Aalto University

Intelligent User Interfaces Studio

ELEC-E7870 - Advanced Topics in User Interfaces

04.02.2018

Kashyap Todi & Sunjun Kim

Lecture 2:
Hello, Hardware

Before start...

Please visit and download

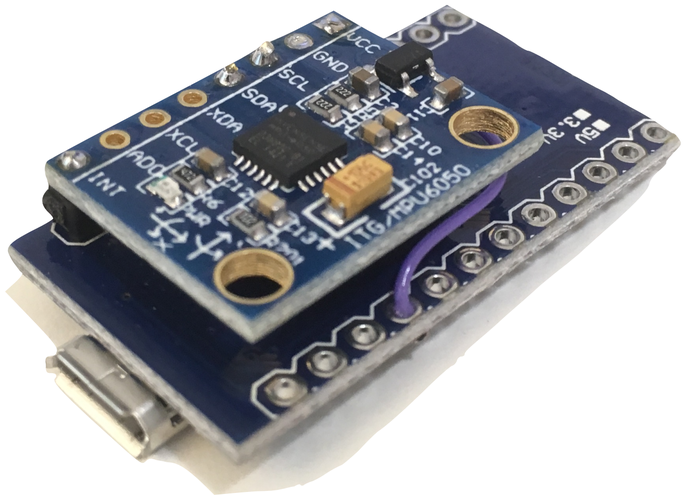
https://www.arduino.cc/
→ Software → Downloads → Arduino IDE v1.8.8

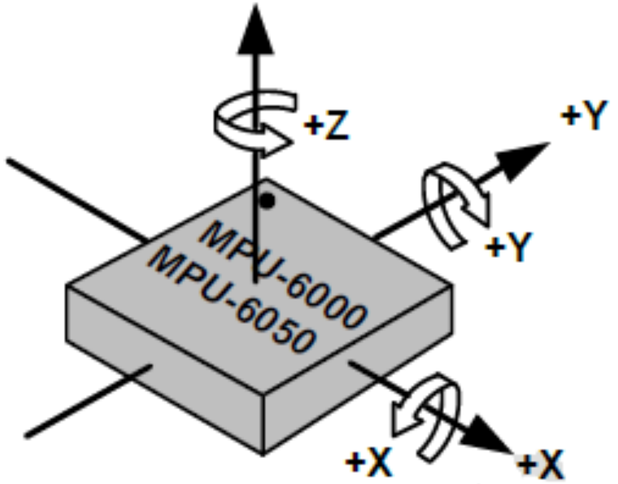
http://processing.org/
→ Download → v3.5.3

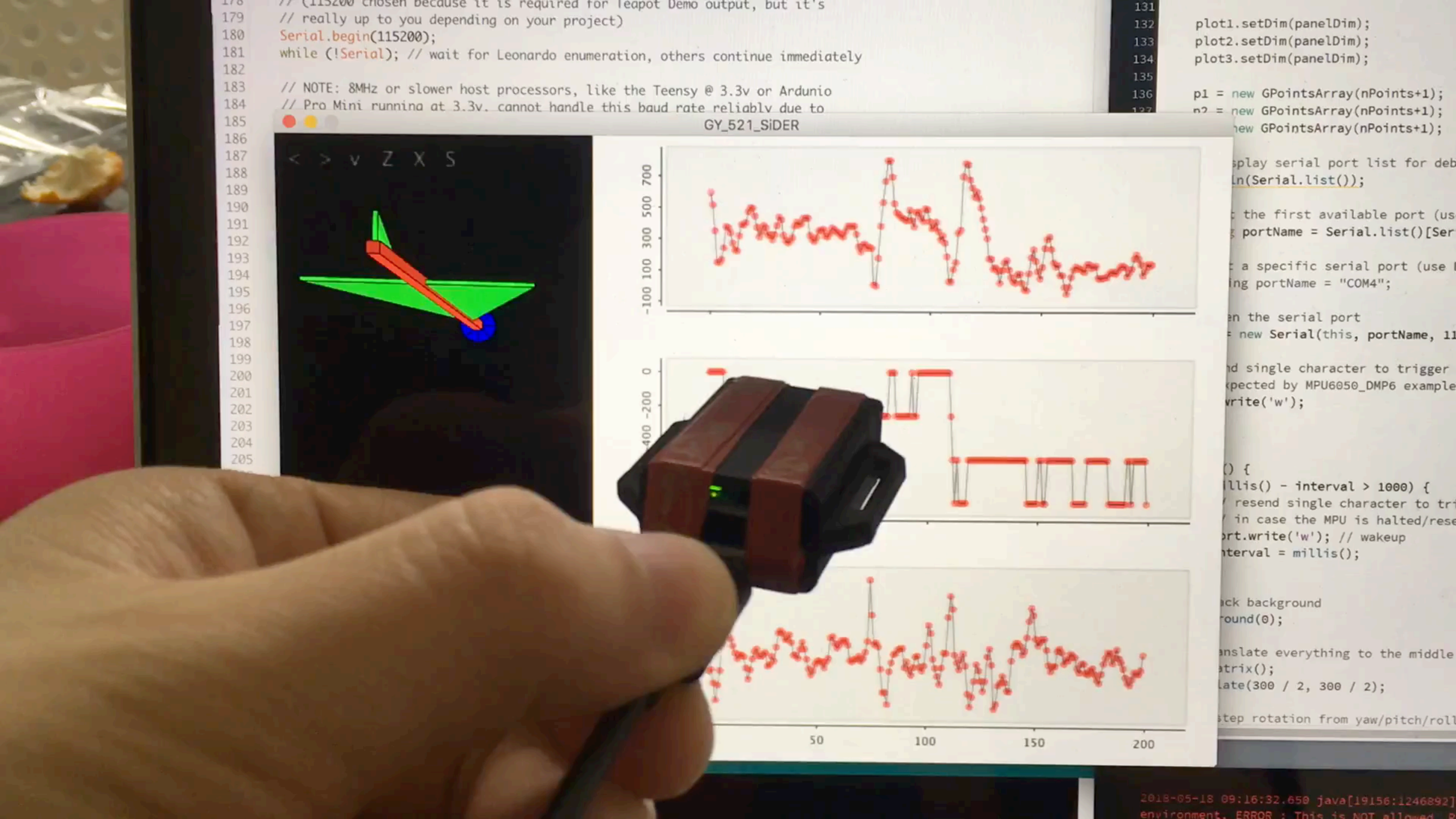
Objectives

- Setup an environment for the following lectures.
- Understand the hardware.
- Understand basic thresholding and hysteresis.

Our Target Device





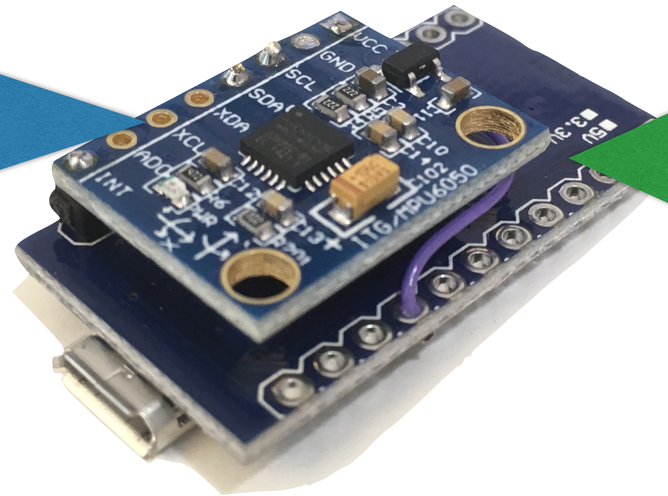

// really up to you depending on your project)
Serial.begin(115200);
while (!Serial); // wait for Leonardo enumeration, others continue immediately
// NOTE: 8MHz or slower host processors, like the Teensy @ 3.3v or Ardunio
plot1.setDim(panelDim);
plot2.setDim(panelDim);
plot3.setDim(panelDim);
p1 = new GPointsArray(nPoints+1);
GY_521_SiDER
< > v Z X S
-100 100 300 500 700
-400 -200 0
50 100 150 200

