```
import smbus                                    #import SMBus module of I2C
from time import sleep          #import
import RPi.GPIO as GPIO
import time as time
#some MPU6050 Registers and their Address
PWR_MGMT_1   = 0x6B
SMPLRT_DIV   = 0x19
CONFIG       = 0x1A
GYRO_CONFIG  = 0x1B
INT_ENABLE   = 0x38
ACCEL_XOUT_H = 0x3B
ACCEL_YOUT_H = 0x3D
ACCEL_ZOUT_H = 0x3F
GYRO_XOUT_H  = 0x43
GYRO_YOUT_H  = 0x45
GYRO_ZOUT_H  = 0x47
rotationanglex = 0
rotationangley = 0
rotationanglez = 0

m1pin1 = 18
m1pin2 = 23
GPIO.setmode(GPIO.BCM)
GPIO.setup(17,GPIO.OUT)
GPIO.setup(m1pin1,GPIO.OUT)
GPIO.setup(27,GPIO.OUT)
GPIO.setup(22,GPIO.OUT)
def MPU_Init():
        #write to sample rate register
        bus.write_byte_data(Device_Address, SMPLRT_DIV, 7)

        #Write to power management register
        bus.write_byte_data(Device_Address, PWR_MGMT_1, 1)

        #Write to Configuration register
        bus.write_byte_data(Device_Address, CONFIG, 0)

        #Write to Gyro configuration register
        bus.write_byte_data(Device_Address, GYRO_CONFIG, 24)

        #Write to interrupt enable register
        bus.write_byte_data(Device_Address, INT_ENABLE, 1)

def read_raw_data(addr):
        #Accelero and Gyro value are 16-bit
    high = bus.read_byte_data(Device_Address, addr)
    low = bus.read_byte_data(Device_Address, addr+1)

    #concatenate higher and lower value
    value = ((high << 8) | low)

    #to get signed value from mpu6050
    if(value > 32768):
            value = value - 65536
    return value
```

```
bus = smbus.SMBus(1) # or bus = smbus.SMBus(0) for older version boards
Device_Address = 0x68   # MPU6050 device address

MPU_Init()

print (" Reading Data of Gyroscope and Accelerometer")

while True:

        #Read Accelerometer raw value
        acc_x = read_raw_data(ACCEL_XOUT_H)
        acc_y = read_raw_data(ACCEL_YOUT_H)
        acc_z = read_raw_data(ACCEL_ZOUT_H)

        #Read Gyroscope raw value
        gyro_x = read_raw_data(GYRO_XOUT_H)
        gyro_y = read_raw_data(GYRO_YOUT_H)
        gyro_z = read_raw_data(GYRO_ZOUT_H)

        #Full scale range +/- 250 degree/C as per sensitivity scale factor
        Ax = acc_x/16384.0
        Ay = acc_y/16384.0
        Az = acc_z/16384.0

        Gx = gyro_x/131.0
        Gy = gyro_y/131.0
        Gz = gyro_z/131.0

        rotationanglez = rotationanglez + (Gz-0.089)/11
        if abs(Gy)>1.75:
                Gy=-0.02
        rotationangley = rotationangley + (Gy-0.02)/11
        rotationanglex = rotationanglex + (Gx+0.242)/11
        print(rotationanglex,rotationangley,rotationanglez)
        sleep(0.0005)
        if (rotationanglex) < -5 :
                GPIO.output(17,GPIO.HIGH)
                GPIO.output(27,GPIO.HIGH)
                GPIO.output(22,GPIO.LOW)
        elif (rotationanglex) >   5 :
                GPIO.output(17,GPIO.HIGH)
                GPIO.output(27,GPIO.LOW)
                GPIO.output(22,GPIO.HIGH)
        else:
                GPIO.output(17,0)
                GPIO.output(27,0)
                GPIO.output(22,0)
```