TREBALL DE FI DE CARRERA

TÍTOL DEL TFC: Validación de las alturas de un plan de vuelo

TITULACIÓ: Enginyeria Tècnica de Telecomunicació, especialitat Sistemes de Telecomunicació

AUTOR: Miguel Mestre Fernández

DIRECTOR: Jorge Ramírez Alcántara

DATA: 15 de juny de 2007

Título: Validación de las alturas de un plan de vuelo

Autor: Miguel Mestre Fernández

Director: Jorge Ramírez Alcántara

Fecha: 15 de junio de 2007

Resumen

El objetivo de este trabajo consiste en verificar un plan de vuelo de un UAV (*Unmanned Aerial Vehicle*) contra posibles colisiones que se puedan producir con el nivel del terreno y teniendo en cuenta que no puede entrar en el espacio aéreo controlado.

Para poder verificarlo se analizan los diferentes requisitos propuestos por el usuario, e.g: detectar posibles conflictos del plan de vuelo propuesto, la presencia de márgenes de seguridad, el formato de los datos de la altura del cartográfico, etc. También, se informará al usuario de la altura mínima obtenida en un *flight plan* completo, ya que será la altura más crítica que adquirirá el avión en todo su recorrido.

El avión tendrá especificado un *flight plan* en el cual podrá realizar dos tipos de trayectorias diferentes (LEGs). La primera consiste en una línea recta (*Track-to-Fix*) y la segunda en un arco de circunferencia (*Radius-to-Fix*).

Los datos de entrada de la aplicación, que verificará el plan de vuelo, estarán compuestos por un conjunto de matrices de alturas (cada matriz corresponde a la elevación del terreno por el que pasa un LEG propuesto) y los datos de cada LEG. En ambos tipos de LEGs será necesario conocer la posición inicial y final, así como la altura en dichas posiciones, y en el caso del *Radius-to-Fix*, además de los datos anteriores, se necesitará el centro y el sentido de giro (*clockwise* o *anticlockwise*).

La implementación de las trayectorias anteriores se ha realizado mediante dos algoritmos que identifican las posiciones por las que volará el avión y así comprobar el principal objetivo, es decir, que no se produzca una colisión.

La aplicación analizará cada uno de los LEGs y comprobará si es seguro (no hay colisión y no entra en espacio aéreo controlado). Si el *flight plan* es correcto se informará de la altura mínima obtenida por el avión en todo el trayecto. Si no es correcto se informará que el plan de vuelo es erróneo.

Title: Validation of the heights of a flight plan

Author: Miguel Mestre Fernández

Director: Jorge Ramírez Alcántara

Date: June, 15th 2007

-

Overview

The main objective of this project is to verify the flight plan of an UAV (Unmanned Aerial Vehicle) against possible collisions that can be produced with the level of the ground and taking into account that it can not enter into the air traffic controlled.

To verify the above mentioned, we have to analyse the different requirements; e.g: to detect conflicts in the flight plan, the security distance, the format of the different notes of height, etc. We will also inform the user of the minimum high of a complete flight plan because that one is going to be the most critical for the aircraft during all the route.

The aircraft will have an specific flight plan that can do two types of trajectory (LEGs). The first one consists in a straight line (Track-to-Fix). The second one consists in a circumference arc (Radius-to-Fix).

The introductory notes of the application that will verify the flight plan will be composed by a group of altitude matrixes (every matrix goes with the elevation of the ground where there is a LEG). Both types of LEGs will be necessary to know the initial and final position, as well as, the height in case of Radius-to-Fix. We will also need the centre and the clockwise.

The above mentioned trajectories have been realised by two logarithms that identify the different positions where the aircraft will fly and verify the main aim, to avoid a collision.

The application will analyse every LEGs and will also verify if is possible (not collision and not enter into air traffic controlled). If the flight plan is correct is necessary to inform about the minimum high of the aircraft during the trajectory. If is not correct is necessary to inform that is incorrect.

ÍNDICE

INTRODUCCIÓN

Este trabajo está pensado para solucionar problemas del mundo de la aeronáutica.

Concretamente, está introducido dentro del grupo de investigación ICARUS, que se centra en el estudio de los vehículos aéreos no tripulados para varias misiones civiles, e.g: proporcionar información visual sobre el foco de un incendio.

La aplicación que se desarrollará será utilizada por un UAV (*Unmanned Aerial Vehicle*), el cual es un vehículo aéreo no tripulado donde la CPU que lleva a bordo se encarga de pilotar sin que sea necesario disponer de un humano a bordo. En nuestro caso, su sistema de posicionamiento es el GPS.
Esta aplicación será programada en lenguaje de programación *C* para un sistema operativo Linux.

Iniciaremos el proyecto con una fase de captura y análisis de requisitos del usuario para asegurarnos que la aplicación satisfaga sus necesidades iniciales. El objetivo es analizar un *flight plan* (plan de vuelo) para detectar posibles colisiones con el terreno y comprobar que no vuela por espacio aéreo controlado. Para ello, se dispondrá de los datos cartográficos del terreno y de la trayectoria propuesta para el avión. Con estos datos se podrá comprobar si el plan de vuelo es seguro o no.

Respecto a la organización del documento, éste está formado por una primera parte donde se especifican los requisitos del usuario para la verificación del plan de vuelo.
En el siguiente apartado del documento, se explicarán las diferentes entradas de las que consta la aplicación (*Digital Elevation Model* y *flight plan*) así como las características de cada una de ellas.
A continuación, se hará referencia a la algorítmica de la aplicación, es decir, analizar y programar las posibles trayectorias del UAV, que en este caso serán dos (recta y arco de circunferencia) y los diferentes algoritmos necesarios para la elaboración de la aplicación (márgenes de seguridad, altura mínima y relación entre coordenadas).
En el capítulo siguiente, se incluirán una serie de pruebas para verificar los requisitos del usuario.
También, como aportación suplementaria, se diseñará la posibilidad de obtener un plan de vuelo alternativo.
Por otro lado, se explicará la planificación llevada a cabo y los posibles costes.
Por último, se analizará el trabajo realizado mediante las conclusiones finales, donde se hará una referencia al posible impacto medioambiental que podría producir la implementación del proyecto.

CAPÍTULO 1. REQUISITOS DEL USUARIO

La aplicación que vamos a desarrollar consiste en verificar un *flight plan* (plan de vuelo), contra posibles colisiones provocadas por la orografía del terreno.

REQ-USU 01: Detectar posibles conflictos del plan de vuelo propuesto.
- *Rationale*: Para verificar si el plan de vuelo es seguro.
- *Additional Info*: Será seguro si no hay colisiones con la elevación del terreno.

Para poder verificar el plan de vuelo, dispondremos de las características del terreno (curvas de nivel) que serán dadas por un Modelo Digital de Elevación denominado MDE o DEM (Digital Elevation Model), el cual proporcionará la altura del terreno en intervalos de 30 m x 30 m.

REQ-USU 02: El modelo de elevación del terreno usa el formato del ICC (*Institut Català de Cartografia*).
- *Rationale*: Es el formato con mejor resolución disponible para Cataluña.
- *Additional Info*: Resolución 30 x 30 metros.

El plan de vuelo estará formado por un conjunto de tramos, denominados LEGs, los cuales unen dos *waypoints* (coordenadas geográficas). El primer tipo de LEG será una línea recta llamada *Track-to-Fix* (*TF*) y el segundo una curva (arco de circunferencia) denominada *Radius-to-Fix* (*RF*). Estos nombres se encuentran referenciados por el estándar 283A de RNAV (ver [1]).

REQ-USU 03: El formato del plan de vuelo se expresará en XML.
- Rationale: Para el intercambio de información con otras aplicaciones.
- Additional Info: La aplicación ha de funcionar cuando se le proporciona más de un LEG.

Los datos necesarios que nos proporcionará el usuario para la realización del *TF* serán la posición inicial y final del avión (geográficas), así como su altura en dichas posiciones. En cambio, en el movimiento curvilíneo (*RF*) necesitaremos además el centro de la circunferencia y el sentido de giro (*clockwise* o *anticlockwise*).

REQ-USU 04: Los *waypoints* estarán expresados en coordenadas geográficas (latitud, longitud).
- *Rationale*: Tipo de coordenadas utilizadas por el UAV.
- *Additional Info*: Se proporcionará la altura del avión por cada *waypoint*.

REQ-USU 05: El avión ha de ser capaz de trazar una línea recta (*TF*).
- *Rationale*: Mínimo recorrido entre dos puntos.
- *Additional Info*: Los datos necesarios serán la posición inicial y final, y la altura en dichas posiciones.

REQ-USU 06: El avión ha de ser capaz de trazar un arco de circunferencia (*RF*).

- *Rationale*: Para poder cambiar de trayectoria de forma progresiva.
- *Additional Info*: Los datos necesarios serán la posición inicial y final, la altura en dichas posiciones, el centro de la circunferencia y el sentido de giro.

Para incrementar la seguridad, el usuario podrá introducir dos tipos de márgenes: lateral y vertical.

Por un lado, el margen lateral corresponde a las posibles desviaciones, respecto al plano horizontal, que pueda sufrir el avión.

REQ-USU 07: El usuario podrá utilizar un margen de seguridad lateral configurable.

- Rationale: Proporcionar mayor seguridad sobre posibles obstáculos.
- Additional Info: Se expresará en metros.

Por otro lado, disponemos de un margen vertical el cual puede ser de dos tipos: el inferior y el superior. El margen inferior servirá para que el avión, en ningún caso, pueda volar demasiado cerca de un obstáculo con altura similar. De forma análoga, la función del margen superior será establecer una distancia de seguridad respecto a los límites establecidos por la legislación española del espacio aéreo libre (altura máxima = 1000 pies = 304.8 m). Dicha altura está referenciada respecto a la orografía del terreno.

La altura de los *waypoints* inicial y final será indicada respecto al nivel del mar, en cambio el margen vertical inferior estará referenciado respecto a la altura del avión en cada momento. Por último, el margen superior será indicado respecto a la altura máxima que podrá alcanzar el avión en cada momento.

REQ-USU 08: El UAV no sobrepasará la altura de 1000 pies.

- Rationale: Para evitar entrar en espacio aéreo controlado.
- Additional Info: Altura definida sobre el nivel del terreno (DEM).

REQ-USU 09: El usuario podrá utilizar un margen de seguridad vertical inferior configurable.

- Rationale: Proporcionar mayor seguridad sobre posibles obstáculos.
- Aditional Info: Se expresará en metros.

REQ-USU 10: El usuario podrá utilizar un margen de seguridad vertical superior configurable.

- Rationale: Para añadir un margen de seguridad que evite entrar en espacio aéreo controlado.
- Aditional Info: Se expresará en metros.

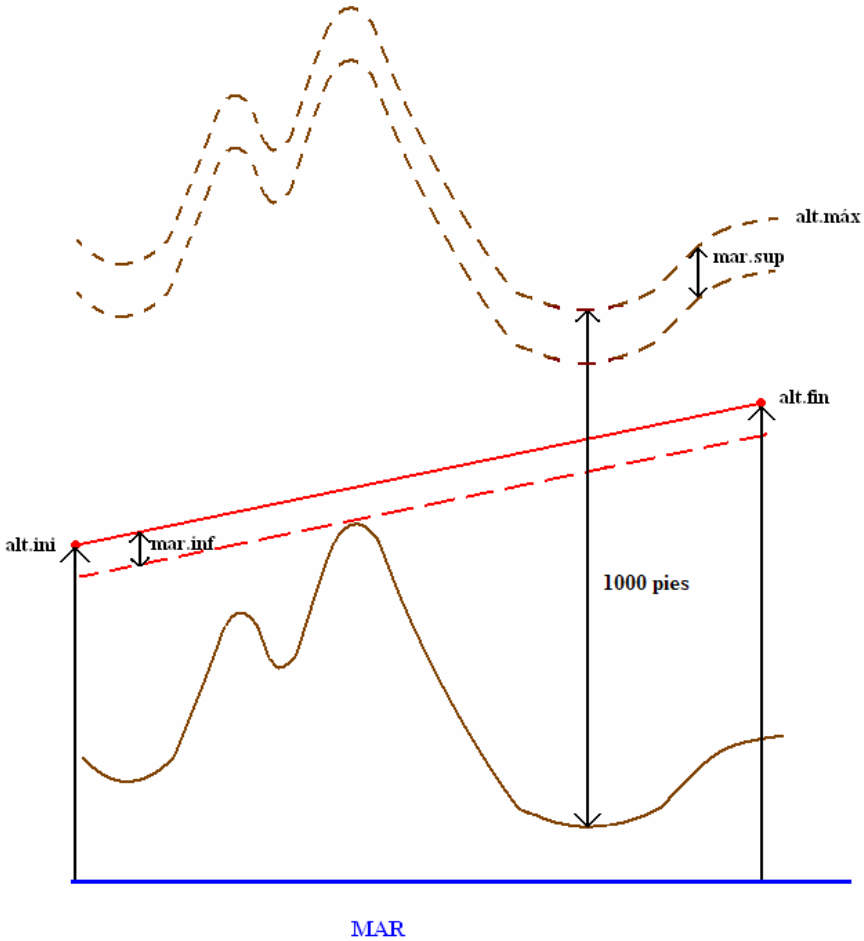


Fig. 1.1 Representación de los márgenes de seguridad y altura máxima.

REQ-USU 11: La resolución de las coordenadas será inferior o igual a un metro.

- Rationale: Resolución válida para nuestro plan de vuelo.
- Additional Info: Aplica tanto a las geográficas como a las coordenadas UTM.

REQ-USU 12: La salida de la aplicación informará de la altura mínima que el avión ha adquirido en un *flight plan*.

- Rationale: Para identificar el punto crítico.
- Additional Info: Esta altura es la menor diferencia entre la altura del avión en un instante y el nivel del terreno en dicha posición.

REQ-USU 13: La aplicación debe estar programada en lenguaje de programación *C*.

- Rationale: Para facilitar eventuales certificaciones.
- Additional Info: La aplicación será utilizada en el sistema operativo Linux.

CAPÍTULO 2. ENTRADAS DEL PROGRAMA

2.1. Digital Elevation Model (DEM)

Para satisfacer el REQ-USU 02 utilizaremos el Modelo Digital de Elevación proporcionado por el ICC.

2.1.1. Modelo Digital del Terreno

Se denomina modelo digital del terreno a una estructura numérica de datos que representa la distribución espacial de una variable cuantitativa, como puede ser la temperatura, la cota, la humedad o la presión atmosférica. En particular, cuando la variable a representar es la cota o altura del terreno se denomina Modelo Digital de Elevaciones (MDE) o DEM en sus siglas en inglés.

