UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

GUIADO AUTOMÁTICO DE AERONAVES UTILIZANDO EL SIMULADOR DE VUELO COMO MODELO DE AERONAVE

TRABAJO FINAL DEL

Grado en Ingeniería Electrónica Industrial y Automática

REALIZADO POR

Joaquin Vico Navarro

TUTORIZADO POR

Juan Vila Carbó

CURSO ACADÉMICO: 2019/2020

Capítulo 1. Resumen

En este documento se expone proceso de diseño y desarrollo de un sistema de guiado y control vertical para el Boeing B737 y Cessna C172 sobre el simulador de vuelo Xplane10 y basado en la base de datos Eurocontrol-BADA.

En la bibliografía actual existe una gran cantidad de sistemas de control para distintas aeronaves, pero generalmente se ven limitados por la necesidad del uso de puntos preprogramados para realizar la transición de un tipo de control al siguiente y por tanto exigen un conocimiento exhaustivo del comportamiento de la aeronave y un precálculo de la ruta. En cambio, este proyecto busca proveer de una herramienta más genérica, capaz de generar un plan de guiado a partir de la información disponible en las cartas de navegación y planes de vuelo y ejecutar el control vertical en función a este y la información de rendimiento que provee la base de datos BADA.

Esta herramienta está a enfocada principalmente a investigación y docencia, siendo las prioridades la flexibilidad, modularidad y adaptabilidad para futuros proyectos.

Primeramente, se describirán los objetivos y requerimientos que se han considerado necesarios para el desarrollo del proyecto.

A continuación, se profundizará en la base teórica del proyecto, describiendo las principales funciones y características de los sistemas de gestión de vuelo, cartas y planes de vuelo, las aeronaves sobre las que se desarrollará el proyecto y los actuadores disponibles para su control.

Posteriormente se describirá el proceso de diseño del sistema, detallando la estructura de los datos de BADA y su modelo de la aeronave. También se decidirá el proceso de ajuste de PIDs y por último se detallará el proceso de diseño de los diferentes tramos de control y el de traducción de plan de vuelo a sistemas de control accionables.

Para finalizar la memoria del proyecto se incluirá la descripción de las distintas clases implementadas en Java y la validación del sistema de control con ambas aeronaves.

Índice

Índice de Ilustraciones

Índice de tablas

Capítulo 2. Objetivos

El objetivo de este proyecto el desarrollo de un sistema de guiado y control vertical para el Boeing B737 y Cessna C172 dentro del simulador Xplane 10 y basado en la base de datos Eurocontrol-BADA que permita la adaptación a distintos planes de vuelo.

El objetivo de este proyecto es desarrollar un sistema que sea capaz de ejecutar un vuelo autónomo a partir de la información proporcionada por el plan de vuelo y la base de datos de prestaciones de Eurocontrol-BADA usando como referencia las aeronaves Boeing B737 y Cessna C172.

Este sistema estará orientado a la docencia e investigación, por lo que deberá de ser altamente configurable, modular y portable.

Para alcanzar estos objetivos, el sistema deberá cumplir con los siguientes requerimientos:

- **Comunicación de datos con el simulador:** El sistema deberá ser capaz de obtener los datos de la situación actual del simulador y enviarle las señales de control.
- **Extracción de información de la base de datos BADA:** El sistema deberá ser capaz de analizar y extraer la información que se encuentra en los archivos de la base de datos BADA para hacerla accesible al resto del programa.
- **Desarrollo de un modelo de aeroplano basado en la base de datos BADA**: A partir de la información disponible a través de la base de datos se deberá desarrollar un modelo que sea capaz de estimar las fuerzas ejercidas sobre la aeronave.
- **Desarrollo de un sistema de predicción de trayectorias:** El sistema ha de ser capaz de generar una serie de trayectorias a partir de un plan de vuelo dado.
- **Desarrollo de los diferentes modelos de control:** Se habrán de desarrollar los diferentes modelos de control para los actuadores.
- **Generación de una Interfaz de usuario**: El sistema deberá incluir una interfaz de usuario que permita visualizar el estado actual de la aeronave y de los sistemas de control.
- Estudio de la posibilidad de generalización del sistema de control y validación sobre otras aeronaves.

Capítulo 3. Estado del Arte

En esta Capítulo se expondrá el estudio de la base teórica de este proyecto y el estado del arte actual, comenzando por una descripción de los sistemas de gestión de vuelo y sus funciones, la definición de Waypoint, carta de navegación y plan de vuelo y la función de EUROCONTROL en el mundo de la aviación y su base de datos BADA.

Por último, se incluye la información sobre las dos aeronaves sobre las que se desarrollará el proyecto

3.1 Sistemas de gestión de vuelo

Las aeronaves modernas vienen equipadas con sistemas que les permiten gestionar de forma autónoma distintos aspectos del vuelo.

Funciones de un Sistema de gestión de vuelo:

- Navegación: El sistema es responsable de establecer la posición de la aeronave durante el vuelo con la mejor precisión posible.
- Plan de vuelo: El sistema permite establecer una ruta prefijada para la aeronave.
- Predicción de trayectoria: El sistema ha de calcular los perfiles de vuelo necesarios para seguir la ruta determinada en el plan de vuelo.
- Rendimiento: El sistema proporciona a la tripulación datos sobre el rendimiento de la aeronave.
- Guiado: El sistema es responsable de generar las señales de control necesarias para guiar a la aeronave a lo largo de las trayectorias calculadas.

Para realizar estas funciones los sistemas de gestión de vuelo se apoyan en dos bases de datos, una base de datos de navegación (Navigation Database) y otra de prestaciones (Performance Database).

La base de datos de navegación le permite conocer la posición de las referencias de vuelo necesarias para conocer la posición durante el vuelo y calcular las trayectorias.

Por otro lado, la base de datos de prestaciones contiene toda la información relativa al comportamiento de la aeronave como velocidades de despegue, altitud máxima o perfiles de vuelo óptimos. Esta base de datos se usa tanto para dar información a la tripulación como para planear y controlar el guiado.

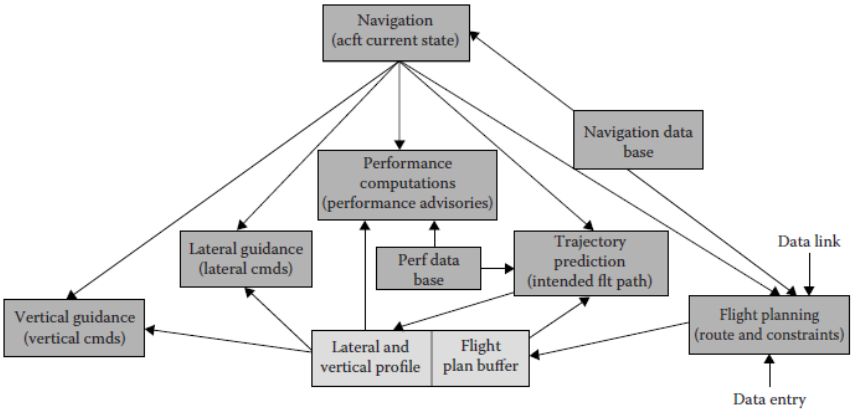


Ilustración 3.1: Bloque funcional del sistema de gestión de vuelo (Walter, 2014)

1. Waypoints

El plan de vuelo generalmente incluye una lista de puntos o Waypoints que establecerán una ruta entre el aeropuerto de salida y el de llegada.

La base de datos de navegación contiene las coordenadas e información de los waypoints, por lo que la tripulación solo tiene que introducir los puntos de la ruta en el sistema de gestión de vuelo para generar el plan.

Existen dos tipos básicos de waypoint, flyby y flyover.

Los waypoints de tipo flyby permiten comenzar el cambio de rumbo antes de haber alcanzado su posición mientras que un waypoint flyover requiere pasar por encima del punto antes de comenzar el cambio de rumbo.

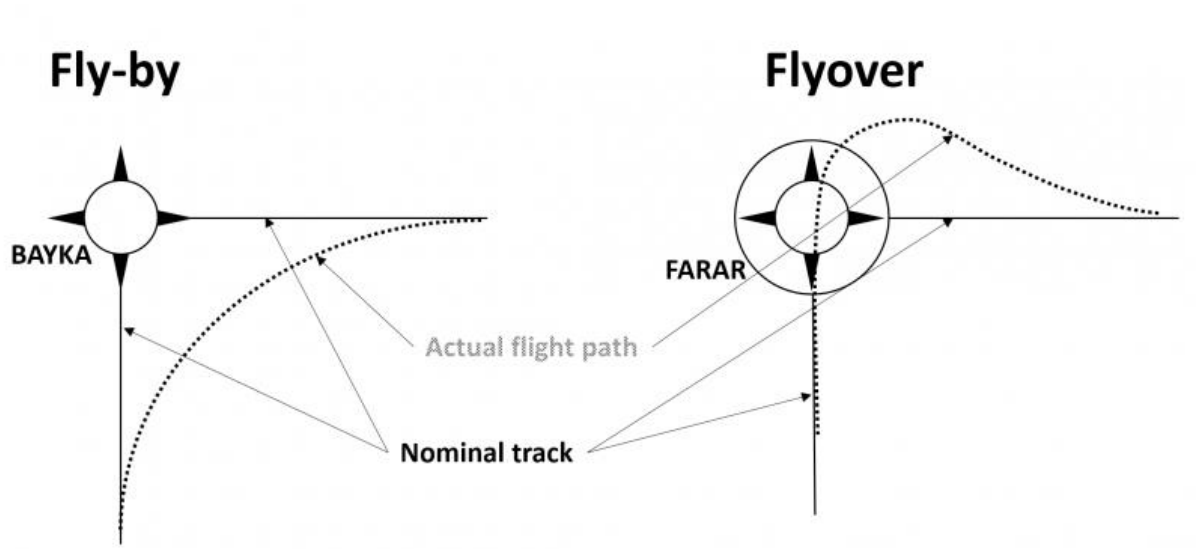


Ilustración 3.2 Cambio de rumbo flyby vs flyover

El cálculo del punto en el que se comienza a realizar el cambio de rumbo se explicará en el apartado 0.

Otra información como altitudes de vuelo máximas y mínimas, velocidades y rumbo pueden ser añadidas al waypoint y se usarán como factores limitantes durante la predicción de la trayectoria.

Además de puntos definidos por coordenadas, también existen otros tipos de waypoints que se pueden generar a partir de ellos, por ejemplo, estipulando un rumbo y distancia desde un waypoint conocido o la intersección de dos rumbos desde dos waypoints. Para reducir la complejidad del sistema se considerará que las coordenadas de todos los waypoints han sido precalculadas y se conocerá la latitud y longitud de estos.

3.2 Cartas de navegación y plan de vuelo

Las cartas de navegación contienen la información necesaria para planear y definir la ruta. Los tipos de carta de vuelo y la información contenida en ellas está definida en el Anexo 4 del Convenio de Aviación Civil Internacional (OACI,Organización de Aviación Civil Internacional, 2009). Estas cartas de navegación incluyen información que varía desde las maniobras en tierra hasta las cartas de navegación visual, pero para el alcance de este proyecto se destaca la función de las Cartas de Salida Normalizada con Vuelo por Instrumentos o SID y las Carta de Aproximación por Instrumentos o AIC.

La carta de Salida Normalizada provee de toda la información necesaria para planear una salida concreta desde una de las pistas de un aeropuerto. Esto incluye waypoints a visitar, las limitaciones de altitud y velocidad en esa zona, obstáculos y otras consideraciones específicas de ese aeropuerto a tener en cuenta.

También incluyen las posiciones y frecuencias de las distintas radioayudas que se encuentran en la zona.

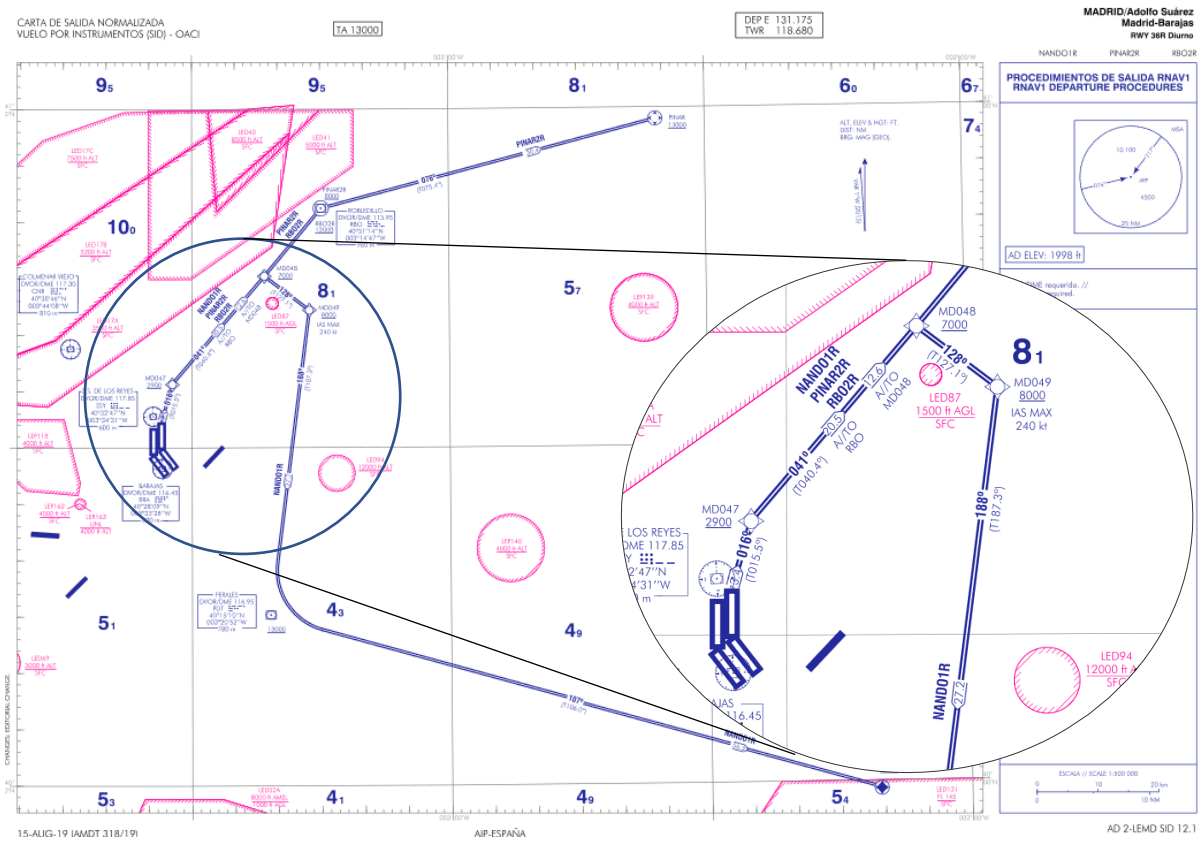


Ilustración 3.3 SID Madrid/Adolfo Suarez Pista 36R a NANDOR1 (AIRAC, 2019)

La carta de Aproximación por Instrumentos contiene la información de los procedimientos de aproximación y los procedimientos a seguir en caso de aproximación frustrada. Igual que la SID, contiene la información de los waypoints y radioayudas y además suele contener un diagrama del perfil vertical a seguir durante el proceso de aproximación y aterrizaje.

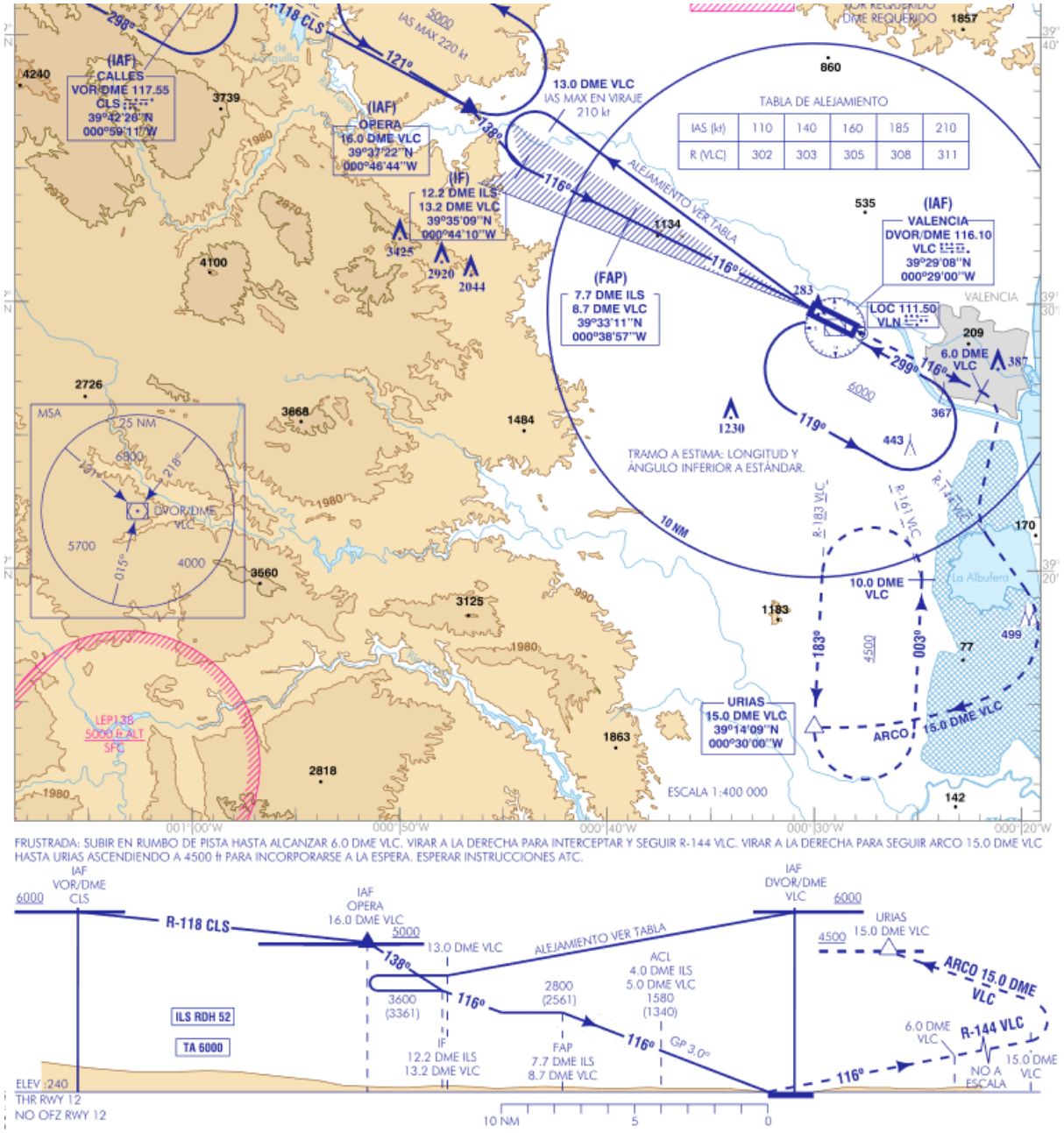


Ilustración 3.4 IAC Valencia Pista 12 desde Calles (AIRAC, 2018)

1. *Plan de vuelo*

El plan de vuelo es un documento estándar definido por la Organización de Aviación Civil internacional o OACI. Este documento contiene información sobre el vuelo, aeronave y piloto y se ha de remitir para su aprobación al Servicio de Tráfico aéreo competente.

