20220802253, Omi Singh

Self Balancing Bot Journal.

I am in charge of the code. The bot basically reads sensor values, calculates the response on the motors using PID control, and emits the response to the stepper motors.

https://github.com/OmiSingh04/design_proj/tree/master/self_balancing_robot

I am mostly done with the code.

PID – Input – Angle

Output – control signal. (I scale it with another variable, to decide its weight to determine the frequency at which to rotate the bot.)

I should probably limit the variables to certain extremes.

Code has 4 parts to it.

Sending debug info via Bluetooth. I think im gonna create a python application to collect this data so I can save it in files. Debug info is probably going to be very useful. Ill have to see. I have maintained the code neatly, by splitting the code into various headers and translation units. It does compile.

2nd October, 2023

I am now nervous, thinking my PID system is very naive. I am relying solely on accelerometer values. Perhaps I should have a look at angular velocity values as well, to calculate my angle. I think I will use this day to create a test.

I will make 2 Arduino sketches. One where I use accelerometer-calculated angles. One where I use gyroscope-calculated angles. See which is better.

Based on the results I may –

i) A simple weighted sum
ii) use the so-called 'Kalman Filter' to combine these values

based on certain weights to maximize accuracy.

I have the MPU6050 already. Poor soldering but it works. Lets get testing.

OK. Gyroscope values are not at all reliable. Even when the chip is at complete rest it generates small outputs, which will overtime really add up when calculating the current angle.

On the other hand, using accelerometer values is the most reliable way to calculate angle.

Now, I wonder how my PID can be made better here. I no longer know. I just have to see how it performs.

I will probably work on debug information, see which variables I should limit and call it a day. I have assignments too.

As a last minute change I might shift from Adafruit_MPU6050 to MPU6050 by ElectronicCats.

Im getting much more accurate angle readings through accelerometer from there. When the chip is almost upright in either direction, im getting up 89 degrees in both sides. This seems more accurate than what im using right now.

Since sudden movements will distort the acceleration values, I need movement RPM to be small.

We will see. The main focus right now is balancing.

I need to write better debug info. I doubt assembling will be done even by tomorrow. Ill have time to make some updates throughout this week. We are 1 team member short. But its okay. I always felt like the team had only 3 members in the first place.

3rd October, 2023

I learned that calibrating the MPU6050 is probably a good choice. It removes… or minimizes zero error. That zero error is the exact reason why I have been getting bad values from the gyro.

Now I have the right offsets which will make my readings accurate. Ill try again, using angular velocity values to find angle.

And while I'm at it, I think I should search for a library that will calculate it for me.

Its tough. Ill need external libraries that implement that part for me. Mid Term is coming.

7th October, 2023

Its too late to work on this for now. Ill put it to after Mid Term – 2.

As star points on what I should be working on –

- Pid Tuning Interface.
- Kalman Filter library to combine gyro + accel to get better angle values. (since stage 2 involves moving the robo)
- Pid library (since mine is probably too basic)

Abstraction is epic.

13th October, 2023.

The mid term 2 is finished. Im back to working on the thing. The above three points are to be worked on.

My priority should be the Kalman filter. I can test value stability at home itself. I have the chip, the microcontroller.

So by priority –

1) Kalman Filter for getting filtered angle values
2) PID library
3) Interface

Lets search for some good Kalman filter library.

STUFF IS FROM LIKE 9 YEARS AGO.

15th October, 2023

I am finally done with the code. I settled with – a complementary filter. No Kalman. There is a PID library which I am using to generate control.

My guess is that I do not need integral term for the PID. I need to tune P and D terms only. I will be 0.

But I cannot say anything for sure until we actually try it out.

At the very least, the behaviour of PID is as predicted. As the offset increases, it generates a higher control signal. And it also generates it in the opposite direction.

The sad news is that im too tired right now to work on the interface.

There is no happy news.

Maybe ill work on the Bluetooth communication thing by tonight. Lets see.