Trabajo Fin de Grado
Grado en Ingeniería Aeroespacial

Estudio y aplicación del Filtro de Kalman en fusión de sensores en UAVs

Autora: María Esther Aranda Romasanta
Tutora: Juana María Martínez Heredia

Dpto. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2017

Trabajo Fin de Grado
Grado en Ingeniería Aeroespacial

Estudio y aplicación del Filtro de Kalman en fusión de sensores en UAVs

Autora:
María Esther Aranda Romasanta

Tutora:
Juana María Martínez Heredia

Dpto. de Ingeniería Electrónica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2017

ÍNDICE

Índice de Figuras 3
1. Motivación 5
2. Introducción 6
3. Descripción general del hardware 8
3.1 GPS 8
3.2 IMU 9
3.3 Arduino 11
4. Adquisición de datos 13
4.1 Módulo GPS de la compañía Adafruit 13
4.2 Módulo IMU de la compañía Adafruit 14
4.3 Montaje del sistema de medición 14
4.4 Programa de obtención de datos 15
4.5 Dónde fueron realizadas las medidas 16
4.6 Incidencias en la toma de los registros 18
5. Tratamiento de los datos obtenidos 19
5.1 Tratamiento de la posición 19
5.1.1 Fórmula de Haversine 19
5.1.2 Rotación de las aceleraciones de la IMU 21
5.1.3 Integración de las aceleraciones de la IMU 23
5.2 Tratamiento de la velocidad 25
5.3 Fusión de datos 26
6. Filtro de Kalman lineal 27
6.1 Introducción 27
6.2 Filtro de Kalman monodimensional 28
6.3 Definición de varianza y covarianza 30
6.4 Filtro de Kalman multidimensional 31
7. Aplicación del filtro de Kalman lineal 35
7.1 Modelo dinámico lineal 35
7.2 Inicialización de las matrices 35
7.3 Cálculo de las matrices de covarianza Q y R 37
7.4 Análisis de los resultados obtenidos 38
7.4.1 Entrada de datos a 100Hz en tramo 1 39
7.4.2 Entrada de datos a 50 Hz en tramo 2 42
7.4.3 Inicialización con distintas matrices de covarianza iniciales 45
7.4.4 Efecto de los vectores de ruido W y Z 48
8. Filtro de Kalman Extendido 51
8.1 Introducción 51
8.2 Idoneidad y limitaciones 53
9. Aplicación del filtro de Kalman Extendido 54
9.1 Modelo dinámico 54
9.2 Modelo utilizado en la medida 55
9.3 Inicialización de las matrices 55

9.4 Análisis de los resultados obtenidos 57
9.4.1 Entrada de datos a 100Hz en tramo 1 57
9.4.2 Entrada de datos a 50 Hz en tramo 2 60
9.4.3 Inicialización con distintas matrices de covarianza iniciales 63
9.4.4 Efecto de los vectores de ruido W y Z 66
10. Comparativa de los resultados obtenidos 69
11. Otros filtros 73
11.1. La transformación Unscented. 73
11.2. Filtro de Kalman Unscented 76
11.3. El filtro de Kalman como ejemplo de filtro adaptativo. 77
11.4. Filtro Schmidt-Kalman. 78
12. Ecuaciones para modelos UAV 79
12.1. Descripción del UAV 79
12.2. Modelado del UAV 79
12.3. Formulación por Newton-Euler 82
13. Conclusiones y líneas futuras 86
Apéndice A: Programa Arduino 88
Apéndice B: Tratamiento de medidas 92
1 Latitud, longitud y altitud a coordenadas $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$. 92
2 Conversión de un vector en función para poder ser integrado 92
3 Rotación de las aceleraciones de la IMU 93
Apéndice C: Filtro de Kalman Lineal 94
Apéndice D: Filtro de Kalman Extendido 97
Apéndice E: Filtro de Kalman Matlab 100
Bibliografía 102

ÍNDICE DE FIGURAS

Figura 2.1: Dron utilizado para fotografía. 7
Figura 3.1: Constelación de satélites GPS 9
Figura 3.2: IMU en formato modular 10
Figura 3.3: Giróscopo MPU-6050, acelerómetro Arduino Adx1335 y magnetómetro HMC5883L 10
Figura 3.4: Placa Arduino Due 12
Figura 4.1: Imagen del sensor GPS utilizado 13
Figura 4.2: Imagen del sensor IMU utilizado 14
Figura 4.3: Diagrama de las conexiones entre la placa Arduino, la IMU y el receptor GPS. 14
Figura 4.4: Montaje del sistema para realizar la medición 15
Figura 4.5: Muestra de archivo de registro de datos 16
Figura 4.6: Tramo 1 de la prueba 17
Figura 4.7: Tramo 2 de la prueba 17
Figura 5.1: Triángulo esférico y curva ortodrómica 19
Figura 5.2: Esquema del ángulo y dirección de avance medido por el GPS 21
Figura 5.3: Ejes de la IMU sobre el vehículo del experimento 22
Figura 5.4: Vista de los ejes de la IMU desde arriba 22
Figura 5.5: Regla del trapecio 24
Figura 5.6: Regla de Simpson 24
Figura 5.7: Esquema del ángulo y velocidad medida por el GPS 25
Figura 6.1: Diagrama de la estructura de un filtro de Kalman 28
Figura 6.2: Esquema de un filtro de Kalman multidimensional 32
Figura 7.1: Posición en el eje x – filtro lineal – 100Hz 39
Figura 7.2: Posición en el eje y – filtro lineal – 100Hz 39
Figura 7.3: Posición en el eje z – filtro lineal – 100Hz 40
Figura 7.4: Velocidad en el eje x - filtro lineal - 100Hz 40
Figura 7.5: Velocidad en el eje y - filtro lineal - 100Hz 41
Figura 7.6: Velocidad en el eje z - filtro lineal - 100Hz 41
Figura 7.7: Posición en el eje x – filtro lineal – 50Hz 42
Figura 7.8: Posición en el eje y – filtro lineal – 50Hz 42
Figura 7.9: Posición en el eje z – filtro lineal – 50Hz 43
Figura 7.10: Velocidad en el eje x - filtro lineal - 50Hz 43
Figura 7.11: Velocidad en el eje y - filtro lineal - 50Hz 44
Figura 7.12: Velocidad en el eje z - filtro lineal - 50Hz 44
Figura 7.13: Posición en el eje x – $P2$– 50Hz 45
Figura 7.14: Posición en el eje x – $P3$– 50Hz 46
Figura 7.15: Posición en el eje x – $P4$– 50Hz 46
Figura 7.16: Posición en el eje x – $P5$– 50Hz 47
Figura 7.17: Posición en el eje x – $P6$– 50Hz 48
Figura 7.18: Posición en el eje x – Efecto de W – 50Hz 49
Figura 7.19: Posición en el eje x – Efecto de Z – 50Hz 49
Figura 7.20: Posición en el eje x – Efecto de W y Z – 50Hz 50
Figura 9.1: Posición en el eje x – filtro extendido – 100Hz 57

Figura 9.2: Posición en el eje y – filtro extendido – 100Hz 58
Figura 9.3: Posición en el eje z – filtro extendido – 100Hz 58
Figura 9.4: Velocidad en el eje x – filtro extendido – 100Hz 59
Figura 9.5: Velocidad en el eje y – filtro extendido – 100Hz 59
Figura 9.6: Velocidad en el eje z – filtro extendido – 100Hz 60
Figura 9.7: Posición en el eje x – filtro extendido – 50Hz 60
Figura 9.8: Posición en el eje y – filtro extendido – 50Hz 61
Figura 9.9: Posición en el eje z – filtro extendido – 50Hz 61
Figura 9.10: Velocidad en el eje x – filtro extendido – 50Hz 62
Figura 9.11: Velocidad en el eje y – filtro extendido – 50Hz 62
Figura 9.12: Velocidad en el eje z – filtro extendido – 50Hz 63
Figura 9.13: Posición en el eje x – $P2$– 50Hz 64
Figura 9.14: Posición en el eje x – $P3$– 50Hz 64
Figura 9.15: Posición en el eje x – $P4$– 50Hz 65
Figura 9.16: Posición en el eje x – $P5$– 50Hz 65
Figura 9.17: Posición en el eje x – $P6$– 50Hz 66
Figura 9.18: Posición en el eje x – Efecto de W – 50Hz 67
Figura 9.19: Posición en el eje x – Efecto de Z – 50Hz 67
Figura 9.20: Posición en el eje x – Efecto de W y Z – 50Hz 68
Figura 10.1: Comparativa de la posición en dirección x 69
Figura 10.2: Comparativa de la posición en dirección y 70
Figura 10.3: Comparativa de la posición en dirección z 70
Figura 10.4: Comparativa de la velocidad en dirección x 71
Figura 10.5: Comparativa de la velocidad en dirección y 71
Figura 10.6: Comparativa de la velocidad en dirección z 72
Figura 11.1: Puntos en el plano de la medida y en el transformado 75
Figura 11.2: Medias ("+" MC, "◊" LIN y "◊" UT) y contornos 1σ estimados para x, yT 76
Figura 11.3: Esquema completo del concepto general del filtrado de Kalman. 78
Figura 12.1: Modelo dinámico del UAV 79

1. MOTIVACIÓN

Un UAV es un vehículo aéreo no tripulado, es decir, una aeronave que vuela sin tripulación. Es capaz de mantener de manera autónoma un nivel de vuelo controlado y sostenido, y es propulsado por un motor de explosión, eléctrico, o de reacción.

Existen dos tipos principales, los controlados desde una ubicación remota y los que son capaces de realizar un vuelo autónomo, a partir de planes de vuelo pre-programados mediante automatización dinámica.

Sus primeros usos fueron militares, ya que posibilitan sobrevolar áreas de alto riesgo o de difícil acceso, además de que no requieren la actuación de pilotos en zonas de combate. Sin embargo, presentan una serie de desventajas o problemas que aún están sin resolver.

Los principales problemas técnicos que presentan derivan del uso de señales para enviar y recibir información, en el caso del control remoto. Las señales para navegación vía satélite pueden ser alteradas externamente, como podría ocurrir en el caso de una guerra, o verse afectados por condiciones atmosféricas adversas, o simplemente ser los causantes de ralentizar las comunicaciones entre el operador en tierra y la aeronave, provocando retrasos entre la emisión de instrucciones y su recepción.

Es por ello que aparece como una necesidad el dotar a la aeronave de la mayor autonomía posible, siempre que esto no viole principios éticos como puede ser la decisión de atacar en el caso de una guerra. Para que esto sea realizable, es necesario equiparla con sensores que le permitan captar toda la información disponible a su alrededor y, por tanto, esta información debe ser lo más fiable posible.

Mediante mejoras en los sistemas de medición y calibración, instalación de sensores que, con sistemas de medición diferentes, proporcionan la misma medida, y la aparición de herramientas que permiten estimar mejor los estados de la aeronave, están apareciendo cada vez más usos para estos sistemas, que ya no solo implican operaciones militares complejas, sino maniobras en zonas pobladas o bajo situación de catástrofe que requieren que la aeronave esté dotada de potentes sistemas de control, estabilizadores y datos correctos para actuar de manera adecuada en un entorno con condiciones complejas y cambiantes.

2. INTRODUCCIÓN

En los últimos años se ha producido un avance importante en cuanto al desarrollo de los sistemas que integran los vehículos aéreos no tripulados. Debido al gran rango de aplicaciones para los que este vehículo está capacitado, pronto dejó de ser de uso exclusivo en el ámbito militar y amplió su campo de actuación al ámbito civil.

Este tipo de aeronaves no tripuladas son utilizadas en tareas de búsqueda y rescate, exploración de edificios, terremotos, control de incendios, salvamento marítimo, fotografía o cartografía aérea, entre otras muchas. Su capacidad de acceder de manera rápida y fácil a zonas de terreno irregular o peligroso, además de ofrecer imágenes aéreas a vista de pájaro, han hecho de este dispositivo una herramienta práctica en este tipo de situaciones.

Para poder llevar a cabo diferentes misiones, la aeronave debe contar con estabilidad suficiente para realizar maniobras complicadas, así como disponer de sistemas de control avanzados que proporcionen precisión a la hora de realizar movimientos complejos. El desarrollo de estos sistemas es un proceso que envuelve multitud de variables dinámicas.

Los sensores utilizados para llevar a cabo estas labores de control presentan errores en los datos proporcionados, debido a la complejidad de modelar el movimiento del vehículo o a la gran cantidad de información que manejan, ya que muchos de ellos trabajan con más de una variable. Es por esta acumulación de errores, que reduce el campo de aplicabilidad de estas aeronaves, que se decide estudiar la utilización una herramienta matemática que los minimice.

El filtro de Kalman es un estimador matemático que utiliza un proceso iterativo para eliminar el error cuadrático medio de un conjunto de datos. Su funcionamiento se basa en realizar, en base a un modelo dinámico del sistema, una predicción de un conjunto de variables de estado. Por otro lado, recibe por parte de los sensores los valores medidos de estas mismas variables. El filtro pondera, mediante un proceso estadístico, cuál de las dos fuentes, si la predicción o la medida, contiene mayor error, y estima un nuevo valor para las variables libre de ruido aleatorio. A pesar de tratarse de un proceso estadístico, constituye hoy en día una de las herramientas más óptimas para la estimación de valores.

Conociendo las características y ventajas que supone aplicar un filtro Kalman a las medidas obtenidas de los sensores con el fin de eliminar el ruido que estas contienen y, con ello, conseguir una mayor precisión en el control de la aeronave, se decide hacer un estudio basado en la implementación de un filtro Kalman a un vehículo en movimiento. Por simplicidad, el vehículo se mueve sobre una carretera realizando un movimiento rectilíneo y uniforme, con el fin de utilizar el filtro para conocer de manera precisa la trayectoria seguida y la velocidad de desplazamiento.

Los sensores utilizados para obtener estos datos de posición y velocidad son un GPS (Global Positioning System), que consiste en un sistema de navegación de gran precisión con capacidad de proporcionar la posición y velocidad de un objeto en coordenadas geográficas, y una IMU (Inertial Measurement Unit), que es una unidad de medida inercial para conocer en todo momento la orientación de un móvil y la trayectoria seguida. El proceso de obtención de datos durante la prueba es realizado mediante una placa Arduino a dos frecuencias distintas.

Dado que los sensores proporcionan el mismo tipo de información acerca del movimiento, es necesario fusionar ambas señales para que entren al filtro como una única fuente de medida de posiciones y velocidades. Para conseguirlo, los datos proporcionados por ambos sensores deben estar expresados en el mismo sistema de coordenadas y en las mismas unidades.

Una vez los datos son tratados para servir de entrada al filtro, se llevan a cabo dos filtrados Kalman diferentes. El filtrado lineal es utilizado en aquellos casos en los que las ecuaciones que constituyen la predicción y la medida son lineales. Según como se ha definido el movimiento, es el filtro más indicado para realizar las estimaciones. El otro filtrado se conoce por Extendido y es válido cuando las ecuaciones de la predicción, de la medida, o ambas, son no lineales. Forzando una entrada de la medida al filtro en un sistema de coordenadas distinto al utilizado para la predicción, se consigue la no-linealidad en el experimento realizado. Finalmente, se realiza un estudio de las variaciones que conlleva sobre la solución la toma matrices de inicialización del filtro de diferentes características.

A la vista de los resultados obtenidos tras el filtrado, se comparan con los proporcionados por la función del programa Matlab que implementa un filtro de Kalman, para comprobar la validez del programa desarrollado. Se plantea también la utilización de otros filtros Kalman que constituyen una mejora a las deficiencias que presentan los dos filtrados realizados.

Para concluir, se plantean las ecuaciones dinámicas de un cuatri-rotor, es decir, de una aeronave que dispone de cuatro rotores para su movimiento. El objetivo es extraer una idea clara del complejo modelo dinámico que sería necesario implementar para realizar la predicción del filtro, en caso de que fuese utilizado para estimar datos obtenidos de sensores instalados en un sistema real.

Figura 2.1: Dron utilizado para fotografía.

3. DESCRIPCIÓN GENERAL DEL HARDWARE

En esta sección se describirán brevemente las características y funcionamiento de los sensores de posicionamiento utilizados en este proyecto, GPS e IMU [1], así como del microcontrolador Arduino, con el objeto de dar una visión genérica sobre su funcionamiento y sus utilidades, ya que luego serán los dispositivos utilizados para la obtención de datos en el estudio llevado a cabo en el presente trabajo.

3.1 GPS

El sistema de posicionamiento global GPS (Global Positioning System), es un sistema de navegación de gran precisión (del orden de metros) y de operación continua, basado en el uso de satélites que proporcionan información sobre la posición en 3D de un objeto, persona o vehículo sobre la superficie terrestre. El sistema fue desarrollado, instalado y empleado por el Departamento de Defensa de los Estados Unidos, y está completamente operativo desde el año 1993.

