Escola Tècnica Superior d'Enginyeries
Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Titulació:

Grau en Enginyeria en Tecnologies Industrials

Alumne:

Joaquim Lobo Besora

Títol TFG:

Projecte d'una planta experimental d'assaig per UAVs

Director/a del TFG:

David González Diez

Convocatòria de lliurament del TFG:

Juny 2016

Contingut d'aquest volum: **ANNEXOS**

Escola Tècnica Superior d'Enginyeries
Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Projecte d'una planta experimental d'assaig per UAVs

Annexos

Escola Tècnica Superior d'Enginyeria
Industrial i Aeronàutica
de Terrassa

Grau en Enginyeria en Tecnologies Industrials

Autor: **Joaquim Lobo Besora**

Director del TFG: **David González Diez**

Juny 2016

Índex

Annex A: Control de la velocitat dels motors brushless 6

Annex B: Codis de programació 8

Codi utilitzat pel control de la posició angular amb un motor 8

Sense Filtre Complementari 8

Amb Filtre Complementari 11

Codi utilitzat pel control de la posició angular amb dos motors 15

Annex C: Efectes en el comportament de l'UAV 20

Proves experimentals amb un motor 20

Efecte del control proporcional 20

Efecte del control proporcional-integral 22

Efecte dels límits de la integral 24

Proves experimentals amb dos motors 25

Efecte dels límits de la integral 26

Annex D: Efecte del filtre complementari 27

Bibliografia 28

Índex de figures

Figura 1. Mostra de senyals PWM. 6

Figura 2. Angle (3,5º/) vs Temps (10s/), (Kp=250; Ki=0; Consigna=10º). 20

Figura 3. Angle (5º/) vs Temps (10s/), (Kp=500; Ki=0; Consigna=10º). 20

Figura 4. Angle (5º/) vs Temps (10s/), (Kp=750; Ki=0; Consigna=15º). 21

Figura 5. Angle (5º/) vs Temps (11s/), (Kp=500; Ki=0; Consigna=20º). 21

Figura 6. Angle (5º/) vs Temps (10s/), (Kp=750; Ki=0; Consigna=20º). 21

Figura 7. Angle (5º/) vs Temps (10s/), (Kp=1200; Ki=0; Consigna=20º). 21

Figura 8. Angle (5º/) vs Temps (11s/), (Kp=250; Ki=25; Consigna=15º). 22

Figura 9. Angle (5º/) vs Temps (15s/), (Kp=75; Ki=30; Consigna=15º). 22

Figura 10. Angle (5º/) vs Temps (13s/), (Kp=75; Ki=30; Consigna=20º). 23

Figura 11. Angle (5º/) vs Temps (14s/), (Kp=8,5; Ki=33; Consigna=20º). 23

Figura 12. Angle (5,5º/) vs Temps (10s/), (Límit positiu=1000; Límit negatiu=-1000; Kp=2,8; Ki=1,18; Consigna=20º). 24

Figura 13. Angle (5,5º/) vs Temps (10s/), (Límit positiu=500; Límit negatiu=-500; Kp=2,8; Ki=2,36; Consigna=20º). 24

Figura 14. Angle (5,5º/) vs Temps (10s/), (Límit positiu=250; Límit negatiu=-250; Kp=2,8; Ki=4,72; Consigna=20º). 24

Figura 15. Angle (5º/) vs Temps (6s/), (Kp=0,4; Ki=0,286; Límit positiu=35; Límit negatiu=-5; Consigna=-20º). 26

Figura 16. Angle (5º/) vs Temps (6s/), (Kp=0,4; Ki=0,13; Límit positiu=75; Límit negatiu=-5; Consigna=-20º). 26

Figura 17. Angle (0,5º/) vs Temps (6s/) sense filtre. 27

Figura 18. Angle (0,5º/) vs Temps (6s/) amb Filtre Complementari. 27

Llistat d'abreviatures

UAV: Unmanned Aerial Vehicle (Vehicle Aeri No Tripulat).

ESC: Electronic Speed controller (Controlador Electrònic de Velocitat).

PWM: Pulse-Width Modulation (Modulació per Ample de Pols).

PI: Proporcional - Integral.

Annex A: Control de la velocitat dels motors brushless

Aquest annex conté una breu explicació sobre el control de la velocitat dels motors brushless.

Els motors brushless són motors elèctrics que funcionen a partir de senyals trifàsiques. El component electrònic capaç de generar aquestes senyals s'anomena ESC. Per tant, la velocitat dels motors es controla a partir d'aquests dispositius.

Els ESCs es controlen mitjançant polsos, és a dir, el control de la velocitat dels motors es basa en la modulació per ample de pols. Per aquest motiu, el microcontrolador utilitzat ha d'incorporar pins PWM.

La modulació per ample de pols (PWM) és una tècnica que permet enviar dades a un dispositiu en forma d'ona quadrada (polsos). Aquesta ona està formada per un valor alt i un baix, en el cas d'Arduino 5V i 0V respectivament. [1] [3]

Els ESC ajusten la velocitat dels motors en funció de la relació entre el temps que la senyal es troba en el valor alt i el temps que es troba en el valor baix.

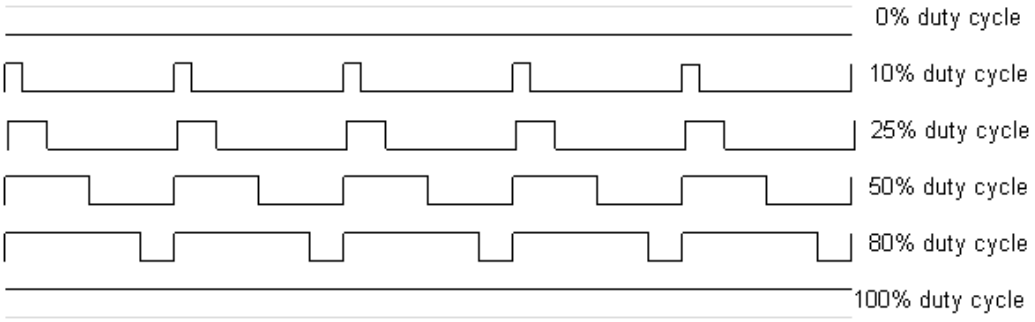

Figura 1. Mostra de senyals PWM.

Font: [2]

Aquesta relació s'anomena cicle de treball, normalment representat en %. En la figura 1 es pot veure un exemple de l'ona quadrada segons el cicle de treball.

Per defecte, en la majoria de motors i ESCs, el rang de funcionament és una senyal alta entre 1000 i 2000 µs d'amplitud, que s'envia cada 20 ms. És a dir, funcionen quan la senyal es manté en el seu valor alt durant un temps comprés entre 1000 i 2000 µs en un període de 20 ms. [4] [5]

Els ESC han de rebre un pols d'activació abans de funcionar. Alguns ESCs s'activen quan reben un pols d'amplitud mínima (1000 µs) durant 2-5 segons,

altres s'activen quan reben un pols d'amplitud màxima (2000 µs) durant el mateix temps. Quan l'ESC estigui activat només s'han d'enviar polsos de diferents amplituds (dins del rang de funcionament) per poder controlar la velocitat dels motors. [4]

En el subapartat 5.2.1.1 de la memòria es poden apreciar els passos seguits per l'activació dels ESCs.

