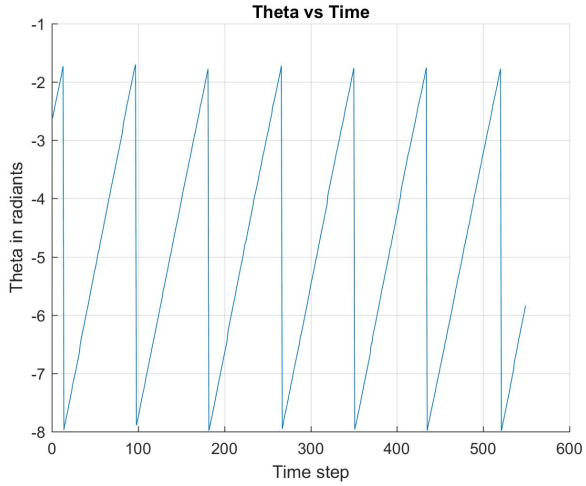

Figure C.18: log3:Theta vs Time

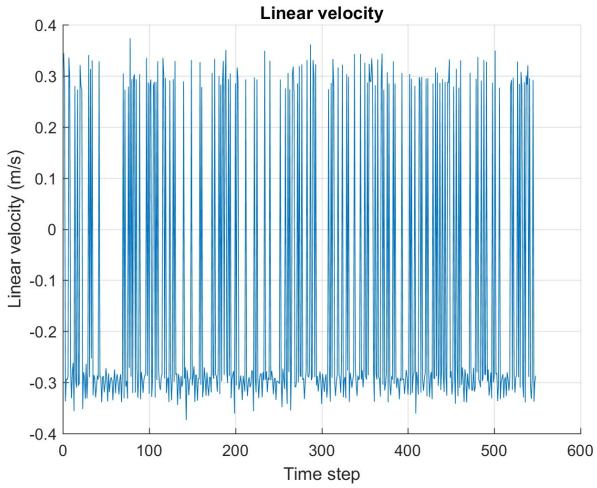

Figure C.19: log3:V vs Time

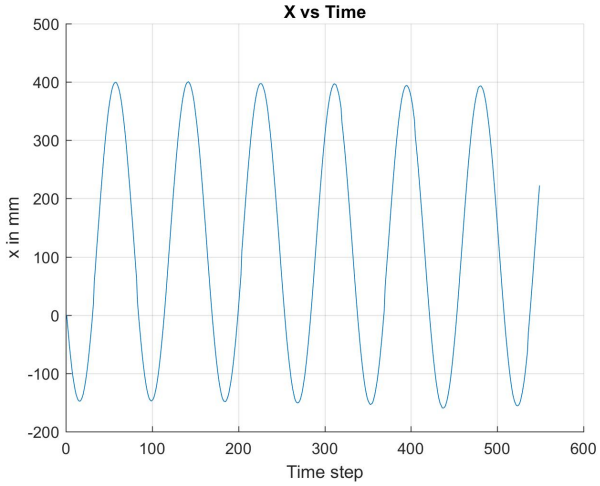

Figure C.20: log3:X vs Time

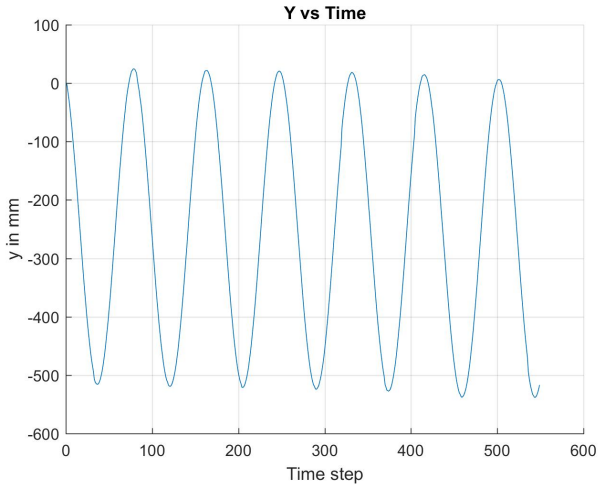

Figure C.21: log3:Y vs Time

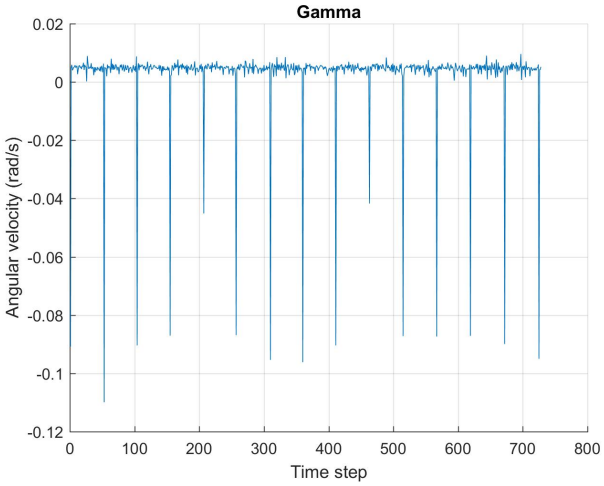

Figure C.22: log4:Gamma vs Time

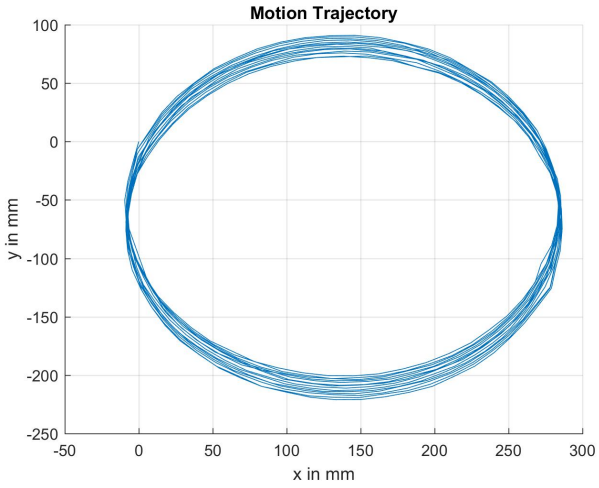

Figure C.23: log4:Motion Trajectory

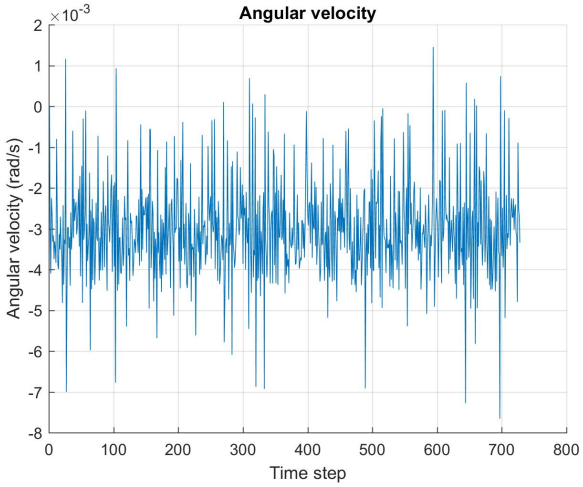

Figure C.24: log4:Omega vs Time

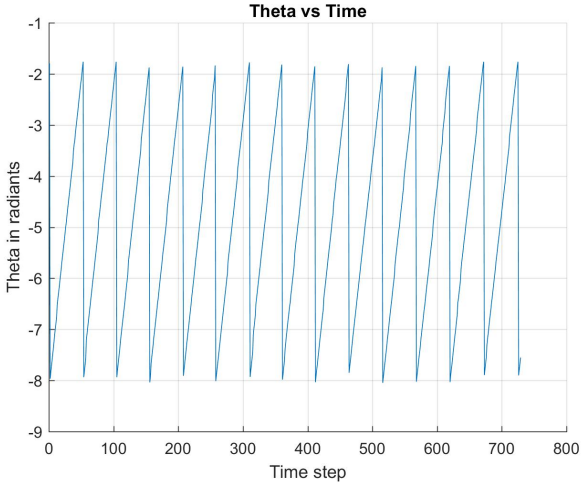

Figure C.25: log4:Theta vs Time