Para determinar las posiciones en el globo, el sistema GPS se sirve de una red de veinticuatro satélites en órbita sobre el planeta Tierra, a 20200 km de altura, con trayectorias sincronizadas para cubrir toda la superficie de la Tierra (Figura 3.1). Utiliza la trilateración, un método matemático para determinar las posiciones relativas de objetos usando la geometría de triángulos de forma análoga a la triangulación.

Los satélites se encuentran dispuestos de tal manera que cualquier punto de la superficie es observado, como mínimo, por cuatro de esos satélites. Cuando se desea determinar la posición, estos satélites envían señales indicando la identificación del satélite y la hora del reloj de cada uno de ellos, que son recibidas por un receptor que se encuentre cercano a la superficie terrestre. Con base a estas señales, el aparato sincroniza el reloj del GPS y calcula el tiempo que tardan en llegar las señales al equipo, y de tal modo mide la distancia al satélite. Puesto que existen errores de sincronización en los relojes tanto de los satélites como de los receptores (a pesar de ser atómicos y tener gran precisión), a la distancia calculada por el tiempo de vuelo de la señal se le denomina pseudorango, para diferenciarlo del verdadero rango (corrigiendo esos errores de sincronización).

Mediante la trilateración se determina la posición del receptor. Cada satélite indica que el receptor se encuentra en un punto en la superficie de la esfera, con centro en el propio satélite y radio la distancia total hasta el receptor. Para calcular una posición en tres dimensiones bastaría con utilizar tres medidas diferentes, así se obtendrían las coordenadas x, y, z o latitud, longitud y altitud. Pero para resolver el problema de sincronización entre los relojes de los receptores GPS y los relojes de los satélites se necesita una medida adicional. De esta forma se obtendría un sistema de cuatro ecuaciones no lineales con cuatro incógnitas: latitud, longitud, altitud y el offset del reloj del receptor.

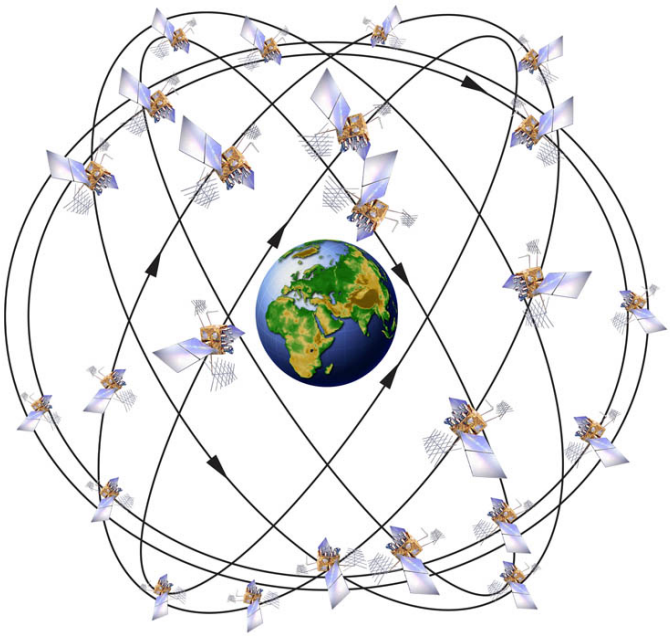

Figura 3.1: Constelación de satélites GPS

Al igual que en todos los sistemas sensoriales, el GPS no se libra de poseer fuentes de errores que son necesarias tener en cuenta para mitigar sus efectos. La precisión de la medida que se obtiene de la posición dependerá tanto de la precisión de los pseudorangos como de la geometría que forman los satélites que el receptor tiene a su alcance. La mejor posición es formando un tetraedro con el receptor en el centro de una de las caras, y el vértice opuesto a dicha cara sobre el usuario.

También se producen errores debido al tiempo de propagación de la señal. Puesto que estas viajan a la velocidad de la luz, porque son señales electromagnéticas, un error de 10ns podría resultar en un error en posición de hasta 3m.

3.2 IMU

Una unidad de medición inercial o IMU (del inglés, Inertial Measurement Unit), es un dispositivo electrónico utilizado para conocer en todo momento la orientación de un móvil y la trayectoria seguida, usando un método conocido como navegación por estima, así como para recibir información acerca de la velocidad y fuerzas gravitacionales del sistema. Las unidades de medición inercial son normalmente usadas para maniobrar aviones, incluyendo vehículos aéreos no tripulados, entre muchos otros usos, y además naves espaciales, incluyendo transbordadores, satélites y aterrizadores.

Estos sistemas son realmente útiles cuando la señal GPS no está disponible o tiene poca calidad, esto es, cuando el móvil se encuentra en zonas donde no existe contacto entre el emisor y el receptor GPS, como pueden ser túneles o dentro de edificios; o existen interferencias debidas a componentes electrónicos.

Como se puede observar en la Figura 3.2, son dispositivos con un tamaño reducido, gracias al uso de tecnología MEMS (del inglés, Microelectromechanical Systems). Además, son bastante versátiles y su costo es accesible.

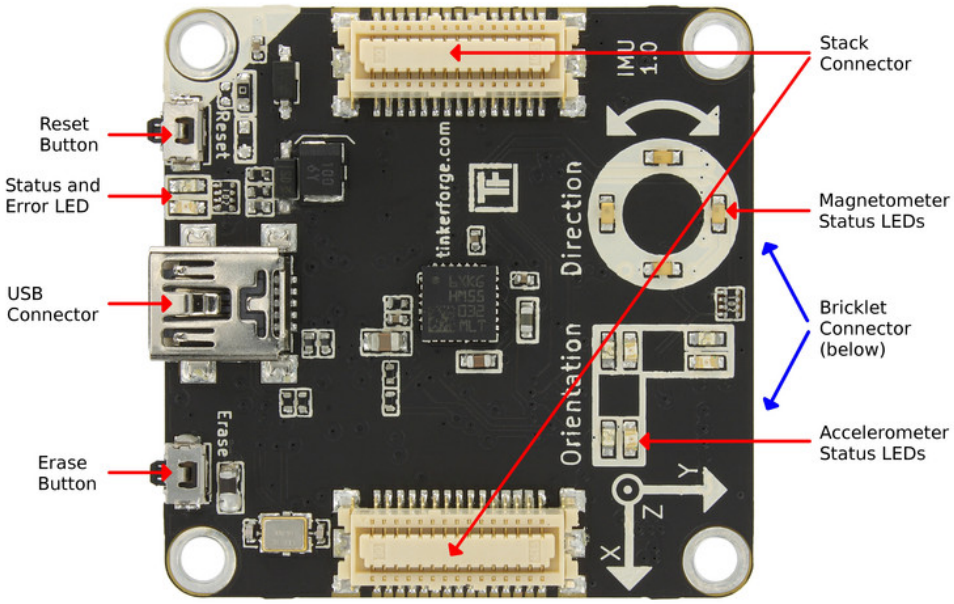

Figura 3.2: IMU en formato modular

Actualmente, los UAVs utilizados a cualquier nivel, tanto comercial como particular, suelen estar equipados con una IMU como mínimo (a veces junto a un receptor GPS), la cual incluye normalmente un giróscopo, un acelerómetro y un magnetómetro de 3 ejes cada uno. Estos sensores son susceptibles a poseer errores debido a varias causas: daños en los sistemas, variaciones de temperatura o vibración excesiva, entre otras.

Figura 3.3: Giróscopo MPU-6050, acelerómetro Arduino Adx1335 y magnetómetro HMC5883L

Una IMU funciona detectando la actual tasa de aceleración utilizando el acelerómetro, y los cambios en atributos rotacionales tales como cabeceo, alabeo y guiñada usando el giróscopo.

En un sistema de navegación, una vez recogidos los datos, son enviados a un ordenador, que calcula la posición actual basada en la velocidad, direcciones viajadas y tiempo.

La principal desventaja de usar una IMU para la navegación es que estas, normalmente, son afectadas por un error acumulativo. Este dispositivo calcula su nueva posición agregando los cambios detectados a las posiciones previamente calculadas. Por lo tanto, cualquier error que se produzca en la medida, por muy pequeño que sea, se va acumulando de punto a punto. Este fenómeno provoca que a medida que pasa más tiempo desde que se tomó la medida inicial, la posición real difiera más y más de la posición calculada.

La tasa de muestreo también es un factor que se encuentra limitado por los giróscopos y acelerómetros que integran la IMU. Esto es debido a que los dispositivos son capaces de tomar datos en un intervalo de tiempo finito. Por ejemplo, si el acelerómetro es capaz de tomar medidas una vez cada segundo, la IMU interpretará que esa aceleración se ha producido durante todo el segundo, incluso cuando puede que ocurran grandes variaciones en ese intervalo de tiempo. La resolución del sensor dependerá de la calidad de los componentes que la integren, esto es, de los acelerómetros y giróscopos.

Cabe destacar que un error constante en la aceleración se traduce en un error lineal en la velocidad y en un error cuadrático en la posición.

Las IMU normalmente son sólo uno de los componentes de un sistema de navegación. Otros sistemas son usados para corregir las imprecisiones que las IMU sufren invariablemente, tales como GPS, sensores de gravedad (para la vertical local), sensores de velocidad externos (para compensar la deriva por velocidad), un sistema barométrico para la corrección de la altitud y un compás magnético.

Un ejemplo de la navegación por estima en la que se basa el funcionamiento de este sensor sería el caso de una IMU instalada en un aeroplano, que informara de que el aparato viajó hacia el oeste durante una hora a una velocidad promedio de 804 kilómetros por hora. El dispositivo de guiado podría deducir que el avión debería estar a 804 kilómetros al oeste de su posición inicial. Si esta información estuviera combinada con un sistema informatizado de mapas (cuyo propósito sería similar al de la utilización del sistema de navegación GPS, aunque sin la necesidad de comunicarse con componentes externos como son los satélites), el dispositivo de guiado podría usar este método para mostrar al piloto donde está localizado geográficamente el avión.

3.3 Arduino

Arduino es una compañía de hardware libre, así como una comunidad tecnológica, que diseña y fabrica placas de desarrollo de hardware. Estas placas están compuestas por microcontroladores, elementos pasivos y activos. Son programadas a través de lo que se conoce como entorno de desarrollo (del inglés, Integrated Development Environment, IDE), el cual se encarga de compilar el código desarrollado para el modelo seleccionado de placa.

La idea bajo la que surgió Arduino fue la de acercar y facilitar el uso de la electrónica y la programación de sistemas a proyectos de diferentes disciplinas. Toda la plataforma Arduino libera sus componentes con licencia de código abierto, lo que permite libertad de acceso a ellos.

El hardware de Arduino está integrado por una placa de circuito impreso que incluye un microcontrolador, puertos digitales y analógicos de entrada y salida, cuatro de los cuales pueden ser conectados a placas de expansión, que permiten ampliar las características de funcionamiento de la placa Arduino. También posee un puerto de conexión USB que permite conectar la placa con el ordenador y alimentarla.

El software consiste en el entorno de desarrollo (IDE) previamente mencionado, que permite el desarrollo del código que programará la función para la que vaya a ser usada la placa. Este IDE está basado en un entorno de "Processing", que es un lenguaje de programación y entorno de desarrollo integrado, también de código abierto, basado en Java, de fácil utilización y que sirve como medio para la enseñanza y producción de proyectos multimedia e interactivos de diseño digital. El microcontrolador incluido en la placa se programa mediante un ordenador.

Arduino se puede utilizar para desarrollar objetos interactivos autónomos o puede ser conectado a algún software, tal como Adobe Flash. Una tendencia tecnológica es utilizar Arduino como tarjeta de adquisición de datos. Las placas pueden ser montadas a mano o adquiridas. El entorno de desarrollo integrado libre se puede descargar gratuitamente.

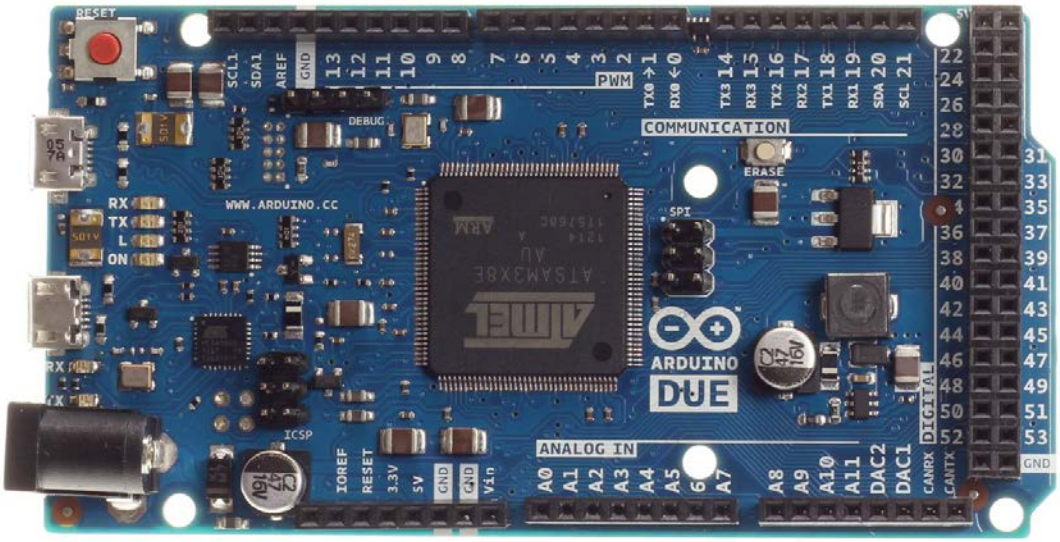

Figura 3.4: Placa Arduino Due

4. ADQUISICIÓN DE DATOS

El proceso de adquisición de datos se realizó a través de una placa Arduino Due conectada a una placa de desarrollo, sobre la que se encontraban insertados el conjunto de sensores, es decir, el GPS y la IMU. En este apartado se detallan las características concretas de los sensores utilizados, así como la estructura del montaje para el sistema de medición. Por otro lado, se comentan los cambios realizados sobre un programa, obtenido de la bibliografía, para obtener los datos a la frecuencia necesaria para el experimento. Por último, se detallará la estructura de los datos obtenidos y dónde fue realizado el experimento.

4.1 Módulo GPS de la compañía Adafruit

Se trata de un módulo de alta calidad que puede realizar seguimiento de hasta 22 satélites sobre 66 canales. Su frecuencia de actualización de datos va de 1 a 10Hz, aunque esta última es demasiado alta, ya que no tiene tiempo suficiente de tomar las medidas y enviarlas. Es por ello que en este trabajo la frecuencia del GPS será fijada a 1Hz. La precisión con la que toma las medidas será de menos de 3m para el caso de la posición, y del orden de 0.1m/s para el caso de la velocidad. El tiempo de arranque del sensor es de unos 34s y la velocidad máxima que es capaz de medir es 515m/s. El resto de características técnicas se detallan a continuación:

- **Tamaño de la antena:** 15mm x 15mm x 4mm.
- **Sensibilidad en adquisición:** -145dBm.
- **Sensibilidad en seguimiento:** -165dBm.
- **Rango de voltaje de entrada:** 3.0 – 5.5VDC.
- **Datos de salida:** NMEA 0183, 9600 baudios por defecto, 3V de salida a nivel lógico, 5V de entrada segura.
- **Soporte de DGPS/WAAS/EGNOS.**
- **Detección y reducción de interferencias.**
- **Detección y compensación de Multi-path.**

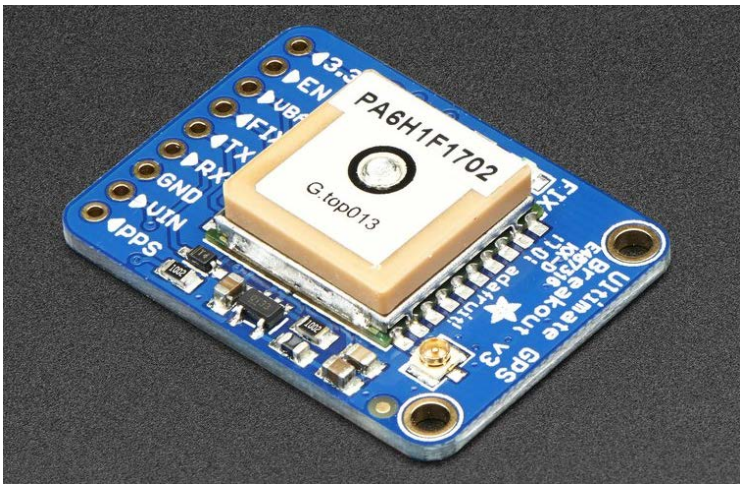

Figura 4.1: Imagen del sensor GPS utilizado

4.2 Módulo IMU de la compañía Adafruit

Esta unidad de medidas inerciales combina tres de los sensores de mayor calidad disponibles en el mercado para ofrecer once ejes de datos: aceleración en tres ejes, tres ejes giroscópicos y tres ejes magnéticos, además de medidas barométricas de presión y altitud, así como medida de temperatura. Las características técnicas de los tres sensores que conforman la IMU son:

- **Giróscopo L3GD20H de tres ejes:** ±250, ±500 o ±2000 grados por segundo de escala.
- **Brújula LSM303 de tres ejes:** de ±1.3 a ±8.1 gauss de escala de campo magnético.
- **Acelerómetro LSM303 de tres ejes:** ±2g, ±4g, ±8g o ±16g de escala seleccionables.
- **Presión barométrica/temperatura BMP180:** de 300 a 1100hPa de presión. De -40 a 85°C de temperatura. 0.17m de resolución.

