ELEC-E8101 Group project:
Lab A report
Group 05

Swaminathan Aravind, Kamath Abhijit Krishnananda, Karal Puthanpura JithinLal Dev

November 3, 2019

Reporting of Task 4.1

The given Linearized equations of Motions are:

$$(I_b + m_b l_b^2)\ddot{\theta_b} = m_b l_b g\theta_b - m_b l_b \ddot{x_w} - \frac{2K_t}{R_m}v_m + (\frac{2K_eK_t}{R_m} + b_f)(\frac{\dot{x_w}}{l_w} - \dot{\theta_b})$$

$$(\frac{I_w}{l_w} + l_w m_b + l_w m_w)\ddot{x_w} = -m_b l_b l_w \ddot{\theta_b} + \frac{2K_t}{R_m}v_m - (\frac{2K_eK_t}{R_m} + b_f)(\frac{\dot{x_w}}{l_w} - \dot{\theta_b})$$

Solving the above equations (as suggested in the hint) we get

$$\begin{pmatrix} m_b l_b & (I_b + m_b l_b^2) \\ (\frac{I_w}{l_w} + l_w m_b + l_w m_w) & m_b l_b l_w \end{pmatrix} \begin{pmatrix} \ddot{x_w} \\ \ddot{\theta_b} \end{pmatrix} =$$

$$\begin{pmatrix} 0 & (\frac{2K_eK_t}{R_m}) + \frac{b_f}{l_w} & m_b l_b g & -\frac{2K_eK_t}{R_m} - b_f \\ 0 & -(\frac{2K_eK_t}{R_m}) - \frac{b_f}{l_w} & 0 & \frac{2K_eK_t}{R_m} + b_f \end{pmatrix} \begin{pmatrix} x_w \\ \dot{x_w} \\ \theta_b \\ \dot{\theta_b} \end{pmatrix} + \begin{pmatrix} -\frac{2K_t}{R_m} \\ \frac{2K_t}{R_m} \end{pmatrix} u \quad (1)$$

4.1.1 The choice of X is

$$\begin{bmatrix} x_w \\ \dot{x_w} \\ \theta_b \\ \dot{\theta_b} \end{bmatrix}$$

The input to the system is

$$u = v_m$$

and the output of the system is

$$y = \theta_b$$

4.1.2 The equations are represented in the State space form. The matrices are

$$\alpha = \begin{bmatrix} 0 & (\frac{2K_eK_t}{R_m}) + \frac{b_f}{l_w} & m_b l_b g & -\frac{2K_eK_t}{R_m} - b_f \\ 0 & -(\frac{2K_eK_t}{R_m}) - \frac{b_f}{l_w} & 0 & \frac{2K_eK_t}{R_m} + b_f \end{bmatrix}$$

$$\beta = \begin{bmatrix} -\frac{2K_t}{R_m} \\ \frac{2K_t}{R_m} \end{bmatrix}$$

$$\gamma = \begin{bmatrix} m_b l_b & (I_b + m_b l_b^2) \\ (\frac{I_w}{l_w} + l_w m_b + l_w m_w) & m_b l_b l_w \end{bmatrix}$$

Now, with the above mentioned matrices,we get

$$A = \gamma^{-1}\alpha$$

$$B = \gamma^{-1}\beta$$

Also we chose the matrix X as 4*1 matrix so we assume ,

$$\dot{X} = \begin{bmatrix} \dot{x_w} \\ \ddot{x_w} \\ \dot{\theta_b} \\ \ddot{\theta_b} \end{bmatrix}$$

After derivations , we get the new matrices as

$$\gamma_{new} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & m_b l_b & (I_b + m_b l_b^2) & 0 \\ 0 & (\frac{I_w}{l_w} + l_w m_b + l_w m_w) & m_b l_b l_w & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\alpha_{new} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & (\frac{2K_e K_t}{R_m}) + \frac{b_f}{l_w} & m_b l_b g & -\frac{2K_e K_t}{R_m} - b_f \\ 0 & -(\frac{2K_e K_t}{R_m}) - \frac{b_f}{l_w} & 0 & \frac{2K_e K_t}{R_m} + b_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\beta_{new} = \begin{bmatrix} 0 \\ -\frac{2K_t}{R_m} \\ 0 \\ \frac{2K_t}{R_m} \end{bmatrix}$$

And the updated matrices for A, B, C and D are

$$A = \gamma_{new}^{-1}\alpha_{new}$$

$$B = \gamma_{new}^{-1}\beta_{new}$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$$

$$D = 0$$

4.1.3 :The numeric forms of A,B,C and D matrices are

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -773.8 & -6.6 & 16.2 \\ 0 & 0 & 0 & 1 \\ 0 & 3313.2 & 63.1 & -69.6 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 36.5980 \\ 0 \\ -156.7072 \end{bmatrix}$$
$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$$
$$D = 0$$

