Inverted Pendulum Position Controller

Submission Document:

Team B:

Bhavina Chechani

Bhumika Chechani

Harekrishna Ray

Jay Mistry

Acknowledgement

In the present world of competition and race of existence, we joined this community with the willingness that it will be a bridge between the theoretical and the practical aspects. We would like to express our sincere gratitude to the Abhiyanta Community, to the core team members and our mentors for helping us out at every stage and properly guiding us at each and appropriate required stages of the task.

Index

Overview

The inverted pendulum is one of the basic problems in control systems because of its theoretical values, and its practical values. The objective of the project is to balance the inverted pendulum if any deflection takes place using a controller.

In the following project, we have carried out this problem in the standard control system stability analysis model:

1. Modelling of the inverted pendulum system
2. Determining the transfer functions from the equations of the motions and analysing the system with the help of impulse response and pole zero plot
3. PD Controller implementation
4. PID Controller implementation
5. Comparing PD and PID Controller and choosing the best for the hardware implementation
6. Hardware Implementation
7. Troubleshooting

In the first step, the equations of motion for the inverted pendulum and the cart are determined. After determining the equations of the model for the system, the transfer function for the system is determined. The non-linear model which is obtained is linearized by considering a few assumptions, then analyzing the system in SCILAB/MATLAB. After this, a comparison of PD and PID controller is done in stabilizing the system and its responses and analysis are carried out in SCILAB/MATLAB. Once the system is stabilized using a properly tuned controller, implementation of this on hardware to demonstrate the system is carried out.

1. Modelling of the inverted pendulum system

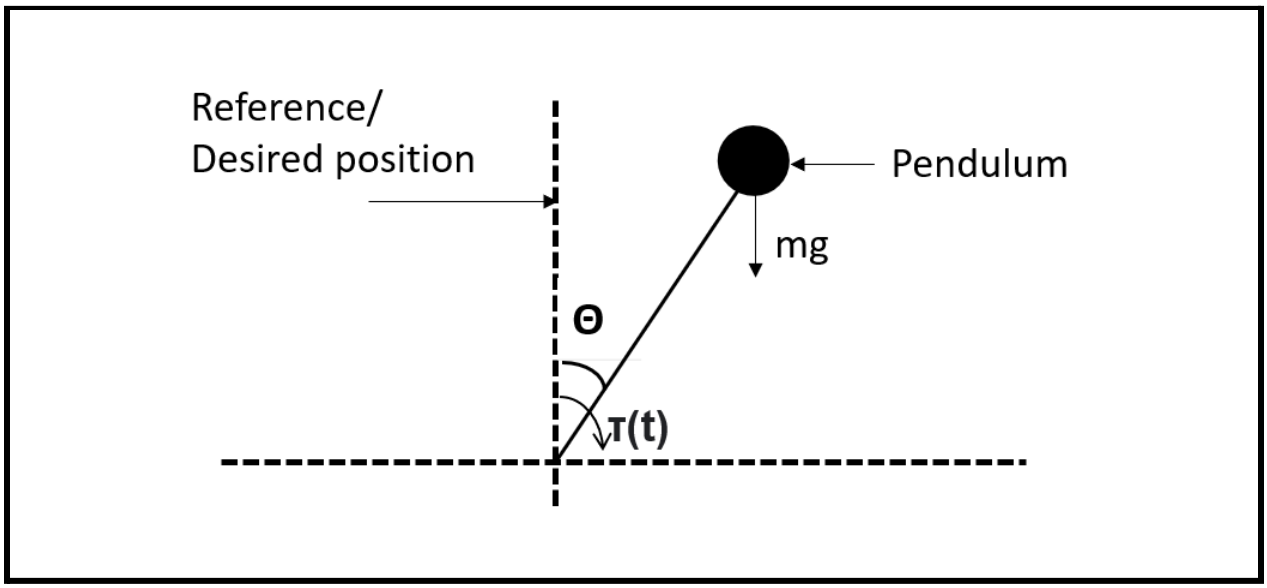


Fig: 1.1 Free Body Diagram

Fig 1.1 shows the cart and inverted pendulum free body diagram.

Parameters of the inverted pendulum and cart:

m = Mass of the pendulum
l = Length of the pendulum
τ = torque applied to the pendulum
Θ = Angle of deflection for pendulum from vertical

The equations of motions[1] that is determined are:

$$\tau(t) = ml^2\Theta''(t) - mgl\Theta(t)$$

For modelling this system in SCILAB/MATLAB the following considerations are done:
M = 1kg
m = 0.3kg
l = 0.5m
g = 9.91m/s^2

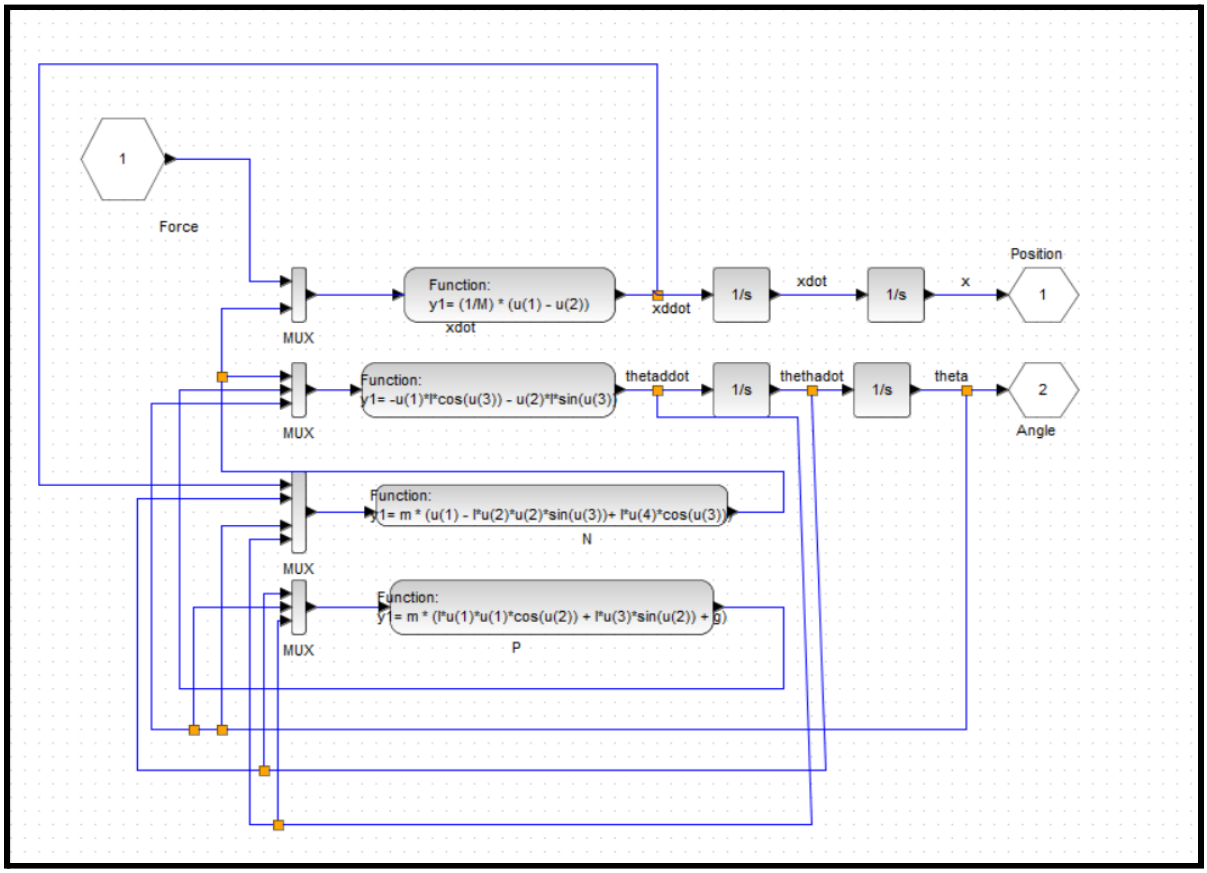


Fig: 1.2 XCOS modelling of the inverted pendulum system

The XCOS block diagram[4] for the inverted pendulum system is shown in figure 1.2.

2. Transfer Function of the system

From the equation of motions of the system, the transfer function[1] of the pendulum can be obtained:

$$\mathrm{TF}_{\mathrm{pend}} = \frac{\frac{mls}{q}}{s^3 + \frac{ml^2s^2 - (M+m)mgls - mgl}{q}}$$

After determining the transfer function of the system, its impulse response (fig 2.1) and pole-zero plot (fig 2.2) is obtained in SCILAB/MATLAB.
In fig 2.3 comparison of the impulse response[2] and the pole-zero plot for the inverted pendulum and cart is done.

