Sistema Embebido de Control de Rumbo y Altitud para Aeronaves No Tripuladas UAV

Ing. Santiago N. Reynoso, Ing. Esp. Gabriel Caballero, Ing. Diego Llorens

Especialización en Sistemas Embebidos, Departamento de Mecánica Aeronáutica
Instituto Universitario Aeronáutico, Córdoba, Argentina.

***Resumen*—Esta publicación presenta la implementación de un sistema embebido de control de rumbo y altitud para una aeronave no tripulada UAV. El sistema de control se diseñó a partir de los modelos propuestos en el libro *Automatic Flight Control Systems*[1]. El sistema se programó en el hardware del autopilto ArduPilotMega, del cual se extrajo la base del cdigo fuente ArduPilotMega 1.02 [2] para la implementación de un nuevo código con sistema de control de rumbo y altitud. Mediante ensayos en vuelo se verificó el funcionamiento del sistema de control y se realizaron los respectivos ajuste de las ganancias de control.**

***Palabras Claves*- Autopiloto, Control Rumbo y Altitud, UAV.**

I. INTRODUCCIÓN

En el desarrollo de los UAV (*Unmanned Aerial Vehicle*) una parte esencial del sistema, es el diseño del software del autopiloto. Dicho software es el encargado de controlar la aeronave para que el mismo realice las maniobras de vuelo que permitan cumplir con las misiones programadas. Las maniobras de vuelo básicas para el cumplimiento de las misiones, son las de control de rumbo y control de altitud.

En la actualidad la información bibliográfica disponible respecto a los sistemas de control aplicados en aviones UAV es escaza y de difícil acceso debido a que se trata de una tecnología nueva en pleno auge y de alto crecimiento a nivel mundial en los últimos años. Dada la importancia del estudio de los sistemas de control en aviones UAV el Departamento de Mecánica Aeronáutica del Instituto Universitario Aeronáutico propone el análisis de aplicación del sistema de control descripto en el libro *Automatic Flight Control Systems*[1] utilizado en aviones convencionales para ser aplicados en aviones UAV de pequeño tamaño denominados UAV Clase I o SUAV (*Small Unmanned Aerial Vehicle*)[3].

En la actualidad existen una amplia lista de sistemas de autopilotos disponibles en el mercado que son de tipo código y hardware abierto tales como: ArduPilotMega, UAV DevBoard, Paparazzi UAV, PX4 FMU. La ventaja de trabajar con sistemas abiertos es que permiten realizar el estudio de su funcionamiento y la posible modificación del código fuente.

Para la realización del siguiente trabajo se seleccionó el uso del autopiloto ArduPilotMega dado que es de bajo costo (Factor importante ya que se debe considerar la posibilidad de la pérdida del hardware en los ensayos en vuelo), posee todos los sensores necesarios para realizar la navegación (acelerómetro 3 ejes, girómetro 3 ejes, magnetómetro 3 ejes, GPS, telemetría de datos, anemómetro y barómetro) y dado su de volumen, peso y consumo de energía lo hace apto para ser utilizado en el avión disponible en el Departamento de Mecánica Aeronáutica con el cual se prevé la realización de los ensayos en vuelo.

El código fuente del ArduPilotMega tiene implementado un control de rumbo y altitud de tipo PID (Control Proporcional Integral y Derivativo [3]) el cual actúa sobre las señales de ángulos de actitud de rolido y cabeceo, y los valores de derivada e integración de dichas señales.

En el presente trabajo se propone como objetivo implementar un sistema de control de tipo de lazos anidados el cual permita controlar rumbo y altitud mediante un sistema de control que actúe sobre las velocidades de rolido y cabeceo dado por la acción de las superficies de control (elevador y alerón). Las velocidades de rolido y cabeceo generan cambios en los ángulos de rolido y cabeceo donde actuaria el segundo lazo de control. Por último, el tercer lazo de control actúa sobre los cambios de rumbo y altitud producidos por los cambios de ángulo de rolido y cabeceo respectivamente. Dicha arquitectura de control es la que plantea el libro *Automatic Flight Control Systems*[1]. La ventaja de implementar este tipo control es la de que el sistema actúa directamente sobre las señales de velocidad y posición angular del avión, a diferencia del sistema de control PID que solo actúa sobre las señales de posición angular y sus valores de derivada e integración.

En este trabajo se describe el diseño e implementación del software del autopiloto en el cual se implementa el sistema de control propuesto en el párrafo anterior.

Dicho nuevo software parte de la base del código del autopiloto ArduPilotMega[2], del cual se han extraído las librerías principales y el núcleo principal del programa. Partiendo de este código base, se plantea el desafío de implementar un sistema de control de rumbo y altitud de mayor complejidad descripto en el libro *Automatic Flight Control Systems*[1].

S-R. Depto. Mecánica Aeronáutica, Instituto Universitario Aeronáutico, Av. Fuerza Aérea Km 6,5. Córdoba, Argentina.(e-mail: sreynoso@iua.edu.ar).

G-C. Depto. Electrónica, Instituto Universitario Aeronáutico, Av. Fuerza Aérea Km 6,5. Córdoba, Argentina.(e-mail: gcavallero@invap.com).

D-L. Depto. Mecánica Aeronáutica, Instituto Universitario Aeronáutico, Av. Fuerza Aérea Km 6,5. Córdoba, Argentina.(e-mail: idllorens@gmail.com).

Para el desarrollo del trabajo se adoptó el modelo Evolutivo Espiral como modelo de ciclo de vida, ya que el mismo permite realizar mejoras por cada ciclo de la espiral, permitiendo asi realizar futuras evoluciones del software, agregando funcionalidades tales como navegación por puntos preestablecidos (*Way Points*), aterrizaje y despegue en modo automático (Ver Fig. 1).

Por lo que en el primer ciclo de la espiral se realiza el desarrollo del Sistema Embebido de Control de Rumbo y Altitud del autopiloto.

Para cada ciclo de la espiral se planteó trabajar acorde al Modelo de Prototipos, teniendo en cuenta de que se trata de un proyecto de corto tiempo y pocos recursos. La construcción del prototipo permite la evaluación del sistema para la retroalimentación, con el fin de refinar los requerimeintos del software implementado.

Fig. 1. Modelo Evolutivo Espiral del desarrollo del proyecto.

II. Descripción General del Sistema de Control

A. *Descripción del Sistema de Control de Altitud y Rumbo*

Para la implementación del sistema control de rumbo y altitud se parte del desarrollo teórico descripto en el libro *Automatic Flight Control Systems*[1]. En el libro se describe el sistema de control de altitud de una aeronave, el cual fue modificado a fin de obtener un modelo que permita ser implementado en un sistema embebido.