Reporting of Task 4.2

4.2.1 : The transfer function of the system is computed from the matrices A,B,C and D using Matlab's inbuilt functions

$$G(s) = \frac{-156.7s}{(s + 843.4)(s + 5.64)(s - 5.679)}$$

For the calculation of the transfer function of the system we used two methods,
Method 1 : sys = ss(A,B,C,D); H = tf(sys); where the above method gave us the result mentioned in 4.2.1
Method 2 : [num den]= ss2tf(A,B,C,D); H = tf(num,den);
The result we got is

$$G(s) = \frac{-156.7s^2 + 1.866e^-11 + 3.48e^-14}{s^4 + 843.3s^3 - 63.07s^2 - 2.702e04s}$$

When we use this G(s) in the next step (4.2.2) and plot the step response of the closed loop system, we weren't getting a proper result. The step response curve was very sharp. We aren't sure what difference does this make of generating the transfer function in two different methods. Since the first method had a proper smooth step response, we are proceeding with the first method and taking that result. 4.2.2 : There wasn't many problems with the matlab numeric form , but it wasn't considering the e^{-11}, e^{-14} terms to zero. we had considered those terms as minimal .i.e to zero.

Reporting of Task 4.3

4.3.1. Equation of PID was considered as:

$$C(s) = K_p + K_i\frac{1}{s} + K_ds = \frac{K_ps + K_i + Kds^2}{s} = \frac{N}{D}$$

The poles of the plant transfer function were:
$P_1 = -843.3571$, $P_2 = -5.6416$, $P_3 = 5.679$
The pole P_3 lies on the right hand side of the s-plane, this causes our system to be unstable. We have to find the value of K_p, K_i and K_d such that the closed loop transfer function is stable. We have to replace P_3 with a pole that lies on the LHS of the s-plane. Here we choose our new pole to be $p = -3$. It is chosen in such a way that, the new pole does not effect the existing dominant pole and also lie on the RHS of s-plane.

$$\text{Closed loop transfer function } \; P(s) = \frac{CG}{1 + CG}$$

4.3.2.Hence Using pole placement method K_p, K_i and K_d were calculated as

$$K_p = -47.2363, \; K_i = -264.7467, \; K_d = -0.0556$$

4.3.3.The resulting closed loop transfer function is:

$$Cs = \frac{s(s + 843.4189)(s + 5.6421)}{s(s + 843.3571)(s + 5.6416)(s + 3.0014)}$$

Reporting of Task 4.4

4.4.1: The equations of motion after adding the disturbance d is

$$(I_b + m_b l_b^2)\ddot{\theta_b} = m_b l_b g \sin\theta_b - m_b l_b \ddot{x_w} \cos\theta_b - \frac{2K_t}{R_m} v_m + (\frac{2K_e K_t}{R_m} + b_f)(\frac{\dot{x_w}}{l_w} - \dot{\theta_b}) + l_b \cos\theta_b d$$

and

$$(\frac{I_w}{l_w} + l_w m_b + l_w m_w)\ddot{x_w} = -m_b l_b \ddot{\theta_b} \cos\theta_b + m_b l_b l_w \dot{\theta_b}^2 \sin\theta_b + \frac{2K_t}{R_m} v_m - (\frac{2K_e K_t}{R_m} + b_f)(\frac{\dot{x_w}}{l_w} - \dot{\theta_b}) + l_w d \quad (2)$$

4.4.2: Linearizing the equations above we consider

$$\sin\theta_b = \theta_b; \theta_b^2 = 0; \ddot{\theta_b}\cos\theta_b = \ddot{\theta_b}; \ddot{x_w}\cos\theta_b = \ddot{x_w}$$

After considering the above equations we get

$$(I_b + m_b l_b^2)\ddot{\theta_b} = m_b l_b g \theta_b - m_b l_b \ddot{x_w} - \frac{2K_t}{R_m} v_m + (\frac{2K_e K_t}{R_m} + b_f)(\frac{\dot{x_w}}{l_w} - \dot{\theta_b}) + l_b d$$

$$(\frac{I_w}{l_w} + l_w m_b + l_w m_w)\ddot{x_w} = -m_b l_b l_w \ddot{\theta_b} + \frac{2K_t}{R_m} v_m - (\frac{2K_e K_t}{R_m} + b_f)(\frac{\dot{x_w}}{l_w} - \dot{\theta_b}) + l_w d$$

Matrices in numeric form:

$$\alpha_d = \begin{bmatrix} 0 & (\frac{2K_e K_t}{R_m}) + \frac{b_f}{l_w} & m_b l_b g & -\frac{2K_e K_t}{R_m} - b_f \\ 0 & -(\frac{2K_e K_t}{R_m}) - \frac{b_f}{l_w} & 0 & \frac{2K_e K_t}{R_m} + b_f \end{bmatrix}$$

$$\beta_d = \begin{bmatrix} -\frac{2K_t}{R_m} & l_b \\ \frac{2K_t}{R_m} & l_w \end{bmatrix}$$

$$\gamma_d = \begin{bmatrix} m_b l_b & (I_b + m_b l_b^2) \\ (\frac{I_w}{l_w} + l_w m_b + l_w m_w) & m_b l_b l_w \end{bmatrix}$$