Los modelos digitales del terreno, también denominados MDT, son simbólicos pues establecen relaciones de correspondencia con el objeto real mediante algoritmos o formalismos matemáticos que son tratados mediante programas informáticos.

2.1.1.1. Características básicas

- Los MDT han de ser estructuras de datos, no son sólo acumulaciones de cifras, sino que deben tener una estructura interna con la cual deben interpretarse dichos datos.
- Los MDT representan distribuciones espaciales de variables, lo que acota su uso a fenómenos geográficos. Son muy usados en ciencias como cartografía y SIG (Sistemas de Información Geográficos).
- La variable a representar ha de ser cuantitativa.

El Modelo Digital de Elevación (Digital Elevation Model) es una estructura numérica de datos que representa la distribución espacial de la altitud de la superficie del terreno. Por lo tanto, es lo mismo que decir un MDT de altitudes, ya que tiene los mismos modelos de datos.

De forma general, la unidad básica de un MDE es un punto acotado, definido como un valor de altitud *z*, al que acompañan los valores correspondientes de *x* e *y*.

2.1.2. Modelo Cartográfico de Cataluña

Como se ha comentado anteriormente, nuestro Modelo Digital de Elevación será proporcionado por el *Institut Català de Cartografia* (ver [4]).

Este modelo hace referencia a los datos del nivel del terreno de Cataluña en intervalos de 30 m x 30 m, es decir, cada 900 m^2. Estos datos están referenciados respecto al nivel medio del mar de Alicante.

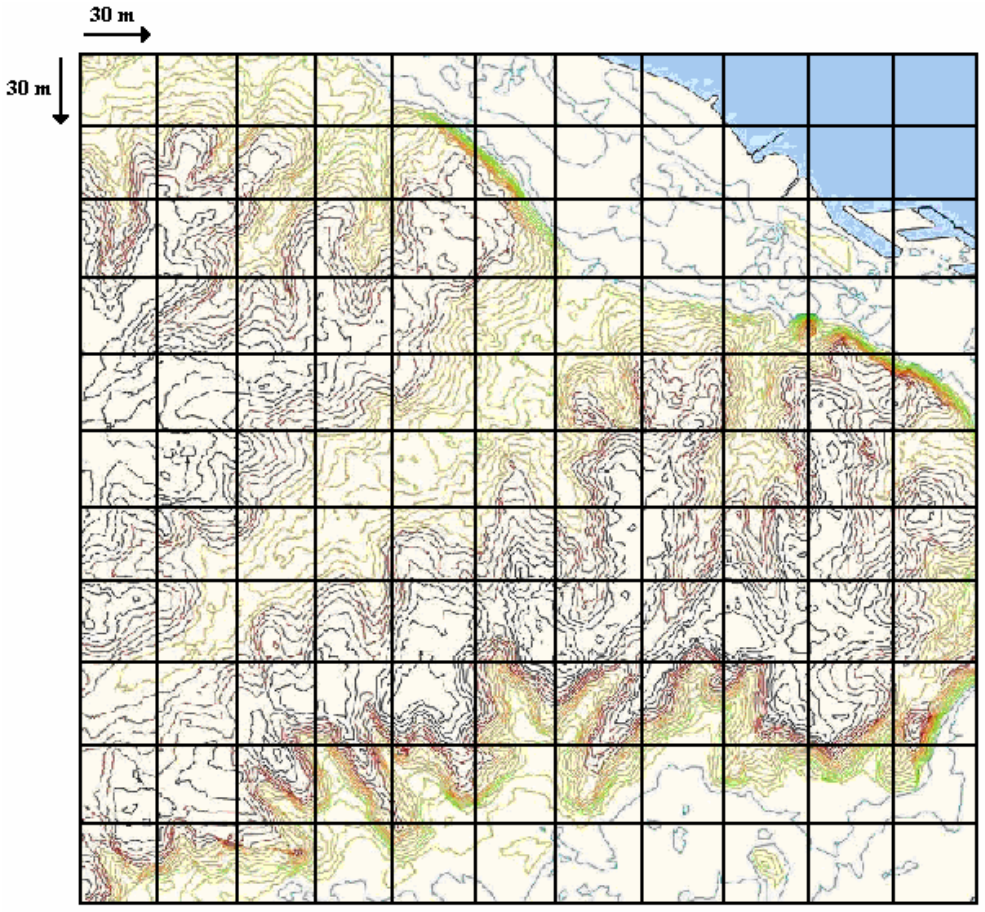


Fig. 2.1 Ejemplo matriz de alturas

2.2. FLIGHT PLAN

El plan de vuelo propuesto por el usuario estará formado por un conjunto de LEGs (rectas o arcos de circunferencia), enlazados entre ellos, es decir, el *waypoint* (coordenada geográfica) final de un LEG ha de coincidir con el inicial del siguiente *waypoint*, y así sucesivamente.

Respecto al formato del fichero de entrada, cabe señalar que el REQ-USU 03 ha sido modificado. (Justificación en Capítulo 6)

Anterior:

REQ-USU 03: El formato del plan de vuelo se expresará en XML.
- Rationale: Para el intercambio de información con otras aplicaciones.
- Additional Info: La aplicación ha de funcionar cuando se le proporciona más de un LEG.

Actual:

REQ-USU 03: El formato del plan de vuelo se expresará en un fichero de texto.
- Rationale: Para poder realizar pruebas de la aplicación desarrollada.
- Additional Info: La aplicación ha de funcionar cuando se le proporciona más de un LEG.

Por tanto, el formato del *flight plan* será un fichero de texto, donde se especificará primero el tipo de LEG utilizado y después las características asociadas. Por un lado, en el caso del *Track-to-Fix* (justificación REQ-USU 05) los datos necesarios serán los *waypoints* inicial y final, y la altura del avión en cada uno. Por otro lado, el *Radius-to-Fix* (justificación REQ-USU 06) además de los datos anteriores incluirá el centro de la circunferencia y el sentido de giro (*clockwise o anticlockwise*).
Por cada LEG del plan de vuelo, nuestra aplicación proporcionará la matriz de alturas (DEM) correspondiente al LEG dado.

Los *waypoints* se expresarán en el fichero como coordenadas geográficas: latitud, longitud- (justificación REQ-USU 04).

2.2.1. Ejemplo formato flight plan en fichero de texto

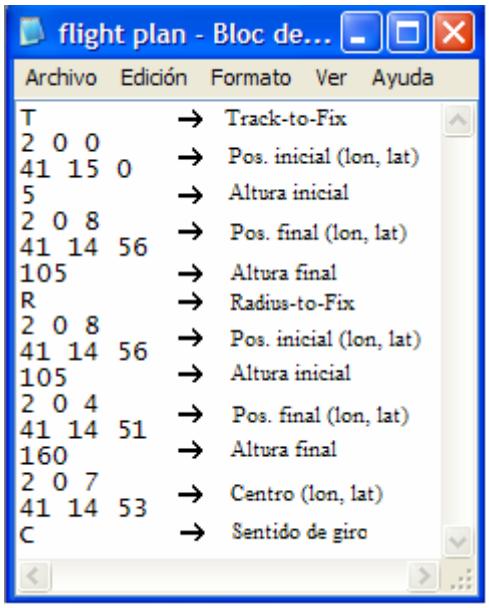


Fig. 2.2 Formato *flight plan*

2.2.2. Ejemplo flight plan representado en un modelo cartográfico

En el siguiente ejemplo se pueden visualizar dos trayectorias diferentes, es decir, dos tipos de LEGs.

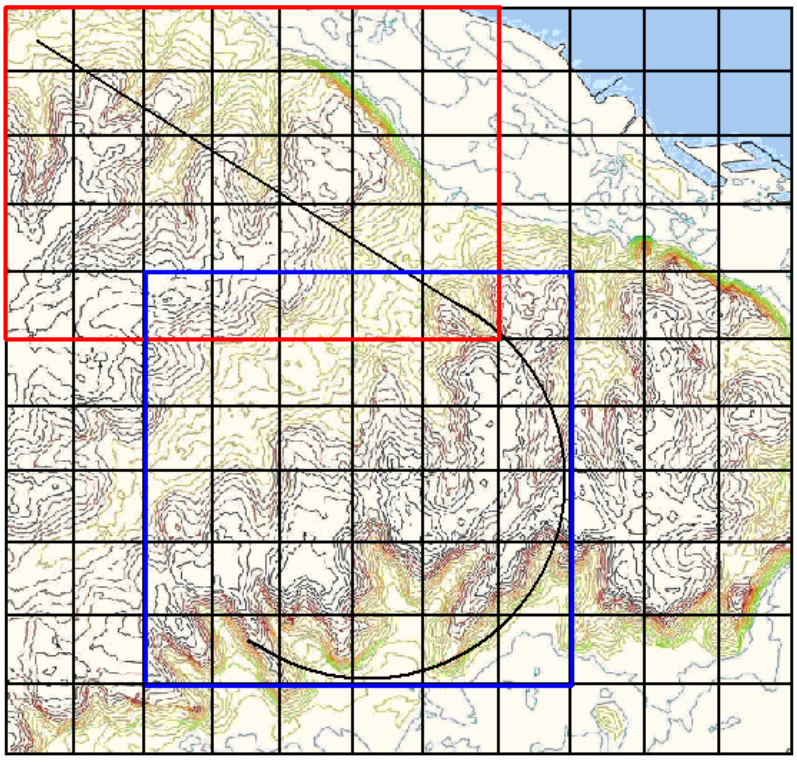


Fig. 2.3 Ejemplo *flight plan* en modelo cartográfico

El primero de ellos (rectángulo rojo) corresponde a un *Track-to-Fix* y el segundo (cuadrado azul) a un *Radius-to-Fix*.

A partir de un *flight plan* en fichero de texto, la aplicación calculará para cada LEG la matriz de alturas que le corresponde. Por un lado, en el primer LEG (rectángulo rojo) la matriz está comprendida entre la posición (0,0) y la (4,6). Por otro lado, en el cuadrado azul, que corresponde al segundo LEG, la matriz empezaría en la (4,2) y acabaría en la (9,7).

Para cada LEG se analizarían las casillas de la matriz de alturas por las que pasaría el avión y se comprobaría si se produce una colisión con la orografía del terreno.

2.3. COORDENADAS GPS Y UTM

El objetivo del proyecto (REQ-USU 01) es comprobar que el plan de vuelo del avión sea seguro, pero para ello necesitamos conocer la posición exacta del avión dentro del planeta Tierra.
Tal y como se explica en el apartado 2.2., para expresar estas posiciones utilizaremos las coordenadas geográficas (justificación REQ-USU 04), las cuales se pasarán a cartesianas y éstas a coordenadas de nuestro DEM.
Las coordenadas geográficas y cartesianas se analizan mediante las tecnologías GPS y UTM respectivamente, que se explican a continuación:

2.3.1. GPS

Es un Sistema Global de Navegación por Satélite (GNSS) el cual permite determinar en todo el mundo la posición de un objeto, una persona, un vehículo o una nave, con una precisión hasta de centímetros, aunque lo habitual son unos pocos metros. Utiliza el sistema de coordenadas geográficas, es decir, se expresan mediante latitud y longitud.

2.3.1.1. Elementos que lo componen

1. *Sistema de satélites*: está formado por 24 unidades (21 operativos y 3 de respaldo) con trayectorias sincronizadas para cubrir toda la superficie del globo terráqueo. Más concretamente, repartidos en 6 planos orbitales de 4 satélites cada uno. La energía eléctrica que requieren para su funcionamiento la adquieren a partir de dos paneles compuestos de celdas solares adosadas a sus costados.
2. *Estaciones terrestres*: envían información de control a los satélites para controlar las órbitas y realizar el mantenimiento de toda la constelación.
3. *Terminales receptores*: indican la posición en la que estamos, conocidas también como Unidades GPS. Se pueden adquirir en tiendas especializadas.

El GPS funciona mediante una red de 24 satélites en órbita sobre el globo a 20.200 km. Cuando se desea determinar la posición, el terminal receptor localiza automáticamente como mínimo tres satélites de la red, de los que recibe unas señales indicando la posición y el reloj de cada uno de ellos. En base a estas señales, el aparato sincroniza el reloj del GPS y calcula el retraso de las señales, es decir, la distancia al satélite. Por "triangulación" calcula la posición en que se encuentra el terminal. La triangulación en el caso del GPS, a diferencia del caso 2-D que consiste en averiguar el ángulo respecto de puntos conocidos, se basa en determinar la distancia de cada satélite respecto al punto de medición. Conocidas las distancias, se determina fácilmente la propia posición relativa respecto a los tres satélites. Conociendo además las coordenadas o posición de cada uno de ellos por la señal que emiten, se obtiene la posición absoluta o las coordenadas reales del punto de medición.

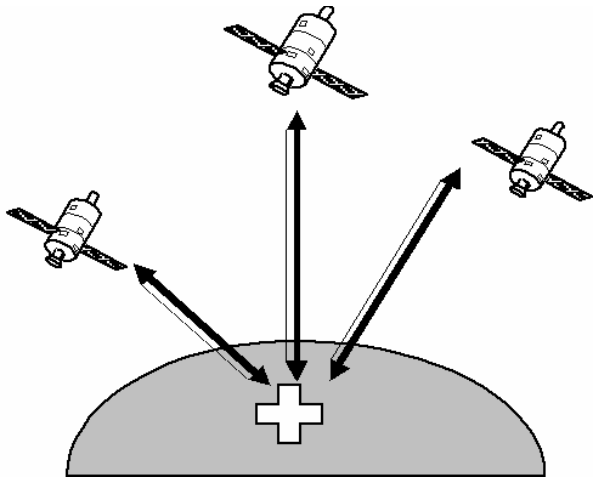


Fig. 2.4 Triangulación GPS

2.3.2. UTM

El Sistema de Coordenadas Universal Transversal de Mercator es un sistema de coordenadas basado en la proyección geográfica transversa de Mercator, que se construye como la proyección de Mercator normal tangente a un meridiano. A diferencia del sistema de coordenadas geográficas, expresadas en longitud y latitud, las magnitudes en el sistema UTM se expresan en unidades de distancia.

