ICMIEE18-266

Self-Balancing Autonomous Unicycle using Raspberry Pi

Pulok Tarafder[1,*], *Moyukh Amin*[1], *Syed Samiul Alam Mehrab*[2], *Abdullah Al Maruf*[1]
[1]Department of Electrical and Electronic Engineering, BRAC University, 66 Mohakhali, Dhaka 1212, BANGLADESH
[2]Department of Computer Science Engineering, BRAC University,66 Mohakhali, Dhaka 1212, BANGLADESH

ABSTRACT
Traditionally mobile devices contain no balancing mechanism as they are already quasi static containing three or four wheels. Using gyroscopes to power human forwards and backwards only a single wheel is needed to make a vehicle which will be much smaller and lighter with less cost. Being an inverted pendulum design it is inherently unstable so it needs constant control to provide appropriate acceleration.Using Kalman filter algorithm on the gyroscopic data this problem can be solved. As the proposed model is automatically controlled it will help reduce our design cost bare bones. The model always ensures that the tilt factor is minimized. We were successfully able to implement balancing during movement and on lateral level during static state.

Keywords: Self balance, Unicycle, Kalman Filter, PID Control, Raspberry Pi.

1. Introduction

Imagine robots of the future cruising through space time with two or even a single wheel. Our effort is to seek a solution to this problem by taking on the challenge of balancing a unicycle in a single direction using the existing motors torque and forward-backward acceleration. An autonomous self-balancing vehicle i.e. a single wheel vertically stable transport is the future. Being an inverted pendulum design it is intrinsically not stable so it needs a steady control to maintain its movements. The single wheel explores the Linear-quadratic-Gaussian control problem.The solution can be found by using a series of Kalman filter algorithm on the gyroscopic data and then apply forward and backward acceleration to balance a load while moving in the rolling direction.The Self balancing unicycle uses raspberry pi to control the motor, process gyro and accelerator data and provide more features for further development. Using these methods we are able to balance it during its movement in the rolling direction.

1.1 Physical Model

The inspiration was Segway type vehicle but with a single wheel. As this is a classic inverted pendulum problem. The body needs a good balance and moment of inertia as not to lean and fall. The lateral balancing is done by measuring the tilt angle and setting and offset in the opposite direction and providing torque in that direction. There are four degrees of freedom for this vehicle as such balancing in the lateral direction is discussed in this paper. The concept of this type of PID control is quite popular and widely used in automated control systems. We get the current tilt and yaw, predict the next position and use this feedback to control the output. This process will prevent the robot from falling by providing acceleration in the wheels corresponding to its inclination from the neutral vertical. As the vehicle gets deviated by an angle, then in the frame of reference of the wheels and the center of mass will encounter a pseudo force and apply a torque opposite to the direction of tilt. The project needs precision vector calculations done in seconds. Moreover, over the years the sensors data has been seen to be populated with noise data. This is where fusion algorithms come which require huge processing power. As such the reasonable choice for our project was a processor of GHz power. So the raspberry pi zero was the obvious choice of embedded controller. This project includes the RPi Zero. The mechanical design problem is solved by keeping the center of gravity (COG) at a single vector over any motion which is described on another section. The motor choice needed to be high RPM, high Torque and only a geared DC motor provided the requirement. Our project implements a 12VDC 120W motor. We needed a suitable motor driver to control both speed and direction. The RPi motor driver satisfied all those requirements and also solved the problem of supplying power to the pi by regulating 12V to 5V using LM2596. An MPU6050 sensor which contains a MEMS accelerometer and a MEMS gyro in a single chip which contains 16-bit analog to digital conversion hardware for each channel was sufficient for our project and it has I2C bus that helps communicate with it in a relevant and fast way.

1.1.1 Raspberry Pi Specification

The project uses Debian OS in Raspberry Pi zero (Linux raspberry pi 4.9.59+ #1047 Sun Oct 29 11:47:10 GMT -

Fig.1 Raspberry Pi Zero

* Pulok Tarafder. Tel.: +88-01682510085
E-mail address: pulok.tarafder@g.bracu.ac.bd

2017 armv6l GNU/Linux). The code is done in Python3.6 and the dependencies are all under GNU. This single core 1.2GHz device with 512MB of RAM including unpopulated 40-pin GPIO connector is capable of applying this complex fusion of Kalman-filter Algorithm and predicts state almost at real-time. The physical look of this Raspberry Pi Zero is shown in Fig.1.

