Self Balancing Robot

Sahil Patil
Electronics & Telecommunication Engineering
Vidyavardhini's College of Engineering & Technology
Vasai(w), India

Dhananjay Bhavsar
Electronics & Telecommunication Engineering
Vidyavardhini's College of Engineering & Technology
Vasai(w), India

Shubham Mantri
Electronics & Telecommunication Engineering
Vidyavardhini's College of Engineering & Technology
Vasai(w), India

Shraddha Gosavi
Electronics & Telecommunication Engineering
Vidyavardhini's College of Engineering & Technology
Vasai(w), India

***Abstract*— Self-balancing robot is an effective approach to the development and advancement in the field of robotics. In this particular model, the concept of inverted pendulum is used. Self-balancing is a process by which a system achieves stability by internal forces. The basic idea of this project is to overcome the challenge of balancing initially unstable system, by providing control mechanism to the robot so that it can balance on its own. The robot uses sensor values provided by accelerometer and gyroscope to find exact position of itself in three-dimensional geometry and send the values to microcontroller. The microcontroller on the other hand uses programs in it to give proper instruction about rotation of wheels to the motor driver module which in turn helps to balance the robot. This robot is advantageous over traditional four wheeled robots as it helps in taking sharp turns and navigating through tighter areas thus, serving as an essential machine for various industrial applications.**

Keywords- ATMEGA328P, Self-Balancing, Accelerometer, Gyroscope, PID, Complementary filter

I. INTRODUCTION

The invention of self-balancing robots has been a massive milestone in the history of robotics. These machines are particularly characterized by their ability to balance on two wheels by the implementation of a closed loop algorithm. A self-balancing robot has an unstable dynamic system unlike any other robot which rests itself on either three or more wheels. It works on the same phenomena as that of an inverted pendulum. Its design is more complex, as it needs to maintain its upright (vertical) position, however this design has many advantages which allows it to be used in practical scenarios. It's ability to turn on the spot and sustainable architecture increases its applications in industries. It is essential for the robot to not only balance but also maintain its position, withstanding external forces or unexpected disturbances if any. Active researches on two wheeled robots have been widely increased since the early versions of the studies on self-balancing robots by JOE[1] and n-BOT[2] complete with inertial activity sensors, encoders and on-wheel microcontrollers. A board of digital signal processor was featured by JOE, which was
based on a controller board on various microprocessors and the ATMEGA series of the Atmel architecture. It is rapidly emerging as a popular platform for both education and product development, with applications ranging from robotics, to process control and network control.

This paper explains the design along with the construction and control mechanism of a two-wheel self-balancing robot. This robot is self-driven by the interaction of the following components: DC motors, a microcontroller, a gyroscope(3-axis) and an accelerometer(3-axis) for attitude determination. To deal with the problem of sudden horizontal movements and gyro drifts in sensors, a complementary filter is implemented[3]. PID(proportional integral derivative), is the feedback mechanism used for this project.

II. BLOCK DIAGRAM OF CONTROL SYSTEM

Fig.1 illustrates the block diagram of the system. This shows the output from certain blocks and their respective inputs. The microcontroller ATMEGA328P acts like the heart of the system. Because it manages to input readings from the Sensors which include Accelerometer and Gyroscope, and Complementary filter , and based on these readings communicates with the motor driver.

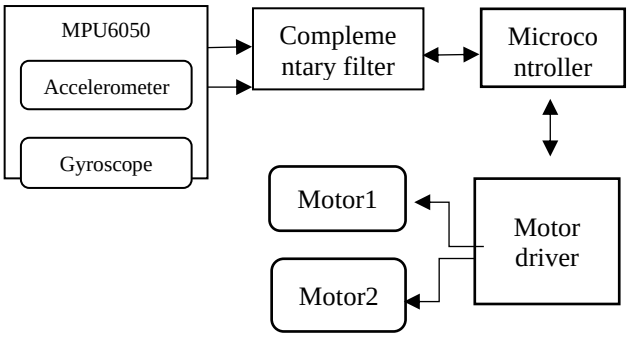

Fig.1 : Block Diagram

III. WORKING AND METHODS

To keep the robot balanced, the motors must counteract the robot falling by rotating the wheels in desired direction as shown in Fig.2. This action requires feedback and correcting elements. The feedback element is the MPU6050 gyroscope + accelerometer, which gives both acceleration and rotation in all three axes. The Arduino uses this to know the current

orientation of the robot. The correcting element is the motor and wheel combination.

Fig.2. : Working Principle

To know the exact position of robot we need to use sensor values. Accelerometer gives an indication of orientation in static conditions. Gyroscope is a good indicator of tilt angle in dynamic conditions. We have to combine both the slow moving signals from accelerometer and fast moving signals from gyroscope. The gyroscope mainly comprises a drift and in a lesser time the values returned are completely wrong. The accelerometer, on the other hand, returns true values when the acceleration is progressive, but it suffers from vibrations, thus resulting in wrong angle values. So in order to have correct values we use a Complementary filter which is a math filter used to combine both signal values. The concept behind Complementary filter is to allow the accelerometer signals to pass through a low-pass filter and the gyroscope signals to pass through a high-pass filter and finally combine the results to give the final signal[4].
Equation for low pass filter[4]:

$$y(n)=[(1- \alpha).x(n)]+ [\alpha.y(n-1)]$$

Where,
x(n)→ pitch/roll/yaw output from the accelerometer
y(n)→ filtered final pitch/roll/yaw which is the input for the next phase of program.

