UNIVERSIDADE DA CORUÑA

Escola Politécnica Superior, Ferrol

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA MECÁNICA

Título: **Desarrollo de un modelo de un vehículo aéreo no tripulado sobre simulador de código abierto**

Autor: **D. José Rubén Vieites Pardo**
Tutores: **D. Álvaro Deibe Díaz**
D. Félix Orjales Saavedra
Fecha: **Julio 2016**

AGRADECIMIENTOS

Quiero aprovechar estas líneas para agradecer a todas las personas que han influido positivamente en esta etapa de mi vida en la Escuela Politécnica Superior de Ferrol.

Gracias a mis padres, por todo lo que me han dado para poder llegar hasta aquí, al resto de mi familia por su preocupación y a mis amigos por su apoyo a lo largo de estos años.

También agradecer a mis tutores Félix y Álvaro, por darme la oportunidad de realizar este proyecto con ellos y por su ayuda durante su desarrollo. A Juan Carlos por aportar el modelo gráfico del UAV, y al resto de personas que hayan aportado algo a este proyecto.

Por último agradecer a mis compañeros de grado durante estos 5 años por hacer más agradable esta etapa.

Índice

Índice de tablas

Índice de ilustraciones

1. INTRODUCIÓN

1.1. UAV

El término *Unmanned Aerial Vehicle* o UAV, hace referencia a los vehículos aéreos propulsados que no llevan un operador humano a bordo, usan la fuerza aerodinámica para sustentarse, pueden volar autónomamente mediante un software pre-programado o ser pilotados de forma remota por un operador humano "fuera de su campo visual" mediante un enlace de vídeo. Y además realizar las tareas para las que ha sido diseñado sin intervención humana, o con intervención humana en determinadas acciones. Pueden ser reemplazables o reutilizables a diferencia de los misiles, y pueden transportar cargas peligrosas, o no peligrosas tanto en ámbito militar como civil, o incluso no transportar cargas en los casos en los que les ha sido encomendada otra tarea.

En español se les denomina Vehículos Aéreos No Tripulados (VANT) o Sistemas Aéreos No Tripulados (SANT), aunque son más conocidos en la sociedad por el nombre de "drones", nombre heredado de los vehículos militares a control remoto usados como blanco de tiro.

Los UAVs actuales están pasando de ser únicamente vehículos aéreos pilotados remotamente a robots cada vez más autónomos y versátiles, aunque este grado de autonomía es aún una cuestión que genera polémica, sobretodo para determinadas aplicaciones en las que pueden ponerse en riesgo vidas humanas.

1.1.1. Clasificación de los UAV

Se podrían hacer diferentes clasificaciones de los UAVs en función de su ámbito de uso, de su alcance, altitud máxima, según el tipo de ala, e incluso autonomía de vuelo. Aquí se ha escogido la clasificación en función de su peso como opción. En ella se pueden distinguir 7 categorías, en las que además se especifican la altitud normal operativa, el radio de operación, la autonomía de vuelo, la altitud y sus usos más comunes:

Categoría	Peso del UAV (kg)	Altitud Normal Operativa (m)	Radio de operación (km)	Autonomía de vuelo	Altitud	Usos más comunes
MICRO	< 2	Más de 60	5	Pocas horas	Muy baja	Reconocimiento, inspección, vigilancia
MINI	2 - 20	Más de 900	25	Más de 2 días	Baja	Vigilancia, recoger información
SMALL	20 - 150	Mas de 1.500	50	Más de 2 días	Baja	Vigilancia, recoger información
TACTICAL	150 - 600	Más de 3050	200	Más de 2 días	Baja	Vigilancia, recoger información
MALE	> 600	Más de 13.700	Ilimitado	Días o semanas	Media	Vigilancia, transporte de cargas
HALE	> 600	Más de 19.800	Ilimitado	Días o semanas	Alta	Vigilancia, recoger información, retransmisión de señal
STRIKE/ COMBAT	> 600	Más de 19.800	Ilimitado	Días o semanas	Alta	Vigilancia, recoger información, retransmisión de señal

Tabla 1.1: Clasificacion UAV

1.1.2. Historia de los UAV

Se puede considerar que los UAVs aparecen en el ámbito militar, aunque existieron creaciones previas de uso civil, pero de menor interés. Se empezaron a usar en el sector militar para realizar tareas "pesadas, sucias o peligrosas" (*dull, dirty or dangerous)* como misiones de ataque, reconocimiento y espionaje sobre objetivos enemigos, sin necesidad de arriesgar vidas humanas para ello, o para el entrenamiento con baterías antiaéreas. Los primeros datos que se tienen sobre el uso de plataformas aéreas no tripuladas datan del año 1894, cuando el ejército austríaco usó globos cargados con explosivos lanzados desde barcos para atacar territorios aprovechando las corrientes de viento, pero era un sistema poco fiable.

Posteriormente, durante la I Guerra Mundial se construyeron las primeras aeronaves no tripuladas, controladas por radiofrecuencia y que servían para guiar explosivos hacia el objetivo. En 1934 la empresa Reginald Denny Industries fue la primera en fabricar vehículos aéreos radiocontrolados en masa. A partir de ese momento, se empezaron a fabricar diversos modelos con fines principalmente militares.

Actualmente, los UAVs son utilizados en el ámbito militar para misiones como las citadas anteriormente, pero también se ha extendido su uso al ámbito civil, como en labores de vigilancia contra incendios, seguridad y control fronterizo, distribución de señal de Internet, cartografía, agricultura, cine, medio ambiente, transporte,....

Algunos de los UAVs más conocidos en el ámbito militar son el MQ-1 Predator usado para misiones de reconocimiento y ataque o el MQ-9 Reaper usado para tareas de vigilancia tanto en el ejército como por el servicio de aduanas y protección fronterizo de EEUU. Este modelo incluso ha sido utilizado por la NASA en su versión desarmada para misiones de carácter científico.

Los modelos más destacados construidos en España son el Milano, capaz de entrar en combate y el SIVA creado para misiones de vigilancia. Ambos fueron desarrollados por el Instituto Nacional de Técnica Aeroespacial (INTA).

Tras los primeros drones militares modernos, una pequeña compañía española desarrolló en 2010 el primer UAV con Certificado de Aeronavegabilidad Experimental, el FT-ALTEA, que incorporaba varios sensores, cámaras térmicas y de HD. Este modelo fue desarrollado por Flightech Systems, pero esta no es la única empresa española que se dedica al desarrollo de esta tecnología. Indra es otra de las grandes que invierte en este sector y ya cuenta con algunos modelos como el MRI y otros en vía de desarrollo, como el Mantis o el Pelicano. Unmanned Solutions S.L. también destaca por la creación de algunos modelos como el K2B y la familia de modelos Serie K.

La gran inversión que se está haciendo en los últimos años en España para investigar esta tecnología se debe a que los UAVs tienen un bajo coste, son rápidos y tienen un mundo de aplicaciones abierto a la imaginación. Tanto universidades como nuevas pymes están investigando y desarrollando esta industria en período de crecimiento y que promete llegar lejos. Claro ejemplo de ello son Dronair, con sede en Barcelona y que se dedica a la grabación y captura de imágenes desde el aire mediante el uso de UAVs para cine, televisión, publicidad o eventos; Cartogalicia, con sede en Santiago de Compostela y que usa tecnologías de vuelo autónomo para realizar proyectos topográficos; o Trabajoscondron S.L., una empresa especializada en el uso de drones para trabajos de captura de imágenes, grabación de vídeo, inspección de infraestructuras industriales para su mantenimiento como parques eólicos, estructuras, bosques, vigilancia de playas (proyecto en fase de pruebas), o publicidad. Incluso existen proyectos iniciados por alumnos de la UDC con los UAVs como protagonistas, como es el caso de DRONLIFE, un UAV con la capacidad de transportar órganos

para trasplantes, desarrollado por alumnas de la EUDI (Escuela Universitaria de Diseño Industrial) en colaboración con la IFFE Bussines School para un concurso de aplicaciones civiles de drones.

1.2. Simuladores

Por otra parte, un simulador es un software informático que permite reproducir virtualmente un sistema real. Los simuladores intentan reproducir de la forma más fiel posible tanto el entorno como el comportamiento del sistema real en el software.

El objetivo es que el usuario adquiera conocimiento a partir del trabajo exploratorio, la inferencia y el aprendizaje por descubrimiento. Los simuladores permiten realizar cambios y ver como varía el comportamiento del modelo simulado ante tales cambios. Si hiciéramos estas modificaciones en el sistema real supondrían en algunos casos un gran gasto económico, serían más laboriosos y serían mucho mas lentos a la hora de ser estudiados o probados. Gracias a los simuladores podemos permitirnos un estudio más detallado y preciso con una inversión menor y en un plazo de tiempo más breve. Además de poder realizar cambios sobre el modelo fácilmente, también podemos hacer cambios sobre el entorno en donde se simula el modelo, como cambios en las condiciones meteorológicas, entre otras cosas.

La simulación implica la creación de un modelo o sistema para la que se puede recurrir a varias técnicas. Se puede elaborar un modelo del equipo y virtualizarlo por hardware con el equipo real o se puede usar el mismo software que se usa en el equipo real pero usándolo en un ordenador convencional, con lo que se abaratan los costes.

A la hora de diseñar una simulación, podemos distinguir varias fases:

1. **Definición del sistema:** se trata de hacer un análisis del sistema con el fin de determinar la interacción del mismo con otros sistemas, sus restricciones, las variables que intervienen y las medidas de efectividad que se van a usar para definir y estudiar el sistema.
2. **Formulación del modelo:** consiste en definir todas las variables que forman parte de el, sus relaciones lógicas y los diagramas de flujo que describen el modelo.
3. **Preparación de los datos:** es la recolección de datos para su implementación en los parámetros de las variables del modelo.
4. **Traslación del modelo:** se define como la programación del modelo mediante algún lenguaje de programación o la utilización de algún simulador para procesarlo en el

ordenador y obtener los resultados.

5. **Validación:** consiste en encontrar las deficiencias del modelo o de los datos implementados y su corrección.
6. **Planificación estratégica:** se encarga del diseño de los experimentos que se van a simular y de determinar el número de simulaciones que se van a realizar.
7. **Planificación táctica:** consiste en determinar las condiciones en las que se ha de realizar cada versión del experimento. Estos factores se escogen según afecten más o menos al comportamiento de nuestro sistema.
8. **Experimentación:** ejecución de los experimentos planificados
9. **Interpretación:** estudio de los resultados obtenidos
10. **Documentación:** redacción del modelo y su traducción informática y los resultados obtenidos tras la simulación.

1.2.1. Tipos de simuladores

- Simuladores sociales: son software que intentan predecir el comportamiento de una sociedad o de un individuo cuando se produce un cambio que les afecta. Se incluyen en este grupo a los simuladores de vida, políticos, de entretenimiento,...

- Simuladores económico-financieros: son aquellos que tienen como objetivo predecir cuales van a ser los resultados de una operación económica o un cambio que pueda afectar a la economía, ya sea de una empresa en particular o de un estado. Podemos agrupar aquí a los simuladores de negocios, políticos, energéticos,...

- Simuladores de procesos: este es el grupo más generalista, porque agrupa simuladores de una gran variedad de sistemas, ya sean simuladores de procesos de fabricación, simuladores industriales, físicos, químicos, simuladores de redes, de ciberdefensa, logística,... intentan reproducir fielmente las condiciones reales de trabajo para obtener los resultados más realistas posibles, y aplicar posteriormente los métodos a la realidad para mejorar un proceso o encontrar fallos y corregirlos.

- Simuladores de aprendizaje: en en se incluyen todo tipo de simuladores de manejo de vehículos para autoescuelas, de barcos, de vuelo, musicales, médicos, etc., y su objetivo es únicamente el aprendizaje o entrenamiento por parte de los usuarios, antes de realizar estas tareas con los sistemas reales.

En este trabajo nos centraremos en los simuladores de vuelo, que están incluidos en el grupo de simuladores de aprendizaje, para el estudio del modelo de los parámetros del modelo del que partimos, y tomaremos datos que compararemos con el modelo real, y modificaremos para intentar mejorar el modelo.

1.2.2. Historia

La historia y la evolución de la simulación por ordenador han ido paralelas a la evolución de la informática. Sus orígenes los encontramos en la Segunda Guerra Mundial cuando los matemáticos J. V. Neuman y S. Ulam, tenían el reto de resolver un problema complejo relacionado con el comportamiento de los neutrones. Los experimentos basados en prueba y error eran muy caros y el problema era demasiado complicado para abordarlo mediante técnicas analíticas. La aproximación que cogieron se basa en la utilización de números aleatorios y distribuciones de probabilidad. El método utilizados fue llamado "método Montecarlo" por el paralelismo entre la generación de números aleatorios y el juego de la ruleta. Durante la Guerra Fría se intensificó el uso de la simulación para resolver problemas de interés militar, como trayectorias y dinámicas de satélites artificiales, guiar misiles,.... A partir de la década de los 60 empiezan a aparecer en el mercado programas de simulación de sistemas de acontecimientos discretos que poco a poco se empezaron a usar para resolver problemas de ámbito civil. En los años 80, la revolución que se produjo en la informática tuvo un gran impacto en la simulación por ordenador, y es durante estos años cuando se generaliza el uso de simuladores a prácticamente todos los ámbitos de la ciencia y la ingeniería mencionados anteriormente.

El ámbito de la ingeniería ha sido el que más se ha aprovechado del desarrollo de esta tecnología y de sus ventajas. Ha sido aplicada en muchos casos para el aprendizaje y entrenamiento de uso de equipos industriales y también en una gran variedad de temas relacionados con la investigación y el desarrollo, como procesos industriales, vehículos, o en nuestro caso, de vuelo.

Algunos de los simuladores de vuelo mas conocidos actualmente son el Micrisoft Flight Simulator, vendido como videojuego, el X-Plane, extremadamente preciso e incluso autorizado por la Administración Federal de Aviación de EEUU para el entrenamiento de pilotos, Rise of Flight, vendido como videojuego de vuelo de combate, o el FlightGear, que será el que utilizaremos para llevar a cabo el desarrollo de nuestro modelo.

En España, las principales empresas que realizan simuladores de vuelo son Indra y EADS. Indra

suministra productos de simulación de vuelo para aeronaves militares de combate (Eurofighter, EF-18, AV-8B,...) , transporte (A400M, C-130,...) y entrenadores (C-101,...) pero también de simulación de hidroaviones contra incendios (CL-415), siempre orientado al aprendizaje del piloto.

1.3. Breve descripción de nuestro UAV

Ilustración 1.1: UAV (1)

Ilustración 1.2: UAV (2)

En estas 2 imágenes podemos observar nuestro modelo de UAV, el Hercules XL 76", con el que trabajaremos a lo largo de este proyecto. A continuación se detallan algunas de sus características:

- Envergadura: 1,94 m (76 pulgadas, de ahí su nombre)
- Posee 2 winglets en las puntas de las alas
- 2 colas verticales que integran en ellas a las derivas. La función de las derivas es provocar giros alrededor del eje vertical
- 2 hélices que giran en sentido contrario. Visto desde atrás, la izquierda gira en sentido horario y la derecha en sentido antihorario.
- 4 elevones en su borde de salida. Son un híbrido entre los alerones y la profundidad, su función es la de realizar giros alrededor del eje longitudinal y provocar ascensos o descensos
- Fuselaje: es un cuerpo de forma geométrica irregular, como se puede observar en la imagen *UAV (1)*

Esto solo es una descripción muy breve y superficial del UAV que modelaremos, simplemente para entender como es geométricamente y algunas de sus características principales. Todos estos datos y otros sobre el UAV los iremos aclarando y explicando como han sido calculados a lo largo del proyecto, cuando surga la necesidad de aplicarlos a nuestro trabajo. Además también iremos explicando otros términos técnicos cuando sea necesario.

Hemos decidido hacerlo así, en lugar de explicar en profundidad todas las características del UAV al principio, porque creemos que de esta forma es más fácil de comprender lo que estamos haciendo en cada momento y se retiene mejor la información técnica.

2. OBJETIVOS

El presente proyecto se integra en la línea de investigación sobre UAVs del GII (Grupo Integrado de Ingeniería) de la UDC. El objetivo global de esta línea es el de desarrollar un enjambre de UAVs capaces de volar de forma autónoma y colaborar entre sí para realizar una tarea común (vigilancia forestal contra incendios, rescate de náufragos,...).

El primer paso para poder construir un grupo de UAVs como el que se pretende, es tener un UAV real, que será objeto de estudio, y con el cual se realizarán las pruebas necesarias para observar como se comportarán todos los UAVs idénticos a este. Para desarrollar el controlador de vuelo autónomo conviene hacer una simulación mediante software previamente. Esta simulación debe ser lo más próxima a la realidad posible, para predecir el comportamiento del modelo real, sin poner en riesgo su integridad. De esta forma se facilita y acelera el proceso de desarrollo del controlador de vuelo autónomo, y las estrategias de coordinación del enjambre de UAVs.

Este trabajo se centrará en la creación de un modelo de un UAV virtual, a partir de un modelo real del que disponemos en el laboratorio del GII, sobre un simulador de código abierto, el FlightGear, con el que podremos simular, cambiar parámetros de vuelo y realizar modificaciones de las características del modelo. También seleccionaremos un modelo de dinámica de vuelo (FDM) que determina el comportamiento físico de la aeronave para llevar a cabo dicha simulación.

Una de las condiciones impuestas para la realización de este trabajo, ha sido la de utilizar en la medida de lo posible software de código abierto,esta ha sido una de las causas principales que han impulsado el uso de FlightGear, junto con el FDM adecuado, el JSBSim para la realización de este proyecto. Cabe mencionar además que ha sido redactado usando OpenOffice, otro software de código abierto.

Por último analizaremos los resultados obtenidos en la simulación del vuelo del modelo y lo contrastaremos con el comportamiento del modelo real en vuelo real, para observar las diferencias existentes e implementar las correcciones en el controlador.

3. METODOLOGÍA

3.1. FlightGear

Ilustración 3.1: Logo de FlightGear

Como ya hemos mencionado anteriormente, para realizar la simulación de nuestra aeronave hemos optado por utilizar FlightGear.

FlightGear es un simulador de vuelo de código abierto. Originalmente fue desarrollado por un equipo de voluntarios anónimos y lanzado al mercado en 1997, ha ido evolucionando gracias a aportaciones de programadores a nivel mundial.

El objetivo de FlightGear era crear un simulador de vuelo de código abierto para contribuir al aprendizaje académico, el entrenamiento de pilotos, o como herramienta aplicada a la ingeniería, de forma que fuese accesible para cualquier persona, independientemente de sus límites económicos.

Uno de los objetivos de este proyecto era el de usar software de código abierto, FlightGear cumple esta condición. Esto significa que es un programa en el que cualquier persona puede acceder a su código y realizar en él cambios para aportar mejoras.

FlightGear está escrito principalmente en C++. Además, se tiene constancia de que funciona con diversos sistemas operativos como Windows, Linux, Mac OS-X, Solaris o IRIX, permitiendo a los usuarios usar la plataforma que cada uno prefiera, para realizar proyectos comerciales, de investigación, privados o por afición, sin que el factor económico suponga una limitación.

Pero FlightGear es solamente el software que ofrece la interfaz gráfica de la simulación, para la aeronave pueda volar siguiendo los principios de la física, es necesario un Modelo de Dinámica de Vuelo.

3.2. Selección de FDM

Un FDM (en español, Modelo de Dinámica de Vuelo) es esencialmente un modelo matemático-físico que define el movimiento y comportamiento de la aeronave bajo las fuerzas y momentos aplicados sobre ella y fuerzas naturales usando varios mecanismos de control.

3.2.1. Flight Dynamics Models (FDM)

Para FlightGear, tenemos cinco FDM compatibles:

1. ***LaRCsim***

   Originalmente fue desarrollado por la NASA. Fue el primer FDM compatible con FlightGear, y el que se usaba por defecto hasta el año 2000. Actualmente ha sido sustiutido por otros dos FDM más avanzados, el YASim y el JSBSim y ha servido de base para la aparición de otro FDM, el UIUC, por lo que se encuentra en desuso.

   Sus principales funciones son:

   - *ls_accel()*: sirve para sumar fuerzas y momentos generados por la aerodinámica y el motor y las posiciona en el centro de masas para calcular las aceleraciones resultantes.
   - *ls_step()*: integra las ecuaciones de las aceleraciones para obtener las velocidades y las ecuaciones de velocidades para obtener la posición de la aeronave.
   - *ls_aux()*: conociendo las velocidades y posiciones calcula otros parámetros de vuelo como ángulos de ataque (alpha) y deslizamiento lateral (beta), presiones, temperaturas, etc.
   - *atmos_62()*: búsquedas de tabla de atmósfera estándar de 1962.
   - *ls_geoc_to_geod (lat_geoc, radius, lat_geod, alt, sea_level_r)* y *ls_geod_to_geoc (lat_geod, alt, sl_radius, lat_geoc)* para calcular posiciones geocéntricas y geodésicas.
   - *ls_gravity(SCALAR radius, SCALAR lat, SCALAR *gravity)* recrea la gravedad local de LaRCsim en función de la latitud y la altitud.

2. ***UIUC***

   Este FDM nació en el UIUC Applied Aerodynamics Group de la Universidad de Illinois en Estados Unidos, basándose en el modelo anterior, LaRCsim. Además incluía la opción de simulación en condiciones de formación de hielo sobre la aeronave. Al igual que el

LaRCsim se encuentra inactivo.

3. YASim

Es un modelo algoritmo matemático y de información que determina el comportamiento físico del vuelo dentro del simulador. Su autor, Andy Ross lo tituló “Yet Another Simulator”, de ahí su nombre.

YASim usa la geometría de la aeronave para generar las características básicas de vuelo. Por ejemplo, si introducimos las dimensiones de sus alas, fuselaje y sistema de propulsión, YASim usa esta información para generar un modelo matemático que describe el comportamiento físico de vuelo. Esto nos puede dar una simulación aproximada, pero para conseguir un comportamiento más parecido a la realidad, necesitamos ajustar muchos parámetros más, y es necesario invertir tiempo investigando y un mayor esfuerzo. Este FDM realiza una simulación a partir del efecto de circulación del aire en distintas partes del avión.

Además es capaz de generar simulaciones de aeronaves de la fija o ala rotatoria.

Como simulador de vuelo, YASim cuenta con algunas limitaciones:

- Originalmente fue desarrollado para simulación de vuelos subsónicos. En caso que que el usuario no necesite demasiado realismo, YASim puede simular vuelos transónicos, pero la precisión de la simulación se vuelve muy baja al tratar de simular vuelos supersónicos. La simulación de vuelos transónicos se consigue gracias a que muchas aeronaves modernas poseen características de diseño que minimizan los efectos transónicos, de forma que los pilotos no los noten. De igual modo, podemos aplicar estas características a nuestra simulación para que no se vea afectada. En nuestro caso, esto no tiene mucha importancia, ya que los vuelos que va a realizar nuestro UAV serán a velocidades muy inferiores.
- YASim, no permite simular un vuelo próximo a la realidad de aeronaves que carezcan de fuselaje, es decir, aeronaves que solo están formadas por un ala (como un ala delta). Necesita un estabilizador horizontal para compensar los momentos generados por el cabeceo de la aeronave durante el vuelo. En una aeronave real que carezca de fuselaje, este cabeceo puede compensarse con la curvatura adecuada en la parte trasera del ala. Para hacer esto con YASim de una forma lo más aproximada posible podemos introducir un estabilizador horizontal cerca de la superficie del ala, pero no obtendremos una buena simulación.

- Actualmente, con YASim no es posible cambiar el ángulo de incidencia del aire sobre el ala durante el vuelo, no se puede simular un movimiento de la superficie alar o una parte de ella como puede ser un estabilizador.
- Muchos características de motores no están bien simuladas. Por ejemplo, en motores de pistones, el rendimiento es realista, pero los resultados de consumos, presiones manométricas y temperatura del motor, se podrían mejorar. Los motores a reacción en cambio, poseen una simulación bastante mejor porque es más simple. Las turbohélices son también bastante rudimentarias. Por último, las hélices todavía no han sido implementadas en este FDM.
- YASim no es un FDM lo suficientemente avanzado para simular ciertos efectos del vuelo sobre superficies de la aeronave, como por ejemplo, las ventajas de añadir una pequeña ala vertical en la punta del ala *(winglet)*, que lo que hace en aeronaves reales es reducir los efectos de los "torbellinos de punta de ala" *(wingtip vortices)* que se producen. Esto no quiere decir que YASim no pueda simular estos efectos, simplemente que no los puede simular automáticamente, deben ser programados.
- Algunas carencias más de este FDM son:
  - El arrastre inducido tiene un "bug".
  - La sustentación producida por el ala y los cálculos de momentos producidos, no son completos, se desprecian algunos de ellos.
  - Es difícil obtener un comportamiento exacto de cabeceo de la aeronave cuando se despliegan los flaps.
  - Durante el despliegue de los flaps, las fuerzas producidas sobre estos, y el arrastre que producen, no pueden ser interpolados para cada punto durante el movimiento de despliegue.

    La mayoría de estos fallos, pueden ser corregidos con algunos ajustes y mucha paciencia.

4. JSBSim

Fue concebido en 1996 como un FDM sencillo, basado en datos, no lineal, con 6 grados de libertad (6DoF), con objeto de simular la dinámica y control de vuelo de aeronaves.

Desde sus primeras versiones, JSBSim se ha beneficiado del hecho de ser un software con

entorno de código abierto para mejorar sus prestaciones y crecer gracias a las contribuciones e ideas de la gran variedad de usuarios de todo el mundo.

Es un código escrito en C++, incluyendo algunas rutinas del lenguaje C, usa ficheros de configuración XML, además de ser usado en FlightGear, también es usado en OpenEagles. Podemos configurar totalmente el modelo de vuelo desde ficheros XML: aerodinámica, propulsión, simulación de los efectos de rotación de la tierra, etc.

JSBSim contiene varios modelos físicos que trabajan conjuntamente para calcular la dinámica de la aeronave general.

- Equilibrio de masas: usa la masa para generar una fuerza de gravedad y momentos de inercia para amortiguar los cambios en la velocidad de rotación.
- Reacciones en tierra: cuando la aeronave está en contacto con la tierra se generan reacciones en el punto de contacto, este modelo consigue calcular estas reacciones. Si el contacto no está en el centro de gravedad, este proporciona también un momento de rotación.
- Propulsión: puede modelar una amplia variedad de motores usando diferentes métodos. Todos los motores ofrecen una fuerza a lo largo del eje X, y crean momentos si esta fuerza resultante no es aplicada en el centro de masas. La hélice y los rotores de propulsión también crean momentos de giro.
- Aerodinámica: aquí partimos de cero para crear el modelo, aunque existen plantillas en la web del programa. Es responsabilidad exclusiva del autor de la aeronave usar fórmulas válidas. La fuerzas aerodinámicas se dividen en sustentación, resistencia, y fuerzas laterales. También se pueden crear momentos de alabeo, cabeceo y guiñada. Además, si el punto de referencia de la fuerza aerodinámica no es coincidente con el centro de masas las fuerzas crearan un momento.
- “Fuerzas de flotación”: corresponde a aeronaves “más ligeras que el aire”, como globos aerostáticos, dirigibles presurizados y dirigibles rígidos.
- Fuerzas externas: Si la sección de aerodinámica es demasiado limitada o se desea simular las cuerdas de remolque de planeadores, catapultas, etc., se pueden agregar fuerzas en puntos arbitrarios.
- Atmósfera: se puede configurar un modelo de atmósfera según se desee, o implementar un modelo concreto externo. Además se pueden simular tres tipos de turbulencia diferentes.

- Gravedad.

5. ***MATLAB***

   Además de los 4 FDM anteriores, también se pueden crear modelos en MATLAB, o modelos para aeronaves específicas, como pueden ser las "más ligeras que el aire" (como globos, etc.).

   Las principales desventajas de MATLAB es que no es un software de código abierto y sería muy laborioso desarrollar un modelo completo en él.

Ahora que tenemos una idea de las capacidades y carencias de cada uno, podemos seleccionar nuestro FDM. Es evidente que los que tienen un desarrollo más activo son YASim y JSBSim, por lo que haremos una comparación entre estos dos para decidir razonadamente cual usaremos.

YASim, a pesar de ser su pobre documentación, es más accesible e intuitivo, en cambio, JSBSim es un FDM más rico en documentación, tiene un entorno de desarrollo más flexible y se necesita un mayor conocimiento en el área de aerodinámica para crear un modelo. Esto hace que JSBSim sea la dinámica de vuelo que mejor puede simular la aeronave, de forma más realista, a pesar de ser necesario un mayor trabajo para su realización. Además, algunas de las carencias de YASim importantes para nuestro caso, como el hecho de que no sea capaz de simular correctamente aeronaves sin fuselaje (sin añadir estabilizadores), no nos conviene, y otras de estas carencias no están presentes en JSBSim, por lo tanto, consideramos que es un modelo más avanzado. Teniendo esto en cuenta, es obvio que el FDM que se utilizará en este proyecto será JSBSim.

	YASim	**JSBSim**
Documentación	Poca documentación	Mucha documentación
Entorno de desarrollo	Más accesible e intuitivo	Más flexible
Nivel de precisión de la información	Medianamente preciso (el nivel de exigencia de datos es menor)	Muy preciso (se exigen un mayor número de datos y mayor conocimiento en aerodinámica)
Capaz de simular aeronaves sin fuselaje?	Si, pero solo si añadimos estabilizadores	Si, sin necesidad de ajustes

Tabla 3.1: Comparación YASim vs JSBSim

3.3. Modelado

Nuestra misión en este trabajo a partir de ahora es determinar las características físicas del UAV real que tenemos, el HERCULES XL 72”/78”, tomando medidas directamente sobre el, utilizando las herramientas de las que disponemos.

Con estos datos, podremos crear el modelo virtual escribiendo los archivos .XML necesarios para que el FDM realice los cálculos que determinarán el comportamiento de la aeronave en el simulador.

Dividiremos el modelado en las siguientes partes del UAV:

- Hélices
- Motores
- Aerodinámica
- Tren de aterrizaje e inercias

3.3.1. Hélices

Para modelar las hélices, tomamos las medidas directamente de una hélice de pruebas de la que disponemos en el laboratorio del GII.

Ilustración 3.2: Hélice de pruebas

En primer lugar, debemos disponer un software que sea capaz de modelizar nuestra hélice a partir de unos parámetros de entrada que le introduciremos. Como siempre, debe cumplir la condición de software libre y existen multitud de programas que cumplen esta condición, pero además, debemos elegir uno que sea capaz de crear archivos .XML o similares pero fáciles de modificar para convertir a .XML para que JSBSim pueda trabajar con ellos y crear el modelo. El que nos recomienda la propia página web de JSBSim es el Aeromatic, pero se trata de un software muy

simple, que recrea una hélice que puede tener mayores diferencias con la realidad. En su lugar hemos encontrado otro de mayor precisión con el que podemos asemejar mejor nuestro modelo a la hélice real, se trata del JavaProp, a continuación se muestra una comparación de las características que nos permite modificar cada programa, en ella queda bastante claro el por qué de nuestra elección:

Datos de entrada	Aeromatic	Javaprop
Potencia del motor	Si	Si
Diámetro	Si	Si
RPM máxima	Si	Si
Incinación de hélice	Si	Si
Consideración de spinner	No	Si
Nº de palas	No (en función de la potencia)	Si
Velocidad del UAV	No	Si
Ángulo de hélice	No	Si (por secciones)
Perfil de hélice	No	Si (por secciones)
Cuerda	No	Si (por secciones)

Tabla 3.2: Comparación Aeromatic vs JavaProp

Como podemos observar, JavaProp nos permite realizar un modelo más detallado, y por lo tanto más preciso de nuestra hélice.