1.1.2 Motor Driving Principle

The main purpose of using a DC motor is to transfer electrical energy into mechanical energy. Whenever a current conductor is placed in magnetic field, it experiences a mechanical force. Although DC motors have low efficiency in our case it serves the best as we don't need servo mechanism to guess the wheel position they are inefficient and our gyroscope and accelerometer data acts as feedback. The RPi motor driver fits as overhead on the pi zero and also provides power. We don't need connecting anything separately. Although the motor is controlled through 3 pins. Table 1 states that the Pulse Width Modulated signal on pin 26 controls the speed and torque of the motor. Making M1 High and M2 Low moves the motor forward and on the other hand M1 Low and M2 High move the motor backward.

Table 1 Controlling Movements usingMotor

Interface	WiringPi	BCM
M1	**P28**	**20**
M2	**P29**	**21**
PWMA	**P25**	**26**

Fig.2RPi Motor Driver

1.1.3 Interfacing with MPU6050

The MPU-6050 devices combine a 3-axis gyroscope and a 3-axis accelerometer on the same silicon die, together with an onboard Digital Motion Processor™ (DMP™), which processes complex 6-axis Motion Fusion algorithms. The device can access external magnetometers or other sensors through an auxiliary master I²C bus, allowing the devices to gather a full set of sensor data without intervention from the system processor. The devices are offered in a 4 mm x 4 mm x 0.9 mm QFN package. The Inter-integrated Circuit (I2C) Protocol is a protocol intended to allow multiple "slave" digital integrated circuits ("chips") to communicate with one or more "master" chips. Like the Serial Peripheral Interface (SPI), it is only intended for short distance communications within a single device. Like Asynchronous Serial Interfaces (such as RS-232 or UARTs), it only requires two signal wires to exchange information. Messages are broken up into two types of frame: an address frame, where the master indicates the slave to which the message is being sent, and one or more data frames, which are 8-bit data messages passed from master to slave or vice versa. Data is placed on the SDA line after SCL goes low, and is sampled after the SCL line goes high. The time between clock edge and data read/write is defined by the devices on the bus and will vary from chip to chip. The RPi reads the specific addresses and collects its data.

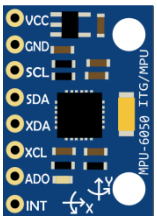

Fig.3MPU-6050 6-Axis MEMS Sensos

1.2 Mechanical Structure

The seat is sourced from a second hand bicycle. The shape of the seat is important as you need to be able to shift your center of gravity forward and backward on the seat. We attached it to the frame with a simple steel pipe. The wheel, sprocket axle all were welded in to the frame. We used 1x1 inch tubing for the frame and the foot pegs. A couple of plates across the middle of the frame served to mount the single motor with the frame to which we glued or screwed the electronics. The axle was set in a channel which allows the two sprockets of the motor and wheel to be moved so the chain can be easily fitted. A bolt above the axle stops it from suddenly upwards when in use. The whole package came to 30 x 7 inch excluding the 3 inch foot pegs. The Mechanical diagram is given in Fig.4.

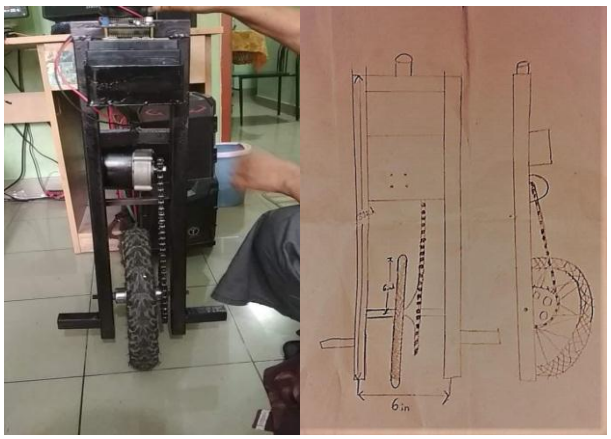

Fig.4 Mechanical Diagram and Physical Product

1.3 Angle Measurement

To measure the angle of tilt we have the data from two sensors, accelerometer and gyroscope. The accelerometer measures the force with respect to gravity, thus we calculate angle of the vehicle. The accelerometer measures angle using the Eq. (1).

$$\theta = arctan(Ay / sqrt(Ax^2 + Az^2)) \quad (1)$$

The problem with this type of sensor is that it accounts for the rest of the forces the vehicle is experiencing i.e., it has a lot of error and noise. The gyroscope can measure the angular velocity by integrating the signal over time by using Eq. (2).

$$\theta_i = \theta_{i-1} + w * dt \quad (2)$$

but the problem is measurement is that is not perfect and has a deviation, such that in short run times the measure is valid, but for long time runs the value will start to deviate much form the real angle. To avoid this situation we integrate these measurements and can accurately obtain the angle of the vehicle at its current position in any instantaneous time. To combine of both sensors, which is called sensor fusion, we use Kalman Filter in this application, as it is an algorithm that gives an accurate data with low computation costs.