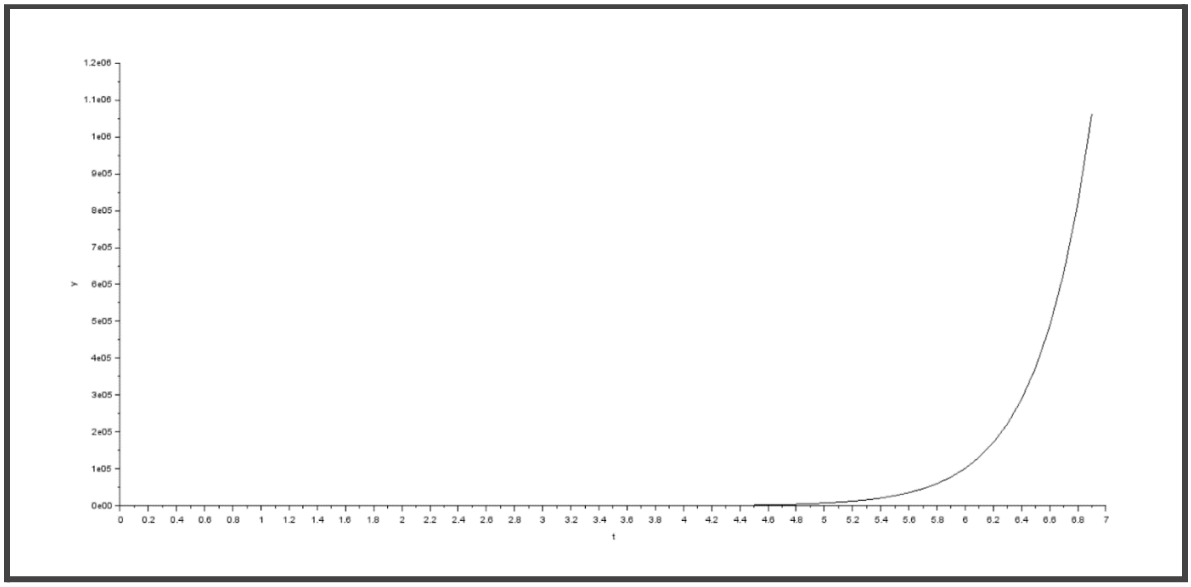


Fig 2.1 Impulse response of the system

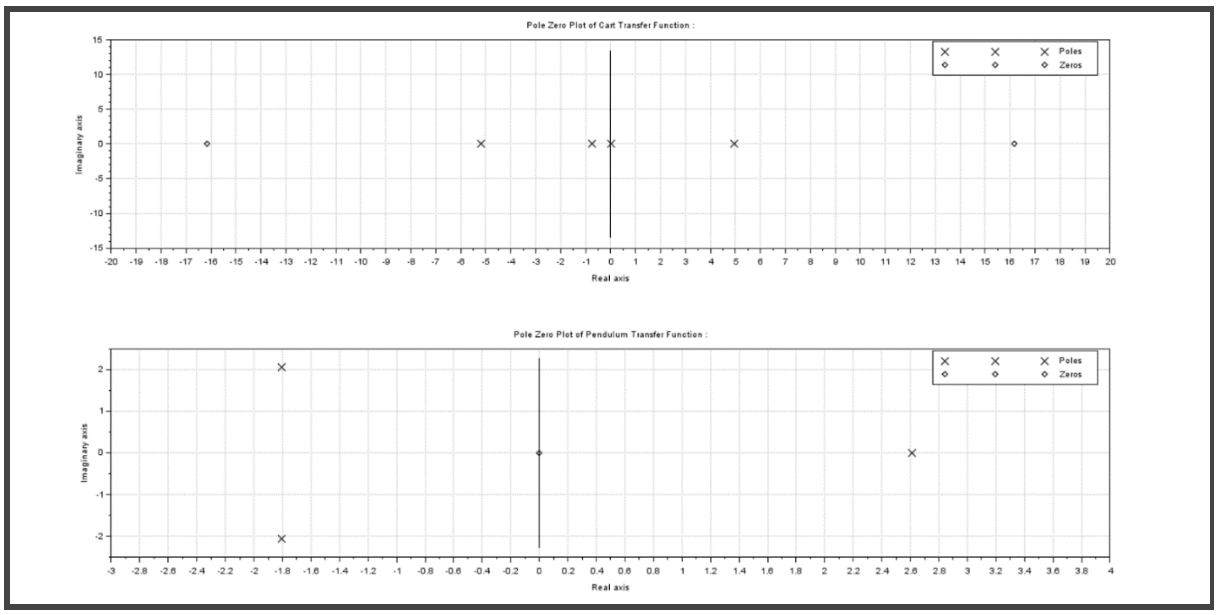


Fig: 2.2 Pole-zero map of the system

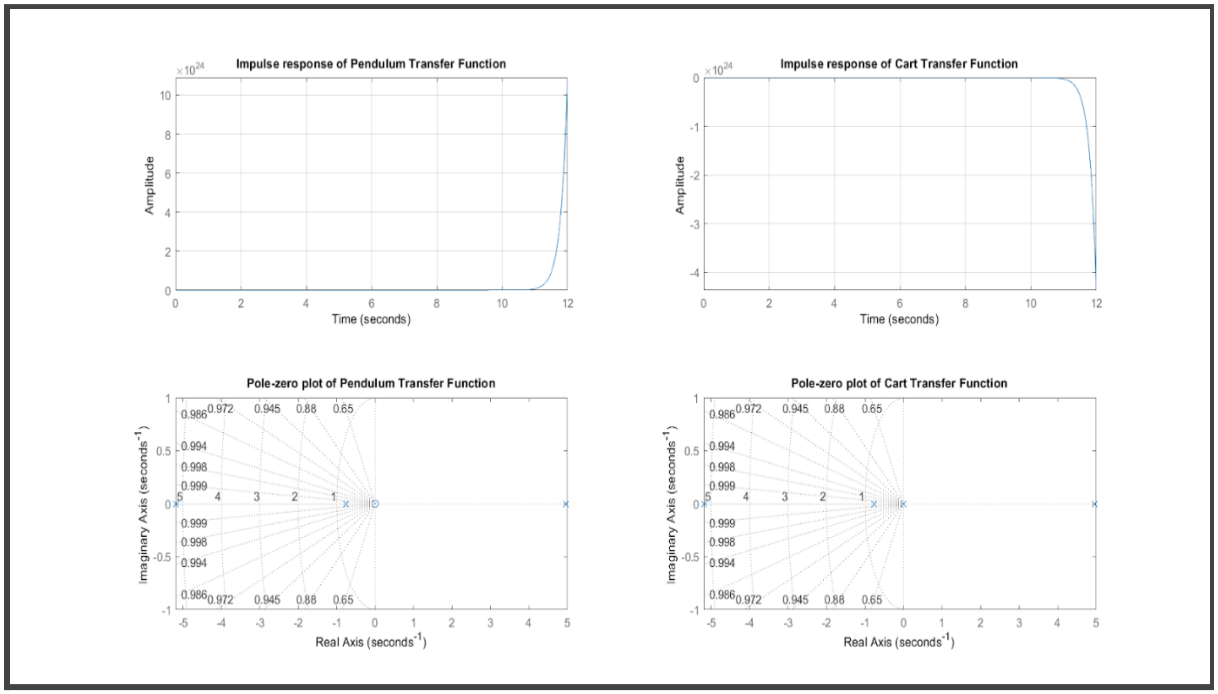


Fig: 2.3 Comparing the impulse response and pole-zero plot of pendulum and cart.

From the above plots fig 2.1 and fig 2.2,

The system impulse response fig 2.1 is entirely undesirable since it is leading to instability. Since the plot is of an open-loop transfer function only, the response suggests that there is a requirement of developing a closed-loop system to make this unstable system stable by making the impulse response valid i.e., not oscillating/unbounded. The poles of a system can also tell us about its time response. Since our system has two outputs and one input, it is described by two transfer functions. From above, it is seen that for both the transfer function, 1 pole lies on RHP, in the case of the cart's transfer function a zero also lies in the RHP, which leads to the instability of the system.

To make the system stable we need to introduce a closed-loop type mechanism by using appropriate controllers so that the poles on the RHP are shifted to the LHP on the plane.

3. PD Controller Implementation

The following fig 3.1 shows the XCOS block diagram of PD Controller[14] implementation.

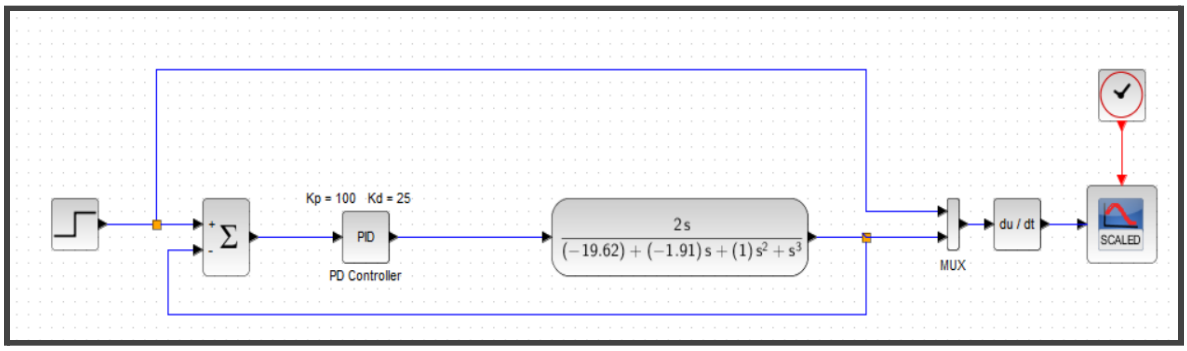


Fig: 3.1 Xcos block diagram for implementing the PD controller

The various values of K_p and K_d were tested for tuning the PD controller.

Table 3.1: Combinations of K_p and K_d

Cases	Comments/Remarks
Case 1: $K_p = 1$, $K_d = 1$	Unbounded output unstable response.
Case 2: $K_p = 10$, $K_d = 1$	Unbounded output unstable response.
Case 3: $K_p = 100$, $K_d = 1$	Oscillatory response for some time, then gets stable.
Case 4: $K_p = 100$, $K_d = 10$	Settling decreases and response contains only maximum overshot and no oscillations.
Case 5: $K_p = 100$, $K_d = 20$	Decreasing maximum overshoot and rise time and settling time.
Case 6: $K_p = 100$, $K_d = 25$	Optimum response, gets stable output near 0 with overshoot **($M_P = 0.0096$)** and rise time **($t_r = 0.03s$)**, but settling time**($t_s = InF$)***.

***Note:** In PD Controller, since t_s = infinity, we need to add integral part to make the system stable with less settling time. This is overcome by using the PID controller covered in section 4.

From above, the best tuned system was found in the case 6. The impulse response(fig 3.4) and the pole-zero plot(fig 3.2&3.3) for the same is:

Pole – Zero Plot for best K_p and K_d(*Case 6*):

