Arduino – stabilizator giroscopic

In acest tutorial iti propun o varianta simpla de stabilizator giroscopic, ce se poate construi cu urmatoarele componente:

- o placa Arduino
- un accelerometru+giroscop MPU6050
- un servomotor Medium
- o sursa de alimentare ce poate fi un acumulator Li-PO de 7.4V
- un breadboard
- fire de conexiune

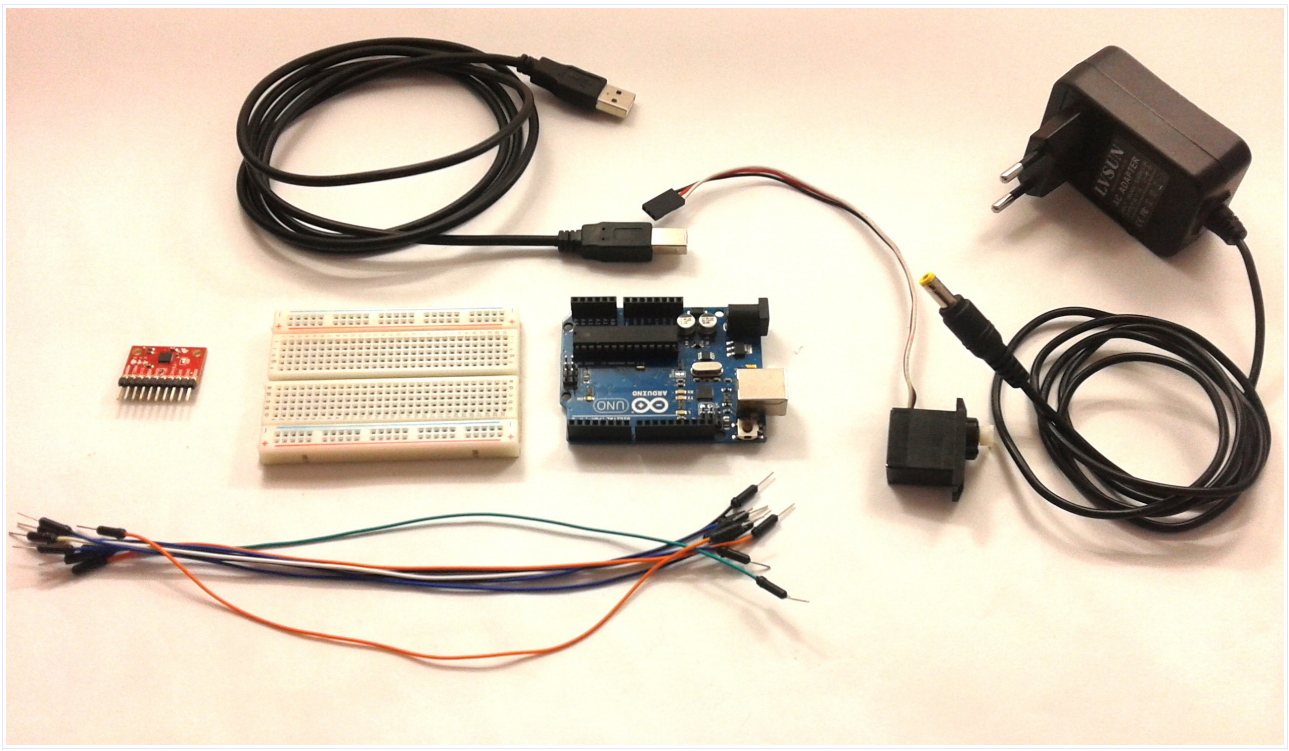
La ce se poate utiliza un stabilizator giroscopic ?

Stabilizatorul giroscopic se poate utiliza pentru a echilibra o camera de filmat, pentru a echilibra o drona ce se afla in zbor, un avion rc sau un robot de tip Segway. Am mentionat numai cateva exemple, dar in continuare vei descoperi cum se poate construi un sistem simplu avand un singur servomotor.

Stabilizatoarele giroscopice performante utilizeaza o gama variata de motoare/servomotoare/motoare pas cu pas, in functie de aplicatie.

Pentru a demonstra principiul, in acest tutorial s-a folosit un singur servomotor care este suficient cat sa echilibreze o camera web, spre exemplu.

Componentele minime necesare pentru stabilizator:



Ce este giroscopul ?

Giroscopul, in termeni simpli, este un mic dispozitiv folosit la masurarea si mentinerea orientatiei. In ziua de azi, aproape orice telefon mobil are un mic giroscop pe care il utilizeaza atunci cand se schimba pozitia ecranului din modul Portrait in Landscape. Avioanele, elicopterele sau aparatele de zbor in general, utilizeaza giroscoape ce le ajuta la orientarea in spatiu.