Annex B: Codis de programació

En aquest annex es troben els codis creats per controlar la posició angular de l'UAV. [4] [6] [7] [8]

Codi utilitzat pel control de la posició angular amb un motor

Sense Filtre Complementari

```
#include <Servo.h> //Llibreria utilitzada pel control dels motors/servomotors.
#include <Wire.h> //Llibreria utilitzada per l'MPU-6050.
#define MPU 0x68 //Direcció I2C de la IMU.
#define S_A 16384 //Valor per defecte de la sensibilitat de l'acceleròmetre [LSB/g].
#define S_G 131 //Valor per defecte de la velocitat del giroscopi [LSB/°/s].
// Els valors anteriors s'utilitzen per obtenir valors coherents de les lectures del giroscopi [°/s] i de l'acceleròmetre [g]. Aquests valors es poden trobar a la web oficial de la IMU [].
#define RAD_A_DEG = 57.295779 // Conversió de radians a graus (180/pi).
#define PIN 10 //PIN PWM on està connectat l'ESC.
#define VSERIAL 9600 //Velocitat del port serial [bps].
#define PWMIN 1140 //Amplitud mínima de pols del rang de funcionament [µs].
#define VMAX 1230 //Limitació de la velocitat màxima del motor (arriba fins a 2000 µs).
#define VMIN 1160 //Limitació de la velocitat mínima del motor (arriba fins a 1140 µs).
#define Consigna 0//Valor de l'angle consigna [°].
#define LimitIntegral1 250 //Límit positiu de la integral.
#define LimitIntegral2 50 //Límit negatiu de la integral.
#define INCREMENT 5 //Increment de l'angle consigna a través del teclat.

Servo motor; //Es crea el motor com element de la llibreria Servo.

//Declaració de variables:
float AngleConsigna = Consigna;
float error; //Error entre l'angle consigna i l'angle d'inclinació actual de l'UAV.
float Integral = 0; //Assignació d'un valor inicial de 0 a la integral.
float U; //Velocitat del motor [µs].
float Kp = 2.8; //Guany proporcional.
float Ki = 33.7; //Guany integral.
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ; //L'MPU-6050 dóna els valors en enters de 16 bits (valors sense refinar).
float Angle_Accel_X; //Angle respecte l'eix X calculat a partir de les lectures de l'acceleròmetre.
boolean inicialitzacio = false;
int EntradaDeDada;
float OrdreTeclat = 0;
unsigned U1; //Variable sense signe.
```

```
//Variables declarades per calcular la mitjana dels últims 10 angles:
float angle1=0;
float angle2=0;
float angle3=0;
float angle4=0;
float angle5=0;
float angle6=0;
float angle7=0;
float angle8=0;
float angle9=0;
float angle0=0;
float Mitjana_Angle=0;


void setup() {

  Serial.begin(VSERIAL); //Iniciar port serial
  //Amb les 5 línies següents s'inicia la comunicació I2C amb l'MPU-6050 i s'activa enviant un 0.
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0); //Activació de l'MPU-6050.
  Wire.endTransmission(true);
  motor.attach(PIN); //Assignació del pin 10 al motor/ESC.
  Serial.println("INICI DEL PROGRAMA \n CONTROL DE PARAMETRES: \n A o a per augmentar l'Angle Consigna 5 graus \n Z o z per disminuir l'Angle Consigna 5 graus");
  Serial.println("Premer A o a per iniciar el programa un cop finalitzat el so de confirmacio");
  while (inicialitzacio == false) {//Creació d'un bucle que evita sortir del void setup si no s'envia una A o a.
    motor.writeMicroseconds(PWMIN); //Activació de l'ESC.
    EntradaDeDada = Serial.read(); //Lectura del byte rebut a través del Monitor Serie.
    if (EntradaDeDada == 65 || EntradaDeDada == 97) { //A o a
      inicialitzacio = true;
    }
  }
  delay(2500);//Temps d'espera de 2,5 segons.
}

void loop()
{

OrdreTeclat = ControlParametres(); //Es crida la funció ControlParametres().

AngleConsigna = AngleConsigna + OrdreTeclat; //Actualització de l'angle consigna si ha estat modificat a través del Monitor Sèrie.
```

```
//L'MPU-6050 emmagatzema les lectures de l'acceleròmetre i del giroscopi en espais de la memòria anomenats registres.
//Per poder llegir els valors de l'acceleròmetre i del giroscopi s'han de demanar els valors d'aquests registres.
//Lectura dels valors de l'acceleròmetre:
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Es comença demanant el registre 0x3B.
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //Es demanen 6 registres per la lectura de l'acceleròmetre (incloent el 0x3B), ja que cada valor ocupa 2 registres.
  AcX=Wire.read()<<8|Wire.read(); //Correspon als registres 0x3B i 0x3C.
  AcY=Wire.read()<<8|Wire.read(); //Correspon als registres 0x3D i 0x3E.
  AcZ=Wire.read()<<8|Wire.read();//Correspon als registres 0x3F i 0x3G.

//A partir dels valors obtinguts de les lectures de l'acceleròmetre, es calcula l'angle en X:
  Angle_Accel_X = atan((AcY / S_A) / sqrt(pow((AcZ / S_A), 2))) * RAD_TO_DEG;

//Mitjana dels 10 últims angles per tal d'atenuar el soroll causat per les interferències electromagnètiques:
  angle0=angle9;  angle9=angle8;  angle8=angle7;  angle7=angle6; angle6=angle5;  angle5=angle4;  angle4=angle3;  angle3=angle2; angle2=angle1; angle1= Angle_Accel_X;
  Mitjana_Angle=(angle1+angle2+angle3+angle4+angle5+angle6+angle7+ angle8+angle9+angle0)/10;

  error = AngleConsigna - Mitjana_Angle; //Error entre l'angle consigna i l'angle d'inclinació actual de l'UAV.
  Integral = error + Integral; // Integral de l'error de l'angle.

  float P = Kp * error; //Part proporcional del control PI.

//Limitació de la integral:
 if (Integral >LimitIntegral1) {

    Integral= LimitIntegral1;
  }
  if (Integral <= -LimitIntegral2){

    Integral= -LimitIntegral2;
  }

  float I = Ki * Integral; //Part integral del control PI.

  U = P + I; //Velocitat del motor [µs] (Senyal de control).

 //Limitació de la velocitat màxima i mínima del motor:
 if (U < VMIN) {
    U = VMIN;
  }
  if (U > VMAX) {
    U = VMAX;
  }
```

```
 U1=(unsigned)U; //S'elimina el signe a la velocitat del motor, ja
que un pols en µs no pot ser negatiu.
  Serial.println(Mitjana_Angle); //Mostrar els valors de l'angle a
través del Monitor Sèrie.
  motor.writeMicroseconds(U1); //Velocitat assignada al motor.
  delay(20); //Temps de cicle (0.02 segons).

}

int ControlParametres(){ //Funció que serveix per augmentar o
disminuir el valor de l'angle consigna a través del Monitor
Serial.

   int  ordre=0;

  if (Serial.available() > 0) { //Si s'ha enviat alguna dada a
través del Monitor Serial:
     EntradaDeDada = Serial.read(); // Lectura del Byte rebut
     if (EntradaDeDada == 65 || EntradaDeDada ==97) { // Si s'ha
enviat una A o a
      Serial.println( " ANGLE CONSIGNA AUGMENTAT");
        ordre = INCREMENT; //Es dóna a la variable ordre un valor
de 5 que servirà per augmentar l'angle consigna.
      }
    if (EntradaDeDada == 90 || EntradaDeDada ==122) { // Si s'ha
enviat una Z o z
      Serial.println( "ANGLE CONSIGNA DISMINUIT");
      ordre = -INCREMENT; //Es dóna a la variable ordre un valor
de -5 que servirà per disminuir l'angle consigna.
    }

  }

  return (ordre); //Enviar el valor de la variable ordre allà on
es cridi la funció.
}
```