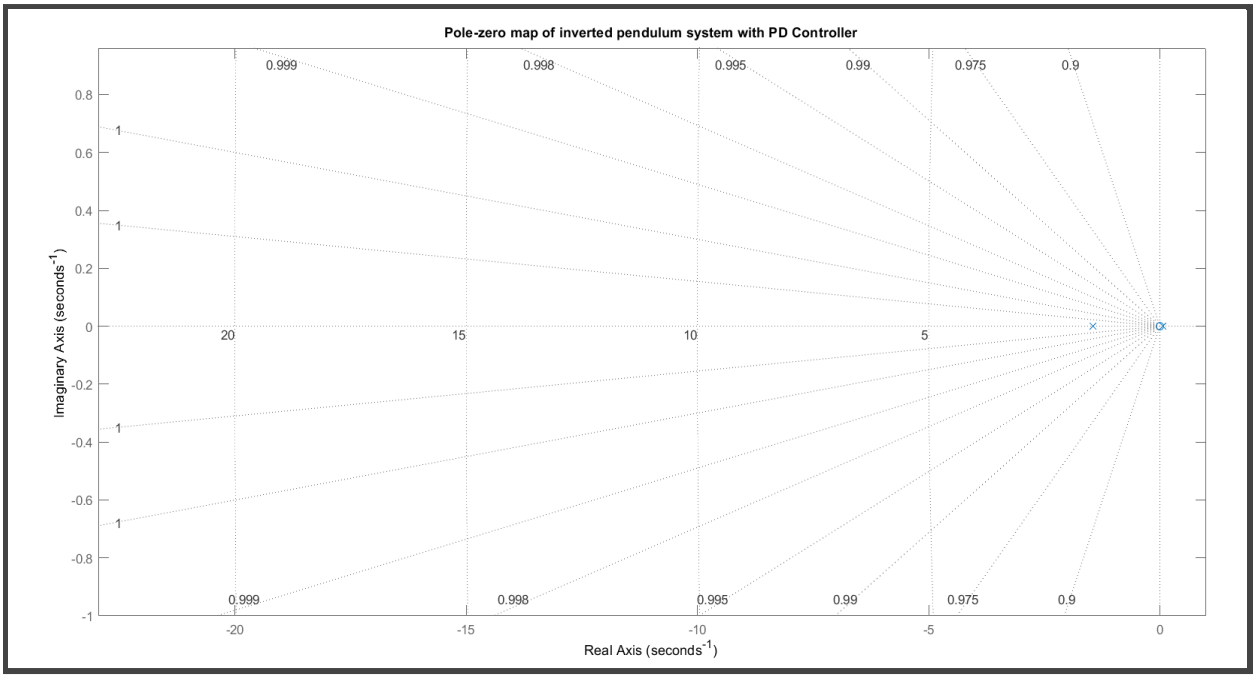


Fig: 3.2 Pole-zero map of the system

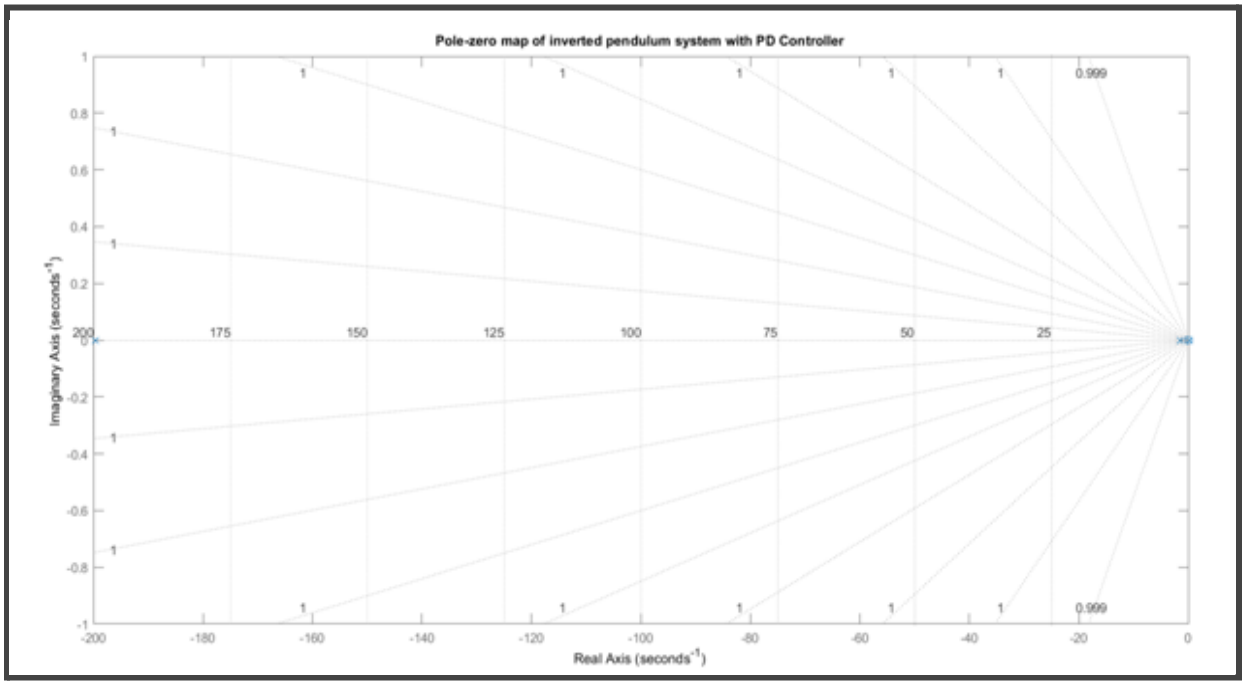


Fig: 3.3 Pole-zero map of the system

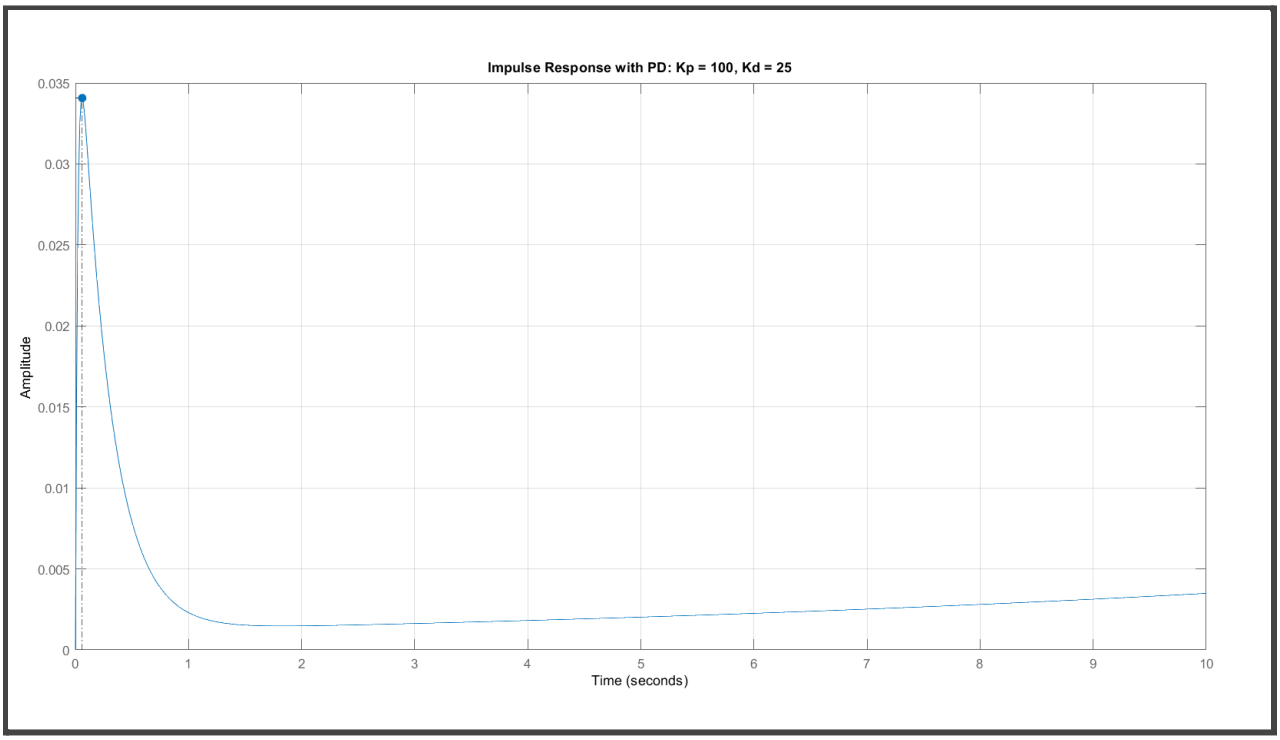


Fig: 3.4 Impulse response for case 6

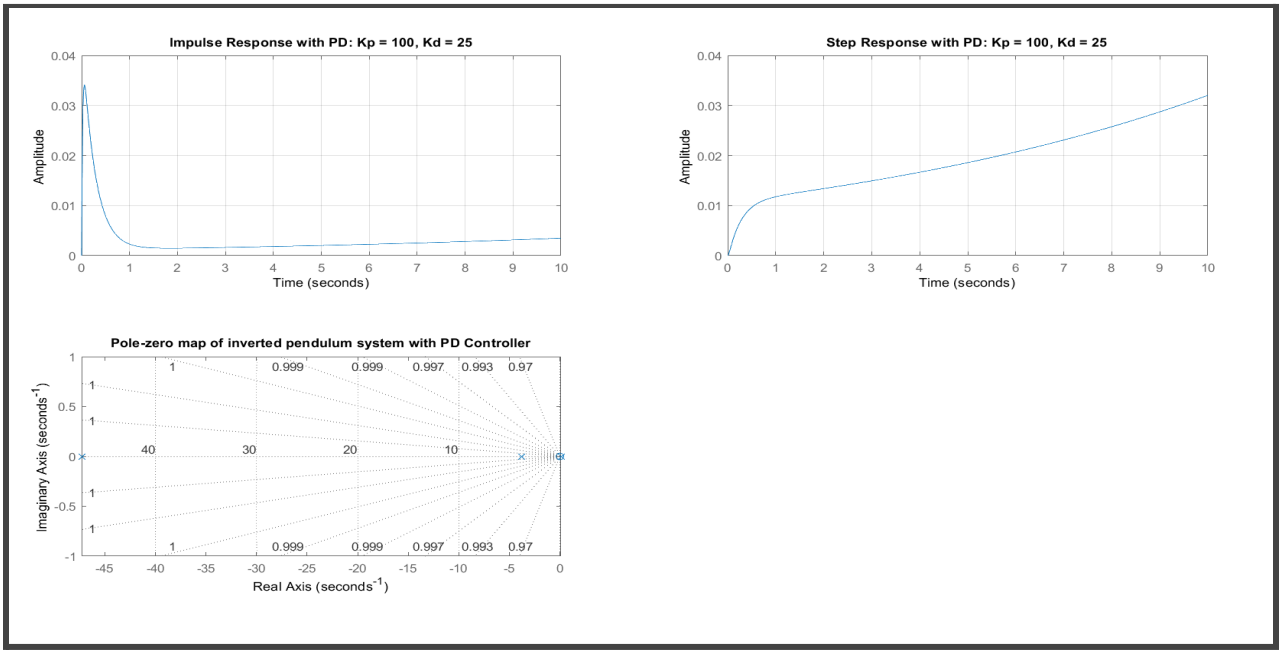


Fig: 3.5 Pole-zero map, impulse and step response case 6

Hence, by using the PD Controller to stabilize the inverted pendulum system, the following outcomes have been concluded:

- **K_p:** On increasing the **proportional gain K_p**, the unbounded output gets bounded at a certain value of K_p, this parameter increases the **peak overshoot M_p.**
- **K_d:** On increasing the **derivative gain K_d,** the transient response gets improves and the oscillations are overcome which were caused due to the K_p gain. This brings the response near to 0 and stabilizes it.
- Hence, by using a PD controller, tried to stabilize the system by observing the impulse response of the system, but one pole of the system lies in the RHP. To bring that pole in LHP on further increasing the K_p *and* K_i values will distort the transient and steady-state behavioR.
- So, this can be overcome by using a PID controller.

4. PID Controller Implementation

The following fig 4.1 shows the XCOS block diagram of PID Controller[1] implementation.

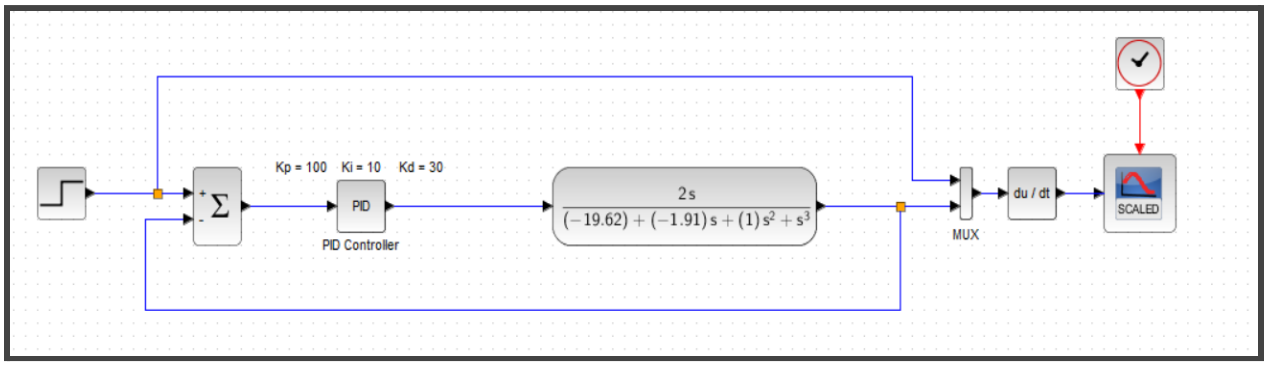


Fig: 4.1 Xcos block diagram for implementing the PID controller

The various values of K_p,K_i and K_d were tested for tuning the PID controller.

Table 4.1: Combinations of K_p, K_i and K_d

Cases	Comments/Remarks
Case 1:K_p = 1, K_i= 1, K_d = 1	Unbounded output unstable response.
Case 2:K_p = 10, K_i= 1, K_d = 1	Unbounded output unstable response.
Case 3:K_p = 100, K_i= 1,K_d = 1	Oscillatory response for some time, then gets stable but does not stabilize at 0.
Case 4:K_p = 100, K_i= 10,K_d = 1	Oscillatory response for some time, then gets stabilized at 0, high settling time.
Case 5:K_p = 100, K_i= 1, K_d = 10	No oscillations, settles at a constant quantity but not at 0.
Case 6:K_p = 100, K_i = 10, K_d = 30	Optimum response, gets stable at 0 with overshoot **(M_P = 0.0114)** and rise time**(t_r = 0s)**, settling time **(t_s = 1.36s)** and peak time**(t_P = 2.4511s)**.

From above, the best tuned system was found in the case 6. The impulse response (fig 4.4) and the pole-zero plot (fig 4.2&4.3) for the same is:

Pole – Zero Plot for best K_p, K_i and K_d(*Case 6*):