2.3.2.1. Husos UTM

Se divide la Tierra en 60 husos de 6º de longitud, la zona de proyección de la UTM se define entre los paralelos 80º S y 84 º N. Cada Huso se numera con un número entre el 1 y el 60, estando el primer huso limitado entre las longitudes 180° y 174° W y centrado en el meridiano 177º W. Ca da huso tiene asignado un meridiano central, que es donde se sitúa el origen de coordenadas, junto con el ecuador. Los husos se numeran en orden ascendente hacia el este. Por ejemplo, la Península Ibérica está situada en los Husos 31 al 29.

2.3.2.2. Zonas UTM

Se divide la Tierra en 16 zonas de 8º Grados de Latitud, que se denominan con letras desde la C hasta la X excluyendo las letras "I" y "O", por su parecido con los números uno y cero, respectivamente. Las zonas polares no están consideradas en este sistema de referencia. Si una zona tiene una letra igual o mayor que la N, la zona está en el hemisferio norte, mientras que está en el sur si su letra es menor que la "N".

2.3.2.3. Notación

Cada cuadrícula UTM se define mediante el número del Huso y la letra de la Zona, por ejemplo Cataluña estaría en la cuadrícula 31T.

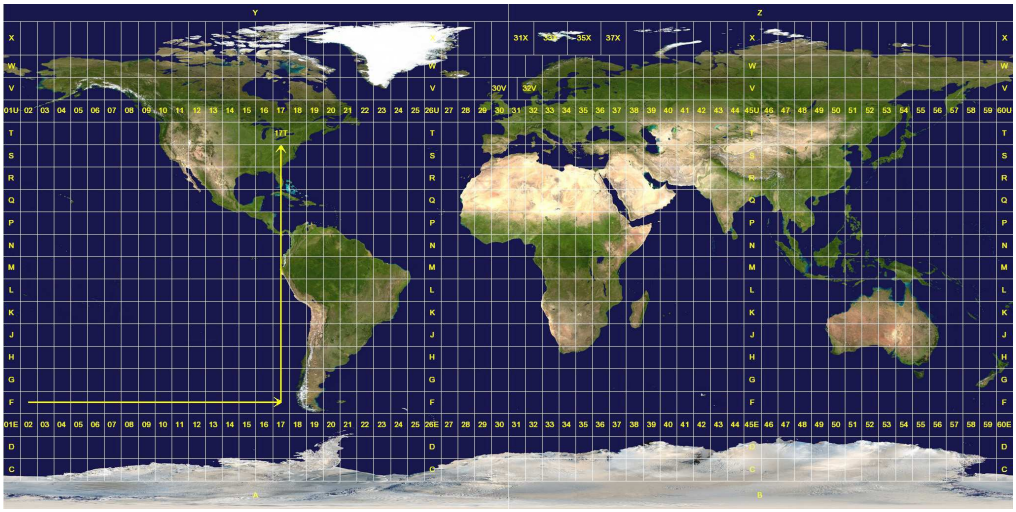


Fig. 2.5 Husos y zonas UTM.
Fuente: wikipedia

2.3.2.4. Resolución UTM

Una coordenada UTM siempre corresponde a un área cuadrada cuyo lado depende del grado de resolución de la coordenada. Cualquier punto comprendido dentro de este cuadrado (a esa resolución en particular) tiene el mismo valor de coordenada UTM. El valor de referencia definido por la coordenada UTM no está localizado en el centro del cuadrado, sino en la esquina inferior izquierda de dicho cuadrado.

La forma de lectura de una zona UTM siempre es de izquierda a derecha (Este), y de arriba a abajo (Norte). Es decir, el valor del Este corresponde a la distancia hacia el Este desde la esquina inferior izquierda de la cuadrícula UTM y el valor del Norte siempre es la distancia hacia el norte del Ecuador (en el hemisferio norte).

Por una parte, una zona UTM siempre comprende una región cuya distancia horizontal al Este es siempre inferior a 1.000.000 metros (de hecho, la "anchura" máxima de una zona UTM tiene lugar en el ecuador y corresponde aproximadamente a 668.000 m). Por otra parte, para cada hemisferio, una zona UTM siempre comprende una región cuya distancia vertical es inferior a 10.000.000 metros (realmente algo más de 9.329.000 metros en la latitud 84° N).

Por definición, el valor del Este del punto central (que coincide con el meridiano central) de la retícula UTM es siempre de 500.000 m. Cualquier punto a la izquierda de éste meridiano central tendrá un valor inferior a 500.000 m, así

mismo, cualquier punto situado a la derecha del meridiano central tendrá un valor superior a 500.000 m.

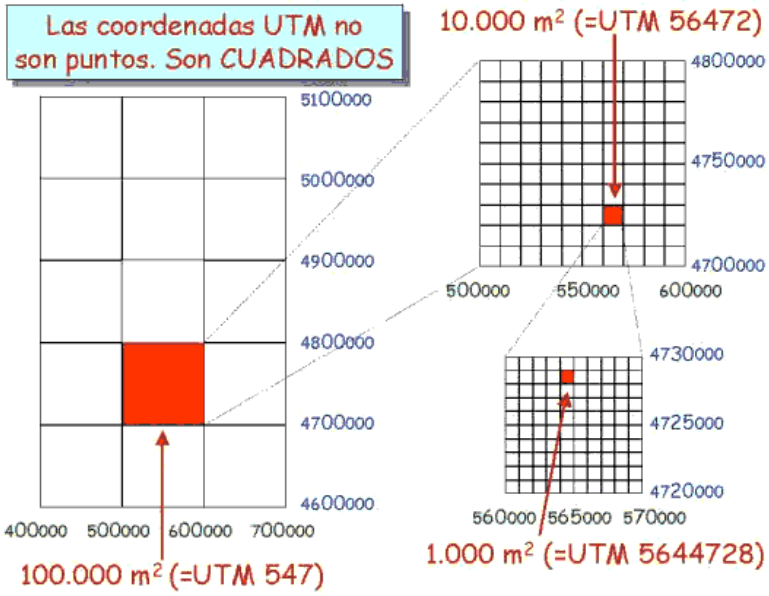


Fig. 2.6 Coordenadas UTM

En la siguiente tabla se describe la misma coordenada UTM con diferentes resoluciones, que oscilan desde áreas cuadradas que sólo tienen 1 metro de lado hasta aquella que tiene 100.000 metros. No hay límite de resolución en una coordenada UTM, ya que se pueden definir áreas cuyos lados sean centímetros, milímetros, etc.

COORDENADAS UTM: LA RESOLUCIÓN DETERMINA EL NÚMERO DE DÍGITOS.

Coordenadas UTM	Zona y banda	Metros al Este	Metros al Norte	Resolución
30S 3546784891567	30 S	354678	4891567	1 metro
30S 35467489156	30 S	354670	4891560	10 m
30S 354648915	30 S	354600	4891500	100 m
30S 3544891	30 S	354000	4891000	1000 m
30S 35489	30 S	350000	4890000	10.000 m
30S 348	30 S	300000	4800000	100.000 m

Fig. 2.7 Resolución coordenadas UTM

2.3.2.5. Ejemplo de una zona específica: Cataluña.

Como este proyecto se centrará básicamente en Cataluña, aquí se proporciona la conversión de coordenadas geográficas a UTM y viceversa, para el huso 31T.

Coordenadas de una zona de Cataluña en GPS y UTM

Fig. 2.8 Coordenadas de Cataluña

2.3.3. Resolución de las coordenadas

Una vez conocido qué representa cada sistema de coordenadas, la pregunta que se nos plantea es la siguiente: ¿con qué resolución hemos de determinar cada tipo de coordenada para poder tener una resolución inferior o igual a un metro (suficiente para nuestro plan de vuelo)?. Con la siguiente demostración justificamos el REQ-USU 11.

Para poder averiguar que resolución necesitamos, haremos una estimación teórica teniendo en cuenta que el peor de los casos se produciría en el Ecuador. Calcularemos cuántos metros representan un segundo, para así poder deducir que resolución necesitamos para las coordenadas GPS.

Radio de la Tierra (en el Ecuador) = $6.378.000\ \mathrm{m}$.

Perímetro en el Ecuador =
= $2\pi \cdot R = 2\pi \cdot 6378\ \mathrm{km} = 40.074{,}15589\ \mathrm{km} = 40.074.155{,}89\ \mathrm{m}$

Circunferencia = $360° = 1.296.000"$

$$resolución = \frac{40.074.155{,}89\ \mathrm{m}}{1.296.000"} = 30{,}9214\, m/seg. \qquad \textbf{(2.1)}$$

A partir de este resultado, podemos deducir que si la resolución de las coordenadas GPS fueran de centésimas de segundo ($0{,}3092\, m/cent.seg$), el error cometido sería inferior a un metro, por tanto, cumple las expectativas propuestas inicialmente.

Como conclusión, para poder obtener una resolución inferior o igual a un metro, necesitamos una resolución en el caso de GPS de centésimas de segundo. En cambio, en UTM sólo será necesario obtener 6 dígitos respecto al Este y 7 respecto al Norte, sin la necesidad de decimales.

2.3.3.1. Verificación de la resolución de las coordenadas

Para poder verificar el REQ-USU 11, pondremos como ejemplo las coordenadas de Cataluña (E: 2º, N: 41º 15', Huso: 31), y utilizaremos el software proporcionado por el *Institut Català de Cartografia* (ver [8]).

Variación de una centésima de segundo respecto a la Longitud

Fig. 2.9 Variación de la longitud

Podemos comprobar que respecto a los valores iniciales, longitud: 2º 0' 0'' y latitud 41º 15' 0'' (apartado "Coordenadas GPS y UTM", Fig. 2.8: Coordenadas de Cataluña), obtenemos una diferencia de menos de un metro ($416.216{,}7\ \mathrm{m} - 416.216{,}4\ \mathrm{m} = 0{,}3\ \mathrm{m}$) en el caso del Este porque sólo variamos la longitud en una centésima de segundo (2º 0' 0.01''), por lo tanto, se consigue el objetivo fijado.
Se obtendría el mismo resultado para la otra coordenada, si en este caso se variara la latitud.

CAPÍTULO 3. ALGORÍTMICA DE LA APLICACIÓN

La aplicación ha sido desarrollada en lenguaje de programación *C* para cumplir el REQ-USU 13. Dada la limitada extensión de este documento no he incluido el código fuente para conseguir una mayor claridad.

3.1. LEG 1: Track-to-Fix (Línea recta)

La implementación de este LEG justifica el REQ-USU 01 y el 05.

3.1.1. Algoritmo de la línea recta

Para poder representar las zonas por las que volará el avión, necesitamos conocer las casillas de nuestra matriz de elevación del terreno por las que pasará. Este objetivo se consigue mediante el algoritmo de la línea recta, es decir, entre los puntos de inicio y destino se traza una línea recta imaginaria la cual marcará las casillas por las que volará nuestro avión.

Fig. 3.1 Track-to-Fix RNAV

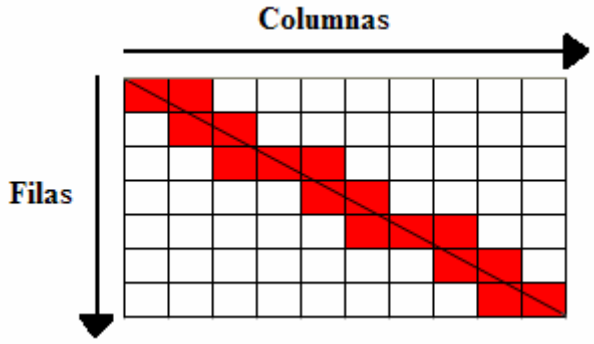


Fig. 3.2 Ejemplo del LEG: Track-to-Fix

Para realizar este algoritmo utilizaremos la parte de las matemáticas que corresponde a la geometría del plano.

Explicación del algoritmo:

Antes que nada, definimos la que será nuestra pendiente (*m*) de la recta imaginaria. Para conseguirla, simplemente hemos de dividir el número de filas entre el número de columnas.

El siguiente paso será realizar dos bucles, el primero recorrerá todas las columnas y el segundo, algunas de las filas.

A partir de aquí se expresa realmente la esencia de este algoritmo. Como se puede ver en la imagen anterior, para cada columna pueden corresponder varias filas (este hecho es irreal en geometría del plano por discretización de valores reales), para averiguarlo calcularemos para cada *x* el valor de *y* (*y=m·x*) que le corresponde. Si el valor de *y* de la fila anterior (*y=m·(x-1)*) es un número decimal, se cogerá la parte entera de dicho número (truncamiento) para empezar a recorrer las filas a partir de ese valor y no dejarnos ninguna casilla sin seleccionar por la que pasa la recta. Este recorrido se realizará hasta que el valor de *y* de dicha columna sea mayor que la variable que realice el bucle anterior.

3.1.1.1. Ejemplo: Track-to-Fix

Trayectoria: NO – SE

$$m = \frac{7}{10} = 0.7$$

Recta: $y = m \cdot x$

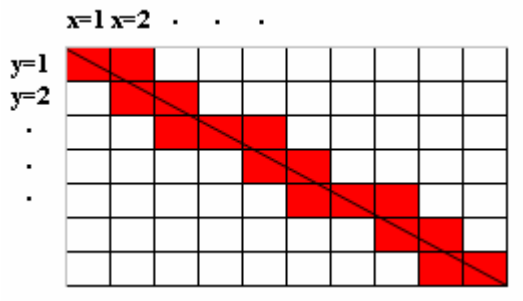


Fig. 3.3 Ejemplo del algoritmo de la línea recta

Bucle 1: Recorrido de columnas (*x*):

$$x = 1 \rightarrow y = 0.7$$

Bucle 2: Recorrido de filas

Definimos una variable nueva *k*.
Su valor inicial corresponderá a la parte entera de *y*, calculado con *x*-1.
Vamos incrementando *k* en una unidad hasta que su valor sea menor o igual que el de *y* del bucle anterior.

Valor inicial de k=0.

$$k = 0 \Rightarrow Casilla(0,0)$$
$$k = 1 >= y = 0.7$$

$$x = 2 \rightarrow y = 1.4$$

Valor inicial de k=0.

$$k = 0 \Rightarrow Casilla(0,1)$$
$$k = 1 \Rightarrow Casilla(1,1)$$
$$k = 2 >= y = 1.4$$

Este proceso se va repitiendo hasta que el valor de *x* sea igual al número de columnas.

3.1.2. Pendiente 3-D

Para poder justificar el REQ-USU 05 en su totalidad necesitaremos, como ya hemos comentado anteriormente, una altura inicial y otra final.

Con dichas alturas y el recorrido total que hará el avión (línea recta) podremos obtener la pendiente vertical del LEG a verificar. Al calcular esta pendiente suponemos que nuestro avión realizará un vuelo en progresión.