En la Fig. 2 podemos observar el diagrama en bloques del modelo del sistema de control de altitud. La señal de referencia h_{ref} representa la altura deseada a la cual se pretende que el avión vuele. A dicha señal se le resta el valor de la altura actual h_m medida por el altímetro, para obtener la señal de error de altura, la cual sera la señal de entrada a la ganancia de control K_c. A la señal de salida de K_c, es decir al ángulo de cabeceo θ_{comm} se le resta la señal proveniente del ángulo de cabeceo obtenida del sensor para obtener el ángulo de cabeceo de error θ_E. Luego se implementa un controlador PI ya que solo con un control proporcional no es suficiente debido al error en estado estacionario que presenta el control aplicado en dicha planta. La señal de salida del controlador PI, sera la señal de entrada a la función de transferencia del servo actuador que genera como señal de salida el ángulo de deflexión δ_E de la superficie de control del empenaje horizontal del avión. Dada la variación de δ_E se producirá un cambio en la dinámica del avión, mediante la cual hara que el mismo cambie su altitud de vuelo.

TABLA I
UNIDADES DE LOS SISTEMAS DE CONTROL DE AERONAVES

Símbolo	Descripción
h	Altura de la aeronave [m]
θ	Ángulo de cabeceo [°]
δ	Ángulo de deflexión de la superficie de control [°]
γ	Ángulo de planeo [°]
q	Velocidad de cabeceo [°/s]
Φ	Ángulo de rolido [°]
p	Velocidad de rolido [°/s]
ψ	Ángulo de guiñada [°]

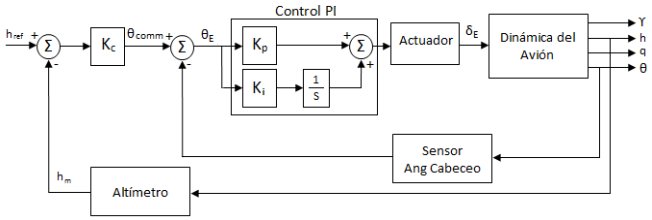


Fig. 2. Diagrama en bloques del sistema de control de altitud.

En la Fig. 3 se observa el diagrama en bloques del sistema de control de rumbo. Si bien el sistema de control de rumbo es similar al control de altitud, el mismo plantea que la ganancia que actúa sobre la velocidad de rolido, no se encuentra en el lazo de realimentación. En este caso el sistema de control de rumbo se basa en tres lazos de control anidados. En el primero de los lazos se controla la velocidad de rolido p_A mediante el ajuste de la ganancia de control K_δ. En el segundo lazo se controla el ángulo de rolido Φ mediante la ganancia de control K_Φ. Por último el lazo externo realiza el control de ángulo de rumbo ψ mediante el ajuste de la ganancia de control K_ψ. Este tipo de control permite controlar el rumbo del avión, teniendo en cuenta no solo el ángulo de rolido, sino también la velocidad angular de rolido. Dicho sistema de control trabaja teniendo en cuenta tanto la dinámica como la cinemática del avión.

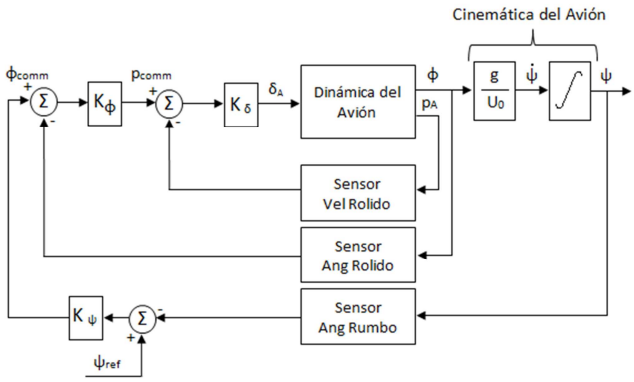


Fig. 3. Diagrama en bloque del sistema de control de rumbo.

B. *Descripción del Hardware del ArduPilotMega*

El hardware a utilizar para la implementación del software es el autopiloto ArduPilotMega. El mismo posee un microcontrolador Atmega 2560 de 8 bits y los sensores necesarios para programar el software del sistema de control. Posee un girómetro de tres ejes IDG500/ISZ500, mediante el cual se obtienen las velocidades de rolido y cabeceo. Mediante el uso del girómetro y el acelerómetro de tres ejes ADXL335, el software de ArduPilotMega realiza el cálculo de AHRS (*Attitude Heading Reference System*) con el que se obtienen los ángulos de actitud de rolido Φ y cabeceo θ. La altura de vuelo se obtiene mediante el sensor de presión barométrica BMP085.

Por último el rumbo de la aeronave se estima mediante el uso de un magnetómetro HMC5883L.

Para la transmisión de datos el autopiloto utiliza los módulos de telemetría XBee Pro de 900Mhz con potencia de transmisión de 50mW. Con dicho sistema se transmiten los datos de navegación y actitudes del avión en vuelo.

El hardware de ArduPilotMega posee además una memoria EEPROM AT45DB161B, mediante la cual se pueden almacenar todos los datos de vuelo para el posterior análisis del funcionamiento y ajuste del sistema de control.

Las dimensiones del hardware del autopiloto son de 105x40x20mm con un peso de 50g, lo que lo hace beneficioso para ser utilizado en el avión disponible en el Departamento de Mecánica Aeronáutica, el cual se describe más adelante.

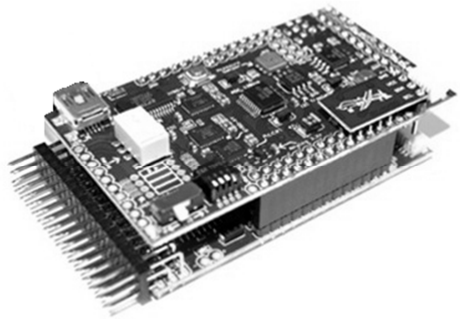


Fig. 4. Autopiloto ArduPilotMega 1.

C. *Descripción del Software de ArduPilotMega*

El código ArduPilotMega está escrito en lenguaje C++ el cual se compila y se carga en el microcontrolador del autopiloto mediante el uso del entorno de desarrollo de programación Arduino. Si bien Arduino no admite la posibilidad de uso de gestión de interrupciones, esto no limita a la realización de un sistema de autopiloto, como ya quedó demostrado en los códigos desarrollado por los programadores de ArduPilotMega.

En la Tabla II se describen los archivos extraídos del autopiloto ArduPilotMega. Estos archivos contienen las funciones principales con los cuales se elabora el código base a partir del cual se agregan las funciones para la programación del sistema de control[4].