El siguiente paso es determinar las características de nuestra hélice para introducirlas en el programa:

- Número de aspas: 2
- Peso: 18g.
- Longitud: 9 pulgadas (0,2286 m)
- Paso: 6 pulgadas (el UAV avanza 0,1524 m por vuelta de la hélice a RPM máxima, de forma ideal)
- Cuerda: Pese a que JavaProp no solicita inicialmente las medidas de cuerda para realizar sus cálculos, si que incluye la opción de importar una tabla de datos que contenga las medidas de cuerda y ángulo de ataque en la pestaña de “geometría”. Nosotros, para tener un modelo

más exacto hemos hecho esto, y para ello hemos dividido la superficie longitudinalmente a cada centímetro obteniendo 11 secciones, y mediante un calibre, hemos tomado las medidas para cada sección, obteniendo los siguientes resultados:

Posición (cm)	Posición (r/R)	Cuerda (mm)
1	0,0875	16,4
2	0,1750	18,5
3	0,2625	23,3
4	0,3500	25,6
5	0,4374	25,5
6	0,5249	24
7	0,6124	21,8
8	0,6999	18,8
9	0,3937	15,4
10	0,8749	12
11	0,9624	9

Tabla 3.3: Cuerda en cada sección de la hélice

En donde "r" es la posición en donde tomamos la medida y "R" el radio de la hélice (11,43 cm).

Ilustración 3.3: Secciones de la hélice

- Ángulo de hélice: Para medir el ángulo de hélice hemos usado un molde de plastilina. Primero situamos la hélice sobre el bloque de plastilina y presionamos verticalmente hacia abajo para plasmar la forma de la hélice sobre el bloque. Además, se ha ido comprobando en cada momento que la forma quedaba plasmada correctamente colocando una tabla sobre la hélice y midiendo con un nivel si existía alguna inclinación de la hélice, para no obtener resultados de ángulos mayores o menores que el real, y obtener el resultado más preciso posible. Posteriormente se mide el ángulo de ataque colocando una varilla delgada que va desde el borde de ataque hasta el borde de salida de la cavidad dejada en el molde por la hélice y medimos el ángulo con un transportador de ángulos, comprobando siempre su horizontalidad, y repetimos este proceso en las 11 secciones.

En las siguientes imágenes podemos ver el procedimiento seguido:

Ilustración 3.4: Molde de la hélice

Ilustración 3.5: Procedimiento de medida del ángulo de hélice

Lo que medimos aquí, estrictamente hablando, es el ángulo de la hélice (que es lo que nos pide el programa) en cada punto, el ángulo de ataque se debería calcular en función de la velocidad del viento, ya que se define como "el ángulo que forman la cuerda geométrica de

un perfil alar con la dirección del viento incidente", y aquí estamos midiendo el ángulo en condiciones de reposo.

En la siguiente imagen se ilustra perfectamente la diferencia:

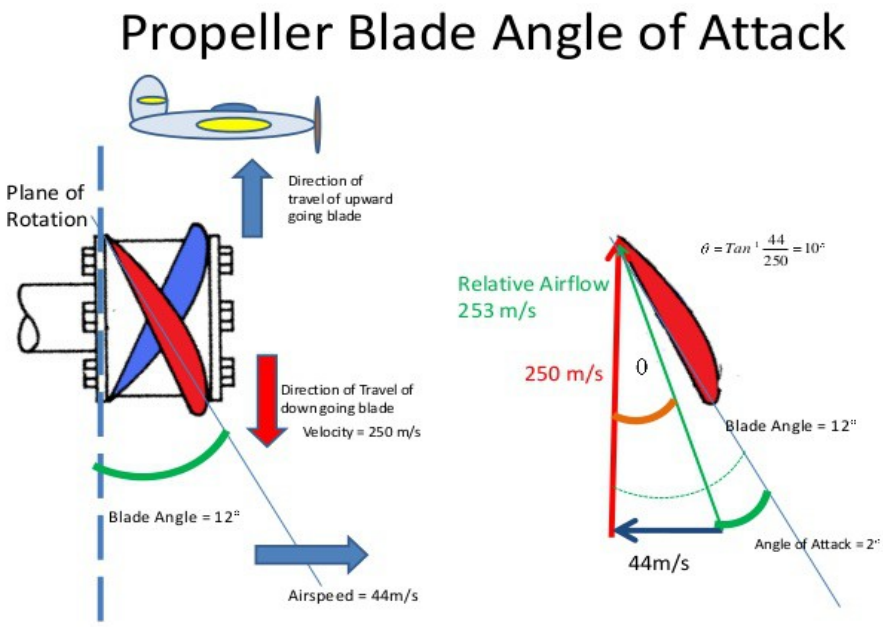

Ilustración 3.6: Diferencia entre ángulo de ataque y ángulo de hélice

Inicialmente, JavaProp solo nos solicita los ángulos de ataque de 4 posiciones de la hélice, pero se le pueden introducir más datos para mejorar la geometría generada importando una tabla que contenga los datos de cuerda y ángulo de ataque, que es lo que hemos hecho nosotros. En la siguiente tabla, podemos ver los ángulos medidos en función de la posición:

Posición (mm)	Cuerda (mm)	Angulo de ataque (º)
10	16.4	32
20	18.5	43
24	21,6	47
30	23.3	35
40	25.6	28
50	25.5	24
60	24	20
70	21.8	18
80	18.8	15
90	15.4	12
100	12	10
110	9	7
113	1	5

Tabla 3.4: Ángulo de ataque y cuerda en función de la posición

Además, como se puede ver en la tabla, hemos añadido un dato más en la posición 24 mm, porque es donde se produce un cambio mayor de ángulo.

- Velocidad máxima de rotación: 10.000 rpm. Medida directamente, una vez montada sobre el UAV, con un tacómetro.
- Velocidad de vuelo del UAV: Esta medida fue tomada en pruebas de vuelo previas, obteniendo una velocidad media de vuelo de 12,5 m/s.
- Diámetro del "spinner": 20 mm. El "spinner" es el cono que hay justo en el centro de la hélice, sobre el que está el eje de giro. En nuestro caso no existe tal cono, pero si una superficie cilíndrica, que es menos eficiente aerodinámicamente, pero que hay que considerar.
- Potencia del motor: 345 W. Dato proporcionado por el fabricante.
- Perfiles: JavaProp, nos solicita el perfil de 4 secciones de la hélice, los perfiles se han determinado teniendo en cuenta el espesor de la pala en cada uno de las 4 secciones que solicita y la forma geométrica:

Posición (r/R)	Perfil
0,000	MH 126, Re=500.000
0,333	MH 114 13%, Re=500.000
0,667	Clark Y, Re=100.000
1,000	MH 116 9,8%, Re=500.000

Tabla 3.5: Perfiles de las secciones

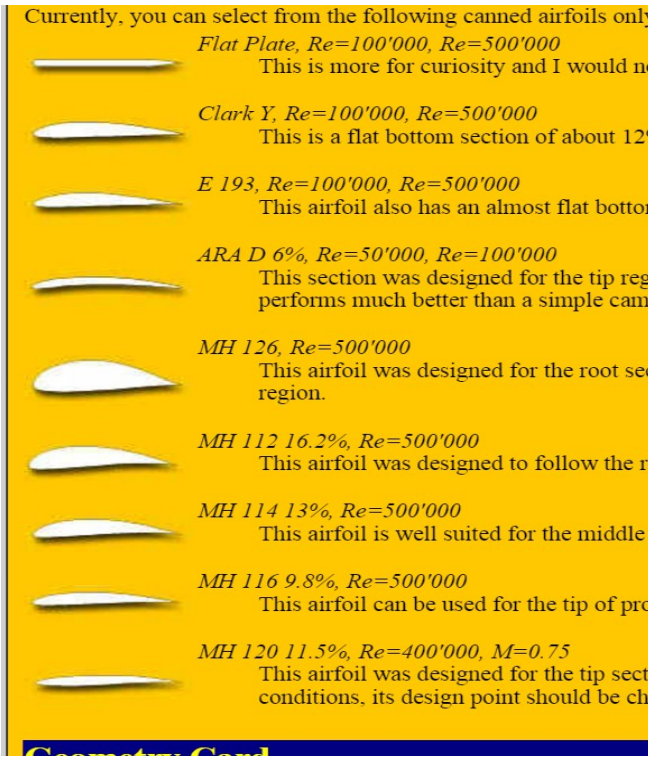

Ilustración 3.7: Lista de perfiles disponibles

Introduciendo todos estos datos en JavaProp, hemos generado un archivo que contiene los datos del diseño principal en formato .XML, otro archivo en formato .IGES para poder exportar el modelo de la hélice a Solid Edge (versión estudiante) y comparar con la real, y otro archivo .HTML que contiene todos los datos de comportamiento del modelo.

A partir de este último archivo, cogeremos los datos necesarios para los cálculos en el Modelo de Dinámica de Vuelo, JSBSim, y lo convertiremos a .XML. Para ello, primero necesitamos generar un modelo de hélice como el que genera Aeromatic, ya que contiene el formato de archivo con el que trabaja JSBSim y posteriormente introducimos los datos calculados por JavaProp en este archivo para conservar el formato pero teniendo una información más detallada y precisa.

Además, necesitamos conocer otros parámetros de la hélice que necesita el JSBSim para recrear el comportamiento del modelo:

- Momento de inercia (Ixx): lo podemos calcular utilizando la fórmula:

  $$I_{xx}=m\cdot k^2$$

  En donde:

  m: masa de la hélice
  k: radio de giro (como area)
  Obteniendo una inercia de:

  $$\mathbf{I_{xx}} = 0{,}018\cdot 0{,}1143^2 = \mathbf{0{,}00023516\ kg\cdot m^2}$$

  Durante el giro de las hélices, se produce una fuerza de inercia proporcional a la masa de las palas de las hélices y al cuadrado de la distancia al centro de masas de la misma. Teniendo esta información presente, sabemos que en caso de que el UAV se impulsara con una sola hélice, o en caso de que tuviera varias pero todas girasen en el mismo sentido, las fuerzas de inercia provocarían un giro del UAV hacia un lado (este giro se conoce como "yaw" o guiñada). En nuestro caso, gracias a que las hélices giran una en sentido horario y otra en sentido antihorario, se contrarrestan entre si los efectos de inercia.

- P-factor: valor 0. Es un dato que no aparece documentado en el JSBSim, pero sin embargo, si lo solicita para completar el archivo de datos de la hélice. Buscando información, hemos determinado que se trata de un coeficiente que sirve para considerar un fenómeno aerodinámico que se produce durante la rotación de las hélices, conocido como efecto de asimetría de pala de hélice. Es el responsable de la localización asimétrica del centro de la

fuerza de empuje cuando la aeronave está en una posición de alto ángulo de ataque. Esta desviación del centro de empuje es responsable del giro de la aeronave durante el vuelo hacia uno de los lados (guiñada).

En nuestro UAV debido a los sentidos de giro contrarios, reducimos el efecto del P-factor, a un valor inapreciable, por ello podemos considerar que es 0.

Con estos datos ya tenemos nuestro modelo de hélice con el que se podrá realizar la simulación.

A continuación se muestran dos gráficas en las que se visualizan los valores de los coeficientes de potencia (Cp) y de empuje (Ct), obtenidos con Aeromatic y con JavaProp, para observar la diferencia entre los resultados que proporcionan ambos programas y justificar nuestra elección.

- *Coeficiente de empuje (Ct)*

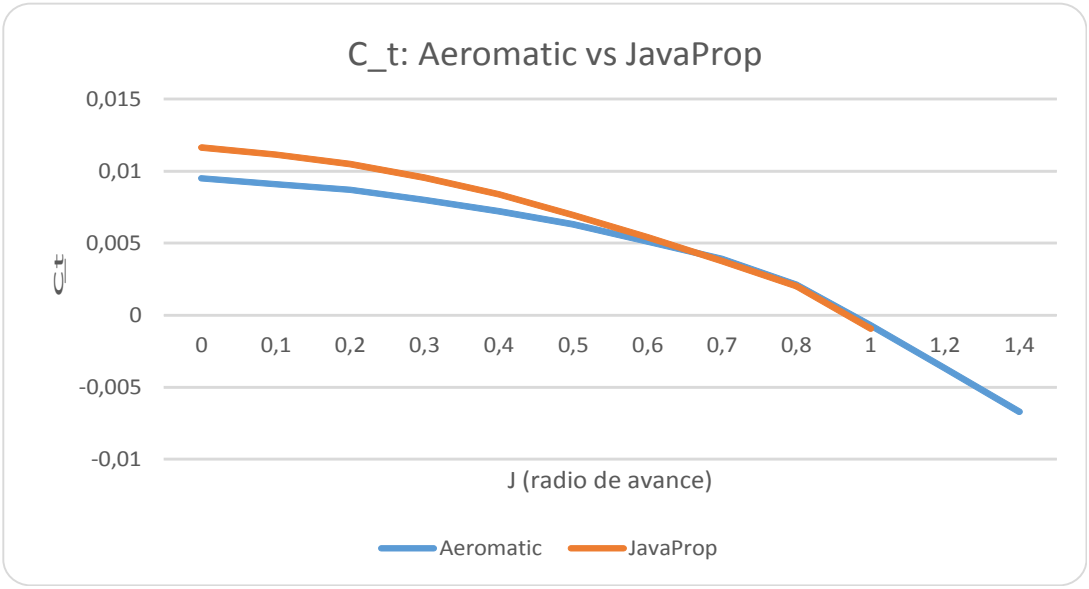

Ilustración 3.8: Comparación del coeficiente de empuje

- *Coeficiente de potencia (Cp)*

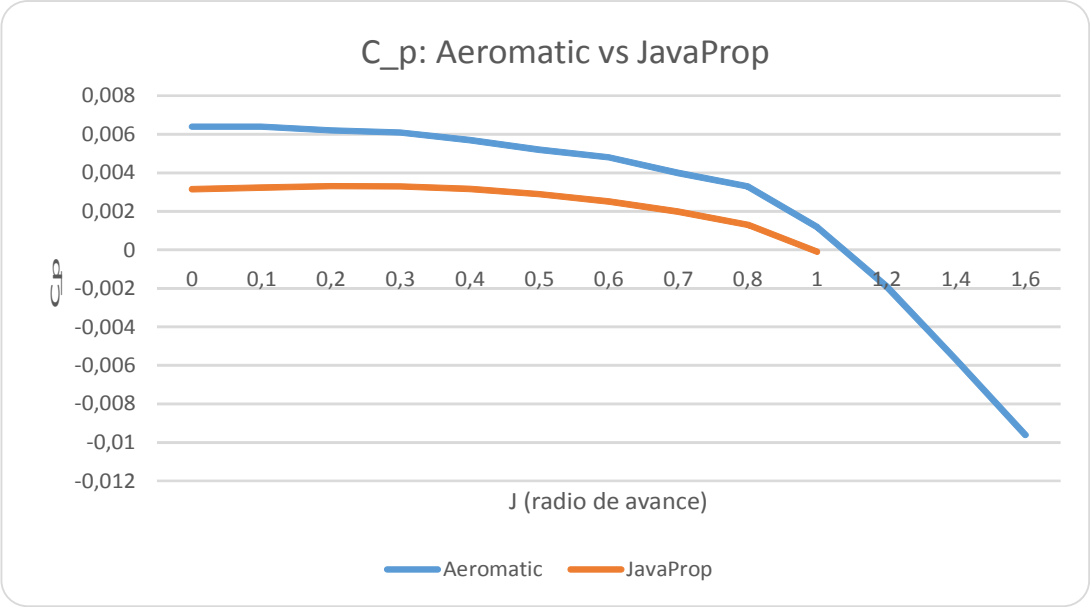

Ilustración 3.9: Comparación del coeficiente de potencia

Podemos apreciar que para el coeficiente de empuje (C_t), JavaProp nos proporciona unos resultados aproximadamente un 20% mayores que Aeromatic al principio, aunque luego esa diferencia se va reduciendo progresivamente hasta igualarse los valores en un radio de avance (J) de 0,65, a partir de ahí se mantiene ligeramente por debajo. Para el coeficiente de potencia (C_p), los resultados de JavaProp son siempre la mitad de los proporcionados por Aeromatic. Dados los resultados, debemos suponer que los del JavaProp son más precisos, porque utiliza una mayor cantidad de información para generar el perfil.

El archivo resultante se puede consultar en el ANEXO I: “ Fichero de hélices”.

3.3.2. Motores

Los motores que utiliza nuestro UAV son como los que se muestran en la imagen de abajo. Usa únicamente 2 motores, uno a cada lado.

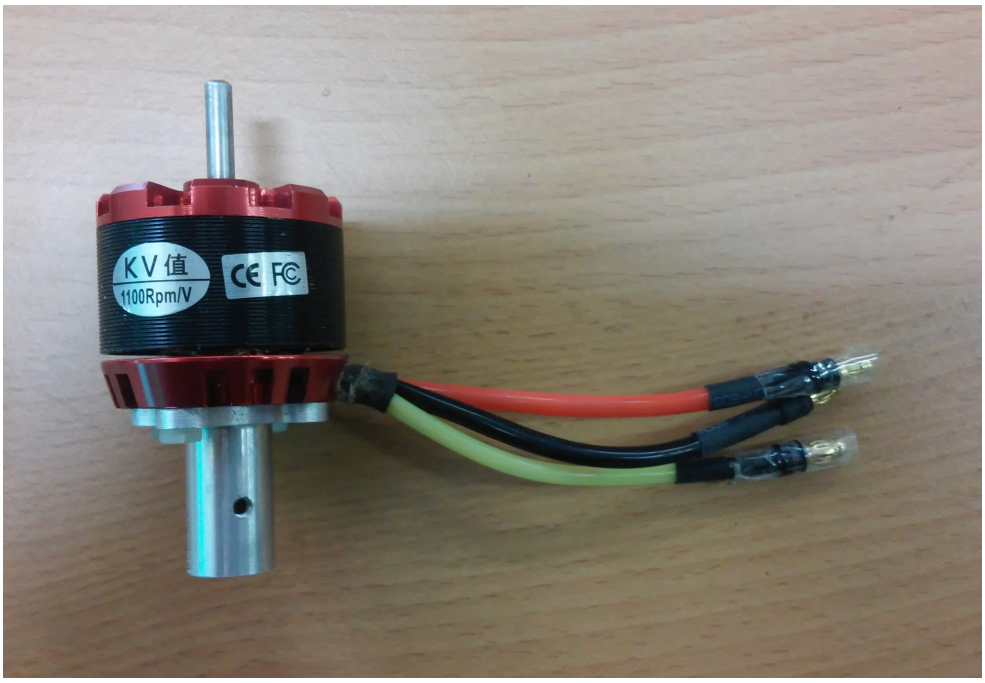

Ilustración 3.10: Motor

Los datos del motor ya nos los proporciona el fabricante, por lo que no es necesario hacer ninguna medida experimental.

JSBSim solo solicita el nombre y la potencia del motor, en el caso de motores eléctricos como el nuestro, por lo que podemos decir que esta es la parte más facil de modelar, y el archivo .XML es el más sencillo de los cuatro. Los datos introducidos en el archivo son:

- Modelo: "n3530".
- Potencia: 345 W.

Finalmente, cabe destacar que FlightGear trabaja con un modelo de motor basado en la potencia que se le especifica. La regulación de la potencia entregada por el motor se hace desde 0 hasta el valor especificado y varia de forma lineal. Internamente, se convierte esta potencia en caballos de fuerza que son aplicados sobre el archivo de las hélices para proporcionar un par y hacer que se muevan. Además, no consideran la existencia de baterías, por lo que es como si no consumieran energía alguna. Tampoco considera la fricción interna del motor.

El archivo resultante se puede consultar en el ANEXO II: "Fichero de motores".

3.3.3. Aerodinámica

El UAV que pretendemos modelar es como el que se puede observar en la imagen.

Ilustración 3.11: Unidad de UAV

Pero además, disponemos de otros 2 iguales a este en el laboratorio, y algunos más diferentes, que conforman el equipo de UAVs de la línea de investigación sobre UAVs del Grupo Integrado de Ingeniería.

Ilustración 3.12: Equipo de UAVs

3.3.3.1. Selección del método para el análisis aerodinámico

Para realizar el modelado debemos analizar las diferentes opciones de software u opciones experimentales que nos servirán para obtener información del UAV. Haremos esta selección de forma razonada, analizando las ventajas e inconvenientes de cada posibilidad, así como sus capacidades y limitaciones, como ya hicimos cuando seleccionamos el software para el modelado de las hélices. Además, debemos recordar uno de los objetivos de este proyecto, el de usar en la medida de lo posible software de código abierto.

Investigando, se han encontrado varios métodos de los que nos podemos servir para obtener la información necesaria:

- DATCOM
- HolyCows
- AVL
- Túnel de viento

Digital DATCOM

El "United States Air Force Stability and Control **Digital DAT**a **COM**pendium" es un programa informático escrito en FORTRAN que implementa los métodos contenidos en el "USAF Stability and Control DATCOM", un libro de más de 3100 páginas de cálculo de estabilidad aerodinámica y métodos de predicción de control para el diseño de aeronaves, redactado por la compañía de aviación estadounidense McDonnell Douglas Corporation en colaboración con los ingenieros del Laboratorio de Dinámica de Vuelo de la Base de la Fuerza Aérea Wright-Patterson.

Este software sirve para calcular la estabilidad estática y dinámica y el control de aeronaves de ala fija. DATCOM requiere un archivo de entrada que contenga la descripción geométrica de la aeronave, y da como datos de salida los coeficientes adimensionales correspondientes en las condiciones de vuelo especificadas.

Algunas de las características principales de los parámetros de entrada que permite introducir son:

- *Condiciones y opciones de vuelo:* más de 400 combinaciones de Match-altitud con más de 20 ángulos de ataque para cada combinación. Vuelos subsónicos o supersónicos, área alar, cuerda aerodinámica principal, envergadura y rugosidad de la superficie.
- *Parámetros de síntesis:* posición del centro de gravedad, vértices de alas, viento, posiciones de cola horizontal y vertical. El origen del sistema de coordenadas no tiene por qué ser

necesariamente la “nariz” de la aeronave, puede serlo cualquier punto, pero todas las dimensiones deben ser referenciadas desde este punto. Además se pueden añadir ángulos de incidencia diferentes del viento y la cola horizontal.

- *Parámetros de cuerpo:* define la geometría del cuerpo, el programa asume una geometría simétrica. Se pueden colocar más de 20 posiciones para definir la forma de la aeronave para una mitad (la otra mitad la supone simétrica).
- *Parámetros de las colas horizontal y vertical:* parámetros como la cuerda de la raíz de cada cola, envergadura media, forma de los perfiles, ángulo de diedro, flecha de las alas (de la cola), giro de las alas. También se pueden introducir perfiles variables a lo largo de la envergadura.
- *Dispositivos de control y sustentación:* se pueden definir flaps (varios tipos), alerones y elevadores. Pero no se puede definir un dispositivo de profundidad o “timón”.

Como parámetros de salida, DATCOM nos puede dar una gran cantidad de información en comparación con la escasa información de entrada que requiere:

- CL: coeficiente de sustentación
- CD: coeficiente de arrastre
- Cm: coeficiente de momento de cabeceo
- CN: coeficiente de fuerza normal
- CA: coeficiente de fuerza axial
- Clα: coeficiente sustentación en función del ángulo de ataque
- Cmα: Coeficiente de momento de cabeceo en función del ángulo de ataque
- CYβ: Coeficiente de la fuerza lateral en función del ángulo de deslizamiento lateral
- Cnβ: Coeficiente del momento de guiñada en función del ángulo de deslizamiento lateral
- Clβ: Coeficiente del momento de alabeo en función del ángulo de deslizamiento lateral.

Más adelante hablaremos de los que coeficientes necesitamos que nos proporcione el programa para implementar en JSBSim y se explicará brevemente cada uno de ellos.

Por otra parte DATCOM también tiene sus limitaciones. Algunas de ellas son:

- No permite colocar los motores de la aeronave, habitáculos externos u otras protuberancias

debido a que analiza el fuselaje como un cuerpo en revolución. Esta simplificación afecta al coeficiente de arrastre.

- No proporciona los resultados de coeficientes derivativos cuando la aeronave modelada no tiene las alas de forma “cónica”, es decir, más estrechas en su punta que en su raíz, o cuando incluyen extensiones o partes móviles en el borde de ataque de las alas (como pueden ser los slats). Esta carencia puede ser sustituida por información experimental sobre el cuerpo alar.
- No es posible añadir dos colas verticales sobre el cuerpo del fuselaje, pero se puede aproximar y crear una sola cola vertical equivalente a las dos colas.
- No proporciona coeficientes derivativos acerca de la superficie de control de profundidad. De acuerdo con el manual, no existe forma de introducir parámetros de entrada que definan la geometría de la profundidad.
- No es capaz de analizar 3 superficies de sustentación al mismo tiempo. Este problema lo podemos solucionar superponiendo superficies, usando datos experimentales.
- El fichero de datos de salida es un archivo de texto difícil de leer, y muy engorroso para el usuario.

HolyCows

HolyCows, Inc. es una compañía estadounidense que, básicamente lo que hizo fue adoptar DATCOM e introducir mejoras en el código. Creó dos versiones, una libre "DATCOM +" y otra de pago "DATCOM + PRO". La documentación encontrada en la web de la compañía hace referencia únicamente a la versión de pago, por lo que solo explicaremos las características de esta versión.

DATCOM +, nace en 1996, y es una extensión de DATCOM que incorpora algunas herramientas para facilitar su uso.

DATCOM + PRO, es la siguiente generación del programa, hecha de forma que aún se facilita más el uso del programa por parte del usuario. Podemos visualizar rápidamente la aeronave, y la información (coeficientes) generados en gráficas bidimensionales, para poderlos interpretar más rápidamente. Además, podemos ejecutar el modelo proporcionado con JSBSim directamente para ver su comportamiento en vuelo. Algunas de sus ventajas son:

- Existe un manual de usuario
- Amplio catálogo de ejemplos

- Detección de errores y mejor visualización
- Los ficheros de datos de entrada no son tan restrictivos (en DATCOM se exigía ponerle nombre determinado para cada archivo de la geometría)
- Modelos 3D mejores, con capacidad para manejar las superficies de control
- Fácil comparación de los coeficientes mediante gráficas
- Imágenes de las secciones transversales del fuselaje y las alas
- Mejor integración con JSBSim
- Ejemplos de maniobras de vuelo en JSBSim
- Fácil comparación de los resultados de los test de vuelo
- Buena documentación, incluye información para construir nuestro propio modelo

Por otro lado, también cuenta con sus limitaciones:

- No puede simular colas verticales gemelas (nos afecta)
- No es capaz de simular misiles (no nos afecta)
- No es capaz de simular helicópteros (no nos afecta)

AVL (Athena Vortex Laticce)

AVL es un software libre para el análisis aerodinámico y dinámica de vuelo de aeronaves rígidas de cualquier tipo de geometría, a partir de archivos de entrada en los que se especifica su geometría, masa y parámetros de vuelo. Usa el método de cálculo de dinámica de fluidos "Vortex Laticce" para realizar el análisis sobre sus superficies. Originalmente fue creado por Harold Youngren en 1988 para el MIT (Massachusetts Institute of Tecnhology), pero desde su creación ha recibido numerosas modificaciones, hechas por el propio Youngreen y por Mark drela.

Este software nos permite modelar a nuestro gusto diferentes características:

- Componentes aerodinámicos: superficies de sustentación y cuerpos delgados de revolución (fuselaje, motores,...)
- Definir los perfiles alares
- Introducir singularidades en la geometría
- Manejo de las superficies móviles de control (flaps, profundidad,...)

- Configuración libre de los parámetros:
  - Ángulos alfa (ángulo de ataque) y beta (ángulo de deslizamiento lateral)
  - Rotaciones en los tres ejes (p=alabeo, q=cabeceo, r=guiñada)
  - Transformación subsónica de compresibilidad usando el método Prandtl-Glauert
- Datos de salida:
  - Fuerzas y momentos provocados por la aerodinámica en el cuerpo o en los ejes
  - Análisis Trefftz del arrastre inducido en la aeronave
  - Derivativos de fuerzas y momentos, así como ángulos, rotaciones y controles
- Ajustes de cálculo:
  - Variables: alfa, beta; rotaciones p, q, r; variación de las superficies de control
  - Posibilidad de realizar diferentes cálculos para diferentes condiciones de vuelo
- Propiedades de masas:
  - Se puede incluir un archivo en donde se definan las unidades usadas, localización de masas e inercias
- Análisis físico:
  - Es capaz de predecir la estabilidad de vuelo
  - Modelos de cuerpo rígido, cuasi-estáticos
  - Proporciona matrices de datos del sistema dinámico

Una de las principales ventajas de AVL, y que va a afectar a nuestra elección, es que, a diferencia de DATCOM u HolyCows, este programa sí está diseñado para velocidades de vuelo bajas, como en nuestro caso, y por lo tanto es un dato importante a considerar.

En la web del desarrollador, no aparecen documentadas limitaciones de este programa, por lo que el propio hecho de no aparecer sus inconvenientes es una limitación para nosotros ya que no sabemos hasta que punto podemos definir parámetros de nuestro modelo, y el grado de realismo al que podemos llegar. Una de las desventajas que se han descubierto tras el uso del programa ha sido que solo se pueden añadir cuerpos de revolución al modelo, por lo que tendremos que adaptar nuestro fuselaje (que no es de revolución) y crear un cuerpo de revolución equivalente.

Este software, da como resultado una gran cantidad de coeficientes que determinan la aerodinámica del UAV, superando a la información proporcionada por DATCOM o su versión modificada, HolyCows.

Túnel de viento

La última opción que manejamos para llevar a cabo esta tarea es hacer un estudio aerodinámico del UAV en el túnel de viento del que disponemos en la UDC. Esta es la única opción experimental de las 4. Recopilar información aerodinámica a partir de ensayos en túnel de viento, es la forma más realista de obtener información, porque se obtiene a partir del modelo real, con sus defectos, a diferencia de los software anteriores, en los que se introduce un archivo con la geometría idealizada. Además, gracias a que nuestro UAV no es muy grande, se podría usar directamente en el túnel, sin necesidad de crear un modelo a escala. Éstas son sus principales ventajas, pero también tiene algunos inconvenientes:

- Necesitamos un período de preparación de las pruebas mucho mayor
- Todos los cálculos de coeficientes deben realizarse uno a uno, a partir de información básica
- Dificultad para realizar las pruebas en comparación con el método informático

Una vez expuestas todas las ventajas e inconvenientes de cada una de las formas que tenemos para obtener la información aerodinámica, ya tenemos una base sobre la que seleccionar razonadamente cual va a ser la herramienta que usemos.