Para el cálculo de la distancia que separa la posición inicial y final utilizaremos Pitágoras, teniendo en cuenta que cada fila y columna tienen una separación en la vida real de 30 metros, y que para ir de una casilla a otra supondremos que estamos en el centro de ella, por tanto, habrá una fila y columna menos.

Por otra parte, para el cálculo de la pendiente volveremos a utilizar la geometría en el plano. En este caso se obtendrá de la división entre la diferencia entre alturas y la distancia total recorrida. En el caso de que la altura inicial sea mayor que la final, indicará que el avión desciende, por tanto, la pendiente será negativa.

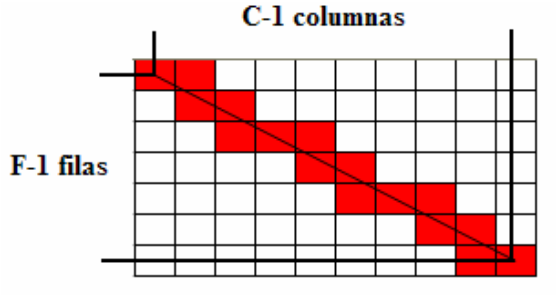


Fig. 3.4 Ejemplo cálculo distancia total recorrida

$$dist.total(m) = \sqrt{(F-1)^2 \cdot 30^2 + (C-1)^2 \cdot 30^2} \qquad \textbf{(3.1)}$$

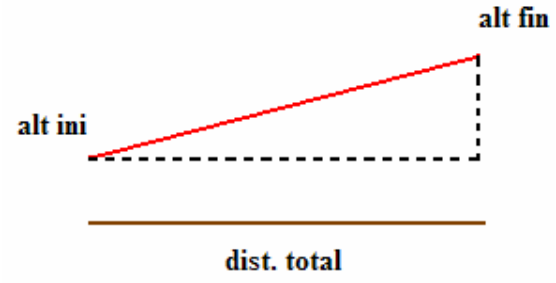


Fig. 3.5 Ejemplo cálculo pendiente vertical

$$m\_alt = \frac{alt\_fin - alt\_ini}{dist.total} \qquad \textbf{(3.2)}$$

Una vez obtenida esta pendiente la utilizaremos para calcular la altura en la que se encontrará el avión por cada casilla (i, j) por la que pase. Para hacer una aproximación lo más exhaustiva posible, consideraremos que las alturas de una misma columna no son iguales, sino que calcularemos la altura que le corresponde a cada casilla por separado, lo que proporcionará más realismo a la hora de comprobar si el plan de vuelo es o no es válido.

$$alt\_inst = alt\_ini + m\_alt \cdot \sqrt{(i-1)^2 \cdot 30^2 + (j-1)^2 \cdot 30^2} \qquad \textbf{(3.3)}$$

Cabe recordar que cuando se compruebe si en la posición por la que vuela el avión es correcta o no, se tendrá en cuenta tanto el margen lateral como los verticales.

3.2. LEG 2: Radius-to-Fix (Arco de circunferencia)

Este segundo tipo de LEG se ha implementado para justificar el REQ-USU 01 y el 06.

3.2.1. Algoritmo del arco de circunferencia

Otro tipo de movimiento que el avión puede realizar es una curva. Para poder representarla será necesario conocer el centro, y las posiciones inicial y final.

Para poder concretar dicho algoritmo definiremos una circunferencia, la cual dividiremos en cuatro partes iguales (cuadrantes), ya que cada una de estas regiones necesitará un algoritmo propio de comprobación de posiciones por las que pasará el avión. También, hay que tener en cuenta que el sentido de la trayectoria puede tener dos posibilidades: en el sentido de las agujas del reloj (*clockwise*) o en el opuesto (*anticlockwise*).

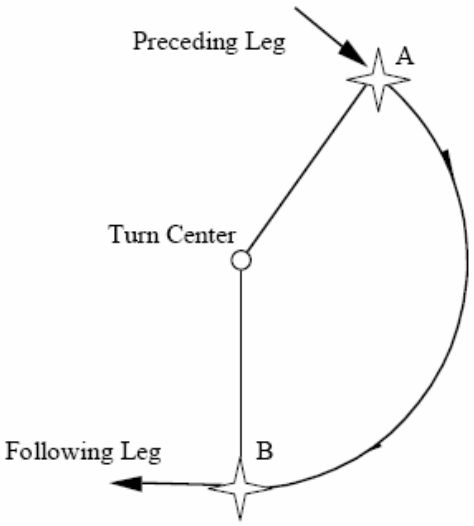


Fig. 3.6 Radius-to-Fix RNAV

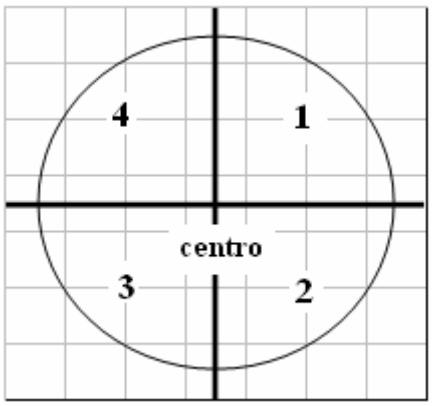


Fig. 3.7 División circunferencia en cuadrantes

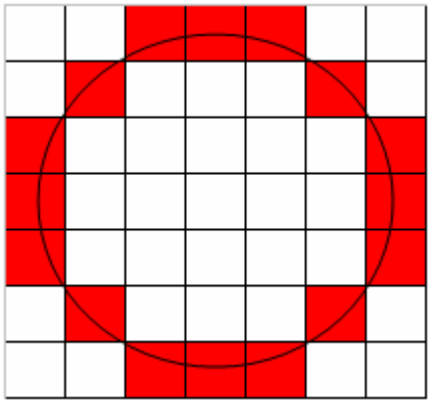


Fig. 3.8 Ejemplo del LEG: Radius-to-Fix

Al encontrarnos en un sistema discreto y no continuo, no podemos dividir la circunferencia en cuadrantes como en la imagen anterior, sino que definiremos un sistema propio para concretar que casillas pertenecen a cada cuadrante.

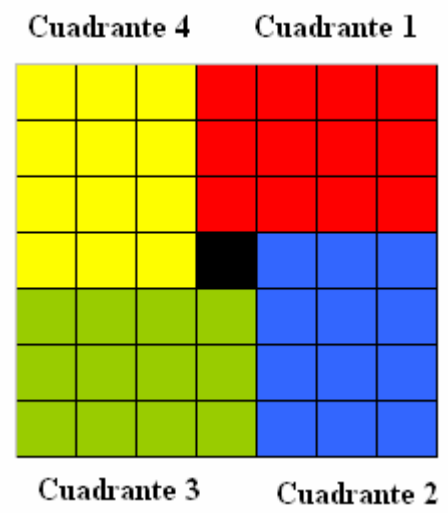


Fig. 3.9 División circunferencia en cuadrantes de forma discreta

De la misma forma que en el algoritmo de la recta, usaremos la geometría del plano para el desarrollo de dicho algoritmo.

Explicación del algoritmo:

Primero de todo, encontramos el radio de la curva aplicando Pitágoras. Para ello, sólo necesitamos la posición inicial (i, j) y el centro (c_x, c_y).

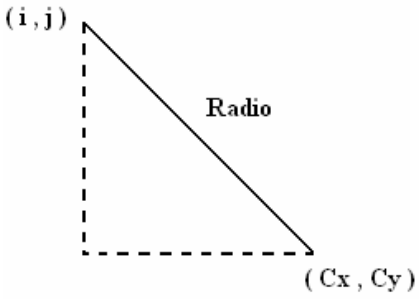


Fig. 3.10 Cálculo radio mediante Pitágoras

$$radio(m) = \sqrt{(i - c_x)^2 \cdot 30^2 + (j - c_y)^2 .30^2} \qquad \textbf{(3.4)}$$

Independientemente del cuadrante en el que nos encontremos, el algoritmo consistirá en comprobar si se cumple la ecuación de la circunferencia en cada casilla teniendo en cuenta un margen de error (dimensiones de la casilla: 30 x 30 m), ya que nuestro sistema es discreto y no continuo. Una casilla sólo pertenecerá a la trayectoria del avión si cumple las dos inecuaciones siguientes:

$$\begin{aligned}(i-c_x)^2\cdot 30^2+(j-c_y)^2\cdot 30^2-radio^2 &\le 30^2\\ (i-c_x)^2\cdot 30^2+(j-c_y)^2\cdot 30^2-radio^2 &\ge -30^2\end{aligned} \qquad \textbf{(3.5)}$$

También se comprobará, antes del desarrollo del algoritmo, que la distancia entre la posición final y el centro cumpla las inecuaciones anteriores.

Seguidamente, identificaremos en que cuadrante nos encontramos y según el sentido de giro desarrollaremos un algoritmo u otro.

En cada iteración comprobaremos siempre tres casillas. La primera corresponde a la más alejada del centro, la segunda a la casilla que se encuentra en diagonal y la tercera a la más cercana al centro. Siempre se ha de hacer en este orden porque es el más parecido a la trayectoria de una circunferencia. Se comprobarán las posiciones que cumplan las fórmulas anteriores, pero se tendrá en cuenta solamente la última posición que lo cumpla, para aplicar nuevamente el algoritmo a partir de esa casilla del cuadrante correspondiente.

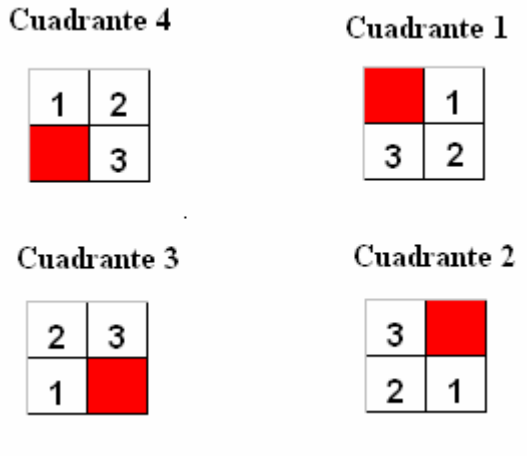


Fig. 3.11 Representación algoritmo según cuadrante con sentido de giro *clockwise*

El algoritmo finalizará cuando una de las posiciones por las que pase el avión corresponda a la posición final definida anteriormente.

3.2.1.1. Ejemplo: Radius-to-Fix

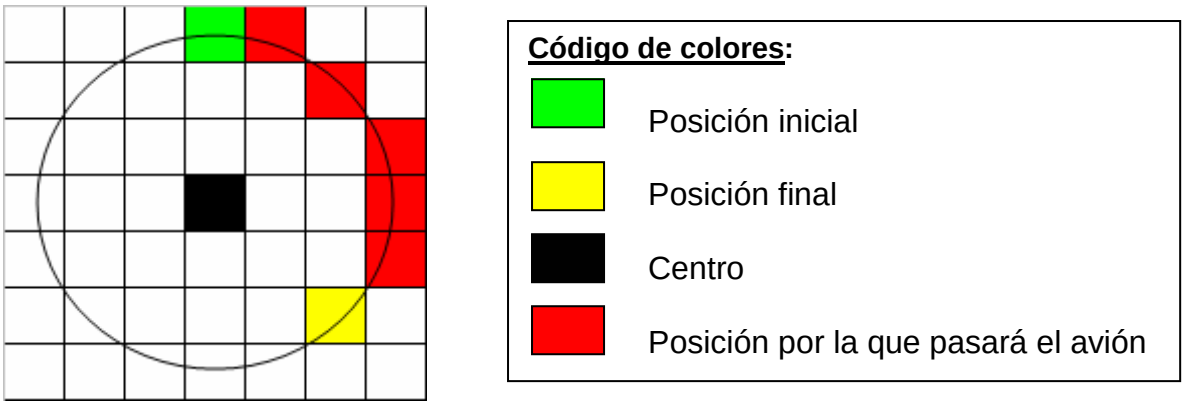


Fig. 3.12 Ejemplo algoritmo del arco de circunferencia

Posición inicial: (0,3)
Posición final: (5,5)
Centro: (3,3)
Sentido de giro: *clockwise*

$$radio(m) = \sqrt{(0-3)^2 \cdot 30^2 + (3-3)^2 \cdot 30^2} = 90m$$

Verificamos que la posición final es una casilla que pertenece a la trayectoria del avión:

$$(5-3)^2 \cdot 30^2 + (5-3)^2 \cdot 30^2 - 90^2 \leq 30^2 \qquad -900 \leq 900$$
$$(5-3)^2 \cdot 30^2 + (5-3)^2 \cdot 30^2 - 90^2 \geq -30^2 \qquad -900 \geq -900$$

Comprobación de las 3 casillas:

Primero comprobamos la más alejada al centro, en este caso la (0,4).

$$(0-3)^2 \cdot 30^2 + (4-3)^2 \cdot 30^2 - 90^2 \leq 30^2 \qquad 900 \leq 900$$
$$(0-3)^2 \cdot 30^2 + (4-3)^2 \cdot 30^2 - 90^2 \geq -30^2 \qquad 900 \geq -900$$

Como se cumplen las dos inecuaciones, verificamos que por esta casilla pasa el avión, por tanto, guardamos esta posición.

Ahora, comprobaremos la casilla que se encuentra en diagonal (1,4)

$$(1-3)^2 \cdot 30^2 + (4-3)^2 \cdot 30^2 - 90^2 \leq 30^2 \qquad -3600 \leq 900$$
$$(1-3)^2 \cdot 30^2 + (4-3)^2 \cdot 30^2 - 90^2 \geq -30^2 \qquad -3600 \geq -900$$

Se puede observar que la segunda inecuación no se cumple, es decir, esta posición no será utilizada por el avión en su vuelo.

Por último, comprobamos la casilla más cercana al centro (1,3)

$$(1-3)^2 \cdot 30^2 + (3-3)^2 \cdot 30^2 - 90^2 \leq 30^2 \qquad -4500 \leq 900$$
$$(1-3)^2 \cdot 30^2 + (3-3)^2 \cdot 30^2 - 90^2 \geq -30^2 \qquad -4500 \geq -900$$

De la misma manera que en la casilla anterior, falla la segunda inecuación, por tanto, tampoco guardamos esta posición.

Una vez comprobadas las 3 casillas, aplicaremos de nuevo este método desde la casilla (0,4), ya que fue la última posición que guardamos.

Este proceso se repetirá de igual manera hasta comprobar que una de las casillas por las que pasa el avión corresponda con la posición final dada anteriormente.