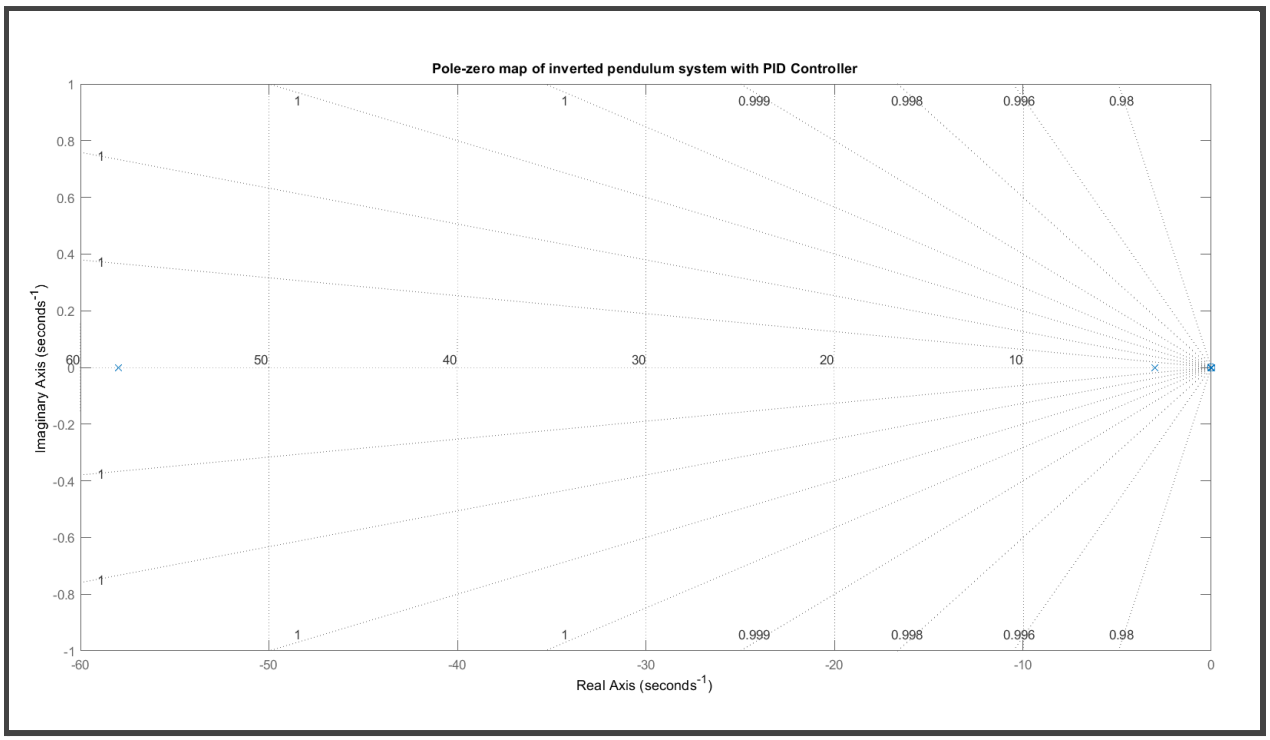


Fig: 4.2 Pole-zero map of the system

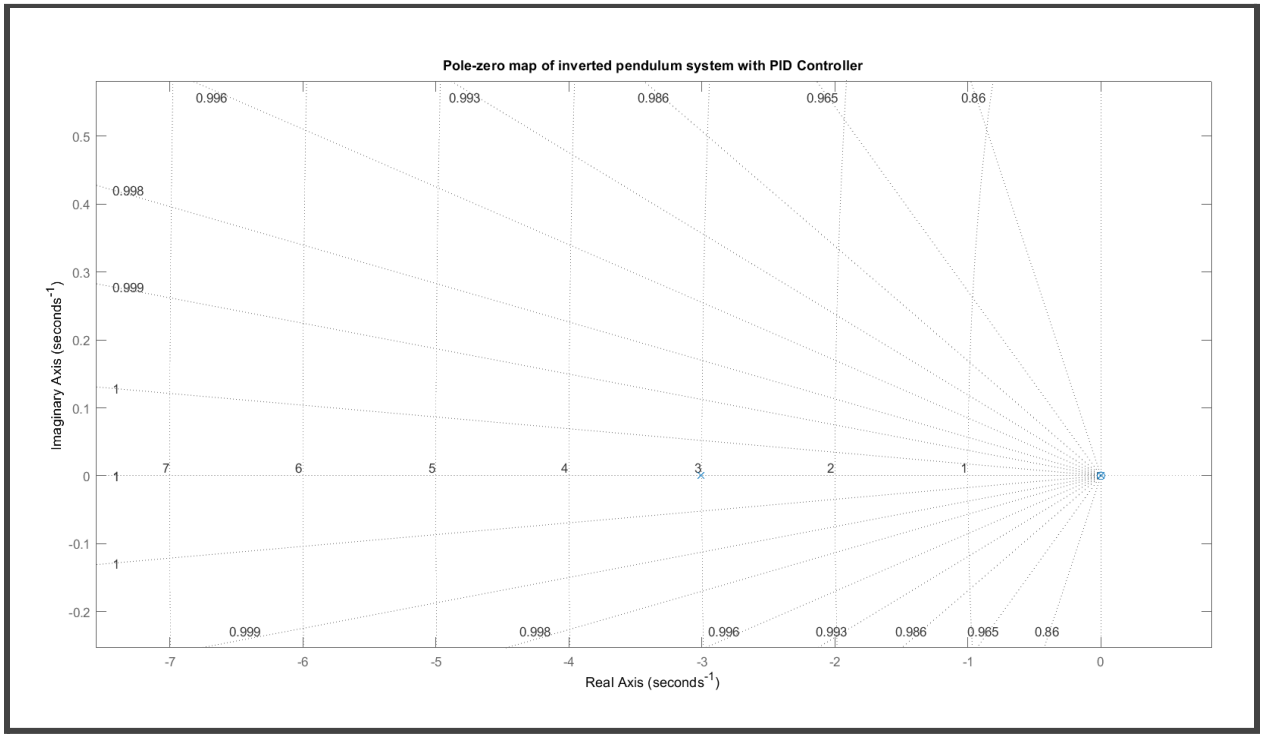


Fig: 4.3 Pole-zero map of the system

Impulse, Step response and pole-zero plot:

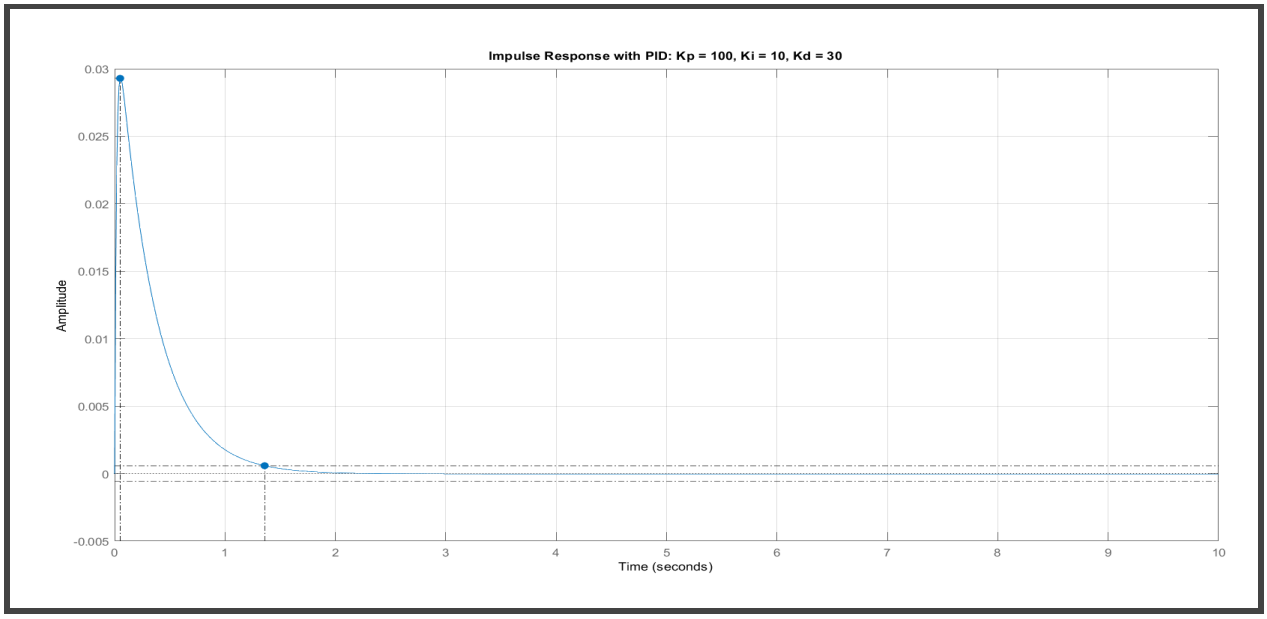


Fig: 4.4 Impulse response for case 6

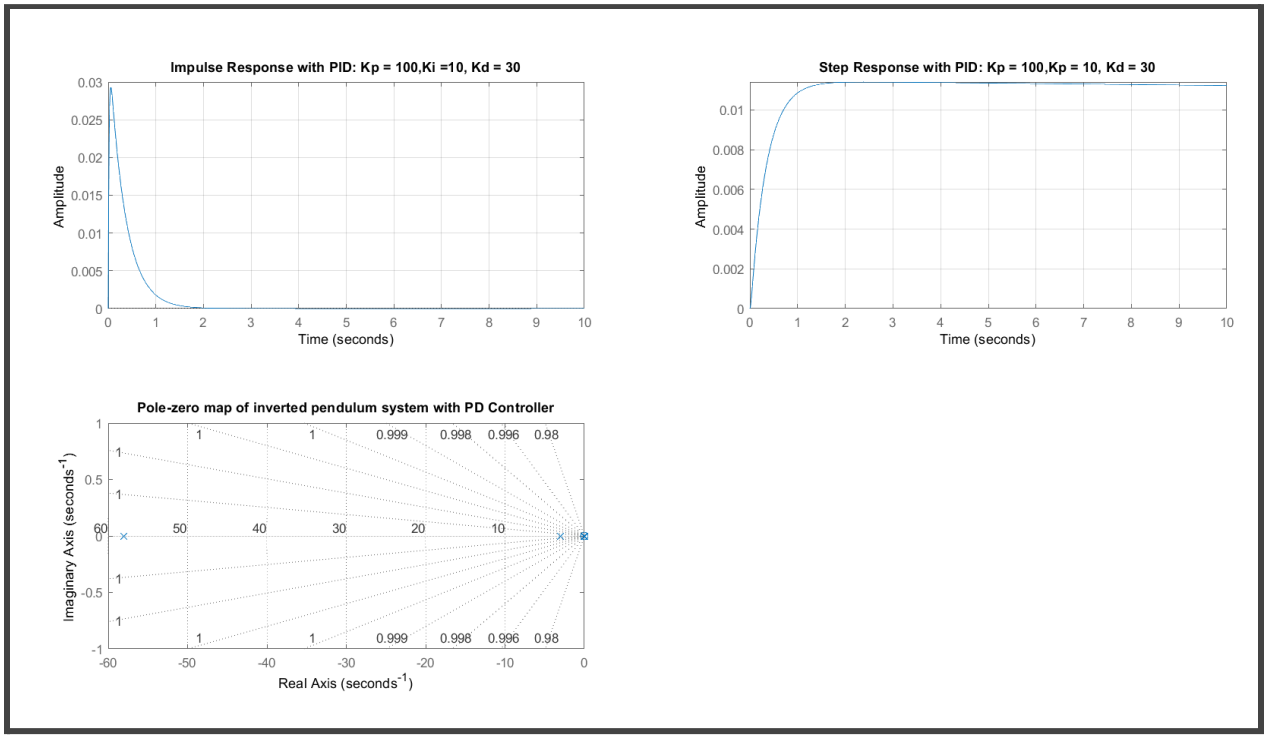


Fig: 4.5 Pole-zero map, impulse and step response case 6

Hence, by using the PID Controller, trying to stabilize the inverted pendulum system and overcome the problem of a pole on RHP in case of using a PD controller, the following outcomes have been concluded.

- **K_p:** On increasing the **proportional gain K_p**, the unbounded output gets bounded at a certain value of K_p, this parameter increases the **peak overshoot M_p.**
- **K_i:** On increasing the **integral gain K_i**, the steady-state response improves and we get a stable output. This parameter brought the pole from RHP to LHP in the s-plane and made the system stable. On further increasing K_i(>10 in our case), the response shifts below 0 and it makes the system less stable, hence proper tuning is a must.
- **K_d:** On increasing the **derivative gain K_d,** the **transient response** gets improves and the oscillations are overcome which were caused due to the K_p gain. This brings the response near to 0 and stabilizes it.
- Hence, by using a PID controller, the inverted pendulum system is stabilized and the PD controller problem is overcome, by observing the impulse response of the system.

5. Comparing PD Controller and PID Controller

Table 5.1: Combination of K_p, K_d for PD and K_p, K_d and K_i for PID

Cases	K_p	K_d	K_p	Ki	K_d
Case 1	1	1	1	1	1
Case 2	10	1	10	1	1
Case 3	100	1	100	1	1
Case 4	100	10	100	10	1
Case 5	100	20	100	1	10
Case 6	100	25	100	10	30