Entre otra información, el documento incluye los aeropuertos de salida y llegada, la ruta a seguir y la altura y velocidad de crucero.

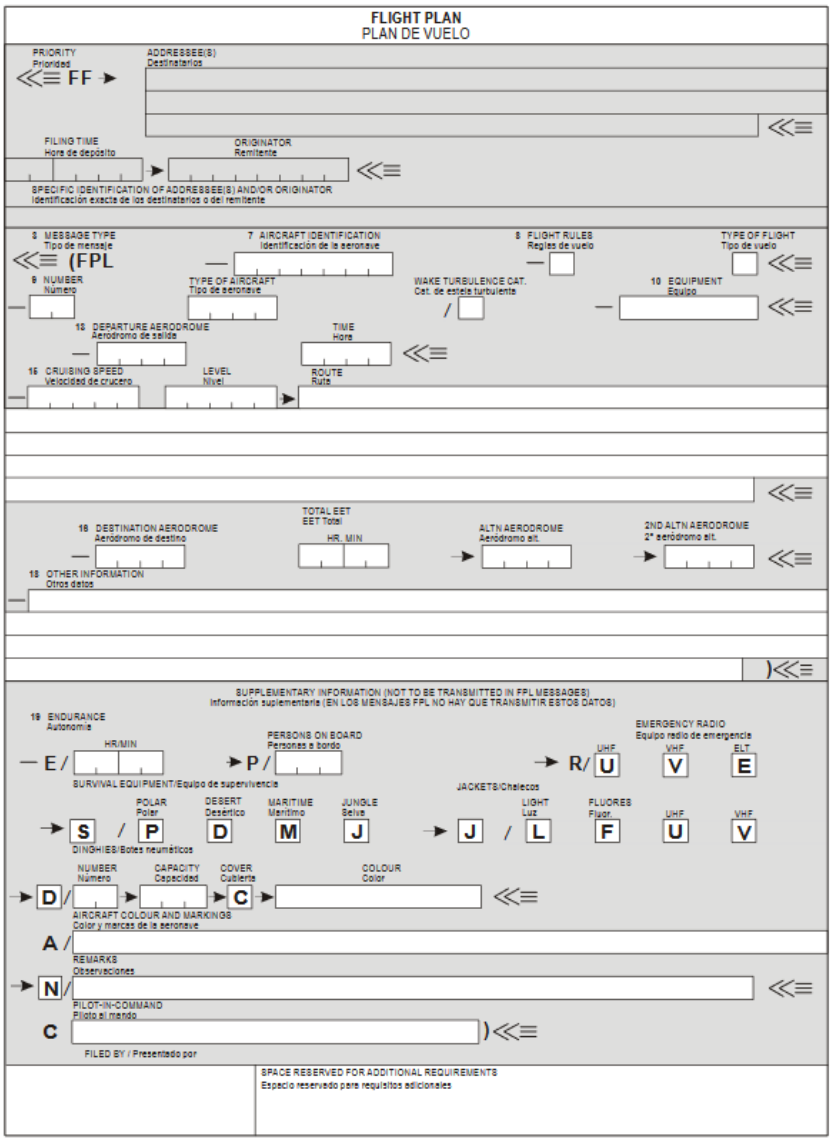


Ilustración 3.5 Plan de vuelo (OACI, 2007)

2. *Predicción de trayectoria*

Una vez se ha establecido un plan de vuelo, el sistema de gestión de vuelo ha de ser capaz de dividir el plan en una serie de tramos. Estos tramos definen el tipo de control que se ejercerá durante el guiado.

Para cada segmento se establece un tipo de control distinto, con un objetivo distinto. En la Ilustración 3.6 se puede apreciar un ejemplo de una separación por segmentos de un perfil vertical, resaltando en negro los segmentos de mantenimiento de altitud (Altitude Hold) y segmentos de ascenso y descenso en naranja y azul respectivamente.

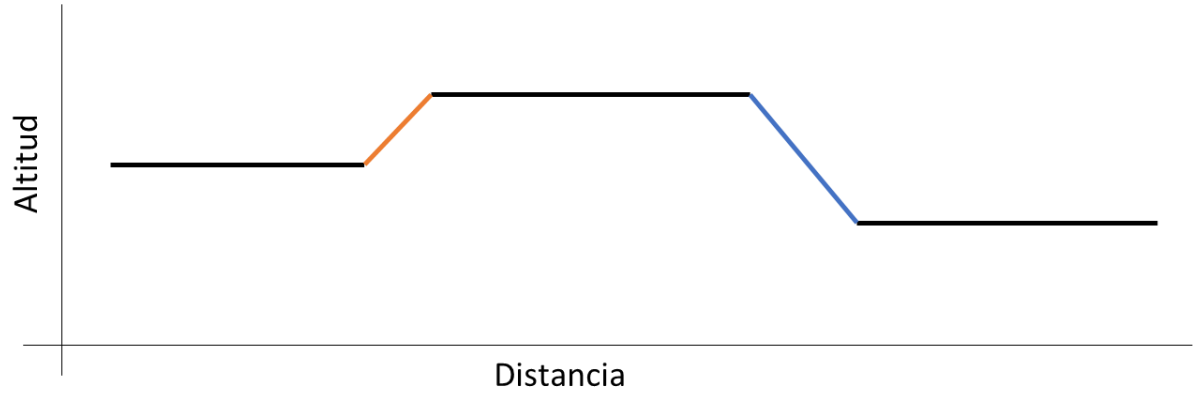


Ilustración 3.6 Ejemplo de separación por segmentos del perfil vertical

3.3 Eurocontrol

Eurocontrol u "Organización Europea para la Seguridad de la Navegación Aérea" una organización con base en Bruselas dedicada al avance y la estandarización de los sistemas de control aéreo en sus países miembros con el fin de mejorar la colaboración e interoperabilidad entre ellos.

Entre otros servicios, Eurocontrol provee a sus miembros de herramientas de simulación y modelado del tráfico aéreo. Una de estas herramientas es la Base de Datos de Aeronaves o BADA por sus siglas en inglés.

EUROCONTROL

Ilustración 3.7 Logotipo de Eurocontrol

1. Eurocontrol BADA

La Base de Datos de Aeronaves o BADA es una base de datos que contiene la información necesaria para la simulación del comportamiento de una aeronave durante todas sus fases de vuelo. Esta información incluye las características físicas de la aeronave, estimaciones de consumo de combustible o las velocidades y alturas típicas a las que vuelan los diferentes operadores.

Por una parte, BADA es una herramienta útil para calcular la trayectoria real y el rendimiento de una aeronave conociendo su trayectoria ideal.

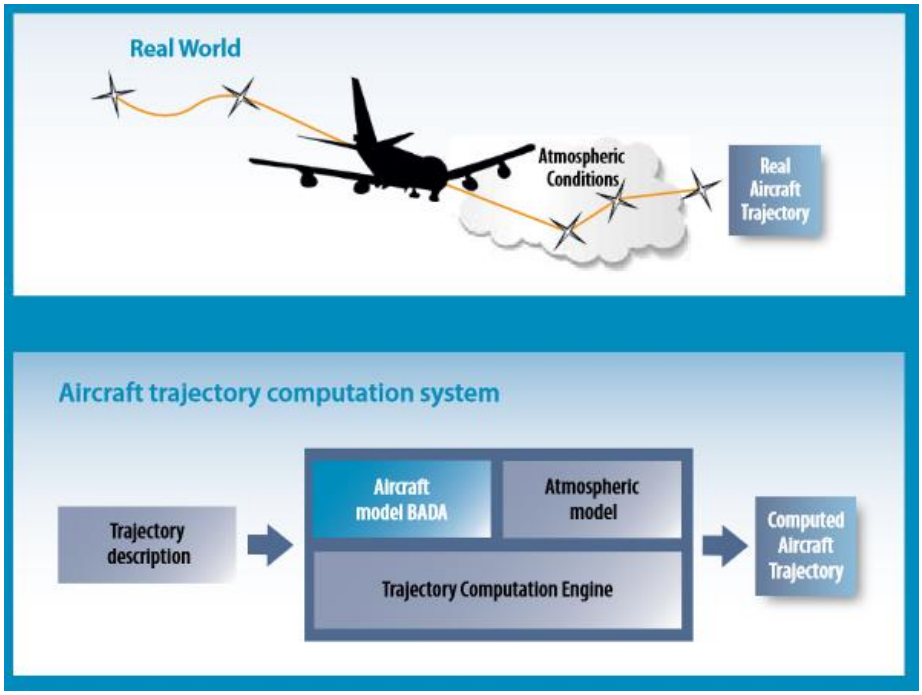


Ilustración 3.8 Diagrama de Uso de BADA (Eurocontrol, 2020)

Para este fin, BADA provee un modelo de la aeronave calculado en base al Modelo de Energía Total explicado en la sección 1.

Esta información puede ser utilizada para tanto ejercicios de simulación y validación de sistemas como para cálculo de trayectorias y optimización de rutas o consumos.

Por otra parte, BADA también provee de valores por defecto para algunas operadoras de cada aeronave. Estos valores son muy prácticos para situaciones en las que se quiere generar un tráfico aéreo simulado relativamente realista.

En el momento de redacción de este trabajo existen dos familias activas de esta base de datos, la familia 3 y la familia 4.

La familia 3 es la versión más madura de la base de datos y provee modelos para la mayoría de las aeronaves utilizadas en los pises miembros, asegurando una cobertura de al menos el 95% del tráfico aéreo.

La actualización a BADA 4 es una versión más reciente y sus avances se centran en vuelo eficiente, aumentando las capacidades del modelo para optimización de costes y uso de combustible, pero todavía no se encuentra disponible para muchas aeronaves.

Dada la amplia gama de aeronaves disponibles se ha decidido basar este proyecto en la familia 3 de BADA

3.4 Aviones de verificación

Los aviones que se usarán durante el desarrollo del proyecto serán el Boeing B737-800 y el Cessna C172. Se han elegido estos aviones porque son unos ejemplos muy representativos de sus categorías, avión comercial de pasajeros y utilitario monomotor.

1. B737-800

Desarrollado por la empresa americana Boeing, el B737 en un avión de pasajeros de fuselaje estrecho, equipado con dos motores a reacción y de corto a medio alcance. Uno de los aviones comerciales más vendidos, el B737 se usa ampliamente en el espacio aéreo europeo.

Ilustración 3.9 Boeing B373

Número de asientos	189
Tipo de motor	Reacción
Peso en Vacío	36,378–44,677 kg
Velocidad Operativa	450 knt
Velocidad nunca excedida	473 knt
Alcance	5,436–5,575 km
Techo de vuelo	41 000 pies

Tabla 3 Características del B737-800

2. Cessna C-172

El Cessna 172 es un avión monomotor desarrollado por la empresa americana Cessna. Fue introducido en el año 1956 y es uno de los aviones más vendidos con al menos 44.000 unidades en 2015.

El éxito de este avión se debe a la combinación de bajo coste y facilidad de manejo, que lo hicieron el vehículo ideal para el entrenamiento de nuevos pilotos. También es ampliamente utilizado en tareas de búsqueda y rescate, monitorización y vigilancia gracias a su baja velocidad de entrada en pérdidas y estabilidad.

Ilustración 3.10 Cessna C-172

Las características principales del Cessna 172 son las siguientes:

Número de asientos	4
Tipo de Motor	Pistón
Peso en Vacío	367 kg
Velocidad Operativa	123 knt
Velocidad nunca excedida	163 knt
Alcance	1270 km
Techo de vuelo	14 200 pies
Régimen de ascenso	720 pies por minuto

Tabla 3.1 características del Cessna 172

3.5 Actuadores de un avión

En el control de un avión en vuelo se utilizan una serie de actuadores, estos son los motores y las superficies de control.

1. Motores

En un avión, los motores generan el empuje o thrust que hace que acelere hacia adelante. Esta fuerza es una de las cuatro fuerzas principales de vuelo.

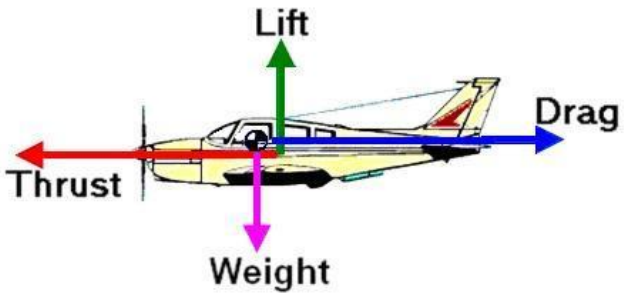


Ilustración 3.11 Fuerzas de vuelo (USA Civil Air Patrol, 2018)

El control sobre el motor es la forma principal que tiene un avión de controlar su energía total en vuelo.

Existen tres tipos principales de motor para aviones comerciales, motores de pistón, turbohélice y reacción.

Motores de pistón

Los primeros motores usados en el mundo de la aviación fueron los motores de pistón, estos motores generan el empuje girando una hélice a altas velocidades. Esta rotación se consigue mediante un sistema de pistón cigüeñal con un motor de combustión interna.

Existen diferentes tipos de configuración con sus propias características.

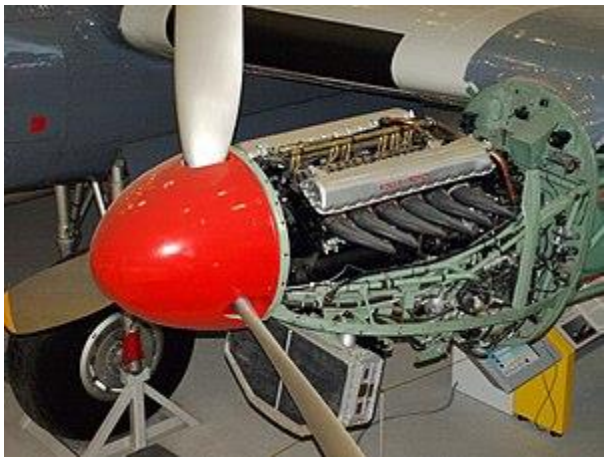


Ilustración 3.12 Motor de pistón

Motores Turbohélice

De forma similar a los motores de pistón, los motores turbohélice generan la mayoría de su empuje girando una hélice a altas velocidades, la diferencia se encuentra en que para generar esta rotación los motores turbohélice usan la expansión del aire generada al quemar el combustible en la cámara de combustión para rotar una turbina, que esta a su vez conectada al hélice.

Los gases de escape también generaran un ligero empuje al salir del motor, aproximadamente un 10% del empuje total.

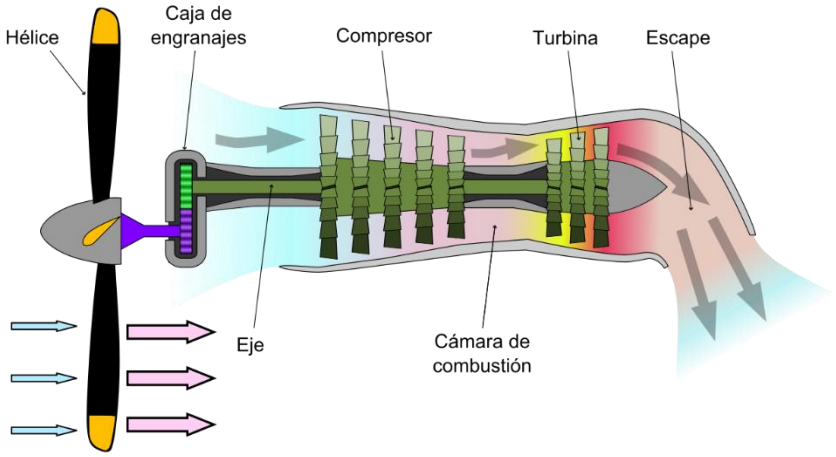


Ilustración 3.13 Funcionamiento de un motor turbohélice (Emoscopes, 2005)

Motores a Reacción

Los motores a reacción a diferencia de los dos anteriores no cuentan con una hélice externa, estos motores generan el empuje expulsando fluidos a alta velocidad. Existen muchos tipos de motores a reacción, pero dentro del mundo de la aviación el más típico es el turbofán.

Los motores turbofán se caracterizan por tener un ventilador en la parte frontal. Este ventilador fuerza aire al interior del motor que se dividirá en dos flujos, el flujo primario y el secundario. El flujo secundario se derivará por la parte exterior del motor, mientras que el flujo primario pasara por una serie de compresores, una sala de combustión donde se quemara junto al combustible para salir a través de unas turbinas.

La relación entre el flujo primario y el secundario se conoce como índice de derivación y definirá gran parte de las prestaciones del motor.

Ilustración 3.14 Motor a reacción en un A320 (Delatorre, 2007)

2. Superficies de control

Las superficies de control permiten controlar la magnitud y la dirección de las diferentes fuerzas de resistencia del aire. Aumentando de forma controlada esta resistencia en diferentes puntos se consigue un cambio de dirección.

Alerones

Los alerones se encuentran situados generalmente en la zona más exterior de las alas y su principal uso es el control del alabeo del aeroplano.

Ilustración 3.15 YAK-52 usando sus alerones (Markin, 2012)

Los alerones suelen estar conectados, de forma que cuando uno baja, el otro sube, esto aumenta la fuerza de sustentación en el ala con el alerón bajo mientras que se recude en la otra, creando un momento de alabeo y haciendo que el aeroplano gire en su eje longitudinal.

Estos giros pueden causar la llamada guiñada adversa, cuando se usan los alerones para girar a la derecha las fuerzas de resistencia harán que el aeroplano comience a guiñar hacia la izquierda.

Timón de dirección

El timón de dirección se encuentra en la cola del avión y se utiliza para el control de guiñada. Aunque rara vez se use para dirigir el aeroplano de forma independiente, el timón se usará para corregir la guiñada adversa generada por un giro de alerón. Este tipo de giros en los que se usan ambas superficies de control son conocidos como “giros coordinados”.

También puede ser usado en aviones con múltiples motores para compensar un empuje asimétrico o para mantener el avión alineado con la pista durante el aterrizaje si hay demasiado viento lateral.

Generalmente se encuentra conectado al giro del tren de aterrizaje, por lo que se usara para dirigir el aeroplano mientras siga en tierra.

Elevadores

Los elevadores se encuentran en la cola y ejercen control sobre el cabeceo del aeroplano. Lo hacen creando una fuerza perpendicular al eje del avión, lo que crea un momento de giro sobre su centro de masa y por tanto afectando al cabeceo.