3.2.2. Pendiente 3-D

Este apartado sirve para justificar el REQ-USU 06 en su totalidad.

En este segundo caso (*RF*), la pendiente se calculará de manera muy similar al primer caso. Sólo hay que tener en cuenta de que se trata de un arco de circunferencia, por tanto, el cálculo de la distancia total será en base a esta característica.

Primero de todo, para el cálculo de la distancia que separa las posiciones inicial y final utilizaremos las propiedades de la circunferencia para poder calcular la longitud de un arco de circunferencia.

En segundo lugar, calcularemos la pendiente de igual forma que en el *TF*, es decir, dividiendo la diferencia de alturas entre la distancia total (longitud arco de circunferencia). Esta manera de calcular la pendiente supondrá que el avión realiza un vuelo en progresión.

La pendiente tendrá dos interpretaciones según el signo que le corresponda. Si es positivo querrá decir que asciende, en cambio, si es negativo estará descendiendo.

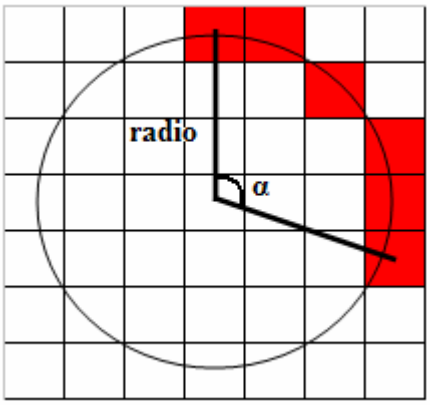


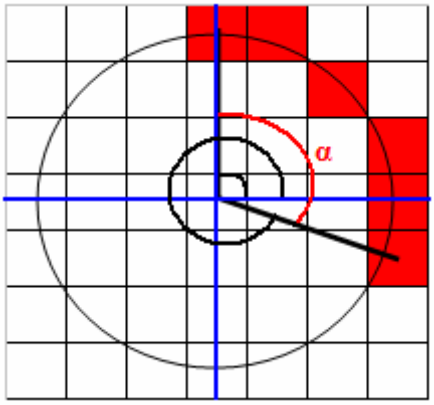
Fig. 3.13 Ejemplo cálculo distancia total recorrida

$$dist.total(m) = radio \cdot \alpha \qquad \textbf{(3.6)}$$

Para el cálculo de α seguiremos el siguiente proceso:

Primero de todo, calcularemos qué ángulo forma la posición inicial y final respecto al eje de abscisas (horizontal), es decir, transformaremos las coordenadas rectangulares a polares.

Una vez obtenido dichos ángulos, podremos obtener el ángulo (α) que forman entre sí la posición inicial y final.



Hay que tener en cuenta que según el sentido de giro el ángulo puede ser α o $360^{o}-\alpha$.

Fig. 3.14 Ejemplo cálculo ángulo entre dos posiciones

Una vez calculada la distancia total podremos calcular la pendiente teniendo en cuenta la altura inicial y final.

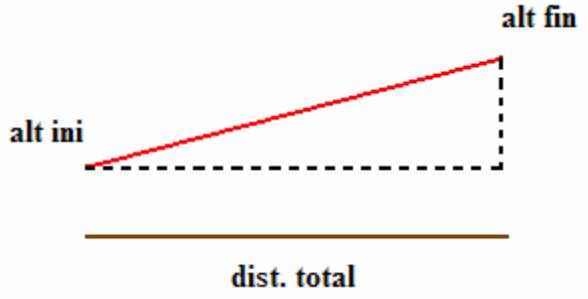


Fig. 3.15 Ejemplo cálculo pendiente vertical

$$m\_alt = \frac{alt\_fin - alt\_ini}{dist.total} \qquad \textbf{(3.7)}$$

Por último, calcularemos la altura que el avión adquirirá en cada posición por la que vuele, teniendo en cuenta que la distancia variará, por tanto, habrá que calcular la longitud del arco de circunferencia en cada posición.

$$alt\_inst = alt\_ini + m\_alt \cdot (radio \cdot \alpha) \qquad \textbf{(3.8)}$$

El parámetro α es el único que cambiará en función del ángulo que se forme entre el origen y la posición del avión en ese momento.

3.3. Margen de seguridad lateral y vertical

En ambos LEGs se podrá introducir un margen lateral y vertical para proporcionar mayor seguridad al avión durante el vuelo.

Esta implementación se ha llevado a cabo para justificar los REQ-USU 07, 08, 09 y 10.

3.3.1. Margen lateral

El usuario podrá configurar cuánto margen de seguridad lateral quiere para su plan de vuelo. Este dato se expresará en metros, y el programa lo transformará a número de casillas ya que se trata de un sistema discreto y no continuo.

Para poder expresar el margen en casillas se realizará un algoritmo que consiste en dividir el margen expresado en metros entre 30, que son las dimensiones en metros de una casilla. Si el número obtenido no es un entero, se redondeará siempre hacia el siguiente entero. Se hará de esta forma porque es más coherente proporcionar mayor seguridad, que no menos de la pedida por el usuario.

$$\frac{\text{margen\_metros}}{30 \text{ metros}} = \text{margen\_casillas} \qquad \textbf{(3.9)}$$

(Si margen_casillas no es un número entero redondear hacia arriba)

Según la trayectoria del avión, las casillas que se comprobarán serán diferentes.

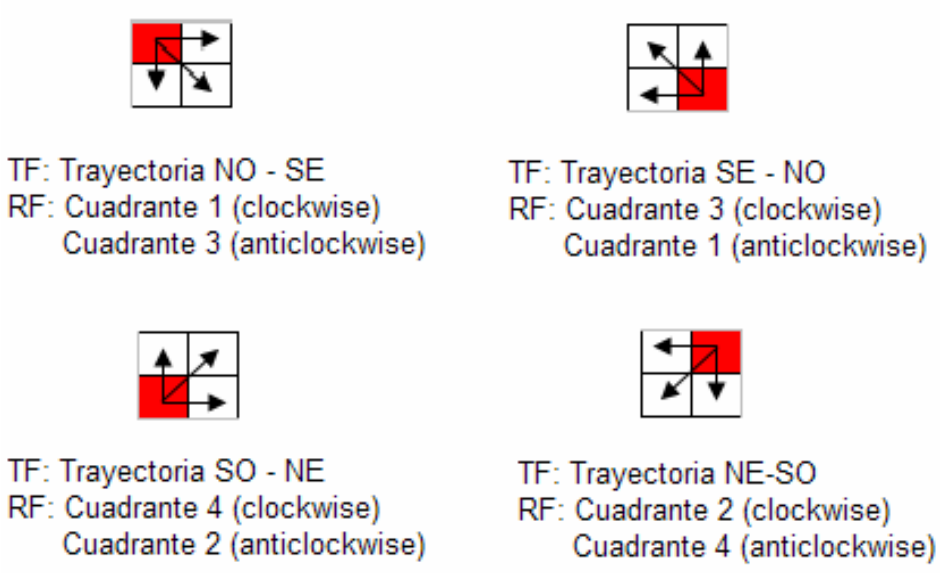


Fig. 3.16 Algoritmo del margen de seguridad lateral

3.3.2. Margen vertical

Este segundo margen se puede dividir en dos diferenciando respecto a que están referenciados. Ambos márgenes (inferior y superior) se expresarán en metros.

Por un lado, el margen vertical inferior se aplica siempre respecto a la altura que tiene el avión en cada instante.

$$alt\_avión = alt\_inst - mar\_vert\_inf \quad \textbf{(3.10)}$$

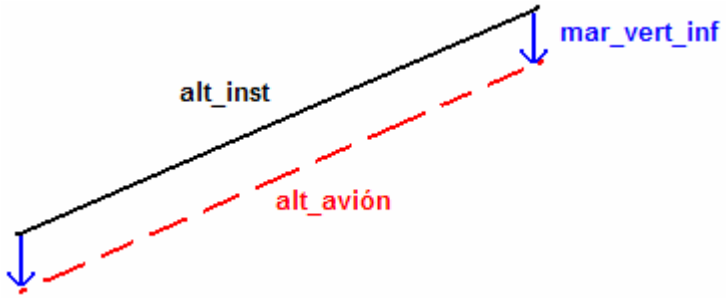


Fig. 3.17 Representación margen vertical inferior

Por otro lado, el margen de seguridad vertical superior será indicado respecto a la altura máxima (1000 pies) que podrá alcanzar el avión en cada momento.

$$alt\_máx\_real = alt\_máx - mar\_vert\_sup \quad \textbf{(3.11)}$$

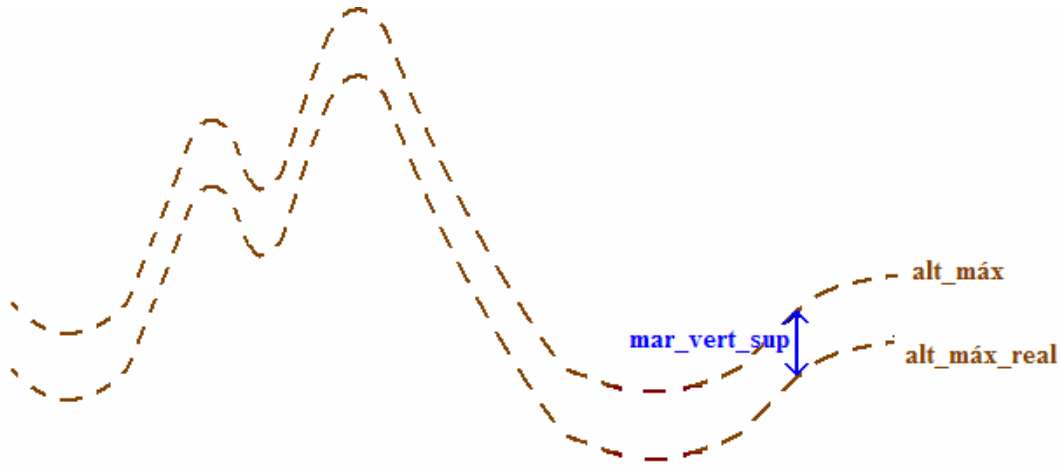


Fig. 3.18 Representación margen vertical superior

3.4. Mínima altura del avión

Para justificar el REQ-USU 12 la aplicación informará al usuario de la altura mínima que adquirirá el avión en todo el plan de vuelo. Esta altura mínima es la menor diferencia que existe entre la altura del avión en un instante y el nivel del terreno en esa misma posición.

Para conseguir esta altura mínima, se analizará cada LEG por separado y se obtendrá de cada LEG su altura mínima. Una vez se hayan analizado todos los LEGs con sus respectivas alturas mínimas, la aplicación informará de la altura más crítica de los diferentes LEGs, es decir, de todo el *flight plan*.

$$\min (\text{alt\_inst}(p) - \text{nivel\_terreno}(p)) \ \forall p \in \text{Flight Plan} \qquad \textbf{(3.12)}$$

En la siguiente ilusatración se puede observar gráficamente como se obtendrían las alturas mínimas de cada LEG. El caso más restrictivo, es decir, la altura mínima de todo el plan de vuelo sería la que pertenece al LEG2 (alt_min_2).

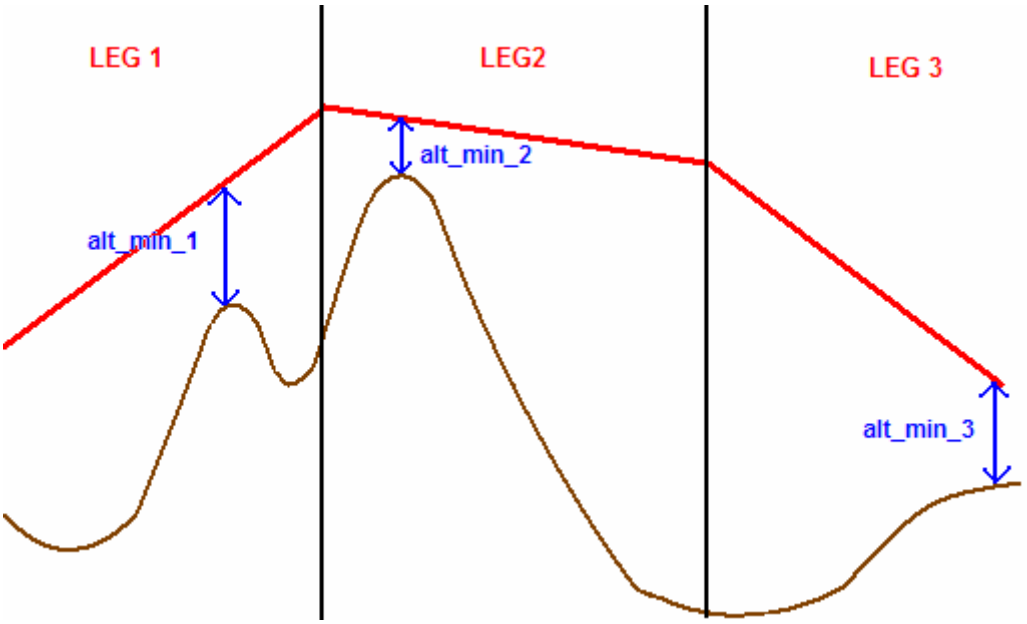


Fig. 3.19 Representación altura mínima de un *flight plan*

3.5. Relación entre los tipos de coordenadas

Las coordenadas proporcionadas por el usuario serán geográficas (latitud, longitud) por lo que se justifica el REQ-USU 04.

Una vez obtenidas estas coordenadas, será necesario transformarlas en coordenadas de nuestro modelo de elevación del terreno. Para ello, se necesitará realizar un paso previo que consiste en transformar primero las coordenadas geográficas a cartesianas (UTM).
El algoritmo que realiza este cambio ha sido proporcionado por el grupo de investigación ICARUS en el lenguaje de programación *C#*. Por lo que, se ha tenido que transformar a *C*, ya que es el lenguaje que utilizaremos en el programa.

Por último, cuando tenemos las coordenadas expresadas en UTM, lo único que tendremos que realizar es dividir cada coordenada entre 30 (resolución en metros de nuestro DEM) para obtener las coordenadas referenciadas a nuestro modelo de elevación del terreno.