TABLA II
ARCHIVOS EXTRAÍDOS DEL AUTOPILOTO ARDUPILOTMEGA

ARCHIVOS	DESCRIPCIÓN
ArduPilotMega.pde	Contiene el bucle principal del programa.
AHRS.pde	Contiene las funciones del horizonte artificial que calcula la actitud del avión.
Attitude.pde	Contiene las funciones que transforman la salida de los lazos de control en señales para los actuadores (servos).
EEPROM.pde	Contiene las funciones para leer y escribir en la memoria EPPROM del autopiloto.
GCS_Ardupilot.pde	Implementa un protocolo con caracteres ASCII de comunicación de telemetría.
log.pde	Contiene funciones para leer y escribir archivos de registro de datos en la memoria flash del piloto automático.
radio.pde	Contiene funciones para leer las señales enviados desde el radio control.
sensors.pde	Contiene las funciones para leer la presión absoluta y la velocidad anemométrica.
setup.pde	Contiene funciones para cargar los valores de configuración por defecto durante la inicialización del piloto automático.
system.pde	Contiene las funciones necesarias para realizar la inicialización del piloto automático en el suelo.
test.pde	Contiene diferentes funciones para probar el sistema de autopiloto utilizando comunicación por puerto serie conectado a una computadora.

III. PROGRAMACIÓN DE LAS FUNCIONES DEL SISTEMA DE CONTROL

A. *Implementación de los Modos de Vuelo*

Para la implementación del autopiloto se establecieron cinco modos de vuelo adicionales además del modo de vuelo Manual:

1. Modo Control de Ángulo de Rolido (`ROLIDO`).
2. Modo Control de Ángulo de Cabeceo (`CABECEO`).
3. Modo Control de Ángulo de Rumbo (`RUMBO`).
4. Modo Control de Altitud (`ALTITUD`).
5. Modo Control de Rumbo – Altitud (`RUMBALT`).

Estos modos de vuelo permiten ir verificando el funcionamiento de las diferentes partes que componen el sistema de control a medida que se incrementa la complejidad del mismo.

El primer modo de vuelo permite realizar un ensayo en vuelo con el fin de hacer solo el ajuste de las ganancias K_δ y K_Φ para controlar solamente el ángulo de rolido. De la misma manera el segundo modo permite el ajuste de las ganancias de control de ángulo de cabeceo K_p y K_i. De esta manera el ajuste de las ganancias de control de ángulo de rolido y cabeceo se realiza de manera desacoplada previo a realizar los ajustes de rumbo y altitud.

Los modos de control de ángulo de rumbo y control de altitud permiten realizar los vuelos de ensayo para ajustar las ganancias de control K_ψ y K_c ambas por separadas, de manera desacoplada y utilizando las ganancias de control de ángulo de rolido y cabeceo fijadas mediante el uso de los dos primeros modos de vuelo.

Finalmente, en el Modo de Control de Rumbo – Altitud, el autopiloto controla tanto el rumbo como la altura de la aeronave de modo acoplado, en cual se realizaría el último ajuste fino de las ganancias de control.

B. Implementación de la Señal de Referencia del Sistema de Control

Para realizar el ajuste del sistema de control es necesario implementar las señales de referencia de entrada al sistema, a fin de evaluar cómo responde el sistema frente a una señal conocida.

En todos los modos de vuelo, se implementó la posibilidad de que la señal de referencias de los sistemas de control puedan ser generadas de dos maneras: mediante el stick del radio control o mediante una señal escalón generada de manera automática al entrar en el modo seleccionado.

La señal escalón es una señal muy utilizada en los sistemas de control para analizar la respuesta del sistema, ya que la misma permite evaluar diferentes tiempos y tipo de respuestas. Para la utilización de dicha señal se configura el tiempo en bajo donde la referencia se mantiene constante y mediante otra variable se selecciona la amplitud del salto de la señal de referencia (Ver Fig. 5).

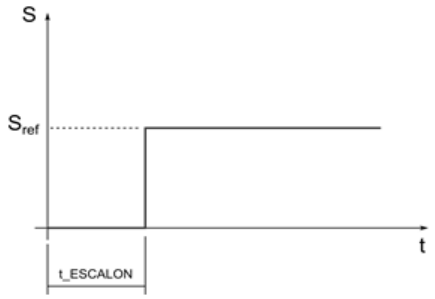


Fig. 5. Señal de control escalón.

```
// CONFIGURACION SEÑAL ESCALON DE REFERENCIA
#define ANG_CABECEO  8  // Salto en grados
#define ANG_ROLIDO   15 // Salto en grados
#define D_RUMBO      45 // Salto en grados
#define D_ALTURA     15 //Saltos en metros
#define t_ESCALON    201// (t = t_ESCALON x 20meg)
```

Mediante el uso de la opción MODO_STICK, la señal de referencia se modifica en tiempo real mediante el uso del stick del radio control, con el cual se selecciona el ángulo de rumbo o altitud de referencia que se desea que al autopiloto controle.

Las señales de referencias se implementan mediante cuatro funciones:

- Función de Señal de Referencia Ángulo de Rolido
- Función de Señal de Referencia Ángulo de Cabeceo
- Función de Señal de Referencia Ángulo de Rumbo
- Función de Señal de Referencia de Altitud

A continuación se muestra la implementación de las cuatro funciones que generan las señales de referencia del sistema de control para cada uno de los modos de vuelo.

```
/////////////// REF. ANGULO DE ROLIDO /////////////////
void RA_phi_ref_RC(void){

#if SELECT_REF == MODO_STICK
   Phi_Ref+=(radio_in[CH_ROLL]-radio_trim[CH_ROLL])*
   (delta_phi_RA/(float)(radio_max[CH_ROLL]-
   radio_trim[CH_ROLL]));

#else SELECT_REF == MODO_ESCALON
     cont_phi++;

     if(cont_phi == t_ESCALON){
        neg_phi = -1*neg_phi;
        Phi_Ref = neg_phi*ANG_ROLIDO;
     }
#endif

   // LIMITACION DE LOS ANGULOS DE ROLIDO
   Phi_Ref = constrain(Phi_Ref,-MAX_PHI_RA,MAX_PHI_RA);
}
/////////////// REF. ANGULO DE CABECEO /////////////////
void RA_theta_ref_RC(void){

#if SELECT_REF == MODO_STICK
   Theta_Ref+=(radio_in[CH_PITCH]-radio_trim[CH_PITCH])*
   (delta_theta_RA/(float)(radio_max[CH_PITCH]-
   radio_trim[CH_PITCH]));

#else SELECT_REF == MODO_ESCALON
   cont_theta++;

   if(cont_theta == t_ESCALON){
      neg_theta = -1*neg_theta;
      Theta_Ref = neg_theta*ANG_CABECEO;
   }
#endif

   // LIMITACION DE LOS ANGULOS DE CABECEO
   Theta_Ref=constrain(Theta_Ref,-
   MAX_THETA_RA,MAX_THETA_RA);
}
/////////////// REF. ANGULO DE RUMBO /////////////////
void RA_psi_ref_RC(void){

#if SELECT_REF == MODO_STICK
   psi_ref_RA+=(radio_in[CH_ROLL]-radio_trim[CH_ROLL])
   *(delta_psi_RA/(radio_max[CH_ROLL]-radio_trim[CH_ROLL]));

#else SELECT_REF == MODO_ESCALON
   cont_psi++;

   if(cont_psi == t_ESCALON)
      psi_ref_RA += D_RUMBO;
#endif

   // CONTROL DE CRUCE DE RUMBO DE 360 A 0 GRADOS
   if( psi_ref_RA < 0.0f )
      psi_ref_RA += 360.0f;
   else if( psi_ref_RA > 360.0f )
      psi_ref_RA -= 360.0f;
}
/////////////////// REF. ALTITUD //////////////////////
voidRA_alt_ref_RC(void){

#if SELECT_REF == MODO_STICK
   alt_ref_RA+=(radio_in[CH_PITCH]-radio_trim[CH_PITCH])
   *(delta_alt_RA/(radio_max[CH_PITCH]-
   radio_trim[CH_PITCH]));

#else SELECT_REF == MODO_ESCALON
   cont_alt++;

   if(cont_alt == t_ESCALON)
   alt_ref_RA += D_ALTURA;
#endif
```

```
   // LIMITACION DE ALTITUD DE VUELO

   alt_ref_RA = constrain(alt_ref_RA, MIN_ALT_RA,
   MAX_ALT_RA);
}
```