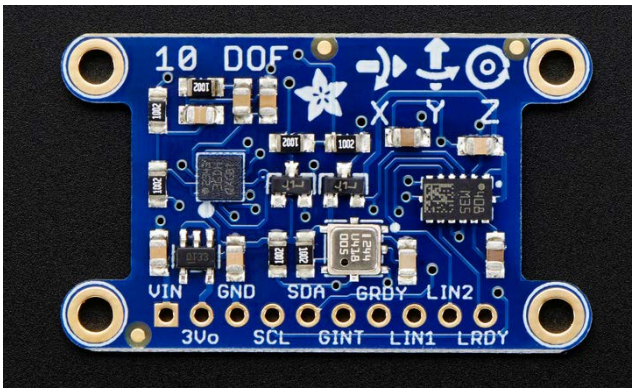

Figura 4.2: Imagen del sensor IMU utilizado

4.3 Montaje del sistema de medición

El montaje del hardware para la obtención de los datos durante el experimento ha sido realizado como aparece en el esquema de la Figura 4.3, según [2].

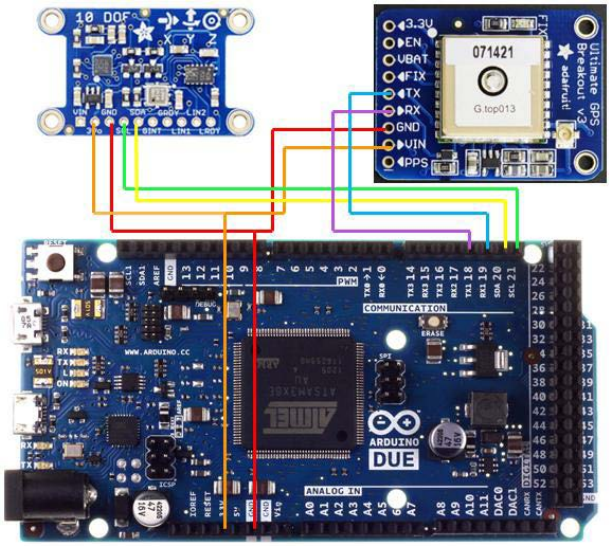

Figura 4.3: Diagrama de las conexiones entre la placa Arduino, la IMU y el receptor GPS.

Para que los ejes de la IMU estuvieran alineados con los ejes del movimiento del vehículo durante el experimento, se realizó un montaje sobre una caja de cartón indicando claramente los ejes del sensor (Figura 4.4). La antena del GPS (dispositivo negro situado fuera de la caja) fue pegada al techo del vehículo, facilitando así la capacidad de recibir señales por parte de los satélites, llegando a recibir en algunos tramos hasta diez señales.

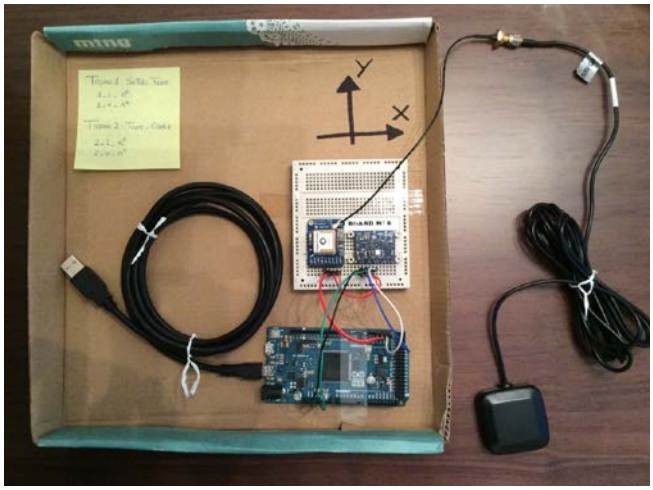

Figura 4.4: Montaje del sistema para realizar la medición

4.4 Programa de obtención de datos

El programa de captura de datos ejecutado sobre Arduino fue obtenido de [2] y aparece detallado en el Apéndice A. Consta de una función setup, donde se configuran las velocidades de la comunicación serie entre Arduino y el ordenador (115200 baudios), y la velocidad entre el canal serie y el módulo GPS (9600 baudios). A continuación, en la función, se configura el GPS para que transmita los mensajes NMEA RMC (recommended mínimum – mínimo recomendado) y GGA (fix data - constantes) mediante la llamada a la librería del módulo GPS sendCommand. Mediante este mismo comando se habilita el envío de los mensajes a 1Hz y se solicita la actualización del status de la antena. Finalmente, se llama a la función initSensors**,** desde donde se inicializan el acelerómetro, el sensor magnético y el medidor de presión atmosférica y temperatura.

Una vez inicializados los sensores, se inicia el bucle principal del programa, llamado loop (), el cual consiste básicamente en la programación de dos timers (temporizadores), que posibilitan la lectura a 1Hz de los datos procedentes del GPS, y a 100Hz de los datos procedentes de la IMU. Estos datos son empaquetados en una única línea de registro y transmitidos al terminal de Arduino. En esa línea se encuentran la última posición recibida del GPS y los datos recién leídos de la IMU. La línea contiene también los tiempos de recepción de los datos del GPS y de la IMU. La separación de los datos dentro de la línea se realiza mediante un carácter de tabulación. Al final de cada línea de registro se inserta un carácter retorno de carro, que hace que el terminal comience a escribir en la siguiente línea.

La estructura de cada línea de registro, numerados de uno a dieciséis y en orden de aparición de los elementos que la componen, es la siguiente:

#	Dato	Descripción	Unidades	Freq. (Hz)
1	TimerGPS	Instante de recepción de los datos del GPS	Milisegundos	1
2	Fix	Indicación de disponibilidad de posición	-	1
3	Quality	Tipo de posición 0 = No válido 1 = Posición GPS (SPS) 2 = Posición DGPS 3 = Posición PPS 4 = Navegación cinética satelital 5 = Navegación cinética satelital flotante 6 = Estima 7 = Modo manual 8 = Modo simulado	-	1
4	Satellites	Nº de satélites disponibles	-	1
5	Latitude	Latitud de la posición actual	Grados sexag.	1
6	Longitude	Longitud de la posición actual	Grados sexag.	1
7	Altitude	Altitud de la posición actual	Metros	1
8	Speed	Velocidad	Nudos	1
9	Angle	Ángulo del vector velocidad respecto al Norte	Grados sexag.	1
10	TimerIMU	Instante de recepción de los datos de la IMU	Milisegundos	100*
11	Heading	Rumbo	Grados	100*
12	Pitch	Ángulo de inclinación respecto al plano horizontal	Grados	100*
13	Roll	Ángulo de inclinación respecto al plano vertical	Grados	100*
14	Acel X	Aceleración en el eje X	Grados	100*
15	Acel Y	Aceleración en el eje Y	Grados	100*
16	Acel Z	Aceleración en el eje Z	Grados	100*

Tabla 4.1: Estructura del registro de datos

* Algunos registros fueron realizados también a 50Hz.

Una muestra de la estructura del registro de datos está representada en la Figura 4.5, donde aparecen las dieciséis columnas que posee cada línea.

```
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2293  206.37  -9.19  2.52  -1.57  0.43  9.69
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2303  206.37  -8.39  0.68  -1.45  0.12  9.85
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2313  206.39  -6.98  -3.48  -1.02  -0.51  8.32
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2323  206.39  -11.19  0.29  -1.49  0.04  7.53
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2333  206.39  -2.28  -1.52  -0.35  -0.24  8.87
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2343  206.39  -4.31  1.81  -0.75  0.31  9.89
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2353  206.39  -13.01  1.05  -1.92  0.16  8.32
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2363  206.39  -10.88  1.30  -1.96  0.24  10.20
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2373  206.39  -7.53  2.87  -1.33  0.51  10.08
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2383  205.88  -10.88  1.24  -1.37  0.16  7.14
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2393  205.88  -9.64  4.43  -1.53  0.71  8.98
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2403  205.88  -5.14  1.58  -1.02  0.31  11.34
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2413  205.88  -5.50  -7.20  -1.02  -1.33  10.51
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2423  205.88  -15.28  -3.20  -2.04  -0.43  7.45
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2433  205.88  -1.77  -1.77  -0.31  -0.31  10.16
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2443  205.88  -3.23  -7.74  -0.71  -1.69  12.40
2278  1  1  9  36.4601479  -6.2473216  4.30  38.19  99.31  2453  205.88  -20.15  2.74  -1.41  0.20  3.84
```

Figura 4.5: Muestra de archivo de registro de datos

4.5 Dónde fueron realizadas las medidas

Para el experimento realizado se estudia el movimiento rectilíneo y uniforme, esto es, a velocidad constante, de un vehículo circulando sobre una carretera sin pendiente ni tramos curvos. El experimento fue realizado de noche, evitando así el tráfico, sin viento ni lluvia, y utilizando el modo velocidad de crucero del que dispone el vehículo para conseguir una velocidad constante.

El registro de datos fue realizado recorriendo la autovía de une San Fernando con Cádiz (España), aprovechando los dos tramos rectos de los que esta carretera dispone (Figura 4.6 y 4.7). Los tramos donde se realizó la prueba se encuentran señalados en amarillo.

Figura 4.6: Tramo 1 de la prueba

Figura 4.7: Tramo 2 de la prueba

Por facilidad en la identificación, se han denominado tramo 1 y tramo 2. Es relevante indicar las orientaciones de los tramos con el fin de analizar los resultados obtenidos posteriormente.

El tramo 1 parte de San Fernando y finaliza en la curva que se dirige hacia Cádiz. En dirección a Cádiz, supone seguir un rumbo de 99º, y en dirección a San Fernando de 279º.

El tramo 2 parte de la curva y finaliza en Cádiz. En dirección a Cádiz tiene un rumbo de 155'5º, mientras que en dirección a San Fernando de 335'5º.

Observando la Figura 4.5 donde se muestran los datos de registro, las columnas nueve y once deberían mostrar los mismos valores, ya que el ángulo que mide el sensor GPS y el heading que mide el sensor IMU son los indicadores del rumbo. Sin embargo, no solo no coindicen, sino que difieren en un ángulo distinto de 180º. Investigando en la librería proporcionada por la compañía Adafruit sobre el módulo IMU, se ha concluido que el ángulo mostrado por el heading corresponde al arcotangente de las proyecciones de la medida tomada por el magnetómetro sobre los ejes x e y de la IMU. Al desconocer el valor de estas proyecciones, ha sido muy difícil determinar cuál es la referencia que toma el sensor para el cálculo del rumbo. Por ello, de ahora en adelante y durante los experimentos realizados en esta memoria, se tomará como valor del rumbo el ángulo proporcionado por el sensor GPS.

Es importante destacar que la aplicación de los filtros Kalman a los datos capturados durante el experimento aquí descrito fue realizada off-line, es decir, se realizó con posterioridad a la captura de los datos. No obstante, el algoritmo de los filtros Kalman que serán implementados está programado para que, únicamente añadiendo un procesador de datos a tiempo real, los filtros puedan ser implementados mientras se toman las medidas.

4.6 Incidencias en la toma de los registros

Es notable reseñar que la toma de registros se vio afectada por la capacidad limitada que el sistema proporcionaba en la captura de datos. Tras el inicio de un nuevo registro y transcurrido un tiempo variable entre 20 y 100 segundos, la captura de datos se interrumpía haciendo necesario un reset de la tarjeta Arduino (desconectando y volviendo a conectar el terminal USB a través del cual la tarjeta Arduino recibe la alimentación). No pudo determinarse si la interrupción del registro era motivada por la conexión serie entre el PC y la placa Arduino, debido al elevado volumen de los datos registrados (aproximadamente 100 líneas de registros por segundo), o por fallos en el acceso a los sensores de la IMU. En numerosas ocasiones, tras el reset de la tarjeta, al iniciar un nuevo registro, se producían fallos de acceso al módulo de la IMU, apareciendo la sentencia "Ooops, no LSM303 detected... Check your wiring!" a consecuencia de fallos en la inicialización del sensor.

Otro efecto que ralentizó y perturbó la toma de registros era la necesidad de esperar entre 30 segundos y un minuto tras la alimentación de la placa para que el GPS finalizara el proceso de adquisición de satélites y se obtuviera una posición GPS disponible.

5. TRATAMIENTO DE LOS DATOS OBTENIDOS

En esta sección se explicarán los cambios realizados sobre los datos obtenidos directamente de los sensores para adaptarlos a las unidades y sistemas de coordenadas en los que van a trabajar los filtros. Se comenzará hablando del tratamiento que han recibido los valores que determinan la posición, tanto de los obtenidos del GPS como de la IMU, para luego de qué manera se han obtenido los datos de velocidad. Finalmente se hablará de la fusión de datos, es decir, del método seguido para pasar a los filtros las medidas de posición y velocidad de los dos sensores al mismo tiempo.

5.1 Tratamiento de la posición

Incluye tres grupos de operaciones, correspondientes a los programas que aparecen en el Apéndice B de esta memoria. Primero, se describe la conversión de las coordenadas geográficas (Lat, Long, Alt) en la que se recibe el dato de posición GPS a coordenadas cartesianas. Seguidamente, se continuará describiendo la rotación de ejes e integración de los valores proporcionados por la IMU.

5.1.1 Fórmula de Haversine

La fórmula de Haversine calcula la distancia del círculo máximo entre dos puntos de una esfera, dadas la longitud y la latitud de dichos puntos. Esta fórmula es ampliamente utilizada en navegación, y representa un caso particular de la ley de Haversine utilizada en trigonometría esférica, que relaciona los lados y ángulos de los triángulos esféricos (Figura 5.1).

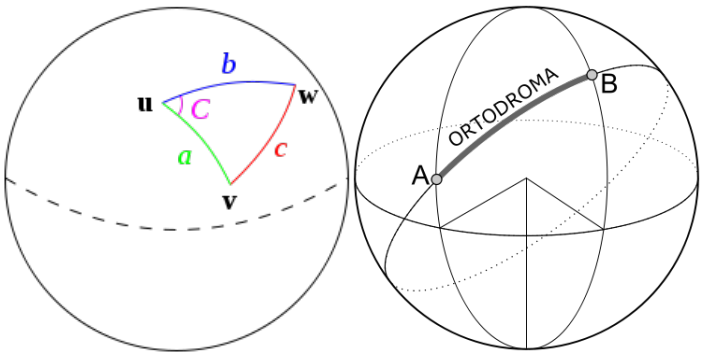

Figura 5.1: Triángulo esférico y curva ortodrómica

La distancia del círculo máximo entre dos puntos es la distancia más corta entre esos dos puntos. Aplicado a la superficie terrestre se hablaría de la ortodrómica (Figura 5.1). Una característica de esta curva es que presenta un ángulo diferente con cada meridiano, excepto cuando dicha ortodrómica coincide con un meridiano o con el ecuador.

Es necesario destacar que para la aplicación de esta fórmula se está suponiendo que la superficie de la Tierra es representada por una esfera, cosa que no es del todo cierto ya que, debido al achatamiento de los polos, posee la forma de un elipsoide. Sin embargo, dado que

esta aproximación esférica supone un error entorno al 0.3%, se considerará como válida en el caso bajo estudio.

Para cualquier pareja de puntos sobre una esfera, la función de Haversine para el ángulo central entre ellos, es decir, el ángulo que formarían esos dos puntos si estuviesen sobre una esfera del mismo radio, es:

$$hav\left(\frac{d}{r}\right) = hav(\varphi_2 - \varphi_1) + \cos\varphi_1 \cos\varphi_2 \, hav(\lambda_2 - \lambda_1)$$

Donde:

- **hav:** Función de Haversine: $haversin(\theta) = \sin(\frac{\theta}{2})^2 = \frac{(1-\cos\theta)}{2}$
- **d:** distancia entre los dos puntos sobre el círculo máximo de la esfera.
- **r:** radio de la esfera, que en este caso será el radio de la Tierra, $R = 6371Km$.
- $\boldsymbol{\varphi_1}, \boldsymbol{\varphi_2}$**:** Latitud de los puntos 1 y 2, respectivamente, en radianes.
- $\boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}$**:** Longitud de los puntos 1 y 2, respectivamente, en radianes.

