Indian Institute of Technology, Mandi

Control Systems (EE301P)

Balanduino

Group - 2

Anand Kumar - B16045

Deepak Jarwal - B16054

Mukesh Kumawat - B15222

Amit Ghanghas - B15208

Contents

Abstract

The purpose is to design a PID controller for an inverted pendulum system. These types of systems are unstable and require active control for stability. For this task we have chosen to work on Balanduino which is an open source balancing robot platform that can be programmed and controlled with ease. Using system dynamics and linearising for small excursion about $180°$, system transfer function is obtained for which proportional, integral, and derivative gains are found out for different settling times and steady state errors. Then the PID gains are manually fed to the robot using the Balanduino Android app and the response of robot is recorded for various disturbances and verified with calculations.

1 Introduction

The inverted pendulum are the systems for which center of mass lies above the pivoted point. The applications of inverted pendulum are numerous ranging from humans balancing their upper body around ankle joints to high-end robots. All of such systems are unstable and require an active control for stability. Balanduino offers an excellent opportunity for Control Systems students to analyse and design a *PID controller* for inverted pendulum based systems. Balanduino comprises of a pendulum system pivoted around two wheel as shown in Fig.1. In order to balance a two-wheeled pendulum system, it is necessary to have accurate and quick information about the tilt angle of the system and compensation of the same using a PID controller. The acccurate estimation of tilt angle is made using Kalman filter implemented in the C++ code . Gyro gets drifted over time , hence Kalman filter uses past and present value of angles to provide correct tilt angle.

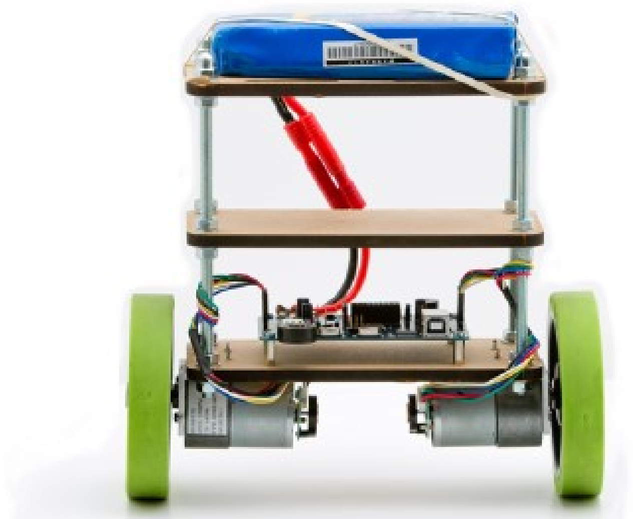


Figure 1: Balanduino [1]

2 Theory

To balance a two-wheeled inverted pendulum, it is foremost necessary to understand the physics of the inverted pendulum. Commonly, such systems are modeled as a rigid rod pivoted at a frictionless joint of a moving carriage as shown in *Figure 3(a)*. For

simplicity, the degree of freedom of pendulum is restricted in the X-Y plane and of the carriage in the X direction. The free-body diagram of the modeled system can be seen in *Figure 3(b)*.

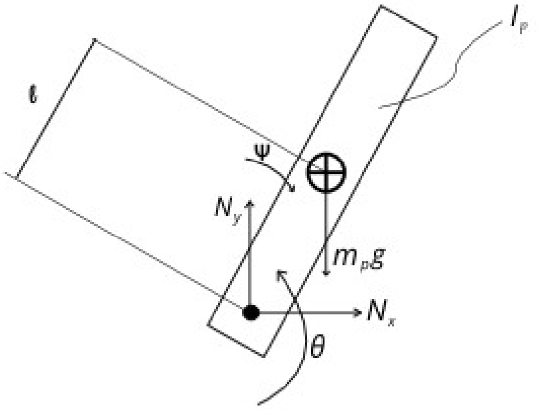


Figure 2: Forces on the Inverted Pendulum.