C. Implementación de la Funciones del Sistema de Control

Para la implementación del sistema de control de rumbo y altitud se dividió al sistema en cuatro funciones:

- Función de Control de Ángulo de Rolido
- Función de Control de Ángulo de Cabeceo
- Función de Control de Ángulo de Rumbo
- Función de Control de Altitud

De esta manera cada modo de vuelo llama a las respectivas funciones de control.

La función de control de ángulo de rolido implementa los lazos internos de control de rumbo para controlar el ángulo de rolido ϕ como se muestra en la Fig. 6. Esta función se utiliza en el modo de control de rolido y forma parte del modo de control de rumbo.

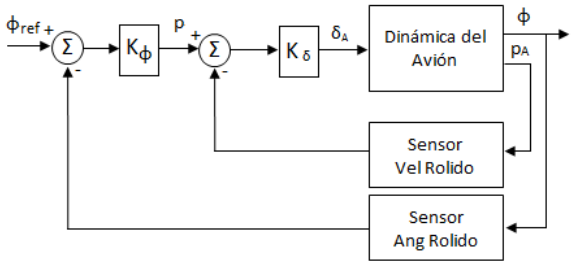


Fig. 6. Diagrama en bloque de control de ángulo rolido.

A continuación se muestra el código fuente de la implementación de la función de control de ángulo de rolido donde **roll_sensor** es el valor de ángulo de actitud en rolido del avión obtenido del AHRS, **Omega_Vector[0]** es la velocidad de rolido del avión y por último **PD_Ale** es la señal que controla el servo actuador de los alerones.

```
////////////// CONTROL ANGULO DE ROLIDO //////////////
void RA_AngRoll(void){
   Er_Phi  = Phi_Ref - roll_sensor / 100.0f;
   PD_Ale  = Er_Phi  * KR_phi;
   PD_Ale  = (PD_Ale-ToDeg(Omega_Vector[0]))*KR_d;
   PD_Ale  = constrain(PD_Ale,-45,45);
}
```

De manera similar se implementó el lazo de control interno del sistema de control de altitud el cual controla el ángulo de cabeceo θ, como se muestra en la Fig. 7.

En este caso el control de ángulo de cabeceo θ tiene un integrador para compensar el error en estado estacionario que presenta la dinámica del avión.

Esta función se llama en el modo de control de cabeceo y también forma parte de las funciones que se llaman en el modo de control de altitud.

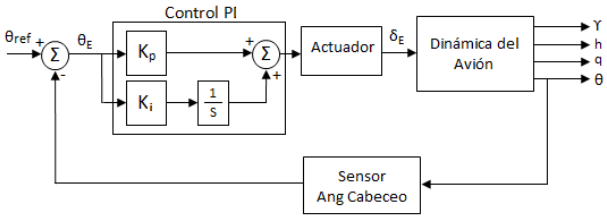


Fig. 7. Diagrama en bloque del control de ángulo de cabeceo.

A continuación se muestra la implementación del sistema de control de cabeceo donde **pitch_sensor** indica el ángulo de actitud en cabeceo del avión obtenido por el AHRS, en **I_Er_Theta** se realiza la integración de la señal de error de θ y por último **PD_Elev** es la señal de control del servo del emepenaje horizontal del avión.

```
//////////////// CONTROL ANGULO CABECEO ////////////////
void RA_AngPitch(void){
   Er_Theta = Theta_Ref - pitch_sensor / 100.0f;
   PD_Elev  = Er_Theta  * KH_p ;

   I_Er_Theta += Er_Theta*KH_i*(deltaMiliSeconds/1000.0);

   PD_Elev += I_Er_Theta;
   PD_Elev  = constrain(PD_Elev,-45,45);
}
```

La función que implementa el sistema de control de rumbo solo controla sobre el ángulo ψ y genera la señal de ϕ_{ref} la cual servirá como señal de entrada a la función de control de ángulo de rolido descripta anteriormente (Ver Fig. 8).

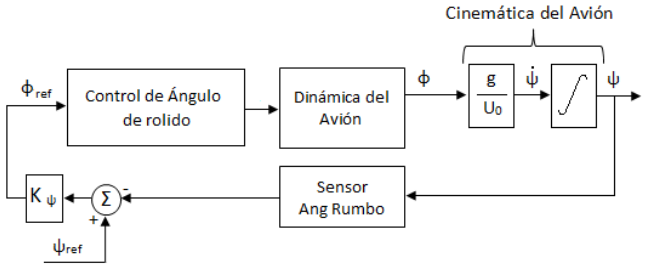


Fig. 8. Diagrama en bloque del control de ángulo de rumbo.

De esta manera el modo control de rumbo primero llama a la función control de rumbo y una vez obtenida la señal de ϕ_{ref} llama a la función control de rolido.

