ETE 320 Project Report

Self-Balancing Robot Using Microcontroller

Rakib Hossain

1808039

May 13, 2023

Abstract

This paper discusses the design, construction, and control of a two-wheel study level, small self-balancing robot. Two-wheel robots are compact and require less number of motors and other equipment in comparison to conventional four-wheel robots. Consequently, they offer a complex control system problem of balancing on two-wheel. The system architecture comprises of Arduino Uno microcontroller board, DC motor, gyroscope and accelerometer sensor. PID controller controls it through the program written in Arduino IDE. A detailed description of constituting materials in making the robot has been provided. A step-by-step process of connecting the hardware components has been listed, and finally, hardware assembly is shown.

1 Introduction

To make a robot that can balance itself on two wheels. There will be only one axle connecting the two wheels and a platform will be mounted on that. There will be another platform above it. The platform will not remain stable itself. Our job will be to balance the platform using Gyroscope sensors and to maintain it horizontal. At first we have decided to just balance the robot on its two wheels. Basically, a two-wheel self-balancing robot is very similar to the inverted pendulum, and which is an important test part in control. A Two-wheeled self-balancing vehicle is commercially known as "Segway".And also the Segway can never stay upright. Besides the development of Segway, studies of two-wheel self-balancing robots have been widely reported. Arduino is an open prototyping platform based up on Atmega processors. And it will be a fast becoming popular platform for both education and product development, with applications ranging from robotics, to process control.In this paper, I report a project on the basis of design, construction and control of a two-wheel selfbalancing robot with Ardunio software. The robot is driven by two DC motors, and is equipped with an Arduino Uno board which is based on the ATmega328 microprocessor, and also included with a MPU-6050 for attitude determination. For two wheel control designs are based on proportional-integral-differential (PID) control.

2 Component & Analysis

2.1 Arduino Uno

An Arduino Microcontroller is a multipurpose tool that can be used for many electronics projects as it provides many functions to a particular circuit. Is used in this Project as the brain of the robot. It is based on the ATmega328 microcontroller, which is capable for several projects. For our project we use it to control the vehicle's Motor Driver and MPU-6050 using Arduino code. we wrote the code to get data from MPU-6050 and control the Motor Driver.

Figure 1: Arduino Uno

2.2 Accelerometer Gyroscope (MPU-6050)

The MPU-6050 is a motion-tracking device. Here, this sensor gets the alignment of the bot, which is further processed to take control actions. The MPU-6050 is a 3-axis accelerometer. It captures the x, y, and z-axis values. But we worked with only one axis. When Bot is unbalanced MPU-6050 delivers a signal of x-axis to Arduino and Arduino takes operation according to this.

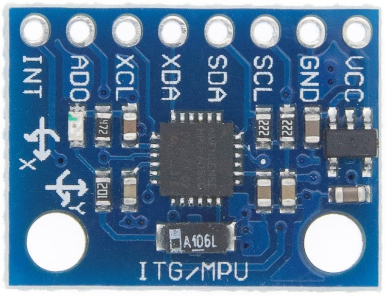

Figure 2: Accelerometer Gyroscope (MPU-6050)

2.3 Motor Driver (L298N)

A Motor Driver that can control 2 motors. It has 12v and 5v input. It has also 6 input pins including two enable pin and four control pins, These 6 pins receive signals from Arduino and drive the motors.

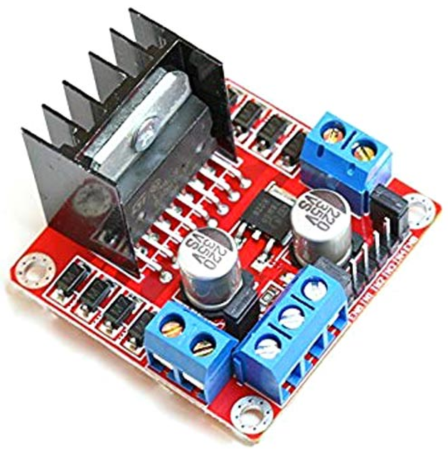

Figure 3: Motor Driver (L298N)

2.4 Motors And Wheels

In electronics, a motor is an electrical machine that converts electrical energy into mechanical energy. For our project, the motors had to be controlled so that their duty cycles could be changed and their direction monitored based on the input from the MPU-6050 sensor. MPU-6050 sensor detects unbalanced it transfer data to Arduino and Arduino control motor via motor driver.

