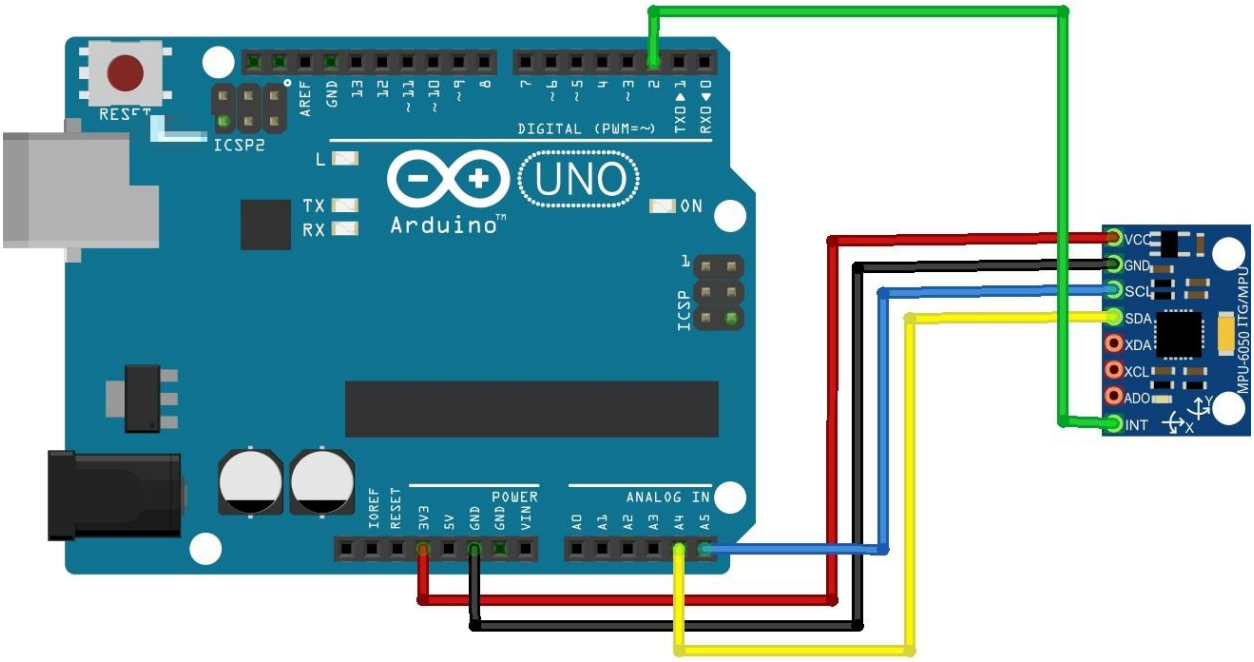
Connecting MPU-6050/GY-521 to Arduino UNO

Note: All the libraries and files are added to shared Google Drive, for your convenience.

Step 1: Arduino connections

Here are schematics and graphics, please notice that you have to connect the power to 3.3V.