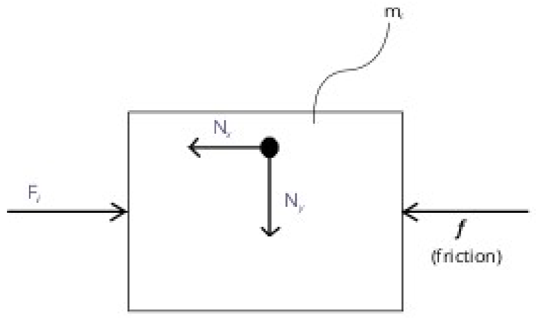


Figure 3: Forces on the Cart.

For the carriage, balancing the forces in the X direction leads to the following equation:

$$F_i = m_c\ddot{x} + f\dot{x} + N_x \tag{1}$$

where F_i is an applied force, m$_c$ is the mass of the carriage, f is a constant of friction and N_x is the contact force in the axis between the carriage and the pendulum in the X direction.

For pendulum, force equations in X direction and the direction perpendicular to the rod leads to following two equations:

$$N_x = m_p(\ddot{x} + \ddot{\theta}cos\theta - \dot{\theta}^2 lsin\theta) \tag{2}$$

$$N_y sin\theta + N_x cos\theta = m_p(\ddot{\theta}l + \ddot{x}cos\theta + gsin\theta) \tag{3}$$

Balancing torque around the center of mass of the rod gives us the following equation:

$$-N_y l sin\theta - N_x l cos\theta = I_p \ddot{\theta} \quad (4)$$

where I_p is the moment of inertia of the pendulum.

Combining equations (1) and (2)

$$F_i = (m_c + m_p)\ddot{x} + f\dot{x} + m_p(l\ddot{\theta}cos\theta - l\dot{\theta}^2 sin\theta) \quad (5)$$

Combining equations (3) and (4)

$$(I_p + m_p l^2)\ddot{\theta} + m_p(glsin\theta + l\ddot{x}cos\theta) = 0 \quad (6)$$

Clearly, equations (5) and (6) are non-linear and linearisation is done about small excursion of 180°. Substituting $\theta = \pi + \psi$ in equations (6) and (5) gives

$$(I_p + m_p l^2)(\pi \ddot{+} \psi) + m_p(glsin(\pi + \psi) + l\ddot{x}cos(\pi + \psi)) = 0$$

$$(I_p + m_p l^2)\ddot{\psi} + m_p(-glsin\psi - l\ddot{x}cos\psi) = 0, \begin{array}{l} cos\psi \approx 1, \\ sin\psi \approx -\psi \end{array}$$

$$(I_p + m_p l^2)\ddot{\psi} - m_p(gl\psi + \ddot{x}) = 0 \quad (7)$$

and

$$\begin{aligned} F_i = (m_c + m_p)\ddot{x} &+ f\dot{x} \\ &+ m_p l(\pi \ddot{+} \psi)cos(\pi + \psi) \\ &- m_p l(\pi \dot{+} \psi)^2 sin(\pi + \psi) \end{aligned}$$

$$F_i = (m_c + m_p)\ddot{x} + f\dot{x} - m_p(l\ddot{\psi}cos\psi + l\dot{\psi}^2 sin\psi)$$

$$F_i = (m_c + m_p)\ddot{x} + f\dot{x} - m_p(l\ddot{\psi} - l\dot{\psi}^2\psi), cos\psi \approx 1, sin\psi \approx -\psi$$

$$F_i = (m_c + m_p)\ddot{x} + f\dot{x} - m_p l\ddot{\psi} = u_i \quad (8)$$

For frequency domain analysis, Laplace transform of 7 and 8 gives the following equations.

$$s^2\Psi(s)(I_p + m_p l^2) - m_p(gl\Psi(s) + s^2 X(s)l) = 0 \quad (9)$$

and

$$U_i(s) = s^2 X(s)(m_c + m_p) + sX(s)f - s^2\Psi(s)m_p l \quad (10)$$

Representing Equation (9) as a Transfer Function gives

$$X(s) = \left(\frac{I_p + m_p l^2}{m_p l} - \frac{g}{s^2} \right) \Psi(s) \tag{11}$$