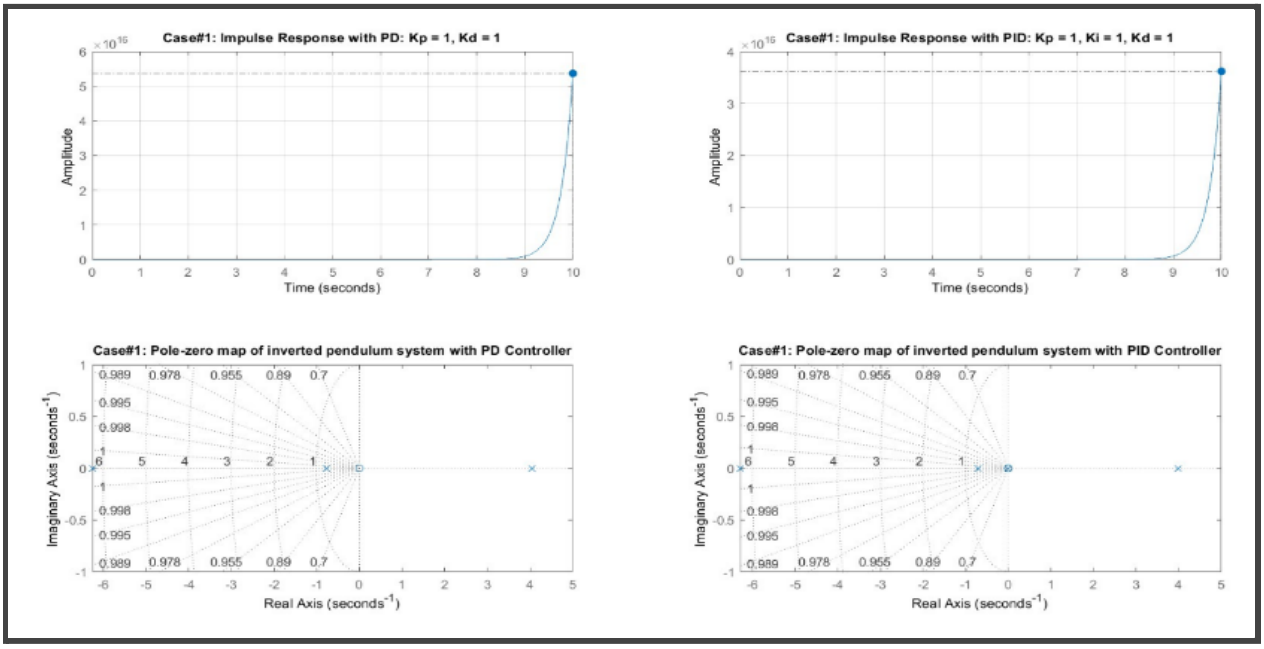


Fig: 5.1 Impulse and pole-zero plot for case1

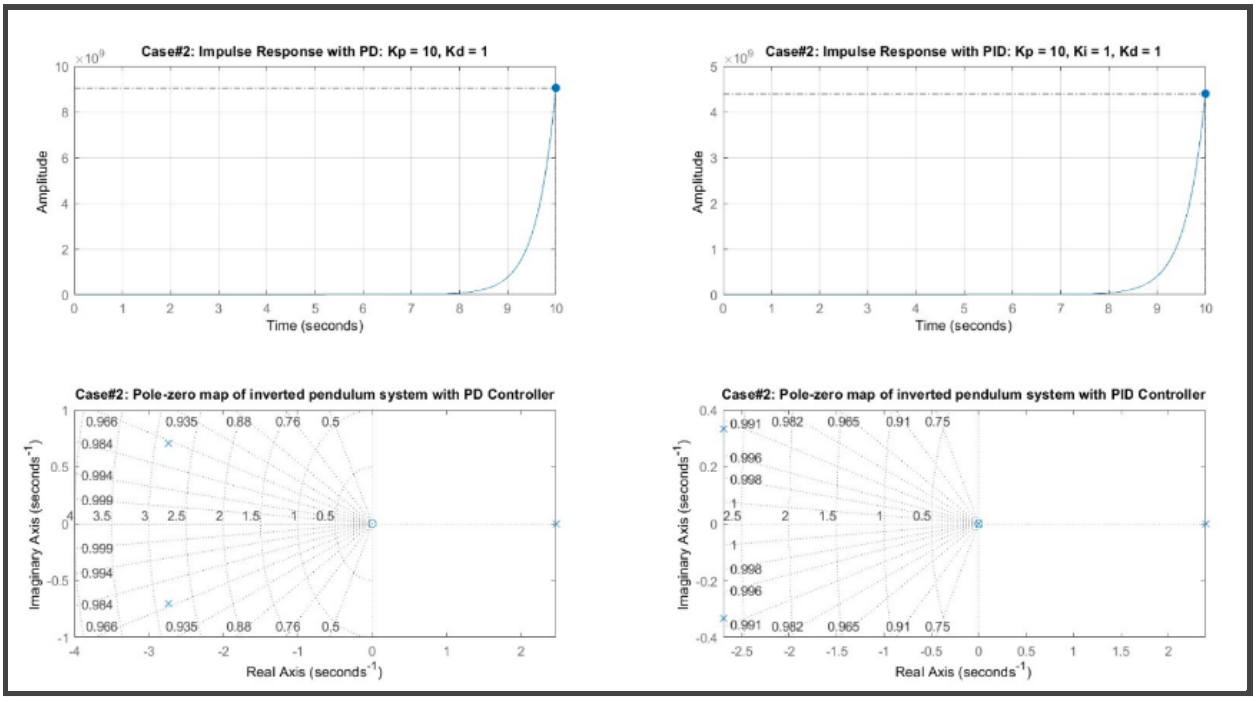


Fig: 5.2Impulse and pole-zero plot for case2

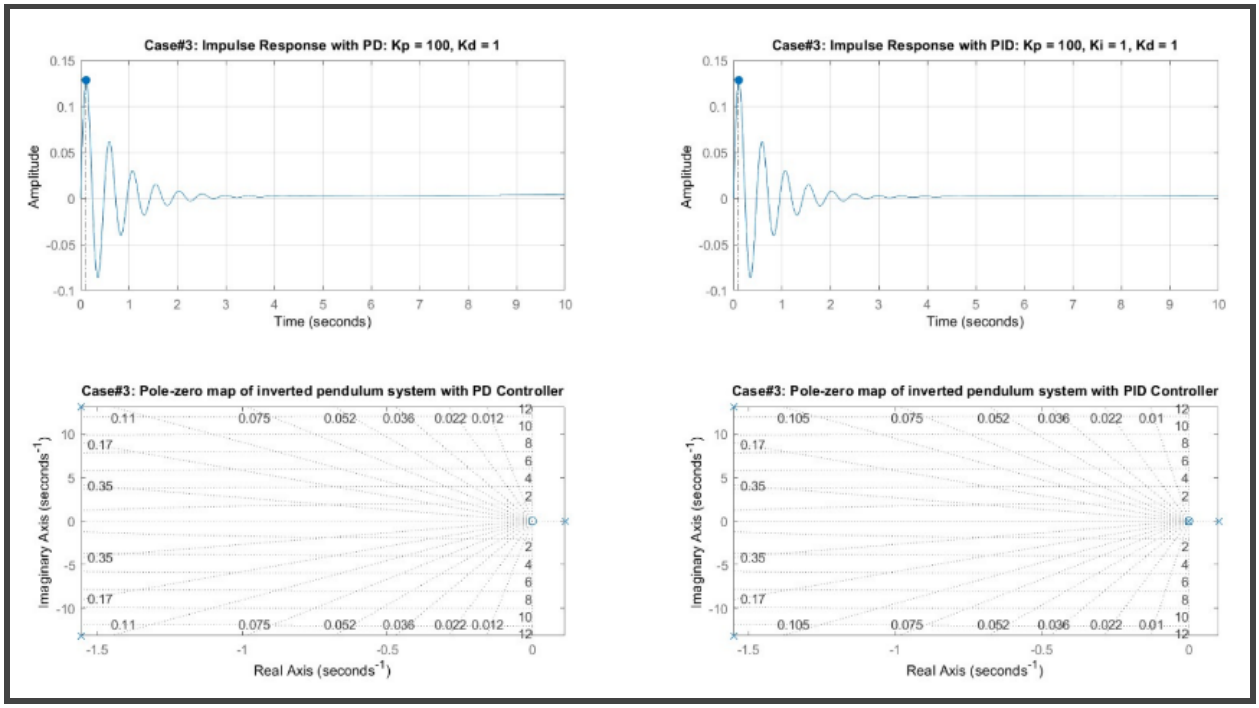


Fig: 5.3 Impulse and pole-zero plot for case3

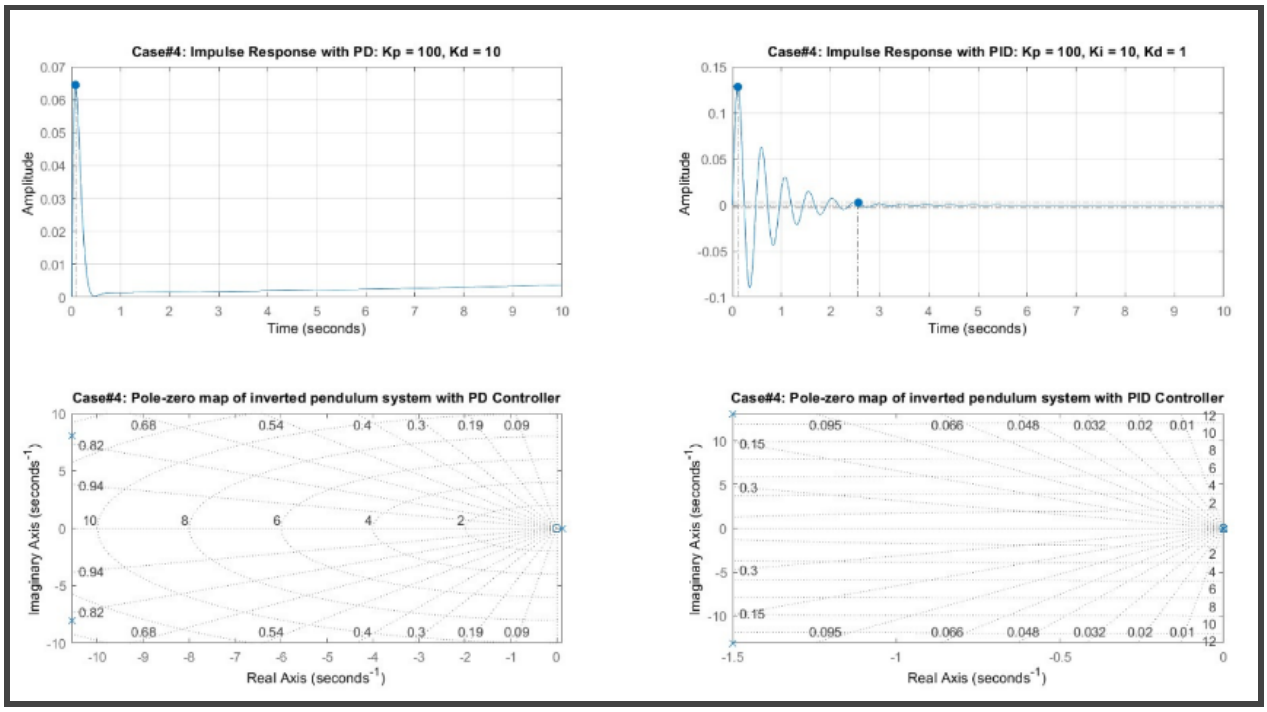


Fig: 5.4 Impulse and pole-zero plot for case4

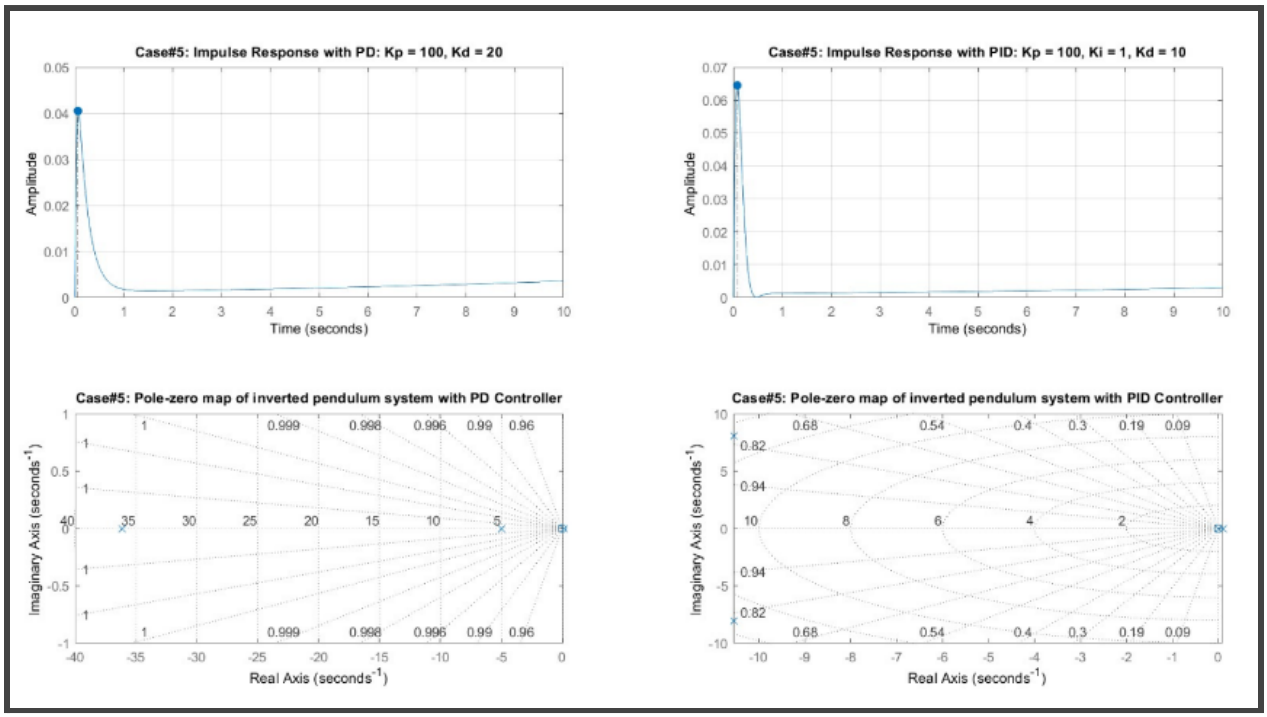


Fig: 5.5 Impulse and pole-zero plot for case5

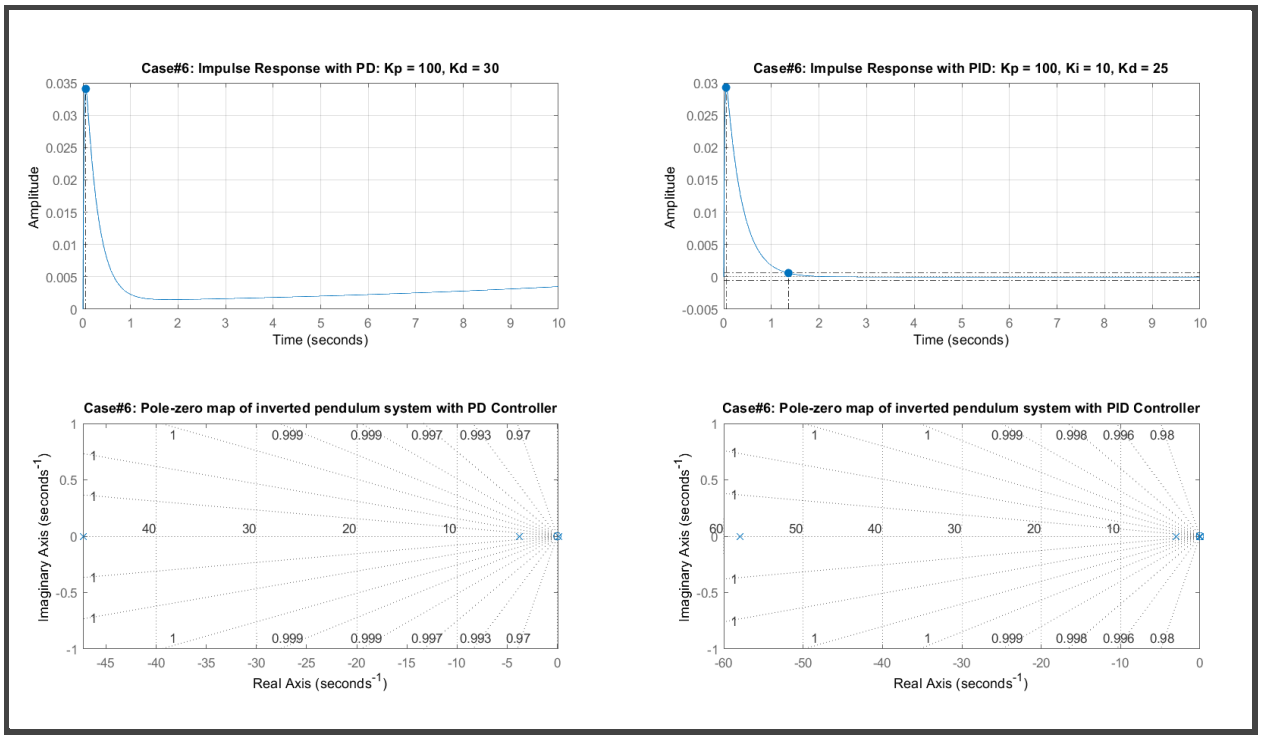


Fig: 5.6 Impulse and pole-zero plot for case6

- From the above results, the **PID controller is the best option** and is the best to stabilize the system as studied above.
- The PD controller also makes the impulse response stable, but one pole is left in the RHP of the s-plane and hence this makes the system unstable.
- On using the PID controller, the integral part of the PID controller shifts that one pole in the RHP of the s-plane to the LHP.
- ***Case #6*** i.e., K_p= 100, K_i = 10, K_d= 25 is the best and the optimum case in tuning the PID controller for the system, since beyond $K_i > 10$ will make the response move below the zero line, and also increasing the $K_p$more will increase oscillations in the system. Increasing $K_d$more will increase the overshot, which is not desired.

6. Hardware Implementation

Component:

- Arduino
- Rotary Potentiometer
- Dc motor 200 rpm
- L298 motor driver IC[12]
- Ice-cream stick (for pendulum part)
- Breadboard
- Connecting wires

Connection and Working[13]:

- We have connected the motor shaft to the shaft of the rotary potentiometer and joined our pendulum to that.
- For Hardware implementation, we have used Arduino for controlling the motor speed, and for feedback, we have used Potentiometer.