Resolviendo la ecuación para d:

$$d = rhav^{-1}\left(hav\left(\frac{d}{r}\right)\right) = 2r\sin^{-1}\left(\sqrt{hav(\varphi_2 - \varphi_1) + \cos\varphi_1 \cos\varphi_2 \, hav(\lambda_2 - \lambda_1)}\right)$$

Una vez conocido el valor de la distancia que separa dos puntos sobre la superficie terrestre dadas sus latitudes y longitudes que proporcionará el sensor GPS, es preciso adaptar esta idea al problema del experimento que se realiza en esta memoria.

Tomando como latitud y longitud iniciales las del origen del movimiento, se calcularán los incrementos siempre respecto a ellas, de tal manera que en cada instante, el valor de la distancia (d) conocido sea el valor de la distancia total recorrida.

Los ejes en los que se estudiará el movimiento del vehículo serán unos ejes cartesianos con origen en el punto de inicio de la trayectoria del vehículo y el eje Y orientado según el Norte geográfico y el X según el Este. Por ello, será necesario descomponer esta distancia recorrida en las componentes según las direcciones Norte-Sur y Este-Oeste. Para ello, será utilizada otra de las medidas proporcionadas por el GPS, el llamado ángulo de GPS (track angle o bearing en inglés).

El bearing es el ángulo que va desde el Norte Geográfico a la dirección de avance del móvil. Como puede verse en la Figura 5.2, la distancia calculada anteriormente, que se corresponde con la distancia total recorrida por el móvil, tendrá la misma dirección que la dirección de avance en el caso de un movimiento rectilíneo.

En el caso de que el sensor GPS no proporcione el valor del ángulo, este puede ser calculado mediante relaciones trigonométricas que involucran los valores de las diferencias entre las latitudes y las longitudes.

$$bearing = \theta = \tan^{-1}(\sin\Delta\lambda * \cos\varphi_2 * \cos\varphi_1 * \sin\varphi_2 - \sin\varphi_1 * \cos\varphi_2 * \cos\Delta\lambda)$$

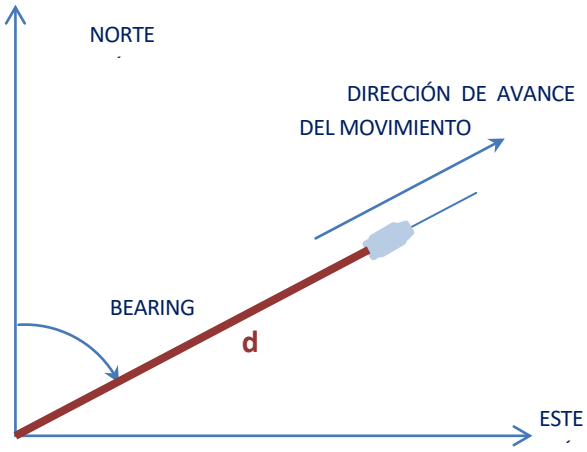

Figura 5.2: Esquema del ángulo y dirección de avance medido por el GPS

Una vez obtenida la distancia recorrida y el ángulo que forma la dirección de avance con la dirección del Norte Geográfico, la descomposición de la distancia recorrida según estos ejes es el resultado de aplicar relaciones trigonométricas. Las componentes de la posición del vehículo son:

$$x = E_G = d * \sin(\theta)$$

$$y = N_G = d * \cos(\theta)$$

En ningún momento en este desarrollo se ha hablado de la tercera componente que proporciona la posición del móvil, esto es, de la altura. El motivo es porque la altura proporcionada por el sensor GPS coincide con la altura en el sistema de ejes geográficos que se ha tomado como referencia, por lo que no es necesario efectuar ningún cambio sobre su valor.

$$z = Altura$$

5.1.2 Rotación de las aceleraciones de la IMU

El siguiente paso será la rotación de las aceleraciones proporcionadas por la IMU. Estas vienen expresadas en los ejes de la IMU, y para que las medidas puedan ser tratadas por los filtros Kalman a la vez que las de GPS, deben estar expresadas en los mismos ejes, es decir, en los ejes geográficos de la Tierra. Es preciso notar que los ejes de la IMU son ejes móviles respecto del sistema de referencia fijo que representan los ejes geográficos de la Tierra.

Los ejes de la IMU orientados según la dirección de avance del vehículo aparecen representados en la Figura 5.3. Por tratarse de un movimiento rectilíneo y uniforme, las aceleraciones únicamente tendrán componente en la dirección del eje y, aunque serán muy pequeñas, ya que al realizarse a velocidad constante el experimento serán aceleraciones

aleatorias provocadas por defectos de rodadura, como las que ocurrirán también en las direcciones x y z.

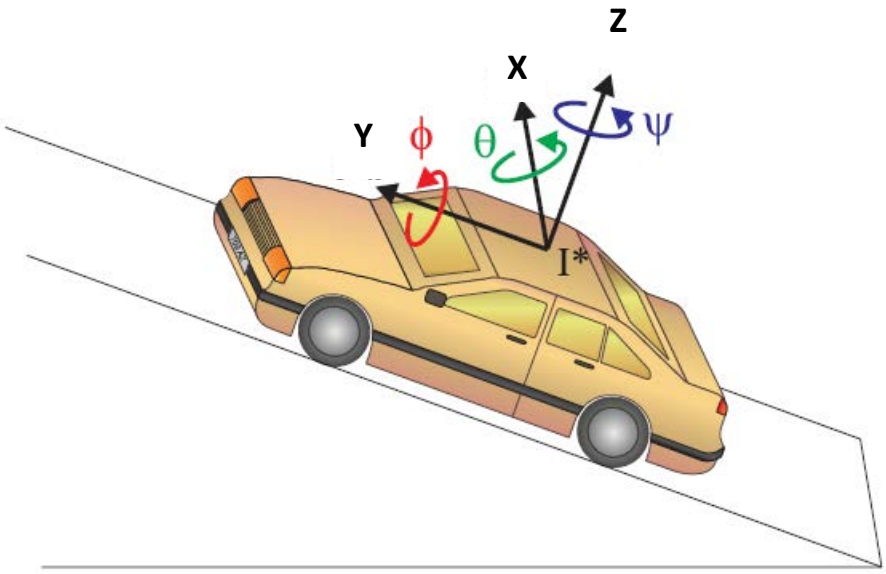

Figura 5.3: Ejes de la IMU sobre el vehículo del experimento

El ángulo que indica rotación respecto del eje x se denomina ángulo de cabeceo (θ, pitch angle), el ángulo que indica rotación en torno al eje y se denomina ángulo de balanceo (ϕ, roll angle) y el ángulo que indica rotación alrededor del eje z se denomina ángulo de guiñada (Ψ, yaw angle). La figura 5.4 muestra estos tres ángulos desde otra perspectiva:

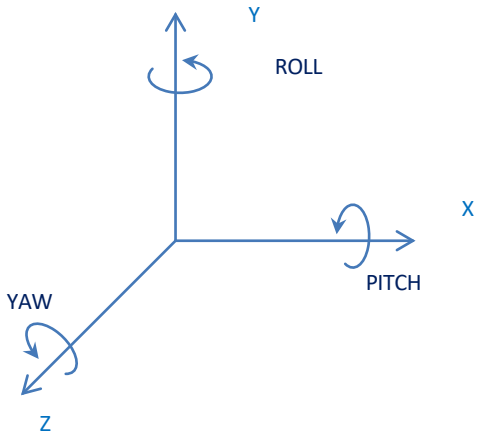

Figura 5.4: Vista de los ejes de la IMU desde arriba

La matriz de rotación que permite expresar los vectores de aceleración medidos en ejes móviles de la IMU a los ejes geográficos terrestres es, según [3]:

$$R_G = R_{\Psi}^{Z} * R_{\phi}^{Y} * R_{\theta}^{X}$$

$$R_G = \begin{pmatrix} \cos\Psi & -\sin\Psi & 0 \\ \sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}$$

$$a_G^{IMU} = R_G * a^{IMU}$$

En estas ecuaciones todos los ángulos deben estar expresados en radianes.

Una cuestión muy importante es que el ángulo de rotación en torno al eje z, o ángulo de guiñada, no es un dato proporcionado directamente por el sensor IMU. Este sensor devuelve un ángulo que recibe el nombre de *heading*, y que representa el ángulo que forma la dirección de avance del vehículo respecto del Norte Magnético terrestre. Por ello, para obtener el ángulo real de rotación del eje z respecto del Norte Geográfico es necesario aplicarle una corrección, que será sumar al *heading* la rotación de eje magnético de la Tierra respecto del eje geográfico en la zona en la que estén siendo tomadas las medidas. Según [4], en la zona geográfica dónde se ha realizado el experimento, detallada en el apartado 4.5, el ángulo que forma el eje magnético respecto del geográfico es:

$$Mag_{angle} = (1^\circ\ 25'W)$$

Por lo que el ángulo de guiñada corregido, y que debería ser introducido para llevar a cabo la rotación de matrices en las ecuaciones anteriores, es:

$$\Psi = heading + Mag_{angle}$$

Como se comentó en el apartado 4.5, en este trabajo se utilizará el ángulo proporcionado por el GPS para realizar esta rotación., teniendo la explicación anterior únicamente carácter ilustrativo en este trabajo.

5.1.3 Integración de las aceleraciones de la IMU

Una vez giradas las aceleraciones a ejes geográficos, es preciso integrar los valores obtenidos respecto del tiempo una vez para obtener los valores de la velocidad, y respecto del tiempo dos veces para obtener los valores de posición. Para obtener los valores de velocidad y aceleración, se aplicará un método de integración numérica, que, según [5], consiste en obtener el valor de la integral:

$$I = \int_a^b f(x)dx$$

Existen diversas reglas de integración. Se comentarán aquí las dos más conocidas y se realizará un estudio sobre cuál de las dos introduce más errores en el resultado.

- **Regla del trapecio:** La curva a integrar se aproxima con trapezoides, cada uno de los cuales posee un área proporcional a la multiplicación de la base por el promedio de la altura de los lados. Este método se muestra en la Figura 5.5.

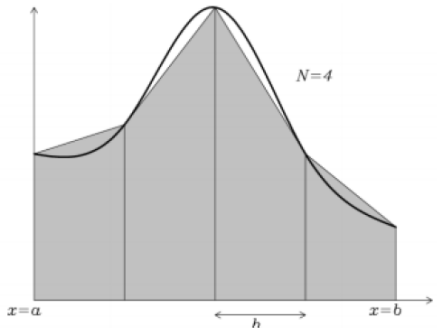

Figura 5.5: Regla del trapecio

$$I \approx \frac{h}{2}\big(f(a) + (b)\big) + h\sum_{j=0}^{N} f(a + jh); \qquad h = \frac{(b-a)}{N}$$

- **Regla de Simpson:** Aproxima la función a integrar en cada intervalo por una parábola, siendo el muestreo entre intervalos homogéneo (Figura 5.6). Se utiliza la función:

$$f(x) \approx \alpha x^2 + \beta x + \gamma$$

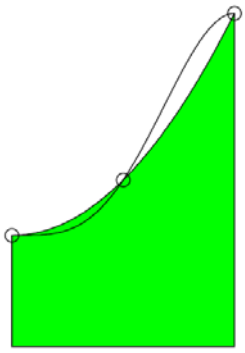

Figura 5.6: Regla de Simpson

Para el intervalo correspondiente a los puntos $i-1$, i e $i-1$, la función $f(x)$ se puede expresar como la suma ponderada de los valores de la función en esos tres puntos.

$$\int_{i-1}^{i+1} \alpha x^2 + \beta x + \gamma = \frac{f(i-1)}{3} + \frac{4f(i)}{3} + \frac{f(i+1)}{3}$$

En la siguiente tabla, obtenida de [6], se muestran los errores obtenidos mediante las aproximaciones de la regla de los trapecios y de Simpson:

Integración	Aproximación ε	Error relativo ϵ
Trapezoidal	$\varepsilon_t = \left(\frac{(b-a)^3}{N^2}\right) f^{(2)}$	$\epsilon_s = \frac{\varepsilon_t}{f}$
Simpson	$\varepsilon_s = \left(\frac{(b-a)^5}{N^4}\right) f^{(4)}$	$\epsilon_s = \frac{\varepsilon_s}{f}$

Tabla 5.1: Error en las reglas de integración

Para un menor número de secciones el error disminuye. Aunque el error depende del caso que se esté tratando, en general la regla de Simpson es una mejora sobre la regla trapezoidal, pese a la adicción de un leve coste computacional del algoritmo.

Es importante destacar que el número de puntos N tiene que ser impar para que la regla de Simpson funcione, lo que es un inconveniente si se tienen datos de campo o de laboratorio.

No obstante, considerando el número de muestras que se pasan a la función, se ha decidido trabajar en esta memoria con la regla de Simpson por las mejoras que supone su implementación. La función $quadv$ de Matlab implementa esta regla y además, permite trabajar con vectores.

5.2 Tratamiento de la velocidad

La velocidad, obtenida del sensor IMU tras la primera integración, está expresada en ejes geográficos ya que la rotación de ejes fue realizada previamente a la integración. Sin embargo, en el caso de la velocidad proporcionada por el GPS, es necesario tanto un cambio de sistema de referencia como un cambio de unidades.

El GPS proporciona la medida de la velocidad en nudos. La relación entre esta unidad y la establecida por el sistema internacional (m/s) es: $1\ knot = 0{,}514444\mathrm{m/s}$

Además, este valor de la velocidad se corresponde con el módulo de un vector en la dirección del movimiento del vehículo. Este módulo debe ser descompuesto según los ejes geográficos para tener la velocidad expresada en los mismos ejes que el resto de variables del problema. Esta velocidad es descompuesta de la misma manera que el valor de la distancia en el apartado 5.1.1. Esta descomposición se ve ilustrada en la Figura 5.7. Las componentes resultantes de la velocidad son:

$$V_x = V * \sin(\theta); \qquad V_y = V * \cos(\theta)$$

Donde V es el valor de la velocidad proporcionado por el GPS y θ el valor del ángulo proporcionado también por el GPS. En caso de no disponer de él, puede realizarse el cálculo del bearing previamente explicado en el apartado 5.1.1. La componente $V_z = 0$, ya que el movimiento se realiza en el plano horizontal.

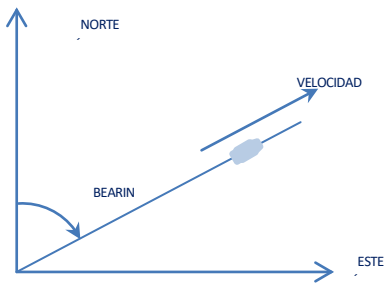

Figura 5.7: Esquema del ángulo y velocidad medida por el GPS

5.3 Fusión de datos

Una vez obtenidos y tratados todos los datos del GPS y la IMU, falta encontrar un método para fusionarlos y pasárselos al filtrado Kalman que se realizará posteriormente. Las operaciones realizadas sobre las medidas han dado lugar al conocimiento de tres posiciones y tres velocidades por cada sensor.

La fusión de datos desarrollada en este experimento se basa en el razonamiento desarrollado en [7], en el que las posiciones y velocidades que se pasan al filtro como medidas son una composición de las posiciones y velocidades el GPS con las de la IMU:

$$Pos = (x_{GPS} + x_{IMU},\ y_{GPS} + y_{IMU},\ z_{GPS} + z_{IMU})$$

$$Vel = \left(V_{x_{GPS}} + V_{x_{IMU}},\ V_{y_{GPS}} + V_{y_{IMU}},\ V_{z_{GPS}} + V_{z_{IMU}}\right)$$

El motivo es que por tratarse de un movimiento lineal, la nueva posición va a estar basada en la anterior. Como la frecuencia a la que se reciben los datos es distinta en ambos sensores, se utiliza uno de ellos, la IMU, para interpolar las posiciones y velocidades entre dos entradas de medida de GPS.

De un modo más concreto, cada segundo se recibirá una entrada de GPS, y cada 10 milisegundos se recibirá una posición de aceleración, que se corresponderá, tras lo visto en las reglas de integración numérica, con el incremento que ha sufrido la posición en el intervalo de tiempo de 10 milisegundos. Tomando como origen la posición indicada por el GPS, durante el segundo que pasa hasta que se recibe la nueva se suman los cien incrementos de posición que se obtienen de la integración de las aceleraciones. El razonamiento para la velocidad es análogo.

Sin embargo, y una vez vistos los resultados de una implementación práctica, se comprobó que al ser los valores de las aceleraciones cercanos a cero, el incremento de posición que se sumaba era prácticamente despreciable tras dos integraciones, en un diferencial de tiempo muy pequeño, de ese valor también pequeño. Por ello, cuando entraba una posición nueva de GPS transcurrido un segundo, el filtro pegaba grandes saltos.

