UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Estudio teórico, simulación y validación de un sistema GBAS para el aeropuerto de Asturias

Trabajo Fin de Grado

Grado en Ingeniería Aeroespacial

AUTOR/A: Diago Iranzo, Albert

Tutor/a: Quintanilla García, Israel

CURSO ACADÉMICO: 2021/2022

Universitat Politècnica de València
Escuela Técnica Superior de Ingeniería del Diseño

Grado en Ingeniería Aeroespacial
Trabajo de Fin de Grado

Estudio Teórico, Simulación y Validación de un Sistema GBAS para el Aeropuerto de Asturias

Autor:

Albert Diago Iranzo .....

Tutor:

Dr. Israel Quintanilla García .....

Curso Académico 2021/2022

Resumen

El presente Trabajo de Fin de Grado, titulado *"Estudio teórico, Simulación y Validación de un sistema GBAS para el Aeropuerto de Asturias"*, tiene por objetivo ofrecer un análisis de este sistema de aumentación de la señal GNSS desde diversas perspectivas, las cuales se identifican en el documento a través de las partes. Estas son: aportar un marco teórico y contextual del sistema, abordar los diferentes factores físicos que intervienen en su operación, así como su implementación real.

En el primer bloque, se introducen los diferentes sistemas de aumentación de la señal GNSS dentro del contexto de la navegación aérea, haciendo énfasis en el sistema GBAS. Dentro de este análisis, se detallan sus características, tanto a nivel de prestaciones como de infraestructuras, funcionamiento y regulación. También se listan las ventajas y limitaciones de estos sistemas frente a las soluciones actuales, como puede ser el ILS.

El segundo bloque desarrolla la simulación del sistema en el entorno de MATLAB, haciendo uso también del simulador de vuelo X-Plane. El objetivo de este es la adquisición de datos verosímiles de trayectorias en un entorno completamente controlado, para después poder procesar y analizar. Se detallan los distintos componentes creados para simular cada uno de los actores involucrados, su física, y las suposiciones y simplificaciones tomadas.

Abstract

The scope of this Thesis, titled *"Theoretical Study, Simulation and Validation of a GBAS system for the Airport of Asturias"* is to offer an analysis of this augmentation system of the GNSS signal from different standpoints, which are identified by the different parts of the document. These are: to present a theoretical frame and contextualize this system in the aerial navigation, to cover the different phisical factors that are involved in its operation and its implementation in a real airport.

The first part introduces the different augmentation systems of the GNSS signal in the context of the aerial navigation, emphasazing on the GBAS system. Throughout this analysis, its characteristics in terms of performance, infrastructure and regulation are described. Furthermore, its advantages and limitations with respect to the legacy alternatives, such as the ILS, are listed.

The development of the simulation of the system is placed at the second part of the thesis. It has been implemented inside the MATLAB environment, with the additional data provided by the X-Plane flight simulator. The purpose of using this software is to adquired realistic trajectory data from aircraft in a fully controlled environment to further process and analyze. The different components that have been created for the simulation are explained as well as their physics, logic and simplifications.

Resum

El present Treball de Fi de Grau, titulat *"Estudi teòric, Simulació i Validació d'un sistema GBAS per a l'Aeroport d'Astúries"*, té com a objectiu oferir un anàlisi d'aquest sistema d'aumentació de la senyal GNSS des de diverses perspectives, les cuals es poden identificar al document a través de les parts. Aquestes son: aportar un marc teòric i contextual del sistema, abordar els diferents factors físics que intervenen en la seua operació, així com la seua implementació real.

A la primera part s'introdueixen els diferents sistemes d'aumentació de la senyal GNSS dins del context de la navegació aérea, enfatitzant en el sistema GBAS. Dins d'aquest anàlisi, es detallen les seues característiques, tant a nivell de prestacions com d'infraestructures, funcionament i regulació. També es listen els diferents avantatges i inconnvenients del sistema comparats amb diferents solucions actuals, com pot ser l'ILS.

La segona part desenvolupa la simulació del sistema en el entorn de MATLAB, fent ús del simulador de vol X-Plane. L'objectiu d'aquest es l'adquisició de dades verosímils de trajectories en un entorn completament controlat, per a ser processades i analitzades posteriorment. Es detallen els diferents components que integren la simulació per als diferents actors involucrats, les seues físiques, lògica i simplificacions tomades.

Agradecimientos

Este Trabajo de Fin de Grado pone el broche final a mi recorrido en el Grado, una etapa preciosa de mi vida. Por eso me gustaría agradecer a todo el mundo que ha hecho posible que llegue hasta aquí.

En primer lugar a mi familia, por apoyarme siempre y darme ánimos cuando más los necesito. Pasar gran parte de la carrera en mitad de una pandemia no ha sido la mejor experiencia pero gracias a ellos ha sido más llevadero.

A mis amigas y amigos de siempre (Ruth, Indra, Lorena(s), Paula, Laura, Maribel, Noelia, Carmen, Naomi ...), por inspirarme cada día a ser mejor persona, iluminar los días nublados y darme los mejores momentos que he tenido el placer de vivir. Un besazo, espero poder seguir en nuestras vidas siempre.

En especial también me gustaría agradecérselo a mi grandísimo compañero y amigo Toni, que ha estado siempre conmigo durante estos cuatro años y es en gran medida responsable de que me lleve un buen recuerdo de ellos. Eres un encanto y siempre he disfrutado mucho del tiempo que he estado contigo, tanto fuera (especialmente) como dentro de la uni.

También quería agradecérselo a los geniales profesores que he tenido a lo largo de mi vida académica (Fernando, Teresa, Juanvi, Juanjo...), por formarme tanto dentro como fuera de las aulas y ser un ejemplo para mí. No podría olvidarme de agradecérselo a María Jesús, por transmitirme su amor por la ingeniería y su ambición por aprender por los medios que sea.

Un abrazo enorme para todos, sois una gran parte de mi vida y no hubiera podido llegar hasta aquí sin vosotros. Muchas gracias.

Albert

Valencia, 2022

Índice general

Lista de Figuras

Lista de Tablas

Siglas

ABAS Aircraft Based Augmentation System 23, 24, 41, 54, 67

ADF Automatic Direction Finder 4

AM Amplitude Modulation 3, 6

ATC Air Traffic Control 3, 17, 20, 37

ATM Air Traffic Management 19

ATS Air Traffic Services 17

DME Distance Measurement Equipment 4, 5, 17, 18, 21

EGNOS European Geostationary Navigation Overlay Service 24–26, 57

FAA Federal Aviation Administration 19, 23, 27, 31

FAS Final Approach Segment 33, 34, 36

FDE Fault Detection and Exclusion 24

FM Frequency Modulation 6

GAST GBAS Approach Service Type 31, 32

GBAS Ground Based Augmentation System 8, 9, 13, 23, 24, 26–31, 33–41, 43–46, 59, 60, 67, 70, 73, 74, 76, 79, 80, 85

GDOP Geometric Dillution of Precision 13

GIMOS GNSS Interference Monitoring System 41

GLONASS Global'naya Navigatsionnaya Sputnikovaya Sistema 10, 15, 73

GLS GBAS Landing System 33–35, 38, 43

GNSS Global Navigation Satellite System 8–15, 17, 18, 22–32, 34, 35, 38, 39, 41, 43, 46, 47, 53, 55–59, 63, 68, 73, 79, 84

GPS Global Positioning System 10–12, 14, 15, 17, 25, 27, 28, 38, 55, 57, 73

GRAS Ground-based Regional Augmentation System 27, 33

GUI Graphic User Interface 47

IGM Independent GBAS Monitoring System 41

ILS Intrumental Landing System 7, 33, 35–39

IRNSS Indian Regional Navigation Satellite System 17

MMR Multi-Mode Receiver 34

NDB Non-Directional Beacon 3–5

NextGen Next Generation Air Transportation System 19

NORAD North American Aerospace Defense Command 44, 48

NPA Non-Precision Approach 22, 24, 25

NSE Navigation System Error 41

OACI Organización de Aviación Civil Internacional 9, 11, 16, 17, 19, 21, 22, 25, 28, 32, 39, 40

PBN Performance Based Navigation 17–19, 21

PFD Primary Flight Display 35, 38

PRC Pseudo-Range Correction 28, 30–33, 70

PRN Pseudo-Random Noise 12, 15, 56

QZSS Quasi Zenit Satellite System 16, 17

R-NAV Area Navigation 18–22

RAAN Ascensión Recta del Nodo Ascendente 51, 54

RAIM Receiver Autonomous Integrity Monitoring 23

RNP Required Navigation Performance 18–22

RRC Range Rate Correction 33

RVR Runway Visual Range 7

SARPs Standards And Recommended Practices 36

SBAS Satellite Based Augmentation System 16, 23, 24, 26–29, 34, 38, 80

SESAR Single European Sky ATM Research 19

SPG4 Simplified General Perturbation 4 48

TDMA Time Division Multiple Access 32

TEC Total Electron Content 57

TECU Total Electron Content Unit 57

TLE Two-Line Elements 48–53, 73

TSE Total System Error 20

VDB VHF Data Broadcast 30–32, 35

VOR Very High Frequency Omnidirectional Range 5–7, 17, 18, 35

Parte I

Sistemas de aumentación GNSS en la navegación aérea

Capítulo 1

Introducción y antecedentes

La navegación aérea siempre ha sido una de las principales cuestiones a resolver en el ámbito de la aeronáutica. Desde sus comienzos, cuando las técnicas de navegación estaban basadas en el conocido dead-reckoning y en el vuelo visual, hasta los sistemas más modernos que se tratarán en este Trabajo de Fin de Grado.

Si bien antaño este problema afectaba en mayor medida a los pilotos y a su habilidad para poder seguir las rutas de forma precisa, ya que los espacios aéreos estaban poco transitados, desde que se estableció como una forma asequible de recorrer largas distancias en poco tiempo, la aviación, y en concreto la comercial, ha experimentado un crecimiento exponencial. Como se puede ver en la Figura 1.1, que muestra la evolución del tráfico en los Estados Unidos de las últimas décadas, exceptuando el parón ocasionado por la pandemia del COVID-19 de 2020, el tráfico no ha dejado de crecer [1]. Esta tendencia también se muestra en gran parte del resto de países del mundo, siendo una excepción aquellos en vías de desarrollo que todavía no cuentan con la infraestructura necesaria.

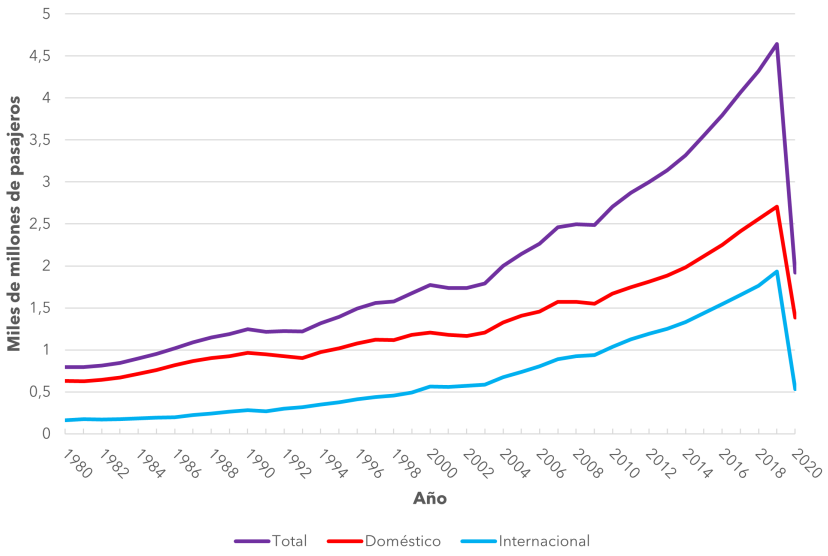


Figura 1.1: Evolución del tráfico entre 1980 y 2020 en EEUU [1]

Las necesidades del sector tanto en materia de seguimiento de trazados precisos y, sobre todo, en cuestiones de seguridad, requieren de técnicas de navegación que permitan a las aeronaves conocer su posición en todo momento con un cierto grado de precisión e integridad. De esta forma, el espacio aéreo, cada vez más congestionado, puede organizarse de forma efectiva y, en consecuencia, derivar en una mayor seguridad.

1.1. Las radioayudas convencionales

Los mayores avances en la navegación aérea civil provienen de la desclasificación de instrumentos que fueron desarrollados para fines militares durante el transcurso de la Segunda Guerra Mundial y la Guerra Fría. Las técnicas de navegación que se derivan de estos se basan, principalmente, en el uso de unas instalaciones denominadas radioayudas. Como su propio nombre indica, las radioayudas están basadas en la emisión de señales radioeléctricas con el fin de conseguir ubicar la posición relativa de la aeronave con respecto a las radioayudas y tienen asociadas unas técnicas de navegación más rudimentarias que las que se verán más adelante. Asimismo, en materia de seguridad también se implementó el uso de radares (primarios y secundarios) con fines de Air Traffic Control (ATC).

1.1.1. NDB

El Non-Directional Beacon (NDB) es radiofaro más sencillo de los principales que son utilizados. Permite conocer el track relativo de la aeronave con respecto a la estación. No obstante, si se tiene acceso a la señal de varios NDB se puede triangular la posición de la aeronave. Esencialmente esta radioayuda consiste en una antena que transmite utilizando modulación Amplitude Modulation (AM) con un tono de 400 Hz el código morse de la radioayuda. La Figura 1.2 muestra una estación NDB.

Figura 1.2: Estación NDB [Fuente: azimut.ru]

El NDB opera en las bandas LF (150-350 kHz) y MF (550-1650 kHz), aunque para navegación aérea solo se utiliza hasta 564 kHz. De esta forma, puede hacer uso del mecanismo de propagación por ondas de superficie, con el objetivo de seguir la curvatura de la Tierra y tener un alcance mayor (hasta 100 NM) [2]. Hace uso de un patrón de radiación omnidireccional, lo cual le permite dar señal en cualquier acimut alrededor de la estación, pero ocasiona un cono de silencio encima de esta. El inconveniente de usar este rango de frecuencias tan bajo es que las antenas adquieren grandes dimensiones y requieren de técnicas de acortamiento para reducir su tamaño.

Para poder hacer uso de esta señal, una aeronave cuenta con un sistema denominado Automatic Direction Finder (ADF). El ADF, en una implementación sencilla, consta de dos antenas de espira colocadas de forma perpendicular, un bloque de procesamiento de la señal, y un monitor (analógico o digital) para que pueda hacer uso de la información en la cabina. Debido a que el patrón de radiación de una antena en espira es direccional, al combinar la señal de ambas antenas se puede discernir el acimut relativo del sistema hasta la estación.

1.1.2. DME

El Distance Measurement Equipment (DME) es una radioayuda que, como su propio nombre indica, permite obtener la distancia (slant-range) entre la estación y la aeronave. No obstante, esta distancia no es en acimut, sino que se trata de la hipotenusa del triángulo rectángulo formado por la distancia horizontal de la aeronave a la estación y su altura relativa a esta. En este caso, su funcionamiento no es tan sencillo como el NDB, ya que la estación requiere de interacción con la aeronave. La Figura 1.3 muestra un ejemplo de estación DME.

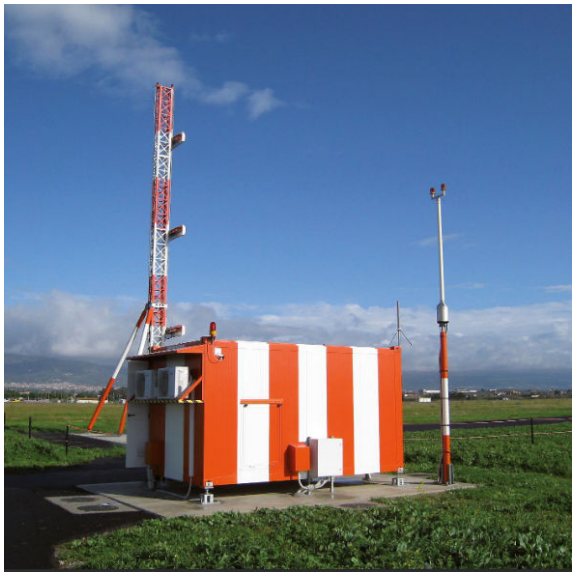


Figura 1.3: Estación DME [Fuente: aeroexpo]

El funcionamiento de una estación DME se basa en la repetición de secuencias de interrogaciones y respuestas. Estas interrogaciones vienen de forma periódica de cada una de las aeronaves que soliciten el servicio a la radioayuda y a medida que son recibidas, la estación

prepara una respuesta la cual será transmitida con un retraso de 50 μs con respecto a la entrada. De esta forma, una vez es recibida por la aeronave, esta puede correlar los retardos todas las respuestas recibidas con sus interrogaciones. De esta forma se puede calcular la distancia slant-range con la expresión de la Ecuación 1.1, donde c es la velocidad de la luz, τ_{int} el retraso que ha sido calculado entre las interrogaciones y la recepción de las respuestas y τ_{ret} son los 50 μs del transpondedor.

$$R = c \cdot \frac{\tau_{int} - \tau_{ret}}{2} \tag{1.1}$$

Para llevar a cabo este propósito, la estación DME cuenta con una única antena que opera en dos bandas: la recepción de interrogaciones tiene a sus portadoras RF en las frecuencias comprendidas entre 1025 y 1087 MHz o bien entre 1088 y 1150 MHz, a las cuales van asociadas respuestas con portadoras RF en 962-1024 MHz o bien 1151-1213 MHz, respectivamente. Con su capacidad de respuesta, un DME puede dar servicio a más de 100 aeronaves simultáneamente, asumiendo que el 95 % de estas se encuentra en seguimiento de la estación (22-30 interrogaciones/s) y el 5 % restante en búsqueda (122-150 interrogaciones/s), para dar un total de alrededor de 3600 interrogaciones por segundo. El error de distancia aproximado de esta radioayuda es menor a 0.2 NM y tiene un alcance aproximado de entre 30 y 100 NM.

1.1.3. VOR

El Very High Frequency Omnidirectional Range (VOR) y su equivalente militar, el TACAN, son radioayudas que suelen estar complementadas de un DME (en el caso del TACAN forma parte del propio sistema), a lo que se le conoce como VOR/DME. Este tipo de radioayudas en su uso con DME permiten obtener la posición de la aeronave tanto en acimut relativo como en distancia.

Si no se dispone de DME, una única estación VOR solo permite conocer el acimut relativo a la estación. Igual que con el caso del NDB, si se dispone de varios se puede conocer la posición de la aeronave en el plano horizontal. La Figura 1.4 muestra una estación VOR en su variante Doppler.

Figura 1.4: Estación VOR Doppler [Fuente: DeimosSpace]

A pesar de que entre las dos variantes del VOR (convencional y Doppler) existen diferencias en cuanto a la implementación, el principio utilizado para la recepción es el mismo. Se tienen dos señales:

- **Señal de referencia:** en el caso del VOR Doppler se emite desde el radiador central con un patrón de radiación omnidireccional. Esta señal, modulada por un tono AM 30 Hz sobre la portadora, que se encuentra entre 108 y 118 MHz (banda aeronáutica) en uno de los 200 canales habilitados.
- **Señal subportadora:** se encuentra desplazada en frecuencia 9960 Hz con respecto a la portadora. Se emite a través de las coronas secundarias del VOR Doppler (esquema omnidireccional) de forma secuencial entre ellas a un ritmo de 1800 rpm, lo que equivale a 30 Hz. De esta forma, se consigue un efecto de modulación espacial en Frequency Modulation (FM) por la rotación del diagrama.

El esquema de las señales resultantes puede verse en la Figura 1.5. Como se ha mencioando anteriormente, el VOR convencional y el Doppler a nivel de usuario son indistinguibles, a pesar de que el papel de la señal de referencia y la subportadora se alternan.

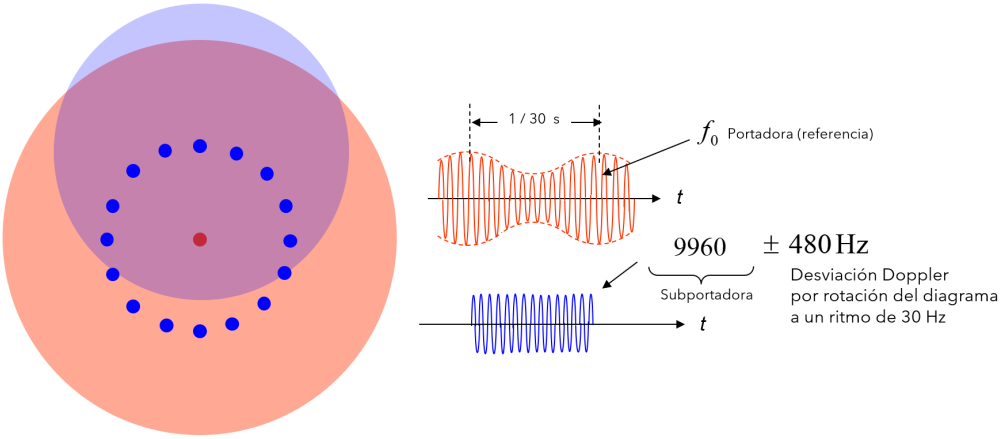


Figura 1.5: Señales generadas por el VOR Doppler [Elaboración propia]

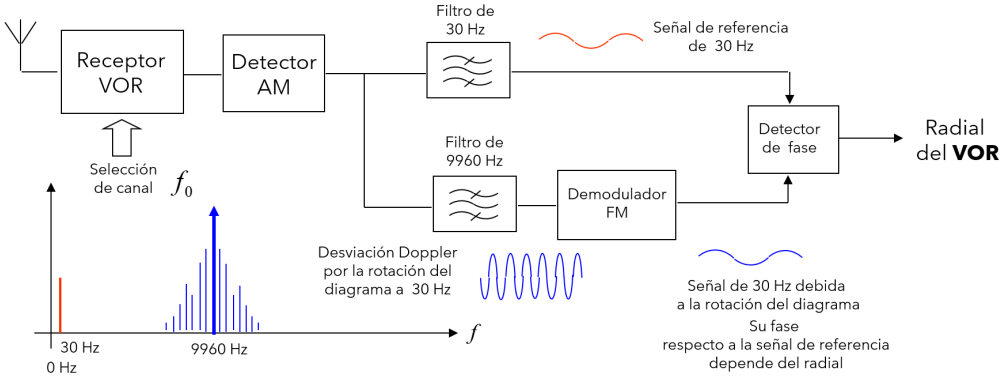


Figura 1.6: Esquema de bloques del receptor VOR de una aeronave [Elaboración propia]

El receptor de las señales VOR de la aeronave debe procesar ambas señales con el objetivo de determinar la diferencia de fase de estas, la cual está directamente relacionada con el

acimut (radial) relativo a la estación. Usando esta técnica, también se puede distinguir si la aeronave se está acercando a la estación o alejándose. Esta información se mostrará al piloto con una indicación parecida a FROM y TO, generalmente. El esquema de bloques básico de un receptor VOR puede verse en la Figura 1.5.

1.1.4. ILS

El Intrumental Landing System (ILS) es la principal radioayuda para aterrizajes en vuelos instrumentales. A pesar de que se pueden realizar otros procedimientos basados en otras radioayudas, entre las convencionales, el ILS permite una mayor precisión. La medida de precisión de los aterrizajes, categorizada por diferentes valores de CAT, está establecida por la OACI [3] y sus valores se recogen en la Tabla 1.1. La Runway Visual Range (RVR) es similar a la visibilidad, pero hace referencia a la visibilidad de las luces de pista.

Categoría de la operación	Altura de decisión (DH)	RVR	Visibilidad
CAT I	>60 m (200 ft)	>550 m	>800 m
CAT II	>30 m (100 ft)	>350 m	-
CAT IIIA	>30 m o sin DH	>200 m	-
CAT IIIB	>15 m o sin DH	>50 m	-
CAT IIIA	>30 m o sin DH	-	-

Tabla 1.1: Categorías de la aproximación de precisión

El ILS está formado por dos radioayudas: la antena de la senda de planeo (glide-slope) y el localizador (en modelo más antiguos, también se incluyen unas balizas de marcaje). Juntas, forman dos planos que determinan la trayectoria óptima de descenso para la aeronave. El receptor ILS muestra al piloto por cabina la posición de la aeronave con respecto a dichos planos. Las Figuras 1.7a y 1.7b muestran estas instalaciones.

(a) Estación del localizador

(b) Estación de la senda de planeo

Figura 1.7: Radioayudas del ILS [Fuente: hispaviacion]

La formación de los planos de referencia es similar en los casos del localizador (plano horizontal) y la senda de planeo (referencia vertical). Utilizando teoría de agrupamiento de antenas, la disposición del localizador emite hasta cinco clases de señales [4]:

- **Carrier Side Band (CSB):** se trata de la señal portadora modulada por la suma de las señales de 90 y 150 Hz. El haz de esta señal es simétrico con respecto al eje de la pista y proporcionan la referencia de fase.
- **Side Band Only (SBO):** en este caso la portadora está modulada por cada una de las señales de 90 y 150 Hz con un cierto desfase (la de 150 Hz está en fase y la de 90 en contrafase). Esta señal se emite con dos haces que forman un diagrama simétrico con respecto al eje de pista, donde se ubica un nulo, como se puede ver en la Figura 1.8.
- **Clearance (CLR):** se trata de una señal auxiliar que sirve para evitar falsos nulos y que causen que el sistema determine que está alineado con el eje de pista.
- **Señal de identificación**
- **Canal de voz**

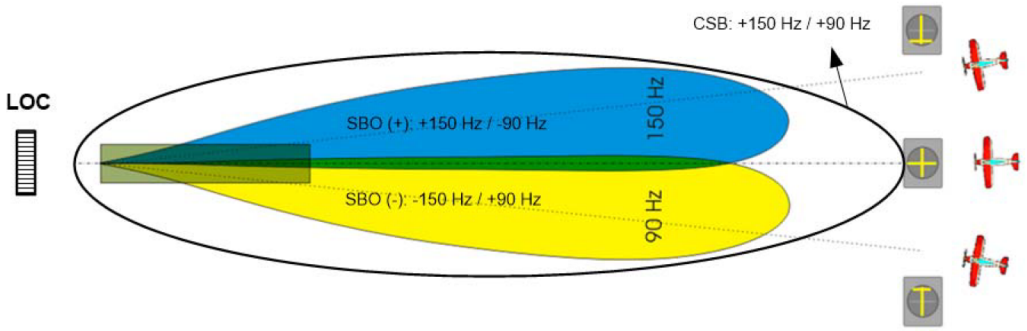


Figura 1.8: Diagrama de radiación de la señal SBO del ILS [4]

Dependiendo de la zona de los haces en la que se encuentre el avión, al comparar la señal de referencia con las SBO el receptor puede discernir cuán alejado se encuentra del eje de pista (en el que solo se recibiría la señal de referencia). Un proceso similar se aplica en la senda de planeo para formar el plano vertical de referencia.

1.2. Motivación y objetivos

El presente Trabajo de Fin de Grado tiene como objetivo estudiar en profundidad el sistema de aumentación basado en tierra, o bien Ground Based Augmentation System (GBAS), desde diferentes puntos de vista: teórico, de implementación y funcionamiento, y de regulación. En concreto, se pretende cubrir los siguientes aspectos:

- Realizar un estudio de la navegación aérea y su tecnología asociada en las últimas décadas, observar las tendencias actuales y entender la necesidad de las técnicas actuales.
- Dar una visión general de las técnicas de navegación basadas en prestaciones, o PBN, y su dependencia con las constelaciones Global Navigation Satellite System (GNSS).

- Reconocer los diferentes requisitos de la Organización de Aviación Civil Internacional (OACI) para poder utilizar sistemas de posicionamiento GNSS para la navegación aérea.
- Estudiar el sistema GBAS en su totalidad: desde los principios de funcionamiento hasta las aplicaciones que tiene.
- Discutir los diferentes elementos que intervienen en el funcionamiento de un GBAS a nivel físico y el proceso de monitorización de este a través de la simulación en el entorno de Matlab y X-Plane.

Capítulo 2

Tendencias actuales en la navegación aérea

Dentro del contexto del incipiente aumento de la demanda de tráfico comercial, el énfasis en la optimización de costes en la elaboración de rutas y la mayor concienciación del impacto medioambiental de la aeronáutica, con la llegada del nuevo milenio y el avance en la tecnología, comienzan a surgir nuevas técnicas de navegación aérea que permiten reducir la huella de las operaciones, mientras se realizan de una forma más segura, organizada y fiable. Estos avances vienen en parte fomentados por la incorporación del uso civil de los sistemas GNSS como es el caso del GPS.

En este capítulo se estudiará el funcionamiento de estos sistemas con sus técnicas de aumentación, y el nuevo paradigma de la navegación que permite hacer un uso más homogéneo de los sistemas convencionales así como la navegación basada en GNSS.

2.1. Global Navigation Satellite Systems (GNSS)

Los sitemas globales de navegación satelital (del inglés, GNSS), se originaron en la época de la Guerra Fría con propósitos militares. Con el tiempo, estos proyectos se fueron desclasificando y con la llegada del nuevo milenio se comenzaron a incorporar con aplicaciones tanto civiles como militares, aunque se verán más adelante casos en los que el sistema es completamente civil.

GNSS es el nombre genérico con el cual se engloban todas las constelaciones de satélites que permiten obtener un posicionamiento geoespacial continuado en el tiempo con una cobertura global (existen sistemas regionales que no cuentan con esta característica) [5]. Atendiendo a su evolución temporal, podemos distinguir tres fases en los sistemas GNSS:

- **Fase 0, partida:** se incluyen dentro de esta fase los primeros sistemas de origen militar creados en la década de 1970, que son el Global Positioning System (GPS) y el Global'naya Navigatsionnaya Sputnikovaya Sistema (GLONASS). Actualmente ambos sistemas tienen aplicaciones tanto civiles como militares (siendo las militares

de mayores prestaciones que las civiles).

Su uso está condicionado, en el caso del GPS por el ejército de los EEUU, por lo cual en cualquier momento, en caso de conflicto o cualquier problema, pueden cortar el servicio civil o hacerlo inservible utilizando la disponibilidad selectiva. Este mecanismo introduce un error sintético en la señal GPS que hace que el error generado en el receptor sea demasiado elevado como para ser utilizado para navegación. La disponibilidad selectiva se desactivó el 2 de mayo del 2000 bajo el mandato de Bill Clinton.

- **Fase 1, GNSS-1 (2002-2015):** esta fase se comienza a iniciar con la desactivación de la disponibilidad selectiva, lo cual permitió el uso de estos sistemas para aplicaciones civiles. En esta fase surgen también los sistemas de aumentación de la señal GNSS que se tratarán más adelante (ejemplos de implementación de estos sistemas son EGNOS, WAAS, etc). Estos permiten que se utilicen los sistemas GNSS para la navegación aérea, ya que sí que cumplen con los requerimientos establecidos por la OACI en materia de precisión, integridad, continuidad y disponibilidad, los cuales se detallan en la el Apartado 2.2.
- **Fase 2, GNSS-2 (2015-presente):** esta fase se caracteriza por la inclusión de los nuevos sistemas GALILEO (Unión Europea) y BeiDou (China), los cuales introducen un mayor número de señales más potentes y seguras, y están orientados a garantizar una serie de servicios en tiempo real y de operaciones críticas como el salvamento. En esta fase también se incluyen las actualizaciones de los sistemas anteriormente citados.

2.1.1. Componentes de los sistemas y funcionamiento

Un sistema GNSS está compuesto por tres segmentos principales [6]:

- **Segmento espacial:** dentro de este componente se encuentran las constelaciones de satélites de las cuales hace uso el sistema. En el caso de los sistemas globales, normalmente las constelaciones están formadas por un total de entre 24 y 28 satélites ubicados en tres planos orbitales de órbita media (MEO), a unos 20000-30000 km de altura. Estos transmiten la (o las señales) GNSS de medición en diferentes bandas de frecuencia dentro del espectro de las microondas.
- **Segmento de control:** es el encargado de monitorear y mantener la salir del sistema a través del control de las señales retransmitidas y proporcionando al segmento espacial los mensajes y datos de navegación. Está compuesto por uno o diversos centros de control que cuentan con relojes atómicos para poder medir las desviaciones del tiempo GNSS; así como un conjunto de estaciones de vigilancia distribuidas alrededor de la Tierra, las cuales cuentan con antenas que sirven de enlace entre ambos segmentos.
- **Segmento de usuario:** está compuesto por los receptores utilizados de forma global para hacer uso del sistema GNSS. Estos receptores, atendiendo a su uso pueden ser militares o civiles (normalmente con prestaciones reducidas si el sistema es principalmente de uso militar).

En cuanto a su funcionamiento, los sistemas GNSS dependen principalmente de un factor: el tiempo. Los satélites que conforman el segmento espacial del sistema cuentan con relojes atómicos para poder medir el tiempo de la forma más precisa posible (dentro de las restricciones de tamaño y el entorno de operación del satélite). En conjunto se establece una medida de tiempo más precisa que la que suele utilizarse en tierra: el conocido como tiempo GNSS[1]. No obstante, no se debe confundir el tiempo GNSS con el tiempo que marcan los relojes de los satélites. Estos presentan una cierta deriva, conocido como *bias*, con respecto a este tiempo. Esta deriva se conoce de forma precisa, como se ha comentado anteriormente, a través de los centros de control.

Así, la función principal de los satélites del segmento espacial es la de transmitir los mensajes de navegación, que suelen incluir el instante de tiempo en el que se transmite el mensaje medido con el reloj del satélite, la posición del satélite en cuestión, unos coeficientes que permiten corregir el *bias* del satélite con gran precisión, así como otros datos de correcciones, como pueden ser las del modelo ionosférico[2].

Esta información, modulada sobre una portadora en el orden de 1 GHz, viaja por el espacio hasta llegar a la zona de cobertura de cada satélite, donde es captada por los receptores del segmento de usuario. Dependiendo de si se trata de una señal de altas prestaciones (normalmente de uso militar) el receptor deberá realizar un proceso de decodificación, pero en el caso general se presentan varios modos de mediación:

- **Medidas de pseudodistancia simples:** este es el método de medición más común. Consiste en medir el retraso entre la emisión de la señal GNSS y la recepción. Sabiendo la velocidad de la luz en el vacío se puede transformar en una medida de unidades de distancia. La Ecuación 2.1 describe este proceso:

$$\rho = c \cdot \left(t + \Delta t_{iono} + \Delta t_{tropo} + \Delta t_{bias}^{sat} + \Delta t_{bias}^{user}\right) \tag{2.1}$$

  Donde se pueden apreciar las distintas fuentes de error de este tipo de medidas. Δt_{iono} es el retraso que sufre la señal en la ionosfera, como es el caso de Δt_{tropo}. En cambio, los últimos dos factores son las desviaciones de los relojes de usuario y satélite con respecto al tiempo GNSS. En el caso de la deriva del satélite, esta se corrige con los coeficientes mencionados anteriormente, pero el reloj del usuario también introduce un error desconocido y mayor que el del satélite al tratarse de un reloj más simple.

  Para realizar la medida del retraso en el envío de la señal, se hace uso del Pseudo-Random Noise (PRN), un código transmitido en la señal GNSS [7]. Este código es conocido por el receptor y sabe a qué instante de tiempo pertenece, por lo que puede aplicar un cierto retardo en el procesado de la señal hasta hacer que coincida. El retardo necesario para que coincida con la señal es precisamente el t medido por el sistema, previo a correcciones por los fenómenos mencionados. La Figura 2.1 muestra este proceso.

  Una vez son conocidas las pseudodistancias de los satélites al receptor, con un mínimo de cuatro satélites se puede triangular la posición del receptor. El error cometido por

[1] Este es un término general, cada constelación utiliza un nombre, por ejemplo el tiempo GPS.

[2] Cada sistema tiene una estructura de mensajes distinta y que puede incluir diferentes tipos de información, como pueden ser los datos de integridad.

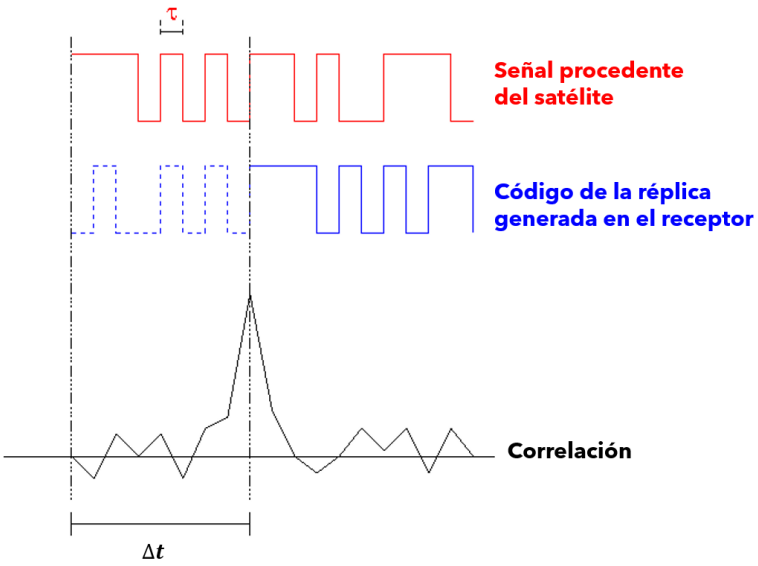


Figura 2.1: Proceso de determinación del retraso de la señal en el receptor [Elaboración propia]

este método de medida es el más alto de los que se discuten. Depende de la geometría de los satélites (normalmente medido a través de una medida denominada Geometric Dillution of Precision (GDOP)). La Figura 2.2 muestra el esquema de funcionamiento de este tipo de medidas.

