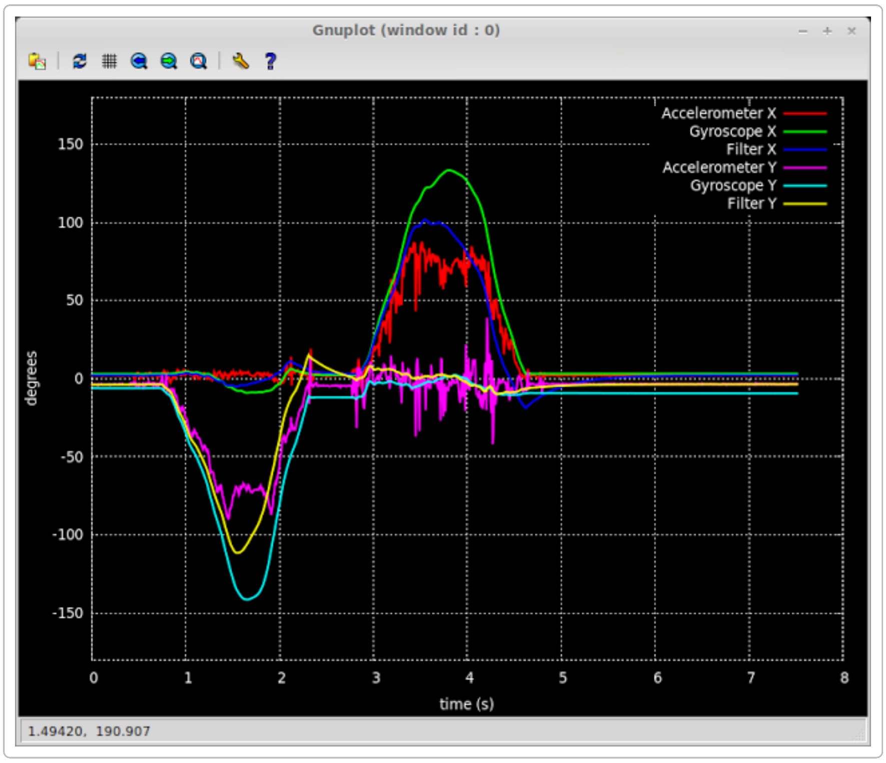
Bitify

Tinkering with the Raspberry Pi and other geeky stuff

Saturday, 16 November 2013

Using a complementary filter to combine Accelerometer and Gyroscopic data

This post shows how to combine data from the accelerometer and gyroscope using a complementary filter to produce a better readings from the MPU-6050.



Complementary filter

The image above shows data for a negative rotation around the Y axis followed by a positive rotation around the X axis. It includes the base accelerometer, gyroscope and the filtered data. Lets look at things in a bit more detail.

The following graph show a simple rotation in X of roughly 90-100 degrees (I didn't measure it accurately). The red line shows the accelerometer data and as we can see from the spikes it's a noisy data set. The green line show the rotation angle calculated from summing the individual angles read from the gyroscope. While this data is less noisy it is prone to drift over time, the gyroscope doesn't return back to zero when not moving.

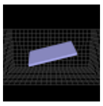
Search

Search

Popular Posts

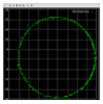
Interfacing Raspberry Pi and MPU-6050

I wanted to interface my Pi to a Six-Axis Gyro + Accelerometer sensor and the one I settled on was based on a MPU-6050 chip. I went for thi...



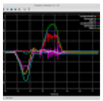
Reading data from the MPU-6050 on the Raspberry Pi

In a previous post I showed how to connect an Accelerometer & Gyro sensor to the Raspberry Pi, in this post I'll show some simple P...



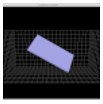
Connecting and calibrating a HMC5883L Compass on the Raspberry Pi

Here is how to connect a HMC5883L Compass to the Raspberry Pi, calibrate it and read the data. Connecting the compass is simple enough, fo...