Amb Filtre Complementari

```
#include <Servo.h> //Llibreria utilitzada pel control dels
motors/servomotors.
#include <Wire.h> //Llibreria utilitzada per l'MPU-6050.
#define MPU 0x68 //Direcció I2C de la IMU.
#define S_A 16384 //Valor per defecte de la sensibilitat de
l'acceleròmetre [LSB/g].
#define S_G 131 //Valor per defecte de la velocitat del giroscopi
[LSB/°/s].
// Els valors anteriors s'utilitzen per obtenir valors coherents
de les lectures del giroscopi [°/s] i de l'acceleròmetre [g].
Aquests valors es poden trobar a la web oficial de la IMU [].
#define RAD_A_DEG = 57.295779 // Conversió de radians a graus
(180/pi).
#define PIN 10 //PIN PWM on està connectat l'ESC.
#define VSERIAL 9600 //Velocitat del port serial [bps].
#define PWMIN 1140 //Amplitud mínima de pols del rang de
funcionament [µs].
#define VMAX 1230 //Limitació de la velocitat màxima del motor
(arriba fins a 2000 µs).
```

```
#define VMIN 1160 //Limitació de la velocitat mínima del motor (arriba fins a 1140 µs).
#define Consigna 0//Valor de l'angle consigna [°].
#define LimitIntegral1 250 //Límit positiu de la integral.
#define LimitIntegral2 50 //Límit negatiu de la integral.
#define INCREMENT 5 //Increment de l'angle consigna a través del teclat.

Servo motor; //Es crea el motor com element de la llibreria Servo.

//Declaració de variables:
float AngleConsigna = Consigna;
float error; //Error entre l'angle consigna i l'angle d'inclinació actual de l'UAV.
float Integral = 0; //Assignació d'un valor inicial de 0 a la integral.
float U; //Velocitat del motor [µs].
float Kp = 2.8; //Guany proporcional.
float Ki = 33.7; //Guany integral.
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ; //L'MPU-6050 dóna els valors en enters de 16 bits (valors sense refinar).
float Angle_Accel_X; //Angle respecte l'eix X calculat a partir de les lectures de l'acceleròmetre.
float Gy_X; //Valor corresponent a les lectures del giroscopi.
float Angle_X;//Angle obtingut amb el Filtre Complementari.
boolean inicialitzacio = false;
int EntradaDeDada;
float OrdreTeclat = 0;
unsigned U1; //Variable sense signe.
//Variables declarades per calcular la mitjana dels últims 10 angles:
float angle1=0;
float angle2=0;
float angle3=0;
float angle4=0;
float angle5=0;
float angle6=0;
float angle7=0;
float angle8=0;
float angle9=0;
float angle0=0;
float Mitjana_Angle=0;


void setup() {

  Serial.begin(VSERIAL); //Iniciar port serial
  //Amb les 5 línies següents s'inicia la comunicació I2C amb l'MPU-6050 i s'activa enviant un 0.
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0); //Activació de l'MPU-6050.
  Wire.endTransmission(true);
  motor.attach(PIN); //Assignació del pin 10 al motor/ESC.
  Serial.println("INICI DEL PROGRAMA \n CONTROL DE PARAMETRES: \n A o a per augmentar l'Angle Consigna 5 graus \n Z o z per disminuir l'Angle Consigna 5 graus");
```

```
  Serial.println("Premer A o a per iniciar el programa un cop
finalitzat el so de confirmacio");
  while (inicialitzacio == false) {//Creació d'un bucle que evita
sortir del void setup si no s'envia una A o a.
    motor.writeMicroseconds(PWMIN); //Activació de l'ESC.
    EntradaDeDada = Serial.read(); //Lectura del byte rebut a
través del Monitor Serie.
    if (EntradaDeDada == 65 || EntradaDeDada == 97) { //A o a
      inicialitzacio = true;
    }
  }
  delay(2500);//Temps d'espera de 2,5 segons.
}

void loop()
{

OrdreTeclat = ControlParametres(); //Es crida la funció
ControlParametres().

AngleConsigna = AngleConsigna + OrdreTeclat; //Actualització de
l'angle consigna si ha estat modificat a través del Monitor Sèrie.


//L'MPU-6050 emmagatzema les lectures de l'acceleròmetre i del
giroscopi en espais de la memòria anomenats registres.
//Per poder llegir els valors de l'acceleròmetre i del giroscopi
s'han de demanar els valors d'aquests registres.
//Lectura dels valors de l'acceleròmetre:
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Es comença demanant el registre 0x3B.
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //Es demanen 6 registres per la
lectura de l'acceleròmetre (incloent el 0x3B), ja que cada valor
ocupa 2 registres.
  AcX=Wire.read()<<8|Wire.read(); //Correspon als registres 0x3B i
0x3C.
  AcY=Wire.read()<<8|Wire.read(); //Correspon als registres 0x3D i
0x3E.
  AcZ=Wire.read()<<8|Wire.read();//Correspon als registres 0x3F i
0x3G.

//A partir dels valors obtinguts de les lectures de
l'acceleròmetre, es calcula l'angle en X:
  Angle_Accel_X = atan((AcY / S_A) / sqrt(pow((AcZ / S_A), 2))) *
RAD_TO_DEG;

 //Lectura dels valors del giroscopi:
   Wire.beginTransmission(MPU);
   Wire.write(0x43); //Es comença demanat el registre 0x43
   Wire.endTransmission(false);
   Wire.requestFrom(MPU,2,true);      //A      diferència      de
l'acceleròmetre, només es demanen 2 registres (incloent el 0x43),
ja que, només es necessiten les lectures del giroscopi en l'eix X.
   GyX=Wire.read()<<8|Wire.read(); //Correspon als registres 0x43
i 0x44.

   //Valor de les lectures del giroscopi en °/s.
   Gy_X= GyX/S_G;//Eix X.
```

```
  //Filtre Complementari per l'angle en l'eix X:
  Angle_X = 0.9 *(Angle_X+Gy_X*0.02) + 0.1*Angle_Accel_X;

  //Mitjana dels 10 últims angles per tal d'atenuar el soroll causat per les interferències electromagnètiques:
  angle0=angle9;   angle9=angle8;   angle8=angle7;   angle7=angle6;   angle6=angle5;   angle5=angle4;   angle4=angle3;   angle3=angle2;   angle2=angle1; angle1=Angle_X;
  Mitjana_Angle=(angle1+angle2+angle3+angle4+angle5+angle6+angle7+angle8+angle9+angle0)/10;

  error = AngleConsigna - Mitjana_Angle; //Error entre l'angle consigna i l'angle d'inclinació actual de l'UAV.
  Integral = error + Integral; // Integral de l'error de l'angle.

  float P = Kp * error; //Part proporcional del control PI.

//Limitació de la integral:
 if (Integral >LimitIntegral1) {

    Integral= LimitIntegral1;
  }
  if (Integral <= -LimitIntegral2){

    Integral= -LimitIntegral2;
  }

  float I = Ki * Integral; //Part integral del control PI.

  U = P + I; //Velocitat del motor [µs] (Senyal de control).

 //Limitació de la velocitat màxima i mínima del motor:
 if (U < VMIN) {
    U = VMIN;
  }
  if (U > VMAX) {
    U = VMAX;
  }

 U1=(unsigned)U; //S'elimina el signe a la velocitat del motor, ja que un pols en µs no pot ser negatiu.
  Serial.println(Mitjana_Angle); //Mostrar els valors de l'angle a través del Monitor Sèrie.
  motor.writeMicroseconds(U1); //Velocitat assignada al motor.
  delay(20); //Temps de cicle (0.02 segons).

}

int ControlParametres(){ //Funció que serveix per augmentar o disminuir el valor de l'angle consigna a través del Monitor Serial.

   int  ordre=0;

  if (Serial.available() > 0) { //Si s'ha enviat alguna dada a través del Monitor Serial:
     EntradaDeDada = Serial.read(); // Lectura del Byte rebut
     if (EntradaDeDada == 65 || EntradaDeDada ==97) { // Si s'ha enviat una A o a
```

```
      Serial.println( " ANGLE CONSIGNA AUGMENTAT");
        ordre = INCREMENT; //Es dóna a la variable ordre un valor
de 5 que servirà per augmentar l'angle consigna.
      }
    if (EntradaDeDada == 90 || EntradaDeDada ==122) { // Si s'ha
enviat una Z o z
      Serial.println( "ANGLE CONSIGNA DISMINUIT");
      ordre = -INCREMENT; //Es dóna a la variable ordre un valor
de -5 que servirà per disminuir l'angle consigna.
    }

  }

  return (ordre); //Enviar el valor de la variable ordre allà on
es cridi la funció.
}
```