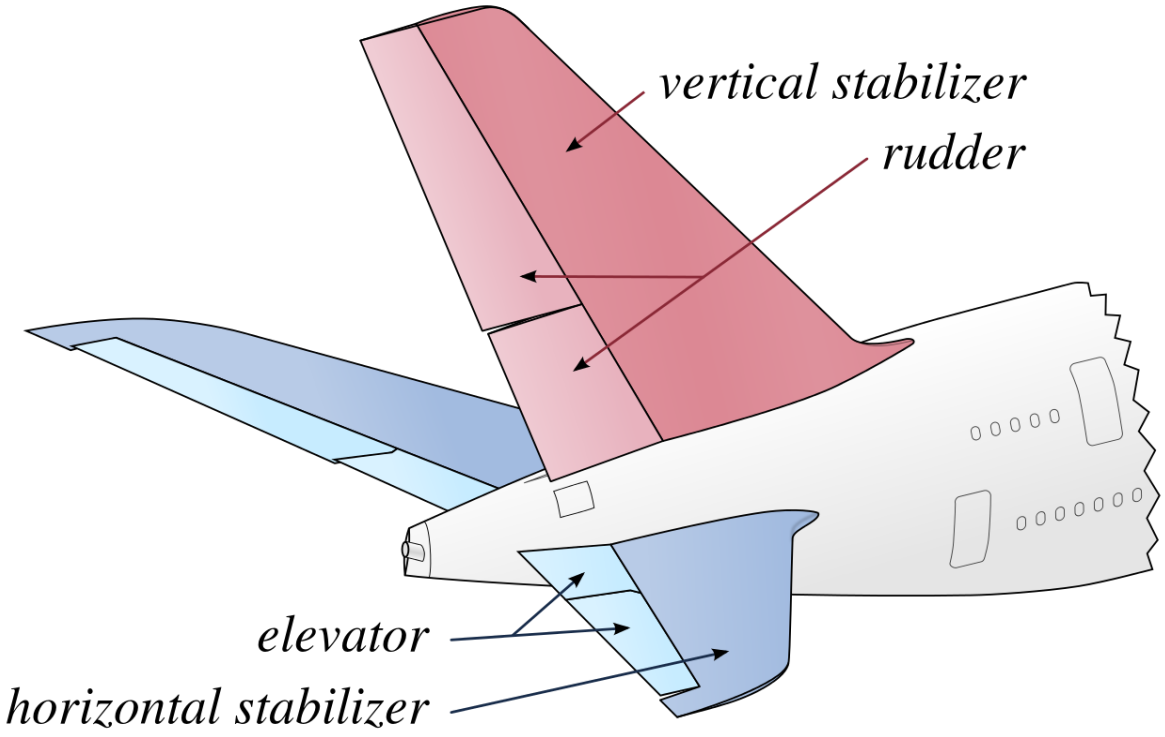


Ilustración 3.16 Cola de un Airbus A380 (Cleynen, 2013)

Los elevadores se accionan de forma simétrica a diferencia de los alerones.

Flaps

Los Flaps o dispositivos de hipersustentación son una superficie de control cuyo principal objetivo es aumentar el área del ala y por tanto la fuerza de sustentación en situaciones de baja velocidad como en aterrizaje y despegue.

Los flaps se encuentran generalmente en la parte interior de las alas y se accionan de forma simétrica.

Ilustración 3.17 Flaps de un B737 desplegados

A diferencia de las superficies de control comentadas anteriormente, los flaps tienen una velocidad máxima a la que pueden accionarse que varía con cada aeronave y en caso de accionarse a velocidades mayores podrían causar graves daños al ala.

3.6 Simulador XPlane 10

Para poder simular el comportamiento de la aeronave durante el vuelo se usará el software XPlane 10. Uno de los puntos fuertes de este simulador es que permite el uso de conexiones externas para la exportación de datos y control de las aeronaves durante el vuelo.

XPlane 10 está catalogado como un simulador apto para el entrenamiento de pilotos por la FAA.

El modelo de vuelo de XPlane se diferencia de otros simuladores en que calcula las fuerzas siguiendo la teoría del elemento pala, mientras que otros se basan en la consulta y aproximación de puntos precalculados en LookUp Tables (LUTs). Con este modelo es capaz de simular situaciones de las que no se tienen datos y de simular de forma fiable el rendimiento de aeronaves con modificaciones.

Otra de las propiedades importantes del simulador XPlane 10 es que permite tanto un control manual mediante joysticks o controladores específicos para el simulador como un control externo a través de una conexión standard. Esto permite que un programa externo intercambie información con el simulador, extrayendo datos y controlando la aeronave como si un piloto se tratara.

Capítulo 4. Diseño

En este capítulo se detallarán las decisiones de diseño que se han seguido durante la elaboración del proyecto. Se comenzará con una descripción de la estructura y contenido de la base de datos de BADA, continuando con el modelo de aeronave que se ha creado en base a esta.

Posteriormente se describirá el concepto de plan de misión y tramo de control, especificando las características de cada tipo de tramo, y se detallara el proceso de traducción de plan de vuelo a un conjunto de tramos de control accionables.

Por último, se describirá el proceso de ajuste de los diferentes controladores PID.

4.1 Base de datos BADA

Para poder modelar correctamente el comportamiento de un aeroplano en vuelo se necesitan conocer sus características y comportamiento en vuelo. Para el desarrollo de este proyecto se ha basado el modelo en la base de datos BADA.

La información de BADA se encuentra separada en una serie de archivos, cada uno con información de una faceta diferente del modelo. Los archivos son los siguientes:

- Archivo de Sinónimos (SYNONIM)
- Archivos de Prestaciones Operativas (Operations Performance Files o OPF)
- Archivos de Procedimiento de Aerolínea (Airline Procedure Files o APF)
- Archivos de Tabla de Prestaciones (Performance Table Files o PTF)
- Archivos de datos de Prestaciones (Performance Table Data o PTD)

4.2 Archivo de Sinónimos

Este archivo contiene la lista de los tipos de aeronave soportados por BADA y si esta soportado directamente o por el modelo de otra aeronave considerado equivalente. Ya que se conoce que nombre tendrá el modelo de BADA para los aviones elegidos el archivo de sinónimos no será utilizado en este proyecto.

4.3 Archivo de Prestaciones Operacional (OPF)

Este archivo contiene los parámetros necesarios para el modelo de masa, envolvente de vuelo, resistencia del aire, empuje y consumo de combustible.

Estos parámetros están divididos por bloques:

Bloque de tipo de aeronave

Este bloque contiene el nombre del modelo, el número de motores que tiene y su tipo.

```
CC====== Actype ===================================================/
CD   A306__          2 engines    Jet                      H        /
CC  A300B4-622    with PW4158 engines                    wake       /
CC                                                                  /
```

Ilustración 4.1 Bloque de tipo de aeronave (EUROCONTROL, 2017)

Bloque de masa

Este bloque contiene la información sobre la masa en toneladas de la aeronave.

```
CC====== Mass (t) ================================================/
CC    reference      minimum      maximum     max payload  mass grad  /
CD     .14000E+03   .87000E+02   .17170E+03   .39000E+02   .15103E+00 /
```

Ilustración 4.2 Bloque de masa (EUROCONTROL, 2017)

Bloque de envolvente de vuelo

Este bloque contiene la información sobre los limites en vuelo de la aeronave, su velocidad máxima en nudos y mach, la altura máxima en pies y el gradiente de temperatura para estas.

```
CC====== Flight envelope ========================================/
CC     VMO(KCAS)       MMO        Max.Alt       Hmax       temp grad  /
CD     .33500E+03   .82000E+00   .41000E+05   .32378E+05   -.2716E+02 /
```

Ilustración 4.3 Bloque de envolvente de vuelo (EUROCONTROL, 2017)

Bloque de aerodinámica

Este bloque contiene las características aerodinámicas del ala y una serie de coeficientes para distintas fases de vuelo. Estos coeficientes se usarán en el modelo para calcular la fuerza de resistencia del aire y para obtener la velocidad mínima de entrada en pérdidas.

En el apartado nombre (name) se incluye la configuración de los Spoilers y Flaps para las que se han calculado los coeficientes.

```
CC====== Aerodynamics ================================================/
CC Wing Area and Buffet coefficients (SIM)                            /
CCndrst Surf(m2)     Clbo(M=0)       k          CM16                  /
CD 5   .26000E+03   .13150E+01   .84080E+00   .00000E+00              /
CC   Configuration characteristics                                    /
CC n Phase  Name    Vstall(KCAS)    CD0          CD2         unused    /
CD 1 CR   Clean     .15100E+03   .20591E-01   .51977E-01   .00000E+00 /
CD 2 IC   S15F00    .11700E+03   .33057E-01   .45362E-01   .00000E+00 /
CD 3 TO   S15F00    .11700E+03   .33057E-01   .45362E-01   .00000E+00 /
CD 4 AP   S15F15    .10900E+03   .38031E-01   .44932E-01   .00000E+00 /
CD 5 LD   S30F40    .97000E+02   .78935E-01   .44822E-01   .00000E+00 /
CC   Spoiler                                                          /
CD 1      RET                                                         /
CD 2      EXT                                .00000E+00   .00000E+00 /
CC   Gear                                                             /
CD 1      UP                                                          /
CD 2      DOWN                    .2250E-01   .00000E+00   .00000E+00 /
CC   Brakes                                                           /
CD 1      OFF                                                         /
CD 2      ON                                 .00000E+00   .00000E+00 /
```

Ilustración 4.4 Bloque de aerodinámica (EUROCONTROL, 2017)

Bloque de empuje de motor

Este bloque contiene los coeficientes necesarios para el cálculo del empuje máximo del motor. Este empuje máximo no es constante y depende de la densidad del aire. Esto a su vez dependerá de la altura y temperatura actuales.

Cada tipo de motor tiene su propio modelo que usa los mismos tipos de coeficientes.

```
CC====== Engine Thrust ================================================/
CC         Max climb thrust coefficients (SIM)                         /
CD     .29716E+06   .51306E+05   .56296E-10   .84814E+01   .44597E-02 /
CC     Desc(low)    Desc(high)   Desc level   Desc(app)     Desc(ld)  /
CD     .32012E-01   .40310E-01   .15161E+05   .13124E+00   .39136E+00 /
CC     Desc CAS     Desc Mach     unused       unused       unused    /
CD     .30000E+03   .78000E+00   .00000E+00   .00000E+00   .00000E+00 /
```

Ilustración 4.5 Bloque de empuje de motor (EUROCONTROL, 2017)

Bloque de consumo de combustible

Los coeficientes de consumo de combustible permiten calcular el consumo durante el vuelo.

Aunque esta información se añadirá leerá y almacenará en el sistema de guiado, la eficiencia y consumo se han dejado fuera del proyecto, por lo que no se usará.

```
CC====== Fuel Consumption ============================================/
CC   Thrust Specific Fuel Consumption Coefficients                    /
CD     .63936E+00   .10047E+04                                        /
CC   Descent Fuel Flow Coefficients                                   /
CD     .21196E+02   .67071E+05                                        /
CC   Cruise Corr.    unused       unused       unused       unused    /
CD     .98852E+00   .00000E+00   .00000E+00   .00000E+00   .00000E+00 /
```

Ilustración 4.6 Bloque de empuje de motor (EUROCONTROL, 2017)

Bloque de movimiento en tierra

El bloque de movimiento en tierra contiene la longitud mínima de la pista de despegue y aterrizaje en el peor caso(TOL/LDL), la longitud del avión y la envergadura de ala. Igual que con el bloque de consumo de combustible, esta información se almacenará, pero no se usará.

```
CC====== Ground ============================================================/
CC         TOL           LDL          span          length        unused     /
CD      .23620E+04    .15550E+04    .44840E+02    .54080E+02    .00000E+00 /
CC=========================================================================/
FI                                                                         /
```

Ilustración 4.7 Bloque de empuje de motor (EUROCONTROL, 2017)

4.4 Archivo de Procedimiento de Aerolínea (APF)

En el APF se proporcionan unos valores nominales de velocidad para las distintas fases de vuelo. Esto permite dar unos valores por defecto a las aeronaves en simuladores y softwares de gestión.

4.5 Archivo de Tabla de Prestaciones (PTF)

Además de los coeficientes necesarios para simular el comportamiento del aeroplano, BADA también proporciona dentro de su tabla de prestaciones una serie de características de vuelo óptimo para diferentes altitudes.

Estos datos se utilizarán como objetivos de control en los diferentes segmentos.

```
BADA PERFORMANCE FILE                                        Apr 01 2010

AC/Type: A306__
                            Source OPF File:                 Sep 05 2008
                            Source APF file:                 Mar 05 2009

 Speeds:   CAS(LO/HI)  Mach   Mass Levels [kg]          Temperature:  ISA
 climb   - 250/310     0.79   low     -  104400
 cruise  - 250/310     0.79   nominal -  140000        Max Alt. [ft]:  41000
 descent - 250/290     0.79   high    -  171700
=============================================================================================
 FL |          CRUISE          |                CLIMB                |       DESCENT
    |  TAS          fuel        |  TAS          ROCD         fuel    |  TAS  ROCD    fuel
    | [kts]       [kg/min]      | [kts]         [fpm]       [kg/min] | [kts] [fpm] [kg/min]
    |          lo   nom    hi   |          lo    nom    hi    nom     |        nom    nom
=============================================================================================
  0 |                           |  157    2454  1925  1556   219.7   |  131    698   84.1
    |                           |                                     |
  5 |                           |  158    2437  1907  1536   217.8   |  132    714   83.3
    |                           |                                     |
 10 |                           |  159    2420  1889  1517   215.9   |  138    730   82.9
    |                           |                                     |
 15 |                           |  166    2530  1974  1588   214.9   |  149    774   82.9
    |                           |                                     |
 20 |                           |  167    2512  1955  1568   213.0   |  181    988   28.3
    |                           |                                     |
 30 |  230    53.3  69.9  88.8  |  190    2940  2289  1852   212.9   |  230   1287   20.2
    |                           |                                     |
 40 |  233    53.4  70.1  89.0  |  225    3474  2695  2191   214.6   |  233   1306   19.9
    |                           |                                     |
```

Ilustración 4.8 Ejemplo de tabla de prestaciones para el A306 (EUROCONTROL, 2017)

Esta tabla provee para cada nivel de vuelo y fase una serie de datos. De estos caben destacar las Velocidades Verdaderas (TAS) y las velocidades de ascenso y descenso (ROCD o Rate Of Climb and Descent).

4.6 Archivo de Datos de Prestaciones (PTD)

Este archivo contiene datos de prestaciones precalculados para diferentes condiciones y es un superset de los datos proporcionados por el PTF. La principal utilidad es proporcionar un conjunto de datos contra el que validar otros sistemas. Este archivo no se usará en este proyecto.

4.7 Modelos de la aeronave

De acuerdo con el modelo de energía total, para el control de la aeronave se necesitará el cálculo del empuje máximo y de la resistencia del aire actual.

Además, se requerirán bucles de control para cada una de las superficies de control.

1. Modelo de Energía Total

El modelo de Energía Total o TEM por sus siglas en ingles es el modelo en que se basa BADA. Este modelo se puede considerar un modelo reducido de punto con masa.

El modelo iguala el ratio de trabajo que ejercen las fuerzas sobre la aeronave a la variación de energía cinética y potencial.

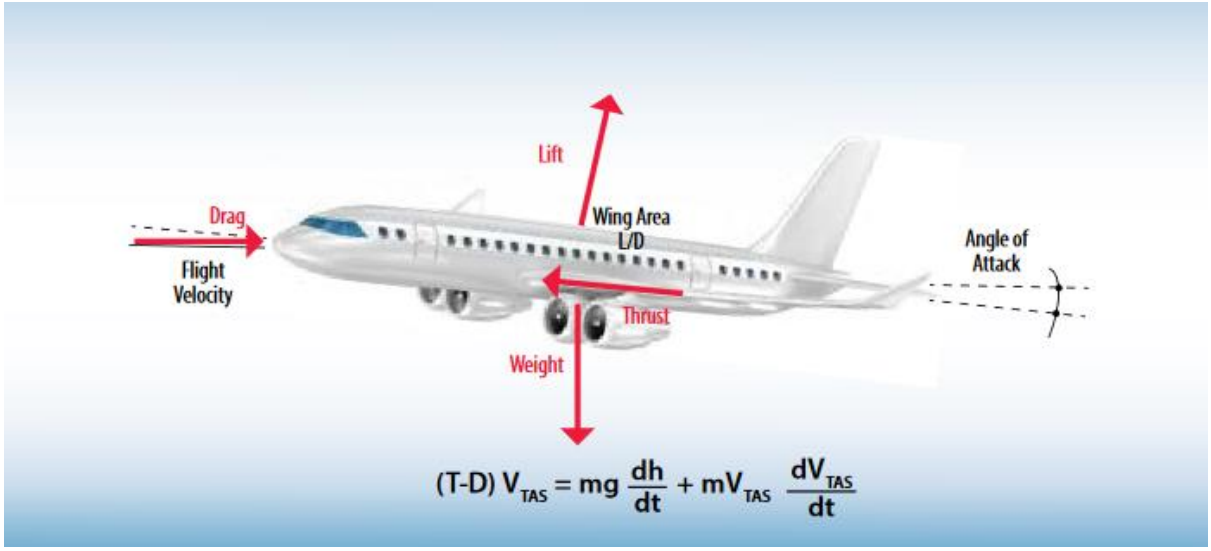


Ilustración 4.9 Modelo de Aeronave (Eurocontrol, 2020)

Ya que la resistencia del aire o drag (D) se puede calcular con las características del avión, este modelo permite estimar el empuje necesario para realizar un cambio de altura o velocidad.

Con esta ecuación se puede calcular la energía total que tendrá la aeronave en cada momento, pero no permite controlar el ratio entre energía cinética y potencial. Para ello se deberán de usar las superficies de control.

2. *Cálculo del empuje máximo*

El modelo de empuje se obtiene de los coeficientes obtenidos en el archivo OPF de BADA.

Los modelos de BADA permiten calcular el empuje o Thrust máximo para la fase de ascenso y extrapolan el máximo en el resto de las fases usando coeficientes de corrección.

El empuje máximo de ascenso se calculará con la siguientes formulas dependiendo del tipo de motor:

Jet: $$(Thr_{max\,climb})_{ISA} = C_{Tc,1} \times \left(1 - \frac{H_p}{C_{Tc,2}} + C_{Tc,3} \times {H_p}^2\right)$$

Turboprop: $$(Thr_{max\,climb})_{ISA} = \frac{C_{Tc,1}}{V_{TAS}} \times \left(1 - \frac{H_p}{C_{Tc,2}}\right) + C_{Tc,3}$$

Piston: $$(Thr_{max\,climb})_{ISA} = C_{Tc,1} \times \left(1 - \frac{H_p}{C_{Tc,2}}\right) + \frac{C_{Tc,3}}{V_{TAS}}$$

Siendo:

- Vtas: la velocidad verdadera en nudos
- H_p: la altura geopotencial en pies
- C_{tc1}, C_{tc2} y C_{tc3}: Coeficientes obtenidos del OPF

A partir del empuje máximo, se calculará la acción sobre el acelerador (throttle) con la siguiente relación:

$$Throttle = \frac{Thrust_{target}}{Thr_{max}}$$

3. *Cálculo de la Resistencia del aire*

De forma similar al empuje, el modelo de BADA permite calcular la fuerza de resistencia del aire ejercida sobre la aeronave.