La estrategia seguida para que no fuera la posición dada por el GPS la que controlara la medida fue la de linealizar el tramo entre dos medidas de GPS, ya que se disponía del dato de la velocidad en cada intervalo entre medidas de IMU. La fusión de sensores para la posición quedó finalmente:

$$Pos = \begin{pmatrix} x_{GPS} + x_{IMU} + Vel_x * (T_{IMU} - T_{GPS}) \\ y_{GPS} + y_{IMU} + Vel_y * (T_{IMU} - T_{GPS}) \\ z_{GPS} + z_{IMU} + Vel_z * (T_{IMU} - T_{GPS}) \end{pmatrix}$$

La fusión de la velocidad permanecerá como fue descrita anteriormente, ya que se supondrá prácticamente constante durante todo el experimento.

6. FILTRO DE KALMAN LINEAL

En esta sección se detallará qué es un filtro de Kalman, cuáles son las ecuaciones que rigen su funcionamiento y para qué se utiliza. Comenzando con una descripción del filtro para el caso monodimensional, para después seguir con una generalización del modelo al caso de trabajar con más de una variable, se explicará el significado y el comportamiento de cada uno de los parámetros que conforman esta herramienta matemática tan ampliamente utilizada en la actualidad. Se realizará también una pequeña introducción a los conceptos estadísticos de varianza y covarianza, básicos para el entendimiento del filtro multidimensional.

6.1 Introducción

Según [8], un filtro de Kalman es un estimador, esto es, una función estadística obtenida a partir de una muestra, para lo que se denomina problema lineal cuadrático. Este problema consiste en estimar el estado instantáneo de un sistema dinámico lineal perturbado por un ruido blanco, es decir, aleatorio, mediante el uso de medidas linealmente relacionadas con el estado, pero corrompidas por el ruido blanco. El estimador, o lo que es lo mismo, la función estadística resultante, es óptimo con respecto a cualquier función cuadrática de estimación de error.

El filtro de Kalman es uno de los descubrimientos más grandes en la historia de la teoría de la estimación estadística, aunque no es únicamente un estimador, ya que tiene la capacidad de propagar el estado actual de conocimiento del sistema dinámico, incluyendo la influencia estadística de las perturbaciones y los efectos de medidas anteriores. Se trata, por tanto, de una herramienta matemática y, aunque no es capaz de encontrar una solución por sí mismo al problema, facilita que el usuario la encuentre.

Sus aplicaciones más importantes han sido el control de sistemas dinámicos complejos, como los procesos de fabricación continua, aviones, barcos y vehículos espaciales, o el análisis de prestaciones de sistemas de estimación, determinando el mejor uso de estos sensores para un conjunto de criterios de diseño, como pueden ser la precisión o el coste de los sistemas. También es usado para predecir la posible evolución de sistemas dinámicos imposibles de controlar, como pueden ser el flujo de los ríos durante inundaciones, la trayectoria de cuerpos celestes o los precios de bienes comercializados.

De un modo más simple respecto a lo definido anteriormente, un filtro de Kalman no es más que un proceso matemático iterativo que, basado en un modelo del sistema dinámico bajo estudio, y de unas medidas realizadas mediante sensores de las variables que modelan este sistema, elimina el ruido blanco o aleatorio de ambas fuentes para dar una estimación del valor real de las variables de dicho sistema. Dicho de otra manera, minimiza el error cuadrático medio de un conjunto de datos para dar la estimación más óptima y fiable hasta ahora conseguida de la realidad.

Un ruido blanco es aquel que tiende a cero con la evolución del sistema, es decir, se trata de un ruido aleatorio que aparece como resultado de errores en las medidas o en la predicción realizada por el modelo dinámico de las variables del sistema, como puede ser el no haber

considerado una pequeña componente de viento en un movimiento de un vehículo a lo largo de una carretera. Se llama ruido o incertidumbre porque se trata de variaciones pequeñas sobre los resultados reales. Los errores en la calibración de los sensores o en el modelado del sistema dinámico que impliquen grandes variaciones sobre los resultados reales no están incluidos aquí, por lo que un filtro de Kalman no es capaz de eliminarlos.

Es importante, por tanto, remarcar la idea de que un filtro de Kalman no es capaz de decidir si la medida o el modelo son acertados o erróneos. Su única tarea es estimar, dentro de los datos de los que dispone, una solución óptima libre de ruido aleatorio.

6.2 Filtro de Kalman monodimensional

En la Figura 6.1, obtenida de [9], se puede ver un diagrama de las etapas que conforman un filtro de Kalman monodimensional:

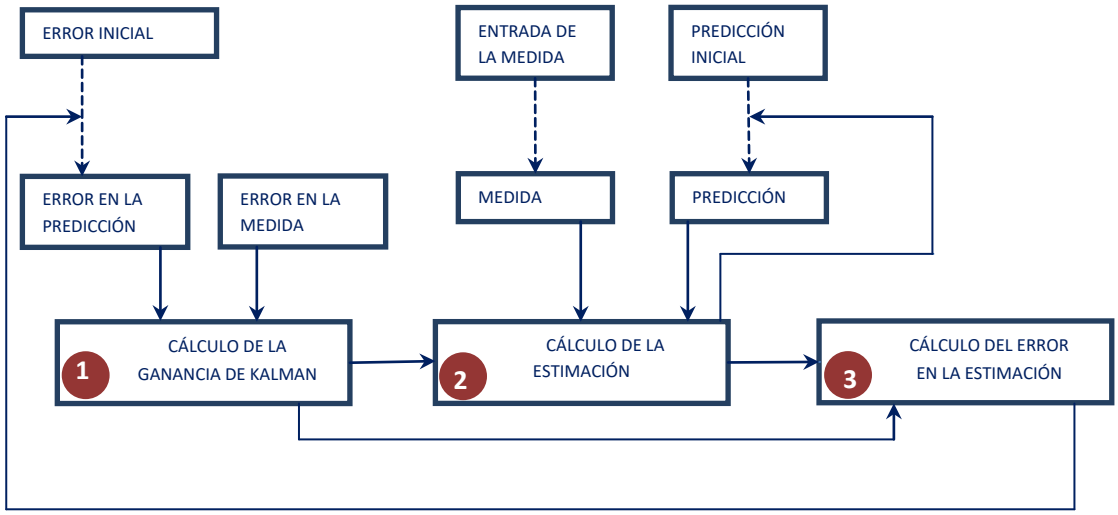

Figura 6.1: Diagrama de la estructura de un filtro de Kalman

En una iteración normal del filtro el primer paso es la predicción, para el instante actual, del valor de la variable de estado, utilizando el modelo dinámico del sistema y en base al valor estimado de la variable (salida del filtro) en la iteración anterior. A continuación, se actualiza el error de la predicción realizada utilizando el error que se obtiene de la salida del filtro como resultado de nuestra estimación.

En la primera iteración, al no disponerse de una salida del filtro, se utilizan valores iniciales previamente establecidos.

Por otro lado, el filtro recibe como dato de entrada la medida de la variable, obtenida del sensor, que será almacenada como valor medido. Se actualiza también el error de este último valor medido.

Siguen ahora los tres procesos principales en los que se basa el filtro.

El primero de ellos será calcular la Ganancia de Kalman que, utilizando los errores calculados, se encarga de ponderar entre la medida y la predicción, según se indica en el cuadro 1 de la Figura 6.1.

Con la Ganancia, la predicción de la variable de estado para esta iteración y la medida, se obtiene la salida del filtro, es decir, se estima el valor de la variable de estado. Finalmente, se actualiza el error cometido en el cálculo de esta estimación. Ambos procesos son indicados en los cuadros 2 y 3 de la Figura 6.1.

Tanto el valor de la estimación como de su error asociado son almacenados para ser utilizados en la siguiente iteración.

- **Ganancia de Kalman:** Es un peso cuyo valor oscila entre cero y uno. Su función dentro del filtro es la de dar más importancia al valor medido o al valor estimado mediante la comparación de los errores (*E*) de cada uno. Se define por el siguiente cociente:

$$\boldsymbol{K} = \frac{\boldsymbol{E_{ESTIMACIÓN}}}{\boldsymbol{E_{ESTIMACIÓN}} + \boldsymbol{E_{MEDIDA}}} \; ; \qquad \boldsymbol{K} \in [\boldsymbol{0}, \boldsymbol{1}]$$

  Al comienzo del proceso iterativo, el error en la estimación será normalmente elevado respecto al de la medida, porque tanto el error como la estimación se corresponderán con un valor que ha sido inicializado previamente por el programador y que estará alejado de la realidad, o por lo menos, más alejado que la medida. Por esta razón, al ser $E_{MEDIDA} \ll E_{ESTIMACIÓN}$, la Ganancia de Kalman tiende a un valor cercano a uno. Esto significará que el filtro dará más peso a la medida respecto de la estimación. A medida que avanza el proceso, la Ganancia de Kalman tiende a disminuir. En el extremo contrario, cuando su valor sea cercano a cero, significará que $E_{MEDIDA} \gg E_{ESTIMACIÓN}$, lo que indicará que la estimación prevalece sobre la medida denotando la buena estimación del modelo. La rápida convergencia del filtro de Kalman permite que el valor de la ganancia alcance un valor próximo a cero en un número relativamente pequeño de iteraciones. Por esta razón es tan ampliamente utilizado como uno de los mejores métodos estimativos.

- **Estimación del estado actual:** Una vez conocido cuál de los dos valores, si la estimación o la medida, es más fiable, y reflejado en el valor de la Ganancia de Kalman, *K*, tiene lugar la estimación del estado actual, o lo que es lo mismo, el cálculo de la variable que rige el comportamiento del sistema. Viene definida por la ecuación:

$$\boldsymbol{EST_{actual}} = \boldsymbol{Predicción} + \boldsymbol{K}\,[\boldsymbol{Medida} - \boldsymbol{Predicción}]$$

  Es importante volver a incidir en que el filtro de Kalman es un estimador o función estadística y, como tal, realizará el tratamiento de la medida. En este sentido, tratará las medidas como un conjunto de valores de una distribución normal para los cuales minimizará su error cuadrático. No puede, por tanto, mejorar mediciones que se

encuentren afectadas por sesgos o errores sistemáticos que no estén dentro del ámbito del ruido blanco.

Analizando la ecuación, cuando la Ganancia de Kalman tiende a uno, la salida estará principalmente determinada por la medida. En el otro extremo, cuando la Ganancia de Kalman esté próxima a cero, la salida del filtro, es decir, la estimación, estará fuertemente condicionada por la predicción.

- **Error en la estimación actual:** Es una indicación de cómo de próximos están los valores estimados a los medidos. Se calcula a partir de la siguiente ecuación:

$$\boldsymbol{E_{EST_{actual}}} = \frac{(\boldsymbol{E_{MEDIDA}})(\boldsymbol{E_{PREDICCIÓN}})}{(\boldsymbol{E_{MEDIDA}}) + (\boldsymbol{E_{PREDICCIÓN}})}; \qquad \boldsymbol{E_{EST_{actual}}} = (\mathbf{1} - \boldsymbol{K})\, \boldsymbol{E_{PREDICCIÓN}}$$

En este caso, cuando la Ganancia de Kalman tiende a uno, el error en la estimación actual va a ser muy pequeño. En efecto, cuando $K = 1$, el filtro considera que el error en la medida es muy pequeño, y el valor que estima es, por tanto, muy cercano a ella. Para la siguiente iteración, conseguirá comparar la medida nueva con un valor de su predicción que, por estar basado en el estado anterior, va a ser también muy próximo a la medida, con lo que *K* tenderá a cero. Cuando $K = 0$, el error en la medida es muy grande y el valor estimado es muy próximo al valor predicho, por lo que el error actual será el error de la predicción. Como conclusión, cuando *K* sea muy grande la estimación se acerca rápidamente al valor medido, mientras que si *K* es pequeña, la estimación se acercará más lentamente a la medida, lo que se traduce por pequeñas oscilaciones en el entorno de la misma.

Es interesante observar que, si bien será muy extraño que *K* tome valores extremos, el error en la estimación actual siempre será más pequeño que el error en la predicción, que está basado en el error de la estimación anterior, lo que garantiza la convergencia del filtro.

6.3 Definición de varianza y covarianza

El filtro de Kalman está basado en una serie de conceptos estadísticos. Su definición facilitará la comprensión del funcionamiento del filtro en los siguientes apartados:

- **Varianza:** Medida de la dispersión de una variable respecto de su media. Se calcula como el cuadrado de la desviación típica, que representa la distancia promedio de cada punto respecto de la media. En una distribución normal, un intervalo de $(-\sigma, +\sigma)$, centrado en la media representa aproximadamente 2/3 de los valores de la muestra, siendo σ la desviación típica. Si sustituimos el valor de la desviación típica por el de la varianza, el nuevo intervalo abarcará el 100 % de los valores de la muestra.

  Siendo x_i los valores individuales, $\bar{x}$ la media de los valores y $\bar{x} - x_i$ la desviación de cada valor respecto de la media:

$$Desviación\ típica = \sigma = \sqrt{\frac{\sum_{i=1}^{N}(\bar{x} - x_i)^2}{N}}$$

$$Varianza = \sigma^2 = \frac{\sum_{i=1}^{N}(\bar{x} - x_i)^2}{N}$$

- **Covarianza:** Indica el grado de variación conjunta de dos variables aleatorias respecto de sus medias, es decir, determina si existe algún tipo de dependencia entre ellas. La varianza se puede definir también como la covarianza de una variable respecto de sí misma.

  Si vale cero, significará que no existe una relación lineal entre ellas. Por otro lado, si grandes o pequeños valores de una se corresponden con grandes o pequeños valores de otra, esto es, varían conjuntamente, la covarianza tomará valores positivos. Si, por el contrario, grandes valores de una se corresponden con pequeños valores de la otra, o viceversa, la covarianza tendrá un valor negativo, expresando esto comportamientos opuestos entre las variables.

$$\sigma_x\sigma_y = \frac{\sum_{i=1}^{N}(\bar{x} - x_i)(\bar{y} - y_i)}{N}$$

Cuando, de ahora en adelante, se mencionen en este trabajo las matrices de covarianza, serán aquellas matrices que incluyen las desviaciones de los parámetros que sean considerados en ese momento como muestra respecto a la media de dicha muestra. Tales matrices estarán formadas por los términos correspondientes a la varianza en la diagonal principal, y por las covarianzas de los parámetros fuera de ella. Un ejemplo de esto en 3D sería:

$$M = \begin{bmatrix} \sigma_x^2 & \sigma_x\sigma_y & \sigma_x\sigma_z \\ \sigma_y\sigma_x & \sigma_y^2 & \sigma_y\sigma_z \\ \sigma_z\sigma_x & \sigma_z\sigma_y & \sigma_z^2 \end{bmatrix}$$

Las matrices de covarianza serán las encargadas de cuantificar el error cometido por el filtro, tanto en la predicción, como en las medidas o en la estimación del estado real. Por ello, quizás el término error no es del todo acertado, refiriéndonos al usarlo a la incertidumbre o dispersión que experimentarán los valores que estén siendo considerados respecto a su media.

6.4 Filtro de Kalman multidimensional

En el apartado 6.2 se realizaba una pequeña introducción al funcionamiento de un filtro de Kalman de una variable. No obstante, en la realidad, cuando un filtro es implementado, no sólo se quiere conocer el valor real de una variable, sino de todas aquellas que sean capaces de medir los sensores y resulten útiles para alguna de las muchas aplicaciones del filtro. Implementar un filtro por cada una de estas medidas sería prácticamente imposible, ya que se necesitaría que fuesen realizadas en paralelo si se habla de una implementación a tiempo real. Es aquí donde nace la idea de realizar un filtro multidimensional, que estime el valor de más de una variable al mismo tiempo. Para ello, será necesario que el modelo lineal del sistema dinámico incluya todas estas variables, por lo que deberán guardar algún tipo de relación entre ellas. Además, será preciso definir una serie de matrices que permitirán el manejo conjunto de

todas las variables que estén siendo tratadas. En la Figura 6.2 se muestra un diagrama de las ecuaciones que son necesarias para elaborar un filtro de estas características, así como el orden de aplicación de las mismas.

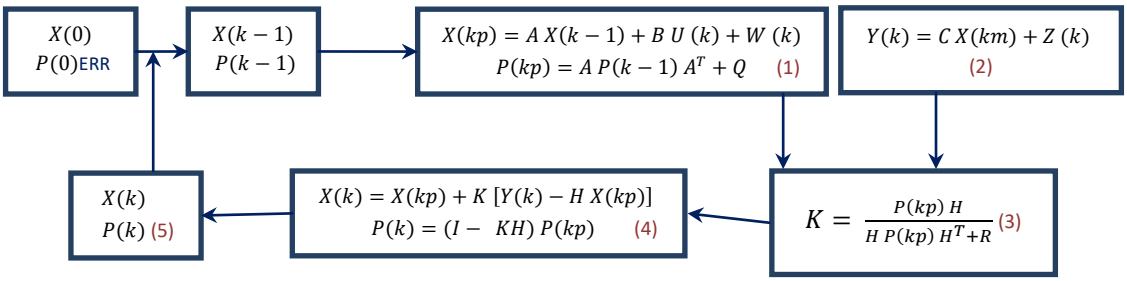

Figura 6.2: Esquema de un filtro de Kalman multidimensional

Como puede observarse, la estructura es igual que la de un filtro de una variable. A continuación se detalla lo que representan cada una de estas matrices y la función que tienen:

- **Creación de un estado anterior:** La salida del filtro, es decir, la estimación realizada por el filtro en la iteración anterior, pasa a convertirse en la nueva entrada del filtro (cuadro 5 de la Figura 6.1). En el caso de la primera iteración, al no existir una salida del filtro, se utilizan valores iniciales previamente establecidos.
    - **X(0):** Vector inicial de estado. En él se incluirán los valores iniciales de las variables que se quieren filtrar.
    - **P(0):** Matriz inicial de la covarianza del proceso. Esta matriz recoge los errores en el cálculo del vector inicial de estado.
    - **X(k-1):** Vector de estado correspondiente al estado anterior. Incluye la estimación de las variables realizada por el filtro en la iteración anterior (*k-1*).
    - **P(k-1):** Matriz de covarianza del proceso del estado anterior. Representa los errores que han sido cometidos en la estimación realizada en la iteración anterior (*k-1*).