Codi utilitzat pel control de la posició angular amb dos motors

```
#include <Servo.h> //Llibreria utilitzada pel control dels
motors/servomotors.
#include <Wire.h> //Llibreria utilitzada per l'MPU-6050.
#define MPU 0x68 //Direcció I2C de la IMU.
#define S_A 16384 //Valor per defecte de la sensibilitat de
l'acceleròmetre [LSB/g].
#define S_G 131 //Valor per defecte de la velocitat del giroscopi
[LSB/°/s].
// Els valors anteriors s'utilitzen per obtenir valors coherents
de les lectures del giroscopi [°/s] i de l'acceleròmetre [g].
Aquests valors es poden trobar a la web oficial de la IMU [].
#define RAD_A_DEG = 57.295779 // Conversió de radians a graus
(180/pi).
#define PIN1 10 //PIN PWM on està connectat un ESC.
#define PIN2 6 //PIN PWM on està connectat l'altre ESC.
#define VSERIAL 9600 //Velocitat del port serial [bps].
#define PWMIN1 1140 //Amplitud mínima de pols per l'activació del
motor CW [µs].
#define PWMIN2 1135 //Amplitud mínima de pols per l'activació del
motor CCW [µs].
#define VMAX 1230 //Limitació de la velocitat màxima del motor
(arriba fins a 2000 µs).
#define VMIN 1160 //Limitació de la velocitat mínima del motor
(arriba fins a 1140 µs).
#define Consigna 0//Valor de l'angle consigna [°].
#define LimitIntegral1 250 //Límit positiu de la integral.
#define LimitIntegral2 50 //Límit negatiu de la integral.
#define INCREMENT 5 //Increment de l'angle consigna a través del
teclat.
#define Uo 1175 //Velocitat base [µs].

Servo motor1, motor2; //Es crea el motor com element de la
llibreria Servo.

//Declaració de variables:
float AngleConsigna = Consigna;
float error; //Error entre l'angle consigna i l'angle d'inclinació
actual de l'UAV.
```

```
float Integral = 0; //Assignació d'un valor inicial de 0 a la integral.
float U; //Diferencial de velocitat entre els dos motors[µs].
float Kp = 0.4; //Guany proporcional
float Ki = 0.04; //Guany integral
int16_t AcX, AcY, AcZ, GyX, GyY, GyZ; //L'MPU-6050 dóna els valors en enters de 16 bits (valors sense refinar).
float Angle_Accel_X; //Angle respecte l'eix X calculat a partir de les lectures de l'acceleròmetre.
float Gy_X; //Valor corresponent a les lectures del giroscopi.
float Angle_X;//Angle obtingut amb el Filtre Complementari.
boolean inicialitzacio = false;
int EntradaDeDada;
float OrdreTeclat = 0;
unsigned U0; //S'elimina el signe de Uo, d'aquesta manera empre serà un valor positiu.
float U1; //Velocitat d'un dels motors.
float U2; //Velocitat de l'altre motor.
//Variables declarades per calcular la mitjana dels últims 10 angles:
float angle1=0;
float angle2=0;
float angle3=0;
float angle4=0;
float angle5=0;
float angle6=0;
float angle7=0;
float angle8=0;
float angle9=0;
float angle0=0;
float Mitjana_Angle=0;


void setup() {

  Serial.begin(VSERIAL); //Iniciar port serial
  //Amb les 5 línies següents s'inicia la comunicació I2C amb l'MPU-6050 i s'activa enviant un 0.
  Wire.begin();
  Wire.beginTransmission(MPU);
  Wire.write(0x6B);
  Wire.write(0); //Activació de l'MPU-6050.
  Wire.endTransmission(true);
  motor1.attach(PIN1); //Assignació del pin 10 a un dels motors/ESCs.
  motor2.attach(PIN2);//Assignació del pin 6 a l'altre motor/ESC.
  Serial.println("INICI DEL PROGRAMA \n CONTROL DE PARAMETRES: \n A o a per augmentar l'Angle Consigna 5 graus \n Z o z per disminuir l'Angle Consigna 5 graus");
  Serial.println("Premer A o a per iniciar el programa un cop finalitzat el so de confirmacio");
  while (inicialitzacio == false) {//Creació d'un bucle que evita sortir del void setup si no s'envia una A o a.
    motor1.writeMicroseconds(PWMIN1); //Activació de l'ESC connectat al motor CW.
    motor2.writeMicroseconds(PWMIN2); //Activació de l'ESC connectat al motor CCW.
EntradaDeDada = Serial.read(); //Lectura del byte rebut a través del Monitor Sèrie.
```

```
    if (EntradaDeDada == 65 || EntradaDeDada == 97) { //A o a
      inicialitzacio = true;
    }
  }
  delay(2500);//Temps d'espera de 2,5 segons.
}

void loop()
{

OrdreTeclat = ControlParametres(); //Es crida la funció
ControlParametres().

AngleConsigna = AngleConsigna + OrdreTeclat; //Actualització de
l'angle consigna si ha estat modificat a través del Monitor Sèrie.

//L'MPU-6050 emmagatzema les lectures de l'acceleròmetre i del