Selección

En primer lugar podemos descartar el uso de Digital DATCOM, porque es la versión más antigua del programa, y en tal caso HolyCows es una versión más avanzada que nos ofrece unos mejores resultados y además nos facilita la lectura de los resultados, que es uno de los grandes problemas de DATCOM. Además, no es posible simular dos colas verticales gemelas en aeronaves usando DATCOM, siendo necesario modelar una sola cola equivalente a las otras dos. Por último, tampoco es posible introducir la superficie de la profundidad, por lo que estaríamos despreciando una superficie de control.

También podemos descartar el método del túnel de viento. Como hemos dicho antes, posiblemente esta sea la forma más realista de obtener la información sobre su aerodinámica, y además podemos trabajar con el UAV real porque no es muy grande. Pero las desventajas pesan demasiado sobre las ventajas de este método. No disponemos del tiempo necesario para la realización de las pruebas en

túnel de viento, y en tal caso, se retrasaría la entrega del proyecto. Además no tenemos la experiencia ni los conocimientos suficientes para hacerlo de este modo, por lo que tendríamos que invertir aún más tiempo. Es un método más difícil, más laborioso y más lento.

Por último nos quedan HolyCows y Athena Vortex Laticce como candidatos. Las ventajas expuestas antes sobre el uso de HolyCows hacían referencia a la versión DATCOM+PRO, que es la versión de pago, de esta forma estaríamos incumpliendo uno de los objetivos de este proyecto. La versión libre (DATCOM+) no tiene tantas ventajas con respecto a DATCOM. Por otro lado tampoco podemos modelar colas verticales gemelas, ni está diseñado para velocidades de vuelo bajas como es nuestro caso. AVL en cambio, sí permite la simulación para velocidades de vuelo bajas, y este va a ser un factor que influya mucho en los resultados. Además, se han encontrado proyectos y tesis anteriores sobre UAVs realizadas con AVL, por lo que deducimos que es un software bastante apropiado para nuestro caso. Cabe destacar también que es el que tiene las actualizaciones y modificaciones hechas más recientemente, por lo que se puede decir que es el de mejor soporte.

Para hacer esta comparación más visual, hemos hecho la siguiente tabla, en la que se puntúan algunos de los aspectos de los diferentes métodos, en donde 1 corresponde a la peor puntuación, y 4 a la mejor.

	DATCOM	**HolyCows**	**Túnel de viento**	**AVL**
Documentación	4	2	3	2
Usado en trabajos anteriores	1	1	2	4
Lectura de resultados	1	3	2	3
Precisión de resultados	1	2	3	2
Adaptabilidad a nuestro modelo	1	2	4	3
Condiciones de vuelo	1	1	4	4
Rapidez de obtención de resultados	3	4	1	4
TOTAL	***12***	***15***	***19***	***22***

Tabla 3.6: Comparación de los 4 métodos de cálculo aerodinámico

En conclusión, usaremos el **Athena Vortex Laticce (AVL)** para introducir un archivo que contenga los datos de geometría del UAV para obtener toda la información aerodinámica necesaria para llevar a cabo la simulación obteniendo los resultados más realistas posibles.

3.3.3.2. Características del UAV

Antes de empezar a modelar el UAV conviene explicar algunas de sus características, como su forma, cuales son sus elementos de control, como funcionan, y su influencia en la dinámica de vuelo del UAV. Primero debemos ver los elementos de control de una aeronave general, como se ven en la siguiente imagen, y después centrarnos en nuestro caso.

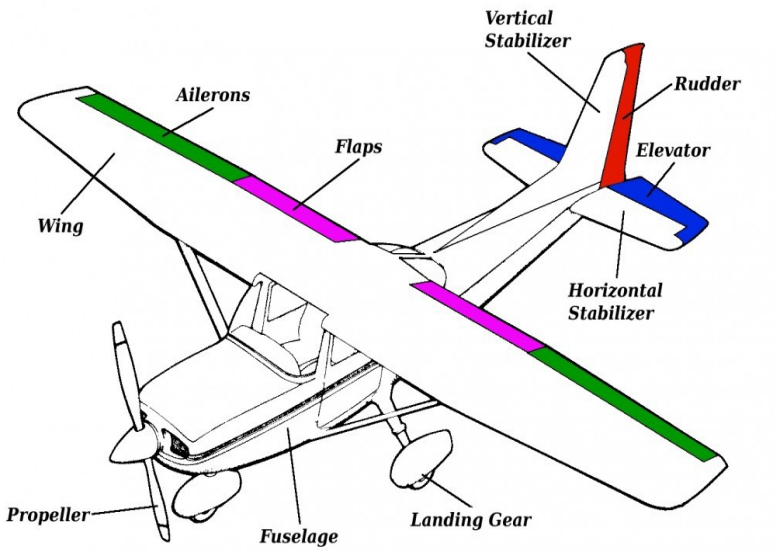

Ilustración 3.13: Principales superficies de control en aeronaves

Las funciones de estas superficies son:

- **Derivas** (rudder): Se encarga de controlar los movimientos de guiñada.
- **Profundidad** (elevator): Son los responsables del movimiento de cabeceo.
- **Alerones** (ailerons): Tienen la función de provocar los movimientos de alabeo de la aeronave.
- **Flaps**: Ejercen de frenos durante el aterrizaje inclinados hacia arriba o generan una mayor sustentación durante el despegue con un ángulo negativo.

Estas superficies controladoras no solo ejercen estos movimientos, si no que provocan otros efectos secundarios debido a los momentos y fuerzas que generan con respecto al centro de gravedad, al no estar el centro de presiones de la superficie en la misma horizontal que el CG. Se podría explicar en mayor profundidad pero a nosotros nos es suficiente con esto para entender nuestro caso.

Nuestro UAV, tiene dos derivas en las colas verticales, y unas superficies de control que son un híbrido entre los elementos de profundidad y los alerones llamado elevones (de sus nombres en inglés elevator+aileron). Están situados en la parte trasera de las alas del UAV, a lo largo de todo el borde de salida como se puede ver en el plano del ANEXO VII: "Plano del UAV". Estos elevones ejercen la función de generar los movimientos de cabeceo y de alabeo según se requiera en cada momento.

En la siguiente tabla se pueden ver algunas características necesarias para la creación del modelo geométrico en AVL.

Área de referencia (m^2)	1,1534
Cuerda Aerodinámica Media (m)	0,431
Envergadura de referencia (m)	1,94
Espesor máximo en la zona central (m)	0,059
Ángulo de dihedro (º)	2
Ángulo de torsión (º)	2
Ángulo de flecha (º)	25

Tabla 3.7: Principales características geométricas

El área de referencia hace referencia a la superficie de sustentación incluida la superficie aportada por los elevones, al igual que la cuerda aerodinámica media.

3.3.3.3. Modelado geométrico del UAV en AVL

Para realizar el modelado, nos servimos de la documentación proporcionada por la web del programa y de ejemplos disponibles también en la web. Introducimos los datos de 18 secciones de la parte alar con ángulo de flecha, mas otras 4 secciones de la parte alar que no tiene flecha. Consideramos que con esto tenemos una superficie suficientemente detallada, ya que la raíz del ala y la punta las unen una linea recta, sin curvatura visto desde arriba, al igual que en la parte central del ala, que tampoco tiene curvatura visto desde arriba.

Para hacer esto, primero tenemos que declarar una serie de parámetros que afectan a todo el UAV, estos son:

- **Número de Mach:** calculado como

$$Mach = V_{UAV} / V_{sonido} = 12{,}5 / 343 = 0{,}036$$

- **Planos de simetría aerodinámica** (*Iysym Izsym Zsym):* en nuestro caso no tenemos

simetría aerodinámica con respecto a ningún plano, porque aunque sí sea geométricamente simétrico, no podemos programarlo como aerodinámicamente simétrico porque en la situación de querer hacer un giro de alabeo (sobre el eje X) uno de los alerones deflecta hacia arriba y el otro hacia abajo para poder provocar el giro, y por lo tanto no es aerodinámicamente simétrico. Posteriormente se explicará como se soluciona esto.

- **Área de referencia** (*Sref*): es el área alar de la vista en planta del UAV y vale 1,1534 m.
- **Cuerda Aerodinámica Media** (*Cref*): es la cuerda media de las alas y vale 0,4306 m. Calculada con la ecuación:

$$CAM = (2/S_{ref}) * \int_0^{b/2} c^2 \, dy = 0{,}4306\, m$$

En donde:

Sref = superficie alar

b = envergadura de la semiala

c = cuerda en la coordenada “y”

y = coordenada a lo largo de la envergadura alar

- **Envergadura** (*Bref*): 1,94 m.
- **Centro de masas:** punto medido tomado como origen de coordenadas la "punta" del UAV.

Xref = 0,254 m *Yref* = 0,0 m *Zref* = 0,0295 m

A partir de aquí se pueden empezar a crear cada una de las superficies (llamadas “SURFACE” en el archivo) que componen el UAV. La primera de ellas es "CENTRAL WING", que corresponde a la superficie para el ala central y otra llamada “WING” para el ala con ángulo de flecha. También creamos otra para las colas verticales llamada “RUDDER”. Posteriormente añadiremos los winglets y el fuselaje. Los parámetros a programar en las superficies son:

- ***Nchordwise***: indica el número de puntos de cálculo que va a realizar el programa en la cuerda de la superficie, es decir, el número de puntos de cálculo entre el borde de ataque y el borde de salida. En nuestro caso ponemos 10, suficiente para proporcionar unos resultados bastante precisos, dado el tamaño del UAV.
- ***Cspase***: establece la densidad y distribución de los puntos de cálculo a lo largo de la cuerda de la superficie alar. Siguiendo la documentación del programa, nosotros escogemos la

distribución *"1.0"* porque es la más eficiente en general para cualquier superficie. Esta distribución concentra los puntos de cálculo en el borde de ataque y en el borde de salida, los puntos más críticos aerodinámicamente.

- ***Nspanwise***: es el número de puntos de cálculo entre cada sección geométrica declarada. Nosotros ponemos 6, porque es el máximo de puntos de cálculo que pudimos establecer sin que el programa diera problemas, con más de 6 se cerraba automáticamente nada más ejecutarlo. Este parámetro se declara en cada sección porque al hacerlo para toda la superficie se cerraba automáticamente el programa.
- ***Sspace***: establece la densidad y distribución de los puntos de cálculo entre cada sección, al igual que en el caso de *Cspace*, escogemos la distribución tipo *"1.0"* porque según la documentación es la más eficiente en general. Este parámetro se declara en cada sección porque al hacerlo para toda la superficie se cerraba automáticamente el programa.
- ***COMPONENT***: Sirve para decirle al programa a que "conjunto de superficies" pertenece la superficie que se está programando, es decir, para que el programa sepa que esa superficie va unida a otra con el mismo número de "COMPONENT" nosotros usaremos pondremos 1 a todas las superficies programadas porque van todas unidas.
- ***YDUPLICATE***: aquí podemos establecer el plano en el eje Y sobre el cual la superficie que estamos modelando es simétrica geométricamente (no aerodinámicamente). En nuestro caso es simétrica con respecto al plano del eje Y que pasa por el origen de coordenadas 0.0.
- ***ANGLE***: es el ángulo de dihedro de la superficie, es decir, el ángulo que forma el ala con respecto al plano XY.
- ***SECTION***: aquí se declaran los parámetros de cada sección:
  1. *Xle, Yle y Zle*: es la posición del borde de ataque en cada sección
  2. *Chord*: cuerda alar, incluida la superficie de control que pueda haber en el borde de salida del ala
  3. *Ainc*: es el ángulo de torsión en cada sección. La zona alar central no tiene ángulo de torsión, pero la zona alar con ángulo de flecha tiene un ángulo de torsión de 2º en la punta de ala. Este ángulo va aumentando progresivamente desde 0º en la raíz de ala hasta los 2º en la punta.
  4. *Nspanwise y Space:* explicados anteriormente.

- **Perfil alar:** es el perfil del ala para cada sección. AVL tiene una base de datos de perfiles NACA, por lo que encontrando un perfil lo más parecido posible al real podemos usar este perfil en nuestro modelo. Tomando medidas del UAV y calculando algunos parámetros, hemos hallado que el perfil NACA más parecido a nuestro perfil alar es el NACA 3210, como el que se ve en la figura:

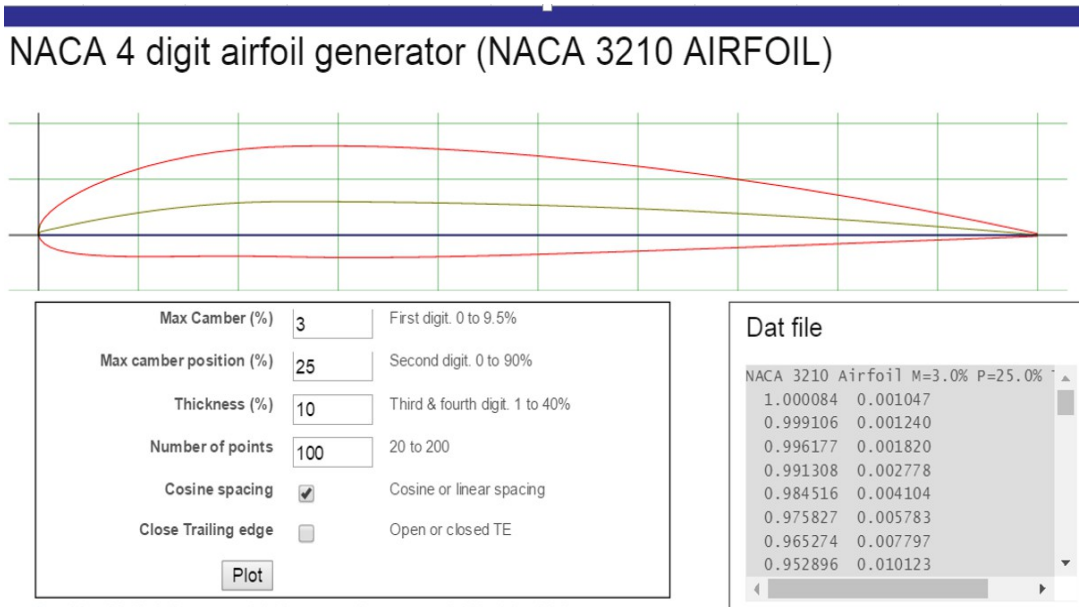

Ilustración 3.14: Perfil NACA 3210 y algunos parámetros

Para las secciones en las que no se declara un perfil determinado, el programa, por defecto usa una placa plana como perfil (esto se usa en las colas verticales y los winglets).

- ***CONTROL***: solo es necesario declarar esto en las superficies que contengan elementos de control, es decir, las alas con ángulo de flecha que contienen los elevones y las colas verticales que contienen la profundidad. AVL permite programar 2 controles en la misma sección, lo que es necesario para los elevones. Si miramos en el anexo *"hercules.avl"* vemos que tenemos el nombre de la superficie de control seguido de 6 números.

  1. *1º número:* es la ganancia del giro de la superficie. En nuestro caso va a ser siempre 1.
  2. *2º número:* es la posición, en tanto por 1, de la cuerda en donde acaba el perfil alar y comienza la superficie de control, en donde 0 representa el borde de ataque y 1 el ángulo de salida.
  3. *3º, 4º y 5º número:* representa el eje de giro de la superficie de control en el mismo origen de coordenadas en el que empezamos a programar la geometría.
  4. *6º número:* sirve para saber como va a deflexionar la superficie de control de la otra

mitad del UAV. Anteriormente explicamos que para que la superficie que estamos creando sea simétrica geométricamente pero no aerodinámicamente, debemos declarar la geometría en *YDUPLICATE*, esto significa que podemos hacer que, por ejemplo, el alerón (*aileron*) de la mitad izquierda deflecte hacia abajo y el derecho hacia arriba. Si colocamos un -1 y le damos una deflexión de 20º, el alerón derecho girará 20º con respecto a su eje de giro y el izquierdo -20º, consiguiendo el movimiento de alabeo. Para la deriva (*elevator*) y profundidad (*rudder*) colocamos un 1 porque en estos casos queremos que giren hacia el mismo lado para conseguir los movimientos de cabeceo y guiñada respectivamente.

- ***TRANSLATE***: con este comando podemos trasladar el borde de ataque de una superficie hacia cualquier punto del espacio. Lo utilizamos para colocar las colas verticales y los winglets.

Con esto ya tenemos un modelo preliminar del que podemos sacar información y hacer una primera simulación de prueba.

A continuación se perfecciona el modelo añadiéndole los winglets a los que hemos llamado "WINGLET" y el fuselaje, a la que llamamos "FUSELAGE". Para los winglets seguimos los mismos pasos que en el proceso de creación de las alas y las colas verticales, porque al igual que estos es una superficie. Sin embargo, el fuselaje es diferente, no lo podemos modelar como un conjunto de superficies, debe ser un cuerpo volumétrico (llamado "BODY" en el archivo), y además AVL solo permite crear cuerpos volumétricos de revolución. El cuerpo de nuestro UAV no es un cuerpo que se pueda representar por revolución, como se puede ver más adelante. La solución más lógica para este problema, y que además viene explicada en la documentación del programa, es construir un cuerpo de revolución que tenga la misma área frontal y la misma área lateral que el real.

El área frontal la podemos calcular fácilmente porque forma un rectángulo de 69 x 92 mm, resultando una área frontal de:

$$A_{frontal} = 0{,}069 * 0{,}092 = 6{,}348 \cdot 10^{-3}\, m^2$$

De aquí podemos obtener el radio de una superficie circular de igual área de la siguiente forma:

$$A_{frontal} = 6{,}348 \cdot 10^{-3} = pi * r^2$$

Obteniendo un radio r=44,95 mm que podemos aproximar como r=45 mm.

Para hallar el área lateral, no podemos usar ningún método de cálculo porque la forma lateral es una

figura irregular como la que se ve en la imagen, lo calcularemos de “forma geométrica”:

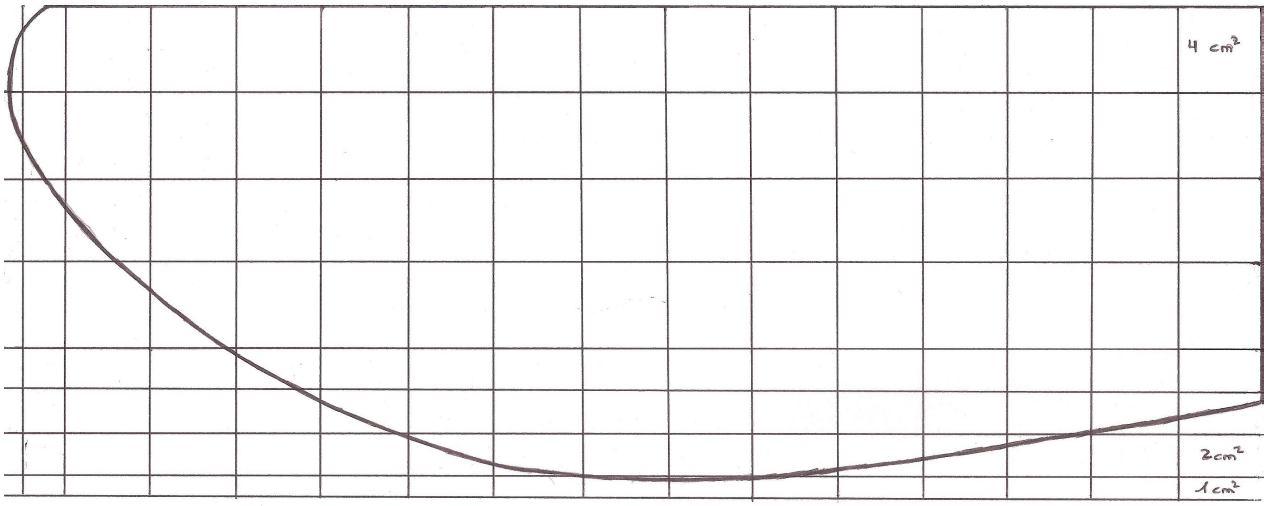

Ilustración 3.15: Forma lateral del fuselaje

Al no tener una forma geométrica reconocible, se aproxima su superficie dividiéndola en rectángulos de área conocida, suficientemente pequeños (400 mm^2 y de 200 mm^2 y 100 mm^2) para que el error de aproximación sea despreciable. Obtenemos un área lateral aproximada de 0,0274 m^2

Para representar en AVL podemos hacerlo mediante una elipse de área equivalente, obteniendo un elipsoide como cuerpo de revolución equivalente. Para calcular los parámetros de la elipse equivalente:

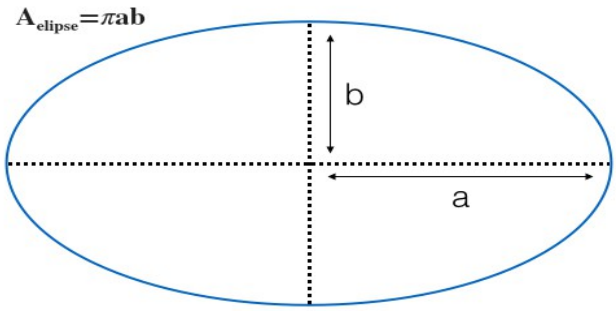

Ilustración 3.16: Parámetros de una elipse

La longitud del semieje vertical “b” ya lo sabemos, es igual al radio “r” de la circunferencia del área frontal, y la longitud del semieje horizontal “a” lo podemos calcular mediante la ecuación del área de una elipse:

$$A_{elipse} = A_{lateral} = 0{,}0274 = pi * 0{,}045 * a$$

De donde obtenemos que la longitud a = 0,194 m = 194 mm.

Con estos datos, podemos sacar los puntos que definen la elipse en una hoja de cálculo de Excel a partir de la ecuación de una elipse:

$$x^2 \div a^2 + x^2 \div b^2 = 1$$

Por último introducimos los puntos de la elipse calculados en el archivo "herc_body.dat" para que posteriormente los pueda leer AVL.

Al representar esto, tuvimos un problema, AVL solo representa la mitad trasera del elipsoide y por la parte delantera aproxima un cuerpo redondeado, pero que no corresponde a un elipsoide, quedando la siguiente representación:

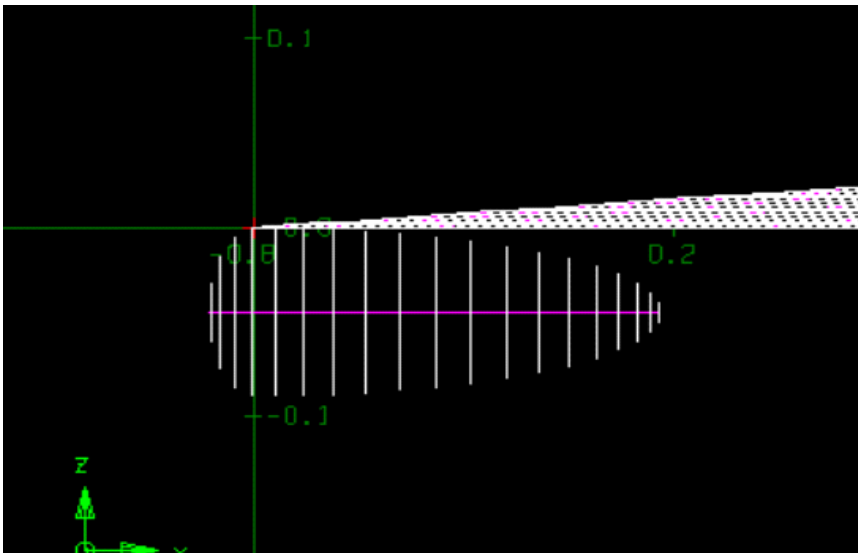

Ilustración 3.17: Cuerpo del UAV

Pero de esta forma, solo tendremos la mitad del área lateral calculada, más una pequeña área de la parte frontal creada automáticamente por AVL.

Para solucionar esto, multiplicamos por 1,8 el área lateral hallada (y no por 2 para tener en cuenta el área de la parte delantera) y volvemos a repetir el proceso para hallar las coordenadas de la nueva elipse. Más adelante veremos la representación.

Los parámetros necesarios para la creación del fuselaje en AVL son:

- ***Nbody***: es el número de puntos de cálculo entre el punto más adelantado del fuselaje y el punto más retrasado. Nosotros consideramos suficientes 30 puntos de cálculo.
- ***Bspace***: es el tipo de distribución de los puntos de cálculo a lo largo del fuselaje. Al igual que hicimos en las superficies alares, le asignamos un valor de "1.0", que concentra estos

puntos en los bordes delantero y trasero. Según la documentación, es el más eficiente.

- ***TRANSLATE***: ya explicado anteriormente.
- ***BFIL***: es el nombre del fichero que contiene las coordenadas que representarán el cuerpo del fuselaje, como hemos dicho antes, este archivo se llamará "*herc_body.dat*"

Como resultado de este modelado obtuvimos la siguiente representación del UAV:

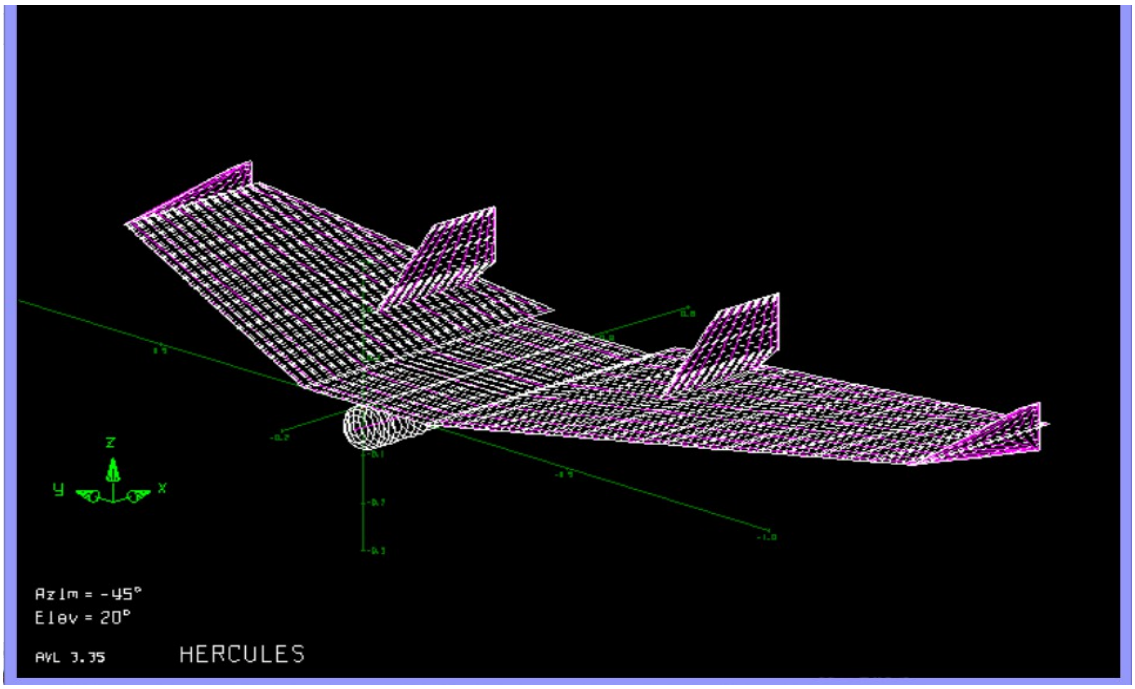

Ilustración 3.18: Representación del UAV

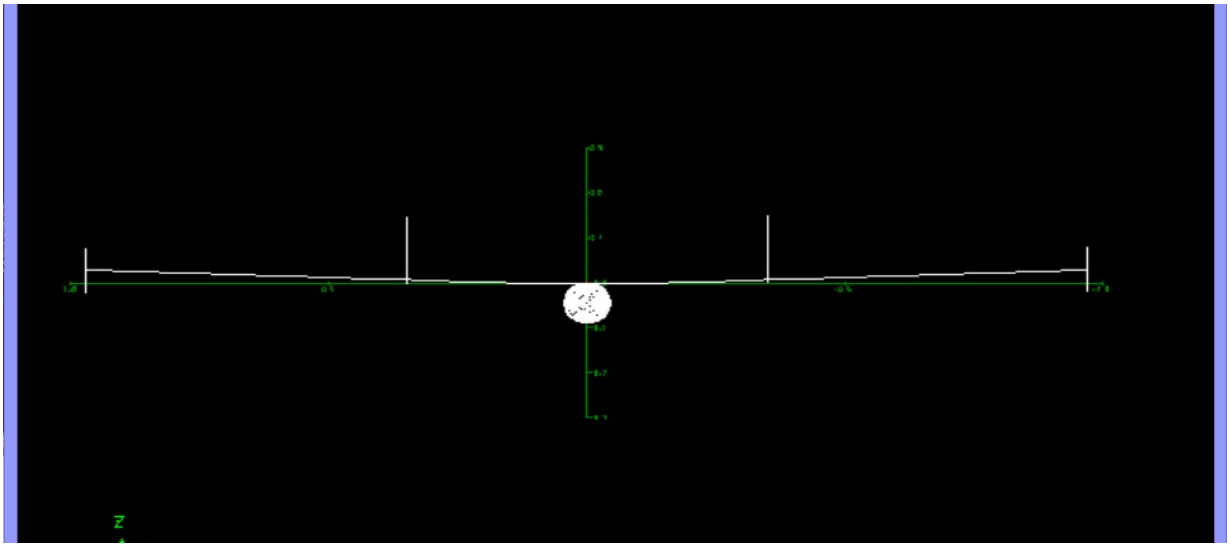

Ilustración 3.19: Vista frontal

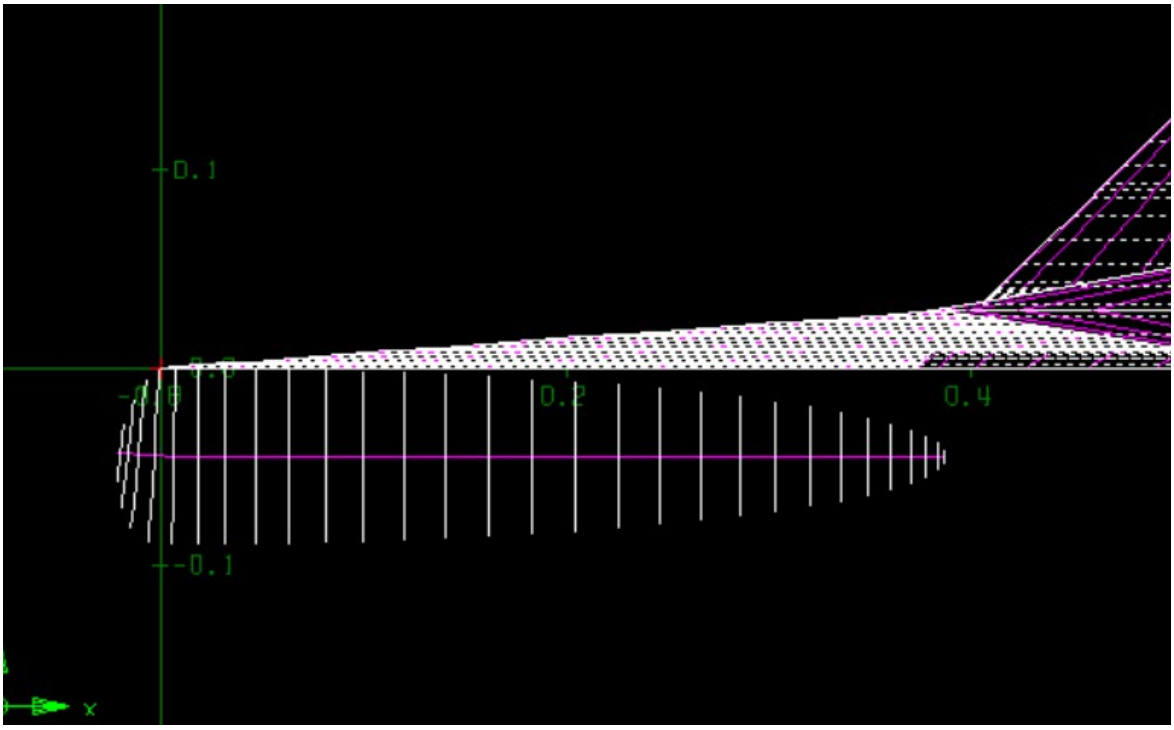

Ilustración 3.20: Vista lateral del fuselaje

El archivo resultante se puede ver en el ANEXO V: “hercules.avl”.