En el cuadro siguiente se muestra la implementación del sistema de control de rumbo, donde se llama la función **RA_Calculo_Heading** la cual estima el rumbo actual de la aeronave mediante magnetómetro. Se calcula la señal error de ψ mediante la diferencia del rumbo actual y el rumbo de referencia, luego se multiplica la señal por la ganancia de control KR_psi. Esta señal será la señal de referencia de ángulo de rolido ϕ_{ref} para la función de control de ángulo de rolido.

```
/////////////// CONTROL ANGULO DE RUMBO ///////////////
void RA_AngRumb(void){
   RA_Calculo_Heading();
   Er_Psi = psi_ref_RA - psi_sen_RA;

   if( Er_Psi > 180 )
      Er_Psi -= 360.0f;
   else if( Er_Psi < -180 )
      Er_Psi += 360.0f;

   Phi_Ref = Er_Psi * KR_psi;
   Phi_Ref = constrain(Phi_Ref, -MAX_PHI_RA, MAX_PHI_RA);
}
```

Del mismo modo que en el control de rumbo, en el control de altitud se implementa la función de control que solo actua sobre la altura h. Dicha función genera el θ_{ref} que es la señal de entrada de la función de cotrol de ángulo de cabeceo (Ver Fig. 9). Por lo que en el modo de control de altitud se llama primero a la función de control de altitud y luego se llama la función de control de ángulo de cabeceo.

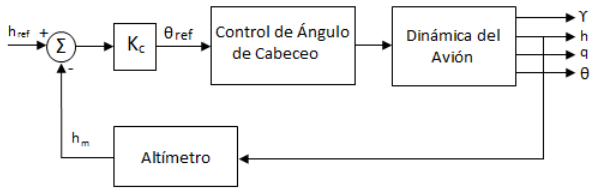


Fig. 9. Diagrama en bloque de control de altitud.

A continuación se muestra la función de control de altitud donde comienza por estimar la señal de error de altitud. La altitud a la que se encuentra el avión es obtenida mediante la diferencia de altitud que mide el barómetro por diferencia de presión denominada **current_loc.alt** y la altitud medida al momento del despegue almacenada en **ground_alt**.

```
////////////////// CONTROL DE ALTITUD //////////////////
void RA_Altitude(void){
   Er_Alt = alt_ref_RA-(current_loc.alt-ground_alt);
   Theta_Ref = Er_Alt * KH;
   Theta_Ref = constrain(Theta_Ref,-MAX_THETA_RA,
   MAX_THETA_RA);
}
```

De esta forma quedan implementadas las cuatro funciones de control que son llamadas por los distintos modos de vuelo del piloto automático.

IV. Sincronización de la Ejecución de las Funciones de Control

El código del autopiloto comienza por inicializar la configuración de todos los sistemas en la función **void setup()**. Luego entre un bucle infinito en el cual se van ejecutando todas las funciones que controlan el vuelo de la aeronave.

El bucle infinito se divide en dos bucles, uno al que se denomina **médium_loop()** y otro **fast_loop()** como se observa en la Fig. 10. En el **medium_loop()** se ejecuta un *switch case* de 5 casos con el fin de ejecutar funciones cada 100ms,

En el primer caso del *switch case* se ejecutan las funciones mediante la cual se estima el rumbo del avión utilizando GPS o magnetómetro.

En el segundo caso se llama la función **control_mode()** donde se ejecutan las funciones de control de rumbo y altitud, las cuales generaran las señales de referencia de ángulo de rolido y cabeceo necesarios para que el avión alcance el rumbo y la altitud de referencia.

Luego en el tercer caso se estima la altitud del avión mediante el uso del barómetro, continua con la función para grabar los datos en memoria y por último en el quinto caso se llama a un **slow_loop()** que actualmente no es utilizado por el autopiloto.

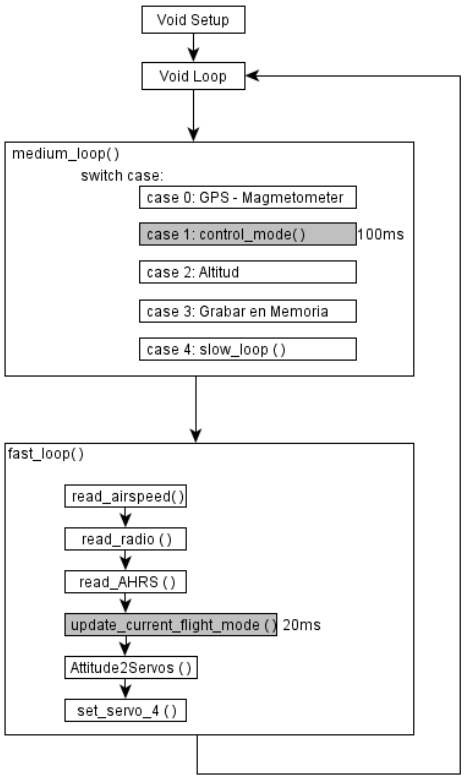


Fig. 10. Tiempo de ejecución de las funciones de control.

La función **fast loop()** se llama cada 20ms y es donde se llaman a las funciones de lectura de velocidad indicada del avión, lectura de las señales de radio control, cálculo de la actitud del avión mediante la función read_AHRS (Attitude Heading Reference System), luego llama a la función **update_current_flight_mode()** en la cual se ejecutan las funciones que controlan los ángulos de rolido y cabeceo de la aeronave.

Por último el **fast_loop()** termina por llamar a la funciones encargadas de posicionar los servos acorde a la actitud que estiman los controladores que debe tener el avión para realizar el control de ángulo de rolido y cabeceo.

De esta manera, la implementación del sistema de control queda dividida en dos partes, en una primera en la cual se controla el rumbo y la altitud cada 100ms y una segunda parte donde se controla el ángulo de rolido y cabeceo cada 20ms.

A continuación se muestra la implementación de la función **control_mode()** en donde se divide en tres posibles casos. El primer caso corresponde al modo de vuelo de control de rumbo y altitud, donde se llama las dos funciones de señal de referencia de selección de rumbo y altitud a controlar, y luego se llama a las funciones de control de ángulo de rumbo y altitud.

Los dos siguientes casos corresponden a los modos de vuelo de rumbo y altitud respectivamente. En cada uno de los casos, primero se llama a la respectiva función de señal de referencia y luego a la función de control.

```
switch(control_mode){
   case RUMBALT:
      RA_psi_ref_RC();   // Senal referencia de rumbo
      RA_alt_ref_RC();   // Senal referencia de altitud
      RA_AngRumb();      // Funcion Control de Rumbo
      RA_Altitude();     // Funcion Control de Altitud
      break;
   case RUMBO:
      RA_psi_ref_RC();   // Senal referencia de rumbo
      RA_AngRumb();      // Funcion Control de Rumbo
      break;
   case ALTITUD:
      RA_alt_ref_RC();   // Senal referencia de altitud
      RA_Altitude();     // Funcion Control de Altitud
      break;
}
```

El siguiente cuadro se muestra la implementación de la función **update_current_flight_mode().** En los primeros tres modos de vuelo, se llaman a las respectivas funciones ya sea para control de rolido como cabeceo, las cuales reciben las señales de referencias determinadas en la función **control_mode()**.