Aalto University

Components

GY-521 MPU-6050

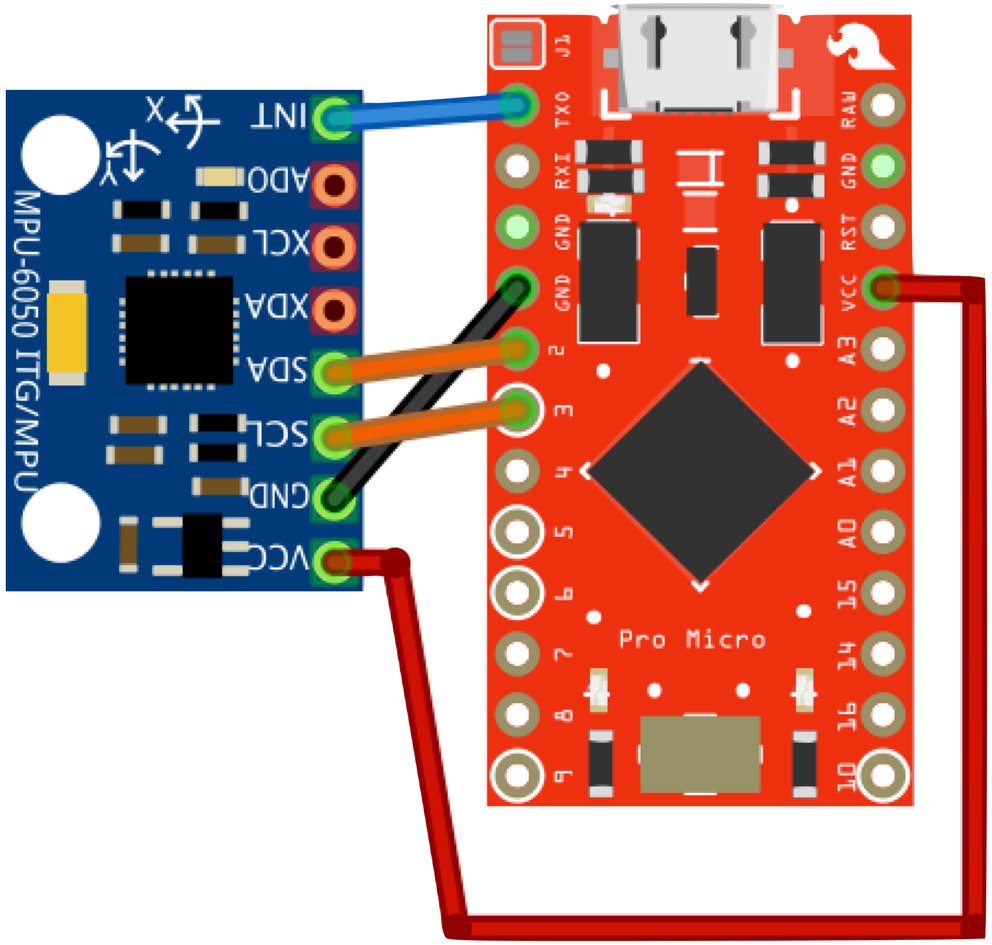
- Motion sensor module
- Documentation: https://playground.arduino.cc/Main/MPU-6050
- 3-axis accelerometer + gyro
- 16-bit analog-digital convert
- Communication: I^2C



Arduino Pro Micro

- Microprocessor board
- Manufacturer: Sparkfun https://www.sparkfun.com/products/12640
- Main MCU: ATmega32u4
- 5V / 16Mhz version
- Operated with Arduino IDE

Schematics



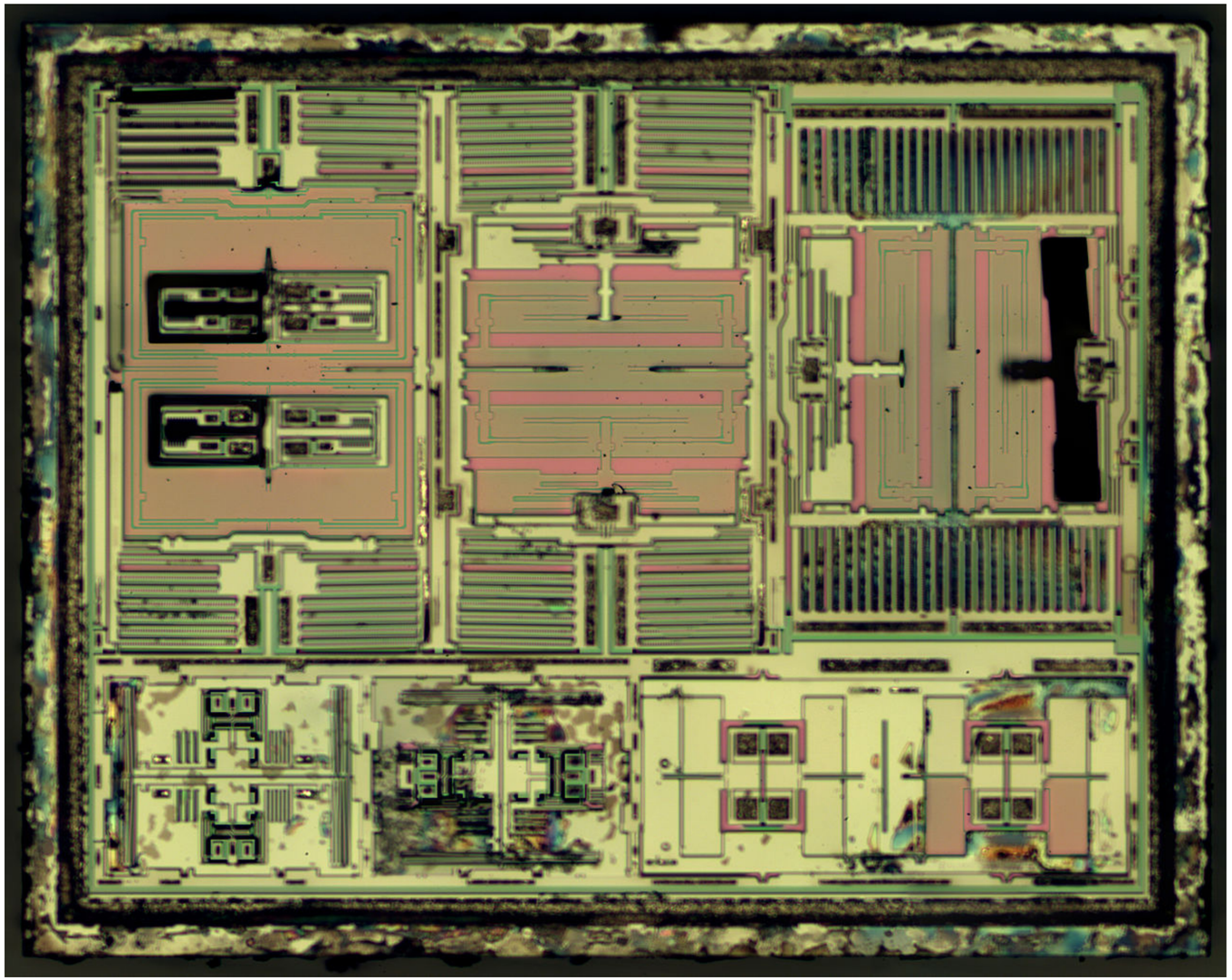
GY-521 Module – Arduino Pro Micro

INT – TXO = D1

SDA – Pin 2: I²C SDA

SCL – Pin 3: I²C SCL

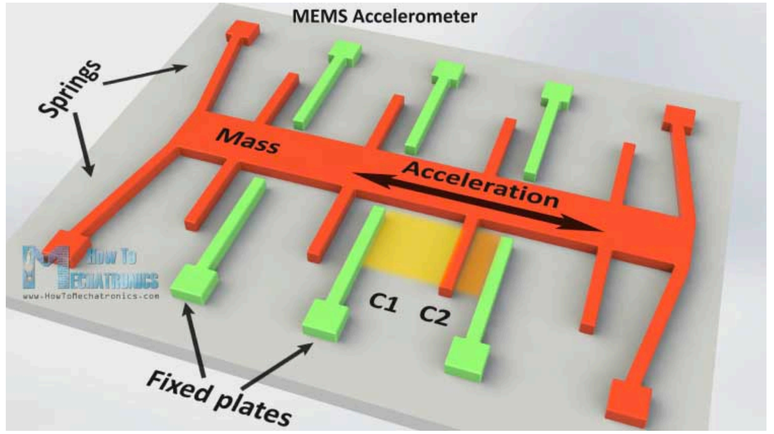
MEMS Sensor (MPU6050)



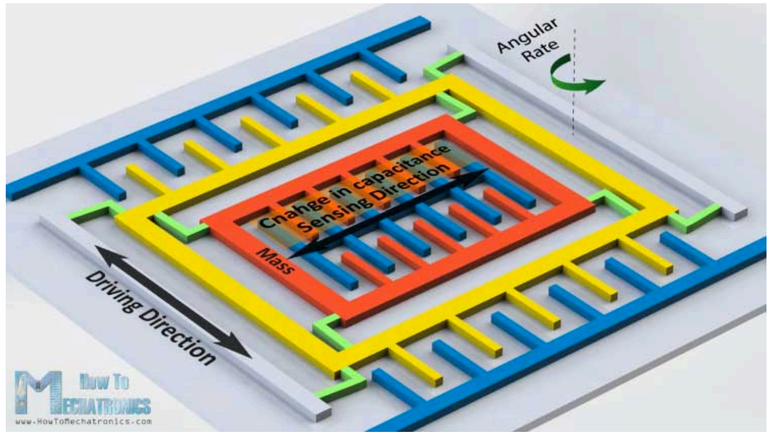
ZeptoBars [CC BY 3.0 (https://creativecommons.org/licenses/by/3.0)], via Wikimedia Commons

Micro-Electro-Mechanical System

Accelerometer



Gyroscope



https://www.youtube.com/watch?v=eqZgxR6eRjo

Arduino Sketch

setup()

This code runs only **once** at startup

loop()

