Self Balancing Robot

Akshat Malviya(B17003)*, Akshita Jain(B17004)†, Divya Gupta(B17083)‡,
Manvi Gupta(B17092)§, Shreya Lanjewar(B17142)¶
* b17003@students.iitmandi.ac.in † b17004@students.iitmandi.ac.in ‡ b17083@students.iitmandi.ac.in
§ b17092@students.iitmandi.ac.in ¶ b17142@students.iitmandi.ac.in

***Abstract*- In this project we aim to make an unstable robotic platform balance on its two wheels in a vertical position. The basic idea involved in a self-balancing robot is simple. It is prevented from falling by giving acceleration to the wheels according to its inclination from the vertical axis. This self-balancing electro-mechanical system has various uses. By using the results from gyroscope and the calculated PID controller results, it can solve the problem of inverted pendulum, in which a large mass is placed at the end of a pole. Such robots can substitute the current day short distance vehicles and with some improvisations, it can also be used as a guide to the physically disabled people.**

I. APPARATUS REQUIRED

TABLE I
COMPONENTS USED IN THE ROBOT

Sr No.	Apparatus	Quantity
1.	Jumper Wires	Approx. 30
2.	DC Motors(200 rpm)	2
3.	Raspberry pi	1
4.	Gyroscope(MPU6050)	1
5.	Hard Board and glue/glue-gun	1
6.	Motor Driver (L298N)	1
7.	Lipo Battery (2200mAh, 5V)	1
8.	Wheels	2
9.	Power bank (5V)	1

II. REQUIREMENT ANALYSIS

A. Raspberry Pi 3 Model

Raspberry Pi 3 Model B is a small single-board computer, developed to promote the teaching of computer science. It uses a Broadcom BCM2837 SoC with a 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor, with 512 KB shared L2 cache. Its power rating is 700mA (3.5W). It is provided with 4*USB2.0 ports which can be used to connect keyboards and mouse. It has inbuilt wifi module and bluetooth module. The Raspberry Pi has a set of GPIO(General Purpose Input Output) pins which can be programmed using the python library GPIO. We can take inputs, give outputs using these GPIO pins. It also has multiple Ground, 5V, and 3.3V pins mounted on it for convenience.

Fig. 1. Raspberry-pi

B. IC L298N Motor Driver:

The L298N is a dual H-Bridge motor driver which allows speed and direction control of two DC motors at the same time. The module can drive DC motors that have voltages between 5 and 35V, with a peak current up to 2A. The module has two

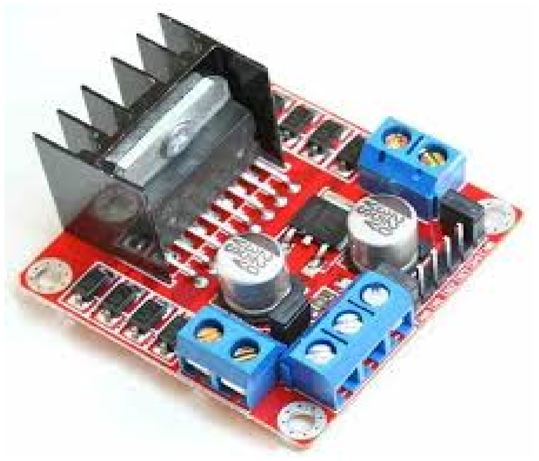


Fig. 2. Motor Driver (L298N)

screw terminal blocks for the 2 motors, and another screw terminal block for the Ground pin, the V_{cc} for motors and a 5V pin which can either be an input or output.

C. *The InvenSense MPU-6050:*

For the robot to be balanced, it requires sensors that allow it to measure its tilt accurately and at a high measurement rate. The MPU-6050(Motion Processing Unit) sensor has a microelectromechanical accelerometer and a microelectromechanical gyroscope in a single chip. It contains 16-bits analog to digital converter for each channel. Therefore it captures the x, y and z channels simultaneously. The sensor uses the I2C-bus to interface with the raspberry-pi.

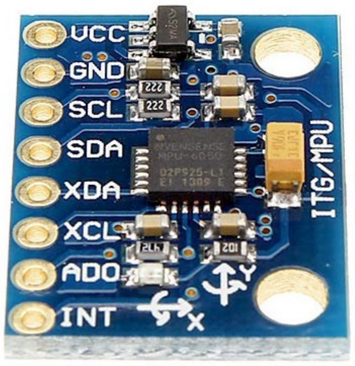


Fig. 3. MPU-6050

III. INTRODUCTION

Self-Balance Robot is somewhat which balances itself and automatically corrects its position on disturbance. The robot body mainly comprises of one base board on the wheels and three other boards. The motors, wheels, gyroscope and motor drivers are attached with base. The DC motors in the base of the device keep the vehicle upright when powered on with balancing enabled. Gyroscopic sensor is used to detect tilting of the device which indicates a deflection from perfect balance. Motors driving the wheels are commanded to change the direction accordingly to bring the vehicle back into balance. The vehicle has electric motors powered by lithium-polymer battery. Steering can be controlled by simply varying the speeds between the dc two motors.