Figure 4: Motor and Wheel

2.5 12v DC Battery

A Battery that provides power to Arduino, Motor Driver, Motors and MPU-6050 sensor. This 12v battery power is directly connected to Motor Driver and 5v output from Motor Driver goes to Arduino Uno.

eXTREME
2200mAh
Powered radio controlled models
High Discharge Lithium Polymer Battery
7.4V 11.1V 14.8V
8C 30C 40C 60C

Figure 5: 12v DC Battery

3 Design Consideration

The primary objective of the vehicle is to balance it with only two wheels. To fulfill this we used one MPU-6050 sensor, one Arduino, one motor driver, two motors, and two wheels. The MPU-6050 sensor observes the sensitivity of the x-axis from the robot's condition. As a result, it is able to keep standing.

4 Circuit Schematics & Explanation

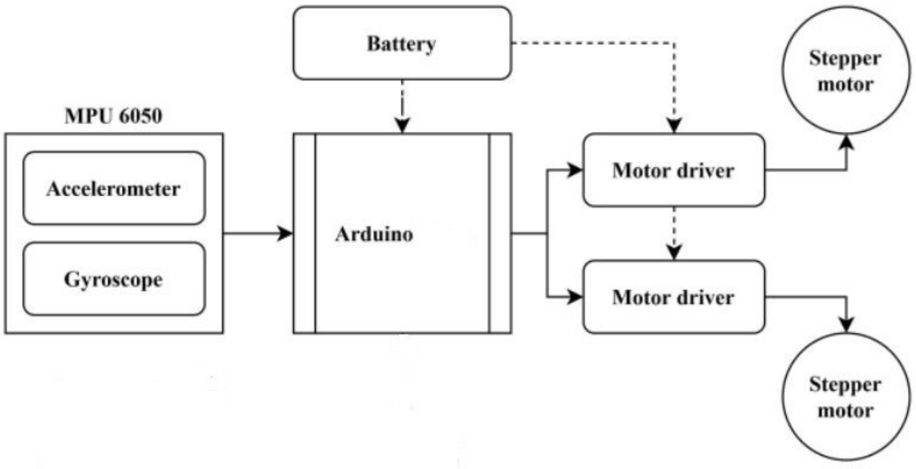

Figure 6: Block diagram of the self-balancing robot

In the circuit, the MPU-6050 sensor receives a signal from the bot's condition. Then it transmits this signal to Arduino. Arduino process this data and according to Arduino code, it transmit the data to Motor Driver to drive the motors. This process repeats again and again.