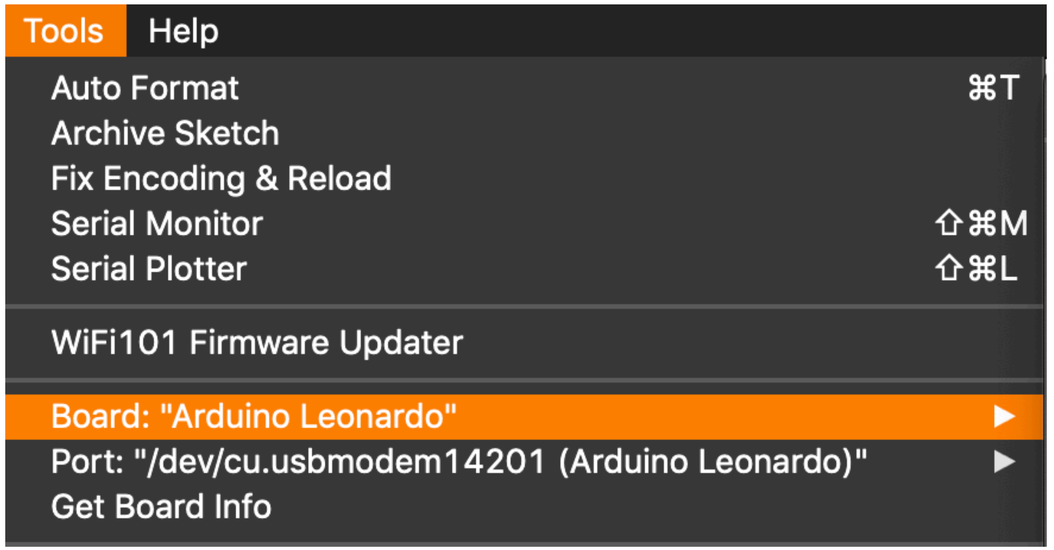
This code **repeated** infinitely

Minimal example

Open **Programs/Arduino/MPU6050_minimal**

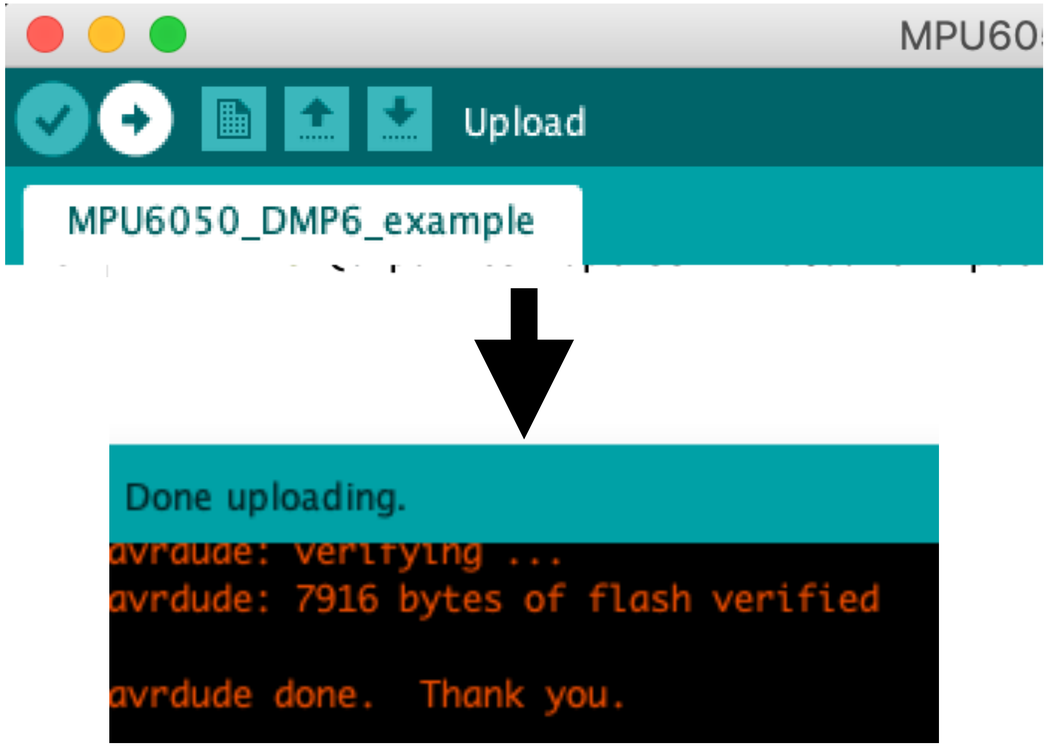
https://playground.arduino.cc/Main/MPU-6050

Select **Tools → Board : Arduino Leonardo**
Select **Port → COM# (or /dev/cu.*) with (Arduino Leonardo)**



Minimal Example

Hit "Upload" button



Outputs on "Serial Monitor"

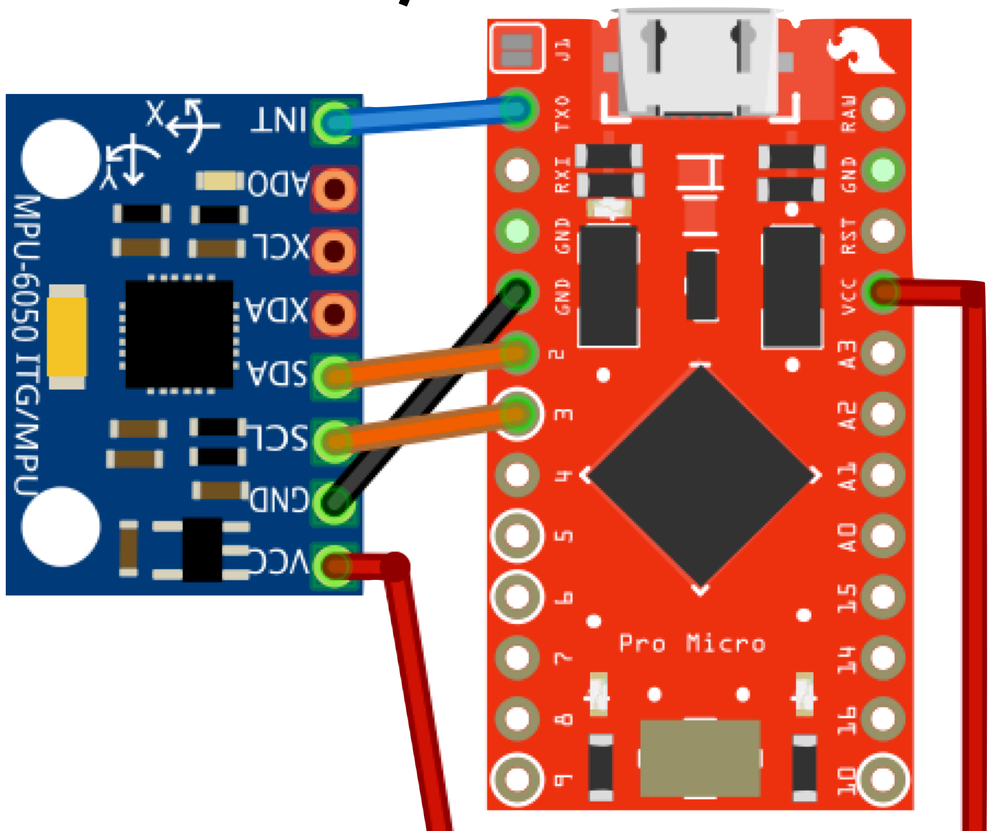

```
AcX = 12216 | AcY = -2000 | AcZ = 12140 | Tmp = 34.27 | GyX = 19 | GyY = -174 | GyZ = -107
AcX = 12144 | AcY = -1836 | AcZ = 12392 | Tmp = 34.41 | GyX = 105 | GyY = -201 | GyZ = 55
AcX = 3884 | AcY = -1476 | AcZ = 15832 | Tmp = 34.37 | GyX = -1509 | GyY = 32767 | GyZ = -726
AcX = 4404 | AcY = -2104 | AcZ = 18256 | Tmp = 34.46 | GyX = 850 | GyY = -5215 | GyZ = -399
AcX = 8808 | AcY = -1724 | AcZ = 14640 | Tmp = 34.32 | GyX = 43 | GyY = -1257 | GyZ = -228
AcX = 8676 | AcY = -1712 | AcZ = 14936 | Tmp = 34.46 | GyX = 134 | GyY = 222 | GyZ = -91
AcX = 8440 | AcY = -1724 | AcZ = 15020 | Tmp = 34.37 | GyX = -100 | GyY = -1717 | GyZ = -113
AcX = 9112 | AcY = -1696 | AcZ = 14532 | Tmp = 34.41 | GyX = -19 | GyY = -1235 | GyZ = -61
AcX = 10856 | AcY = -1428 | AcZ = 13096 | Tmp = 34.32 | GyX = 1080 | GyY = 2636 | GyZ = -2303
AcX = 11148 | AcY = -1784 | AcZ = 13240 | Tmp = 34.41 | GyX = -317 | GyY = -2020 | GyZ = -144
AcX = 11412 | AcY = -1748 | AcZ = 13012 | Tmp = 34.41 | GyX = -18 | GyY = -2611 | GyZ = -39
AcX = 10456 | AcY = -2192 | AcZ = 13336 | Tmp = 34.37 | GyX = -959 | GyY = -1489 | GyZ = -534
AcX = 11448 | AcY = -1880 | AcZ = 12880 | Tmp = 34.46 | GyX = 250 | GyY = 887 | GyZ = -25
AcX = 11344 | AcY = -1732 | AcZ = 12948 | Tmp = 34.41 | GyX = 136 | GyY = -114 | GyZ = -57
```

I²C Communication

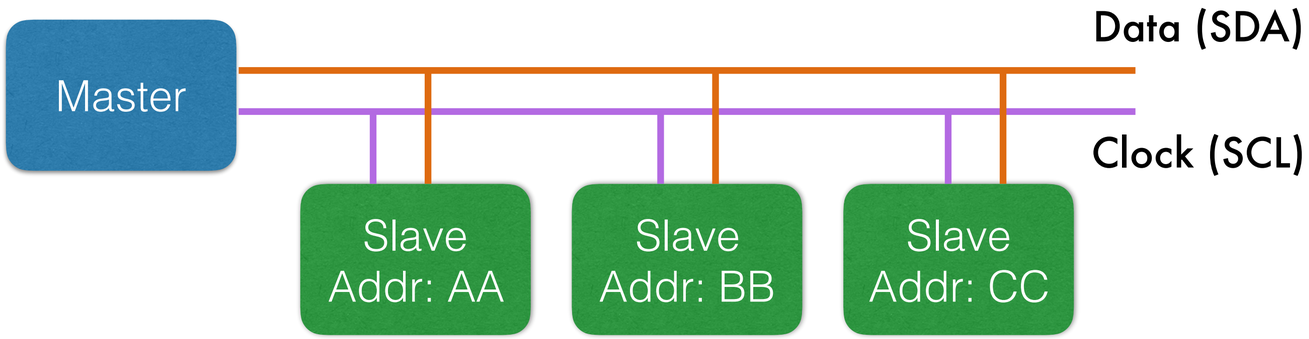
(= TWI, Two-Wire Interface)