Substituting Equation (11) in Equation (10)

$$\begin{aligned} U_i(s) = {} & s^2 \left(\frac{I_p + m_p l^2}{m_p l} - \frac{g}{s^2} \right) \Psi(s)(m_c + m_p) \\ & + s \left(\frac{I_p + m_p l^2}{m_p l} - \frac{g}{s^2} \right) \Psi(s) f - s^2 \Psi(s) m_p l \end{aligned} \tag{12}$$

$$\frac{\Psi(s)}{U_i(s)} = G(s) = \frac{s m_p l}{\begin{array}{c} s^3((m_c + m_p)(I_p + m_p l^2) - (m_p l)^2) - f m_p g l \\ + s^2 f (I_p + m_p l^2) + s(m_c + m_p) m_p g l \end{array}} \tag{13}$$

and the Transfer Function describing the position $X(s)$ looks like

$$\frac{X(s)}{U_i(s)} = \frac{s^2 (I_p + m_p l^2) - m_p g l}{\begin{array}{c} s^4((m_c + m_p)(I_p + m_p l^2) - (m_p l)^2) - s f m_p g l \\ + s^3 f (I_p + m_p l^2) + s^2 (m_c + m_p) m_p g l \end{array}} \tag{14}$$

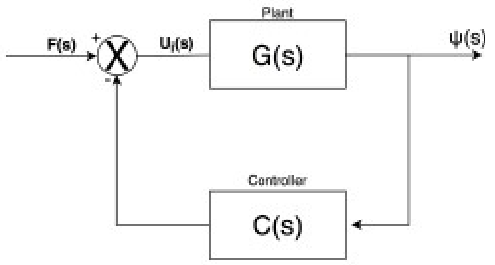


Figure 4: Reduced Block Diagram of the Feedback Control

Quantity	**Value**
m_c	0.380 Kg
m_p	0.970 Kg
I_p	0.045 Kg m^2
l	0.180 m

Table 1: Parameters list

Substituting the value of parameters in 14, we have

$$G(s) = \frac{0.0009226s}{0.0003841s^3 + 0.000004s^2 - 0.0122s - 0.000091} \tag{15}$$

Note: The order of the sytem is 3.

Pole zero map for open loop system G(s) is plotted in *MATLAB* and it is found that the open loop system is unstable as some of the poles lie on right hand side of the s-plane as shown in *Figure 4*.

MATLAB code for pole-zero plot:

```
%MatLab code to get pole-zero plot for open loop system

m_c = 0.380; % mass of carriage
m_p = 0.970; % mass of pendulum
I_p = 0.045; % moment of inertia of pendulum
l = 0.180; % length of pendulum
f = 0.1; % frictional coefficient
g = 9.81; % acceleration due to gravity

q = (m_c+m_p)*(I_p+m_p*l^2)-(m_p*l)^2;

%define open loop transfer function for theta
s = tf('s');
G = (m_p*l*s/q)/(s^3 + (f*(I_p + m_p*l^2))*s^2/q - ((m_c + m_p)*m_p
    *g*l)*s/q - f*m_p*g*l/q);

%plot pole-zero plot for G
h = pzplot(G);
```

Listing 1: MATLAB code for plotting pole-zero map.

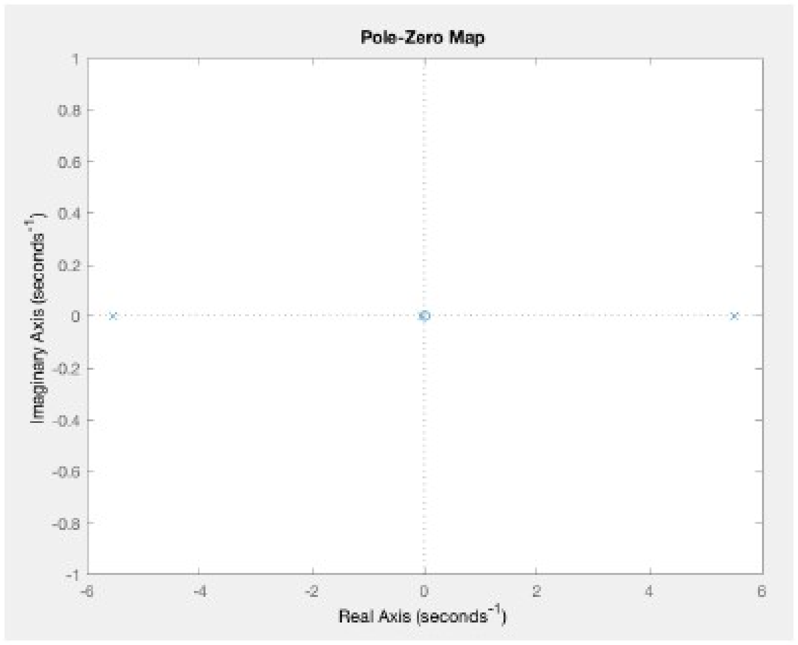


Figure 5: Pole-Zero plot for open loop system.