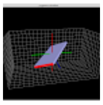
Using a complementary filter to combine Accelerometer and Gyroscopic data

This post shows how to combine data from the accelerometer and gyroscope using a complementary filter to produce a better readings from the...



3D OpenGL visualisation of the data from an MPU-6050 connected to a Raspberry Pi

In this post I'll show how to serve the data over http and display a 3D representation in OpenGL extending on a previous blog post det...



Pitch, Roll and Yaw using MPU6050 & HMC5883L (with tilt compensation and complementary filter)

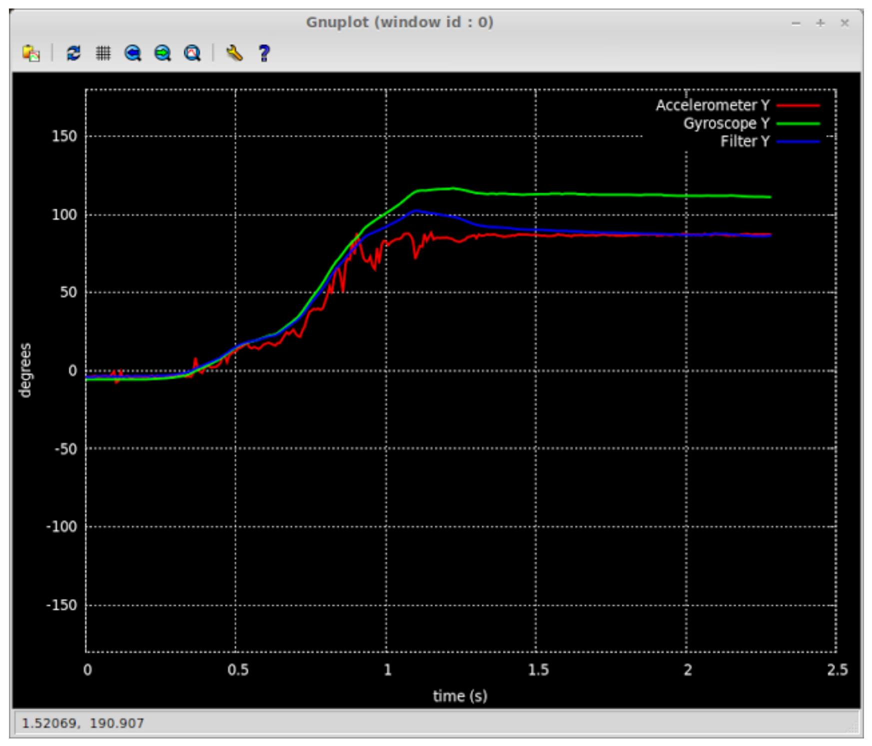
Combining the data from an MPU605 and a HMC5883L to give tilt compensated pitch, roll and yaw. Pitch, roll and yaw (with tilt compensati...

GY80 (L3G4200D, ADXL345, HMC5883L, BMP085) Python library for Raspberry Pi

A while back I bought a GY80 board, which comprises of: L3G4200D - Three axis Gyroscope ADXL345 - Three axis accelerometer HMC5883L - C...

Temperature logging with a DS18B20 and a Raspberry Pi

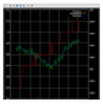
I wanted to do some temperature logging so I hooked up a DS18B20 temperature sensor to a Raspberry Pi. About the DS18B20 Dallas DS18B...



The blue line shows the complementary filter at work. It combines the two data sets by merging fast rotations from the gyroscope with the slower trends from the accelerometer and we get the best of both worlds. For a full explanation of the theory behind this type of filter I recommend reading this excellent paper. If you just want some simple code then read on.