Slave
(addr: 0x68)

Master



SDA – Pin 2: I²C SDA
SCL – Pin 3: I²C SCL



Only one master and one slave can communicate at same time

setup()

Wire

Arduino library for I²C

```
#include<Wire.h>
const int MPU_addr=0x68;  // I2C address of the MPU-6050
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
void setup(){
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B);  // PWR_MGMT_1 register
  Wire.write(0);     // set to zero (wakes up the MPU-6050)
  Wire.endTransmission(true);
  Serial.begin(9600);
}
```

I²C Write

https://www.arduino.cc/en/reference/wire

```
Wire.beginTransmission(MPU_addr);
Wire.write(0x3B);
Wire.endTransmission(false);
```

beginTransmission
set the slave addr.

write
Queue the data
(from master to slave)

endTransmission
Ends a transmission.
➔ Flushes the bytes that were queued by **write()**

true
send a stop message
false
keep the connection active

I²C Read

https://www.arduino.cc/en/reference/wire

Set the target register

```
Wire.beginTransmission(MPU_addr);
Wire.write(0x3B);
Wire.endTransmission(false);
```

requestFrom()

Request 14 bytes from the MPU

```
Wire.requestFrom(MPU_addr,14,true);  // request a total of 14 registers
AcX=Wire.read()<<8|Wire.read();  // 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
AcY=Wire.read()<<8|Wire.read();  // 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
AcZ=Wire.read()<<8|Wire.read();  // 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
Tmp=Wire.read()<<8|Wire.read();  // 0x41 (TEMP_OUT_H) & 0x42 (TEMP_OUT_L)
GyX=Wire.read()<<8|Wire.read();  // 0x43 (GYRO_XOUT_H) & 0x44 (GYRO_XOUT_L)
GyY=Wire.read()<<8|Wire.read();  // 0x45 (GYRO_YOUT_H) & 0x46 (GYRO_YOUT_L)
GyZ=Wire.read()<<8|Wire.read();  // 0x47 (GYRO_ZOUT_H) & 0x48 (GYRO_ZOUT_L)
```

read()

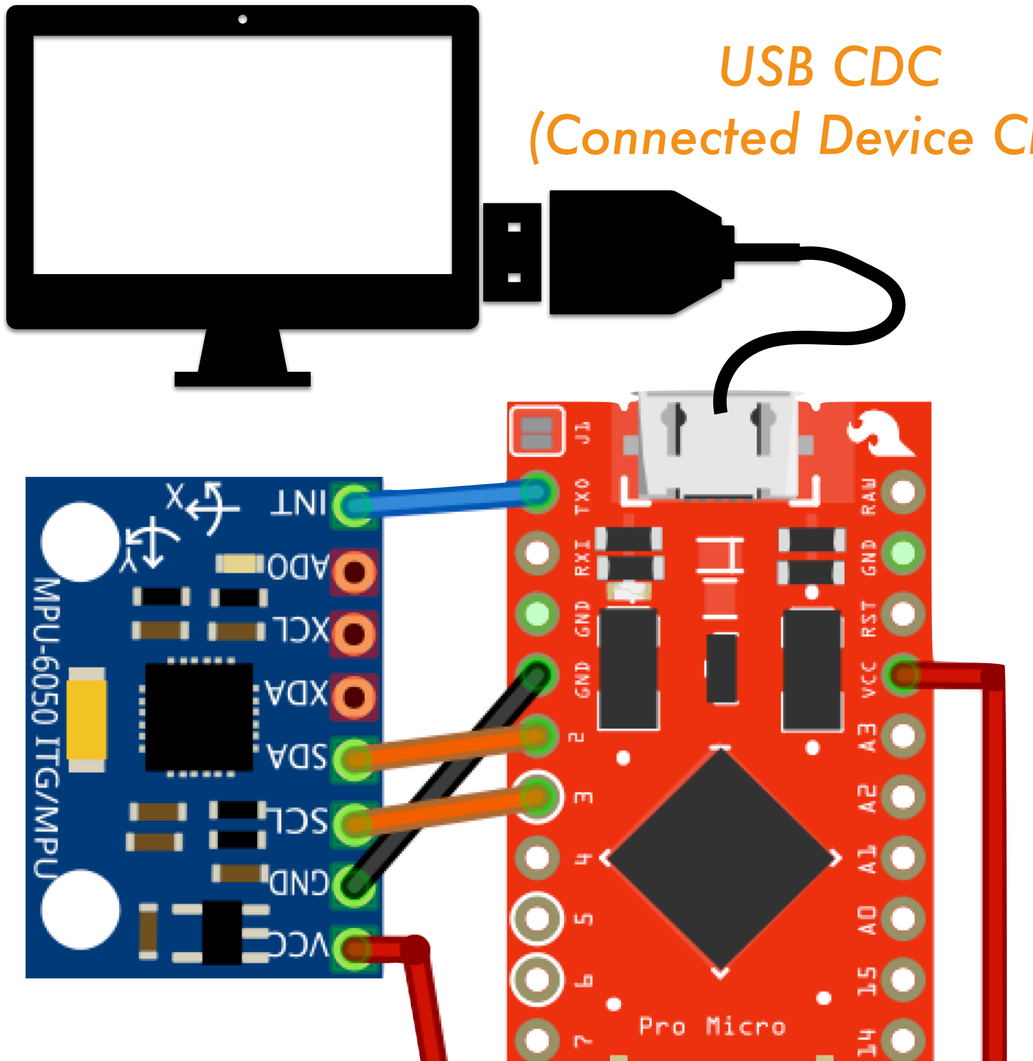
read 1 byte

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F
3B	59	ACCEL_XOUT_H	R
3C	60	ACCEL_XOUT_L	R
3D	61	ACCEL_YOUT_H	R
3E	62	ACCEL_YOUT_L	R
3F	63	ACCEL_ZOUT_H	R
40	64	ACCEL_ZOUT_L	R
41	65	TEMP_OUT_H	R
42	66	TEMP_OUT_L	R
43	67	GYRO_XOUT_H	R
44	68	GYRO_XOUT_L	R
45	69	GYRO_YOUT_H	R
46	70	GYRO_YOUT_L	R
47	71	GYRO_ZOUT_H	R
48	72	GYRO_ZOUT_L	R

Documents/Datasheet - MPU-6000-Register-Map1.pdf (page 6)

Serial Output

Processed information are sent through serial output



USB CDC
(Connected Device Class)

setup()

```
Serial.begin(9600);
```

loop()

```
Serial.print("AcX = "); Serial.print(AcX);
Serial.print(" | AcY = "); Serial.print(AcY);
Serial.print(" | AcZ = "); Serial.print(AcZ);
Serial.print(" | Tmp = "); Serial.print(Tmp/340.00+36.53);
Serial.print(" | GyX = "); Serial.print(GyX);
Serial.print(" | GyY = "); Serial.print(GyY);
Serial.print(" | GyZ = "); Serial.println(GyZ);
```

The temperature in degrees C for a given register value may be computed as:

Temperature in degrees C = (TEMP_OUT Register Value as a signed quantity)/340 + 36.53

Documents/Datasheet - MPU-6000-Register-Map1.pdf (page 30)

Exercise

Identify the axis and their directions

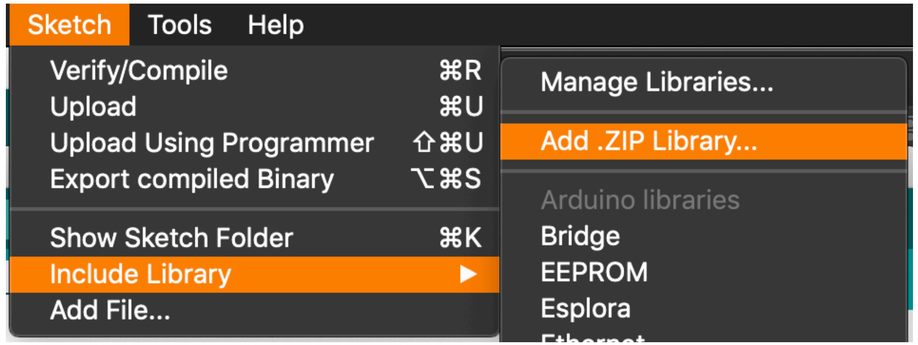
```
AcX = 12216 | AcY = -2000 | AcZ = 12140 | Tmp = 34.27 | GyX = 19 | GyY = -174 | GyZ = -107
AcX = 12144 | AcY = -1836 | AcZ = 12392 | Tmp = 34.41 | GyX = 105 | GyY = -201 | GyZ = 55
AcX = 3884 | AcY = -1476 | AcZ = 15832 | Tmp = 34.37 | GyX = -1509 | GyY = 32767 | GyZ = -726
AcX = 4404 | AcY = -2104 | AcZ = 18256 | Tmp = 34.46 | GyX = 850 | GyY = -5215 | GyZ = -399
AcX = 8808 | AcY = -1724 | AcZ = 14640 | Tmp = 34.32 | GyX = 43 | GyY = -1257 | GyZ = -228
AcX = 8676 | AcY = -1712 | AcZ = 14936 | Tmp = 34.46 | GyX = 134 | GyY = 222 | GyZ = -91
AcX = 8440 | AcY = -1724 | AcZ = 15020 | Tmp = 34.37 | GyX = -100 | GyY = -1717 | GyZ = -113
AcX = 9112 | AcY = -1696 | AcZ = 14532 | Tmp = 34.41 | GyX = -19 | GyY = -1235 | GyZ = -61
AcX = 10856 | AcY = -1428 | AcZ = 13096 | Tmp = 34.32 | GyX = 1080 | GyY = 2636 | GyZ = -2303
AcX = 11148 | AcY = -1784 | AcZ = 13240 | Tmp = 34.41 | GyX = -317 | GyY = -2020 | GyZ = -144
AcX = 11412 | AcY = -1748 | AcZ = 13012 | Tmp = 34.41 | GyX = -18 | GyY = -2611 | GyZ = -39
AcX = 10456 | AcY = -2192 | AcZ = 13336 | Tmp = 34.37 | GyX = -959 | GyY = -1489 | GyZ = -534
AcX = 11448 | AcY = -1880 | AcZ = 12880 | Tmp = 34.46 | GyX = 250 | GyY = 887 | GyZ = -25
AcX = 11344 | AcY = -1732 | AcZ = 12948 | Tmp = 34.41 | GyX = 136 | GyY = -114 | GyZ = -57
```