3 PID control

The aim of the project is to control the angle deviation ψ for making the Balanduino stable by using an active PID control method. The block diagram of the required closed loop system is shown in Figure 4.

MATLAB Code:

```
% MatLab code for simulation of inverted pendulum

m_c = 0.380; % mass of carriage
m_p = 0.970; % mass of pendulum
I_p = 0.045; % moment of inertia of pendulum
l = 0.180; % length of pendulum
f = 0.1; % frictional coefficient
g = 9.8; % acceleration due to gravity

q = (m_c+m_p)*(I_p+m_p*l^2)-(m_p*l)^2;

%define open loop transfer function for theta
s = tf('s');
G = (m_p*l*s/q)/(s^3 + (f*(I_p + m_p*l^2))*s^2/q - ((m_c + m_p)*m_p
    *g*l)*s/q - f*m_p*g*l/q);

%defining controller
Kp = 20;
Ki = 2;
Kd = 3;
```

```
C = pid(Kp,Ki,Kd);
T = feedback(G,C);

% Plotting the response
t=0:0.01:10;
impulse(T,t)
%Replace T by T*(180/3.14)  to get amplitude in  degrees.
axis([0, 5, -10, 10]);
title('Kp = 20, Ki = 2, Kd = 3');
-------------------------------------------------------------
```

Listing 2: MATLAB code for Balanduino

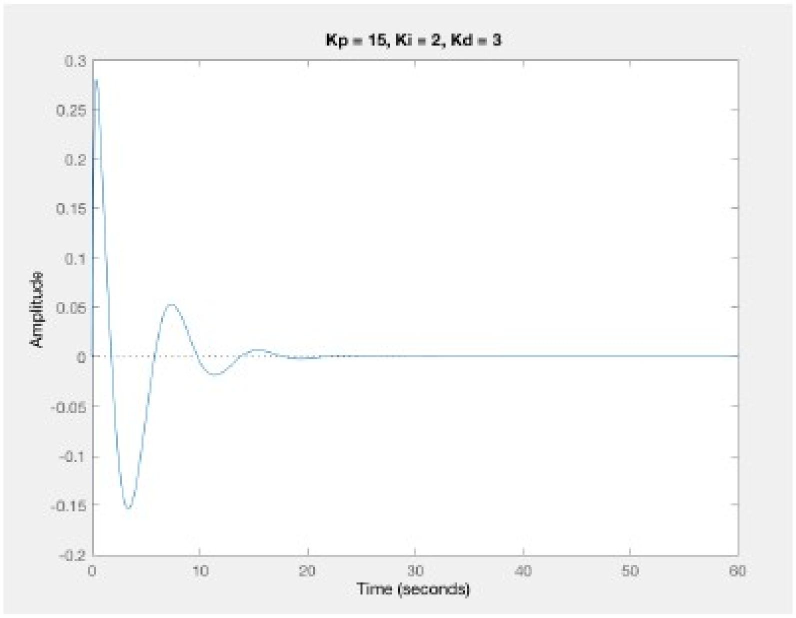


Figure 6: Feedback Control System is stable with zero steady state error and settling time of 18 sec

