```
// 1-Axis Brushless Gimbal with Arduino
// Copyright (C) 2015 ArduinoDeXXX All Rights Reserved.
// materials:
// "Arduino UNO",
// a 3-phase brushless motor with 14-pole and 12-slot,
// two gyro modules with "L3GD20" chips, and
// a motor driver IC "L298"
// information:
// The gyro which CS pin is connected to digital pin 8 is attached to the base of motor.
// The other gyro which CS pin is connected to digital pin 9 is attached to top of motor.
// Eight AA-batteries for 12V would work better than 6V with four batteries.
// Don't touch gyros for 8 seconds after reset Arduino.
// Three wires from the motor are connected to digital pins 3, 5 and 6 appropriately.
// If the motor rotates wrong direction, the connection of these 3 wires should be changed.
// If the motor does not stop at start, two alternative lines in "void chkAndCtl()" should be used instead of line A and B.
// View the sites bellow to see more detail.
// http://www.instructables.com/id/DIY-Brushless-Gimbal-with-Arduino/
// http://www.instructables.com/id/DIY-Brushless-Gimbal-with-Arduino-in-Japanese/

#include <SPI.h>
long recOmegaD = 0;
long omegaD = 0;
long thetaM = 0;
long recMicros = 0;
long recOmegaDa = 0;
long omegaDa = 0;
long thetaMa = 0;
long deg1000 = 0;
int variPwm = 0;
byte phase = 0;
int rz, rza, dRz, dRza;
long R, Ra;
const int motorPin1 = 3;
const int motorPin2 = 5;
const int motorPin3 = 6;
const long L1000 = 51429;
const int nLnT1000 = 2140;
const int  LnT1000 = 4291;
const int  LnLvl10 = 390;
const long slope1000 = 41245;

void L3GD20_write(byte  reg,byte val) {
  digitalWrite(8, LOW);
  SPI.transfer(reg);
  SPI.transfer(val);
  digitalWrite(8, HIGH);
}
byte L3GD20_read(byte reg) {
  byte ret = 0;
  digitalWrite(8, LOW);
  SPI.transfer(reg | 0x80);
   ret =SPI.transfer(0);
  digitalWrite(8, HIGH);
  return ret;
}
void L3GD20a_write(byte  reg,byte val) {
  digitalWrite(9, LOW);
  SPI.transfer(reg);
  SPI.transfer(val);
  digitalWrite(9, HIGH);
}
byte L3GD20a_read(byte reg) {
  byte ret = 0;
  digitalWrite(9, LOW);
  SPI.transfer(reg | 0x80);
   ret =SPI.transfer(0);
  digitalWrite(9, HIGH);
  return ret;
}
```

```
void setup () {
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);
  pinMode(motorPin3, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  digitalWrite(8, HIGH);
  digitalWrite(9, HIGH);
  SPI.begin();
  SPI.setBitOrder(MSBFIRST);
  SPI.setDataMode(SPI_MODE3);
  SPI.setClockDivider(SPI_CLOCK_DIV2);
  L3GD20_write(0x20, B11001111);
  L3GD20_write(0x23, B00000000);
  L3GD20a_write(0x20, B11001111);
  L3GD20a_write(0x23, B00000000);
  calibrate();
  delay(50);
   recMicros =micros();
}

void loop (){
  chkAndCtl();
  calcPwms();
  if ( phase == 1 ) {
    analogWrite(motorPin1, variPwm);
    analogWrite(motorPin2, 255);
    analogWrite(motorPin3, 0);
  }
  if ( phase == 2 ) {
    analogWrite(motorPin1, 255);
    analogWrite(motorPin2, variPwm);
    analogWrite(motorPin3, 0);
  }
  if ( phase == 3 ) {
    analogWrite(motorPin1, 255);
    analogWrite(motorPin2, 0);
    analogWrite(motorPin3, variPwm);
  }
  if ( phase == 4 ) {
