Lab 6: Data Filtering and Motion Gesture Control

A) Objectives

- To obtain the Acceleration data and Euler angles by sensor module GY-25Z.
- To filter the noisy data by Low Pass Filter and Moving Average Filter.
- To make a motion gesture control of a car.

B) Background

We can obtain 3-Axis accelerometer data (X-, Y- and Z-axis) and Euler Angles (Roll, Pitch and Yaw) from GY-25Z module as shown in Fig. 1.

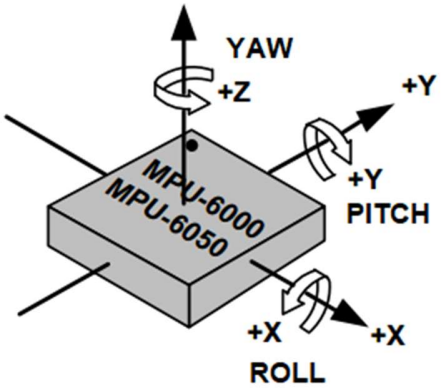

Fig. 1 – Orientation of Axis of Sensitivity and Polarity of Rotation

Task 1 – Get Acceleration data and Euler angles

Like the setup and procedures in Lab 5, we can receive the 3-axis acceleration data and Euler angles in the data packet from GY-25Z through UART.

Hardware connection:

- Place GY-25Z at the center of the breadboard
- Connect GY-25Z (VCC) to 5V
- Connect GY-25Z (TX) to Arduino Nano D10
- Connect GY-25Z (RX) to Arduino Nano D11
- Connect GY-25Z(GND) to GND

Procedure:

- Set software UART pin as D10 and D11. (By library SoftwareSerial.h)

- Set baud rate as 115200.
- Wait a few seconds after power on.
- Send command (0x55, 0x11) to set GY-25Z for output acceleration data and Euler angles.
- Send command (0x56, 0x02) to set GY-25Z for output data automatically.
- Wait and receive the data packet sent from GY-25Z.

Header		Data Type	Nums of Data	Acceleration at X-axis		Acceleration at Y-axis		Acceleration at Z-axis		Roll		Pitch		Yaw		Checksum
0x5A	0x5A	[7:0]	[7:0]	AX[15:8]	AX[7:0]	AY[15:8]	AY[7:0]	AZ[15:0]	AZ[7:0]	R[15:0]	R[7:0]	P[15:0]	P[7:0]	Y[15:0]	Y[7:0]	[7:0]

- Combine and calculate the acceleration of 3-axis in g.
  - The range of received data is -32768 ~ 32767 which represents the acceleration range from -2g to 2g. Hence, the acceleration in g = received_data / (32767/2).
- Combine and rescale the Euler angles.
- Print out the 3-axis acceleration data.
- Place your breadboard on a horizontal surface. The acceleration of X- and Y-axis should be equal to 0. And the acceleration of Z-axis should be 1g.

Task 2 – Apply low-pass filer and moving average filter

The value of ADC measurements can change quickly. We can apply the filter to remove the short-term fluctuations and leave the long-term trend.

(a) Low-pass filter:

A low-pass filter (LPF) is a filter that passes signals with a frequency lower than a selected cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency.

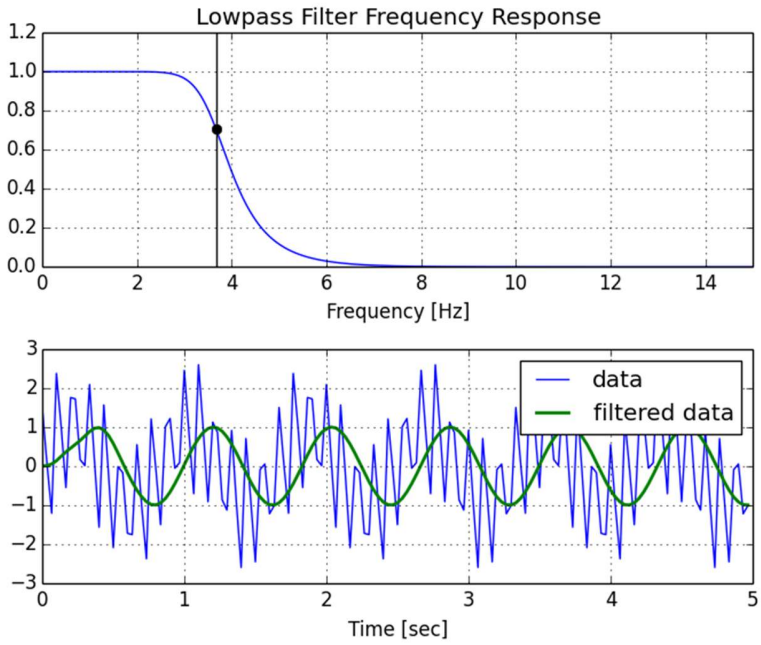

Fig. 2 – Low-pass filter in frequency and time domain

Since we're taking discrete-time samples, to 'smooth' out the data, a weighted average between filtered value and current value (the data from the current time step) is needed. We will discuss more examples of low pass filter sampling and cut-off frequencies with respect to data filtering in class. For now, the filtering can be achieved by the following equation:

filtered_value = (1.0 – weight) * filtered_value + weight * current_value;

where 'weight' represents how much the value is smoothed.