The interesting parts are lines 76 to 91.

```
01 #!/usr/bin/python
02 
03 import smbus
04 import math
05 import time
06 
07 # Power management registers
08 power_mgmt_1 = 0x6b
09 power_mgmt_2 = 0x6c
10 
11 gyro_scale = 131.0
12 accel_scale = 16384.0
13 
14 address = 0x68  # This is the address value read via the i2cdetect command
15 
16 def read_all():
17     raw_gyro_data = bus.read_i2c_block_data(address, 0x43, 6)
18     raw_accel_data = bus.read_i2c_block_data(address, 0x3b, 6)
19 
20     gyro_scaled_x = twos_compliment((raw_gyro_data[0] << 8) + raw_gyro_data[1]) / gyro_scale
21     gyro_scaled_y = twos_compliment((raw_gyro_data[2] << 8) + raw_gyro_data[3]) / gyro_scale
22     gyro_scaled_z = twos_compliment((raw_gyro_data[4] << 8) + raw_gyro_data[5]) / gyro_scale
23 
24     accel_scaled_x = twos_compliment((raw_accel_data[0] << 8) + raw_accel_data[1]) / accel_scale
25     accel_scaled_y = twos_compliment((raw_accel_data[2] << 8) + raw_accel_data[3]) / accel_scale
26     accel_scaled_z = twos_compliment((raw_accel_data[4] << 8) + raw_accel_data[5]) / accel_scale
27 
28     return (gyro_scaled_x, gyro_scaled_y, gyro_scaled_z, accel_scaled_x, accel_scaled_y, accel_scaled_z)
29 
30 def twos_compliment(val):
31     if (val >= 0x8000):
32         return -((65535 - val) + 1)
33     else:
34         return val
35 
36 def dist(a, b):
37     return math.sqrt((a * a) + (b * b))
38 
39 
40 def get_y_rotation(x,y,z):
41     radians = math.atan2(x, dist(y,z))
42     return -math.degrees(radians)
43 
44 def get_x_rotation(x,y,z):
45     radians = math.atan2(y, dist(x,z))
46     return math.degrees(radians)
47 
48 bus = smbus.SMBus(0)  # or bus = smbus.SMBus(1) for Revision 2 boards
49 
```



Interfacing a BMP085 Digital Pressure sensor to the Raspberry Pi

I recently bought a sensor with a BMP085 Digital Pressure sensor on it so I thought I'd write a post on how to read the data from the R...

Labels

1-wire (1)
ADXL345 (1)
BMP085 (2)
DS18B20 (1)
gnuplot (3)
GY80 (1)
HMC5883L (3)
L3G4200D (1)
MPU-6050 (6)
OpenGL (2)
Python (8)
Raspberry Pi (8)
Raspbian (4)
temperature (1)

Blog Archive

About Me

Andrew Birkett

View my complete profile

```
# Now wake the 6050 up as it starts in sleep mode
bus.write_byte_data(address, power_mgmt_1, 0)

now = time.time()

K = 0.98
K1 = 1 - K

time_diff = 0.01

(gyro_scaled_x, gyro_scaled_y, gyro_scaled_z, accel_scaled_x, accel_scaled_y, accel_scaled_z) = read_all()

last_x = get_x_rotation(accel_scaled_x, accel_scaled_y, accel_scaled_z)
last_y = get_y_rotation(accel_scaled_x, accel_scaled_y, accel_scaled_z)

gyro_offset_x = gyro_scaled_x
gyro_offset_y = gyro_scaled_y

gyro_total_x = (last_x) - gyro_offset_x
gyro_total_y = (last_y) - gyro_offset_y

print "{0:.4f} {1:.2f} {2:.2f} {3:.2f} {4:.2f} {5:.2f} {6:.2f}".format( time.time() - now, (last_x), gyro_total_x, (last_x), (last_y), gyro_total_y, (last_y))

for i in range(0, int(3.0 / time_diff)):
    time.sleep(time_diff - 0.005)

    (gyro_scaled_x, gyro_scaled_y, gyro_scaled_z, accel_scaled_x, accel_scaled_y, accel_scaled_z) = read_all()

    gyro_scaled_x -= gyro_offset_x
    gyro_scaled_y -= gyro_offset_y

    gyro_x_delta = (gyro_scaled_x * time_diff)
    gyro_y_delta = (gyro_scaled_y * time_diff)

    gyro_total_x += gyro_x_delta
    gyro_total_y += gyro_y_delta

    rotation_x = get_x_rotation(accel_scaled_x, accel_scaled_y, accel_scaled_z)
    rotation_y = get_y_rotation(accel_scaled_x, accel_scaled_y, accel_scaled_z)

    last_x = K * (last_x + gyro_x_delta) + (K1 * rotation_x)
    last_y = K * (last_y + gyro_y_delta) + (K1 * rotation_y)

    print "{0:.4f} {1:.2f} {2:.2f} {3:.2f} {4:.2f} {5:.2f} {6:.2f}".format( time.time() - now, (rotation_x), (gyro_total_x), (last_x), (rotation_y), (gyro_total_y), (last_y))
```