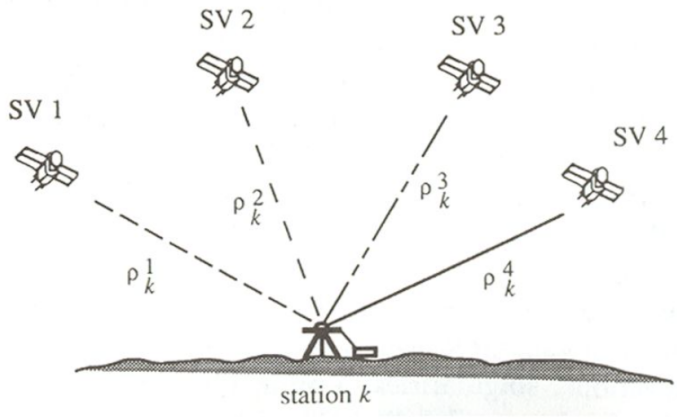


Figura 2.2: Toma de medidas de pseudodistancia en una estación estática [7]

- **Medidas de fase:** este es un método que no suele emplearse en navegación aérea, sino más bien para fines topográficos. El motivo principal es que este método de medida matemáticamente no permite el posicionamiento cinemático de un receptor cuando se toman medidas de fase simples [5]. Se requiere de medidas de fase diferenciales (estación de referencia). Este tipo de soluciones se plantea incorporarlo en los sistemas de aumentación GNSS como parte del sistema GBAS.

  En lugar de medir un retraso, como su propio nombre indica, en este caso se mide la fase de la señal GNSS a través de un seguidor de fase en el receptor. La ventaja

de utilizar técnicas de medida basadas en medidas de fase es que son más precisas que las de pseudodistancia ya que se puede seguir la fase con un mayor grado de precisión, del orden de una décima parte de un ciclo. No obstante, en este caso se requiere de determinar el número de ciclos contenidos en la distancia que separa cada satélite con el receptor, lo cual requiere de un tiempo del orden de un minuto o bien de tener un sistema de medida de pseudodistancia de apoyo para poder estimar este valor de forma prácticamente instantánea. La Figura 2.3 muestra gráficamente este problema [8].

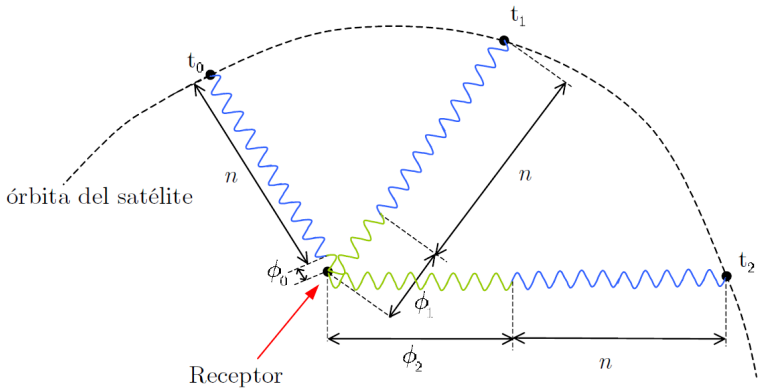


Figura 2.3: Toma de medidas de fase en una estación estática [8]

- **Medidas diferenciales:** este método de medida engloba a las medidas diferenciales de pseudodistancia y a las medidas diferenciales de fase. No obstante, ya que este método requiere de una estación de referencia, será tratado en mayor profundidad en el Apartado 2.3

2.1.2. Sistemas globales

En esta sección se hará un repaso general de los sistemas GNSS desplegados actualmente, estableciendo sus características principales así como los servicios que ofrecen. Existen cuatro sistemas GNSS:

- **GPS - NAVSTAR:** este sistema comenzó a operar en 1978 como parte del programa NAVSTAR. A lo largo de los años, la constelación ha ido actualizándose en los denominados Bloques (I, II, IIA, IIR, IIR-M, IIF y III), cada uno con sus peculiaridades y mejoras. Principalmente con el avance de los bloques se aumenta la capacidad de los paneles solares, pasando de 710 W por panel en el Bloque I a 4480 W en el III y se aumenta la esperanza de vida del satélite (de 7.5 a 15 años). Asimismo, también se elaboran nuevas señales que proporcionan nuevos servicios y se aumenta la precisión de los relojes de los satélites. Algunas de sus características son [9]:

  - Constelación nominal formada por 32 satélites distribuidos en 6 órbitas circulares (excentricidad inferior a 0.02) con una inclinación de 55º. Se encuentran ubicadas a unos 20200 km de altura y cuentan con un periodo de 11 horas y 58

minutos, para aproximarse a medio día sidéreo[3]. Actualmente solo se encuentran operativos 31, el restante se encuentra en mantenimiento.

- Cada satélite cuenta con cuatro relojes atómicos internos, todos con una precisión mayor del nanosegundo. Se encuentran dos tipos de relojes: los de cesio, con una precisión de $1 \cdot 10^{-13}$ s y los de rubidio, de $1 \cdot 10^{-14}$ s, siendo estos últimos los principales. Los osciladores de estos relojes generan la frecuencia fundamental que se utilizará para formar las diferentes señales del sistema.

- Las señales que se generan contienen tres componentes: la portadora, el código PRN y el mensaje de navegación. Actualmente se transmiten en tres bandas: la L1, cuya frecuencia central se encuentra en 1575.42 MHz (154 veces la frecuencia fundamental), en la cual modula el código C/A (civil) y se cifra el P (precisión, militar). Por otro lado, la L2, a 1227.60 MHz y la L5, a 1176.45 MHz. Esta última es más moderna y transmite mensajes de navegación civil (CNAV).

- Los servicios que ofrece el sistema GPS son Standard Positioning Service (SPS, disponible para todos los consumidores); Precise Positioning Service (PPS, para consumidores autorizados); Precise Point Positioning (PPP, en desarrollo, más preciso); y GPS diferencial, con estaciones de referencia.

- **GLONASS:** es la respuesta rusa al sistema GPS estadounidense. Se comenzó a desarrollar en la Unión Soviética en 1976 y su primer satélite operativo entró en funcionamiento en 1982. No obstante, con la caída de la URSS se abandonó temporalmente el programa hasta principios del nuevo milenio. Algunas de sus características principales son:

  - La constelación nominal de GLONASS está compuesta por 24 satélites distribuidos en 3 planos orbitales con una inclinación de 64.8^{o} para dar mejor cobertura a las zonas de mayor latitud, entre las cuales se encuentra Rusia. La altura de los satélites es de 19100 km. Para garantizar total cobertura de la Federación Rusa precisa de 18 satélites, y 24 permiten cobertura global. Del mismo modo que el sistema GPS, el GLONASS, al ser un programa con un mayor recorrido, también se ha actualizado en Bloques (GLONASS-M y GLONASS-K1 y GLONASS-K2 en desarrollo).

  - El sistema GLONASS transmite cuatro señales ubicadas en dos bandas, la L1 y la L2, pero separadas de sus equivalentes estadounidenses en 0.5625 y 0.4375 MHz, respectivamente. Otra característica similar al GPS es el uso que se hace de estas frecuencias, ya que en la L1 se incluye el código C/A y P, mientras que en la L2 solo el P. Actualmente también se ha añadido una tercera banda, la L3 a 1201 MHz, aunque todavía está en fase de pruebas.

- **GALILEO:** se trata del sistema GNSS de la Unión Europea. Proporciona un posicionamiento global preciso y garantizado, ya que a diferencia de otras constelaciones, GALILEO está bajo control civil y será totalmente compatible e interoperable con otras constelaciones, como pueden ser GPS o GLONASS. Está íntimamente relacionado con otro proyecto de la Unión denominado EGNOS, que se trata de un sistema

[3]Un día sidéreo hace referencia al periodo de rotación de la Tierra sobre sí misma con respecto a un sistema de coordenadas fijo, no respecto al Sol, como es el caso de los días naturales.

de aumentación que se tratará más adelante. El objetivo de GALILEO es ofrecer un sistema que sea capaz de satisfacer con los requisitos de la OACI para poder ser utilizado para navegación aérea, así como cubrir los diferentes vacíos que dejan el resto de constelaciones (no tienen garantías legales de funcionamiento, responsabilidades en caso de fallo no claras). Algunas de sus características son [10]:

- La constelación nominal estará formada por 30 satélites (27 de los cuales serán de medida y 3 geoestacionarios de observación). Las órbitas tienen una inclinación de 56^{o} y se ubican a 23222 km de altura, con una excentricidad media de 0.002 (prácticamente circular).

- Cuentan también con cuatro relojes atómicos, dos pasivos de hidrógeno y dos de rubidio. GALILEO transmite 11 señales en 5 bandas, cuya nomenclatura principal es E1 (banda de la L1), E5 (banda de la L5) y E6 (cercana a la L2 en frecuencia).

- Cada una de dichas señales está orientada a ofrecer un tipo determinado de servicio: Servicio Abiero (OS), proporciona datos de posicionamiento y sincronización; Servicio Comercial (CS), tiene mejores prestaciones que el abierto y su señal se encuentra codificada; Servicio de Salvamento de Vida (SoL), este servicio se encarga de operaciones críticas como pueden ser la navegación civil y marítima, cuenta con sistema de integridad; Servivio de búsqueda y Salvamento (SARS), permite a los operadores responder de forma más rápida a las llamadas de socorro; y por último el Servicio Público Regulado (SRS), se encuentra encriptado solo para usuarios autorizados con fines de defensa, entre otros.

- **BeiDou - COMPASS:** es el sistema global chino originado en 1997. En 2020 completó su constelación y en 2015 comenzó a estar operativo en diferentes países asiáticos. Algunas de sus características son:

  - Su constelación nominal está formada por 35 satélites, de los cuales 27 son BeiDou-M localizados en tres planos orbitales de inclinación 55^{o} y una altura de 21528 km. El resto de ellos son geoestacionarios, de forma similar al sistema GALILEO.

  - Transmite señales en tres bandas, la B1 (1561.098 MHz), la B1-2 (1589.742 MHz), la B2 (1207.14 MHz) y la B3 (1268.52 MHz). Ofrecen tanto servicios civiles como militares.

2.1.3. Sistemas regionales

Estos sistemas se diferencian de los globales, como su propio nombre indica, en que no dan cobertura global para la navegación, si no que se centran en las áreas geográficas de los países que los desarrollan. Estos son:

- **Quasi Zenit Satellite System (QZSS):** se trata de un sistema regional japonés, aunque principalmente actúa como sistema de aumentación de tipo Satellite Based Augmentation System (SBAS), de forma similar al sistema EGNOS en Europa. Su

segmento espacial está compuesto por 3 satélites geosíncronos de órbita inclinada y un cuarto geoestacionario. Se plantea como un sistema complementario al GPS y también será compatible con GALILEO. Transmite seis señales, similares a las de otros sistemas para aumentar su compatibilidad: L1-C/A, L1C, L2C, L5, L1-SAIF y LEX [11].

- **Indian Regional Navigation Satellite System (IRNSS)-NAVIC:** el sistema IRNSS es propiedad del gobierno índio y tiene como objetivo fundamental proporcionar un sistema de posicionamiento independiente para no quedarse aislado en caso de conflicto bélico. Da cobertura a la zona de la Índia y aproximadamente 1500 km alrededor suyo. De forma similar al QZSS, cuenta con 7 satélites, 3 geoestacionarios y 4 geosíncronos con órbita inclinada. Trabaja con dos señales, la L5 (misma banda que las homólogas) y la S (2492.028 MHz). Da servicio tanto civil como militar.

2.2. Performance Based Navigation (PBN)

Tal y como se estableció anteriormente, el crecimiento sostenido de la aviación requiere de un mayor aprovechamiento del espacio aéreo disponible, con el fin de aumentar la seguridad operacional así como reducir la carga de trabajo de los controladores aéreos y reducir el impacto de las operaciones en materia de ruido y huella de carbono. Bajo estas premisas y apoyado por la desactivación de la Disponibilidad Selectiva del sistema GPS, en 2003 se comienza a elaborar una nueva filosofía para la creación de rutas aéreas [12].

Esta es precisamente el Performance Based Navigation (PBN), definida por la OACI en su Doc. 9613 [13] como la navegación de área basada en los requerimientos de prestaciones de las aeronaves operando en una ruta Air Traffic Services (ATS), ya sea en aproximación por instrumentos o bien en el espacio aéreo designado. Aquí yace la diferencia fundamental que separa el PBN de la navegación convencional. Como se ha visto en el Capítulo 1, entre las radioayudas convencionales existen muchas diferencias en cuanto a su funcionamiento y especialmente en cuanto a las técnicas para navegar utilizándolas.

Por ejemplo, para seguir una aerovía se puede estar siguiendo un radial VOR, pero no se utilizarían tres DME a pesar de que se pudiera triangular la posición. Este comportamiento heterogéneo condiciona la conformación del espacio aéreo, haciendo que las rutas establecidas por ATS estén basadas en la posición de las radioayudas. También condiciona la creación de procedimientos debido a que los franqueamientos de obstáculos o el tipo de técnica utilizada dependen en gran medida de la radioayuda utilizada. Como se puede ver en la Figura 2.4, la mayor parte de las aerovías se establecen entre radioayudas de tipo VOR, lo cual puede ocasionar en zonas congestionadas cuellos de botella debido a las aglomeraciones, que requieren de un mayor trabajo de separación por parte de ATC.

El PBN pretende cambiar este paradigma haciendo que el diseño del espacio aéreo sea dependiente de la capacidad de la aeronave de conocer su posición utilizando todos los medios que tiene disponible (sistema de radioayudas convencionales, sistemas inerciales, receptor GNSS). De esta forma, se puede tener una mayor libertad a la hora de crear rutas debido a que no se depende tanto del tipo de radioayuda sino que esa responsabilidad recae sobre la aeronave y su habilidad para seguirla. Esto se traduce a efectos prácticos en la incorporación progresiva de los denominados *waypoints* en contraposición a los *fixes*.

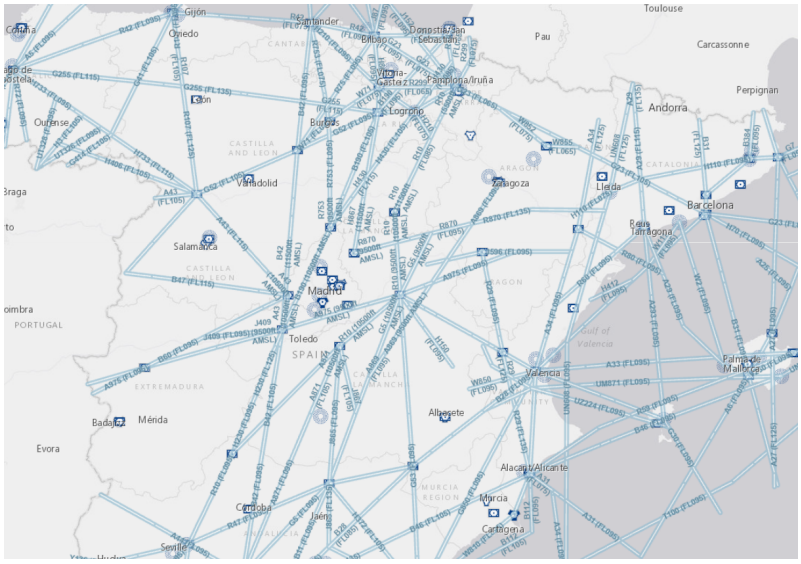


Figura 2.4: Aerovías y radioayudas en parte de España [Fuente de la figura]

Los *fixes* se definen como una intersección de valores de radioayudas, por ejemplo, un radial VOR y una distancia a un DME. En cambio, los *waypoints* se definen por coordenadas y es el sistema de la aeronave el que debe localizar este punto utilizando ya sean las propias radioayudas convencionales u otros sistemas. De esta forma, navegar un espacio aéreo con PBN se hace de una forma más homogénea, y aunque muchas veces está orientado al uso del sistema GNSS, se intenta que sea independiente de cualquier tipo de sistema, haciéndolo más versátil. La Figura 2.5 muestra la diferencia entre ambos tipos de navegación en el caso de los denominados *VOR fantasma*.

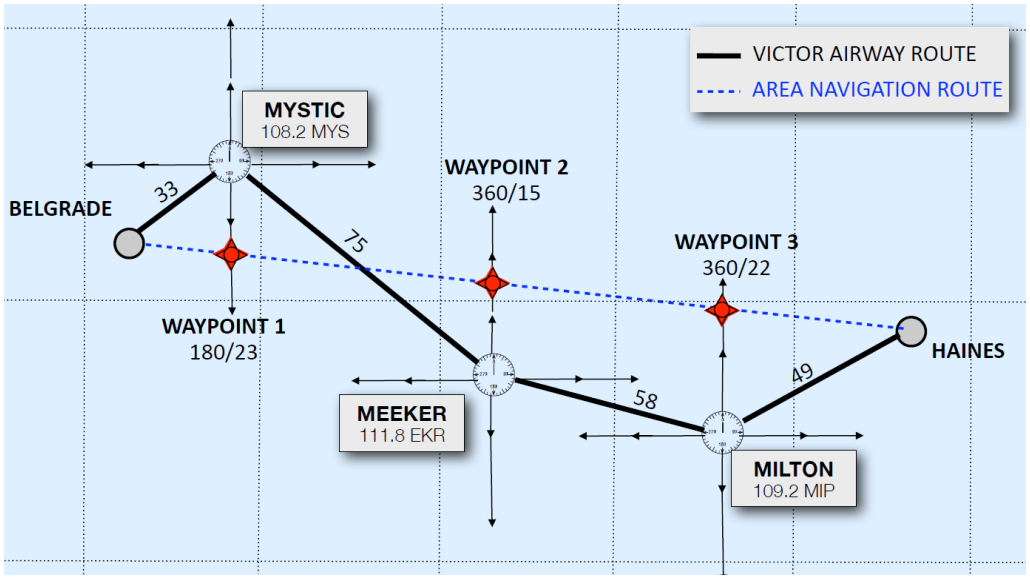


Figura 2.5: Diferencia entre esquemas de navegación convencionales y PBN [Fuente de la figura]

La navegación PBN es un concepto, pero se pueden distinguir dos implementaciones distintas: Area Navigation (R-NAV) y Required Navigation Performance (RNP) [14]. El tipo de implementación que tenga una ruta específica determinará los requerimientos específicos

de los sistemas de la aeronave. La principal diferencia entre ambas es que las rutas R-NAV no disponen de requisitos de monitoreo y alerta, mientras que las de RNP sí.

La implementación de PBN y otra serie de medidas enfocadas a la mejora de la navegación aérea y la reducción del impacto de esta se materializaron principalmente en la introducción de dos planes:

- **Single European Sky ATM Research (SESAR):** es el proyecto colaborativo europeo para renovar su espacio aéreo y su sistema de Air Traffic Management (ATM). La fase de implementación a gran escala debería haber finalizado en 2020 en el plan inicial, pero la pandemia ha alargado este proceso.

- **Next Generation Air Transportation System (NextGen):** es el equivalente estadounidense impulsado por la Federal Aviation Administration (FAA). En este caso, la fase de implementación debería estar lista para 2025.

2.2.1. Area Navigation (R-NAV)

La OACI define la navegación de área como un método de navegación que permite la operación de aeronaves en cualquier ruta aérea deseada dentro de la cobertura de ayudas a la navegación referenciadas a radioayudas o bien dentro de los límites de la capacidad de ayudas auto-contenidas, así como una combinación de ambas. Los requerimientos de un sistema capacitado para realizar operaciones R-NAV dependen del contexto de dicha operación. No pueden tener los mismos requerimientos, por ejemplo, la fase de crucero que la de aproximación.

Nav. Spec.	**Fase de vuelo**							
	OCN	**CNT**	**ARR**	**Aproximación**				**DEP**
				Inicial	**Int.**	**Final**	**Missed**	
RNAV 10 (RNP 10)	10	-	-	-	-	-	-	-
RNAV 5	-	5	5	-	-	-	-	-
RNAV 2	-	2	2	-	-	-	-	-
RNAV 1	-	1	1	1	1	-	1	1
RNP 4	4	-	-	-	-	-	-	-
RNP 2	2	2	-	-	-	-	-	-
RNP 1	-	-	1	1	1	-	1	1
Advanced RNP	2	2 o 1	1	1	1	0.3	1	1
RNP APCH	-	-	-	1	1	0.3	1	-
RNP AR APCH	-	-	-	1-0.1	1-0.1	0.3-0.1	1-0.1	-
RNP 0.3	-	0.3	0.3	0.3	0.3	-	0.3	0.3

Tabla 2.1: Requerimientos de precisión para RNP y R-NAV

Estos requerimientos, en el caso de R-NAV se materializan en una precisión determinada,

que define la probabilidad con la que la aeronave se encuentra a una cierta distancia de la posición computada por el sistema, asumiendo que el sistema no falla. La Tabla 2.1 recoge los diferentes requisitos de precisión R-NAV y RNP dependiendo de la fase de vuelo. La precisión se mide como el error total del sistema o bien Total System Error (TSE) en un 95 % del tiempo medido en millas náuticas. Esto hace referencia al peor escenario, en un caso real la precisión de estos sistemas es mucho mayor. OCN hace referencia a las rutas oceánicas y CNT a las continentales; estas son las operaciones con menores requisitos de precisión debido a que no requieren de tanto control.

Las rutas R-NAV posibilitan dos nuevas técnicas de navegación:

- **Navegación punto a punto:** son procedimientos como los representados en la Figura 2.5. Se trata de rutas introducidas en las cartas aeronáuticas en las cuales se utilizan *waypoints* colocados a conveniencia de ATC utilizando las radioayudas existentes (aunque como ya se ha visto se puede utilizar otro sistema para llegar a estos puntos). Algunos de los ejemplos de implementación de este tipo de procedimientos se dan en las llegadas de aeropuertos con gran congestión, para ayudar a la secuenciación del tráfico:
  - Point-merge: consiste en desplazar a todo el tráfico en un arco definido por una serie de *waypoints* hasta que pueda ser introducido en el punto de unión. El diagrama de una estrategia de convergencia a punto puede verse en la Figura 2.6.

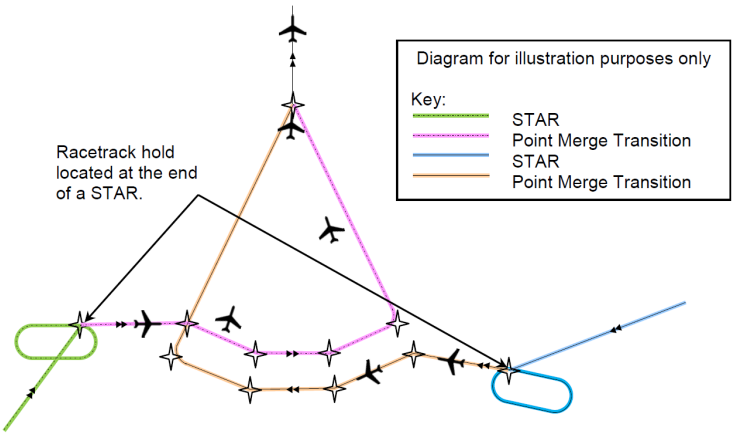
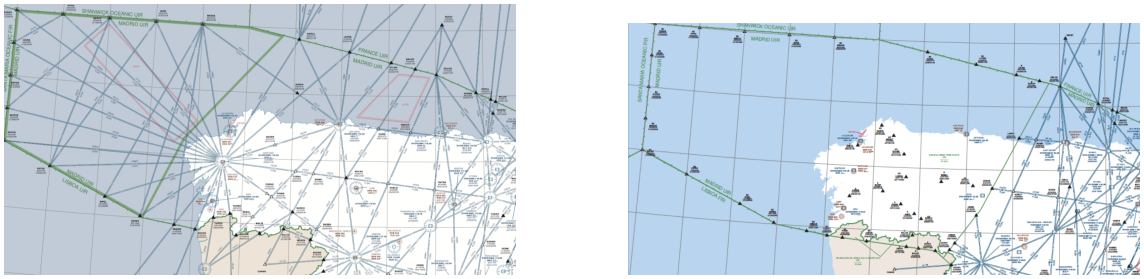


Figura 2.6: Estrategia de convergencia a punto R-NAV [Fuente: PoliformaT]

  - Estrategia de trombón: similar al point-merge, en lugar de dirigir a todo el tráfico hacia un arco, se desvía por un tramo recto en dirección contraria a la de aterrizaje y cuando se puede secuenciar el tráfico se les ordena virar hacia un *waypoint*.

- **Free-routing:** es una consecuencia de poder realizar navegación punto a punto. Este esquema permite definir una serie de *waypoints* que delimitan una zona de rutas libres. A partir de ese momento y sujeto a aprobación del plan de vuelo con la ruta a seguir (que sea libre no significa que no esté controlada), una aeronave puede seguir aerovías ficticias entre dichos puntos, o combinaciones de estas. Las Figuras 2.7a y 2.7b reflejan como cambia el espacio aéreo tras aplicar esta medida.



(a) Santiago con aerovías convencionales

(b) Santiago con free-routing

Figura 2.7: Cambio del espacio aéreo con la llegada del free-routing [Fuente: PoliformaT]

2.2.2. Required Navigation Performance (RNP)

RNP es un tipo de navegación PBN más completo y preciso en el que no solo se tiene en cuenta la habilidad del sistema para seguir una ruta determinada con un cierto nivel de precisión, sino que además debe cumplir con una serie de características que permiten al sistema cuál es el error que está cometiendo y si es fiable su medida o no. Estos son los requisitos de alarma y monitoreo de la OACI:

- **Precisión:** aplica la misma definición que la vista en el apartado anterior. Se define como la probabilidad con la que la posición calculada por el sistema de la aeronave difiera más de una cierta distancia de la posición de la aeronave real asumiendo que el sistema está libre de fallos.
- **Integridad:** se trata de la habilidad del sistema de desactivarse de forma automática o de proveer de advertencias dentro de un margen de tiempo determinado (límite de alerta, depende de la operación RNP) cuando no se deba utilizar el sistema como medio de navegación principal, ya sea por falta de datos, fallo del sistema o fallos detectados en los datos recibidos.
- **Disponibilidad:** es la habilidad del sistema para estar disponible para hacer uso de este en el momento en el que sea requerido y se expresa en porcentaje del tiempo en el cual el sistema está operando cumpliendo con todas las prestaciones requeridas.
- **Continuidad:** se refiere a la probabilidad con la que se puede dar una pérdida del servicio mientras este está en uso.

Las rutas RNP permiten una mayor precisión a la hora de establecer procedimientos, lo cual posibilita maniobras más complejas con tramos curvos o bien descensos contínuos. Los tipos de tramos que se pueden ejecutar utilizando RNP son los siguientes:

- **Tramos rectos (Fly-To):** este es el tipo de tramos que también se podían volar con un R-NAV. Se trata de dirigirse a unas coordenadas especificadas por un *waypoint*, en este caso con garantías de que si el sistema falla se emitirá un aviso.
- **Radio constante a fijo (Radius-to-fix):** a pesar de que este tipo de tramos se asemejan a aquellos que en tipo de navegación convencional se realizarían alrededor de un DME, en el caso de RNP tenemos una menor dispersión, ya que se consigue

un error menor a 0.5 NM en el 95 % del tiempo a distintas velocidades. En el caso de la navegación convencional dependería de la precisión de la radioayuda utilizada así como la del sistema de vuelo que determine el radio de giro en función de la velocidad.

- **Transiciones Fly-By y Fly-Over:** son diferentes formas de afrontar los cambios de *track* a lo largo de una ruta. En el caso del Fly-Over la transición entre ángulos de **track** se efectúa después de pasar por el *waypoint*; en el caso de los Fly-By este cambio se establece antes, lo cual optimiza la operación ya que la transición es más suave y hay un menor cambio de rumbo.
- **Advanced RNP (A-RNP):** es un clasificador especial dentro de las operaciones RNP que permite afrontar todas las fases de vuelo con diferentes valores de precisión lateral. Cuenta con una serie de funcionalidades entre las cuales se incluyen:

  - R-NAV Holding: reduce las áreas de espera para poder mantenerlas más cercanas entre ellas y así optimizar la secuenciación de tráfico entrante. Asimismo, una aeronave con este sistema puede definir un *waypoint* como inicio de una espera.

  - Offset paralelo: permite realizar una ruta ya planificada de forma paralela a la original, con el objetivo de separar la aeronave del tráfico que realice un recorrido similar sin requerir de cambios de nivel.

  - Time of Arrival Control (TOAC): permite controlar la hora de llegada a un *waypoint* determinado variando la velocidad y recorrido de la ruta, para poder evitar congestiones.

Tal y como se vio en el caso de la precisión, dependiendo del tipo de operación que se esté realizando se establecerán diferentes requerimientos en cuanto a integridad, continuidad y disponibilidad, los cuales se recogen en la Tabla 2.2 y provienen del Anexo 10 de la OACI [15]. Estos están referidos a sistemas GNSS ya que, aunque se busca la independencia del sistema, para RNP este suele ser el principal.

Tipo de operación	Precisión horizontal 95 %	Precisión vertical 95 %	Integridad	Tiempo de alerta	Continuidad	Disponibilidad
En-route	3.7 km (2.0 NM)	N/A	10^{-7}/h	5 min	10^{-4}/h - 10^{-8}/h	0.99 - 0.99999
En-route, terminal	0.74 km (0.4 NM)	N/A	10^{-7}/h	15 s	10^{-4}/h - 10^{-8}/h	0.99 - 0.99999
Aproximación inicial e intermedia, Non-Precision Approach (NPA), Despegue	220 m (720 ft)	N/A	10^{-7}/h	10 s	10^{-4}/h - 10^{-8}/h	0.99 - 0.99999
Aproximación con guiado vertical APV-I	16.0 m (52 ft)	20.0 m (66 ft)	$1-2\cdot10^{-7}$ por op.	10 s	$1-8\cdot10^{-6}$ por 15 s	0.99 - 0.99999

...

Tipo de operación	Precisión horizontal 95 %	Precisión vertical 95 %	Integridad	Tiempo de alerta	Continuidad	Disponibilidad
Aproximación con guiado vertical APV-II	16.0 m (52 ft)	8.00 m (26 ft)	$1-2\cdot10^{-7}$ por op.	6 s	$1-8\cdot10^{-6}$ por 15 s	0.99 - 0.99999
Aproximaciones de precisión de CAT I	16.0 m (52 ft)	6.0 m a 4.0 m (20 ft a 13 ft)	$1-2\cdot10^{-7}$ por op.	6 s	$1-8\cdot10^{-6}$ por 15 s	0.99 - 0.99999

Tabla 2.2: Requisitos marcados por la OACI para operaciones RNP con GNSS

2.3. Sistemas de aumentación de la señal GNSS

Como se ha visto en el Apartado 2.1, ninguno de los sistemas GNSS cumple con los requerimientos expuestos en la Tabla 2.2 debido a que generalmente se trata de sistemas de origen militar o bien porque no disponen de información de integridad. Esto hace que por lo que por sí solos, no se pueden utilizar para la navegación aérea.

En este contexto surgen los sistemas de aumentación de la señal GNSS, que pretenden cubrir estas carencias a diferentes niveles, dependiendo de las necesidades de la operación para la que estén diseñados. Atendiendo a su principio de operación y su infraestructura asociada, se pueden dividir en tres categorías: Aircraft Based Augmentation System (ABAS), SBAS y GBAS, los cuales se introducen a continuación.

2.3.1. Aircraft Based Augmentation System (ABAS)

El ABAS es el sistema de aumentación portable más sencillo de implementar y al mismo tiempo el que proporciona prestaciones más sencillas pero suficientes para cubrir su propósito. Este sistema de aumentación no mejora la calidad de la señal GNSS, si no que únicamente se encarga de cubrir los apartados que impiden usar las constelaciones GNSS para la navegación. Consiste en un receptor GNSS con la particularidad de que cumple la Technical Standard Order (TSO) C-129 de la FAA [16]. La principal consecuencia de esto es que el receptor incorpora un sistema denominado Receiver Autonomous Integrity Monitoring (RAIM). Este sistema se encarga de advertir al piloto de que la señal GNSS es errónea y por lo tanto debe dejar de usarse.

El proceso de detección de fallo de la señal se realiza realizando comparaciones de los datos almacenados de la posición y tiempo de diferentes conjuntos de 4 satélites. En el caso de que haya discrepancias entre estos, se trata porque alguno de ellos es defectuoso y por lo tanto no se puede determinar la posición con exactitud. Este esquema de funcionamiento hace que el mínimo de satélites necesarios para la operación del sistema pase de 4 (mínimo matemático para posicionamiento cinemático de una aeronave en el espacio) a 5 (para realizar subconjuntos de cuatro).

No obstante, los sistemas ABAS también pueden incorporar un sistema adicional que permita detectar el satélite defectuoso con el propósito de eliminarlo del cálculo y poder continuar con la operación si se continúan teniendo suficientes medidas. A este sistema se le conoce como Fault Detection and Exclusion (FDE), aunque no es un requisito para poder usar el ABAS como instrumento principal de navegación. Añadir este sistema aumenta el número mínimo de satélites necesarios a 6 debido a que uno podría ser eliminado y no se quiere perder el servicio del anterior hasta que se vuelva a encontrar otro.

Como se ha comentado anteriormente, el ABAS no mejora las prestaciones de la señal GNSS en tanto a precisión, dado que solo incorpora un monitor de integridad de la señal. Es por este motivo que el ABAS suele utilizarse para la fase de ruta, ya sea continental o oceánica, en la que los requisitos de precisión son notoriamente menos restrictivos que en otras fases así como en salidas y llegadas con menores requerimientos de precisión, es decir, NPA.

2.3.2. Satellite Based Augmentation System (SBAS)

El SBAS es un sistema de aumentación de la señal GNSS diferencial, como es el caso del GBAS. Este tipo de sistemas tienen una mayor precisión que los convencionales ya que se apoyan en estaciones de referencia cuya localización está perfectamente medida mediante técnicas cartográficas. En el caso del SBAS se busca dar servicio a una zona extensa (aproximadamente de la extensión de un continente). La Figura 2.8 muestra los sistemas actuales y sus fechas estimadas de implementación.

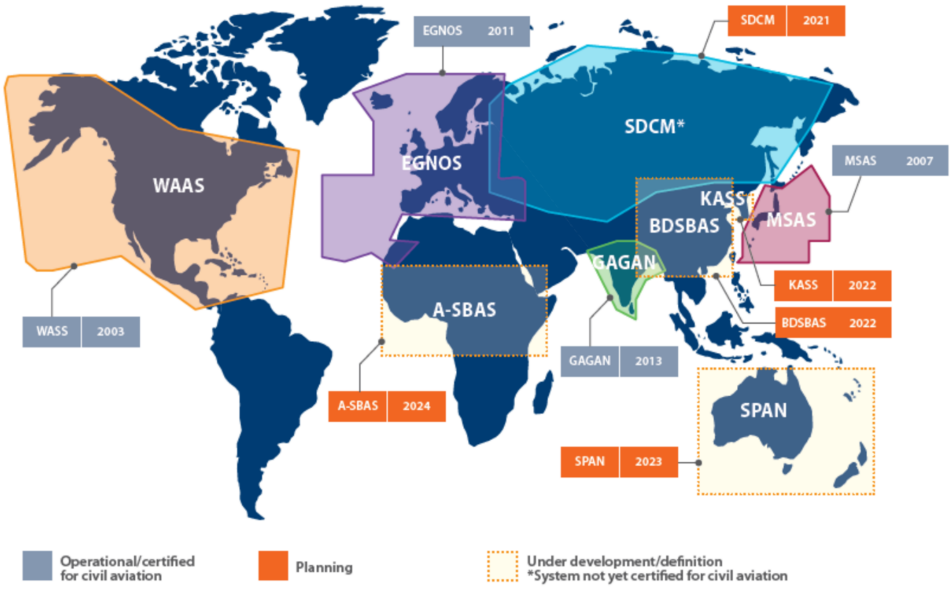


Figura 2.8: Servicios actuales de aumentación SBAS y su área de cobertura [Fuente: extracrew]

Para explicar el funcionamiento de este sistema nos centraremos en el European Geostationary Navigation Overlay Service (EGNOS), que es la implementación europea. A diferencia del ABAS, el SBAS sí que mejora la precisión de posicionamiento. Asimismo, un sistema SBAS valida la integridad de las señales GNSS así como su disponibilidad y continuidad,

llegando a poder utilizarse para aproximaciones LPV-200 (CAT I), si el aeropuerto firma un acuerdo de responsabilidad ya que la continuidad del sistema no cumple con los mínimos de la OACI. En cuanto a las prestaciones generales del sistema EGNOS [17]:

- **Precisión:** los errores de navegación del sistema, esto es, la combinación de los errores en las efemérides de los satélites, geometría del satélite-usuario, etc; mejora en gran medida gracias a las correcciones diferenciales ofrecidas por el sistema EGNOS. Esto le permite alcanzar una precisión horizontal y vertical previstas de 1-2 m y 1-3 m respectivamente en un 95 % del tiempo.
- **Integridad:** el riesgo de integridad, esto es, el número de eventos que pueden suponer una pérdida de integridad del sistema (no tener una precisión utilizable) para un tiempo de alarma máximo de 6 s y unos límites verticales y horizontales de 50 y 40 m respectivamente, es de $2 \cdot 10^{-7}$ cada 150 s.
- **Continuidad:** el valor que ofrece el proveedor de servicios de EGNOS diferencia la continuidad del sistema entre el tipo de operaciones que se pretenden realizar y la zona geográfica en la que se requiere del servicio. Para gran parte de la zona de cobertura esta es de $2{,}5 \cdot 10^{-4}/h$ para NPA y de $1{,}0 \cdot 10^{-4}$ por 15 s para APV-I.
- **Disponibilidad**: en el caso de NPA la disponibilidad es del 99.9 % en toda la zona de cobertura y del 99 % para APV-I.

El sistema EGNOS cuenta con dos segmentos:

- **Segmento espacial:** está compuesto por dos elementos, aunque en un sentido estricto solo uno de estos pertenece al sistema. El principal componente para el funcionamiento del sistema es la señal GNSS de referencia, en este caso el GPS a esperas de que GALILEO esté completamente operativo y también comience a formar parte de este segmento. Por otro lado, el sistema EGNOS cuenta con una red de satélite geoestacionarios que son los encargados de transmitir las correcciones diferenciales a los usuarios.
- **Segmento de tierra:** está compuesto por un conjunto de instalaciones que en conjunto analizan la señal GPS, calculan las correcciones de estas y las envían a los satélites geoestacionarios para que lleguen a los usuarios. Las estaciones principales que lo componen son las siguientes:

  - Ranging & Integrity Monitoring Stations (RIMS): son instalaciones con antenas receptoras que suministran las medidas crudas de los satélites (RIMS A), aportan los datos necesarios para el cálculo de los datos de integridad del sistema (RIMS B) y detectan fallos conocidos como *malas formas de onda* (RIMS C). Son un total de 40 RIMS distribuidas en 19 países europeos o circundantes.

