International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 07 | July 2020 www.irjet.net p-ISSN: 2395-0072

Designing PID Controller for Balancing Nonlinear beam Model using Arduino

Ashish Vishwakarma[1], Saurabh Yadav[2], Omkar Sawalkar[3], Niranjan Samudre[4]

[1,2,3]Student, Department of Electronics Engineering, Atharva College OF Engineering, Mumbai
[4]Professor, Dept. of Electronics Engineering, Atharva College of Engineering, Maharashtra, India

---

Abstract - *The balancing Beam system with BLDC motors is a nonlinear unstable system of 2-degreee of freedom which has wide application in control engineering. In this paper we design a balanced beam system such that the irrespective of weight and angle of the beam is able to move freely with respect to pivot point on beam without losing the momentum of the beam with the help of BLDC motor flpas. Therefore, we have designed a dedicated PID controller that implement a system which balancing at any point. We have used "Arduino Kit" to design code for PID turner controller*

In our project is the mechanism is to control two brushless motor in order to calibrate the beam. And using the BLDC motors with flaps to balance and calculate the angle using the MPU6050 or the MPU9250 IMU module. The value that we will control is the inclination angle of our beam. The e(t) error will be the difference between the angle of the beam and the desired one. The desired one will be 0, which means that the beam is perfectly horizontal.

Key Words: **Arduino Uno, Proportional-integral-derivate (PID), BLDC motor, beam, MPU 6050, ESC;**

1. INTRODUCTION

The balancing Beam system is a nonlinear unstable system which has wide application in robotics control engineering. There exists different types of balancing beam PID controller system which balances the beam at a pre-defined position. In this paper we design a weight balanced beam system such that the irrespective of weight of the beam is able to move freely at any angle without losing the momentum of the beam. Therefore, we have designed a dedicated PID controller that implement a system which balances a point on a beam. We have used "Arduino Kit" to design code for PID turner controller and

Generally, the balancing beam system is connected to real control problem such as horizontally stabilizing an airplane during landing and in turbulent airflow. There are two degrees of freedom in this system. First is the Brushless motor rotating on the beam and second is the beam rotating through its center axis. The main purpose of the system is to control the angle of the beam to a desired angle by manipulating the rotation speed of Brushless motor(BLDC). The angle of the beam can be measured using a special sensor .i.e. MPU 6050. The control voltage signal is send to the ESC motor controller through a power supply of 12V 10A and then the torque generated by the motor drive the beam to rotate to the desired angle. Thus unstable problem of the system can be solved by closing the open loop with a feedback controller. There are many types of feedback control can be used to stabilize the system such as Proportional Integral Derivative (PID) Control.

2. SYSTEM PRATICAL MODEL

System Practical model is separated into two parts; the mechanical part which consists of beam, lever arm (aluminum bar) and BLDC Motor and electrical part which consists of DC motor, sensor and Arduino microcontroller.

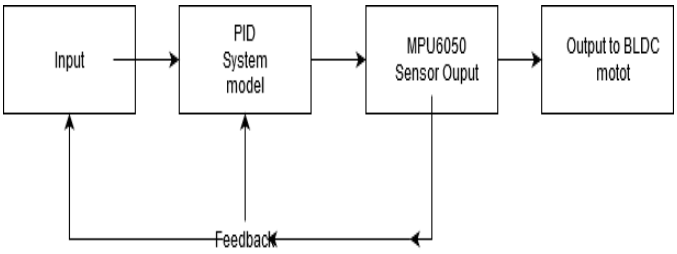

Fig1: Basic Block Diagram

2.1. System Mechanical Part Selection of appropriate material for a mechanical part is an essential element of all engineering projects. The main mechanical parts of the system are the base support, Following materials are used in to build the system:

- Base support made of wood has length of (40 cm) and width of (20 cm).
- Aluminum beam which has length of (34 cm), and width of (2 cm).
- Motor controller (ESC).

• Lever horn. 3.2. System Electrical Part.

The Electrical circuit consists of:

• Arduino Uno • BLDC Motor (2200Kv)

• Accelerometer and Gyroscope sensor (MPU 6050).

The accelerometer measures the acceleration of the 3 axes: X, Y and Z, the three dimensions of our 3D space. The IMU movement is upwards, the Z axis, forward and backward the Y axis and side to side the X axis. The IMU also detects the acceleration of terrestrial gravity. But the values of IMU is "g" units. Therefore, this module works with only 8 bits registers that are stored in two registers of low and high bits. The sum of the registers give us 16 bits of data. This data will have a maximum of 16 bits: 2^16=65536 maximum acceleration value including the positive/negative sign. The terrestrial gravity is used for the accelerometer readings to know the inclination angle with respect to the X axis or Y axis. Considering, IMU is perpendicular to the ground axis (floor). Therefore the Z axis 1g = 9.8 m/s^2, and the two X&Y - axes is 0. Now suppose we turn the IMU 90 degrees such that the X axis is perpendicular to the ground, therefore it will mark the acceleration of 1gm/s^2 gravity.