First we read all the scaled data from the device.

```
(gyro_scaled_x, gyro_scaled_y, gyro_scaled_z, accel_scaled_x, accel_scaled_y, accel_scaled_z) = read_all()
```

Adjust the gyroscope data by the offset.

```
gyro_scaled_x -= gyro_offset_x
gyro_scaled_y -= gyro_offset_y
```

The offset is the value of the gyroscope reading when it's not moving and is taken from the very first reading. This is fine for simple testing but ideally a true offset value should be determined by calibrating the sensor.
Now calculate the gyroscope delta, this is how much the sensor has rotated since the last sample was taken and then add it to a running total

```
gyro_x_delta = (gyro_scaled_x * time_diff)
gyro_y_delta = (gyro_scaled_y * time_diff)

gyro_total_x += gyro_x_delta
gyro_total_y += gyro_y_delta
```

This gives us a rotation angle just from reading from the gyroscope (the green line in the graph above)
Next read the rotation values from the accelerometer just like we did in the previous post

```
rotation_x = get_x_rotation(accel_scaled_x, accel_scaled_y, accel_scaled_z)
rotation_y = get_y_rotation(accel_scaled_x, accel_scaled_y, accel_scaled_z)
```

Now the complementary filter is used to combine the data.

```
last_x = K * (last_x + gyro_x_delta) + (K1 * rotation_x)
last_y = K * (last_y + gyro_y_delta) + (K1 * rotation_y)
```

We take the previous readings (last_x, last_y) and add in the gyroscope data then scale this by K, then add in the accelerometer data scaled by K1 and this value is our new angle. The coefficients K and K1 should add up to 1, in this case they are 0.98 and 0.02 respectively. You can change the values of K and K1 to suit your application as described in the previously linked article. The time intervals for the loop needs to be reasonably accurate for this to work well and the sample rate should be 100Hz or higher.

If you run the code and direct the output to a file

```
sudo ./filter-test.py > plot.dat
```

you can then generate gnuplot diagrams similar to those above, save the following to a file gnuplot-command.plg

```
set terminal wxt persist size 800,600 background '#000000' # enhanced font 'Consolas,10'

set style line 99 linecolor rgb "#ffffff" linetype 0 linewidth 2
set key top right textcolor linestyle 99
```

```
set grid linestyle 99
set border linestyle 99

set xlabel "time (s)" textcolor linestyle 99
set ylabel "degrees" textcolor linestyle 99

set yrange [-180:180]

plot filename using 1:2 title "Accelerometer X" with line linewidth 2 , \
  filename using 1:3 title "Gyroscope X" with line linewidth 2 , \
  filename using 1:4 title "Filter X" with line linewidth 2

plot filename using 1:5 title "Accelerometer Y" with line linewidth 2 , \
  filename using 1:6 title "Gyroscope Y" with line linewidth 2 , \
  filename using 1:7 title "Filter Y" with line linewidth 2
```

then to generate a graph

```
gnuplot -e "filename='plot.dat'" gnuplot-command.plg
```

In the next post I show how to hook up a HMC5883L Compass module and incorporate it into the code so we can get a true bearing. Well that is when I get a new one as I seem to have fried mine.

Posted by Andrew Birkett at 23:11

Labels: gnuplot, MPU-6050, Python, Raspberry Pi

27 comments

Add a comment as Julien Louette

Top comments

James Smith 5 months ago - Shared publicly
awesome do you have the equivalent in C?

