



Escola Politècnica Superior
de Castelldefels

UNIVERSITAT POLITÈCNICA DE CATALUNYA

TREBALL DE FI DE CARRERA

TÍTOL DEL TFC : Modelado de aeronaves usando las librerías JSBSim

TITULACIÓ: Ingeniería Técnica Aeronáutica, especialidad Aeronavegación

AUTORS: Maria Teresa Marín, Mar Noguera

DIRECTOR: Dagoberto Salazar

DATA: 25 de octubre de 2008

Título : Modelado de aeronaves usando las librerías JSBSim

Autores: Maria Teresa Marín, Mar Noguera

Director: Dagoberto Salazar

Data: 25 de octubre de 2008

Resum

Este Trabajo de Fin de Carrera (TFC) tiene como uno de sus objetivos ser una guía a la hora de utilizar JSBSim. JSBSim es un conjunto de librerías escritas en C++ que permiten simular el comportamiento de una aeronave a partir de unos archivos de configuración que contienen todas las características físicas y aerodinámicas de dicha aeronave.

El proyecto consiste entonces en:

1. Crear los archivos de configuración de un UAV.
2. Aprender a usar JSBSim.
3. Realizar varias simulaciones de vuelo.

Title : Aircraft modeling using JSBSim libraries

Authors: Maria Teresa Marín, Mar Noguera

Director: Dagoberto Salazar

Date: October 25, 2008

Overview

This Final Project is a guide about using JSBSim. JSBSim is a set of libraries written in C++ that allows to simulate the behaviour of an aircraft using configuration files which contains the physical and aerodynamical characteristics of such aircraft.

The Project consists of:

1. Creating configuration files for an UAV.
2. Learning how to use JSBSim.
3. Performing some flight simulations.

Dedicado al 10 de Julio.

ÍNDICE GENERAL

INTRODUCCIÓN	1
CAPÍTULO 1. Primeros pasos en JSBSim	3
1.1. ¿Qué es JSBSim?	3
1.2. ¿A quién está dirigido?	3
1.3. Obtención del Programa	4
1.4. Sistemas de referencia	4
1.4.1. Sistema Ejes - Estructural	4
1.4.2. Sistema Ejes - Cuerpo	5
1.4.3. Sistema Ejes - Estabilidad	5
1.4.4. Sistema Ejes - Viento	5
CAPÍTULO 2. Configurando JSBSim	7
2.1. Fuentes de información	7
2.2. Descripción de los ficheros utilizados	8
2.2.1. Código “XML”	8
2.2.2. Archivos principales de configuración	10
2.2.3. Archivo de configuración principal de la c172x	10
2.2.4. Archivo de configuración del piloto automático de la c172x	20
2.2.5. Archivo de configuración del motor y de la hélice de la c172x	22
2.3. Ejemplo de creación de un avión en JSBSim.	23
2.3.1. Crear la carpeta 1-uav.	23
2.3.2. Parámetros del UAV.	24
2.3.3. Crear archivos del UAV en JSBSim.	24
CAPÍTULO 3. Simulaciones con JSBSim	29
3.1. Descripción de las aeronaves utilizadas	29
3.1.1. Cessna 172	29
3.1.2. OV-10 “Bronco”	30
3.2. Herramientas utilizadas	31
3.2.1. Código Bash	32

3.2.2. Scripts	32
3.2.3. Familiarización con los scripts	32
3.2.4. Script de “limpieza”	34
3.2.5. Generación de gráficos	35
3.3. Simular “FIN” con una Cessna 172	37
3.3.1. Script FIN	37
3.3.2. Gráficas “FIN”	41
3.4. Simular una caída en barrena con una Cessna 172	41
3.4.1. Script “Barrena”	43
3.4.2. Gráficas “Barrena”	44
3.5. Simular un “looping” con un OV-10 “Bronco”	44
3.5.1. Estabilidad y Maniobrabilidad	45
3.5.2. Script “Looping”	46
3.5.3. Gráficas “Looping”	48
3.6. Google Earth	48
CAPÍTULO 4. Simulando con el UAV	53
4.1. Implementar las simulaciones con el UAV.	53
CAPÍTULO 5. Conclusiones	57
GLOSARIO	59
BIBLIOGRAFÍA	61
APÉNDICE A. Cálculo de los momentos y productos de inercia del UAV	63
A.1. Cálculo de los momentos de inercia.	63
A.1.1. Cálculo del momento de inercia en el eje X (I_x)	63
A.1.2. Cálculo del momento de inercia sobre el eje Z (I_z)	64
A.1.3. Cálculo de los productos de inercia	66
APÉNDICE B. Archivos del código del UAV	69
B.1. Archivos de la UAV creados a partir de la Cessna	69
B.1.1. Archivo de configuración principal	69
B.1.2. Archivo del piloto automático	84

B.1.3. Archivo de la hélice	88
B.1.4. Archivo del motor	89
B.2. Archivos de la UAV creados a partir del Aeromatic	89
B.2.1. Archivo principal de configuración	89
B.2.2. Archivo del motor	102
B.2.3. Archivo de la hélice	103
APÉNDICE C. Simulaciones	105
C.1. Simulación “FIN”	105
C.2. Simulación “BARRENA”	110
C.3. Simulación “LOOPING”	112
C.4. Simulación con el UAV	114

ÍNDICE DE FIGURAS

1.1 Logo de JSBSim	3
2.1 Dimensiones del UAV	25
3.1 Tabla de propiedades de la C172	30
3.2 Cessna 172	30
3.3 OV-10 “Bronco”	31
3.4 Tabla de propiedades del OV-10	31
3.5 Ejemplo Gráfica 2D: Tiempo vs. Altura	36
3.6 Ejemplo Gráfica 3D Latitud-Longitud-Altura	37
3.7 Simulación “FIN” en función de la latitud-altitud	41
3.8 Simulación “FIN” en función de la longitud-altitud	42
3.9 Simulación “FIN” en función de la latitud-longitud	42
3.10 Simulación “FIN” en 3D, en función de la latitud-longitud-altura	42
3.11 Simulación “Barrena” en función de la latitud-altitud	44
3.12 Simulación “Barrena” en función de la longitud-altitud	45
3.13 Simulación “Barrena” en función de la latitud-longitud	45
3.14 Simulación “Barrena” en 3D, en función de la latitud-longitud-altura	45
3.15 Simulación “Looping” en función de la latitud-altitud	48
3.16 Simulación “Looping” en función de la longitud-altitud	48
3.17 Simulación “Looping” en función de la latitud-longitud	49
3.18 Simulación “Looping” en 3D, en función de la latitud-longitud-altura	49
3.19 Simulación del “looping” en Google Earth	51
4.1 Parte de la web de JSBSim	54
4.2 Simulación realizada con el UAV	55
A.1 Medidas del UAV	64

ÍNDICE DE TABLAS

A.1 Momentos de Inercia en el Eje X	65
A.2 Cálculo de Steiner en el Eje Z	65
A.3 Momento de Inercia en el Eje Z	66
A.4 Momentos de Inercia y distancias a los ejes X y Z	68
A.5 Producto de Inercia respecto al Eje XZ	68

INTRODUCCIÓN

En este proyecto se pretende crear virtualmente un UAV (Unmanned Aerial Vehicle, vehículo aéreo no tripulado), para posteriormente poder hacer simulaciones con él. De esta forma se podrá estudiar cómo el avión responde frente a determinados estímulos y situaciones.

Para poder poner en marcha esta fase del desarrollo del UAV se ha utilizado el programa de código abierto JSBSim. JSBSim es un conjunto de librerías escritas en C++ que permiten simular el comportamiento de una aeronave a partir de unos archivos de configuración que contienen todas las características físicas y aerodinámicas de dicha aeronave.

Los objetivos generales del proyecto consisten en aprender a usar JSBSim, completar los datos físicos y aerodinámicos del UAV, introducir los datos en los archivos de configuración adecuados, y realizar varias simulaciones de vuelo.

Para poder realizar esta tarea ha sido necesario seguir una serie de pasos esenciales antes de poder crear y posteriormente simular una aeronave con JSBSim.

Se podría resumir cada paso del proyecto en los siguientes puntos:

1. Descripción de los ficheros utilizados.
2. Familiarización con los *Scripts*.
3. Realización de los tutoriales *Bash*, *XML* y *Gnuplot*.
4. Creación del script **limpieza.xml**.
5. Creación de las simulaciones.
6. Ejemplo de creación de un avión en JSBSim.
 - * Crear la carpeta **1-uav**.
 - * Medidas del UAV.
 - * Crear archivos del UAV en JSBSim.
7. Usar la nueva versión de JSBSim.
8. Implementar las simulaciones en *Google Earth*.

El proyecto está dividido en 4 capítulos:

1. Primeros pasos en JSBSim.
2. Configurando JSBSim.
3. Aprendiendo a simular.
4. Simulando el UAV en JSBSim.

En el primero de ellos se hace una breve explicación histórica de JSBSim y de su funcionamiento. También se explica cómo se puede instalar y los pasos a seguir para ello.

En el segundo capítulo se definen los archivos básicos de configuración de las aeronaves en JSBSim y se muestra la creación de un avión, en este caso un UAV (primer y sexto puntos de los pasos indicados anteriormente).

En el tercer capítulo se explican las aplicaciones utilizadas para la elaboración y comprensión de las simulaciones. A la vez se describe cómo realizar nuevas simulaciones a partir de varios ejemplos.

En el cuarto capítulo se describe cómo se han adaptado las simulaciones del tercer punto al UAV y los problemas que han surgido durante el proceso.

Finalmente se concluirá el trabajo con una valoración de los nuevos conocimientos adquiridos.

CAPÍTULO 1. PRIMEROS PASOS EN JSBSIM

1.1. ¿Qué es JSBSim?

JSBSim fue creado en 1996 como una aplicación sencilla en código abierto, controlada por datos, no lineal, con seis grados de libertad (6DOF), con el objeto de simular el control y la dinámica de vuelo de una aeronave.

Desde su creación JSBSim se ha beneficiado del entorno de código abierto (open source development environment) y ha crecido con la ayuda de una gran variedad de usuarios que han ido contribuyendo con nuevas ideas a su creación y desarrollo.

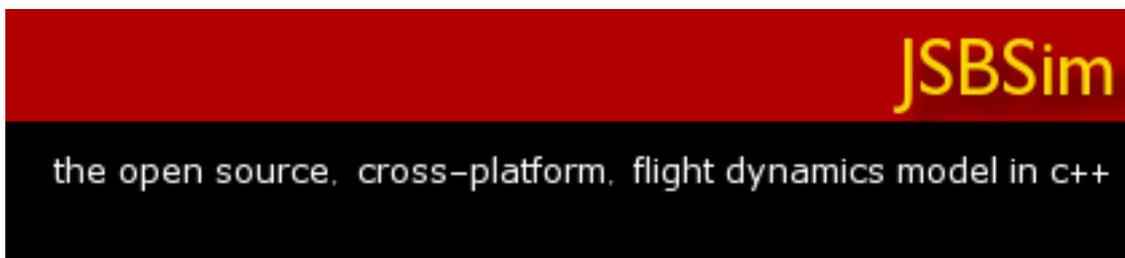


Figura 1.1: Logo de JSBSim

JSBSim está constituido por una colección de programas (bloques) escritos en su gran mayoría en lenguaje C++, incluyendo algunas rutinas del lenguaje C. Algunos de los bloques de C++ que están comprendidos en JSBSim son para simular la atmósfera, un sistema de control de vuelo o un motor. Otros encierran construcciones matemáticas como por ejemplo la ecuación del movimiento, una matriz e incluso un vector.

Todos estos elementos juntos hacen que JSBSim, apartir de los “inputs” (entradas de datos) de control, calcule y sume las fuerzas y momentos que se obtienen de ellos y su entorno, obteniendo de esta forma el movimiento del vehículo (velocidad, orientación, posición, etc.) en escalones discretos de tiempo.

JSBSim ha sido compilado en una gran variedad de plataformas, tanto en Linux como en Windows, y es igual de eficaz en ordenadores Macintosh.

1.2. ¿A quién está dirigido?

La arquitectura de JSBSim está pensada para su fácil comprensión y está diseñada para ser lo más útil posible para los estudiantes de aeronáutica. Debido a su fácil configuración también está preparada para su utilización por profesionales de áreas diferentes.

1.3. Obtención del Programa

El código fuente del JSBSim puede ser descargado como un “release”, es decir, un conjunto de archivos de código fuente que se hicieron públicos en algún momento cuando se consideró que el programa era estable. Este tipo de “releases” se van preparando ocasionalmente a criterio del programador. El código fuente puede descargarse en:

http://sourceforge.net/project/showfiles.php?group_id=19399

El código fuente también puede obtenerse directamente gestor de revisiones utilizando una herramienta denominada CVS (Concurrent Version System). CVS está disponible como herramienta opcional en entorno Cygwin bajo Windows, pero es habitual encontrarla en sistemas Linux.

Las instrucciones para descargar JSBSim desde el gestor de revisiones pueden encontrarse en la página web del proyecto en:

<http://www.jsbsim.org>

Cuando se baja el código fuente directamente desde el gestor de revisiones, es necesario compilarlo. Las instrucciones para hacer esto también se encuentran en la URL anterior. En el mismo sitio se pueden descargar archivos ejecutables para algunas plataformas (Windows, Cygwin, Linux, IRIX, etc.).

Debe tenerse en cuenta que actualmente la aeronave de ejemplo y las rutinas deben ser descargadas como parte del código fuente del “release”.

1.4. Sistemas de referencia

JSBSim permite diseñar las aeronaves según los sistemas de referencia elegidos. Entre ellos tenemos:

1. Estructural.
2. Cuerpo.
3. Estabilidad.
4. Viento.

1.4.1. Sistema Ejes - Estructural

Este sistema es el más común de todos, y se utiliza para definir puntos de importancia en el avión, como por ejemplo el centro de gravedad (CG), la localización del tren de aterrizaje, el punto de vista del piloto, distintas masas puntuales, los motores, etc.

El eje X tiene sentido creciente desde el morro hasta la cola. El eje Y es positivo desde el fuselaje hacia la derecha (en el sentido del vuelo) y por supuesto el eje Z es positivo hacia arriba. Normalmente el origen de este sistema de referencia (Datum) está en el frente del avión, ya sea en el extremo del morro, o a cierta distancia delante del mismo.

Por otro lado, el eje X normalmente coincide con el eje de simetría del fuselaje y pasa a través del cono de la hélice (de existir ésta). Todas las posiciones referentes al eje X se denominan "stations". Las posiciones referentes al eje Z se denominan "waterline positions". Por último, las posiciones a lo largo del eje Y se denominan "buttline positions".

1.4.2. Sistema Ejes - Cuerpo

El sistema de ejes "cuerpo" es similar al sistema "estructural" pero rotándolo 180 grados respecto al eje Y, con el origen en el centro de gravedad.

Es en referencia a este sistema de ejes donde las fuerzas del avión y los momentos son sumados, y la aceleración resultante es integrada para obtener las velocidades.

1.4.3. Sistema Ejes - Estabilidad

Este sistema es similar al de ejes "cuerpo", excepto que el eje X apunta según la proyección del vector velocidad-viento sobre el plano XY. El eje Y se mantiene en la misma posición y el eje Z completa el triedro dextrógiro (a derechas).

1.4.4. Sistema Ejes - Viento

Este sistema es similar al anterior, con la diferencia de que el eje X apunta directamente según el vector velocidad viento (velocidad aerodinámica real del avión). El eje Z está contenido en el plano de simetría del avión (plano XZ en ejes cuerpo). El eje Y completa el triedro dextrógiro.

CAPÍTULO 2. CONFIGURANDO JSBSIM

En este capítulo se describe como está estructurada una aeronave en JSBSim, además de los pasos a seguir para crear una nueva.

2.1. Fuentes de información

Antes de poder empezar cualquier tarea del proyecto, es importante conocer con detalle cómo funciona JSBSim. Para ello, primero se optó por la comprensión y aprendizaje de los ficheros descriptivos de las aeronaves.

JSBSim incluye una serie de aeronaves reales, dadas por defecto, que siguen un formato de ficheros descriptivos similares. Se optó por empezar el trabajo con los archivos descriptivos de una Cessna 172x (c172x), puesto que al ser una de las aeronaves más conocidas, de ella se puede obtener más información aerodinámica y estructural que de muchos otros modelos.

Una vez escogido el avión que se iba a utilizar como modelo, sólo quedaba encontrar un manual o información donde se explique cómo se crea un avión en JSBSim, o que ayude a entender dichos archivos descriptivos.

En la web oficial de JSBSim¹ existe un manual en el apartado “*Documentation*” que incluye este tipo de información.

Cuando este proyecto comenzó, el citado manual estaba disponible en una versión antigua y por lo tanto no tan completa como la que necesitábamos. Entonces, para poder obtener más información sobre los archivos de JSBSim también se recurrió a otros manuales encontrados en la red:

- a. <http://en.wikipedia.org/wiki/JSBSim>
- b. <http://jsbsim.wiki.sourceforge.net/>

Sin embargo, desgraciadamente para nuestros fines todos ellos procedían de la fuente original, por lo que no aportaban mucha ayuda.

Puesto que la información de la que se disponía estaba incompleta, se decidió pedir ayuda a los creadores de JSBSim. Se envió un e-mail a cada uno de los 8 miembros solicitando la posibilidad de obtener alguna otra información que complementara la que ya se tenía obtenida en la red.

Al poco tiempo, varios de los creadores respondieron a las cuestiones planteadas. Todos los miembros que contestaron coincidieron en que la información publicada en la web era pobre y que JSBSim está poco documentado.

A pesar de la falta de información, se intentó entender y explicar esos archivos. Todas las

¹<http://www.jsbsim.org>

partes de los mismos que no se explicaban en los manuales se han ido dejando de lado para después consultarlas con libros de aerodinámica o con profesionales en la materia.

Una de las personas a las que se consultó, aparte de los creadores de JSBSim, fue *Alex Ruíz*, piloto y mecánico de la compañía aérea **Vueling**. El señor Ruíz ayudó a comprender el archivo principal de descripción de una aeronave, y el archivo dónde se describe el piloto automático de dicha aeronave.

Finalmente, los creadores del programa publicaron un nuevo manual, más completo que las fuentes de información anteriores, para entender JSBSim y los archivos que lo componen. De esta forma, las partes del proyecto cuya comprensión y análisis no resultaba nada fácil pudieron ser utilizadas para los fines de este trabajo, aunque es necesario indicar que todavía hay puntos por aclarar en este último manual.

2.2. Descripción de los ficheros utilizados

Los archivos que definen las características de las aeronaves en JSBSim tienen unas características muy concretas y están escritos en el lenguaje **XML**, que se describe a continuación.

2.2.1. Código “XML”

XML proviene de las siglas en inglés *Extensible Markup Language* (“lenguaje de marcas ampliable”)², es un metalenguaje³ extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C).

Es una simplificación y adaptación del SGML⁴ y permite definir la gramática de lenguajes específicos.

Por lo tanto, XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG y MathML.

La tecnología XML busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible. Que la información sea estructurada quiere decir que se compone de partes bien definidas, y que esas partes se componen a su vez de otras partes. Entonces se tiene un árbol de pedazos de información.

En XML la información habitualmente se encuentra organizada en “elementos”, que están definidos mediante un par de “etiquetas” de apertura y cierre. Por ejemplo:

²Lenguaje de marcas es una forma de codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación.

³Metalenguaje es un lenguaje usado para hacer referencia a otros lenguajes.

⁴SGML son las siglas de *Standard Generalized Markup Language* o “Lenguaje de Marcación Generalizado”. Consiste en un sistema para la organización y etiquetado de documentos.

```
<description>Models a 1997 Cessna 172R.</description>
```

Nótese que el nombre del elemento es *description*, y que en la etiqueta de cierre el nombre del elemento está precedido por una barra inclinada (</description>).

Como la información está estructurada formando una especie de árbol, entonces se tiene un elemento “raíz” que envuelve a los demás elementos:

```
<fileheader>
  <author>Tony Peden</author>
  <filecreationdate>1999-01-01</filecreationdate>
  <description>Models a 1997 Cessna 172R.</description>
</fileheader>
```

La estructura general de un elemento completo es la siguiente:

```
<nombre_elemento nombre_atributo="valor_atributo">
  Contenido del elemento
</nombre_elemento>
```

Se puede ver que, a parte del nombre del elemento y su contenido, se pueden especificar “atributos” que proporcionan información adicional.

De esta manera en JSBSim se puede indicar, por ejemplo:

```
<mass_balance>
  <pointmass name="PILOT">
    <weight unit="LBS">180.0</weight>
    <location name="POINTMASS" unit="IN">
      <x> 36.0 </x>
      <y> -14.0 </y>
      <z> 24.0 </z>
    </location>
  </pointmass>
</mass_balance>
```

Que indica que el peso del piloto son 180 libras, y está sentado (con respecto al “datum” de la aeronave) a 36 pulgadas en la dirección X, -14 pulgadas en la dirección Y, y 24 pulgadas en la dirección Z. Las unidades usadas han sido proporcionadas en forma de “atributos” del elemento en cuestión.

Finalmente, es posible insertar comentarios en un archivo XML mediante la sintaxis:

```
<!-- Esto es un comentario. -->
```

2.2.2. Archivos principales de configuración

Todo avión de JSBSim está compuesto por 4 archivos principales:

1. El archivo que define la configuración principal del avión.
2. El archivo que define el piloto automático del avión.
3. El archivo que define el motor del avión.
4. El archivo que define la propulsión del avión.

En el primero de ellos se define el avión, tanto estructuralmente como aerodinámicamente. En el segundo se declara el piloto automático del avión a crear. En el tercero se describe el motor del avión, y en el cuarto la hélice o tobera del mismo.

A continuación se procede a una explicación más detallada de las partes principales de cada uno de los archivos citados.

Dado que hemos tomado como archivo base los de la c172x para entender como se crea en JSBSim una aeronave, las explicaciones y ejemplos dados en este punto se refieren a dicho avión.

2.2.3. Archivo de configuración principal de la c172x

Este archivo esta formado por 8 secciones:

- A. Encabezado.
- B. Medidas.
- C. Balance de masas.
- D. Reacciones en tierra.
- E. Propulsión.
- F. Piloto automático.
- G. Control de vuelo.
- H. Aerodinámica.
- I. Otros.

2.2.3.1. Encabezado

Esta sección incluye información sobre quién ha creado el modelo de avión, cuándo, qué versión es, qué licencias se usan, qué referencias han sido usadas al crear este modelo, notas y limitaciones. Por ejemplo, en la c172x encontramos:

```
<fileheader>
  <author>Tony Peden</author>
  <license>
    <licenseName>GPL</licenseName>
    <licenseURL>http://www.gnu.org/licenses/gpl.html</licenseURL>
  </license>
  <filecreationdate>1999-01-01</filecreationdate>
  <version>$Revision: 1.65 $</version>
  <description>Models a 1997 Cessna 172R.</description>
</fileheader>
```

También se puede añadir información extra como:

E-mail y organización:

```
<email> e-mail del creador </email>
<organization> lugar de trabajo del autor </organization>
```

Notas que quiera añadir el autor que puedan ser interesantes:

```
<note> ... </note>
<note> ... </note>
```

Limitaciones que tenga el avión que se está creando:

```
<limitation> Spoilers not modeled </limitation>
<limitation> ... </limitation>
```

Referencias utilizadas como ayuda para el diseño de este avión:

```
<reference refID="identificador"
  author="autor del libro/web..."
  title="titulo del libro/web... "
  date="fecha del libro/web"/>
```

2.2.3.2. Medidas

En esta sección se definen las medidas características del vehículo y las localizaciones de los puntos clave. Por ejemplo, en la c172x:

```
<metrics>
```

```

<wingarea unit="FT2"> 174.0 </wingarea>
<wingspan unit="FT"> 35.8 </wingspan>
<chord unit="FT"> 4.9 </chord>
<htailarea unit="FT2"> 21.9 </htailarea>
<htailarm unit="FT"> 15.7 </htailarm>
<vtailarea unit="FT2"> 16.5 </vtailarea>
<vtailarm unit="FT"> 15.7 </vtailarm>
<location name="AERORP" unit="IN">
  <x> 43.2 </x>
  <y> 0.0 </y>
  <z> 59.4 </z>
</location>
<location name="EYEPOINT" unit="IN">
  <x> 37.0 </x>
  <y> 0.0 </y>
  <z> 48.0 </z>
</location>
<location name="VRP" unit="IN">
  <x> 42.6 </x>
  <y> 0.0 </y>
  <z> 38.5 </z>
</location>
</metrics>

```

NOTA: Es importante remarcar que en cada apartado se tienen que especificar el tipo de unidades utilizadas (*unit*).