3.3.3.4. Cálculo de los coeficientes aerodinámicos

Una vez que tenemos el modelo geométrico similar, de un UAV de características similares, necesitamos programar unas condiciones de vuelo constantes (número de Mach, velocidad de vuelo, densidad del aire, aceleración de la gravedad y masa) del UAV para los cuales el programa hará el cálculo de coeficientes aerodinámicos que serán introducidos en el archivo que viene en el ANEXO III: "*Fichero de aerodinámica*" y determinarán la dinámica de vuelo en el simulador. Se crearán varios casos variando los diferentes parámetros (ángulo de ataque, ángulo de incidencia lateral del viento, deflexión de alerones, de profundidad y de deriva, velocidad de alabeo, de cabeceo y de guiñada) del UAV para obtener el efecto que produce la modificación de cada parámetro, por ejemplo, crearemos un caso en el que el ángulo de ataque sea 0º, después otro en el que sea 2º, 4º, 8º,... sin variar los demás parámetros, obteniendo una tabla de resultados que nos dice como influye la variación de este parámetro en el comportamiento aerodinámico. Posteriormente otros casos en donde obtendremos los resultados del cálculo para giros del alerón de 10º, 20º y 30º, obteniendo como resultado final otras tablas de datos que indicarán como afecta cada parámetro en el comportamiento del UAV. Debemos mencionar que AVL no proporciona todos los coeficientes que

necesitamos para la simulación en FlightGear, solo obtenemos de él el coeficiente de arrastre total (CDtot), coeficiente de sustentación (CL), coeficiente de fuerza lateral (CY), coeficiente de momento de alabeo (Cl), coeficiente de momento de cabeceo (Cm) y coeficiente de momento de guiñada (Cn) y la eficiencia (e). A partir de estos 7 datos podemos obtener el resto de coeficientes calculándolos en hojas de Excel, simplemente restándole a los resultados obtenidos tras la modificación de cada parámetro, el valor del mismo coeficiente cuando todos los parámetros estaban a 0.

Las condiciones de vuelo que serán constantes para todas los cálculos son:

Número de Mach	0,036
Velocidad (m/s)	12,5
Densidad del aire (kg/m^3)	1,225
Aceleración de la gravedad (m/s^2)	9,81
Masa (kg)	3

Tabla 3.8: Condiciones de vuelo constantes

Debemos destacar, que no solo hemos hecho un cálculo de los coeficientes y lo hemos dado por bueno. En primer lugar hemos hecho un modelo sin los winglets y sin el fuselaje del UAV en el que hemos obtenido unos valores y hemos puesto en el archivo de aerodinámica para probar en el simulador que el comportamiento durante el vuelo se aproxima a los vuelos realizados con el UAV real. En un principio, el comportamiento del UAV en el simulador poco se acercaba a la realidad, era muy inestable durante el vuelo. Usando la documentación de FlightGear hemos encontrado un modelo similar y hemos comparado nuestros resultados con estos para asegurar que los resultados no eran resultados sin sentido, que los signos coincidieran y que tuvieran un orden de magnitud aproximado. A la vista del comportamiento extraño en el simulador, fuimos sustituyendo uno por uno todos los valores que obtuvimos por los aportados en la documentación de FlightGear, hasta encontrar los valores que hacían que se comportara así nuestro modelo. Hemos observado que uno de los valores, el "momento de cabeceo debido a la velocidad de cabeceo" era un orden de magnitud menor que el valor de la documentación. Modificamos este valor multiplicándolo por 10 de forma que se aproximase más al valor aportado en la documentación y probamos de nuevo a simular el modelo, obteniendo comportamiento mucho más estable. También fue necesario modificar otros valores que se especificarán más adelante uno por uno, y explicando el porqué de su modificación.

El siguiente paso fue introducir los winglets y el cuerpo del UAV, volver a realizar el cálculo aerodinámico en AVL e introducir los valores obtenidos en el archivo "*Fichero de aerodinámica*" y

volver a simular.

A continuación se redacta una breve explicación de cada uno de los coeficientes aerodinámicos que hemos considerado para el desarrollo del modelo virtual, de los resultados obtenidos en AVL y de sus correcciones en los casos en los que fue necesario. Han sido agrupados en 6 grupos, atendiendo a la fuerza o movimiento que afecta cada uno: fuerza de arrastre, fuerza lateral, fuerza de sustentación, alabeo, cabeceo y guiñada.

FUERZAS DE ARRASTRE

- **CD0:** *coeficiente de arrastre a sustentación cero.*

Es un parámetro adimensional que relaciona la fuerza de arrastre de una aeronave en reposo (sustentación cero), con su forma, tamaño y rugosidad de superficie. Refleja el arrastre "parasitario", que es muy útil para entender como de aerodinámica es el UAV.
Se han obtenido los resultados de este valor para variando el ángulo de ataque para valores de 0º, 2º, 4º, 8º, 10º y 45º, aunque posteriormente le hemos añadido el resultado para 90º. Lo hemos calculado usando la ecuación:

$$CD0 = CD - CL^2 / (pi * B_{ref}^2 / S_{ref} * e)$$

Los resultados fueron:

α (º)	CD0
0	0,0008
2	0,0005
4	0,0005
8	0,0009
10	0,0013
45	0,0146

Tabla 3.9: CD0 en función de alfa

Este parámetro, tiene especial influencia sobre la velocidad de vuelo del UAV, cuanto menor sea el valor, mayor la velocidad que es capaz de alcanzar, y viceversa. Durante las simulaciones con estos datos en FlightGear, hemos observado las velocidades que alcanzaba, siendo como máximo de 100 nudos (=185,2 km/h) aproximadamente, lo cual no se corresponde con la realidad. En pruebas de vuelo realizadas, se ha visto que la velocidad del UAV en vuelo con los motores a media potencia es aproximadamente de 50 km/h, por lo que este valor puede ser el causante de nuestro error.

Para encontrar el error de estos resultados los comparamos con la siguiente tabla de coeficientes de arrastre parasitarios de diferentes geometrías:

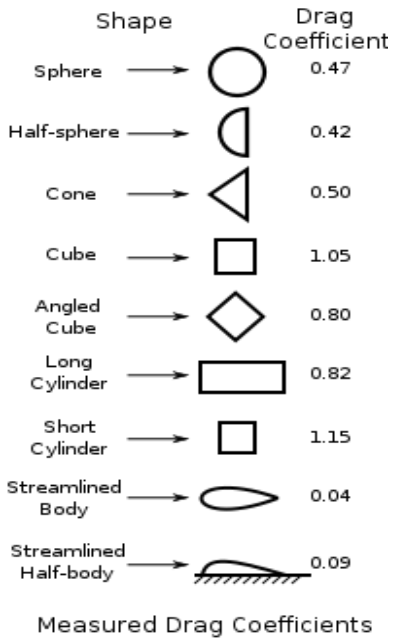

Ilustración 3.21: CD0 de diferentes geometrías

En comparación, nuestros resultados son 2 órdenes de magnitud más pequeños que los indicados en la tabla para cuerpos aerodinamizados.

AVL no solicita datos sobre la rugosidad de la superficie, ni es posible introducir las cavidades existentes en la superficie del UAV, estas limitaciones afectan a los resultados de CD0 obtenidos. Teniendo en cuenta la ilustración anterior y las limitaciones de AVL citadas, decidimos aumentar progresivamente los valores de CD0 para los ángulos de ataque de 0º, 8º y 10º (obviamos los resultados para 2º y 4º ya que son casi igual que para 0º), de forma que tuviésemos unos resultados próximos a los que se ven en la tabla y que mostraran un comportamiento más realista del modelo a simple vista.

Para seguir una curva coherente, añadimos también un ángulo de ataque de 90º (tomando el valor de arrastre de una placa plana) y calculamos el arrastre para el ángulo de alfa=45º mediante la ecuación:

$$CD_0 = 1{,}28 * \sin(alfa)$$

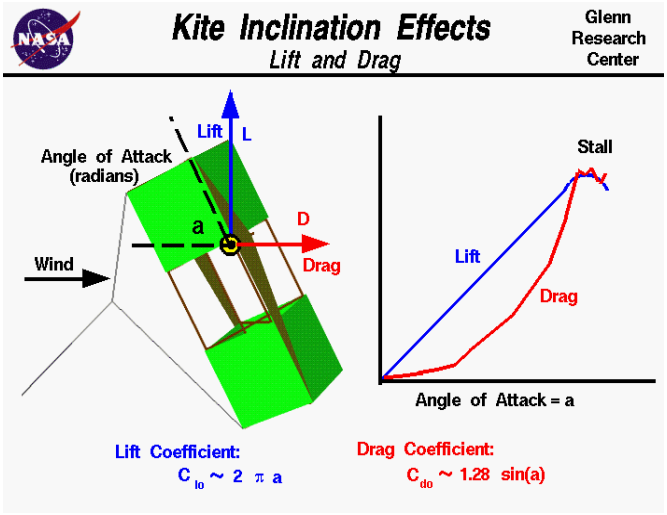

Ilustración 3.22: Aproximación de CD0 en función de alfa

Para ángulos de ataque >50º no es posible realizar el cálculo en AVL, aparece el mensaje: "*Cannot trim. Alpha too large.*"

Por lo tanto los resultados que introducimos finalmente en el archivo son:

α (º)	CD0
0	0,08
8	0,09
10	0,13
45	0,905
90	1,28

Tabla 3.10: CD0 modificados

Una vez introducidos, comprobamos que el UAV alcanza una velocidad razonable en el simulador, de entre 24 y 30 nudos (entre 45 y 55 km/h aproximadamente) con un ángulo de ataque de 0º, lo que coincide con las velocidades alcanzadas en las pruebas de vuelo reales. De esta forma damos por buenos estos resultados.

- **CDi:** *coeficiente de arrastre inducido.*

Es un parámetro adimensional que aparece cuando un flujo de aire circula alrededor del cuerpo del UAV y genera una fuerza de sustentación. A diferencia que en el caso anterior, para que se de este efecto debe existir una velocidad relativa entre el cuerpo y el flujo que circula a su alrededor. También es el arrastre producido por la punta de ala cuando existe sustentación.

Los resultados que obtuvimos se calcularon mediante la ecuación siguiente, y se ven en la siguiente table, no necesitaron modificación alguna:

α (º)	CD
0	0,0083
2	0,0152
4	0,024
8	0,047
10	0,061
45	0,3598
90	0,3704

Tabla 3.11: CD inducido en función de alfa

El arrastre inducido para alfa=90º fue introducido posteriormente.

- **CDβ**: *coeficiente de arrastre debido al deslizamiento lateral*

Cuando el UAV se encuentra en vuelo, la dirección de vuelo no tiene por que ser la misma que la del viento, cuando la dirección del viento es oblicua a la trayectoria de la aeronave, se ejercen fuerzas que provocan un giro, y por lo tanto un cambio de trayectoria. Para considerar este efecto se considera este coeficiente.

Este ángulo entre la dirección del viento y la trayectoria de la aeronave se denomina β, y es positivo cuando el viento incide por la derecha de la aeronave (visto desde atrás), y negativo desde la izquierda.

Los resultados obtenidos para este parámetro fueron:

β (º)	CD
0	0
2	1,00E-04
4	0,00039
6	0,00087
8	0,00154
15	0,0500
90	1,2300

Tabla 3.12: CD en funición de beta

Los ángulos de15º y 90º se añadieron posteriormente en el archivo con los datos del ejemplo aportado por la documentación.

- **CDde**: *Coeficiente de arrastre debido al ángulo de profundidad*

Es la contribución al efecto de arrastre que hace el elemento de profundidad (elevator) del UAV. El valor de este coeficiente tiene importancia a la hora de determinar el comportamiento durante el despegue y el aterrizaje y en las maniobras de coger altura o disminuirla durante el vuelo. Su valor crece en función de las solicitaciones de estas operaciones (STOL "Short Take Off and Landing").
Los resultados que obtuvimos fueron los siguientes:

Giro de profundidad (º)	CD
10	0,01304
20	0,03276
30	0,05915

Tabla 3.13: CD debido a profundidad

Estos valores forman una recta con una pendiente de valor **0,1321.** Para el cálculo de la pendiente se utilizará el valor de giro del elemento de control correspondiente en radianes, y no en grados como aparece en las tablas. Este valor será el que introducimos en el archivo. Las siguientes tablas de datos que representadas gráficamente formen una recta tendrán una pendiente, que será el dato a introducir en el archivo, en lugar de la tabla de datos.

FUERZA LATERAL

- **CYβ:** *Fuerza lateral debido al ángulo de deslizamiento lateral beta*

Es el coeficiente que considera los efectos de la fuerza que viento que incide lateralmente con un ángulo β, ejerce sobre el UAV.
Los resultados obtenidos fueron:

β (º)	CY
0	0
2	-0,0044
4	-0,00879
6	-0,01317
8	-0,01753

Tabla 3.14: Coeficiente de fuerza lateral en función de beta

Estos valores forman una recta con una pendiente de valor **-0,1254.**

FUERZAS DE SUSTENTACIÓN

CLα: *Coeficiente de sustentación debido al ángulo de ataque*

Es el efecto de sustentación (fuerza vertical) que se produce debido a la inclinación o ángulo de ataque de la aeronave a través del aire.

Los resultados obtenidos fueron:

α (º)	CL
0	0,27343
2	0,37099
4	0,46751
8	0,65613
10	0,7476
45	1,81912

Tabla 3.15: CL en función de alfa

Durante la simulación, comprobamos que para ángulos de ataque altos, el UAV generaba más sustentación y por lo tanto ascendía rápidamente, cuando debería entrar en pérdida. Ante este fenómeno, buscamos información sobre el efecto del ángulo de ataque sobre la sustentación, y algunas fuentes proporcionaban información que establecían que la sustentación máxima para aeronaves en general se produce para ángulos de ataque de entre 15º y 20º aproximadamente. En los resultados proporcionados por AVL vemos que la sustentación sigue creciendo a partir de ese punto (por ejemplo, obtuvimos un CL=1,81912 para un ángulo de 45º). A continuación vemos una gráfica que relaciona el CL con el ángulo α, sacada de la web de la NASA:

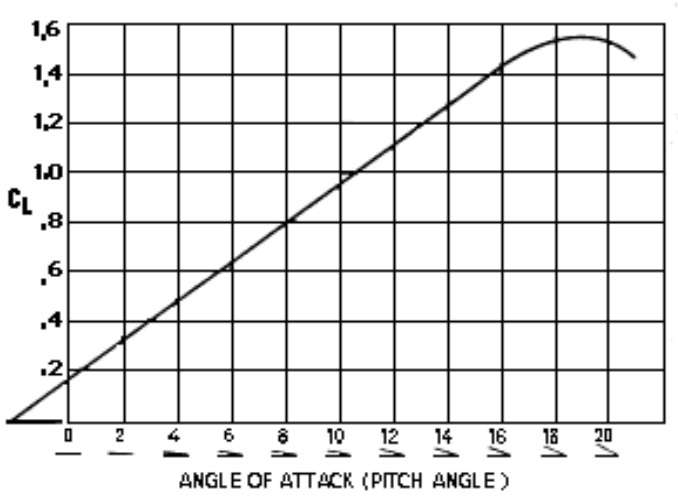

Ilustración 3.23: CL en función de α

Los valores de la gráfica no tienen por que corresponder con los nuestros, lo que sí tiene que ser parecido es la curva que forma la gráfica. Como podemos ver crece linealmente hasta los 19º de α, y a partir de ahí decrece. Para obtener un comportamiento realista en nuestro modelo tenemos que hacer que nuestros datos formen una curva parecida a esta, por lo tanto, añadiremos tres datos más de CL para un ángulo de ataque de 18º con CL=1 (siguiendo la tendencia de la recta de la gráfica que se ve un poco más adelante), y para 25º con CL=1 también, para que la curva establezca el máximo entre estos dos valores, que es aproximadamente en donde tiene que estar el máximo, y otro dato para 30º con CL=0,7476 siguiendo la tendencia de la curva. También añadimos otro dato para un ángulo de ataque de 45º para continuar la curva por la derecha, que vemos que tiene una gran pendiente negativa cuando entra en pérdida. Por último, añadimos un dato para un ángulo de -10º con CL=-1,2 y otro para -5º con CL=-0,5 (para que continúe la curva por la izquierda, siguiendo su tendencia) para reflejar la sustentación negativa cuando el ángulo de ataque es negativo menor de -3º, que es cuando empieza a existir sustentación negativa.

Después de las modificaciones nos queda una tabla como la siguiente:

α (º)	CL
-10	-1,2
-5	-0,5
0	0,27343
2	0,37099
4	0,46751
8	0,65613
10	0,7476
18	1
25	1
30	0,7476
45	-0,9

Tabla 3.16: CL en función de α finalmente

Si lo representamos gráficamente:

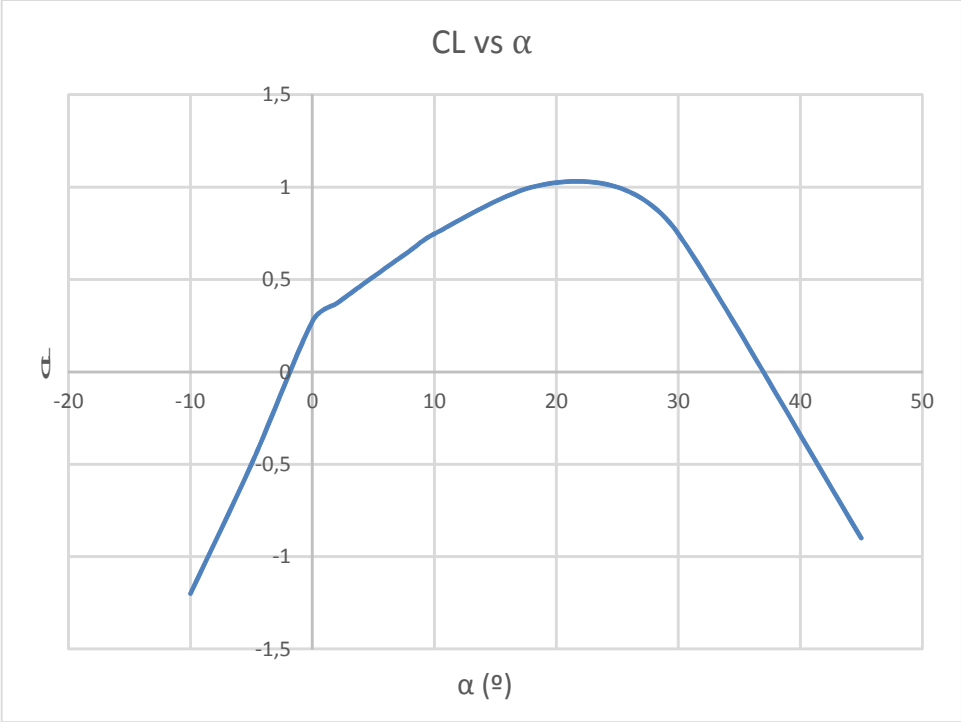

Ilustración 3.24: CL vs alfa

Probamos a realizar una nueva simulación en FlightGear con estos datos y verificamos que se comporta de forma parecida al modelo real, por lo que damos por buenos estos resultados y sus modificaciones.

- **CLde:** *Coeficiente de sustentación debido al ángulo del elemento de profundidad*

Es la sustentación generada por la inclinación del elemento de profundidad (elevator) que sirve para hacer que la aeronave realice un movimiento de cabeceo (suba o baje en el aire), y a su vez, genera una sustentación.

Los resultados fueron:

Ángulo de deflexión (º)	CL
10	0,151
20	0,3021
30	0,4533

Tabla 3.17: CL en función del ángulo de la profundidad

Estos valores forman una recta con una pendiente de valor de **0,8660**.

ALABEO (ROLL)

- **Clβ:** *Momento de alabeo debido al ángulo de deslizamiento lateral beta*

Durante el vuelo se considera que la dirección de vuelo no es totalmente recta, sino que existe una diferencia entre la dirección del viento y la del UAV. El UAV alabea hacia la dirección de deslizamiento lateral, debido principalmente al ángulo de dihedro, al ángulo de flecha y el flujo de aire inducido alrededor del fuselaje. El alabeo debido al deslizamiento lateral es proporcional al ángulo β (ángulo de deslizamiento lateral).

Los resultados obtenidos fueron:

β (º)	Cl
0	0
2	-0,00221
4	-0,00441
6	-0,00659
8	-0,00873

Tabla 3.18: Cl en función de beta

Estos valores forman una recta con una pendiente de valor **-0,0623**.

- **Clp:** *Momento de alabeo debido a la velocidad de alabeo*

Hace referencia al derivativo del amortiguamiento del movimiento de alabeo, es decir, el propio movimiento de alabeo genera un momento de alabeo debido a que el área de sustentación cambia al girar la aeronave cuando tenemos un ángulo de diedro, como es nuestro caso. Se puede observar en la siguiente imagen:

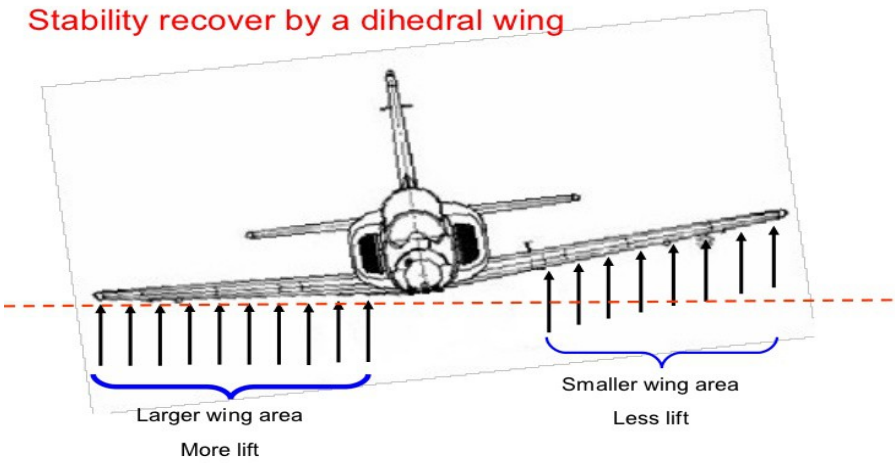

Ilustración 3.25: Efecto del ángulo de dihedro

- *Ángulo de diedro:* es el ángulo que forman las superficies inferiores de las alas con la horizontal. Este ángulo es negativo cuando el ángulo es hacia arriba (como en la imagen) y positivo cuando es hacia abajo. En nuestro caso tenemos un pequeño ángulo de diedro negativo de -2º.

Los resultados obtenidos fueron:

velocidad (rad/s)	Velocidad (º/s)	Cl
0,261799388	15	-0,1049
0,523598776	30	-0,2097
0,785398163	45	-0,3146
1,047197551	60	-0,4195

Tabla 3.19: Cl en función de la velocidad de alabeo

Estos valores forman una recta con una pendiente de valor **-0,4006.**

- **Clr:** *Momento de alabeo debido a la velocidad de guiñada*

Al girar la aeronave sobre el eje Z (guiñada), la presión ejercida por el viento sobre la superficie de la cola vertical y otras superficies laterales, genera un momento sobre el UAV (al no estar el centro de presiones en la misma horizontal que el centro de masas del UAV) que provoca un movimiento de alabeo.

Los resultados obtenidos fueron:

velocidad (rad/s)	Cl
0,261799388	0,02452
0,523598776	0,04904
0,785398163	0,07357
1,047197551	0,09810

Tabla 3.20: Cl en función de la velocidad de guiñada

Estos valores forman una recta con una pendiente de valor **0,0937**.

- **Clda:** *Momento de alabeo provocado por el ángulo del alerón*

Para hacer girar una aeronave sobre el eje X (alabeo) se inclinan los alerones, uno hacia arriba y otro hacia abajo, provocando el giro hacia el lado del alerón levantado. Este coeficiente considera el momento generado por los alerones para hacer girar la aeronave.

Los resultados obtenidos fueron:

Giro del alerón (º)	Cl
10	0,03529
20	0,07058
30	0,10587

Tabla 3.21: Cl en función del giro del alerón

Estos valores forman una recta con una pendiente de valor **0,2022**.

- **Cldr:** *Momento de alabeo provocado por el ángulo de la deriva*

Se trata del momento generado por la incidencia del viento sobre el elemento de deriva o "timón" de la aeronave cuando este se encuentra girado un cierto ángulo. No solo afecta a la guiñada, si no que también tiene influencia sobre el movimiento de alabeo.

Los resultados obtenidos fueron:

Giro de la profundidad (º)	Cl
10	0,00127
20	0,00253
30	0,0038

Tabla 3.22: Cl en función del giro de la profundidad

Estos valores forman una recta con una pendiente de valor **0,0072**.

CABECEO (PITCH)

- **Cmα:** *Momento de cabeceo debido al ángulo de ataque*

Es el momento que provoca la fuerza del viento debido a la inclinación del UAV (ángulo de ataque), que provoca un movimiento de cabeceo.

Los resultados obtenidos fueron:

α (º)	Cm
0	-0,06177
2	-0,0639
4	-0,06562
8	-0,06783
10	-0,0683
45	0,01415
90	0,09682

Tabla 3.23: Cm en función de alfa

Estos datos no forman una recta, por lo que se introduce como una tabla de datos en el archivo.

- **Cmde:** *Momento de cabeceo provocado por el elemento de profundidad*

Hace referencia al momento generado por la inclinación de la profundidad, situada en la parte trasera del UAV y cuya misión es la de hacer que ascienda o descienda. Es el elemento que más influencia tiene en el movimiento de cabeceo.

Los resultados obtenidos fueron:

Giro de la profundidad (º)	Cm
10	-0,07262
20	-0,14551
30	-0,21866

Tabla 3.24: Cm en función del giro de la profundidad

Estos valores forman una recta con una pendiente de valor **-0,4184**.

- **Cmq:** *Momento de cabeceo debido al la velocidad de cabeceo*

Expresa el cambio en el momento de cabeceo provocado por el propio movimiento de cabeceo, es

decir, cuando la aeronave realiza un movimiento de cabeceo (gira sobre el eje "Y"), el viento incidirá con un cierto ángulo sobre la parte superior o inferior del fuselaje y las alas, provocando un momento que tendrá efecto sobre el movimiento de cabeceo.
Los resultados obtenidos fueron:

Velocidad (rad/s)	Cl
0,261799388	-0,27998
0,523598776	-0,56502
0,785398163	-0,85509
1,047197551	-1,1502

Tabla 3.25: Cl en función de la velocidad de cabeceo

Los valores forman una curva que se puede aproximar a una recta con una pendiente de valor **-1,1080**.
Tras realizar las primeras pruebas en el simulador, y usando el método de sustitución y prueba, es decir, sustituyendo uno a uno los valores calculados en AVL por los aportados en la documentación y simulando de nuevo en FlightGear, determinamos que este valor era el causante del comportamiento extraño del UAV, provocaba una gran inestabilidad durante el vuelo que imposibilitaba su control. Si lo comparamos con el parámetro de uno de los modelos de UAV de características similares que aparece en la documentación del programa (valor=12) vemos que el valor de nuestro resultado es aproximadamente un orden de magnitud inferior. Si multiplicamos por 10 nuestro resultado (**11,080**), lo sustituimos y hacemos la simulación, vemos que el comportamiento del UAV es mucho más estable, y por lo tanto, este valor era el causante de la irregularidad.

GUIÑADA (YAW)

- **Cnb:** *Momento de guiñada debido al ángulo de deslizamiento lateral β*

El viento lateral incidente con ángulo beta ejerce una fuerza sobre las superficies laterales del UAV, como pueden ser las colas verticales o los winglets, que provoca un momento que a su vez genera un giro de guiñada (eje "Z").
Los resultados obtenidos fueron:

β (º)	Cn
0	0
2	0,00053
4	0,00106
6	0,00158
8	0,0021

Tabla 3.26: Cn en función de beta

Estos valores forman una recta con una pendiente de valor **0,0150**.

- **Cnr:** *Momento de guiñada debido a la velocidad de guiñada*

Al realizar un movimiento de guiñada, el viento que inicialmente incidía de frente, ahora incide con un ángulo que va creciendo según va girando el UAV sobre el eje "Z", o bien, si el viento incidía con un ángulo beta, ahora incidirá con un ángulo mayor o menor dependiendo del sentido de giro. Por esto es necesario considerar el momento de guiñada debido al propio movimiento de guiñada.
Los resultados obtenidos fueron:

Velocidad de guiñada (rad/s)	Cl
0,261799388	-0,00289
0,523598776	-0,00578
0,785398163	-0,08808
1,047197551	-0,01155

Tabla 3.27: Cl debido a la velocidad de guiñada

Estos valores forman una recta con una pendiente de valor **-0,0110**.

- **Cndr:** *Momento de guiñada provocado por el ángulo de la deriva*

La deriva es el principal elemento generador de guiñada del UAV, al estar situado en la parte trasera de las colas verticales, cuando se giran estas superficies hacia uno de los lados, el viento que incide sobre ellas provoca un momento gracias al cual la aeronave puede realizar el movimiento de guiñada.
Los resultados obtenidos fueron:

Giro de la deriva (º)	Cn
10	-0,00208
20	-0,00416
30	-0,00624

Tabla 3.28: Cn en función del giro de la deriva

Estos valores forman una recta con una pendiente de valor **-0,0119**.