Para ello primero se ha de obtener el coeficiente de sustentación:

$$C_L = \frac{2 \cdot m \cdot g_0}{\rho \cdot V_{TAS}^{\ 2} \cdot S \cdot \cos\phi}$$

Siendo:

- M: la masa de la aeronave en kilos
- g_0: la fuerza de la gravedad
- ρ: la densidad del aire en kg/m^3
- Vtas: Velocidad verdadera en m/s
- S: la superficie del área del ala de referencia en m^2

Una vez obtenido el coeficiente de sustentación se obtendrá el coeficiente de resistencia dependiendo de la fase en la que se encuentre:

- Crucero

$$C_D = C_{D0,CR} + C_{D2,CR} \times (C_L)^2$$

- Aproximación

$$C_D = C_{D0,AP} + C_{D2,AP} \times (C_L)^2$$

- Aterrizaje

$$C_D = C_{D0,LDG} + C_{D0,\Delta LDG} + C_{D2,LDG} \times (C_L)^2$$

Siendo C_{d0} y C_{d1} coeficientes proporcionados por BADA.

Con el C_D calculado se obtendrá la fuerza de resistencia Newtons utilizando la siguiente formula:

$$D = \frac{C_D \cdot \rho \cdot V_{TAS}^{\ 2} \cdot S}{2}$$

Una vez calculada la resistencia del aire, se podrá utilizar para calcular el empuje necesario de los motores.

4.8 Plan de misión

Una de las partes más importantes del desarrollo de este proyecto es la traducción de la información contenida en el plan de vuelo a una serie de tramos de control accionables por el sistema de autopiloto. A esta serie de tramos independientes se le ha llamado **Plan de Misión**.

Para generar el plan de misión, primero se dividirá el control en dos dominios independientes, el control vertical y el control horizontal. Estos dominios estarán a su vez divididos en una serie de tramos, cada uno con un tipo de control, unos objetivos de control y una terminación concretos.

Por tanto, el Plan de Misión vendrá definido por dos listas como se muestra en la siguiente figura:

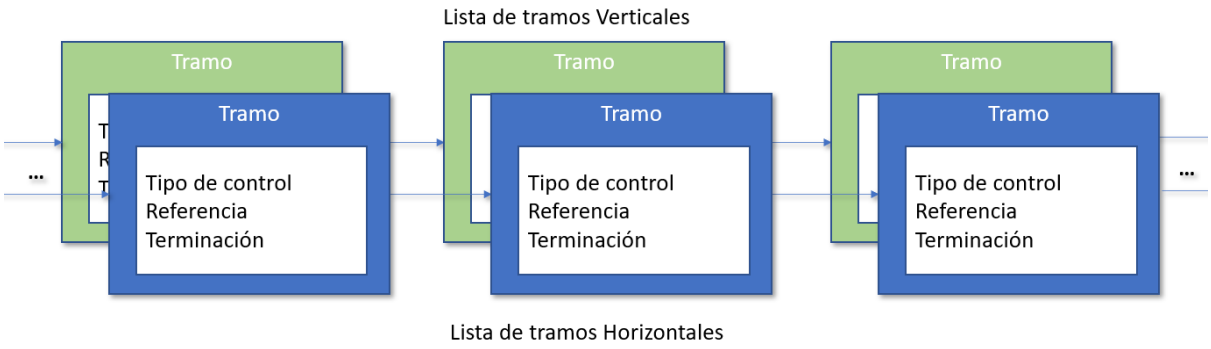


Ilustración 4.10 Listas de tramos del plan de misión

Esta estructura permite compartimentalizar las distintas fases de vuelo y generar una serie de tramos que contengan las complejidades y particularidades de cada fase, manteniendo el sistema de autopiloto simple. Además, al separar el control vertical del horizontal, se permite que se haga una transición de un tipo de tramo a otro de forma independiente en cada uno de los planos, por ejemplo, pasar de ascender a mantener altura en el plano vertical mientras el plano horizontal mantiene un rumbo o cambiar de rumbo sin cambiar el control vertical.

1. Tramo

Los tramos han de contener la información necesaria para poder generar las acciones de control necesarias hasta que alcancen su condición de terminación.

Los tramos se especializarán en tramos horizontales y verticales, pero ambos comparten las siguientes características:

- Tienen una referencia de control.
- Tienen una condición de terminación que indica cuando se ha de pasar al siguiente tramo.
- Contienen un algoritmo de control adecuado para su tipo de referencia.

A continuación, se expondrá en más detalle las particularidades de cada tipo de tramo.

2. Tramos Horizontales

Los tramos horizontales son los encargados de realizar el control en el plano horizontal de la aeronave. Estos tramos ejercerán su control sobre el timón de cola, los alerones y los frenos.

Dentro de los tramos horizontales podemos encontrar los siguientes tipos de tramo:

Tramo de despegue

El tramo de despegue mantiene el freno accionado hasta que el motor alcance las revoluciones mínimas y ejerce control sobre el timón para mantener el aeroplano alineado con la pista de despegue.

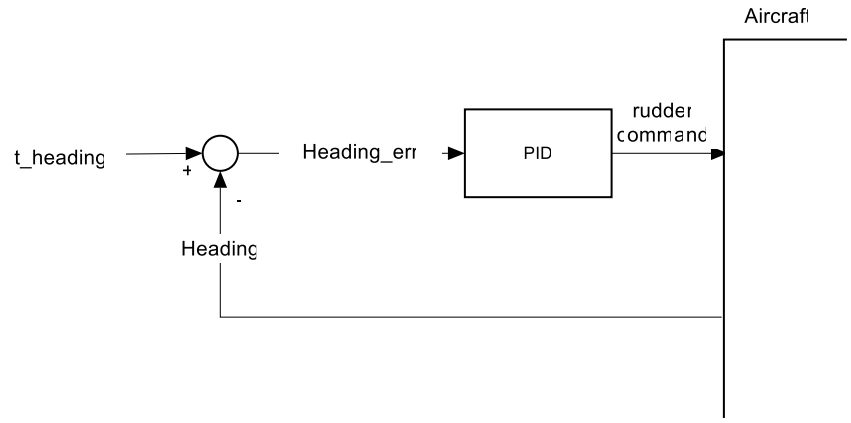


Ilustración 4.11 Bucle de control de rumbo en tierra

Tramo de mantenimiento de rumbo

Los tramos por rumbo se usan para mantener un rumbo estable, principalmente durante las fases de ascenso y descenso. Para ello se ha diseñado el siguiente bucle de control ilustrado en la figura

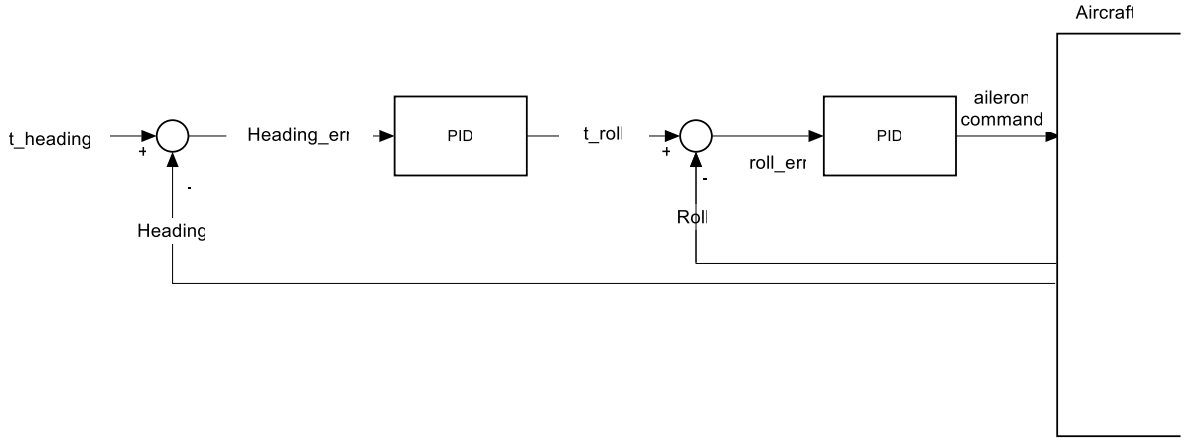


Ilustración 4.12 Bucle de control de los tramos de mantenimiento de rumbo

Este bucle consta de dos PIDs, uno de ellos para obtener el roll o alabeo deseado para un determinado error en rumbo y otro para obtener la acción deseada para el error de alabeo.

Tramo horizontal por Waypoint

Durante estos tramos, la aeronave intentará alcanzar un Waypoint concreto. Dado que el foco del proyecto se encuentra en el control vertical, se ha optado por un sistema de control sencillo, tomando como referencia de control el rumbo entre el waypoint y la situación actual de la aeronave. Por ello, el sistema de control será igual al mostrado en la Ilustración 4.12.

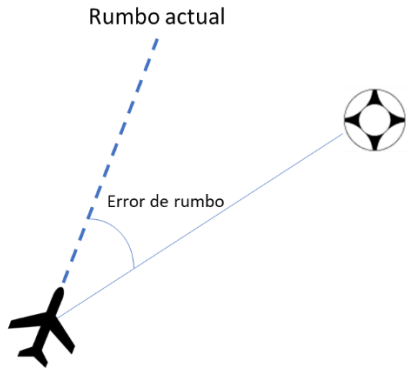


Ilustración 4.13 Error de rumbo para los tramos horizontales por Waypoint

Tramo de aterrizaje??

Segmento

3. Tramos Vertical Verticales

De forma análoga a los tramos horizontales, los tramos verticales se encargan del control en el plano vertical de la aeronave. Esto incluye el control del acelerador, elevadores y flaps.

Tramo de despegue

Este tramo se usa para la sección de tierra del despegue. Durante este tramo se mantendrán los flaps extendidos para aumentar la sustentación y se mantendrá el acelerador constante al máximo de despegue hasta alcanzar la velocidad mínima de despegue. Durante este tramo no se accionarán los elevadores.

Tramo SPD

Los tramos SPD o tramos de control por velocidad son segmentos en los que se tendrá como objetivo mantener una velocidad de ascenso/descenso concreta.

En los casos en que el plan de vuelo lo permita, la velocidad de ascenso y descenso se obtendrá de la Tabla de Prestaciones de Bada, descrita en el apartado 4.5. Esta tabla provee las velocidades optimas de vuelo para cada nivel de vuelo, por lo que la referencia de control podría cambiar durante el tramo.

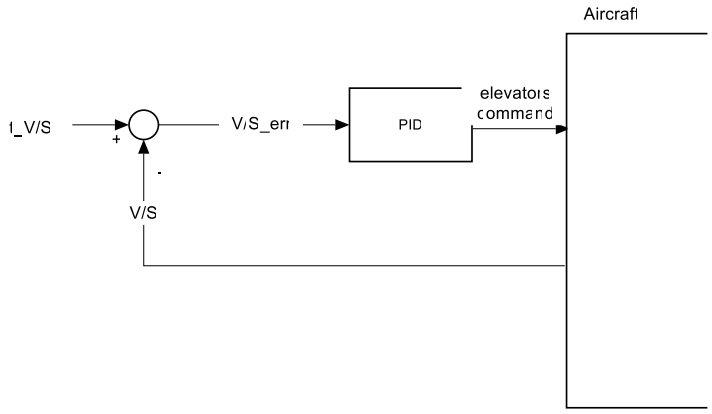


Ilustración 4.14 Bucle de control de los elevadores en tramo SPD

Tramo PTH

Los Tramos PTH tienen como objetivo mantener un ángulo de trayectoria de vuelo constante. Estos tramos se usan generalmente como descensos. En este proyecto solo se usaran como parte de la aproximación y aterrizaje y estas etapas se suelen realizar con los motores al en “iddle”, por tanto el control de throttle será un control fijo a 0.

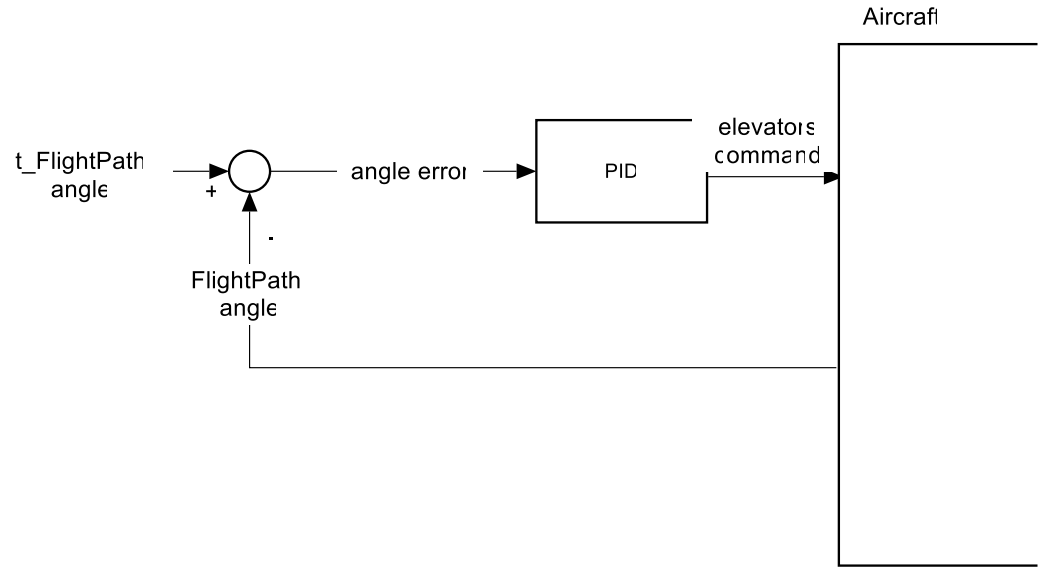


Ilustración 4.15 Bucle de control de elevadores para un tramo PTH

Tramo Hold

Los tramos Hold o mantenimiento tienen como objetivo mantener una altitud y velocidad concretas. La altura a mantener se obtendrá a partir del plan de vuelo mientras que la velocidad de mantenimiento se obtendrá de la Tabla de Prestaciones de Bada de forma similar al Tramo SPD.

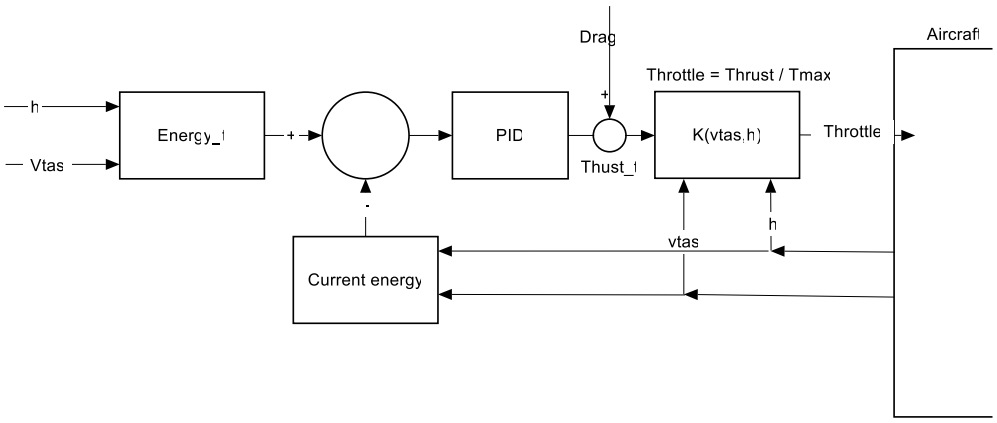


Ilustración 4.16 Bucle de control del Throttle para un tramo Hold

4. Terminaciones

Dependiendo del tipo de segmento y fase se requerirá un tipo de terminación distinto para iniciar la transición al siguiente.

Esta terminación puede ser distinta para el segmento vertical y horizontal ya que tienen objetivos distintos.

Las terminaciones implementadas son terminación por Altura, Velocidad, Waypoint e Inicio del Descenso.

Altura y velocidad

Esta terminación se considerará cumplida cuando el aeroplano alcance el objetivo que se define al crear la terminación.

Dado que estas terminaciones pueden ser usadas tanto para ascenso y aceleración como descenso y deceleración se dará el segmento por terminado cuando se cumpla la siguiente condición:

$$ABS(objetivo - \ actual) < objetivo * e\_termination$$

El porcentaje de error (e_termination) se define por defecto como el 5%.

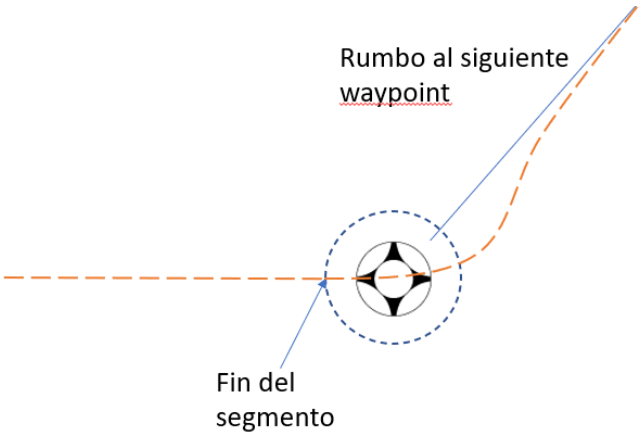
Waypoint

La terminación de waypoint será diferente dependiendo de si se trata de un waypoint de tipo flyover o flyby.

En el caso de flyover la condición será la siguiente:

$$Distancia(aeronave, waypoint) < flyoverMinimumDistance$$

La distancia mínima por defecto son 250 m.



Para flyby se deberá de calcular la distancia a la que se debe comenzar el giro coordinado. Esta distancia depende de la velocidad actual de la aeronave, la diferencia de rumbo antes y después del punto y la velocidad actual.

Para establecer la distancia del waypoint desde donde empezar a girar, primero se ha de calcular radio de giro. Para ello se usará la siguiente ecuación:

$$\mathrm{R} = \frac{\mathrm{Vtas}}{\mathrm{g} \cdot \tan(\phi_{\mathrm{nominal}})}$$

En esta ecuación ϕ_{nominal} es el ángulo de alabeo nominal para un giro coordinado. Este valor dependerá de la aeronave actual.

La distancia de inicio del giro dependerá de la diferencia entre el rumbo actual del avión y el rumbo hacia el siguiente punto

$$\mathrm{Distancia} = \mathrm{R} \cdot \tan(\frac{\mathrm{cambio\ de\ rumbo}}{2}) + \mathrm{distancia\ de\ preparacion}$$

La distancia de preparación de giro se suele estimar como 5 segundos a la velocidad de vuelo actual.

Inicio del Descenso (TOD)

El inicio del descenso o Top Of Descent en inglés se usa como final para la fase de crucero y viene determinado por el ultimo punto en el que se puede comenzar el descenso y alcanzar el primer punto de la fase de aproximación con la altura adecuada.