Equation for High pass filter[4]:

$$y(n)=[(1- \alpha).y(n-1)]+[(1- \alpha)(x(n)-x(n-1))]$$

Where,
x(n) → is the pitch/roll/yaw output from the gyroscope
y(n) → is the filtered final pitch/roll/yaw which is the input, for the next phase of program.

Here, 'n' depicts the indication of current samples and 'α' defines the boundary limit where the accelerometer readings tends to stop and the gyroscope readings take over and vice-versa. It mainly deals about how much of the output should depend on the current value or a new value that arrives. Both the values of 'α' need to be same, and it is usually greater than 0.5 from the definitions above.

$$\alpha = \frac{\tau}{\tau + dt}$$

This equation is derived from Filter/Control theory. Where, τ is the desired time constant(determines the response speed of the readings),and

$$dt = \frac{1}{fs}$$

Where, fs →sampling Frequency.

The current angle of inclination(Θ) is determined by combining the values of angles from both gyroscope and accelerometer, according to the formula:

$$\Theta = [(1- \alpha).(previousAngle + \beta1)] + [\alpha.(\beta2)]$$

Where,
β1 →angle obtained from gyroscope
β2 →angle obtained from accelerometer

First reading is the angle as resulted from gyroscope(β1) by integration. Accelerometer gives the second reading. For instance when the output of gyroscope is null(0), angle will converge to that of the result given by accelerometer(β2). If the value of 'α' is very small the output angle will not trust the reading from the accelerometer and eventually it will trust the gyroscope.

In order to control the speed at which wheels should pivot,PID (proportional integral derivative), which is a control loop feedback mechanism largely used in control systems, is used. This concept is mainly used for calculation of 'error' value. Here firstly the measured and desired set points are calculated; the difference is obtained for further calculations. The proportional(P) term which is derived from PID is based on the current angle difference from Zeroth Point. The integral(I) term derived from the PID relies on the current angle difference or error(e(t)) from point zero which is then multiplied by the gain, and subsequently accumulated over time. The integral control also contributes in balancing the robot if it is moving. The derivative (D) term which is derived from PID is the current rate of rotation. In order for the robot to balance properly the hardware/physical orientation of the robot plays a vital role. In other words the center of gravity of the robot should be balanced. The PID constants i.e, kp, kd and ki can be determined by trial and error method. Fig.3 briefly represents the flowchart of the PID algorithm.

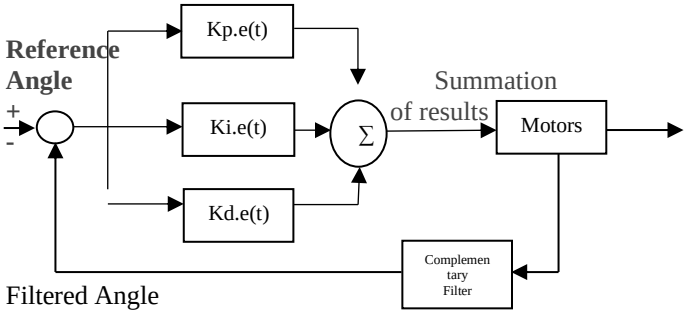

Fig.3 : PID Flowchart

IV. COMPONENTS

The microcontroller ATMEGA328P along with MPU6050 (accelerometer and gyroscope) and L298N motor driver is responsible for balancing of the robot. The sensor module MPU6050 and motor driver L298N are programmed by the

help of microcontroller. The position of the robot in 3-dimensional geometry(angle of inclination) and the angular velocity of the wheels(speed by which the robot is falling) is determined by the sensor MPU6050. The microcontroller, by utilizing these values instructs the motor driver module to rotate the motors in order to counteract the falling motion of the robot.

1) ATMEGA-328P [5]

It is a high performance 8-bit microcontroller based on AVR RISC architecture. It is a 28 pin microcontroller which has 32KB flash memory and runs on a frequency of 20Hz with external crystal within 1.8v to 5v operating voltage.

2) MPU6050 IMU [6]

It is an Inertial measurement Unit commonly known as IMU, which is used to measure gravitational acceleration and rotational velocity. It is a 6-axis sensor module which contains 3-axis accelerometer and 3-axis gyroscope integrated on a single chip. It is basically used to determine the exact location of the robot in 3-dimensional space. The outputs of the gyroscope are defined in degrees per second. Hence in order to get the angular position integration of the angular velocity is needed.

Gravitational acceleration can be measured by the Accelerometer along the 3 axes and thereby by using some sort of math we can calculate the sensor position. So by combining the accelerometer data values and gyroscope data values we can get very precise information defining orientation of sensor.

3) L298N Motor Driver[7]

This module is used to drive DC motors. As the module uses dual H-bridge drive, it is possible to drive two motors at the same time. It supports driver voltage from 5v to 35v.