  - Mission Control Centres (MCC): son 4 centros encargados de controlar toda la operación del sistema así como de realizar los cálculos de correcciones diferenciales así como de comprobar la integridad de la señal en sus centros de control y procesamiento.

- Navigation Land Earth Stations (NLES): reciben los datos de corrección de los centros de control y procesamiento preparados para ser enviados a los satélites geoestacionarios. Es por este motivo que estas instalaciones cuentan con grandes antenas transmisoras capaces de enviar los datos al segmento espacial. Son un total de 6 NLES, con 2 de ellas para cada satélite.

- EGNOS Wide Area Network (EWAN): es la red encargada de conectar todas las estaciones de tierra para el correcto funcionamiento del sistema.

El esquema de funcionamiento descrito puede apreciarse gráficamente en la Figura 2.9.

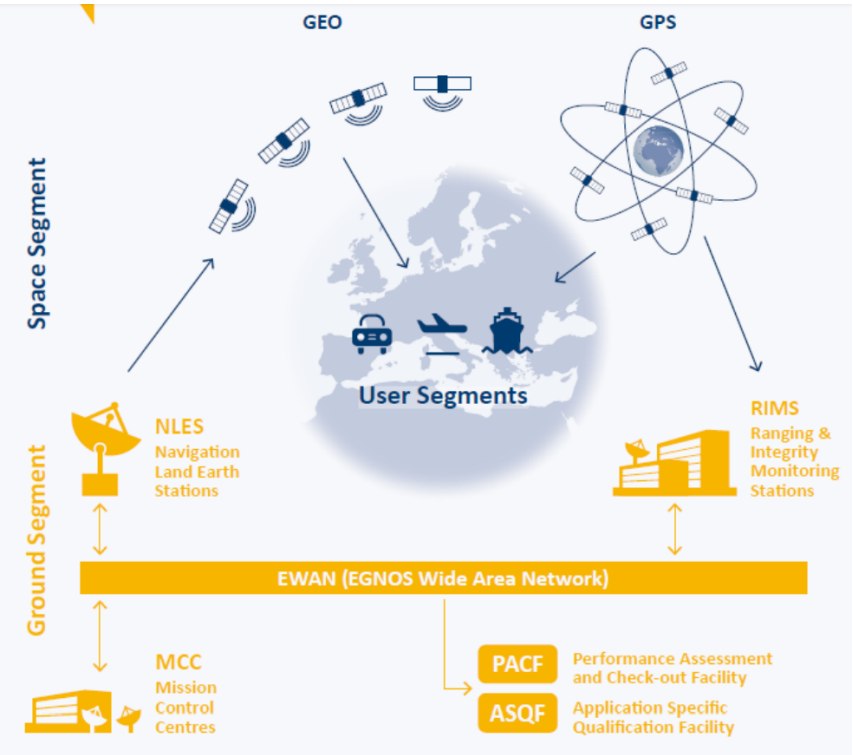


Figura 2.9: Esquema de funcionamiento de EGNOS [Fuente: researchgate]

2.3.3. Ground Based Augmentation System (GBAS)

El GBAS es otro sistema de aumentación de la señal GNSS que aumenta la precisión del posicionamiento al hacer uso de medidas diferenciales. La diferencia principal entre este sistema y el SBAS es el área cubierta (en este caso limitada a la zona de terminal de un aeropuerto) y el sistema utilizado para dar su servicio. El funcionamiento de este sistema así como sus características serán analizadas en profundidad en el Capítulo 3.

Capítulo 3

El sistema GBAS

A lo largo del presente capítulo se discutirá el Ground Based Augmentation System (GBAS). Este sistema de aumentación de la señal GNSS basado en medidas diferenciales busca ofrecer servicio a un área mucho más reducida que un sistema SBAS, alrededor de la zona terminal del espacio aéreo de un aeropuerto, con el beneficio de tener una infraestructura mucho más sencilla que el sistema SBAS y con unas prestaciones mayores.

El GBAS, de la misma forma que otros sistemas de aumentación, comienza a desarrollarse con la desaparición de la disponibilidad selectiva de la señal GPS en 2002. No obstante, el GPS diferencial, precursor del GBAS se comenzó a desarrollar en la década de 1980. Debido a que la disponibilidad selectiva se puede interpretar como un error sintético que es introducido en la señal GPS, si se dispone de una estación de referencia cuya posición es conocida con gran precisión, este error se puede estimar y eliminar para computar la posición de un móvil.

Actualmente el sistema GBAS está en continuo estudio y desarrollo a lo largo del mundo. Uno de los grandes contribuyentes a este ha sido Australia, que certificó sus sistemas GBAS para el aeropuerto de Sydney en 2008. En Europa, las pruebas del GBAS están principalmente supervisadas por EUROCONTROL. Países como Francia, España e Italia ya disponen de casos de implementación de este sistema, siendo un ejemplo el ubicado en el aeropuerto de Málaga desde 2014, siendo en aquel entonces el cuarto aeropuerto del mundo en contar con este sistema. A nivel internacional, desde 2004, la FAA y EUROCONTROL crean eventos a través de la International GBAS Working Group (IGWG) para poner en común avances en esta tecnología.

Volviendo al caso de Australia, su implementación a nivel regional, denominada Ground-based Regional Augmentation System (GRAS), presenta características que lo convierten en una mezcla de las soluciones SBAS y GBAS. Al igual que el sistema SBAS cuenta con una serie de estaciones monitoras en toda Australia que se encargan de observar la señal GPS para enviársela a una unidad central de cómputo. No obstante, a diferencia del SBAS, no cuenta con un sistema de satélites geoestacionarios, sino que utiliza las propias estaciones que en el SBAS servirían para enviar la información a los satélites para transmitir las correcciones diferenciales. Es por este motivo que reciben el nombre de pseudo-satélites.

En primer lugar, para dar una visión global del sistema y se explicarán sus principios de

funcionamiento. A continuación, de tratará la infraestructura del sistema, sus aplicaciones y ventajas e inconvenientes frente a otras soluciones disponibles. Por último, se detallarán las prestaciones del sistema en términos de requerimientos de la OACI y los procesos de monitorización del sistema.

3.1. Principios de funcionamiento

El funcionamiento del sistema GBAS se asimila al del SBAS. Así, como este, cuenta con un segmento de espacio y uno de tierra. La diferencia fundamental es el objetivo de cada sistema; en el caso del SBAS es ofrecer diferentes servicios de prestaciones considerables a una área geográfica extensa. En el caso del GBAS se centra en ofrecer mejores prestaciones en un area crítica de uso, como es la zona terminal del espacio aéreo de un aeropuerto concreto.

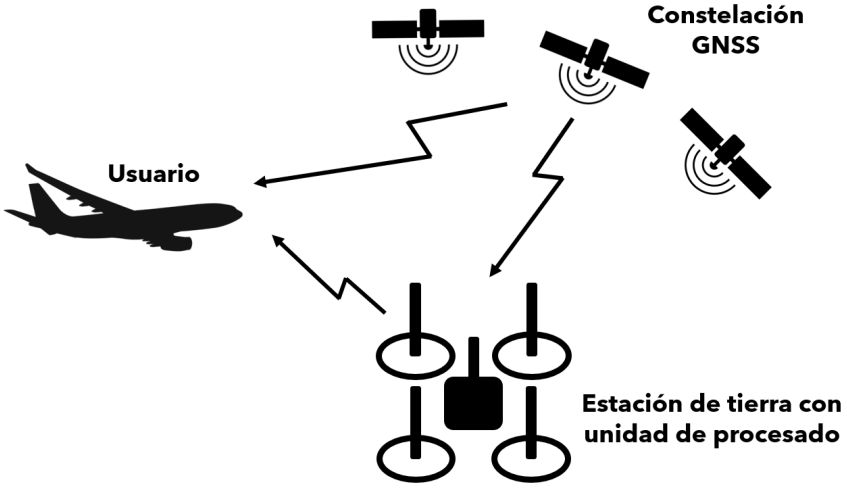


Figura 3.1: Esquema de funcionamiento de un sistema GBAS [Elaboración propia]

La Figura 3.1 muestra el esquema de funcionamiento general del sistema. Como ya se ha mencionado anteriormente, el GBAS se utiliza para hacer medidas GNSS diferenciales, utilizando una estación de referencia. Esta es la estación de tierra, la cual cuenta con cuatro[1] antenas receptoras de la señal GNSS, normalmente GPS. Estos datos se procesan en una unidad de cómputo, la cual se encarga de calcular las correcciones diferenciales que serán transmitidas, y de comprobar la integridad de la señal GNSS[2]. Esta suele encontrarse dentro de la caseta que monta la antena transmisora de la estación de tierra, aunque si se requiriera, podría instalarse en otro lugar y conectarse con la estación mediante un enlace de datos.

El componente fundamental dentro de los cálculo de la unidad de procesado es la denominada Pseudo-Range Correction (PRC), que consiste en una medida de corrección que

[1] Generalmente son cuatro los receptores de la estación de tierra del GBAS, pero podría variar dependiendo de la implementación

[2] La unidad de cómputo también genera otros parámetros dentro del mensaje GBAS, y serán discutidos en el Apartado 3.2.1

cubre en grandes rasgos todas las fuentes de error del segmento de espacio, que son los siguientes [18]:

- **Retraso ionosférico:** como se verá en el Capítulo 5, la ionosfera es un medio dispersivo, lo cual introduce un retraso en la señal del GNSS que depende de la frecuencia y además este varía según la actividad solar, entre otros factores. Gracias a los modelos ionosféricos con los cuales operan las constelaciones GNSS se puede reducir hasta cierta medida, pero con la estación de tierra se puede mitigar aún más.

  En el caso del retraso ionosférico, según los modelos matemáticos establecidos, se ha comprobado que este depende de la elevación con la que se observe el satélite que trasmite la señal. Así, para elevaciones similares en la misma zona geográfica los retrasos ionosféricos estarán altamente correlados.

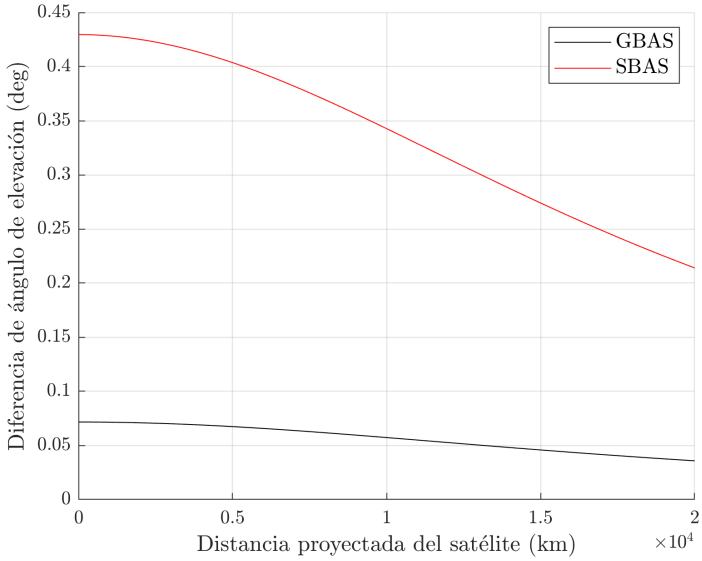


Figura 3.2: Diferencia de ángulo de elevación entre usuario y estación [Elaboración propia]

La Figura 3.2 muestra la diferencia en el ángulo de elevación medido entre un usuario y la estación de referencia separados en una distancia de 25 km, aproximadamente el rango de un sistema GBAS, para mostrar el peor caso de diferencia de distancia. También se puede apreciar como, en el caso del SBAS, esta diferencia, en términos relativos, es significativa, lo cual deriva en mayores discrepancias.

Esto se debe a que el SBAS, al tener una área de cobertura tan extensa en comparación al GBAS, no puede disponer de instalaciones tan cercanas a los usuarios como este. Para el cálculo de la Figura 3.2 se ha asumido una distancia al usuario de la estación más cercana de 150 km para el SBAS.

- **Retrado troposférico:** de forma similar al retraso ionosférico, la presencia de humedad en las capas más bajas de la atmósfera ocasiona un retraso en la señal que debe ser considerado. En este caso, como depende del área geográfica en la que se encuentre el usuario, se puede modelar a través de la estación o bien aproximar a que este será muy similar al del usuario debido a la cercanía.

- **Error de reloj del satélite:** este error se puede corregir mediante el mensaje de navegación de la constelación. No obstante, el error de reloj de la estación GBAS es desconocido. Sin embargo, dado que trabajamos en diferencial, este error se minimiza al juntarse con el error de reloj del usuario.

Introduciendo estos datos, la PRC se puede calcular con la fórmula dada por la Ecuación 3.1 teniendo en cuenta el resultado de la Ecuación 2.1 y sustituyendo el error de satélite del reloj de usuario por el de la estación.

$$PRC = \rho - c \cdot t = c \cdot \left(\Delta t_{iono} + \Delta t_{tropo} + \Delta t_{bias}^{sat} + \Delta t_{bias}^{station}\right) \tag{3.1}$$

Una vez calculados los parámetros del mensaje de la estación de tierra, este se emite a través de la antena VHF Data Broadcast (VDB) para que llegue al usuario, que recibe también los datos de la constelación con su receptor GNSS.

3.2. Infraestructura requerida

Los tres elementos principales de la infraestructura del sistema GBAS, como ya se ha comentado, son: la constelación GNSS, la estación de tierra y el receptor del usuario, que si bien no forma parte de la infraestructura del proveedor de servicios, es necesario para el correcto posicionamiento de este [19]. En este apartado se cubrirán en profundidad los dos últimos, ya que la constelación GNSS ya ha sido tratada en el Apartado 2.1.

3.2.1. Estación de tierra

La estación de tierra del GBAS se encarga de procesar toda la información que recibe a través de sus receptores GNSS, hacer los cálculos de las correcciones diferenciales y el monitoreo de la integridad del sistema, y transmitirlo por VDB al usuario. Un ejemplo de estación de tierra del GBAS se puede ver en la Figura 3.3.

Figura 3.3: Estación de tierra del GBAS de Bremen [20]

Los componentes principales de la estación de tierra del GBAS son:

- **Antenas receptoras GNSS:** son un total de cuatro antenas receptoras de la señal GNSS cuya posición ha sido calibrada mediante técnicas topográficas para ofrecer una referencia fija para la toma de medidas diferenciales. Como ya se ha introducido en el Apartado 2.1, dado que varias constelaciones tienen compatibilidad entre sí, las antenas receptoras de la estación de tierra del GBAS trabajan con un modelo de multiconstelación. Esto permite una mayor redundancia en caso de fallo de una de ellas.
- **Antena transmisora VDB:** transmite los mensajes provenientes de la unidad de cómputo. Se trata de una antena VHF que opera dentro de la banda aeronáutica (108-117.950 MHz), con una división de canales de 25 kHz. Como se explicará cuando se hable de los requisitos radioeléctricos del sistema, esta antena debe ubicarse de forma que no haya ningún obstáculo alrededor que pueda bloquear la señal de cara a los usuarios.
- **Unidad de cómputo:** es el componente encargado de procesar la señal GNSS y generar los mensajes que la estación emitirá a través de la antena VDB. Dado que esta unidad tiene incorporada en su base de datos la posición real de los receptores GNSS, puede descartar este dato como incógnita para calcular la PRC. Asimismo, utilizando sus algoritmos internos y conociendo de antemano la posición de los satélites de la constelación, es capaz de monitorizar la integridad de las mediciones, para poder ser transmitida como parte del mensaje.

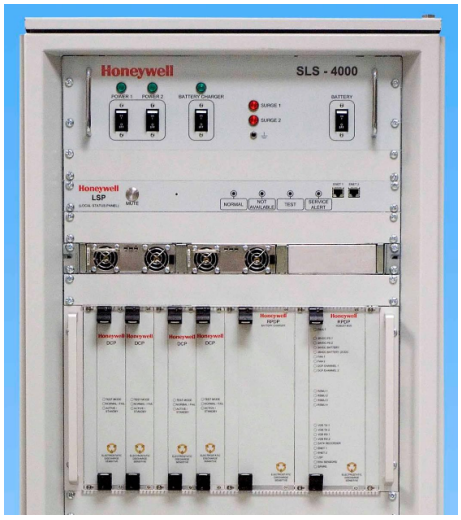


Figura 3.4: SLS-4000 de Honeywell, unidad de cómputo para GBAS [21]

Dependiendo de la certificación de este componente, el GBAS pùede dar servicio a aproximaciones de precisión de distinta categoría, hasta llegar a CAT III. Un ejemplo de unidad de cómputo es el modelo SLS-4000 del fabricante Honeywell (también incluye las antenas receptoras y transmisora, coincide con el modelo de la Figura 3.3), el cual se puede ver en la Figura 3.4. Este modelo,en 2009, se conviertió en el primero en el mundo en ser certificado por la FAA para aproximaciones de precisión de CAT I (definido por el GBAS Approach Service Type (GAST)-C). En este momento su modelo SLS-5000, con soporte para CAT II/III continua en desarrollo (recogido

en el GAST-D) [21]. En España AENA lo incorporó en 2014, para el aeropuerto de Málaga.

Algunas de las características del SLS-4000 de Honeywell son [22]:

- Antenas receptoras GNSS con un diseño limitador del efecto multipath[3]. Se pueden ubicar a una distancia máxima de 1300 m de la unidad de cómputo y su separación entre ellas es de entre 100 y 200 m.
- Monta un Intel Pentium M de 1.8 GHz como CPU de la unidad de cómputo. Le permite calcular la PRC en tiempo real. El acceso a la unidad se establece a través de la terminal de mantenimiento, que se conecta a la red LAN del aeropuerto vía Ethernet.
- La antena VDB hace uso de un Time Division Multiple Access (TDMA) con 2 *slots* de forma predeterminada y puede ajustarse. Las correcciones se actualizan hasta 2 veces por segundo. Puede estar a una distancia máxima de 200 m de la unidad central de cómputo.

A grandes rasgos, el diagrama de bloques general de la estación de tierra puede verse en la Figura 3.5.

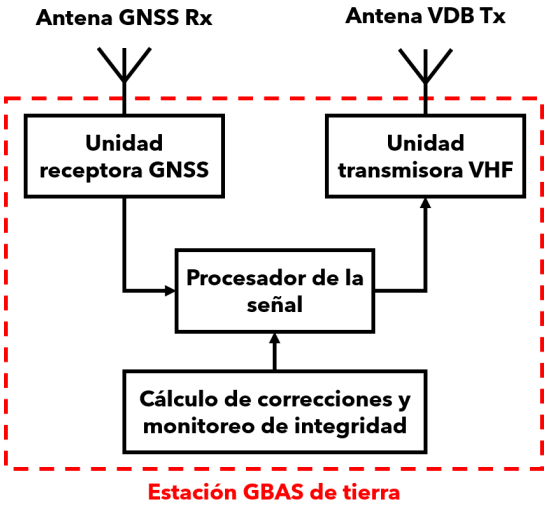


Figura 3.5: Diagrama de bloques de la estación de tierra del GBAS [Elaboración propia]

En cuanto a los mensajes que genera la estación de tierra, estos vienen definidos en el primer volumen del Anexo 10 de la OACI y se recogen en la Tabla 3.1.

Identificador de tipo de mensaje	Nombre del mensaje
0	Libre
...	

[3]Es un error derivado de rebotes que puede tener la señal GNSS en diferentes obstáculos que encuentre a su paso.

Identificador de tipo de mensaje	Nombre del mensaje
1	Correcciones de pseudo-distancia (PRC)
2	Datos relacionados con el GBAS
3	Mensaje vacío
4	Datos del Final Approach Segment (FAS)
5	Predicción de la disponibilidad de datos
6	Reservado
7	Reservado para aplicaciones nacionales
8	Reservado para tests
9 a 100	Libre
101	PRC de un GRAS
102 a 255	Libre

Tabla 3.1: Tipos de mensaje de la estación GBAS

Los mensajes de Tipo 1 (muy similares a los de tipo 101) incluyen las medidas de pseudo-distancia de todos los satélites que recoge la estación de tierra, así como su identificador y la fecha en la que se toman las medidas. También se incluye un 'paquete de datos para cada satélite medido que incluye la PRC, la Range Rate Correction (RRC), que mide el cambio que sufre la PRC con el tiempo y tiene unidades de velocidad; aparte de una serie de parámetros de dispersión de las medidas.

Los mensajes de Tipo 2, ofrecen datos actualizados de la estación de tierra del GBAS. Entre estos datos se incluyen:

- **Características físicas de la estación:** número de receptores de referencia (entre 2 y 4), latitud y longitud de la estación, variación magnética en el punto de referencia, altura del punto de referencia.
- **Datos para el uso de la instalación:** distancia máxima que garantiza la integridad de las medidas expresada en términos del slant-range (no distancia horizontal). Este valor está codificado para llegar hasta los 510 km, sin embargo, esto solo sucede en sistemas como el GRAS australiano, en un GBAS de un aeropuerto suele ser alrededor de 25-50 km.

Los mensajes de Tipo 4 permiten hacer uso de del GBAS como GBAS Landing System (GLS). Incluyen los mensajes de tramo final de aproximación para todos los procedimientos codificados en la estación con GLS. Este tipo de bloques de datos se encargan de virtualizar la senda de planeo para ofrecer unos datos similares a los del ILS. La Figura 3.6 muestra la configuración de estas aproximaciones. Así, se deben definir una serie de puntos que son [23]:

- **Landing / Ficticious Threshold Point (LFP/FTP):** define en latitud, longitud y altitud el punto que será considerado como umbral de la pista de aterrizaje. Este

puede coincidir o no con el real (por ejemplo para modificaciones temporales que desplacen el umbral o por cuestiones de eficiencia), de ahí el acrónimo de FTP.

- **Threshold Crossing Height (TCH):** como su propio nombre indica, es la altura a la que en la trayectoria nominal se cruza el umbral de la pista. Suele tener un valor de unos 15 m.
- **Glide Path Intersection Point (GPIP):** punto de intersección de la senda de planeo con la pista definida por el FAS.
- **Flight Path Aligment Point (FPAP):** es un punto localizado en el mismo plano horizontal que el LTP y que permite definir el alineamiento del procedimiento con la pista. Si este está alineado con el el centro de la pista suele estar ubicado más allá del final de pista.

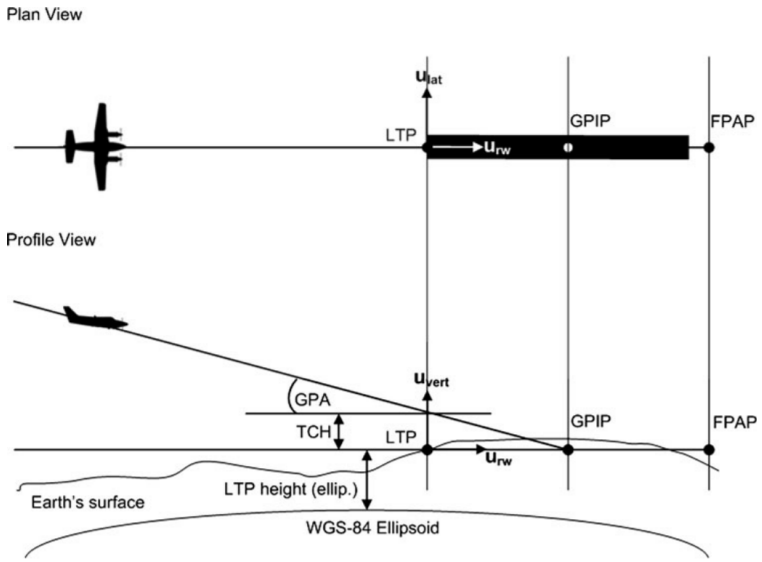


Figura 3.6: Esquema de parámetros del bloque de datos del FAS [23]

Dentro de este bloque de datos también se incluye la pista a la que pertenece, la ruta que marca el procedimiento hasta llegar al tramo de FAS, el tipo de operación (en este caso siempre será GLS, pero los bloques de datos FAS son comunes a otros procedimientos como los basados en SBAS) así como el ID del aeropuerto [24].

3.2.2. Receptor de la aeronave

El receptor de la aeronave no forma parte intrínseca de un GBAS pero es indispensable para su uso. Esta formado por un Multi-Mode Receiver (MMR), una unidad de procesamiento y los monitores de la interfaz del piloto. Su diagrama de bloques general puede verse en la Figura 3.7.

El MMR se encarga de recibir todos los datos necesarios para los cálculos de la posición de la aeronave (utilizando GNSS, si se disponen de unidades inerciales no forman parte del receptor) para enviarlos a la unidad de procesamiento. Por este motivo, cuenta con dos

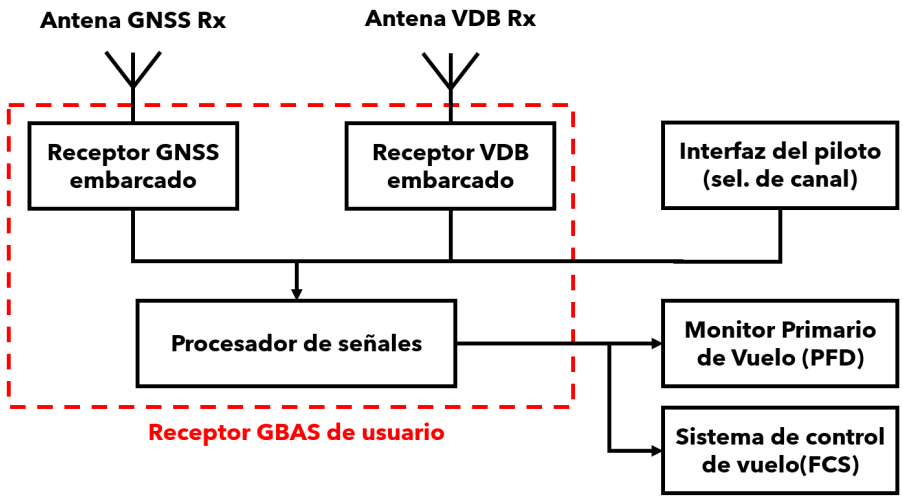


Figura 3.7: Diagrama de bloques del receptor de usuario [Elaboración propia]

antenas: una para las señales de GNSS multiconstelación y la otra para los paquetes de datos transmitidos por las antenas VDB de la estación GBAS. Esta última debe operar en toda la banda aeronáutica para poder seleccionar el canal adecuado.

La unidad de procesamiento es la encargada de computar tanto la posición de la aeronave utilizando las correcciones diferenciales como de comprobar estos datos para comprobar la integridad de la información que suministra.

Esta información se envía finalmente tanto al ordenador a bordo para poder tomar las acciones de control correspondientes (por ejemplo, virar para volver a la trayectoria nominal fijada) y al Primary Flight Display (PFD), para poder mostrarle al piloto su ubicación, o en el caso de tratarse de un GLS, su posición con respecto a la senda de planeo nominal [25].

3.3. Aplicaciones

Las principales aplicaciones de un GBAS son dos: la implementación de un GLS y el servicio de posicionamiento para aeronaves en la zona de cobertura VDB.

La función principal de un GBAS es permitir hacer uso de aproximaciones GLS, anteriormente conocido como GNSS Landing System [26]. Este tipo de aproximaciones permiten un mayor grado de libertad a la hora de diseñar los procedimientos, ya que no tiene una senda de planeo fija ni la necesidad de descender en tramos rectos. Asimismo, debido a las prestaciones del sistema, el franqueamiento de obstáculos es menos restrictivo (a mayor prestación del sistema de aterrizaje, mayor control se tiene sobre las zonas por las que va a pasar la aeronave).

Una aproximación GLS trata de asemejarse en la medida de lo posible a una ILS en cuanto a la interacción del piloto con esta. Así, la implementación de este tipo de sistemas es más sencilla. En lugar de seleccionar un canal de frecuencia, como se haría en una ILS para seleccionar el VOR del aeropuerto, se debe seleccionar un canal VDB, como se puede ver

en la Figura 3.8 [27].

Figura 3.8: Selección del canal VDB en un A-320 [Fuente: airservices australia]

Al seleccionar el canal VDB del procedimiento de aproximación determinado, se obtienen los datos de FAS provenientes del Mensaje de tipo 4 de la estación GBAS.

Asimismo, el GBAS puede utilizarse también para dar servicio de posicionamiento preciso a todas las aeronaves que se encuentren dentro de la zona de cobertura. Esto es especialmente interesante para los despegues y la fase de taxi de las aeronaves, en aeropuertos con sistemas de calles más complejos.

3.4. Ventajas e inconvenientes del sistema

El sistema GBAS ofrece grandes ventajas con respecto a los sistemas ILS convencionales, lo cual lo convierte en un buen reemplazo de estos en aeropuertos especialmente congestionados o que cuenten con múltiples pistas. Entre sus ventjas destacan [28]:

- **Servicio simultáneo a múltiples pistas:** a diferencia del ILS, el funcionamiento del sistema GBAS no depende del posicionamiento fijo de unas radioayudas alrededor de una pista, sino que a grandes rasgos solo debe transmitir una señal que incluye información con respecto a correcciones que realizar y acerca de los diferentes procedimientos de aterrizaje disponibles. Según el manual de Standards And Recommended Practices (SARPs) del GBAS, los mensajes de este pueden incluir información relativa de hasta 48 aproximaciones distintas con diferentes configuraciones [29].

  De esta forma, un único sistema GBAS puede utilizar algunos de estos espacios para diferentes pistas, lo cual reduce de forma considerable los costes de operación, ya que únicamente se requiere de un sistema para dar servicio potencialmente a todo un aeropuerto. Al necesitar de menos instalaciones, también se reducen los costes de mantenimiento de estas así como la inversión inicial. La instalación de un sistema GBAS y su validación pueden costar alrededor de los 300000 €. En el caso de un ILS de CAT II/III, este coste se puede elevar hasta 1000000 € por pista que precise de este.

- **Reducción del mantenimiento:** en el caso del GBAS los costes de mantenimiento se reducen de dos formas, siendo la primera únicamente aplicable a aeropuertos de múltiples pistas ya que, como se ha comentado, no se requieren de múltiples instalaciones. Sin embargo, en el caso de comparar los costes de mantenimiento de un sistema GBAS con respecto a otro ILS único, el primero también requiere de menor mantenimiento.

  Esto se debe a que, más allá de detectar averías en el sistema, el funcionamiento a nivel electromagnético del sistema GBAS es mucho más sencillo, ya que solo debe recibir una señal, procesarla y emitir otra. En cambio, en el caso del ILS, al basarse en la emisión de señales con unas características muy marcadas, las pruebas a realizar en este sistema son más complicadas y por ende, más costosas.

- **Aproximaciones en curva:** al transmitir información de aproximaciones mediante coordenadas a las que llegar y no disponer de haces como el ILS, un sistema GBAS posibilita la incorporación de aproximaciones más complejas, como pueden ser las curvas, con guiado tanto lateral como vertical.

  Esta capacidad permite, consecuentemente, tener una mayor libertad para optimizar la utilización del espacio aéreo para el sector de ATC así como reducir el impacto de las operaciones en el aeropuerto, ya que potencialmente se pueden crear procedimientos que eviten zonas urbanas (reducción de la huella sonora) o bien que requieran de menor combustible (reducción de emisiones). Un ejemplo de este tipo de procedimientos puede verse en la Figura 3.9, que muestra un fragmento de la carta del procedimiento GLS RWY 32R del aeropuerto de código OACI KMWH (Grant County International Airport).

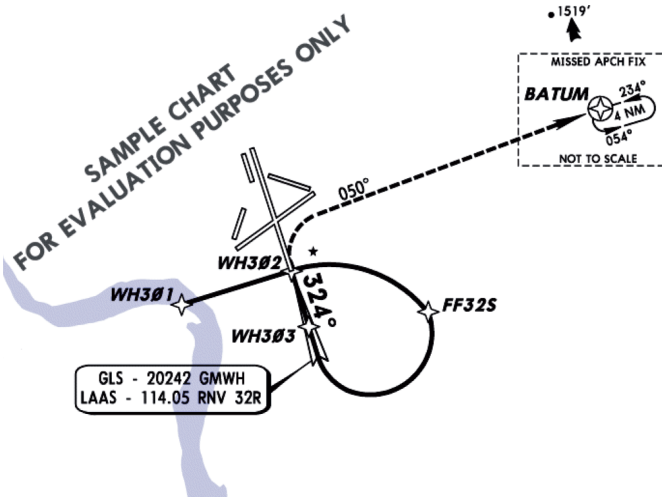


Figura 3.9: Procedimiento en curva GLS del aeropuerto de KMWH [26]

En general se puede hablar de que el sistema GBAS ofrece una mayor versatilidad que los esquemas convencionales a la hora de diseñar el espacio aéreo de la zona aeroportuaria, ya sea con implementaciones específicas como puede ser el aterrizaje curvo, o bien a la hora de realizar cambios. Más allá del procedimiento para publicar cambios en las cartas aeronáuticas, en el caso del sistema GBAS, este es un procedimiento relativamente sencillo, ya que solo se debe modificar la información de los mensajes

transmitidos por la estación, sin modificaciones del hardware requeridas, como sería en el caso del ILS para, por ejemplo, variar el ángulo de la senda de planeo.

- **Diversos ángulos de la senda de planeo y umbrales desplazados:** como ya se ha introducido, al tratarse de una radioayuda fundamentalmente digital en cuanto a su operación, el ángulo de la senda de planeo puede establecerse simplemente cambiando los mensajes transmitidos y se pueden elaborar aproximaciones distintas con diversos ángulos de planeo entre sí. Lo mismo sucede para los umbrales desplazados, si esto ocurre por ejemplo, por la realización de obras en la cabecera de pista, en el caso del ILS es más complicado tener esto en cuenta.
- **Guiado en el tramo de frustrada:** a diferencia del ILS, un sistema GBAS puede ofrecer guiado en cualquier zona dentro de su rango de cobertura, lo cual incluye el guiado preciso en el tramo de frustrada. Como ya se ha comentado, también se puede utilizar el sistema para guiado en la fase de taxi.
- **Reducción de zonas críticas y sensibles:** relacionada con los aspectos radioeléctricos del ILS, hace referencia a que debido a su naturaleza, cualquier perturbación de los haces del localizador o de la senda de planeo puede alterar la operación del sistema y hacer que no ofrezca un guiado correcto. En el caso del GBAS las interferencias son mucho más controlables dado que no se requiere de ningún patrón cuya forma sea crítica, es omnidireccional.
- **Integración:** la integración del GBAS dentro de la interfaz de la aeronave con el piloto puede llevarse a cabo de forma que este no aprecie a simple vista ninguna diferencia con respecto al ILS convencional. La Figura 3.10 muestra la comparación del PFD de un Airbus A-320 cuando realiza una aproximación GLS con la de un ILS.

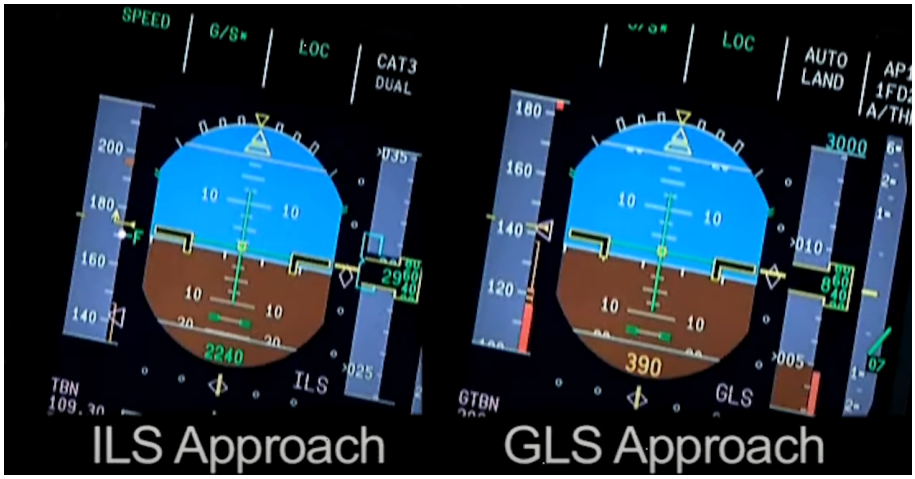


Figura 3.10: Comparación de interfaces entre ILS y GLS [Fuente: hughes aerospace]

La principal desventaja del GBAS es que se pierde independencia en cuanto a la operación del sistema de aterrizaje. Si bien el GBAS trata de ser una versión auto-contenida del más amplio SBAS, hay un factor que es constante en estos sistemas: la dependencia con el segmento espacial, es decir, la/las constelaciones GNSS a partir de las cuales se elaboran los mensajes y correcciones.

Como ya he comentó en el Apartado 2.1, no existen garantías legales de que el sistema GPS esté siempre en funcionamiento y las responsabilidades legales no están completa-

mente definidas. La constelación GALILEO intentará solucionar estos problemas, pero aún así cualquier fallo en este segmento puede ocasionar que aunque la estación de tierra no presente fallos internos, no pueda dar ningún servicio.

Este factor no estaba presente en el ILS, ya que su operación únicamente depende de las interferencias que pueda haber con las señales involucradas o bien del suministro eléctrico del aeropuerto (factor que afecta a cualquier sistema de aterrizaje). En el caso de estas infraestructuras, que son críticas para la operación del aeropuerto, esto no debería suponer un problema ya que se suelen conectar a Sistemas de Alimentación Ininterrumpida o SAIs, pero siempre puede darse un fallo incluso en estos.

Asimismo, el GBAS también presenta una serie de inconvenientes a nivel de usuario, típicos de un sistema en fase de implementación. Y es que para poder hacer uso de este sistema, la aeronave necesita estar equipada con la aviónica necesaria, entre la cual se encuentra el Receptor Multi-Modo, no implementado en muchas aeronaves actualmente. Por otro lado, debido al amplio número de aproximaciones que puede ofrecer el GBAS, se debe llevar cuidado con cuál de estas se selecciona a la hora de aterrizar.