Los últimos dos casos corresponde a los modos de vuelo de control de rolido y control de cabeceo. En estos modos primero se llama a la respectiva función de referencia para luego llamar a la función de control correspondiente.

```
voidupdate_current_flight_mode(void)
{  switch(control_mode){
     case RUMBALT:
        RA_AngRoll();   // Controlar angulo de rolido PHI
        RA_AngPitch();  // Controlar angulo de cabeceo THETA
        break;
     case RUMBO:
        RA_AngRoll();   // Controlar angulo de rolido PHI
        break;
     case ALTITUD:
        RA_AngPitch();  // Controlar angulo de cabeceo THETA
        break;
     case ROLIDO:
        RA_phi_ref_RC();// Ingresar angulo de ref de rolido
        RA_AngRoll();   // Controlar angulo de rolido PHI
        break;
     case CABECEO:
        RA_theta_ref_RC(); // Ingresar angulo ref de cabeceo
        RA_AngPitch();  // Controlar angulo de cabeceo THETA
        break;
   }
}
```

V. Rango de Ganancias de Control con Matlab

Mediante el uso de Matlab se calcula el rango de las ganancias de control para luego ser ensayadas en vuelo[5].

Se crea un archivo *script* donde se carga las ecuaciones que modelan al avión [6]. Luego se discretizan las funciones con el tiempo de muestreo 20ms y 100ms que es el período de muestreo de las señales de control con las que corren los bucles medium_loop() y el fast_loop(). De esta manera se obtiene el modelo digital de la planta.

Mediante el análisis en lugar de raíces de las funciones de transferencias en el plano discreto, se estima el valor máximo y mínimo de las ganancias para mantener el sistema dentro del círculo unitario donde el mismo se mantiene estable[7].

Este análisis se realiza para cada uno de los modos de vuelo, por lo que se establecen las ganancias para los ensayos de control de ángulo de rolido, control ángulo de cabeceo, control de rumbo y control de altitud, cada análisis con su respectivo modelado de la función de transferencia del avión[6]. En las siguientes figuras se muestran los lugares de raíces estudiados para establecer los rangos de ganancias de los sistemas de control de ángulo de rolido y ángulo de cabeceo.

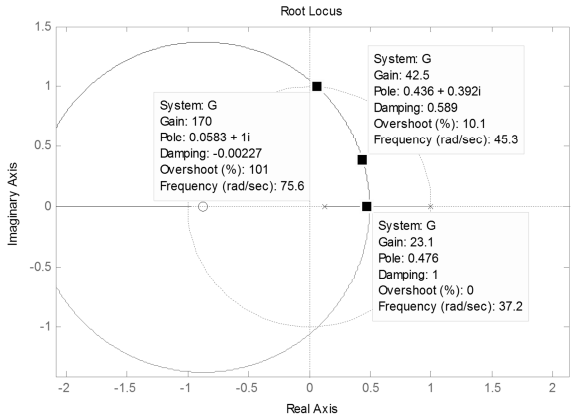


Fig. 11. Lugar de raíces del sistema para control del ángulo de rolido.

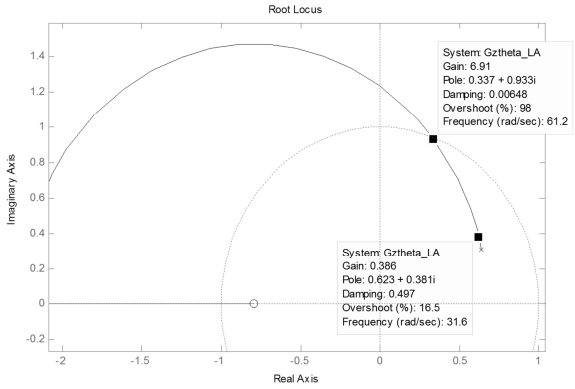


Fig. 12. Lugar de raíces del sistema para control del ángulo de cabeceo.

Una vez analizados los lugares de raíces se establecieron los valores de ganancia para ser ensayados en vuelo, los cuales se muestran en la Tabla III.

TABLA III
GANANCIAS DE LOS SISTEMAS DE CONTROL DE ROLIDO Y CABECEO

	K_ϕ	
Kδ	Min.	Max.
0.2	5	42
0.3	5	47
0.4	4	50

	K_p	
Ki	Min.	Max.
3.0	0.3	2.0
4.0	0.5	2.5
8.0	0.7	3.8

Del mismo modo se establecieron los rangos de ganancias del sistema de control de rumbo y el sistema de control de altitud mediante el análisis de los lugares de raíces correspondiente a cada función de transferencia del avión.

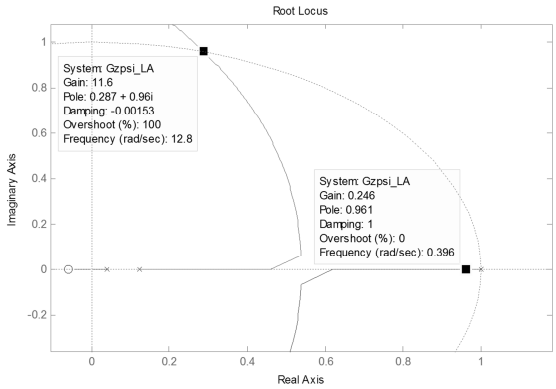


Fig. 13. Lugar de raíces del sistema para control del ángulo de rumbo.

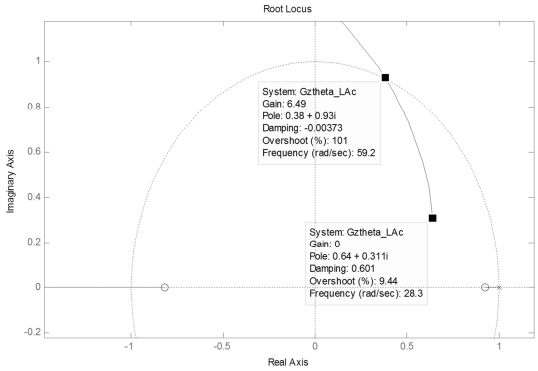


Fig. 14. Lugar de raíces del sistema para control de altitud.

Los valores de ganancias establecidos para los sistemas de control de rumbo y altitud se observan en la Tabla IV.