The updated matrices after the 4*4 contraint is

$$\gamma_{dnew} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & m_b l_b & (I_b + m_b l_b^2) & 0 \\ 0 & (\frac{I_w}{l_w} + l_w m_b + l_w m_w) & m_b l_b l_w & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\alpha_{dnew} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & (\frac{2K_e K_t}{R_m}) + \frac{b_f}{l_w} & m_b l_b g & -\frac{2K_e K_t}{R_m} - b_f \\ 0 & -(\frac{2K_e K_t}{R_m}) - \frac{b_f}{l_w} & 0 & \frac{2K_e K_t}{R_m} + b_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\beta_{dnew} = \begin{bmatrix} 0 & 0 \\ -\dfrac{2K_t}{R_m} & l_b \\ 0 & \\ \dfrac{2K_t}{R_m} & l_w \end{bmatrix}$$

The final matrices value for disturbance input is

$$A = \gamma_{dnew}^{-1}\alpha_{dnew}$$

$$B = \gamma_{dnew}^{-1}\beta_{dnew}$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$$

$$D = 0$$

Values for matrices:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -773.8 & -6.6 & 16.2 \\ 0 & 0 & 0 & 1 \\ 0 & 3313.2 & 63.1 & -69.6 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 36.5980 & 1.8795 \\ 0 & 0 \\ -156.7072 & 1.0797 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$$

$$D = 0$$

Reporting of Task 4.5

4.5.1: The simulink model we used is shown in Figure 1.

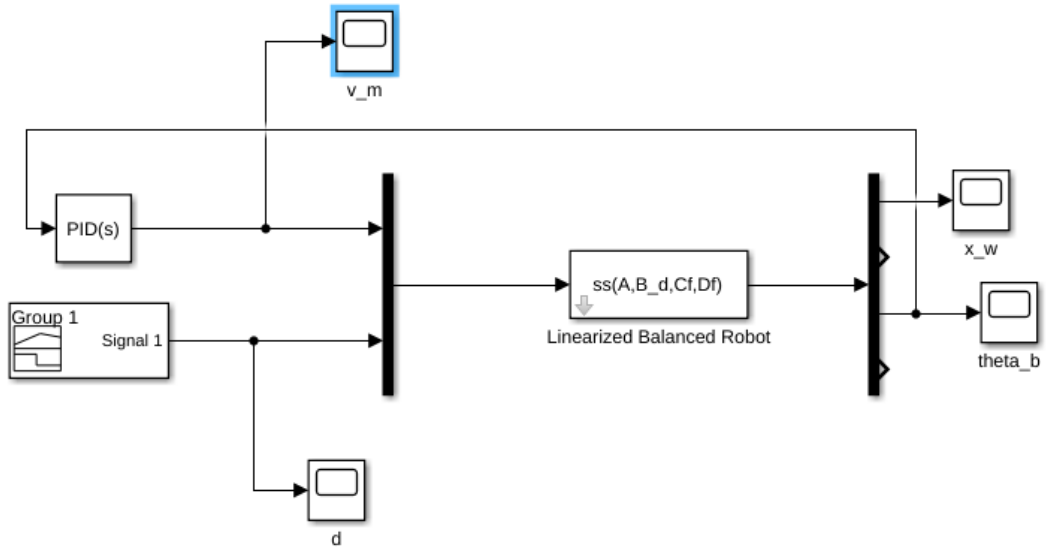

Figure 1: Simulink model

We initially had some trouble with setting the values of kp,ki,kd in PID block, because when we tried with the same negative values what we got in the previous steps, our system response was negative. So we had to change the values of PID parameters to positive.

4.5.2: The realization of $d(t), v_m(t), x_m(t), \theta_b(t)$ are shown in Figure 2,Figure 3,Figure 4,Figure 5 respectively

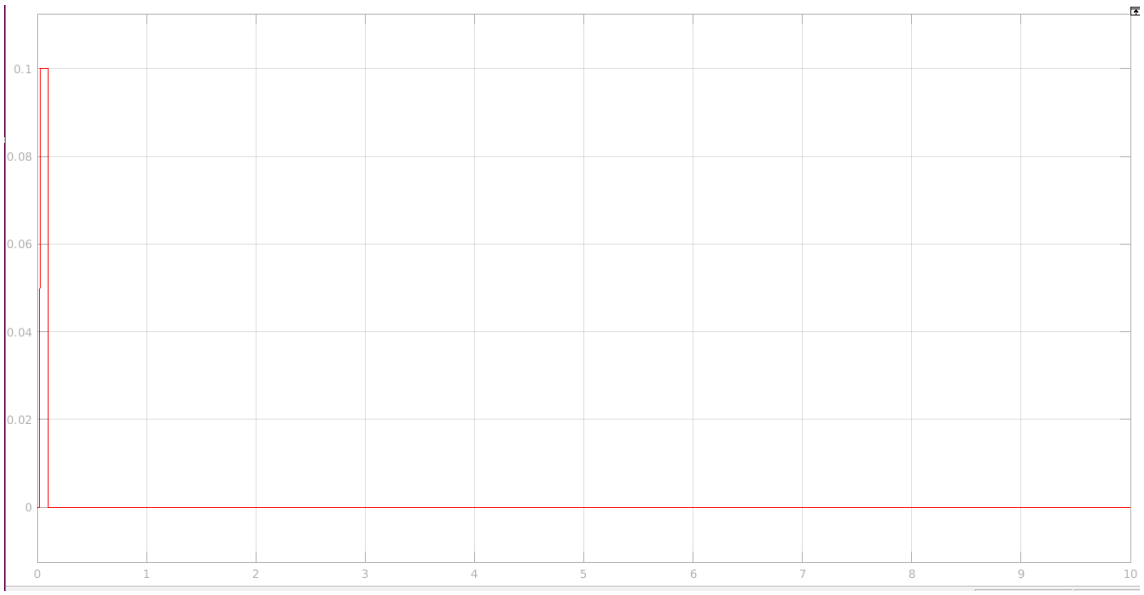

Figure 2: d(t)