Advanced example

Open **Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino,**

* Modified from https://github.com/jrowberg/i2cdevlib by Jeff Rowberg

add **Libraries,**

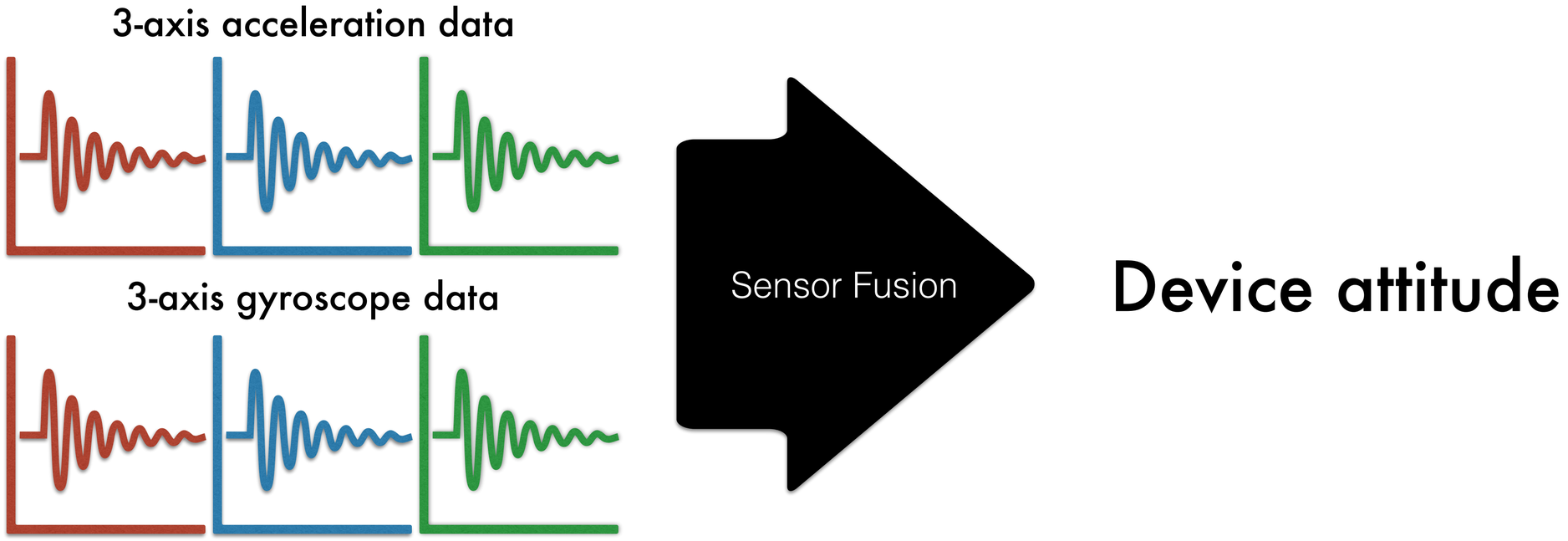


Programs/Arduino/Lib/
I2Cdev.zip
MPU6050.zip

and Upload.

Digital Motion Processor (DMP)

DMP does a complex calculation with raw sensor values.



https://github.com/jrowberg/i2cdevlib/blob/master/Arduino/MPU6050/MPU6050_6Axis_MotionApps20.h

Digital Motion Processor (DMP)

DMP does a complex calculation with raw sensor values.

```
/* ============================================================================================ *
 | Default MotionApps v2.0 42-byte FIFO packet structure:                                       |
 |                                                                                              |
 | [QUAT W][      ][QUAT X][      ][QUAT Y][      ][QUAT Z][      ][GYRO X][      ][GYRO Y][      ] |
 |   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  |
 |                                                                                              |
 | [GYRO Z][      ][ACC X ][      ][ACC Y ][      ][ACC Z ][      ][      ]                     |
 |  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41                      |
 * ============================================================================================ */
```

https://github.com/jrowberg/i2cdevlib/blob/master/Arduino/MPU6050/MPU6050_6Axis_MotionApps20.h

Digital Motion Processor (DMP)

DMP does a complex calculation with raw sensor values.

Quaternion of the device attitude

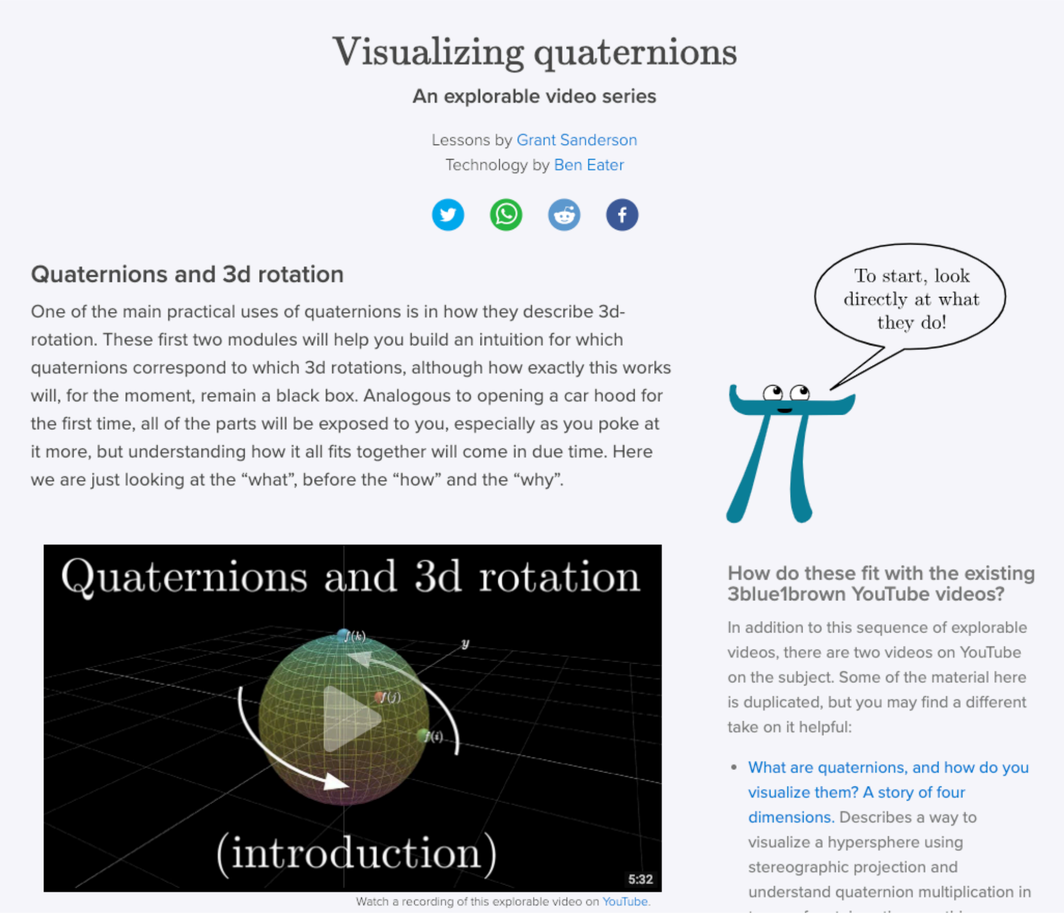
```
/* ============================================================================================ *
 | Default MotionApps v2.0 42-byte FIFO packet structure:                                       |
 |                                                                                              |
 | [QUAT W][      ][QUAT X][      ][QUAT Y][      ][QUAT Z][      ][GYRO X][      ][GYRO Y][      ] |
 |   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  |
 |                                                                                              |
 | [GYRO Z][      ][ACC X ][      ][ACC Y ][      ][ACC Z ][      ][      ]                     |
 |  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41                      |
 * ============================================================================================ */
```

```
rawPacket[2] = fifoBuffer[0];
rawPacket[3] = fifoBuffer[1];
rawPacket[4] = fifoBuffer[4];
rawPacket[5] = fifoBuffer[5];
rawPacket[6] = fifoBuffer[8];
rawPacket[7] = fifoBuffer[9];
rawPacket[8] = fifoBuffer[12];
rawPacket[9] = fifoBuffer[13];
```

/Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino

Wait, what's quaternion?

