**Bölüm: Kendi kendini dengeleyen robotun açısının bulunması**

Arduino Kodu:

#include <Wire.h>

const int MPU **=** 0x68**;** // MPU6050 I2C address

float AccX**,** AccY**,** AccZ**,** GyroX**,** GyroY**,** GyroZ**;**

float accAngleX**,** accAngleY**,** gyroAngleX**,** gyroAngleY**,** angleX**,** angleY**;**

float AccErrorX**,** AccErrorY**,** GyroErrorX**,** GyroErrorY**;**

float elapsedTime**,** currentTime**,** previousTime**;**

int c **=** 0**;**

struct IMU **{**

float angleX**,** angleY**;**

unsigned long timeStamp**;**

**};**

IMU imu**;**

void setup**()** **{**

Serial**.**begin**(**57600**);**

initialize\_MPU6050**();**

// Call this function if you need to get the IMU error values for your module

//calculate\_IMU\_error();

**}**

void loop**()** **{**

read\_IMU**();**

//Serial.print(imu.angleX); Serial.print(' '); Serial.print(imu.angleY); Serial.print(' '); Serial.println(imu.timeStamp);;

Serial**.**write**(**'h'**);** Serial**.**write**((**byte**\*)(&**imu**),** **sizeof(**imu**));**

**}**

void initialize\_MPU6050**()** **{**

Wire**.**begin**();** // Initialize comunication

Wire**.**beginTransmission**(**MPU**);** // Start communication with MPU6050 // MPU=0x68

Wire**.**write**(**0x6B**);** // Talk to the register 6B

Wire**.**write**(**0x00**);** // Make reset - place a 0 into the 6B register

Wire**.**endTransmission**(true);** //end the transmission

// Configure Accelerometer

Wire**.**beginTransmission**(**MPU**);**

Wire**.**write**(**0x1C**);** //Talk to the ACCEL\_CONFIG register

Wire**.**write**(**0x10**);** //Set the register bits as 00010000 (+/- 8g full scale range)

Wire**.**endTransmission**(true);**

// Configure Gyro

Wire**.**beginTransmission**(**MPU**);**

Wire**.**write**(**0x1B**);** // Talk to the GYRO\_CONFIG register (1B hex)

Wire**.**write**(**0x10**);** // Set the register bits as 00010000 (1000dps full scale)

Wire**.**endTransmission**(true);**

**}**

void calculate\_IMU\_error() {

// We can call this funtion in the setup section to calculate the accelerometer and gury data error. From here we will get the error values used in the above equations printed on the Serial Monitor.

// Note that we should place the IMU flat in order to get the proper values, so that we then can the correct values

// Read accelerometer values 200 times

while (c < 200) {

Wire.beginTransmission(MPU);

Wire.write(0x3B);

Wire.endTransmission(false);

Wire.requestFrom(MPU, 6, true);

AccX = (Wire.read() << 8 | Wire.read()) / 4096.0 ;

AccY = (Wire.read() << 8 | Wire.read()) / 4096.0 ;

AccZ = (Wire.read() << 8 | Wire.read()) / 4096.0 ;

// Sum all readings

AccErrorX = AccErrorX + ((atan((AccY) / sqrt(pow((AccX), 2) + pow((AccZ), 2))) \* 180 / PI));

AccErrorY = AccErrorY + ((atan(-1 \* (AccX) / sqrt(pow((AccY), 2) + pow((AccZ), 2))) \* 180 / PI));

c++;

}

//Divide the sum by 200 to get the error value

AccErrorX = AccErrorX / 200;

AccErrorY = AccErrorY / 200;

c = 0;

// Read gyro values 200 times

while (c < 200) {

Wire.beginTransmission(MPU);

Wire.write(0x43);

Wire.endTransmission(false);

Wire.requestFrom(MPU, 4, true);

GyroX = Wire.read() << 8 | Wire.read();

GyroY = Wire.read() << 8 | Wire.read();

// Sum all readings

GyroErrorX = GyroErrorX + (GyroX / 32.8);

GyroErrorY = GyroErrorY + (GyroY / 32.8);

c++;

}

//Divide the sum by 200 to get the error value

GyroErrorX = GyroErrorX / 200;

GyroErrorY = GyroErrorY / 200;

// Print the error values on the Serial Monitor

Serial.print("AccErrorX: ");

Serial.println(AccErrorX);

Serial.print("AccErrorY: ");

Serial.println(AccErrorY);

Serial.print("GyroErrorX: ");

Serial.println(GyroErrorX);

Serial.print("GyroErrorY: ");