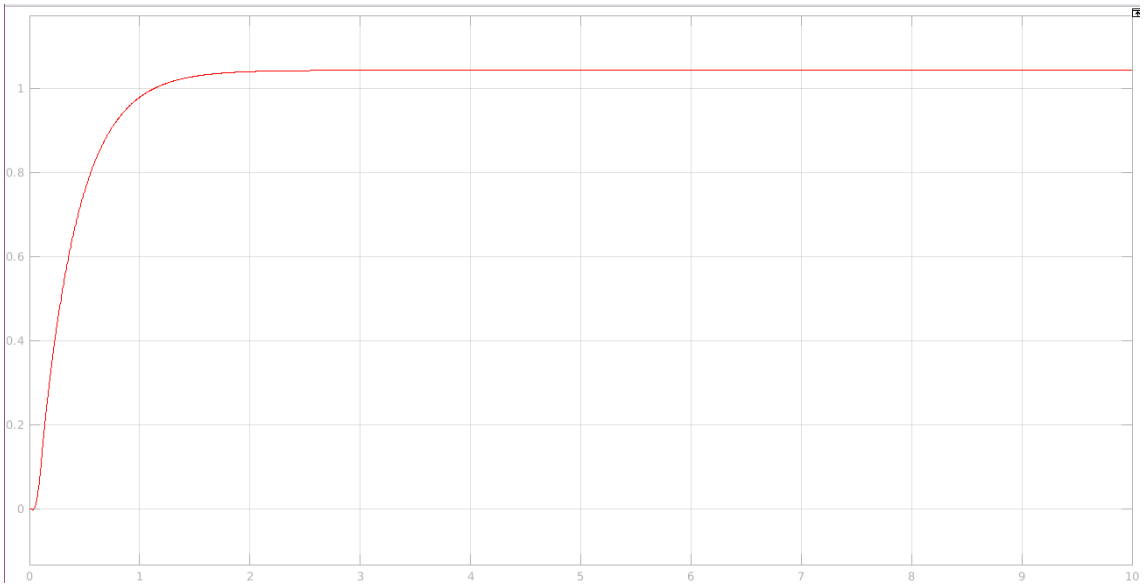

Figure 3: $v_m(t)$

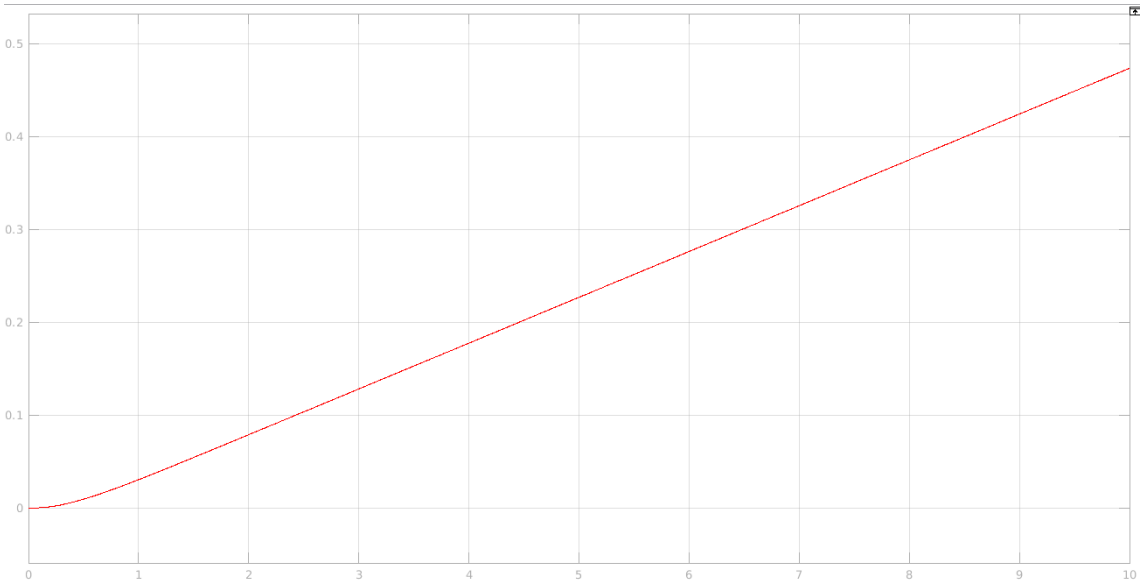

Figure 4: $x_w(t)$

Figure 5: $\theta_b(t)$

Reporting of Task 4.6

4.6.1: The calculated bandwidth of the system is $11.4rad/s \approx 1.8Hz$.