En nuestro caso, utilizaremos el modelo cartográfico de Cataluña, que tiene como vértice inicial del fichero la coordenada (257995, 4752005). Por tanto, antes de dividir entre 30, habrá que restarle la coordenada inicial para obtener como punto de referencia la casilla (0,0).

$$\text{Coordenada X} = \frac{\text{UTMx - 257995}}{30} \qquad \text{Coordenada Y} = \frac{\text{4752005 - UTMy}}{30} \qquad \textbf{(3.13)}$$

CAPÍTULO 4. VERIFICACIÓN DE LA APLICACIÓN

4.1. Pruebas Track-to-Fix

Para comprobar el buen funcionamiento del algoritmo realizaremos una serie de pruebas según los datos iniciales proporcionados por el usuario.

Las siguientes pruebas han sido realizadas para verificar el REQ-USU 01.

Código de colores:

- Posición por la que pasará el avión
- Posición de margen de seg. lateral.
- Obstáculo.

4.1.1. Proyección horizontal

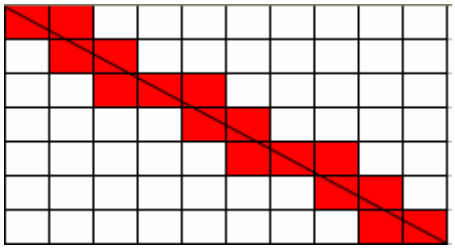
A continuación, se mostrarán algunos ejemplos de un vuelo horizontal (proyección sobre el suelo) considerando que la altura a la que vuela el avión es siempre la misma.

En estos ejemplos se podrá observar que se cumple el REQ-USU 05.

Prueba 1:

En este primer ejemplo la trayectoria que se realizará será la NO – SE teniendo en cuenta dos aspectos, la presencia de obstáculos y el margen de seguridad lateral. Esta trayectoria será similar a la opuesta: SE – NO.

Sin obstáculos
mar_seg_lat = 0 m.



Con obstáculos
mar_seg_lat = 0 m.

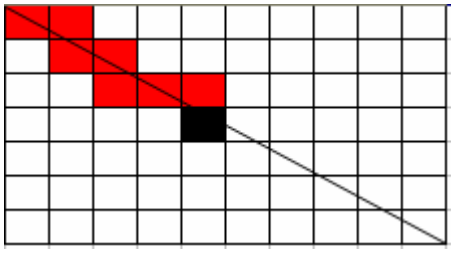
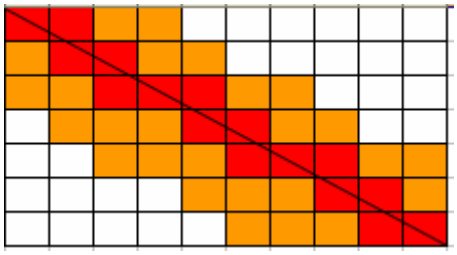


Fig. 4.1 Ejemplo *TF* sentido NO-SE

En la imagen de la izquierda se muestran las casillas por las que pasa el avión asumiendo que no hay colisión y por este motivo puede realizar todo el trayecto. En cambio, en la ilustración de la derecha se introduce un obstáculo con una altura superior a la del vuelo del avión, por tanto, se produce un conflicto y la verificación del plan de vuelo dará error.

En las siguientes figuras se ha introducido un margen lateral para verificar el REQ-USU 07.

Sin obstáculos (NO – SE)
mar_seg_lat = 50 m. (2 casillas)



Sin obstáculos (SE – NO)
mar_seg_lat = 50 m. (2 casillas)

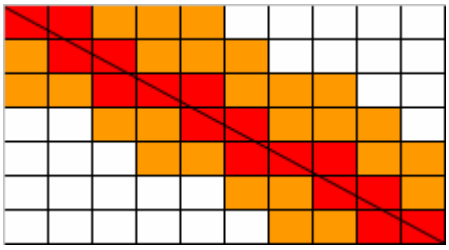


Fig. 4.2 Ejemplo *TF* con margen lateral

En estas dos imágenes con trayectorias opuestas, observamos que las posiciones sobrevoladas por el avión son las mismas, pero las casillas que indican los márgenes laterales no lo son ya que cada trayectoria comprueba distintas posiciones en función de si el sentido es NO – SE o SE – NO.

Con obstáculos (NO – SE)
mar_seg_lat = 50 m. (2 casillas)

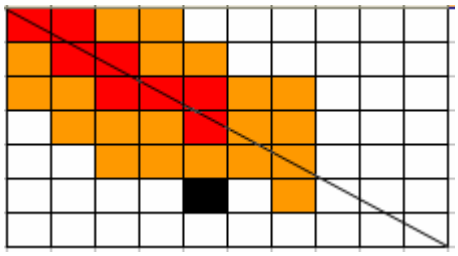


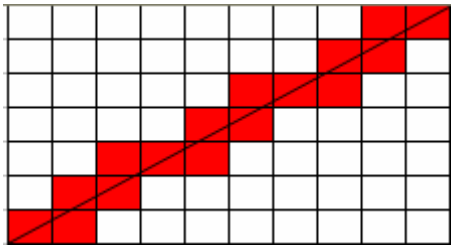
Fig. 4.3 Ejemplo *TF* con margen lateral y colisión

Por último, en la tercera ilustración se produce un error en el plan de vuelo por la presencia de un obstáculo que, aunque no se encuentra en la trayectoria del avión, sí que obstaculiza por la presencia del margen de seguridad.

Prueba 2:

En este segundo ejemplo se visualizará la trayectoria NE – SO (similar a la trayectoria opuesta SO – NE) con las mismas condiciones que en el ejemplo anterior.

Sin obstáculos.
mar_seg_lat = 0 m.



Con obstáculos.
mar_seg_lat = 0 m.

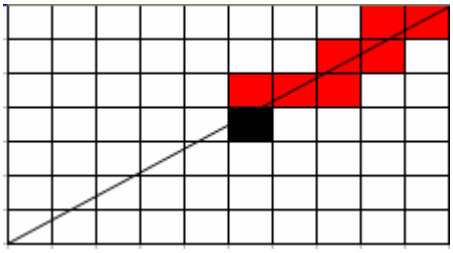
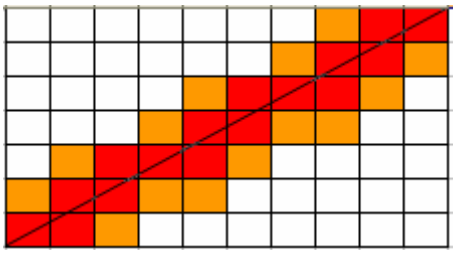


Fig. 4.4 Ejemplo *TF* sentido NE-SO

Sin obstáculos.
mar_seg_lat = 25 m. (1 casilla)



Con obstáculos.
mar_seg_lat = 25 m. (1 casilla)

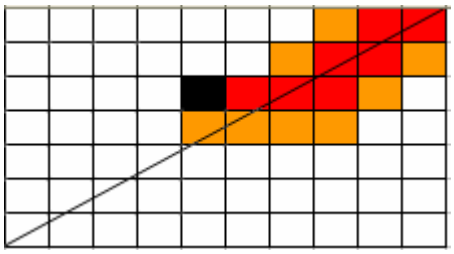


Fig. 4.5 Ejemplo TF con margen lateral

4.1.2. Proyección vertical

Para poder representar el vuelo que realiza el avión entre la altura inicial y final, utilizaremos el mismo método aunque ahora el plano que se mostrará será el plano que contiene a la recta y que a su vez es perpendicular al suelo.

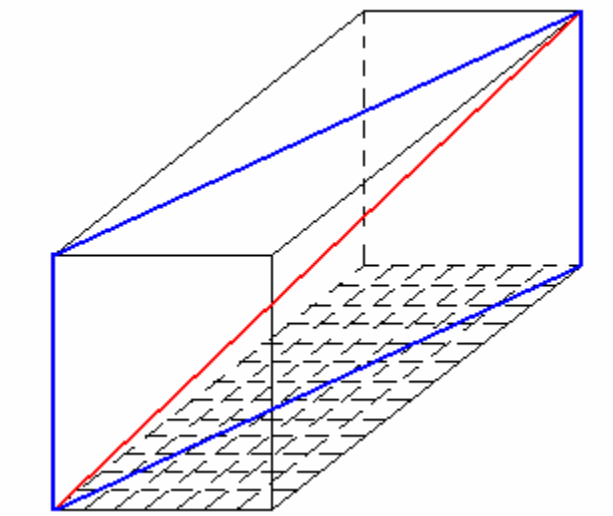
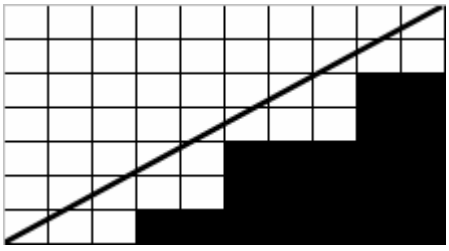


Fig. 4.6 Plano proyección vertical *TF*

Prueba 1:

En este ejemplo se mostrará la influencia del margen vertical inferior para proporcionar seguridad ante un posible choque.

Sin obstáculos.
Sin mar_vert_inf.



Con obstáculos.
Sin mar_vert_inf.

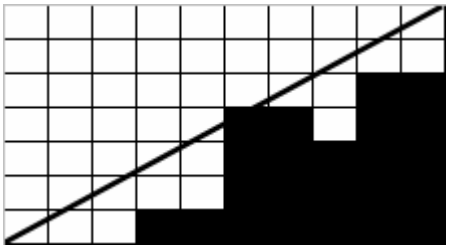
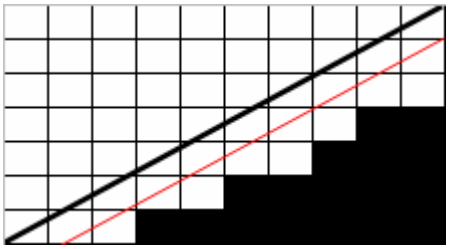


Fig. 4.7 Ejemplo *TF* proyección vertical

En el primer caso no se produce colisión porque las alturas que va adquiriendo el avión son superiores a los obstáculos de la orografía del terreno. En cambio, en el segundo caso se produce una obstrucción en la trayectoria, por tanto, el plan de vuelo sería erróneo.

La siguiente prueba sirve para verificar el REQ-USU 09.

Sin obstáculos.
Con mar_vert_inf.



Con obstáculos.
Con mar_vert_inf.

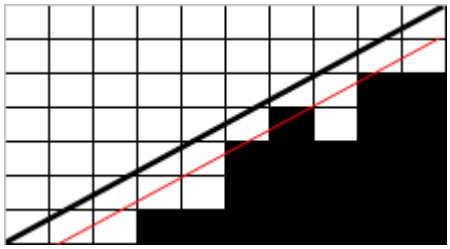


Fig. 4.8 Ejemplo *TF* con margen vertical inferior

En estos dos nuevos casos se ha introducido un margen de seguridad vertical. Se puede apreciar que los obstáculos en realidad no obstruyen el trayecto del avión, pero en el segundo caso al introducir el margen de seguridad la verificación del plan de vuelo no sería correcta., porque se produce una obstrucción con el margen vertical.

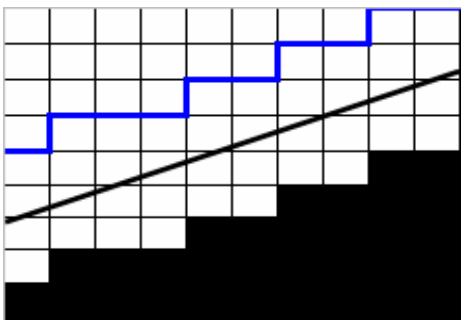
En el caso del sentido descendente se analizaría de forma análoga, pero ahora teniendo en cuenta que el avión cada vez tendrá una altura inferior (pendiente negativa).

Prueba 2:

En este segundo ejemplo se podrá visualizar cómo afecta la altura máxima establecida por la legislación española (1000 pies), así como el margen vertical superior.

Para verificar el REQ-USU 08 se han realizado las siguientes pruebas.

Sin obstáculos.
Sin mar_vert_sup.



Con obstáculos.
Sin mar_vert_sup.

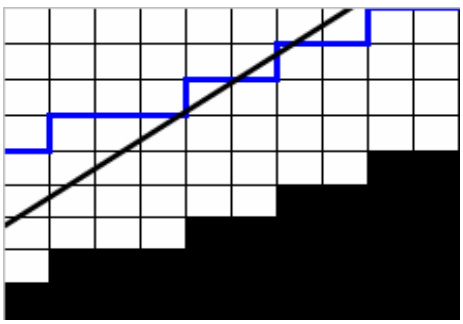
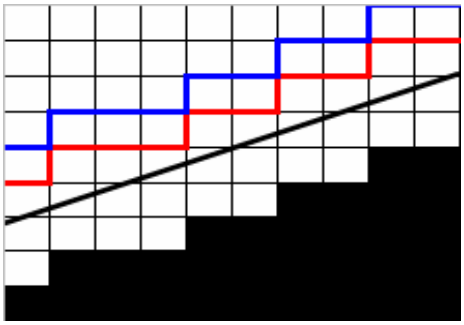


Fig. 4.9 Ejemplo *TF* con altura máxima

En estas dos primeras imágenes se puede observar que existe un límite de altura, en el que en el primer caso no se sobrepasa, por tanto, el vuelo del avión es correcto. En cambio, en la segunda imagen se puede ver como el avión sí que lo sobrepasa, por lo que, entraría en espacio aéreo controlado, es decir, el plan vuelo no sería correcto.

En el siguiente caso se ha introducido un margen vertical superior para poder verificar el REQ-USU 10.

Sin obstáculos.
Con mar_vert_sup.



Con obstáculos.
Con mar_vert_sup.

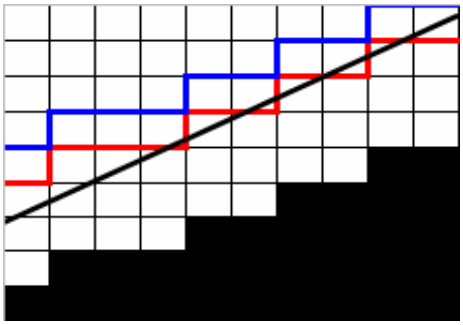


Fig. 4.10 Ejemplo *TF* con margen vertical superior

En las dos últimas ilustraciones se introduce el margen de seguridad vertical superior, el cual en el primer trayecto no afectaría al plan de vuelo porque no consigue sobrepasarlo. Por el contrario, en el segundo caso, aunque la altura del avión sea menor que la altura máxima, éste sobrepasa el margen de seguridad, por tanto, el *flight plan* sería erróneo.

4.2. Pruebas Radius-to-Fix

De la misma forma que en el primer tipo de LEG (*TF*), se harán una serie de pruebas para comprobar que el algoritmo desarrollado anteriormente es correcto.