V. RESULTS

For the robot to stand upright, it is necessary that the microcontroller generates an appropriate output which is then sent as a command to drive the motors. This command is generated by multiplying each of the PID(proportional, derivative and differential) response components by their corresponding constants (Kp, Kd and Ki) and summing the result. The values of these constants is obtained by performing the following steps.

STEPS FOR TUNING THE PID CONSTANTS

Step 1: Set Ki and Kd to zero and gradually increase Kp so that the robot starts to oscillate about the zero position.
Step 2: Increase Ki so that the response of the robot is faster when it is out of balance. Ki should be large enough so that the angle of inclination does not increase. The robot should come back to zero position if it is inclined.
Step 3: Increase Kd so as to reduce the oscillations. The overshoots should also be reduced by now.
Step 4: Repeat the above steps by fine tuning each parameter to achieve the best result.

By executing the above mentioned steps, we obtained the suitable values of the PID constants(in double data type) as:

- Kp = 60
- Kd = 1.4
- Ki = 70

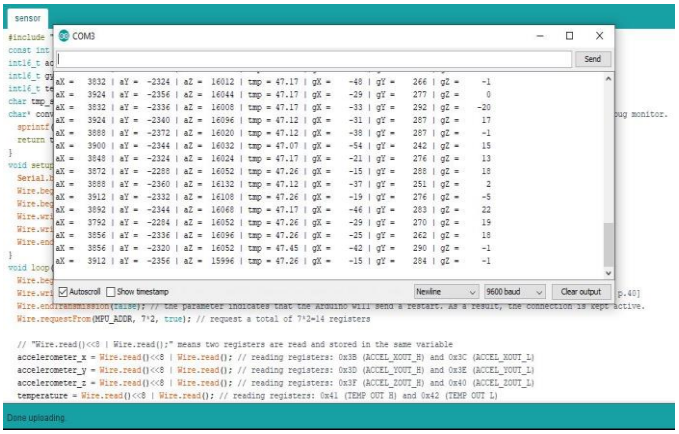

Fig.4 : Results obtained from Sensors

Fig.4 shows the acceleration components obtained from accelerometer and gyro outputs across all three axes in coordinate planes. This is basically the data/information regarding the position of the robot , which is achieved from the 6-axis motion tracking device(MPU6050).

VI. CONCLUSION AND FUTURE SCOPE

In this project we were able to implement a self-balancing robot by using PID tuning method. The assembled configuration of the robot is shown in Fig.5. The bot balances itself effectively while leaning in forward or backward directions by the implementation of a closed loop algorithm.

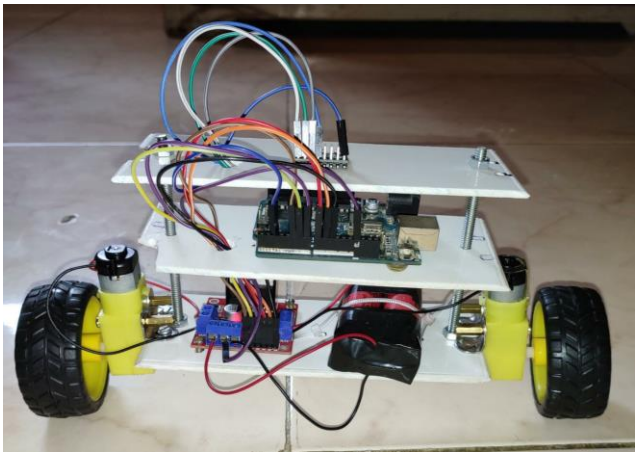

Fig.5 : Assembled model

The performance can be further improved by enhancing the precision of motor speed readings, thereby improving stability. Also by installing a Bluetooth module or some wireless technology it is possible to create a remote controlled prototype, increasing its array of applications.

The above mentioned features can be used as a modification to the existing system while designing a more efficient system in the future. These robots can be used for smart gardening purposes; autonomous trolleys in malls, hospitals and airports; an intelligent robot for various purposes. Currently popularized as “Segways”, these

machines are mostly used for travel and tourism purposes and by private security services. It has been put to use by a range of private and military organizations since its invention.

VII. REFERENCES

[1] F. Grasser, A. D'Arrrigo, S. Colombi, and A. C. Rufer, "JOE: A mobile, inverted pendulum," IEEE Transactions on Industrial Electronics,vol. 49, no. 1, pp. 107–14, 2002.
[2] D. P. Anderson. (2003, Aug.) nBot balancing robot. Online. [Online].Available: http://www.geology.smu.edu/ dpa-www/robo/nbot.
[3] A.-J. Baerveldt and R. Klang, "A low-cost and low-weight attitude estimation system for an autonomous helicopter," in IEEE InternationalConference on Intelligent Engineering Systems, sep 1997, pp. 391–5.
[4] https://sites.google.com/site/myimuestimationexperience/filters/complementary-filter
[5] [https://www.sparkfun.com/products/9061]
[6] [https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/]
[7] [http://wiki.sunfounder.cc/index.php?title=Motor_Driver_Module-L298N]