4.6.2: The sampling period we calculated is $108Hz \approx 110Hz$.

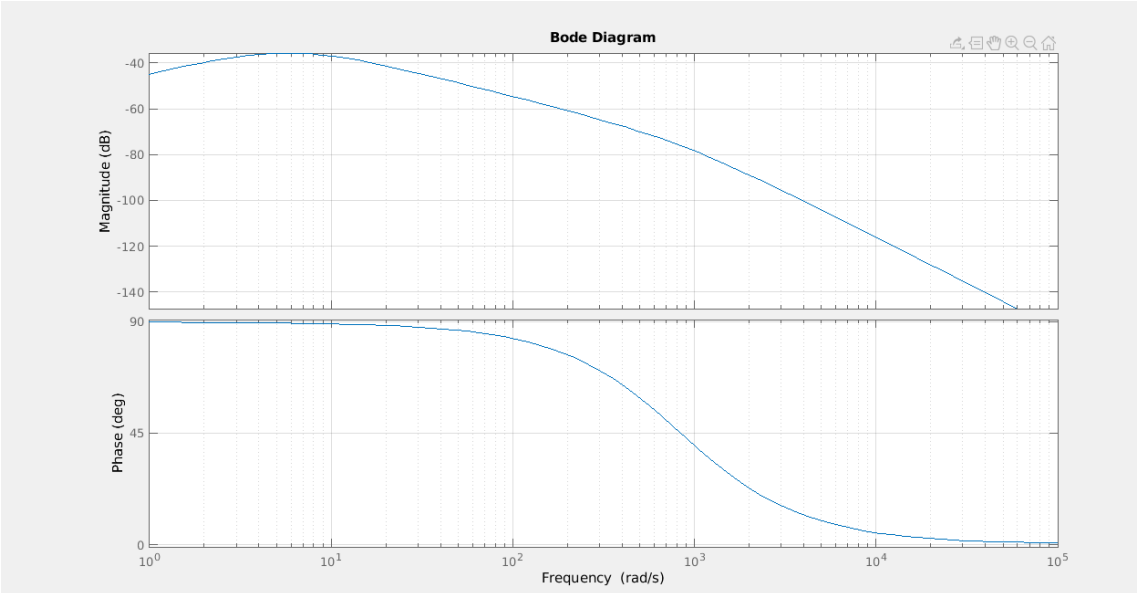

Figure 6: bode plot for the transfer function

4.6.3: Justification for 4.6.1 and 4.6.2 - We were initially confused with the question as in what to proceed with, to compute the bandwidth of the system or compute the bandwidth of open loop system. We discussed with research group and concluded that we

have to get the bandwidth of the open loop system We get the bode plot of the open loop system(plant) response in Figure 6. So Bandwidth is as follows: Lower Freq = $2.31rad/s$ Upper Freq = $13.8rad/s$ BW = Upper freq - Lower freq = $11.4rad/s = 1.8287Hz$.

To get the sampling period, with Nyquist Theorem, also considering the Arduino's processing speed as mentioned in the lab book, we take 30 times the value we got from Nyquist theorm. We compute BW $= 2 * 30 * 1.8287 = 108Hz \approx 110Hz$. So the sampling time we chose is $T_s = 1/BW = 0.01s$

4.6.4: The final discrete time transfer function is :

$$H(Z) = \frac{-0.001639z^2 + 0.001419z + 0.0002199}{z^3 - 2.004z^2 + 1.001z - 0.0002174}$$