1.4 Kalman Filter

Kalman filtering also known as Linear Quadratic Estimation (LQE) is a widely used algorithm used to model data accurately in a series of measurements observed over time containing statistical noise and other inaccuracies. It finds the most optimum averaging factor for each consequent state. Also somehow remembers a little bit about the past states. Here the equation below is a basic model of Kalman filter. The Kalman gain eliminates the noise by applying co-variance principle between measured data and previous estimation and iterates the Kalman gain to be accurate over time which in turn accurately predicts future states. It treats each of the data in discrete time interval and calculates Kalman gain at each consequent state getting smarter gain each iteration using Eq. (3).

$$X_k = K_k . Z_k + (1 - K_k) . X_{k-1} \quad (3)$$

Fig.5 Sensor Data(RG) and Filter Data(B)

Fig.5 shows how Kalman filter works on the data. The red line shows how much noisy the acceleration data is. And the drift is clearly visible in the Gyroscope readings. The green line shows simple complimentary filter output while Kalman filter output is visible in blue. The blue data is quite accurate and noise less. As such Kalman filter algorithm is fitted to our data by iterating through gyroscope and accelerometer readings.

1.5 Mathematical Model

1. If the tilt angle is to the right, the vehicle must decelerate to the left and vice versa.

2. The position of the vehicle relative to track center is stabilized by slightly modulating the null angle (the angle error that the control system tries to null) by the position of the vehicle, that is, (null angle =tilt angle +k * position) where k is small. This makes the pole want to lean slightly toward track center and stabilize at track center where the tilt angle is exactly vertical. Any offset in the tilt sensor or track slope that would otherwise cause instability translates into a stable position offset. A further added offset gives position control.

3. A normal pendulum subject to a moving pivot point such as a load lifted by a crane, has a peaked response at the pendulum radian frequency of omega=sqrt {g / ell }. To prevent uncontrolled swinging, the frequency spectrum of the pivot motion should be suppressed near omega. The inverted pendulum requires the same suppression filter to achieve stability.

1.6 PID Control of the Motor

The PID controller is widely employed because it is very understandable and because it is quite effective. One attraction of the PID controller is that all engineers understand conceptually differentiation and integration, so they can implement the control system even without a deep understanding of control theory. Further, even though the compensator is simple, it is quite sophisticated in that it captures the history of the system (through integration) and anticipates the future behavior of the system (through differentiation). The output of a PID controller, which is equal to the control input to the plant, is calculated in the time domain from the feedback error as follows in Eq. (4).

$$u(t) = Kp * e(t) + Ki \int e(t)dt + Kd \frac{de(t)}{dt} \quad (4)$$

Increasing the proportional gain (K_p) has the effect of proportionally increasing the control signal for the same level of error. The fact that the controller will "push" harder for a given level of error tends to cause the closed-loop system to react more quickly, but also to overshoot more. Another effect of increasing K_p is that it tends to reduce, but not eliminate, the steady-state error.

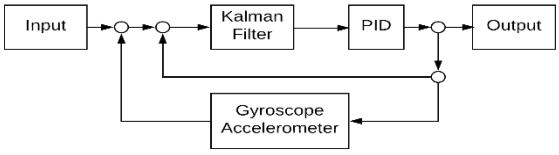

Fig.6 Control Diagram

The addition of a derivative term to the controller (K_d) adds the ability of the controller to "anticipate" error. With simple proportional control, if K_p is fixed, the only way that the control will increase is if the error increases. With derivative control, the control signal can become large if the error begins sloping upward, even while the magnitude of the error is still relatively small. This anticipation tends to add damping to the system, thereby decreasing overshoot. The addition of a derivative term, however, has no effect on the steady-state error.

The addition of an integral term to the controller (K_i) tends to help reduce steady-state error. If there is a persistent, steady error, the integrator builds and builds, thereby increasing the control signal and driving the error down. A drawback of the integral term, however, is that it can make the system more sluggish since when the error signal changes sign, it may take a while for the integrator to "unwind."