3.5. Monitorización del sistema

Una vez estudiados los componentes y aplicaciones del GBAS este apartado se va a dedicar a describir las diferentes formas de monitorizar el sistema y de mostrar la información de estas. En primer lugar se analizarán los diagramas más comunes de representación de las prestaciones del sistema y a continuación los instrumentos utilizados para la monitorización.

A lo largo del Capítulo 2 se ha insistido en la importancia de los sistemas de aumentación de la señal GNSS para poder implementarse como medio de navegación aérea primaria en diferentes operaciones. Por este motivo, la forma más utilizada para mostrar las prestaciones reales de un sistema no solo recoge datos de precisión, si no también los compara con los niveles de protección del sistema (requisitos como la continuidad o la disponibilidad se representan a parte, ya sea geométricamente como un área de servicio o como valor en un área fija). De esta forma, se puede comprobar en un solo vistazo tanto los requisitos de precisión como de integridad del sistema. Este es el denominado diagrama de Standford.

Este diagrama representa los errores de sistema en el eje horizontal (horizontales o verticales, representados por PE) y los compara con los niveles de protección generados por el sistema en el eje vertical (PL). Si a este factor se le añaden los valores de alerta exigidos por la OACI (AL), el gráfico se divide en una serie de zonas, que son las siguientes:

- **Operaciones nominales:** PE<PL<AL, este es el objetivo que debe alcanzar el sistema. En todo momento el nivel de protección supera al error cometido por el sistema y no alcanza el límite de alerta que haría que el sistema se encuentre no disponible.
- **Información contradictoria:** PL<PE<AL, el sistema no funciona correctamente a pesar de tener unos valores de error y nivel inferiores al límite de operación ya que

no es capaz de calcular correctamente el nivel de protección, que en todo momento debería ser mayor que el de error.

- **Sistema no disponible:** PE o PL >AL, en el momento en el que se superan los límites de alerta, ya sea por el error del sistema o el de protección, se debe alertar al usuario en un tiempo máximo dependiente de la operación. Existen diferentes sub-regiones dentro de esta sección y se pueden ver en la Figura 3.11.

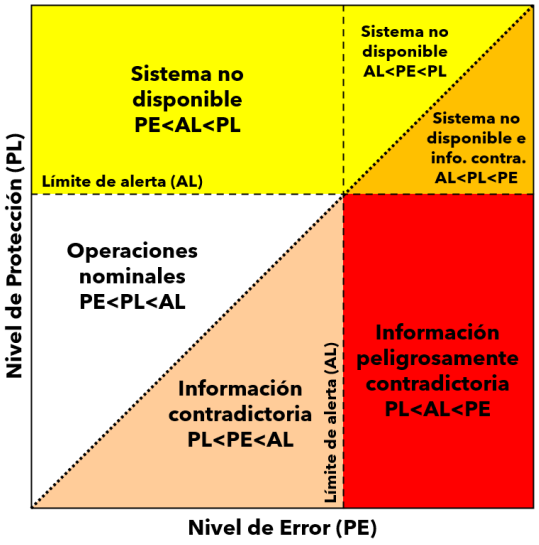


Figura 3.11: Regiones del diagrama de Standford [Elaboración propia]

Un ejemplo práctico de este tipo de diagramas es el presentado por un equipo del Electronic Navigation Research Institute [30] cuando evaluaron las prestaciones del GBAS del aeropuerto de Sendai para aproximaciones de CAT II/III, cuya estandarización ya ha sido redactada (RTCA/DO-245) pero a fecha de redacción de este documento todavía no ha sido aprobada por la OACI. Este diagrama puede verse en la Figura 3.12.

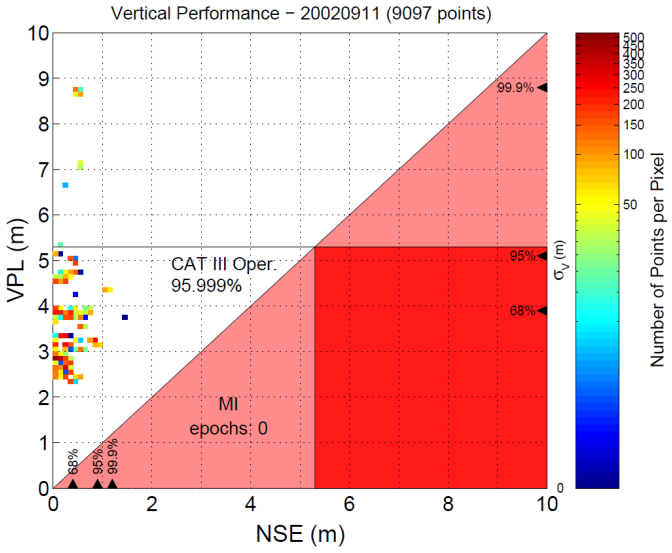


Figura 3.12: Diagrama de Standford para aproximaciones de CAT II/III [30]

Como se puede ver en la Figura, el GBAS permitía conseguir un Navigation System Error (NSE) vertical de 0.85 m en un 95 % del tiempo probado, lo cual cumple con los estándares requeridos por el RTCA/DO-245. No obstante, la disponibilidad del sistema no supera el estándar, ya que solo se encuentra disponible con los niveles de protección requeridos en un 95.99 % del tiempo cuando el mínimo exigible es 99 %.

Otro tipo de gráficos que se pretendan utilizar suelen reflejar aspectos más concretos de la instalación en general o de pruebas realizadas, como pueden ser distribuciones de los errores, gráficos de niveles de interferencia de la señal, etc [31].

Para monitorizar la estación de forma más rutinaria se suelen utilizar dos sistemas, Independent GBAS Monitoring System (IGM) y el GNSS Interference Monitoring System (GIMOS)[20].

- **IGM:** está compuesto por dos partes: la receptora, formada por dos receptores GNSS y uno GBAS (parte superior) y un servidor de datos (parte inferior). Asimismo, incorpora una unidad de alimentación ininterrumpida o SAI para garantizar su operación en caso de pérdida de corriente. Este sistema permite simular los datos de navegación que generaría un usuario en aproximaciones de precisión sin requerir de ensayos de vuelo. Se trata de un sistema portátil, como se puede ver la Figura 3.13.

Figura 3.13: IGM instalado en un vehículo [20]

- **GIMOS:** se encarga de analizar el nivel de interferencia de las señales GNSS recibidas por el sistema. Cuenta con un analizador espectral que opera en tiempo real, un receptor GNSS que cumple con el estándar TSO C-129 (ABAS) y un PC para servir de interfaz con el sistema. De nuevo, se trata de una estación portátil para poder hacer pruebas alrededor de toda la instalación.

Parte II

Simulación de un sistema GBAS con MATLAB y X-Plane

Capítulo 4

Marco general de la simulación

Esta parte se centra en el desarrollo de una simulación de un sistema GBAS así como la aplicación de un GLS dentro del entorno de MATLAB y haciendo uso de datos de Celestrak y X-Plane 11.

En primer lugar, se concretarán los objetivos de la simulación así como su funcionamiento general y las simplificaciones que incorpora en su base. Seguidamente, se introducen cada uno de los componentes que forman parte de la simulación y por último se mostrarán los resultados de la misma.

4.1. Objetivos

El objetivo principal de esta simulación es medir las prestaciones de un sistema GBAS en su uso como instrumento de aterrizaje, es decir, como GLS. Es por este motivo que se incorpora como resultado de la simulación un monitor de los parámetros relevantes del sistema (cilindro de protección, medida de integridad, etc) basado en el mostrado en [32]. Asimismo, dentro del contexto de estudio del documento, esta simulación permite hacer hincapié en aspectos del sistema más allá de sus principios físicos, ya que se van a abordar los diferentes mecanismos que intervienen en su operación como puede ser la interferencia de la ionosfera, el posicionamiento de los satélites, etc.

De esta forma, aunque el objetivo final de la simulación se centra en la obtención de resultados, el objetivo del planteamiento de esta es configurar una descripción realista del sistema en su conjunto, para entender a través de la explicación de esta el GBAS más allá del segmento de la estación de tiera:

- Segmento de espacio: constituido por los satélites de la constelación GNSS a partir de los cuales se toman medidas así como los fenómenos de propagación acontecidos por el medio en el que se encuentran.
- Segmento de tierra: la estación GBAS en sí. Interviene en la toma de medidas de los satélites, en el procesamiento de los mismos y en su envío al usuario.

- Segmento de usuario: de este segmento, constituido por una aeronave, se simulará el receptor GBAS en tanto a la estimación de la posición del usuario y la obtención de las diferentes prestaciones requeridas por la OACI.

4.2. Esquema de funcionamiento

En esta sección se pretende describir el funcionamiento general de la simulación realizada en MATLAB. Para poder ejecutar la simulación se requiere de tres componentes principales:

- **Archivos de efemérides:** son los datos de posición de los satélites activos más cercanos al momento de ejecución de la simulación. Provienen de la base de datos de North American Aerospace Defense Command (NORAD) Celestrak y sirven de base para la simulación de las órbitas de dichos satélites.
- **X-Plane 11:** se trata de un simulador de vuelo comercial que permite el envío y recepción de datos con aplicaciones externas a través de su interfaz. Es el encargado de generar las coordenadas que simularán la posición de un usuario que esté haciendo uso de la estación GBAS.
- **Archivos de MATLAB:** son el núcleo de la simulación. Los diferentes ficheros creados son los encargados de procesar la información de los componentes anteriores y presentar los resultados. Dentro de la interfaz básica de la simulación, si se consideran los archivos de NORAD así como los datos de X-Plane como los componentes destinados a la entrada, dentro de los ficheros de MATLAB se pueden encontrar una serie de módulos destinados a la salida de resultados: esta serie de gráficas permiten evaluar el funcionamiento de las correcciones ofrecidas por la estación GBAS a nivel de usuario para poder monitorizar así su integridad, continuidad, precisión y disponibilidad.

  La interacción de los distintos componentes introducidos se puede ver en la Figura 4.1. A nivel interno, la simulación de MATLAB está gestionada a través de un script llamado `magicSIMMON.m` el cual sirve de nexo de unión entre los distintos componentes que conforman la simulación. La primera fase de la ejecución de la simulación consiste en configurar todos los módulos que van a requerirse. Este proceso incluye inicializar la variable encargada de gestionar el tiempo de simulación, las diferentes clases que simulan a cada uno de los agentes activos en la simulación (satélites, receptores de la estación y usuario) así como establecer el canal de comunicación entre X-Plane y MATLAB.

Una vez este proceso se ha realizado se necesita llegar al punto de partida de la simulación. Para ello se deben determinar las posiciones de los distintos satélites que serán incluidos en la simulación, lo cual incluye descargar los datos de las efemérides de NORAD Celestrak, transformar estos datos a vectores de posición y velocidad así como calcular la posición de estos en el instante inicial de la simulación, puesto que los datos descargados normalmente no coinciden con la época de inicio de la simulación. Los datos de la web pueden actualizarse del orden de una vez por día.

Pasado este punto es cuando empieza la fase de ejecución que genera los gráficos y analiza los datos de X-Plane. Esta está compuesta por un bucle en el cual se actualiza el tiempo

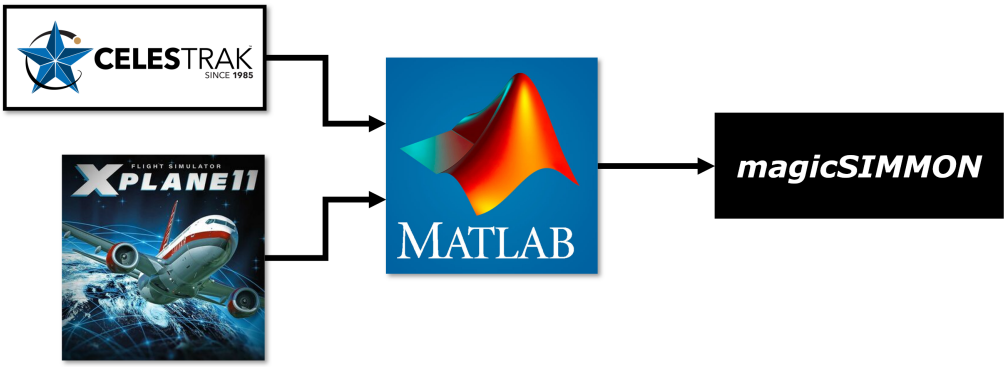


Figura 4.1: Esquema de interacción de la simulación [Elaboración propia]

de simulación con los datos obtenidos a través de X-Plane para dar paso a la actualización del estado de los distintos actores. Este proceso incluye:

- La posición de los satélites activos a lo largo de sus órbitas con el paso temporal establecido.
- La posición de los receptores de la estación de tierra ya que el sistema de referencia de la tierra utilizado es fijo con respecto a las estrellas y por lo tanto una pocisión en términos de latitud y longitud fijas no lo son en este sistema.
- La posición del usuario en base a los datos obtenidos de X-Plane y transformado al sistema de referencia de la simulación.

Tras actualizar las diferentes posiciones reales de los actores, se procede a estimar la posición a través de las medidas de la señal en el espacio. Es en este momento en el que actúa el filtro de mínimos cuadrados que se ha incluido en el segmento de usuario para estimar la posición del mismo a través de las medidas de pseudodistancia ofrecidas por los satélites una vez aplicadas las correcciones obtenidas de la estación GBAS.

Finalmente, se trasladan todos los parámetros relacionados con las prestaciones a los monitores de integridad para actualizar las gráficas con los nuevos valores obtenidos. Tras hacer esto se vuelve al principio del bucle para pedir a X-Plane más datos de posición.

4.3. Simplificaciones aplicadas

En esta sección se pretende hacer un compendio de las diferentes simplificaciones que han sido tomadas a la hora de elaborar la simulación así como su justificación:

- **La Tierra considerada como una esfera perfecta:** el factor de achatamiento de la Tierra es del orden de 0.0033. Dado que el radio máximo de la zona que se va

a simular no supera los 50 km, no considerar la Tierra como un elipsoide tiene un impacto muy reducido a la hora de realizar cálculos. Es por este motivo, sumado a la complejidad que añadiría a los cálculos de transformaciones de coordenadas, que se ha mantenido la Tierra como una esfera perfecta.

- **Órbitas de los satélites:** las órbitas de los satélites se calculan integrando las ecuaciones del movimiento relativo de Newton para dos masas entre las cuales actúa una fuerza de atracción gravitatoria. Si bien estas ecuaciones son válidas para dos objetos esféricos puntuales, diferentes correcciones deben aplicarse para corregir los efectos derivados del hecho de que la Tierra no es un objeto esférico perfecto y existen otros cuerpos alrededor que ejercen un efecto sobre los satélites.

  Entre estos efectos, debido al tiempo considerado para las simulaciones, que es de aproximadamente 10-20 minutos, solo se ha incluido el modelado por el coeficiente J2. Este modela los efectos sobre los satélites debido al achatamiento de los polos y para las altitudes de órbita de los satélites GNSS, es el más pronunciado. La Figura 5.4 recoge el impacto de estos efectos y justifica por qué solo ha sido inclido este.

- **Modelado de la atmosfera:** la mayoría de los algoritmos utilizados para modelar la ionosfera y la troposfera se basan en observaciones tomadas a través de diferentes estaciones. Debido a que en el momento de realización de este Trabajo de Fin de Grado no se disponían de los datos sufiencientes, los modelos utilizados para la ionosfera y la troposfera se basan en una adaptación aproximada de aquellos descritos, siendo incapaces por el momento de simular, por ejemplo, efectos como las tormentas solares, las cuales tienen un gran impacto en las prestaciones de los sistemas basados en GNSS.

- **Efecto multipath:** el efecto multipath o multi-camino se produce al haber reflexiones en distintas superficies de la señal proveniente de los satélites GNSS. Este efecto, dada su complejidad de modelado en un proyecto de esta escala se ha dejado fuera de la simulación.

- **Interacción entre estación de referencia y usuario:** para ser completamente fiel al proceso seguido en un caso real, la interacción entre la estación y el usuario debería darse a través de los diferentes mensajes emitidos por la estación GBAS, que deben ser decodificados e interpretados por el receptor del usuario.

  Para simplificar este proceso y reducir el tiempo de procesado de los datos, este paso se omite, de forma que el usuario recibe directamente las correcciones de cada uno de los satélites de forma que puedan ser aplicadas y no como valores que debe transformar internamente.

Capítulo 5

Desarrollo de los componentes en MATLAB

La implementación de la simulación en MATLAB ha sido desarrollada en el entorno de App Designer. Se trata de un módulo del programa que permite integrar diferentes componentes dentro de una Graphic User Interface (GUI), o interfaz gráfica de usuario, de forma sencilla. Esta es una evolución del anterior sistema de desarrollo de interfaces denominado GUIDE, que no permitía exportar la aplicación de forma semi-independiente (puesto que cierto uso de funciones todavía requiere de MATLAB para poder ejecutarse) de MATLAB.

Los bloques que componen la simulación se agrupan en distintos grupos, y su implementación se corresponde en mayor medida a una serie de clases (aunque la funcionalidad de una misma clase puede incluir componentes de los grupos que se describen a continuación.

5.1. Constelaciones GNSS

La constelación GNSS que forma parte de la simulación está implementada en MATLAB a través de la clase `GNSS`. Esta incluye toda la funcionalidad necesaria para poder calcular la posición y estado de todos los satélites que la componen en el tiempo de la simulación. Sus atributos y métodos están listados en la Tabla 5.1. La implementación total de la clase se puede ver en el Apéndice A.

Nombre	Tipo	Acceso	Descripción
SAT	Atributo	Público[1]	Struct que recoge los datos más importantes de cada satélite de la constelación como pueden ser su ID, la posición en coordenadas geocéntrico-ecuatoriales y su velocidad
sat_count	Atributo	Público	Número de satélites de la constelación
time	Atributo	Público	tiempo actual de la simulación
			...

[1] Puede accederse directamente y su valor es modificable externamente.

Nombre	Tipo	Acceso	Descripción
log_flag	Atributo	Privado[2]	Variable bandera que indica si se quiere almacenar toda la información de la simulación posible (estados históricos y valores de caracterización extraídos de los archivos Two-Line Elements (TLE))
LOG	Atributo	Privado	Struct de datos de la simulación. De no estar activada la log_flag se mantiene en cero
err_flag	Atributo	Privado	Variable que recoge el código de error actual del componente de los satélites. Marcada en 0 si el proceso de inicio se ha efectuado sin errores
t_step	Atributo	Privado	paso temporal anterior de la simulación
GNSS	Método	Público	Función constructora de la clase
SAT_Integrate	Método	Público	Integra la posición de los satélites para el siguiente paso temporal de la simulación
GetLOGflag	Método	Público	Obtiene el estado de la log_flag
GetLOG	Método	Público	Obtiene el struct de archivo de datos
GetERR	Método	Público	Obtiene el estado de la err_flag
GetSTEP	Método	Público	Obtiene el paso de integración de la simulación de la constelación GNSS

Tabla 5.1: Estructura de la clase GNSS

5.1.1. Descarga de datos de la constelación: NORAD Celestrak

Para poder realizar una simulación correcta de la constelación GNSS escogida se debe conocer una posición inicial de esta que se ajuste en mayor medida a la realidad. Con este objetivo y teniendo en cuenta que el uso de la simulación está orientado a escenarios relativamente recientes, se ha optado por la descarga de datos a través del portal de NORAD Celestrack. Esta página, entre otras funciones, hace accesible de forma sencilla la información de las características orbitales de diferentes satélites que publica el NORAD.

Estos datos están publicados en la forma de TLE y son resultado del empleo del modelo Simplified General Perturbation 4 (SPG4), el cual permite obtener aproximaciones rápidas y relativamente precisas de movimientos de cuerpos orbitales [33]. La estructura de estos mensajes se puede ver en las Tablas 5.2 y 5.3.

Línea y campo	Columnas	Contenidos
01-01	01-01	Número de línea
01-02	03-07	Número de catálogo del satélite
01-03	08-08	Clasificación (desclasificado, clasificado, secreto)
...		

[2]Su valor no es visible directamente, solo puede accederse a través de métodos internos.

Línea y campo	Columnas	Contenidos
01-04	10-11	Designador internacional: últimos dos dígitos del año de lanzamiento
01-05	12-14	Designador internacional: número de lanzamiento del año
01-06	15-17	Designador internacional: pieza de lanzamiento
01-07	19-20	Año de EPOCH: últimos dos dígitos
01-08	21-32	EPOCH: día como fracción de año
01-09	34-43	Primera derivada del movimiento medio, coeficiente balístico
01-10	45-52	Decimales de la segunda derivada del movimiento medio
01-11	54-61	Término de arrastre o coeficiente de presión por radiación
01-12	63-63	Tipo de efemérides (siempre 0)
01-13	65-68	Número de TLE del elemento
01-14	69-69	Checksum en módulo 10

Tabla 5.2: Estructura de la línea 1 de un mensaje TLE

Línea y campo	**Columnas**	**Contenidos**
02-01	01-01	Número de línea
02-02	03-07	Número de catálogo del satélite
02-03	09-16	Inclinación en grados
02-04	18-25	Ascensión recta del nodo ascendente en grados
02-05	27-33	Decimales de la excentricidad (asumido 0.X)
02-06	35-42	Argumento del perigeo en grados
02-07	44-51	Anomalía media en grados
02-08	53-63	Movimiento medio en revoluciones por día
02-09	64-68	Número de revoluciones en el EPOCH
02-10	69-69	Checksum en módul 10

Tabla 5.3: Estructura de la línea 2 de un mensaje TLE

Asimismo, un ejemplo del tipo de mensajes que lee el programa es el visto en la Figura 5.1.

```
GPS BIIR-2  (PRN 13)
1 24876U 97035A   22092.57823984  .00000028  00000+0  00000+0 0  9990
2 24876  55.4979 158.6334 0056419  52.2649 308.3227  2.00563518181130
```

Figura 5.1: Ejemplo de mensaje en formato TLE [Elaboración propia]

Este proceso se realiza mediante el comando `webread(url)` de MATLAB, que permite almacenar el texto de una página web para su posterior procesado. La implementación de esta sección, que se encuentra dentro de la función constructora de la clase `GNSS`, se

muestra a continuación:

```
switch sat_const

    case 'GPS'
        url = "https://tinyurl.com/GPS-TLE";
        data = webread(url);

    case 'GLONASS'
        url = "https://tinyurl.com/GLONASS-TLE";
        data = webread(url);

    otherwise
        fprintf('\nIncorrect command\n\n');
        obj.error_flag = 1;
end
```

5.1.2. Obtención del vector de estados inicial: Sistema de referencia

La obtención del vector de estados en el momento inicial de la simulación,esto es, la posición y velocidad de cada uno de los satélites en un momento de tiempo, requiere que se comente la relación de estos con los parámetros mostrados en los archivos TLE.

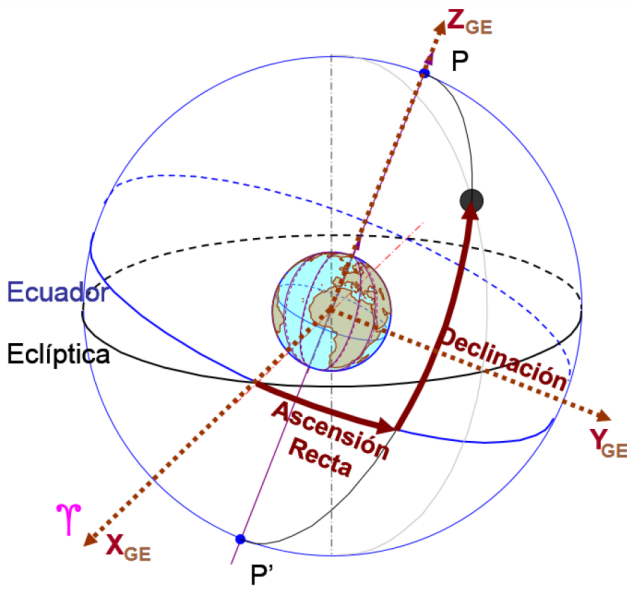


Figura 5.2: Sistema de referencia geocéntrico-ecuatorial [34]

El sistema de referencia empleado en la simulación es el Geocéntrico-Ecuatorial con coordenadas cartesianas. Su esquema general se puede ver en la Figura 5.2. Se trata de un sistema de referencia cuyo origen se sitúa en el centro de masas de la Tierra y tiene como plano de referencia el ecuatorial. La dirección del eje X_{GE} viene dado por el denominado punto vernal o Aries, y es definido como el punto del plano de la eclíptica por el cual

pasa el Sol del hemisferio sur al hemisferio norte, lo cual ocurre en el equinoccio de marzo. Esta dirección se considera fija con respecto a las estrellas (especialmente en la ventana de tiempo de la simulación, que es del orden de minutos). La dirección del eje Z es el eje de rotación de la Tierra y la Y es perpendicular a los anteriores. La ventaja de este sistema de referencia es que es inercial, por lo cual la descripción del problema y su integración se simplifica.

La obtención de los estados iniciales del satélite se basa en la transformación de las coordenadas de este desde el sistema de referencia perifocal, el cual depende de los parámetros descritos en el TLE. El esquema de este sistema de referencia puede verse en la Figura 5.3. En este caso el sistema no es inercial, ya que su origen se ubica en el centro de masas de la Tierra (el cual es uno de los focos de la órbita). La dirección del eje X_{PF} viene dada por el perigeo de la órbita y se encuentra dentro del plano de la órbita.

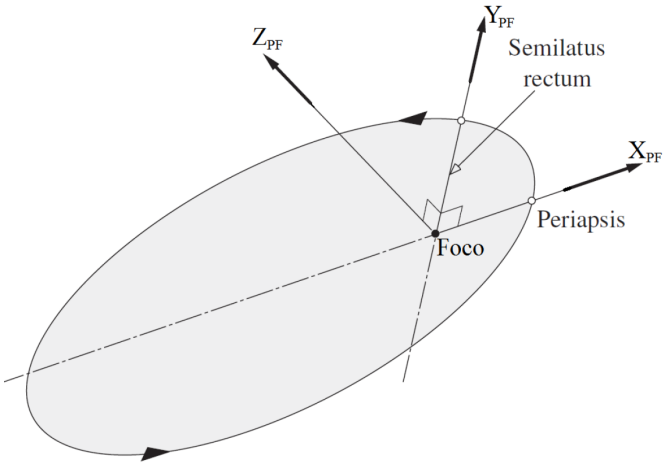


Figura 5.3: Sistema de referencia perifocal [35]

Denotando la Ascensión Recta del Nodo Ascendente (RAAN) como Ω, la inclinación de la órbita como i y el argumento del perigeo como ω, la Ecuación 5.1 que describe el cambio de sistema de referencia de la posición tanto como de la velocidad entre ambos sistemas de referencia.

$$\begin{bmatrix} X_{GE} \\ Y_{GE} \\ Z_{GE} \end{bmatrix} = R_z(\Omega) \cdot R_x(i) \cdot R_z(\omega) \cdot \begin{bmatrix} X_{PF} \\ Y_{PF} \\ Z_{PF} \end{bmatrix} \tag{5.1}$$

Donde las matrices $R_z(\Omega)$, $R_x(i))$ y $R_z(\omega)$ representan las matrices de rotación con respecto a cada uno de los ejes que constan en el subíndice. Asimismo, la relación de los parámetros orbitales con las coordenadas perifocales puede verse en la Ecuación 5.2.

$$\begin{bmatrix} X_{PF} \\ Y_{PF} \\ Z_{PF} \end{bmatrix} = \begin{bmatrix} \frac{h^2/\mu}{1+e\cos\theta}\cos\theta \\ \frac{h^2/\mu}{1+e\cos\theta}\sin\theta \\ 0 \end{bmatrix} \tag{5.2}$$

En la Ecuación 5.2, h es el momento angular del satélite, θ es su anomalía verdadera y μ

es el parámetro gravitacional de la Tierra, que corresponde a la constante de la gravedad por la masa de la Tierra, por lo que tiene un valor de 398600,5 km^3s^{-2}. La ecuación 5.3 relaciona estos con los valores del TLE.

$$h = \sqrt{\mu \cdot a(1-e^2)} \qquad \theta = 2\arctan\left(\sqrt{\frac{1+e}{1-e}} \cdot \tan\frac{E}{2}\right) \qquad E = M + e\sin E \tag{5.3}$$

Donde E es la anomalía excéntrica, que se calcula de forma iterativa a partir de la anomalía media, M. La implementación de todo lo descrito en este apartado en MATLAB se puede ver a continuación:

```
% SEMI-MAJOR AXIS (KM)
GNSS_ST(i).a = (mu/(GNSS_ST(i).mm*2*pi/(24*3600))^2)^(1/3);

% ECCENTRIC ANOMALY (DEG)
err = 1e-10;
Ecc_an_0 = GNSS_ST(i).ma * pi/180;
convergence_flag = 1;
n_it = 0;

while(convergence_flag)
    Ecc_an =  GNSS_ST(i).ma * pi/180 + GNSS_ST(i).e ...
        * sin(Ecc_an_0);
    if ( abs(Ecc_an - Ecc_an_0) < err)
        convergence_flag = 0;
    end
    Ecc_an_0 = Ecc_an;
    n_it = n_it + 1;
end

GNSS_ST(i).ecc_an = Ecc_an * 180/pi;

% TRUE ANOMALY (DEG)
GNSS_ST(i).theta  = 2*atan(sqrt((1+GNSS_ST(i).e)/...
    (1-GNSS_ST(i).e))*tand(GNSS_ST(i).ecc_an/2)) * 180/pi;

% ANGULAR MOMENT
GNSS_ST(i).h = sqrt(mu*GNSS_ST(i).a*(1 - GNSS_ST(i).e^2));

% G.E. POSITION AND VELOCITY AT EPOCH (KM & KM/S)
r = (GNSS_ST(i).h^2/mu)/(1 + ...
    GNSS_ST(i).e*cosd(GNSS_ST(i).theta));

r_pf = [r*cosd(GNSS_ST(i).theta); r*sind(GNSS_ST(i).theta); 0];

v_pf = [ -mu/GNSS_ST(i).h*sind(GNSS_ST(i).theta);...
    mu/GNSS_ST(i).h*(GNSS_ST(i).e + cosd(GNSS_ST(i).theta)); 0];

GNSS_ST(i).XYZ = rotz(GNSS_ST(i).RAAN) * rotx(GNSS_ST(i).i)...
    * rotz(GNSS_ST(i).w) * r_pf;

GNSS_ST(i).vel = rotz(GNSS_ST(i).RAAN) * rotx(GNSS_ST(i).i)...
    * rotz(GNSS_ST(i).w) * v_pf;
```

5.1.3. Integración de la posición: perturbaciones

Hasta el momento se ha obtenido el vector de estados de los distintos satélites para la época (tiempo) en la que se obtuvo el TLE. Para poder calcular la posición y velocidad de la constelación el cualquier momento de la simulación se necesitará integrar la ecuación del movimiento de los satélites. Aplicando la segunda ley de Newton sobre dos masas puntuales en los que únicamente actúa la fuerza de atracción gravitatoria entre ellos y expresando esta relación en forma de movimiento relativo se llega al resultado de la Ecuación 5.4 [36].

$$\ddot{\vec{r}} = -\frac{\mu}{r^3}\vec{r} = -\frac{K \cdot (M_{Tierra} + m_{sat})}{r^3}\vec{r} \tag{5.4}$$

Donde se asume que la masa del satélite es despreciable frente a la de la Tierra. Sin embargo, como ya se ha mencionado, la Ecuación 5.4 asume dos masas puntuales, y esto podría aplicarse si la Tierra fuera una esfera perfecta, pero no es el caso, ya que se encuentra achatada por los polos. Asimismo, ambos cuerpos no se encuentran en el vacío, existen otras atracciones que intervienen en el movimiento del satélite, como puede ser la de la Luna, el Sol (y la radiación solar), etc. Estos factores habitualmente se modelan como perturbaciones de la ecuación fundamental y tienen asociado un parámetro que se puede incluir de forma sencilla en la Ecuación 5.4. La Figura 5.4 muestra la influencia de estos factores en términos de aceleración [37].

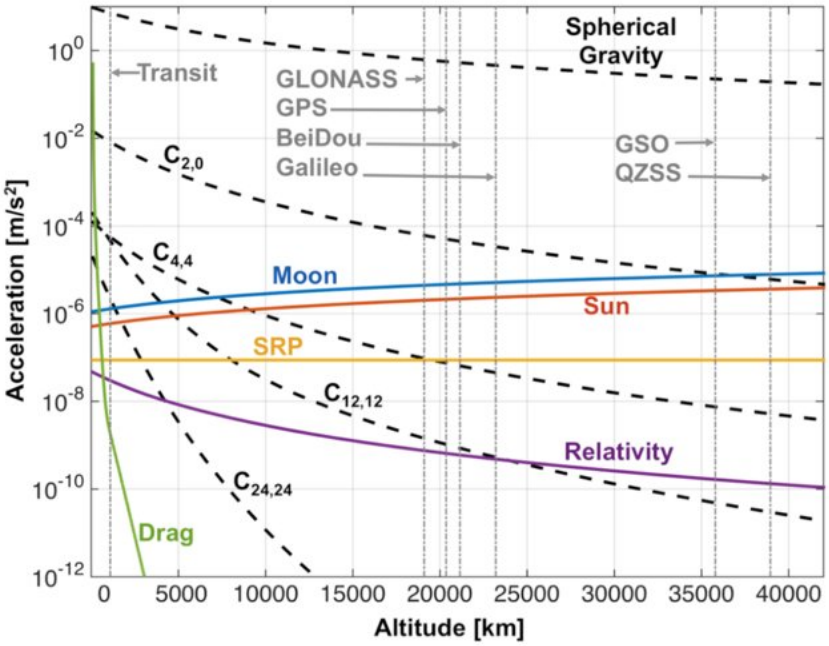


Figura 5.4: Aceleración derivada de las diferentes perturbaciones [37]

Como se puede ver en la Figura 5.4, para la altura a la que se encuentran las constelaciones globales GNSS, la primera perturbación más relevante después de la atracción fundamental (descrita como gravedad esférica) es el $C_{2,0}$, comúnmente denominado J_2 o segundo armónico zonal. Esta perturbación modela el efecto del achatamiento en los polos de la Tierra, que es de alrededor de 0,0033, si se comparan los radios ecuatorial y polar. Este efecto tiene dos consecuencias en las órbitas de las constelaciones:

- **Regresión/Avance de nodos:** la RAAN de la órbita, para aquellas posígradas, es decir, con la inclinación inferior a 90º, decrecerá de forma sostenida. En el caso de las retrógradas, será lo contrario, y la RAAN aumentará. Esto ocasiona un desplazamiento en la traza de los satélites. La Ecuación 5.5 recoge esta variación de la RAAN.

$$\frac{d\Omega}{dt} = -\left(\frac{3}{2}\frac{\sqrt{\mu}J_2R_T^2}{(1-e^2)^2a^{\frac{7}{2}}}\right)\cos i \quad \text{rad/s} \tag{5.5}$$

- **Avance/Retroceso del perigeo:** este es un efecto más notable en las órbitas con grandes excentricidades, ya que ven su perigeo desplazado. En el caso de las circulares, está presente, pero no tiene un impacto significativo en la traza. La Ecuación 5.6 muestra este cambio.

$$\frac{d\omega}{dt} = -\left(\frac{3}{2}\frac{\sqrt{\mu}J_2R_T^2}{(1-e^2)^2a^{\frac{7}{2}}}\right)\left(\frac{5}{2}\sin i^2 - 2\right) \quad \text{rad/s} \tag{5.6}$$

Donde el parámetro J_2 tiene un valor de $1.08263{\cdot}10^{-3}$. Según lo visto en la Figura 5.4, tiene un impacto del orden de 10^{-4} m/s^2 en el cálculo de las órbitas. Teniendo en cuenta que las simulaciones únicamente van a cubrir procesos de aproximación a distancias relativamente cortas (en términos orbitales), esta será la única que se tenga en consideración. Si se tratara de simular el comportamiento de un sistema ABAS durante todas las fases del vuelo, entonces podría ser interesante añadir otras. La Ecuación 5.7 amplía la 5.4 con la perturbación del segundo armónico zonal [38], [39].

$$\ddot{\vec{r}} = -\frac{\mu}{r^3}\vec{r} - \begin{bmatrix} \frac{\mu x}{r^3}\left[\frac{3}{2}J_2\left(\frac{R_T}{r}\right)^2\left(1-5\left(\frac{z}{r}\right)^2\right)\right] \\ \frac{\mu y}{r^3}\left[\frac{3}{2}J_2\left(\frac{R_T}{r}\right)^2\left(1-5\left(\frac{z}{r}\right)^2\right)\right] \\ \frac{\mu z}{r^3}\left[\frac{3}{2}J_2\left(\frac{R_T}{r}\right)^2\left(3-5\left(\frac{z}{r}\right)^2\right)\right] \end{bmatrix} \tag{5.7}$$

La implementación de esta ecuación en MATLAB se puede ver a continuación. La función `Relative_movement_eq` es llamada por la función `ode45` de MATLAB para integrar la posición de los satélites.

```
1  function [Ydot] = Relative_movement_eq(t,Y)
2
3      mu = 398600.5;
4
5      Yd(1) = Y(4); % x
6      Yd(2) = Y(5); % y
7      Yd(3) = Y(6); % z
8
9      r     = sqrt(Y(1)^2+Y(2)^2+Y(3)^2);
10     RT    = 6378; % km
11     J2    = 1.08263e-3;
12     fj21  = (mu*Y(1))/r^3*(1.5*J2*(RT/r)^2*(1-5*(Y(3)/r)^2));
13     fj22  = (mu*Y(2))/r^3*(1.5*J2*(RT/r)^2*(1-5*(Y(3)/r)^2));
14     fj23  = (mu*Y(3))/r^3*(1.5*J2*(RT/r)^2*(3-5*(Y(3)/r)^2));
```

```
15
16      Yd(4) = - mu * Y(1)/r^3 - fj21; % dx
17      Yd(5) = - mu * Y(2)/r^3 - fj22; % dy
18      Yd(6) = - mu * Y(3)/r^3 - fj23; % dz
19
20      Ydot = [Yd(1) ; Yd(2) ; Yd(3) ; Yd(4) ; Yd(5) ; Yd(6)];
21
22  end
```

El resultado final de este componente es una serie de vectores de posición que representan a todos los satélites operativos de una constelación GNSS determinada. La Figura 5.5 muestra un ejemplo gráfico de la constelación GPS y la Tierra con el sistema de referencia geocéntrico-ecuatorial. Se pueden apreciar las trazas de algunos de dichos satélites, representando los planos orbitales principales de la constelación. Esta gráfica se ha obtenido en el fichero `ORBIT_representation`, disponible en el Apéndice A.