2.2.3.3. Balance de masas

Aquí es dónde se definen las características sobre las masas del avión, estando incluidas, por ejemplo, la masa del avión en vacío, los momentos y productos de inercia, la localización del centro de gravedad, y cualquier otra definición al respecto.

Como ejemplo, en la c172x encontramos:

```

<mass_balance>
  <ixx unit="SLUG*FT2"> 948.0 </ixx>
  <iyy unit="SLUG*FT2"> 1346.0 </iyy>
  <izz unit="SLUG*FT2"> 1967.0 </izz>
  <ixz unit="SLUG*FT2"> 0.0 </ixz>
  <emptywt unit="LBS"> 1620.0 </emptywt>
  <location name="CG" unit="IN">
    <x> 41.0 </x>
    <y> 0.0 </y>
    <z> 36.5 </z>
  </location>
  <pointmass name="PILOT">

```

```

    <weight unit="LBS">180.0</weight>
    <location name="POINTMASS" unit="IN">
        <x> 36.0 </x>
        <y> -14.0 </y>
        <z> 24.0 </z>
    </location>
</pointmass>
<pointmass name="CO-PILOT">
    <weight unit="LBS">180.0</weight>
    <location name="POINTMASS" unit="IN">
        <x> 36.0 </x>
        <y> 14.0 </y>
        <z> 24.0 </z>
    </location>
</pointmass>
<pointmass name="LUGGAGE">
    <weight unit="LBS">120.0</weight>
    <location name="POINTMASS" unit="IN">
        <x> 95.0 </x>
        <y> 0.0 </y>
        <z> 24.0 </z>
    </location>
</pointmass>
</mass_balance>

```

A continuación se adjunta como quedaría el código general, para cualquier avión:

```

<mass_balance>
    <ixx unit="{SLUG*FT2 | KG*M2}"> {number} </ixx>
    <iyy unit="{SLUG*FT2 | KG*M2}"> {number} </iyy>
    <izz unit="{SLUG*FT2 | KG*M2}"> {number} </izz>
    <ixy unit="{SLUG*FT2 | KG*M2}"> {number} </ixy>
    <ixz unit="{SLUG*FT2 | KG*M2}"> {number} </ixz>
    <iyz unit="{SLUG*FT2 | KG*M2}"> {number} </iyz>
    <emptywt unit="{LBS | KG}"> {number} </emptywt>
    <location name="CG" unit="{IN | M}">
        <x> {number} </x>
        <y> {number} </y>
        <z> {number} </z>
    </location>
    <pointmass name="{string}">
        <weight unit="{LBS | KG}"> {number} </weight>
        <location name="POINTMASS" unit="{IN | M}">
            <x> {number} </x>
            <y> {number} </y>
            <z> {number} </z>
        </location>
    </pointmass>

```

```

    ... other point masses ...
</mass_balance>

```

2.2.3.4. Reacciones en tierra

En este apartado se explican los diferentes casos en los que el avión puede contactar con la superficie del suelo: contacto normal, aterrizaje, impacto violento, etc., y cómo es su respuesta a dichos posibles contactos⁵.

El código general será:

```

<contact type="{BOGEY | STRUCTURE}" name="{string}">
  <location unit="{IN | M}">
    <x> {number} </x>
    <y> {number} </y>
    <z> {number} </z>
  </location>
  <static_friction> {number} </static_friction>
  <dynamic_friction> {number} </dynamic_friction>
  <rolling_friction> {number} </rolling_friction>
  <spring_coeff unit="{LBS/FT | N/M}"> {number} </spring_coeff>
  <damping_coeff unit="{LBS/FT/SEC | N/M/SEC}">
    {number}
  </damping_coeff>
  <damping_coeff_rebound unit="{LBS/FT/SEC | N/M/SEC}">
    {number}
  </damping_coeff_rebound>
  <max_steer unit="DEG"> {number | 0 | 360} </max_steer>
  <brake_group> {NONE|LEFT|RIGHT|CENTER|NOSE|TAIL} </brake_group>
  <retractable>{0 | 1}</retractable>
  <table type="{CORNERING_COEFF}">
  </table>
  <relaxation_velocity>
    <rolling unit="{FT/SEC | KTS | M/S}"> {number} </rolling>
    <side unit="{FT/SEC | KTS | M/S}"> {number} </side>
  </relaxation_velocity>
  <force_lag_filter>
    <rolling> {number} </rolling>
    <side> {number} </side>
  </force_lag_filter>
  <wheel_slip_filter> {number} </wheel_slip_filter>

```

⁵Puesto que sólo hay dos partes del avión que pueden hacer contacto con el suelo, las ruedas o la estructura, sólo se definen dos tipos de contacto: BOGEY y STRUCTURE.

“STRUCTURE” se refiere a cuando es una parte de la estructura (alas o fuselaje, por ejemplo) la que toca el suelo, mientras que “BOGEY” se refiere al contacto con las ruedas.

Por lo tanto, si el tipo de contacto es BOGEY es importante tener en cuenta el posterior movimiento de avance que originará el giro de las ruedas.

```
</contact>
```

Como ejemplo, en la c172x habría que definir el contacto de las tres ruedas, la del tren delantero y las del tren principal.

Se adjunta solo el código que define el contacto del tren delantero:

```
<contact type="BOGEY" name="Nose Gear">
  <location unit="IN">
    <x> -6.8 </x>
    <y> 0.0 </y>
    <z> -20.0 </z>
  </location>
  <static_friction>0.8</static_friction>
  <dynamic_friction>0.5</dynamic_friction>
  <rolling_friction>0.02</rolling_friction>
  <spring_coeff unit="LBS/FT">1800</spring_coeff>
  <damping_coeff unit="LBS/FT/SEC">600</damping_coeff>
  <relaxation_velocity>
    <rolling unit="FT/SEC">0.75</rolling>
    <side unit="FT/SEC">0.75</side>
  </relaxation_velocity>
  <force_lag_filter>
    <rolling> 120 </rolling>
    <side> 120 </side>
  </force_lag_filter>
  <wheel_slip_filter> 120 </wheel_slip_filter>
  <max_steer unit="DEG">10</max_steer>
  <brake_group>NONE</brake_group>
  <retractable>0</retractable>
</contact>
```

Pero como es explico anteriormente, no sólo se tiene que explicar el contacto de las ruedas con el suelo, también puede ser que el avión choque o roce el suelo con la cola, o con la punta de las alas. JSBSim también tiene en cuenta esta posibilidad.

A continuación se describe el contacto con un patín de cola:

```
<contact type="STRUCTURE" name="TAIL_SKID">
  <location unit="IN">
    <x> 188.0 </x>
    <y> 0.0 </y>
    <z> 8.0 </z>
  </location>
  <static_friction>0.2</static_friction>
  <dynamic_friction>0.2</dynamic_friction>
  <spring_coeff unit="LBS/FT">20000</spring_coeff>
  <damping_coeff unit="LBS/FT/SEC">1000</damping_coeff>
```

```
</contact>
```

En este ejemplo es importante remarcar que no hay que añadir toda la parte relacionada con la velocidad (como en el caso del tren de aterrizaje), puesto que al ser un contacto estructural (STRUCTURE) no se tiene que declarar todo lo relativo a un posible giro de las ruedas.

2.2.3.5. Propulsión

En este apartado se hace referencia a los archivos en los que se definen las características correspondientes al motor, por un lado, y a su hélice por el otro⁶.

En el caso de la c172x será como se indica a continuación:

```
<propulsion>
  <engine file="engIO360C">
    <location unit="IN">
      <x>-19.7</x>
      <y>0</y>
      <z>26.6</z>
    </location>
    <orient unit="DEG">
      <roll>0</roll>
      <pitch>0</pitch>
      <yaw>0</yaw>
    </orient>
    <feed>0</feed>
    <feed>1</feed>
    <thruster file="prop_Clark_Y7570">
      <location unit="IN">
        <x>-37.7</x>
        <y>0</y>
        <z>26.6</z>
      </location>
      <orient unit="DEG">
        <roll>0</roll>
        <pitch>0</pitch>
        <yaw>0</yaw>
      </orient>
      <sense>1</sense>
      <p_factor>10.0</p_factor>
    </thruster>
  </engine>
  <tank type="FUEL">    <!-- Tank number 0 -->
    <location unit="IN">
```

⁶Los archivos sobre el motor y sobre la hélice se encuentran dentro de JSBSim en la carpeta "ENGINE"; en dicha carpeta además se encuentran definidos todos los motores de las aeronaves existentes en JSBSim.

```

        <x>48.0</x>
        <y>-112.0</y>
        <z>59.4</z>
    </location>
    <radius unit="IN">29.4</radius>
    <capacity unit="LBS">168</capacity>
    <contents unit="LBS">168</contents>
</tank>
<tank type="FUEL">    <!-- Tank number 1 -->
    <location unit="IN">
        <x>48.0</x>
        <y>112.0</y>
        <z>59.4</z>
    </location>
    <radius unit="IN">29.4</radius>
    <capacity unit="LBS">168</capacity>
    <contents unit="LBS">168</contents>
</tank>
</propulsion>

```

Se puede ver en este ejemplo que no sólo se llama a los archivos de motor (“engIO360C”) y propulsión (“prop.Clark.Y7570”) de la aeronave, sino que también están definidas su situación y orientación en la misma.

Además, también se define la cantidad de combustible usada y la capacidad de los depósitos del avión.

2.2.3.6. *Piloto automático*

En este apartado sólo se hace referencia, como en la sección 2.2.3.5., al archivo donde está definido el piloto automático⁷.

La línea de código utilizada es la siguiente:

```
<autopilot file="c172ap"/>
```

2.2.3.7. *Control de Vuelo*

En esta sección se definen las diferentes las estructuras del avión que sirven para controlar su movimiento en las diferentes fases del vuelo. Éstas puede ser el timón de profundidad (elevator), el timón de dirección (rudder), los alerones (ailerons), flaps, etc.

A continuación se incluye parte del código que describe el movimiento y control de los flaps:

⁷Este archivo se encuentra en la misma carpeta del archivo principal de la aeronave.

```

<channel name="Flaps">
  <kinematic name="Flaps Control">
    <input>fcs/flap-cmd-norm</input>
    <traverse>
      <setting>
        <position> 0 </position>
        <time> 0 </time>
      </setting>
      <setting>
        <position> 10 </position>
        <time> 2 </time>
      </setting>
      <setting>
        <position> 20 </position>
        <time> 1 </time>
      </setting>
      <setting>
        <position> 30 </position>
        <time> 1 </time>
      </setting>
    </traverse>
    <output>fcs/flap-pos-deg</output>
  </kinematic>
  <aerosurface_scale name="Flap Position Normalizer">
    <input>fcs/flap-pos-deg</input>
    <domain>
      <min>0</min>
      <max>30</max>
    </domain>
    <range>
      <min>0</min>
      <max>1</max>
    </range>
    <output>fcs/flap-pos-norm</output>
  </aerosurface_scale>
</channel>

```

Se puede ver que primero se definen las diferentes posiciones de los flaps, así como el tiempo de respuesta en alcanzar dichas posiciones.

En la siguiente parte (denominada "Flap Position Normalizer") se define la posición de los flaps, pero normalizada (entrada en el rango [0,30], con salida dentro del rango [0,1]).

2.2.3.8. Aerodinámica

Aquí se describe la respuesta del avión a las diferentes posiciones de los controles de vuelo.

A continuación se incluye un ejemplo de cómo se define una tabla de propiedades aerodinámicas. Específicamente, se define la tabla de respuesta de la resistencia aerodinámica de la c172x en función los diferentes ángulos de ataque ("alpha"), según sea la posición de los flaps.

```
<function name="aero/coefficient/CDwbh">
  <description>Drag due to alpha</description>
  <product>
    <property>aero/function/ground-effect-factor-drag</property>
    <property>aero/qbar-area</property>
    <table>
      <independentVar lookup="row">aero/alpha-rad</independentVar>
      <independentVar lookup="column">fcs/flap-pos-deg</independentVar>
      <tableData>
        0.0          10.0         20.0         30.0
        -0.0872664  0.00407143  3.8165e-05  0.000512444  0.00137231
        -0.0698132  0.0013424  0.000440634  0.00248067  0.00412341
        -0.0523599  8.96747e-05  0.00231942  0.0059252  0.00835082
        -0.0349066  0.000313268  0.00567451  0.0108461  0.0140545
        -0.0174533  0.00201318  0.0105059  0.0172432  0.0212346
        0.0          0.0051894  0.0168137  0.0251167  0.0298909
        0.0174533  0.00993967  0.0247521  0.0346492  0.0402205
        0.0349066  0.0162201  0.0342207  0.0457119  0.0520802
        0.0523599  0.0240308  0.0452195  0.0583047  0.0654701
        0.0698132  0.0333717  0.0577485  0.0724278  0.0803902
        0.0872664  0.0442427  0.0718077  0.088081  0.0968405
        0.10472     0.056644  0.0873971  0.105265  0.114821
        0.122173   0.0705754  0.104517  0.123978  0.134332
        0.139626   0.086037  0.123166  0.144222  0.155373
        0.15708    0.096239  0.135317  0.157346  0.168984
        0.174533   0.106859  0.147857  0.170848  0.182966
        0.191986   0.118034  0.160954  0.184905  0.197503
        0.209439   0.129765  0.174606  0.199517  0.212596
        0.226893   0.142366  0.189176  0.215072  0.228643
        0.244346   0.156452  0.205363  0.23231  0.246406
        0.261799   0.172732  0.223956  0.252059  0.266733
        0.279253   0.178209  0.230187  0.258667  0.273529
        0.296706   0.171598  0.222665  0.25069  0.265325
        0.314159   0.161786  0.211468  0.2388  0.253089
        0.331613   0.147493  0.19508  0.221364  0.235129
        0.349066   0.109678  0.151171  0.174408  0.186649
      </tableData>
    </table>
  </product>
</function>
```

2.2.3.9. Otros

Para terminar, se proporciona la relación de las diferentes “propiedades”⁸ creadas y utilizadas a lo largo de este fichero de configuración principal de la c172x, y que pueden ser almacenadas en archivos de salida.

Es importante enfatizar que, a parte de las secciones aquí definidas, JSBSim también permite la posibilidad de modelar secciones para el diseño de un amplio tipo de objetos, como pueden ser (además de todo tipo de aeronaves) globos aeroestáticos, cohetes, pelotas, proyectiles, etc.

Por ejemplo, para el caso de los globos sólo se tiene que incluir una sección denominada *Buoyant Forces*, en donde se define todo tipo de datos atmosféricos como la densidad del aire, el calor, etc.

También se puede añadir una última sección llamada *External Reactions*⁹. En ella se definen los elementos externos que pueden afectar a la estructura de la aeronave o a su movimiento, como pueden ser rafagas de viento, objetos con posibilidad de colisión, etc.

2.2.4. Archivo de configuración del piloto automático de la c172x

Una de las grandes ventajas de JSBSim es que permite el vuelo automático, lo cual en el avión se consigue mediante un piloto automático. El trabajo de éste consiste en mantener en todo momento las características de vuelo requeridas: rumbo, altitud, ángulo de ataque, etc., y en caso de que alguno de estos parámetros sea alterado, el piloto automático debe corregirlo devolviendo la aeronave al estado esperado, lo cual traducido en código es llevar a cero las variables afectadas.

En su definición se han de tener en cuenta todos los “inputs” de entrada, tanto internos (las ordenes desde cabina, variación de la carga de combustible por tiempo de vuelo, par torsor de los motores, desequilibrios estructurales, etc.) como externos (corrientes de aire, variaciones de presión, etc.), de modo que la respuesta a las posibles variaciones sea la más exacta posible.

Analizando el archivo donde se define el piloto automático se pueden identificar claramente tres partes principales.

- Sensores.
- Roll Channel.
- Pitch Channel.

La primera parte es donde se declaran las propiedades que se requieren para los diversos

⁸En JSBSim, el término “property” se refiere a parámetros que pueden ser leídos, calculados y/o manipulados por las funciones del simulador.

⁹En este proyecto no se profundiza en las secciones “Buoyant Forces” y “External Reactions”.

sensores. En el caso de la c172x sólo se define un sensor, que es el sensor que calcula la actitud de la aeronave:

```
<sensor name="fcs/attitude/sensor/phi-rad">
  <input> attitude/phi-rad </input>
  <lag> 0.5 </lag>
  <noise variation="PERCENT"> 0.05 </noise>
  <quantization name="attitude/sensor/quantized/phi-rad">
    <bits> 12 </bits>
    <min> -180 </min>
    <max> 180 </max>
  </quantization>
  <bias> 0.001 </bias>
</sensor>
```

En la segunda y tercera parte se describen las diferentes funciones del piloto automático que forman parte de la actuación de los canales de alabeo ("Roll Channel": wing leveler, heading hold) y de picado ("Pitch Channel": altitude hold).

Como ejemplo se expone el código utilizado para controlar el ángulo de alabeo ("wing leveler").

```
<channel name="fcs/ap-roll-wing-leveler">
  <pure_gain name="fcs/roll-ap-wing-leveler">
    <input>fcs/attitude/sensor/phi-rad</input>
    <gain>2.0</gain>
    <clipto>
      <min>-0.255</min>
      <max>0.255</max>
    </clipto>
  </pure_gain>

  <integrator name="fcs/roll-ap-error-integrator">
    <input> attitude/phi-rad </input>
    <c1> 0.125 </c1>
  </integrator>

  <summer name="fcs/roll-ap-error-summer">
    <input> velocities/p-rad_sec</input>
    <input> fcs/roll-ap-wing-leveler</input>
    <input> fcs/roll-ap-error-integrator</input>
    <clipto>
      <min>-1.0</min>
      <max> 1.0</max>
    </clipto>
  </summer>

  <switch name="fcs/roll-ap-autoswitch">
```

```
<default value="0.0"/>
<test value="fcs/roll-ap-error-summer">
  ap/attitude_hold == 1
</test>
</switch>

<pure_gain name="fcs/roll-ap-aileron-command-normalizer">
  <input>fcs/roll-ap-autoswitch</input>
  <gain>-1</gain>
</pure_gain>
</channel>
```

Como se puede ver, en primer lugar se definen cuales serán los máximos y los mínimos operativos.

Seguidamente se definen los posibles desajustes y errores que, por construcción, se pueden dar en el “wing leveler”. Se utiliza un circuito integrador en el lazo de control y su salida se combina en un sumador que une varias entradas. Finalmente, se define el “switch” con el que se pone en marcha esta función.

La gran mayoría de los sistemas definidos en este archivo hacen referencia a muchas de las propiedades (“properties”) declaradas en la sección de Control de Vuelo del archivo principal del avión.

Es importante notar que en la versión de JSBSim utilizada en este proyecto el piloto automático es un prototipo en pruebas, y por ello no está suficientemente afinado.

2.2.5. Archivo de configuración del motor y de la hélice de la c172x

Estos dos archivos están, lógicamente, muy relacionados entre si, y en este caso el motor de la c172x es de combustión interna que arrastra una hélice.

En el archivo del motor se requieren los siguientes datos:

- a. minmp: La mínima presión del aire a la entrada del motor.
- b. maxmp: La máxima presión del aire a la entrada del motor.
- c. displacement: El volumen total de los cilindros.
- d. maxhp: La potencia máxima del motor.
- e. cycles: Los ciclos del motor.
- f. idlerpm: La cantidad mínima de revoluciones por minuto.
- g. maxthrottle: La aceleración máxima.
- h. minthrottle: La aceleración mínima.

En el archivo de la hélice se describe cómo es, cuántas palas tiene, su momento de inercia, si tiene paso variable, su diámetro, etc.

2.3. Ejemplo de creación de un avión en JSBSim.

Una vez entendido cómo se describe un avión en JSBSim, se procede a crear los archivos de configuración de una nueva aeronave, en este caso un UAV.

Los pasos a seguir son:

1. Crear la carpeta **1-uav**.
2. Determinar los parámetros físicos y aerodinámicos del UAV.
3. Crear archivos del UAV en JSBSim.

2.3.1. Crear la carpeta 1-uav.

El primer paso a la hora de crear un avión nuevo es entrar en el apartado donde JSBSim guarda las descripciones de los diferentes archivos donde se declaran los diferentes aviones.

Una vez dentro de esta carpeta, se tiene que crear la carpeta con el nombre del avión a modelar. En este ejemplo, se creó la carpeta **1-uav**. Este nombre fue escogido por comodidad de las proyectistas. Al ser el único avión con un nombre empezando por un número, esta carpeta siempre sería fácil de localizar.

Una vez hecha la carpeta, a su vez, hay que crear en ella una serie de archivos esenciales para la simulación:

1. El archivo principal del avión,
2. el archivo del piloto automático, y,
3. un archivo de inicialización¹⁰.

Los archivos de la hélice y del motor deben introducirse por separado, pero también dentro de JSBSim, dentro de su carpeta carpeta "engine".

Para empezar a crear el prototipo, se tomaron como apoyo los archivos de la c172x, por ser ya conocidos. Para ello se copiaron todos los archivos esenciales de la Cessna dentro de la carpeta del UAV.

¹⁰El archivo de inicialización sólo es necesario en caso de que se pida en la simulación.

De esta forma se tiene la rampa de lanzamiento para crear el nuevo avión. A medida que se avance en la creación del UAV, los datos reales del mismo reemplazarán paulatinamente a los copiados de la c172x.

2.3.2. Parámetros del UAV.

Antes de poder continuar con la creación del nuevo avión, es necesario tomar ciertas medidas en el modelo físico para poder rellenar la primera parte del archivo principal del UAV: *metrics*.

Como ya se ha explicado en el punto 2.2.3., este archivo requiere una serie de datos referentes a su estructura.

Para obtener estos datos se hizo referencia al TFC de Marc Espuga Zaragoza, *Identificación del modo longitudinal de un UAV*, puesto que en este TFC se explican sus características. Sin embargo, sólo se detallan las relacionadas con el modo longitudinal, por lo tanto era necesario encontrar el resto de medidas que pide el archivo principal y que no se encontraban en el TFC.

Para ello las proyectistas fueron a tomar medidas directamente en el propio UAV para completar el apartado “metrics” del archivo principal. La figura 2.1 muestra un esquema de las medidas asociadas a esta aeronave

2.3.3. Crear archivos del UAV en JSBSim.

Después de ver cómo se crea un avión en JSBSim, y de tomar los datos necesarios para el UAV, ya se estaba en condiciones para escribir sus archivos de configuración.

Una de las ventajas de la c172x es que es muy similar al UAV, tanto en su configuración como en su estructura general. Por lo tanto es factible utilizar los archivos de la c172x como modelo para el UAV. Para ello había que *transformar* los siguientes archivos:

1. El archivo principal de la c172x,
2. el piloto automático,
3. el motor y
4. la hélice.

2.3.3.1. Archivo de configuración principal del UAV

Aprovechando la similitud de las aeronaves, se puede ver que las partes que necesitan ser modificadas son solamente *metrics*, *mass_balance*, *ground_reaction*, y por último *propulsion*. Todos los cambios realizados se pueden ver en el Apéndice B.