Cum construiesti stabilizatorul giroscopic ?

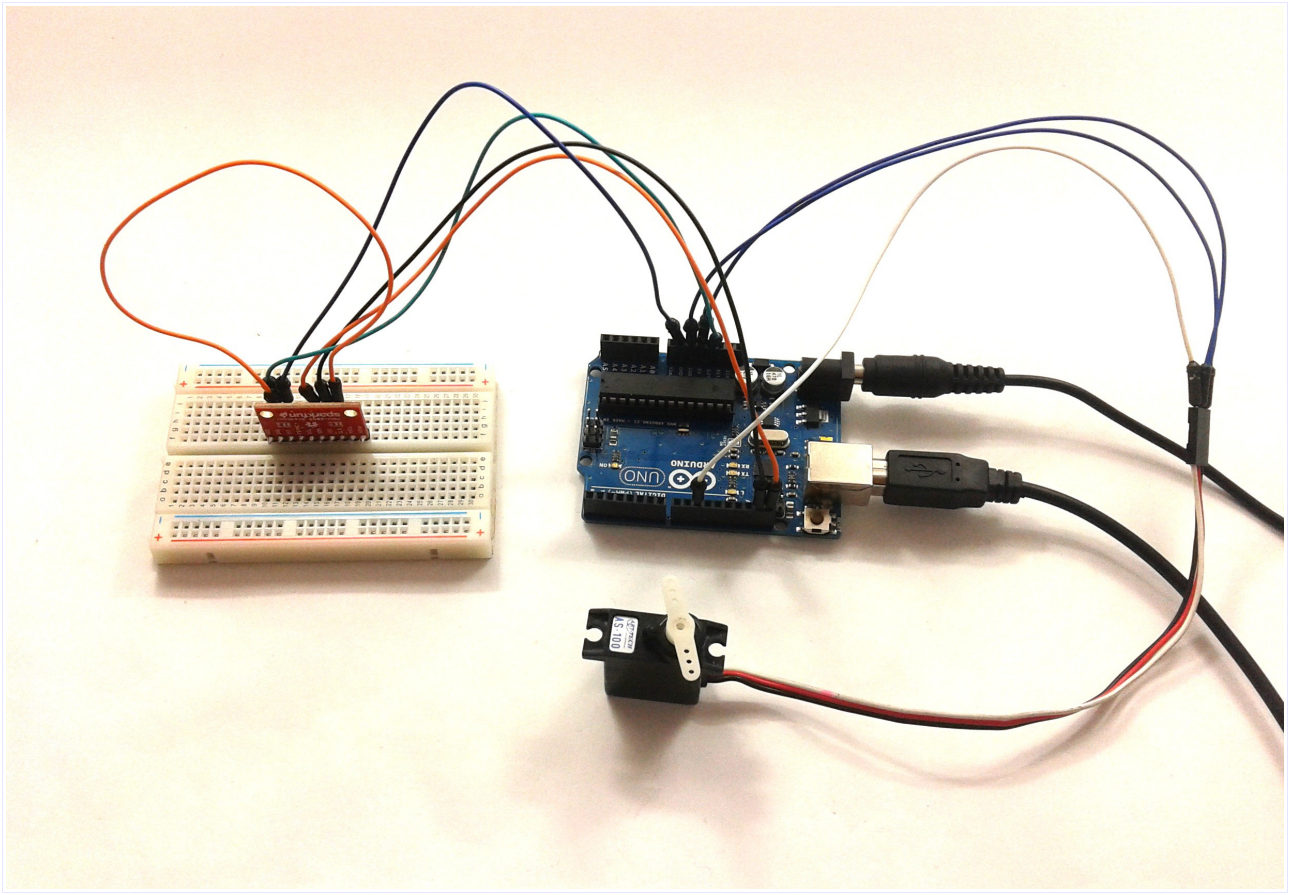
Avand componentele de mai sus, urmeaza sa le conectezi folosind diagrama de mai jos. Poti alimenta servomotorul fie din placa Arduino, fie dintr-un stabilizator de 5V separat, in functie de servomotorul pe care l-ai ales si consumul acestuia.