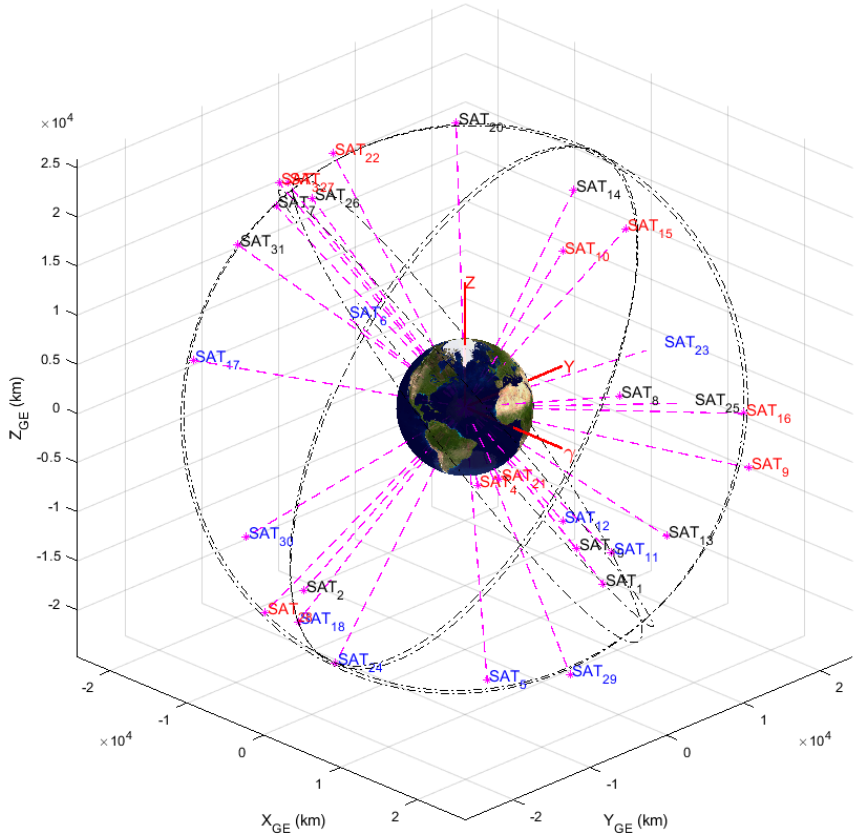


Figura 5.5: Representación de la constelación GPS obtenida a través de MATLAB [Elaboración propia]

5.2. Entorno espacial y atmosférico: signal-in-space

A diferencia de las constelaciones GNSS, el entorno espacial y atmosférico no está modelado a través de una clase en MATLAB, sino que se incluye en funciones específicas de diferentes clases. La atmósfera atendiendo a su refractividad se puede dividir en dos capas:

la troposfera (0-80 km) y la ionosfera (80-1000 km), más allá de estas se considera vacío. Como ya se comentó en el Apartado 2.1, los efectos de propagación de la señal GNSS afectan al sistema de los receptores en forma de retrasos ocasionados por estas capas; y ocasionan los mayores errores en el posicionamiento GNSS (el efecto multi-path no será incluído en la simulación). En este apartado se discutirán los modelos de estos sistemas y su implementación en el código de la simulación.

5.2.1. Efecto de la ionosfera, un medio dispersivo

La ionosfera es una capa la atmósfera que se extiende aproximadamente desde los 80-100 km de altura hasta los 1000 km. En ella, los gases presentes, debido a la radiación solar y, en consecuencia, a las altas temperaturas, se encuentran ionizados. Debido a la baja densidad de la ionosfera, se pueden encontrar electrones e iones libres en el espacio. La ionosfera es un sistema dinámico cuya actividad se ve principalmente afectada por las variaciones en la radiación solar, pero también está afectada por otros factores como el campo magnético terrestre.

A nivel electromagnético, la ionosfera es un medio dispersivo. Esto quiere decir que su índice de refracción (del cual depende la velocidad de propagación de las ondas en un medio que no sea el vacío) depende de la frecuencia y principalmente está ocasionado por la cantidad de electrones libres que se encuentren en esta capa. Este fenómeno produce retrasos distintos en el ancho de banda de la señal GNSS, por lo que se puede recibir el código PRN con un cierto retraso y los mensajes de navegación con otro distinto (retardo de grupo). Si se dispone de un receptor multi-frecuencia, se puede calcular este retraso. Sin embargo, en el momento de redacción de este documento no se ha desarrollado ningún receptor GNSS embarcado y certificado para la navegación aérea que disponga de dicha funcionalidad.

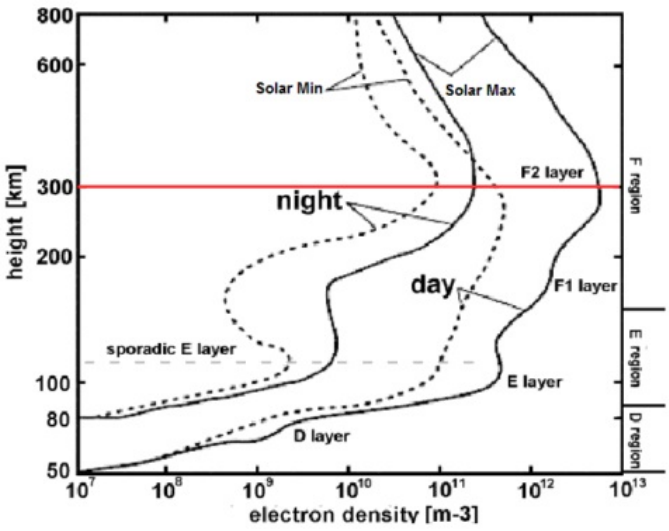


Figura 5.6: Variación de la densidad de electrones en función de la altura [Fuente: roma2.ingv.it]

Asimismo, las variaciones de actividad de la ionosfera varían de forma significativa en cortos periodos de tiempo, como se puede ver en la Figura 5.6. La unidad de medida más extendida para el Total Electron Content (TEC) o densidad de electrones es el Total Electron Content Unit (TECU), que equivale al contenido total de electrones libres en una columna de un metro cuadrado de base que se establece en la dirección de observación entre, en este caso, el receptor GNSS y el satélite; en forma matemática, 1 TECU = 10^6 electrones/m^2. La Ecuación 5.8 modela el error de pseudo-distancia asociado al TEC a una determinada frecuencia [40].

$$\Delta\rho_{iono} = 40{,}3 \cdot \frac{TEC}{f^2} \tag{5.8}$$

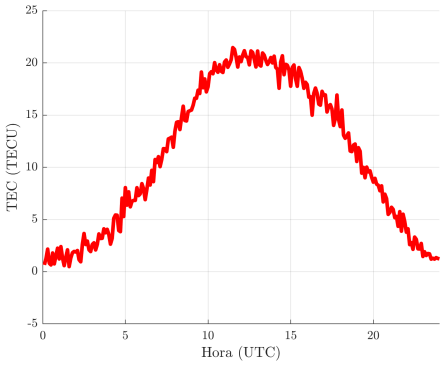
Al describirse el TEC en función de la línea de observación, como ya se adelantaba en el Capítulo 3, la elevación juega un papel importante en este retraso [41]. Existen diferentes modelos que tratan de modelar el TEC para calcular el error cometido por la ionosfera y que se han utilizado en aplicaciones GNSS (existen otros modelos como el International Reference Ionosphere o IRI):

- **Klobuchar:** es el modelo utilizado por el sistema GPS y para receptores de una sola frecuencia. Se trata de un modelo empírico cuyos parámetros se envían a través del mensaje de navegación. Es capaz de corregir hasta un 50 % RMS del error de la ionosfera. Está basado principalmente en una onda coseno para representar los cambios en el TEC con un máximo a las 14:00 locales, y su periodo y amplitud se calculan en base a los parámetros del mensaje de navegación.
- **NeQuick:** es un modelo empírico más reciente que se implementa en la constelación GALILEO y en el sistema EGNOS. Mejora los resultados del Klobuchar, haciendo que el error cometido se pueda corregir hasta en un 75 % RMS. Utiliza la integración numérica para proporcionar el TEC en función de la posición del usuario, la hora y la actividad solar [42].

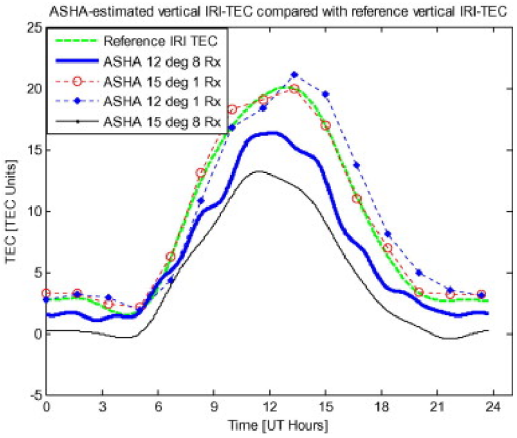
Para la simulación en MATLAB, se va a optar por un modelo ionosférico basado en el Klobuchar, utilizando una onda coseno de amplitud variable, pero dado que no se disponen de los parámetros de este modelo para cada instante de tiempo (tal es el caso del modelo Nequick), se tratará de estimar en base a mediciones empíricas. La implementación de este modelo viene recogida por la Ecuación 5.9.

$$TEC = \frac{VTEC}{\cos Z_{usu}^{sat}} + A = \frac{(A_m \cos\Delta\theta + \Delta A)\cos(\omega_T \cdot t + \Delta\lambda_{UTC}) + \Delta TEC}{\cos Z_{usu}^{sat}} + A \tag{5.9}$$

Donde Z_{usu}^{sat} representa la elevación existente entre el receptor y el satélite, las variables ΔTEC y ΔA son variables aleatorias que modifican ligeramente la amplitud y el ruido de la señal. $\Delta\theta$ mide la diferencia entre la latitud del receptor y el plano de la eclíptica, para determinar la posición relativa del Sol con respecto al usuario en el plano vertical, para modelar el efecto de la estación en la que se realicen las medidas. Finalmente, $\Delta\lambda_{UTC}$ se utiliza para desplazar el pico de la onda en función de la diferencia de longitudes entre el receptor y el pico de referencia. La Figura 5.7 muestra un ejemplo de las medidas que se pueden tomar utilizando este modelo comparado con las de un estudio empírico [43].



(a) Modelo de MATLAB para el TEC [Elaboración propia]



(b) Medidas empíricas del TEC [43]

Figura 5.7: Comparación del modelo implementado con medidas experimentales

5.2.2. La absorción atmosférica en la troposfera

La troposfera es la capa más baja de la atmósfera. En aspectos radioeléctricos, también nombrada como atmósfera inerte, se asume que se extiende hasta los 80 km para incluir así también la estratosfera y poder modelar los retrasos derivados de la dinámica atmosférica. La troposfera contiene aproximadamente el 99 % del vapor de agua de la atmósfera en sus primeros 12 km de extensión, aunque su grosor varía entre los 7 y los 17 km, dependiendo de la latitud a la que se mida. A nivel de propagación, la troposfera es un medio no dispersivo (hasta los 15 GHz aproximadamente).

Las condiciones de temperatura, presión y contenido de vapor de agua en el camino de la señal GNSS producen cambios en el índice de refracción de la troposfera y, al tratarse de un medio no dispersivo, afecta en la misma medida a todo el mensaje. El error inducido, que puede ser de entre 2 y 15 m, aunque puede corregirse hasta en un 90 % con los modelos troposféricos, depende de la longitud de troposfera que atraviese la señal, por lo que se reduce en el cenit y aumenta a medida que se distancia de este. Para modelar este retraso normalmente se asumen las siguientes condiciones [44]:

- El retraso troposférico puede dividirse en dos componentes, uno debido al contenido en vapor de agua en la región que atraviesa la señal (atmósfera húmeda); y otro debido a las condiciones de presión y temperatura de estas zonas (atmósfera seca).
- Ambos tipos de retraso troposférico dependen del ángulo de elevación entre el receptor y la fuente de la señal, en este caso, el satélite de la constelación GNSS.
- Se produce una asimetría en el plano horizontal de la troposfera dado que las condiciones climáticas cambian entre regiones.

5.3. Toma de medidas de los receptores GNSS

Los diferentes receptores GNSS dentro del contexto de la simulación tienen la misión de gestionar la posición de sus respectivos portadores, ya sea el caso del usuario o el de la estación GBAS. Debido a que, dependiendo de la aplicación, tienen una serie de características, para modelar a estos actores se ha creado una clase abstracta denominada `RECEIVER` de la cual las clases `GBAS` y `USER` heredan sus propiedades e implementan aquellas que son características de estos. Las funciones y atributos principales de la superclase `RECEIVER` se pueden ver en la Tabla 5.4 y su implementación completa se encuentra en el Apéndice A.

Nombre	Tipo	Acceso	Descripción
RHO_MEAS	Atributo	Público	Vector con medidas de pseudodistancias crudas de la constelación GNSS seleccionada
SAT_XYZ	Atributo	Público	Struct con las posiciones de los satélites que han sido utilizados para tomar medidas
LOC	Atributo	Privado	Struct que almacena la latitud, longitud y altura del receptor en cuestión medidas en grados y en metros
LOC_XYZ	Atributo	Privado	Vector/Matriz que almacena la posición de los receptores en el sistema de coordenadas Geocéntrico-Ecuatorial
LOG	Atributo	Privado	Struct de datos almacenados a lo largo de la simulación
time	Atributo	Privado	Tiempo actual de la simulación
t_step	Atributo	Privado	Paso temporal de la simulación
elevation_mask	Atributo	Privado	Máscara de elevación del receptor, es la elevación mínima observable para la cual se considera que el receptor es capaz de recibir la señal
n_receivers	Atributo	Privado	Número de receptores del componente
est_state	Atributo	Privado	Estado estimado por el filtro de Kalman del receptor, incluye tanto la posición como la velocidad del mismo
est_pos	Atributo	Privado	Vector de posición del receptor estimado por el filtro de Kalman en el sistema de coordenadas Geocéntrico-Ecuatorial. Es una parte del vector de estados
GetLOC	Método	Público	Obtiene la posición del receptor en formato latitud, longitud y altura
GetLOC_XYZ	Método	Público	Obtiene la posición del receptor en formato Geocéntrico-Ecuatorial
GetLOG	Método	Público	Obtiene el struct LOG
GetREC	Método	Público	Obtiene el número de receptores
GetEmask	Método	Público	Obtiene la máscara de elevación del receptor
update_SAT	Método	Público	Actualiza el struct de SAT_XYZ de la clase RECEIVER

...

Nombre	Tipo	Acceso	Descripción
latlon2XYZ	Método abstracto[3]	Protegido[4]	Convierte las medidas de latitud, longitud y altura al sistema de coordenadas Geocéntrico-Ecuatoriales
update_RHO	Método abstracto	Protegido	Actualiza las medidas de pseudodistancia del receptor
update_location	Método abstracto	Protegido	Actualiza LOC del receptor

Tabla 5.4: Estructura de la clase RECEIVER

Más allá de todos los métodos que se encargan de gestionar las diferentes variables dentro de la clase, las funciones principales del sistema son `latlon2XYZ` y una función externa a la clase que se encarga de aplicar las medidas, llamada `MEASURE`.

Como se ha visto en el Apartado 5.1.2, el sistema de referencia utilizado para la integración del movimiento de los satélites es el Geocéntrico-Ecuatorial. Este es un sistema considerado inercial, por lo que no rota solidario con la Tierra. Esto dificulta la conversión entre coordenadas en términos de latitud, longitud y altura a este sistema debido a que un objeto estacionario con respecto a la Tierra (como puede ser la estación GBAS) esté en movimiento en el sistema Geocéntrico-Ecuatorial (describiendo una senda circular a medida que rota la Tierra). Además, como ya se ha visto anteriormente, la rotación de la Tierra va ligada a los días siderales, por lo que el momento en el que se ejecute la simulación determina la posición de la Tierra con respecto a las estrellas.

Para poder solucionar estos problemas se hace uso del concepto de los días Julianos. La fecha juliana mide el número de días (y la fracción de estos) que han transcurrido desde el mediodía del 1 de enero de 4713 a.C. del calendario juliano proléptico. Este tipo de numeración ofrece una escala basada en días solares y permite reducir los errores en la medida de tiempos para eventos astronómicos al no tener que considerar años bisiestos, etc. No obstante, actualmente en la astronomía, la época estándar que sirve de referencia de tiempos es el denominado J2000.0 (o simplemente J2000), que corresponde al mediodía del 1 de enero del año 2000 del calendario gregoriano.

De esta forma, calculando la fecha juliana con respecto al J2000 del instante de la simulación y conociendo con precisión el periodo de rotación de la Tierra, se puede calcular el número de rotaciones completas y la fracción de vuelta que ha realizado desde dicha fecha, permitiendo conocer la posición relativa del Meridiano de Greenwich a la hora de la simulación. Las ecuaciones 5.10 y 5.11 permiten obtener la esa posición relativa, θ_G, una vez calculado el número de vueltas exacto desde el J2000, T_0.

$$\theta_{G0} = \mathrm{mod}\left(\frac{100{,}4606184 + 36000{,}77004\,T_0 + 0{,}000387933\,T_0^2 - 2{,}583\cdot 10^{-8}\,T_0^3}{360}\right) \quad (5.10)$$

[3]Debe ser implementado por las clases que heredan esta función.
[4]Solo puede ser accedido desde la propia clase o por subclases que hereden los métodos.

$$\theta_G = \theta_{G0} + 360{,}98564724 \cdot \frac{UT}{24} \tag{5.11}$$

Donde θ_{G0} representa la distancia angular del meridiano de Greenwich con respecto a la dirección del punto vernal a las 00 h del día de la simulación y el paso de esta a θ_G añade la fracción de día, representado por UT. Por otro lado, la operación mod() representa el resto de la división.

Una vez calculada, simplemente se debe trasladar esto al meridiano local para obtener la posición angular del receptor con respecto al punto vernal, visto en la Ecuación 5.12. Finalmente, solo se deben rotar estos datos angulares sabiendo que el vector debe encontrarse encima de la superficie de la Tierra.

$$\theta_{LOCAL} = \text{mod}\left(\frac{\theta_G + \lambda_{receptor}}{360}\right) \tag{5.12}$$

La implementación de esta función difiere en el caso de ambos receptores en cuanto al número de posiciones que se deben traducir, pero el proceso es el mismo para ambas. La implementación para el caso de un único receptor se puede ver a continuación.

```
function pos_XYZ = latlon2XYZ(obj)
%Obtiene las coordenadas geocentrico-ecuatoriales de los receptores de la
%estacion GBAS a partir de su definicion y el tiempo de la simulacion
    J2000 = juliandate(datetime('2000-01-01 12:00:00'));
    RT = 6371;
    t00 = obj.time;
    t00(1,4:6) = 0;
    % Dia juliano de la medianoche del dia de la simulacion
    J0 = juliandate(t00);
    UT = obj.time(4) + obj.time(5)/60 + obj.time(6)/3600; % Hora UT;

    T0 = double((J0-J2000)/36525);
    % Grados, tiempo sidereo de Greenwich a las 0h
    theta_G0 = mod(100.4606184 + 36000.77004*T0 + 0.000387933*T0^2 - ...
        2.583*10^-8*T0^3,360);
    theta_G = theta_G0 + 360.98564724*(UT/24);
    % Distancia angular del meridiano local al punto vernal
    theta_local = mod(theta_G + obj.LOC.lon,360);
    pos_XYZ = (RT + obj.LOC.alt)*[sind(obj.LOC.lat)*cosd(theta_local);...
        sind(obj.LOC.lat)*sind(theta_local); cosd(obj.LOC.lat)];
end
```

Por otro lado, como se ha mencionado anteriormente, está la función `MEASURE`. Esta se encarga de recoger los valores de las pseudodistancias medidas de los satélites visibles para el receptor y aplicarles los retrasos modelados pertinentes. Para determinar si un satélite es visible para el receptor en cuestión se requiere de la función `SAT_Visible`.

El esquema de funcionamiento de esta se basa en aplicar una máscara de elevación, modelada a través del atributo `elevation_mask` de la clase `RECEIVER`. Dado que se considera que este valor es el más limitante y no se tiene en cuenta la orografía del terreno (la Tierra está modelada como una esfera lisa), esta máscara de elevación mínima define un cono alrededor del receptor en el cual los satélites que se encuentren serán visibles para el receptor.

El esquema de funcionamiento de la función `SAT_Visible` puede verse en la Figura 5.8, seguida de su implementación en MATLAB.

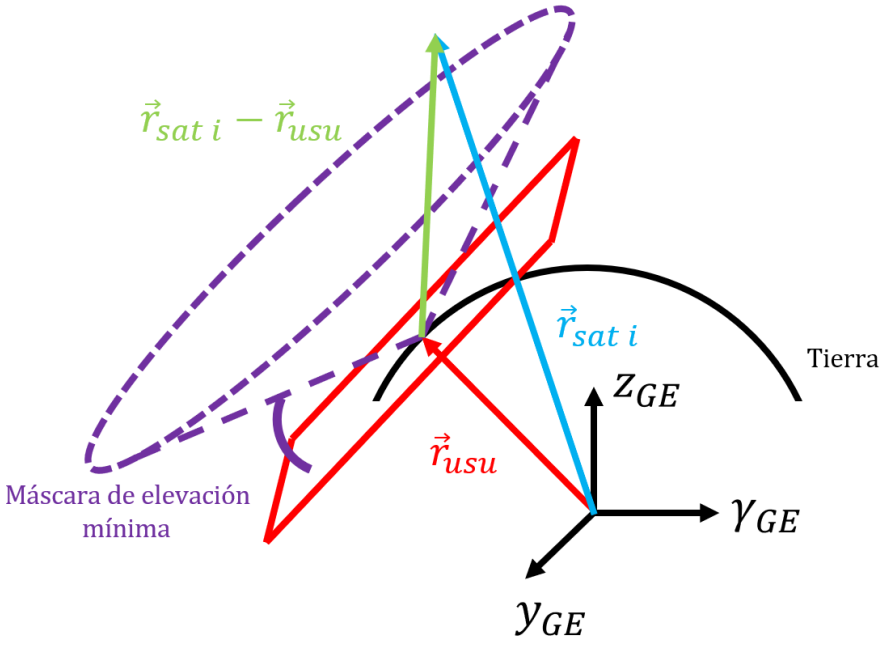


Figura 5.8: Diagrama de la lógica de la función SAT_Visible [Elaboración propia]

```
function visible_vector = SAT_Visible(obj,sat_obj)
%La funcion sat_visible determina la visibilidad del satelite de
%coordenadas satelite en base a una mascara de elevacion propia del
%receptor. El resultado es una matriz de forma 1 x nsat con unos y ceros
%dependiendo de si el satelite es visible o no

    visible_vector = zeros(obj.GetREC,sat_obj.sat_count);
    rec_xyz = obj.GetLOC_XYZ;
    for i = 1 : obj.GetRECEIVERS
        for j = 1:sat_obj.sat_count
            v_dif = sat_obj.SAT(j).XYZ - rec_xyz(:,i);
            projection = transpose(v_dif) * rec_xyz(:,i);
            if(projection > 0)
                if(abs(acosd(projection/(norm(v_dif)*norm...
                        (rec_xyz(:,i))))) < 90 - obj.GetEmask)
                    visible_vector(i,j) = 1;
                end
            end
        end
    end
end
```

Una vez obtenido el vector de satélites visibles solo queda calcular la pseudodistancia geométrica entre el receptor (no se han considerado los efectos de multi-camino que podrían dar pie a errores en las medidas) y cada uno de dichos satélites para después añadirle los retrasos ionosférico y troposférico, así como el error de reloj del usuario (se desprecia el error de reloj del satélite debido a que este se corrige a través del mensaje de navegación). La implementación de esta función se puede ver a continuación.

```
function MEASURE (obj,sat_obj)
%Determina las mediciones de pseudodistancia que es capaz de medir el
%receptor, tanto si esta formado por un conjunto de receptores como si
%solo cuenta con uno (caso usuario). Estas medidas incluyen un cierto
%ruido debido a los efectos del reloj, ionosfera, etc.

    sat_vis = SAT_Visible(obj,sat_obj);
    rec_xyz = obj.GetLOC_XYZ;
    for j = 1:sat_obj.sat_count
        prod = 1;
        for i = 1 : obj.GetREC
            prod = prod * sat_vis(i,j);
        end
        sat_vis(:,j) = prod;
    end
    obj.RHO_MEAS = zeros(obj.GetREC,sat_obj.sat_count);
    for i = 1:obj.GetREC
        for j = 1:sat_obj.sat_count
            if (sat_vis(i,j)==1)
                obj.RHO_MEAS(i,j) = norm(sat_obj.SAT(j).XYZ - ...
                    rec_xyz(i)) + 2.99792458e5 * ...
                    sat_obj.SAT(j).offset/1e5 + 1e-3*randn;
            end
        end
    end
end
```

5.3.1. Receptor del usuario: prueba de un filtro de Kalman extendido

Para el receptor del usuario, que modela toda la interacción de este con el resto de la simulación, en primera instancia se ha implementado un filtro de Kalman extendido para estimar la posición del usuario mediante GNSS diferencial. Los filtros de Kalman ofrecen una respuesta óptima para sistemas lineales en los que aparece un ruido gaussiano. En este caso, debido a la naturaleza no lineal de las medidas de pseudodistancia, se debe implementar un filtro de Kalman extendido. Esta clase de filtros no está demostrado matemáticamente que sean óptimos, pero ofrecen buenas prestaciones si las estimaciones iniciales no incluyen un error considerable.

Nuestro objetivo es implementar un filtro de seis estados que permita estimar la posición y velocidad del receptor de usuario en coordenadas geocéntrico-ecuatoriales. Para ello, se asumirá que el receptor se mueve a una velocidad constante, pero se admitirá una cierta aceleración desconocida y que no será estimada. La Ecuación 5.13 muestra el sistema expresado en forma de espacio de estados y la 5.14 amplía estas expresiones sustituyendo los valores del sistema.

$$\dot{x} = Fx + w \tag{5.13}$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{y} \\ \ddot{y} \\ \dot{z} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \end{bmatrix} + \begin{bmatrix} 0 \\ u_s \\ 0 \\ u_s \\ 0 \\ u_s \end{bmatrix} \tag{5.14}$$

Donde F representa la matriz dinámica del sistema, que relaciona el vector de estados x con su derivada, $\dot{x}$ y w es un proceso de ruido blanco expresado en forma de vector, denominado en muchas ocasiones como ruido de proceso. Sus valores están recogidos como u_s. El vector de ruido de proceso permite transmitirle al filtro de Kalman que la dinámica descrita en las ecuaciones diferenciales puede no ser del todo correcta. En este caso, se utiliza para modelar una aceleración desconocida y que no necesariamente debe ser cero. Cuanto mayor sea el ruido del proceso, menor precisión tendrá el filtro, pero mayor será su robustez ante cambios bruscos de la dinámica del sistema.

Una vez vista la ecuación del sistema, describiremos la ecuación de medida, que se puede ver en la Ecuación 5.15 y se amplía en la 5.16. La matriz de medida, H, juega un papel similar a F y relaciona el vector de estados del sistema con las medidas, z. Al tratarse de medidas no lineales, se trata del jacobiano de las medidas de pseudodistancia evaluadas en el estado estimado, que será representado como $\hat{x}$. El vector de ruido de las medidas se modela con v y en este caso recoge todos los errores a la hora de calcular las pseudodistancias, como puede ser el retraso ionosférico. En el caso de las medidas, para representar las ecuaciones se han considerado m satélites visibles.

$$z = Hx + v \tag{5.15}$$

$$\begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_m \end{bmatrix} = \begin{bmatrix} \frac{\partial \rho_1}{\partial x} & 0 & \frac{\partial \rho_1}{\partial y} & 0 & \frac{\partial \rho_1}{\partial z} & 0 \\ \frac{\partial \rho_2}{\partial x} & 0 & \frac{\partial \rho_2}{\partial y} & 0 & \frac{\partial \rho_2}{\partial z} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \rho_m}{\partial x} & 0 & \frac{\partial \rho_m}{\partial y} & 0 & \frac{\partial \rho_m}{\partial z} & 0 \end{bmatrix} \cdot \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ z \\ \dot{z} \end{bmatrix} + \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_m \end{bmatrix} \tag{5.16}$$

En el caso de los ruidos introducidos, ya sea el ruido de proceso o el ruido de medidas, a pesar de que en las ecuaciones aparezcan como vectores no se va a trabar con estos, sino con las matrices Q y R respectivamente, que incluyen el valor esperado de la auto-correlación de estos ruidos, como se puede ver en las Ecuaciones 5.17 y 5.18.

$$Q = E\left[ww^T\right] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \Phi_{s1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Phi_{s2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Phi_{s3} \end{bmatrix} \tag{5.17}$$

$$R = E\left[vv^T\right] = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \sigma_m^2 \end{bmatrix} \tag{5.18}$$

Debido a que se aplica el valor esperado del resultado y en principio las medidas provenientes de cada satélite son independientes, se ha asumido que las covararianzas de estos productos son nulas. Ahora bien, hasta ahora se ha tratado el sistema como continuo, pero en realidad su naturaleza es discreta, ya que se emplean receptores digitales. Por este motivo, se deben discretizar las matrices mostradas. La ecuación fundamental del filtro de Kalman extendido cuando la matriz dinámica del sistema es lineal viene dada por la Ecuación 5.19 y siempre incluye un término de predicción (proyección de los estados hacia el futuro) y una parte de corrección, en la que se incluye el concepto de la ganancia de Kalman, K_k. Para un instante k, se tiene:

$$\hat{x}_k = \Phi_k \hat{x}_{k-1} + K_k \left(z_k - H\Phi_k \hat{x}_{k-1}\right) \tag{5.19}$$

Donde Φ_k es la versión discretizada de la matriz fundamental del sistema, $\Phi(t)$, también denominada matriz de transición, ya que relaciona el vector de estados en un instante de tiempo con el siguiente. En el caso de que no fuera lineal, no se haría uso de esta matriz sino que se implementaría un método numérico para integrar las ecuaciones diferenciales de los estados. Para obtenerla, se hace uso de las expresiones recogidas en la Ecuación 5.20. Para un tiempo de muestreo T_s, la matriz fundamental de este sistema se puede ver en la Ecuación 5.21.

$$\Phi(t) = \mathcal{L}^{-1}\left[(sI - F)^{-1}\right] = e^{Ft} = I + Ft + \frac{(Ft)^2}{2!} + \cdots + \frac{(Ft)^n}{n!} + \cdots \tag{5.20}$$

$$\Phi_k = \Phi(T_s) = \begin{bmatrix} 1 & T_s & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T_s & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.21}$$

En el caso de las matrices de ruido, la del ruido de medida mantiene su forma. No obstante, la matriz de ruido de proceso, Q, requiere de un cambio más sustancial, que se expresa en la Ecuación 5.22.

$$Q_k = \int_0^{T_s} \Phi(\tau)\, Q\, \Phi^T(\tau)\, d\tau = \begin{bmatrix} T_s^3/3 & T_s^2/2 & 0 & 0 & 0 & 0 \\ T_s^2/2 & T_s & 0 & 0 & 0 & 0 \\ 0 & 0 & T_s^3/3 & T_s^2/2 & 0 & 0 \\ 0 & 0 & T_s^2/2 & T_s & 0 & 0 \\ 0 & 0 & 0 & 0 & T_s^3/3 & T_s^2/2 \\ 0 & 0 & 0 & 0 & T_s^2/2 & T_s \end{bmatrix} \tag{5.22}$$

Finalmente, se listan en la Ecuación 5.23 las expresiones recursivas de Riccati [45], que permiten obtener los valores de la ganancia Kalman y las matrices de covarianza de los errores cometidos antes de la estimación, M_k, y después de la estimación, P_k. Estos valores, si el sistema fuera lineal y se tratara de un filtro de Kalman polinomial o lineal,nos permitirían evaluar los errores cometidos por el sistema para poder trasladarlo a un diagrama de Standford.

$$\begin{aligned} M_k &= \Phi_k \, P_{k-1} \, \Phi_k^T + Q_k \\ K_k &= M_k \, H^T \left(H \, M_k \, H^T + R_k \right)^{-1} \\ P_k &= \left(I - K_k \, H \right) M_k \end{aligned} \tag{5.23}$$

Dado que en este caso tratamos con un filtro extendido por la ecuación de las medidas, las matrices de estimación de los errores cometidos no nos brindan una aproximación fiel de estos. Como se indica en las expresiones de la Ecuación 5.23, los valores de la matriz P_k dependen de los valores de ruido de proceso y de medidas, los cuales son parámetros fijos. Esto deriva en una sobre estimación de los valores de error del filtro, los cuales se pueden ver en la Figura 5.9. El error estimado para un 95 % del tiempo (2σ) se mantiene en 45 m, pese a que el error real del sistema no supera en ningún momento los 20 m.

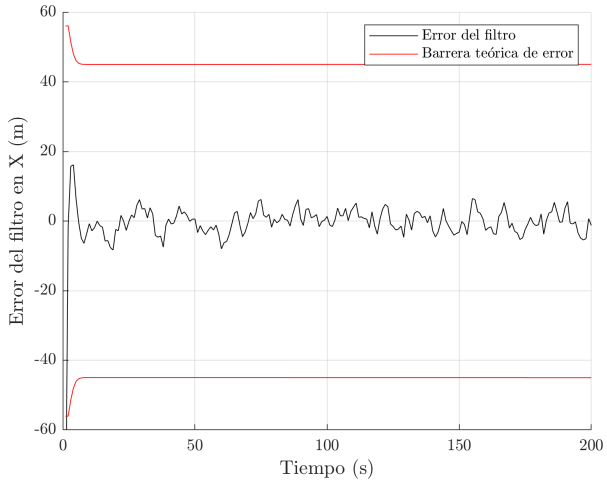


Figura 5.9: Errores estimados por la matriz P_k frente a errores reales [Elaboración propia]

Con el objetivo de estimar el error del sistema sin conocer el error real se optará por utilizar las medidas estimadas por el filtro con las medidas reales [46]. La diferencia entre estas medidas, también llamadas residuales, están altamente correladas con los errores del sistema, ya que es una medida directa de lo bien que sigue el filtro de Kalman lo que está ocurriendo en el sistema real. No obstante, como estos residuales son medidas ruidosas, para normalizar las medidas y acotar el error en al menos un 95 % del tiempo se hará uso de un filtro de ventana deslizante.

De esta forma, se calculará el percentil 95 % de la media de los residuales de cada satélite dentro del ancho de la ventana, que en este caso será de 50 medidas, para suavizar las curvas obtenidas y tener una cota conservadora del error. La Figura 5.10 muestra las nuevas barreras de error implementando este método. Cabe destacar que en la monitorización del sistema se comparará esta con el error real, que no estaría disponible para el usuario.

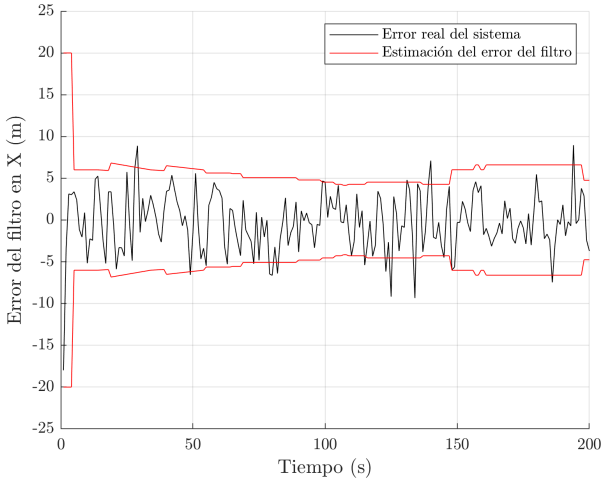


Figura 5.10: Errores estimados por el sistema y error real [Elaboración propia]

Como se puede apreciar en la Figura 5.10, la cota del error es consistente con el error del sistema. El primer escalón de 20 m de error ha sido añadido de forma artificial dado que el filtro necesita un periodo de 5 s para estabilizar sus medidas y ofrecer un valor de la posición sin un error considerable. Estudiando los picos que penetran en la barrera del error sobre un periodo de una hora con un ratio de medidas de 1 Hz se ha obtenido que el error cometido por el sistema se encuentra acotado por las barreras en un 97,033 % del tiempo, superando el objetivo del 95 %.

No se debe confundir el error estimado del sistema con el nivel de protección del mismo, que se establece en base al error estimado. Si se cometiera el error de confundirlos, el sistema resultante se encontraría en cada uno de estos eventos dentro de la región de información contradictoria, dado que el nivel de protección sería inferior al error del sistema.

Una vez estudiada la base teórica se explicará el funcionamiento de la implementación en MATLAB, la cual está principalmente ubicada dentro de la función `ESTIMATE`. El esquema de funcionamiento del filtro puede verse en la Figura 5.11 y se divide en distintas fases:

- **Fase de inicialización:** simula el momento en el que se pone en marcha el sistema de posicionamiento y se consigue enganchar la frecuencia de la estación GBAS. Inicializa todos los parámetros del filtro que no dependen del tiempo y ofrece una estimación de aquellos que sí. Debido a que el sistema GBAS se usa en regiones limitadas, este error inicial no podrá ser superior a los 20 km en el peor de los casos. No obstante, se asumirá un error inicial de posicionamiento de 20 m al considerar que el usuario cuenta con un ABAS.
- **Bucle temporal:** representa el periodo de tiempo en el cual el programa va a estar repitiendo el bucle de funcionamiento. Así, cuando se decida desconectar la simulación pasado un tiempo arbitrario t_f, dejará de funcionar.
- **Obtención de medidas:** cuando se actualiza la posición de los satélites, se debe simular la obtención de las medidas de pseudodistancia crudas de los satélites y aplicarles las correcciones diferenciales recibidas de la estación de tierra GBAS.