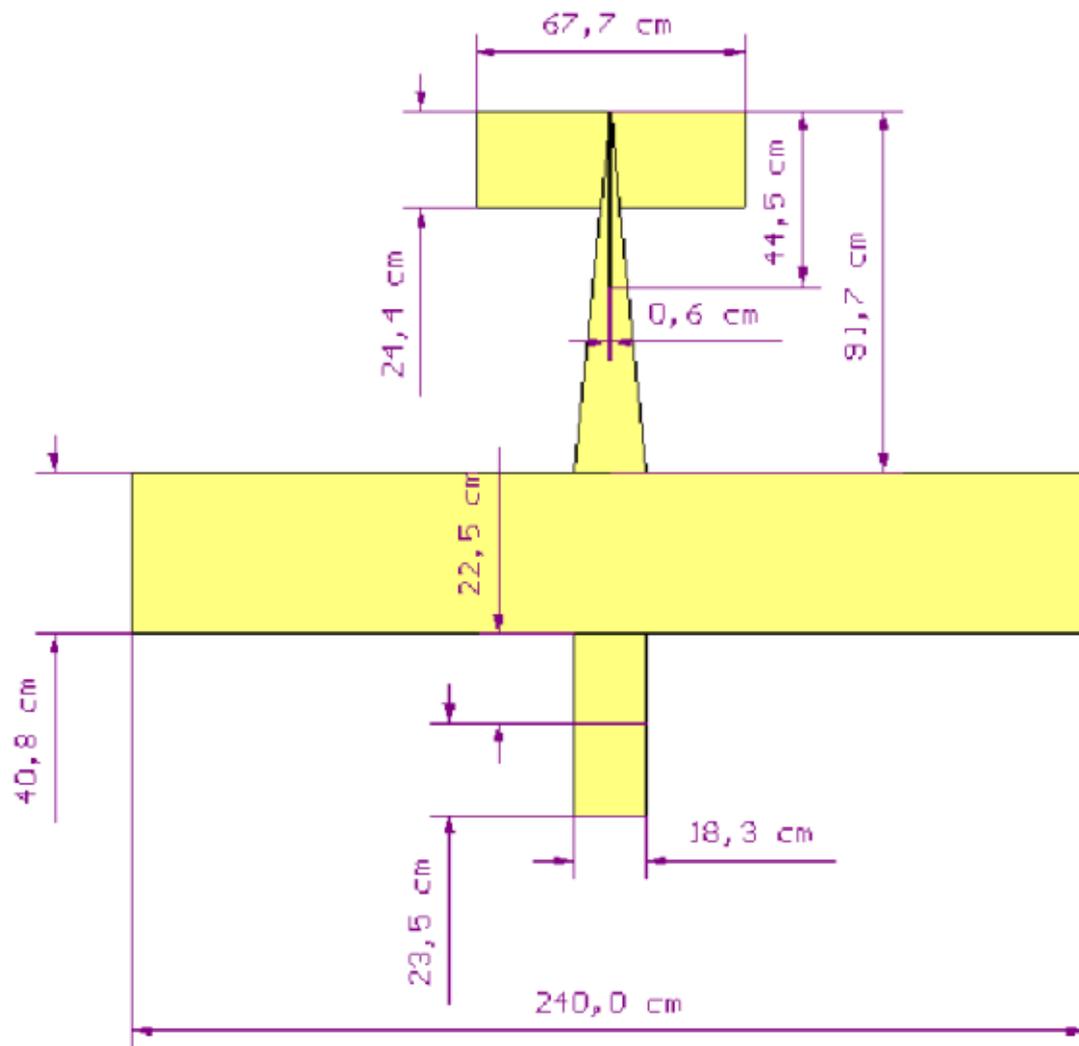


Figura 2.1: Dimensiones del UAV

En el apartado *aerodinamics* no ha sido necesario cambiar ninguno de los datos. Sólo fue necesario eliminar todas las funciones en donde había alguna tabla o referencia a los flaps, puesto que el UAV no tiene flaps.

El resto de las tablas podía conservarse puesto que las respuestas aerodinámicas de los dos aviones se consideran similares.

La tabla donde se han realizado cambios está en la siguiente parte del código.

```
<function name="aero/coefficient/CDwbh">
  <description>Drag due to alpha</description>
  <product>
    <property>aero/function/ground-effect-factor-drag</property>
    <property>aero/qbar-area</property>
  <table>
    <independentVar lookup="row">aero/alpha-rad</independentVar>
    <independentVar lookup="column">fcs/flap-pos-deg</independentVar>
```

```

    <tableData>
        0.0
        -0.0872664 -0.5500
        -0.0698132 -0.4400
        -0.0523599 -0.3300
        -0.0349066 -0.2200
        -0.0174533 -0.1100
        0.0 0.0000
        0.0174533 0.1100
        0.0349066 0.2200
        0.0523599 0.3300
        0.0698132 0.4400
        0.0872664 0.5500
        0.10472 0.6600
        0.122173 0.7700
        0.139626 0.8800
        ...
    </tableData>
</table>
</product>
</function>

```

En esta tabla, se compara la configuración de los flaps con el ángulo “alpha”. Puesto que hay una sola posición de flaps (0), sólo habrá una columna en la tabla. En ella se define la respuesta del coeficiente de resistencia aerodinámica del perfil alar del UAV.

Para encontrar los datos correctos de esta tabla ha sido necesario tener en cuenta el tipo de perfil utilizado. Gracias a la web indicada abajo se han podido completar los datos que faltaban:

<http://www.cyberiad.net/library/airfoils/foildata/n0012cl.htm>

Por otro lado, hay que tener en cuenta que en el apartado “mass_balance” hizo falta calcular los momentos de inercia del UAV. Estos cálculos fueron bastante complicados, y se explican en el Apéndice A.

2.3.3.2. Archivo de configuración del piloto automático del UAV

En este caso, y dado que el piloto automático sólo depende de cómo reacciona el avión y no de sus dimensiones, se puede utilizar el mismo archivo de la c172x modificando únicamente el nombre.

2.3.3.3. Archivo de configuración del motor y la hélice del UAV.

En estos archivos hay que reemplazar los datos de la c172x con los correspondientes del UAV en los lugares al efecto.

En el archivo donde se define la hélice, hay que poner el momento de inercia de las palas, el diámetro de las mismas y el número de ellas:

```
<ixx> 0.1 </ixx>
<diameter unit="IN"> 7.20 </diameter>
<numblades> 2 </numblades>
```

Para conseguir los datos del motor del UAV se recurrió de nuevo al TFC de Marc Espuga Zaragoza, puesto que en este documento se describen las especificaciones y datos más importantes del motor: motor de hélice de diámetro (D_p) de 50,1 cm, con una potencia máxima de 4,2 HP a 7500 rpm.

Con estos datos se puede completar el archivo del motor del UAV. El código quedaría así:

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

<piston_engine name="1-uaveng">
  <minmp unit="INHG"> 6.5 </minmp>
  <maxmp unit="INHG"> 29.92 </maxmp>
  <displacement unit="IN3"> 1.0 </displacement>
  <maxhp> 4.2 </maxhp>
  <cycles> 2.0 </cycles>
  <idlerpm> 550.0 </idlerpm>
  <maxthrottle> 1.0 </maxthrottle>
  <minthrottle> 0.2 </minthrottle>
</piston_engine>
```

En la segunda línea del código se especifica qué tipo de motor es. Como en este caso es un motor de pistones se tiene que definir con "piston_engine". Si el avión fuera una turbina se definiría con "turbine_engine".

CAPÍTULO 3. SIMULACIONES CON JSBSIM

En este apartado se describirá qué es un “script”, cómo se interpretan las simulaciones¹ y por último se presentarán los datos conseguidos con las ya mencionadas simulaciones y las diferentes aeronaves utilizadas.

En todas las simulaciones el fin principal es conseguir observar cómo reaccionan las aeronaves estudiadas ante ciertos estímulos.

Este capítulo debe facilitar a futuros proyectistas la elaboración de simulaciones y el estudio de cómo reaccionará el UAV (u otra aeronave de interés) ante los distintos estímulos y maniobras. En las siguientes simulaciones se estudian principalmente los movimientos de las aeronaves respecto al alabeo y cabeceo, e incluso cómo responden a un “looping”.

Finalmente, se aplicarán las simulaciones creadas para ver cómo reacciona el UAV introducido en este mismo proyecto, descrito en el apartado 2.3..

3.1. Descripción de las aeronaves utilizadas

A lo largo del proyecto se han utilizado para las diferentes simulaciones varias aeronaves. Entre ellas se encuentran la Cessna 172, el OV10 “Bronco” y finalmente el UAV, avión comprado por la EPSC e introducido en JSBSim en este proyecto.

Con la Cessna 172 se dan los primeros pasos con las simulaciones, se estudia el fichero “c172_runway_untilcrash.xml” y se crea una nueva simulación que escribe con la trayectoria del avión la palabra **FIN** en el cielo.

También, se intenta simular con este avión un “looping”, pero al no conseguirlo se termina estudiando la entrada en pérdida en barrena. Finalmente, la simulación del “looping” se consiguió realizar con otro avión más adecuado para dicha maniobra (el OV10 “Bronco”).

3.1.1. Cessna 172

La Cessna 172 Skyhawk (C172) es un avión monomotor de ala alta con capacidad para 4 personas. Es el avión más fabricado de la historia y probablemente el avión de entrenamiento más popular del mundo. El número total de aparatos modelo 172 fabricados a lo largo de 30 años asciende a la sorprendente cantidad de 40.000 unidades.

La 172 es descendiente directo de la Cessna 170, con la única diferencia del tren “triciclo” en lugar del convencional original propio de la 170. El Skyhawk forma parte de una gran familia de aviones Cessna monomotores de ala alta y tren triciclo.

Una variante de la C172, el T-41, es empleado como avión de entrenamiento en la Fuerza Aérea y el Ejército de los Estados Unidos.

¹Se describen varias simulaciones: desde las utilizadas para familiarizarse con el código hasta llegar a las creadas por las proyectistas.

Propiedades	
Tripulación	1 o 2
Planta motriz	1 motor Lycoming IO-360-L2A, 120 kW
Peso	1043 kg peso máximo al despegue.
Hélices	bipala de velocidad constante, por motor.
Carga útil	4 personas
Actuaciones	
Velocidad	máxima: 224 km/h
Autonomía	1030 km
Techo de servicio	3995 m
Dimensiones	
Envergadura	10,68 m
Longitud	8,20 m
Altura	2,68 m

Figura 3.1: Tabla de propiedades de la C172

A causa de su diseño de ala alta, su estabilidad en vientos de baja velocidad y su relativamente baja velocidad de entrada en pérdida, la C-172 es un excelente aparato para operaciones de búsqueda y rescate.



Figura 3.2: Cessna 172

3.1.2. OV-10 “Bronco”

El North American Rockwell OV-10 “Bronco” es un avión de apoyo en el campo de batalla.

Una de sus versiones, el OV-10A, empezó sirviendo con el Cuerpo de Marines y la Fuerza Aérea estadounidenses en 1968. Era una versión de remolque de blancos suministrada a la Alemania Federal.



Figura 3.3: OV-10 “Bronco”

Por otro lado, el OV-10C era otra versión para la Real Fuerza Aérea de Tailandia, donde se usa todavía en operaciones fronterizas. Otras versiones fueron el OV-10E para Venezuela y el OV-10F para Indonesia. En E.E.U.U., se usan actualmente algunos OV-10 en operaciones de vigilancia anti-incendios.

El OV-10 Bronco era extremadamente eficaz en misiones de apoyo cercano en zonas de combate y se usó en las últimas etapas de la guerra de Vietnam.

Propiedades	
Tripulación	2
Planta motriz	2 motores turbohélice AiResearch T76-410/411 de 533kW.
Peso	6.536 kg cargado.
Armamento	2 ametralladoras de 7,62 mm 1 máximo de 1632 kg de bombas y/o cohetes
Actuaciones	
Velocidad	máxima 452 km/h
Autonomía	960 km
Techo de servicio	9150 m
Dimensiones	
Envergadura	12,19 m
Longitud	12,67 m
Altura	4,62 m

Figura 3.4: Tabla de propiedades del OV-10

3.2. Herramientas utilizadas

En esta sección describiremos brevemente algunas de las herramientas que se utilizaron para facilitar el llevar a cabo las simulaciones.

3.2.1. Código Bash

Una de las herramientas utilizadas a lo largo del proyecto es el llamado “shell” **Bash**. Esta herramienta es un shell² de Unix³ escrito para el proyecto GNU⁴.

Bash es el intérprete predeterminado en la mayoría de sistemas GNU/Linux, además de Mac OS X Tiger, y puede ejecutarse en la mayoría de los sistemas operativos tipo Unix. También se ha llevado a Microsoft Windows mediante el proyecto “Cygwin”.

En este proyecto hemos usado el lenguaje asociado a Bash para escribir los scripts. Por lo tanto las simulaciones siguen el patrón de este lenguaje, a la vez que ha ayudado a crear pequeños programas que permiten administrar varias funciones, utilizando otros programas que ya están instalados en el sistema.

Un ejemplo de programa creado con el lenguaje o código Bash es el script *limpieza*, que permite facilitar el proceso de filtración de datos del fichero obtenido después de la realización de una simulación. Este script se explica con más detalle en la sección 3.2.4.

3.2.2. Scripts

Según la *Wikipedia*, el término **Script** se refiere a:

“Un guión o conjunto de instrucciones que permiten la automatización de tareas creando pequeñas utilidades.”

En JSBSim, un script es un archivo de texto que permite crear las tareas que realizarán los aviones además de, como dice la definición, pequeñas utilidades que han ayudado a facilitar el trabajo repetitivo de este proyecto.

Más adelante, en la sección 3.2.4., se describe paso a paso el script *limpieza*. Éste ha sido creado para agilizar la limpieza de los ficheros de datos obtenidos después de la simulación, y poder posteriormente hacer gráficos con los parámetros obtenidos.

3.2.3. Familiarización con los scripts

Para poder familiarizarse con los scripts, primero era necesario entender perfectamente su funcionamiento. El tutor del proyecto proporcionó como ayuda un primer script con comentarios que permitían entender la estructura y funciones de estos ficheros. El nombre de este primer fichero es *c172_runway_untilcrash.xml*.

En este archivo se describe un despegue, ascenso, una parada de motor y un posterior impacto de la aeronave contra el suelo.

²Shell es un intérprete de órdenes de Unix.

³Unix es un sistema operativo portable, multitarea y multiusuario.

⁴GNU es un proyecto iniciado por Richard Stallman con el objetivo de crear un sistema operativo completamente libre.

Para poder simular dicho fichero en JSBSim, éste se tiene que almacenar en la carpeta "scripts"⁵. Toda simulación tiene que hacer referencia a uno u otro avión de los almacenados en la carpeta "aircraft". En este caso, era la c172x.

En cualquier simulación es importante fijar unos parámetros iniciales como altitud, posición del avión, velocidad inicial, rumbo etc., cuyos datos puedan variarse para facilitar todo tipo de estudio que quiera analizarse con dicha simulación.

Cualquier parámetro inicial se tiene que configurar y guardar en un fichero dentro de la carpeta dónde esté declarada la aeronave, en este caso dentro de la carpeta "c172x".

Para una mejor comprensión de lo dicho anteriormente, se proporciona un ejemplo de un fichero, que en este caso se trata de "reset00.xml":

```
<?xml version="1.0"?>
<initialize name="reset00">
<!--(U, V, W) velocidad del CG del avión en ref.avión/ref.tierra-->
  <ubody unit="FT/SEC"> 0.0 </ubody>
  <vbody unit="FT/SEC"> 0.0 </vbody>
  <wbody unit="FT/SEC"> 0.0 </wbody>
  <latitude unit="DEG"> 47.0 </latitude>
  <longitude unit="DEG"> 122.0 </longitude>
  <phi unit="DEG"> 0.0 </phi>
  <theta unit="DEG"> 0.0 </theta>
  <psi unit="DEG"> 225.0 </psi>
  <altitude unit="FT"> 4.8 </altitude>
</initialize>
```

Como se puede ver, este fichero inicializa el avión en una latitud, longitud y altura concretas.

Una vez dentro del script de la simulación, la orden que hace referencia a la aeronave con la que se desea simular, y que además llama al fichero de inicialización es:

```
<use aircraft="c172x" initialize="reset00"/>
```

Una vez se tiene la simulación preparada (en este caso la proporcionada por el tutor) y elegido el avión con el que se va a realizar, solo queda ponerla en funcionamiento.

A continuación se explica como se pone en marcha una simulación en el sistema operativo "Linux".

Primero ha de abrirse un "Terminal", para acceder a la carpeta dónde está guardado JSBSim. En este caso, la orden sería:

```
cd JSBSim-0.9.13/
```

Una vez dentro de la carpeta, hay que llamar al script que permite ejecutar la simulación.

⁵"scripts" se encuentra dentro de JSBSim.

Para ello la instrucción que hay que utilizar es:

```
JSBSim --script=scripts/c172_runway_untilcrash.xml
```

Una vez ejecutada dicha orden se inicia la simulación.

Mientras se va ejecutando la simulación, ésta crea un fichero muy extenso **.csv** dónde se almacenan todos los parámetros obtenidos. Gracias al mismo, se podrá posteriormente aislar los datos que se necesiten, como velocidad, posición, rumbo, etc.

Este fichero **.csv** estará compuesto por una cabecera y los datos resultantes de la simulación. Hay dos formas para poder leer este tipo de ficheros: vía Terminal, o vía hoja de cálculo.

Si accedemos por el Terminal, la orden a escribir sería:

```
less JSBout172.csv
```

En la cabecera se indican los parámetros a tener en cuenta en la simulación. Los datos resultantes se almacenan secuencialmente cada uno en la fila de su correspondiente parámetro.

Si accedemos vía hoja de cálculo, se mostrarán los mismos datos ordenados por columnas.

Este **.csv** contiene gran cantidad de información que conduce a lecturas erróneas del mismo, por lo tanto es necesario eliminar todos los datos sobrantes. Para agilizar el proceso de limpieza del fichero y entender la metodología de los scripts y el código Bash, se crea el fichero “limpieza”, que se describe en el siguiente apartado.

3.2.4. Script de “limpieza”

Para poder agilizar procesos y familiarizarse con los scripts se crea “limpieza.xml”. Éste es un script que filtra los datos del fichero **.csv** (comentado anteriormente), separando la cabecera de los datos, y almacenándolo todo en dos ficheros nuevos.

En el primero de ellos se almacena la cabecera, asignándole un número natural consecutivo a cada parámetro, y presentándolo en forma de una sola columna.

En el segundo de ellos se guardarán los datos correspondientes a cada parámetro. Como el fichero de datos viene estructurado por defecto de la siguiente forma:

```
dato, dato, dato, etc
```

se debe almacenar del mismo modo, pero en lugar del doble espacio y la coma entre parámetros, hay que guardarlos con un espacio simple entre datos.

```
dato dato dato etc
```

Para poder ejecutar el script “limpieza”, es necesario indicar lo siguiente: fichero que se ha de limpiar, fichero de cabecera y fichero de datos, por lo que se debe escribir en el Terminal la siguiente instrucción:

```
./limpieza JSBout172B.csv cabecera.txt JSBout172M.csv
```

Esta orden llama al script que contiene el siguiente código:

```
#!/bin/bash

# $1 = fichero a modificar
# $2 = encabezado
# $3 = fichero limpio (sin encabezado)

# contabiliza todas las líneas del fichero
TODASLINEAS=`wc -l $1 | gawk '{print $1}'`

# contabiliza todas las líneas del fichero menos una
SNCABEZA=${TODASLINEAS-1}

# aquí se cambia el ", " por un salto de línea(\n) y lo guarda en $2
head -1 $1 | sed "s/, /\n/g" | gawk '{print NR,$0 }' > $2

# aquí se cambia el ", " por " " y lo guarda en $3
tail -$SNCABEZA $1 | sed "s/, / /g" > $3

# esto aparece en pantalla
echo
echo "FIN"
echo
```

3.2.5. Generación de gráficos

Para poder ver de forma gráfica la trayectoria de una simulación realizada, o cualquier otro parámetro, se usa un programa denominado “Gnuplot”.

Gnuplot es un programa muy flexible para generar gráficas de funciones y datos, y es compatible con los sistemas operativos más populares (Linux, UNIX, Windows, Mac OS X, etc.). El origen de Gnuplot data de 1986.

Gnuplot puede producir sus resultados directamente en pantalla, así como en multitud de formatos de imagen, como PNG, EPS, SVG, JPEG, etc. Se puede usar interactivamente o en modo por lotes, usando scripts. Este programa tiene una gran base de usuarios y está convenientemente mantenido por sus desarrolladores.

Para hacer los gráficos de la trayectoria simulada, lo primero que se debe tener en cuenta es cuáles parámetros se van a representar. Gracias al script “limpieza”, se crea un archivo “cabecera” donde se guardan estos datos. Para acceder a este archivo hay que escribir

en el Terminal la siguiente instrucción:

```
less cabecera.txt
```

Less permite mostrar en pantalla el contenido del fichero indicado.

Después de saber cuáles datos se van a representar, es necesario primero abrir el programa Gnuplot desde el Terminal:

```
gnuplot
```

Indicando seguidamente lo que se quiere representar. Por ejemplo:

```
plot "JSBout172M.csv" u 1:37 with lines
```

Primero se indica el archivo para la toma de datos y las posiciones (columnas) a representar. Con la orden "with lines", Gnuplot dibujará la gráfica con una línea continua, en vez de con puntos, que es el comportamiento por defecto.

En la figura 3.5 se representa la altura en función del tiempo.

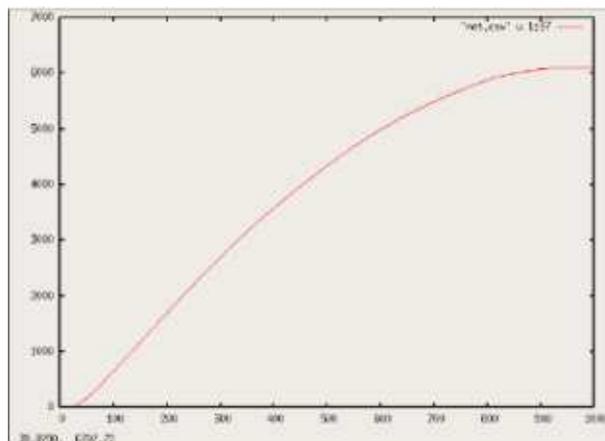


Figura 3.5: Ejemplo Gráfica 2D: Tiempo vs. Altura

En caso de querer una gráfica en 3D, los pasos a realizar son los mismos, pero poniendo "splot" en vez de "plot", y añadiendo los tres datos que se desean representar.

En el caso de que se quisiera representar la latitud en función de la longitud y distancia desde el suelo se haría mediante la instrucción siguiente:

```
splot "JSBout172M.csv" u 35:36:37 with lines
```

En la figura 3.6 se ve la imagen generada.

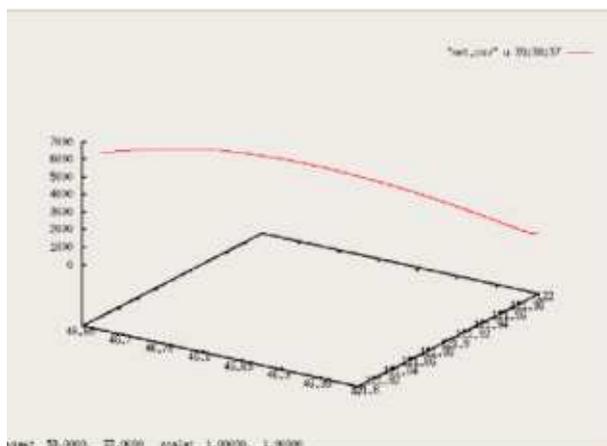


Figura 3.6: Ejemplo Gráfica 3D Latitud-Longitud-Altura

3.3. Simular “FIN” con una Cessna 172

En base al fichero facilitado por el tutor, “c172_runway_untillcrash.xml”, se desarrolla el script “FIN”. En esta simulación se juega con los movimientos de alabeo y guiñada, haciendo que el avión dibuje la palabra “FIN” vista en planta (representando la latitud-longitud).

Mediante esta simulación se describirá la estructura de los ficheros script y los comandos usados.

Más adelante, se repetirá esta simulación con el avión UAV creado en capítulos anteriores, aunque en este apartado haya sido ejecutada con una Cessna 172x por la fiabilidad en los datos de este avión.

3.3.1. Script FIN

Los ficheros utilizados para la realización de las simulaciones usan el lenguaje XML descrito en el apartado 2.2.1.

Estos archivos contienen una cabecera dónde se declara la versión XML con la que se va a trabajar, y la codificación utilizada. Estos dos campos, versión y codificación, son necesarios para que JSBSim interprete el código que acompaña a la cabecera.

En la siguiente línea de código se redirecciona al usuario usuario a la página que ofrece las pautas generales de creación de XML.

A continuación se incluye un ejemplo:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="http://jsbsim.sourceforge.net/JSBSimScript.xsl"?>
<runscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sf.net/JSBSimScript.xsd"
```

```
name="luav">
```

Después de la cabecera hay que declarar el avión con el que se va a trabajar en esta simulación y, en caso de necesitarse, los parámetros iniciales. En este caso se utilizará la “c172x” y los datos descritos en “reset00”⁶:

```
<use aircraft="1-uav" initialize="reset00"/>
```

A continuación, es obligatorio fijar el tiempo de duración de la simulación, así como la frecuencia de toma de datos. Dicha instrucción se estructura de la siguiente forma:

```
<run start="0.0" end="10000" dt="0.0083333">
```

En este ejemplo la simulación irá desde 0 a 10000 segundos, tomando datos cada 0.0083333 segundos.

