Assignment 4 Gyroscope and accelerometer sensing and filter

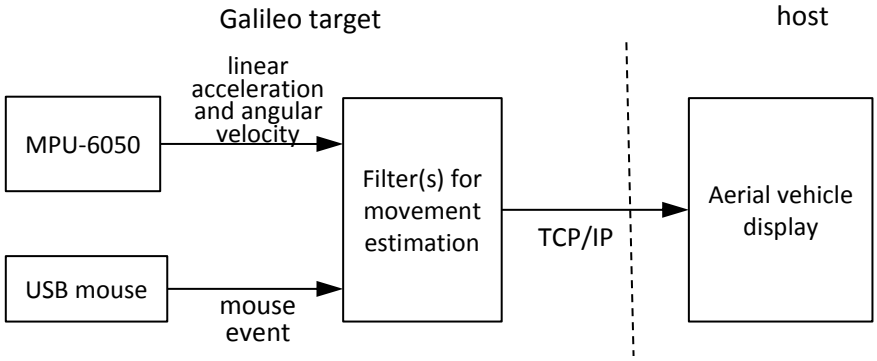
Assignment Objectives

1. To compute body movement via gyroscope and accelerometer sensors.
2. To learn input device driver in Linux
3. To apply filter design for state estimation.

Lab Assignment

It may be evitable to have noise in the signals collected from sensors. The noise could be caused by the accuracy of sensor devices, the white noise from the environment, or the sampling period variations. To deal with noisy measurement, we often apply statistical properties of the signal to derive the best guess of the true system state. A simple example is to take the weighted average of the measured values over a running window. This approach can result in an improvement of the signal-to-noise ratio. The second approach of dealing with noisy measurement is to use sensor fusion in which noisy measurements are collected from multiple sensors. The subsequent estimation based on the combined information can lead more accurate and complete view of the systems.

In this assignment, you are asked to develop an application program to measure the movement of a rigid body (e.g., an aerial vehicle) in 3-dimensional space. The rigid body is free to move in and rotate about three perpendicular axes. An accelerometer and gyroscope sensor is attached on the body. Sampling data is collected from the sensor and is processed via a signal processing module (filter). The position and orientation of the body is then sent to a display program to illustrate the movement. Giving that the display program, an executable built with Unity, is running in the host, packets should be sent via a stream socket to a specified port at the host.



When your program begins to run, the MPU-6050 accelerometer and gyroscope sensor module should be configured properly and initialized with a sampling rate of 200, approximately. Then, your program can read in a stream of accelerometer and gyroscope rate data and estimate the position and orientation of the rigid body. To estimate position changes, you can perform double integration of acceleration data in X, Y, and Z axes. For orientation, you can

1. Integrate angular velocities that are sampled from gyroscope sensor, or
2. Use a Kalman filter with both accelerometer data and gyroscope data.

To switch between the two approaches, your program should read in button events from a mouse (a USB mouse connected to the micro USB port of Galileo board). When either left or right button is “released”, the position is reset to (0,0,0). Then, the computation of orientation follows the first approach if the left button is released, or the second one otherwise.

Here are some useful links of using MPU 6050

1. A MPU-650 demo program with the RRaspberryPi https://github.com/richardghirst/PiBits/tree/master/MPU6050-Pi-Demo
2. A practical approach to Kalman filter and how to implement it http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/
3. An Galileo sketch implementation of Kalman filter for MPU 6050. https://github.com/TKJElectronics/KalmanFilter
4. A quadrocoper control example which includes MPU 6050 and Kalman filer code. https://github.com/big5824/Quadrocopter (see the description at http://www.botched.co.uk/pic-tutorials/mpu6050-setup-data-aquisition/)
5. Arduino I2Cdevlib. http://www.i2cdevlib.com/devices/mpu6050#source.

[Additional requirement for CSE 598 students]

- Your program should read sensor data from MPU 6050 only when the data is ready. This can be done with *Data Ready Interrupt* signal from MPU 6050.
- Optimize your program such that it can run at a high sampling rate and give a test what the highest sampling rate your program can attain. The result should be stated in readme file.

Due Date

This assignment will be due at 11:59pm on Nov. 24.

What to Turn in for Grading

- Create a working directory to include your source files, makefiles, and readme. Your source files should include the main program *main_4.c* (the user-level program), and any other .c or .h files for the assignment. (please clean up all object code from your submission and we will recompile your code using your makefiles.)
- Comment your source files properly and rewrite the readme file to describe how to use your software.
- Compress the directory into a zip archive file named cse438(598)-lastname-assgn04.zip. Note that any object code or temporary build files should not be included in the submission.
- Submit the zip archive to Blackboard by the due date and time.
- Failure to follow these instructions may cause an annoyed and cranky TA or instructor to deduct points while grading your assignment.