- **Ecuaciones de Ricatti y estimación:** esta representa la parte de cálculo principal del filtro. Debe calcular la ganancia de Kalman y las matrices de covarianza de los errores cometidos para, seguidamente, estimar la posición y velocidad del usuario.
- **Almacenar y enviar resultados:** esta parte no es intrínsecamente necesaria para el funcionamiento del filtro pero en esta implementación se incorpora para conectar este módulo con el resto de componentes de la simulación, como puede ser la interfaz gráfica para mostrar los parámetros de errores, etc.

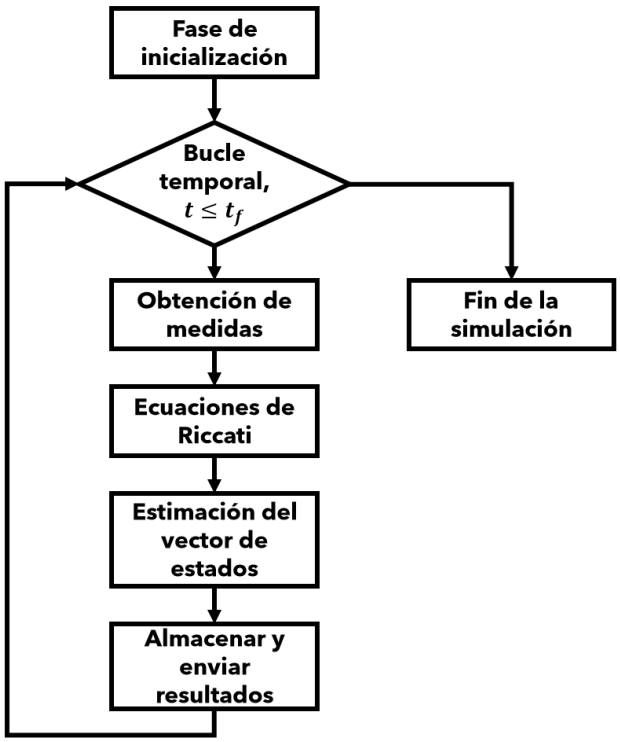


Figura 5.11: Esquema de funcionamiento del filtro de Kalman extendido [Elaboración propia]

No obstante, los resultados obtenidos en la Figura 5.10 son válidos cuando el ruido del sistema se modela como un ruido gaussiano de media nula. Cuando se aplica el modelo descrito para el error atmosférico el filtro comienza a presentar malas estimaciones, muy dependientes de la región en la que nos encontremos dentro del diagrama del retraso ionosférico de la Figura 5.7. Es por este motivo que en lugar de un filtro de Kalman se ha implementado un filtro de mínimos cuadrados, el cual se puede ver en la función `ESTIMATE_LQM` de la clase `USER`, la cuál será comentada más adelante.

5.3.2. Algoritmo de cálculo de niveles de protección

Uno de los puntos principales del sistema de aumentación de la señal GNSS es la implementación de unos niveles de protección que garantizan la integridad del sistema, siendo estos siempre superiores al error del sistema (desconocido para el mismo). El cálculo de los niveles de protección en la simulación se realiza tanto a nivel de usuario como de estación, pero estos convergen en la función `ESTIMATE_LQM()`:

```
function EST_LQM(obj,obj_sat)
%Estima la posicion del usuario haciendo uso de un filtro de minimos
%cuadrados y ofrece una aproximacion del error que comete el filtro
    index = 1;
    c = physconst('LightSpeed')*1e-9;
    bias_usu = 11.3; % us
    bias_est = 7;    % us
    for i = 1:obj_sat.sat_count
        if (obj.RHO_MEAS(i)~=0)
            rho_valid(index) = obj.RHO_MEAS(i) + c * bias_usu;
            sat_pos(:,index) = obj.SAT_XYZ(:,i);
            index = index + 1;
        end
    end
    m = index - 1; % Numero de medidas de satelites visibles
    for iter = 1:30
        A = [0,0,0,0];
        rho_res = [0;0];
        for i = 1:m
            A(i,1) = -(sat_pos(1,i) - obj.est_pos_XYZ(1))/(rho_valid(i)...
                - c*bias_est);
            A(i,2) = -(sat_pos(2,i) - obj.est_pos_XYZ(2))/(rho_valid(i)...
                - c*bias_est);
            A(i,3) = -(sat_pos(3,i) - obj.est_pos_XYZ(3))/(rho_valid(i)...
                - c*bias_est);
            A(i,4) =  c;
            rho_res(i,1) = rho_valid(i) - (norm(sat_pos(:,i) - ...
                obj.est_pos_XYZ) + c*bias_est);
        end
        medidas_res = (transpose(A)*A)\(transpose(A)*rho_res);
        obj.est_pos_XYZ  = obj.est_pos_XYZ + medidas_res(1:3,1);
        bias_est = bias_est + medidas_res(4,1);
    end
    obj.res = medidas_res(1:3,1);
    obj.real_error = obj.LOC_XYZ - obj.est_pos_XYZ;

    obj.pl_temp = [0;0;0];
    nlos = 0;
    for j=1:length(sat_pos)
        if rho_valid(j)~=0
            pseudo_est = norm(sat_pos(:,j) - obj.est_pos_XYZ) +c*bias_est;
            pl_raw = rho_valid(j) - pseudo_est;
            obj.pl_temp = obj.pl_temp + abs(pl_raw.*((sat_pos(:,j) - ...
                obj.est_pos_XYZ)/norm(sat_pos(:,j)...
                - obj.est_pos_XYZ)));
            nlos = nlos + 1;
        end
    end
    obj.pl_temp = obj.pl_temp * 1/nlos;
    obj.pl_user(:,end+1) = obj.pl_temp;

    w_len = 25;
    perc = 95;
    initial = 10;
    if length(obj.PL) < initial
        obj.PL(:,end+1)=[18;8]*1e-3;
    elseif length(obj.PL) <= w_len
        pl_x = 1.5*max(prctile(obj.pl_user(1,initial:end),perc) + ...
            randn*1.75e-3,prctile(obj.GBAS_PL(1,initial:end),perc)*2);
        pl_y = 1.5*max(prctile(obj.pl_user(2,initial:end),perc) + ...
```

```
61              randn*1.75e-3,prctile(obj.GBAS_PL(2,initial:end),perc)*2);
62          pl_z = 1.5*max(prctile(obj.pl_user(3,initial:end),perc)*1.3 + ...
63              randn*0.7e-3,prctile(obj.GBAS_PL(3,initial:end),perc)*2);
64          obj.PL(:,end+1) = [norm([pl_x pl_y]) sqrt(2)*pl_z];
65      else
66          pl_x = 1.5*max(prctile(obj.pl_user(1,(end-w_len):end),perc) + ...
67              randn*1.75e-3,prctile(obj.GBAS_PL(1,(end-w_len):end),perc)*2);
68          pl_y = 1.5*max(prctile(obj.pl_user(2,(end-w_len):end),perc) + ...
69              randn*1.75e-3,prctile(obj.GBAS_PL(2,(end-w_len):end),perc)*2);
70          pl_z = 1.5*max(prctile(obj.pl_user(3,(end-w_len):end),perc)*1.3+...
71              randn*0.7e-3,prctile(obj.GBAS_PL(3,(end-w_len):end),perc)*2);
72          obj.PL(:,end+1) = [norm([pl_x pl_y]) sqrt(2)*pl_z];
73      end
74  end
```

El proceso de cálculo (simplificado) de los niveles de protección es el siguiente:

- La estación de tierra del GBAS proyecta los errores medidos de los satélites en vista sobre ella misma para calcular una media de error por eje. Estos parámetros se computan tanto para el plano horizontal (combinando los ejes X e Y) y el vertical.
- El usuario realiza este mismo proceso una vez ha estimado su posición con las correcciones del PRC de la estación y los almacena.
- Sobre una ventana de desplazamiento, se coge el valor del percentil 95 de ambos casos incluyendo ruido del sistema y se escoge el valor más alto, para tener una medida conservadora del nivel de protección.

En un sistema real el proceso se complica haciendo uso los mensajes emitidos por la estación de tierra, que incluye una serie de matrices que se calculan para cada uno de los satélites en vista y que permiten obtener valores de UDRE y GIVE. En este caso, dada la extensión del proyecto, este proceso se ha simplificado a computar una serie de medias de los errores sobre los ejes en que se cometen (recordemos que dado que no se aplican errores como el efecto multipath, todos los errores se aplican en la línea de vista entre receptor y satélite).

5.4. Comunicación con X-Plane

El objetivo principal de esta simulación es poder estimar las prestaciones de un sistema GBAS sin la necesidad de tener datos de campo reales. Esta simulación tiene, por lo tanto, un nivel de abstracción mayor que otras soluciones de monitorización, que están pensados para el momento de implementación.

Es por este motivo que los valores de entrada para el programa tienen que ser puros, en el sentido de que se plantea tener un control total sobre la situación real del usuario para poder compararlo realmente con el sistema. Es en este punto en el que entra el simulador de vuelo X-Plane 11. A través de una serie de librerías específicas para MATLAB y su interfaz UDP, este programa puede enviar las coordenadas exactas de un usuario sintético. La interfaz de X-Plane 11 mientras se ejecuta puede verse en la Figura 5.12.

Figura 5.12: Interfaz de X-Plane 11 durante una simulación [Elaboración propia]

5.4.1. La interfaz de comunicación UDP

X-Plane, además de permitir el uso de pluggins, incorpora una interfaz de comunicación UDP (User Datagram Protocol), la cual permite conectar aplicaciones como MATLAB a esta e intercambiar información. Esta interfaz está principalmente diseñada para el control de aeronaves y las pruebas generales de algoritmos de control.

No obstante, llevado a un extremos más simple, en este caso solo se tomarán tres datos de forma periódica: la latitud, la longitud y la altitud de la aeronave. Para poder habilitar esta funcionalidad simplemente hay que seleccionar estos outputs en la lista disponible dentro del simulador y habilitar los puertos de comunicación (49000 por defecto para X-Plane). Gracias a las librerías ya implementadas en MATLAB la recepción de datos se puede implementar en unas líneas:

```
1 function update_LOC(obj)
2 %Recibe informacion de la posicion real del usuario de X-Plane y la
3 %convierte al formato correspondiente
4     datos = dref_readXPLANE();
5     obj.LOC.lat = datos(1);
6     obj.LOC.lon = datos(2);
7     obj.LOC.alt = datos(3)/3280.84; % De ft a km
8     obj.LOC_XYZ = latlon2XYZ(obj);
9 end
```

5.5. Interfaz gráfica y manejo de la simulación: magicSIMMON

Esta sección esta dedicada al nexo de unión de todos los módulos de los que cuenta la herramienta de simulación, que ha recibido el nombre de MATLAB GBAS Integrity Corrections & Simulation Monitor, magicSIMMON para abreviar. En primer lugar se describirá la interfaz gráfica de la misma, para pasar a la introducción de parámetros de la simulación y por último explicar la estructura del script que controla la simulación.

5.5.1. La interfaz gráfica de la aplicación

La herramienta magicSIMMON ha sido desarrollada enteramente dentro del entorno de MATLAB en la forma de un script principal que controla todos sus módulos acompañado de una serie de funciones que implementan los diferentes actores dentro de la simulación y algunos aspectos comunes como la toma de medidas o la obtención de los errores atmosféricos simulados. La interfaz gráfica de magicSIMMON ha sido generada en la forma de un objeto de tipo figura de MATLAB que se actualiza de forma dinámica a medida que se procesan los datos. Esta puede verse en la Figura 5.13.

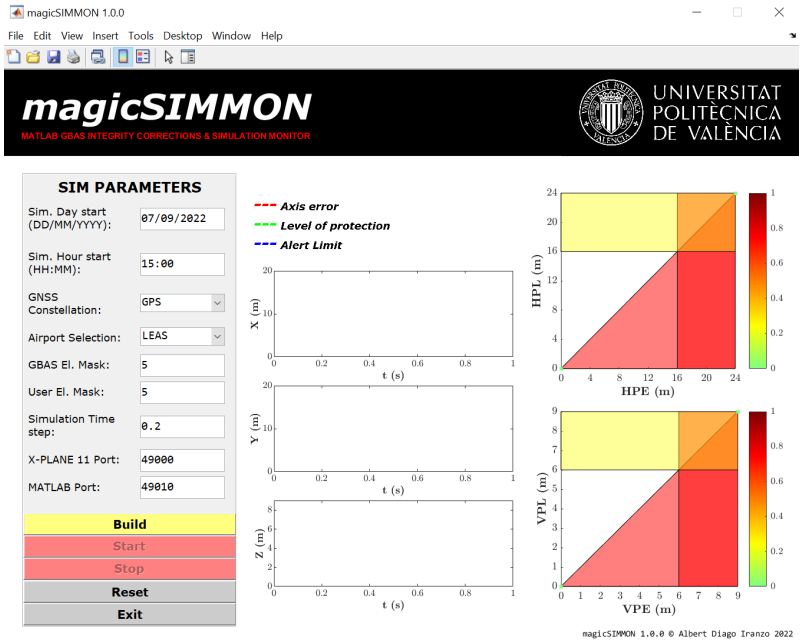


Figura 5.13: Interfaz gráfica de magicSIMMON [Elaboración propia]

El motivo por el cual se ha desarrollado siguiendo esta estructura en lugar de hacer uso de los módulos específicos de MATLAB para la generación de interfaces gráficas o GUIs como son App Designer o su antecesora, GUIDE, ha sido la diferencia de latencia de cada una de estas opciones. Los objetos de tipo figura nativos de MATLAB están desarrollados utilizando Java mientras que los módulos citados implementan tecnologías basadas en HTML y CSS, lo cual aumenta el tiempo de procesado de cada uno de los fotogramas, haciendo que no pueda rendir en los tiempos requeridos. Al hacer uso de figuras y de actualizar los datos de estas directamente accediendo a los objetos que las forman en lugar de utilizar comandos como plot se puede conseguir alcanzar frecuencias de refresco del orden de 5 Hz y mostrar los resultados de los datos obtenidos de X-Plane con una latencia del orden de décimas de segundo.

La interfaz gráfica se divide en dos secciones claramente separadas: por un lado se encuentra la sección de SIM PARAMETERS que, como su propio nombre indica, se utiliza para introducir los diferentes parámetros de configuración de la simulación antes de construirla. La sección de la derecha muestra los resultados de la simulación. Consta de cinco gráficas agrupadas en dos partes; por un lado se muestran los diferentes errores y nivelees de protección que se ofrecen a nivel de usuario en los tres ejes locales la usuario (X,Y y Z).

Por otro lado se incluyen los diagramas de Standford de los planos horizontal y vertical, los cuales constituyen la parte de monitorización del sistema y permiten analizar la disponibilidad, integridad y precisión del sistema. La continuidad del sistema no puede ser medida de forma precisa con la simulación implementada ya que está orientada a simular periodos cortos de tiempo (i.e.e una aproximación por simulación) y no incluye suficientes muestras como para conseguir cifras significativas de este parámetro. No obstante, existen dos alternativas para estimar este parámetro con los medios disponibles:

- Adaptar el modelo de simulación para calcular diferentes aproximaciones con diferentes configuraciones de satélites y rutas hasta tener un número significativo de muestras como para poder evaluar el sistema en conjunto.
- Utilizar un modelo de continuidad sintética basado en las medidas de disponibilidad del sistema. Existen diferentes curvas experimentales que miden el riesgo de continuidad en una determinada operación en base a la disponibilidad del sistema.

Para una determinada operación podemos medir la disponibilidad y la continuidad de forma discreta observando los resultados de los diagramas de Standford. Siempre y cuando no se presenten épocas en la región no disponible se podrá decir que la disponibilidad del sistema ha sido del 100 % así como que el servicio ha sido continuo.

5.5.2. Introducción de condiciones generales en la aplicación

La introducción de parámetros de entrada de la simulación se gestiona a través de una serie de objetos de clase uicontrol de MATLAB. Uno de los campos de esta clase es string, que recoge el texto que se introduce en estos. Los campos que permiten controlar la simulación son los siguientes:

- **Simulation Day Start:** fecha de inicio de la simulación en formato DD/MM/YYYY. Permite establecer, junto a la hora de la simulación, el instante inicial en el cual se comenzará a realizar la simulación.
- **Simulation Hour Start:** configura la hora exacta de comienzo de la simulación en formato HH:MM y 24 horas. El efecto de estos parámetros en los ficheros es la configuración de la constelación GNSS elegida, haciendo que se integre su posición desde la fecha de emisión de los ficheros TLE hasta el valor establecido. Para una mayor veracidad de los resultados de las órbitas se recomienda usar una fecha cercana a la actual, debido a que los archivos TLE, como se ha mencionado anteriormente, se extraen de una página web que los actualiza diariamente.
- **GNSS Constellation:** permite escoger la constelación GNSS de referencia para la simulación. Permite seleccionar dos constelaciones, GPS o GLONASS, quedándose fuera de la simulación las posibilidades de GALILEO por no tener todos los servicios operativos en el momento de redacción de este documento o la opción de multiconstelación.
- **Airport Selection:** escoge la ubicación de la estación de tierra del sistema GBAS. En la herramienta están implementados LEAS, LEMD y LEVC; pero estos campos se podrían ampliar a otros aeropuertos o ubicaciones según se requiera.

- **GBAS & User Elevation Mask:** configura la máscara de elevación de los diferentes receptores de la simulación, permitiendo separar la máscara de elevación del usuario de la del sistema de tierra del GBAS.
- **Simulation Time Step:** establece el paso de tiempo con el que se actualizarán tanto las medidas de X-Plane así como el paso de integración del filtro y de los satélites.
- **X-Plane 11 & MATLAB Ports:** determina los puertos que se utilizarán para realizar la simulación y conectar X-Plane con MATLAB.

5.5.3. Manejador de la simulación

El archivo `magicSIMMON_Manager.m` controla toda la herramienta y por ende la simulación. Consta de una primera fase en la que se genera la interfaz gráfica vacía y una segunda en la que el usuario puede introducir los datos para comenzar la simulación. Una vez se inicia esta segunda fase los diferentes estados de la aplicación se encuentran guiados a través de los botones de la esquina inferior izquierda de la interfaz.

En primer lugar se deben introducir los parámetros de configuración de la simulación para poder pasar a construir el modelo e inciar las clases con el botón de Build. El resto de botones se encuentran bloqueados para evitar fallos debidos a cambios de estado no permitidos (i.e. parar la simulación antes de que esta comience). La transición entre estados y su representación en la interfaz gráfica se pueden ver en la Figura 5.14.

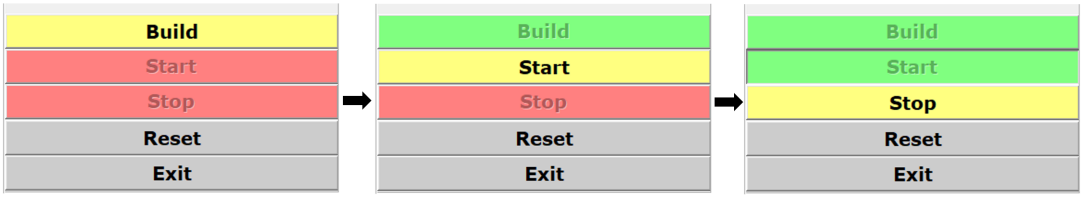


Figura 5.14: Transición entre estados de la herramienta de magicSIMMON [Elaboración propia]

En cuanto a la gestión de los estados de la herramienta el programa implementado cuenta con un bucle semi-infinito que se ejecute siempre y cuando no se active el flag del botón de salida, que cierra la herramienta. Para simplificar la implementación de las funciones de callback de los elementos interactivos la gestión de los eventos de pulsación de botones se gestiona a través de flags y sus valores se actualizan a través del comando drawnow, ya que MATLAB comprueba el estado de los eventos al realizar este comando. Dentro de dicho bucle se va comprobando en cada iteración los valores de las flags de cada botón y se realizan las acciones pertinentes:

- **Build:** recoge los valores de entrada introducidos en SIM PARAMETERS y los utiliza para inciar las clases de usuario, constelación y GBAS que gestiona la herramienta y que han sido descrita a lo largo del presente capítulo.
- **Start:** una vez construido el modelo de la simulación al presionar el botón de Start comienza el bucle de toma de medida y representación. Este no tiene límite de tiempo.

- **Stop:** detiene la toma de medidas y deja de actualizar las gráficas de la herramienta.
- **Reset:** vuelve a la configuración inicial de las gráficas y desecha las clases creadas hasta el momento.
- **Exit:** cierra la herramienta.

Capítulo 6

Resultados de la simulación

6.1. Configuración de la simulación

Con el objetivo de probar la herramienta desarrollada y evaluar las prestaciones del sistema GBAS presentado, se realizará una simulación con los siguientes parámetros de entrada:

- **Aproximación:** CAT I
- **Día de simulación:** 07/09/2022
- **Hora de inicio de la simulación:** 15:00 UTC
- **Constelación seleccionada:** GPS
- **Aeropuerto seleccionado:** LEAS
- **Máscara de elevación de la estación GBAS:** 5º
- **Máscara de elevación del usuario:** 5º
- **Paso temporal de la simulación:** 0.2 s
- **Puerto de X-Plane 11:** 49000
- **Puerto de MATLAB:** 49010
- **Aeronave seleccionada:** Cessna 172-SP

6.2. Prestaciones obtenidas

Los resultados directos de la herramienta magicSIMMON son gráficas de errores y los diagramas de Stanford para el plano tanto vertical como horizontal. Estos se pueden ver en las Figuras 6.1, 6.2a y 6.2b. Asimismo, se calcularán los percentiles principales de los errores obtenidos para la aproximación y los valores de los riesgos de continuidad y de disponibilidad.

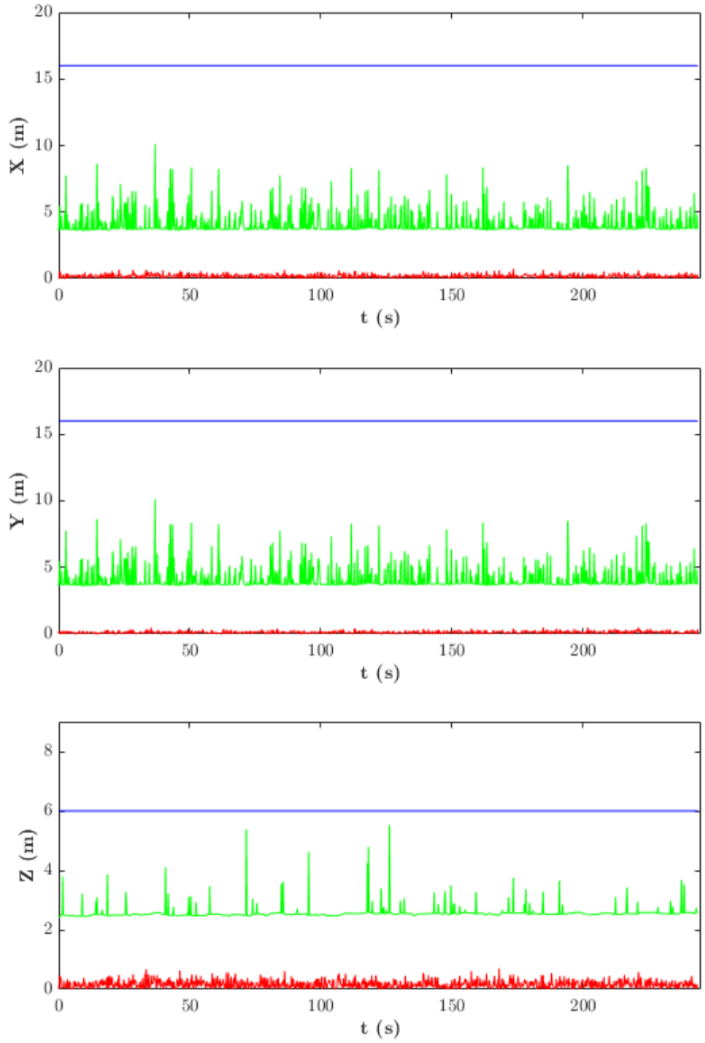


Figura 6.1: Errores calculados por magicSIMMON. En rojo se representa el error real, en verde el nivel de protección y en azul el límite de alerta [Elaboración propia]

Como se puede ver en los gráficos, la simplificación realizada para calcular los niveles de protección ha resultado ser demasiado conservadora. El error máximo en los ejes no alcanza el metro en magnitud y sin embargo el nivel de protección es en todo momento superior a los 2 m para el plano vertical y 4 en el caso del horizontal.

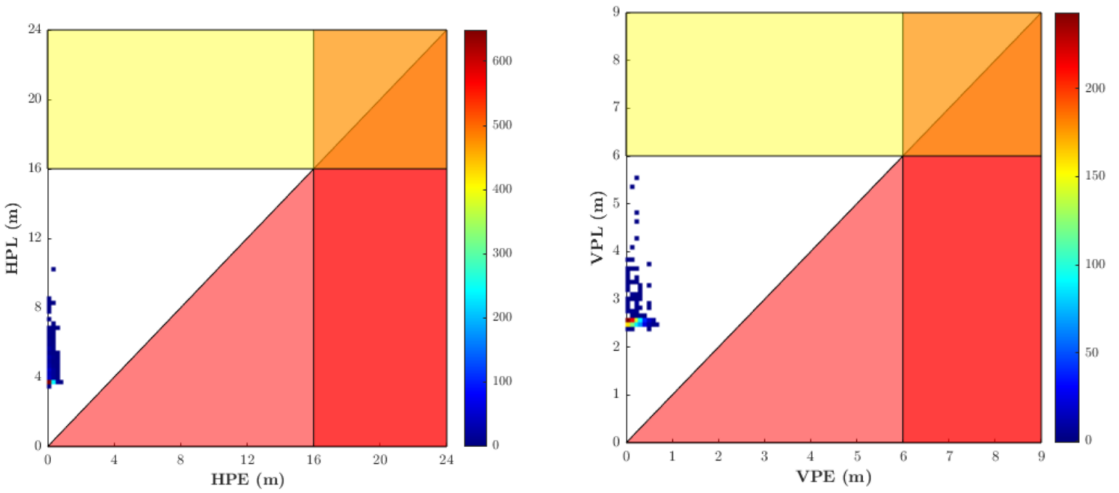
	Error real del sistema		
Sigma	**Eje X (m)**	**Eje Y (m)**	**Eje Z (m)**
1σ (68 %)	0.2301	0.1483	0.2164
2σ (95 %)	0.4350	0.2874	0.4131
3σ (99.7 %)	0.6603	0.4405	0.6119
máx. (100 %)	0.7387	0.4798	0.6985

Tabla 6.1: Percentiles de error en los distintos ejes

	Nivel de protección del sistema		
Sigma	**Eje X (m)**	**Eje Y (m)**	**Eje Z (m)**
1σ (68 %)	3.7911	3.7911	2.5498
2σ (95 %)	5.7605	5.7605	2.67021
3σ (99.7 %)	8.3496	8.3496	4.4863
máx. (100 %)	10.1191	10.1191	5.5212

Tabla 6.2: Percentiles de nivel de protección en los distintos ejes

No obstante, este ajuste también se ha realizado para albergar ciertos errores de transición de satélites, que no se observan en la simulación realizada debido al corto tiempo que ocupa (una operación frente al periodo orbital de un satélite). Las Tablas 6.1 y 6.2 recogen los percentiles de estos.



(a) Plano Horizontal (b) Plano vertical

Figura 6.2: Diagramas de Stanford de la simulación [Elaboración propia]

Como se puede ver, en términos del Diagrama de Stanford el comportamiento conservador del nivel de protección hace que no haya ninguna época de las 1300 analizadas que haga que el sistema pase a no estar disponible. Por lo tanto, para la operación analizada, el riesgo de continuidad así como de integridad sería de 0, con una disponibilidad del 100 %, lo cual haría que el servicio CAT I estuviera disponible en todo el recorrido de la operación. No obstante, estos valores aislados no son suficientes para validar a nivel operacional el sistema, ya que se debería testear para todo el rango de geometrías de satélites, con un periodo de observación mucho mayor al implementado en esta simulación.

Capítulo 7

Conclusiones generales del proyecto

A lo largo del presente Trabajo de Fin de Grado se han discutido los diferentes sistemas de aumentación de la señal GNSS actuales y se ha profundizado en el GBAS. La navegación aérea en los últimos años cada vez se mueve más hacia el diseño de rutas y operaciones que sean independientes de las radioayudas utilizadas y que requieran la menor infraestructura posible. En este sentido, la tendencia hacia los sistemas y ayudas basadas en navegación inercial y GNSS se vé beneficiada.

Con el objetivo de ampliar el conocimiento en los diferentes actores que forman parte del proceso de determinación de la posición en uno de estos sistemas de aumentación, se ha desarrollado una herramienta llamada magicSIMMON para poder evaluar de forma sencilla las prestaciones que puede ofrecer un sistema GBAS sin tener datos de implementación reales. Los resultados obtenidos han superado los requisitos para ofrecer servicio de aproximaciones de precisión CAT I. No obstante, se han detectado una serie de defectos y posibles mejoras que harían estos resultados más cercanos a la realidad, y como ejercicio de autocrítica, se listan a continuación:

- **Emisión de mensajes cuantificados por parte de la estación GBAS:** la gestión de las correcciones en la presente implementación de la herramienta traslada directamente estas medidas crudas al usuario, sin ninguna limitación ni de tiempo ni de precisión. En el caso de una estación real, los mensajes emitidos por esta tienen unos tiempos determinados y no pueden ser actualizados a la misma frecuencia que las medidas del usuario.

  Este factor hace que las correcciones reales de la estación sean menos precisas que las ofrecidas en la herramienta, lo cual degradaría las prestaciones hasta un punto más cercano al estado del arte del sistema.

- **Uso de medidas suavizadas con medidas de fase provenientes de diferentes receptores:** tal y como se ha explicado al hablar de la infraestructura del sistema GBAS, las estaciones de tierra suelen contar cuatro antenas receptoras, las cuales, además de servir a la redundancia total del sistema, se utilizan para suavizar las correcciones calculadas al provenir de la ponderación de todas estas.

  Asimismo, las medidas de fase de estas pueden utilizarse para suavizar el resultado final, cosa que no ha sido incluída en esta simulación, ya que las medidas han sido

puramente de pseudodistancia con un único receptor en la estación del GBAS.

- **Implementación de DFMC:** con el inminente despliegue operativo de la constelación de satélites GALILEO y la implementación de la señal L5 de forma más extendida, podría ser interesante incluir la posibilidad de hacer uso de un modelo de Doble Frecuencia, Multi-Constelación (DFMC) para mejorar las prestaciones del sistema, para aquellos usuarios que contaran con el receptor necesario. Esta posibilidad ya se está comenzando a plantear, por ejemplo, en el sistema SBAS de SouthPAN para Australia y Nueva Zelanda.
- **Errores de código y multi-camino:** no incluir cualquier fuente de error puede hacer que las prestaciones finales del sistema sean superiores a aquellas que realmente se observarían de desplegarse. El caso del error del multicamino es especialmente importante debido a que se trata de un error que no se aplica directamente sobre la línea de vista e incrementaría el error en el plano horizontal.

Bibliografía

[1] OACI, «ICAO Economic Impact Analysis of COVID-19 on Civil Aviation,» ICAO, inf. téc., 2020, Online. dirección: https://www.iea.org/data-and-statistics/charts/world-air-passenger-traffic-evolution-1980-2020.

[2] L. Sempere, *Radioayudas convencionales*, Transparencias de la Universitat Politècnica de València de la asignatura de Infraestructuras para la Navegación Aérea, 2021.

[3] OACI, *Anexo 6 al Convenio sobre Aviación Civil Internacional: Operación de aeronaves, Parte I - Transporte aéreo comercial internacional*, 10.ª ed. Quebec: OACI, jun. de 2016.

[4] A. Vidal, *Sistemas de aterrizaje instrumental*, Transparencias de la Universitat Politècnica de València de la asignatura de Infraestructuras para la Navegación Aérea, 2018.

[5] J. L. Berné Valero, N. Garrido Villén y R. Capilla Romá, *GNSS: GPS, GALILEO, GLONASS, BEIDOU : fundamentos y métodos de posicionamiento*, spa, ép. Colección Académica. Valencia: Editorial de la Universidad Politécnica de Valencia, 2019, ISBN: 8490487774.

[6] E. D. Kaplan y C. J. Hegarty, *Understanding GPS/GNSS: Principles and Applications*. Boston: Artech House, 2017, ISBN: 9781630810580.

[7] A. Vidal, *Sistemas GNSS*, Transparencias de la Universitat Politècnica de València de la asignatura de Infraestructuras para la Navegación Aérea, 2018.

[8] U. Fernández, «Técnicas de resolución de la ambigüedad de las medidas de fase en sistemas de navegación por satélite,» Tesis Doctoral, Universidad de Málaga, 2005.

[9] B. Hofmann-Wellenhof, H. Lichtenegger y E. Wasle, *GNSS - Global Navigation Satellite Systems : GPS, GLONASS, Galileo & more*. Deutschland: Springer Verlag, 2008.

[10] EUSPA, «European GNSS (GALILEO) Open Service: Service Definition Document,» European GNSS (GALILEO) Open Service, inf. téc., nov. de 2021.

[11] G. d. J. Cabinet Office, «Quasi-Zenith Satellite System: Interface Specification IS-QZss-PNT-004,» Office of National Space Policy, Gobierno de Japón, inf. téc., ene. de 2021.

[12] M. Watkins, *Performance Based Navigation (PBN)*, ICAO PBN Workshop for Air Traffic Controllers, jun. de 2017.

[13] OACI, *Performance-Based Navigation (PBN) Manual*, 3.ª ed. Quebec: OACI, 2008.

[14] ——, *Aprobación Operacional PBN: Introducción y visión*, Curso de la OACI para la Aprobación Operacional PBN, 2018.

[15] ——, *Anexo 10 al Convenio sobre Aviación Civil Internacional: Telecomunicaciones Aeronáuticas, Volumen I*, 6.ª ed. Quebec: OACI, jun. de 2006.

[16] I. Quintanilla, *BLOQUE III: Navegación Aérea*, Transparencias de la Universitat Politècnica de València de la asignatura de Navegación Aérea, Cartografía y Cosmografía, 2021.

[17] ESSP, «EGNOS Monthly Performance Report April 2022,» European Satellite Service Provider, inf. téc., abr. de 2022.

[18] I. Quintanilla, *GPS: MODELOS MATEMÁTICOS*, Transparencias de la Universitat Politècnica de València de la asignatura de Navegación Aérea, Cartografía y Cosmografía, 2021.

[19] OACI, *GBAS for ATCO*, online, jun. de 2017. dirección: https://www.icao.int/APAC/APAC-RSO/PBN%5C%20Workshop%5C%20for%5C%20Air%5C%20Traffic%5C%20Controllers/GBAS%5C%20for%5C%20ATCOv2.pdf.

[20] O. Weber, *GBAS Operational Implementations*, DFS experiences and capabilities, 2015.

[21] P. Nef, «Honeywell GBAS System Development,» Honeywell, inf. téc., jun. de 2019.

[22] P. Reines y A. Witt, «Honeywell SLS-4000 GBAS,» Honeywell, inf. téc., jun. de 2008.

[23] OACI, «Advisory Circular, AC 91-011,» Organización de la Aviación Civil Internacional, inf. téc., mayo de 2012.

[24] R. Geister, «Segmented Steep Precision Approaches Based on GLS,» *CEAS Aeronautical Journal*, vol. 2, dic. de 2011. DOI: 10.1007/s13272-011-0016-6.

[25] P. A. Ferreira, «Operational Evaluation of a GBAS System,» Trabajo de Fin de Máster, Universidad Técnica de Lisboa, sep. de 2007.

[26] J. Ackland, T. Imirich y T. Murphy, «Global Navigation Satellite System Landing System,» Boeing, Aeromagazine, inf. téc., ene. de 2003.

[27] IFALPA, «Ground Based Augmentation System (GBAS) and GBAS Landing System (GLS),» International Federation of Air Line Pilots' Associations, inf. téc., jul. de 2015.

[28] OACI, «GUIA PARA LA IMPLEMENTACION DE SISTEMAS DE AUMENTACION BASADOS EN TIERRA,» Organización de Aviación Civil Internacional, inf. téc., mayo de 2013.

[29] SFO, *San Francisco International Airport GBAS Procedure Review*, SFO Roundtable Technical Working Group, nov. de 2020.

[30] S. Saitoh, S. Fukushima, T. Yoshihara y N. Fujii, «Experimental GBAS Performance at the Approach Phase,» *Proceedings of the 2003 National Technical Meeting of The Institute of Navigation*, págs. 317-325, 2003.

[31] U. Bestmann, P. M. Schachtebeck, T. Feuerle y P. Hecker, «Making the Case for GBAS, Experimental Aircraft Approaches in Germany,» *Inside GNSS*, págs. 42-45, 2006.

[32] I. of Communications & Navigation, *Ground Based Augmentation System (GBAS)*. dirección: https://www.dlr.de/kn/en/desktopdefault.aspx/tabid-7567/12810_read-32118/.

[33] D. Vallado y P. Cefola, «Two-line element sets - Practice and use,» *Proceedings of the International Astronautical Congress, IAC*, vol. 7, págs. 5812-5825, ene. de 2012.

[34] J. A. M. Fernández, «Órbitas en tres dimensiones: Sistemas de Coordenadas,» *Universitat Politècnica de València*, mar. de 2021. dirección: http://hdl.handle.net/10251/164075.

[35] ——, «Órbitas en tres dimensiones: Elementos orbitales,» *Universitat Politècnica de València*, 2021. dirección: http://hdl.handle.net/10251/158425.

[36] ——, «Ecuaciones del movimiento en un marco inercial y en uno relativo,» *Universitat Politècnica de València*, 2021.

[37] T. G. René, «Orbital Diversity for Global Navigation Satellite Systems,» Tesis Doctoral, Universidad de Standford, 2017.

[38] P. B. Mangas, «Desarrollo de un simulador de vuelo orbital. Estudio de perturbaciones,» Trabajo de Fin de Grado, Universidad Carlos III de Madrid, jun. de 2013.