    analogWrite(motorPin1, variPwm);
    analogWrite(motorPin2, 0);
    analogWrite(motorPin3, 255);
  }
  if ( phase == 5 ) {
    analogWrite(motorPin1, 0);
    analogWrite(motorPin2, variPwm);
    analogWrite(motorPin3, 255);
  }
  if ( phase == 6 ) {
    analogWrite(motorPin1, 0);
    analogWrite(motorPin2, 255);
    analogWrite(motorPin3, variPwm);
  }
}

void calibrate(){
  analogWrite(motorPin1, 217);
  analogWrite(motorPin2, 0);
  analogWrite(motorPin3, 255);
  delay (3000);
  R = 0;
  Ra = 0;
  for  (long i = 0 ; i < 4000 ; i++ ){
    if ( i > 1000 ) {
      rz = ( (L3GD20_read(0x2D) << 8) | L3GD20_read(0x2C) );
      R = R + rz;
      rza = ( (L3GD20a_read(0x2D) << 8) | L3GD20a_read(0x2C) );
      Ra = Ra + rza;
      delayMicroseconds( 25 );
```

```
    }
  }
  dRz  = R  / 750;
   dRza = Ra / 750;
}

void chkAndCtl() {
  R = 0;
  Ra = 0;
  for  (int i = 0 ; i < 4 ; i++ ) {
    rz = ( (L3GD20_read(0x2D) << 8) | L3GD20_read(0x2C) );
    R = R + rz;
    rza = ( (L3GD20a_read(0x2D) << 8) | L3GD20a_read(0x2C) );
    Ra = Ra + rza;
    delayMicroseconds( 25 );
  }
  omegaD  = ( R  - dRz  ) * 0.025;
  omegaDa = ( Ra - dRza ) * 0.025;
   recMicros =micros() - recMicros;
  thetaM = thetaM + ( (omegaD+recOmegaD)/20 ) * recMicros;
  thetaMa = thetaMa + ( - (omegaDa+recOmegaDa)/20 ) * recMicros;// Line A
//thetaMa = thetaMa + (   (omegaDa+recOmegaDa)/20 ) * recMicros; // The Alternative Line for Line A
  deg1000 = thetaM / 1000 - 1*omegaDa * recMicros /500 + 1*thetaMa/1000;// Line B
//deg1000 = thetaM / 1000 + 1*omegaDa * recMicros /500 + 1*thetaMa/1000; // The Alternative Line for Line B
   recMicros =micros();
  recOmegaD = omegaD;
  recOmegaDa = omegaDa;
}

void calcPwms() {
  int degPwm1000 = deg1000 % 8571;
  if ( degPwm1000 < 0 ) { degPwm1000 = degPwm1000 + 8571; }
  long linePwm = ( LnLvl10 + ( slope1000 * degPwm1000 ) / 100000 + 5 ) / 10;
  long degNL1000  =abs( degPwm1000 - LnT1000 - nLnT1000 );
  long nLinePwm = 255 - ( ( -  2935 * (sq( sq(degNL1000)/1000 ) /1000 )
                           +  8413 * ( (sq(degNL1000)/1000 ) * degNL1000 /1000 )
                           - 11421 * (sq(degNL1000)/1000 )
                          + 32929 * ( degNL1000 )
                        ) / 100000 + 5
                      ) / 10;
  int pwmShape = 0;
  if ( degPwm1000 < LnT1000 ) { pwmShape = linePwm; }else { pwmShape = nLinePwm; }
  int dDegPwm1000 = deg1000 % 17143;
  if ( dDegPwm1000 < 0 ) { dDegPwm1000 = dDegPwm1000 + 17143; }
  if ( dDegPwm1000 < 8571 ) { variPwm = 255 - pwmShape; }else { variPwm = pwmShape; }
  long shftDeg1000 = ( deg1000 + L1000/2 + nLnT1000 ) % L1000;
  if ( shftDeg1000 < 0 ) { shftDeg1000 = shftDeg1000 + L1000; }
  phase = 0;
  if ( shftDeg1000 < 8571 ) { phase = 1; }else {
    if ( shftDeg1000 < 17143 ) { phase = 2; }else {
      if ( shftDeg1000 < 25714 ) { phase = 3; }else {
        if ( shftDeg1000 < 34286 ) { phase = 4; }else {
          if ( shftDeg1000 < 42857 ) { phase = 5; }else {
            phase = 6; }
        }
      }
    }
  }
}
// Copyright (C) 2015 ArduinoDeXXX All Rights Reserved.
```