giroscopi en espais de la memòria anomenats registres.
//Per poder llegir els valors de l'acceleròmetre i del giroscopi
s'han de demanar els valors d'aquests registres.
//Lectura dels valors de l'acceleròmetre:
  Wire.beginTransmission(MPU);
  Wire.write(0x3B); //Es comença demanant el registre 0x3B.
  Wire.endTransmission(false);
  Wire.requestFrom(MPU, 6, true); //Es demanen 6 registres per la
lectura de l'acceleròmetre (incloent el 0x3B), ja que cada valor
ocupa 2 registres.
  AcX=Wire.read()<<8|Wire.read(); //Correspon als registres 0x3B i
0x3C.
  AcY=Wire.read()<<8|Wire.read(); //Correspon als registres 0x3D i
0x3E.
  AcZ=Wire.read()<<8|Wire.read();//Correspon als registres 0x3F i
0x3G.

//A partir dels valors obtinguts de les lectures de
l'acceleròmetre, es calcula l'angle en X:
  Angle_Accel_X = atan((AcY / S_A) / sqrt(pow((AcZ / S_A), 2))) *
RAD_TO_DEG;

 //Lectura dels valors del giroscopi:
   Wire.beginTransmission(MPU);
   Wire.write(0x43); //Es comença demanat el registre 0x43
   Wire.endTransmission(false);
   Wire.requestFrom(MPU,2,true);      //A      diferència      de
l'acceleròmetre, només es demanen 2 registres (incloent el 0x43),
ja que, només es necessiten les lectures del giroscopi en l'eix X.
   GyX=Wire.read()<<8|Wire.read(); //Correspon als registres 0x43
i 0x44.

   //Valor de les lectures del giroscopi en °/s.
   Gy_X= GyX/S_G;//Eix X.
  //Filtre Complementari per l'angle en l'eix X:
  Angle_X = 0.9 *(Angle_X+Gy_X*0.02) + 0.1*Angle_Accel_X;

  //Mitjana dels 10 últims angles per tal d'atenuar el soroll
causat per les interferències electromagnètiques:
  angle0=angle9;  angle9=angle8;  angle8=angle7;  angle7=angle6;
angle6=angle5;  angle5=angle4;  angle4=angle3;  angle3=angle2;
angle2=angle1; angle1=Angle_X;
```

```
  Mitjana_Angle=(angle1+angle2+angle3+angle4+angle5+angle6+angle7+
angle8+angle9+angle0)/10;

  error = AngleConsigna - Mitjana_Angle; //Error entre l'angle
consigna i l'angle d'inclinació actual de l'UAV.
  Integral = error + Integral; // Integral de l'error de l'angle.

  float P = Kp * error; //Part proporcional del control PI.

//Limitació de la integral:
 if (Integral >LimitIntegral1) {

    Integral= LimitIntegral1;
  }
  if (Integral <= -LimitIntegral2){

    Integral= -LimitIntegral2;
  }

  float I = Ki * Integral; //Part integral del control PI.

  U = P + I; //Diferencial de velocitat entre els dos motors [µs]
(Senyal de control).

//Els següents "if" s'utilitzen per arribar als angles consigna
més grans sense haver d'augmentar la velocitat base (Uo):
  if(AngleConsigna>0){
  U1= Uo+U;
  U2=(Uo-U)-12;
  }
  if (AngleConsigna==0){
  U1= Uo+U; // Velocitat del motor connectat al pin 10.
  U2=Uo-U; // Velocitat del motor connectat al pin 6.
  }
  if (AngleConsigna<0){
  U1= (Uo+U)-12;
  U2=Uo-U;
  }

//Limitació de la velocitat màxima i mínima de cada motor:
if (U1<VMIN){

  U1= VMIN;

  }

  if (U2<VMIN){

  U2= VMIN;

  }
  if (U1>VMAX){

  U1= VMAX;

  }

  if (U2>VMAX){
```

```
  U2= VMAX;

  }
  Serial.println(Mitjana_Angle); //Mostrar els valors de l'angle a
través del Monitor Sèrie.
  motor1.writeMicroseconds(U1);  //Velocitat  assignada  al  motor
connectat al pin 10.
  motor2.writeMicroseconds(U2);  //Velocitat  assignada  al  motor
connectat al pin 6.
  delay(20); //Temps de cicle (0.02 segons).

}

int  ControlParametres(){  //Funció  que  serveix  per  augmentar  o
disminuir  el  valor  de  l'angle  consigna  a  través  del  Monitor
Serial.

   int  ordre=0;

  if  (Serial.available()  >  0)  {  //Si  s'ha  enviat  alguna  dada  a
través del Monitor Serial:
     EntradaDeDada = Serial.read(); // Lectura del Byte rebut
     if (EntradaDeDada == 65 || EntradaDeDada ==97) { // Si s'ha
enviat una A o a
      Serial.println( " ANGLE CONSIGNA AUGMENTAT");
        ordre = INCREMENT; //Es dóna a la variable ordre un valor
de 5 que servirà per augmentar l'angle consigna.
      }
    if (EntradaDeDada == 90 || EntradaDeDada ==122) { // Si s'ha
enviat una Z o z
      Serial.println( "ANGLE CONSIGNA DISMINUIT");
      ordre = -INCREMENT; //Es dóna a la variable ordre un valor
de -5 que servirà per disminuir l'angle consigna.
    }

  }

  return (ordre); //Enviar el valor de la variable ordre allà on
es cridi la funció.
}
```