IV. CALCULATIONS

The main motive is to calculate the angle of tilt from it's reference position and according to that data we can decide the direction of rotation of motor.This is done using gyroscope.

A. *The Inverted Pendulum Problem*

Balancing robots represent the classic inverted pendulum problem, in which a large mass is placed above the pivoted point of the pole. The pole is free to rotate about the pivot(base), and the base is free to move in the plane perpendicular to the vertical. The goal of the problem is to keep the pole vertical by moving the base in response to changes in the angle.

B. *Angle Estimation:*

To find the direction and the tilt angle, an accelerometer and a gyroscope sensor is used. Although both accelerometer and gyroscope can be individually used to calculate tilt angle, in practice, it is often used in collaboration.

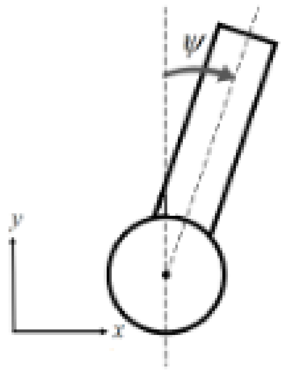
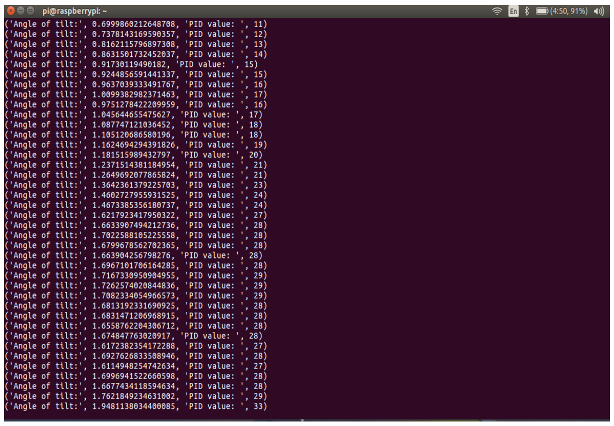


Fig. 4. Tilt Angle

C. *Maximum angle of tilt:*

In order to determine he maximum angle of tilt beyond which the robot(system) will not be able to come back to its original position of equilibrium, we conducted different experiments. Many times the robot balanced and many times it couldn't since the torque provided by the motors wasn't sufficient to bring it back once the angle exceeded a particular value. The approximate maximum angle of tilt came out to be around 20 to 25 degrees.

D. *PID calculations*



```
('Angle of tilt:', 0.6999860212648708, 'PID value: ', 11)
('Angle of tilt:', 0.7378143169590357, 'PID value: ', 12)
('Angle of tilt:', 0.8162115796897308, 'PID value: ', 13)
('Angle of tilt:', 0.8631501732452037, 'PID value: ', 14)
('Angle of tilt:', 0.91730119490182, 'PID value: ', 15)
('Angle of tilt:', 0.9244856591441337, 'PID value: ', 15)
('Angle of tilt:', 0.9637039333491767, 'PID value: ', 16)
('Angle of tilt:', 1.0099382982371463, 'PID value: ', 17)
('Angle of tilt:', 0.9751278422209959, 'PID value: ', 16)
('Angle of tilt:', 1.045644655475627, 'PID value: ', 17)
('Angle of tilt:', 1.087747121036452, 'PID value: ', 18)
('Angle of tilt:', 1.105120686580196, 'PID value: ', 18)
('Angle of tilt:', 1.1624694294391826, 'PID value: ', 19)
('Angle of tilt:', 1.181515989432797, 'PID value: ', 20)
('Angle of tilt:', 1.2371514381184954, 'PID value: ', 21)
('Angle of tilt:', 1.2649692077865824, 'PID value: ', 21)
('Angle of tilt:', 1.3642361379225703, 'PID value: ', 23)
('Angle of tilt:', 1.4602727955931525, 'PID value: ', 24)
('Angle of tilt:', 1.4673385356180737, 'PID value: ', 24)
('Angle of tilt:', 1.6217923417950322, 'PID value: ', 27)
('Angle of tilt:', 1.6633907494212736, 'PID value: ', 28)
('Angle of tilt:', 1.7022588105225558, 'PID value: ', 28)
('Angle of tilt:', 1.6799678562702365, 'PID value: ', 28)
('Angle of tilt:', 1.663904256798276, 'PID value: ', 28)
('Angle of tilt:', 1.6967101706164285, 'PID value: ', 28)
('Angle of tilt:', 1.7167330950904955, 'PID value: ', 29)
('Angle of tilt:', 1.7262574020844836, 'PID value: ', 29)
('Angle of tilt:', 1.7082334054966573, 'PID value: ', 29)
('Angle of tilt:', 1.6813192331690925, 'PID value: ', 28)
('Angle of tilt:', 1.6831471206968915, 'PID value: ', 28)
('Angle of tilt:', 1.6558762204306712, 'PID value: ', 28)
('Angle of tilt:', 1.674847763020917, 'PID value: ', 28)
('Angle of tilt:', 1.6172382354172288, 'PID value: ', 27)
('Angle of tilt:', 1.6927626833508946, 'PID value: ', 28)
('Angle of tilt:', 1.6114948254742634, 'PID value: ', 27)
('Angle of tilt:', 1.6996941522660598, 'PID value: ', 28)
('Angle of tilt:', 1.6677434118594634, 'PID value: ', 28)
('Angle of tilt:', 1.7621849234631002, 'PID value: ', 29)
('Angle of tilt:', 1.9481138034400085, 'PID value: ', 33)
```