Estos son todos los coeficientes necesarios para la creación del archivo que determinará la dinámica de vuelo en el simulador. Los valores aquí expuestos no los hemos incluido en ningún anexo porque como veremos más adelante, tras las pruebas de vuelo que hicimos para comprobar la similitud entre el modelo del simulador y el real, hemos tenido que modificar algunos de estos parámetros para conseguir un mayor realismo. El archivo definitivo se puede ver en el ANEXO III: “Fichero de aerodinámica”.

3.3.4. Tren de aterrizaje e inercias

Por último, para completar nuestro UAV virtual, necesitamos un archivo que reúna las características geométricas principales del UAV, tenga definido un tren de aterrizaje que sirva de apoyo durante el despegue y aterrizaje, contenga las inercias, que tienen gran influencia en el comportamiento durante el vuelo, y declarar las posiciones en las que actúan los motores y su sentido de giro. Este archivo sirve como "coordinador" de los otros 3, ejerce de archivo principal que enlaza al resto.

Este archivo ha sido creado siguiendo la estructura de otros encontrados en la documentación de FlightGear, y sustituyendo en él los datos de nuestro UAV. Este archivo se puede consultar en el ANEXO IV: "Fichero de tren de aterrizaje e inercias"

Lo primeros datos a introducir en el cabecero del fichero es el nombre del autor, fecha versión y descripción del modelo a crear.

A continuación se describen algunos datos sobre la geometría del UAV, medidos directamente sobre nuestro modelo, que es necesario introducir:

- **Área alar** (*wingarea*): 1,1534 m^2. Es la superficie proyectada del modelo (visto desde arriba).
- **Envergadura** (*wingspan*): 1,94 m.
- **Incidencia del ala** (*wingincidence*): 0º. Es el ángulo con el que incide el UAV con respecto a la horizontal en el simulador.
- **Cuerda** (*chord*): 0,56 m. Necesita el valor de la longitud de la cuerda en la zona central del UAV. Medida directamente sobre el UAV.
- **Área de la cola horizontal** (*htailarea*): 0,0984 m^2. Hace referencia a la superficie de la cola horizontal del UAV. En nuestro caso no tenemos cola, pero si las superficies de control que controlan el movimiento de cabeceo, llamadas elevones (que además realizan el alabeo). Ésta será la superficie que consideramos para este dato.
- **Brazo de la cola horizontal** (*htailarm*): 0,372 m. Es la distancia desde el centro de masas del UAV hasta el centro de presiones de los 4 elementos de profundidad, en nuestro caso son los elevones.
- **Área de cola vertical** (*vtailarea*): 0,0627 m^2. Considera el área de las 2 colas verticales que

tenemos en el UAV, incluidas las superficies de las derivas.

- **Brazo de la cola vertical** (*vtailarm*): 0,365 m. Es la distancia desde el centro de masas hasta el centro de presiones de las derivas.
- **Punto de Referencia Aerodinámica** (*AERORP*): es el punto sobre el cual se aplican las fuerzas y momentos generados por las superficies y superficies de control declaradas anteriormente, por lo tanto este punto será el centro de masas del UAV. Sus coordenadas son:
  - X = 0,254 m
  - Y = 0 m
  - Z = 0 m

  Este punto ha sido medido empíricamente sobre el UAV, ya que su geometría no permitía calcularlo de otra forma. Hemos buscado el punto de equilibrio del UAV con todas sus masas (baterías, motores, etc.) colocándolo encima de una pequeña base circular de aproximadamente 2 cm de diámetro hasta encontrar el punto en donde estaba en equilibrio. El centro de masas coincide aproximadamente con el centro de esta circunferencia, resultando ser el valor anterior.
- **"Punto de vista del piloto"** (*EYEPOINT*): punto desde el cual el piloto vería desde dentro del UAV. En nuestro caso lo hacemos coincidir con el centro de masas, al igual que antes, porque aunque no llevamos piloto, el programa requiere situar este punto.
- **Punto de Referencia Visual** (*VRP*): es un punto que sirve para conciliar el sistema de coordenadas del sistema mecánico y el sistema de coordenadas del modelo gráfico. Por norma general en aeronáutica, este punto se sitúa en el morro de la aeronave a modelar, por lo que en nuestro caso las coordenadas de este punto serán:
  - X = 0 m
  - Y = 0 m
  - Z = 0 m
- **Balance de masas:** en este apartado se declaran las inercias, el peso total, el centro de masas y la carga y su punto de aplicación sobre el UAV.
  - Inercias: es la facilidad con que un cuerpo realiza un giro, y son una característica

fundamental en aeronaves. Tiene gran importancia para nosotros porque influye mucho en el comportamiento del UAV en el simulador. Al ser una geometría compleja no es posible calcularlo teóricamente, pero sí podemos usar métodos experimentales para su cálculo. Usaremos un péndulo bifilar para hallarlas, el proceso completo así como la justificación de los resultados se puede ver en el ANEXO VI: Cálculo de inercias

Inercia	Valor (kg*m^2)
Ixx	1,5956
Iyy	0,4208
Izz	1,7366

Tabla 3.29 Inercias

- Peso y punto de aplicación: el peso en vacío del UAV es de 3 kg. Este peso tiene en cuenta todos los elementos que porta normalmente cuando se hace volar (baterías, motores, hélices, servomotores, cámara, etc.). El punto de aplicación es el centro de masas, cuyas coordenadas ya sabemos.
- Carga y punto de aplicación: no portamos ningún tipo de carga, por lo que el valor de esta masa es 0 kg. El punto de aplicación es indiferente al ser la masa nula, lo establecemos en el centro de masas igualmente.

- **Tren de aterrizaje** (*ground_reacctions*): Necesitamos declarar unos puntos de contacto entre el UAV y el suelo que servirán como tren aterrizaje en la simulación. El UAV real no dispone de tren de aterrizaje propiamente dicho, si no que aterriza sobre el fuselaje. Para la simulación, establecemos estos puntos para que durante el despegue o el aterrizaje el UAV tenga una cierta estabilidad. En nuestro caso tendremos 3 puntos de contacto virtuales (llamados “BOGEY” en el código) entre el UAV y el suelo, uno en la punta trasera del ala izquierda (“LEFT_MLG”), otro en la punta trasera del ala derecha (“RIGHT_MLG”) y otro en el morro del UAV (“NOSE_LG”). Las coordenadas de estos puntos se miden en unidades de pulgadas (") y medidas directamente sobre el UAV (salvo la coordenada Z que se coloca aproximadamente para situar el UAV con un cierto ángulo de ataque en reposo) resultaron ser:

  1. LEFT_MLG: en la punta trasera del ala izquierda
     - X = 28"

- Y = -38"
- Z = -6"

2. RIGHT_MLG: en la punta trasera del ala derecha
   - X = 28"
   - Y = 38"
   - Z = -6"
3. NOSE_LG: en el centro del morro
   - X = 0"
   - Y = 0"
   - Z = -10"

Pero además de establecer los puntos, es necesario añadirle algunas características físicas que determinarán la forma de comportarse de estos puntos de contacto:

- **Coeficiente de fricción estática** (*static_friction*): le asignamos el mismo valor para los 3 puntos de contacto suponiendo que son de caucho y va a aterrizar sobre cemento seco. Su valor es 1.
- **Coeficiente de fricción dinámica** (*dynamic_friction*): haciendo la misma suposición que antes, el coeficiente de fricción dinámica caucho - cemento seco es 0,8
- **Coeficiente de fricción de rodadura** (*rolling_friction*): suponiendo lo mismo que en el coeficiente anterior, su valor es de 0,02.
- **Coeficiente de rigidez** (*spring_coeff*): para el caucho es de $4{,}1 \cdot 10^9$ Pa, que en unidades del sistema internacional es $4{,}1 \cdot 10^9$ kg/cm2. En nuestro caso no tenemos superficie de contacto, ya que hemos supuesto un contacto puntual que en el simulador es ideal, por lo tanto debemos considerar un coeficiente de rigidez en unidades de kg/cm, como si fuese un muelle. Este muelle debemos considerar que no tiene un coeficiente de rigidez ni un coeficiente de amortiguamiento muy alto, porque el modelo real aterriza sobre su propio fuselaje, hecho de poliestireno y rodeado de una capa de proliporileno (normalmente sobre hierba) y no rebota excesivamente. Tampoco queremos un rebote excesivo en el simulador para no tener comportamientos raros durante las maniobras de aterrizaje y despeje. Asignamos un coeficiente de 60 lb/ft para los 3 apoyos.

- **Coeficiente de amortiguamiento** (*damping_coeff*): por las razones explicadas en el apartado anterior, tampoco queremos un valor demasiado alto de este coeficiente, por lo tanto su valor para los 3 puntos de reacción será 10 lb/ft·s.

- **Sistema de propulsión** (*propulsion*): hasta ahora hemos visto como hemos creado los archivos que contienen los modelos de motor y hélices, pero es necesario darles una posición, orientación y sentido de giro. Los 2 motores que tenemos están ubicados a 25 cm a los lados desde el medio del UAV y sobresalen hacia delante del borde de ataque de las alas, quedando las hélices en el mismo plano que el morro del UAV. Por lo tanto las coordenadas tanto de motores como de hélices son

  - X = 0 m
  - Y = 0,25 m (motor y hélice derecha) y -0,25 m (motor y hélice izquierda)
  - Z = 0 m

  Propulsan completamente en el eje X, por lo que no están girados en ninguno de los ejes:

  - Roll = 0º
  - Pitch = 0º
  - Yaw = 0º

  El sentido de giro es antihorario para el motor de la derecha (visto desde atrás) y horario para el motor izquierdo.

- **Control de vuelo** (*flight_control*): finalmente, se han programado los controles de las superficies de control del UAV. En esta parte controlan las derivas y se integran los alerones (ailerons) y los elementos de profundidad (elevators) en un único elemento al que llamamos elevones, como ya hemos visto, de forma que las señales de entrada que llegan desde el teclado o ratón durante la simulación se transformen en una señal y se giren estas superficies en el sentido adecuado para conseguir el giro deseado. Además tienen integrados “trims” que sirven para corregir la posición de cada superficie de control en caso de que existan desviaciones de su inclinación.

4. ANALISIS Y PRUEBAS

Una vez que tenemos un modelo virtual del UAV listo para ser usado en el simulador, hechas ya varias simulaciones, y comprobado que los valores obtenidos del simulador son globalmente cercanos o similares a los obtenidos en los vuelos reales, es necesario validar nuestro trabajo con datos, comparando los datos obtenidos durante vuelos de prueba con el UAV real con los datos exportados durante vuelos del UAV en el simulador.

Para obtener datos de vuelos reales, y respetando la legislación vigente sobre vuelo de Vehículos Aéreos No Tripulados, hemos ido a la pista del Club de Aeromodelismo "A Pombiña" situada en Narón para realizar nuestros vuelos de prueba.

Llevamos una unidad de UAV equipado con una cámara de vídeo, un tubo de Pitot para medir la velocidad relativa con respecto al aire, un altímetro barométrico para poder medir la altitud de vuelo y un controlador de velocidad con un PID para controlar la velocidad del UAV en función de nuestras necesidades. Todos los datos medidos por estos sensores, necesarios para nuestro análisis, se transmiten mediante un enlace de radiofrecuencia de largo alcance a 868 MHz hasta una estación en tierra receptora, diseñada y construida por Daniel García Gonçalves en un proyecto anterior a este de esta misma línea de investigación.

Ilustración 4.1: Equipo de UAVs y estación de control en tierra

Para realizar el análisis, primero necesitamos diseñar unas pruebas de vuelo que sirvan para comparar diferentes parámetros de forma sencilla. Hemos diseñado 4 pruebas:

1. Prueba 1: realizaremos un vuelo a velocidad constante con el motor al 100% de su capacidad, y medimos el tiempo necesario para que el UAV ascienda una determinada altura, y mediremos el tiempo de subida.
2. Prueba 2: realizaremos un vuelo con el motor al 0% (apagado) y velocidad constante, dejando descender el UAV una altura determinada y mediremos el tiempo necesario para la bajada.
3. Prueba 3: realizaremos un vuelo a altura constante, con el motor al 38% de su capacidad y mediremos la velocidad que es capaz de alcanzar.
4. Prueba 4: realizaremos un vuelo en el que se haga un giro de 180º en el aire, con los alerones girados un 10% a velocidad constante con el motor al 38% de su capacidad y altura constante.

La información obtenida durante los vuelos se ha exportado a hojas de Excel para poder trabajar con los datos más fácilmente, y visualizarlos mejor. A continuación se muestran algunos gráficos en donde se comparan algunos parámetros de vuelo con respecto al tiempo.

4.1. Vuelos con el UAV real

4.1.1. Primer vuelo: subida y bajada

Los datos de este primer vuelo, los podremos usar para realizar las pruebas 1 y 2, porque en el se ha sometido al UAV a un ascenso y a un descenso con las características anteriormente citadas para cada prueba.

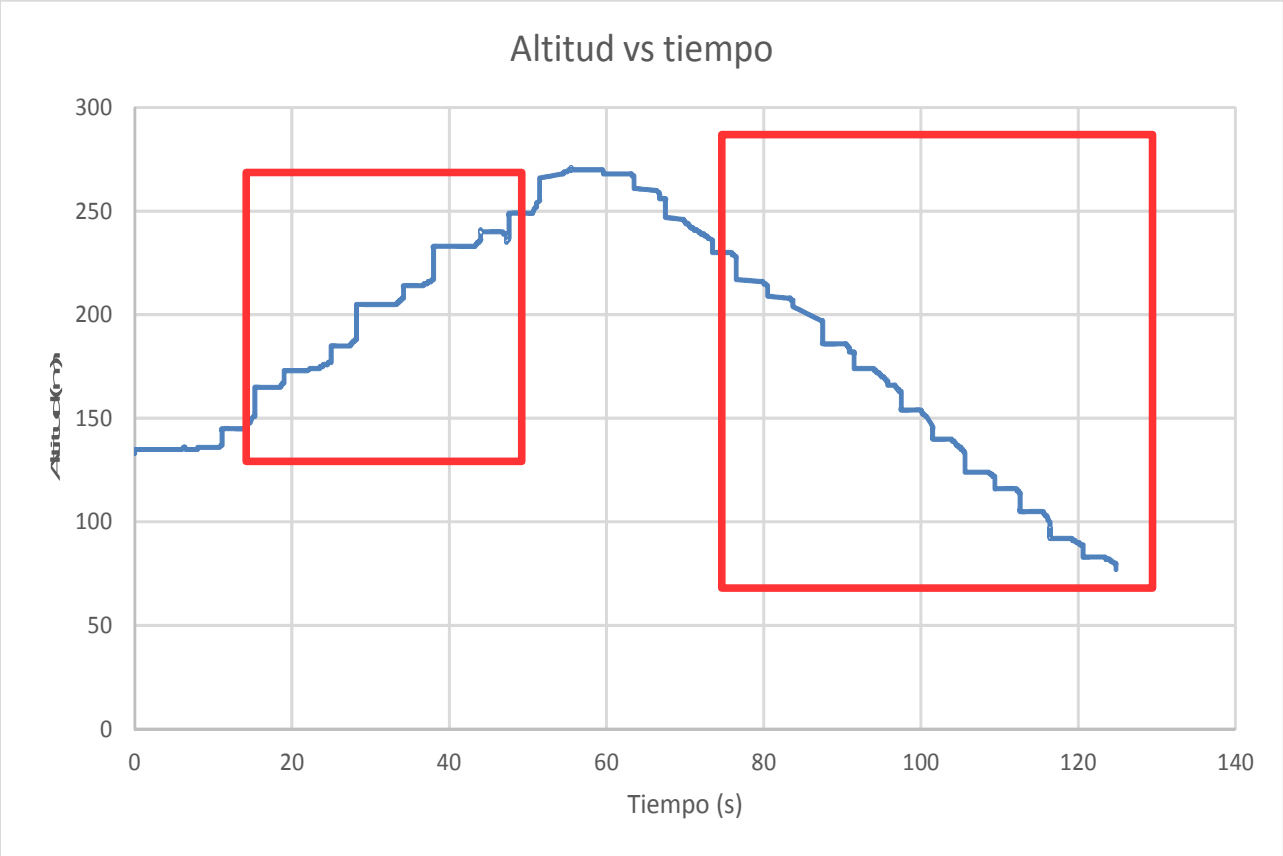

Ilustración 4.2: Altitud vs tiempo

En la gráfica anterior podemos observar la altitud (en metros) representado frente al tiempo (en s). Vemos que se ha llevado el UAV hasta una altitud de 133 m y luego se ha hecho ascender hasta los 271 m, esta ascendencia se produce con el motor al 100% de su capacidad y a velocidad constante como requiere la prueba. Una vez alcanzada la altura máxima, se apagan los motores y se deja descender hasta los 80 m con el motor apagado.

Para la Prueba 1 tomaremos los datos que están dentro del recuadro pequeño, mientras que para la Prueba 3 tomaremos los datos incluidos en el recuadro grande.

La forma de "pequeños escalones" que se observa en la gráfica es debido a que son momentos en

los que la radio está enviando datos en vez de recibirlos. Tambien existen momentos en los que los que el GPS tarda en actualizar el tiempo y envía varios datos diferentes para un mismo instante, por eso existen subidas verticales en la gráfica.

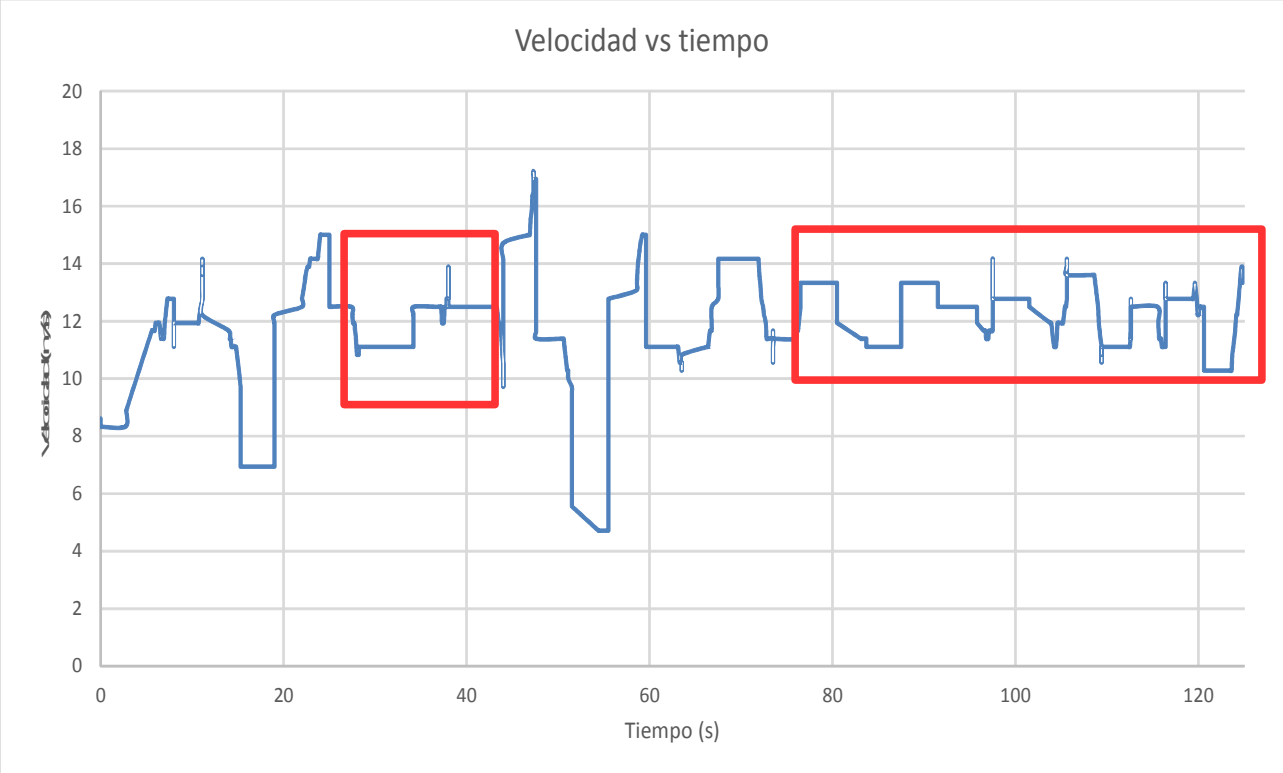

Ilustración 4.3: Velocidad vs tiempo

En el gráfico anterior se puede ver representada la velocidad del UAV (en m/s) frente al tiempo (en s) a lo durante el vuelo. Pese a que a simple vista pueda parecer que la velocidad no es constante, sí lo es en algunos tramos. El tramo del recuadro de la izquierda corresponde al tiempo de ascendencia del UAV a velocidad constante. La velocidad la medimos mediante un tubo de Pitot, lo que proporciona la velocidad relativa del UAV con respecto al aire, la variación de velocidad que se puede ver en los 2 recuadros dibujados se debe a rachas de viento, que falsean nuestras mediciones, lo sabemos porque son pequeños cambios en la velocidad de 2-3 m/s aproximadamente. Los cambios de la velocidad que se pueden ver justo a la izquierda del recuadro izquierdo y entre los 2 recuadros sí que son cambios provocados.

El recuadro de la izquierda corresponde al período de ascendencia que usaremos para la Prueba 1 y el recuadro de la derecha corresponde al período de descendencia, que usaremos para la Prueba 2.

Cuando hagamos las comparaciones entre los vuelos del modelo real y en el simulador, tomaremos solamente los rangos de datos contenidos en los recuadros para centrarnos en cada parte.

4.1.2. Segundo vuelo: altura constante

Los datos del 2º vuelo nos servirán para la Prueba 3, porque en él hemos mantenido la altura de vuelo constante, con el motor al 38,75% de su capacidad.

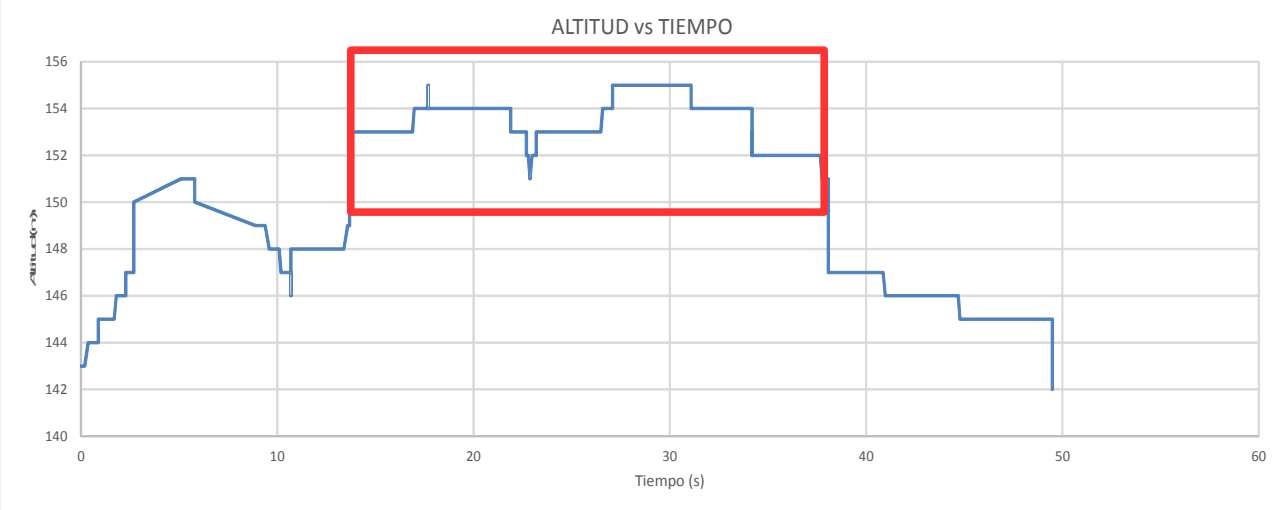

Ilustración 4.4: Altitud vs tiempo

En este vuelo, si medimos la altura vemos que la altura más baja es de 145 m y la más alta de 155 m aproximadamente. Como los extremos de la gráfica muestran una bajada de la altura, y lo que buscamos es altura constante, solo tomaremos el rango de datos del recuadro rojo, que corresponde al intervalo entre los 13,7 y los 37,7 segundos para nuestro análisis. En este rango de valores la altitud media es de 153,4 m.

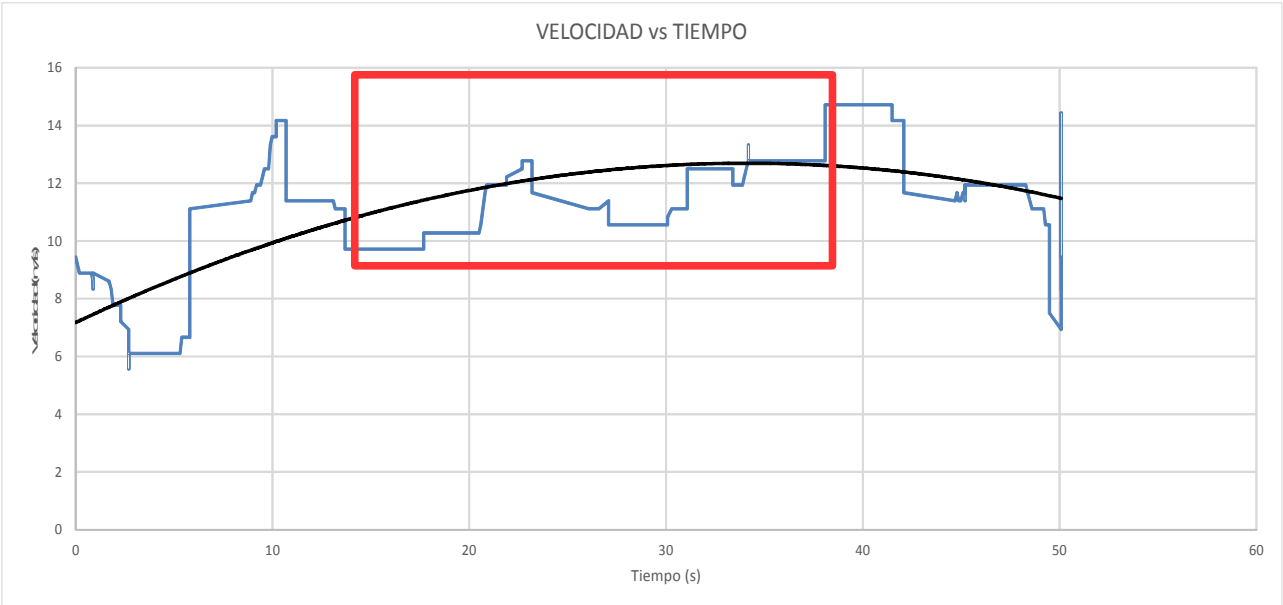

Ilustración 4.5: Velocidad vs tiempo

En la gráfica de la velocidad con respecto al tiempo sucede lo mismo, no es perfectamente constante durante todo el vuelo debido a las rachas de viento, por lo que tomaremos solo el rango correspondido entre los segundos 13,7 y 37,7 para realizar el análisis ya que es el rango en donde la velocidad se mantiene más constante.

4.1.3. Tercer vuelo: giro

Por último, para realizar la Prueba 4, necesitamos realizar un vuelo en el que hagamos un cambio de rumbo de 180º en la trayectoria, con los alerones girados un 10% y manteniendo la potencia del motor. Con esto podemos observar la caída que sufre el UAV durante el giro y el tiempo necesario para completarlo.

Desgraciadamente, debido a la falta de tiempo y a las condiciones meteorológicas adversas durante la semana anterior a la entrega del proyecto, no hemos podido realizar este vuelo, por lo que no tendremos una 4ª prueba en la que comparar el comportamiento durante el giro. En los otros 2 vuelos del UAV se realizaron cambios de rumbo, pero de forma descontrolada, sin mantener la potencia del motor constante y sin controlar que los alerones estuviesen girados un 10% aproximadamente, por lo que no podemos usar los datos de esos vuelos para esta prueba.

Nos conformaremos con la comparación visual durante el vuelo y el criterio personal para validar el modelo en maniobras de cambios de rumbo.

4.2. Vuelo en el simulador

Para realizar los vuelos en el simulador de forma que se respestase los parámetros de los vuelos reales en el simulador, hemos usado un software desarrollado en JAVA y ejecutado en Netbeans IDE 8.1 (de código libre), que al mismo tiempo ejecuta FlightGear, y muestra todos los parámetros de vuelo (altura, velocidad, etc.) por pantalla durante la simulación, de modo que podemos ajustarlos de forma precisa a las condiciones de vuelo que necesitamos. Posteriormente se guardan en un archivo todos estos parámetros medidos durante el vuelo y desde este archivo copiamos la información a una hoja de Excel en donde es más fácil de tratar para analizarlo posteriormente.

4.2.1. Vuelo: subida, bajada y altura constante

Para poder comparar los vuelos del UAV real y el virtual, debemos realizar vuelos los más semejantes posible, y analizar los resultados. Por ello, hemos realizado un vuelo en donde provocamos una ascendencia, una descendencia y un vuelo a altura constante del UAV, respetando

los parámetros del vuelo real.

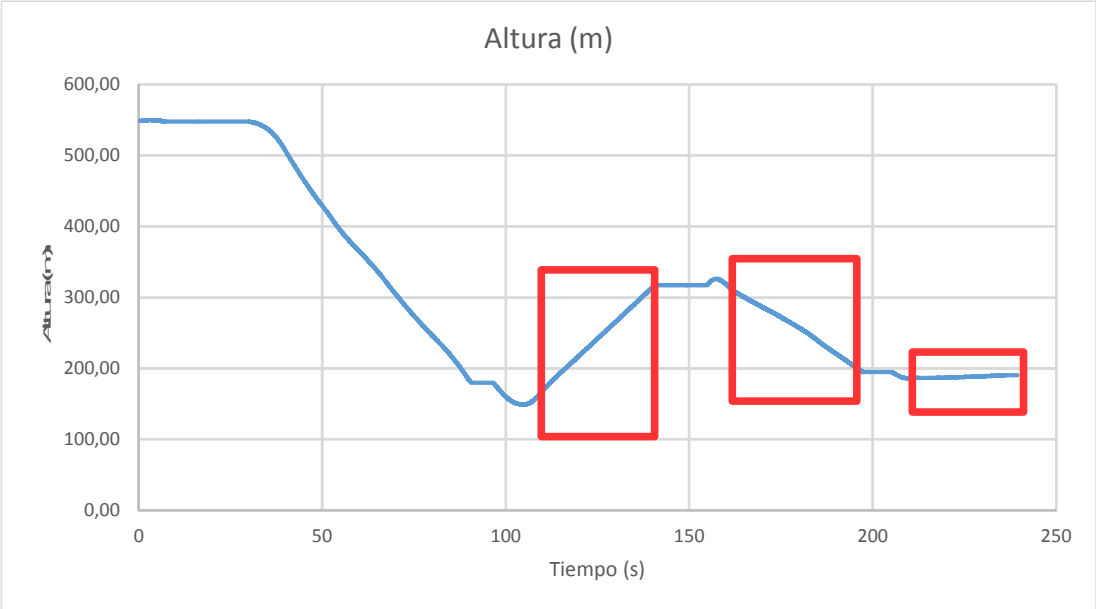

Ilustración 4.6: Altura vs tiempo: subida, bajada y altura cosntante

Inicialmente, FlightGear nos sitúa a una altura de 550 m aproximadamente, a mayor altitud, mayor energía tiene el sistema, para trabajar en unas condiciones semejantes a las del vuelo real, debemos realizar las pruebas a una altura aproximada, por este motivo descendemos el UAV hasta los 149 m, y a partir de ahi realizamos nuestras pruebas.