1 · Reply

View all 4 replies

Andrew Birkett 5 months ago
Have you installed the gnuplot application ? It isn't usually installed by default in Linux.

James Smith 5 months ago
Yes I did

anshul sanam 1 year ago - Shared publicly
Great tutorial, I am doing this on a Beaglebone Black and it works well, but is it possible to make the values smoother, because the the values seem to "jitter" a lot for me.

+1 1

Andrew Birkett via Google+ 2 years ago - Shared publicly
#RaspberryPi

1 · Reply

Lucas Leite 4 months ago - Shared publicly
Andrew, thank you.
I learned a lot by these series of articles on the MPU6050.
It was very insightful and taught me a lot on digital filters

1 · Reply

Doug Blanding 11 months ago - Shared publicly
You have done a really nice job on this series covering the MPU-6050. The one thing I discovered (that wasn't explained) was that I needed to install gnuplot-x11.

1 · Reply

Tim Rackers 1 year ago - Shared publicly

In regards to the theory behind the complementary filter, you included a link to find the paper that explains the process, "this excellent paper". However, the link isn't currently working and I was hoping that you could reply with a link to the paper.

Thanks,

1 · Reply

Andrew Birkett 1 year ago

Thanks for pointing out the link isn't working, I managed to track another copy down https://googledrive.com/host/0B0ZbiLZrqVa6Y2d3UjFVWDhNZms/filter.pdf I'll update the blog to link to this new version

Tim Rackers 1 year ago

Thanks!

---

bernardo jaccoud 8 months ago - Shared publicly

Great tutorial. I've used a lot of your code so let me thank you in advanced. However, I have a question, why did you use the range that you did in the "for" loop? Thank you.

1 · Reply

Andrew Birkett 8 months ago

It was a little bit arbitrary, i think I just found the values gave reasonable results for demonstration purposes.

Josse Pyfferoen 9 months ago - Shared publicly

Nice code!
I just don't get the twos_complement function.
What is the meaning of the 0x8000?

1 · Reply

View all 5 replies

Josse Pyfferoen 8 months ago

+Andrew Birkett
Thank you for answering!
(probably) last question: why do you put a - (line 42) in get y rotation?

Andrew Birkett 8 months ago

+Josse Pyfferoen To be honest I can't remember, I'm guessing the data was coming out with the signed flipped so I just flipped it back.

Andrew Huff 11 months ago - Shared publicly

Hi, thanks for this guide. I'm planning on using it soon but before I start, could you give me an idea as to how much of the CPU running this uses? It's just that the rpi will be doing other things while this is running and I'm worried that it will cause a lag in those tasks. Hopefully it's not CPU intensive. Thanks!

1

Kieran Cranley 1 year ago - Shared publicly

Hi Andrew - I'm a newbie, but I did manage to get the MPU-6050 working with your python programs. I haven't tried OpenGL visualisation on my Windows PC but it sounds like it isn't possible.

What I wanted to ask you, is - do you have any python code that would allow output from the 6050 to drive a pair of servos?

1 · Reply

View all 4 replies

Sparks N Smoke 1 year ago

Hi Andrew - I'm making some progress with servos etc, but in the meantime, I just wanted to ask you if the angles above which you are calculating for the accelerometer - get_x_rotation(x,dist(y,z)) and get_y_rotation(y, dist(x,z)) are angles between the vector R and the x and y axes, rather than the angles calculated for the gyro. Should they not be the exact same angles for inputting to the

Sparks N Smoke 1 year ago

+Andrew Birkett Me again Andrew - I have the servos going - you can see the post on my blog: http://smokespark.blogspot.co.uk/2015/01/61-exploring-pi-bs-ports-vii.html
Kieran

Roberto Montiel Camargo 11 months ago - Shared publicly

thanks for this great post, I'm trying to occupy the MPU6050 to measure the displacement of a car shock absorbers, how I can do to make me scroll instead of angular rotation?

1 · Reply

Newer Post | Home | Older Post

Subscribe to: Post Comments (Atom)