Fig. 5. Screenshot Of Tilt Angle and Corresponding PID values

PID control, which stands for Proportional-Integral-Derivative control, is a control loop feedback mechanism. As the name suggests, PID algorithm consists of three basic coefficients: proportional, integral and derivative we varied

which to get optimal response.
After building the dynamic model and tuning the robot, we found that it was extremely difficult to balance the robot even if the centre of mass is within the estimated range to satisfy the small-angle approximation. We tried to approximate the value of k_p,k_i,k_d to balance the robot with hit and trial method.
We realized that too High K_p leads to oscillation in values and tends to generate an offset. K_i counteracted the offset. Increasing Kd improved rise time, overshoot, and settling time (to a point). We took a lot of values for trial and from observation we finally came to the following values of the constants:
k_p=10, k_d=3, k_i=3 .

E. *Assumptions*

These assumptions hold for small tilt angles:

- We have assumed the robot as a linearly translating body without taking into consideration it's moment of inertia.
- We are controlling the motor torque by varying the input voltage, assuming the relation between the torque and the voltage to be linear.

V. Block Diagram

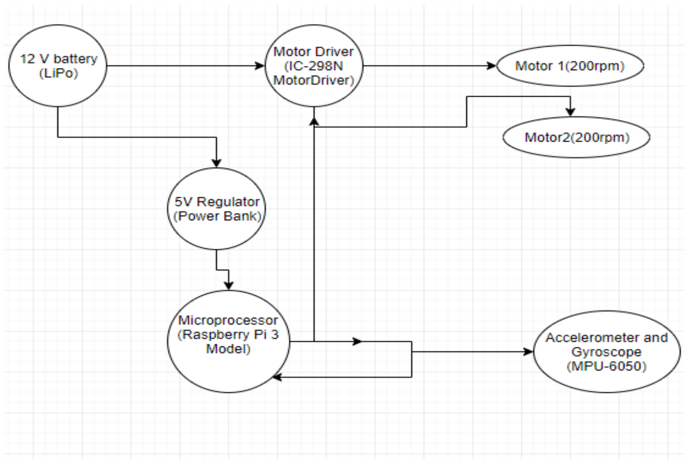

12 V battery (LiPo)
Motor Driver (IC-298N MotorDriver)
Motor 1(200rpm)
Motor2(200rpm)
5V Regulator (Power Bank)
Microprocessor (Raspberry Pi 3 Model)
Accelerometer and Gyroscope (MPU-6050)

Fig. 6. Block Diagram

VI. Hardware Design

The balancing robot we built is basically a solution of a simple inverted pendulum on two wheels. Though there were many constraints while making the body and balancing it, we finally balanced it to a great extent. The body of the robot consists of rectangular boards. The dimensions of rectangular sheets used are two 25cmx7cm sheets, one 15cmx25cm and two 7cmx15cm sheets. The sheet with dimension 7cmx15cm was used as the base of robot and another was used as a roof as shown in the figure.
Initially, we tried placing the battery on top of the body for better oscillations (as the centre of mass shifts upwards) but it wasn't successful since the counter torque required to balance the robot was much more than the what motors could provide. Hence, the motor driver and Li-Po was attached to the base board. The other 2 boards with dimensions 25cmx7cm were vertically attached with the base. DC motors and wheels were also attached to it. The board with dimensions 15cmx25cm was attached to the back part of the robot. And the raspberry-pi was attached to this vertical board.
We tried to use 500rpm motors but it failed to balance as the motors were not suitable according to the weight and dimensions of the robot. So, we decided to use motors of a lower rpm and ended up using those with a value of 200rpm which improved the performance.