Si se desea que mientras se ejecuta la simulación se vean en pantalla algunos de los resultados de la misma, se debe inicializar la propiedad “*time notify*”, que se declara mediante el siguiente elemento:

```
<property> simulation/notify-time-trigger </property>
```

Después de definir las condiciones de la simulación, se empiezan a escribir los diferentes eventos que se pretende simular mediante bloques de instrucciones. En todos los eventos ha de incluirse una condición (asociada al tiempo, a la presión, a la altura, etc.) que permite indicar cuándo se ejecutará.

Para crear un evento lo primero que se debe hacer es declararle un nombre, seguidamente, si se desea, se escribe una breve descripción de su funcionalidad, y posteriormente se indica la condición que tiene que cumplir. El primer evento de esta simulación se denomina “*engine start*”, el cual inicia el proceso de puesta en marcha el motor cuando han pasado 0.25 segundos de la simulación, como se muestra en el ejemplo que sigue:

```
<event name="engine start">
  <description>Start the engine</description>
  <notify/>
  <condition>
    sim-time-sec >= 0.25
  </condition>
```

A continuación se declaran las acciones a realizar para finalizar la puesta en marcha.

Será necesario describir el movimiento de la palanca del acelerador (throttle) del “Flight Control System” (fcs) desde 0.0 hasta su valor máximo de 1.0. La forma de moverlo es con una función “rampa” que dura 0.5 segundos.

```
<set name="fcs/throttle-cmd-norm" value="1.0" action="FG_RAMP" tc="0.5"/>
```

⁶Recordar que aquí se describe el estado inicial de la aeronave

También se debe describir el movimiento de la palanca de mezcla aire/combustible, desde 0.0 hasta el valor óptimo para el ascenso (0.87), y se usa para ello una función “rampa” que dura 0.5 segundos.

```
<set name="fcs/mixture-cmd-norm" value="0.87" action="FG_RAMP" tc="0.5"/>
```

Seguidamente se activan las magnetos:

```
<set name="propulsion/magneto_cmd" value="3"/>
```

Y se activa el motor de arranque (value="1"):

```
<set name="propulsion/starter_cmd" value="1"/>
```

Finalmente se cierra este evento con la instrucción:

```
</event>
```

Otro evento que encontramos en esta simulación es el que describe el cambio de rumbo. Como ejemplo se describirá el evento que cambia el rumbo del avión a 45 grados y activa el piloto automático.

En este caso en lugar de definir un tiempo como condición para la realización del evento, se define una altura sobre el terreno (AGL), que para este ejemplo será de 5 pies o mayor.

```
<event name="Set heading hold 1">  
  <description>Set Heading to 45 when 5 ft AGL is reached</description>  
  <notify/>  
  <condition>  
    position/h-agl-ft >= 5  
  </condition>
```

Cuando la altura AGL sea 5 ft o mayor, se programa al piloto automático para que mantenga el rumbo a 45 grados.

```
<set name="ap/heading_setpoint" value="45"/>  
<set name="ap/heading_hold" value="1"/>  
</event>
```

A continuación se describe un nuevo evento que se activa cuando el tiempo de simulación es 2500 segundos o mayor. En él se reprograma al piloto automático para que la aeronave cambie el rumbo a 180 grados y lo mantenga.

```
<event name="Set heading hold 2">  
  <description>Set Heading to 180</description>  
  <notify/>  
  <condition>  
    sim-time-sec >= 2500
```

```

    </condition>
    <set name="ap/heading_setpoint" value="180"/>
    <set name="ap/heading_hold" value="1"/>
</event>

```

De esta manera se realizan los giros pertinentes para escribir en el cielo la palabra FIN, adecuando las duraciones de cada tramo y los rumbos pertinentes para que realice la palabra con la mayor semejanza posible.

Otro tipo de condición que encontramos en esta simulación se refiere a la presión de impacto dinámica. En este evento, si ésta es mayor o igual a 4 psf, se programa el piloto automático para que la aeronave se eleve a 100 pies de altitud, y una vez en ella, debe mantenerla en ese nivel por estar activado, en el evento anterior, el 'altitude_hold'.

Como a lo largo de la simulación no se alcanzan los 4 psf, este evento no llegará a ejecutarse nunca.

```

<condition>
    aero/qbar-psf >= 4
</condition>

```

En otro de los eventos, al llegar a 100 pies sobre el terreno se empobrece la mezcla hasta llevarla al nivel óptimo de crucero (69 por ciento), mediante la utilización de una función exponencial con un tiempo característico de 400 segundos. A la vez se repliegan completamente los flaps (0 por ciento).

```

<event name="Adjust throttle/flaps">
    <description>Remove flaps at 100 ft</description>
    <notify/>
    <condition>
        position/h-agl-ft >= 100
    </condition>
    <set name="fcs/mixture-cmd-norm" value="0.69" action="FG_EXP" tc ="400.0"/>
    <set name="fcs/flap-cmd-norm" value="0"/>
</event>

```

Con la finalidad de que escriba la palabra FIN en un plano, al transcurrir 100 segundos de simulación se reprograma al piloto automático para que lleve la aeronave a 7000 pies de altitud y la mantenga allí, como se indica en el siguiente evento:

```

<event name="Set autopilot for 7000 ft.">
    <description>Set Autopilot for 7000 ft</description>
    <notify/>
    <condition>
        sim-time-sec >= 100
    </condition>
    <set name="ap/altitude_setpoint" value="7000.0"/>
</event>

```

Finalmente, el evento “Time Notify” se encarga persistentemente de generar mensajes en pantalla cada 100 segundos de simulación.

Los valores reportados por este último evento en este ejemplo son “Velocidad Calibrada” (vc) en nudos (kts), y la altura de la aeronave sobre el terreno, en pies (ft).

```
<event name="Time Notify" type="FG_DELTA" persistent="true">
  <description>Output message at 100 second intervals</description>
  <notify>
    <property>velocities/vc-kts</property>
    <property>position/h-agl-ft</property>
  </notify>
  <condition>
    sim-time-sec >= simulation/notify-time-trigger
  </condition>
  <set name="simulation/notify-time-trigger" value="100" type="FG_DELTA"/>
</event>
```

Para proceder a cerrar la simulación se utiliza la siguiente instrucción:

```
</run>
</runscript>
```

3.3.2. Gráficas “FIN”

A continuación se muestran las gráficas de la simulación “FIN”.

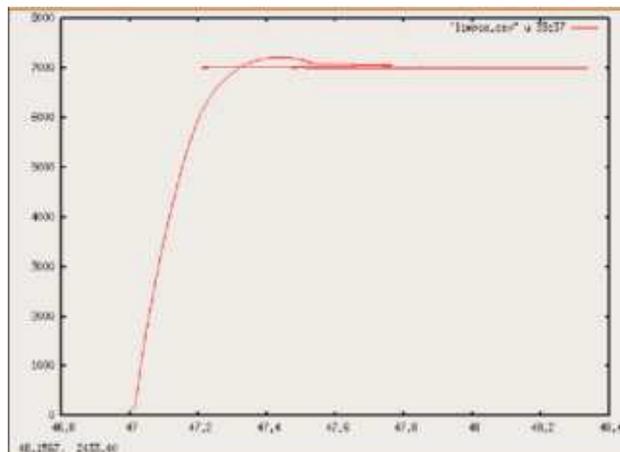


Figura 3.7: Simulación “FIN” en función de la latitud-altitud

3.4. Simular una caída en barrena con una Cessna 172

La siguiente meta a alcanzar fue la de realizar un “looping” con este avión. Después de muchas pruebas no se conseguía realizar sin que el avión entrara en pérdida.

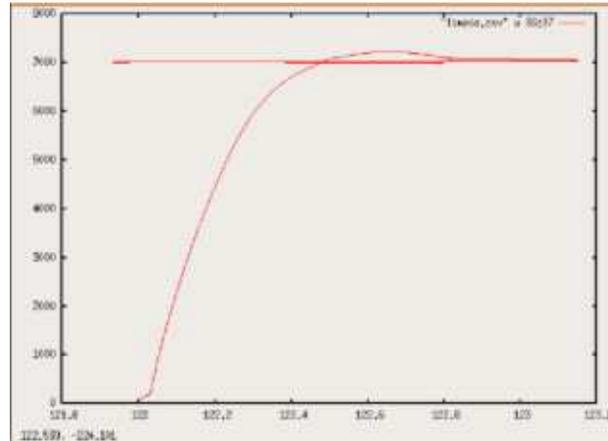


Figura 3.8: Simulación "FIN" en función de la longitud-altitud

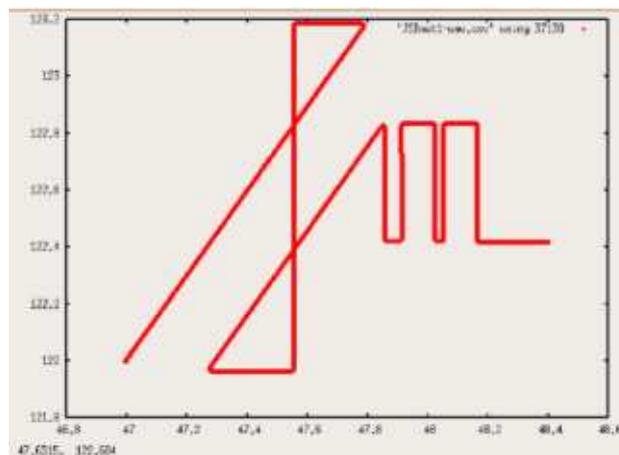


Figura 3.9: Simulación "FIN" en función de la latitud-longitud

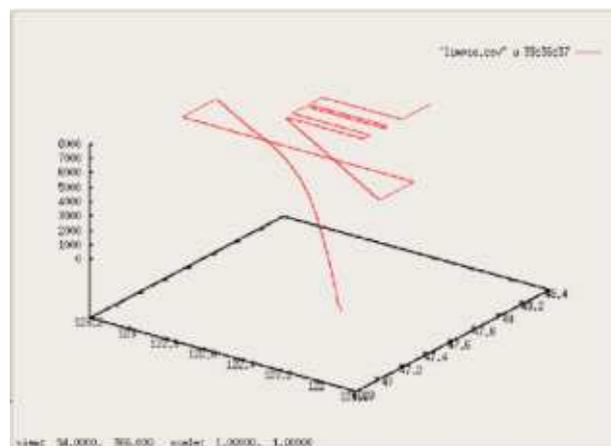


Figura 3.10: Simulación "FIN" en 3D, en función de la latitud-longitud-altura

Por otro lado, el avión sufría una desviación durante la realización del looping. En uno de los intentos la Cessna entró en pérdida consiguiéndose simular una caída en barrena. Por ello se llegó a la conclusión de que mediante este simulador no era posible la realización

de un looping con la Cessna⁷.

De forma somera, una barrena (en inglés “spin”), es una situación de vuelo en la que, tras entrar el avión en pérdida (por un excesivo ángulo de ataque), si a la vez se provoca un momento de guiñada mediante el timón de dirección, obtenemos que las fuerzas de sustentación generadas sobre las alas sean diferentes.

Mientras un ala está “en pérdida” y genera mucha resistencia y poca sustentación, la otra está “volando”, generando sustentación, lo cual crea un par de fuerzas que provoca un giro a lo largo del eje longitudinal.

Si la entrada en barrena se produce (voluntaria o inadvertidamente) por debajo de un umbral de seguridad, puede resultar imposible recuperar a tiempo la actitud de vuelo normal del avión.

Dicha recuperación depende del tipo de barrena, y especialmente del avión (algunos aviones no pueden recuperarse de una barrena). Para recuperar la posición del avión y salir de la barrena, suele ser suficiente mantener alerones y timón de profundidad en posición neutral, a la vez que se mueve el timón de dirección en el sentido opuesto al giro para detener la rotación, y a partir de aquí, recuperar la pérdida de forma normal.

3.4.1. Script “Barrena”

Al iniciar esta simulación, el primer evento es el de “engine start” para poner en marcha el motor cuando han pasado 0.25 segundos del inicio de la simulación (igual que en la simulación vista en la sección 3.3.).

A los 15 segundos del inicio de la simulación se mueve la palanca que controla el timón de profundidad hasta el punto óptimo, calculado experimentalmente, para picar el avión y que le permita tener la energía cinética necesaria para poder posteriormente intentar realizar el looping. Se usa una función rampa que dura 0.6 segundos, también calculada experimentalmente.

```
<event name="Dive the airplane">
  <description>Dive the airplane 1</description>
  <notify/>
  <condition>
    sim-time-sec >= 15
  </condition>
  <set name="fcs/elevator-cmd-norm" value="0.04" action="FG_RAMP" tc ="0.6"/>
</event>
```

A continuación, se estabiliza la palanca del timón de profundidad, aunque el avión sigue con su inercia de picado.

```
<set name="fcs/elevator-cmd-norm" value="0.0" action="FG_RAMP" tc ="0.6"/>
```

⁷O al menos, se necesita mucha pericia para hacerlo.

A los 21 s del inicio de la simulación se mueve la palanca que controla el timón de profundidad hasta el punto óptimo (-0.6), calculado experimentalmente, lo cual eleva el morro del avión como se hacía para intentar hacer el looping.

Se usa una función rampa que dura 0.6 segundos, también calculada experimentalmente.

```
<event name="push the airplane">
  <description>push the airplane 1</description>
  <notify/>
  <condition>
    sim-time-sec >= 21
  </condition>
  <set name="fcs/elevator-cmd-norm" value="-0.6" action="FG_RAMP" tc ="0.6"/>
</event>
```

Cuando el tiempo de simulación sea 30 segundos o mayor, se reprograma al piloto automático para que lleve la aeronave a 6000 pies de altitud y la mantenga allí, e intentar que éste recupere el avión de la caída inmediata. Pero como se ve en las gráficas que se muestran a continuación el avión no consigue recuperar la pérdida.

3.4.2. Gráficas “Barrena”

Seguidamente se muestran las gráficas de la entrada en pérdida en barrena en función de la latitud, longitud y altura.

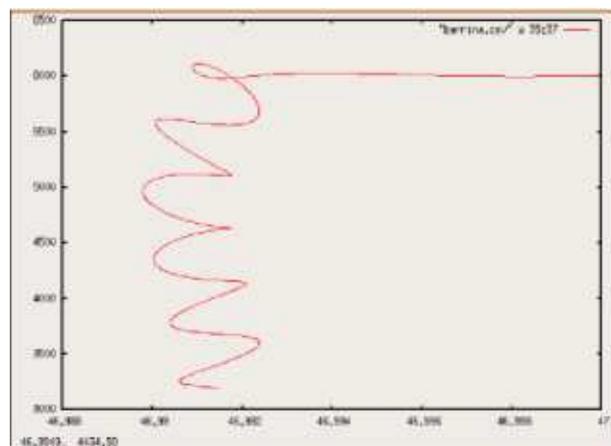


Figura 3.11: Simulación “Barrena” en función de la latitud-altitud

3.5. Simular un “looping” con un OV-10 “Bronco”

Como no se pudo realizar el looping con la Cessna, y teniendo en cuenta los conceptos de maniobrabilidad y estabilidad que se describen a continuación, se procede a realizar la acrobacia con el OV-10 “Bronco”, un avión más adecuado para la ejecución de dicho movimiento.

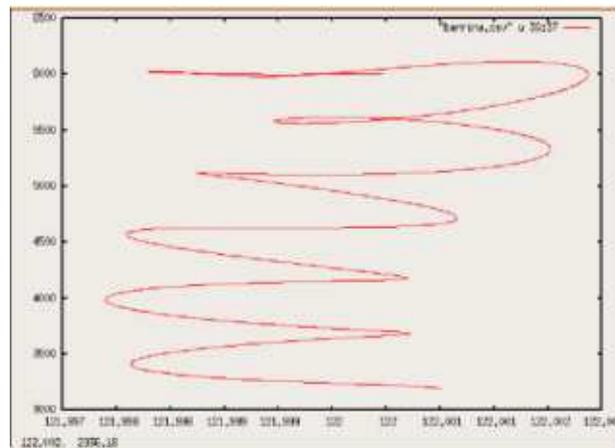


Figura 3.12: Simulación “Barrena” en función de la longitud-altitud

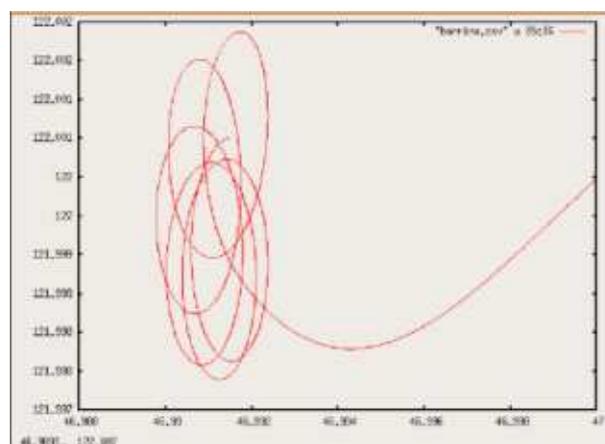


Figura 3.13: Simulación “Barrena” en función de la latitud-longitud

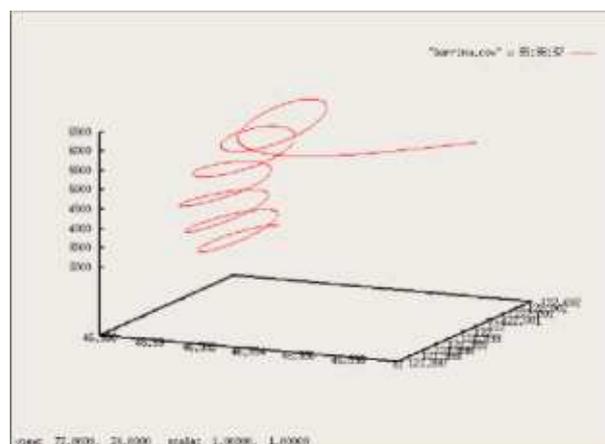


Figura 3.14: Simulación “Barrena” en 3D, en función de la latitud-longitud-altura

3.5.1. Estabilidad y Maniobrabilidad

El *equilibrio* se define como el estado de un cuerpo o sistema cuando la resultante de las fuerzas que actúan sobre él es nula. Se pueden definir dos tipos de estabilidad:

- Por **Estabilidad Estática** se entiende la respuesta de un sistema cuando se le saca de una posición de equilibrio. En nuestro caso, la estabilidad que nos interesa es la capacidad del avión para recobrar una posición de equilibrio después de sufrir una perturbación que la haya modificado (turbulencia, ráfaga de viento, etc.).
 - a. *Estabilidad positiva* significa que si un sistema es desplazado de su posición de equilibrio, genera fuerzas tendentes a volver a la posición inicial.
 - b. *Estabilidad neutra* se da cuando un sistema desplazado de su posición de equilibrio no genera ninguna fuerza y permanece equilibrado en esta nueva posición.
 - c. *Estabilidad negativa* es cuando un sistema desplazado de su posición de equilibrio genera fuerzas que tienden a desplazarlo aún más.
- * **Estabilidad Dinámica:** se dice que un sistema posee estabilidad dinámica si el movimiento del sistema produce una fuerza que se opone a ese movimiento. También la estabilidad dinámica puede ser positiva, neutra y negativa.

Por otro lado, la *maniobrabilidad* es la facilidad de un móvil para ser dirigido.

La conclusión que se obtiene de lo anterior es que cuanto más estable es un móvil, más difícil es sacarlo de la trayectoria inicial que seguía. Por lo que la Cessna, al ser más estable que el OV-10, es menos adecuada para recuperarla del looping.

3.5.2. Script “Looping”

En esta simulación se juega con la potencia de los motores, y después se recrea un looping usando el avión OV-10.

En este caso los parámetros iniciales se encuentran definidos en el archivo “reset02”, guardado dentro de la carpeta del avión “OV10”. En él vemos que se inicializa a una altitud de 10000 pies y en su velocidad máxima de crucero:

```
<?xml version="1.0"?>
<initialize name="reset02">
  <ubody unit="KTS"> 244.0 </ubody>
  <latitude unit="DEG"> 45.0 </latitude>
  <longitude unit="DEG"> 90.0 </longitude>
  <psi unit="DEG"> 200.0 </psi>
  <altitude unit="FT"> 10000.0 </altitude>
</initialize>
```

La simulación, que se inicia como todas las demás, dura 250 segundos. En este caso el avión tiene otra configuración de motor, por lo que su arranque es distinto, así que en primer lugar se da al interruptor “starter”:

```
<set name="propulsion/starter_cmd" value="1"/>
```

```
<set name="fcs/right-brake-cmd-norm" value="0"/>
<set name="fcs/left-brake-cmd-norm" value="0"/>
```

Una vez activado el motor de puesta en marcha (es decir, el “starter”), se permite el paso de combustible al motor. Una vez arrancados los motores se estabiliza su potencia a ralentí, iniciando así la simulación.

```
<set name="propulsion/cutoff_cmd" value="0"/>
<set name="propulsion/starter_cmd" value="1"/>
<set name="fcs/throttle-cmd-norm[0]" value="0.4"/>
<set name="fcs/throttle-cmd-norm[1]" value="0.4"/>
```

Dado que la simulación se inicia a 10000 pies, con la condición anterior el avión pierde altitud y al cabo de 15 segundos se ponen los motores a máxima potencia.

```
<condition> sim-time-sec >= 15 </condition>
<set name="fcs/throttle-cmd-norm[0]" value="1.0"/>
<set name="fcs/throttle-cmd-norm[1]" value="1.0"/>
```

Posteriormente, a 100 segundos del inicio de la simulación, una vez el avión estabilizado en altura se mueve la palanca que controla el timón de profundidad hasta el punto óptimo, calculado experimentalmente, para picar el avión y que le permita tomar la energía cinética necesaria para poder realizar el looping.

Se usa una función rampa que dura 0.6 segundos, también calculada por ensayo y error.

```
<condition>
  sim-time-sec >= 100
</condition>
<set name="fcs/elevator-cmd-norm" value="0.04" action="FG_RAMP" tc ="0.6"/>
```

Cinco segundos después movemos la palanca que controla el timón de profundidad hasta el punto óptimo (-0.5, calculado experimentalmente), para elevar el morro del avión y poder iniciar el looping. Para este movimiento se usa una función rampa que dura 0.6 segundos.

```
<condition>
  sim-time-sec >= 115
</condition>
<set name="fcs/elevator-cmd-norm" value="-0.5" action="FG_RAMP" tc ="0.6"/>
```

Finalmente, para estabilizar la aeronave y que no entre en pérdida, se debe colocar la palanca del timón de profundidad hasta el punto -0.05 usando una función rampa de 0.6 segundos.

```
<condition>
  sim-time-sec >= 140
</condition>
<set name="fcs/elevator-cmd-norm" value="-0.05" action="FG_RAMP" tc ="0.6"/>
```

Para terminar, se prosigue con la simulación el tiempo suficiente para asegurarse de que el avión está estabilizado.

3.5.3. Gráficas “Looping”

Seguidamente se muestran las gráficas de la simulación del looping en función de la latitud, longitud y altura.

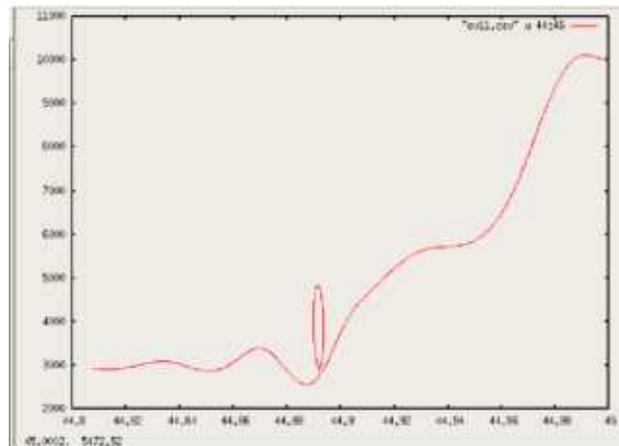


Figura 3.15: Simulación “Looping” en función de la latitud-altitud

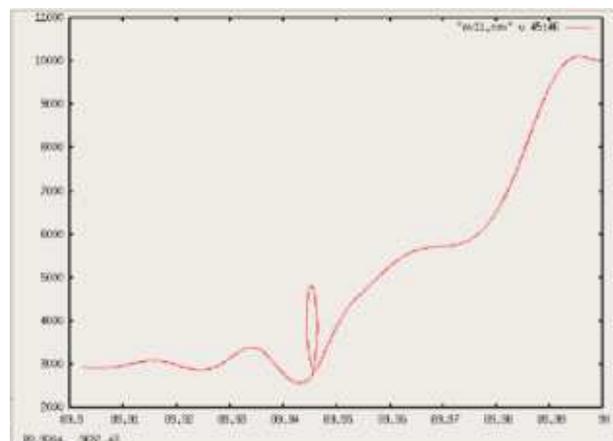


Figura 3.16: Simulación “Looping” en función de la longitud-altitud

3.6. Google Earth

A medida que se avanzaba en el proyecto, en la web oficial del programa se publicó una nueva versión de JSBSim. La novedad de esta nueva versión es que permite que las trayectorias simuladas sean proyectadas en **Google Earth**.