5 Code

```
#include "I2Cdev.h"
#include <PID_v1.h>
#include "MPU6050_6Axis_MotionApps20.h"
MPU6050 mpu;

bool dmpReady = false;
uint8_t mpuIntStatus;
uint8_t devStatus;
uint16_t packetSize;
uint16_t fifoCount;
uint8_t fifoBuffer[64];

Quaternion q;
VectorFloat gravity;

float ypr[3];
double setpoint= 176;
double Kp = 21;
double Kd = 0.8;
double Ki = 140;
double input, output;

PID pid(&input,&output,&setpoint,Kp,Ki,Kd,DIRECT);

volatile bool mpuInterrupt = false;

void dmpDataReady(){
    mpuInterrupt = true;
}

void setup() {
  mpu.initialize();
  devStatus = mpu.dmpInitialize();
  mpu.setXGyroOffset(220);
  mpu.setYGyroOffset(76);
  mpu.setZGyroOffset(-85);
  mpu.setZAccelOffset(1688);

  if (devStatus == 0) {
    mpu.setDMPEnabled(true);
    attachInterrupt(0, dmpDataReady, RISING);
    mpuIntStatus = mpu.getIntStatus();
    dmpReady = true;
    packetSize = mpu.dmpGetFIFOPacketSize();
    pid.SetMode(AUTOMATIC);
    pid.SetSampleTime(10);
    pid.SetOutputLimits(-255, 255);
  }
  pinMode (6, OUTPUT);
  pinMode (9, OUTPUT);
  pinMode (10, OUTPUT);
  pinMode (11, OUTPUT);
  analogWrite(6,LOW);
  analogWrite(9,LOW);
  analogWrite(10,LOW);
  analogWrite(11,LOW);
}

void loop() {
  if (!dmpReady) return;
  while (!mpuInterrupt && fifoCount < packetSize){
    pid.Compute();
    if (input>150 && input<200){
    if (output>0){
      Forward();
    }
    else if (output<0){
      Reverse();
    }
    Stop();
  }
  mpuInterrupt = false;
  mpuIntStatus = mpu.getIntStatus();
  fifoCount = mpu.getFIFOCount();
  if ((mpuIntStatus&0x10) || fifoCount==1024){
      mpu.resetFIFO();
  }
  else if (mpuIntStatus & 0x02){
    while(fifoCount<packetSize){
      fifoCount = mpu.getFIFOCount();
    }
    mpu.getFIFOBytes(fifoBuffer, packetSize);
    fifoCount -= packetSize;
    mpu.dmpGetQuaternion(&q, fifoBuffer);
    mpu.dmpGetGravity(&gravity,&q);
    mpu.dmpGetYawPitchRoll(ypr,&q,&gravity);
    input = ypr[1] * 180/M_PI + 180;
  }
}
void Forward(){
  analogWrite(6,output);
  analogWrite(9,0);
  analogWrite(10,output);
  analogWrite(11,0);
}
void Reverse(){
  analogWrite(6,0);
  analogWrite(9,output*-1);
  analogWrite(10,0);
  analogWrite(11,output*-1);
}
void Stop(){
  analogWrite(6,0);
  analogWrite(9,0);
  analogWrite(10,0);
  analogWrite(11,0);
}
```

6 Conclusion

This project presents an Arduino based, low cost self-balancing robot as an educational control system. And this robot was designed and established as desired with the limited resources possible. It is able to balance on the surface because the PID controller was successfully implemented on the robot. The implemented PID controller was also able to handle a varying amount of load on the structure. It could also handle disturbances when someone was picking load from the structure.

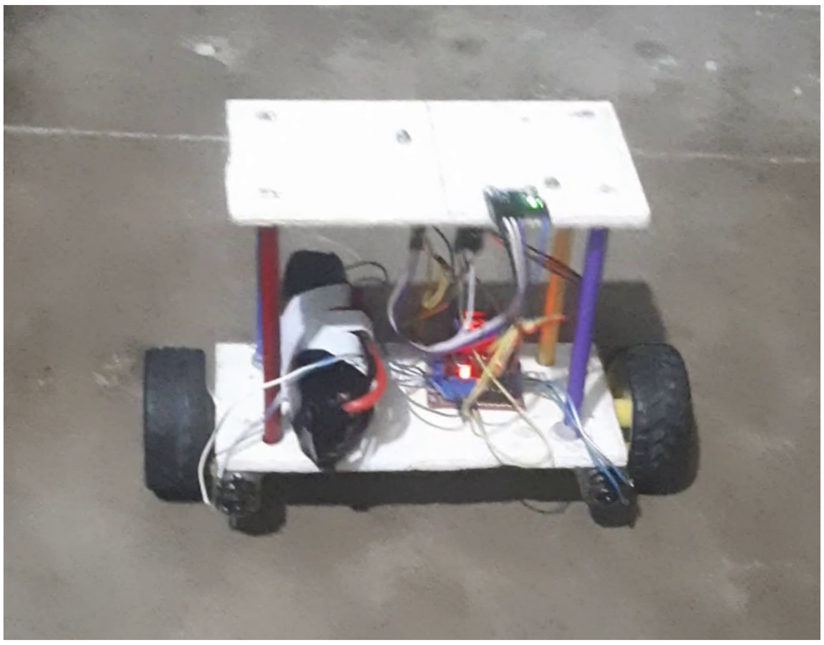

Figure 7: Photos of Final Lab Project

7 References

1 MPU6050 - Interface MPU6050 Accelerometer and Gyroscope Sensor
`https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/`

2 L298N - Interface L298N DC Motor Driver Module
`https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutor`

3 Arduino Uno - Arduino Guide
`https://docs.arduino.cc/hardware/uno-rev3`

4 DIY Self Balancing Robot using Arduino
`https://circuitdigest.com/microcontroller-projects/arduino-based-self`