En las pruebas siguientes se podrá observar que se verifica el REQ-USU 01.

Código de colores:

- Posición por la que pasará el avión
- Posición de margen de seg. lateral.
- Obstáculo.

4.2.1. Proyección horizontal

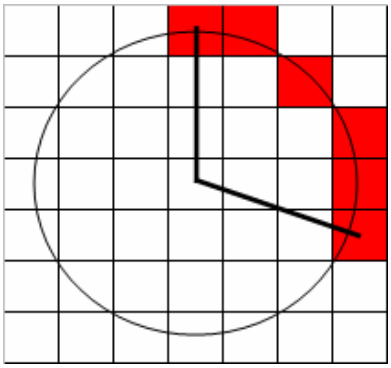
En esta primera proyección (realizada sobre el suelo) se analizará el movimiento del avión y el posible margen de seguridad introducido por el usuario, así como posibles colisiones con obstáculos.

Las pruebas realizadas en esta proyección sirven para verificar el REQ-USU 06.

Prueba 1:

En este ejemplo se mostrarán las casillas que forman parte de la trayectoria del avión, teniendo en cuenta que la dirección de giro que se seguirá será el de las agujas del reloj (*clockwise*) y que no habrá presencia del margen de seguridad.

Sin obstáculos
mar_seg_lat = 0 m.



Con obstáculos
mar_seg_lat = 0 m.

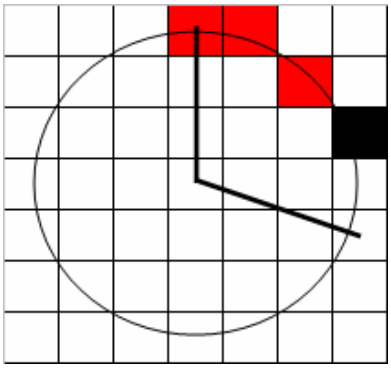


Fig. 4.11 Ejemplo *RF* sentido *clockwise*

En la ilustración de la izquierda se muestran las casillas por las que pasa el avión asumiendo que no hay colisión y por este motivo puede realizar todo el trayecto. En cambio, en la ilustración de la derecha se introduce un obstáculo con una altura superior a la del vuelo del avión, por tanto, se produce un conflicto y la verificación del plan de vuelo dará error.

Prueba 2:

En este segundo ejemplo la trayectoria que se realizará será la contraria a la de las agujas del reloj (*anticlockwise*) con la presencia, en este caso, de un margen de seguridad lateral.

El REQ-USU 07 será verificado con la siguiente prueba.

Sin obstáculos
mar_seg_lat = 25 m. (1 casilla)

Con obstáculos
mar_seg_lat = 25 m. (1 casilla)

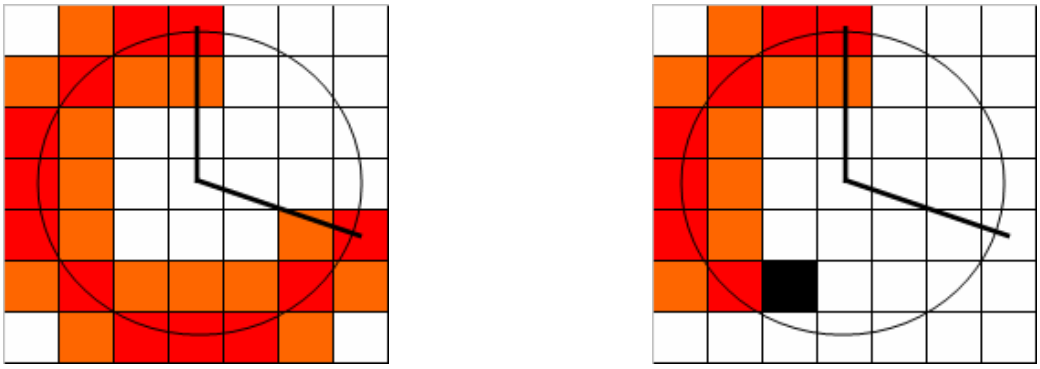


Fig. 4.12 Ejemplo *RF* sentido *anticlockwise* con margen lateral

En estas dos imágenes se puede observar que la primera puede realizar de forma segura el plan de vuelo, porque no detecta la presencia de obstáculos. En cambio, en la segunda se produce un error en el plan de vuelo propuesto a causa de una colisión, que si bien no se encuentra en la trayectoria exacta del avión, sí que afecta, ya que se ha introducido un margen de seguridad.

4.2.2. Proyección vertical

En esta proyección se mostrará la utilización de los márgenes verticales, así como la altura máxima establecida (1000 pies).

Si se produjera un descenso, la pendiente sería negativa aunque las siguientes pruebas se llevarían a cabo de la misma manera.

En este caso, el plano que se mostrará será el que se obtiene de realizar un corte al cilindro en el que está inscrito el arco de circunferencia.

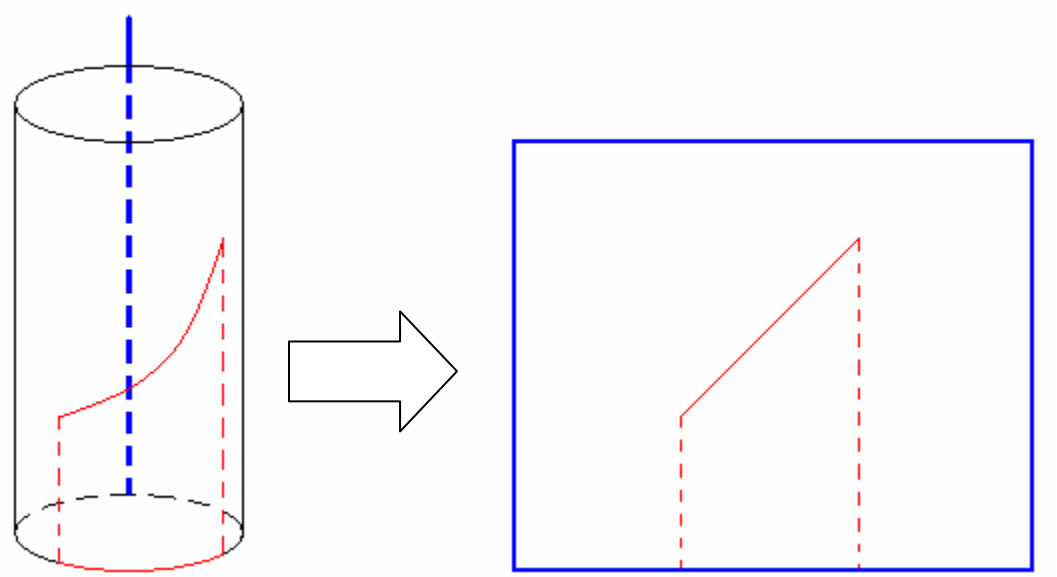
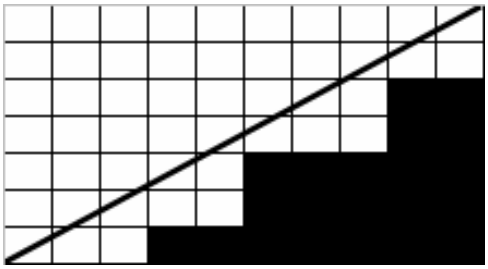


Fig. 4.13 Plano proyección vertical *RF*

Prueba 1:

En este primer ejemplo se podrán visualizar las diferentes alturas que adquiere el avión. En estos ejemplos el sentido es ascendente, aunque el descendente se analizaría de igual forma.

Sin obstáculos.
Sin mar_vert_inf.



Con obstáculos.
Sin mar_vert_inf.

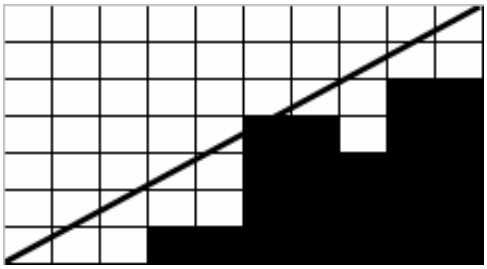
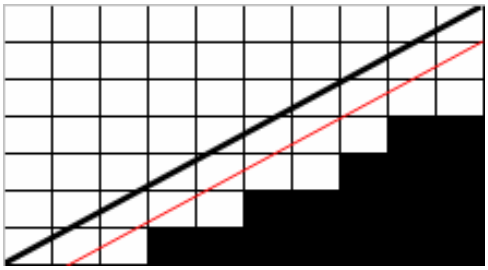


Fig. 4.14 Ejemplo *RF* proyección vertical

En estas dos imágenes podemos observar que en la primera de ellas no se produce colisión entre el avión y la elevación del terreno. Por el contrario, en la segunda imagen sí que se detecta una colisión por lo que el *flight plan* propuesto no sería válido.

Las dos ilustraciones siguientes se han realizado para verificar el REQ-USU 09.

Sin obstáculos.
Con mar_vert_inf.



Con obstáculos.
Con mar_vert_inf.

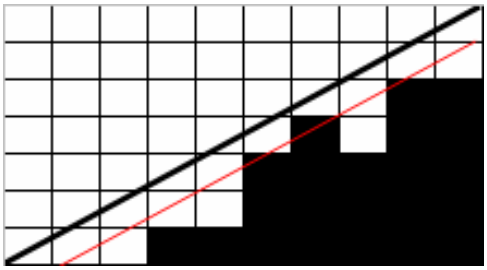


Fig. 4.15 Ejemplo *RF* con margen vertical inferior

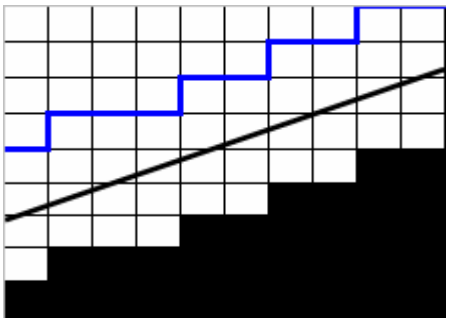
En la figura anterior se puede observar que está presente el margen de seguridad vertical inferior. En la primera ilustración no se produciría colisión, por tanto, el plan de vuelo sería correcto, en cambio, en la segunda no lo sería porque se produce una obstrucción entre el margen inferior y el nivel del terreno.

Prueba 2:

Este segundo ejemplo servirá para poder observar la relevancia del margen de seguridad vertical superior y el límite de altura que son 1000 pies.

En las siguientes figuras (Fig. 4.16 y 4.17) se podrá verificar el REQ-USU 08.

Sin obstáculos.
Sin mar_vert_sup.



Con obstáculos.
Sin mar_vert_sup.

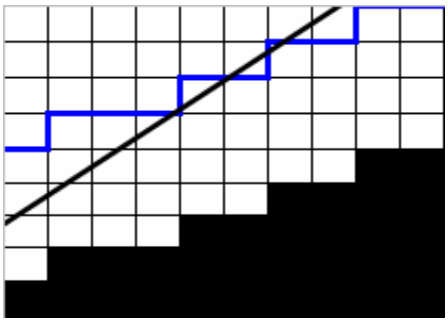
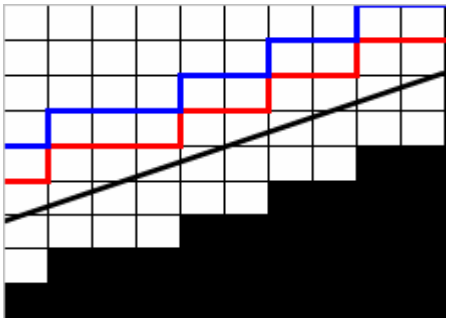


Fig. 4.16 Ejemplo *RF* con altura máxima

En este caso se puede ver que está presente un límite de altura, donde en la primera imagen el avión no lo sobrepasa, pero que en la segunda sí que vuela por encima de ese límite establecido, por tanto, en el segundo caso el plan de vuelo sería erróneo.

Esta última figura se ha realizado para observar que se verifica el REQ-USU 10.

Sin obstáculos.
Con mar_vert_sup.



Con obstáculos.
Con mar_vert_sup.

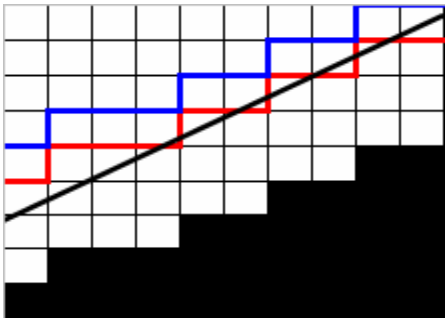


Fig. 4.17 Ejemplo *RF* con margen vertical superior

En este último caso se tiene en cuenta el margen de seguridad vertical superior, el cual a causa de su presencia haría que el *flight plan* de la segunda imagen no fuera correcto, en cambio, en la primera no afectaría y sería válido.

CAPÍTULO 5. FLIGHT PLAN ALTERNATIVO

Este capítulo se ha realizado porque al haberse producido un cambio en un requisito (REQ-USU 03), se ha querido mostrar una nueva aportación suplementaria como puede ser la realización de un *flight plan* alternativo.

En este apartado se explicará como se podría trazar un camino alternativo si se detectara un obstáculo en la trayectoria del avión.

El algoritmo que se explicará a continuación corresponde a una primera aportación para el desarrollo de una futura aplicación relacionada con la posibilidad de obtener un nuevo *flight plan*. Por último, dicho algoritmo sólo se aplicará para el LEG *Track-to-Fix*.

5.1. Algoritmo trayectoria alternativa

El algoritmo se diseña de la misma forma que en el algoritmo de la línea recta (apartado 3.1.1.) hasta que se detecta una colisión.
A partir de ese momento, la aplicación será capaz de conocer la posición dónde se produce la obstrucción y trazar un nuevo trayecto hasta el destino final.

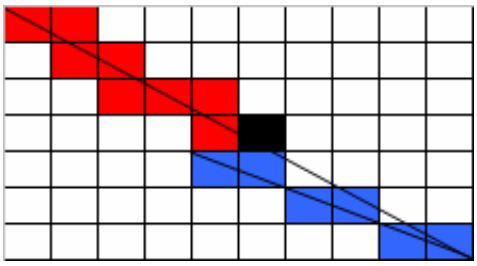


Fig. 5.1 Ejemplo trayectoria alternativa

Explicación del algoritmo:

Tal y como se ha comentado anteriormente, el algoritmo se desarrolla igual que en el de la línea recta. La única diferencia es que cuando se detecta un obstáculo y según la trayectoria que siga el avión, éste se desviará para un lado o para otro.