Para determinar si se ha de comenzar el descenso se compara el tiempo restante para alcanzar el waypoint en el eje horizontal y el tiempo necesario para descender a la altura adecuada a la velocidad de descenso (ROD) indicada para esta altura por el perfil de prestaciones.

$$\frac{Distancia(aeronave, waypoint)}{Vtas} < \frac{altura\ actual - altura\ waypoint}{ROCD(altura)} + Margen\ TOD$$

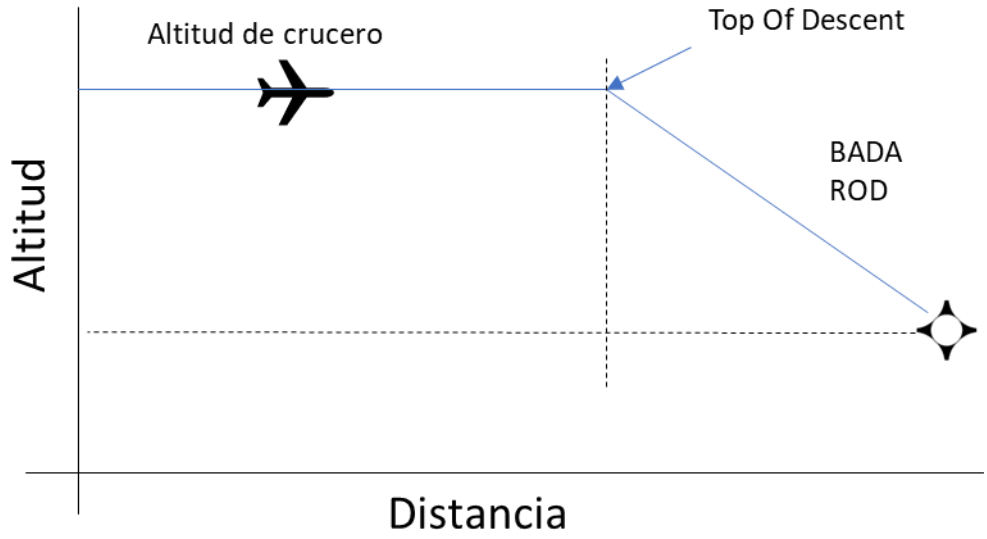


Ilustración 4.17 Top Of Descent

Para asegurar que se alcanza la altura antes de llegar al punto se añadirá un margen de 120 segundos.

5. *Generación del plan de misión a partir del plan de vuelo*

Una vez se ha definido el plan de vuelo y se conocen los diferentes tramos disponibles para el control, se procede a detallar el proceso de generación del plan de misión a partir del plan de vuelo.

Los tramos verticales se generarán de forma independiente a los horizontales.

Al resultar más simples comenzaremos con el proceso de traducción de lista de waypoints a lista de tramos horizontales.

Para generar la lista horizontal se comenzará con un tramo de despegue. Este tramo mantendrá al avión alineado con la pista mientras se encuentra en tierra. Una vez se alcance la velocidad mínima de despegue se pasará a un tramo de control por rumbo, manteniendo el avión en el mismo rumbo en el aire hasta alcanzar la altitud mínima para comenzar el giro.

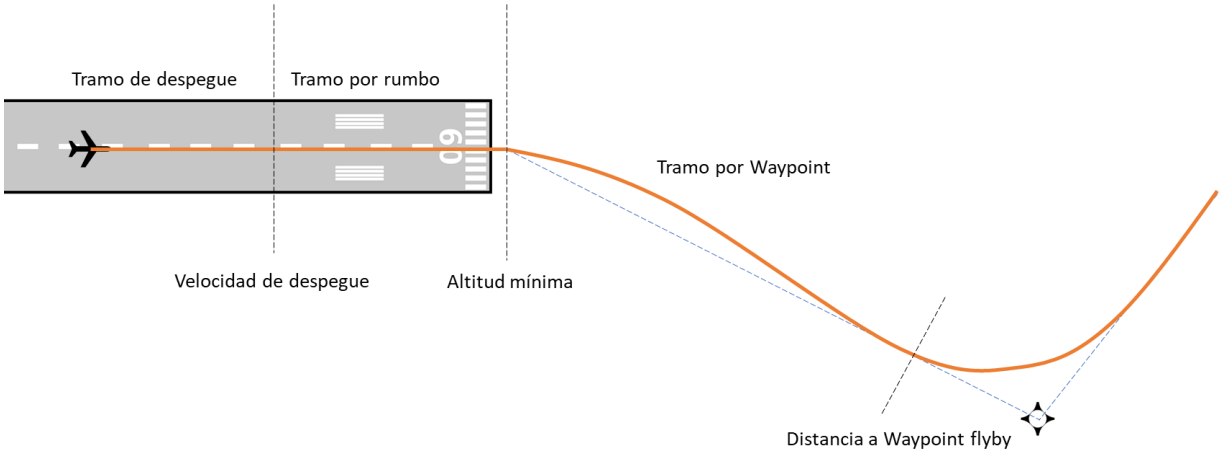


Ilustración 4.18 División por tramos en el eje horizontal

Una vez alcanzada la altitud mínima se procederá a añadir un nuevo tramo por waypoint por cada tramo hasta alcanzar la zona de aterrizaje.

En el plano vertical se comenzará con un tramo de despegue, que acelerará el avión hasta alcanzar la velocidad mínima de despegue. Una vez alcanzada esta velocidad se comenzará el ascenso, con un tramo de velocidad constante. Al llegar a la altitud mínima, se comenzará a ascender a la altura de crucero. Esto podría realizarse en un único tramo de ascenso si no existen limitaciones de altura en las cartas de salida, pero esto no es frecuente.

Para poder resolver los posibles límites de altura de las cartas de salida se recorrerá la lista de waypoints de la fase de ascenso y se añadirán dos nuevos tramos por cada cambio en los límites de altura.

En caso de encontrarse un Waypoint con un límite de altura máximo superior a la altura de vuelo actual se añadirá un tramo nuevo de ascenso por velocidad constante con terminación al alcanzar dicha altura y un tramo de mantenimiento de altura hasta alcanzar dicho Waypoint.

Una vez se han añadido todos los tramos derivados de las limitaciones de ascenso, se añadirá un nuevo tramo de ascenso hasta la altitud de crucero. En cuanto se alcance esta altitud, se considerará que el aeroplano se encuentra en fase de crucero y no variará la altura hasta llegar al Top Of Descent.

En la Ilustración 4.19 se puede ver un ejemplo de división de un plan de vuelo ejemplo. Comienza con el tramo de despegue y el ascenso a la altitud mínima de vuelo. Después del tramo de ascenso a la altura mínima, se buscara la siguiente limitación superior, añadiendo un tramo de ascenso hasta esta altitud y un tramo de mantenimiento de altitud hasta salir de la zona con limitación de altitud. Este proceso se repetirá hasta que no queden waypoints en fase de ascenso.

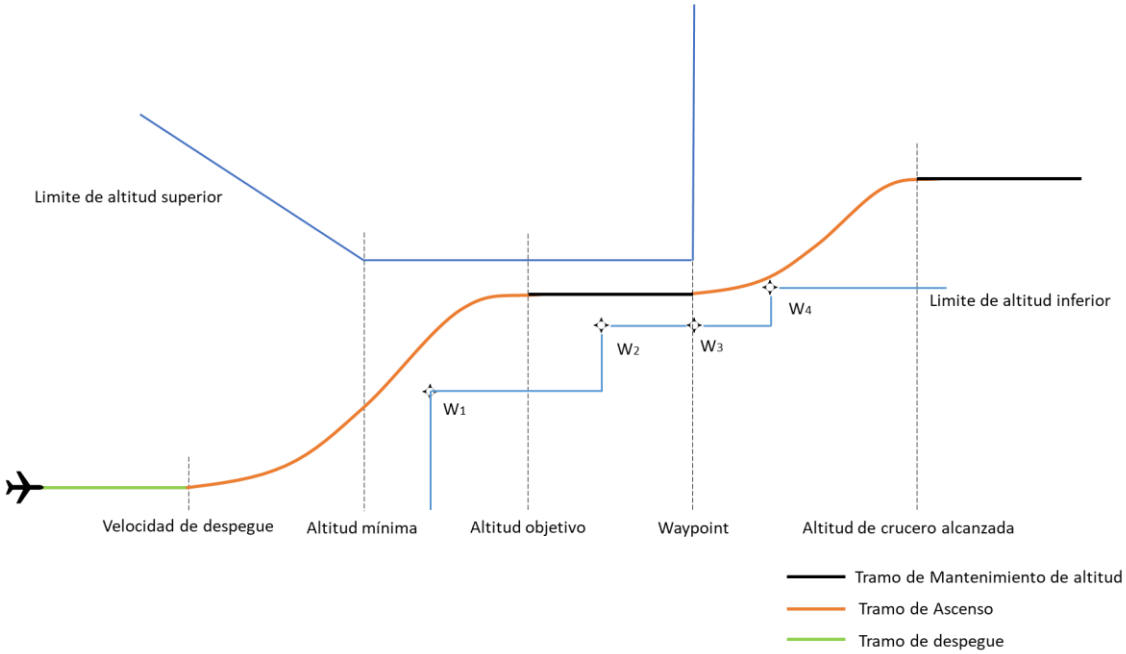


Ilustración 4.19 División por tramos del despegue y ascenso en el plano vertical

En caso de querer realizar un cambio de nivel de vuelo, solo se requerirá añadir un tramo de ascenso/descenso al nivel de vuelo deseado y un tramo de mantenimiento de altitud hasta ToD.

Una vez se alcanza el Top Of descent, se seguirá el proceso inverso al ascenso, añadiendo tramos de descenso por velocidad constante hasta la altitud mínima del siguiente punto y después manteniendo la altura hasta alcanzarlo. Esto se repetirá hasta el penúltimo waypoint (siendo el ultimo la posición de la pista de aterrizaje) y se añadirá un ultimo tramo de descenso por ángulo de vuelo que descenderá hasta la altura de la pista.

4.9 Controladores PID

Los controladores PID o controladores Proporcional, Integral y Derivativo son mecanismos de control ampliamente utilizados en sistemas de control industrial. Para realizar un control por PID se necesitará un error, es decir, la diferencia entre el estado actual y la referencia de control. A este error se le aplicara cada uno de los componentes del PID y la suma de los resultados de los tres componentes constituirá la acción de control que se usara para controlar el proceso.

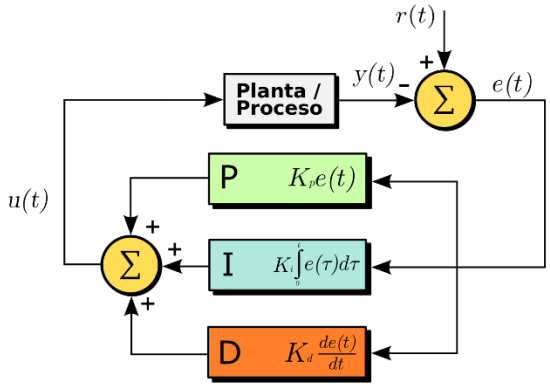


Ilustración 4.20 Diagrama de bloques de un PID en lazo cerrado

1. Ajustado de los PID

Para que un PID sea capaz de controlar un proceso de forma adecuada, cada uno de los elementos del control han de ser ajustados para el sistema concreto. Para ello, primero se ha definir qué tipo de respuesta se espera del aeroplano cuando este en vuelo.

Oscilaciones ante un cambio de objetivo son indeseables, ya que inducen desgaste sobre los actuadores y pueden resultar incomodas para la tripulación, por lo que se buscara siempre una respuesta sobreamortiguada, acercándose lo máximo posible a una respuesta críticamente amortiguada.

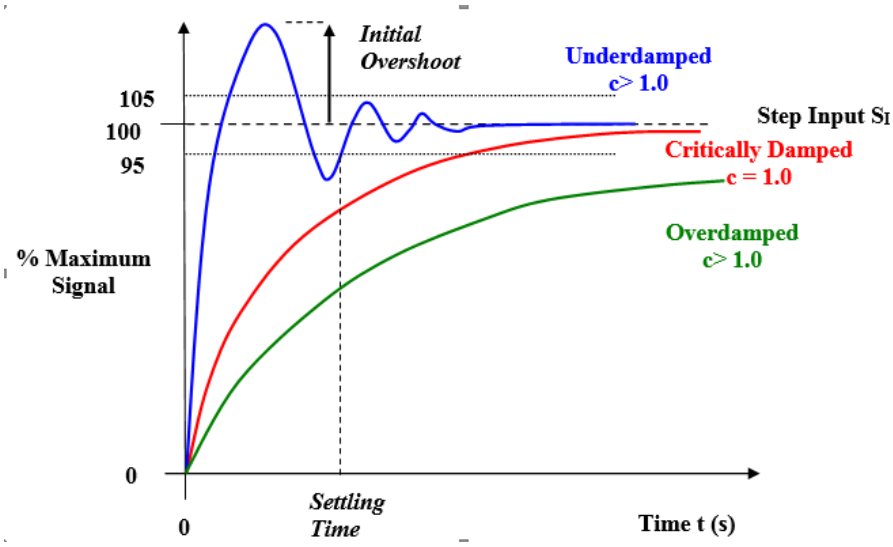


Ilustración 4.21 Respuesta ante escalón dependiendo del factor de amortiguación

Existen métodos de cálculo numérico para controladores PID, pero estos requieren un conocimiento exhaustivo de la respuesta del sistema y esa información no se encuentra disponible para cada uno de los actuadores de la aeronave.

Una alternativa es el método de Ziegler-Nichols. Este método permite obtener unos primeros valores de ajuste para un PID sin conocer las condiciones concretas del sistema.

El método consiste en realizar un control proporcional con una ganancia arbitraria e ir aumentando (o reduciendo) esta ganancia hasta que el sistema comience a oscilar de forma estable. En este punto se medirán la ganancia (K_u) y el periodo de oscilación (T_u).

Una vez obtenidas las características del sistema, se puede aplicar la regla de Ziegler-Nichols para obtener unos primeros valores para un sistema de control adecuado. Desde la publicación de este método otros investigadores han expandido el método con otros conjuntos de reglas dependiendo de la respuesta deseada para el sistema. En la siguiente tabla se pueden ver algunas de las más destacadas:

Tipo	K_P	K_I	K_D
Ziegler-Nichols (PID Clásico)	0.6 * K_u	1.2 * K_u/ T_u	3/40 * K_u*T_u
Pessen Integral Rule	0.7 * K_u	1.75 * K_u/ T_u	21/200 * K_u*T_u
Ligera sobreoscilación	K_u/3	2/3 * K_u/ T_u	K_u*T_u/9
Sin sobreoscilación	K_u/5	2/5 * K_u/ T_u	K_u*T_u/15

Tabla 4.1 Valores iniciales para PIDs (McCormack & Godfrey, 1998)

Para el calculo de estas acciones se desarrollo una interfaz de usuario que permite editar los valores de las ganancias PID de los distintos controladores de los tramos y así estimar los valores del modelo

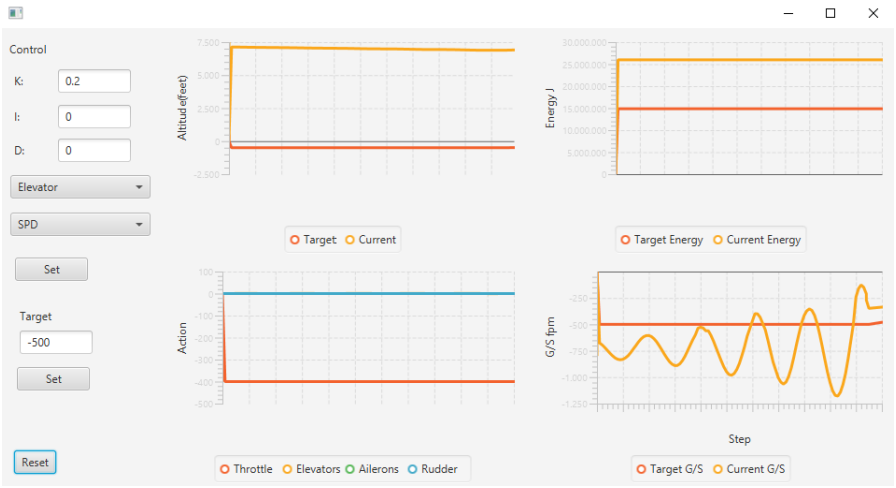


Ilustración 4.22 Estimación de Ku y Tu para un tramo de SPD en el Cessna C172

Una vez encontrada al ganancia y periodo, se ha aplicado la regla Sin Sobreoscilación (McCormack & Godfrey, 1998) ya que se busca una respuesta subamortiguada. Para este caso los resultados son los siguientes:

SPD			
elevators	VGS		
Ku	0,19		
Tu	3,95	Sampling Period	0,2
LO_OV	0,063333	0,0320675	0,0053446
NO_OV	0,038	0,0192405	0,0032068

Tabla 4.2 Calculo de los valores del PID

A partir de estos valores se ajustara el control de forma manual hasta obtener un resultado aceptable.

Tabla 4.3 Repuesta del controlador despues de la primera ronda de ajustes

Posteriormente se realizara el ajuste de los PIDs para el resto de actuadores y modos.

Capítulo 5. Implementación

En este capítulo se detallará la implementación del sistema, detallando el procedimiento de comunicación con el simulador, las estructuras de clases y de datos que se han desarrollado en base a los modelos definidos en el capítulo anterior y la interacción entre los diferentes módulos que lo componen.

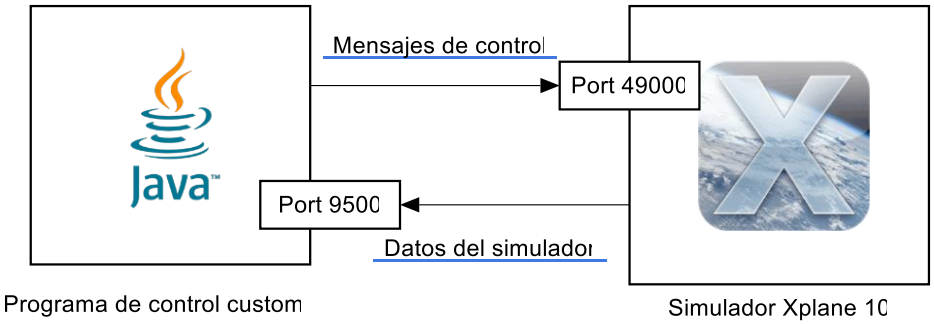
5.1 Comunicación con el simulador Xplane10

La comunicación con el simulador se realiza mediante el intercambio de paquetes con el protocolo TCP/IP.

El protocolo TCP/IP es comúnmente utilizado para comunicación entre computadores a través de internet y permite establecer conexiones fiables. Para identificar al destinatario se utiliza una combinación de IP y puerto.

La IP sirve como identificador de la interfaz a la que esta conectada y viene expresada por cuatro bloques de números de 0 a 255 separados por puntos en IPV4, por ejemplo 192.168.1.1.

Para identificar cual de los programas que se encuentran en ejecución actualmente en la máquina es el destinatario del mensaje se utiliza el concepto de puerto. Estos puertos se pueden reservar utilizando sockets y permiten que diferentes programas puedan recibir información sin que se vean afectados.