Procedure:

- Apply the low-pass filter (by the above equation) on the acceleration data of Z-axis.
- Use 'weight' = 0.9.
- Print out the values before and after the low-pass filter.
- Move the breadboard and observe the results by Serial Plotter.
- Repeat the observation with 'weight' = 0.1.

(b) Moving average filter:

It is an average of a sub-set of data. Assume we create an array to hold 8 data. On each pass of the loop, the latest data replaces the oldest data of the array and recalculate the average of the data in the array. It is a common filter due to its simplicity and is effective reducing random noise while retaining a sharp step response especially for signals recorded in time-domain.

Procedure:

- Create an array of 8 acceleration data of Z-axis.
- Shift the array to vacate the oldest data.
- Insert the latest data into the array.
- Calculate the average value of the array.
- Print out the values of latest data, low-pass filter (weight = 0.1) and moving average filter.
- Observe the results by Serial Plotter.

Task 3 – Motion gesture control

Imagine the breadboard is a remote control of a car, like Nintendo Switch Controller in Fig. 3.

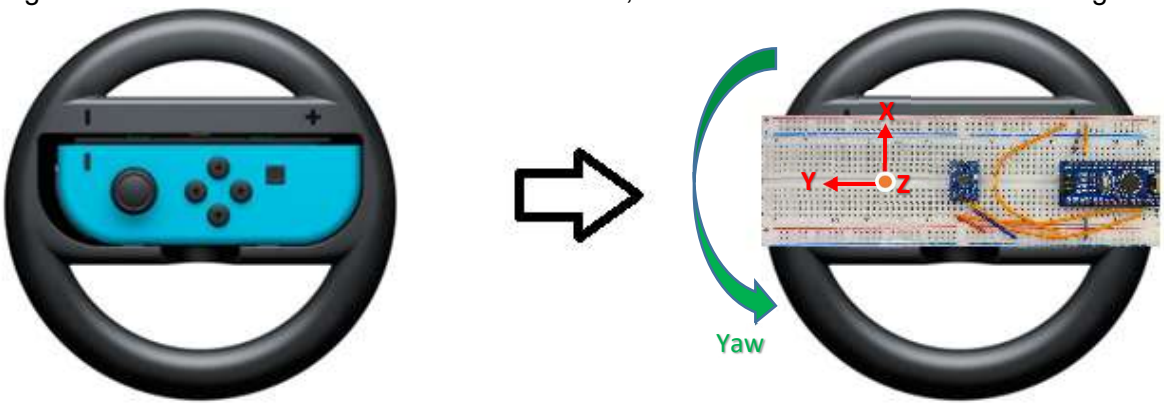

Fig. 3 – Use the Arduino Nano with GY-25Z as motion gesture control

We define the control of the car as followings:

(i) The car moves forward when the breadboard flips forward over a threshold value.
- acceleration of z-axis > threshold

(ii) The car stops when the breadboard holds as Fig. 3.
- - threshold < acceleration of z-axis < threshold

(iii) The car moves backward when the breadboard flips backward over a threshold value.
- acceleration of z-axis < - threshold

Assume the maximum steering angle is 90°:

(iv) The car turns left when the breadboard steers left.
- When the sensor faces upward (i.e. acceleration of z-axis > 0), Yaw angle increases and the range is from 0° to 90°.
- When the sensor faces downward (i.e. acceleration of z-axis < 0), Yaw angle value is 180° inversed. Yaw angle decreases and the range is from 90° to 180°.

(v) The car turns right when the breadboard steer right.
- When the sensor faces upward (i.e. acceleration of z-axis > 0), Yaw angle decreases and the range is from 0° to -90°.
- When the sensor faces downward (i.e. acceleration of z-axis < 0), Yaw angle increases and the range is from -90° to -180°.

Procedure:

- Receive the acceleration data of Z-axis and Yaw angle from GY-25Z.
- For every power on, reset the Yaw angle to 0 by calibration command in setup().
- Apply the moving average filter on the acceleration data of Z-axis.
- Check the above definitions (i)-(v) to determine the action of the car.
- The sensitivity of acceleration can be adjusted by the threshold value.
- Print out the action of the car by the motion gesture control.

Optional (not included in the skeleton code of Lab 6):

- *Connect two motors as in Lab 3 which represent left and right wheels of the car.*
- *Control the car using the differential drive by the motion gesture.*