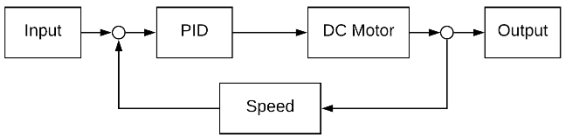

Fig.7 Block diagram of feedback mechanism of PID controller.

1.7 Computation

The program starts by adjusting the optimum position for the first 5 iterations. Then reading and adjusting motor offset. The motor then keeps adjusting the speed of acceleration and deceleration to get the optimum torque and adjust tilt angle. The block diagram of Fig 8 shows the computation process.

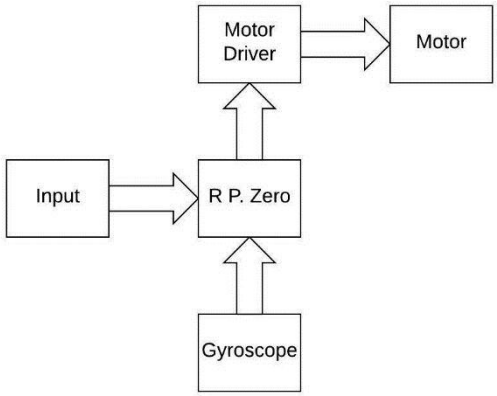

Fig.8 Computation process of the RPi Zero

2. Conclusion

In this paper, we proposed a model for solving the inverted pendulum problem on a single point of inertia. Using Kalman filter over gyroscope and accelerometer data to provide an optimum motor offset that the motor driver adjusts while correcting the Kalman gain over time. We designed and implemented this model considering factors like low cost, reliability and automatic control. As the proposed model is automatically controlled it will help reduce our design cost to bare bones reducing both motor, gear and wheel costs. The model always ensures that the tilt factor is minimized. We were successfully able to implement balancing during movement and on lateral level during static state. We hope that this control system will be perfected and everything will be single wheeled in the near future.

NOMENCLATURE

X_k= Current estimation
Z_k= Measured value
K_k= Kalman gain
X_{k-1}= Previous estimation
K_p= Proportional gain
K_i= Controller integral term
K_d= Differential term
$u(t)$ = Output of the PID controller

REFERENCES

[1] M. Baloh& M. Parent, Modeling and model verification of an intelligent self-balancing two-wheeled vehicle for an autonomous urban transportation system, *The conference on computational intelligence robotics andautonomous systems* pp. 1-7 2003.

[2] C. -N. Huang, The development of self-balancing controller for one-wheeled vehicles; *Engineering vol. 2 no.4 2010*.

[3] J. Juan Rincon Pasaye, J. Alberto Bonales Valencia & F. Jimenez Perez Tilt measurement based on an accelerometer a gyro and a Kalman filter to control a self-balancing vehicle; *Power Electronics and Computing (ROPEC) 2013 IEEE International Autumn Meeting* on pp. 1-5 2013.

[4] B. Mahler & J. Haase, Mathematical model and control strategy of a two-wheeled self-balancing robot"; *Industrial Electronics Society IECON 2013–39th Annual Conference of the IEEE* pp. 4198-4203 2013.

[5] A. M. Mohtasib& M. H. Shawar, Self-balancing two-wheel electric vehicle (steve);*Mechatronics and its Applications (ISMA) 2013 9th International Symposium* on pp. 1-8 2013.

[6] C. Sun, T. Lu & K. Yuan Balance control of two-wheeled self-balancing robot based on linear quadratic regulator and neural network; *Intelligent Control and Information Processing (ICICIP)*

2013 Fourth International Conference on pp. 862-867 2013.

[7] L. Sun J. Gan Researching of two-wheeled self-balancing robot base on lqr combined with pid;*Intelligent Systems and Applications (ISA) 2010 2nd International Workshop* on pp. 1-5 2010.

[8] R. Xiaogang, L. Jiang, D. Haijiang& L. XinYuan Design and lqr control of a two-wheeled self-balancing robot; *Control Conference 2008. CCC 2008. 27th Chinese* pp. 275-279 2008.

[9] Q. Yong, L. Yanlong, Z. Xizhe et al. Balance control of two-wheeled self-balancing mobile robot based on tsfuzzy model; S*trategic Technology (IFOST) 2011 6th International Forum* on vol. 1 pp. 406-409 2011.

[10] N. Yu, Y. Li, X. Ruan& C. Wang Research on attitude estimation of small self-balancing two-wheeled robot; *Control Conference (CCC) 2013 32nd Chinese* pp. 5872-5876 2013.