Para intercambiar mensajes con Xplane se ha de configurar una conexión de salida y otra de entrada, los puertos por defecto son 49000 y 9500 respectivamente.

Xplane permite configurar las categorías de datos que se desea extraer del simulador.

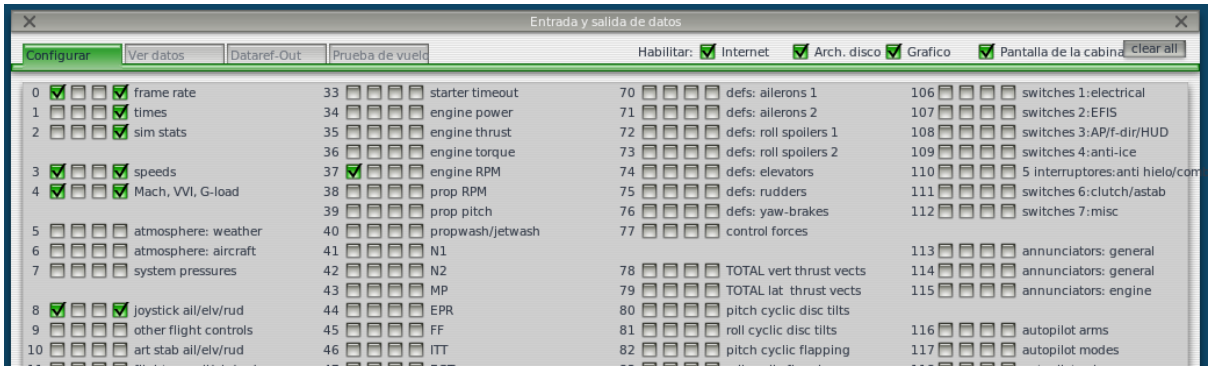


Ilustración 5.1 Pantalla de configuración de IO de Xplane10

Como se puede observar en la figura anterior, los diferentes datos se separan en diferentes categorías. Para poder obtener correctamente estos datos se ha de configurar tanto el simulador como la sección de código encargada de decodificar los mensajes.

Esta comunicación se establece a partir de una librería configurable llamada XPlane-Interface (XPI) (Cantoni, 2015). Esta librería se encarga de configurar las conexiones de entrada y salida para el software controlador y lanzar los hilos de envío, recepción y decodificación de los mensajes.

Una vez se ha recibido y decodificado un mensaje, su información se almacena en un objeto de la clase DataXplane para poder usarlo en el programa. De forma análoga, a la hora de enviar ordenes al simulador se guardarán en la clase ActionXplane y la librería se encargará de realizar el empaquetado y envío al simulador Xplane.

5.2 Diagramas de Flujo

En la siguiente sección se resumirá el flujo del programa de autopiloto

5.3 Ejecución principal

El flujo principal del programa se muestra en la siguiente figura:

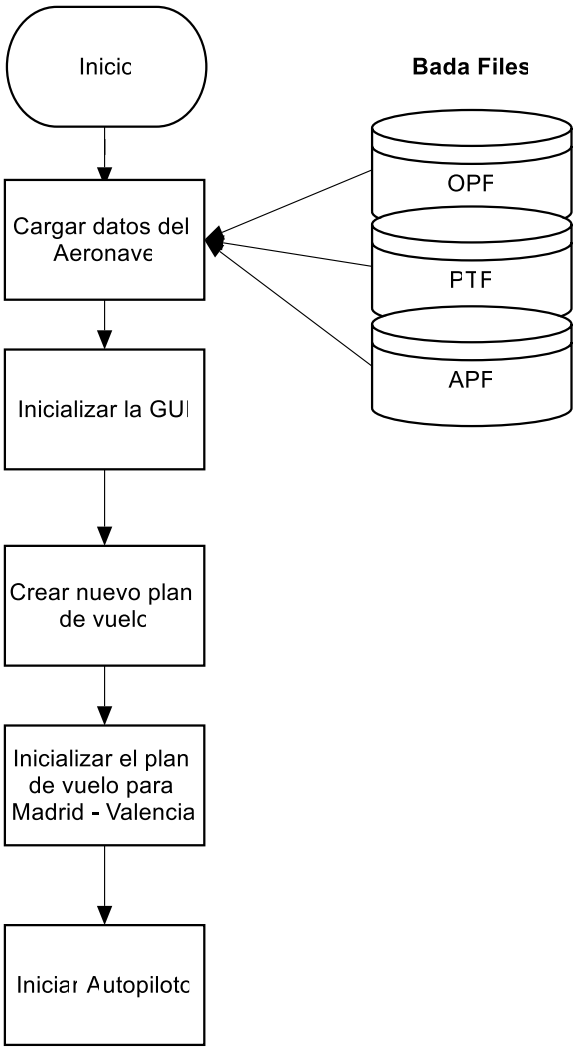


Ilustración 5.2 Diagrama de flujo principal

El programa comenzará cargando los coeficientes de los diferentes archivos de BADA y se almacenan en la clase Aircraft, detallada en la sección 5.7. Esta clase contendrá los modelos de la aeronave y los métodos necesarios para calcularlos.

A continuación, se inicializará la interfaz de usuario. Esta

Posteriormente se creará un nuevo plan de vuelo y se inicializará con los datos del set de test. Este plan de vuelo realizara el trayecto Madrid-Valencia.

Una vez se han cargado e inicializado los datos se puede comenzar el bucle de control del autopiloto.

5.4 División por paquetes

Para mejorar la mantenibilidad y facilitar el uso del código de forma modular, las distintas clases desarrolladas se han dividido en paquetes. Los paquetes un sistema muy utilizado en la programación orientada a objetos para agrupar conjuntos de clases con dependencias mutuas y funcionalidades similar

Los paquetes principales son:

- **gui**: contiene las clases de la interfaz gráfica de usuario y se encarga del arranque del programa.

- **autopilot**: el paquete autopilot contiene la principal clase de autopiloto, los controladores P y PID y las clases de interfaz de lectura/escritura con Xplane.
- **bada**: el paquete bada contiene las clases que leen los distintos archivos de la base de datos y la clase Aircraft que almacenará esos datos y los métodos de cálculo del modelo de aeroplano.
- **flightPlan**: este paquete contiene las clases que implementan los planes de vuelo y de misión. Esto incluye Waypoints, Legs (tramos) y sus terminaciones.
- **xplane**: el paquete Xplane contiene la librería de comunicación con XPlane10 XPI (Cantoni, 2015).
- **utilities**: el paquete utilities contiene los métodos de conversión de unidades en la clase UnitConverter y las constantes utilizadas por el resto del programa.

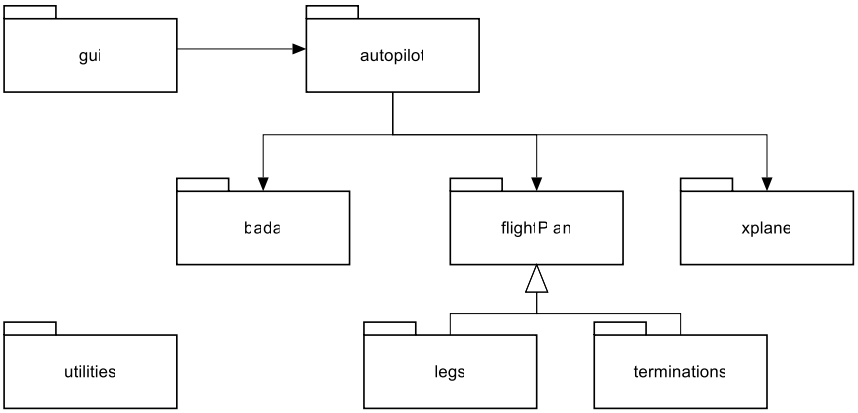


Ilustración 5.3 Diagrama de paquetes

5.4.1 Diagrama de clases principal

En el siguiente diagrama se enumeran las clases del proyecto y la relación entre ellas:

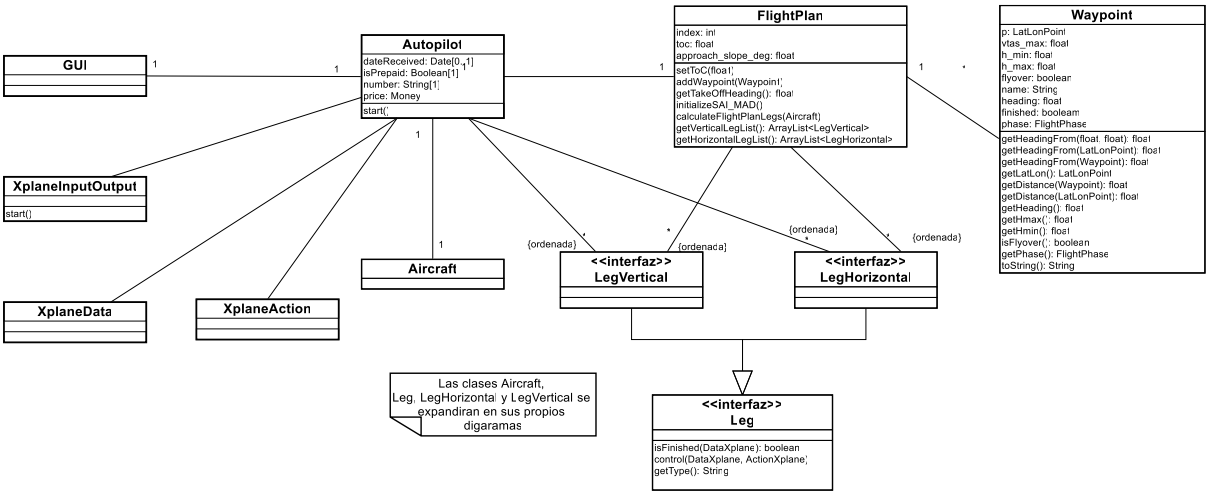


Ilustración 5.4 Diagrama de clases principal

En este diagrama cabe destacar la clase Autopilot, esta es la clase principal del proyecto y contiene el flujo principal del sistema de control y guiado.

5.5 Paquete GUI

La interfaz de usuario se ha desarrollado usando el esquema FXML. Este esquema esta compuesto por una clase lanzadora, una clase controlador y un archivo FXML que contiene la disposición de los diferentes elementos de la interfaz gráfica.

Para la Interfaz de usuario se ha priorizado que se puedan consultar el estado actual de la aeronave en el plano vertical y la acción ejercida sobre los distintos actuadores.

Ilustración 5.5 Interfaz gráfica del sistema de autopiloto

También permite introducir un nuevo objetivo de altura para realzar un cambio de altitud durante la fase de crucero.

5.6 Paquete autopilot

Clase Autopilot.java

La clase Autopilot contiene el bucle de control y guiado por tramos que se seguirá durante el vuelo. Gracias al sistema de control por tramos, la implementación de cada tipo de control se realiza dentro de las clases LEG, por lo que la fase de vuelo y tipo de guiado que se esté llevando a cabo a cada momento resulta transparente para el autopiloto.

En la Ilustración 5.6 se puede ver el diagrama de flujo de este del autopiloto:

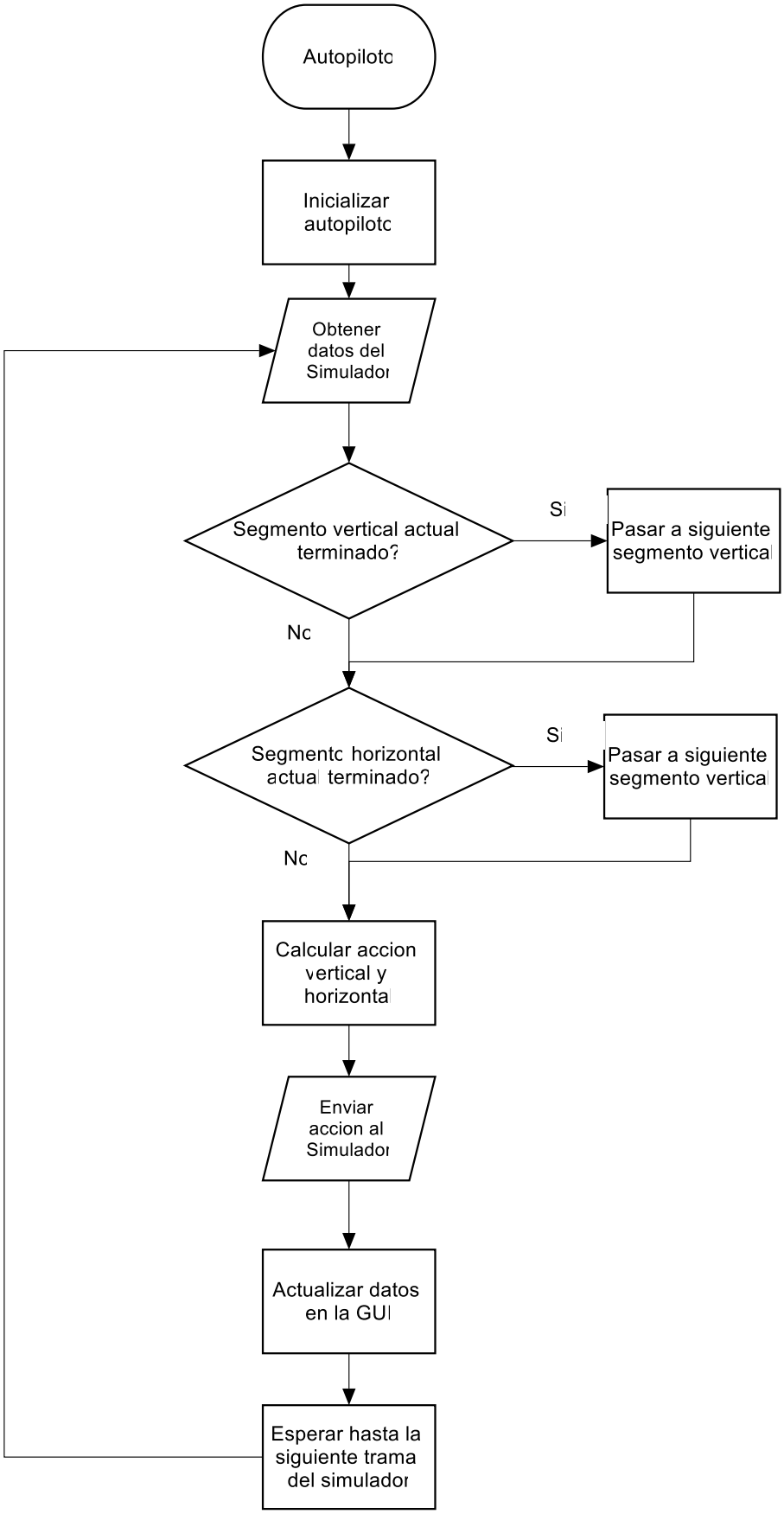


Ilustración 5.6 Diagrama de Flujo del autopiloto

Durante su inicialización, el autopiloto generará las listas de segmentos (vertical y horizontal) del plan de misión acorde con el plan de vuelo y creará una instancia de la clase *XplaneInputOutput* para comunicarse con el simulador.

Una vez se han inicializado los componentes comienza el bucle de control del autopiloto. Este bucle es sencillo, realizando los siguientes pasos en cada iteración:

1. Obtener los datos actuales del simulador
2. Consultar si los segmentos actuales han terminado, pasando al siguiente si lo han hecho.
3. Calcular la respuesta de control, llamando a la función de control del segmento actual
4. Enviar la acción calculada al simulador
5. Actualizar los datos en la GUI
6. Esperar hasta que el simulador envíe la siguiente trama

Atributos

Atributo	Valor por defecto	Descripción
private FlightPlan flightPlan	null	Objeto FlightPlan que contendrá el plan de vuelo y de misión
private Aircraft ai	null	Objeto Aircraft que contendrá el modelo de BADA para este vuelo
private XplaneInputOutput xplane	null	Objeto interfaz con el simulador
private DataXplane data;	null	Objeto DataXplane para la lectura de datos del simulador
private ActionXplane action	null	Objeto ActionXplane para el envío de ordenes al simulador
private FXMLDocumentController gui	null	Instancia del controlador de la gui para actualizar los datos de esta
private ArrayList<LegHorizontal> horizontalLegList	null	Lista de tramos horizontales para el control del vuelo
private int h_index	0	Índice actual de la lista de tramos horizontales
private ArrayList<LegVertical> verticalLegList;	null	Lista de tramos verticales para el control del vuelo
private int v_index	0	Índice actual de la lista de tramos verticales

Tabla 5.1 Tabla de Atributos de Autopilot

Constructores

```
public Autopilot(FlightPlan f, Aircraft ai, FXMLDocumentController gui)
```

Constructor de la clase Autopilot Inicializa el autopiloto y lanza el hilo de control

Parametros:
`f` – Plan de vuelo

`ai` – Objeto Aricraft con el modelo de BADA de la aeronave actual `gui` – Controlador de la interfaz gráfica actual

Métodos

Método	Parámetro	Descripción
run	-	Lanza el hilo de ejecución del autopiloto
flightEnded	-	Detiene el hilo de ejecución del autopiloto
setTarget	float alt	Añade un nuevo tramo de ascenso/descenso a la altitud alt (en metros)

Tabla 5.2 Tabla de Métodos de Autopiloto

Clase ControlP.java

La clase ControlP implementa un control proporcional.

Atributos

Atributo	Valor por defecto	Descripción
private float Kp	0	Ganancia proporcional del controlador
Private float minv	0	Valor mínimo de la acción del control
Private float maxv	0	Valor máximo de la acción del control

Tabla 5.3 Tabla de Atributos de ControlP

Constructores

`public ControlP(float Kp, float minv, float maxv)` Constructor de un nuevo control proporcional Parametros: `Kp` – Ganancia proporcional `minv` – Valor mínimo de la acción de control `maxv` – Valor máximo de la acción de control

Tabla 5.4 Constructor de ControlP

Métodos

Método	Parametro	Descripción
float Control	Float err	Calcula y devuelve la acción del controlador para un error err

Tabla 5.5 Tabla de Métodos de ControlP

Clase ControlPID.java

La clase ControlPID implementa un control de tipo PID.

Atributos

Atributo	Valor por defecto	Descripción
private float Kp	0	Ganancia proporcional del controlador
Private float KI	0	Ganancia integral del controlador
Private float Kd	0	Ganancia derivativa del controlador
Private float minv	0	Valor mínimo de la acción del control
Private float maxv	0	Valor máximo de la acción del control

Tabla 5.6 Tabla de Atributos de ControlPID

Constructores

```
public ControlP(float Kp, float Ki, float Kd, float minv, float maxv)
```

Constructor de un nuevo control proporcional

Parametros:
`Kp` – Ganancia proporcional
`Ki` – Ganancia integral
`Kd` – Ganancia derivativa
`minv` – Valor mínimo de la acción de control
`maxv` – Valor máximo de la acción de control

Tabla 5.7 Constructor de ControlPID

Métodos

Método	Parametros	Descripción
float Control	Float err	Calcula y devuelve la acción del controlador para un error err

Tabla 5.8 Tabla de Métodos de ControlPID

5.7 Paquete bada

El paquete bada contiene las clases que se encargan de leer los coeficientes de la base de datos e implementa el modelo de aeroplano para el calculo de la resistencia del aire y el empuje.