- **Calculo de la predicción:** El filtro, basándose en el modelo lineal definido, y utilizando como base el valor estimado de la iteración anterior, realiza una predicción de cuál será el siguiente estado del sistema (primera ecuación del cuadro 1 de la Figura 6.2).

    - **X(kp):** Vector de estado predicho (*p*) para la iteración actual (*k*). Vector que recoge el valor predicho de cada una de las variables de estado, obtenido en este caso mediante la aplicación del modelo lineal que representa el comportamiento del sistema en estudio.
    - **A:** Es la matriz de coeficientes de aquellos términos del modelo lineal que dependen de las variables de estado, es decir, es la matriz que relaciona las variables entre sí.
    - **B:** Es la matriz de coeficientes de aquellos términos del modelo lineal que no dependen de las variables de estado, es decir, de aquellos términos que

permanecerán constantes durante la implementación del filtro. Puede servir también para modificar las dimensiones del vector *U(k)*.

- **U(k):** Vector de variables de control. Incluye los términos del modelo lineal que permanecen constantes durante la predicción de las variables, por ejemplo, aceleraciones conocidas como la gravedad o una fuerza de empuje constante.
- **W(k):** Vector de ruido del modelo dinámico. Pasar de un modelo real a un modelo matemático implica errores de aproximaciones, redondeos o simplificaciones, como puede ser una pequeña componente de viento aleatorio que modificaría la trayectoria en un movimiento. En definitiva, se asume como ruido la diferencia entre el modelo real sobre el que se realiza el estudio, y el modelo matemático que lo representa. Esa diferencia está compuesta por todos aquellos elementos del sistema real no considerados en el modelo matemático, y por las aproximaciones realizadas para aquellos elementos que sí están incluidos.

- **Predicción de la matriz de covarianza:** Una vez realizada la predicción de las variables del vector de estado, también es preciso predecir cuál será la matriz de covarianza, es decir, predecir cuál será el grado de dispersión que tendrá la estimación del filtro (segunda ecuación del cuadro 1 de la Figura 6.2).
    - **Q:** Matriz de ruido de la covarianza del proceso. Se suma a la predicción de la matriz de covarianza del vector de estado, basada en la matriz de covarianza de la estimación anterior. Modela los errores o dispersión esperados para el ruido del modelo dinámico. Su función es importante, cuando la matriz de covarianza de la salida del filtro, P(k) (cuadro 4 de la Figura 6.2), tiende a cero, su realimentación en la siguiente iteración en el cálculo de la predicción de la matriz de covarianza, P(kp) (cuadro 1 de la Figura 6.2), haría que esta fuese también próxima a cero. La suma de la matriz Q introduce un factor de error o dispersión asociado al ruido del proceso que permite al filtro seguir comparando su predicción con la medida.
    - **P(kp):** Matriz de covarianza del proceso. Predicción de la matriz de covarianza basada en el modelo dinámico y en la matriz de covarianza de la iteración anterior.

- **Entrada de la medida:** Es la segunda ecuación que proporciona información sobre el sistema bajo estudio. Incluye los cambios que deben ser realizados sobre la medida obtenida de los sensores para adaptarla a la forma y estructura del vector de estado.

    - **Y(k):** Vector de la observación para la iteración actual (*k*). Vector que recoge el valor que proporciona la medida sobre cada una de las variables de estado (cuadro 2 de la Figura 6.2).
    - **C:** Matriz que adapta las medidas obtenidas directamente de los sensores a la estructura del vector de estado. Puede implicar un cambio de unidades, de ejes, de posición dentro del vector o, en el caso de que la misma variable sea medida por más de un sensor, una fusión de datos.

- **X(km):** Vector que contiene las medidas obtenidas directamente de los sensores.
- **Z(k):** Vector de ruido de la medida. Al igual que ocurría con el vector *W* para la predicción, la medición puede tener unos pequeños errores asociados a los sensores, como pueden ser errores de calibración.

- **Matriz de covarianza de la medida:** Determina el grado de dispersión que tienen los valores proporcionados por los sensores. Se utiliza para calcular la Ganancia de Kalman (cuadro 3 de la Figura 6.2).

    - **R:** Matriz de covarianza de la medida. Modela los errores o dispersión presentes en la medida, tanto por causa de los sensores como de los cálculos, al modelar el vector de ruido *Z*.

- **Cálculo de la Ganancia de Kalman:** Utilizando las matrices de covarianza calculadas, se encarga de ponderar entre la medida y la predicción, para decidir cuál de las dos presenta mayor incertidumbre y, por tanto, es menos fiable (cuadro 3 de la Figura 6.2).
    - **K:** Ganancia de Kalman. Su valor oscila entre cero y uno.
    - **H:** Matriz que adapta la estructura de la matriz de covarianza del proceso para que coincida con la matriz de covarianza de la medida. También se encarga de adaptar la forma del vector de estado a la forma del vector de la medida en el cálculo de la estimación, ya que puede ser que las dimensiones de los vectores de estado de la medida y la predicción no coincidan. Normalmente, si las matrices tienen la misma dimensión, se toma como una matriz identidad.

- **Estimación del estado actual:** Solución del filtrado. Una vez ponderada la medida y la predicción, el filtro estima una solución del vector de estado para la iteración actual (k) (cuadro 4 de la Figura 6.2).
    - **X(k):** Vector de estado que contiene la estimación del estado actual.

- **Matriz de covarianza de la estimación actual:** Matriz que representa el grado de dispersión que tienen los valores estimados del vector de estado respecto a la media de los mismos. Cuantifica cómo de buena es la estimación (cuadro 4 de la Figura 6.2).
    - **P(k):** Matriz de la covarianza de la estimación del estado actual.
    - **I:** Matriz identidad de la misma dimensión de P(k).

Tanto la matriz de la estimación del estado actual como la matriz de la covarianza de la estimación pasarán a llamarse estado anterior y realimentarán al filtro en la siguiente iteración.

7. APLICACIÓN DEL FILTRO DE KALMAN LINEAL

Una vez estudiado qué es un filtro de Kalman y el álgebra detrás de su implementación, se detallará en este apartado una aplicación a un modelo real. Para ello, definiremos el modelo y las ecuaciones que van a regir su comportamiento. Posteriormente, se explicarán las pautas seguidas para la inicialización de los vectores y matrices, así como la manera de calcular las matrices de covarianza, tanto de la predicción como de la medida. Finalmente, se mostrarán las gráficas de los resultados obtenidos y se analizarán para determinar su grado de corrección y si se corresponden con los datos conocidos, obtenidos de la experiencia.

7.1 Modelo dinámico lineal

El experimento realizado consiste, como ya fue explicado en el apartado 4.5, en un movimiento rectilíneo y uniforme realizado con un vehículo a lo largo de una autovía. El vehículo se desplazaba a velocidad constante ($\sim 75 Km/h$), y presentaba algunas aceleraciones aleatorias, derivadas del estado del firme y de su pendiente, así como otros fenómenos no parametrizables, que no han sido tenidas en cuenta en la ecuación del modelo, sino que aparecerán incluidas de las ecuaciones de ruido. Por ello, las variables de estado del sistema serán la posición en los tres ejes coordenados (x, y, z), así como las tres velocidades correspondientes a cada uno de estos ejes.

Las ecuaciones que rigen la posición en el movimiento rectilíneo y uniforme del vehículo son:

$$x(k) = x(k-1) + v_x * t\,; \qquad y(k) = y(k-1) + v_y * t\,; \qquad z(k) = z(k-1) + v_z * t\,;$$

Mientras que las ecuaciones que rigen la velocidad son:

$$v_x = v_x\,; \qquad v_y = v_y\,; \qquad v_z = v_z\,;$$

Expresando esto de forma matricial en función de las variables de estado:

$$\underbrace{\begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{bmatrix}}_{X(kp)} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & t & 0 & 0 \\ 0 & 1 & 0 & 0 & t & 0 \\ 0 & 0 & 1 & 0 & 0 & t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{bmatrix}}_{X(k-1)}$$

Al no tener este sistema ninguna fuerza externa actuando, como puede ser el considerar la acción de la gravedad en un móvil volador, o una aceleración, que se añadirían como función de las variables a las ecuaciones del modelo, la matriz $U(k) = 0$.

7.2 Inicialización de las matrices

Una vez obtenido el modelo dinámico, falta definir cuáles serán los criterios para inicializar el resto de matrices que necesita el filtro para comenzar su funcionamiento. Algunas de ellas

permanecerán constantes durante todo el filtrado, mientras que otras serán inicializadas para la primera iteración siendo posteriormente actualizadas por el filtro.

- **Matrices constantes durante el funcionamiento del filtro:**
    - **H:** utilizada para adaptar las dimensiones entre dos matrices. En el ejemplo lineal, no es necesaria ninguna modificación, por lo que:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

    - **C:** realiza las operaciones necesarias sobre las medidas directamente obtenidas de los sensores para adaptarlas a la forma, unidades y ejes en los que esté expresado el vector de estado. En este caso, las transformaciones fueron realizadas fuera del filtro, según el apartado 5, por lo que no es necesario ningún cambio adicional:

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

    - **W(k):** Corrección de la predicción del vector de estado basada en el modelo. Se trata de una desviación constante aplicada directamente sobre el vector de estado que modela la contribución al movimiento de algún agente externo, como podría ser una componente de viento en la dirección x de $5Km/h$. En este caso, al no ser medible esta contribución, se ha tomado una desviación simbólica:

$$W = \begin{bmatrix} 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \end{bmatrix}$$

    - **Z(k):** Corrección de la medida, a partir de una desviación conocida del sensor, que se desea corregir. Hace que la estimación del filtro se desplace una constante. El sistema GPS introduce errores variables intencionadamente para uso comercial, a fin de que el sistema no pueda ser utilizado para aplicaciones militares. Para tales usos militares existe un modo denominado modo P o de "disponibilidad condicional", en el cual se dispone de la máxima precisión del sistema (entorno y por debajo del metro). A fin de tener en cuenta este error intencionado introducido en el sistema se ha considerado una desviación simbólica:

$$Z = \begin{bmatrix} 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \end{bmatrix}$$

- **Matrices válidas para la primera iteración:**
  - **X(0):** Considerando que se trata del inicio del movimiento, y que además el experimento está realizado a nivel del mar, todas las variables de estado serán inicializadas a cero. Se comprobará así la capacidad de convergencia del filtro a la estimación correcta.

$$X(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

  - **P(0):** La matriz de la covarianza del proceso debe ser cuadrada, de la misma dimensión que la mayor del vector de estado, e incluirá las varianzas y covarianzas entre las distintas variables. En muchos de los ejemplos de implementación del filtro consultados en la literatura toman la matriz inicial como diagonal, con un valor aleatorio y grande. Esto es porque se supone que no existe relación alguna entre las variables del vector de estado, por lo que sus covarianzas son cero y solo es necesario representar las varianzas. En el apartado 7.4 se realizará un estudio de la rapidez de convergencia del filtro según los valores iniciales de esta matriz. Un ejemplo sería:

$$P(0) = \begin{bmatrix} 7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 \end{bmatrix}$$

7.3 Cálculo de las matrices de covarianza Q y R

Existen diversos métodos para el cálculo de estas matrices. En el estudio realizado para su cálculo se ha observado que, a veces, por las condiciones del problema considerado y a la espera de unos resultados concretos, pueden tomarse como matrices diagonales constantes durante toda la implementación del filtro, donde cada elemento de dicha diagonal representa la posible varianza de la variable a la que está asociado. En otros casos, son calculadas en cada iteración mediante un método adaptado al problema. En este estudio, se ha decidido tomar como método para su cálculo el desarrollado en [10], que es detallado a continuación.

El cálculo de las matrices Q y R va a ser análogo y está basado en la definición de varianza y covarianza vista en el apartado 6.3. Por ello, las matrices tomarán la forma:

$$Q = \frac{\sum_0^{n-1}(w_i - \bar{w})(w_i - \bar{w})^T}{n-1}; \qquad R = \frac{\sum_0^{n-1}(v_i - \bar{v})(v_i - \bar{v})^T}{n-1}$$

Donde

$$w_i = X(kp) - X(kp-1); \qquad \overline{w} = \frac{\sum_{i=0}^{n-1}(X(kp)-X(kp-1))}{n}$$

$$v_i = Y(k) - Y(k-1); \qquad \bar{v} = \frac{\sum_{i=0}^{n-1}(Y(k)-Y(k-1))}{n}$$

El cálculo de la covarianza en estos casos, tanto en la predicción como en la medida, se realiza tomando como variable la diferencia entre los valores de la iteración actual y los valores de la iteración anterior.

Lo que se indica en estas matrices Q y R es el grado de dispersión de las diferencias de los valores predichos y medidos, respectivamente, entorno a su media, de una iteración a otra. Cuando esta diferencia se mantiene constante y en un entorno razonable de la media, la covarianza mantendrá un valor estable. En el momento en que una de estas diferencias sea muy diferente a la tendencia que se seguía, el filtro asume que el valor de la iteración actual tiene mucho error o incertidumbre. Sin embargo, si no se trata de un caso aislado y a partir de ese momento las diferencias comienzan a seguir esa nueva tendencia, el filtro se irá ajustando a estos nuevos datos para volver a dar una estimación correcta.

Las matrices de covarianza permiten también observar el mecanismo de adaptación del filtro. El filtrado Kalman se considera como uno de los paradigmas de filtro adaptativo. Así, en el caso bajo estudio, un cambio a una trayectoria rectilínea tras el trazado de una curva provocará que el filtro registre un error en su predicción, que estaba basada en la trayectoria curva seguida por el vehículo hasta ese momento. Reaccionará dando más peso a la medida en detrimento de la predicción, hasta que esta última se adapte a la nueva trayectoria. Estos cambios en la ponderación de la medida y de la predicción del modelo quedan reflejados en los valores que adoptan, durante los ciclos que dura esta adaptación, las correspondientes matrices de covarianza.

7.4 Análisis de los resultados obtenidos

En esta sección se muestran los resultados gráficos obtenidos tras la implementación del filtro de Kalman lineal detallado en el Apéndice C en el caso práctico bajo estudio.

De todas las pruebas realizadas, se tomarán aquellas que recogieron mayor número de datos para realizar las gráficas. Se mostrarán todos los resultados de posición y velocidad para una muestra de datos tomada a 100Hz y otra a 50Hz. Posteriormente, se realizará un estudio sobre cómo afectan los cambios en la matriz de covarianza inicial del filtro P(0) y, por último, se comprobará el efecto que tienen los vectores de ruido sobre los resultados.

Los dos experimentos mostrados aquí están realizados en tramos de vuelta, por lo que se podrá observar como la posición en el eje y varía hacia valores negativos, representando esto una dirección de avance hacia el Sur Geográfico.