Annex C: Efectes en el comportament de l'UAV

En el següent annex, es mostren alguns dels efectes observats durant les proves experimentals en modificar els diferents paràmetres.

Proves experimentals amb un motor

Efecte del control proporcional

Inicialment es va definir un valor del guany proporcional (Kp) igual a 250 i un angle consigna de 10º, tal i com es pot veure en les figures 2 i 3.

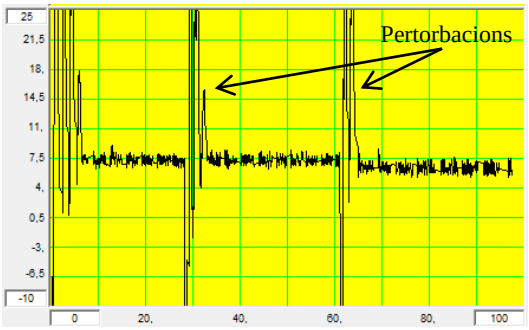

Figura 2. Angle (3,5º/) vs Temps (10s/), (Kp=250; Ki=0; Consigna=10º).

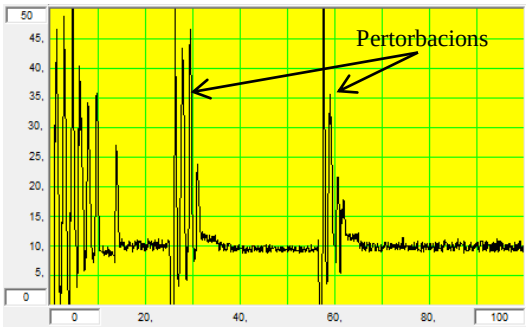

Figura 3. Angle (5º/) vs Temps (10s/), (Kp=500; Ki=0; Consigna=10º).

En la figura 2, amb un angle consigna de 10º i Kp=250, l'angle tarda uns 7 segons a estabilitzar-se des de l'instant inicial i, uns 4 segons després d'aplicar una pertorbació. Tot i que són temps petits, l'angle no oscil·la al voltant de l'angle consigna en el règim permanent.

En canvi, en la figura 3, amb un angle consigna de 10º i Kp= 500, l'angle tarda uns 11 segons a estabilitzar-se des de l'instant inicial i, uns 6 segons després d'aplicar una pertorbació. En aquest cas, l'angle oscil·la al voltant de l'angle consigna (10º) en el règim permanent.

En ambdós casos l'amplitud de les oscil·lacions és molt petita, encara que en la figura 2, l'angle oscil·la al voltant de 7,5º i no de l'angle consigna.

En les figures 2 i 3 també s'observa que, com més gran és el valor del guany proporcional, més tarda a arribar al règim permanent i a estabilitzar-se.

Aquest fenomen ja s'esperava, perquè com s'ha explicat en el subapartat 5.2.3.1 de la memòria, un augment del guany proporcional ajuda a l'angle a arribar a

l'angle consigna més ràpidament, però pot ocasionar un sobreimpuls que n'augmenti la inestabilitat. A més a més, com més petit és el valor del guany, més petit és l'angle on queda estabilitzat.

A primera vista, els resultats obtinguts en la figura 3 semblen acceptables, ja que, les oscil·lacions al voltant de l'angle consigna tenen molt poca amplitud i, els temps que tarda l'angle a estabilitzar són petits.

El següent pas va ser comprovar l'estabilitat amb un angle consigna de 15 i 20º.

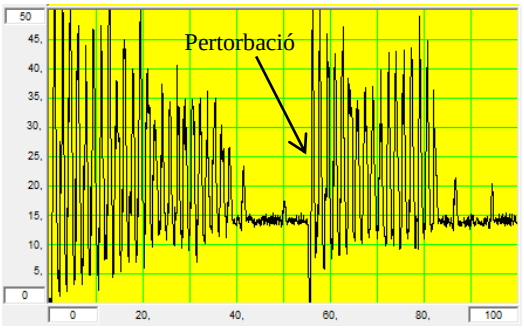

Figura 4. Angle (5º/) vs Temps (10s/), (Kp=750; Ki=0; Consigna=15º).

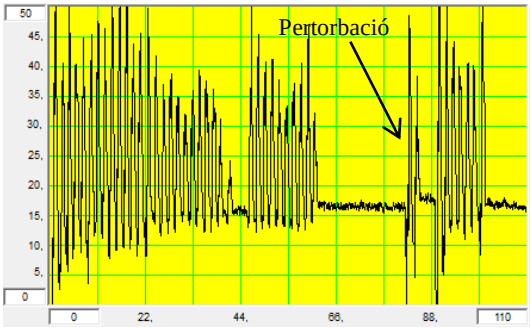

Figura 5. Angle (5º/) vs Temps (11s/), (Kp=500; Ki=0; Consigna=20º).

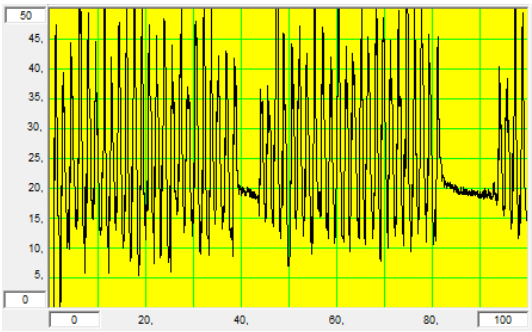

Figura 6. Angle (5º/) vs Temps (10s/), (Kp=750; Ki=0; Consigna=20º).

Figura 7. Angle (5º/) vs Temps (10s/), (Kp=1200; Ki=0; Consigna=20º).

En la figura 5, amb un angle consigna de 20º, es pot apreciar que, tot i utilitzar el mateix valor del guany proporcional que ha donat bons resultats en la figura 3, l'angle no arriba als 20º en el règim permanent. A més a més, l'angle presenta una gran inestabilitat.

A partir de les figures 4, 5, 6 i 7, es pot afirmar que per angles i valors de Kp grans, l'angle és cada cop més inestable, en canvi, per valors de Kp petits, és més estable però no arriba l'angle consigna en el règim permanent.

En conclusió, amb un control proporcional no n´hi ha prou per controlar la posició angular de l'UAV.

Efecte del control proporcional-integral

Un cop vist que el control proporcional no és suficient, ha estat necessari implantar un control proporcional-integral.

El procediment consisteix en anar augmentant el valor del guany integral (Ki) i disminuir el del proporcional (Kp) fins a obtenir un resultat acceptable.

En les figures 8, 9, 10 i 11 es poden veure alguns exemples.

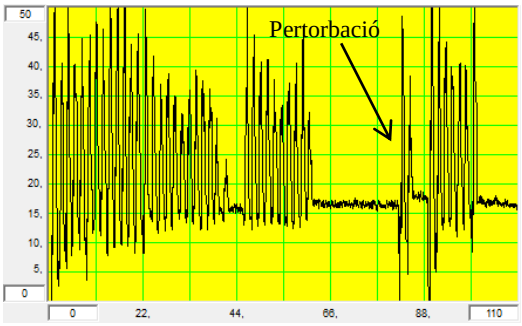

Figura 8. Angle (5º/) vs Temps (11s/), (Kp=250; Ki=25; Consigna=15º).

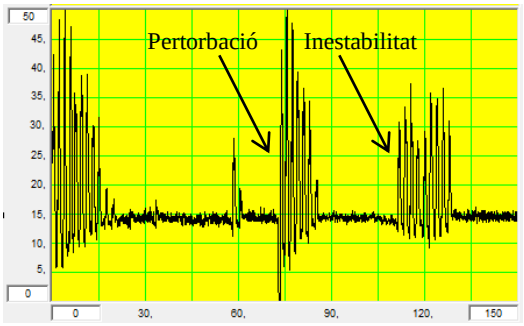

Figura 9. Angle (5º/) vs Temps (15s/), (Kp=75; Ki=30; Consigna=15º).

A partir de les figures 8 i 9, amb un angle consigna de 15º, es pot observar que per un mateix angle consigna, en augmentar el valor del guany integral i disminuir el del proporcional, el comportament de l'angle millora.

En la figura 8, el comportament és força inestable degut al sobreimpuls ocasionat per l'elevat valor del guany proporcional. L'angle es desestatitza amb molta facilitat sense necessitat d'aplicar una pertorbació. A més a més, en el règim permanent l'angle no arriba al valor de l'angle consigna.