- The motor input is coming from a motor driver which is controlled by Arduino.
- As the position of the pendulum changes the resistance of the potentiometer also changes which leads to a change in voltage and this changed voltage is given as feedback to Arduino.
- The error is generated by comparing the reference voltage and the feedback voltage by the Arduino.
- According to the error, the output that is the speed and the polarity of the motor is decided and is and is controlled by the PID controller.
- Following is the link for the arduino code: https://github.com/Abhiyanta-Community/Trainee-B-JAN2021/blob/Task-1.3/Bhavina/Code_arduino.ino

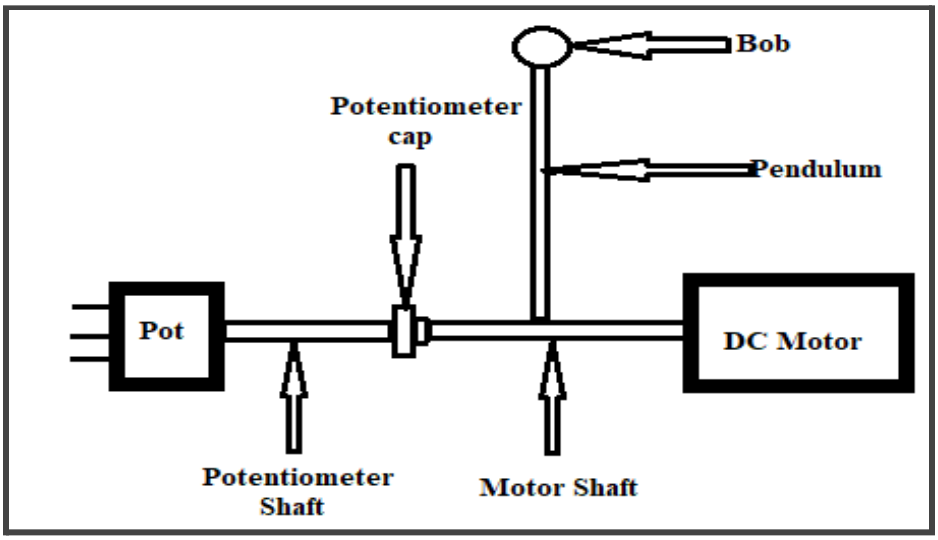


Fig: 6.1 Side view of circuit

7. Tuning

Y-axis : PID controller response **X-axis : Time**
Scale : **Y-axis : 1 unit = 3**

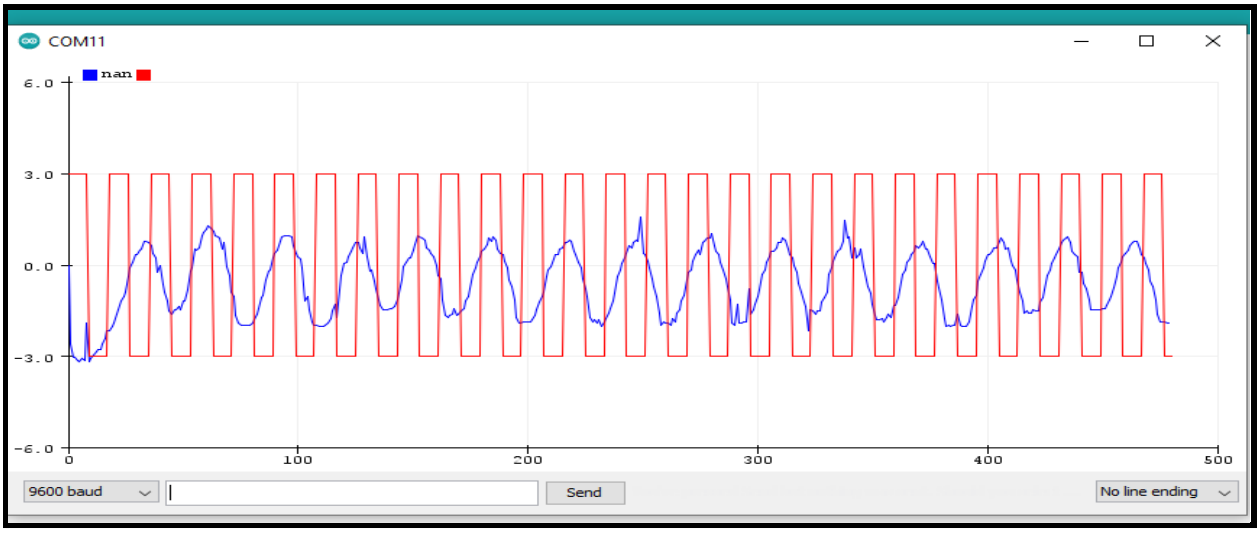


Fig: 7.1 Graph for kp=10,ki=0,kd=0

By observing fig.7.1 oscillations are obtained, overshoot is large and settling time is infinite, Now increasing kd to reduce overshoot and settling time.

Y-axis : PID controller response **X-axis : Time**
Scale : **Y-axis : 1 unit = 2.5**

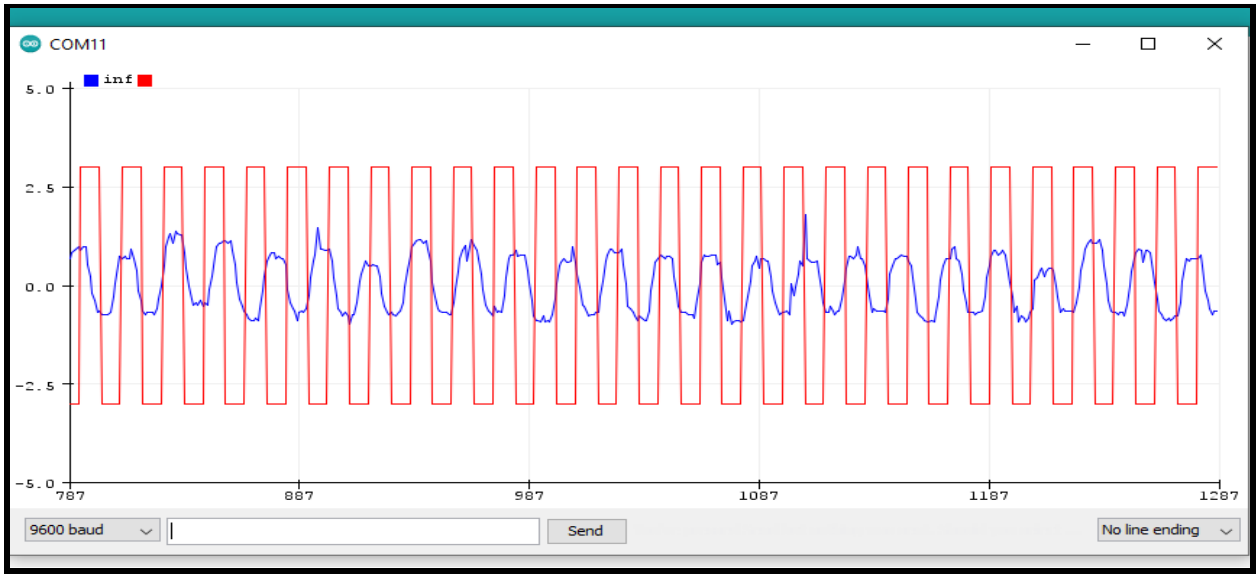


Fig: 7.2 Graph for kp=10,ki=0,kd=0.1

By observing fig.7.2 as kd is increased to 0.1 which helps in decreasing overshoot but settling time is still infinite. Now still increasing kd.

Y-axis : PID controller response **X-axis : Time**
Scale : **Y-axis : 1 unit = 2.5**

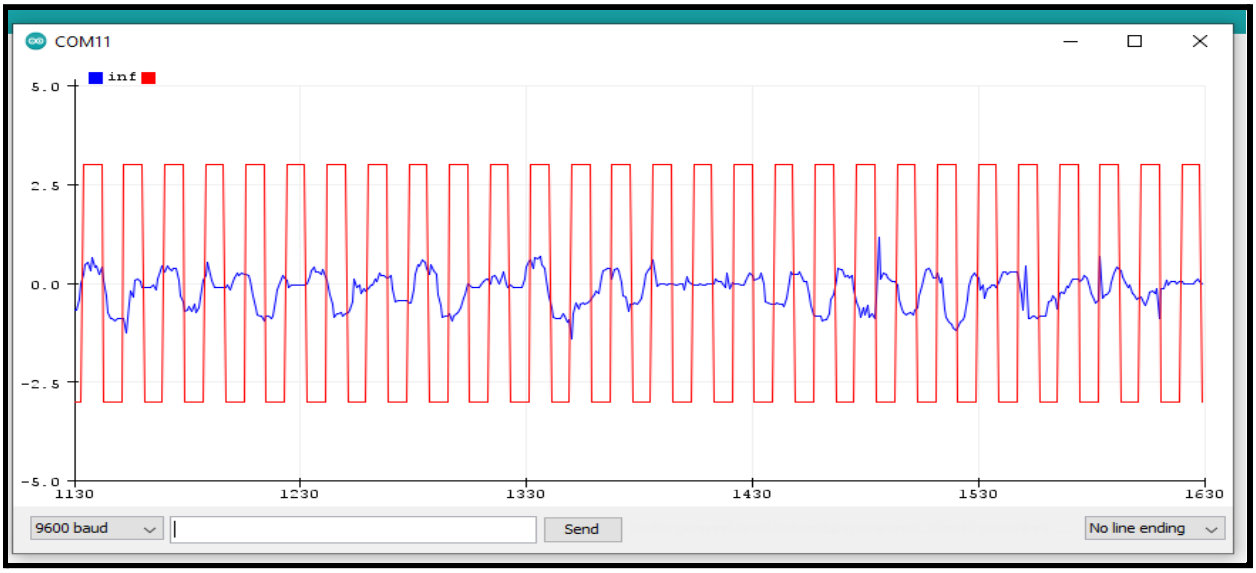


Fig: 7.3 Graph for kp=10,ki=0,kd=10

By observing fig.7.3 concluded that there is not much difference in the response for kd=0.1 and kd=10 so now increasing kp value to 50.