Primero de todo, se aplica el algoritmo de la línea recta. Cuando se detecta un obstáculo, el avión se desplaza hacia la casilla que corresponda según la dirección y sentido de la trayectoria. A partir de aquí se asume que esta posición es la posición inicial y se vuelve a realizar el algoritmo de la línea recta, es decir, se obtiene una submatriz y se vuelve a calcular la nueva pendiente.

Este proceso se repetirá tantas veces como obstáculos haya en la trayectoria del avión.

Por último, también se podrán introducir los márgenes de seguridad que el usuario crea conveniente.

5.2. Cambio de posición según trayectoria

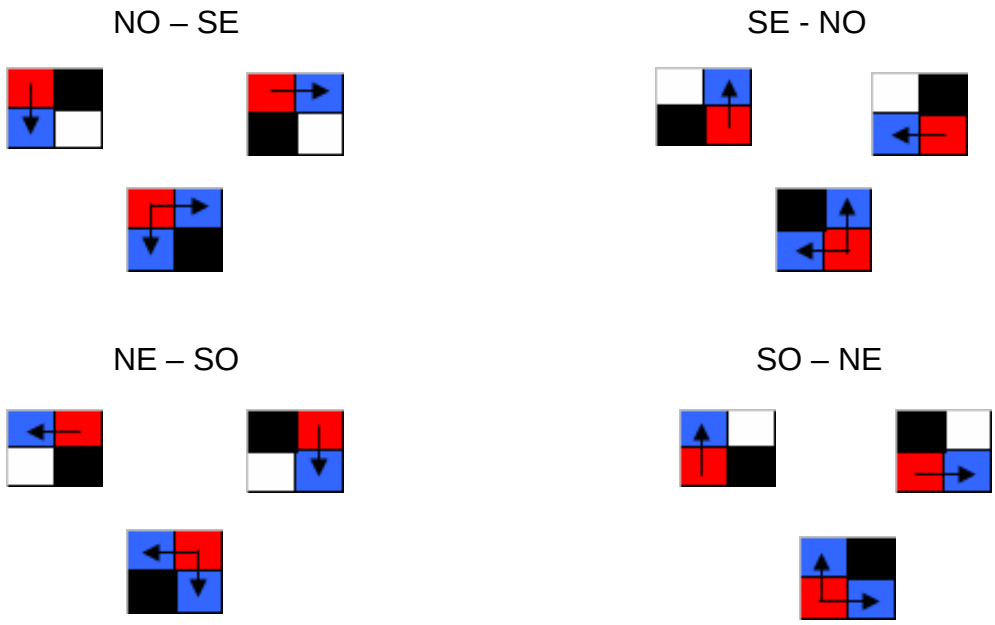


Fig. 5.2 Algoritmo de cambio de posición

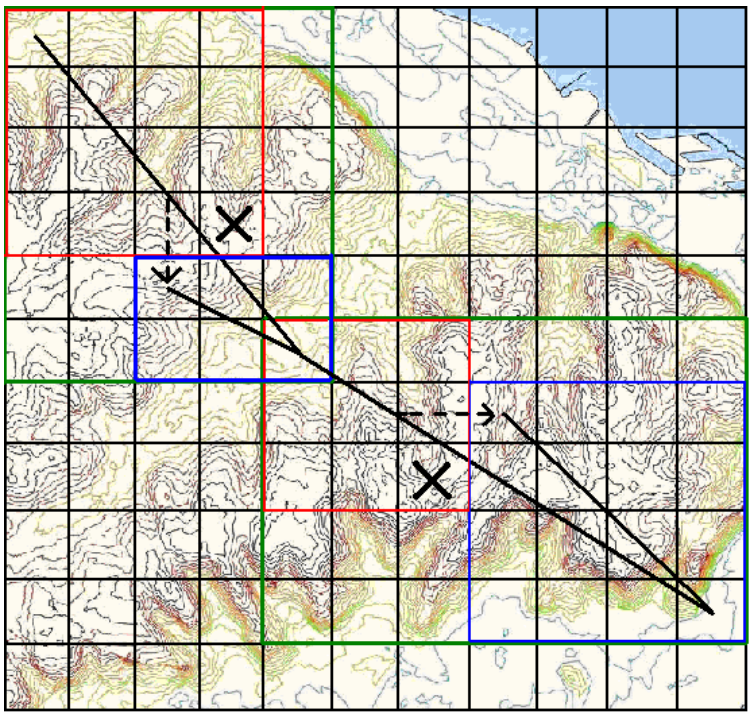


Fig. 5.3 Ejemplo *flight plan* alternativo en modelo cartográfico

5.3. Ejemplo algorítmico de un flight plan alternativo

Trayectoria: NO – SE

Algoritmo hasta colisión:

$$m = \frac{7}{10} = 0.7$$

Recta: $y = m \cdot x$

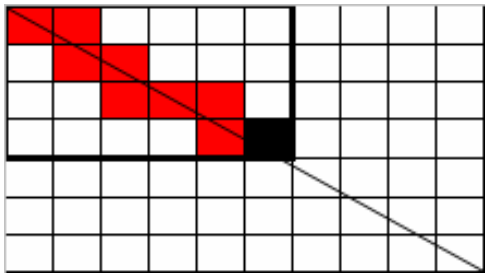


Fig. 5.4 Ejemplo primer algoritmo de la línea recta

Bucle 1: Recorrido de columnas (*x*):

$x = 1 \rightarrow y = 0.7$

Bucle 2: Recorrido de filas

Valor inicial de k=0.

$k = 0 \Rightarrow Casilla(0,0)$

$k = 1 >= y = 0.7$

$x = 2 \rightarrow y = 1.4$

Valor inicial de k=0.

$k = 0 \Rightarrow Casilla(0,1)$

$k = 1 \Rightarrow Casilla(1,1)$

$k = 2 >= y = 1.4$

$x = 3 \rightarrow y = 2.1$

Valor inicial de k=1.

$k = 1 \Rightarrow Casilla(1,2)$

$k = 2 \Rightarrow Casilla(2,2)$

$k = 3 >= y = 2.1$

$x = 4 \rightarrow y = 2.8$

Valor inicial de k=2.

$k = 2 \Rightarrow Casilla(2,3)$

$k = 3 >= y = 2.8$

$x = 5 \rightarrow y = 3.5$

Valor inicial de k=2.

$k = 2 \Rightarrow Casilla(2,4)$

$k = 3 \Rightarrow Casilla(3,4)$

$k = 4 >= y = 3.5$

Se detecta colisión en la casilla (3,5)

Desarrollo del nuevo algoritmo

Algoritmo hasta el final del trayecto:

Casilla inicial (4,4)

$$m = \frac{3}{6} = 0.5$$

Recta: $y = m \cdot x$

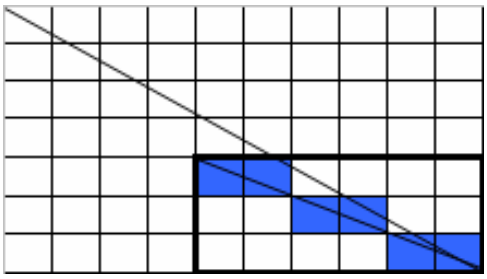


Fig. 5.5 Ejemplo segundo algoritmo de la línea recta

Bucle 1: Recorrido de columnas (*x*):

$$x = 1 \rightarrow y = 0.5$$

Bucle 2: Recorrido de filas

Valor inicial de k=0.

$$k = 0 \Rightarrow Casilla(4,4)$$
$$k = 1 >= y = 0.5$$

$$x = 2 \rightarrow y = 1$$

Valor inicial de k=0.

$$k = 0 \Rightarrow Casilla(4,5)$$
$$k = 1 >= y = 1$$

$$x = 3 \rightarrow y = 1.5$$

Valor inicial de k=1.

$$k = 1 \Rightarrow Casilla(5,6)$$
$$k = 2 >= y = 1.5$$

$$x = 4 \rightarrow y = 2$$

Valor inicial de k=1.

$$k = 1 \Rightarrow Casilla(5,7)$$
$$k = 2 >= y = 2$$

$$x = 5 \rightarrow y = 2.5$$

Valor inicial de k=2.

$$k = 2 \Rightarrow Casilla(6,8)$$
$$k = 3 >= y = 2.5$$

$$x = 6 \rightarrow y = 3$$

Valor inicial de k=2.

$$k = 2 \Rightarrow Casilla(6,9)$$

Casilla (6,9) es la posición final de la trayectoria

Finaliza el algoritmo

Plan de vuelo alternativo correcto

CAPÍTULO 6. PLANIFICACIÓN Y COSTES

El plan de trabajo inicial se componía de un total de 16 semanas (desde el 1 de marzo al 30 de junio) estructuradas de la siguiente forma:

Tabla 6.1. Plan de trabajo inicial

Número de semanas	Plan de trabajo
2	Implementar el algoritmo de la recta.
2	Implementar el algoritmo de la circunferencia.
1	Añadir márgenes de seguridad.
1	Realizar pruebas y documentación.
1	Poner en funcionamiento un *parser* de XML.
3	Generar planes de vuelo en XML, probarlos en la aplicación y documentarlos.
3	Incorporar el formato del Cartográfico a la aplicación y documentar.
2	Completar la documentación del TFC.
1	Preparar la presentación del TFC.

Respecto a esta estructura inicial se han producido algunos cambios a lo largo de la elaboración del proyecto. Uno de ellos, es la modificación del REQ-USU 03. Este cambio se produjo por orden del director del trabajo.

Por tanto, el plan de trabajo final es el siguiente:

Tabla 6.2. Plan de trabajo final

Número de semanas	Plan de trabajo
1	Estudiar requisitos del usuario, y las entradas y salidas de la aplicación.
1	Implementar el algoritmo de la recta.
1	Implementar la pendiente vertical de la recta.
1	Implementar el algoritmo de la circunferencia.
1	Implementar la pendiente vertical de la circunferencia
1	Implementar los algoritmos del margen de seguridad, cambio de coordenadas y altura mínima.
2	Realizar pruebas y documentar.
1	Implementar un plan de vuelo alternativo.
3	Comprobar un plan de vuelo dado un fichero de texto.
1	Incorporar el formato del Cartográfico a la aplicación.
2	Completar la documentación del TFC.
1	Preparar la presentación del TFC.

Para la elaboración del proyecto se han empleado diversas horas tanto para la redacción del documento como para el desarrollo de la aplicación.

Tabla 6.3. Tiempo de dedicación

Esbozos y pruebas iniciales	50 horas
Redacción del documento	70 horas
Desarrollo de la aplicación	100 horas

Por último, para la implementación de la aplicación se ha utilizado el compilador *gcc*. Este compilador se ha utilizado para compilar el programa en lenguaje *C*, mediante un emulador de Linux que se puede utilizar en Windows (*Cygwin*). Por otro lado, el programa se ha implementado en un editor de texto orientado a la programación (*UltraEdit-32*). También se ha utilizado el *Office* para el desarrollo del documento.

CONCLUSIONES

En este trabajo se ha desarrollado una aplicación que comprueba si un plan de vuelo, propuesto por el usuario, es seguro o no. Es decir, no se produce colisión y no vuela por espacio aéreo controlado.
Para el desarrollo del programa se han tenido que cumplir la totalidad de los requisitos propuestos por el usuario: comprobar si el plan de vuelo es seguro, entradas de la aplicación (DEM y fichero de texto con el *flight plan*), trayectorias del avión (*Track-to-Fix*, *Radius-to-Fix*), márgenes de seguridad (lateral y vertical), las coordenadas (geográficas y cartesianas, y su resolución) y la salida de la aplicación (altura mínima).

La realización de la aplicación se ha podido llevar a cabo cumpliendo la planificación previa del trabajo y satisfaciendo cada uno de los requisitos derivados del análisis.

En el apartado de las entradas de la aplicación y sus características, se produjo un cambio ya que el fichero de entrada adoptó el formato de un fichero de texto y no de un XML, por orden del director del trabajo. Este hecho produjo un cambio en el plan de trabajo inicial.

Durante el desarrollo de los algoritmos de la línea recta y la circunferencia se presentaron problemas, por una parte, diferenciar que se trataba de un sistema discreto y no continuo, y por otra parte, en el análisis matemático que suponía conocer las características de la geometría del plano.

Una vez realizados y verificados los cinco algoritmos principales (*Track-to-Fix*, *Radius-to-Fix*, márgenes de seguridad, altura mínima y cambio de coordenadas), se llevó a cabo el desarrollo de la aplicación juntando las variables de entrada y los algoritmos anteriores. La aplicación verifica el *flight plan* e informa al usuario de la altura mínima que el avión ha adquirido durante todo el plan de vuelo.

También, se diseñó un *flight plan* alternativo, válido para un *Track-to-Fix*, que varia el rumbo del avión según el sentido de la trayectoria cuando detecta la presencia de un obstáculo. Este algoritmo es una primera aportación para futuras implementaciones relacionadas con la posibilidad de obtener un nuevo *flight plan*.

Respecto al impacto medioambiental, este proyecto en sí mismo no repercutiría a dañar el medio ambiente ya que se trata del diseño de un *software*. En cambio, esta aplicación está pensada para un UAV. Al ser un vehículo aéreo no tripulado es más ligero comparado con un avión tripulado (no hay piloto ni cabina) lo que permite un menor gasto de energía en la operación y de materias primas en la fabricación del aparato. Además, dicho UAV podría ser muy útil para ayudar a preservar el medio ambiente, ya que una posible misión del avión podría ser informar visualmente sobre el foco de un incendio. Esta ayuda favorecería la intervención de las autoridades en el lugar exacto del incendio y permitiría una extinción del fuego más rápida o controlarlo de forma más eficiente.

BIBLIOGRAFÍA

[1] *Minimum Operational Performance Standards for Required Navigation Performance for Area Navigation*, 1828 L Street, NW, Suite 805, Washington DC, October 28, 2003.

[2] http://www.geoportal-idec.net/geoportal/IDECServlet?pag=cataleg&home=s

[3] http://www.thedigitalmap.com/EasyDEM/download/ayuda/html/entendiendo/MDE.htm

[4] http://www.icc.es

[5] http://es.wikipedia.org/wiki/Sistema_de_posicionamiento_global

[6] http://es.wikipedia.org/wiki/Coordenadas_UTM

[7]http://www.elgps.com/documentos/utm/coordenadas_utm.html

[8] http://www.icc.cat/web/content/php/geotex/geoutm.php