Pel que fa a la figura 9, amb un guany proporcional menor i un guany integral major que els de la figura 8, el comportament és més estable i l'angle oscil·la més proper a l'angle consigna. El temps que tarda a estabilitzar-se de del moment inicial és d'uns 20 segons i, d'uns 13 segons després d'aplicar una pertorbació. Són temps massa elevats i, per aquests valors dels guanys, l'angle es desestabilitza sense necessitat d'aplicat una pertorbació.

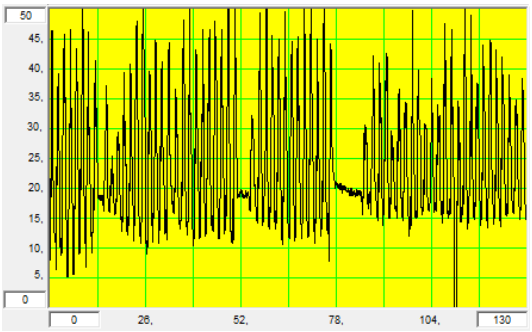

Figura 10. Angle (5º/) vs Temps (13s/), (Kp=75; Ki=30; Consigna=20º).

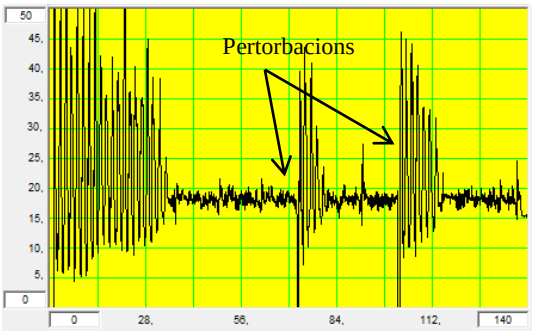

Figura 11. Angle (5º/) vs Temps (14s/), (Kp=8,5; Ki=33; Consigna=20º).

En les figures 10 i 11, amb un angle consigna de 20º, es pot apreciar un efecte similar al observat en les figures 8 i 9. Per un mateix angle consigna, en disminuir el valor del guany proporcional i augmentar el del guany integral, el comportament de l'angle millora.

En la figura 10, l'angle és molt inestable, pràcticament no s'estabilitza en cap moment.

En la figura 11, en canvi, l'angle guanya estabilitat, encara que, tant el temps que tarda en estabilitzar-se des de l'instant inicial (uns 36 segons) com el temps que tarda després d'aplicar una pertorbació (entre 10 i 14 segons) és massa gran.

En les figures 9 i 10, amb un angle consigna de 15 i 20º respectivament, s'han utilitzat els mateixos valors dels guanys, però el resultat obtingut ha estat totalment diferent. En la figura 9 l'angle té un comportament relativament estable, mentre que en la figura 10 és totalment inestable. Aquest fet ha obligat a obtenir els valors dels guanys a partir de l'angle consigna màxim (20º), i així trobar uns valors que funcionin amb tots els angles definits.

Per tant, el següent pas ha estat mantenir l'angle consigna a 20º, anar augmentant el valor del guany integral i disminuint el del guany proporcional fins aconseguir els valors que han permès obtenir els millors resultats possibles amb tots els angles consigna.

Els resultats definitius d'aquestes proves pes poden consultar en l'apartat 6.1.1 de la memòria.

En conclusió, a mesura que disminueix el valor del guany proporcional (Kp) i augmenta el del guany integral (Ki), les oscil·lacions disminueixen més ràpidament, per tant, l'angle s'estabilitza més ràpid. Aquest fenomen ja s'esperava, perquè com s'ha explicat en el subapartat 5.2.3.1 de la memòria, la

part integral del control PI proporciona una major estabilitat, suavitzant l'efecte de la part proporcional.

També es pot veure com la inestabilitat augmenta a mesura que augmenta l'angle consigna, sobretot en les figures 9 i 10.

Efecte dels límits de la integral

Durant les primeres proves experimentals realitzades amb un motor, es va observar que l'UAV no s'estabilitzava. El motor tardava massa temps en augmentar de velocitat per tornar a situar-se a l'angle consigna.

Aquest fet era degut a l'enorme valor de la integral de la part integral del control PI, per aquest motiu va ser necessari limitar-lo, tant per la part positiva com per la negativa.

En les figues 12, 13 i 14, es pot veure l'efecte produït en disminuir el valor dels límits positiu i negatiu (en valor absolut) de la integral.

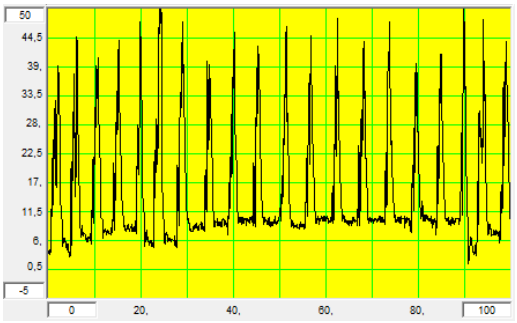

Figura 12. Angle (5,5º/) vs Temps (10s/), (Límit positiu=1000; Límit negatiu=-1000; Kp=2,8; Ki=1,18; Consigna=20º).

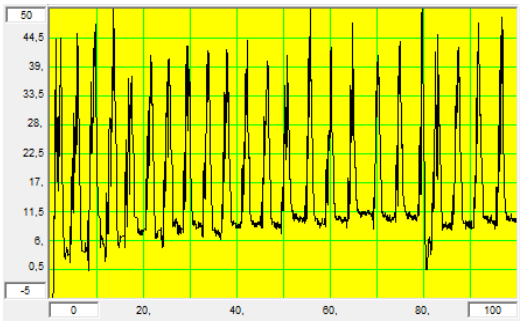

Figura 13. Angle (5,5º/) vs Temps (10s/), (Límit positiu=500; Límit negatiu=-500; Kp=2,8; Ki=2,36; Consigna=20º).

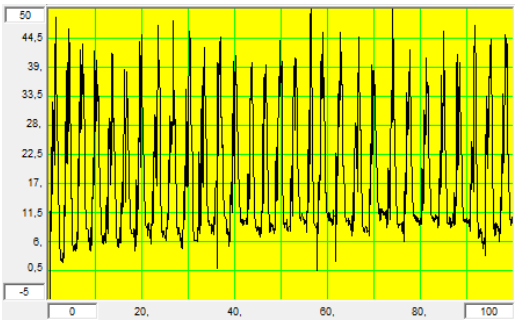

Figura 14. Angle (5,5º/) vs Temps (10s/), (Límit positiu=250; Límit negatiu=-250; Kp=2,8; Ki=4,72; Consigna=20º).

La velocitat del motor en aquestes proves experimental correspon a la suma de la part integral i de la part proporcional del control PI. En les figures anteriors s'ha mantingut constant el valor del guany proporcional i s'ha variat el del guany integral per tal d'obtenir la mateixa velocitat en cada cas.

A partir de les figures 12, 13 i 14, es pot observar clarament que en augmentar el valor dels límits de la integral (en valor absolut), el temps que tarda el motor a canviar de velocitat també augmenta i, com a conseqüència tarda més temps a situar-se a l'angle consigna.

