En esta guía se detalla la localización de los principales parámetros del piloto automático y cómo afectan algunos de ellos al funcionamiento del programa.

Los datos de los vuelos se guardan en la carpeta Visual\bin\Debug\Logs

Namespace GroundStation

Carpeta: Configuration. Clase: *A/Cname*ACPerformance (ej. C172ACPerformance, CirrusACPerformace, etc.) derivada de ACPerformance:

StallSpeed: velocidad de pérdida en nudos (kt), los ángulos de cabeceo y alabeo se reducirán mientras la velocidad de la aeronave no esté por encima de la velocidad de pérdida más un margen de seguridad. Este margen se encuentra en las clases Altitude y Heading en la carpeta Navigation:

```
if(currTas < vStall + 5)
                    {

                        if (currRoll > 1)
                            ans = currRoll -1;
                        else if (currRoll < 1)
                            ans = currRoll + 1;
                        else
                            ans = 0;
                        Console.WriteLine("WARNING: Heading change not
set due to low airspeed --> roll reference set to " + ans + " degrees");
                        this.pid.SetRef(PIDManager.Ctrl.ROLL, ans);
                        this.pid.SetRef(PIDManager.Ctrl.YAW,
this.selHeading);
                    }
```

MaxBank: ángulo máximo de alabeo en grados.

Carpeta: Configuration. Clase: *A/Cname*PidConfig (ej. C172PidConfig, CirrusPidConfig, etc.) derivada de PidConfig:

imuTs: tiempo de muestreo de la unidad inercial

adcTs: tiempo de muestreo de ADC

Los siguientes parámetros son precedidos por una “t” para apuntar que se trata de los parámetros pertenecientes al acelerador (throttle), los mismos parámetros se encuentran precedidos por una ”p”, “r” o “y” para los ángulos de cabeceo (pitch), alabeo (roll) y guiñada (yaw) respectivamente.

tkp: constante proporcional

tki: constante integrativa

tkd: constante derivativa

tCh: número del canal

tOffset: este valor se resta a la salida del controlador PID

tSpan: este valor divide la salida del controlador PID después de restarle el Offset. Se utiliza para reducir la cantidad de bits necesarios para transmitir la señal de control. Al aumentar este valor se disminuye la resolución de la señal de control (mayores saltos entre valores)

tMin: valor mínimo de la salida del controlador PID

tMax: valor máximo de la salida del controlador PID

tMean: valor central en el rango de valores de la salida del controlador PID

tInitialRef: valor de la referencia por defecto. Este valor será la referencia del controlador al activarlo hasta que se introduzca una nueva referencia por parte del usuario o por una de las capas superiores.

Carpeta: Configuration. Clase: NavConfig y XNavConfig:

adcTs, akp, aki, akd, aInitialRef: ver apartado anterior.

gpsTs: tiempo de muestreo del GPS

MinPitch: ángulo de cabeceo mínimo permitido en modo dirigido y autónomo.

MaxPitch: ángulo de cabeceo máximo permitido en modo dirigido y autónomo.

MaxRoll: ángulo de alabeo máximo (a ambos lados) permitido en modo dirigido y autónomo.

OriginLat: latitud del punto inicial del plan de vuelo en grados.

OriginLon: longitud del punto inicial del plan de vuelo en grados.

FPLatitude: vector con la latitud de los puntos del plan de vuelo en grados.

FPLongitude: vector con la longitud de los puntos del plan de vuelo en grados.

Carpeta: Navigation. Clase: Altitude y XAltitude derivadas de UpperLayer:

deltaAlt: margen de error aceptable en el control de la altitud. Reducirlo a 0 puede producir oscilaciones en el control de altitud si el controlador PID no está bien configurado.

En la línea 61: `if(diffAlt > 0 && currTas < (this.ap.stallTas + 5))` El número 5 indica el **margen de seguridad en nudos que se la aplica a la velocidad de entrada en pérdida** por debajo del cual se reducirá el ángulo de cabeceo.