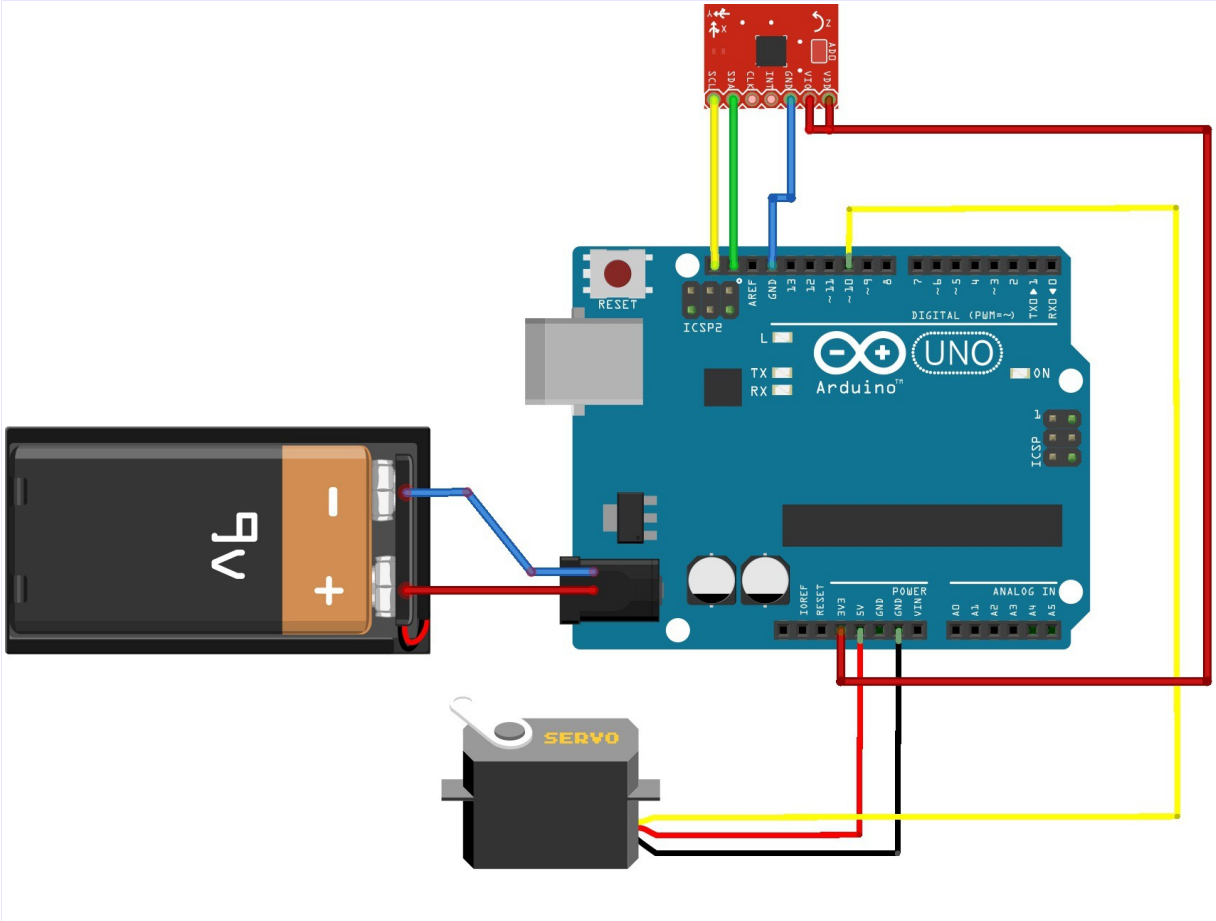
Placuta accelerometru+giroscop se conecteaza la placa Arduino dupa urmatorul tabel:

Arduino pin 3.3V	MPU 6050 pin VDD
Arduino pin 3.3V	MPU 6050 pin VIO
Arduino pin GND	MPU 6050 pin GND
Arduino pin SDA	MPU 6050 pin SDA
Arduino pin SCL	MPU 6050 pin SCL

Majoritatea servomotoarelor necesita alimentare printr-un stabilizator de 5V separat. Poti opta pentru stabilizatorul de aici http://www.robofun.ro/bricks/stabilizator-5v (cel din imaginea de mai sus).

Dupa ce ai realizat diagrama cu componentele giroscopului, vei obtine o schema asemanatoare ca in imagine:




RESET
ICSP2
AREF
GND
DIGITAL (PWM=~)
TXD
RXD
L
TX
RX
UNO
Arduino
ON
ICSP
9V
SERVO
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5

Ceea ce vezi in imagine sunt doar componentele de baza, iar tu vei avea nevoie de asemenea de un cadru de sustinere, o carcasa sau o constructie mecanica simpla care poate fi utilizata la ceva anume.