Y-axis : PID controller response **X-axis : Time**
Scale : **Y-axis : 1 unit = 15**

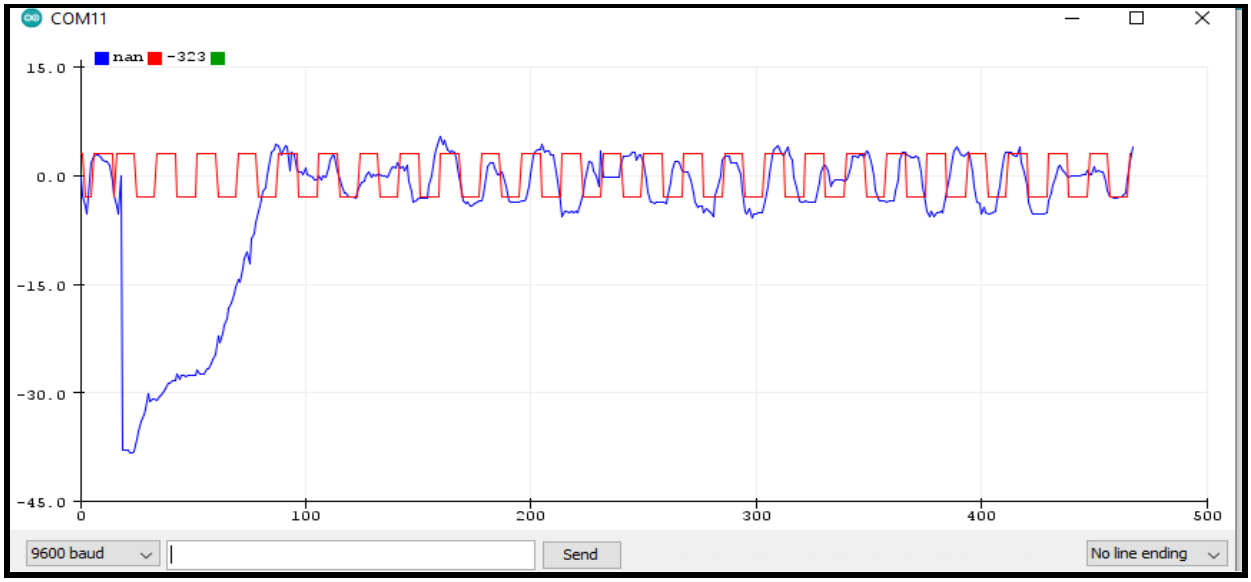


Fig: 7.4 Graph for kp=50,ki=0,kd=0

By observing fig.7.4 as kp is increased kp steady state error is less but oscillates too much.
In this way tried with different values for kp, kd and ki and got stable response around kp=500, kd=20 and ki=0.4

Y-axis : PID controller response **X-axis : Time**
Scale : **Y-axis : 1 unit = 50**

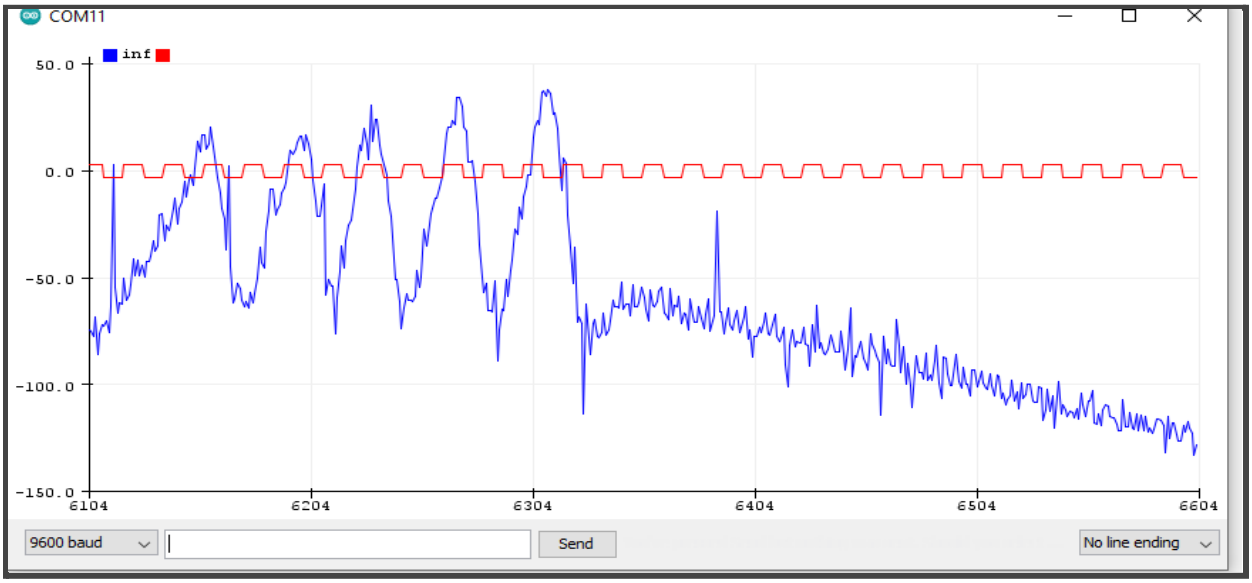


Fig: 7.5 Graph for kp=500,ki=0.4,kd=15

By observing fig.7.5 settling time is reduced to very small but there is some overshoot.so by increasing the value of kd it helps to reduce overshoot and bring the oscillation to damp.

Y-axis : PID controller response **X-axis : Time**
Scale : **Y-axis : 1 unit = 40**

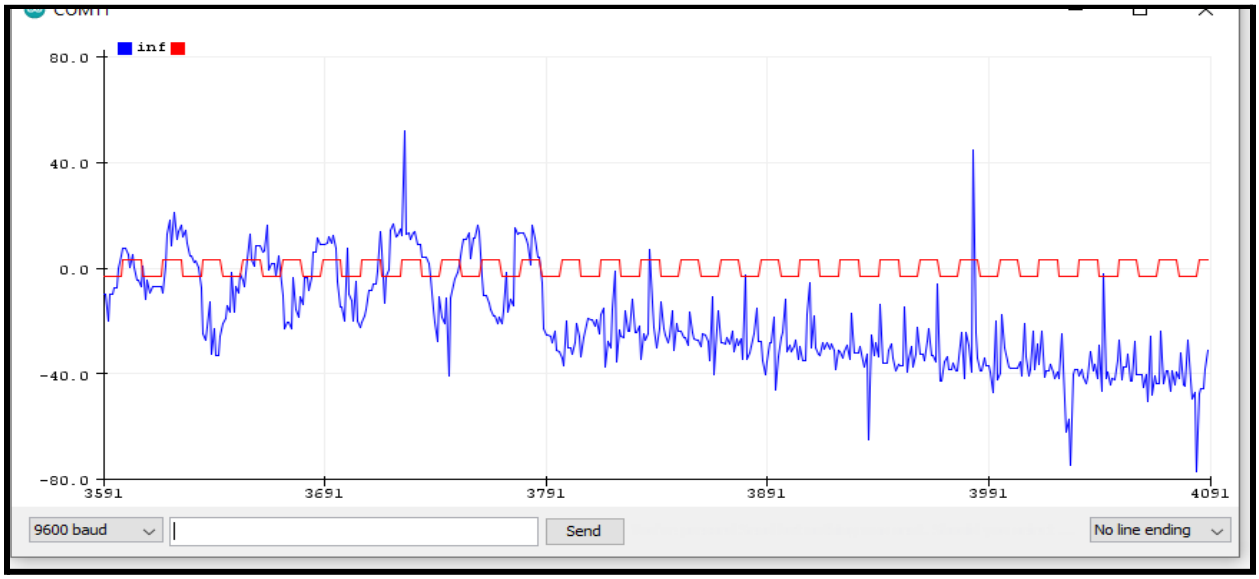


Fig: 7.6 Graph for kp=500,ki=0.4,kd=20

In fig.7.6 it is observed that overshoot and settling time is reduced to very small as desired so this is the most optimum value of kp, ki, kd for this system.

Y-axis : Feedback Voltage **X-axis : Time**
Scale : **Y-axis : 1 unit = 2.5 Volt**

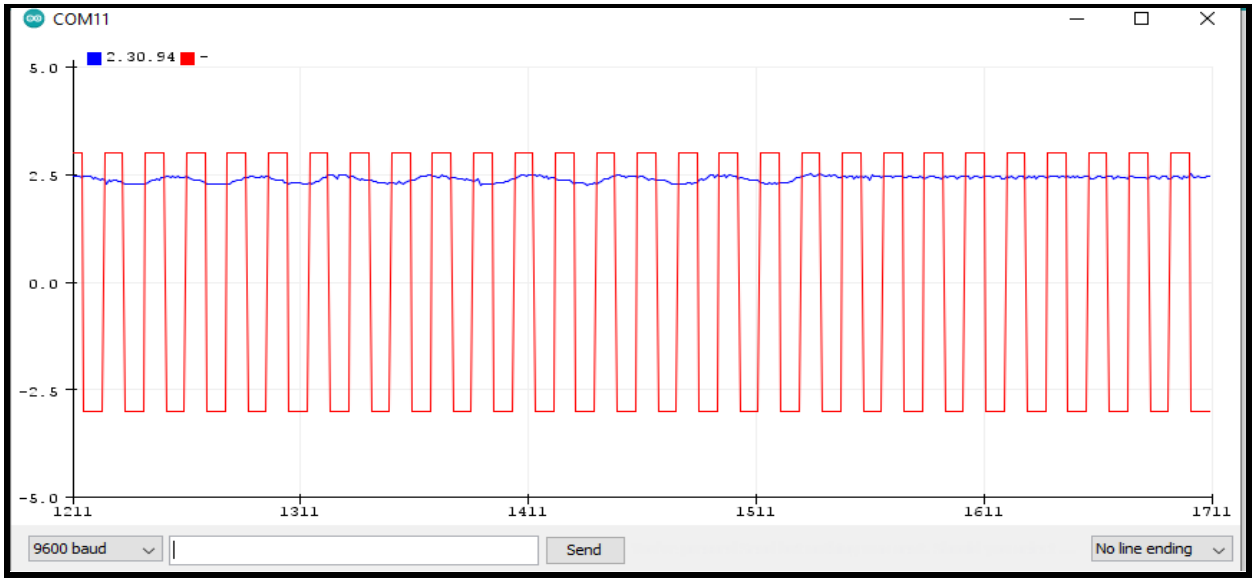


Fig: 7.7 Graph for kp=500,ki=0.4,kd=20

Fig,7.7 shows the graph for feedback voltage at the most stable value of this system.