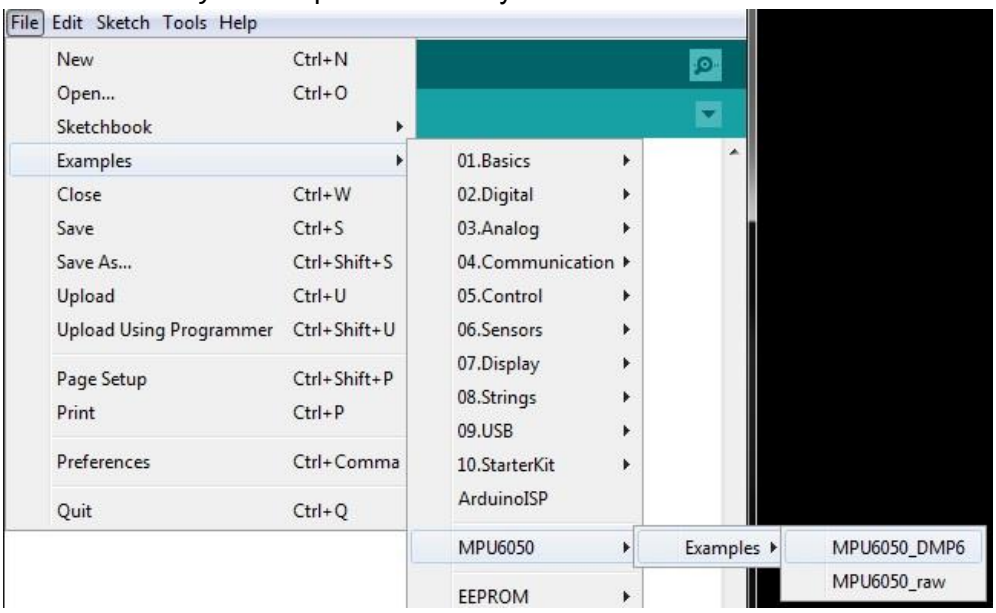


MPU-6050/GY-521	Arduino UNO
VCC	3.3V
GND	GND
SCL	A5
SDA	A4
INT	2

Step 2: Upload the code and test

1. Download the Arduino Library for MPU-6050/GY-521, extract the library. Copy/Cut the folder ‘MPU6050’ and paste it in C:\Users\UserName\Documents\Arduino\libraries (Windows OS), or in the Arduino’s library folder (for other OS).

You should see the library’s examples added to your menu.

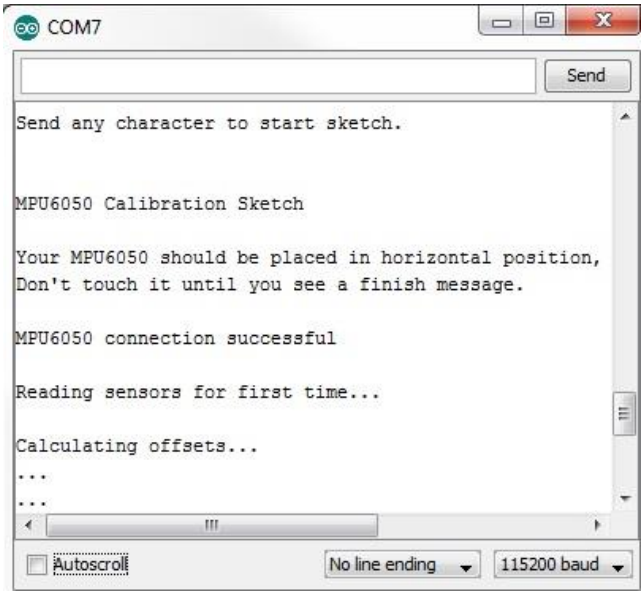


The other way to do this after you download the zip file, without extracting it, go to Sketch > Import Library... > Add Library... > Select the location of MPU6050.zip file.

Add I2Cdev library following the same instructions.

2. Before we start, let’s calibrate MPU6050. Open AccelGyro_calibration file and upload it to Arduino.

Next, open up a Serial Monitor and set the baud rate to 115200. Make sure, the sensor is placed on leveled surface in horizontal position, with package letters facing up, and don't touch it until you see a finish message.



```
FINISHED!

Sensor readings with offsets:   -5      8       16374   0       -1      1
Your offsets:   -2465   1342    1341    -4      23      -9

Data is printed as: acelX acelY acelZ giroX giroY giroZ
Check that your sensor readings are close to 0 0 16384 0 0 0
If calibration was succesful write down your offsets so you can set them i
```

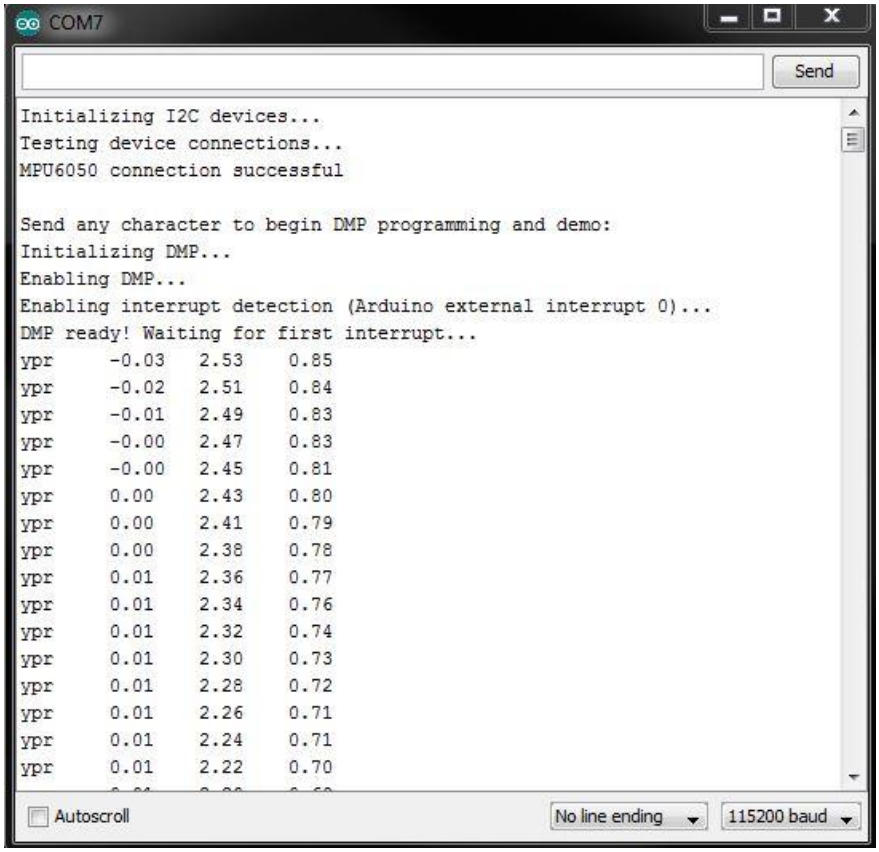
Make sure you note the offsets and close the sketch.