Serial.println(GyroErrorY);

}

void read\_IMU() {

// === Read acceleromter data === //

Wire.beginTransmission(MPU);

Wire.write(0x3B); // Start with register 0x3B (ACCEL\_XOUT\_H)

Wire.endTransmission(false);

Wire.requestFrom(MPU, 6, true); // Read 6 registers total, each axis value is stored in 2 registers

//For a range of +-8g, we need to divide the raw values by 4096, according to the datasheet

AccX = (Wire.read() << 8 | Wire.read()) / 4096.0; // X-axis value

AccY = (Wire.read() << 8 | Wire.read()) / 4096.0; // Y-axis value

AccZ = (Wire.read() << 8 | Wire.read()) / 4096.0; // Z-axis value

// Calculating angle values using

accAngleX = (atan(AccY / sqrt(pow(AccX, 2) + pow(AccZ, 2))) \* 180 / PI) + 1.15; // AccErrorX ~(-1.15) See the calculate\_IMU\_error()custom function for more details

accAngleY = (atan(-1 \* AccX / sqrt(pow(AccY, 2) + pow(AccZ, 2))) \* 180 / PI) - 0.52; // AccErrorX ~(0.5)

// === Read gyro data === //

previousTime = currentTime; // Previous time is stored before the actual time read

imu.timeStamp = millis(); // Current time actual time read

currentTime = float(imu.timeStamp);

elapsedTime = (currentTime - previousTime) / 1000; // Divide by 1000 to get seconds

Wire.beginTransmission(MPU);

Wire.write(0x43); // Gyro data first register address 0x43

Wire.endTransmission(false);

Wire.requestFrom(MPU, 4, true); // Read 4 registers total, each axis value is stored in 2 registers

GyroX = (Wire.read() << 8 | Wire.read()) / 32.8; // For a 1000dps range we have to divide first the raw value by 32.8, according to the datasheet

GyroY = (Wire.read() << 8 | Wire.read()) / 32.8;

GyroX = GyroX + 1.85; //// GyroErrorX ~(-1.85)

GyroY = GyroY - 0.15; // GyroErrorY ~(0.15)

// Currently the raw values are in degrees per seconds, deg/s, so we need to multiply by sendonds (s) to get the angle in degrees

gyroAngleX = GyroX \* elapsedTime;

gyroAngleY = GyroY \* elapsedTime;

// Complementary filter - combine acceleromter and gyro angle values

imu.angleX = 0.98 \* (imu.angleX + gyroAngleX) + 0.02 \* accAngleX;

imu.angleY = 0.98 \* (imu.angleY + gyroAngleY) + 0.02 \* accAngleY;

}

MATLAB kodu:

clear all**;** close all**;** clc**;**

s **=** serialport**(**'COM4'**,** 57600**);** flush**(**s**);**

stopTime **=** 15**;**

rollAngle **=** single**(**0.0**);** pitchAngle **=** single**(**0.0**);** time **=** 0**;**

i **=** 0**;** % package number

**while** **(** time**(**end**)** **<** stopTime **)**

**if** **(**read**(**s**,** 1**,** 'uint8'**)** **==** 'h'**)**

i **=** i **+** 1**;**

rollAngle**(**1**,**i**)** **=** read**(**s**,** 1**,** 'single'**);**

pitchAngle**(**1**,**i**)** **=** read**(**s**,** 1**,** 'single'**);**

time**(**i**)** **=** double**(**read**(**s**,** 1**,** 'uint32'**))** **/** 1000**;**

fprintf**(**'paket = %i roll = %.2f pitch = %.2f zaman = %.2f\n'**,** ...

i**,** rollAngle**(**1**,**i**),** pitchAngle**(**1**,**i**),** time**(**i**));**

**end**

**end**

delete**(**s**);** clear s**;**

%% plot the logged angles

figure**(**1**);**

subplot**(**1**,**2**,**1**);**

plot**(**time**,** rollAngle**,** 'k-'**);**

xlabel**(**'time (s)'**);** ylabel**(**'roll angle (degrees)'**);**

grid on**;**

subplot**(**1**,**2**,**2**);**

plot**(**time**,** pitchAngle**,** 'k-'**);**

xlabel**(**'time (s)'**);** ylabel**(**'pitch angle (degrees)'**);**

grid on**;**



**Şekil 1:** MPU6050 hareket sensöründen gelen verilerle Arduino üzerinde iki açının bulunmasından sonra Arduino’dan MATLAB’a seri port aracılığıyla hesaplanan iki açının ve zaman verisinin yollanması ve çizdirilmesi.