[39] S. Biswas, L. Qiao y A. Dempster, «Real-time On-board Satellite Navigation using GPS and Galileo Measurements,» oct. de 2014. DOI: 10.13140/2.1.1414.9125.

[40] S. Gaglione, *NeQuick model performance analysis for GNSS mass market receivers positioning*, UN/ICTP Workshops on GNSS, 2014.

[41] M. Reyes Quintana, N. Ramírez Sánchez, O. D. Bolívar Fonseca y A. Cárdenas Contreras, «Diseño de un mapa ionosférico como soporte al desarrollo y la implementación de GBAS, precisión de aproximación categoría I en Colombia,» *UD y la geomática*, n.º 5, págs. 15-28, dic. de 2012. DOI: 10.14483/23448407.3642. dirección: https://revistas.udistrital.edu.co/index.php/UDGeo/article/view/3642.

[42] EUSPA, «Ionospheric Correction Algorithm for GALILEO Single Frequency Users,» European GNSS (GALILEO) Open Service, inf. téc., 2016.

[43] B. Opperman, P. J. Cilliers, L.-A. McKinnell y R. Haggard, «Development of a regional GPS-based ionospheric TEC model for South Africa,» *Advances in Space Research*, vol. 39, págs. 808-815, 2007.

[44] G. Xu, *GPS: Theroy, Algorithms and Applications*. New York: Springer Books, 2007, ISBN: 9783540727149.

[45] P. Zarchan y H. Musoff, *Fundamentals of Kalman Filtering: A Practical Approach*, 3.ª ed. Reston, Virginia: American Institute of Aeronautics y Astronautics, Inc., 2013, ISBN: 9781600867187.

[46] J. Blanch, T. Walter y P. Enge, «Protection Level Calculation Using Measurement Residuals: Theory and Results,» ene. de 2005.

Capítulo 8

Pliego de condiciones

8.1. Disposiciones generales

El pliego de condiciones incluye la declaración y justificación de las distintas ordenanzas y condiciones en materia de seguridad, calidad, salud, entre otras, las cuales han sido consideradas a la hora de realizar el Trabajo de Fin de Grado: *Estudio teórico, simulación y validación de un sistema GBAS para el aeropuerto internacional de Asturias.*

El presente proyecto se basa en el desarrollo de una herramienta de *software* y su correspondiente memoria por lo que ha ha sido enteramente realizado en un ordenador, con el consecuente uso de una Pantalla de Visualización de Datos (PVD de aquí en adelante) mientras se permanece sentado en sesiones largas. Este tipo de actividades pueden conllevar un riesgo para la salud de los trabajadores como se recoge en el Real Decreto 488/1997. Este documento dictamina los requerimientos en términos de seguridad y salud conforme la Ley 31/1995 de Prevención de Riesgos Laborales, del 8 de noviembre.

8.2. Condiciones de ejecución

Engloban todos los aspectos relacionados con la realización del propio proyecto así como la estructura del mismo. De acuerdo con lo descrito anteriormente y con el objetivo de mitigar los riesgos del uso continuado de PVDs se ha limitado el número de horas diarias de exposición a estos dispositivos con parones intermitentes para relajar la vista. Asimismo, se ha mantenido una iluminación uniforme para evitar daño por destellos del PVD. Por otro lado, ya que el proyecto se realiza en gran parte sentado, se ha hecho uso de una silla que cumple con el estándar UNE 89401-1:2021 para sillas de oficina.

El presente proyecto se realiza bajo la supervisión del tutor del mismo. Queda bajo la responsabilidad del ingeniero redactor del documento tener una base de conocimientos suficiente para poder interpretar todos los datos recogidos y realizar tomas de decisiones responsables con estos. El tutor del proyecto, a su vez, debe tener amplios conocimientos en el campo de los sistemas de aumentación de la señal GNSS.

8.3. Condiciones materiales

8.3.1. Software utilizado

A continuación se listan los diferentes programas informáticos que se requieren para el desarrollo del proyecto en su forma completa así como su justificación:

- **MATLAB R2019b:** producto software de cómputo numérico y programación en el cual se ha desarrollado la herramienta de simulación del sistema GBAS.
- **X-Plane 11:** simulador de vuelo de uso comercial a través del cual se han obtenido las medidas necesarias para realizar la simulación.
- **Google Chrome:** navegador utilizado tanto para la búsqueda de información así como la redacción de la memoria a través de la plataforma gratuita de Overleaf.
- **Microsoft PowerPoint:** aplicación ofimática para la generación de esquemas para la memoria así como la presentación de la misma.

8.3.2. Hardware mínimo

El hardware que ha sido necesario para la realización de la memoria ha sido únicamente un ordenador portátil. Este ha de ser capaz de ejecutar los programas de software mencionados en el apartado anterior. Entre ellos, los más exigentes son MATLAB y X-Plane, y debido a que se tienen que ejecutar simultáneamente, los requisitos mínimos del equipo vendrán marcados por estos programas:

- Sistema operativo Windows 10 de 64 bits
- Procesador x86-64 con al menos cuatro núcleos lógicos y soporte para instrucciones AVX2
- 8 GB de memoria RAM
- Tarjeta gráfica compatible con DirectX 11 y al menos 512 MB de memoria VRAM
- 28 GB de espacio disponible, preferiblemente SSD

8.3.3. Material académico adicional

A lo largo del proyecto se hará uso de documentos oficiales con diferentes normativas, trabajos académicos, entre otros. Todos estos deben estar debidamente referenciados dentro del proyecto en el apartado de bibliografía.

Capítulo 9

Presupuesto del proyecto

Este capítulo está dedicado a desglosar el coste total de la elaboración del presente Trabajo de Fin de Grado. Se ha dividido en dos categorías: por un lado los costes directos, que engloban el equipo utilizado, así como las licencias y el personal implicado en el proyecto. Los costes indirectos se asocian con aquellos aspectos no directamente relacionados con el Trabajo pero que son necesarios para la realización del mismo, como pueden ser los costes de la luz y administración.

El tiempo total para la realización del proyecto ha sido de 360 horas distribuidas a lo largo de seis meses, lo que equivale a 12 créditos ECTS que marca la asignatura de Trabajo de Fin de Grado.

9.1. Costes directos

9.1.1. Coste de equipo

Como ya se ha mencionado anteriormente, el único equipo requerido para la realización del proyecto ha sido un portátil así como una serie de licencias. Debido a la compaginación del proyecto con un trabajo de jornada completa, el periodo de amortización de las licencias de software es mayor que si se hubieran concentrado en un periodo de tiempo menor. La Tabla 9.1 muestra el desglose de estos costes.

Costes de equipo				
Nombre del equipo	**Coste total (€)**	**Periodo de amortización total (meses)**	**Tiempo de uso (meses)**	**Importe total (€)**
Portátil HP OMEN 15-DC1000NS	1099.00	72	6	91.59
...				

Nombre del equipo	Coste total (€)	Periodo de amortización total (meses)	Tiempo de uso (meses)	Importe total (€)
Licencia MATLAB r2019b	840.00	12	6	420.00
X-Plane 11	59.99	6	6	59.99
Licencia Microsoft Powerpoint	79.00	12	6	39.50
		Subtotal		**611.08**

Tabla 9.1: Desglose de costes de equipo

9.1.2. Coste del personal

En este apartado se incluyen los costes derivados del personal involucrado en el proyecto y se recogen en la Tabla 9.2.

Costes de personal			
Recurso	**Coste (€/h)**	**Horas empleadas**	**Importe total (€)**
Ingeniero técnico titulado	10.00	360	3600.00
Doctor Ingeniero	25.00	10	250.00
	Subtotal		**3850.00**

Tabla 9.2: Desglose de costes de personal

9.1.3. Desglose de los costes directos

El total de los costes directos se recoge en la Tabla 9.3.

Costes directos	
Subsección	**Coste total (€)**
Costes de equipo	611.08
Costes de personal	3850.00
Subtotal	**4461.08**

Tabla 9.3: Desglose de costes directos

9.2. Costes indirectos

En esta sección se incluyen los costes que no están relacionados directamente con el proyecto pero que aportan al mismo. Entre ellos se incluye el gasto eléctrico y de oficina que se realiza durante la realización del proyecto. Estos costes se expresan como una fracción de los directos y se han estimado en un 10 % de estos. Se recogen en la Tabla 9.4.

Costes indirectos	
Subtotal	**446.11 €**

Tabla 9.4: Desglose de costes indirectos

9.3. Presupuesto total

Presupuesto total	
Costes directos	4461.08 €
Costes indirectos	446.11 €
IVA (21 %)	1030.51 €
Total	**5937.70 €**

Tabla 9.5: Desglose de costes totales

El coste total del proyecto de acuerdo con este presupuesto asciende a un total de: **CINCO MIL NOVECIENTOS TREINTA Y SIETE EUROS Y SETENTA CÉNTIMOS**

Apéndice A

Archivos de MATLAB

A.1. Script principal: magicSIMMON.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% magicSIMMON_MANAGER.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%             Variable cleanse and library definition                    %%
%-------------------------------------------------------------------------%
clear all
addpath('LIBRARIES/xplane');
addpath('FUNCTIONS');
%-------------------------------------------------------------------------%

%%                       Constants definition                             %%
%-------------------------------------------------------------------------%
final = 1000;
hal   = 16;
val   = 6;
t0    = 10;
dummy = [0 0];
nbins = 50;
sz    = 15;
i     = 1;
initialization_flag = 0;
build_flag = 0;
start_flag = 0;
stop_flag  = 0;
%-------------------------------------------------------------------------%

%%                        GUI initialization                              %%
%-------------------------------------------------------------------------%
fig = figure('Name','magicSIMMON 1.0.0','NumberTitle','off',...
    'Color','white','Position',[0 0 900 650]);
fig.Resize = "off";
fig.UserData = struct;
upv = axes(fig,'Position',[0.7 0.84974 0.3 0.151]);
image(upv,imread('marcaupv.png'));
set(gca,'Visible','off');

```

```
38 magicTITLE = annotation(fig,'textbox',[0 0.85 0.7 0.17],'string',...
39     ' magicSIMMON');
40 magicTITLE.HorizontalAlignment = 'left';
41 magicTITLE.VerticalAlignment = 'middle';
42 magicTITLE.Color = 'w';
43 magicTITLE.FontName = 'Verdana';
44 magicTITLE.FontSize = 30;
45 magicTITLE.FontWeight = 'b';
46 magicTITLE.FontAngle = 'i';
47 magicTITLE.BackgroundColor = [0 0 0];
48
49 magicsub = annotation(fig,'textbox',[0.015 0.798 1 0.17],'string',...
50     'MATLAB GBAS INTEGRITY CORRECTIONS & SIMULATION MONITOR');
51 magicsub.HorizontalAlignment = 'left';
52 magicsub.VerticalAlignment = 'middle';
53 magicsub.FontName = 'Arial';
54 magicsub.FontSize = 7.4;
55 magicsub.Color = 'r';
56 magicsub.FontWeight = 'b';
57 magicsub.LineStyle = 'none';
58
59 magicp = annotation(fig,'textbox',[0.6 0 0.4 0.1],'string',...
60     'magicSIMMON 1.0.0 c Albert Diago Iranzo 2022 ');
61 magicp.HorizontalAlignment = 'right';
62 magicp.VerticalAlignment = 'bot';
63 magicp.FontName = 'Consolas';
64 magicp.FontSize = 7.4;
65 magicp.LineStyle = 'none';
66
67 inpt = uibuttongroup(fig,'Visible','on','Position',...
68             [0.024 0.021*900/650 0.27 0.79]);
69
70 BTN_Build = uicontrol(inpt,'Style','togglebutton','String','Build',...
71     'Units','normalized','Position',[0 0.20 1 0.05],'Visible','on',...
72     'Enable','off');
73 BTN_Build.FontName = 'Verdana';
74 BTN_Build.FontSize = 10;
75 BTN_Build.FontWeight = 'b';
76 BTN_Build.BackgroundColor = [1.0 1 0.5];
77 BTN_Build.Value = 0;
78
79 BTN_Start = uicontrol(inpt,'Style','togglebutton','String','Start',...
80     'Units','normalized','Position',[0 0.15 1 0.05],'Visible','on',...
81     'Enable','off');
82 BTN_Start.FontName = 'Verdana';
83 BTN_Start.FontSize = 10;
84 BTN_Start.FontWeight = 'b';
85 BTN_Start.BackgroundColor = [1.0 0.5 0.5];
86 BTN_Start.Value = 0;
87
88 BTN_Stop = uicontrol(inpt,'Style','togglebutton','String','Stop',...
89     'Units','normalized','Position',[0 0.1 1 0.05],'Visible','on',...
90     'Enable','off');
91 BTN_Stop.FontName = 'Verdana';
92 BTN_Stop.FontSize = 10;
93 BTN_Stop.FontWeight = 'b';
94 BTN_Stop.BackgroundColor = [1.0 0.5 0.5];
95 BTN_Stop.Value = 0;
96
97 BTN_Reset = uicontrol(inpt,'Style','togglebutton','String','Reset',...
98     'Units','normalized','Position',[0 0.05 1 0.05],'Visible','on',...
```

```
99      'Enable','on');
100 BTN_Reset.FontName = 'Verdana';
101 BTN_Reset.FontSize = 10;
102 BTN_Reset.FontWeight = 'b';
103 BTN_Reset.BackgroundColor = [0.8 0.8 0.8];
104 BTN_Reset.Value = 0;
105
106 BTN_Exit = uicontrol(inpt,'Style','togglebutton','String','Exit',...
107     'Units','normalized','Position',[0 0 1 0.05],'Visible','on',...
108     'Enable','on');
109 BTN_Exit.FontName = 'Verdana';
110 BTN_Exit.FontSize = 10;
111 BTN_Exit.FontWeight = 'b';
112 BTN_Exit.BackgroundColor = [0.8 0.8 0.8];
113 BTN_Exit.Value = 0;
114
115 pos0 = 0.28;
116
117 IN_Port1 = uicontrol(inpt,'Style','edit','String','49010',...
118     'Units','normalized','Position',[0.55 pos0 0.4 0.05],'Visible','on',...
119     'Enable','on');
120 IN_Port1.FontName = 'Consolas';
121 IN_Port1.FontSize = 10;
122 IN_Port1.HorizontalAlignment = 'left';
123 lbl_Port1 = uicontrol(inpt,'Style','text','String','MATLAB Port:',...
124     'Units','normalized','Position',[0.02 pos0-0.008 0.4 0.05],...
125     'FontName','Verdana','FontSize',9,'HorizontalAlignment','left');
126
127 IN_Port2 = uicontrol(inpt,'Style','edit','String','49000',...
128     'Units','normalized','Position',[0.55 pos0+0.06 0.4 0.05],'Visible',...
129     'on','Enable','on');
130 IN_Port2.FontName = 'Consolas';
131 IN_Port2.FontSize = 10;
132 IN_Port2.HorizontalAlignment = 'left';
133 lbl_Port2 = uicontrol(inpt,'Style','text','String','X-PLANE 11 Port:',...
134     'Units','normalized','Position',[0.02 pos0+0.06-0.008 0.5 0.05],...
135     'FontName','Verdana','FontSize',9,'HorizontalAlignment','left');
136
137 IN_tstep = uicontrol(inpt,'Style','edit','String','0.2',...
138     'Units','normalized','Position',[0.55 pos0+0.135 0.4 0.05],...
139     'Visible','on','Enable','on');
140 IN_tstep.FontName = 'Consolas';
141 IN_tstep.FontSize = 10;
142 IN_tstep.HorizontalAlignment = 'left';
143 lbl_t_step = uicontrol(inpt,'Style','text','String',...
144     'Simulation Time step:','Units','normalized','Position',...
145     [0.02 pos0+0.12-0.008 0.5 0.08],'FontName','Verdana','FontSize',...
146     9,'HorizontalAlignment','left');
147
148 IN_ELMU = uicontrol(inpt,'Style','edit','String','5',...
149     'Units','normalized','Position',[0.55 pos0+0.21 0.4 0.05],...
150     'Visible','on','Enable','on');
151 IN_ELMU.FontName = 'Consolas';
152 IN_ELMU.FontSize = 10;
153 IN_ELMU.HorizontalAlignment = 'left';
154 lbl_ELMU = uicontrol(inpt,'Style','text','String','User El. Mask:',...
155     'Units','normalized','Position',[0.02 pos0+0.21-0.008 0.5 0.05],...
156     'FontName','Verdana','FontSize',9,'HorizontalAlignment','left');
157
158 IN_ELMG = uicontrol(inpt,'Style','edit','String','5',...
159     'Units','normalized','Position',[0.55 pos0+0.27 0.4 0.05],...
```

```
    'Visible','on','Enable','on');
IN_ELMG.FontName = 'Consolas';
IN_ELMG.FontSize = 10;
IN_ELMG.HorizontalAlignment = 'left';
lbl_ELMG = uicontrol(inpt,'Style','text','String','GBAS El. Mask:',...
    'Units','normalized','Position',[0.02 pos0+0.27-0.008 0.5 0.05],...
    'FontName','Verdana','FontSize',9,'HorizontalAlignment','left');

IN_AIRP = uicontrol(inpt,'Style','popupmenu','String',...
    'LEAS|LEMD|LEVC','Units','normalized','Position',...
    [0.55 pos0+0.33 0.4 0.05],'Visible','on','Enable','on');
IN_AIRP.FontName = 'Consolas';
IN_AIRP.FontSize = 10;
IN_AIRP.HorizontalAlignment = 'left';
lbl_LEAS = uicontrol(inpt,'Style','text','String',...
    'Airport Selection:','Units','normalized','Position',...
    [0.02 pos0+0.33-0.008 0.5 0.05],'FontName','Verdana',...
    'FontSize',9,'HorizontalAlignment','left');

IN_GNSS = uicontrol(inpt,'Style','popupmenu','String',...
    'GPS|GLONASS','Units','normalized','Position',...
    [0.55 pos0+0.405 0.4 0.05],'Visible','on','Enable','on');
IN_GNSS.FontName = 'Consolas';
IN_GNSS.FontSize = 10;
IN_GNSS.HorizontalAlignment = 'left';
lbl_GNSS = uicontrol(inpt,'Style','text','String','GNSS Constellation:',...
    'Units','normalized','Position',[0.02 pos0+0.39-0.008 0.5 0.08],...
    'FontName','Verdana','FontSize',9,'HorizontalAlignment','left');

IN_HR = uicontrol(inpt,'Style','edit','String','15:00',...
    'Units','normalized','Position',[0.55 pos0+0.495 0.4 0.05],...
    'Visible','on','Enable','on');
IN_HR.FontName = 'Consolas';
IN_HR.FontSize = 10;
IN_HR.HorizontalAlignment = 'left';
lbl_HR = uicontrol(inpt,'Style','text','String',...
    'Sim. Hour start (HH:MM):','Units','normalized','Position',...
    [0.02 pos0+0.48-0.008 0.5 0.08],'FontName','Verdana','FontSize',9,...
    'HorizontalAlignment','left');

IN_DAY = uicontrol(inpt,'Style','edit','String','07/09/2022',...
    'Units','normalized','Position',[0.55 pos0+0.595 0.4 0.05],...
    'Visible','on','Enable','on');
IN_DAY.FontName = 'Consolas';
IN_DAY.FontSize = 10;
IN_DAY.HorizontalAlignment = 'left';
lbl_DAY = uicontrol(inpt,'Style','text','String',...
    'Sim. Day start (DD/MM/YYYY):','Units','normalized','Position',...
    [0.02 pos0+0.58-0.008 0.5 0.08],'FontName','Verdana','FontSize',...
    9,'HorizontalAlignment','left');

lbl_title = uicontrol(inpt,'Style','text','String','SIM PARAMETERS',...
    'Units','normalized','Position',[0.0 0.94 1 0.05],'FontName',...
    'Verdana','FontSize',12,'HorizontalAlignment','center',...
    'FontWeight','b');
%-------------------------------------------------------------------------%

%%                              MAIN loop                                  %%
%-------------------------------------------------------------------------%
while ~BTN_Exit.Value

```

```
221     if BTN_Reset.Value || ~initialization_flag
222         if initialization_flag
223             delete(XNE_g);
224             delete(YNE_g);
225             delete(ZNE_g);
226             delete(HPL_g.Children(4));
227             delete(VPL_g.Children(4));
228             delete(HPL_g);
229             delete(VPL_g);
230             delete(cbv);
231             delete(cbh);
232             clear N1 N2 sh sv;
233         end
234         IN_DAY.Enable = 'on';
235         IN_HR.Enable = 'on';
236         IN_ELMG.Enable = 'on';
237         IN_ELMU.Enable = 'on';
238         IN_GNSS.Enable = 'on';
239         IN_AIRP.Enable = 'on';
240         IN_Port1.Enable = 'on';
241         IN_Port2.Enable = 'on';
242         IN_tstep.Enable = 'on';
243         BTN_Build.BackgroundColor = [1.0 1 0.5];
244         BTN_Build.Value = 0;
245         BTN_Build.Enable = 'on';
246         BTN_Start.BackgroundColor = [1.0 0.5 0.5];
247         BTN_Start.Value = 0;
248         BTN_Start.Enable = 'off';
249         BTN_Stop.BackgroundColor = [1.0 0.5 0.5];
250         BTN_Stop.Value = 0;
251         BTN_Stop.Enable = 'off';
252         BTN_Reset.Value = 0;
253         BTN_Exit.Value = 0;
254         XNE_g = axes(fig,'Position',[0.34 0.5 0.3 0.15]);
255         xne = plot(XNE_g,dummy,dummy,'r',dummy,dummy,'g',dummy,dummy,'b');
256         XNE_g.XLimMode = 'auto';
257         XNE_g.YLim     = [0 20];
258         XNE_g.XLim     = [0 1];
259         XNE_g.TickLabelInterpreter = 'latex';
260         XNE_g.BoxStyle = 'full';
261         xlabel('\textbf{t (s)}','Interpreter','latex');
262         ylabel('\textbf{X (m)}','Interpreter','latex');
263
264         YNE_g = axes(fig,'Position',[0.34 0.3 0.3 0.15]);
265         yne = plot(YNE_g,dummy,dummy,'r',dummy,dummy,'g',dummy,dummy,'b');
266         YNE_g.XLimMode = 'auto';
267         YNE_g.YLim     = [0 20];
268         YNE_g.XLim     = [0 1];
269         YNE_g.TickLabelInterpreter = 'latex';
270         YNE_g.BoxStyle = 'full';
271         xlabel('\textbf{t (s)}','Interpreter','latex');
272         ylabel('\textbf{Y (m)}','Interpreter','latex');
273
274         ZNE_g = axes(fig,'Position',[0.34 0.1 0.3 0.15]);
275         zne = plot(ZNE_g,dummy,dummy,'r',dummy,dummy,'g',dummy,dummy,'b');
276         ZNE_g.XLimMode = 'auto';
277         ZNE_g.YLim     = [0 9];
278         ZNE_g.XLim     = [0 1];
279         ZNE_g.TickLabelInterpreter = 'latex';
280         ZNE_g.BoxStyle = 'full';
281         xlabel('\textbf{t (s)}','Interpreter','latex');
```

```
282         ylabel('\textbf{Z (m)}','Interpreter','latex');
283
284         HPL_g = axes(fig,'Position',[0.7 0.48 0.29 0.22*900/650]);
285         HPL_g.XLimMode = 'manual';
286         HPL_g.YLimMode = 'manual';
287         HPL_g.XLim     = [0 24];
288         HPL_g.YLim     = [0 24];
289         HPL_g.TickLabelInterpreter = 'latex';
290         HPL_g.BoxStyle = 'full';
291         xlabel('\textbf{HPE (m)}','Interpreter','latex');
292         ylabel('\textbf{HPL (m)}','Interpreter','latex');
293         xticks([0  4  8  12  16  20  24]);
294         yticks([0  4  8  12  16  20  24]);
295         xmax_h     = ceil(1.5*hal);
296         ymax_h     = ceil(1.5*hal);
297         xgridmax_h = xmax_h;
298         ygridmax_h = ymax_h;
299         MISLEADH.Vertices = [0 0; xmax_h 0; xmax_h ymax_h];
300         MISLEADH.Faces = [1 2 3];
301         MISLEADH.FaceColor = 'red';
302         MISLEADH.FaceAlpha = .5;
303         HPEA.Vertices = [hal 0; hal ymax_h; xmax_h ymax_h; xmax_h 0];
304         HPEA.Faces = [1 2 3 4];
305         HPEA.FaceColor = 'red';
306         HPEA.FaceAlpha = .5;
307         HPLA.Vertices = [0 hal; 0 ymax_h; xmax_h ymax_h; xmax_h hal];
308         HPLA.Faces = [1 2 3 4];
309         HPLA.FaceColor = 'yellow';
310         HPLA.FaceAlpha = .4;
311         h1 = patch(HPL_g,MISLEADH);
312         hold on
313         h2 = patch(HPL_g,HPEA);
314         h3 = patch(HPL_g,HPLA);
315         sh = scatter(HPL_g,[0 xmax_h],[0 ymax_h],sz,[0 0],'s','filled');
316         colormap('jet');
317         cbh = colorbar;
318         cbh.TickLabelInterpreter = 'latex';
319         cbh.Limits = [0 1];
320         hold off
321
322         VPL_g = axes(fig,'Position',[0.7 0.1 0.29 0.22*900/650]);
323         VPL_g.XLimMode = 'manual';
324         VPL_g.YLimMode = 'manual';
325         VPL_g.XLim     = [0 9];
326         VPL_g.YLim     = [0 9];
327         VPL_g.TickLabelInterpreter = 'latex';
328         VPL_g.BoxStyle = 'full';
329         xlabel('\textbf{VPE (m)}','Interpreter','latex');
330         ylabel('\textbf{VPL (m)}','Interpreter','latex');
331         xticks([0 1 2 3 4 5 6 7 8 9]);
332         yticks([0 1 2 3 4 5 6 7 8 9]);
333         xmax_v     = ceil(1.5*val);
334         ymax_v     = ceil(1.5*val);
335         xgridmax_v = xmax_v;
336         ygridmax_v = ymax_v;
337         MISLEADV.Vertices = [0 0; xmax_v 0; xmax_v ymax_v];
338         MISLEADV.Faces = [1 2 3];
339         MISLEADV.FaceColor = 'red';
340         MISLEADV.FaceAlpha = .5;
341         VPEA.Vertices = [val 0; val ymax_v; xmax_v ymax_v; xmax_v 0];
342         VPEA.Faces = [1 2 3 4];
```

```
343         VPEA.FaceColor = 'red';
344         VPEA.FaceAlpha = .5;
345         VPLA.Vertices = [0 val; 0 ymax_v; xmax_v ymax_v; xmax_v val];
346         VPLA.Faces = [1 2 3 4];
347         VPLA.FaceColor = 'yellow';
348         VPLA.FaceAlpha = .4;
349         v1 = patch(VPL_g,MISLEADV);
350         hold on
351         v2 = patch(VPL_g,VPEA);
352         v3 = patch(VPL_g,VPLA);
353         sv = scatter(VPL_g,[0 xmax_v],[0 ymax_v],sz,[0 0],'s','filled');
354         colormap('jet');
355         cbv = colorbar;
356         cbv.TickLabelInterpreter = 'latex';
357         cbv.Limits = [0 1];
358         hold off
359         initialization_flag = 1;
360         build_flag = 0;
361         start_flag = 0;
362         stop_flag  = 0;
363         i = 1;
364     elseif BTN_Start.Value && build_flag && ~start_flag
365         user.update_LOC;
366         gbas.update_location;
367
368         gnss.SAT_Integrate;
369         gbas.update_SAT(gnss);
370         user.update_SAT(gnss);
371
372         gbas.update_RHO_MEAS(gnss);
373         gbas.update_PRC;
374         user.update_PRC(gbas);
375         user.update_PL(gbas);
376
377         user.update_RHO_MEAS(gnss);
378         user.EST_LQM(gnss);
379         user_error(:,i) = user.Get_ERROR();
380         prc(:,i) = gbas.PRC * 1000;
381         protec_level(:,i) = gbas.PL * 1000;
382         user_pl(:,i) = user.Get_PL * 1000;
383         t(i)=tstep*(i-1);
384         if t(i)>= XNE_g.XLim(2)
385                 XNE_g.XLim = [0 t(i)+max(0.2*200,0.4*t(i))];
386                 YNE_g.XLim = [0 t(i)+max(0.2*200,0.4*t(i))];
387                 ZNE_g.XLim = [0 t(i)+max(0.2*200,0.4*t(i))];
388         end
389         if i>=t0
390             x_error(i-t0+1) = abs(user_error(1,i)*1000);
391             y_error(i-t0+1) = abs(user_error(2,i)*1000);
392             z_error(i-t0+1) = abs(user_error(3,i)*1000);
393             h_bound(i-t0+1) = user_pl(1,i);
394             v_bound(i-t0+1) = user_pl(2,i);
395             ha(i-t0+1)      = hal;
396             va(i-t0+1)      = val;
397
398             xne(1).XData(i-t0+1) = t(i-t0+1);
399             xne(2).XData(i-t0+1) = t(i-t0+1);
400             xne(3).XData(i-t0+1) = t(i-t0+1);
401             xne(1).YData(i-t0+1) = x_error(i-t0+1);
402             xne(2).YData(i-t0+1) = h_bound(i-t0+1);
403             xne(3).YData(i-t0+1) = ha(i-t0+1);
```

```
404
405             yne(1).XData(i-t0+1) = t(i-t0+1);
406             yne(2).XData(i-t0+1) = t(i-t0+1);
407             yne(3).XData(i-t0+1) = t(i-t0+1);
408             yne(1).YData(i-t0+1) = y_error(i-t0+1);
409             yne(2).YData(i-t0+1) = h_bound(i-t0+1);
410             yne(3).YData(i-t0+1) = ha(i-t0+1);
411
412             zne(1).XData(i-t0+1) = t(i-t0+1);
413             zne(2).XData(i-t0+1) = t(i-t0+1);
414             zne(3).XData(i-t0+1) = t(i-t0+1);
415             zne(1).YData(i-t0+1) = z_error(i-t0+1);
416             zne(2).YData(i-t0+1) = v_bound(i-t0+1);
417             zne(3).YData(i-t0+1) = va(i-t0+1);
418
419             if mod(i,10)==0
420                 cpxerr     = x_error;
421                 cphbound   = h_bound;
422                 cpxerr(end+1)   = 0; cpxerr(end+1)   = xmax_h;
423                 cphbound(end+1) = 0; cphbound(end+1) = ymax_h;
424                 N1 = hist3([cpxerr',cphbound'],[nbins,nbins]);
425
426                 cpzerr     = z_error;
427                 cpvbound   = v_bound;
428                 cpzerr(end+1)=0;cpzerr(end+1)=xmax_v;
429                 cpvbound(end+1)=0;cpvbound(end+1)=ymax_v;
430                 N2 = hist3([cpzerr',cpvbound'],[nbins,nbins]);
431                 index_h = 1;
432                 index_v = 1;
433                 for k = 1:nbins
434                     for j = 1:nbins
435                         if(N1(k,j)>0 && ~(k==1 && j ==1 && N1(1,1)<2)...
436                                 && ~(k==nbins && j ==nbins ...
437                                 && N1(nbins,nbins)<2))
438                             hpe_grid(index_h)=xgridmax_h/nbins*(1/2+(k-1));
439                             hpl_grid(index_h)=ygridmax_h/nbins*(1/2+(j-1));
440                             colorh(index_h)=N1(k,j);
441                             index_h = index_h + 1;
442                         end
443
444                         if(N2(k,j)>0 && ~(k==1 && j ==1 && N2(1,1)<2)...
445                                 && ~(k==nbins && j ==nbins...
446                                 && N2(nbins,nbins)<2))
447                             vpe_grid(index_v)=xgridmax_v/nbins*(1/2+(k-1));
448                             vpl_grid(index_v)=ygridmax_v/nbins*(1/2+(j-1));
449                             colorv(index_v)=N2(k,j);
450                             index_v = index_v + 1;
451                         end
452                     end
453                 end
454                 sh.XData = hpe_grid;
455                 sh.YData = hpl_grid;
456                 sh.CData = colorh;
457                 cbh.Limits(2) = max(colorh);
458
459                 sv.XData = vpe_grid;
460                 sv.YData = vpl_grid;
461                 sv.CData = colorv;
462                 cbv.Limits(2) = max(colorv);
463             end
464
```

```
465             drawnow
466         end
467         i = i + 1;
468         if i==2
469             BTN_Start.BackgroundColor = [0.5 1 0.5];
470             BTN_Start.Enable = 'off';
471             BTN_Stop.BackgroundColor = [1 1 0.5];
472             BTN_Stop.Enable = 'on';
473         end
474     elseif BTN_Build.Value
475 
476         IN_DAY.Enable = 'off';
477         IN_HR.Enable = 'off';
478         IN_ELMG.Enable = 'off';
479         IN_ELMU.Enable = 'off';
480         IN_GNSS.Enable = 'off';
481         IN_AIRP.Enable = 'off';
482         IN_Port1.Enable = 'off';
483         IN_Port2.Enable = 'off';
484         IN_tstep.Enable = 'off';
485 
486         sim_t0 = datevec(IN_DAY.String,'dd/mm/yyyy');
487         sim_hour = datevec(IN_HR.String,'HH:MM');
488         sim_t0(4:6) = sim_hour (4:6);
489         portmat = str2double(IN_Port1.String);
490         portxpl = str2double(IN_Port2.String);
491         elmu  = str2double(IN_ELMU.String);
492         elmg  = str2double(IN_ELMG.String);
493         tstep = str2double(IN_tstep.String);
494 
495         switch IN_GNSS.Value
496             case 1
497                 constellation = 'GPS';
498             case 2
499                 constellation = 'GLONASS';
500             otherwise
501                 constellation = 'GPS';
502         end
503 
504         switch IN_AIRP.Value
505             case 1 % LEAS
506                 AIRP.lat = 43+33/60+49/3600;
507                 AIRP.lon = -6-2/60-5/3600;
508                 AIRP.alt = 126;
509             case 2 % LEMD
510                 AIRP.lat = 40+28/60+20/3600;
511                 AIRP.lon = -3-33/60-39/3600;
512                 AIRP.alt = 609;
513             case 3 % LEVC
514                 AIRP.lat = 39+29/60+22/3600;
515                 AIRP.lon = -0-28/60-54/3600;
516                 AIRP.alt = 73;
517             otherwise
518                 AIRP.lat = 40+28/60+20/3600;
519                 AIRP.lon = -3-33/60-39/3600;
520                 AIRP.alt = 609;
521         end
522 
523         gnss = GNSS(constellation,0,sim_t0,tstep);
524         gbas = GBAS(AIRP,0,elmg,sim_t0,tstep);
525         user = USER(AIRP,0,elmu,sim_t0,tstep,[portxpl,portmat]);
```

```
526
527         BTN_Build.BackgroundColor = [0.5 1 0.5];
528         BTN_Build.Enable = 'off';
529         BTN_Build.Value = 0;
530         BTN_Start.BackgroundColor = [1 1 0.5];
531         BTN_Start.Enable = 'on';
532         build_flag = 1;
533
534     elseif BTN_Stop.Value && ~start_flag && ~stop_flag
535
536         start_flag = 1;
537         stop_flag = 1;
538         BTN_Start.Value = 0;
539         BTN_Stop.BackgroundColor = [0.5 1 0.5];
540         BTN_Stop.Enable = 'off';
541         BTN_Stop.Value = 0;
542
543     end
544     drawnow;
545 end
546 close all;
547 %--------------------------------------------------------------------------%
```

A.2. Declaración de clases

A.2.1. GNSS.m

```
1  classdef GNSS < handle
2  % La clase GNSS engloba a la constelacion GNSS de la simulacion,
3  % incluyendo la obtencion de sus datos en el momento inicial de la
4  % simulacion, asi como su posicion y su integracion.
5
6      properties
7          SAT;       % Struct de datos de los satelites de la constelacion
8          sat_count; % Numero de satelites de la constelacion en total
9          time;      % Tiempo actual de los satelites
10     end
11
12     properties (Access = private)
13         log_flag;     % Flag del estado de log
14         LOG;          % Struct con los datos a loggear y el estado de log
15         error_flag;   % Flag que indica error en la introduccion de datos
16         t_step;       % Paso temporal para realizar las integraciones
17     end
18
19 methods
20 function obj = GNSS(sat_const,log_state,t0_sim,t_step)
21 %GNSS_Init inicia la clase de tipo GNSS y calcula el vector de estados
22 %(posicion y velocidad en el sistema geocentrico-ecuatorial) de la
23 %constelacion GNSS seleccionada para el instante inicial de la simulacion
24 %extrayendo sus datos de NORAD Celestrack
25 %
26 %STRUCTURE: obj = GNSS_Init(sat_const,t0_sim,t_step)
27 %
28 %INPUT:
```

```
29 %sat_const: constelacion seleccionada. Puede tomar los valores 'GPS' o
30 %'GLONASS'.
31 %log_state: estado del log de datos
32 %t0_sim: vector de fecha con el instante inicial de la simulacion, el cual
33 %se deriva de X-Plane. t0_sim = datevec(ConvertSerialYearToDate(2022,53));
34 %t_step: paso de tiempo en la simulacion;
35 %
36 %OUTPUT:
37 %obj: instancia de la clase GNSS.
38
39     obj.error_flag = 0;
40     switch sat_const
41
42         case 'GPS'
43             url = "https://tinyurl.com/GPS-TLE";
44             data = webread(url);
45
46         case 'GLONASS'
47             url = "https://tinyurl.com/GLONASS-TLE";
48             data = webread(url);
49
50         otherwise
51             fprintf('\nIncorrect command\n\n');
52             obj.error_flag = 1;
53     end
54
55     if (obj.error_flag == 0)
56         GNSS_ST = struct;
57         obj.SAT = struct;
58         length_TLE = 24 + 71 + 71 + 2;
59         mu = 398600.5;
60         obj.sat_count = length(data)/length_TLE;
61         JD_0    = juliandate(t0_sim);
62         options = odeset('AbsTol',1e-5,'RelTol',1e-5);
63         index = 1;
64         for i = 1 : obj.sat_count
65
66             GNSS_ST(i).ID = data(index:index+24); % NAME
67             GNSS_ST(i).ID = GNSS_ST(i).ID(1:strfind(GNSS_ST(i).ID,')'));
68
69             index = index + 44;     % EPOCH (JULIAN DAYS)
70             date_vector = datevec(ConvertSerialYearToDate(...
71                 str2double(data(index:index+1)) + 2000,str2double(...
72                 data(index+2:index+14))-1));
73             GNSS_ST(i).JD_TLE = juliandate(date_vector);
74
75             index = index + 61;     % INCLINATION (DEG)
76             GNSS_ST(i).i = str2double(data(index:index+8));
77
78             index = index + 8;      % RAAN (DEG)
79             GNSS_ST(i).RAAN = str2double(data(index:index+8));
80
81             index = index + 10;     % ECCENTRICITY
82             GNSS_ST(i).e = str2double(data(index:index+7))*1e-7;
83
84             index = index + 8;      % ARGUMENT OF THE PERIGEE  (DEG)
85             GNSS_ST(i).w = str2double(data(index:index+7));
86
87             index = index + 9;      % MEAN ANOMALY  (DEG)
88             GNSS_ST(i).ma = str2double(data(index:index+7));
89
```