7.4.1 Entrada de datos a 100Hz en tramo 1

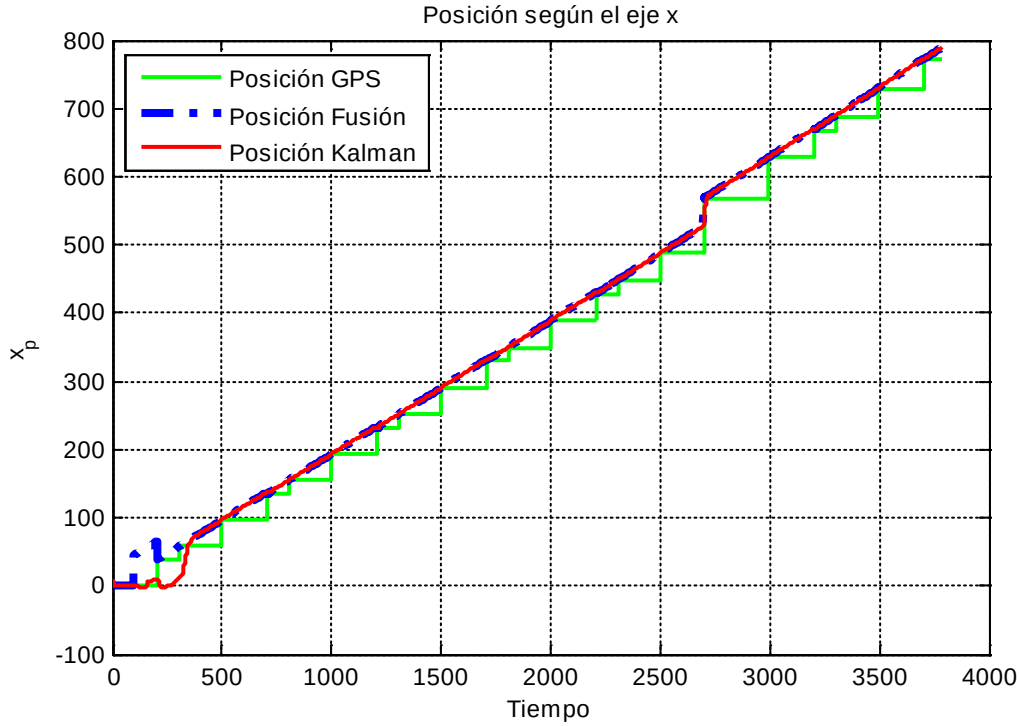

Figura 7.1: Posición en el eje x – filtro lineal – 100Hz

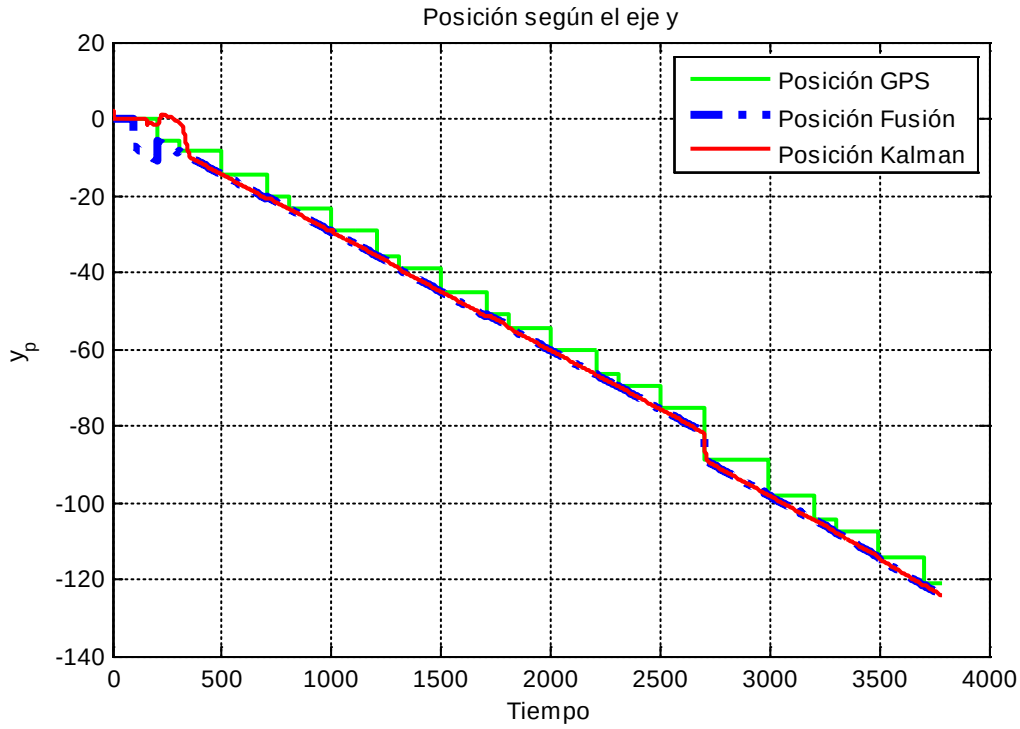

Figura 7.2: Posición en el eje y – filtro lineal – 100Hz

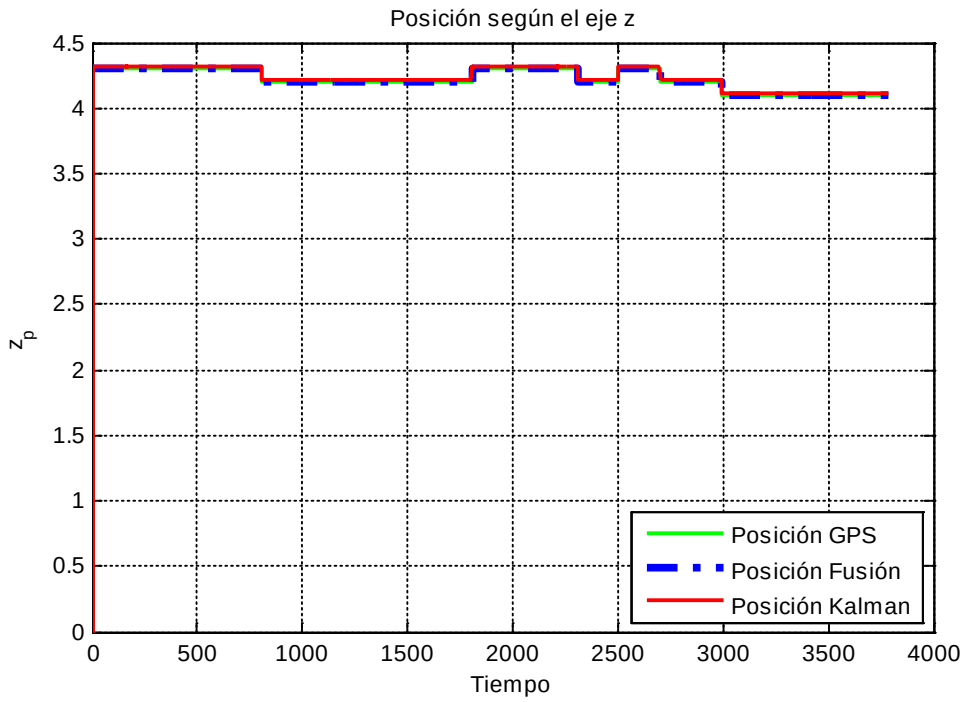

Figura 7.3: Posición en el eje z – filtro lineal – 100Hz

En las gráficas obtenidas para la posición se comprueba que el filtro sigue bien a las medidas obtenidas de los sensores, las cuales, según las matrices W y Z definidas, tienen poco ruido. Las variaciones en la altura, es decir, en la coordenada z pueden ser debidas a desperfectos de la carretera o de rodadura del vehículo.

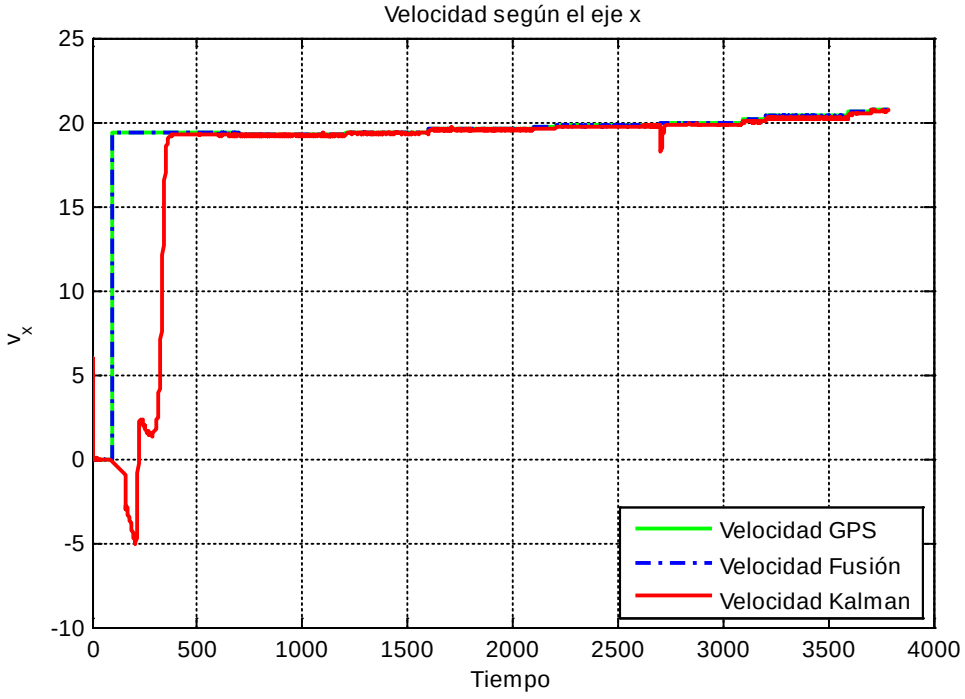

Figura 7.4: Velocidad en el eje x - filtro lineal - 100Hz

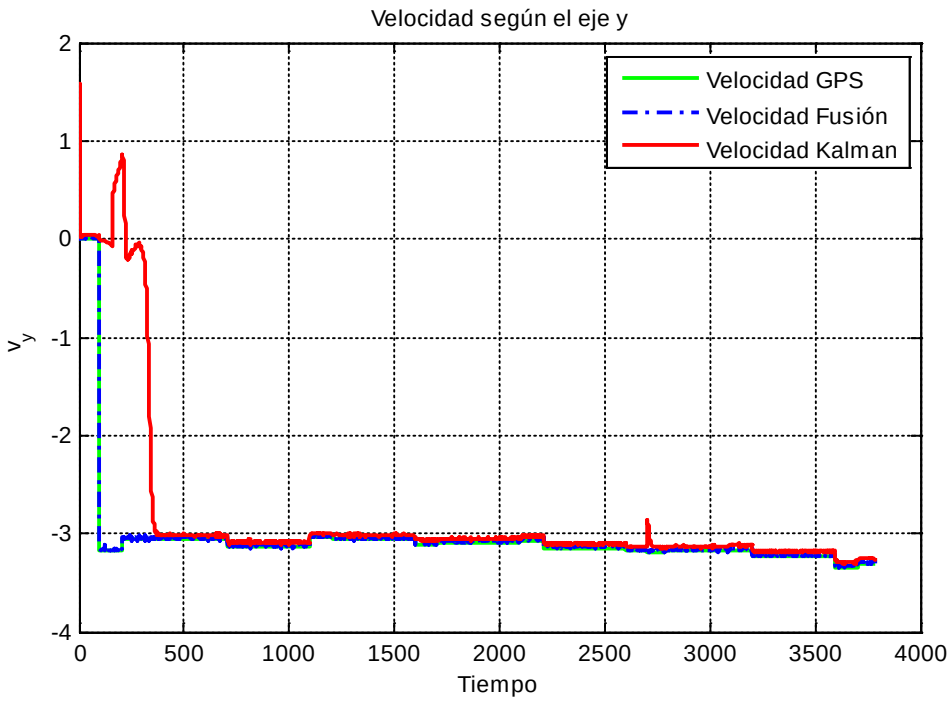

Figura 7.5: Velocidad en el eje y - filtro lineal - 100Hz

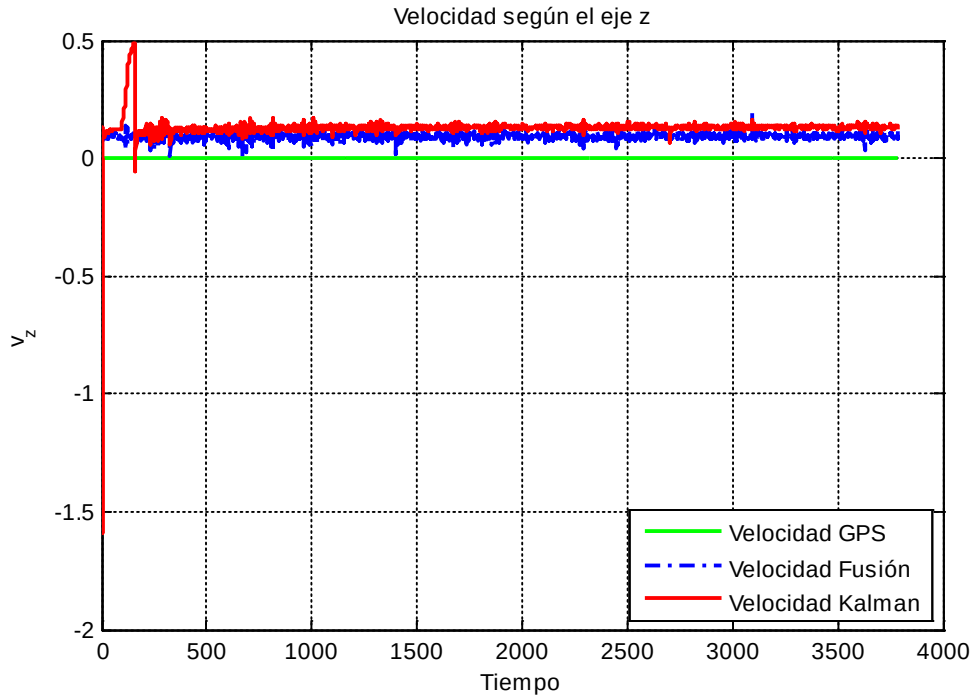

Figura 7.6: Velocidad en el eje z - filtro lineal - 100Hz

Las grandes oscilaciones que tienen lugar al comienzo del filtrado son el resultado de la inicialización de las matrices. En las primeras iteraciones, los valores de la Ganancia de Kalman oscilan rápidamente y el filtro no sabe si hacer más caso a las medidas que al filtro. No obstante, se observa que el filtro se estabiliza antes de llegar a los 5s.

7.4.2 Entrada de datos a 50 Hz en tramo 2

Dado que la frecuencia a la que fueron recogidas las muestras es más baja, pudieron recogerse mayor número de datos y, por tanto, la distancia recorrida fue mayor.