Para la prueba de ascendencia, tomamos el intervalo que va desde los 112 s hasta los 140 s, en donde se asciende desde los 179 m hasta los 314 m. Para el ensayo de descenso tomamos los valores que van desde el los 168 s hasta los 199 s en donde se desciende desde los 292 m hasta los 195 m. Por último para el ensayo de vuelo a altura constante, tomamos el tramo comprendido entre los segundos 211 y 239, en donde volamos a una altura media de 188,24 m.

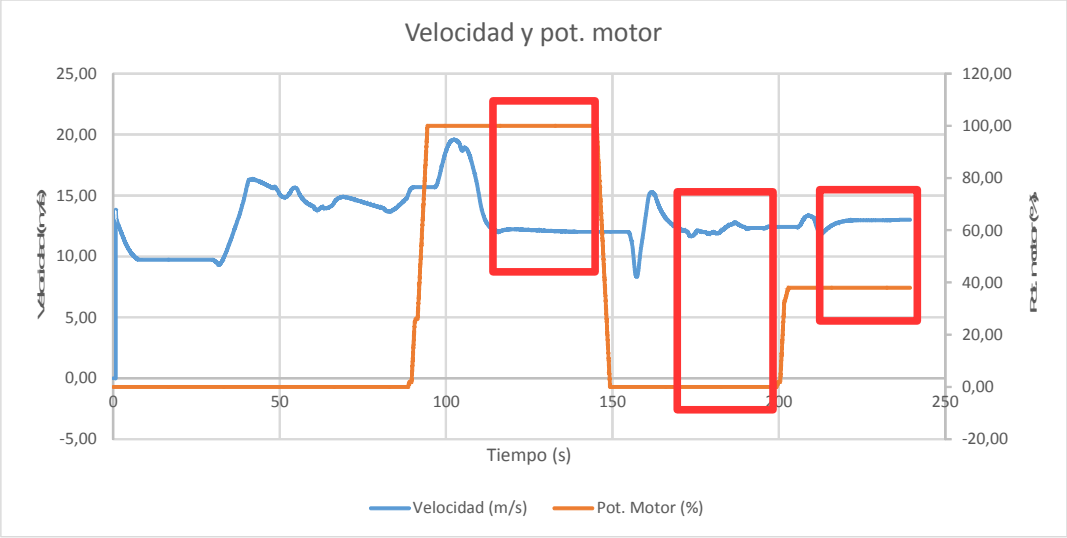

Ilustración 4.7: Velocidad y potencia del motor en cada tramo

En este gráfico en donde representamos la velocidad y la potencia del motor frente al tiempo podemos ver las condiciones de velocidad y potencia a las que realizamos la prueba de vuelo en los diferentes tramos. Lo hemos hecho lo más parecido posible a los vuelos con el UAV real.

Durante el tramo de ascendencia, el motor funcionaba al 100% de su potencia y volaba a una velocidad media de 12,15 m/s. Durante el descenso, el motor estaba apagado, es decir al 0% y bajaba con una velocidad de 12,24 m/s. Finalmente, en el tramo de vuelo a altura constante, el motor funcionaba al 38% de su capacidad y la velocidad media fue de 12,81 m/s.

4.3. Pruebas y análisis

Una vez vistos los vuelos que hemos realizado y sus características, podemos comenzar a comparar los valores de estos vuelos del UAV real con los resultados del simulador.

4.3.1. Prueba 1

Para realizar esta prueba necesitamos un vuelo a velocidad constante y con el motor al 100% de su capacidad. Como hemos visto, tomaremos los datos correspondiente a la zona de ascendencia del primer vuelo, y más concretamente de la zona en donde la velocidad es más uniforme. Este período corresponde al intervalo de tiempo comprendido entre los 28 y los 44 segundos del tramo de vuelo que estamos analizando

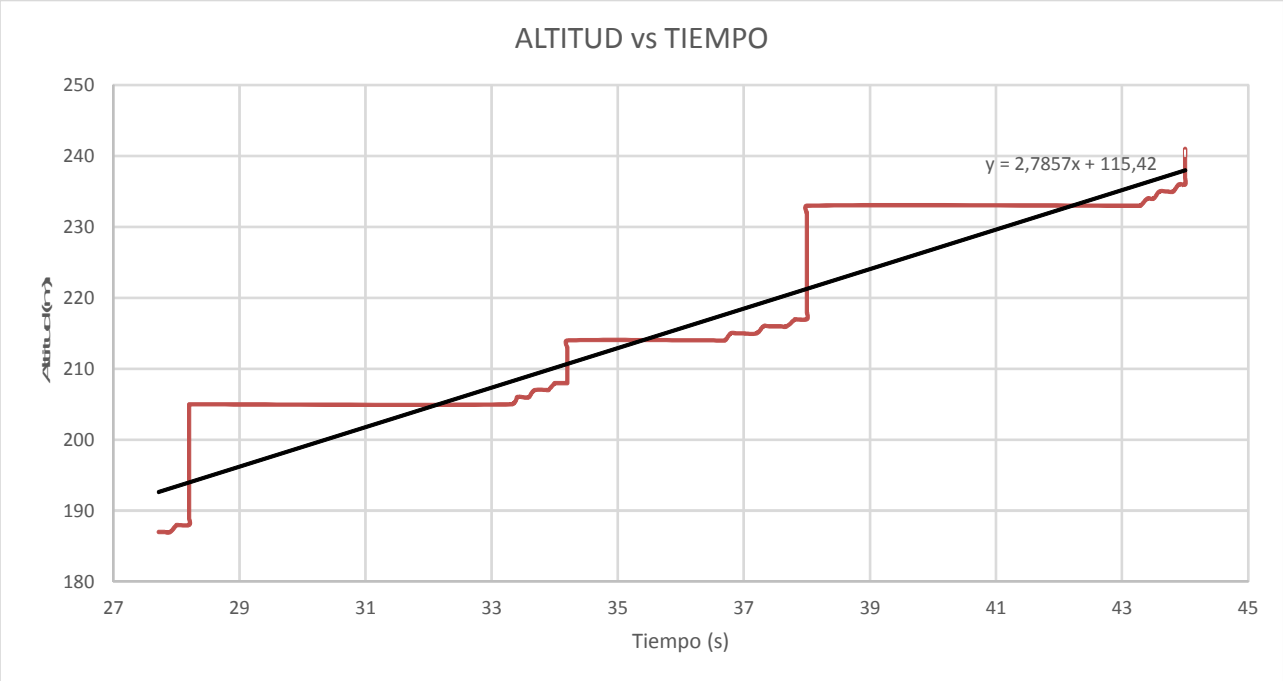

Ilustración 4.8: Altitud vs tiempo en el tramo de ascendencia

Aquí vemos como ha sido el aumento de la altitud del UAV durante estos 16 segundos, observamos como ha ascendido 53 metros en ese tiempo. La forma escalonada del gráfico se debe a la frecuencia de recepción de datos, que no es perfecta, pero si agregamos una línea de tendencia y su ecuación, vemos que tiene una pendiente de 2,7857, lo que indica que asciende aproximadamente a 2,8 metros verticales por segundo.

Si nos fijamos ahora en la velocidad del UAV en este intervalo, seguimos observando variaciones pequeñas en la velocidad (2 m/s), causadas por el viento. La velocidad media durante este intervalo es de 11,82 m/s, esta será la velocidad aproximada que usaremos en el simulador.

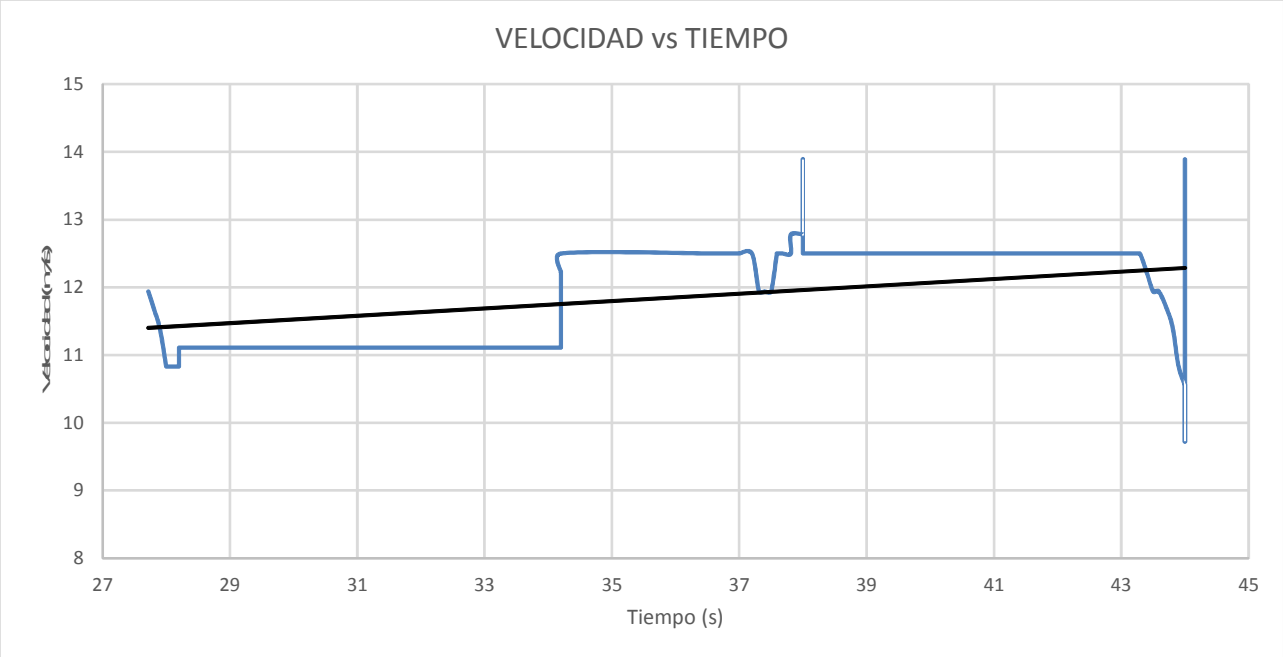

Ilustración 4.9: Velocidad vs tiempo en el tramo de ascendencia

Si ahora nos fijamos en los datos exportados del simulador:

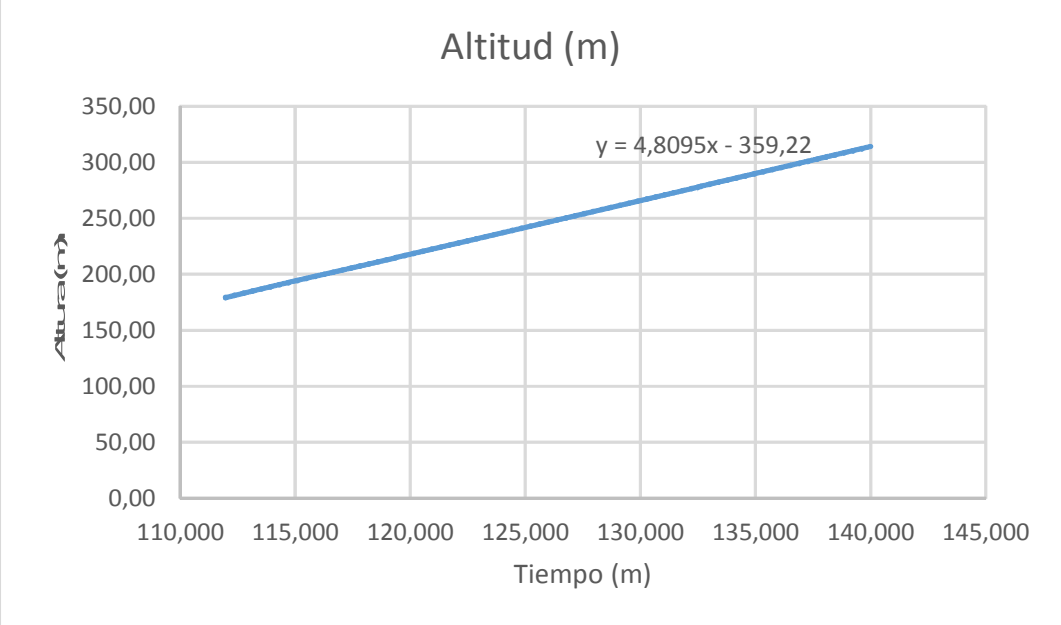

Ilustración 4.10: Altura vs tiempo durante subida en simulador

A una velocidad media de 12,15 m/s, vemos que el UAV ha ascendido 135 m en 28 segundos, lo que nos da una pendiente de 4,8095. Esto significa que el UAV asciende 4,8095 metros en vertical por segundo. Un valor muy isuperior a los 2,7857 m/s del UAV real.

4.3.2. Prueba 2

Esta prueba consiste en apagar los motores del UAV durante el vuelo y dejarlo caer a velocidad constante una cierta distancia y medir el tiempo necesario. Para ello tomamos los datos correspondientes al período de descenso del primer vuelo, que cumple con estas condiciones. Tomando sólo e rango de datos que nos interesa:

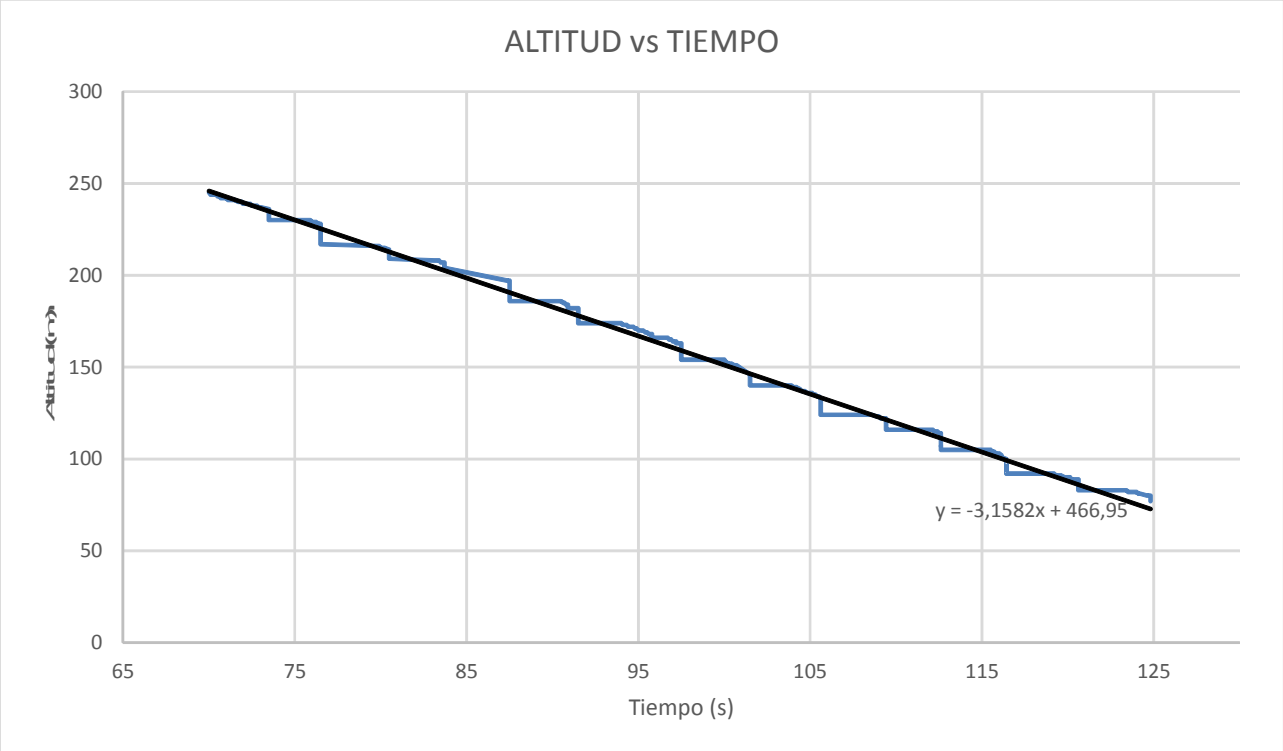

Ilustración 4.11: Altitud vs tiempo durante el descenso

En este caso vemos una gráfica sin escalones tan grandes como en el caso anterior, esto se debe a que tanto el tiempo de descenso (unos 55 s) como la distancia de descenso (unos 168 m) son mayores que antes. Si nos fijamos en la ecuación de la línea de tendencia, observamos que la pendiente es -3,1582, lo que nos indica que desciende aproximadamente unos 3,2 metros verticales por segundo.

La velocidad es constante, aunque nos encontramos con las mismas variaciones de antes debido al viento. En este caso la velocidad media es de 12,3 m/s, por lo que será la velocidad que usaremos en el simulador.

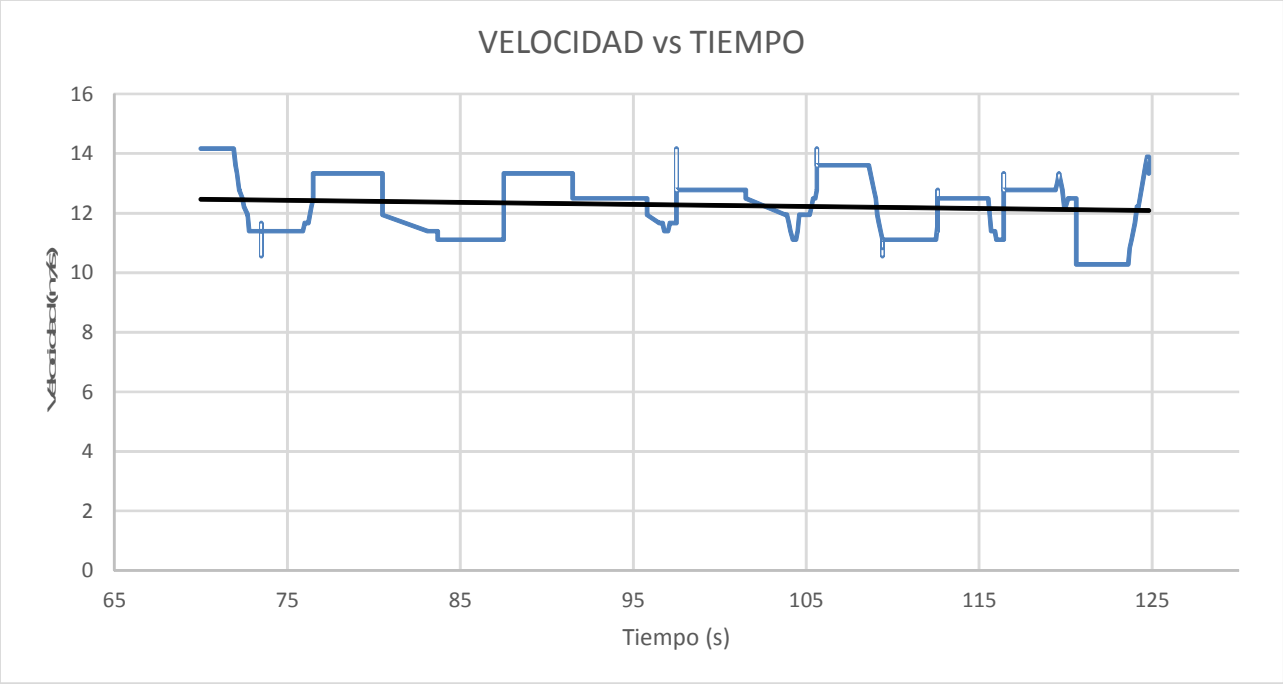

Ilustración 4.12: Velocidad vs tiempo durante el descenso

Si lo comparamos con los datos exportados del simulador:

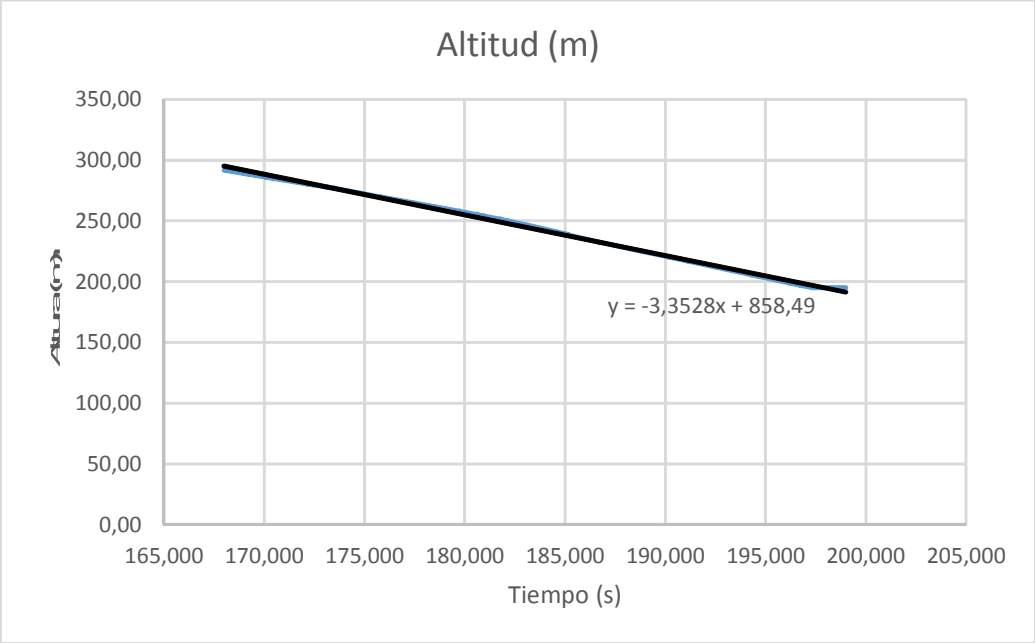

Ilustración 4.13: Altura vs tiempo durante el descenso en simulador

Durante el tramo de bajada, a una velocidad media de 12,24 m/s hemos descendido 97 m en 31 s, lo que nos da una pendiente de -3,3528. Nuestro UAV virtual desciende 3,3528 metros verticales por segundo, un valor un poco superior a los 3,1582 m/s del UAV real.

4.3.3. Prueba 3

La Prueba 3 consiste en realizar un vuelo a altitud constante y con el motor al 38,75 % de su capacidad, par ver cual es la velocidad que es capaz de alcanzar el UAV con estas condiciones. Para analizar esto tomamos los datos del segundo vuelo y los representamos gráficamente.

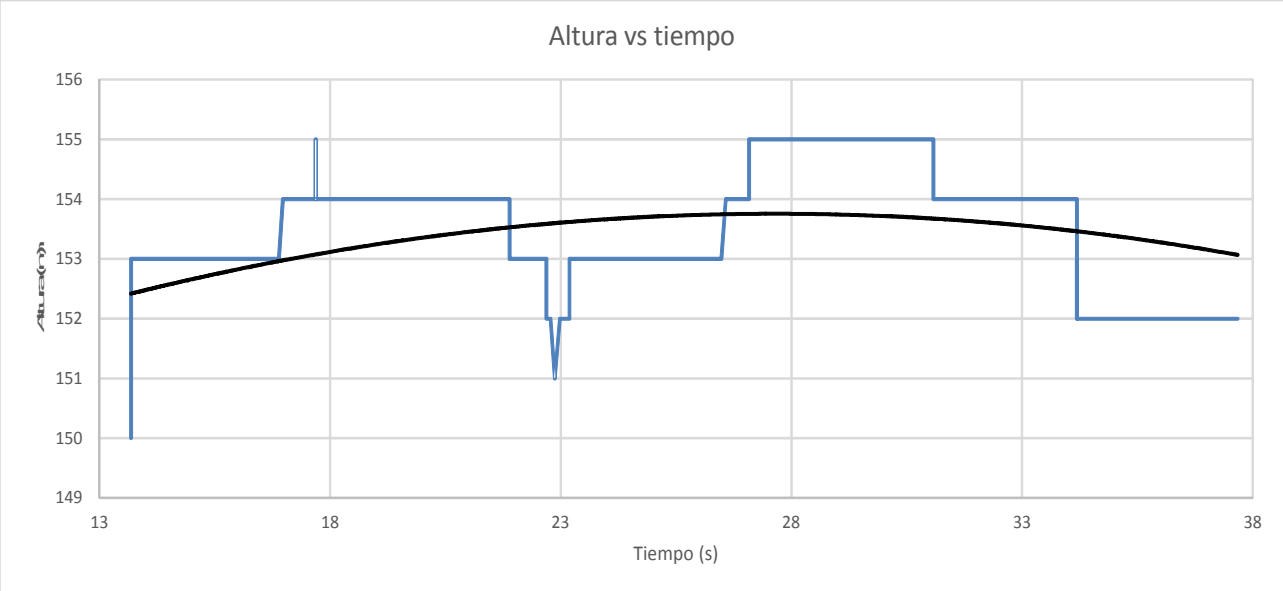

Ilustración 4.14: Altitud vs tiempo a altitud constante

La altitud medida durante el vuelo no es constante en este período de 20 segundos, varía entre los 152 y los 155 m, una diferencia insignificante. La altitud media es de 153,4 m.

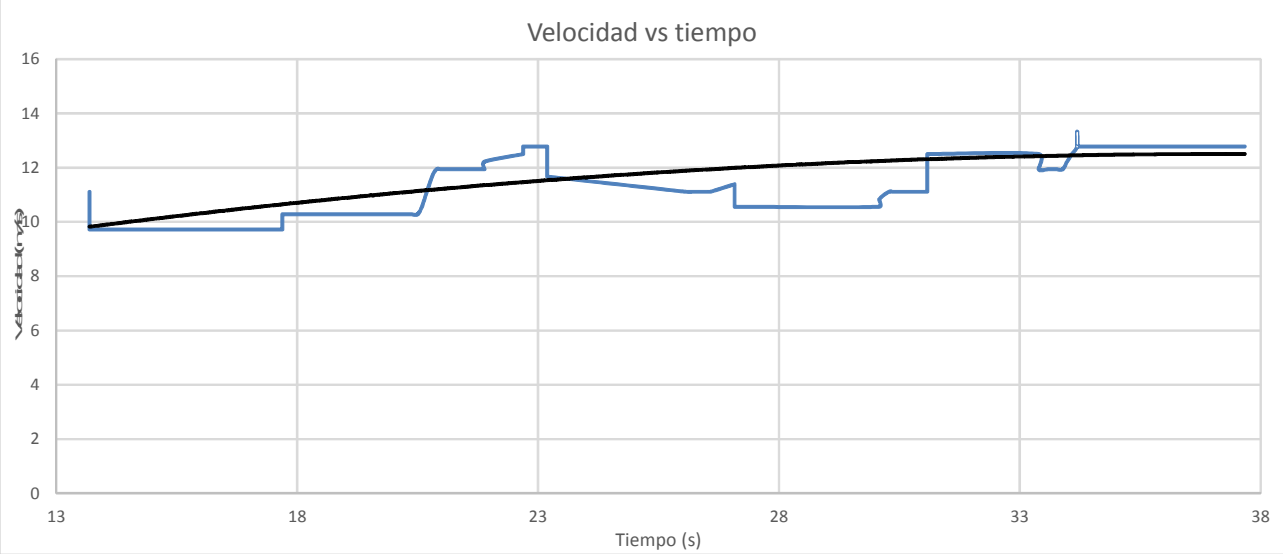

Ilustración 4.15: Velocidad vs tiempo a altitud constante

La velocidad la podemos considerar constante, se mantiene entre los 10 y los 13 m/s en este período de 20 s. la velocidad media es de 11,6 m/s.

Si lo comparamos con los resultados obtenidos en el simulador:

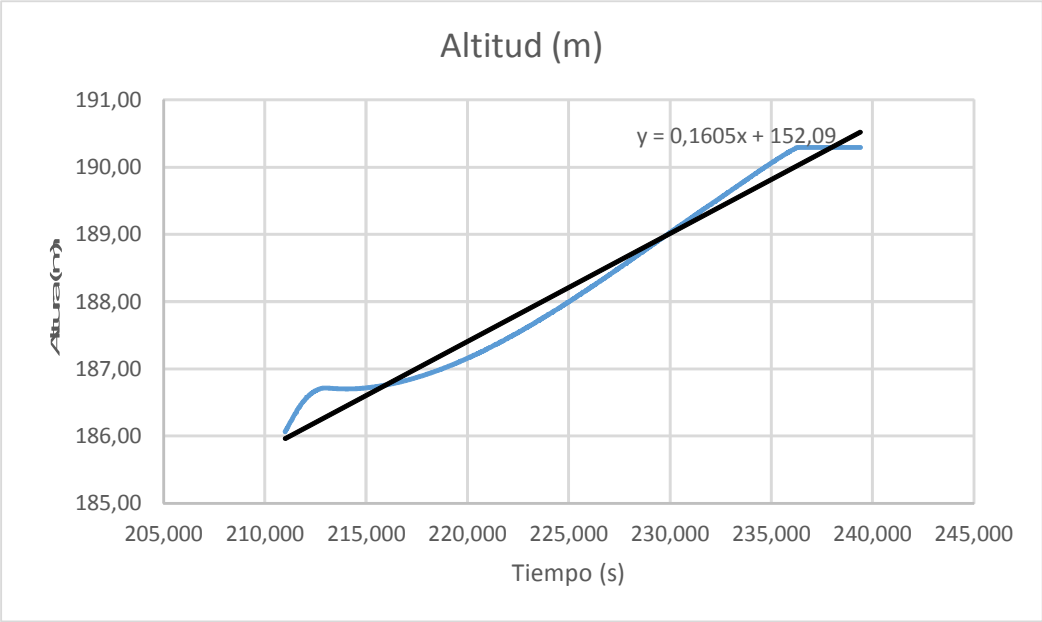

Ilustración 4.16: Altura vs tiempo durante el tramo a altura constante

Evidentemente, es casi imposible mantener una altitud perfectamente constante durante el vuelo en el simulador, por lo que lo hemos hecho lo más preciso posible. En el tramo que nos interesa, sufre una ascendencia de 4 m aproximadamente, lo cual podemos considerar despreciable en los casi 30 s que dura el vuelo a altura constante.