```
90              index = index + 9;       % MEAN MOTION (REV/DAY)
91              GNSS_ST(i).mm = str2double(data(index:index+10));
92
93              index = i*length_TLE+1; % PREPARE NEXT SATELLITE
94
95              % SEMI-MAJOR AXIS (KM)
96              GNSS_ST(i).a = (mu/(GNSS_ST(i).mm*2*pi/(24*3600))^2)^(1/3);
97
98              % ECCENTRIC ANOMALY (DEG)
99              err = 1e-10;
100             Ecc_an_0 = GNSS_ST(i).ma * pi/180;
101             convergence_flag = 1;
102             n_it = 0;
103
104             while(convergence_flag)
105                 Ecc_an =  GNSS_ST(i).ma * pi/180 + GNSS_ST(i).e ...
106                     * sin(Ecc_an_0);
107                 if ( abs(Ecc_an - Ecc_an_0) < err)
108                     convergence_flag = 0;
109                 end
110                 Ecc_an_0 = Ecc_an;
111                 n_it = n_it + 1;
112             end
113
114             GNSS_ST(i).ecc_an = Ecc_an * 180/pi;
115
116             % TRUE ANOMALY (DEG)
117             GNSS_ST(i).theta  = 2*atan(sqrt((1+GNSS_ST(i).e)/...
118                 (1-GNSS_ST(i).e))*tand(GNSS_ST(i).ecc_an/2)) * 180/pi;
119
120             % ANGULAR MOMENT
121             GNSS_ST(i).h = sqrt(mu*GNSS_ST(i).a*(1 - GNSS_ST(i).e^2));
122
123             % G.E. POSITION AND VELOCITY AT EPOCH (KM & KM/S)
124             r = (GNSS_ST(i).h^2/mu)/(1 + ...
125                 GNSS_ST(i).e*cosd(GNSS_ST(i).theta));
126
127             r_pf = [r*cosd(GNSS_ST(i).theta); r*sind(GNSS_ST(i).theta); 0];
128
129             v_pf = [ -mu/GNSS_ST(i).h*sind(GNSS_ST(i).theta);...
130                 mu/GNSS_ST(i).h*(GNSS_ST(i).e + cosd(GNSS_ST(i).theta)); 0];
131
132             GNSS_ST(i).XYZ = rotz(GNSS_ST(i).RAAN) * rotx(GNSS_ST(i).i)...
133                 * rotz(GNSS_ST(i).w) * r_pf;
134
135             GNSS_ST(i).vel = rotz(GNSS_ST(i).RAAN) * rotx(GNSS_ST(i).i)...
136                 * rotz(GNSS_ST(i).w) * v_pf;
137
138             GNSS_ST(i).offset = 1; % Offset del reloj del satelite
139
140             delta_JD = GNSS_ST(i).JD_TLE - JD_0;
141             delta_t  = abs(delta_JD)*24*3600;
142             t0 = 0;
143             tf = delta_t;
144             y0 = [GNSS_ST(i).XYZ; GNSS_ST(i).vel];
145             [t,y] = ode45('Relative_movement_eq',[t0,tf],y0,options);
146             obj.SAT(i).ID       = GNSS_ST(i).ID;
147             obj.SAT(i).XYZ(:,1) = y(end,1:3)';
148             obj.SAT(i).vel(:,1) = y(end,4:6)';
149             obj.SAT(i).offset   = GNSS_ST(i).offset;
150             obj.time            = t0_sim;
```

```
151             obj.t_step          = t_step;
152             obj.log_flag        = log_state;
153         end
154
155         if (obj.log_flag == 0)
156             obj.LOG = 0;
157         else
158             obj.LOG = struct;
159             obj.LOG = GNSS_ST;
160         end
161     end
162 end
163
164 function SAT_Integrate(obj)
165 %SAT_Integrate integra la posicion y velocidad de una constelacion de
166 %satelites orbitando la Tierra despues de un intervalo de tiempo.
167 %
168 %STRUCTURE: SAT_Integrate(obj)
169 %
170 %INPUT:
171 %obj: objeto de tipo GNSS con el vector de estados inicial de la
172 %constelacion y donde se anyade el vector de estados integrado.
173
174     options  = odeset('AbsTol',1e-10,'RelTol',1e-10);
175
176     for i = 1 : obj.sat_count
177         y0    = [obj.SAT(i).XYZ; obj.SAT(i).vel];
178         [t,y] = ode45('Relative_movement_eq',[0,obj.t_step],y0,options);
179         obj.SAT(i).XYZ = y(end,1:3)';
180         obj.SAT(i).vel = y(end,4:6)';
181     end
182
183     if (obj.log_flag ~= 0)
184         for i = 1 : obj.sat_count
185             obj.LOG(i).XYZ(:,end+1) = obj.SAT(i).XYZ;
186             obj.LOG(i).vel(:,end+1) = obj.SAT(i).vel;
187         end
188     end
189 end
190
191 function log_val = GetLOGflag(obj)
192 % Obtiene el flag de log del objeto.
193     log_val = obj.log_flag;
194 end
195
196 function log_struct = GetLOG(obj)
197 % Obtiene el struct de datos guardados. Sera 0 si no se guardan.
198     log_struct = obj.LOG;
199 end
200
201 function log_struct = GetERR(obj)
202 % Obtiene el codigo de error de la clase.
203     log_struct = obj.error_flag;
204 end
205
206 function t_step_val = GetSTEP(obj)
207 % Obtiene el paso de integracion de la simulacion
208     t_step_val = obj.t_step;
209 end
210 end
211 end
```

A.2.2. RECEIVER.m

```
classdef (Abstract) RECEIVER < handle
% La clase RECEIVER engloba el conjunto de operaciones y atributos
% basicos de los receptores que forman parte de la simulacion

    properties (Access = public)
        RHO_MEAS;         % Medidas de pseudodistancia
        SAT_XYZ;          % Medidas de posicion de los satelites
    end

    properties (Access = protected)
        LOC;              % Struct (lat,lon,alt) de cada receptor
        LOG;              % Struct con datos de simulacion
        LOC_XYZ;          % Vector en el sistema G.E. de cada receptor
        n_receivers;      % Numero de receptores de la instancia
        elevation_mask;   % Mascara de elevacion del receptor
        time;             % Tiempo actual de la simulacion
        hour;             % Hora actual en la simulacion
        t_step;           % Paso temporal de la simulacion
        P;                % Matriz P del filtro de Kalman extendido
        est_pos;          % Estimacion de la posicion del filtro
        est_vel;          % Estimacion de la ground speed del filtro
        est_state;        % Estado estimado del filtro de Kalman
    end

methods (Access = public)
    function loc_struct = GetLOC(obj)
%Obtiene el struct de localizacion del receptor (o receptores).
        loc_struct = obj.LOC;
    end

    function update_SAT(obj,obj_sat)
%Actualiza el almanaque de los receptores con las nuevas posiciones
%de los satelites
        obj.SAT_XYZ = obj_sat.GetXYZ;
        obj.hour = obj.hour + obj.t_step/3600;
        if obj.hour >= 24
            obj.hour = 0;
        end
    end

    function log_struct = GetLOG(obj)
%Obtiene el struct de datos guardados de simulacion.
        log_struct = obj.LOG;
    end

    function loc_struct = GetLOC_XYZ(obj)
%Obtiene el struct de localizacion del receptor en G.E (o receptores).
        loc_struct = obj.LOC_XYZ;
    end

    function n_rec = GetREC(obj)
% Obtiene el numero de receptores del receptor
        n_rec = obj.n_receivers;
    end

    function elev = GetEmask(obj)
% Obtiene la mascara de elevacion del receptor
```

```
58          elev = obj.elevation_mask;
59      end
60
61      function rho = GetRHO_MEAS(obj)
62  % Obtiene las medidas de pseudodistancia del receptor
63          rho = obj.RHO_MEAS;
64      end
65
66  end
67
68  methods (Abstract, Access = protected)
69      latlon2XYZ(obj); % Paso de LOC a LOC_XYZ
70  end
71
72  methods (Abstract, Access = public)
73      update_RHO_MEAS(obj,sat_obj);
74  end
75
76  end
```

A.2.3. GBAS.m

```
1   classdef GBAS < RECEIVER
2   % La clase GBAS engloba el conjunto de operaciones y atributos asociados
3   % al receptor GBAS de la simulacion.
4
5       properties (Access = private)
6           PRC;      % Correcciones de la medida de pseudodistancia
7           PL;       % Correcciones de integridad crudas
8       end
9
10  methods (Access = public)
11  function obj = GBAS(localization,log_flag,el_mask,t0_sim,paso)
12  %Constructor de la clase GBAS.
13      obj.LOC            = localization;
14      obj.time           = t0_sim;
15      obj.hour           = obj.time(4)+ obj.time(5)/60 + obj.time(6)/3600;
16      obj.n_receivers    = length(localization);
17      obj.LOC_XYZ        = obj.latlon2XYZ;
18      obj.elevation_mask = el_mask;
19      obj.t_step         = paso;
20      obj.LOG            = struct;
21      obj.P              = [900 0 0 0 0 0; 0 1 0 0 0 0; 0 0 900 0 0 0;...
22          0 0 0 1 0 0; 0 0 0 0 900 0; 0 0 0 0 0 1];
23      obj.est_pos        = obj.LOC_XYZ+randn*30;
24      obj.est_vel        = [1;1;1];
25      obj.est_state      = [obj.est_pos(1);obj.est_vel(1);...
26          obj.est_pos(2);obj.est_vel(2);obj.est_pos(3);obj.est_vel(3)];
27
28      if (log_flag)
29          obj.LOG.flag           = 1;
30          obj.LOG.LOC            = localization;
31          obj.LOG.n_receivers    = obj.n_receivers;
32          obj.LOG.LOC_XYZ(:,1)   = obj.LOC_XYZ;
33          obj.LOG.elevation_mask = obj.elevation_mask;
34          obj.LOG.RHO_MEAS_hand  = 1;
35      else
```

```
36          obj.LOG.flag = 0;
37      end
38  end
39
40  function update_location(obj)
41  %Actualiza la posicion del receptor GBAS para contar con la rotacion de
42  %la Tierra
43      w_earth = 360/(23.9345*3600);
44      obj.LOC_XYZ = rotz(w_earth*obj.t_step)*obj.LOC_XYZ;
45      if (obj.LOG.flag)
46          obj.LOG.LOC_XYZ(:,end+1) = obj.LOC_XYZ;
47      end
48  end
49
50  function update_RHO_MEAS(obj,sat_obj)
51  %Actualiza las medidas de pseudodistancia del receptor
52      MEASURE (obj,sat_obj);
53      if (obj.LOG.flag)
54          obj.LOG.RHO_MEAS(obj.LOG.RHO_MEAS_hand,:)  = obj.RHO_MEAS;
55          obj.LOG.RHO_MEAS_hand = obj.LOG.RHO_MEAS_hand + 1;
56      end
57  end
58
59  function update_PRC(obj)
60  %Actualiza las correcciones de los satelites en vista por la estacion
61      pl_v = [0;0;0];
62      nlos = 0;
63      obj.PRC = 0*obj.RHO_MEAS;
64      for i=1:obj.n_receivers
65          for j=1:length(obj.SAT_XYZ)
66              if obj.RHO_MEAS(i,j)~=0
67                  pseudo_real = norm(obj.SAT_XYZ(:,j) - obj.LOC_XYZ(:,i));
68                  pl_raw = obj.RHO_MEAS(i,j) - pseudo_real;
69                      obj.PRC(j) = pl_raw;
70                      pl_v = pl_v + abs(pl_raw.*((obj.SAT_XYZ(:,j) - ...
71                          obj.LOC_XYZ(:,i))/norm(obj.SAT_XYZ(:,j)...
72                          - obj.LOC_XYZ(:,i))));
73                      nlos = nlos + 1;
74              end
75          end
76          obj.PL = pl_v * 1/nlos;
77      end
78  end
79
80  function prc = GetPRC(obj)
81  % Obtiene las correcciones de pseudodistancia del receptor GBAS
82      prc = obj.PRC;
83  end
84
85  function pl = GetPL(obj)
86  % Obtiene los valores para calcular el PL
87      pl = obj.PL;
88  end
89
90  end
91
92  methods (Access = protected)
93  function pos_XYZ = latlon2XYZ(obj)
94  %Obtiene las coordenadas geocentrico-ecuatoriales de los receptores de la
95  %estacion GBAS a partir de su definicion y el tiempo de la simulacion
96      for i = 1 : obj.n_receivers
```

```
97          J2000 = juliandate(datetime('2000-01-01 12:00:00'));
98          RT = 6371;
99          t00 = obj.time;
100         t00(1,4:6) = 0;
101         J0 = juliandate(t00);  % Dia juliano de la medianoche sim day
102         UT = obj.time(4) + obj.time(5)/60 + obj.time(6)/3600; % Hora UT;
103
104         T0 = double((J0-J2000)/36525);
105         theta_G0 = mod(100.4606184 + 36000.77004*T0 + 0.000387933*T0^2 - ...
106             2.583*10^-8*T0^3,360); % Tiempo sidereo de Greenwich a las 0h
107         theta_G = theta_G0 + 360.98564724*(UT/24);
108         theta_local = mod(theta_G + obj.LOC(i).lon,360);
109         pos_XYZ(:,i) = (RT + obj.LOC(i).alt)*[sind(obj.LOC(i).lat)*...
110             cosd(theta_local); sind(obj.LOC(i).lat)*sind(theta_local);...
111             cosd(obj.LOC(i).lat)];
112     end
113 end
114 end
115 end
```

A.2.4. USER.m

```
1  classdef USER < RECEIVER
2  % La clase USER engloba el conjunto de operaciones y atributos asociados al
3  % receptor del usuario de la simulacion.
4
5      properties (Access = private)
6          port;          % Puerto de comunicacion con X-Plane
7          est_pos_XYZ;   % Posicion estimada por el algoritmo en G.E.
8          est_error;     % Error estimado por el algoritmo
9          real_error;    % Error real del sistema
10         est_p_lvl;     % Nivel de proteccion estimado por el algoritmo
11         error;         % Error real del filtro
12     end
13     properties (Access = protected)
14         PRC;           % Correcciones provenientes de la estacion GBAS
15         GBAS_PL;       % Nivel de proteccion proveniente del GBAS
16         PL;            % Nivel de proteccion final
17         resm;          % Residuo medio del filtro
18         res;           % Residuo del filtro
19         pl_temp;       % Valor temporal del PL
20         pl_user;       % Nivel de proteccion calculado por el usuario
21     end
22
23 methods (Access = public)
24
25 function obj = USER(localization,log_flag,el_mask,t0_sim,tstep,puerto)
26 %Constructor de la clase USER.
27     obj.LOC            = localization;
28     obj.time           = t0_sim;
29     obj.hour           = obj.time(4)+ obj.time(5)/60 + obj.time(6)/3600;
30     obj.LOC_XYZ        = latlon2XYZ(obj);
31     obj.elevation_mask = el_mask;
32     obj.n_receivers    = length(localization);
33     obj.port           = puerto;
34     obj.est_pos_XYZ    = obj.LOC_XYZ + 20e-3*rand;
35     obj.est_vel        = [1;1;1];
```

```
36      obj.resm            = 0;
37      obj.res             = [0;0;0];
38      obj.est_state       = [obj.est_pos_XYZ(1);obj.est_vel(1);...
39          obj.est_pos_XYZ(2);obj.est_vel(2);obj.est_pos_XYZ(3);...
40          obj.est_vel(3)];
41      obj.P               = [900 0 0 0 0 0; 0 1 0 0 0 0; 0 0 900 0 0 0;...
42          0 0 0 1 0 0; 0 0 0 0 900 0; 0 0 0 0 0 1];
43      obj.LOG             = struct;
44      obj.GBAS_PL         = [18;18;8]*1e-3;
45      obj.PL              = [18;8]*1e-3;
46      obj.pl_user         = [18;18;8]*1e-3;
47      ip='localhost';
48      dataref{1}='sim/flightmodel/position/latitude';  % Latitud
49      dataref{2}='sim/flightmodel/position/longitude'; % Longitud
50      dataref{3}='sim/flightmodel/position/elevation'; % Altitud
51      dref_requestXPLANE(ip,puerto(1),puerto(2),1/tstep,dataref);
52      obj.t_step = tstep;
53      if (log_flag)
54          obj.LOG.flag           = 1;
55          obj.LOG.LOC            = localization;
56          obj.LOG.n_receivers    = obj.n_receivers;
57          obj.LOG.LOC_XYZ(:,1)   = obj.LOC_XYZ;
58          obj.LOG.elevation_mask = obj.elevation_mask;
59      else
60          obj.LOG.flag = 0;
61      end
62  end
63
64  function EST_LQM(obj,obj_sat)
65  %Estima la posicion del usuario haciendo uso de un filtro de minimos
66  %cuadrados y ofrece una aproximacion del error que comete el filtro
67      index = 1;
68      c = physconst('LightSpeed')*1e-9;
69      bias_usu = 11.3; % us
70      bias_est = 7;    % us
71      for i = 1:obj_sat.sat_count
72          if (obj.RHO_MEAS(i)~=0)
73              rho_valid(index) = obj.RHO_MEAS(i) + c * bias_usu;
74              sat_pos(:,index) = obj.SAT_XYZ(:,i);
75              index = index + 1;
76          end
77      end
78      m = index - 1; % Numero de medidas de satelites visibles
79      for iter = 1:30
80          A = [0,0,0,0];
81          rho_res = [0;0];
82          for i = 1:m
83              A(i,1) = -(sat_pos(1,i) - obj.est_pos_XYZ(1))/(rho_valid(i)...
84                  - c*bias_est);
85              A(i,2) = -(sat_pos(2,i) - obj.est_pos_XYZ(2))/(rho_valid(i)...
86                  - c*bias_est);
87              A(i,3) = -(sat_pos(3,i) - obj.est_pos_XYZ(3))/(rho_valid(i)...
88                  - c*bias_est);
89              A(i,4) =  c;
90              rho_res(i,1) = rho_valid(i) - (norm(sat_pos(:,i) - ...
91                  obj.est_pos_XYZ) + c*bias_est);
92          end
93          medidas_res = (transpose(A)*A)\(transpose(A)*rho_res);
94          obj.est_pos_XYZ  = obj.est_pos_XYZ + medidas_res(1:3,1);
95          bias_est = bias_est + medidas_res(4,1);
96      end
```

```
97      obj.res = medidas_res(1:3,1);
98      obj.real_error = obj.LOC_XYZ - obj.est_pos_XYZ;
99
100     obj.pl_temp = [0;0;0];
101     nlos = 0;
102     for j=1:length(sat_pos)
103         if rho_valid(j)~=0
104             pseudo_est = norm(sat_pos(:,j) - obj.est_pos_XYZ) +c*bias_est;
105             pl_raw = rho_valid(j) - pseudo_est;
106             obj.pl_temp = obj.pl_temp + abs(pl_raw.*((sat_pos(:,j) - ...
107                 obj.est_pos_XYZ)/norm(sat_pos(:,j)...
108                 - obj.est_pos_XYZ)));
109             nlos = nlos + 1;
110         end
111     end
112     obj.pl_temp = obj.pl_temp * 1/nlos;
113     obj.pl_user(:,end+1) = obj.pl_temp;
114
115     w_len = 25;
116     perc = 95;
117     initial = 10;
118     if length(obj.PL) < initial
119         obj.PL(:,end+1)=[18;8]*1e-3;
120     elseif length(obj.PL) <= w_len
121         pl_x = 1.5*max(prctile(obj.pl_user(1,initial:end),perc) + ...
122             randn*1.75e-3,prctile(obj.GBAS_PL(1,initial:end),perc)*2);
123         pl_y = 1.5*max(prctile(obj.pl_user(2,initial:end),perc) + ...
124             randn*1.75e-3,prctile(obj.GBAS_PL(2,initial:end),perc)*2);
125         pl_z = 1.5*max(prctile(obj.pl_user(3,initial:end),perc)*1.3 + ...
126             randn*0.8e-3,prctile(obj.GBAS_PL(3,initial:end),perc)*2);
127         obj.PL(:,end+1) = [norm([pl_x pl_y]) sqrt(2)*pl_z];
128     else
129         pl_x = 1.5*max(prctile(obj.pl_user(1,(end-w_len):end),perc) + ...
130             randn*1.75e-3,prctile(obj.GBAS_PL(1,(end-w_len):end),perc)*2);
131         pl_y = 1.5*max(prctile(obj.pl_user(2,(end-w_len):end),perc) + ...
132             randn*1.75e-3,prctile(obj.GBAS_PL(2,(end-w_len):end),perc)*2);
133         pl_z = 1.5*max(prctile(obj.pl_user(3,(end-w_len):end),perc)*1.3+...
134             randn*0.8e-3,prctile(obj.GBAS_PL(3,(end-w_len):end),perc)*2);
135         obj.PL(:,end+1) = [norm([pl_x pl_y]) sqrt(2)*pl_z];
136     end
137 end
138
139 function ESTIMATE(obj,obj_sat)
140 %Estima la posicion del usuario haciendo uso de un filtro de Kalman
141 %extendido y ofrece una aproximacion del error que comete el filtro
142     index = 1;
143     for i = 1:obj_sat.sat_count
144         if (obj.RHO_MEAS(i)~=0)
145             rho_valid(index) = obj.RHO_MEAS(i);
146             sat_pos(:,index) = obj.SAT_XYZ(:,i);
147             index = index + 1;
148         end
149     end
150     m = index - 1; % Numero de medidas de satelites visibles
151     n = 6;         % Numero de estados del sistema
152     R = eye(m)*5e-3;
153     PHI = eye(n);
154     PHI(1,2) = obj.t_step; PHI(3,4) = PHI(1,2); PHI(5,6) = PHI(1,2);
155     Q = zeros(n,n);
156     t5step = 1*obj.t_step;
157     Q(1,1) = t5step^3/3; Q(3,3) = Q(1,1); Q(5,5) = Q(1,1);
```

```
158     Q(1,2) = t5step^2/2; Q(2,1) = Q(1,2); Q(3,4) = Q(1,2);
159     Q(4,3) = Q(1,2); Q(5,6) = Q(1,2); Q(6,5) = Q(1,2);
160     Q(2,2) = t5step; Q(4,4) = Q(2,2); Q(6,6) = Q(1,1);
161     Q=0.01*Q;
162     proy_state = PHI*obj.est_state;
163     H = zeros(m,n);
164     for i = 1 : m
165         var = 1;
166         for j = 1 : 2 : n
167             H(i,j) = (proy_state(j)-sat_pos(var,i))/rho_valid(i);
168             var = var + 1;
169         end
170     end
171     M     = PHI*obj.P*PHI' + Q;
172     K     = M*H'*inv(H*M*H'+R);
173     obj.P = (eye(n)-K*H)*M;
174     est_meas = zeros(m,1);
175     for i = 1 : m
176         est_meas (i,1) = sqrt((proy_state(1)-sat_pos(1,i))^2+...
177             (proy_state(3)-sat_pos(2,i))^2+(proy_state(5)-sat_pos(3,i))^2);
178     end
179     obj.res = rho_valid' - est_meas;
180     obj.est_state = proy_state + K*obj.res;
181     obj.est_pos_XYZ=[obj.est_state(1); obj.est_state(3); obj.est_state(5)];
182     obj.real_error = obj.LOC_XYZ - obj.est_pos_XYZ;
183     obj.resm(end+1) = mean(obj.res)*1000;
184     w_len = 50;
185     perc = 95;
186     initial = 10;
187     if length(obj.resm) < initial
188         obj.est_error = 20;
189         obj.PL(:,end+1)=[18;8]*1e-3;
190     elseif length(obj.resm) <= w_len
191         obj.est_error = prctile(obj.resm(initial:end),perc);
192         pl_x = prctile(obj.GBAS_PL(1,initial:end),perc);
193         pl_y = prctile(obj.GBAS_PL(2,initial:end),perc);
194         pl_z = prctile(obj.GBAS_PL(3,initial:end),perc);
195         obj.PL(:,end+1) = [norm([pl_x pl_y]) pl_z];
196     else
197         obj.est_error = prctile(obj.resm(max(length(obj.resm)-...
198             w_len+1,initial):end),perc);
199         pl_x = prctile(obj.GBAS_PL(1,(end-w_len):end),perc);
200         pl_y = prctile(obj.GBAS_PL(2,(end-w_len):end),perc);
201         pl_z = prctile(obj.GBAS_PL(3,(end-w_len):end),perc);
202         obj.PL(:,end+1) = [norm([pl_x pl_y]) pl_z];
203     end
204
205 end
206
207 function update_LOC(obj)
208 %Recibe informacion de la posicion real del usuario de X-Plane y la
209 %convierte al formato correspondiente
210     datos = dref_readXPLANE();
211     obj.LOC.lat = datos(1);
212     obj.LOC.lon = datos(2);
213     obj.LOC.alt = datos(3)/3280.84; % De ft a km
214     obj.LOC_XYZ = latlon2XYZ(obj);
215 end
216
217 function pos = Get_POS_real(obj)
218 %Obtiene la posicion segun X-Plane en coordenadas G.E.
```

```
219     pos = obj.LOC;
220 end
221
222 function pos = Get_POS_real_XYZ(obj)
223 %Obtiene la posicion segun X-Plane en forma de struct lat,lon,alt
224     pos = obj.LOC_XYZ;
225 end
226
227 function [r_error] = Get_ERROR(obj)
228 %Obtiene el error real de la simulacion
229     r_error = obj.real_error;
230 end
231
232 function pos = Get_POS_est_XYZ(obj)
233 %Obtiene la posicion segun el algoritmo de estimacion en coordenadas G.E.
234     pos = obj.est_pos_XYZ;
235 end
236
237 function gls_data = Get_GLS(obj)
238 %Obtiene los datos relativos al GLS
239     gls_data = obj.est_pos_XYZ;
240 end
241
242 function update_RHO_MEAS(obj,sat_obj)
243 % Actualiza las medidas de pseudodistancia del receptor
244     MEASURE (obj,sat_obj);
245     prc_corr = 0*obj.RHO_MEAS;
246     for i= 1:length(obj.RHO_MEAS)
247         if obj.PRC(i)==0
248             obj.RHO_MEAS(i)=0;
249         elseif obj.RHO_MEAS(i)~=0
250             prc_corr(i)=obj.PRC(i);
251         end
252     end
253     obj.RHO_MEAS = obj.RHO_MEAS - prc_corr;
254 end
255
256 function update_PRC(obj,obj_gbas)
257     obj.PRC = obj_gbas.GetPRC;
258 end
259
260 function update_PL(obj,obj_gbas)
261     obj.GBAS_PL(:,end+1) = obj_gbas.GetPL;
262 end
263
264 function pl = Get_PL(obj)
265     pl = obj.PL(:,end);
266 end
267
268 end
269
270 methods (Access = protected)
271
272 function pos_XYZ = latlon2XYZ(obj)
273 %Obtiene las coordenadas geocentrico-ecuatoriales de los receptores de la
274 %estacion GBAS a partir de su definicion y el tiempo de la simulacion
275     J2000 = juliandate(datetime('2000-01-01 12:00:00'));
276     RT = 6371;
277     t00 = obj.time;
278     t00(1,4:6) = 0;
279     % Dia juliano de la medianoche del dia de la simulacion
```

```
280      J0 = juliandate(t00);
281      UT = obj.time(4) + obj.time(5)/60 + obj.time(6)/3600; % Hora UT;
282
283      T0 = double((J0-J2000)/36525);
284      % Grados, tiempo sidereo de Greenwich a las 0h
285      theta_G0 = mod(100.4606184 + 36000.77004*T0 + 0.000387933*T0^2 - ...
286          2.583*10^-8*T0^3,360);
287      theta_G = theta_G0 + 360.98564724*(UT/24);
288      % Distancia angular del meridiano local al punto vernal
289      theta_local = mod(theta_G + obj.LOC.lon,360);
290      pos_XYZ = (RT + obj.LOC.alt)*[sind(obj.LOC.lat)*cosd(theta_local);...
291          sind(obj.LOC.lat)*sind(theta_local); cosd(obj.LOC.lat)];
292  end
293
294  end
295  end
```

A.3. Funciones auxiliares

A.3.1. SAT_Visible.m

```
1  function visible_vector = SAT_Visible(obj,sat_obj)
2  %La funcion sat_visible determina la visibilidad del satelite de
3  %coordenadas satelite en base a una mascara de elevacion propia del
4  %receptor. El resultado es una matriz de forma 1 x nsat con unos y ceros
5  %dependiendo de si el satelite es visible o no
6
7      visible_vector = zeros(obj.GetREC,sat_obj.sat_count);
8      rec_xyz = obj.GetLOC_XYZ;
9      for i = 1 : obj.GetREC
10         for j = 1:sat_obj.sat_count
11             v_dif = sat_obj.SAT(j).XYZ - rec_xyz(:,i);
12             projection = transpose(v_dif) * rec_xyz(:,i);
13             if(projection > 0)
14                 if(abs(acosd(projection/(norm(v_dif)*norm...
15                         (rec_xyz(:,i))))) < 90 - obj.GetEmask)
16                     visible_vector(i,j) = 1;
17                 end
18             end
19         end
20     end
21 end
```

A.3.2. Relative_movement_eq.m

```
1  function [Ydot] = Relative_movement_eq(t,Y)
2
3      mu = 398600.5;
4
5      Yd(1) = Y(4); % x
6      Yd(2) = Y(5); % y
```

```
    Yd(3) = Y(6); % z

    r     = sqrt(Y(1)^2+Y(2)^2+Y(3)^2);
    RT    = 6378; % km
    J2    = 1.08263e-3;
    fj21  = (mu*Y(1))/r^3*(1.5*J2*(RT/r)^2*(1-5*(Y(3)/r)^2));
    fj22  = (mu*Y(2))/r^3*(1.5*J2*(RT/r)^2*(1-5*(Y(3)/r)^2));
    fj23  = (mu*Y(3))/r^3*(1.5*J2*(RT/r)^2*(3-5*(Y(3)/r)^2));

    Yd(4) = - mu * Y(1)/r^3 - fj21; % dx
    Yd(5) = - mu * Y(2)/r^3 - fj22; % dy
    Yd(6) = - mu * Y(3)/r^3 - fj23; % dz

    Ydot = [Yd(1) ; Yd(2) ; Yd(3) ; Yd(4) ; Yd(5) ; Yd(6)];

end
```

A.3.3. MEASURE.m

```
function MEASURE (obj,sat_obj)
%Determina las mediciones de pseudodistancia que es capaz de medir el
%receptor, tanto si esta formado por un conjunto de receptores como si
%solo cuenta con uno (caso usuario). Estas medidas incluyen un cierto
%ruido debido a los efectos del reloj, ionosfera, etc.

    sat_vis = SAT_Visible(obj,sat_obj);
    rec_xyz = obj.GetLOC_XYZ;
    for j = 1:sat_obj.sat_count
        prod = 1;
        for i = 1 : obj.GetREC
            prod = prod * sat_vis(i,j);
        end
        sat_vis(:,j) = prod;
    end
    obj.RHO_MEAS = zeros(obj.GetREC,sat_obj.sat_count);
    for i = 1:obj.GetREC
        for j = 1:sat_obj.sat_count
            if (sat_vis(i,j)==1) && ...
                    (atmos_check(obj,obj.SAT_XYZ(:,j))<=16e-3)
                obj.RHO_MEAS(i,j) = norm(obj.SAT_XYZ(:,j) - ...
                    rec_xyz(:,i)) + atmos_error(obj,obj.SAT_XYZ(:,j));
            end
        end
    end
end
```

A.3.4. ConvertSerialYearToDate.m

```
function [num] = ConvertSerialYearToDate(year,days)
  date0 = datenum(num2str(year),'yyyy');
  num = date0 + days;
end
```

A.3.5. atmos_error.m

```
function atm_error = atmos_error(obj,sat)
%La funcion atmos_error calcula el error que se le debe aplicar a una
%determinada LoS en funcion de la posicion del usuario y de la hora

    rec_xyz = obj.GetLOC_XYZ;
    for i = 1 : obj.GetREC
        v_dif = sat - rec_xyz(:,i);
        projection = transpose(v_dif) * rec_xyz(:,i);
        elev(i) = abs(acosd(projection/(norm(v_dif)*norm(rec_xyz(:,i)))));
        lat(i)  = obj.LOC(i).lat;
        lon(i)  = obj.LOC(i).lon;
        t = obj.hour;
        A = 10;
        tec(i) = 1.1*A + (A + cos(4*pi/24*t+pi/2 + lat(i)*pi/180))...
            *cos(2*pi/24*t + lon(i)*pi/180) + rand*1.5;
        atm_error(i) = 40.3e13/(1.57542e9)^2*tec(i)/abs(sind(2*elev(i)));
    end
end
```

A.3.6. atmos_check.m

```
function atm_error = atmos_check(obj,sat)
%La funcion atmos_error verifica el error combinado sin ruido para
%descartar satelites en vista con mucho error

    rec_xyz = obj.GetLOC_XYZ;
    for i = 1 : obj.GetREC
        v_dif = sat - rec_xyz(:,i);
        projection = transpose(v_dif) * rec_xyz(:,i);
        elev(i) = abs(acosd(projection/(norm(v_dif)*norm(rec_xyz(:,i)))));
        lat(i)  = obj.LOC(i).lat;
        lon(i)  = obj.LOC(i).lon;
        t = obj.hour;
        A = 10;
        tec(i) = 1.1*A + (A + cos(4*pi/24*t+pi/2 + lat(i)*pi/180))...
            *cos(2*pi/24*t + lon(i)*pi/180);
        atm_error(i) = 40.3e13/(1.57542e9)^2*tec(i)/abs(cosd(2*elev(i)));
    end
end
```

A.4. Complementos

A.4.1. ORBIT_representation.m

```
%% Representacion de las orbitas de los satelites en un periodo

% Toma de datos
```

```
a=GNSS('GPS',1,datevec(ConvertSerialYearToDate(2022,53)),100);
for i = 1:432
    a.SAT_Integrate
end
b = a.GetLOG;

% Representacion grafica animada

figure('Color','w');
r = 27e3 + 6.37e3;
% Representacion de la Tierra
general_axes = axes('Xlim', [-r r],'Ylim',[-r r],'Zlim',[-r r],...
    'Color','w','XColor','k','Ycolor','k','Zcolor','k');
[x,y,z] = ellipsoid(0,0,0,6370,6370,6370,30);
view(45,25);
hold on;
imagen = 'land_ocean_ice_2048.jpg';
textura = imread(imagen);
tierra = surface(x,y,-z,'FaceColor','none', 'EdgeColor',0.3*[1 1 1]);

set(tierra,'FaceColor','texturemap','CData',textura,...
    'FaceAlpha',0.9,'EdgeColor','none');
grid on;
axis equal;
hg = hgtransform('Parent',general_axes);
set(tierra,'Parent',hg);
%set(gcf,'Rendered','opengl');

% Representacion de los ejes
axis([-r r -r r -r r]);
plot3([0,2*6370],[0,0],[0,0],'r','LineWidth',2);  % X Vernal equinox
plot3([0,0],[0,2*6370],[0,0],'r','LineWidth',2);  % Y
plot3([0,0],[0,0],[0,2*6370],'r','LineWidth',2);  % Z
text(2*6370+120,10,0,texlabel('gamma'),'Color','r','FontSize',16);
text(10,2*6370+120,0,texlabel('Y'),'Color','r','FontSize',12);
text(0,0,2*6370+150,texlabel('Z'),'Color','r','FontSize',12);

% Representacion de las trazas de las orbitas por plano orbital
for i=[1 11 16 18 24 28]
    plot3(b(i).XYZ(1,2:end),b(i).XYZ(2,2:end),b(i).XYZ(3,2:end),...
        '-.k','LineWidth',0.5);
end
xlabel('X_G_E (km)')
ylabel('Y_G_E (km)')
zlabel('Z_G_E (km)')

% Representacion de las posiciones puntuales
estilo_texto = ['k' 'k' 'r' 'r' 'b' 'b' 'k' 'k' 'r' 'r' 'b' 'b'...
    'k' 'k' 'r' 'r' 'b' 'b' 'k' 'k' 'r' 'r' 'b' 'b' 'k' 'k' 'r' 'r' ...
    'b' 'b' 'k' 'k' 'r' 'r' 'b' 'b' 'k' 'k' 'r' 'r' 'b' 'b'];

for i=1:length(b(1).XYZ)-1
   nsat = 1;
   for k = 1:a.sat_count
            XS(i,nsat) = plot3([0 b(nsat).XYZ(1,i+1)],...
                [0 b(nsat).XYZ(2,i+1)],[0 b(nsat).XYZ(3,i+1)],'--m',...
                'LineWidth',1,'Marker','*');
            YS(i,nsat) = text(b(nsat).XYZ(1,i+1)+150,b(nsat).XYZ(2,i+1)...
                +150,b(nsat).XYZ(3,i+1)+150,texlabel(strcat('SAT',...
                num2str(nsat))),'Color',estilo_texto(k),'FontSize',11);
            nsat = nsat + 1;
```

```
65      end
66
67      drawnow
68
69      for j=1:a.sat_count
70          delete(XS(i,j));
71          delete(YS(i,j));
72      end
73  end
```