Aquest fenomen té sentit, ja que, per arribar a una mateixa velocitat, transcorrerà més temps com més grans siguin els límits de la integral (en valor absolut).

S'ha de tenir en compte que les figures anteriors només són un exemple per mostrar l'efecte dels límits de la integral. Els valors dels guanys utilitzats no són els definitius.

Els valors finals s'han obtingut a partir del valor definitiu del guany proporcional i el del guany integral que es poden consultar en el subapartat 6.1.1 de la memòria.

Proves experimentals amb dos motors

Aquestes proves s'han dut a terme a partir del codi obtingut de les proves amb un motor, realitzant les pertinents modificacions.

Els passos a seguir per obtenir el valor definitiu del guany proporcional i el del guany integral són pràcticament els mateixos que en les proves anteriors.

No es mostren els efectes del control proporcional ni del control proporcional-integral, perquè són molt similars als obtinguts anteriorment. L'estabilitat també augmenta a mesura que disminueix el guany proporcional i augmenta el valor del guany integral.

Efecte dels límits de la integral

Els valors dels límits de la integral s'han augmentat (en valor absolut) respecte als valors obtinguts en les proves amb un motor, això és degut al funcionament dels dos motors alhora, ja que, quan un dels dos motors ha de baixar d'una posició més elevada, ho fa més ràpidament gràcies a la força creada per l'altre motor. Per tant, el temps que tarden els motors en canviar de velocitat per tornar a situar-se a l'angle consigna és inferior que en les proves amb un motor.

Es figures 15 i 16 mostren l'efecte que produeix augmentar el límit positiu de la integral.

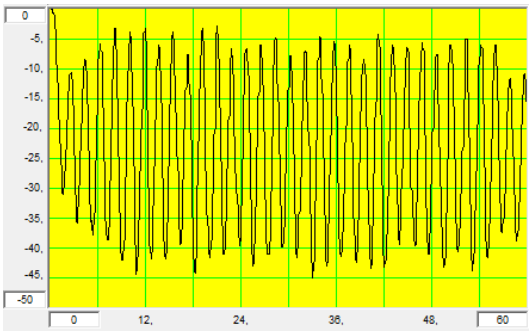

Figura 15. Angle (5º/) vs Temps (6s/), (Kp=0,4; Ki=0,286; Límit positiu=35; Límit negatiu=-5; Consigna=-20º).

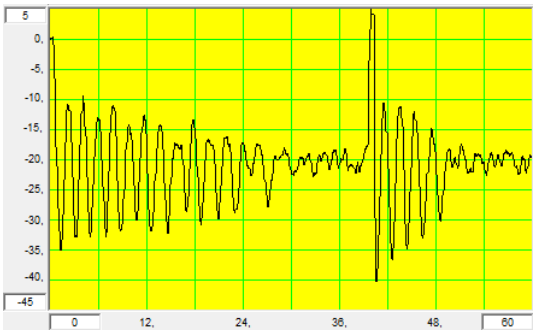

Figura 16. Angle (5º/) vs Temps (6s/), (Kp=0,4; Ki=0,13; Límit positiu=75; Límit negatiu=-5; Consigna=-20º).

A partir de les figures 15 i 16 es pot veure clarament com, a mesura que augmenta el límit positiu de la integral, l'amplitud de les oscil·lacions és menor i més aviat s'arriba un estat més o menys estable.

El valor del guany integral s'ha canviat en cada cas perquè al multiplicar-lo pel límit de la integral doni un valor similar en els dos casos i així poder apreciar l'efecte.

S'ha de tenir en compte que les figures anteriors només són un exemple per mostrar l'efecte dels límits de la integral. Els valors dels guany utilitzats no són els definitius.

Els valors finals s'han obtingut a partir del valor definitiu del guany proporcional i integral que es poden veure en el subapartat 6.1.2 de la memòria.

Annex D: Efecte del filtre complementari

En aquest annex es mostra un exemple de l'efecte que causa utilitzar el Filtre Complementari en la lectura de l'angle d'inclinació.

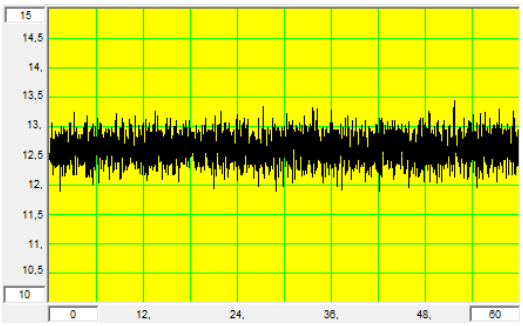

Figura 17. Angle (0,5º/) vs Temps (6s/) sense filtre.

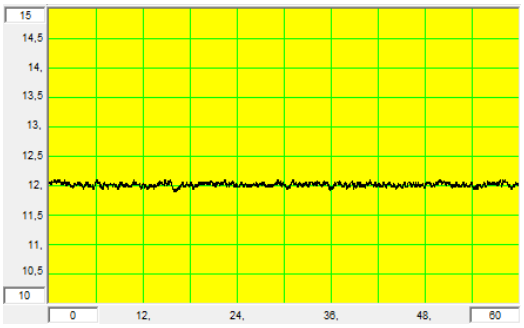

Figura 18. Angle (0,5º/) vs Temps (6s/) amb Filtre Complementari.

En la figura 18 es pot veure clarament una dràstica reducció de soroll respecte a la figura 17. A més a més, l'absència de soroll en la figura 18 dóna lloc a un angle molt més precís i més proper a la realitat.

Bibliografia

[1] Arduino.cc. (2016). *Arduino - PWM*. [En línia] Disponible a: https://www.arduino.cc/en/Tutorial/PWM [Data de consulta: abr. 2016].

[2] Arduino.cc. (2016). *Arduino - SecretsOfArduinoPWM*. [En línia] Disponible a: https://www.arduino.cc/en/Tutorial/SecretsOfArduinoPWM [Data de consulta: abr. 2016].

[3] Electroensaimada DIY, Raspberry Pi, Arduino. (2016). *Tutorial Arduino PWM*. [En línia] Disponible a: http://www.electroensaimada.com/pwm.html [Data de consulta: abr. 2016].

[4] Robologs.net. (2016). *Programación de un ESC con Arduino – robologs*. [En línia] Disponible a: http://robologs.net/2016/02/01/programacion-de-un-esc-con-arduino/ [Data de consulta: abr. 2016].

[5] Mechatronics Project Site. (2012). *ESC Calibration & Programming*. [En línia] Disponible a: http://robots.dacloughb.com/project-2/esc-calibration-programming/ [Data de consulta: abr. 2016].

[6] Invensense.com. (2016). *MPU-6050 | InvenSense*. [En línia] Disponible a: https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/ [Data de consulta: abr. 2016].

[7] Playground.arduino.cc. (2016). *Arduino Playground - MPU-6050*. [En línia] Disponible a: http://playground.arduino.cc/Main/MPU-6050 [Data de consulta: abr. 2016].

[8] Robologs.net. (2014). *Tutorial de Arduino y MPU-6050 – robologs*. [En línia] Disponible a: http://robologs.net/2014/10/15/tutorial-de-arduino-y-mpu-6050/ [Data de consulta: abr. 2016].