Quaternion is a four-dimensional vector to represent 3D rotation.



Visualizing quaternions

An explorable video series

Lessons by Grant Sanderson
Technology by Ben Eater

Quaternions and 3d rotation

One of the main practical uses of quaternions is in how they describe 3d-rotation. These first two modules will help you build an intuition for which quaternions correspond to which 3d rotations, although how exactly this works will, for the moment, remain a black box. Analogous to opening a car hood for the first time, all of the parts will be exposed to you, especially as you poke at it more, but understanding how it all fits together will come in due time. Here we are just looking at the "what", before the "how" and the "why".

Watch a recording of this explorable video on YouTube.

How do these fit with the existing 3blue1brown YouTube videos?

In addition to this sequence of explorable videos, there are two videos on YouTube on the subject. Some of the material here is duplicated, but you may find a different take on it helpful:

- What are quaternions, and how do you visualize them? A story of four dimensions. Describes a way to visualize a hypersphere using stereographic projection and understand quaternion multiplication in

https://eater.net/quaternions

Digital Motion Processor (DMP)

DMP does a complex calculation with raw sensor values.

```
/* ============================================================================================ *
 | Default MotionApps v2.0 42-byte FIFO packet structure:                                       |
 |                                                                                              |
 | [QUAT W][      ][QUAT X][      ][QUAT Y][      ][QUAT Z][      ][GYRO X][      ][GYRO Y][      ] |
 |   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  |
 |                                                                                              |
 | [GYRO Z][      ][ACC X ][      ][ACC Y ][      ][ACC Z ][      ][      ]                     |
 |  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41                      |
 * ============================================================================================ */
```

Raw Gyroscope values

```
rawPacket[10] = fifoBuffer[16];
rawPacket[11] = fifoBuffer[17];
rawPacket[12] = fifoBuffer[20];
rawPacket[13] = fifoBuffer[21];
rawPacket[14] = fifoBuffer[24];
rawPacket[15] = fifoBuffer[25];

rawPacket[16] = fifoBuffer[28];
rawPacket[17] = fifoBuffer[29];
rawPacket[18] = fifoBuffer[32];
rawPacket[19] = fifoBuffer[33];
rawPacket[20] = fifoBuffer[36];
rawPacket[21] = fifoBuffer[37];
```

Raw Accelerometer values

/Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino

Wrap data in a packet

```
uint8_t rawPacket[24] = { '$', 0x03,
                          0,0,0,0,0,0,0,0, // quat
                          0,0,0,0,0,0,     // gyro
                          0,0,0,0,0,0,     // accel
                          '\r', '\n' };
```

'$', 0x03 → Start bytes
+ 8 bytes → 4 * 16-bit integers (quaternion vector)
+ 6 bytes → 3 * 16-bit integers (raw gyroscope values)
+ 6 bytes → 3 * 16-bit integers (raw accelerometer values)
+ '\r', '\n' → End bytes

⋮

```
Serial.write(rawPacket, 24);
```

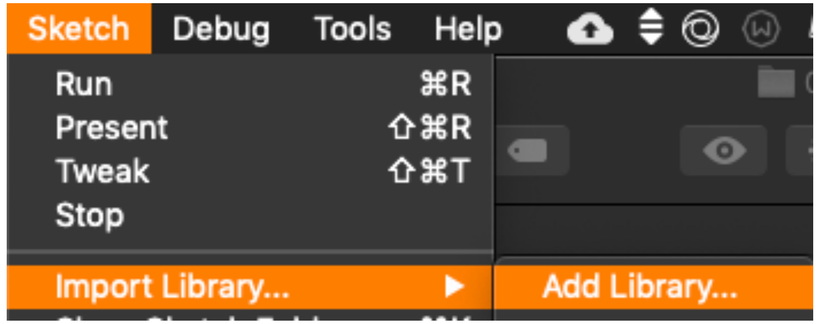
/Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino

Visualizer

Open **Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde**

* Modified from https://github.com/jrowberg/i2cdevlib by Jeff Rowberg

add **Libraries,**

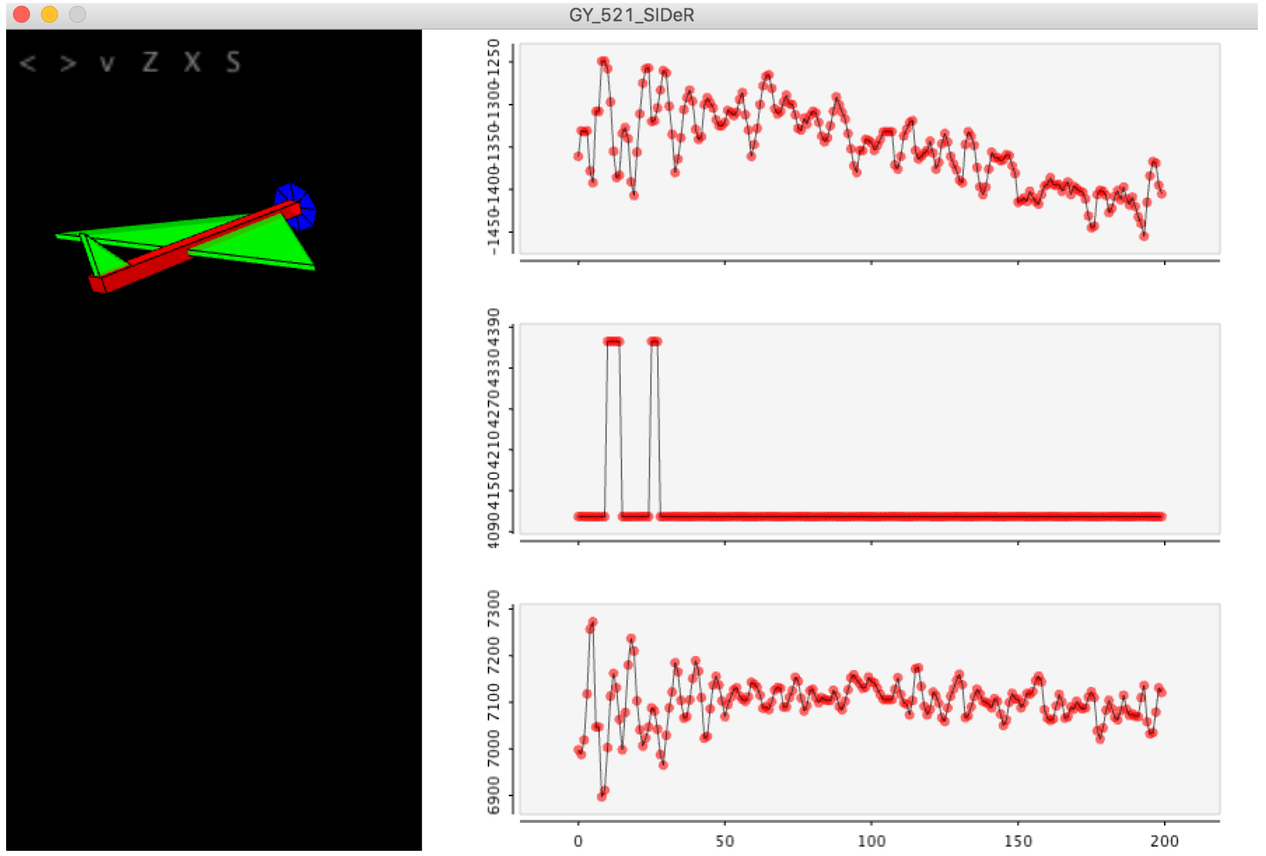


Search for
grafica
ToxicLibs

and **Run**

Visualizer

- A airplane visualizes the orientation of the sensor.
- Key activation is visualized.