TABLA IV
GANANCIAS DE LOS SISTEMAS DE CONTROL DE RUMBO Y ALTITUD

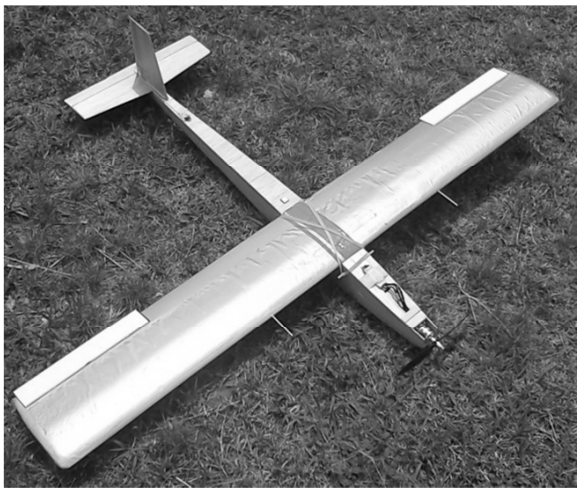
K_ψ	
Min.	Max.
0,5	10
K_c	
Min.	Max.
0,5	8

VI. ENSAYO EN VUELO DEL SISTEMA

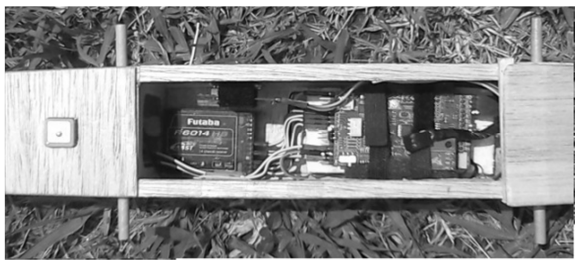
Una vez obtenidos los valores de ganancias para la realización de los ensayos, se realizaron los vuelos de prueba utilizando el avión KJOTA del Departamento de Mecánica Aeronáutica IUA (Ver Fig. 15).

TABLA V
CARACTERÍSTICAS GENERALES AVIÓN KJOTA

Largo	1,06 m
Envergadura Alar	1,50 m
Peso	1,290 Kg
Motor	Eléctrico BL2215/20 1200rpm/V
Nª Canales	4



A)



B)

Fig. 15. A) Avión para ensayo en vuelo KJOTA, B) Instalación del autopiloto ArduPilotMega en el avión KJOTA.

El primer vuelo se configuró con los modos de vuelo manual y modo control ángulo de rolido. Durante los ensayos se evaluaron los valores de ganancias preestablecidos con Matlab hasta allar la combinación de las ganancias K_ϕ y K_δ óptimas para el control de rolido. El ajuste de las ganancias se realizó utilizando como señal de referencia una entrada escalón con un salto de señal de 0º a15º.

Mediante las ganancias K_ϕ = 20 y K_δ = 0.2 se obtuvo la mejor respuesta del sistema de control en ángulo de rolido, la cual se observa en la Fig. 16[8].

Del mismo modo se realizó el ensayo para el control de ángulo de cabeceo. Se configuraron los modos de vuelos manual y de control de ángulo de cabeceo, y se realizó el ensayo con los valores de ganancias preestablecidos. Ademas se configuró en señal de referencia para un salto de 8º y de -8º para realizar los ensayos del sistema.

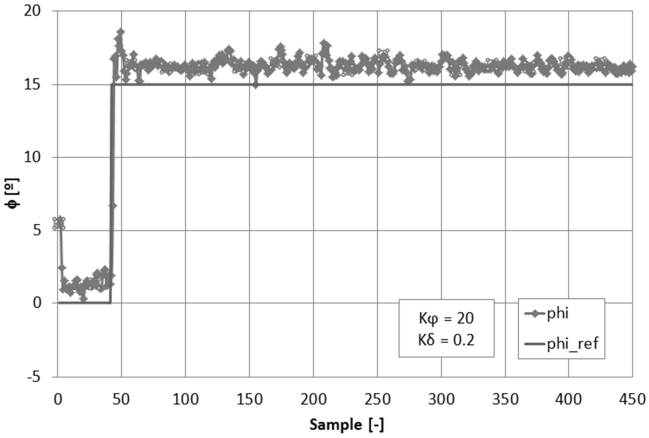


Fig. 16. Ensayo en vuelo de control de rolido con un salto de 15º.

Los resultados obtenidos en el sistema de control de ángulo de cabeceo se observa en la Fig. 17 y Fig. 18. Dicho resultado demuestra que el sistema entra en oscilaciones de ±5º cuando comienza a controlar el ángulo indicado por la señal de referencia y luegos estas oscilaciones se van atenuando hasta mantener el ángulo de cabeceo[9].

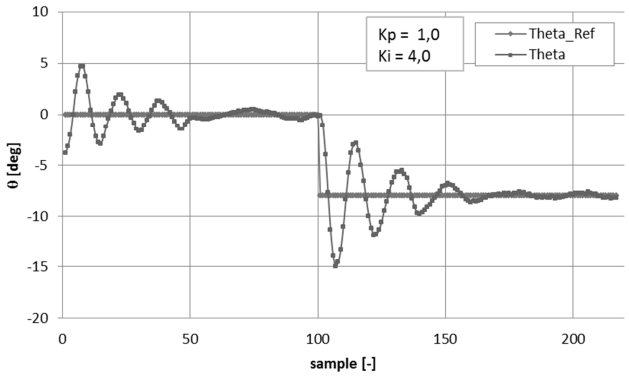


Fig. 17. Ensayo en vuelo de control de cabeceo con un salto de -8º.

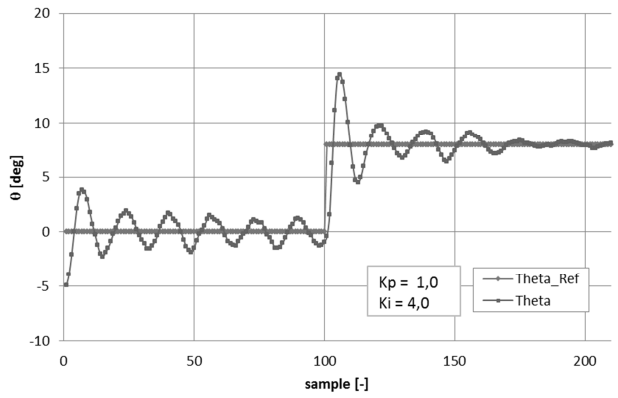


Fig. 18. Ensayo en vuelo de control de cabeceo con un salto de +8º.

Luego se realizó el ensayo para el ajuste en vuelo de la ganancia del controlador de rumbo. Se configuró nuevamente el avión para volar en el modo de control de rumbo y se configuró la señal de referencia para realizar un salto de 45º de rumbo con respecto al rumbo en que se encuentra volando la aeronave.

Utilizando los valores obtenidos mediante el análisis realizado en Matlab, se realizaron distintos vuelos hasta encontrar el valor de K_ψ = 1.2 como el valor de ganancia con el cual el sistema de control respondía de manera aceptable. Los resultados obtenidos se observan en la Fig. 19 donde el avión cambia de rumbo 90º a rumbo 135º. El sistema de control mantiene el rumbo con pequeñas oscilaciones de ±6º que pueden ser atribuidas a rafagas de viento, pero que aun así son aceptables para controlar el rumbo del avión KJOTA[10].