Carpeta: Navigation. Clase: ConsoleInput y XConsoleInput:

En la línea 139 se puede configurar el **rango de velocidades aceptado por la consola**:

```
if (v < 0 || v > 220)
                                {
                                        Console.WriteLine("Velocity    out
of range. It must be range between 0 and 220 knots");
```

En la línea 165 se puede configurar el **rango de altitudes aceptado por la consola**:

```
if (a < 0 || a > 10000)
                                {
                                        Console.WriteLine("Altitude    out
of range. It must be range between 0 and 10000 m");
```

En la línea 192 se puede configurar el **rango del ángulo de guiñada aceptado por la consola**:

```
if (ny < -25 || ny > 25)
                                {
                                                Console.WriteLine("Yaw  out
        of range. It must be range between -25 and 25 degrees");
```

En la línea 218 se puede configurar el **rango del ángulo de cabeceo aceptado por la consola**:

```
if (np < -15 || np > 15)
                                {
                                                Console.WriteLine("Pitch
        out of range. It must be range between -15 and 15 degrees");
```

En la línea 244 se puede configurar el **rango del ángulo de alabeo aceptado por la consola**:

```
if (nr < -25 || nr > 25)
                                {
                                                Console.WriteLine("Roll out
        of range. It must be range between -25 and 25 degrees");
```

Carpeta: Navigation. Clase: Heading derivada de UpperLayer:

En la línea 53: `if(currTas < vStall + 5)` El número 5 indica el **margen de seguridad que se la aplica a la velocidad en nudos** de entrada en pérdida por debajo del cual se reducirá el ángulo de cabeceo.

Carpeta: Navigation. Clase: LatNav y XLatNav:

En las líneas 137 y 138 podemos cambiar el **ángulo máximo entre el rumbo de la aeronave y el plan de vuelo**.

```
ans = ans > 30 ? ans = 30 : ans;
ans = ans < -30 ? ans = -30 : ans;
```

En la línea 189 podemos introducir la **variación magnética** (diferencia entre el rumbo al norte geográfico y al magnético)

```
az1 += -0.45;
```

En la línea 207 podemos configurar la **distancia de *fly by*** en metros. Esta será la distancia a la que la aeronave se tendrá que acercar a cada uno de los puntos de la ruta antes de dirigirse al siguiente. Reducirla demasiado puede

hacer que la aeronave no avance al siguiente punto y se quede volando alrededor de uno de los puntos.

```
if(dist < 200) //Modify here maximum distance to WP before turning
```

Carpeta: Navigation. Clase: Speed y XSpeed derivadas de UpperLayer:

deltaSpeed: margen de error aceptable en el control de la velocidad. Reducirlo a 0 puede producir oscilaciones en el control de velocidad si el controlador PID no está bien configurado.

Carpeta: Processing. Clase: PID

En las líneas 187 y 191 podemos configurar el **margen de velocidades** en nudos por encima y por debajo de la velocidad de referencia en el que actuará el controlador **PID del acelerador, fuera de ese margen se utilizará la máxima potencia por debajo y la mínima por encima.**

```
if (error > 10) //Slower than selected speed->max throttle

else if (error < -10) //Faster than selected speed->Idle
```

Carpeta: Processing. Clase: ThrottlePID derivada de PID:

En la línea 17 podemos incluir los modelos de aeronave cuyo control del acelerador deba ser igual al del resto de parámetros (alabeo, cabeceo, rumbo…). El resto funcionará del modo explicado en el apartado anterior

```
if(model>=20 || model==3)//Copters and jets
```

Se aconseja que se incluyan en esta línea las aeronaves de ala rotatoria y aquellas aeronaves de ala fija con motores a reacción.

Clase: MainProgram:

En las líneas 302 y 310 se debe configurar de forma que el número final coincida con el valor de tMax que se encuentra en la clase derivada de PidConfig para cada modelo de aeronave.

```
Throttle = (float)((float)Convert.ToInt32(pidctrl[0]) * SpanT / 2000);
```

Namespace XPlane

Carpeta: XPlane. Clase: XplaneConnection

En las primeras líneas se encuentran los **puertos utilizados para intercambiar información entre el programa y el simulador**.

En la línea 28 se encuentra la **dirección IP a la que envía el simulador** por defecto.