These are the plot for angle response

Y-axis : Pendulum angle **X-axis : Time**
Scale : **Y-axis : 1 unit = 25 degree**

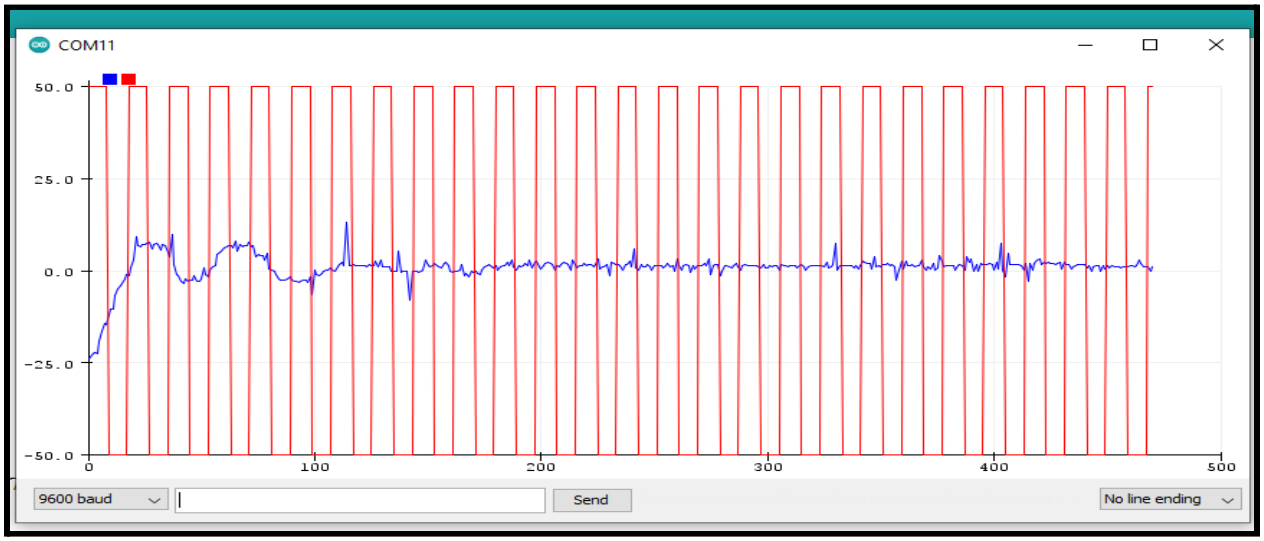


Fig: 7.8 Graph for kp=500,ki=0.4,kd=20

Y-axis : Pendulum angle **X-axis : Time**
Scale : **Y-axis : 1 unit = 25 degree**

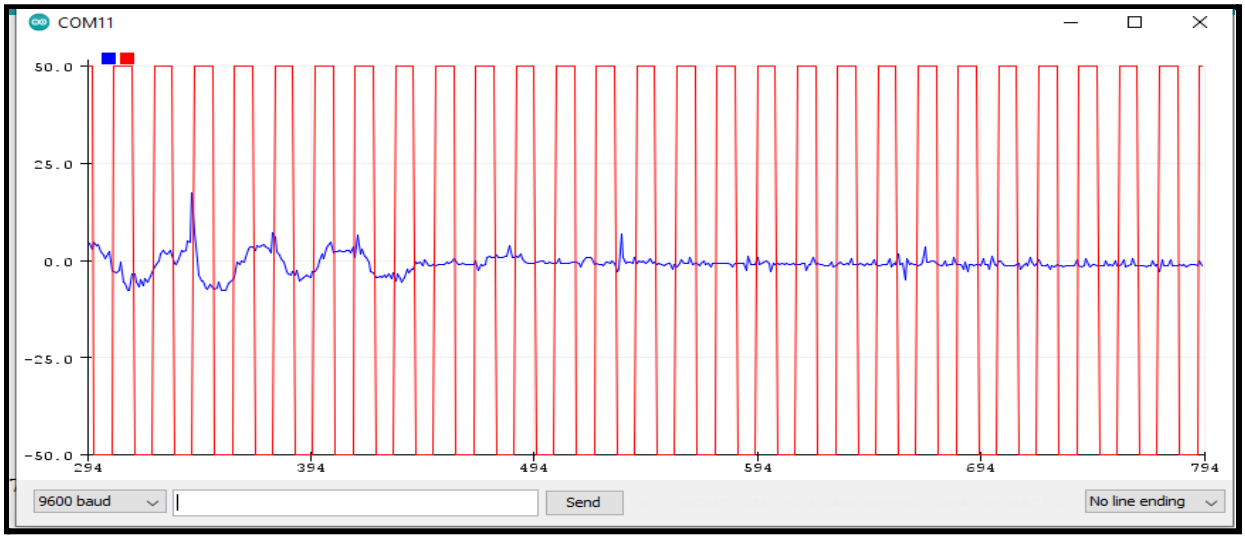


Fig: 7.9 Graph for kp=500,ki=0.4,kd=20

Y-axis : Pendulum angle **X-axis : Time**
Scale : **Y-axis : 1 unit = 25 degree**

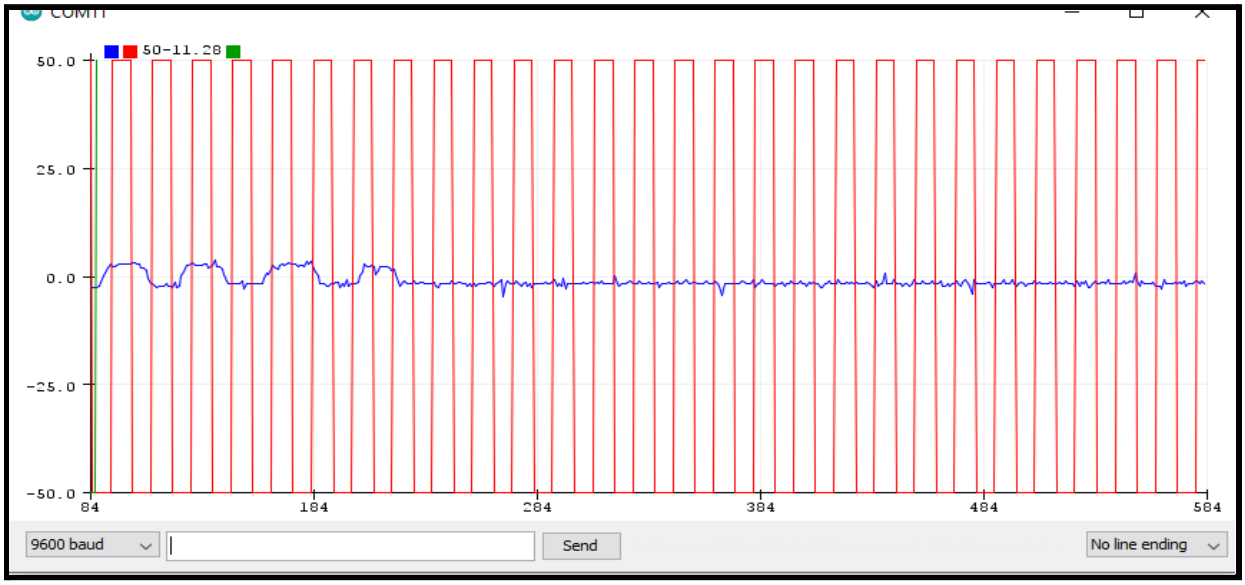


Fig: 7.10 Graph for kp=500,ki=0.4,kd=20

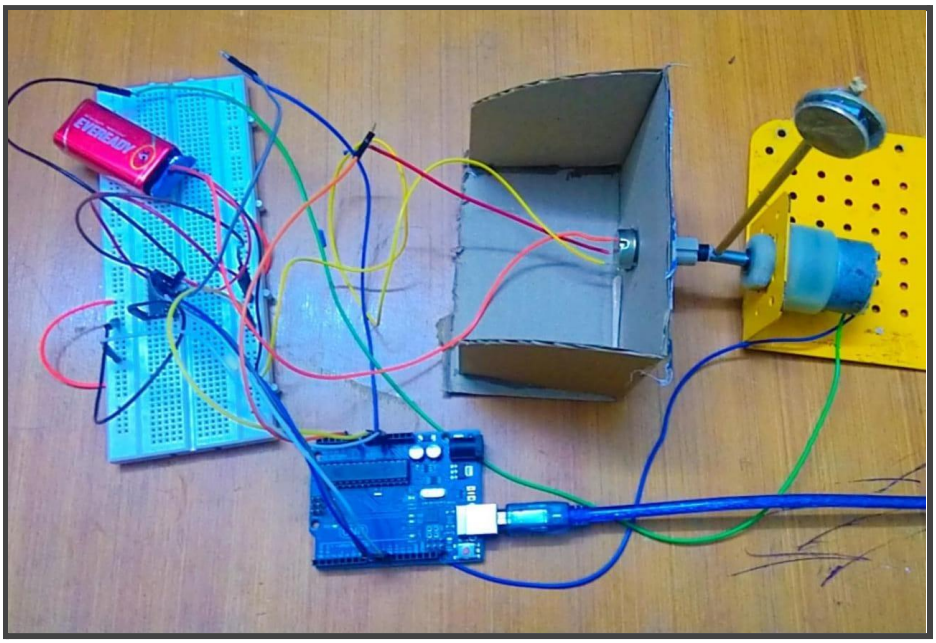
Observation Table:
(Based on fig:7.8, 7.9, 7.10)

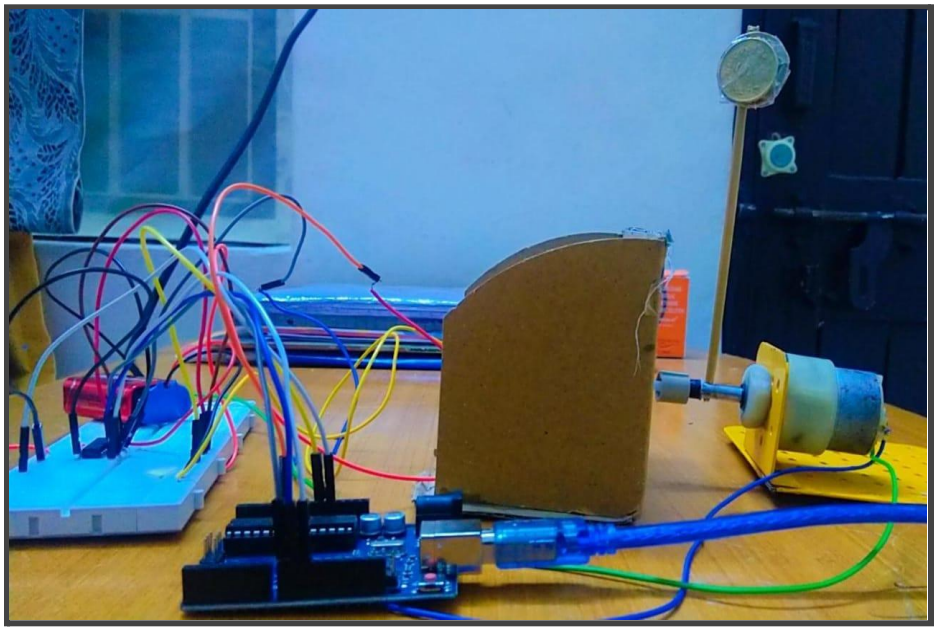
Trail	Overshoot(Degree)	Settling Time(milli sec)	Rise Time(milli sec)
1(7.8)	10	1600	300
2(7.9)	20	1700	100
3(7.10)	7	1600	50

8. Conclusion

- From several test case the conclusion made is that:
- Proportional controller is used to decrease rise time but it increases overshoot .
- Derivative controller is used mainly to reduce the overshoot and it also decreases the settling time, that is it makes the response faster.
- Integral controller is used to reduce the steady state error but in turn it increases overshoot.
- The optimum value of kp, ki, kd for our system is 500, 0.4, 20.
- With these values of kp, ki, kd the rise time obtained is around 100 to 300 millisecond, the settling time is 1600 millisecond and the maximum overshoot is around 10 to 20 degree.

9. Circuit Implementation





10. Troubleshooting

- Thought of using an accelerometer for giving the feedback but that would increase the complexity of the system so switched to potentiometer.
- Faced difficulty while coupling the shaft of Potentiometer and DC Motor, used Potentiometer cap for the same.
- Used DC Motor but it was having a very thin shaft and that's why it was not able to rotate the potentiometer shaft, then used Gear Motor.
- Due to the Gear Motor, the pendulum was not freely falling so increased the mass of the bob, also increased the length of the pendulum stick.
- As the mass of the bob increased , the motor was not able to bring the pendulum to desire position from horizontal position. For that pendulum must be at the recovery angle of the motor.

References

1. Inverted Pendulum: System Modeling
2. Inverted Pendulum: A system with innumerable applications
3. The Inverted Pendulum System
4. Control systems
5. csim - Simulation (time response) of linear system
6. Pole-zero plot of dynamic system - MATLAB
7. Transfer function model - MATLAB
8. Control Systems - MATLAB & Simulink Solutions - MATLAB & Simulink
9. ss2tf - Conversion from state-space to transfer function
10. State-space realization of an impulse response
11. Transfer Functions and Frequency Responses –
12. L293D Motor Driver IC Pinout, Equivalent ICs, Features
13. Inverted Pendulum or Kapitza Pendulum
14. Balancing of an Inverted Pendulum Using PD Controller