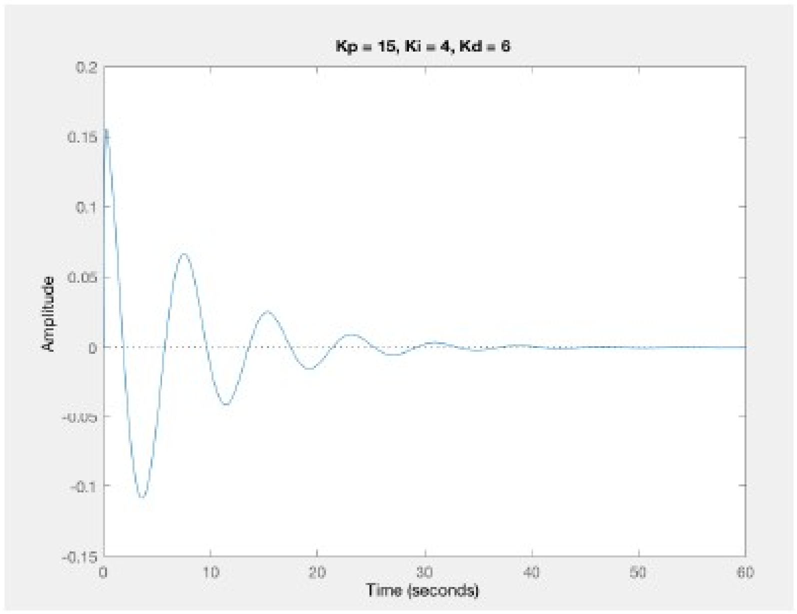


Figure 7: Feedback Control System is stable with zero steady state error and settling time of 40 sec.

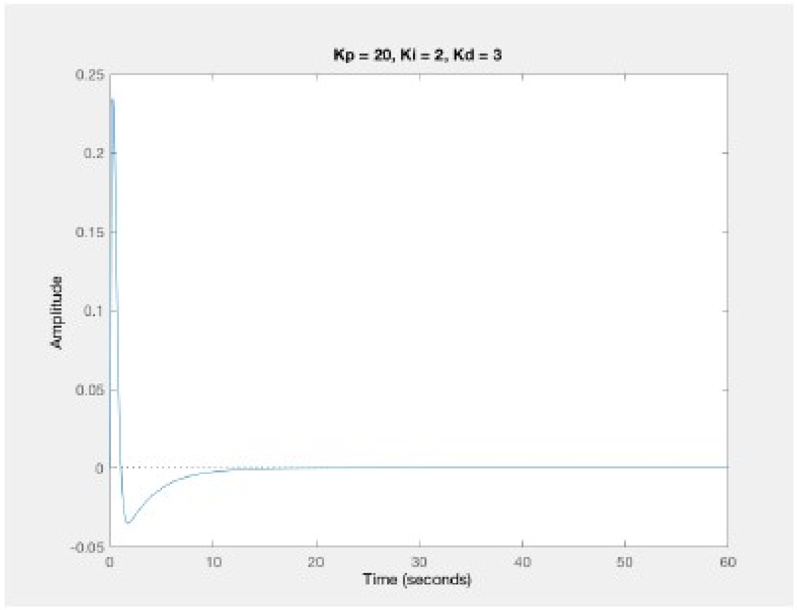


Figure 8: Feedback Control System is stable with zero steady state error and settling time of 9 sec.

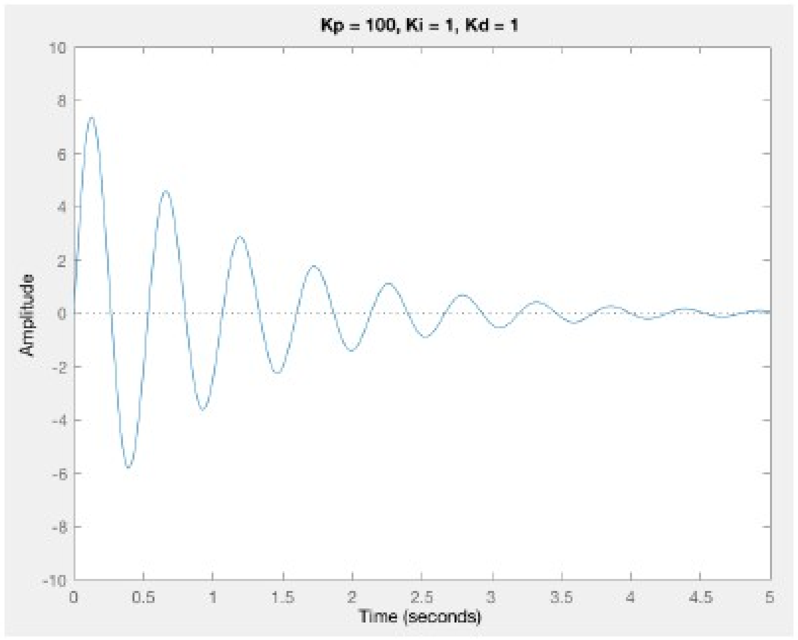


Figure 9: Feedback Control System is stable with zero steady state error and settling time of 5 sec.