A partir de ahora, JSBSim permite poder visualizar todas las simulaciones que se han creado a lo largo del TFC como si el avión realizara dichas maniobras en la vida real y se

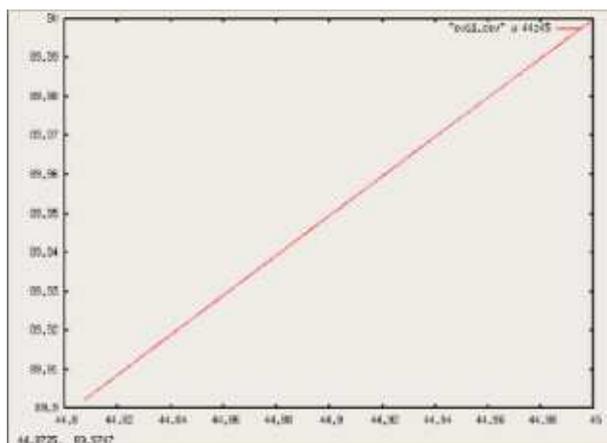


Figura 3.17: Simulación “Looping” en función de la latitud-longitud

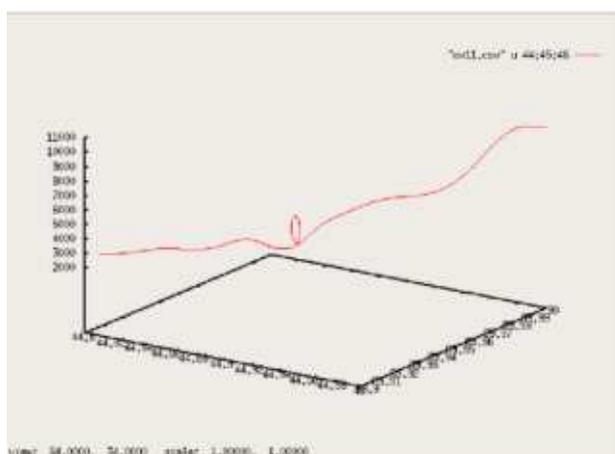


Figura 3.18: Simulación “Looping” en 3D, en función de la latitud-longitud-altura

hubiera tomado una fotografía del momento.

Para poder plasmar una simulación de esta nueva forma, se tienen que seguir una serie de pasos que ayudan a crear los ficheros *.kml*, puesto que es éste el formato que sabe interpretar Google Earth.

El primer paso permite cambiar la extensión del fichero original (*.csv*), modificándola en otra terminada en *.kml*. Para realizar este cambio se debe escribir en el Terminal de Linux la siguiente instrucción:

```
src/JSBSim --script=scripts/c1723.xml --logdirectivefile=kml_output
```

Es importante remarcar que esta instrucción altera la extensión del archivo y elimina todos los parámetros a excepción del tiempo, latitud, longitud y altura. Dichos parámetros fueron indicados dentro del archivo “kml_output”.

En los ficheros *.kml* utilizados en Google Earth los datos importantes son los de longitud, latitud y altura. Con estos datos, Google Earth puede leer sus archivos y por lo tanto generar las gráficas.

Por este motivo tenemos que eliminar la primera columna de datos del fichero, que es donde se registra el tiempo.

Para realizar dicha tarea se escribe la siguiente instrucción:

```
cut -d, -f2- GroundTrack.kml.csv > GroundTrack2.kml
```

Y para terminar, se tiene que poner una cabecera y una cola adecuadas en el documento para que esté completo. Para añadir estas partes del fichero se tiene que usar la siguiente instrucción:

```
cat Prefix.kml GroundTrack2.kml Postfix.kml > FinalGroundTrack.kml
```

Esta orden añade al fichero la siguiente cabecera:

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.2">
  <Document>
    <name>JSBSim Trajectory</name>
    <description>This trajectory was generated using JSBSim, and
was modified by Dago.</description>
    <Style id="yellowLineGreenPoly">
      <LineStyle>
        <color>7f00ffff</color>
        <width>4</width>
      </LineStyle>
      <PolyStyle>
        <color>7f00ff00</color>
      </PolyStyle>
    </Style>
    <Placemark>
      <name>JSBSim Trajectory</name>
      <description>This trajectory was generated using JSBSim.</description>
      <styleUrl>#yellowLineGreenPoly</styleUrl>
      <LineString>
        <extrude>0</extrude>
        <tessellate>0</tessellate>
        <altitudeMode>absolute</altitudeMode>
        <coordinates>
```

La cola del fichero será:

```
      </coordinates>
    </LineString>
  </Placemark>
</Document>
</kml>
```

De esta manera podemos obtener un fichero que se podrá abrir desde Google Earth como

se ve en la figura 3.19.

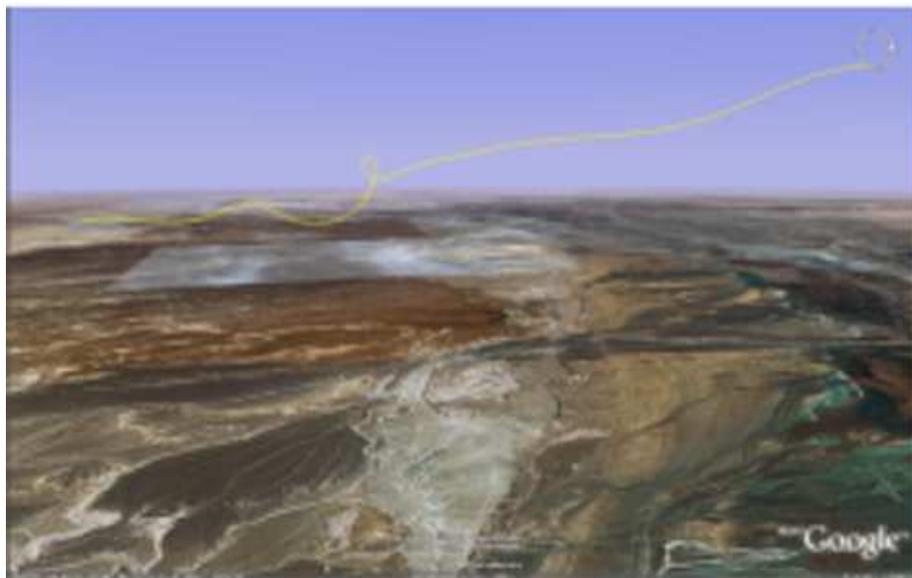


Figura 3.19: Simulación del "looping" en Google Earth

CAPÍTULO 4. SIMULANDO CON EL UAV

Llegado este punto, sólo queda comprobar que el UAV está bien creado. Para ello es necesario someter el avión a las simulaciones correspondientes.

4.1. Implementar las simulaciones con el UAV.

Antes de realizar las simulaciones con el UAV, ya se sabía de antemano que se necesitarían modificar muchos parámetros y que otros tantos serían incorrectos. Esta parte del proyecto se estimaba de larga duración.

Durante los últimos meses del proyecto y casi hasta el día anterior de entregar el escrito, se realizaron muchas pruebas con los scripts y los archivos de configuración del UAV. Gracias a estos ensayos, se pudieron corregir varios errores de escritura de los archivos de configuración.

Con las primera pruebas el programa falló dando errores de segmentación: Para intentar solucionarlo, se revisó todo el código y se descubrió que algunos comentarios introducidos (para ayudar a las proyectistas a entender los archivos del UAV) provocaban fallos de compilación. Se optó por quitar todos esos comentarios y probar cómo respondía la aeronave en el siguiente intento.

En las siguientes pruebas el fallo de segmentación persistía: A continuación se revisaron los scripts, pero no se encontró el fallo en ellos, ni en el código del UAV. Por lo tanto se decidió pedir ayuda nuevamente a los creadores de JSBSim. Después de enviarles un nuevo e-mail a cada uno de ellos adjuntando todos los archivos utilizados, respondieron aportando nuevas ideas de por qué estaban fallando los ficheros.

Lee Duke, uno de los creadores de JSBSim, respondió al e-mail comentando que una de las simulaciones estaba mal terminada, y que además no se habían eliminado todos los comentarios de los archivos de configuración del UAV.

Jon Berndt, el creador de la arquitectura original de JSBSim, también respondió al e-mail informando que dentro del archivo de configuración principal del UAV se había declarado un solo depósito de combustible, cuando se decía que el motor estaba siendo alimentado por dos depósitos.

Por otro lado, una de las simulaciones requería un archivo de inicialización que indirectamente provocaba que el avión se estrellase.

Después de solucionar todos estos problemas, el fallo de segmentación se solventó, pero apareció un nuevo fallo:

```
*** stack smashing detected ***: JSBSim terminated Cancelado (core dumped)
```

En este punto, las proyectistas volvieron a comunicarse con los creadores de JSBSim, explicándoles el nuevo fallo y pidiendo nuevamente colaboración.

Esta vez solo contestó *Jon Berndt*. En este caso, el señor *Berndt* comentó que el fallo podría estar en los datos utilizados para definir el UAV.

Como este avión había sido creado a partir de los archivos de una Cessna, era muy probable que los parámetros de base fueran excesivamente grandes para un avión de dimensiones reducidas como es el UAV. También nos comentó que, desde hace poco, han puesto en funcionamiento un nuevo apartado dentro de la web oficial de JSBSim para crear los archivos de configuración de los aviones de JSBSim: “Aeromatic”.

<http://jsbsim.sourceforge.net/aeromatic2.html>

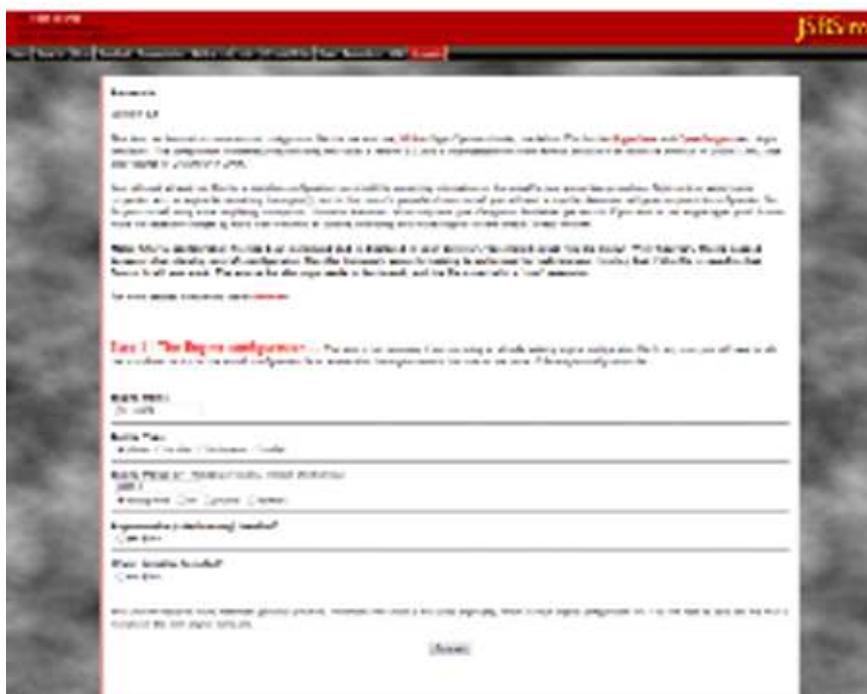


Figura 4.1: Parte de la web de JSBSim

En esta página se pueden introducir algunos datos como el tipo de motor, el diámetro de las palas, las revoluciones máximas, el modelo de avión utilizado, el peso del avión, etc., y se puede crear, sin ninguna dificultad, el archivo de configuración principal del UAV, el archivo del propulsión y el archivo del motor.

Una vez creados los archivos de configuración del avión, que se pueden ver en el Apéndice B, apartado B.2., era necesario probar si de esta forma el programa funcionaba correctamente.

El problema de los ficheros que genera “Aeromatic” es que muchos de los parámetros son estimados, ya que la página sólo permite introducir 4 datos principales, y a partir de éstos estima el resto.

Para tener unos archivos completos, hubo que comparar los nuevos ficheros con los ya existentes, procediendo a cambiar los datos erróneos o estimados por los calculados correctamente a partir del modelo original del avión.

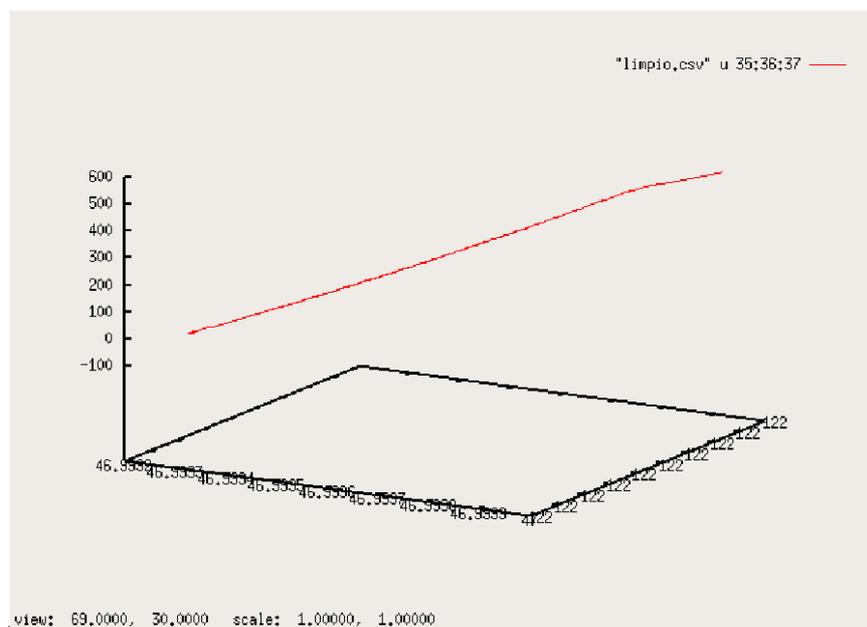
Estos datos se referían principalmente a los momentos y productos de inercia, situación del motor, algunas dimensiones del avión, etc.

En un primer lugar, al realizar el ensayo, el programa realizaba la simulación de forma incompleta, además de no guardar los datos obtenidos del mismo. Por eso se consultó nuevamente con *Jon Berndt* exponiéndole la situación actual.

El problema esta vez era que el "Aeromatic" presuponía que el avión tenía unas dimensiones más grandes de lo que realmente eran, así que el punto de referencia aerodinámico estaba sobredimensionado. A partir de esta pista se encontraron otros errores de programación.

Finalmente, al pulir los ficheros de configuración del UAV y retocar la simulación, se consiguió que el avión volara y guardara los datos en el fichero .csv.

Se escogió una simulación de las que ya venían predeterminadas dentro de la carpeta *scripts*, y se probó con los nuevos ficheros con dicha simulación. Después de comprobar que la simulación se realizaba con éxito, el resultado obtenido se puede observar en la figura 4.2, y el script utilizado se encuentra en la sección C.4. del Anexo C.



CAPÍTULO 5. CONCLUSIONES

A lo largo del proyecto se han ido viendo diferentes etapas del desarrollo del mismo.

En la primera de ellas era estrictamente necesario aprender qué es JSBSim. Además de ser el primer objetivo planteado, dicho conocimiento era imprescindible para poder comprender más adelante todo el funcionamiento del programa.

Gracias al trabajo de búsqueda bibliográfica realizado fue posible aprender con el mayor detalle que se pudo el mencionado programa, consiguiendo así dar paso a la segunda fase del proyecto.

En esta segunda parte se pretendía conocer cómo se configura una aeronave en JSBSim, y aprender a leer los archivos utilizados en él. Para poder llegar a cumplir este objetivo fue necesario estudiar varios manuales que ayudaron a comprender dichos archivos. También fue necesario ponerse en contacto con los creadores de JSBSim puesto que los manuales encontrados en la red estaban incompletos.

Después de comprender cómo se configuran los aviones en JSBSim, se creó un avión propio sobre el que se puso en práctica los conocimientos adquiridos.

El siguiente paso a realizar, después de cumplir los dos primeros objetivos, fue llevar a cabo una simulación. Para ello, se tuvo que estudiar como funcionan los códigos fuentes en los que JSBSim está diseñado. Para saber si todo había sido comprendido correctamente, se realizaron con éxito varias pruebas con diferentes aeronaves.

Finalmente, se trató de acoplar los archivos de la aeronave diseñada con los scripts creados. En este punto la aeronave consigue ejecutar el script y proporciona información del estado del UAV durante la simulación.

Entendemos que los objetivos planteados en este TFC han sido logrados, si bien, disponiendo de más tiempo y recursos se puede proseguir en la experimentación y el estudio del proyecto.

GLOSARIO

Shell: Intérprete de órdenes de Unix.

Unix: Sistema operativo portable, multitarea y multiusuario.

GNU: fue iniciado por Richard Stallman con el objetivo de crear un sistema operativo completamente libre.

EML: Siglas de “Extensible Markup Language” (“lenguaje de marcas ampliable”)

Lenguaje de marcas: Forma de codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación. Es un metalenguaje.

Metalenguaje: Lenguaje usado para hacer referencia a otros lenguajes.

SGML: Siglas de “Standard Generalized Markup Language” o “Lenguaje de Marcación Generalizado”. Consiste en un sistema para la organización y etiquetado de documentos.

DTD: Siglas en inglés de “Document Type Definition”. La definición de tipo de documento (DTD) es una descripción de estructura y sintaxis de un documento XML o SGML. Su función básica es la descripción del formato de datos, para usar un formato común y mantener la consistencia entre todos los documentos que utilicen la misma DTD.

BIBLIOGRAFÍA

Libros

- [1] K. Ogata *Sistemas de Control en Tiempo Discreto*. Segunda edición. Prentice may. 1996.
- [2] G.F Franklin, J.D. Powell, A. Emani-Naeini *Feedback Control of Dynamic Systems*. Cuarta edición. Prentice-Hall, 2002.

Páginas Web

- www.wikipedia.org
- <http://jsbsim.sourceforge.net/>
- <http://jsbsim.wiki.sourceforge.net/JSBSim+Users+Manual>
- <http://jsbsim.sourceforge.net/JSBSim/classes.html>
- <http://jsbsim.sourceforge.net/AutomaticFlightInJSBSim.pdf>

Otros documentos

- JSBSim: An Open Source Flight Dynamics Model in C++.
- Identificación del modo longitudinal de un UAV. Marc Espuga Zaragoza. TFC.
- JSBSim Reference Manual.
- Tutorial de LaTeX.
- Tutorial de Bash.
- Tutorial de Guplot.

APÉNDICE A. CÁLCULO DE LOS MOMENTOS Y PRODUCTOS DE INERCIA DEL UAV

A.1. Cálculo de los momentos de inercia.

A.1.1. Cálculo del momento de inercia en el eje X (I_x)

Siguiendo con la misma tónica y procedimiento que se ha aplicado para obtener el momento de inercia sobre el eje Y (I_y) en el TFC de Marc Espuga Zaragoza, *Identificación del modo longitudinal de un UAV*, se procede de la siguiente manera:

Se descompone la geometría del UAV en figuras geométricas simples obtenidas mediante la proyección del avión, en cada caso, sobre el plano correspondiente, y calculando por separado el momento de inercia de cada una de las figuras.

Concretamente la proyección sobre el plano XY, nos da figuras que son rectangulares y triangulares como se ve en la figura A.1.

Se aplicará, en base a los resultados de las dimensiones de diseño aceptadas para el UAV, las mismas o similares fórmulas que las usadas para el cálculo del momento I_y , con la adaptación adecuada de las coordenadas correspondientes al nuevo plano y eje de trabajo.

Como resultado se obtienen los momentos de los siguientes elementos geométricos que se denominarán así:

- Rectángulo correspondiente al motor.
- Rectángulo correspondiente al cajón del fuselaje, unión alas-motor.
- Rectángulo formado por las alas.
- Rectángulo formado por el soporte de las alas, unión fuselaje-alas.
- Triángulo tramo final fuselaje.
- Rectángulo estabilizador horizontal.
- Rectángulo estabilizador vertical.

Dada la simetría del avión con respecto al eje X, todos los centros de masas de las figuras en que se ha descompuesto el UAV, están sobre el citado eje X, por lo que los momentos calculados con respecto al centro de masas de cada figura son a su vez los momentos sobre el eje X, no habiendo, por ello, necesidad de aplicar el teorema de los ejes paralelos.

Los resultados de los cálculos se reflejan en la tabla A.1.

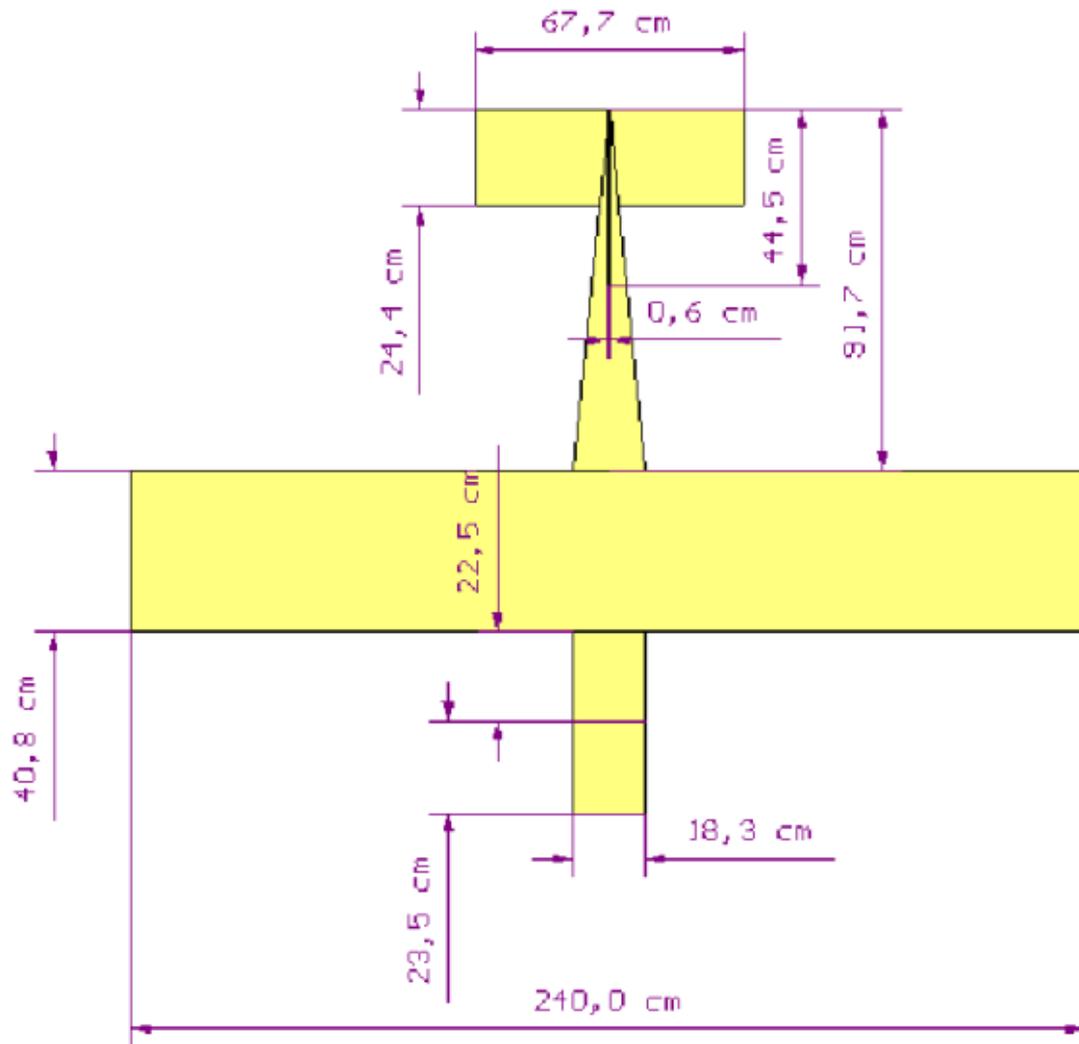


Figura A.1: Medidas del UAV

El momento sobre el eje X será la suma de todos los momentos:

$$I_x = 9,70500 \cdot 10^6 \text{ gr} \cdot \text{cm}^2 \quad (\text{A.1})$$

$$I_x = 0,9705 \text{ Kg} \cdot \text{m}^2 \quad (\text{A.2})$$

A.1.2. Cálculo del momento de inercia sobre el eje Z (I_z)

Para este eje las figuras son las mismas que las obtenidas para el eje X, aunque viendo la proyección del avión en el plano YZ. En este caso, para la obtención del momento de inercia sobre el eje Z se tiene que aplicar el Teorema de Steiner, por estar todos los centros de masas de las mismas fuera del eje.

A continuación se procede a explicar brevemente el Teorema de Steiner.

Nombre Figura Simple	Momento de Inercia ($gr \cdot cm^2$)
Rectángulo Motor	32404.37
Rectángulo Tramo Fuselaje	2298.18
Rectángulo Alas	9407232.0
Rectángulo Soporte Alas	4170.435
Triángulo Tramo Final Fuselaje	987.36
Rectángulo Estab Horizontal	258132.72
Rectángulo Estab Vertical	0.1458
Total	$9.70522 \cdot 10^6$

Tabla A.1: Momentos de Inercia en el Eje X

El Teorema de Steiner (o teorema de los ejes paralelos) establece que el momento de inercia con respecto a cualquier eje paralelo a un eje que pasa por el centro de gravedad, es igual al momento de inercia con respecto al eje que pasa por el centro de gravedad, más el producto de la masa por el cuadrado de la distancia entre los dos ejes, como se ve en la ecuación A.3.

$$I_{eje} = I_{eje}^{(CM)} + Mh^2 \quad (A.3)$$

Donde:

- I_{eje} es el momento de inercia respecto al eje que no pasa por el centro de masa,
- $I_{eje}^{(CM)}$ es el momento de inercia para un eje paralelo al anterior que pasa por el centro de gravedad,
- M es la masa total y,
- h es la distancia entre los dos ejes paralelos considerados.

Por lo tanto, para calcular el momento de inercia I_z será necesario obtener, primero, el momento de inercia de cada figura simple del UAV y la distancia al eje Z. En la tabla A.2 se pueden ver los resultados obtenidos.

Nombre Figura Simple	Momento de la figura ($gr \cdot cm^2$)	Distancia(cm) ($gr \cdot cm^2$)
Rectángulo Motor	85840.77	46.45
Rectángulo Tramo Fuselaje	5772.32	23.45
Rectángulo Alas	9674562.03	8.2
Rectángulo Soporte Alas	4170.4	8.2
Triángulo Tramo Final Fuselaje	69989.8	67.15
Rectángulo Estab Horizontal	278825.9	132.27
Rectángulo Estab Vertical	664.24	124

Tabla A.2: Cálculo de Steiner en el Eje Z

Nombre Figura Simple	Momento de Inercia ($gr \cdot cm^2$)
Rectángulo Motor	2591108.55
Rectángulo Tramo Fuselaje	51056.79
Rectángulo Alas	986274.4
Rectángulo Soporte Alas	17197.5
Triángulo Tramo Final Fuselaje	1027321.6
Rectángulo Estab Horizontal	7606229.6
Rectángulo Estab Vertical	75391.6
Total	$21.1738 \cdot 10^6$

Tabla A.3: Momento de Inercia en el Eje Z

Finalmente, los resultados calculados de los momentos de inercia I_z se reflejan en la tabla A.3.

El momento total será:

$$I_z = 21,1738 \cdot 10^6 gr \cdot cm^2 I_z = 2,11738 Kg \cdot m^2 \quad (A.4)$$

A.1.3. Cálculo de los productos de inercia

Por definición el producto de inercia de un sistema con respecto a dos planos es la suma de los productos de las masas de las partículas del sistema por su distancia a ambos planos (mirar la formula A.5).

$$P_{xy} = \sum_{i=1}^n m_i \cdot x_i \cdot y_i \quad (A.5)$$

Donde:

- m_i es la masa de las partículas, y
- x_i e y_i son la distancia de dicha masa a ambos planos.

Aprovechando la descomposición del UAV en figuras simples que ya han sido usadas en los apartados anteriores, y que son perfectamente conocidos sus centros de masas y las distancias de éstos a los planos formados por los ejes de coordenadas en el centro de masas del avión, se calcularán los productos de inercia considerando que cada figura simple concentra su masa en su CM, multiplicando la misma por sus coordenadas con respecto los planos de referencia del producto.

$$P_{xy} = \sum_{i=1}^n m_i \cdot x_i \cdot y_i \quad (\text{A.6})$$

$$P_{xy} = \sum_{i=1}^n m_i \cdot x_i \cdot z_i \quad (\text{A.7})$$

$$P_{xy} = \sum_{i=1}^n m_i \cdot y_i \cdot z_i \quad (\text{A.8})$$

Donde:

- m_i es la masa de las figuras simples obtenidas para el cálculo del momento de inercia correspondiente a cada eje.

Aplicando la fórmula A.6 para calcular el producto de inercia en el plano XY, se obtiene que $P_{xy} = 0$ debido a la simetría del avión en este plano, ya que todo producto m_{xy} a un lado del eje tiene el opuesto $-m_{xy}$ en el otro lado, siendo por tanto la suma total igual a cero.

Por la misma circunstancia que el caso anterior, si se calcula el producto de inercia para el plano YZ, se obtiene que $P_{yz} = 0$.

Por lo tanto, sólo se tiene que calcular el producto P_{xz} , ya que sobre el UAV no hay simetría en este eje.

Los elementos a considerar en este plano son:

- Rectángulo del motor
- Rectángulo primer tramo fuselaje
- Rectángulo fuselaje bajo las alas
- Rectángulo alas
- Triángulo parte final fuselaje
- Estabilizador horizontal
- Estabilizador vertical

Asimismo, por la simetría existente respecto al eje x no habrá necesidad de hacer los cálculos de producto de inercia de todos aquellos elementos cuyo centro de masas esté sobre el propio eje x, ya que por esta circunstancia el producto $I_{xz} = 0$ ya que $Z = 0$.

Por tanto calculamos solamente los productos de aquellas figuras simples cuyo centro de masas está fuera del eje x, que son: *Rectángulo alas* y *Estabilizador vertical*.

A ambas las dividimos en dos partes: A *Rectángulo alas* lo consideramos dividido en la parte anterior, que es la que está entre el Datum¹ y el CM del UAV, y la parte posterior, que es la que queda por detrás del CM del avión.

Por su parte, al *Estabilizador vertical* lo dividimos en su parte rectangular y su parte triangular.

Primero habrá que calcular los momentos de inercia en el centro de masas de cada figura simple, y calcular la distancia que hay desde ese CM hasta el eje X y el eje Z. Todos estos resultados se ven reflejados en la tabla A.4

Nombre Figura Simple	Momento en CM ($gr \cdot cm^2$)	Distancia X (cm)	Distancia Z (cm)
Rectángulo Estab Vertical	151	-134.4	15.2
Triángulo Estab Vertical	62.46	-115.6	10
Rectángulo Ala Anterior	18.73	6.4	2.5
Rectángulo Ala Posterior	43.72	-14	2.5

Tabla A.4: Momentos de Inercia y distancias a los ejes X y Z

Después de calcular el momento de inercia de cada figura y su distancia a cada eje, es necesario calcular el producto de inercia. Los resultados se ven reflejados en la tabla A.5

Nombre Figura Simple	Producto de Inercia ($gr \cdot cm^2$)
Rectángulo Estab Vertical	-308474.88
Triángulo Estab Vertical	-72203.76
Rectángulo Ala Anterior	299.68
Rectángulo Ala Posterior	-1530.2
Total	-381909.16

Tabla A.5: Producto de Inercia respecto al Eje XZ

De esta forma obtenemos que el producto de inercia en el eje XZ es:

$$P_{xz} = -381909,16gr \cdot cm^2 \quad P_{xz} = -0,03819Kg \cdot m^2 \quad (A.9)$$

Y el producto P_{xy} y P_{yz} es igual a 0.

A la vista de los productos de inercia obtenidos, y dado que éstos son una medida de desequilibrio dinámico del sistema del que se obtiene, se puede afirmar que el UAV es un avión dinámicamente muy equilibrado.

¹Punto de referencia para medir las distancias, que en este avión es el (0,0,0) y queda en la punta del "spinner" de la hélice

APÉNDICE B. ARCHIVOS DEL CÓDIGO DEL UAV

A continuación se adjuntan los códigos completos del UAV introducido en JSBSim.

B.1. Archivos de la UAV creados a partir de la Cessna

B.1.1. Archivo de configuración principal

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="http://jsbsim.sourceforge.net/JSBSim.xsl"?>
<fdm_config name="Megastar XL-120" version="2.0" release="BETA"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sourceforge.net/JSBSim.xsd">

  <fileheader>
    <author>Mar Noguera and Teresa Marin</author>
    <filecreationdate>2008-06-04</filecreationdate>
    <version>$Revision: 1 $</version>
    <description>To model a Megastar XL-120.</description>
  </fileheader>

  <metrics>
    <wingarea unit="FT2"> 10.54 </wingarea>
    <wingspan unit="FT"> 7.87 </wingspan>
    <chord unit="FT"> 1.35 </chord>
    <htailarea unit="FT2"> 0.69 </htailarea>
    <htailarm unit="FT"> 3.38 </htailarm>
    <vtailarea unit="FT2"> 0.599 </vtailarea>
    <vtailarm unit="FT"> 1.017 </vtailarm>
  <location name="AERORP" unit="IN">
    <x> 0.0 </x>
    <y> 0.0 </y>
    <z> 0.0 </z>
  </location>
  <location name="EYEPOINT" unit="IN">
    <x> 0.0 </x>
    <y> 0.0 </y>
    <z> 0.0 </z>
  </location>
  <location name="VRP" unit="IN">
    <x> 0.0 </x>
    <y> 0.0 </y>
    <z> 0.0 </z>
  </location>
</metrics>
```

```

<mass_balance>
  <ixx unit="SLUG*FT2"> 0.715 </ixx>
  <iyy unit="SLUG*FT2"> 1.146 </iyy>
  <izz unit="SLUG*FT2"> 1.560 </izz>
  <ixz unit="SLUG*FT2"> -0.028 </ixz>
  <emptywt unit="LBS"> 15.54 </emptywt>
  <location name="CG" unit="IN">
    <x> 22.91 </x>
    <y> 0.0 </y>
    <z> 0.0 </z>
  </location>
  <pointmass name="PILOT">
    <weight unit="LBS">0.0</weight>
    <location name="POINTMASS" unit="IN">
      <x> 0.0 </x>
      <y> 0.0 </y>
      <z> 0.0 </z>
    </location>
  </pointmass>
  <pointmass name="CO-PILOT">
    <weight unit="LBS">0.0</weight>
    <location name="POINTMASS" unit="IN">
      <x> 0.0 </x>
      <y> 0.0 </y>
      <z> 0.0 </z>
    </location>
  </pointmass>
  <pointmass name="LUGGAGE">
    <weight unit="LBS">1.0</weight>
    <location name="POINTMASS" unit="IN">
      <x> 25.98 </x>
      <y> 0.0 </y>
      <z> -1.97 </z>
    </location>
  </pointmass>
</mass_balance>

<ground_reactions>
  <contact type="BOGEY" name="Main Gear Left">
    <location unit="IN">
      <x> 18.11 </x>
      <y> -10.04 </y>
      <z> -4.96 </z>
    </location>
    <static_friction>0.8</static_friction>
    <dynamic_friction>0.5</dynamic_friction>
    <rolling_friction>0.02</rolling_friction>
    <spring_coeff unit="LBS/FT">1800</spring_coeff>
    <damping_coeff unit="LBS/FT/SEC">600</damping_coeff>
    <relaxation_velocity>
      <rolling unit="FT/SEC"> 0 </rolling>
    </relaxation_velocity>
  </contact>