En la Ilustración 5.7 se muestra el diagrama de clases del paquete bada. Como se puede apreciar en el diagrama, la clase principal Aircraft contendrá una serie de clases auxiliares que permiten almacenar los diferentes coeficientes y datos de forma modular.

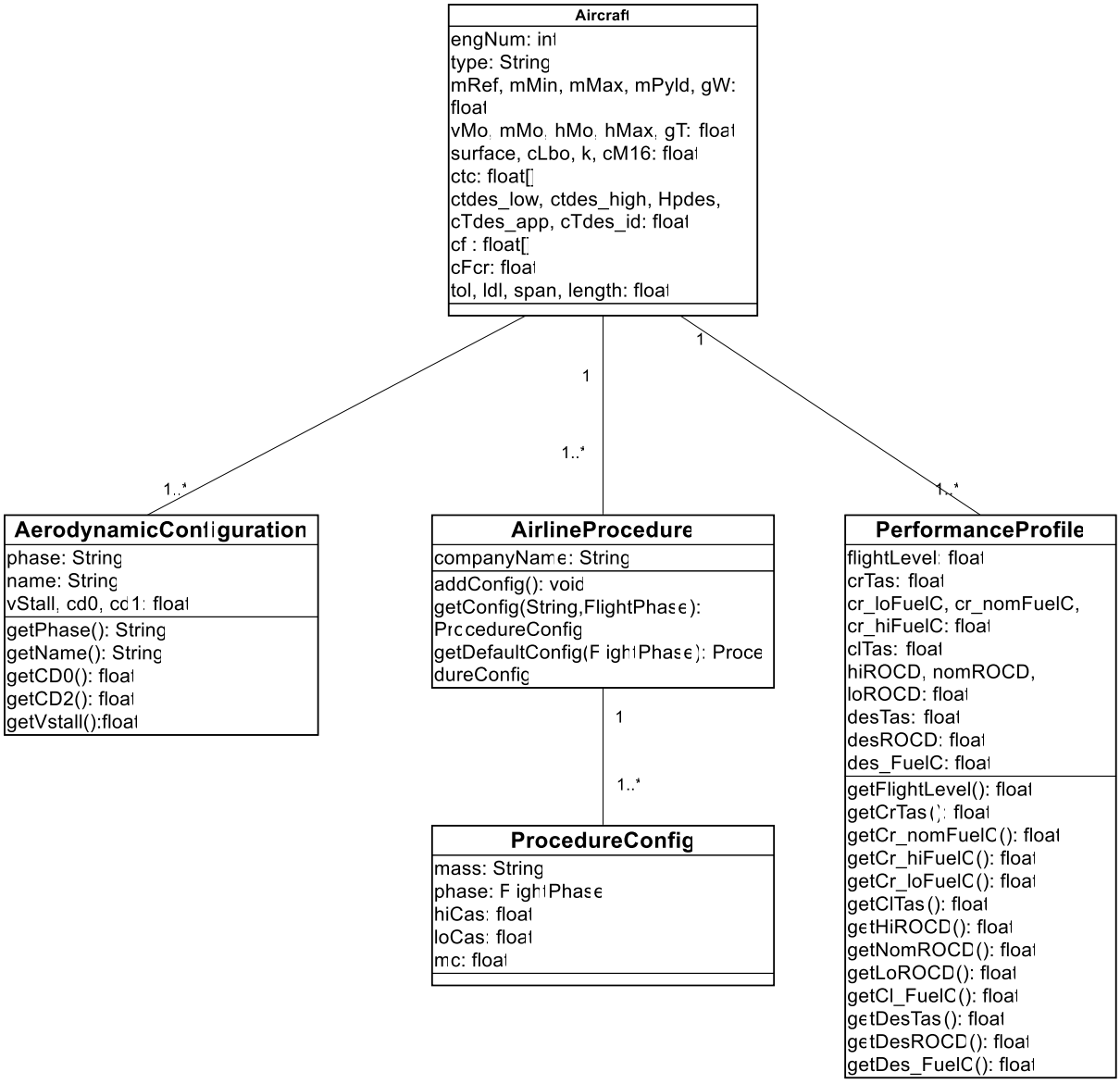


Ilustración 5.7 Diagrama de clases del paquete BADA

Clase Aircraft.java

La clase Aircraft almacena la información extraída de la base de datos BADA y contiene los métodos de calculo del modelo de aeronave.

Atributos

Los atributos de la clase Aircraft se pueden consultar en su diagrama de clases (Ilustración 5.7).

Constructores

```
public Aircraft(String filename)
```

Constructor de un nuevo objeto Aircraft Parametros: `filename` – dirección de los archivos de modelo de bada en el sistema excluyendo la extensión.

Tabla 5.9 Constructor de Aircraft

Métodos

Método	Parámetro	Descripción

addAerodinamicConfiguration	String Phase, String name, float vStall, float cd0, float cd2	Crea y añade una nueva configuración aerodinámica
addAirlineProcedure	AirlineProcedure p	Añade un procedimiento de aerolínea
addPerformanceProfile	PerformanceProfile p	Añade un perfil de rendimiento
float calculateLiftCoef	float rho, float Vtas, float mass	Calcula el coeficiente de sustentación a partir de la densidad del aire, la velocidad y masa
float calculateLiftCoef	float rho, float Vtas, float mass, float bank	Calcula el coeficiente de sustentación a partir de la densidad del aire, la velocidad, masa y ángulo de alabeo
AerodynamicConfiguration getAC_byPhase	String phase	Busca la configuración aerodinámica para la fase de vuelo actual
int getEngNum	-	Devuelve el número de motores de la aeronave
PerformanceProfile getPerformanceProfile_byHeight	Float m	Busca en la lista de perfiles de rendimiento el adecuado para la altitud m
Float getVmin	String Phase	Calcula la velocidad de vuelo mínima para la fase Phase
Float maxTrhust	Float vtas, float Hp	Calcula el empuje máximo de los motores para la velocidad de vuelo vtas y la altitud geodésica Hp
setEngNum	Int eng	Configura el numero de motores
setFlightEnvelopeParameters	float vMo, float mMo, float hMo, float hMax, float gT	Configurara las caracteristicas del envoltorio de vuelo
	`float[] cf, float cfcr`	Configura los parámetros de consuimo de combustible
setGroundParameters	`float tol, float ldl, float span, float length`	Configura los parametros de tierra
setMassParameters	`float mRef, float mMin, float mMax, float mPyld, float gW`	Configura los parametros de masa

setThrustParametes	`float[] ctc,`
	`float ctdes_low,`
	`float ctdes_high,`
	`float Hpdes,`
	`float cTdes_app,`
	`float cTdes_id`
		Configura los parametrso de empuje

setType	`String type`	Configura el tipo de avión

setWingAndBuffer	`float surface,`
	`float cLbo, float k,`
	`float cM16`
		Configura los parametros de las alas

Tabla 5.10 Tabla de Métodos de Aircraft

5.8 Paquete flightPlan

El paquete flightPlan contiene las clases que se encargan de contener y generar la información del plan de vuelo y el plan de misión.

Clase FlightPlan.java

Atributos

Atributo	Valor por defecto	Descripción
private List<Waypoint> waypoints	null	Lista de Waypoints que seguirá el plan de vuelo
Int index	0	Índice en la lista de waypoints
private float toc	0	Altitud del Top Of Climb en metros
pivate float approach_slope_deg	0	Pendiente en grados de la fase de approach

Tabla 5.11 Tabla de Atributos de FlightPlan

Constructores

`public FlightPlan()`

Constructor de la clase FlightPlan que inicializa un plan de vuelo vacío

Métodos

Método	Parámetros	Descripción
addWaypoint	Waypoint w	Añade el waypoint al final de la lista
calculateFlightPlanLegs	Aircraft ai	Calcula los tramos del plan de misión a partir de la lista de waypoints del plan de vuelo
List <LegHorizontal> getHorizontalLegList	-	Devuelve la lista de tramos horizontales del plan de misión
Waypoint getPreviousWaypoint	-	Devuelve el waypoint anterior
float getTakeOffHeading	-	Devuelve el rumbo de despegue
List <Legvertical> getVerticalLegList	-	Devuelve la lista de tramos verticales del plan de misión
initializeSAI_MAD	-	Inicializa el plan de vuelo al para un vuelo Madrid-valencia
Boolean nextWaypoint	-	Avanza el índice al siguiente Waypoint de la lista, devuelve false si se ha alcanzado el final, true si no
setTOC	Float toc	Configura la altitude del Top Of Climb

Tabla 5.12 Tabla de Métodos de FlightPlan

Clase Waypoint.java

Constantes

Constante	Valor	Descripción
Static Float MAX_H	FLOAT.MAX	Valor máximo posible para la altura
Static Float MIN_H	FLOAT.MIN	Valor minimo posible para la altura
Static Float MAX_VTAS	FLOAT.MAX	Valor máximo posible para la velocidad

Tabla 5.13

Atributos

Atributo	Valor por defecto	Descripción
private LatLonPoint p	null	Posición del waypoint
Private float vtas_max	0	Limite de velocidad para este waypoint
private float h_min	0	Altitud mínima para este waypoint
private float h_max	0	Altitud maxima para este waypoint
private Boolean flyover	False	Es este waypoint flyover o flyby (true para flyover, false para flyby)
private String name	""	Nombre del waypoing (e.g.: NANDO)
private Boolean finished	False	Este parámetro se pondrá a true cuando se halla alcanzado el Waypoint
private FlightPhase phase	Null	Fase de vuelo en la que se alcanzara este waypoint

Tabla 5.14 Tabla de Atributos de Waypoint

Constructores

```
public Waypoint(LatLonPoint p,float heading, float h_min, float h_max,
float vtas_max, boolean flyover, FlightPhase phase)
```

Constructor de un nuevo Waypoint

Parámetros:

`p` – LatLonPoint
`heading` - rumbo de salida del waypoint
`h_min` - Altura minima en metros
`h_max` - Altura maxima en metros
`vtas_max` - Velocidad maxima en metros por segundo
`flyover` - es flyover
`phase` - fase de vuelo

Métodos

Método	Parámetros	Descripción
Float getDistance	Waypoint w	Calcula la discancia desde otro Waypoint
Float getDistance	LatLonPoint p	Calcula la distancia desde un punto p
float getHeading	-	Devuelve el rumbo de salida del waypoint
Float getHeadingFrom	Float lat, float lon	Calcula el rumbo desde una latitud lat y longitud lon
Float getHeadingFrom	LatLonPoint p	Calcula el rumbo desde un punto p
Float getHeadingFrom	Waypoint w	Calcula el rumbo desde otro waypoint
Float getHmax	-	Devuelve el limite máximo de altitud para este waypoint
Float getHmin	-	Devuelve el limite mínimo de altitud para este waypoint
Float getVtasMax	-	Devuelve el limite máximo de velocidad para este waypoint
LatLonPoint getLatLon	-	Devuelve la posición del Waypoint
FlightPhase getPhase	-	Devuelve la fase de vuelo en la que se recorrerá el waypoint
Boolean isFinished	-	Devuelve el valor de Finished
isFinished	Boolean f	Cambia el valor de Finised a f
Boolean isFlyover	-	Devuelve true para Waypoints flyover y false para flyby
String toString	-	Sobrecarga del metodo standard toString, devuelve el nombre del waypoint

Tabla 5.15 Tabla de Métodos de Waypoint

5.9 Paquete flightPlan.leg

Interfaz Leg.java

Las interfaces son herramientas de java que permiten crear una serie de clases distintas que compartan unos métodos concretos, de esta forma, toda clase que herede de Leg deberá implementar sus métodos.

Métodos

Método	Parámetros	Descripción
Control	DataXplane data, ActionXplane action	Calcula la acción de control para los datos data y la escribe sobre el parámetro action
String getType	-	Devuelve el tipo de control de este tramo
Boolean isFinished	DataXplane data	Comprueba si el tramo ha terminado con los datos data

Tabla 5.16 Tabla de Métodos de Leg

Interfaz LegVertical.java

La interfaz LegVertical extiende a la interfaz Leg, añadiendo métodos especificos para los tramos del plano vertical.

Cada tipo distinto de control vertical tendrá una clase que implementará esta interfaz.

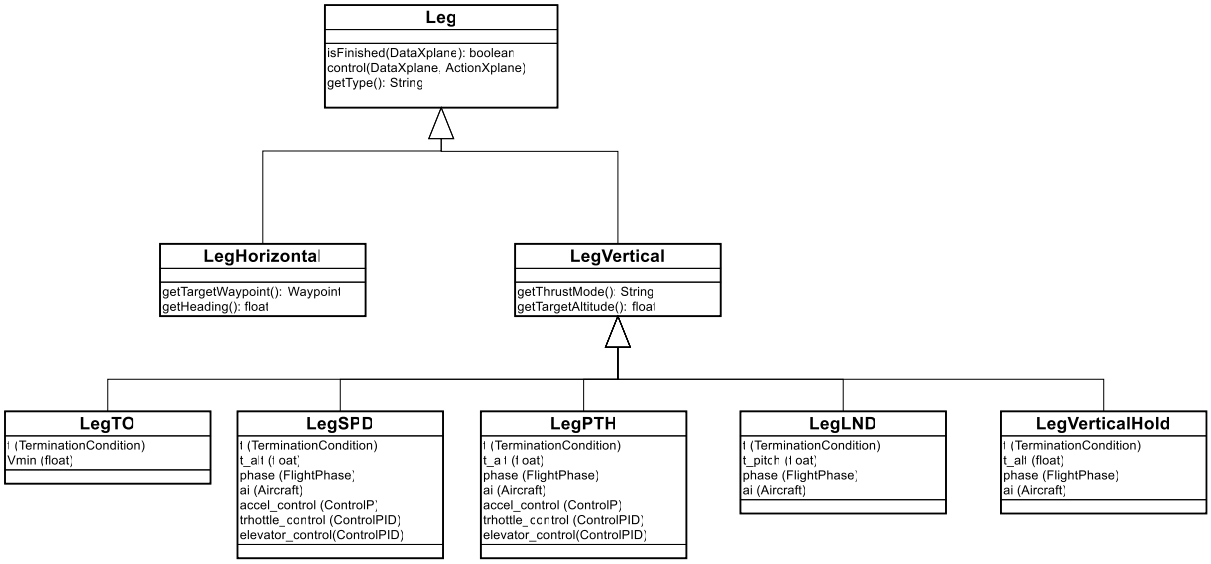


Ilustración 5.8 Diagrama de clases de LegVertical

Métodos

Método	Parámetros	Descripción
String getThrustMode	-	Devuelve el tipo de control sobre el empuje
Float getTargetAltitude	-	Devuelve la altura objetivo para este tramo

Tabla 5.17 Metodos de LegVertical

Interfaz LegHorizonal.java

De forma similar a la interfaz LegVertical, las clases de los tramos horizontales implementaran la interfaz LegHorizontal con sus tipos de control específicos.

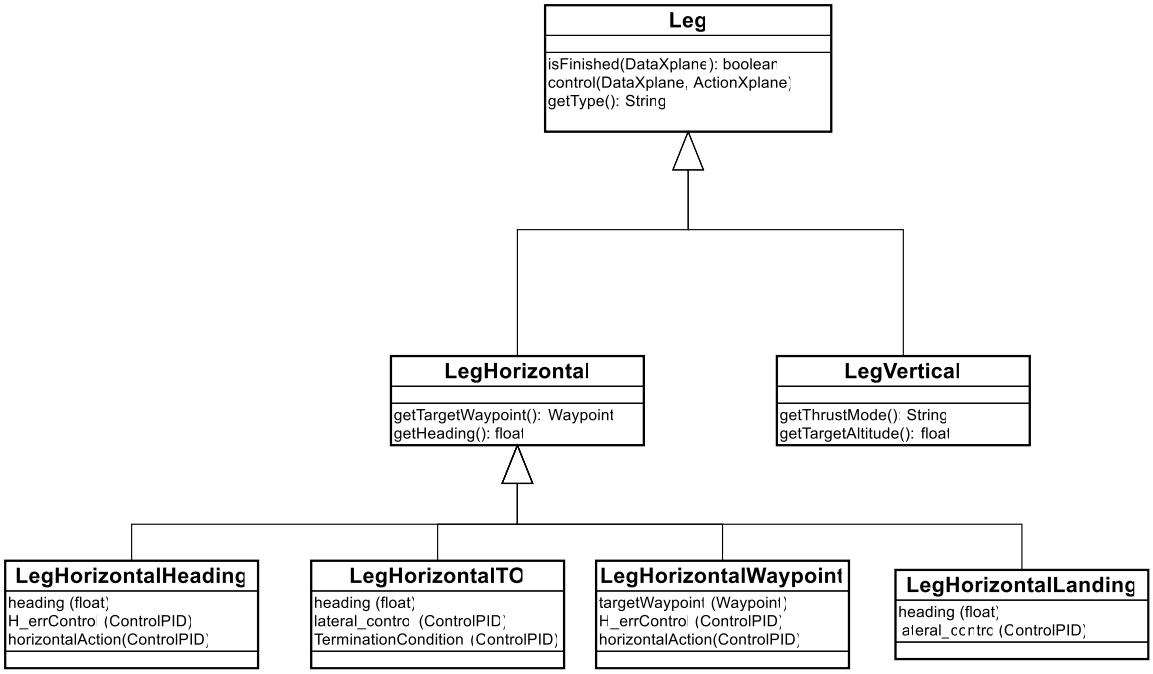


Ilustración 5.9 Diagrama de clases de LegHorizontal

Métodos

Método	Parámetros	Descripción
Waypoint getTargetWaypoint	-	Devuelve el Waypoint objetivo, null si no existe
Float getHeading	-	Devuelve el rumbo de este tramo

Tabla 5.18 Metodos de LegHorizontal

5.10 Paquete flightPlan.terminations

Las terminaciones de los tramos también se implementaran a través de una interfaz común, de forma que puedan ser utilizadas de forma intercambiable.

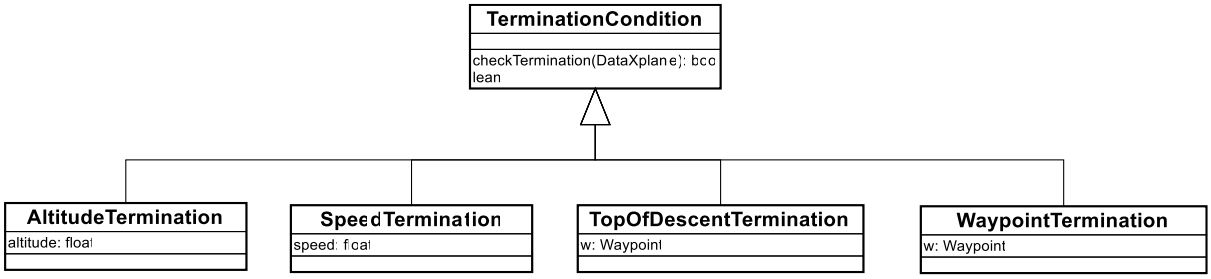


Ilustración 5.10 Diagrama de clases de TerminationCondition

Interfaz TerminationCondition.java

Los diferentes tipos de terminación implementaran el siguiente método:

Método	Parámetros	Descripción
boolean CheckTermination	DataXplane data	El método comprobara si se cumple la condición de terminación con los datos data

Tabla 5.19 Tabla de métodos de TerminationCondition

Capítulo 6. Validación y resultados

En el capitulo de validación se expondrá el proceso que se ha seguido para comprobar que el programa y sus diferentes componentes se comportan de forma adecuada.