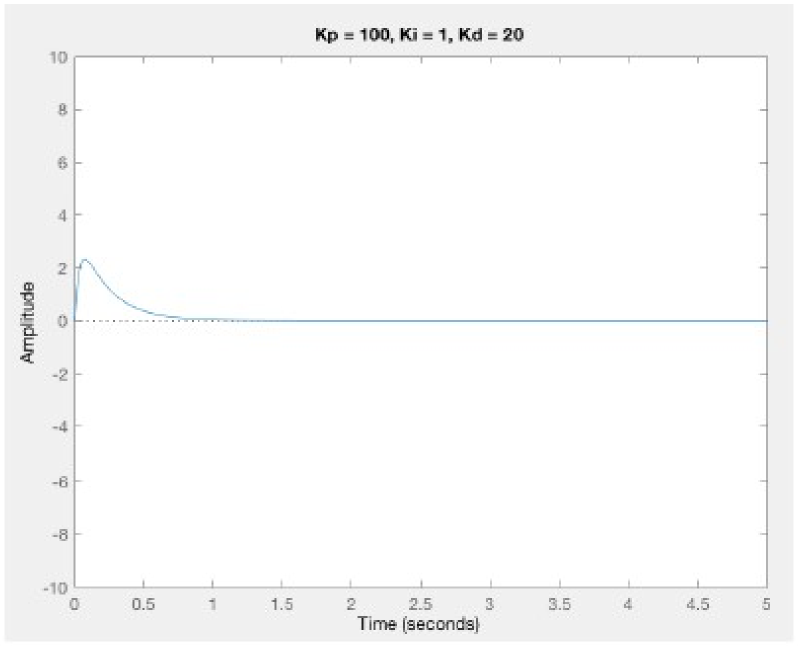


Figure 10: Feedback Control System is stable with zero steady state error and settling time of 0.7 sec.

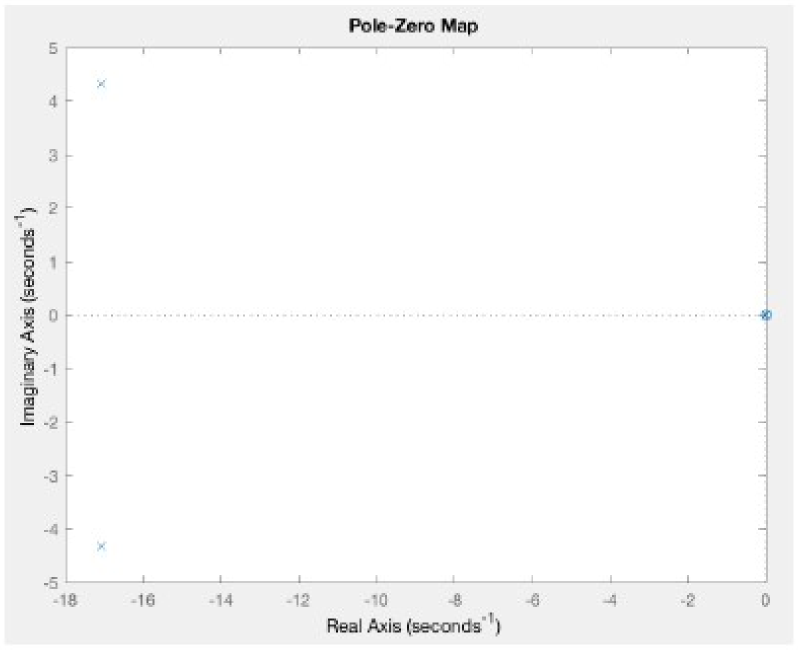


Figure 11: Pole-Zero plot for Figure 8

4 Tuning the Balanduino

The PID values are adjusted using the Balanduino Android application [2]. Balanduino has an inbuilt Bluetooth module which can be paired with a smartphone via the Balanduino app. Once connected with the smartphone, the PID values can be updated directly and the robot can be easily controlled with a joystick or the accelerometer of the smartphone. This Bluetooth connectivity allows us to get the response curve for the Balanduino in real time for the given PID values.

