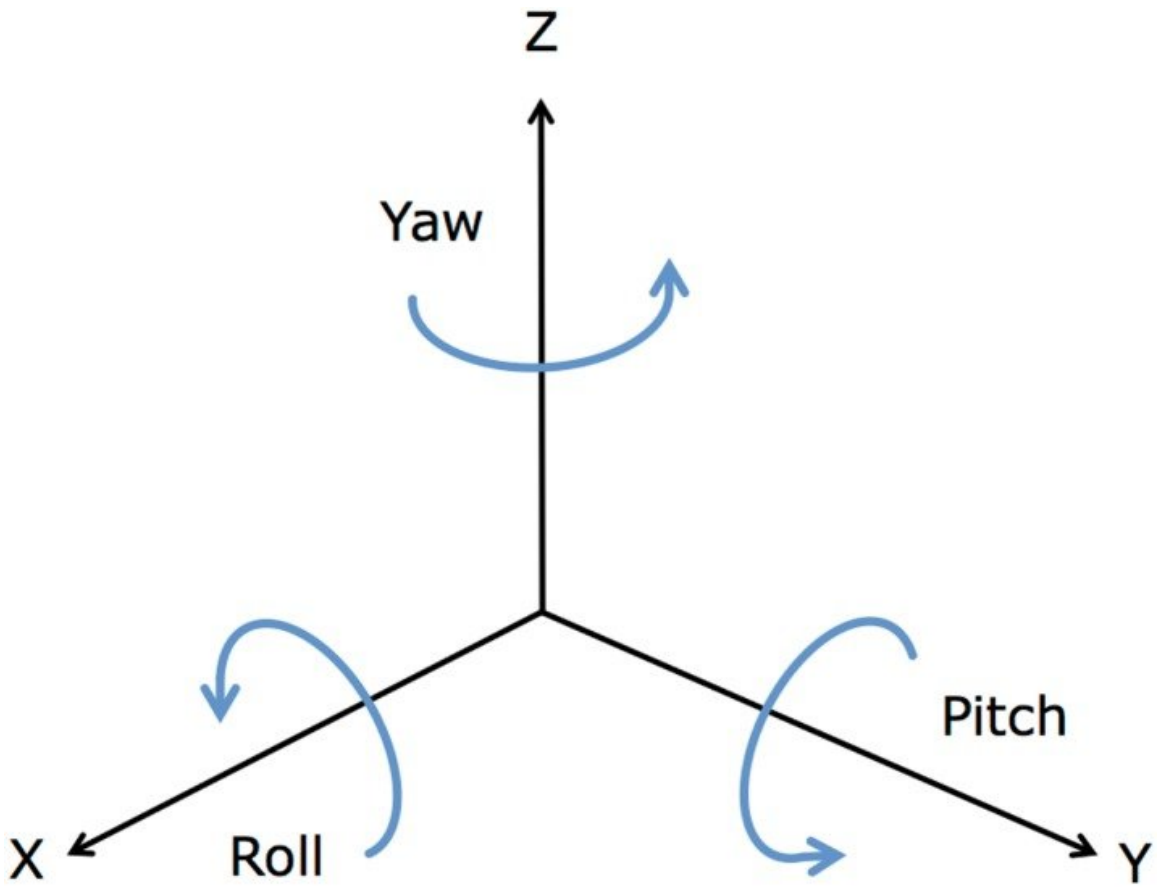
IMU Documentation

YAW angle

this documentation is made to understand the IMU code to get the yaw angle

the Yaw angle is the angle perpindicular to the Z-axis



Looking at the photo we can understand the angle and its axis

Coding:

Next we are going to get the code and document it.

```
#define IMU_ADD 0X68
#define Gyro_XOUT_H 0x43
#define MPU6050_GYRO_CONFIG_VALUE 0x08
#define MPU6050_REG_GYRO_CONFIG 0x1B
```

Here we are identifing the adresses of the IMU to use in our code including

1. the adress of the IMU from the data sheet
2. the axis axis output
3. the configuration value
4. the register configuration

```
void MPU_readGyro(){
Wire.beginTransmission(IMU_ADD);
Wire.write(Gyro_XOUT_H);
Wire.endTransmission();
Wire.requestFrom(IMU_ADD, 6);
while(Wire.available()<6);
gyro_X= Wire.read()<<8 | Wire.read();
gyro_Y= Wire.read()<<8 | Wire.read();
gyro_Z = Wire.read() << 8 | Wire.read();

}
```

this part of the code we made a function to read the values of the IMU in x or y or z.

and we read the values using i2c and getting data bits from the IMU

```
while(Wire.available()<6);
gyro_X= Wire.read()<<8 | Wire.read();
gyro_Y= Wire.read()<<8 | Wire.read();
gyro_Z = Wire.read() << 8 | Wire.read();
```

this part of code is used because the readings is 16 bits and we only have 8 bits so we increased the size of the number of bits we can read

```
void MPU_Init() {
  // Wake up the MPU6050 and configure gyro sensitivity
  Wire.beginTransmission(IMU_ADD);
  Wire.write(0x6B); // PWR_MGMT_1 register
  Wire.write(0);    // Clear sleep mode bit (activate device)
  Wire.endTransmission();

  Wire.beginTransmission(IMU_ADD);
  Wire.write(MPU6050_REG_GYRO_CONFIG);
  Wire.write(MPU6050_GYRO_CONFIG_VALUE);
  Wire.endTransmission();
}
```

here this is a code to wake the imu and configure the sensitivity and to begin its transmission the send data to the arduino

```
void setup() {
Wire.begin();
  Serial.begin(9600);
  MPU_Init();

}

void loop() {
MPU_readGyro();
//the delta here shows the interval i want to read the data in which here is 50ms
double deltaT=0.05;
//using integration which is summation of the changes in velocity
yaw +=(gyro_Z*deltaT);
 Serial.print("Yaw Angle: ");
    Serial.println(yaw);

    delay(50);
}
```

this is our main code consisting of the setup and loop of the code.

- in the setup i used the MPU_init function to start the coding

- in the main loop we start by reading the gyro data from the function that we made.Then we use delta to mulltiply it with the gyro Z data
- the yaw is the summation of the small changes in velocity (Z-axis) and the Delta is the interval i want to read the data in.

resolving the noise

- The one we are going to use is known as **complementary filter**. Idea behind complementary filter is to take slow moving signals from accelerometer and fast moving signals from a gyroscope and combine them. It is ideal to implement with Arduino: easy to use, low cost of processing and good precision.
- also the datasheet recommends the use of the low pass filter and that it is programmable

Cutoff frequency:

in the data sheet the bandwidth is between 1 Hz and 10Hz and the cutoff frequency is 10Hz