6.1 Validación de ruta valencia Madrid

Para la validación del sistema se creó un plan de vuelo basado en la ruta Madrid-Valencia.

Este plan de vuelo se ha obtenido mediante el uso de las cartas de navegación correspondientes a la salida estándar de la pista 36R del aeropuerto de Madrid Barajas hasta el waypoint NANDO, la aerovía NANDO-ABOSI y la aproximación por instrumentos ILS de la pista 12 del aeropuerto de Valencia.

Con estas cartas se ha obtenido la siguiente serie de Waypoints como vuelo de validación para el sistema de control y guiado:

```
index = 0;
waypoints.add(new Waypoint("RWY 36", new LatLonPoint.Float(40.50f, -
3.559f), 359.5f, UnitConversor.feetToM(1918f), Waypoint.MAX_H ,
Waypoint.MAX_VTAS, false, FlightPhase.TAKE_OFF));

waypoints.add(new Waypoint("MD047", new DMSLatLonPoint(false, 40, 35,
37.1f, true, 3, 32, 17.6).getLatLonPoint(), 128,
UnitConversor.feetToM(2900f), Waypoint.MAX_H , UnitConversor.kntToMps(240)
, false, FlightPhase.CLIMB));

waypoints.add(new Waypoint("MD048", new DMSLatLonPoint(false, 40, 45,
13.2f, true, 3, 21, 33.3).getLatLonPoint(), 188,
UnitConversor.feetToM(7000), UnitConversor.feetToM(8000) ,
UnitConversor.kntToMps(240) , false, FlightPhase.CLIMB));

waypoints.add(new Waypoint("MD049", new DMSLatLonPoint(false, 40, 42,
12.4f, true, 3, 16, 19.9).getLatLonPoint(), 107,
UnitConversor.feetToM(8000), Waypoint.MAX_H , UnitConversor.kntToMps(240) ,
false, FlightPhase.CRUISE));

//Radioayuda para correccion de curso
waypoints.add(new Waypoint("PERALES", new DMSLatLonPoint(false, 40, 15,
10f, true, 3, 20, 52).getLatLonPoint(), 107, UnitConversor.feetToM(13000f),
Waypoint.MAX_H , UnitConversor.kntToMps(240) , false, FlightPhase.CRUISE));
```

```
waypoints.add(new Waypoint("NANDO", new DMSLatLonPoint(false, 39, 59,
19.9f, true, 2, 10, 28.4).getLatLonPoint(), 107,
UnitConversor.feetToM(13000f), Waypoint.MAX_H , UnitConversor.kntToMps(240)
, true, FlightPhase.CRUISE));

waypoints.add(new Waypoint("ABOSI", new DMSLatLonPoint(false, 39, 46, 45f,
true, 1, 17, 4).getLatLonPoint(),  107, Waypoint.MIN_H, Waypoint.MAX_H ,
Waypoint.MAX_VTAS , false, FlightPhase.CRUISE));

waypoints.add(new Waypoint("CALLES", new DMSLatLonPoint(false, 39, 42,
25.9f, true, 0, 59, 11.4f).getLatLonPoint(), 118,
UnitConversor.feetToM(6000), Waypoint.MAX_H , Waypoint.MAX_VTAS , true,
FlightPhase.DESCENT));

waypoints.add(new Waypoint("OPERA", new DMSLatLonPoint(false, 39, 37,
21.9f, true, 0, 46, 44.1f).getLatLonPoint(), 138,
UnitConversor.feetToM(5000), Waypoint.MAX_H , Waypoint.MAX_VTAS , true,
FlightPhase.DESCENT));

waypoints.add(new Waypoint("IF_DME", new DMSLatLonPoint(false, 39, 35, 9f,
true, 0, 44, 10.2f).getLatLonPoint(), 116, Waypoint.MIN_H,
UnitConversor.feetToM(3600) , Waypoint.MAX_VTAS , true,
FlightPhase.DESCENT));

waypoints.add(new Waypoint("FAP", new DMSLatLonPoint(false, 39, 33, 10.7f,
true, 0, 38, 57.2f).getLatLonPoint(), 116, Waypoint.MIN_H,
UnitConversor.feetToM(2800) , Waypoint.MAX_VTAS , true,
FlightPhase.DESCENT));

waypoints.add(new Waypoint("RWY 12", new LatLonPoint.Float(39.48359f, -
0.4666f), 116, Waypoint.MIN_H, Waypoint.MAX_H , UnitConversor.feetToM(240)
, true, FlightPhase.DESCENT));

approach_slope_deg = 3;
toc = UnitConversor.flToQNEM(100);
```

Ilustración 6.1 Codificación del plan de vuelo Madrid – Valencia

Esta ruta se ha realizado con ambos aviones, el Cessna C172 y el Boeing B737

6.2 Tramos con la ruta de validación

A partir de la ruta descrita en el apartado anterior se generarán las siguientes listas de tramos:

Lista de tramos horizontales

Tipo	Referencia de control	Terminación
TakeOff	Rumbo 365	Velocidad = vMin despegue
Heading	Rumbo 365	Altitud mínima de vuelo
Waypoint	MD047	Waypoint flyby alcanzado
Waypoint	MD048	Waypoint flyby alcanzado
Waypoint	MD049	Waypoint flyby alcanzado
Waypoint	PERALES	Waypoint flyby alcanzado
Waypoint	NANDO	Waypoint flyover alcanzado
Waypoint	ABOSI	Waypoint flyby alcanzado
Waypoint	CALLES	Waypoint flyover alcanzado
Waypoint	OPERA	Waypoint flyover alcanzado
Waypoint	IF_DME	Waypoint flyover alcanzado
Waypoint	FAP	Waypoint flyover alcanzado
Heading	Rumbo 116	-

Tabla 6.1 Lista de tramos horizontales para el vuelo Madrid Valencia

Lista de tramos verticales

Tipo	Referencia de control	Terminación
TakeOff	-	Velocidad = vMin despegue
Ascenso	Velocidad de ascenso definida por el perfil de rendimiento para la altitud actual	Altitud mínima de vuelo
Tramo de retracción de flaps	-	Un solo ciclo
Ascenso	Velocidad de ascenso definida por el perfil de rendimiento para la altitud actual	Limite de altitud superior de MAD048
HOLD	Limite de altitud superior de MAD048 7500	Waypoint MAD048 alcanzado
Ascenso	Velocidad de ascenso definida por el perfil de rendimiento para la altitud actual	ToC alcanzado
HOLD	Altitud de crucero	ToD alcanzado
Descenso	Velocidad de descenso definida por el perfil de rendimiento para la altitud actual	Altitud de OPERA alcanzada
HOLD	Altitud de OPERA	Waypoint OPERA alcanzado
Descenso	Velocidad de descenso definida por el perfil de rendimiento para la altitud actual	Altitud de IF_DME alcanzada
HOLD	Altitud de IF_DME	Waypoint IF_DME alcanzado

Descenso	Pendiente de descenso de 3°	Altitud de pista alcanzada
Aterrizaje	-	-

Tabla 6.2 Lista de tramos horizontales para el vuelo Madrid Valencia

Capítulo 7. Presupuesto

En el siguiente capítulo se detalla el análisis de costes y presupuesto para el proyecto. Las principales fuentes de costes son los costes de personal, costes de equipo y oficina y costes de licencia de software.

7.1 Costes de Personal

Durante el desarrollo del proyecto, dos roles de personal han participado en él, estudiante y profesor. La dedicación al proyecto se mediará en horas/persona.

A continuación, se desglosa el cálculo del coste de hora/persona para cada rol a partir del salario bruto anual:

Estudiante	Estudiante	Profesor
Salario anual	16.000,00 €	45.000,00 €
Coste anual	20.800,00 €	58.500,00 €
Horas al año	1080	1080
Coste por hora	19,26 €	54,17 €

Tabla 7.1 Calculo del coste por hora/persona de cada rol

7.2 Costes de equipo y oficina

Se repercutirán al proyecto los costes derivados de la amortización del equipo y del uso de instalaciones.

Para el equipo informatico, se ha realizado una amortización lineal a 5 años de los equipos y software:

Amortizacion	Gasto	Meses a amortizar	Coste de amortizacion al mes
Pantalla	110	60	1,83 €
Equipo	750	60	12,50 €
Licencia Windows 10 Home	145	60	2,42 €
		Total	16,75 €

Tabla 7.2 Amortización de equipos

Para contabilizar los gastos de oficina se han consultado la oferta actual para valencia y se ha estimado un gasto mensual de alquiler de 280€ (EasyOffices, 2020).

7.3 Costes de licencia de software

Los siguientes costes de licencia se repercutirán por completo al proyecto dado que se adquirirían de forma específica para realizarlo:

Licencias	**Cantidad**	**Coste unitario**	**Subtotal**
Licencia Xplane 10	1,00	44,47 €	44,47 €
Add-on Cessna C172	1,00	22,99 €	22,99 €
Licencia Microsoft 365 Empresa Estándar (meses)	4,00	10,50 €	42,00 €
Licencia ClickCharts	1,00	69,99 €	69,99 €
		Total	179,45 €

Tabla 7.3 Costes de licencia de software

7.4 Presupuesto Completo

DESCRIPCIÓN	CANTIDAD	PRECIO UNITARIO	SUBTOTAL
Desarrollo del proyecto (horas/alumno)	350,00	26,00 €	9.100,00 €
Formación y supervisión (horas/profesor)	35,00	54,17 €	1.895,83 €
amortización del equipo informático (meses)	4,00	26,00 €	104,00 €
Gastos de Oficina (meses)	4,00	250,00 €	1.000,00 €
Licencias	1,00	179,45 €	179,45 €
		Base Imponible	12.279,28 €
		Tipo Impositivo	0,21 %
		Total	**14.857,93 €**

Tabla 7.4 Presupuesto

Capítulo 8. Conclusiones

El proyecto ha cumplido con su objetivo principal que es proveer de un sistema de guiado y control vertical basado en la base de datos de Eurocontrol-BADA para los aviones B737 y C172 sobre el simulador de vuelo Xplane.

Además de esto, se ha desarrollado un sistema de control horizontal viable que ha demostrado ser más valido de lo esperado, circulando el vuelo de validación con precisión.

Por otro lado, el resultado ha sido un sistema modular, separado por paquetes que se podrían integrar fácilmente en otros proyectos y otorgarles nuevas funcionalidades. Entre estos caben destacar los conceptos de plan de misión, tramo y terminación, que otorgan una gran flexibilidad al independizar unos tramos de otros y no limitar un tipo de tramo a una condición de final concreta.

Durante el desarrollo del sistema se ha realizado un esfuerzo para desligar lo máximo el sistema de los aviones para los que se estaba desarrollando, utilizando al máximo la información que provee BADA para usarlo como capa de abstracción entre el sistema y los aviones simulados. Esto ha resultado en un éxito parcial, ya que el sistema de guiado y la predicción de trayectorias podría realizarse para cualquier avión siempre y cuando exista dentro de la base de datos, pero el control resulta demasiado especifico, necesitando usar unos controladores PID distintos para cada aeronave.

Capítulo 9. Trabajo Futuro

Durante la realización del proyecto se han detectado posibles mejoras que por limitaciones de tiempo y alcance se han excluido de la implementación. A continuación se enumeran algunas de las oportunidades de mejora más destacables:

- **La implementación de un sistema de control y guiado horizontal más complejo.** El guiado horizontal desarrollado para este proyecto es muy básico y, aunque funcional, limita el potencial del sistema. El desarrollo de un sistema de guiado horizontal capaz de ejecutar maniobras complejas como aterrizajes, sobrevuelos o giros de distintos radios ampliaría la funcionalidad del software.

- **Expandir el modelo de BADA para incluir Consumos y eficiencia**. El modelo de BADA permite hacer un calculo del consumo estimado del avión durante todo su vuelo. El proyecto entregado es capaz de obtener los coeficientes de los archivos, pero el modelo no aprovecha estos coeficientes. Añadir estos cálculos al sistema permitiría comparar los consumos de diferentes **maniobras o realizar estudios de eficiencia de las simulaciones**

- **Filtrado de mensajes inválidos.** En la implementación actual de este proyecto solo se han filtrado el ejemplo más común de mensajes inválidos, los mensajes con altitud “0.0”. Xplane envía estos mensajes mientras carga los assets y por esa razón ha resultado fácil detectar y eliminar estos mensajes que no son útiles para el sistema de control.
  Con cierta frecuencia se reciben otro tipo de mensajes erróneos, estos tienen valores anómalos o fuera de rango. Debido a la baja frecuencia, dificultad de predicción y el reducido efecto que tienen sobre el sistema se ha considerado aceptable ignorarlos en esta implementación.

  En la siguiente grafica se puede apreciar la recepción de un dato inválido durante una ejecución:

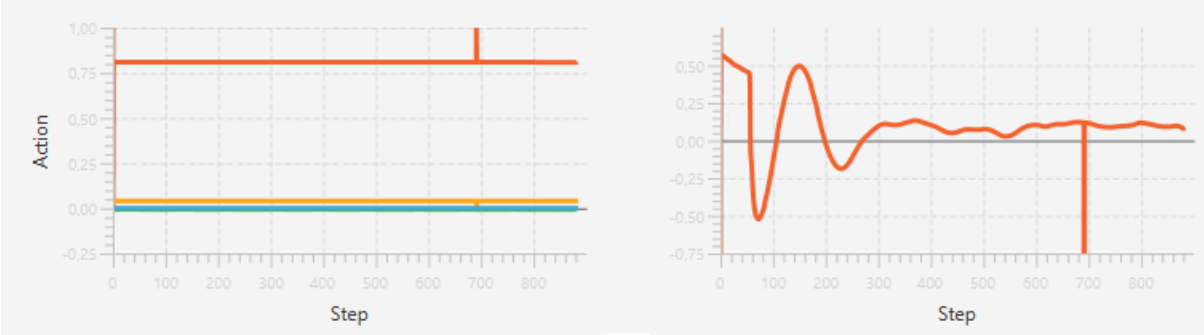


Ilustración 9.1 Gráficas de salida de los sistemas de control y V/S sin filtrar

- **La inclusión en el sistema de errores de medida:** El simulador provee al controlador de datos “perfectos”, es decir, cuando se lee la posición del simulador, el dato que se obtiene es la posición exacta que esta usando el simulador para sus cálculos. Esto simplifica el control, pero limita la validez de los resultados obtenidos. La inyección de ruido en los datos leídos del simulador podría solucionar esta limitación.

Capítulo 10. Referencias

AIRAC, 2018. *LEVC IAC 1- IAC Valencia RWY12.* [En línea]
Available at:
https://ais.enaire.es/AIP/AIPS/AMDT_329_2020_AIRAC_06_2020/AIP/aip/ad/ad2/LEVC_VALENCIA/LE_AD_2_LEVC_IAC_1_en.pdf
[Último acceso: Mayo 2020].

AIRAC, 2019. *AD 2-LEMD SID 12.3- SID Madrid/Adolfo Suaerz RWY36R.* [En línea]
Available at:
https://ais.enaire.es/AIP/AIPS/AMDT_327_2020_AIRAC_04_2020/AIP/aip/ad/ad2/LEMD_MADRID_A_S_Madrid_Barajas/LE_AD_2_LEMD_SID_12_en.pdf
[Último acceso: Mayo 2020].

Airliners.net, s.f. *Cessna 172 Skyhawk (early models) & 175 Skylark.* [En línea]
Available at: https://www.airliners.net/aircraft-data/cessna-172-skyhawk-early-models-175-skylark/140
[Último acceso: Junio 2020].

Cantoni, L., 2015. *X-Plane Interface (X-Pi) Github page.* [En línea]
Available at: https://github.com/luizcantoni/x-pi
[Último acceso: May 2020].

Cessna, s.f. *Cessna Skyhawk.* [En línea]
Available at: https://cessna.txtav.com/en/piston/cessna-skyhawk
[Último acceso: 2020].

Cleynen, O., 2013. *Cola de un Airbus A380, con indicación de la 'elevador' ("Stabilizer" medios estabilizador).* [Arte].

Delatorre, 2007. *Motor en un avión (A320 de Clickair).* [En línea]
Available at: https://commons.wikimedia.org/w/index.php?curid=3430243
[Último acceso: 2020].

EasyOffices, 2020. *EasyOffices.* [En línea]
Available at: https://www.easyoffices.com/es/oficinas/valencia/calle-de-las-barcas
[Último acceso: June 2020].

Emoscopes, 2005. *Funcionamiento de un motor turbohélice.* [En línea]
Available at: https://commons.wikimedia.org/w/index.php?curid=7975732
[Último acceso: 2020].

Enaire, 2019. *PROCEDIMIENTOS DE SALIDA RNAV1 (SID 12 - RWY 36R DIURNO RNAV 1).* [En línea]
[Último acceso: Mayo 2020].

EUROCONTROL, 2017. *USER MANUAL FOR THE BASE OF AIRCRAFT DATA (BADA).* REVISION 3.14 ed. s.l.:s.n.

Eurocontrol, 2020. *BADA FactSheet.* [En línea]
Available at: https://simulations.eurocontrol.int/wp-content/uploads/2018/01/BADA-factsheet_web.pdf
[Último acceso: Mayo 2020].

Markin, A., 2012. *Flight. Yak-52.* [Arte].

McCormack, A. S. & Godfrey, K. R., 1998. Rule-Based Autotuning Based on Frequency. *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY,* 6(1), pp. 43-61.

Nuic, A., Poles, D. & Mouillet, V., 2010. BADA: An advanced aircraft performance model for present and future. *INTERNATIONAL JOURNAL OF ADAPTIVE CONTROL AND SIGNAL PROCESSING.*

OACI,Organización de Aviación Civil Internacional, 2009. Anexo 4: Cartas aeronáuticas. En: *Convenio sobre Aviación Civil Internacional.* Quebec: s.n.

OACI, 2007. Gestión del tránsito aéreo. En: *Procedimientos para losservicios de navegación aérea.* s.l.:OACI.

TravTigerEE, s.f. *Diagrama de bloques de un controlador PID en un lazo realimentado.* [Arte].

USA Civil Air Patrol, 2018. *Forces of Flight.* [En línea]
Available at: https://www.cap-ny153.org/forcesweight.htm
[Último acceso: 2020].

Walter, R., 2014. Flight Management Systems. En: C. R. Spitzer, ed. *Digital Avionics Handbook.* s.l.:CRC Press.