3. Open the example program from File > Examples > MPU6050 > Examples > MPU6050_DMP6.

Inside the code, supply your own offsets from the calibration, make sure you type them in the right field.

```
// load and configure the DMP
Serial.println(F("Initializing DMP..."));
devStatus = mpu.dmpInitialize();

// supply your own gyro offsets here, scaled for min sensitivity
mpu.setXGyroOffset(-3);
mpu.setYGyroOffset(23);
mpu.setZGyroOffset(-9);
mpu.setXAccelOffset(-2482);
mpu.setYAccelOffset(1303);
mpu.setZAccelOffset(1343);

// make sure it worked (returns 0 if so)
if (devStatus == 0) {
    // turn on the DMP, now that it's ready
    Serial.println(F("Enabling DMP..."));
    mpu.setDMPEnabled(true);

    // enable Arduino interrupt detection
    Serial.println(F("Enabling interrupt detection (Arduino external interrupt 0)..."));
    attachInterrupt(0, dmpDataReady, RISING);
    mpuIntStatus = mpu.getIntStatus();

    // set our DMP Ready flag so the main loop() function knows it's okay to use it
    Serial.println(F("DMP ready! Waiting for first interrupt..."));
    dmpReady = true;
```

After all is set, run Serial Monitor (115200 baud). At this point you should see values coming in from MPU-6050/GY-521.



We read the values at yaw, pitch, roll:

- Rotation around the vertical axis is called yaw.
- Rotation around the side-to-side axis is called pitch.
- Rotation around the front-to-back axis is called roll.

Step 3: Model values from MPU-6050 using processing.

Download processing IDE. Open up MPU6050_DMP6 from examples library. Uncomment *#define OUTPUT_READABLE_YAWPITCHROLL* by *//#define OUTPUT_READABLE_YAWPITCHROLL.*

Uncomment: *//#define OUTPUT_TEAPOT* by *#define OUTPUT_TEAPOT*.