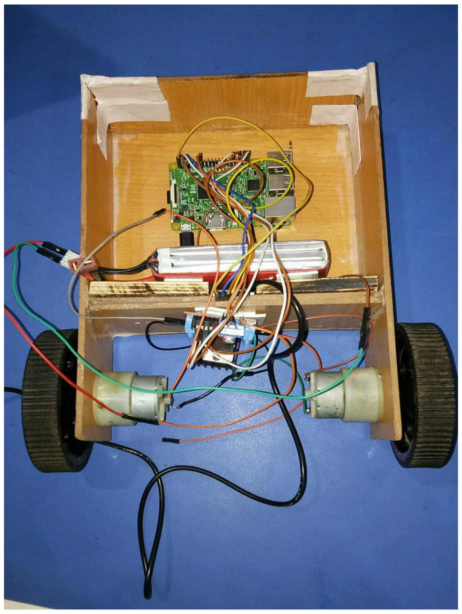


Fig. 7. Robot

VII. Motivation

Since childhood, we have unknowingly been practicing to balance various objects. It may be balancing a stick on the palm, moving with a glass of water filled up to the brim, walking on a narrow wall, cycling, etc. All of it requires a balancing algorithm and with time, we have trained our brain to do so.
Self-balancing Robot is the topic of interest among students and researchers. We, in this project, were working on a similar concept with a focus on handicapped people who find it difficult to drive a two-wheeler vehicle. The available two Wheeler for them with two supporting wheels takes too much space for parking and driving on the road. This project presents an attempt on developing a scaled down version of an autonomous self-balancing robot, which takes less space and balances itself on just 2 wheels, incurring minimum cost. To be more precise, it is a two-wheeled platform required to balance vertically. The self-balancing robot can be used for an transportation. It is used in many industries such as inside

factory floors or for tourism in the park. It is more attractive compared to four or three wheeled vehicles as they can take sharp turns and navigate in congested spaces.

VIII. APPLICATIONS

- **Segway Robot:** Two- wheeled self balancing scooter is used nowadays for security purposes.
- **An intelligent gardener:** In agricultural fields, it can ease the work of farmers and thus increase the productivity.
- **An autonomous trolley:** Used in hospitals, shopping malls for better efficiency.
- **Two-wheel self-balancing vehicle** that has seen deployment for law-enforcement, tourism.
- **Self balancing wheelchair**
- **Handicapped people:** Two-wheeled self balanced vehicle for them as it takes more space on roads and in parking area
- **Battery-run vehicle :** The current day short distance vehicles can be substituted by them and thus fuel can be conserved for other purposes.

IX. PYTHON LIBRARIES USED

- MPU6050
- Time
- PIDcontroller (as an algorithm script)

X. CONCLUSION

We successfully modeled and demonstrated a self balancing robot with control feedback loop system. It is able to balance smoothly till a pitch angle around 20 degrees. It was a great learning experience for all of us. However there are some limitations.
Currently, such system is suitable only for flat ground. In order to make it work on slant surface, the angle of slope needs to be fed into the system, either manually or using some intelligence.
This is just a prototype of a fully balanced system which can be improvised further to increase its utility in terms of various aspects. Since our motive was to help the physically disabled, keeping that in mind, we can install a camera module in it, so that the robot can act as a guide for the blind people.

XI. ACKNOWLEDGEMENT

We would like to express a big thanks to Dr.Hitesh Shrimali, Assistant Professor IIT Mandi, without whose expert guidance and suggestions, this project would never have been possible. Also, we would like to thank our seniors, Jitesh Kumar Gupta (4th year, EE), Pulkit Rajgadiya(4th year, EE), V Sai Subba Rao(3rd year, ME), Sanidhya Aggarwal(3rd year, CSE) and Suryavanshi Virendrasingh(2nd year, CSE) who helped us a lot as and when we asked them despite having a busy schedule.

REFERENCES

[1] **Figure 1-Raspberry-pi**
https://uk.rs-online.com/web/p/processor-microcontroller-development\-kits/8111284/

[2] **Figure 2-Motor Driver**
https://www.amazon.in/REES52-Driver-Controller-Stepper-Arduino/dp/B0197PNK9Q

[3] **Figure 3-MPU-6050**
https://diygeeks.org/wp-content/uploads/\2016/08/mpu6050.jpg

[4] **Figure 5-Tilt Angle**
https://kth.diva-portal.org/smash/get/diva2:916184/FULLTEXT01.pdf

[5] **Raspberry pi theory**
https://www.raspberrypi.org/forums/viewtopic.php?t=102173\

[6] **Installing Raspbian OS On SD Card**
https://www.youtube.com/watch?v=B5wkXu6tmb4

[7] **Balancing Theory**
http://ozzmaker.com/success-with-a-balancing-robot-using-a-raspberry-pi/

[8] **Circuit Connections**
http://www.instructables.com/id/Building-a-segway-in-Raspberry-Pi/