Un exemplu despre cum poti construi un robot care se echilibreaza singur pe 2 roti este aici:

http://letsmakerobots.com/node/1505

Un exemplu despre cum poti construi un robot care sa se echilibreze singur pe o minge, prin intermediul a mai multor roti este aici:

http://spectrum.ieee.org/automaton/robotics/robotics-software/042910-a-robot-that-balances-on-a-ball

Spre exemplu, folosind mai multe servomotoare (sau chiar si motoare) se poate construi un robot care isi poate mentine echilibrul pe o minge. Acest tip de robot nu implica doar utilizarea unui giroscop dar si a unui accelerometru. Utilizarea celor 2 senzori in paralel presupune fiziunea acestora intr-o unitate de masura inertiala sau IMU.

In functie de robotul pe care doresti sa il realizezi, exista o gama variata de senzori IMU cu grade de libertate de 6 sau 9:

http://www.robofun.ro/senzori/imu

Pentru tutorialul de fata este absolut necesar sa studiezi catusi de putin cum se implementeaza o unitate de masura inertiala (IMU) si faptul ca acest lucru implica utilizarea unor filtre (Kalman fiind unul foarte des utilizat).

Un material de studiu foarte util despre fuziunea senzorilor se afla aici:

http://web.mit.edu/scolton/www/filter.pdf

Pentru a simplifica lucrurile, afla ca exista deja implementari cu privire la filtrul Kalman daca accesezi link-ul de mai jos:

https://github.com/TKJElectronics/KalmanFilter

Mai multe detalii despre implementarea filtrului Kalman (libraria de mai sus) pe o placa Arduino se afla aici:

http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/

Revenind la exemplul stabilizarii unei camere, nu avem nevoie decat de un servomotor. In codul Arduino vei include libraria si vei declara un obiect servoToControl:

```
#include <Servo.h>
Servo servoToControl;
```

Pentru filtrul Kalman vei include libraria si vei declara 2 obiecte kalmanX si kalmanY:

```
#include "Kalman.h"
Kalman kalmanX;
Kalman kalmanY;
```

Pentru placa MPU-6050 vei include libraria <Wire> care permite placii sa comunice prin magistrala I2C si vei declara cateva variabile cum ar fi adresa giroscopului si locatiile de memorie prin care vei citi valorile giroscopului si ale accelerometrului.

```
#include <Wire.h>
uint8_t IMUAddress = 0x68; // adresa placutei MPU6050
int16_t accX;
int16_t accY;
int16_t accZ;
int16_t tempRaw;
int16_t gyroX;
int16_t gyroY;
int16_t gyroZ;
```

In rutina setup() vei initializa servomotorul atasandu-l la pinul 10 de pe placa Arduino si vei porni monitorul serial la viteza de 115200 baud (prin care poti sa vizualizezi informatii sau datele de iesire):

```
servoToControl.attach(10);
Serial.begin(115200);
```

Stabilizarea giroscopica

Stabilizarea presupune citirea accelerometrului si a giroscopului din placa MPU-6050, calculul tangajului si a ruliului (pitch and roll) prin functii trigonometrice inverse si filtrarea valorilor pentru o corectie buna.

In rutina loop() placa Arduino va citi acceleratia si giratia celor 3 axe de coordonate respectiv in cod vor aparea variabilele: accX, accY, accZ, gyroX, gyroY, gyroZ.

```
uint8_t* data = i2cRead(0x3B,14);
accX = ((data[0] << 8) | data[1]);
accY = ((data[2] << 8) | data[3]);
accZ = ((data[4] << 8) | data[5]);
tempRaw = ((data[6] << 8) | data[7]);
gyroX = ((data[8] << 8) | data[9]);
gyroY = ((data[10] << 8) | data[11]);
gyroZ = ((data[12] << 8) | data[13]);
```

Tangajul si riuliul se calculeaza prin urmatoarele formule:

```
accYangle = (atan2(accX,accZ)+PI)*RAD_TO_DEG;
accXangle = (atan2(accY,accZ)+PI)*RAD_TO_DEG;
```

Limitele functiei atan2 sunt de la -π la π asa ca pentru a realiza conversia din radiani in grade nu trebuie decat sa faci inmultirea cu: 57.295779513082320876798154814105 valoare care este deja declarata in Arduino: RAD_TO_DEG.

Totusi nu este suficient deoarece trebuie sa iei in calcul si datele generate de catre giroscop, pentru a utiliza cu succes filtrul Kalman.

Pentru mai multe detalii despre echilibrarea unei camere de filmat viziteaza urmatoarele adrese:

http://www.instructables.com/id/Guide-to-gyro-and-accelerometer-with-Arduino-inclu/

http://www.instructables.com/id/Gyro-Camera-for-Motorcycle/