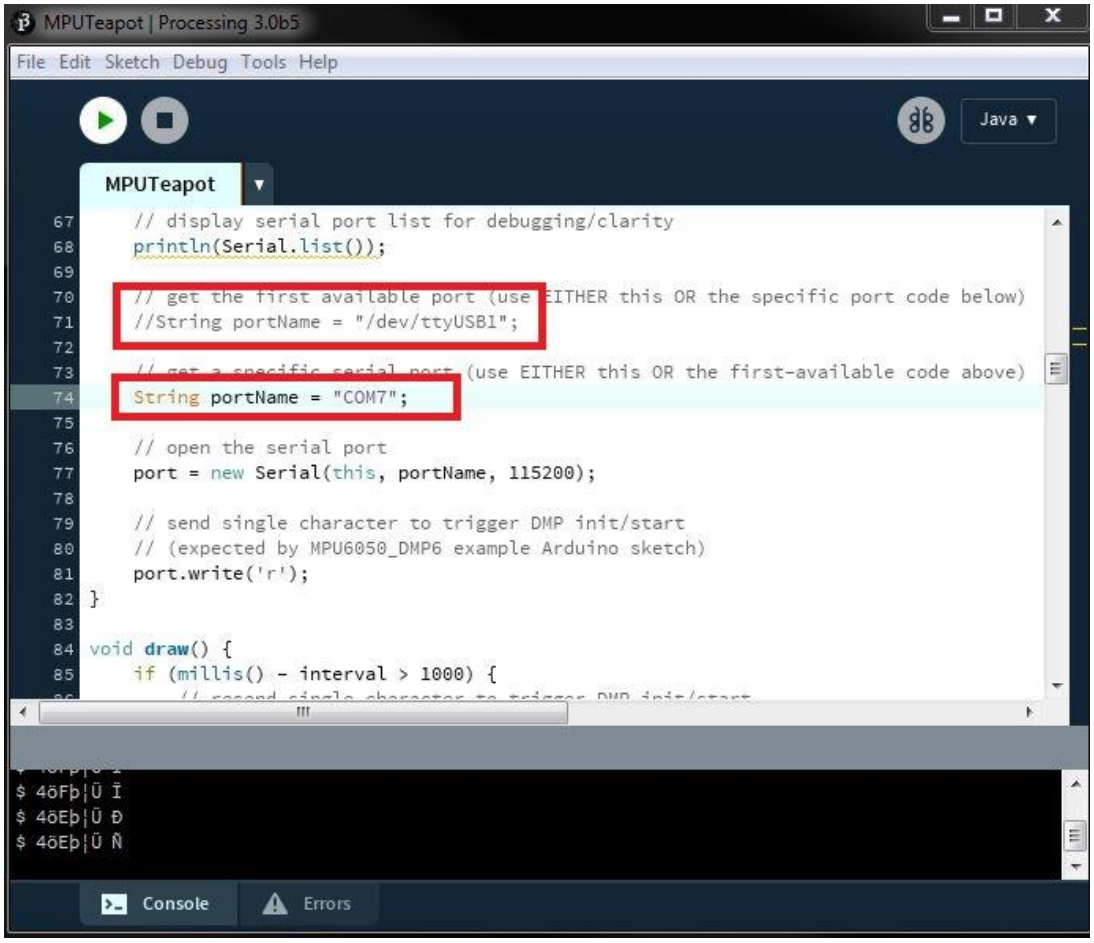
```
// uncomment "OUTPUT_READABLE_QUATERNION" if you want to see the actual
// quaternion components in a [w, x, y, z] format (not best for parsing
// on a remote host such as Processing or something though)
//#define OUTPUT_READABLE_QUATERNION

// uncomment "OUTPUT_READABLE_EULER" if you want to see Euler angles
// (in degrees) calculated from the quaternions coming from the FIFO.
// Note that Euler angles suffer from gimbal lock (for more info, see
// http://en.wikipedia.org/wiki/Gimbal_lock)
//#define OUTPUT_READABLE_EULER

// uncomment "OUTPUT_READABLE_YAWPITCHROLL" if you want to see the yaw/
// pitch/roll angles (in degrees) calculated from the quaternions coming
// from the FIFO. Note this also requires gravity vector calculations.
// Also note that yaw/pitch/roll angles suffer from gimbal lock (for
// more info, see: http://en.wikipedia.org/wiki/Gimbal_lock)
//#define OUTPUT_READABLE_YAWPITCHROLL

// uncomment "OUTPUT_READABLE_REALACCEL" if you want to see acceleration
// components with gravity removed. This acceleration reference frame is
// not compensated for orientation, so +X is always +X according to the
// sensor, just without the effects of gravity. If you want acceleration
// compensated for orientation, us OUTPUT_READABLE_WORLDACCEL instead.
//#define OUTPUT_READABLE_REALACCEL

// uncomment "OUTPUT_READABLE_WORLDACCEL" if you want to see acceleration
// components with gravity removed and adjusted for the world frame of
// reference (yaw is relative to initial orientation, since no magnetometer
// is present in this case). Could be quite handy in some cases.
//#define OUTPUT_READABLE_WORLDACCEL

// uncomment "OUTPUT_TEAPOT" if you want output that matches the
// format used for the InvenSense teapot demo
#define OUTPUT_TEAPOT
```

Upload the sketch to Arduino. Then open processing example for the MPU-6050. File > Open > navigate the folder where your MPU6050 library for the Arduino (Windows: C:\Users\User_Name\Documents\Arduino\libraries\MPU6050\Examples\MPU6050_DMP6\Processing\MPUTeapot).

Inside the code, change the values (notice, you're not in Arduino software now, you're in Processing):

Comment - *String portName = "/dev/ttyUSB1"* by //String portName = "/dev/ttyUSB1";

And uncomment - //String portName = "COM4"; by *String portName = "COM4*
Replace COM4 with COM port on which your Arduino is connected (Tools > Serial Port).



Download toxiclibs-complete-0020 folder and copy it to Processing libraries folder (Windows: C:\Users\UserName\Documents\Processing\libraries).

You are ready to run processing code by clicking play symbol. Wait for about 10 seconds for the sensor to get stabilized, after that you can see 3D model of your MPU-6050/GY521.