Reporting of Task 4.7

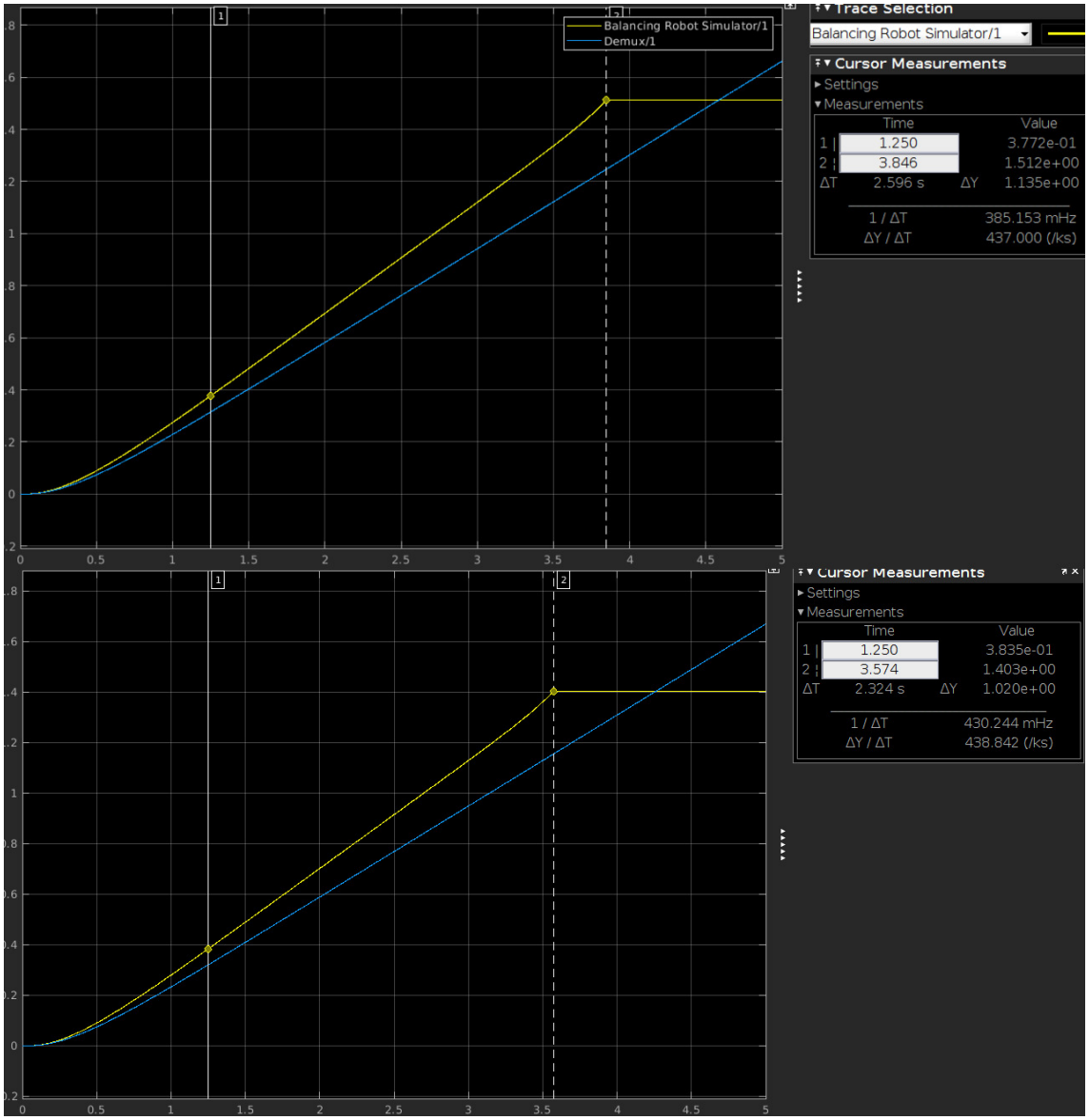

Figure 7: Difference in robot fall time value for continuous and discrete

4.7.1: The smallest values of $\bar{d}$ for which the robot falls : in continuous case: 0.584 in discrete case : 0.584 (Both the cases, it was same , thought it may differ) The only difference we observe is In the Continuous case, the robot falls at 3.84s. But at the discrete case, it falls early 3.57s as shown in Figure 7.

4.7.2 : Realizations for Continuous case: $\theta_b(t) vs \theta_b^{lin}(t), v_m(t) vs v_m^{lin}(t)$ are found in Figure 8, Figure 9 respectively. Realizations for Discrete case: $\theta_b(t) vs \theta_b^{lin}(t), v_m(t) vs v_m^{lin}(t)$