```

```
        <side unit="FT/SEC"> 0 </side>
    </relaxation_velocity>
    <force_lag_filter>
        <rolling> 0 </rolling>
        <side> 0 </side>
    </force_lag_filter>
    <wheel_slip_filter> 0 </wheel_slip_filter>
<max_steer unit="DEG"> 0 </max_steer>
    <brake_group> NONE </brake_group>
    <retractable>0</retractable>
</contact>
<contact type="BOGEY" name="Main Gear Righth">
    <location unit="IN">
        <x> 18.11 </x>
        <y> 10.04 </y>
        <z> -4.96 </z>
    </location>
    <static_friction>0.8</static_friction>
    <dynamic_friction>0.5</dynamic_friction>
    <rolling_friction>0.02</rolling_friction>
    <spring_coeff unit="LBS/FT">1800</spring_coeff>
    <damping_coeff unit="LBS/FT/SEC">1600</damping_coeff>
    <relaxation_velocity>
        <rolling unit="FT/SEC"> 0 </rolling>
        <side unit="FT/SEC"> 0 </side>
    </relaxation_velocity>
    <force_lag_filter>
        <rolling> 0 </rolling>
        <side> 0 </side>
    </force_lag_filter>
    <wheel_slip_filter> 0 </wheel_slip_filter>
<max_steer unit="DEG"> 0 </max_steer>
    <brake_group> NONE </brake_group>
    <retractable>0</retractable>
</contact>
<contact type="BOGEY" name="Tail Gear">
    <location unit="IN">
        <x> 74.29 </x>
        <y> 0.0 </y>
        <z> -4.96 </z>
    </location>
    <static_friction>0.8</static_friction>
    <dynamic_friction>0.5</dynamic_friction>
    <rolling_friction>0.02</rolling_friction>
    <spring_coeff unit="LBS/FT">1800</spring_coeff>
    <damping_coeff unit="LBS/FT/SEC">1600</damping_coeff>
    <relaxation_velocity>
        <rolling unit="FT/SEC"> 0 </rolling>
        <side unit="FT/SEC"> 0 </side>
    </relaxation_velocity>
    <force_lag_filter>
```

```

        <rolling> 0 </rolling>
        <side> 0 </side>
    </force_lag_filter>
    <wheel_slip_filter> 0 </wheel_slip_filter>
    <max_steer unit="DEG"> 0 </max_steer>
    <brake_group> NONE </brake_group>
    <retractable>0</retractable>
</contact>
<contact type="STRUCTURE" name="TAIL_SKID">
    <location unit="IN">
        <x> 70.19 </x>
        <y> 0.0 </y>
        <z> -4.33 </z>
    </location>
    <static_friction>0.2</static_friction>
    <dynamic_friction>0.2</dynamic_friction>
    <spring_coeff unit="LBS/FT">20000</spring_coeff>
    <damping_coeff unit="LBS/FT/SEC">1000</damping_coeff>
</contact>
<contact type="STRUCTURE" name="LEFT_TIP">
    <location unit="IN">
        <x> 22.13 </x>
        <y> -47.24 </y>
        <z> 5.51 </z>
    </location>
    <static_friction>0.2</static_friction>
    <dynamic_friction>0.2</dynamic_friction>
    <spring_coeff unit="LBS/FT">10000</spring_coeff>
    <damping_coeff unit="LBS/FT/SEC">2000</damping_coeff>
</contact>
<contact type="STRUCTURE" name="RIGHT_TIP">
    <location unit="IN">
        <x> 22.13 </x>
        <y> 47.24 </y>
        <z> 5.51 </z>
    </location>
    <static_friction>0.2</static_friction>
    <dynamic_friction>0.2</dynamic_friction>
    <spring_coeff unit="LBS/FT">10000</spring_coeff>
    <damping_coeff unit="LBS/FT/SEC">2000</damping_coeff>
</contact>
</ground_reactions>

<propulsion>
    <engine file="1-uaveng">
        <location unit="IN">
            <x> 5.51 </x>
            <y> 0.0 </y>
            <z> 1.18 </z>
        </location>
        <orient unit="DEG">

```

```
<roll> 0 </roll>
<pitch> 0 </pitch>
<yaw> 0 </yaw>
</orient>
<feed>0</feed>
<thruster file="1_uav_prop">
  <location unit="IN">
    <x> 0.0 </x>
    <y> 0.0 </y>
    <z> 0.0 </z>
  </location>
  <orient unit="DEG">
    <roll> 0 </roll>
    <pitch> 0 </pitch>
    <yaw> 0 </yaw>
  </orient>
  <sense>1</sense>
  <p_factor>10.0</p_factor>
</thruster>
</engine>
<tank type="FUEL">
  <location unit="IN">
    <x> 12.69 </x>
    <y> 0.0 </y>
    <z> 0.0 </z>
  </location>
  <radius unit="IN"> 2.75 </radius>
  <capacity unit="LBS"> 1.296 </capacity>
  <contents unit="LBS"> 1.296 </contents>
</tank>
</propulsion>

<autopilot file="1-uavap"/>

<flight_control name="1-uav">
  <channel name="Pitch">
    <summer name="Pitch Trim Sum">
      <input>ap/elevator_cmd</input>
      <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>-28</min>
        <max> 23</max>
      </range>
    </aerosurface_scale>
  </channel>
</flight_control>
```

```

        <gain>0.01745</gain>
        <output>fcs/elevator-pos-rad</output>
    </aerosurface_scale>
</channel>
<channel name="Roll">
    <summer name="Roll Trim Sum">
        <input>ap/aileron_cmd</input>
        <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>
        <gain>0.01745</gain>
        <range>
            <min>-20</min>
            <max> 15</max>
        </range>
        <output>fcs/left-aileron-pos-rad</output>
    </aerosurface_scale>
    <aerosurface_scale name="Right Aileron Control">
        <input>-fcs/roll-trim-sum</input>
        <gain>0.01745</gain>
        <range>
            <min>-20</min>
            <max> 15</max>
        </range>
        <output>fcs/right-aileron-pos-rad</output>
    </aerosurface_scale>
</channel>
<channel name="Yaw">
    <summer name="Yaw Trim Sum">
        <input>fcs/rudder-cmd-norm</input>
        <input>fcs/yaw-trim-cmd-norm</input>
        <clipto>
            <min>-1</min>
            <max>1</max>
        </clipto>
    </summer>
    <aerosurface_scale name="Rudder Control">
        <input>fcs/yaw-trim-sum</input>
        <gain>0.01745</gain>
        <range>
            <min>-16</min>
            <max> 16</max>
        </range>
        <output>fcs/rudder-pos-rad</output>
    </aerosurface_scale>

```

```

    </channel>
  </flight_control>

  <aerodynamics>
    <alphalimits unit="DEG">
      <min>-0.087</min>
      <max>0.28</max>
    </alphalimits>
    <hysteresis_limits>
      <min>0.09</min>
      <max>0.36</max>
    </hysteresis_limits>
    <function name="aero/function/ground-effect-factor-lift">
      <description>Change in lift due to ground effect factor.</description>
      <table>
        <independentVar>aero/h_b-mac-ft</independentVar>
        <tableData>
          0.0  1.203
          0.1  1.127
          0.15 1.090
          0.2  1.073
          0.3  1.046
          0.4  1.055
          0.5  1.019
          0.6  1.013
          0.7  1.008
          0.8  1.006
          0.9  1.003
          1.0  1.002
          1.1  1.0
        </tableData>
      </table>
    </function>
    <function name="aero/function/ground-effect-factor-drag">
      <description>Change in drag due to ground effect</description>
      <table>
        <independentVar>aero/h_b-mac-ft</independentVar>
        <tableData>
          0.0  0.480
          0.1  0.515
          0.15 0.629
          0.2  0.709
          0.3  0.815
          0.4  0.882
          0.5  0.928
          0.6  0.962
          0.7  0.988
          0.8  1.0
          0.9  1.0
          1.0  1.0
          1.1  1.0
        </tableData>
      </table>
    </function>
  </aerodynamics>

```

```

        </tableData>
    </table>
</function>
<axis name="LIFT">
    <function name="aero/coefficient/CLwbh">
        <description>Lift due to alpha</description>
        <product>
            <property>aero/function/ground-effect-factor-lift</property>
            <property>aero/qbar-area</property>
            <table>
                <independentVar lookup="row">
                    aero/alpha-rad
                </independentVar>
                <independentVar lookup="column">
                    aero/stall-hyst-norm
                </independentVar>
                <tableData>
                    0.0  1.0
                    -0.09 -0.22 -0.22
                    0.0  0.25 0.25
                    0.09 0.73 0.73
                    0.1  0.83 0.78
                    0.12 0.92 0.79
                    0.14 1.02 0.81
                    0.16 1.08 0.82
                    0.17 1.13 0.83
                    0.19 1.19 0.85
                    0.21 1.25 0.86
                    0.24 1.35 0.88
                    0.26 1.44 0.9
                    0.28 1.47 0.92
                    0.3  1.43 0.95
                    0.32 1.38 0.99
                    0.34 1.3  1.05
                    0.36 1.15 1.15
                </tableData>
            </table>
        </product>
    </function>
    <function name="aero/coefficient/CLDe">
        <description>Lift due to Elevator Deflection.</description>
        <product>
            <property>aero/qbar-area</property>
            <property>fcs/elevator-pos-rad</property>
            <value>0.347</value>
        </product>
    </function>
    <function name="aero/coefficient/CLadot">
        <description>Lift due to alpha rate</description>
        <product>
            <property>aero/qbar-area</property>

```

```

        <property>aero/alphadot-rad_sec</property>
        <property>aero/ci2vel</property>
        <value>1.7</value>
    </product>
</function>
<function name="aero/coefficient/CLq">
    <description>Lift due to pitch rate</description>
    <product>
        <property>aero/qbar-area</property>
        <property>velocities/q-aero-rad_sec</property>
        <property>aero/ci2vel</property>
        <value>3.9</value>
    </product>
</function>
</axis>
<axis name="DRAG">
    <function name="aero/coefficient/CD0">
        <description>Drag at zero lift</description>
        <product>
            <property>aero/qbar-area</property>
            <value>0.026</value>
        </product>
    </function>
    <function name="aero/coefficient/CDwbh">
        <description>Drag due to alpha</description>
        <product>
            <property>aero/function/ground-effect-factor-drag</property>
            <property>aero/qbar-area</property>
            <table>
                <independentVar lookup="row">
                    aero/alpha-rad
                </independentVar>
                <independentVar lookup="column">
                    fcs/flap-pos-deg
                </independentVar>
                <tableData>
                    0.0
                    -0.0872664 -0.5500
                    -0.0698132 -0.4400
                    -0.0523599 -0.3300
                    -0.0349066 -0.2200
                    -0.0174533 -0.1100
                    0.0 0.0000
                    0.0174533 0.1100
                    0.0349066 0.2200
                    0.0523599 0.3300
                    0.0698132 0.4400
                    0.0872664 0.5500
                    0.10472 0.6600
                    0.122173 0.7700
                    0.139626 0.8800
                </tableData>
            </table>
        </product>
    </function>
</axis>

```

```

        0.15708    0.9900
        0.174533   1.0727
        0.191986   1.1539
        0.209439   1.2072
        0.226893   1.2169
        0.244346   1.1614
        0.261799   1.0478
        0.279253   0.9221
        0.296706   0.7826
        0.314159   0.7163
        0.331613   0.7091
        0.349066   0.7269
    </tableData>
  </table>
</product>
</function>
<function name="aero/coefficient/CDDe">
  <description>Drag due to Elevator Deflection</description>
  <product>
    <property>aero/qbar-area</property>
    <property>fcs/mag-elevator-pos-rad</property>
    <value>0.06</value>
  </product>
</function>
<function name="aero/coefficient/CDbeta">
  <description>Drag due to sideslip</description>
  <product>
    <property>aero/qbar-area</property>
    <property>aero/mag-beta-rad</property>
    <value>0.17</value>
  </product>
</function>
</axis>
<axis name="SIDE">
  <function name="aero/coefficient/CYb">
    <description>Side force due to beta</description>
    <product>
      <property>aero/qbar-area</property>
      <table>
        <independentVar>aero/beta-rad</independentVar>
        <tableData>
          -0.349  0.108
           0.0    0.0
           0.349 -0.108
        </tableData>
      </table>
    </product>
  </function>
  <function name="aero/coefficient/CYda">
    <description>Side force due to aileron</description>
    <product>