Therefore the total angle is give by formula:

Total_angleX = 0.98(Total angleX + GyroDataY.elasped Time) + 0.02.atan(y/sqrt(X^2+Z^2))

Total_angleY = 0.98(Total angleY + GyroDataX.elasped Time) + 0.02.atan(y/sqrt(Y^2+Z^2))

The beam changes to correct position by manipulating BLDC angle rotation speed. In Arduino code we can set the desired angles “0” degrees or 180 radians and also seting the initial throttle speed of BLDC motor to 800rpm. We can get the beam angle by MPU 6050 with angle error. Angle Error is corrected difference between the total angle and desired angle for the balancing the beam on X-axis and Y-axis. We have chosen the x-axis plane angle to implement the PID i.e. the x axis of the IMU has to be parallel to the balance plane. First calculate the error between the desired angle and the real measured angle (error = Total_angle - desired_angle) and proportional to PID constant (PID_p = Kp*error). The integral error is the part that should only act if it is close to the desired positioned angle but fine tuning the error is primary goal of PID. If operation for an error is between (-3 to 3) degree. To integrate sum the previous integral values with the errors multiplied by the integral constant. This will integrate the each value loop till it reaches to 0 point:

```
If (-3 <error <3)
{
  pid_i = pid_i+(ki*error);
}
```

The derivate error acts upon the speed of the error. The speed is the amount of error that produced in a certain amount of the time divided by that time, for that defined a variable called previous error.

Taking the difference between the value from the actual error and dividing it by time and multiply the result by the derivate constant:

```
PID_d = Kd*[(error - Previous_error)//Time];
```

Since PID is the sum of each values: [PID = pid_p + pid_i + pid_d.] The Arduino has the min value of PWM (Pulse Width Modulation) signal of 1000us and the max 2000us such that the PID values can oscillate between less than (-1000us to 1000us) because when values of 2000us the maximum value that could subtract is 1000 and when the value of a 1000us for the PWM signal, the maximum value that we could add is 1000 to reach the maximum 2000us:

```
if(PID < -1000)
{
  PID=-1000;
}
if(PID > 1000)
{
  PID=1000;
}
```

Summing the total desired throttle and the PID values

```
Left = throttle + PID;
Right = throttle - PID;
```

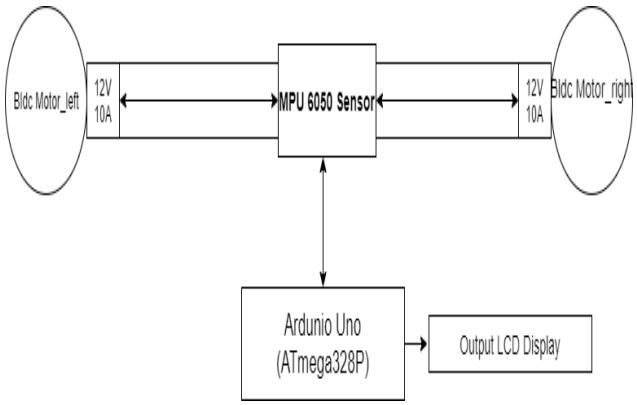

Fig2: Practical model diagram

Pseudo Code for PID:

1. previous_error = 0
2. integral = 0

3. while loop:
4. {
5. error = (setpoint − measured_value)
6. integral += (integral + error × dt)
7. derivative = [(error − previous_error) / dt]
8. output = (Kp × error + Ki × integral + Kd × derivative)
9. previous_error += error
10. wait for (dt)
11. }
12. goto (loop)

3. Result

Fig3: Final Model

With the placement of powerful sensors (MPU 6050), our mechanical model was able to react to the changing angle of the beam correctly providing sufficient resolution to implement this technique. PID algorithm in Arduino code proofed to be very fast which gains when tuned properly. The placement of two motor with the sensor reading both accelerometer and gyroscope with feedback response of changing throttle speed of BLDC motor. Hence we implement the balancing beam of two degree freedom using PID algorithm.

4. CONCLUSIONS

Balancing system in a 2-degree of freedom platform is of unique importance in understanding the control system applications and robotic benchmark problem in control engineering. Horizontally stabilizing an airplane during landing and in turbulent airflow. Stabilizing the drone if strong wind is blowing. Provide better stability of the vehicle increases riding comfort. Our model can be easy implemented in any environment given better sensors and microcontroller.

REFERENCES

[1] A. Zeeshan, N. Nauman and M. Jawad Khan, "Design, control and implementation of a ball on plate balancing system," Proceedings of 2012 9th International Bhurban Conference on Applied Sciences & Technology (IBCAST), Islamabad, 2012, pp. 22-26.M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[2] M. Amjad, M. I. Kashif, S. S. Abdullah and Z. Shareef, "A simplified intelligent controller for ball and beam system," 2010 2nd International Conference on Education Technology and Computer, Shanghai, 2010, pp. V3-494-V3-498.K. Elissa, "Title of paper if known," unpublished.

[3] Advanced PID Control Book by Karl Johan Åström and Tore Hägglund

[4] https://en.wikipedia.org/wiki/PID_controller#Loop_tuning

[5] https://in.mathworks.com/discovery/PID

[6] Advanced PID Control Book by Karl Johan Åström and Tore Hägglund