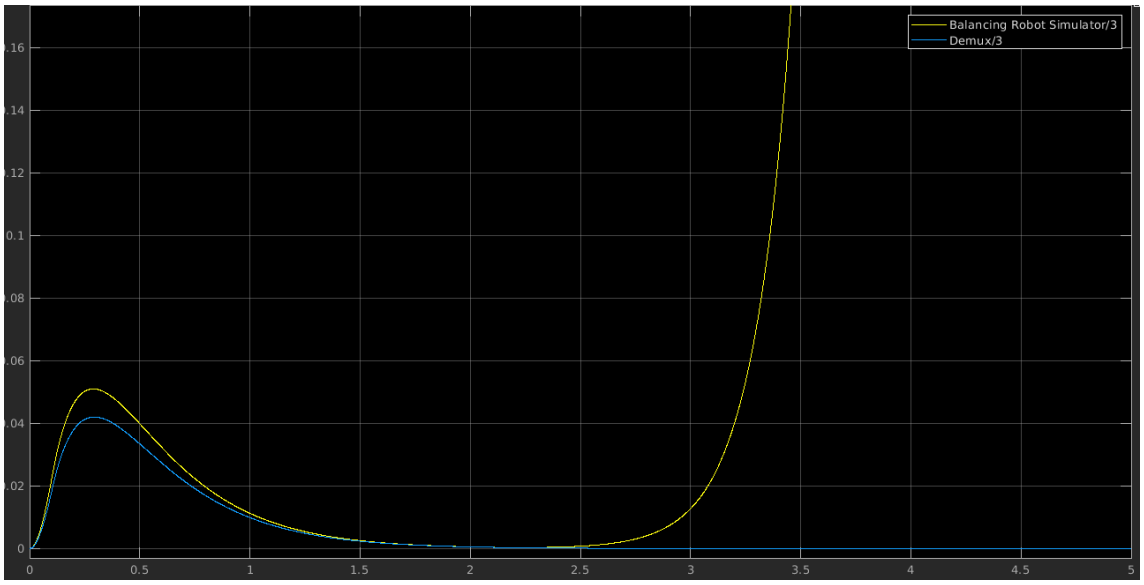

Figure 8: $\theta_b(t) vs \theta_b^{lin}(t)$

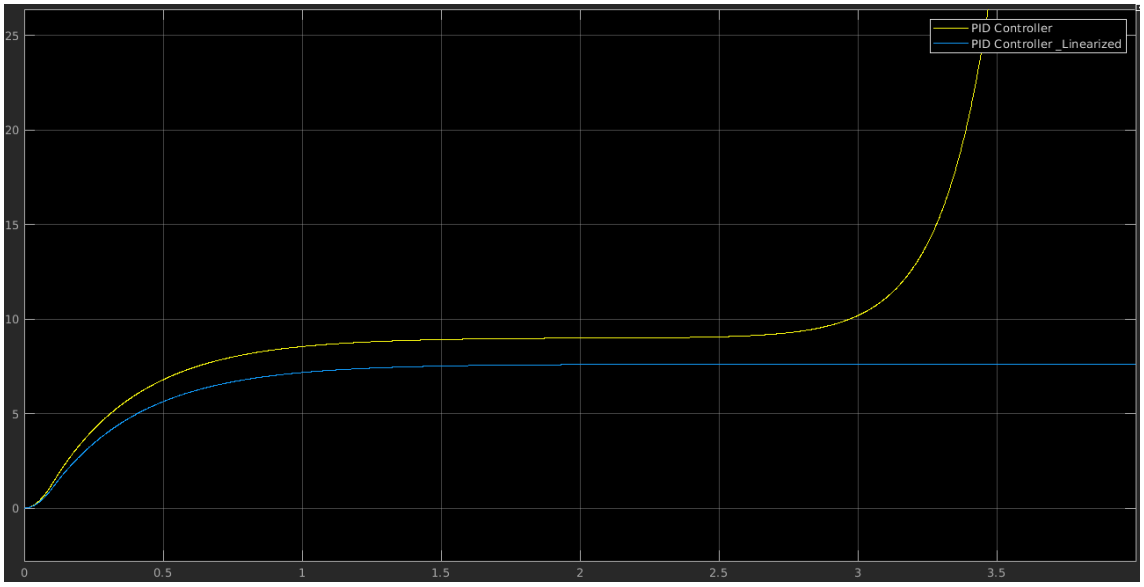

Figure 9: $v_m(t) vs v_m^{lin}(t)$

are found in Figure 10, Figure 11 respectively.