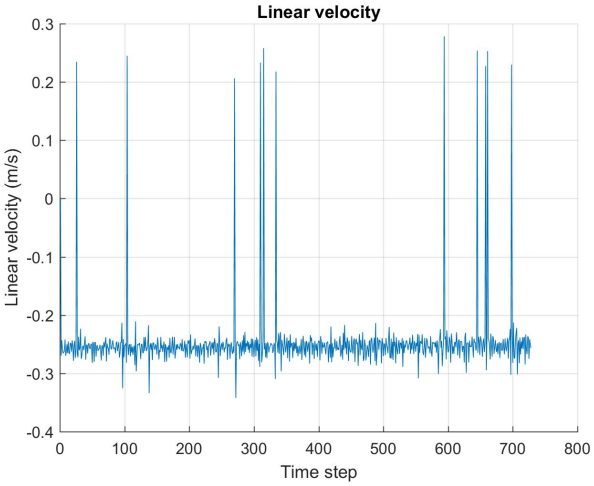

Figure C.26: log4:V vs Time

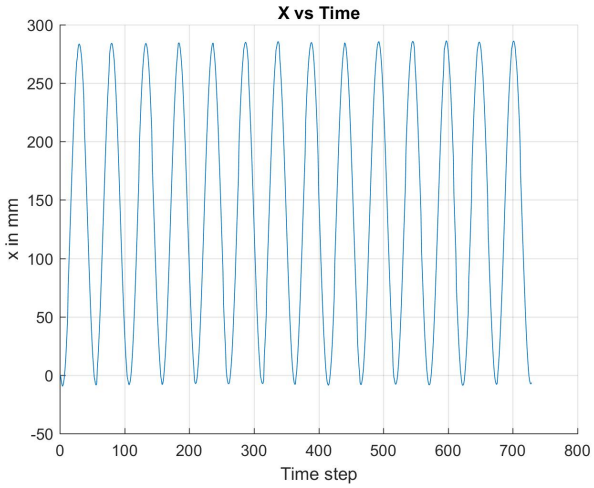

Figure C.27: log4:X vs Time

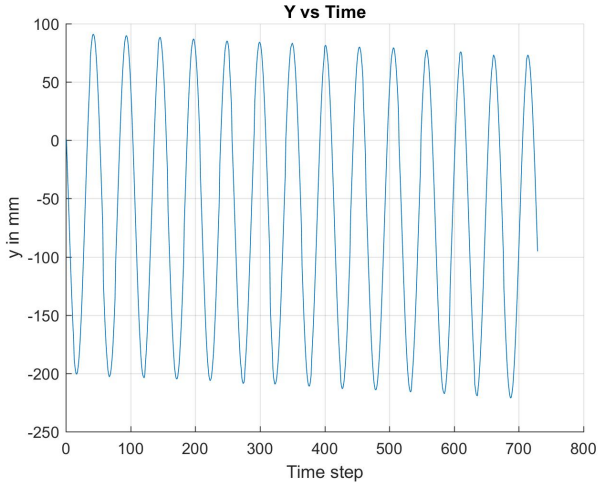

Figure C.28: log4:Y vs Time

D. Source Code

D.1. Compute Alpha

```
data1 = analyzeMotionLog('./logs1/right.log',1);
data2 = analyzeMotionLog('./logs2/right.log',9);
data3 = analyzeMotionLog('./logs1/left.log',17);
data4 = analyzeMotionLog('./logs2/left.log',25);

%A.ALPHA = SIGMA
Ar = abs([data1.v data1.omega 0 0 0 0;
     0 0 data1.v data1.omega 0 0;
     0 0 0 0 data1.v data1.omega;
     data2.v data2.omega 0 0 0 0;
     0 0 data2.v data2.omega 0 0;
     0 0 0 0 data2.v data2.omega]);
SIGMAr = [data1.sigma1; data1.sigma2; data1.sigma3; data2.sigma1; data2.sigma2; data2.sig
ALPHAr = inv(Ar)*SIGMAr;

Al = abs([data3.v data3.omega 0 0 0 0;
     0 0 data3.v data3.omega 0 0;
     0 0 0 0 data3.v data3.omega;
     data4.v data4.omega 0 0 0 0;
     0 0 data4.v data4.omega 0 0;
     0 0 0 0 data4.v data4.omega]);
SIGMAl = [data3.sigma1; data3.sigma2; data3.sigma3; data4.sigma1; data4.sigma2; data4.sig
ALPHAl = pinv(Al)*SIGMAr;
```

```
A = vertcat(Ar,Al);
SIGMA = vertcat(SIGMAr, SIGMAl);
ALPHA = pinv(A)*SIGMA;
display(ALPHA);
```