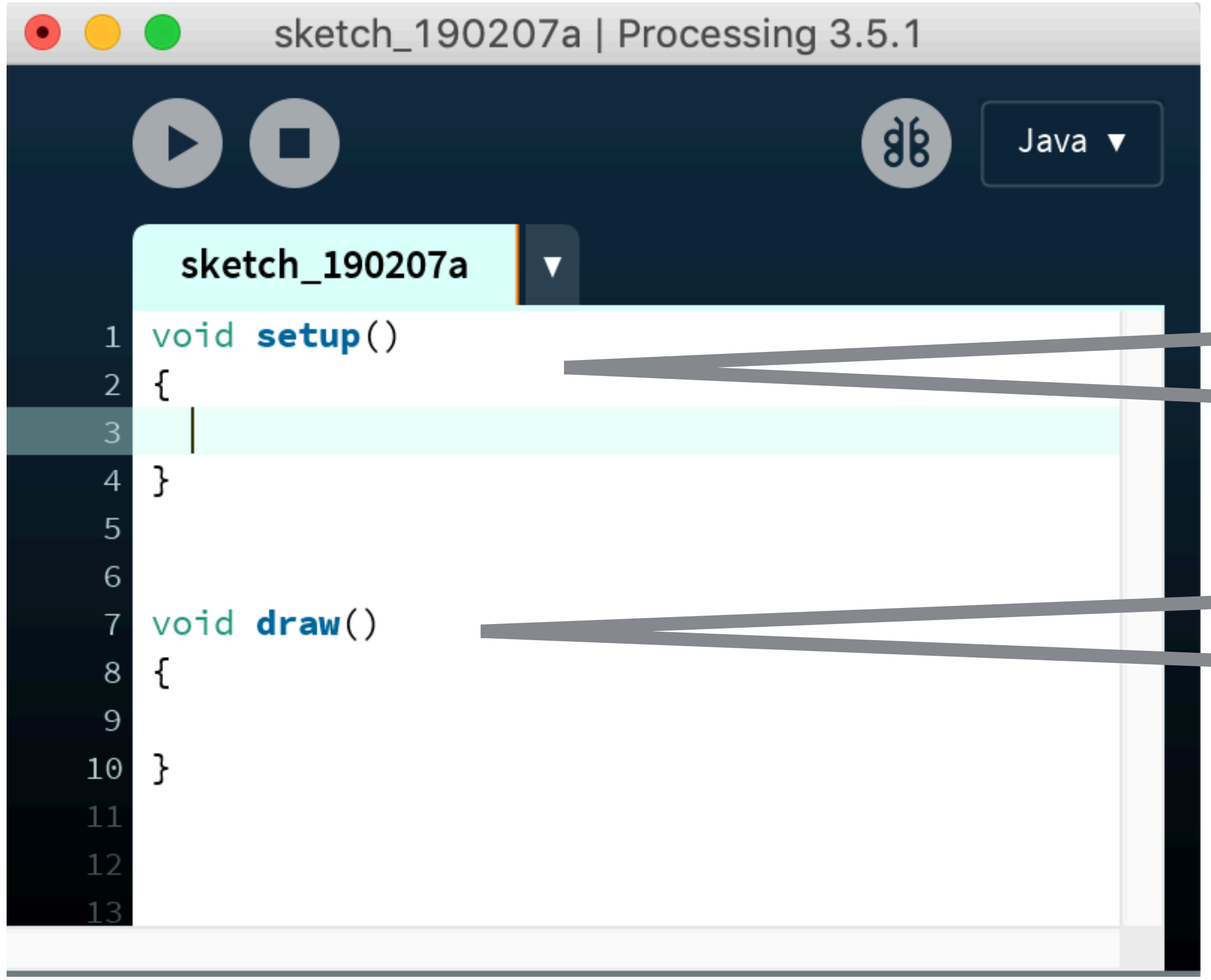
Value plotter

- Useful to monitor value change over time.
- Monitor up to 3 values

```
// plot points
add_p1 = gyro[0];
add_p2 = gyro[1];
add_p3 = gyro[2];
```

- Please tune your method according to the value range.

Processing Sketch



setup()
This code runs
only **once** at startup

draw()
This code **repeated**
infinitely

serialEvent()

This function is executed with a new serial packet.

/Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino

```
uint8_t rawPacket[24] = { '$', 0x03,
                          0,0,0,0,0,0,0,0, // quat
                          0,0,0,0,0,0,     // gyro
                          0,0,0,0,0,0,     // accel
                          '\r', '\n' };
```

* Arduino part

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

```
if (ch == '$') {serialCount = 0;} // this will help with alignment
if (aligned < 4) {
    // make sure we are properly aligned on a 24-byte packet
    if (serialCount == 0) {
        if (ch == '$') aligned++; else aligned = 0;
    } else if (serialCount == 1) {
        if (ch == 3) aligned++; else aligned = 0;
    } else if (serialCount == 22) {
        if (ch == '\r') aligned++; else aligned = 0;
    } else if (serialCount == 23) {
        if (ch == '\n') aligned++; else aligned = 0;
    }
    //println(ch + " " + aligned + " " + serialCount);
    serialCount++;
    if (serialCount == 24) serialCount = 0;
} else {
```

* Processing part
→ align the packet

serialEvent()

Value retrieval

/Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino

```
uint8_t rawPacket[24] = { '$', 0x03,
                          0,0,0,0,0,0,0,0, // quat
                          0,0,0,0,0,0,     // gyro
                          0,0,0,0,0,0,     // accel
                          '\r', '\n' };
```

* Arduino part

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

```
// get quaternion from data packet
q[0] = ((packet[2] << 8) | packet[3]) / 16384.0f;
q[1] = ((packet[4] << 8) | packet[5]) / 16384.0f;
q[2] = ((packet[6] << 8) | packet[7]) / 16384.0f;
q[3] = ((packet[8] << 8) | packet[9]) / 16384.0f;
for (int i = 0; i < 4; i++) if (q[i] >= 2) q[i] = -4 + q[i];

gyro[0] = convert_uint_to_int((packet[10] << 8) | packet[11]);
gyro[1] = convert_uint_to_int((packet[12] << 8) | packet[13]);
gyro[2] = convert_uint_to_int((packet[14] << 8) | packet[15]);

acc[0] = convert_uint_to_int((packet[16] << 8) | packet[17]);
acc[1] = convert_uint_to_int((packet[18] << 8) | packet[18]);
acc[2] = convert_uint_to_int((packet[20] << 8) | packet[21]);
```

* Processing part
→ read numbers from bytes

Gravity, Euler angle, YPR

Following values can be calculated from quaternion vector, q[4].

Gravity vector

```
gravity[0] = 2 * (q[1]*q[3] - q[0]*q[2]);
gravity[1] = 2 * (q[0]*q[1] + q[2]*q[3]);
gravity[2] = q[0]*q[0] - q[1]*q[1] - q[2]*q[2] + q[3]*q[3];
```

Euler angle

```
euler[0] = atan2(2*q[1]*q[2] - 2*q[0]*q[3], 2*q[0]*q[0] + 2*q[1]*q[1] - 1);
euler[1] = -asin(2*q[1]*q[3] + 2*q[0]*q[2]);
euler[2] = atan2(2*q[2]*q[3] - 2*q[0]*q[1], 2*q[0]*q[0] + 2*q[3]*q[3] - 1);
```

Yaw, Pitch, Roll

```
ypr[0] = atan2(2*q[1]*q[2] - 2*q[0]*q[3], 2*q[0]*q[0] + 2*q[1]*q[1] - 1);
ypr[1] = atan(gravity[1] / sqrt(gravity[0]*gravity[0] + gravity[2]*gravity[2]));
ypr[2] = -1*atan(gravity[0] / sqrt(gravity[1]*gravity[1] + gravity[2]*gravity[2]));
```

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

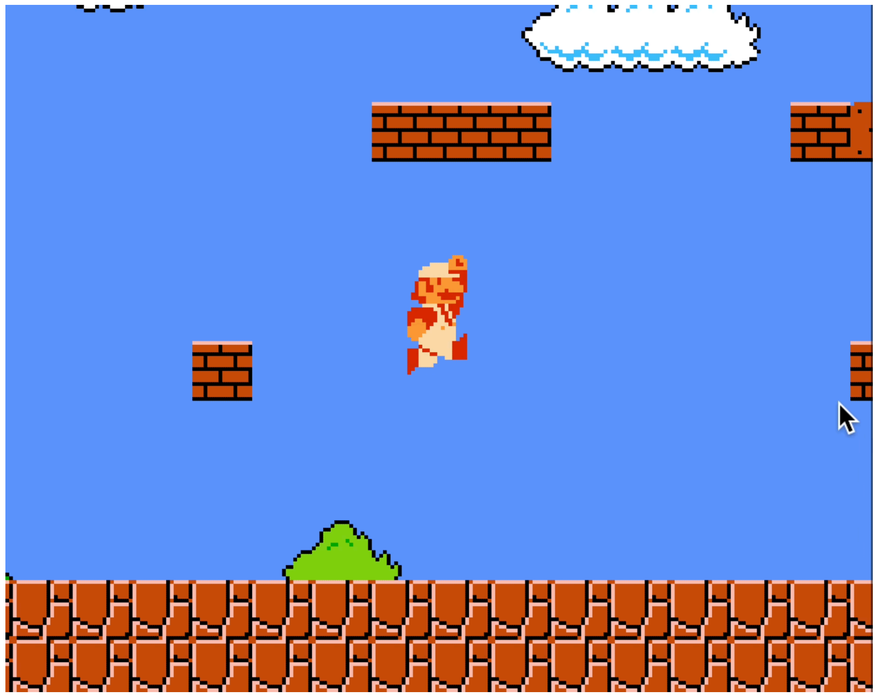
Available pre-calculated values

```
gyro[3]       --> raw gyroscope sensor values (x, y, z)
acc[3]        --> raw accelerometer values (x, y, z)

q[4]          --> Quaternion that represents rotation
gravity[3]    --> vector that represents the direction of the gravity
euler[3]      --> euler angle represents the rotation
ypr[3]        --> (yaw, pitch, role)
linearAcc[3] --> acceleration with removed gravity
linaerAccWorld[3] --> acceleration with removed gravity and motion
```

Exercise: Mario

- http://www.freemario.org/
- Ad will bother you only once
- "TUTORIALS OFF" option will save your time



- LEFT, RIGHT, DOWN : move / Z: jump / X: dash, shoot / S: special skill (don't use)

Keyboard functions

Arduino Leonardo (using ATMega32u4) board can be used as a native USB input devices.

```
49 #include <Keyboard.h>
```

```
246     Keyboard.begin();
```

```
425         case 'l':
426           Keyboard.press(KEY_LEFT_ARROW);
427           break;
428         case 'L':
429           Keyboard.release(KEY_LEFT_ARROW);
430           break;
```

/Programs/Arduino/MPU6050_DMP/MPU6050_DMP.ino

Send command in Processing

sendCommend() function

```
429 void sendCommand(int cmd, boolean isPress)
430 {
431   if(cmd >= pressedKeys.length)
432   {
433     println("ERROR: WRONG COMMAND");
434     return;
435   }
436 
437   char send = '\0';
438   if      (cmd == CMD_LEFT) send = 'l';
439   else if (cmd == CMD_RIGHT) send = 'r';
440   else if (cmd == CMD_DOWN) send = 'd';
441   else if (cmd == CMD_JUMP) send = 'z';
442   else if (cmd == CMD_DASH) send = 'x';
443   else if (cmd == CMD_SKILL) send = 's';
444 
445   if(isPress && pressedKeys[cmd] == false)  // got press command && not have been pressed
446   {
447     pressedKeys[cmd] = true;
448     port.write(send);
449   }
450   if(!isPress && pressedKeys[cmd] == true)   // got release command && have been pressed
451   {
452     pressedKeys[cmd] = false;
453     port.write(Character.toUpperCase(send));
454   }
455 }
```

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

Exercise

Play the Mario game with the device.