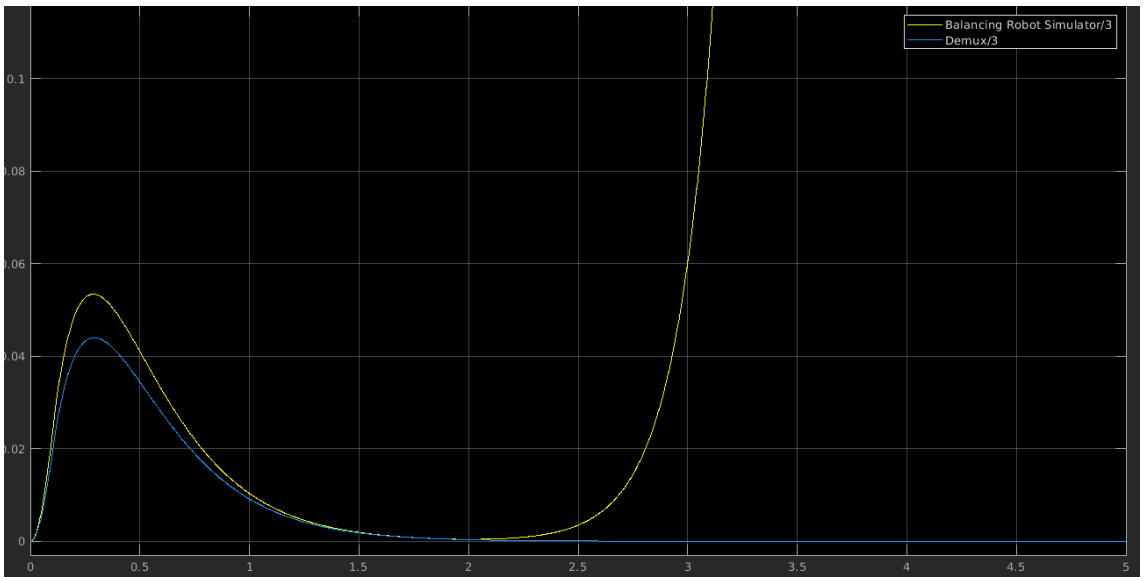

Figure 10: $\theta_b(t) vs \theta_b^{lin}(t)$

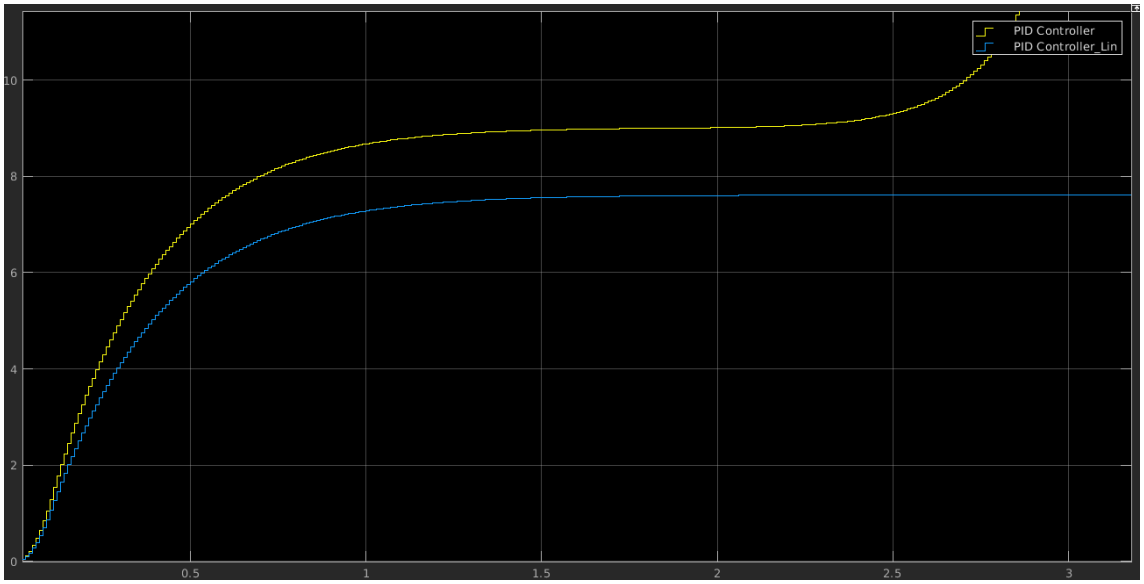

Figure 11: $v_m(t) vs v_m^{lin}(t)$

4.7.3: These experiments were used to relate how the linearization of the equations of motion helped in stabilizing the robot. From the Figure 7, we can see that linearizing the robot will still have the robot in stable position. But anyways, this is simulating the robot, we don't know, how it works in the real case. Also to know if we can tune the PID controller, we are not sure how to tune the PID controller with this setup.

Appendix A : Matlab Codes for the solutions:

Constants.m:

```
clc;
clear all;
g = 9.8;
b_f = 0;
m_b = 0.463;
l_b = 0.113;
I_b = 0.00767;
m_w = 0.026;
l_w = 0.021;
I_w = 0.00000573;
R_m = 4.4;
L_m = 0;
b_m = 0;
K_e = 0.444;
K_t = 0.470;
```

Task_4_1.m:

```
clc;
% %syms g b_f m_b l_b I_b m_w l_w I_w R_m L_m b_m K_e K_t
a_11 = 0;
a_12 = (2*K_e*K_t/R_m + b_f)/I_w;
a_13 = m_b*l_b*g;
a_14 = -(2*K_e*K_t/R_m + b_f);
a_21 = 0;
a_22 = -(2*K_e*K_t/R_m + b_f)/I_w ;
a_23 = 0;
a_24 = (2*K_e*K_t/R_m + b_f);
b_1 = -(2*K_t)/R_m;
b_2 = (2*K_t)/R_m;
g_11 = m_b*l_b;
g_12 = I_b + (m_b*(l_b^2));
g_21 = (I_w/I_w)+(I_w*m_b)+(I_w*m_w);
g_22 = m_b*l_b*I_w;
Gamma = [1 0 0 0;0 g_11 0 g_12;0 0 1 0;0 g_21 0 g_22];
Alpha = [0 1 0 0; a_11 a_12 a_13 a_14;0 0 0 1;a_21 a_22 a_23 a_24];
Beta = [0;b_1;0;b_2];
A = inv(Gamma)*Alpha;
B = inv(Gamma)*Beta;
C= [0 0 1 0];
D = 0;
```

Task_4_2.m:

```
clc;
sys = ss(A, B, C, D);
H = tf(sys);
pzplot(sys);
e = eig(A);
%sys = ss(A,B,C,D)
%[num den] = ss2tf(A,B,C,D);
%H = tf(num,den)
```

Task_4_3.m:

```
clc;
p1= e(2);
p2= e(3);
p3 = e(4);
p3n = -3.0;
k = -156.7;
kp = ((p1*p3n)+(p2*p3n) - (p1*p3) - (p2*p3))/k;
ki = ((p1*p2*p3 )- (p1*p2*p3n))/k;
kd = (p3-p3n)/k;
controller_s = pid(kp,ki,kd);
P_S = feedback((controller_s*H),1)
step(P_S);
```

Task_4_4.m:

```
clc;
b_21 = b_1;
b_22 = l_b;
b_41 = b_2;
b_42 =  l_w;
Beta_d = [0 0;b_21 b_22;0 0;b_41 b_42];
A = inv(Gamma)*Alpha;
B_d = inv(Gamma)*Beta_d;
Cf= [1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1];
Df = [0 0;0 0;0 0;0 0];
```

Task_4_6.m:

```
clc;
bode(H);
fSamplingPeriod = 0.01
H_Z = c2d(H,fSamplingPeriod);
```