Por último se realizó el ensayo en vuelo del sistema de control de altitud. Para realizar el ensayo se configuró el sistema en el modo de control de altitud para controlar mediante una señal escalón, la cual en un principio mantenie la altura en la que se encuentra volando el avión y luego realiza un salto de 45m. Los resultados del ensayo se observan en la Fig. 20. Mediante la realización de estos ensayos se hizo evidente la necesidad de que el sistema cuente con un control de velocidad indicada IAS (*Indicated Air Speed*) para el correcto control de altitud ya que es un factor indispensable para controlar la altitud de vuelo[11].

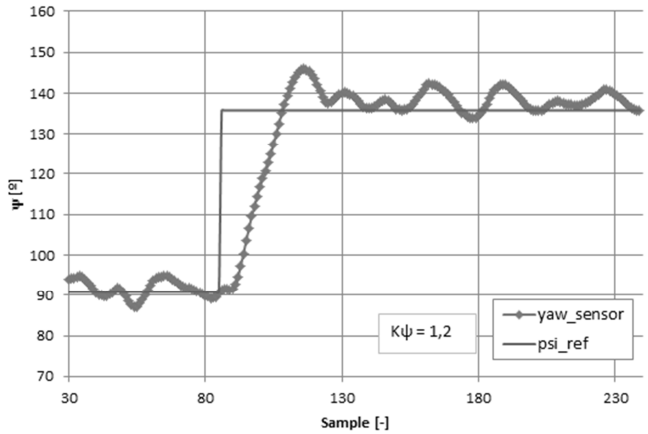


Fig. 19. Ensayo en vuelo de control de rumbo con un salto de 45º.

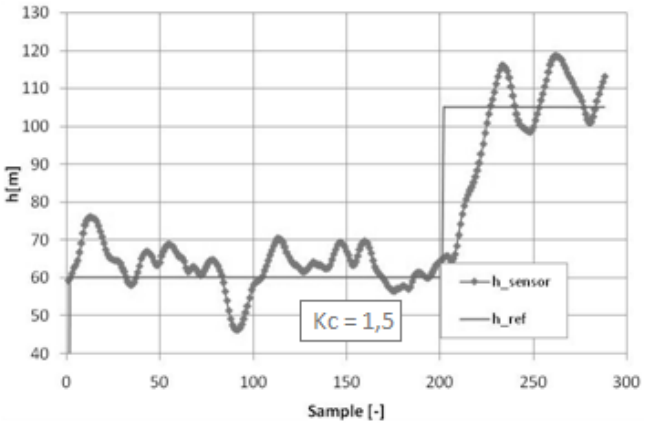


Fig. 20. Ensayo en vuelo de control de altitud con un salto de 45m.

VII. Conclusión

En la finalización del trabajo se puede concluir que se ha logrado diseñar e implementar un sistema embebido de control de rumbo y altitud en la plataforma de ArduPilotMega.

Utilizando la base del código del autopiloto ArduPilotMega fue posible reescribir un nuevo código en el bucle principal con fin el de implementar el sistema de control de rumbo y altitud acorde al propuesto por el libro *Automatic Flight Control Systems* [1].

Mediante la utilización de las herramientas de Matlab se logró definir los rangos de ganancias del sistema de control a ser ajustadas en vuelo. La definición de los rangos de las ganancias fue de suma importancia para la realización de los ajustes en vuelo dado que los modelos de control presentan lazos anidados, y la sintonización de este tipo de ganancias no es posible sin el estudio previo de los posibles valores y sus límites de estabilidad.

Realizados los ensayos en vuelo se logró realizar el ajuste fino de las ganancias del sistema de control para obtener un funcionamiento correcto del sistema embebido de control, salvo en el control de altura en el cual se hace notoria la necesidad de agregar un sistema un control de velocidad IAS.

Queda como trabajo a futuro la implementación de un sistema de control de velocidad IAS para ser utilizado en un nuevo sistema de control de altura el cual controle no solo el ángulo de cabeceo, sino también la velocidad IAS que le permita realizar un ascenso controlado.

Otras aplicaciones a desarrollar son el sistema de navegación por puntos preestablcidos (*Way Points*) utilizando los sitemas de control ya desarrollados y mejorando el sistema de control de altura. Se puede implementar a fututro el modo de emergencia Return To Land el cual utilizan los UAV en caso de emergencia para que el vehículo vuelva de forma autónoma al lugar de despegue, como así también implementar los sistemas de control para la realización de aterrizaje y despegue en modo automático.

VIII. Bibliografía

[1] D. McLean, Automatic Flight Control Systems, Pretince Hall, 1990.

[2] «DiyDrones - Ardupilot Mega,» Official ArduPlane Repository, 2012. [En línea]. Available: https://code.google.com/p/ardupilot-mega/downloads/list?can=2&q=1.02&colspec=Filename+Summary+Uploaded+ReleaseDate+Size+DownloadCount.

[3] K. Ogata, Ingenieria de Control Moderna, Pearson Alhambra, 2003.

[4] D. Llorens, «Descripcion General del Funcionamiento del Programa Ardupilotmega,» IUA - DMA, Cordoba, 2011.

[5] «MathWorks SISOTOOL,» Matlab, 2013. [En línea]. Available: http://www.mathworks.com/help/control/ref/sisotool.html.

[6] D. Llorens, «Funciones de Tramsferencia Avión KJOTA,» IUA, 2013.

[7] K. Ogata, «Sistemas de Control en Tiempo Discreto,» Prentice Hall, 2001.

[8] D. Llorens y S. Reynoso, «Ajuste de Ganancias y Ensayo en Vuelo del Sistema de Control de Rolido,» IUA - DMA, Córdoba, 2013.

[9] D. Llorens y S. Reynoso, «Ajuste de Ganancias y Ensayo en Vuelo del Sistema de Control de Cabeceo,» IUA - DMA, Córdoba, 2013.

[10] D. Llorens y S. Reynoso, «Ajuste de Ganancias y Ensayo en Vuelo del Sistema de Control de Rumbo,» IUA - DMA, Córdoba, 2013.

[11] D. Llorens y S. Reynoso, «Ajuste de Ganancias y Ensayo en Vuelo del Sistema de Control de Altitud,» IUA - DMA, Córdoba, 2013.

[12] P. Y. O. Kimon P. Valavanis, Unmanned Aircraft Systems, Springer, 2009.

[13] R. Austin, Unmanned Aircraft Systems, Wiltshire: Wiley, 2010.

[14] J. H. Blakelock, Automatic Control of Aircraft and Missile, New York: Wiley Interscince Publication, 1990.