Find appropriate features, and map them to commands.

```
void press(int cmd)
{
  sendCommand(cmd, true);
}
void release(int cmd)
{
  sendCommand(cmd, false);
}
```

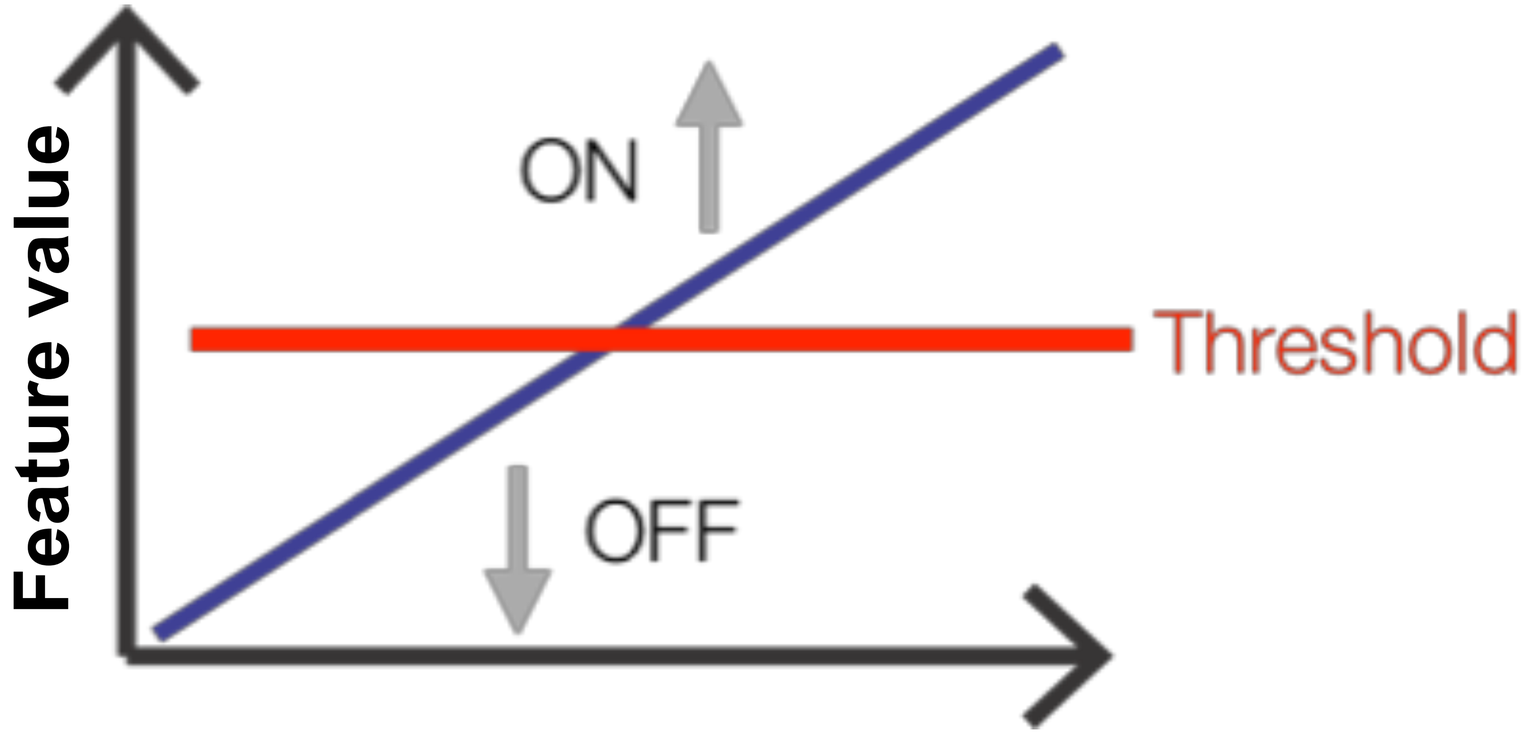
Command list:
CMD_LEFT (=left key),
CMD_RIGHT (=right key),
CMD_DOWN (=down key)

CMD_JUMP (=z key)
CMD_DASH (=x key)

```
if(ypr[2] > 0.2)
{
  press(CMD_RIGHT);
}
else if(ypr[2] < -0.2)
{
  press(CMD_LEFT);
}
else
{
  release(CMD_LEFT);
  release(CMD_RIGHT);
}
```

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

Thresholding



Simple thresholding

Let Mario runs back/forward with X-axis accelerometer.
Please modify **processCommand()** function

* ypr[2] → roll

```
if(ypr[2] > 0.2)
{
  press(CMD_RIGHT);
}
else if(ypr[2] < -0.2)
{
  press(CMD_LEFT);
}
else
{
  release(CMD_LEFT);
  release(CMD_RIGHT);
}
```

Thresholding

Thresholding

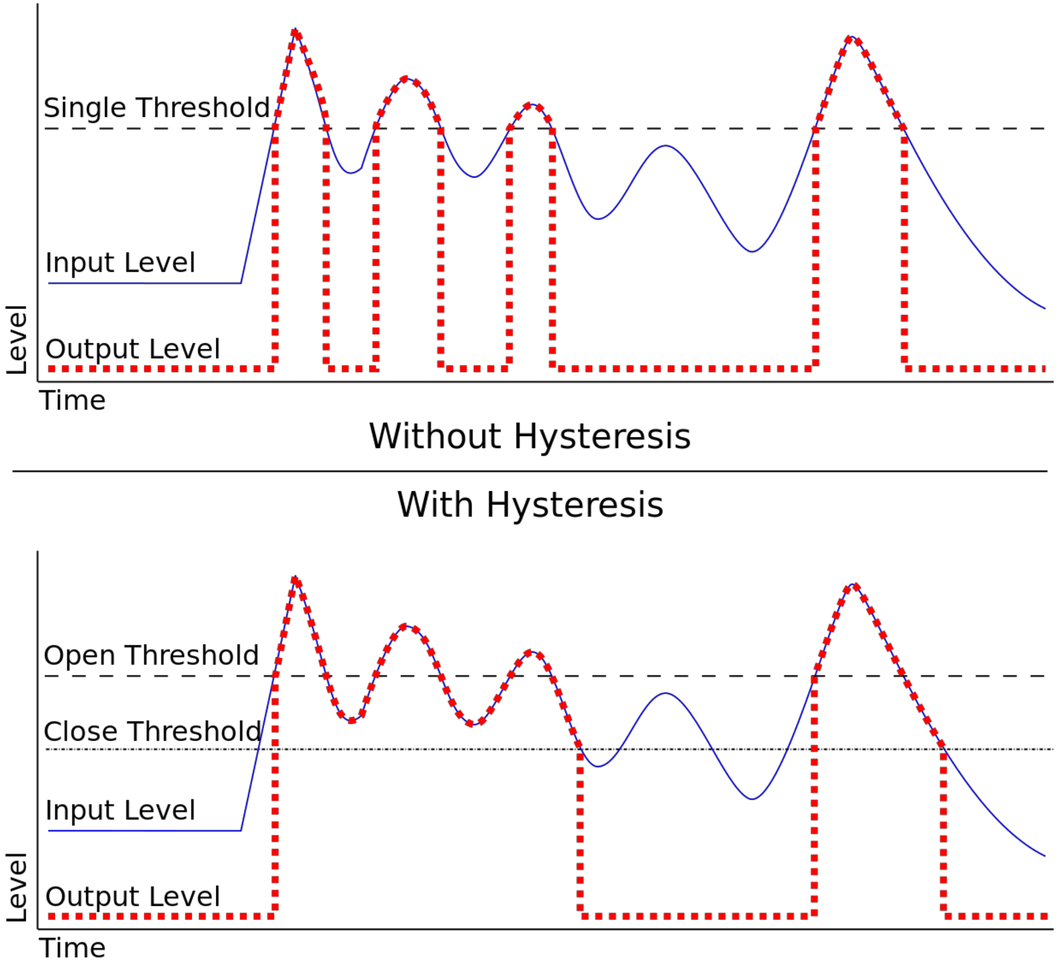
Release keys after use

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

Hysteresis

Simple thresholding suffers by "chatter"

Two-threshold (=hysteresis) can stabilize the input



By Iainf - Own work, CC BY 3.0, https://commons.wikimedia.org/w/index.php?curid=5750524

Thresholding with Hysteresis

Let Mario runs back/forward with X-axis accelerometer.
Please modify **processCommand()** function

* global variables

```
float thre_low = 0.2;
float thre_high = 0.4;
Boolean pressed_right = false;
Boolean pressed_left = false;

void processCommand()
{
```

processCommand()

```
if(ypr[2] > thre_high)
{
  press(CMD_RIGHT);
  pressed_right = true;
}
else if(ypr[2] < -1 * thre_high)
{
  press(CMD_LEFT);
  pressed_left = true;
}
else if((pressed_right || pressed_left) && abs(ypr[2]) < thre_low)
{
  release(CMD_LEFT);
  release(CMD_RIGHT);
  pressed_left = false;
  pressed_right = false;
}
```

/Programs/Processing/GY_521_Skeleton/GY_521_Skeleton.pde

Assignment

Complete your controller to clear the first stage.

Report your method and encountered challenges.