```

```

        <property>aero/qbar-area</property>
        <property>fcs/left-aileron-pos-rad</property>
        <value>-0.05</value>
    </product>
</function>
<function name="aero/coefficient/CYdr">
    <description>Side force due to rudder</description>
    <product>
        <property>aero/qbar-area</property>
        <property>fcs/rudder-pos-rad</property>
        <value>0.098</value>
    </product>
</function>
<function name="aero/coefficient/CYp">
    <description>Side force due to roll rate</description>
    <product>
        <property>aero/qbar-area</property>
        <property>aero/bi2vel</property>
        <property>velocities/p-aero-rad_sec</property>
        <value>-0.037</value>
    </product>
</function>
<function name="aero/coefficient/CYr">
    <description>Side force due to yaw rate</description>
    <product>
        <property>aero/qbar-area</property>
        <property>aero/bi2vel</property>
        <property>velocities/r-aero-rad_sec</property>
        <value>0.21</value>
    </product>
</function>
</axis>
<axis name="ROLL">
    <function name="aero/coefficient/Clb">
        <description>Roll moment due to beta</description>
        <product>
            <property>aero/qbar-area</property>
            <property>metrics/bw-ft</property>
            <table>
                <independentVar>aero/beta-rad</independentVar>
                <tableData>
                    -0.349  0.0311
                    0.0    0.0
                    0.349 -0.0311
                </tableData>
            </table>
        </product>
    </function>
    <function name="aero/coefficient/Clp">
        <description>Roll moment due to roll rate (roll damping)</description>
        <product>

```

```

        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>aero/bi2vel</property>
        <property>velocities/p-aero-rad_sec</property>
        <value>-0.47</value>
    </product>
</function>
<function name="aero/coefficient/Clr">
    <description>Roll moment due to yaw rate</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>aero/bi2vel</property>
        <property>velocities/r-aero-rad_sec</property>
        <table>
            <independentVar>aero/alpha-rad</independentVar>
            <tableData>
                0.000  0.08
                0.094  0.19
            </tableData>
        </table>
    </product>
</function>
<function name="aero/coefficient/Clda">
    <description>Roll moment due to aileron</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>fcs/left-aileron-pos-rad</property>
        <value>0.23</value>
    </product>
</function>
<function name="aero/coefficient/Cldr">
    <description>Roll moment due to rudder</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>fcs/rudder-pos-rad</property>
        <value>0.0147</value>
    </product>
</function>
</axis>
<axis name="PITCH">
    <function name="aero/coefficient/Cmalph">
        <description>Pitch moment due to alpha</description>
        <product>
            <property>aero/qbar-area</property>
            <property>metrics/cbarw-ft</property>
            <property>aero/alpha-rad</property>
            <value>-1.8</value>
        </product>
    </function>
</axis>

```

```

</function>
<function name="aero/coefficient/Cmq">
  <description>Pitch moment due to pitch rate</description>
  <product>
    <property>aero/qbar-area</property>
    <property>metrics/cbarw-ft</property>
    <property>aero/ci2vel</property>
    <property>velocities/q-aero-rad_sec</property>
    <value>-12.4</value>
  </product>
</function>
<function name="aero/coefficient/Cmadot">
  <description>Pitch moment due to alpha rate</description>
  <product>
    <property>aero/qbar-area</property>
    <property>metrics/cbarw-ft</property>
    <property>aero/ci2vel</property>
    <property>aero/alphadot-rad_sec</property>
    <value>-5.2</value>
  </product>
</function>
<function name="aero/coefficient/Cmo">
  <description>Pitching moment at zero alpha</description>
  <product>
    <property>aero/qbar-area</property>
    <property>metrics/cbarw-ft</property>
    <value>0.1</value>
  </product>
</function>
<function name="aero/coefficient/Cmde">
  <description>Pitch moment due to elevator deflection</description>
  <product>
    <property>aero/qbar-area</property>
    <property>metrics/cbarw-ft</property>
    <property>fcs/elevator-pos-rad</property>
    <value>-1.28</value>
  </product>
</function>
</axis>
<axis name="YAW">
  <function name="aero/coefficient/Cnb">
    <description>Yaw moment due to beta</description>
    <product>
      <property>aero/qbar-area</property>
      <property>metrics/bw-ft</property>
      <table>
        <independentVar>aero/beta-rad</independentVar>
        <tableData>
          -0.349  -0.0227
           0.0    0.0
           0.349  0.0227
        </tableData>
      </table>
    </product>
  </function>
</axis>

```

```

        </tableData>
    </table>
</product>
</function>
<function name="aero/coefficient/Cnp">
    <description>Yaw moment due to roll rate</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>aero/bi2vel</property>
        <property>velocities/p-aero-rad_sec</property>
        <value>-0.03</value>
    </product>
</function>
<function name="aero/coefficient/Cnr">
    <description>Yaw moment due to yaw rate</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>aero/bi2vel</property>
        <property>velocities/r-aero-rad_sec</property>
        <value>-0.099</value>
    </product>
</function>
<function name="aero/coefficient/Cnda">
    <description>Yaw moment due to aileron</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>fcs/left-aileron-pos-rad</property>
        <value>0.0053</value>
    </product>
</function>
<function name="aero/coefficient/Cndr">
    <description>Yaw moment due to rudder</description>
    <product>
        <property>aero/qbar-area</property>
        <property>metrics/bw-ft</property>
        <property>fcs/rudder-pos-rad</property>
        <value>-0.043</value>
    </product>
</function>
</axis>
</aerodynamics>
<input port="1137"/>
<output name="JSBout1-uav.csv" type="CSV" rate="20">
    <property> aero/qbar-psf </property>
    <property> attitude/phi-rad </property>
    <property> fcs/attitude/sensor/phi-rad </property>
    <property> position/h-sl-ft </property>
    <property> velocities/vc-kts </property>

```

```

    <property> fcs/throttle-cmd-norm </property>
    <property> propulsion/total-fuel-lbs </property>
    <property> flight-path/gamma-rad </property>
    <property> fcs/mixture-cmd-norm </property>
    <property>ap/heading_setpoint</property>
    <property>ap/heading_hold</property>
    <property>ap/altitude_setpoint</property>
    <property>ap/altitude_hold</property>
    <rates> ON </rates>
    <velocities> ON </velocities>
    <forces> ON </forces>
    <moments> ON </moments>
    <position> ON </position>
    <fcs> ON </fcs>
    <propulsion> ON </propulsion>
    <aerosurfaces> OFF </aerosurfaces>
    <fcs> OFF </fcs>
    <ground_reactions> ON </ground_reactions>
    <!-- Use the cycle_duration property for standalone only. -->
    <property> simulation/cycle_duration </property>
</output>
<output name="localhost" type="SOCKET" port="1138" rate="20">
    <simulation> OFF </simulation>
    <atmosphere> OFF </atmosphere>
    <massprops> OFF</massprops>
    <rates> OFF </rates>
    <velocities> OFF </velocities>
    <forces> OFF </forces>
    <moments> OFF </moments>
    <position> OFF </position>
    <propulsion> OFF </propulsion>
    <aerosurfaces> OFF </aerosurfaces>
    <fcs> OFF </fcs>
    <ground_reactions> OFF </ground_reactions>
    <coefficients> OFF </coefficients>
    <property> position/h-agl-ft </property>
    <property> velocities/vc-kts </property>
    <property> attitude/phi-rad </property>
    <property> fcs/attitude/sensor/phi-rad </property>
</output>
<output name="localhost" type="SOCKET" port="1140" rate="40">
    <simulation> OFF </simulation>
    <atmosphere> OFF </atmosphere>
    <massprops> OFF</massprops>
    <aerosurfaces> OFF </aerosurfaces>
    <rates> OFF </rates>
    <velocities> ON </velocities>
    <forces> OFF </forces>
    <moments> OFF </moments>
    <position> OFF </position>
    <propulsion> OFF </propulsion>

```

```

    <fcs> OFF </fcs>
    <ground_reactions> OFF </ground_reactions>
    <coefficients> OFF </coefficients>
  </output>
</fdm_config>

```

B.1.2. Archivo del piloto automático

```

<?xml version="1.0"?>
<!--

  Authors:   Teresa Marin and Mar Noguera
  Date:     18 August 2002
  Function:  UAV autopilot test file
-->

<autopilot name="1-uav Autopilot">

<!-- INTERFACE PROPERTIES -->

  <property>ap/attitude_hold</property>
  <property>ap/altitude_hold</property>
  <property>ap/heading_hold</property>
  <property>ap/altitude_setpoint</property>
  <property>ap/heading_setpoint</property>
  <property>ap/aileron_cmd</property>
  <property>ap/elevator_cmd</property>
  <property>ap/airspeed_setpoint</property>
  <property>ap/airspeed_hold</property>
  <property>ap/throttle-cmd-norm</property>

<!-- <property>attitude/sensor/phi-rad</property> -->

<!-- SENSOR -->

  <sensor name="fcs/attitude/sensor/phi-rad">
    <input> attitude/phi-rad </input>
    <lag> 0.5 </lag>
    <noise variation="PERCENT"> 0.05 </noise>
    <quantization name="attitude/sensor/quantized/phi-rad">
      <bits> 12 </bits>
      <min> -180 </min>
      <max> 180 </max>
    </quantization>
    <bias> 0.001 </bias>
  </sensor>

<!--
=====

```

ROLL CHANNEL

=====

-->

```

<channel name="fcs/ap-roll-wing-leveler">
  <pure_gain name="fcs/roll-ap-wing-leveler">
    <input>fcs/attitude/sensor/phi-rad</input>
    <gain>2.0</gain>
    <clipto>
      <min>-0.255</min>
      <max>0.255</max>
    </clipto>
  </pure_gain>
  <integrator name="fcs/roll-ap-error-integrator">
    <input> attitude/phi-rad </input>
    <c1> 0.125 </c1>
  </integrator>
  <summer name="fcs/roll-ap-error-summer">
    <input> velocities/p-rad_sec</input>
    <input> fcs/roll-ap-wing-leveler</input>
    <input> fcs/roll-ap-error-integrator</input>
    <clipto>
      <min>-1.0</min>
      <max> 1.0</max>
    </clipto>
  </summer>
  <switch name="fcs/roll-ap-autoswitch">
    <default value="0.0"/>
    <test value="fcs/roll-ap-error-summer">
      ap/attitude_hold == 1
    </test>
  </switch>
  <pure_gain name="fcs/roll-ap-aileron-command-normalizer">
    <input>fcs/roll-ap-autoswitch</input>
    <gain>-1</gain>
  </pure_gain>
</channel>
<channel name="fcs/ap-roll-heading-hold">
  <pure_gain name="fcs/heading-true-degrees">
    <input>attitude/heading-true-rad</input>
    <gain>57.3</gain> <!-- convert to degrees -->
  </pure_gain>
  <summer name="fcs/heading-error">
    <input> -fcs/heading-true-degrees</input>
    <input> ap/heading_setpoint </input>
  </summer>
  <switch name="fcs/heading-error-bias-switch">
    <default value="0.0"/>
    <test value="360.0">
      fcs/heading-error lt -180
    </test>

```

```

    <test value="-360.0">
      fcs/heading-error gt 180
    </test>
  </switch>
</summer name="fcs/heading-corrected">
  <input> fcs/heading-error-bias-switch </input>
  <input> fcs/heading-error </input>
  <clipto>
    <min>-30</min>
    <max> 30</max>
  </clipto>
</summer>
<pure_gain name="fcs/heading-command">
  <input> fcs/heading-corrected </input>
  <gain> 0.01745 </gain>
</pure_gain>
<lag_filter name="fcs/heading-roll-error-lag">
  <input> fcs/heading-command </input>
  <c1> 0.50 </c1>
</lag_filter>
<summer name="fcs/heading-roll-error">
  <input> fcs/heading-roll-error-lag </input>
  <input> -attitude/phi-rad </input>
</summer>
<switch name="fcs/heading-roll-error-switch">
  <default value="0.0"/>
  <test value="fcs/heading-roll-error">
    ap/heading_hold == 1
  </test>
</switch>
<pid name="fcs/roll-pi-controller">
  <input> fcs/heading-roll-error-switch </input>
  <kp> 1.0 </kp>
  <ki> 0.10 </ki>
</pid>
<summer name="fcs/heading-error-summer">
  <input> fcs/roll-pi-controller </input>
  <clipto>
    <min>-1.0</min>
    <max> 1.0</max>
  </clipto>
</summer>
<pid name="fcs/heading-pi-controller">
  <input> fcs/heading-roll-error-switch </input>
  <kp> 1.0 </kp>
  <ki> 0.10 </ki>
</pid>
<switch name="fcs/roll-command-selector">
  <default value="0.0"/>
  <test value="fcs/heading-pi-controller">
    ap/heading_hold == 1

```

```

    </test>
    <test value="fcs/roll-ap-aileron-command-normalizer">
      ap/attitude_hold == 1
    </test>
    <output>ap/aileron_cmd</output>
  </switch>
</channel>

<!--
=====
PITCH CHANNEL
=====
-->
<channel name="fcs/ap-pitch-altitude-hold">
  <summer name="fcs/altitude-error">
    <input> ap/altitude_setpoint </input>
    <input> -position/h-sl-ft </input>
    <clipto>
      <min>-100</min>
      <max> 100</max>
    </clipto>
  </summer>
  <lag_filter name="fcs/alt-error-lag">
    <input> fcs/altitude-error </input>
    <c1> 1 </c1>
  </lag_filter>
  <scheduled_gain name="fcs/hdot-command">
    <input> fcs/alt-error-lag </input>
    <table>
      <independentVar>position/h-sl-ft</independentVar>
      <tableData>
        0.0  0.12
        1000.0  0.11
        2000.0  0.10
        3000.0  0.096
        4000.0  0.093
        5000.0  0.086
        6000.0  0.078
        7000.0  0.069
        8000.0  0.061
        9000.0  0.053
        10000.0  0.045
        11000.0  0.037
        12000.0  0.028
      </tableData>
    </table>
  </scheduled_gain>
  <summer name="fcs/hdot-error">
    <input> fcs/hdot-command </input>
    <input> -velocities/h-dot-fps </input>

```

```

</summer>
<switch name="fcs/ap-alt-hold-switch">
  <default value="0.0"/>
  <test value="fcs/hdot-error">
    ap/altitude_hold == 1
  </test>
</switch>
<deadband name="fcs/windup-trigger">
  <input> fcs/elevator-pos-deg </input>
  <width>46.0</width>
</deadband>
<pid name="fcs/altitude-hold-pid">
  <input> fcs/ap-alt-hold-switch </input>
  <kp> 0.031 </kp>
  <ki> 0.000028 </ki>
  <!-- <kd> 0.00005 </kd> -->
  <trigger> fcs/windup-trigger </trigger>
  <clipto> <min>-1.0</min>
    <max> 1.0</max> </clipto>
</pid>
<pure_gain name="fcs/elevator">
  <input> fcs/altitude-hold-pid </input>
  <gain> -1.0 </gain>
  <output> ap/elevator_cmd </output>
</pure_gain>
</channel>
</autopilot>

```

B.1.3. Archivo de la hélice

```

<?xml version="1.0"?>
<propeller name="UAV motor helice">
  <ixx> 0.01 </ixx>
  <diameter unit="IN"> 7.20 </diameter>
  <numblades> 2 </numblades>
  <minpitch> 21.6 </minpitch>
  <maxpitch> 21.6 </maxpitch>
  <table name="C_THRUST" type="internal">
    <tableData>
      0.0 0.108
      0.2 0.104
      0.4 0.100
      0.6 0.080
      0.8 0.052
      1.0 0.020
      1.11 0.000
    </tableData>
  </table>
  <table name="C_POWER" type="internal">

```

```

    <tableData>
      0.0  0.080
      0.2  0.075
      0.4  0.069
      0.6  0.061
      0.8  0.050
      1.0  0.022
      1.11 0.000
    </tableData>
  </table>
</propeller>

```

B.1.4. Archivo del motor

```

<?xml version="1.0"?>
<piston_engine name="1-uaveng">
  <minmp unit="INHG"> 6.5 </minmp>
  <maxmp unit="INHG"> 29.92 </maxmp>
  <displacement unit="IN3"> 1.0 </displacement>
  <maxhp> 4.2 </maxhp>
  <cycles> 2.0 </cycles>
  <idlerpm> 550.0 </idlerpm>
  <maxthrottle> 1.0 </maxthrottle>
  <minthrottle> 0.2 </minthrottle>
</piston_engine>

```

B.2. Archivos de la UAV creados a partir del Aeromatic

B.2.1. Archivo principal de configuración

```

<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="http://jsbsim.sourceforge.net/JSBSim.xsl"?>
<fdm_config name="1-uav" version="2.0" release="ALPHA"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sourceforge.net/JSBSim.xsd">
  <fileheader>
    <author>Mar Noguera and Teresa Marin</author>
    <filecreationdate>2008-06-04</filecreationdate>
    <version>$Revision: 1 $</version>
    <description>Models a 1-uav</description>
  </fileheader>

  <!-- 1 lbs = 453.592 g = 0.453 kg -->
  <!-- 1 ft = 0.3048 m -->
  <!-- 1 m^2= 10.76 ft^2 -->
  <!-- 1 in = 0.0254 m = 2.54 cm -->

```

```

<!-- 1 slug = 14.5939 kg -->

  <metrics>
    <wingarea unit="FT2"> 10.54 </wingarea>
    <wingspan unit="FT"> 7.87 </wingspan> <!-- envergadura -->
  <wing_incidence> 2.00 </wing_incidence>
    <chord unit="FT"> 1.35 </chord> <!-- cuerda -->
    <htailarea unit="FT2"> 0.69 </htailarea> <!-- area estabilizador horizontal-->
    <htailarm unit="FT"> 3.38 </htailarm> <!-- brazo cola horizontal -->
    <vtailarea unit="FT2"> 0.599 </vtailarea> <!-- area estabilizador vertical -->
  <vtailarm unit="FT"> 1.017 </vtailarm> <!-- brazo cola vertical -->
  <location name="AERORP" unit="IN">
    <x> 4.76 </x>
    <y> 0.00 </y>
    <z> 0.00 </z>
  </location>
  <location name="EYEPOINT" unit="IN">
    <x> 109.62 </x>
    <y> -18.00 </y>
    <z> 45.00 </z>
  </location>
  <location name="VRP" unit="IN">
    <x>0</x>
    <y>0</y>
    <z>0</z>
  </location>
</metrics>

  <mass_balance>
    <ixx unit="SLUG*FT2"> 0.715 </ixx>
    <iyy unit="SLUG*FT2"> 1.146 </iyy>
    <izz unit="SLUG*FT2"> 1.560 </izz>
    <ixz unit="SLUG*FT2"> -0.028 </ixz>
    <emptywt unit="LBS"> 15.54 </emptywt>
    <location name="CG" unit="IN">
      <x> 22.91 </x>
      <y> 0.0 </y>
      <z> 0.0 </z>
    </location>
    <pointmass name="LUGGAGE">
      <weight unit="LBS">1.0</weight>
      <location name="POINTMASS" unit="IN">
        <x> 25.98 </x>
        <y> 0.0 </y>
        <z> -1.97 </z>
      </location>
    </pointmass>
  </mass_balance>

```

```
<ground_reactions>

  <contact type="BOGEY" name="LEFT_MAIN">
    <location unit="IN">
      <x> 18.11 </x>
      <y> -10.04 </y>
      <z> -4.96 </z>
    </location>
    <static_friction> 0.80 </static_friction>
    <dynamic_friction> 0.50 </dynamic_friction>
    <rolling_friction> 0.02 </rolling_friction>
    <spring_coeff unit="LBS/FT"> 17.84 </spring_coeff>
    <damping_coeff unit="LBS/FT/SEC"> 3.57 </damping_coeff>
    <max_steer unit="DEG">0</max_steer>
    <brake_group>LEFT</brake_group>
    <retractable>FIXED</retractable>
  </contact>

  <contact type="BOGEY" name="RIGHT_MAIN">
    <location unit="IN">
      <x> 18.11 </x>
      <y> 10.04 </y>
      <z> -4.96 </z>
    </location>
    <static_friction> 0.80 </static_friction>
    <dynamic_friction> 0.50 </dynamic_friction>
    <rolling_friction> 0.02 </rolling_friction>
    <spring_coeff unit="LBS/FT"> 17.84 </spring_coeff>
    <damping_coeff unit="LBS/FT/SEC"> 3.57 </damping_coeff>
    <max_steer unit="DEG">0</max_steer>
    <brake_group>RIGHT</brake_group>
    <retractable>FIXED</retractable>
  </contact>

  <contact type="BOGEY" name="TAIL">
    <location unit="IN">
      <x> 70.2 </x>
      <y> 0.0 </y>
      <z> -4.66 </z>
    </location>
    <static_friction> 0.80 </static_friction>
    <dynamic_friction> 0.50 </dynamic_friction>
    <rolling_friction> 0.02 </rolling_friction>
    <spring_coeff unit="LBS/FT"> 17.84 </spring_coeff>
    <damping_coeff unit="LBS/FT/SEC"> 14.27 </damping_coeff>
    <max_steer unit="DEG"> 5.00 </max_steer>
    <brake_group>NONE</brake_group>
    <retractable>FIXED</retractable>
  </contact>

  <contact type="STRUCTURE" name="LEFT_WING">
```

```

<location unit="IN">
  <x> 22.13 </x>
  <y> -47.24 </y>
  <z> 5.51 </z>
</location>
<static_friction> 0.80 </static_friction>
<dynamic_friction> 0.50 </dynamic_friction>
<spring_coeff unit="LBS/FT"> 17.84 </spring_coeff>
<damping_coeff unit="LBS/FT/SEC"> 3.57 </damping_coeff>
</contact>

<contact type="STRUCTURE" name="RIGHT_WING">
  <location unit="IN">
    <x> 22.13 </x>
    <y> 47.24 </y>
    <z> 5.51 </z>
  </location>
  <static_friction> 0.80 </static_friction>
  <dynamic_friction> 0.50 </dynamic_friction>
  <spring_coeff unit="LBS/FT"> 17.84 </spring_coeff>
  <damping_coeff unit="LBS/FT/SEC"> 3.57 </damping_coeff>
</contact>

</ground_reactions>

<propulsion>
  <engine file="1-uaveng">
    <location unit="IN">
      <x> 5.51 </x>
      <y> 0.0 </y>
      <z> 1.18 </z>
    </location>
    <orient unit="DEG">
      <roll> 0 </roll>
      <pitch> 0 </pitch>
      <yaw> 0 </yaw>
    </orient>
    <feed>0</feed>
    <thruster file="1_uav_prop">
      <location unit="IN">
        <x> 5.51 </x>
        <y> 0.0 </y>
        <z> 0.0 </z>
      </location>
      <orient unit="DEG">
        <roll> 0 </roll>
        <pitch> 0 </pitch>
        <yaw> 0 </yaw>
      </orient>
      <sense>1</sense>
      <p_factor>10.0</p_factor>
    </thruster>
  </engine>
</propulsion>

```

```
        </thruster>
    </engine>

    <tank type="FUEL"      <!-- Tank number 0 -->
        <location unit="IN">
            <x> 12.69 </x>
            <y> 0.0 </y>
            <z> 0.0 </z>
        </location>
        <radius unit="IN"> 2.75 </radius>
        <capacity unit="LBS"> 1.296 </capacity>
        <contents unit="LBS"> 1.296 </contents>
    </tank>
</propulsion>

<!--
    <autopilot file="1-uavap"/>
-->
<flight_control name="FCS: 1-uav">

    <channel name="Pitch">
        <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> 0.35 </max>
            </range>
            <output>fcs/elevator-pos-rad</output>
        </aerosurface_scale>

        <aerosurface_scale name="elevator normalization">
            <input>fcs/elevator-pos-rad</input>
            <domain>
                <min> -0.35 </min>
                <max> 0.35 </max>
            </domain>
            <range>
                <min> -1 </min>
                <max> 1 </max>
            </range>
        </aerosurface_scale>
    </channel>
</flight_control>
```

```

    </range>
    <output>fcs/elevator-pos-norm</output>
</aerosurface_scale>

</channel>

<channel name="Roll">
  <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>

</channel>

<channel name="Yaw">
  <summer name="Rudder Command Sum">
    <input>fcs/rudder-cmd-norm</input>
    <input>fcs/yaw-trim-cmd-norm</input>
    <clipto>
      <min> -0.35 </min>
      <max> 0.35 </max>
    </clipto>
  </summer>

<aerosurface_scale name="Rudder Control">
  <input>fcs/rudder-command-sum</input>
  <range>
    <min> -0.35 </min>
    <max> 0.35 </max>
  </range>
  <output>fcs/rudder-pos-rad</output>
</aerosurface_scale>

<aerosurface_scale name="rudder normalization">
  <input>fcs/rudder-pos-rad</input>
  <domain>
    <min> -0.35 </min>
    <max> 0.35 </max>
  </domain>
  <range>
    <min> -1 </min>
    <max> 1 </max>
  </range>
  <output>fcs/rudder-pos-norm</output>
</aerosurface_scale>