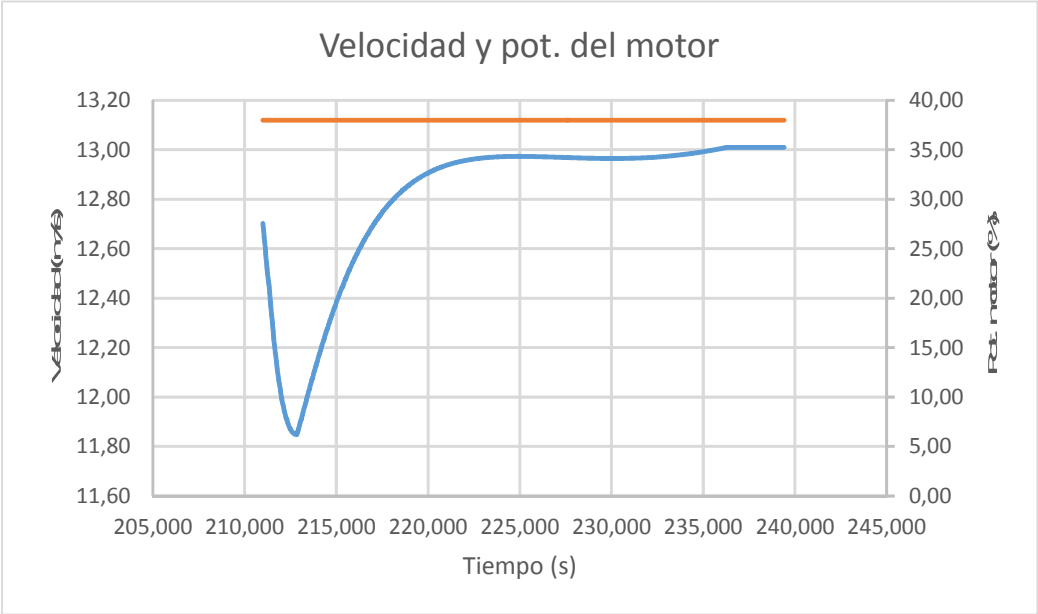

Ilustración 4.17: Potencia del motor y velocidad a altura constante

La potencia del motor se mantiene al 38% y la velocidad media es de 12,81 m/s en ese tramo. En el vuelo del UAV real, la velocidad media era de 11,6 m/s, una velocidad considerablemente inferior.

4.3.4. Prueba 4

Esta última prueba que diseñamos consiste en realizar un cambio de rumbo de 180º con un giro de los alerones del 10% y con los motores al 38% de su capacidad. De esta forma podremos observar la pérdida de altura durante el vuelo y el tiempo necesario para realizar el cambio de rumbo, y compararlo con el modelo virtual.

Para esta prueba podemos usar la transmisión de datos GPS del UAV, que nos aporta la longitud y la latitud en cada instante. El problema es que no encontramos la manera de traducir estas coordenadas en otro tipo de información con el que podamos trabajar fácilmente, como un radio de giro por ejemplo, que nos permita observar la trayectoria que realiza el UAV real y reproducirla posteriormente en el simulador.

De esta forma nos vemos obligados a anular esta prueba.

4.4. Análisis de los resultados del modelo

Con el objetivo de hacer una comparación más intuitiva y fácil de asimilar a simple vista, representamos las diferencias entre los resultados de las pruebas del modelo real y el virtual:

Velocidad de:	Virtual (m/s)	Real (m/s)	Diferencia (%)
Ascenso	4,8095	2,7857	73%
Descenso	3,2528	3,1582	3%
Altura cte.	12,81	11,6	10%

Tabla 4.1: Diferencias entre el modelo virtual y el real

Tanto en el ascenso como en el descenso, lo que medimos es la velocidad en metros verticales por segundo.

Vemos que durante el ascenso, la velocidad de ascendencia es un 73% mayor que en el UAV real, lo que es una gran diferencia que hay que corregir. Para poder corregir esto tenemos tres opciones, la primera sería disminuir la eficiencia de las hélices para obtener un empuje menor, opción que descartamos porque hemos dedicado una cantidad de tiempo y de ajuste de las hélices considerable, y creemos que son lo suficientemente realistas. La segunda opción sería disminuir la sustentación para disminuir la velocidad de elevación, y la tercera aumentar el arrastre para aumentar la dificultad de elevación del UAV.

Por otro lado, durante el descenso tenemos una diferencia del 3%, resultado que podemos considerar como bastante preciso, y que no necesita ajustes, se encuentra en un rango que creemos que es muy preciso.

En tercer lugar, durante el vuelo a altura constante, vemos que bajo las mismas condiciones que en el vuelo real, en el simulador obtenemos una velocidad un 10% mayor. Debemos ajustarlo más ya que es una diferencia considerable. Para ello podemos disminuir la sustentación o bien aumentar el arrastre.

4.5. Ajuste del modelo

Como hemos visto en el punto anterior, el camino a seguir para ajustar nuestro modelo, y hacer que su comportamiento en el simulador sea más parecido a comportamiento real, es disminuir la sustentación o aumentar el arrastre, tanto para el caso de vuelo durante ascenso como para el vuelo a altura constante, pero debemos tener cuidado, porque al variar estos valores, también cambiaremos el comportamiento durante el descenso, que es bastante preciso ya.

Tras hacer varios cambios siguiendo este criterio, y comprobar las velocidades y pendientes de variación de altura durante el ascenso, descenso y vuelo a altura constante, hemos conseguido llegar a un modelo bastante preciso, aumentando los coeficientes de arrastre a pequeños ángulos de ataque (entre -10º y 10º) y disminuyendo el máximo del coeficiente de sustentación. También hemos aumentado el coeficiente de sustentación para ángulos de ataque negativos (descenso del UAV) para compensar la pérdida de precisión producida por la el aumento del ángulo de arrastre. El resultado de estas modificaciones es el siguiente:

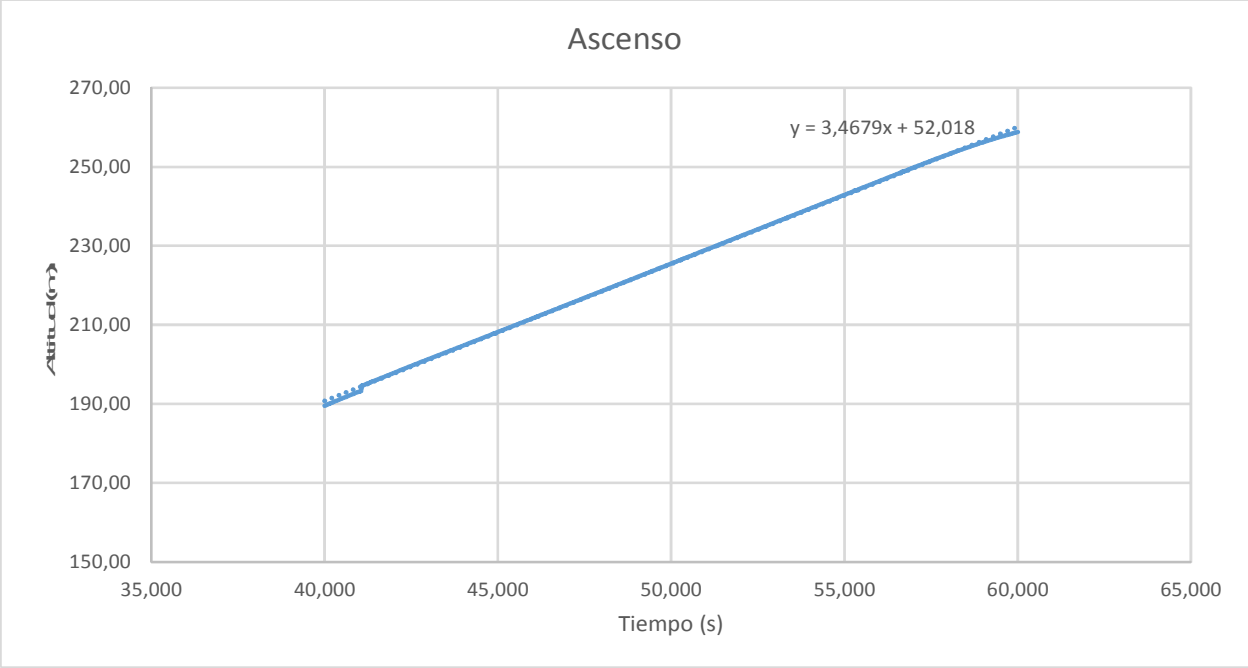

Ilustración 4.18: Período de ascenso

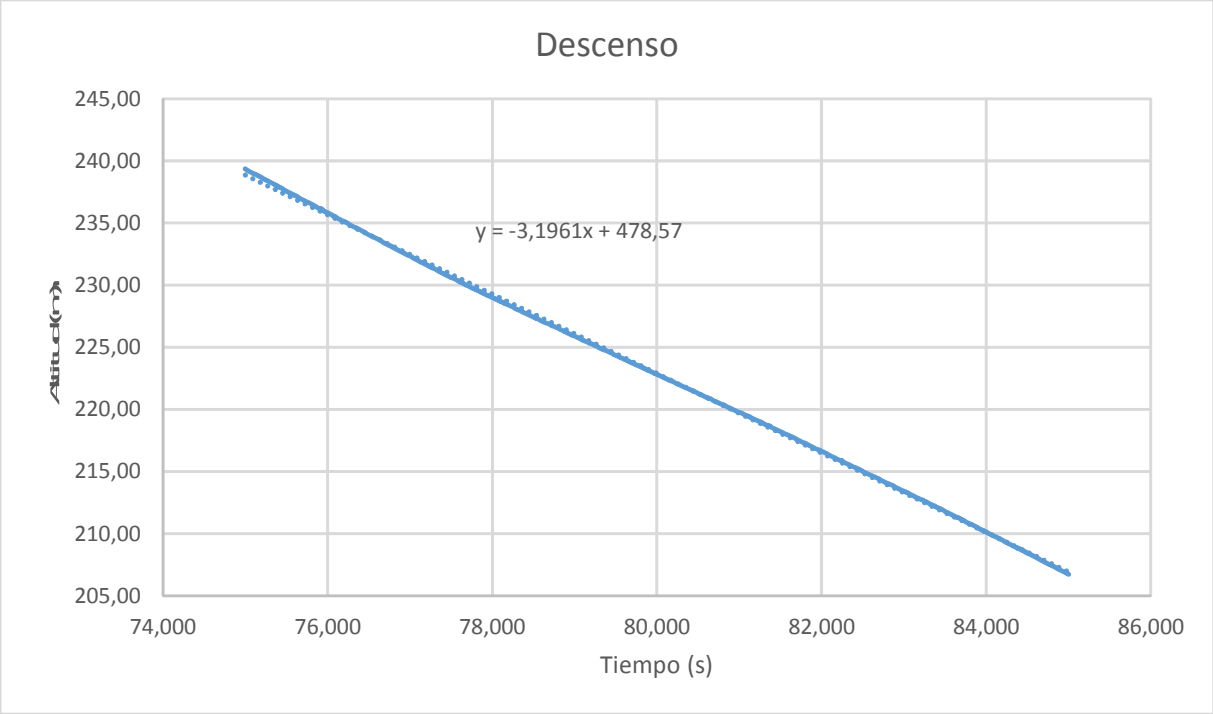

Ilustración 4.19: Período de descenso

Si nos fijamos en las pendientes de las gráficas y las comparamos:

Velocidad de:	Virtual (m/s)	Real (m/s)	Diferencia (%)
Ascenso	3,4679	2,7857	24%
Descenso	3,1961	3,1582	1%
Altura cte.	11,58	11,6	1%

Tabla 4.2: Diferencias despues del ajuste

Las modificaciones sufridas por los coeficientes de arrastre y sustentación fueron:

ARRASTRE		
Ángulo (º)	**Antiguo**	**Nuevo**
-90	1,280	1,280
-45	0,905	0,905
-10	0,130	0,130
-8	0,090	**0,120**
0	0,080	**0,110**
8	0,090	**0,150**
10	0,130	**0,300**
45	0,905	0,905
90	1,280	1,280

Tabla 4.3: Modificaciones del arrastre (en rojo)

SUSTENTACIÓN		
Ángulo (º)	**Antiguo**	**Nuevo**
-10	-1,2000	**-1,0000**
-5	-0,5000	**-0,2000**
0	0,2734	0,2734
2	0,3710	0,3710
4	0,4675	0,4675
8	0,6561	0,6561
10	0,7476	0,7476
18	1,0000	**0,8000**
25	1,0000	**--**
30	0,7476	**--**
45	-0,9000	-0,9000
90	-2,0000	-2,0000

Tabla 4.4: Modificaciones de la sustentación (en rojo)

Con estas modificaciones conseguimos una precisión muy alta para las maniobras de vuelo a altura constante (diferencia del 1%) y descenso (diferencia del 1%) del UAV, y un comportamiento un poco menos preciso para la maniobra de ascenso (diferencia del 24%).

Esta diferencia es justificable, durante el vuelo del UAV real, las condiciones meteorológicas no fueron perfectas, se sucedieron rachas de viento que pudieron influir en las medidas de velocidad y altura de forma negativa. Además también influye el hecho de que se producen pequeños giros del UAV durante su vuelo que contrarrestan el ascenso, haciéndolo más suave.

También asumimos parte de la culpa de estas desviaciones porque en AVL, es imposible introducir información sobre la rugosidad de la superficie, o de las imperfecciones de la superficie (cavidades, cortes, sensores instalados, etc,), acuerdos aerodinámicos entre el fuselaje y la superficie alar, aristas vivas en la parte inferior del fuselaje, uniones imperfectas entre las alas y las superficies de control, y discrepancias entre el perfil alar utilizado y el real, dándonos como resultado una aerodinámica con un cierto error.

Por otra parte parece obvio que las discrepancias aumentan al aumentar el motor, por lo que deducimos que parte de la culpa de las diferencias están en el modelado de las hélices, probablemente debido a que hemos tomado algunas medidas de forma experimental de ellas (ángulos de incidencia, p.e.). Probablemente obtendríamos mejores resultados si dispusiésemos de mejores recursos, con los que tomar medidas más precisas.

Debemos mencionar también que el método experimental mediante el cual hemos calculado las inercias tiene un cierto error, por el hecho de ser experimental, que influye notablemente en la respuesta de las rotaciones del UAV.

5. CONCLUSIONES

Iniciamos este proyecto con objeto de desarrollar un modelo de vehículo aéreo no tripulado en un simulador de vuelo, usando en la medida de lo posible software de código abierto y de la forma más precisa posible. La finalidad de esto es poder realizar pruebas de vuelo con un modelo virtual, en un simulador, sin necesidad de sacar el UAV real a volar, con los inconvenientes que eso conlleva.

Hemos conseguido realizar el desarrollo completo de la dinámica de vuelo del UAV virtual seleccionando el FDM más adecuado y usando únicamente software de código libre, o versiones gratuitas del software (versiones de estudiante), haciendo una selección razonada de entre las diferentes opciones disponibles, en función de sus ventajas e inconvenientes y su adaptabilidad a nuestro caso.

Por otro lado, hemos investigado y desarrollado las diferentes partes del UAV de la forma más realista posible, teniendo en cuenta las limitaciones del software de código libre. Dedicamos una gran cantidad de tiempo a la toma de datos y posterior desarrollo de las hélices, para que aportara una información fiable de sus características a la simulación. El modelado de los motores apenas conllevó dificultades, fue la parte más breve de desarrollar. A la parte aerodinámica del modelo hemos necesitado una cantidad de tiempo aún mayor que para las hélices, su desarrollo implicó invertir más tiempo en investigación sobre aerodinámica y en tomar datos del UAV real. Fue la parte mas laboriosa y con mayores discrepancias de las 4, debido a su complejidad, y a nuestras limitaciones. Las inercias las hemos hallado mediante un método experimental, con el posible error que eso conlleva en los resultados. El tren de aterrizaje lo modelamos como una serie de puntos que daban estabilidad al modelo en reposo y le hemos otorgado unas características de contacto (rozamiento, amortiguamiento,...) que dieran como resultado un comportamiento parecido a la realidad, aunque sabemos que el UAV real carece de tren de aterrizaje, aterriza directamente sobre su “fuselaje”. Y por último, la programación del control de las superficies de control del UAV, para la cual nos hemos basado en documentación existente del programa.

Por último hemos hecho un análisis objetivo y autocrítico de los resultados de comportamiento del modelo obtenido en comparación con el modelo real, explicando que las diferencias están causadas por las limitaciones de recursos.

Pese a todo ello, podemos decir que hemos cumplido todos los objetivos de este proyecto de la forma más correcta dentro de nuestras posibilidades, y hemos obtenido un modelo de UAV realista.

6. BIBLIOGRAFÍA

https://es.wikipedia.org

http://wiki.flightgear.org/

http://www.flightgear.org/

http://www.nasa.gov/

http://www.dtic.upf.edu/~gvirtual/master/rv/seccio2/seccio2.htm

http://cdigital.uv.mx/bitstream/123456789/31665/1/mendezrodriguezberenice.pdf

http://jsbsim.sourceforge.net/

http://www.buckarooshangar.com/flightgear/yasimtut_introduction.html

http://www.alasewm.com.ar/helicewm.htm

http://www.mh-aerotools.de/airfoils/javaprop.htm

http://www.crashtesthobby.com/hercules-xl-7278.html

http://www.allstar.fiu.edu/aero/BA-Background.htm

http://www.dronair.es/servicios-aereos-drones

http://www.cartogalicia.com.mialias.net/cartouav/

http://www.atiak.com/products/resolution-3-airframe/

http://arc.aiaa.org/doi/abs/10.2514/6.2016-0947

http://www.holycows.net/datcom/

http://flitetest.com/articles/beginner-series-choosing-a-plane

http://web.mit.edu/drela/Public/web/avl/

http://outerra.shoutwiki.com/wiki/JSBSim_Properties

http://www.manualvuelo.com/TCV/TCV57.html

http://www.flightlab.net/Flightlab.net/Home.html

http://www.city-gallery.com/knoblock/projects/flightgear/Docs/coords.html

http://www.colegiocristorey.com/alumnos/f22/clase/fisicaavion.htm

http://cinematicaydinamicabrmc200619921.blogspot.com.es/2010/07/formulas-de-momento-de-inercia-el.html

http://www.ima.org.uk/_db/_documents/maths07_williams_huw.pdf

http://naca.central.cranfield.ac.uk/reports/1948/naca-tn-1629.pdf

https://www.youtube.com/watch?v=m9iHEanmNWc

http://www.me.utexas.edu/~me244L/labs/filar/filaroverview.html

www.ijarcet.org

Methods for predicting the aerodynamic and stability control characteristics of stol aircraft (volume III)

7. ANEXOS

7.1. ANEXO I: " Fichero de hélices"

```
<?xml version="1.0"?>
<propeller name="helices_9x6">
  <ixx unit="KG*M2"> 0.00023516 </ixx>  <!--Caculado como masa por radio^2-->
  <diameter unit="IN"> 9 </diameter>
  <numblades> 2 </numblades>
  <gearratio> 1 </gearratio>
  <minpitch> 30 </minpitch>
  <maxpitch> 30 </maxpitch>
  <minrpm> 1 </minrpm>
  <maxrpm> 10000 </maxrpm>
  <constspeed> 0 </constspeed>         <!--1 = constant speed mode, 0 = manual
pitch mode-->
  <reversepitch> 0 </reversepitch>         <!-- angulo en radianes sobre el cual
impulsa la hélice cos(0)=1 sobre el eje x-->
  <p_factor> 0 </p_factor>                 <!-- Efecto aerodinámico explicado en
la memoria. No tiene influencia en nuestro caso-->

  <table name="C_THRUST" type="internal">
    <tableData>
                0       0.011644837
                0.01    0.011524754
                0.02    0.011404672
                0.03    0.011284589
                0.04    0.011164507
                0.05    0.011044425
                0.06    0.011066151
                0.07    0.011087877
                0.08    0.011109604
                0.09    0.01113133
                0.1     0.011153057
                0.11    0.011094762
                0.12    0.011036468
                0.13    0.010978174
                0.14    0.01091988
                0.15    0.010861585
                0.16    0.010787715
                0.17    0.010713846
                0.18    0.010639976
                0.19    0.010566106
                0.2     0.010492236
                0.21    0.010394629
                0.22    0.010297022
                0.23    0.010199414
                0.24    0.010101807
                0.25    0.0100042
                0.26    0.009913639
                0.27    0.009823078
                0.28    0.009732517
                0.29    0.009641956
                0.3     0.009551395
                0.31    0.009443967
                0.32    0.009336539
                0.33    0.009229111
```

0.34	0.009121682
0.35	0.009014254
0.36	0.008886978
0.37	0.008759703
0.38	0.008632427
0.39	0.008505152
0.4	0.008377876
0.41	0.008240515
0.42	0.008103155
0.43	0.007965794
0.44	0.007828433
0.45	0.007691072
0.46	0.007544419
0.47	0.007397766
0.48	0.007251113
0.49	0.007104459
0.5	0.006957806
0.51	0.006805398
0.52	0.006652991
0.53	0.006500583
0.54	0.006348175
0.55	0.006195767
0.56	0.006037282
0.57	0.005878797
0.58	0.005720312
0.59	0.005561826
0.6	0.005403341
0.61	0.005239865
0.62	0.005076388
0.63	0.004912912
0.64	0.004749435
0.65	0.004585959
0.66	0.00441915
0.67	0.004252341
0.68	0.004085532
0.69	0.003918723
0.7	0.003751914
0.71	0.003581127
0.72	0.00341034
0.73	0.003239553
0.74	0.003068766
0.75	0.002897979
0.76	0.002725489
0.77	0.002551971
0.78	0.002377866
0.79	0.002202807
0.8	0.002026354
0.81	0.00184924
0.82	0.001671098
0.83	0.001492149
0.84	0.001312686
0.85	0.001131975
0.86	0.00095053
0.87	0.000768278
0.88	0.00058676
0.89	0.000400324
0.9	0.000214989
0.91	9.84E-06
0.92	-0.000180491
0.93	-0.000368541
0.94	-0.000557913

```
                0.95    -0.000747652
                0.96    -0.000940694
  </tableData>
</table>

<table name="C_POWER" type="internal">
  <tableData>
                0       0.00314904
                0.01    0.003095208
                0.02    0.003041376
                0.03    0.002987544
                0.04    0.002933712
                0.05    0.00287988
                0.06    0.002950423
                0.07    0.003020965
                0.08    0.003091508
                0.09    0.003162051
                0.1     0.003232593
                0.11    0.003241946
                0.12    0.003251298
                0.13    0.00326065
                0.14    0.003270003
                0.15    0.003279355
                0.16    0.003284571
                0.17    0.003289787
                0.18    0.003295003
                0.19    0.003300219
                0.2     0.003305435
                0.21    0.003303029
                0.22    0.003300623
                0.23    0.003298217
                0.24    0.003295811
                0.25    0.003293404
                0.26    0.003291905
                0.27    0.003290405
                0.28    0.003288905
                0.29    0.003287406
                0.3     0.003285906
                0.31    0.00327835
                0.32    0.003270794
                0.33    0.003263238
                0.34    0.003255682
                0.35    0.003248126
                0.36    0.003231588
                0.37    0.00321505
                0.38    0.003198513
                0.39    0.003181975
                0.4     0.003165437
                0.41    0.003142093
                0.42    0.003118749
                0.43    0.003095405
                0.44    0.003072062
                0.45    0.003048718
                0.46    0.003019161
                0.47    0.002989604
                0.48    0.002960047
                0.49    0.00293049
                0.5     0.002900933
                0.51    0.002864982
                0.52    0.002829031
                0.53    0.00279308
```

```
                0.54    0.002757129
                0.55    0.002721178
                0.56    0.002678923
                0.57    0.002636668
                0.58    0.002594413
                0.59    0.002552159
                0.6     0.002509904
                0.61    0.002460456
                0.62    0.002411008
                0.63    0.002361559
                0.64    0.002312111
                0.65    0.002262663
                0.66    0.002206507
                0.67    0.002150352
                0.68    0.002094196
                0.69    0.00203804
                0.7     0.001981885
                0.71    0.001918544
                0.72    0.001855203
                0.73    0.001791862
                0.74    0.001728521
                0.75    0.00166518
                0.76    0.001597695
                0.77    0.001528602
                0.78    0.001458027
                0.79    0.001385927
                0.8     0.001312014
                0.81    0.001236577
                0.82    0.001159492
                0.83    0.0010808
                0.84    0.001000666
                0.85    0.000918719
                0.86    0.000835165
                0.87    0.000749964
                0.88    0.000663897
                0.89    0.000574081
                0.9     0.000483564
                0.91    0.000393336
                0.92    0.000297958
                0.93    0.000202457
                0.94    0.000104978
                0.95    5.93E-06
                0.96    -9.61E-05
    </tableData>
  </table>

</propeller>
```

7.2. ANEXO II: "Fichero de motores"

```
<?xml version="1.0"?>
<!-- EMP N3530 KV1100 motor -->
<!-- web:   http://www.megastorerc.com/motores-brushless/4072-motor-n353013-
1100.html -->
<!-- motor de 345 W -->

<electric_engine name="n3530">
  <power unit="WATTS">  345 </power>
</electric_engine>
```

7.3. ANEXO III: "Fichero de aerodinámica"

```
<?xml version="1.0"?>
    <aerodynamics>

        <axis name="DRAG">
            <function name="aero/coefficient/CD0">
                <description>Drag_at_zero_lift</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
                      <table>
                          <independentVar>aero/alpha-rad</independentVar>
                          <tableData>
                              -1.5708   1.28
                              -0.7854   0.905
                              -0.1745   0.13
                              -0.1396   0.12
                              0.0000    0.11
                              0.1396    0.15
                              0.1745    0.30
                              0.7854    0.905
                              1.5708    1.28
                          </tableData>
                      </table>
                </product>
            </function>
            <function name="aero/coefficient/CDi">
                <description>Induced_drag</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
                    <property>aero/cl-squared</property>
                    <table>
                          <independentVar>aero/alpha-rad</independentVar>
                          <tableData>
                              -1.5708   0.3704
                              -0.7854   0.3598
                              -0.1745   0.061
                              -0.1396   0.047
                              -0.0698   0.024
                              -0.0349   0.0152
                              0.0000    0.0083
                              0.0349    0.0152
                              0.0698    0.024
                              0.1396    0.047
                              0.1745    0.061
                              0.7854    0.3598
                              1.5708    0.3704
                          </tableData>
                      </table>
                </product>
            </function>
            <function name="aero/coefficient/CDbeta">
                <description>Drag_due_to_sideslip</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
```

```
                <table>
                    <independentVar>aero/beta-rad</independentVar>
                    <tableData>
                        -1.5700   1.2300
                        -0.2600   0.0500
                        -0.1396   0.00154
                        -0.1047   0.00087
                        -0.0698   0.00039
                        -0.0349   0.0001
                        0.0000    0.0000
                        0.0349    0.0001
                        0.0698    0.00039
                        0.1047    0.00087
                        0.1396    0.00154
                        0.2600    0.0500
                        1.5700    1.2300
                    </tableData>
                </table>
            </product>
        </function>
        <function name="aero/coefficient/CDde">
            <description>Drag_due_to_Elevator_Deflection</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>fcs/elevator-pos-norm</property>
                <value>0.1321</value>
                        </product>
        </function>
    </axis>

    <axis name="SIDE">
        <function name="aero/coefficient/CYb">
            <description>Side_force_due_to_beta</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>aero/beta-rad</property>
                <value>-0.1254</value>
            </product>
        </function>
    </axis>

    <axis name="LIFT">
        <function name="aero/coefficient/CLalpha">
            <description>Lift_due_to_alpha</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                  <table>
                    <independentVar>aero/alpha-rad</independentVar>
                    <tableData>
                        -0.1745   -1.0
                        -0.0873   -0.2
                        0.0000    0.27343
                        0.0349    0.37099
                        0.0698    0.46751
                        0.1396    0.65613
                        0.1745    0.7476
                        0.3142    0.8
                        0.7854    -0.9
```

```
                        1.5708    -2
                    </tableData>
                </table>
            </product>
        </function>
        <function name="aero/coefficient/CLde">
            <description>Lift_due_to_Elevator_Deflection</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>fcs/elevator-pos-rad</property>
                <value>0.8660</value>
            </product>
        </function>
    </axis>

    <axis name="ROLL">
        <function name="aero/coefficient/Clb">
            <description>Roll_moment_due_to_beta</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>aero/beta-rad</property>
                <value>-0.0623</value>
            </product>
        </function>
        <function name="aero/coefficient/Clp">
            <description>Roll_moment_due_to_roll_rate</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>aero/bi2vel</property> <!-- wing span/
(2*velocity)-->
                <property>velocities/p-aero-rad_sec</property>
                <value>-0.4006</value>
            </product>
        </function>
        <function name="aero/coefficient/Clr">
            <description>Roll_moment_due_to_yaw_rate</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>aero/bi2vel</property> <!-- wing span/
(2*velocity)-->
                <property>velocities/r-aero-rad_sec</property>
                <value>0.0937</value>
            </product>
        </function>
        <function name="aero/coefficient/Clda">
            <description>Roll_moment_due_to_aileron</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>fcs/left-aileron-pos-rad</property>
                <value>0.2022</value>
            </product>
        </function>
```

```
            <function name="aero/coefficient/Cldr">
                <description>Roll_moment_due_to_rudder</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
                    <property>metrics/bw-ft</property>
                    <property>fcs/rudder-pos-rad</property>
                    <value>0.0072</value>
                </product>
            </function>
        </axis>

        <axis name="PITCH">
            <function name="aero/coefficient/Cmalpha">
                <description>Pitch_moment_due_to_alpha</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
                    <property>metrics/cbarw-ft</property>
                    <property>aero/alpha-rad</property>
                    <table>
                          <independentVar>aero/alpha-rad</independentVar>
                          <tableData>
                              0.0000    -0.06177
                              0.0349    -0.0639
                              0.0698    -0.06562
                              0.1396    -0.06783
                              0.1745    -0.0683
                              0.7854    0.01415
                              1.5708    0.09682
                          </tableData>
                    </table>
                </product>
            </function>
            <function name="aero/coefficient/Cmde">
                <description>Pitch_moment_due_to_elevator</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
                    <property>metrics/cbarw-ft</property>
                    <property>fcs/elevator-pos-rad</property>
                     <value>-0.4184</value>
                </product>
            </function>
            <function name="aero/coefficient/Cmq">
                <description>Pitch_moment_due_to_pitch_rate</description>
                <product>
                    <property>aero/qbar-psf</property>
                    <property>metrics/Sw-sqft</property>
                    <property>metrics/cbarw-ft</property>
                    <property>aero/ci2vel</property> <!--wing chord/
(2*velocity)-->
                    <property>velocities/q-aero-rad_sec</property>
                    <value>-11.080</value>
                </product>
            </function>
        </axis>

        <axis name="YAW">
            <function name="aero/coefficient/Cnb">
                <description>Yaw_moment_due_to_beta</description>
```

```
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>aero/beta-rad</property>
                <value>0.0150</value>
            </product>
        </function>
        <function name="aero/coefficient/Cnr">
            <description>Yaw_moment_due_to_yaw_rate</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>aero/bi2vel</property>
                <property>velocities/r-aero-rad_sec</property>
                <value>-0.0110</value>
            </product>
        </function>
        <function name="aero/coefficient/Cndr">
            <description>Yaw_moment_due_to_rudder</description>
            <product>
                <property>aero/qbar-psf</property>
                <property>metrics/Sw-sqft</property>
                <property>metrics/bw-ft</property>
                <property>fcs/rudder-pos-rad</property>
                <value>-0.0119</value>
            </product>
        </function>

    </axis>
</aerodynamics>
```