5 Results

The controller designed for Balanduino offering settling time of 9 sec and zero steady state error is as follows.

$$C(s) = K_p + \frac{K_i}{s} + K_d s \tag{16}$$

$$= 20 + \frac{2}{s} + 3s \tag{17}$$

The overall transfer function of Balanduino with feedback controller is shown below.

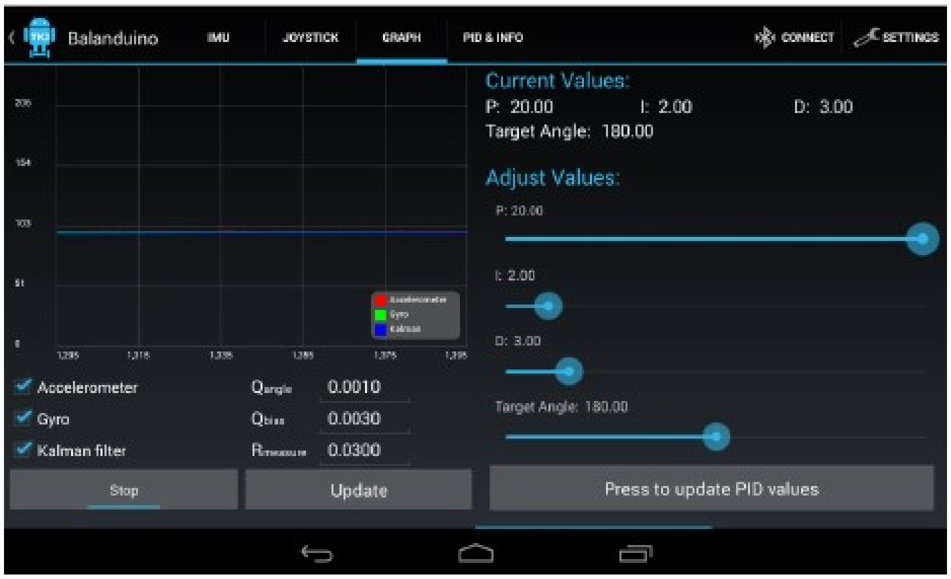
$$T(s) = G(s) * C(s) \tag{18}$$

$$= \frac{0.0009226s^2}{0.0003841s^4 + 0.00554s^3 + 0.001621s^2 + 0.0036s} \tag{19}$$

Note: The order of the system has changed from 3 to 4 and poles of T(s) lies in left hand plane as shown in Figure 11.

With increase in K_p , the rise and settling time is reduced significantly and the steady state error is zero for any value of $K_i \geq 1$. The overshoots are minimized and transient response is improved while keeping $K_d \leq 20$.

The actual responses for the Balanduino are shown below.



Current Values:
P: 20.00 I: 2.00 D: 3.00
Target Angle: 180.00

Adjust Values:
P: 20.00
I: 2.00
D: 3.00
Target Angle: 180.00

✓ Accelerometer Q_{angle} 0.0010
✓ Gyro Q_{bias} 0.0030
✓ Kalman filter $R_{measure}$ 0.0300

Figure 12: Screenshot of the Balanduino app.

The response obtained from the Balanduino application and those simulated in *MATLAB* were quite similar. Also the PID gains for which the Balanduino is found stable were found to be **20, 2, and 3** respectively. Also, the Balanduino gets unbalanced and falls for large excursions about 180°.

6 Conclusion

Inverted pendulum systems are unstable and need active feedback control methods to achieve stability. A simple PID-controller can be implemented, but only to control the angle deviation, ψ. The control the position too, cascade PID controller is to be used. The model is quite reliable as the implemented PID controller was able to stablise the Balanduino. Also, for handling impulses larger than 10°, motors with high torque and high angular speed are recommended with better system modelling and minimum assumptions involved.

References

[1] http://www.balanduino.com/images/front_facing.png

[2] https://github.com/TKJElectronics/BalanduinoAndroidApp
Link to the application on Android Play Store

[3] https://in.mathworks.com/help/control/ref/pzmap.html

[4] http://in.mathworks.com/help/control/examples/using-feedback-to-close-feedback-loops.html