D.2. ANAYLIZE MOTION LOG

```
%Read the log file
%file = './logs1/right.log';
function [data] = analyzeMotionLog(file,startingFigure)
read = importdata(file,' ',0);
c = 400; % The number of log entries to be cropped at start and end
e = size(read,1) - c;
data.time = read(c:e,1);
data.x = read(c:e,2);
data.y = read(c:e,3);
data.orientation = read(c:e,4);
data.duration = data.time(end) - data.time(1);

%Make the start at 0,0,0
% Get orientation
    offsetX = data.x(1);
    offsetY = data.y(1);
    offsetOrientation = data.orientation(1) - atan2(data.y(2)-data.y(1),data.x(2)-data.x


    for j = 1:length(data.time)
        % translate
        data.x(j)= data.x(j) - offsetX;
        data.y(j) = data.y(j) - offsetY;
        data.orientation(j) = data.orientation(j) - offsetOrientation;

%         % rotate
%         data.x(j) = cos(-offsetOrientation )*x...
%             - sin(-offsetOrientation)*y;
%         data.y(j) = sin(-offsetOrientation)*x...
%             + cos(-offsetOrientation)*y;
%         data.orientation(j) = orientation;
    end

%Visualize the motion trajectory
% Plot
figure(startingFigure+1);clf;hold on;
```

```
plot(data.x, data.y);
title('Motion Trajectory');
xlabel('x in mm');
ylabel('y in mm');
grid on;

figure(startingFigure+2);clf;hold on;
plot(data.x);
title('X vs Time');
xlabel('Time step');
ylabel('x in mm');
grid on;

figure(startingFigure+3);clf;hold on;
plot(data.y);
title('Y vs Time');
xlabel('Time step');
ylabel('y in mm');
grid on;

figure(startingFigure+4);clf;hold on;
plot(data.orientation);
title('Theta vs Time');
xlabel('Time step');
ylabel('Theta in radiants');
grid on;



%Calculate the motion errors
data.vHat = double.empty(length(data.time)-1,0);
data.omegaHat = double.empty(length(data.time)-1,0);
data.gammaHat = double.empty(length(data.time)-1,0);
data.deltaT = double.empty(length(data.time)-1,0);

for i = 2:length(data.time)
    x = data.x(i-1);
    xNew = data.x(i);
    y = data.y(i-1);
    yNew = data.y(i);
    theta = data.orientation(i-1);
    thetaNew = data.orientation(i);
    deltaT = (data.time(i) - data.time(i-1));
```

```
    mu = 0.5*((x-xNew)*cos(theta) + (y-yNew)*sin(theta))/...
        ((y-yNew)*cos(theta) - (x-xNew)*sin(theta));
    xCircle = (x+xNew)/2 + mu*(y-yNew);
    yCircle = (y+yNew)/2 + mu*(xNew-x);
    rCircle = sqrt((x-xCircle)^2 + (y-yCircle)^2);
    deltaTheta = angdiff(atan2(yNew - yCircle, xNew - xCircle) ,... %Important .. wrong
        atan2(y-yCircle, x-xCircle));
    data.vHat(i-1) = deltaTheta/deltaT * rCircle;
    data.omegaHat(i-1) = deltaTheta/deltaT;
    data.gammaHat(i-1) = (thetaNew - theta)/deltaT - data.omegaHat(i-1);
        data.deltaT(i-1) = theta;
end

%Visualize the linear velocity
figure(startingFigure+5);clf;hold on;
plot(data.vHat);
title('Linear velocity');
xlabel('Time step');
ylabel('Linear velocity (m/s)');
grid on;

%Visualize the angular velocity
figure(startingFigure+6);clf;hold on;
grid on;
plot(data.omegaHat);
title('Angular velocity');
xlabel('Time step');
ylabel('Angular velocity (rad/s)');

%Visualize the gamma
figure(startingFigure+7);clf;hold on;
grid on;
plot(data.gammaHat);
title('Gamma');
xlabel('Time step');
ylabel('Angular velocity (rad/s)');

data.v = mean(data.vHat);
data.omega = mean(data.omegaHat);
data.gamma = mean(data.gammaHat);
data.sigma1 = std(data.vHat);
data.sigma2 = std(data.omegaHat);
data.sigma3 = std(data.gammaHat);
```

```
end
```