```

```

</channel>

<channel name="Landing Gear">
</channel>

<channel name="Speedbrake">
  <kinematic name="Speedbrake Control">
    <input>fcs/speedbrake-cmd-norm</input>
    <traverse>
      <setting>
        <position> 0 </position>
        <time> 0 </time>
      </setting>
      <setting>
        <position> 1 </position>
        <time> 1 </time>
      </setting>
    </traverse>
    <output>fcs/speedbrake-pos-norm</output>
  </kinematic>

</channel>

</flight_control>

<aerodynamics>

  <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 lookup="row">aero/alpha-rad</independentVar>
          <tableData>
            -0.20 -0.750
            0.00 0.250
            0.23 1.400
            0.60 0.710
          </tableData>
        </table>
      </product>
    </function>

    <function name="aero/coefficient/dCLsb">
      <description>Delta_Lift_due_to_speedbrake</description>
      <product>
        <property>aero/qbar-psf</property>

```

```

        <property>metrics/Sw-sqft</property>
        <property>fcs/speedbrake-pos-norm</property>
        <value>0</value>
    </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>0.2</value>
    </product>
</function>

</axis>

<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 lookup="row">aero/alpha-rad</independentVar>
                <tableData>
                    -1.57      1.500
                    -0.26     0.036
                    0.00      0.028
                    0.26     0.036
                    1.57      1.500
                </tableData>
            </table>
        </product>
    </function>

<function name="aero/coefficient/CDwbh">
    <description>Drag due to alpha</description>
    <product>
        <property>aero/qbar-area</property>
        <table>
            <independentVar lookup="row">aero/alpha-rad</independentVar>
            <independentVar lookup="column">fcs/flap-pos-deg</independentVar>
            <tableData>
                0.0
                -0.0872664  0.00407143
                -0.0698132  0.0013424
                -0.0523599  8.96747e-05
                -0.0349066  0.000313268
            </tableData>
        </table>
    </product>
</function>

```

```

-0.0174533  0.00201318
  0.0        0.0051894
  0.0174533  0.00993967
  0.0349066  0.0162201
  0.0523599  0.0240308
  0.0698132  0.0333717
  0.0872664  0.0442427
  0.10472    0.056644
  0.122173   0.0705754
  0.139626   0.086037
  0.15708    0.096239
  0.174533   0.106859
  0.191986   0.118034
  0.209439   0.129765
  0.226893   0.142366
  0.244346   0.156452
  0.261799   0.172732
  0.279253   0.178209
  0.296706   0.171598
  0.314159   0.161786
  0.331613   0.147493
  0.349066   0.109678
</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>
    <value>0.04</value>
  </product>
</function>

<function name="aero/coefficient/CDmach">
  <description>Drag_due_to_mach</description>
  <product>
    <property>aero/qbar-psf</property>
    <property>metrics/Sw-sqft</property>
    <table>
      <independentVar lookup="row">velocities/mach</independentVar>
      <tableData>
        0.00    0.000
        0.7    0.000
        1.10   0.023
        1.80   0.015
      </tableData>
    </table>
  </product>
</function>

```

```

    </product>
</function>

<function name="aero/coefficient/CDsb">
  <description>Drag_due_to_speedbrakes</description>
  <product>
    <property>aero/qbar-psf</property>
    <property>metrics/Sw-sqft</property>
    <property>fcs/speedbrake-pos-norm</property>
    <value>0.028</value>
  </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 lookup="row">aero/beta-rad</independentVar>
      <tableData>
        -1.57      1.230
        -0.26     0.050
         0.00     0.000
         0.26     0.050
         1.57     1.230
      </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>0.04</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>
    </product>
  </function>

```

```
        <property>aero/beta-rad</property>
        <value>-1</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>-0.1</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>
            <property>velocities/p-aero-rad_sec</property>
            <value>-0.4</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>
            <property>velocities/r-aero-rad_sec</property>
            <value>0.15</value>
        </product>
    </function>

    <function name="aero/coefficient/Clr">
        <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>0.01</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>
            <value>-0.5</value>
        </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>
            <table>
                <independentVar lookup="row">velocities/mach</independentVar>
                <tableData>
                    0.0    -1.100
                    2.0    -0.275
                </tableData>
            </table>
        </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>
            <property>velocities/q-aero-rad_sec</property>
            <value>-12</value>
        </product>
    </function>

    <function name="aero/coefficient/Cmadot">
        <description>Pitch_moment_due_to_alpha_rate</description>

```

```
<product>
  <property>aero/qbar-psf</property>
  <property>metrics/Sw-sqft</property>
  <property>metrics/cbarw-ft</property>
  <property>aero/ci2vel</property>
  <property>aero/alphadot-rad_sec</property>
  <value>-7</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>0.12</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>-0.15</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>-0.1</value>
    </product>
  </function>

  <function name="aero/coefficient/Cnda">
    <description>Adverse_yaw</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>-0.01</value>
    </product>
</function>

</axis>

</aerodynamics>

    <output name="JSBout1-uav.csv" type="CSV" rate="4">
        <rates> ON </rates>
        <velocities> ON </velocities>
        <forces> ON </forces>
        <moments> ON </moments>
        <position> ON </position>
        <fcs> ON </fcs>
        <propulsion> ON </propulsion>
        <aerosurfaces> OFF </aerosurfaces>
        <fcs> OFF </fcs>
        <ground_reactions> ON </ground_reactions>

    </output>

</fdm_config>

```

B.2.2. Archivo del motor

```

<?xml version="1.0"?>
<!--
File:      1-uaveng.xml
Author:    Aero-Matic v 0.81

Inputs:
name:      1-uaveng
type:      piston
power:     4.2 hp
augmented? no
injected?  no
-->

<piston_engine name="1-uaveng">
  <minmp unit="INHG">      6.0 </minmp><!-- Deprecated -->
  <maxmp unit="INHG">     29.9 </maxmp><!-- Deprecated -->
  <displacement unit="IN3"> 7.98 </displacement>

```

```

<!-- Defining <bsfc> over-rides the bsfc value calculated from <maxhp> -->
  <!--<bsfc>          0.45 </bsfc>-->
  <maxhp>          4.20 </maxhp>
  <cycles>         4.0 </cycles>
  <idlerpm>        700.0 </idlerpm>
  <maxrpm>         2800.0 </maxrpm>
  <maxthrottle>    1.0 </maxthrottle>
  <minthrottle>    0.2 </minthrottle>
  <sparkfaildrop> 0.0 </sparkfaildrop>
</piston_engine>

```

B.2.3. Archivo de la hélice

```

<?xml version="1.0"?>
<!-- Generated by Aero-Matic v 0.8

  Inputs:
    horsepower: 4.2
    pitch: fixed
    max engine rpm: 7500
    prop diameter (ft): 1.6433333333333333

  Outputs:
    max prop rpm:11417.65
    gear ratio: 0.66
    Cp0:0.011762
    Ct0:0.010116
    static thrust (lbs): 6.35
-->

<propeller name="1_uav_prop">
  <ixx> 0.0011 </ixx>
  <diameter unit="IN"> 19.7 </diameter>
  <numblades> 2 </numblades>
  <gearratio>0.66 </gearratio>

  <table name="C_THRUST" type="internal">
    <tableData>
      0.0 0.0101
      0.1 0.0097
      0.2 0.0093
      0.3 0.0085
      0.4 0.0077
      0.5 0.0068
      0.6 0.0055
      0.7 0.0041
      0.8 0.0022
      1.0 -0.0008
      1.2 -0.0040
    </tableData>
  </table>

```

```
    1.4 -0.0072
  </tableData>
</table>
```

```
<table name="C_POWER" type="internal">
  <tableData>
    0.0 0.0118
    0.1 0.0118
    0.2 0.0115
    0.3 0.0112
    0.4 0.0106
    0.5 0.0097
    0.6 0.0089
    0.7 0.0075
    0.8 0.0061
    1.0 0.0022
    1.2 -0.0035
    1.4 -0.0105
    1.6 -0.0178
  </tableData>
</table>

</propeller>
```

APÉNDICE C. SIMULACIONES

A continuación adjuntamos el código entero de todas las simulaciones creadas a lo largo de este proyecto

C.1. Simulación “FIN”

Seguidamente encontrará el código de la simulación que dibuja “FIN” en el cielo:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
href="http://jsbsim.sourceforge.net/JSBSimScript.xsl"?>
<runscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sf.net/JSBSimScript.xsd"
  name="fin">

  <use aircraft="c172x" initialize="reset00"/>

  <run start="0.0" end="10000" dt="0.0083333">

    <property> simulation/notify-time-trigger </property>

    <event name="engine start">
      <description>Start the engine</description>
      <notify/>
      <condition>
        sim-time-sec >= 0.25
      </condition>
      <set name="fcs/throttle-cmd-norm" value="1.0" action="FG_RAMP" tc="0.5"/>
      <set name="fcs/mixture-cmd-norm" value="0.87" action="FG_RAMP" tc="0.5"/>
      <set name="propulsion/magneto_cmd" value="3"/>
      <set name="propulsion/starter_cmd" value="1"/>
    </event>

    <event name="Set heading hold 1">
      <description>Set Heading to 45 when 5 ft AGL is reached</description>
      <notify/>
      <condition>
        position/h-agl-ft >= 5
      </condition>
      <set name="ap/heading_setpoint" value="45"/>
      <set name="ap/attitude_hold" value="0"/>
      <set name="ap/heading_hold" value="1"/>
    </event>

    <event name="Set heading hold 2">
      <description>Set Heading to 180</description>
      <notify/>
```

```
<condition>
  sim-time-sec >= 2500
</condition>
<set name="ap/heading_setpoint" value="180"/>
<set name="ap/attitude_hold" value="0"/>
<set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 3">
  <description>Set Heading to 270</description>
  <notify/>

  <condition>
    sim-time-sec >= 3000
  </condition>
  <set name="ap/heading_setpoint" value="270"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 4">
  <description>Set Heading to 180</description>
  <notify/>
  <condition>
    sim-time-sec >= 4500
  </condition>
  <set name="ap/heading_setpoint" value="180"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 5">
  <description>Set Heading to 45</description>
  <notify/>
  <condition>
    sim-time-sec >= 5000
  </condition>
  <set name="ap/heading_setpoint" value="45"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 6">
  <description>Set Heading to 270</description>
  <notify/>
  <condition>
    sim-time-sec >= 6500
  </condition>
  <set name="ap/heading_setpoint" value="270"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>
```

```
</event>

<event name="Set heading hold 7">
  <description>Set Heading to 0</description>
  <notify/>
  <condition>
    sim-time-sec >= 7000
  </condition>
  <set name="ap/heading_setpoint" value="0"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 8">
  <description>Set Heading to 90</description>
  <notify/>
  <condition>
    sim-time-sec >= 7100
  </condition>
  <set name="ap/heading_setpoint" value="90"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 9">
  <description>Set Heading to 0</description>
  <notify/>
  <condition>
    sim-time-sec >= 7600
  </condition>
  <set name="ap/heading_setpoint" value="0"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 10">
  <description>Set Heading to 270</description>
  <notify/>
  <condition>
    sim-time-sec >= 7800
  </condition>
  <set name="ap/heading_setpoint" value="270"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 11">
  <description>Set Heading to 0</description>
  <notify/>
  <condition>
    sim-time-sec >= 8300
```

```
</condition>
<set name="ap/heading_setpoint" value="0"/>
<set name="ap/attitude_hold" value="0"/>
<set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 12">
  <description>Set Heading to 90</description>
  <notify/>
  <condition>
    sim-time-sec >= 8350
  </condition>
  <set name="ap/heading_setpoint" value="90"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 13">
  <description>Set Heading to 0</description>
  <notify/>
  <condition>
    sim-time-sec >= 8850
  </condition>
  <set name="ap/heading_setpoint" value="0"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 14">
  <description>Set Heading to 270</description>
  <notify/>
  <condition>
    sim-time-sec >= 9050
  </condition>
  <set name="ap/heading_setpoint" value="270"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>

<event name="Set heading hold 15">
  <description>Set Heading to 0</description>
  <notify/>
  <condition>
    sim-time-sec >= 9550
  </condition>
  <set name="ap/heading_setpoint" value="0"/>
  <set name="ap/attitude_hold" value="0"/>
  <set name="ap/heading_hold" value="1"/>
</event>
```

```
</event>

<event name="Set autopilot for 20 ft.">
  <description>Set Autopilot for 20 ft</description>
  <notify/>
  <condition>
    aero/qbar-psf >= 4
  </condition>
  <set name="ap/altitude_setpoint" value="100.0"/>
  <set name="ap/altitude_hold" value="1"/>
  <set name="fcs/flap-cmd-norm" value=".33"/>
</event>

<event name="Adjust throttle/flaps">
  <description>Remove flaps at 50 ft</description>
  <notify/>
-
  <condition>
    position/h-agl-ft >= 100
  </condition>
  <set name="fcs/mixture-cmd-norm" value="0.69" action="FG_EXP" tc ="400.0"/>
  <set name="fcs/flap-cmd-norm" value="0"/>
</event>

<event name="Set autopilot for 7000 ft.">
  <description>Set Autopilot for 2000 ft</description>
  <notify/>
  <condition>
    sim-time-sec >= 100
  </condition>
  <set name="ap/altitude_setpoint" value="7000.0"/>
</event>

<event name="Time Notify" type="FG_DELTA" persistent="true">
  <description>Output message at 100 second intervals</description>
  <notify>
    <property>velocities/vc-kts</property>
    <property>position/h-agl-ft</property>
  </notify>
  <condition> sim-time-sec >= simulation/notify-time-trigger </condition>
  <set name="simulation/notify-time-trigger" value="100" type="FG_DELTA"/>
</event>

</run>
</runscript>
```

C.2. Simulación “BARRENA”

A continuación se adjunta la simulación de la caída en barrena de la Cessna:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="http://jsbsim.sourceforge.net/JSBSimScript.xsl"?>
<runscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sf.net/JSBSimScript.xsd"
  name="barrina">

<use aircraft="c172x" initialize="reset01"/>

<run start="0.0" end="100" dt="0.0083333">

  <property> simulation/notify-time-trigger </property>

  <event name="engine start">
    <description>Start the engine</description>
    <notify/>
    <condition>
      sim-time-sec >= 0.25
    </condition>
    <set name="fcs/throttle-cmd-norm" value="1.0" action="FG_RAMP" tc="0.5"/>
    <set name="fcs/mixture-cmd-norm" value="0.87" action="FG_RAMP" tc="0.5"/>
    <set name="propulsion/magneto_cmd" value="3"/>
    <set name="propulsion/starter_cmd" value="1"/>
  </event>

  <event name="Dive the airplane">
    <description>Dive the airplane 1</description>
    <notify/>
    <condition>
      sim-time-sec >= 15
    </condition>
    <set name="fcs/elevator-cmd-norm" value="0.04" action="FG_RAMP" tc="0.6"/>
  </event>

  <!--
Estabilizamos la palanca, pero el avion sigue con su inercia de picado.
  <event name="xxx the airplane">
    <description>xxx the airplane 1</description>
    <notify/>
    <condition>
      sim-time-sec >= 20
    </condition>
    <set name="fcs/elevator-cmd-norm" value="0.0" action="FG_RAMP" tc="0.6"/>
  </event>
-->
```

```
<event name="push the airplane">
  <description>push the airplane 1</description>
  <notify/>
  <condition>
    sim-time-sec >= 20
  </condition>
  <set name="fcs/elevator-cmd-norm" value="-0.6" action="FG_RAMP" tc ="0.6"/>
</event>

<!--
<event name="push control the airplane">
  <description>push the airplane 1</description>
  <notify/>
  <condition>
    sim-time-sec >= 23
  </condition>
  <set name="fcs/elevator-cmd-norm" value="0" action="FG_RAMP" tc ="0.6"/>
</event>
-->

<event name="autopilot">
  <description>start autopilot</description>
  <notify/>
  <condition>
    sim-time-sec >= 30
  </condition>
  <set name="ap/altitude_hold" value= "0"/>
  <altitude unit="FT"> 6000.0 </altitude>
</event>

<event name="Time Notify" type="FG_DELTA" persistent="true">
  <description>Output message at 250 second intervals</description>
  <notify>
    <property>velocities/vc-kts</property>
    <property>position/h-agl-ft</property>
  </notify>
  <condition> sim-time-sec >= simulation/notify-time-trigger </condition>
  <set name="simulation/notify-time-trigger" value="250" type="FG_DELTA"/>
</event>

</run>
</runscript>
```

C.3. Simulación “LOOPING”

A continuación se adjunta la simulación completa de la acrobacia del looping con el OV-10 “Bronco”:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl"
  href="http://jsbsim.sourceforge.net/JSBSimScript.xsl"?>
<runscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sf.net/JSBSimScript.xsd"
  name="OV-10 runway test">

  <description>OV-10 Looping</description>

  <use aircraft="OV10" initialize="reset01"/>
  <run start="0.0" end="250" dt="0.0083333">

    <property> simulation/notify-time-trigger </property>

    <event name="engine start">
      <description>Start the engine</description>
      <notify/>
      <condition>
        sim-time-sec >= 0.25
      </condition>
      <set name="propulsion/starter_cmd" value="1"/>
      <set name="fcs/right-brake-cmd-norm" value="0"/>
      <set name="fcs/left-brake-cmd-norm" value="0"/>
    </event>

    <event name="Start engine: Cutoff off">
      <description>Cutoff is turned off, allowing fuel to flow.</description>
    <!--      <condition>propulsion/engine[0]/n2 ge 15.0</condition> -->
      <condition>sim-time-sec ge 2.0</condition>
      <set name="propulsion/cutoff_cmd" value="0"/>
      <set name="propulsion/starter_cmd" value="1"/>
      <set name="fcs/throttle-cmd-norm[0]" value="0.4"/>
      <set name="fcs/throttle-cmd-norm[1]" value="0.4"/>
      <notify>
        <property>propulsion/engine[0]/n2</property>
        <property>propulsion/engine[1]/n2</property>
        <property>propulsion/starter_cmd</property>
        <property>propulsion/engine[0]/thrust</property>
        <property>propulsion/engine[1]/thrust</property>
      </notify>
    </event>

    <event name="Accelerate engine">
      <description>Accelerate engine.</description>
      <condition> sim-time-sec >= 15 </condition>
```

```

    <set name="fcs/throttle-cmd-norm[0]" value="1.0"/>
    <set name="fcs/throttle-cmd-norm[1]" value="1.0"/>
    <notify>
      <property>propulsion/engine[0]/n2</property>
      <property>propulsion/engine[1]/n2</property>
      <property>propulsion/starter_cmd</property>
      <property>propulsion/engine[0]/thrust</property>
      <property>propulsion/engine[1]/thrust</property>
    </notify>
  </event>

<event name="Dive the airplane">
  <description>Dive the airplane 1</description>
  <notify/>
  <condition>
    sim-time-sec >= 100
  </condition>
  <set name="fcs/elevator-cmd-norm" value="0.04" action="FG_RAMP" tc ="0.6"/>
</event>
<!--
<event name="xxx the airplane">
  <description>xxx the airplane 1</description>
  <notify/>
  <condition>
    sim-time-sec >= 18
  </condition>
  <set name="fcs/elevator-cmd-norm" value="0.0" action="FG_RAMP" tc ="0.6"/>
</event>
-->

<event name="push the airplane 2">
  <description>push the airplane 2</description>
  <notify/>
  <condition>
    sim-time-sec >= 115
  </condition>
  <set name="fcs/elevator-cmd-norm" value="-0.5" action="FG_RAMP" tc ="0.6"/>
</event>

<event name="stb the airplane ">
  <description>push the airplane 3</description>
  <notify/>
  <condition>
    sim-time-sec >= 140
  </condition>
  <set name="fcs/elevator-cmd-norm" value="-0.05" action="FG_RAMP" tc ="0.6"/>
</event>

<event name="Time Notify" type="FG_DELTA" persistent="true">
  <description>Output message at 10 second intervals</description>

```

```

    <notify>
      <property>velocities/vc-kts</property>
      <property>position/h-agl-ft</property>
      <property>propulsion/engine[0]/n2</property>
      <property>propulsion/engine[1]/n2</property>
      <property>propulsion/engine[0]/thrust</property>
      <property>propulsion/engine[1]/thrust</property>
    </notify>
    <condition> sim-time-sec >= simulation/notify-time-trigger </condition>
    <set name="simulation/notify-time-trigger" value="10" type="FG_DELTA"/>
  </event>

</run>

</runscript>

```

C.4. Simulación con el UAV

A continuación se adjunta la simulación completa realizada con el UAV:

```

<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="http://jsbsim.sourceforge.net/JSBSimScript.xsl"?>
<runscript xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://jsbsim.sf.net/JSBSimScript.xsd"
  name="C172-01A takeoff run">

  <use aircraft="1-uav" initialize="reset00"/>
  <run start="0.0" end="1000" dt="0.0083333">

    <property> simulation/notify-time-trigger </property>

    <event name="engine start">
      <description>Start the engine</description>
      <notify/>
      <condition>
        sim-time-sec >= 0.25
      </condition>
      <set name="fcs/throttle-cmd-norm" value="1.0" action="FG_RAMP" tc ="0.5"/>
      <set name="fcs/mixture-cmd-norm" value="0.87" action="FG_RAMP" tc ="0.5"/>
      <set name="propulsion/magneto_cmd" value="3"/>
      <set name="propulsion/starter_cmd" value="1"/>
    </event>

    <event name="Set heading hold">
      <description>Set Heading when 5 ft AGL is reached</description>
      <notify/>

```

```
<condition>
  position/h-agl-ft >= 0.01
</condition>
<set name="ap/heading_setpoint" value="200"/>
<set name="ap/attitude_hold" value="0"/>
<set name="ap/heading_hold" value="1"/>
</event>

<event name="Set autopilot for 20 ft.">
  <description>Set Autopilot for 20 ft</description>
  <notify/>
  <condition>
    aero/qbar-psf >= 4
  </condition>
  <set name="ap/altitude_setpoint" value="100.0"/>
  <set name="ap/altitude_hold" value="1"/>
  <set name="fcs/flap-cmd-norm" value=".33"/>
</event>

<event name="Set autopilot for 6000 ft.">
  <description>Set Autopilot for 6000 ft</description>
  <notify/>
  <condition>
    aero/qbar-psf > 5
  </condition>
  <set name="ap/altitude_setpoint" value="6000.0"/>
</event>

<event name="Adjust throttle/flaps">
  <description>Remove flaps at 50 ft</description>
  <notify/>
  <condition>
    position/h-agl-ft >= 100
  </condition>
  <set name="fcs/mixture-cmd-norm" value="0.69" action="FG_EXP" tc ="400.0"/>
  <set name="fcs/flap-cmd-norm" value="0"/>
</event>

<event name="Set autopilot for 4,000 ft.">
  <description>Set Autopilot for 4000 ft</description>
  <notify/>
  <condition>
    sim-time-sec >= 1500
  </condition>
  <set name="ap/altitude_setpoint" value="4000.0"/>
</event>

<event name="Set autopilot for 7000 ft.">
  <description>Set Autopilot for 7000 ft</description>
  <notify/>
  <condition>
```

```
        sim-time-sec >= 2500
    </condition>
    <set name="ap/altitude_setpoint" value="7000.0"/>
</event>

<event name="Time Notify" type="FG_DELTA" persistent="true">
    <description>Output message at 100 second intervals</description>
    <notify>
        <property>velocities/vc-kts</property>
        <property>position/h-agl-ft</property>
    </notify>
    <condition> sim-time-sec >= simulation/notify-time-trigger </condition>
    <set name="simulation/notify-time-trigger" value="100" type="FG_DELTA"/>
</event>

</run>
</runscript>
```