7.4. ANEXO IV: "Fichero de tren de aterrizaje e inercias"

```
<?xml version="1.0"?>

<?xml-stylesheet href="http://jsbsim.sourceforge.net/JSBSim.xsl"
type="text/xsl"?>
<fdm_config name="herculesgii" version="2.0" release="BETA"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="http://jsbsim.sourceforge.net/JSBSim.xsd">

    <fileheader>
        <author> José Rubén Vieites Pardo </author>
        <filecreationdate> 07-06-2016 </filecreationdate>
        <version> 1.0 </version>
        <description> Modelo Hércules GII </description>
    </fileheader>

    <metrics>
        <wingarea unit="M2"> 1.1534 </wingarea><!--Superficie proyectada del
modelo del UAV-->
        <wingspan unit="M"> 1.94 </wingspan><!--Envergadura-->
            <wingincidence unit="DEG"> 0 </wingincidence><!--Incidencia del ala,
tomado valor aproximado-->
        <chord unit="M"> 0.56 </chord> <!--Cuerda en la zona central del ala-->
        <htailarea unit="M2"> 0.0984 </htailarea><!--No tenemos cola. Es el área
de los 4 elementos de profundidad-->
        <htailarm unit="M"> 0.372 </htailarm><!--No tenemos cola. Distancia en
metros desde el CG hasta el centro de presiones de la profundidad-->
        <vtailarea unit="M2"> 0.0627 </vtailarea><!--Área de las 2 colas
verticales, teniendo en cuenta las derivas-->
        <vtailarm unit="M"> 0.365 </vtailarm><!--Distancia desde el CG hasta el
centro de las derivas, aproximada-->
            <location name="AERORP" unit="M"> <!--Aerodynamic Reference Point,
origen para todo lo relacionado con la aerodinamica-->
            <x> 0.254 </x>
            <y> 0 </y>
            <z> 0 </z>
        </location>
        <location name="EYEPOINT" unit="M"> <!--"Punto de vista del piloto"-->
            <x> 0.254 </x>
            <y> 0 </y>
            <z> 0 </z>
        </location>
        <location name="VRP" unit="M"> <!--Punto de Referencia Visual. Concilia
el modelo mecánico y el gráfico-->
            <x> 0 </x>
            <y> 0 </y>
            <z> 0 </z>
        </location>
    </metrics>

    <mass_balance>
        <ixx unit="KG*M2"> 1.5956 </ixx>
        <iyy unit="KG*M2"> 0.4208 </iyy>
        <izz unit="KG*M2"> 1.7366 </izz>
        <ixy unit="KG*M2"> 0 </ixy>
        <ixz unit="KG*M2"> 0 </ixz>
        <iyz unit="KG*M2"> 0 </iyz>
        <emptywt unit="KG"> 3 </emptywt>
        <location name="CG" unit="M">
```

```
            <x> 0.254 </x>
            <y> 0 </y>
            <z> 0 </z>
        </location>
        <pointmass name="payload">
            <weight unit="KG"> 0 </weight>
            <location name="payload" unit="M">
                <x> 0.254 </x>
                <y> 0 </y>
                <z> 0 </z>
            </location>
        </pointmass>
    </mass_balance>

    <ground_reactions><!--Tren de aterrizaje-->
        <contact type="BOGEY" name="LEFT_MLG">
            <location unit="IN">
                <x> 28.7 </x>
                <y> -38 </y>
                <z> -6 </z>
            </location>
            <static_friction> 1 </static_friction>
            <dynamic_friction> 0.8 </dynamic_friction>
            <rolling_friction> 0.02 </rolling_friction>
            <spring_coeff unit="LBS/FT"> 60 </spring_coeff>
            <damping_coeff unit="LBS/FT/SEC"> 10 </damping_coeff>
            <max_steer unit="DEG"> 0.0 </max_steer>
            <brake_group> LEFT </brake_group>
            <retractable>0</retractable>
        </contact>
        <contact type="BOGEY" name="RIGHT_MLG">
            <location unit="IN">
                <x> 28.7 </x>
                <y> 38 </y>
                <z> -6 </z>
            </location>
            <static_friction> 1 </static_friction>
            <dynamic_friction> 0.8 </dynamic_friction>
            <rolling_friction> 0.02 </rolling_friction>
            <spring_coeff unit="LBS/FT"> 60 </spring_coeff>
            <damping_coeff unit="LBS/FT/SEC"> 10 </damping_coeff>
            <max_steer unit="DEG"> 0.0 </max_steer>
            <brake_group> RIGHT </brake_group>
            <retractable>0</retractable>
        </contact>
        <contact type="BOGEY" name="NOSE_LG">
            <location unit="IN">
                <x> 0 </x>
                <y> 0 </y>
                <z> -10 </z>
            </location>
            <static_friction> 1 </static_friction>
            <dynamic_friction> 0.8 </dynamic_friction>
            <rolling_friction> 0.02 </rolling_friction>
            <spring_coeff unit="LBS/FT"> 60 </spring_coeff>
            <damping_coeff unit="LBS/FT/SEC"> 10 </damping_coeff>
            <max_steer unit="DEG"> 360.0 </max_steer>
            <brake_group> NONE </brake_group>
            <retractable>0</retractable>
        </contact>
    </ground_reactions>
```

```
    <propulsion>
      <engine n="0" file="n3530">
            <location unit="M">
                <x> 0.0 </x>
                <y> 0.25 </y>
                <z> 0.0 </z>
            </location>
            <orient unit="DEG">
                <roll>  0.0 </roll>
                <pitch> 0.0 </pitch>
                <yaw>   0.0 </yaw>
            </orient>
            <feed>0</feed>
            <thruster file="9x6">
                <location unit="M">
                    <x> 0.0 </x>
                    <y> 0.25 </y>
                    <z> 0.0 </z>
                </location>
                <orient unit="DEG">
                    <roll>  0.0 </roll>
                    <pitch> 0.0 </pitch>
                    <yaw>   0.0 </yaw>
                </orient>
                <sense> -1 </sense>     <!--sentido de giro visto desde atras
1:horario ; -1:antihorario-->
            </thruster>
        </engine>
              <engine n="1" file="n3530">
            <location unit="M">
                <x> 0.0 </x>
                <y> -0.25 </y>
                <z> 0.0 </z>
            </location>
            <orient unit="DEG">
                <roll>  0.0 </roll>
                <pitch> 0.0 </pitch>
                <yaw>   0.0 </yaw>
            </orient>
            <feed>1</feed>
            <thruster file="9x6">
                <location unit="M">
                    <x> 0.0 </x>
                    <y> -0.25 </y>
                    <z> 0.0 </z>
                </location>
                <orient unit="DEG">
                    <roll>  0.0 </roll>
                    <pitch> 0.0 </pitch>
                    <yaw>   0.0 </yaw>
                </orient>
                <sense> 1 </sense>
            </thruster>
        </engine>
    </propulsion>



    <flight_control name="FCS: herculesgii_alerones">
     <channel name="All">
```

```
<summer name="pitch-trim-sum">
    <input>fcs/elevator-cmd-norm</input>
    <input>fcs/pitch-trim-cmd-norm</input>
    <clipto>
        <min>-1</min>
        <max>1</max>
    </clipto>
</summer>

<aerosurface_scale name="Elevator Control">
    <input>fcs/pitch-trim-sum</input>
    <range>
        <min>-0.35</min>
        <max>0.3</max>
    </range>
    <output>fcs/elevator-pos-rad</output>
</aerosurface_scale>

<aerosurface_scale name="Elevator Normalized">
    <input>fcs/elevator-pos-rad</input>
    <domain>
        <min>-0.3</min>
        <max> 0.3</max>
    </domain>
    <range>
        <min>-1</min>
        <max> 1</max>
    </range>
    <output>fcs/elevator-pos-norm</output>
</aerosurface_scale>

<summer name="Roll Trim Sum">
    <input>fcs/aileron-cmd-norm</input>
    <input>fcs/roll-trim-cmd-norm</input>
    <clipto>
        <min>-1</min>
        <max>1</max>
    </clipto>
</summer>

<aerosurface_scale name="Left Aileron Control">
    <input>fcs/roll-trim-sum</input>
    <range>
        <min>-0.35</min>
        <max>0.35</max>
    </range>
    <output>fcs/left-aileron-pos-rad</output>
</aerosurface_scale>

<aerosurface_scale name="Right Aileron Control">
    <input>-fcs/roll-trim-sum</input>
    <range>
        <min>-0.35</min>
        <max>0.35</max>
    </range>
    <output>fcs/right-aileron-pos-rad</output>
</aerosurface_scale>

<aerosurface_scale name="Left aileron Normalized">
    <input>fcs/left-aileron-pos-rad</input>
```

```
            <domain>
                <min>-0.35</min>
                <max> 0.35</max>
            </domain>
            <range>
                <min>-1</min>
                <max> 1</max>
            </range>
            <output>fcs/left-aileron-pos-norm</output>
        </aerosurface_scale>

        <aerosurface_scale name="Right aileron Normalized">
            <input>fcs/right-aileron-pos-rad</input>
            <domain>
                <min>-0.35</min>
                <max> 0.35</max>
            </domain>
            <range>
                <min>-1</min>
                <max> 1</max>
            </range>
            <output>fcs/right-aileron-pos-norm</output>
        </aerosurface_scale>

        <!--Derivas-->
     <summer name="rudder-sum">
      <input>fcs/rudder-cmd-norm</input>
      <clipto>
        <min>-1.0</min>
        <max> 1.0</max>
      </clipto>
    </summer>
    <aerosurface_scale name="Rudder Deg">
      <input>fcs/rudder-sum</input>
      <domain>
        <min>-1</min>
        <max> 1</max>
      </domain>
      <range>
        <min>-20.0</min>
        <max> 20.0</max>
      </range>
      <output>fcs/rudder-pos-deg</output>
    </aerosurface_scale>
    <pure_gain name="Rudder Position">
     <input>fcs/rudder-pos-deg</input>
     <gain>0.017453293</gain>
     <output>fcs/rudder-pos-rad</output>
    </pure_gain>

      </channel>
     </flight_control>

    <aerodynamics file="herculesgii_aero.xml"/>

</fdm_config>
```

7.5. ANEXO V: "hercules.avl"

```
!************************************

!AVL dataset for Hercules UAV
!************************************
HERCULES
!Mach = V/Vs = 12.5/343 (m/s)
0.03644
!IYsym   IZsym   Zsym
0        0       0.0
!Sref    Cref    Bref
1.1534   0.4306  1.94
!Xref    Yref    Zref
0.254    0.0      0.0295
#--------------- ALA CENTRAL -----------------
SURFACE
CENTRAL WING
!Nchordwise  Cspace             Nspanwise       Sspace
10            1.0         !96           1.0
COMPONENT
1
YDUPLICATE
0.0
ANGLE
2.0
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.000  0.000   0.560   0.    6         1.0
NACA
3210
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.050  0.000   0.560   0.    6         1.0
NACA
3210
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.100  0.000   0.560   0.    6         1.0
NACA
3210
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.150  0.000   0.560   0.    6         1.0
NACA
3210
#--------------- ALA CON ELEVONS ------------------
SURFACE
WING
!Nchordwise  Cspace             Nspanwise       Sspace
10            1.0         !96           1.0
COMPONENT
1
YDUPLICATE
0.0
ANGLE
2.0
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.1501  0.000   0.620   0.    6         1.0
```

```
NACA
3210
CONTROL
aileron  1.000 0.903  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.903  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.023   0.200  0.002   0.603   0.122    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.900  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.900  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.047   0.250  0.003   0.585   0.244    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.897  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.897  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.070   0.300  0.005   0.568   0.366    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.894  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.894  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.093   0.350  0.007   0.550   0.488   6         1.0
NACA
3210
CONTROL
aileron  1.000 0.891  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.891  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.117   0.400  0.009   0.533   0.610    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.887  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.887  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.140   0.450  0.010   0.516   0.732    6         1.0
NACA
```

```
3210
CONTROL
aileron  1.000 0.884  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.884  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.163   0.500  0.012   0.498   0.854    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.880  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.880  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.187   0.550  0.014   0.481   0.976    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.875  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.875  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.210   0.600  0.016   0.464   1.098    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.871  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.871  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.233   0.650  0.017   0.446   1.220    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.866  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.866  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.256   0.700  0.019   0.429   1.341    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.860  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.860  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.280   0.750  0.021   0.411   1.463    6         1.0
NACA
3210
```

```
CONTROL
aileron  1.000 0.854  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.854  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.303   0.800  0.023   0.394   1.585    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.848  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.848  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.326   0.850  0.024   0.377   1.707    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.841  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.841  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.350   0.900  0.026   0.359   1.829    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.833  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.833  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.373   0.950  0.028   0.342   1.951    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.825  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.825  0.47 1. 0.03  +1
#-------
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.382   0.970  0.029   0.335   2.000    6         1.0
NACA
3210
CONTROL
aileron  1.000 0.821  0.47 1. 0.03  -1
CONTROL
elevator  1.000 0.821  0.47 1. 0.03  +1
#----------------- ALA VERTICAL CON PROFUNDIDADE -------------
SURFACE
RUDDER
!Nchordwise  Cspace             Nspanwise       Sspace
10            1.0         !96           1.0
COMPONENT
1
```

```
YDUPLICATE
0.0
TRANSLATE
0.373 0.350 0.000
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.000  0.000   0.220   0.    6          1.0
CONTROL
rudder  1.000 1.000  0. 0. 1.  -1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.020   0.000  0.020   0.246   0.    6          1.0
CONTROL
rudder  1.000 0.814  0. 0. 1.  -1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.035   0.000  0.035   0.265   0.    6          1.0
CONTROL
rudder  1.000 0.698  0. 0. 1.  -1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.040   0.000  0.040   0.260   0.    6          1.0
CONTROL
rudder  1.000 0.692  0. 0. 1.  -1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.090   0.000  0.090   0.210   0.    6          1.0
CONTROL
rudder  1.000 0.619  0. 0. 1.  -1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.140   0.000  0.140   0.160   0.    6          1.0
CONTROL
rudder  1.000 0.500  0. 0. 1.  -1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.150   0.000  0.150   0.150   0.    6          1.0
CONTROL
rudder  1.000 0.467  0. 0. 1.  -1.0
#----------------- WINGLET -------------
SURFACE
WINGLET
!Nchordwise  Cspace             Nspanwise       Sspace
10            1.0         !96           1.0
COMPONENT
1
YDUPLICATE
0.0
TRANSLATE
0.382 0.970 0.029
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.313   0.000  -0.050   0.000   0.    6          1.0
```

```
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.000   0.000  0.000   0.313   0.    6         1.0
#----
SECTION
!Xle    Yle    Zle     Chord   Ainc  Nspanwise  Space
0.267   0.000  0.050   0.046   0.    6         1.0
#----------------- BODY -------------
BODY
FUSELAGE   !body name string
30  1.0    !Nbody  Bspace
#
TRANSLATE
0.0  0.0  -0.0449514
#
BFIL
herc_body.dat
```

7.6. ANEXO VI: Cálculo de inercias

Las inercias tienen una gran importancia cuando estamos diseñando cualquier tipo de estructura o máquina, reflejan la distribución de masa de un cuerpo o de un sistema de partículas en rotación.
En nuestro caso, las rotaciones del UAV en cualquiera de los 3 ejes principales de rotaciones están enormemente influenciadas por las inercias. Podemos decir que es una característica fundamental que rige el comportamiento del UAV.
En un cuerpo complejo como es un avión, un coche, o en nuestro caso un UAV, es imposible calcular las inercias de forma teórica, aplicando sólo ecuaciones, necesitamos de un método experimental para hallarlas. Uno de los métodos experimentales más utilizados para esto es el método del péndulo bifilar, también se pueden usar péndulos trifilares o cuadrifilares, pero por falta de tiempo y sencillez, nosotros crearemos un péndulo bifilar.

Para construir un péndulo bifilar y hallar las inercias seguimos los siguientes pasos:

1. Colgamos de una estructura (o del techo en nuestro caso) el UAV mediante dos hilos de igual longitud y paralelos entre si de forma que quede completamente en horizontal
2. Hacemos girar el UAV en el eje vertical como en la imagen:

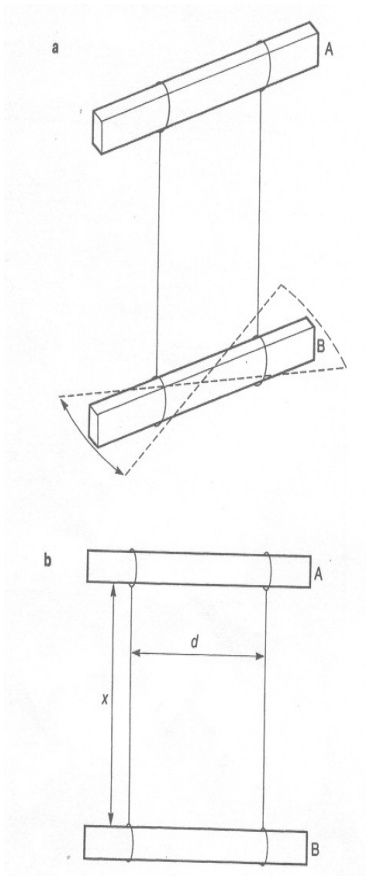

Ilustración 7.1: Péndulo bifilar

3. Contamos 10 oscilaciones y medimos el tiempo para las 3 posiciones.
4. Hallamos el período mediante la ecuación:

$$T = 1 / \ frecuencia$$

en donde:

$$frecuencia = oscilaciones / tiempo$$

5. Una vez que tenemos el período calculamos la inercia en el eje correspondiente mediante la ecuación:

$$I = (m \cdot g \cdot T^2 \cdot b^2) / (4 \cdot pi^2 \cdot L)$$

En donde:

m: masa en kg

g: aceleración de la gravedad en m/s^2

T: período en s

b: distancia entre el centro de masas y los hilos en m

L: longitud de los hilos en m

Las medidas que obtuvimos en las pruebas del experimento para cada eje fueron:

Datos	Ixx	Iyy	Izz
b (m)	0,57	0,5	0,57
L (m)	2,38	1,59	2,33
Tiempo (s)	22,34	10,69	23,06
Oscilaciones	10	10	10

Tabla 7.1: Medidas del experimento

A continuación, se muestran las imágenes del montaje del experimento para realizar las pruebas y poder medir la inercia en cada eje:

- **Ixx**:

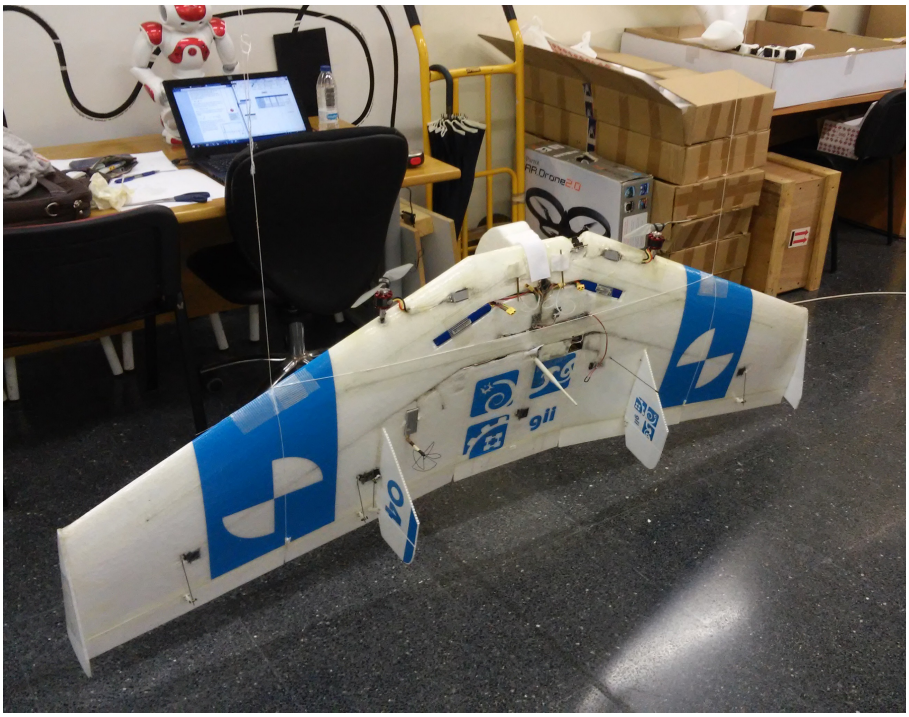

Ilustración 7.2: Posición para la inercia del eje x del UAV (1)

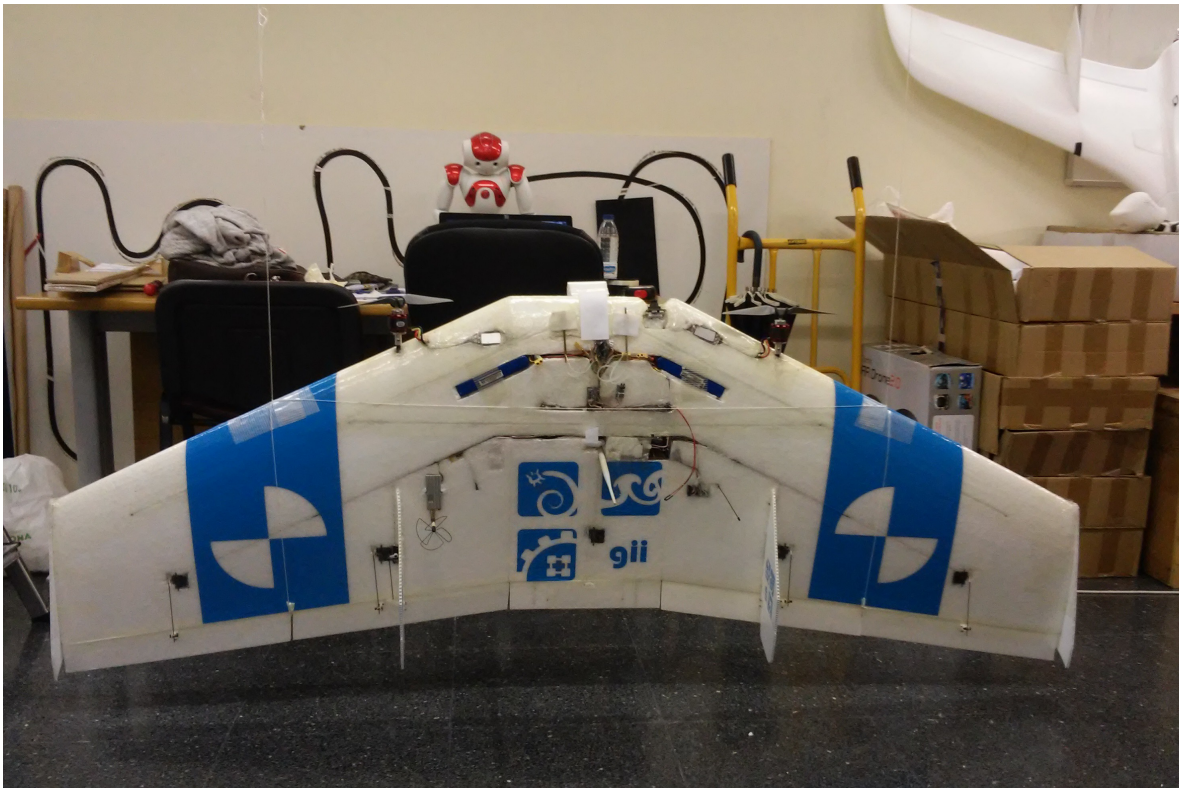

Ilustración 7.3: Posición para el eje x del UAV (2)

- **Iyy:**

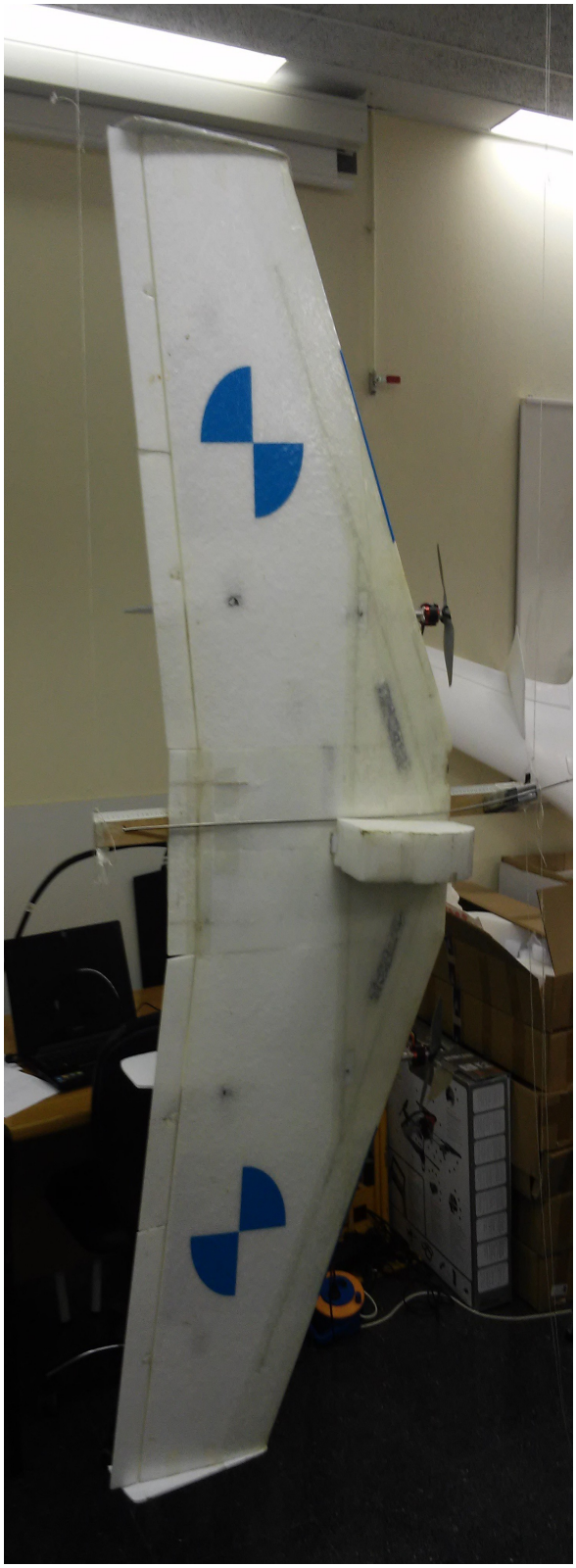

Ilustración 7.4: Posición para el eje Y

- **Izz**:

Ilustración 7.5: Posición para el eje z (1)

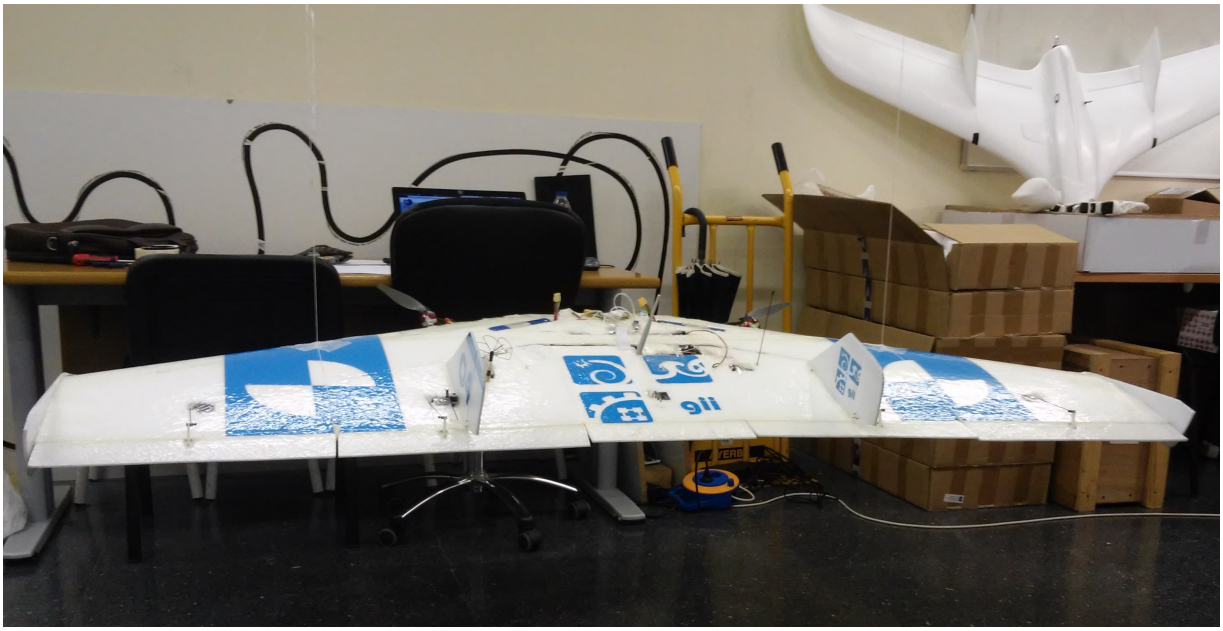

Ilustración 7.6: Posición para el eje z (2)

En cada momento, antes de realizar las pruebas fuimos comprobando el paralelismo de los dos hilos que sostenían el UAV, que fueran equidistantes al centro de masas y que tuvieran la misma longitud hasta el techo con un metro, y la horizontalidad y verticalidad del UAV mediante el uso de un nivel, para que las mediciones fuesen lo más precisas posibles. Además, medimos el tiempo para 10 oscilaciones en cada eje, porque a mayor número de oscilaciones, mayor será la precisión con la que

calculemos el período, de forma que el error del experimento fuese el menor posible.
Los resultados obtenidos fueron:

Datos	Ixx	Iyy	Izz
b (m)	0,57	0,5	0,57
L (m)	2,38	1,59	2,33
Tiempo (s)	22,34	10,69	23,06
Oscilaciones	10	10	10
Frecuencia (Hz)	0,4476	0,9355	0,4337
Periodo (s)	2,234	1,069	2,306
I (kg·m2)	**1,5956**	**0,4208**	**1,7366**

Tabla 7.2: Medidas e inercias resultantes

Datos:

- masa = 3 kg
- gravedad = 9,81 m/s^2

Teniendo en cuenta la definición de momento de inercia (de forma simplificada $I = m \cdot d^2$), analizamos los resultados desde un punto de vista lógico. El eje en el que más fácilmente puede realizar un giro es el de menor momento de inercia, el eje Y, debido a que las masas principales (baterías y motores) están más cerca del eje de rotación Y que pasa por el centro de masas que de los otros 2. Por otro lado, vemos que la inercia en el eje X es ligeramente inferior a la inercia en el eje Z, esto se debe a que las masas principales se encuentran más cerca del eje de rotación X del UAV que del eje Z, por lo tanto, el esfuerzo necesario para iniciar o para detener la rotación en el eje Z será mayor que en el eje X, y mucho mayor que en el eje Y.

7.7. ANEXO VII : Plano del UAV