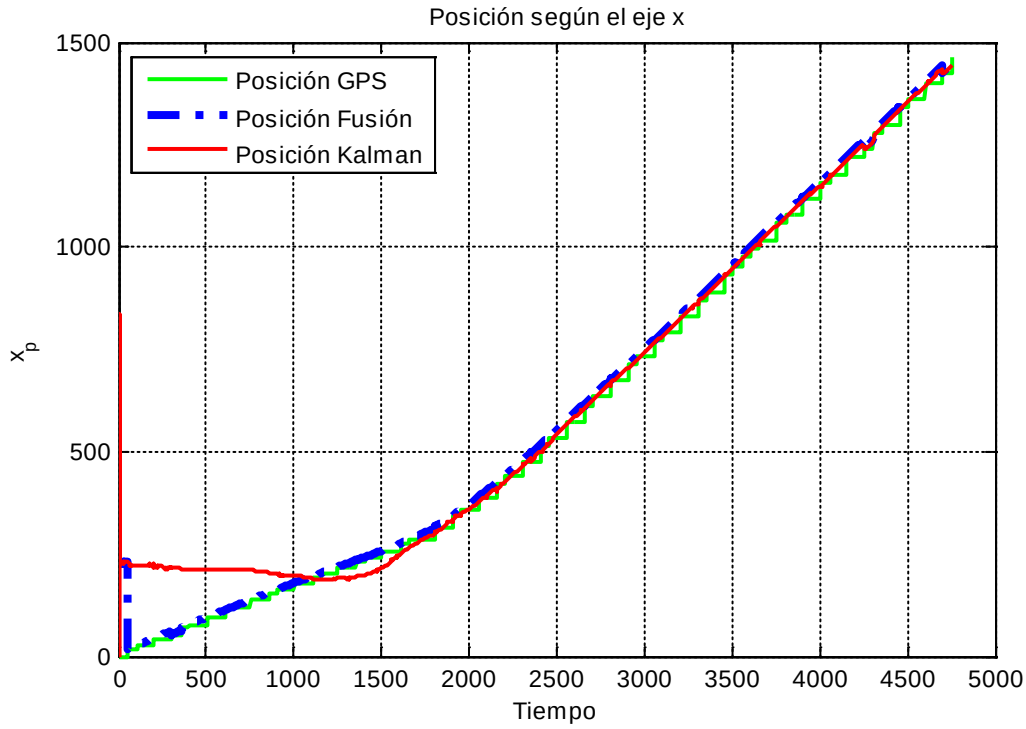

Figura 7.7: Posición en el eje x – filtro lineal – 50Hz

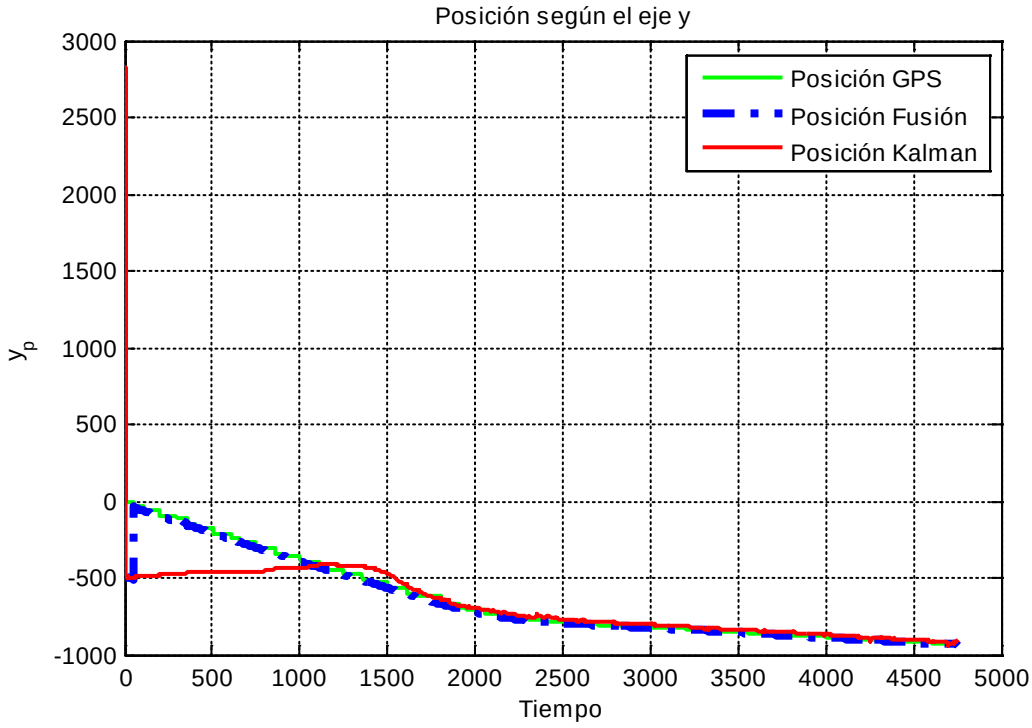

Figura 7.8: Posición en el eje y – filtro lineal – 50Hz

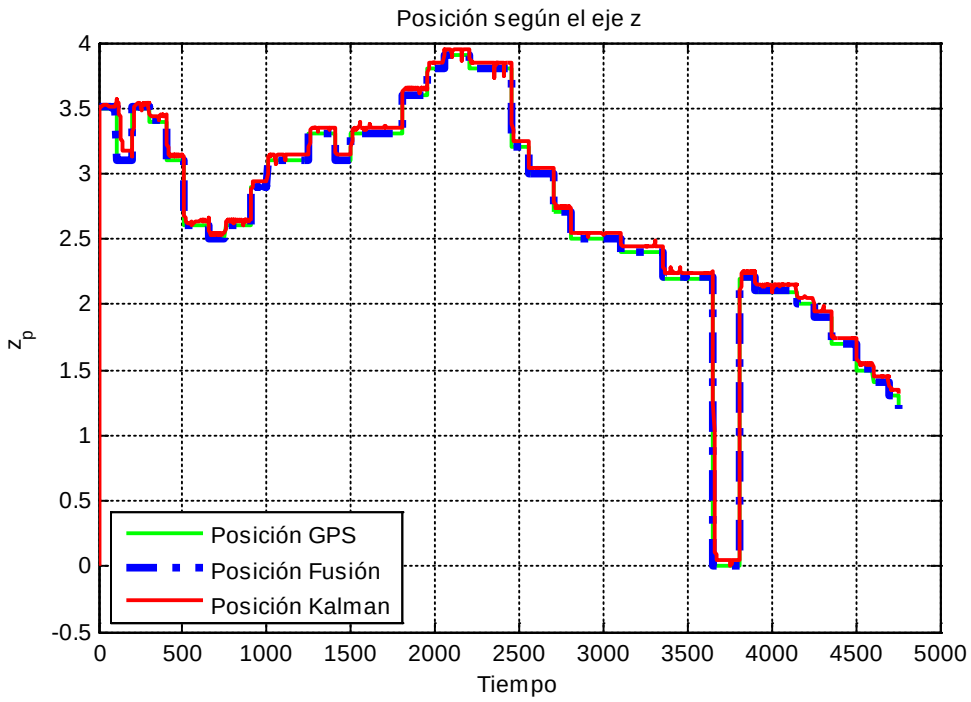

Figura 7.9: Posición en el eje z – filtro lineal – 50Hz

Llama la atención que al comienzo de las estimaciones de la posición en los ejes x e y no solo oscila la salida del filtro, también lo hace la señal fusionada. Puede tratarse de un error en la recorrida de datos, porque durante un segundo se perdiera la señal y no hubiese entrada de dato nuevo de GPS. Al ser la frecuencia de datos menor se comprueba que el filtro tarda más en establecerse, porque pasa el doble de tiempo entre dos estimaciones.

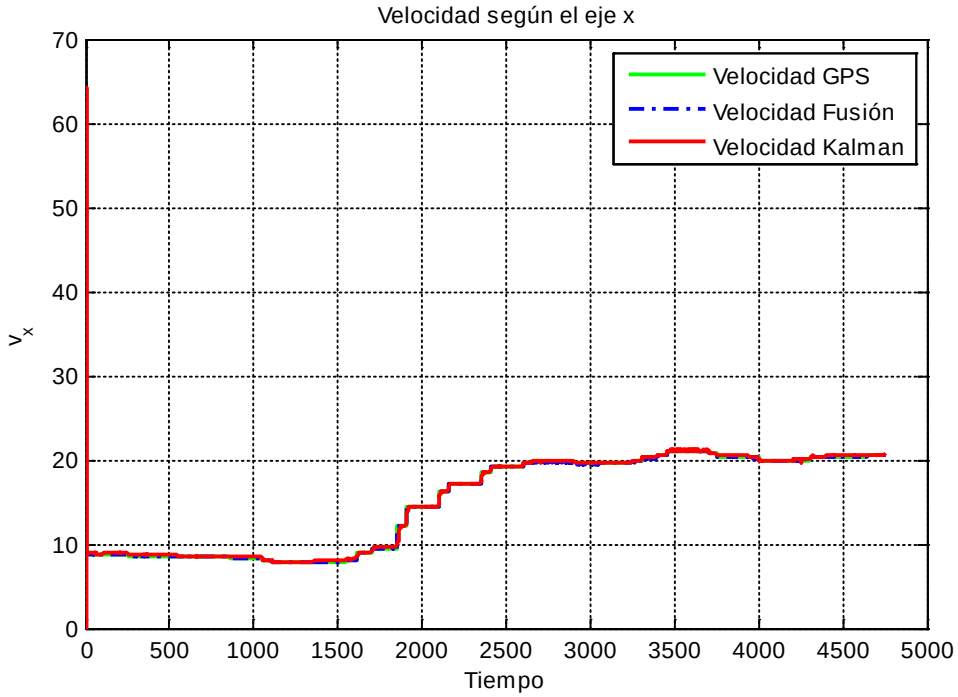

Figura 7.10: Velocidad en el eje x - filtro lineal - 50Hz

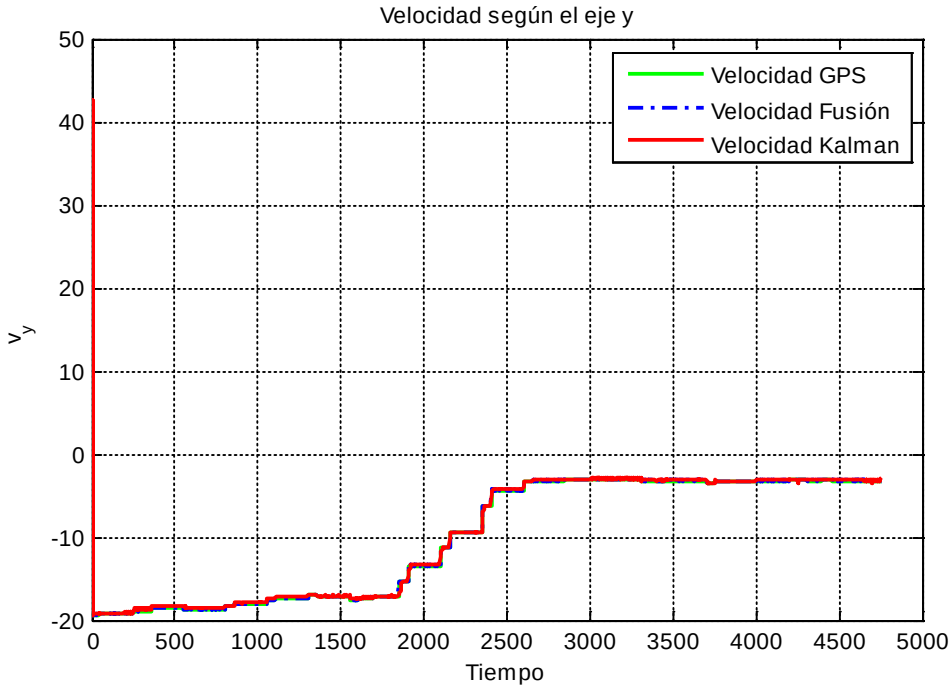

Figura 7.11: Velocidad en el eje y - filtro lineal - 50Hz

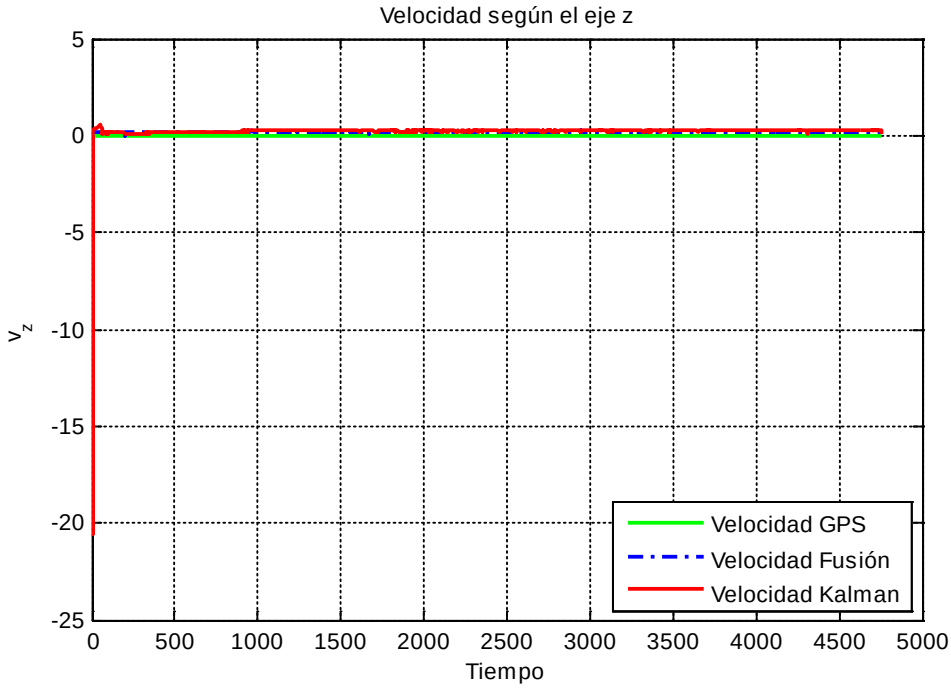

Figura 7.12: Velocidad en el eje z - filtro lineal - 50Hz

A pesar de las grandes oscilaciones que ocurren en estos casos al comienzo de las iteraciones, las velocidades en el caso de un muestreo más lento se estabilizan antes. El error entre la estimación del filtro de Kalman y la medida es del mismo orden que en caso de 100Hz. Ya que la gran variación de altura de la Figura 7.9 no se refleja en una variación de la velocidad, podemos concluir que se trata de un error en la recepción de las medidas.

7.4.3 Inicialización con distintas matrices de covarianza iniciales

Se estudiará el comportamiento del filtro con una matriz de covarianza inicial diagonal de pequeño tamaño, como se definió en el apartado 7.2, para después continuar con cinco pruebas más.

Por mayor claridad en el tiempo de convergencia de las matrices se realizará la implementación del filtro para los datos obtenidos a una frecuencia de 50Hz. Por otro lado, se ha decidido realizar el experimento con una única variable de estado, siendo esta representativa del comportamiento del filtro con el resto de variables. A la vista de los resultados obtenidos en el apartado 7.4.2, se ha decidido que esa variable sea la posición según el eje x.

- **Matriz diagonal de elevados valores:**

$$P_2(0) = 500 * \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

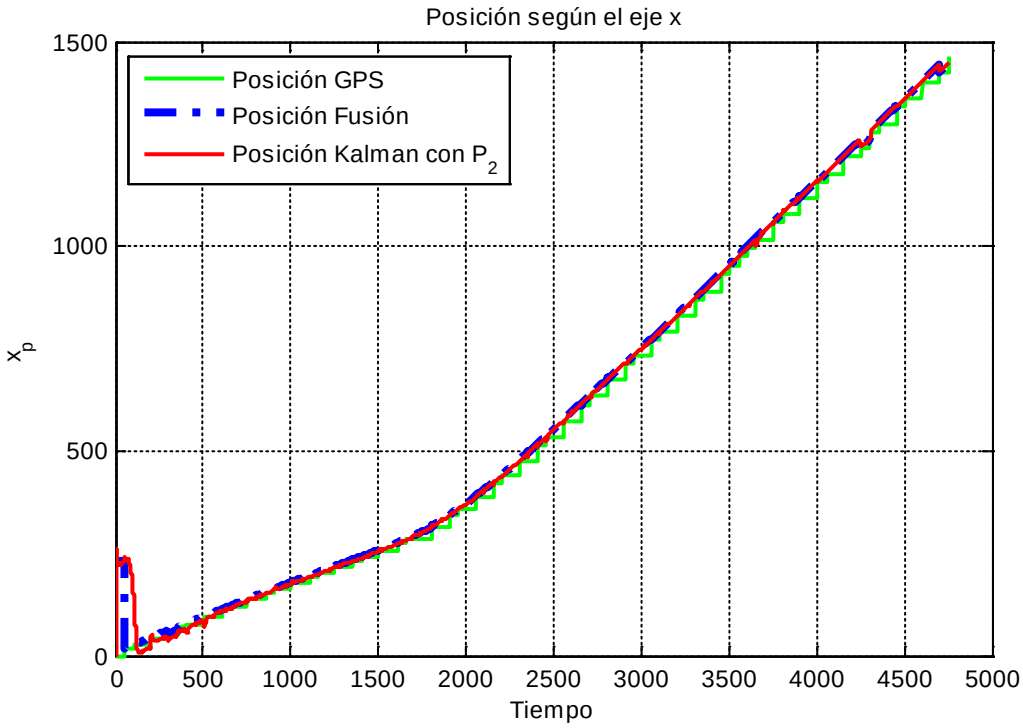

Figura 7.13: Posición en el eje x – P_2 – 50Hz

- **Matriz completa con valores bajos:**

$$P_3(0) = 7 * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

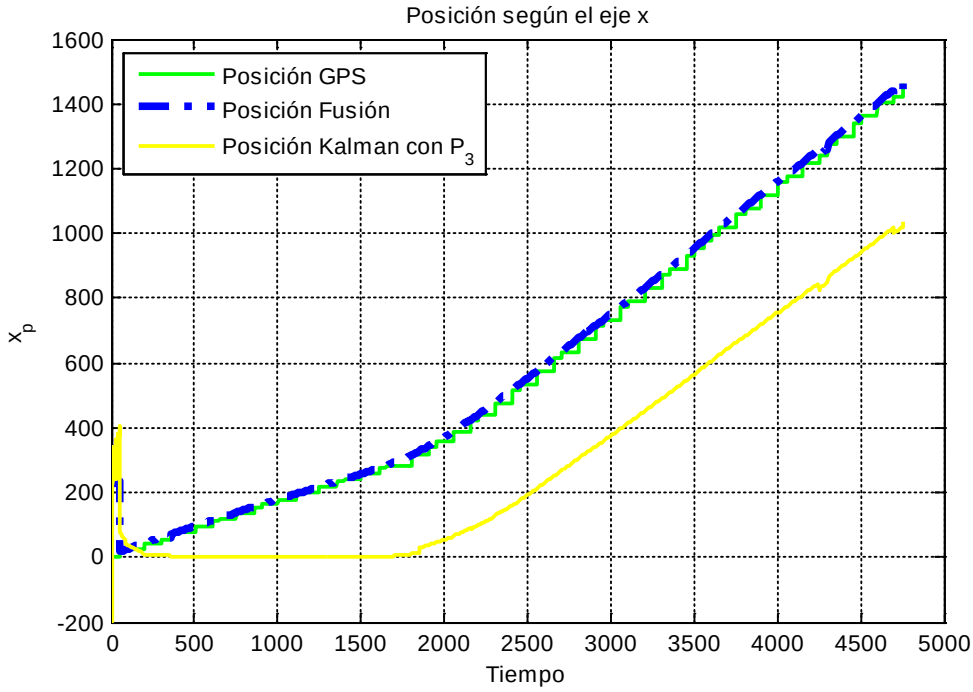

Figura 7.14: Posición en el eje x – P_3 – 50Hz

- **Matriz completa con valores elevados:**

$$P_4(0) = 500 * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

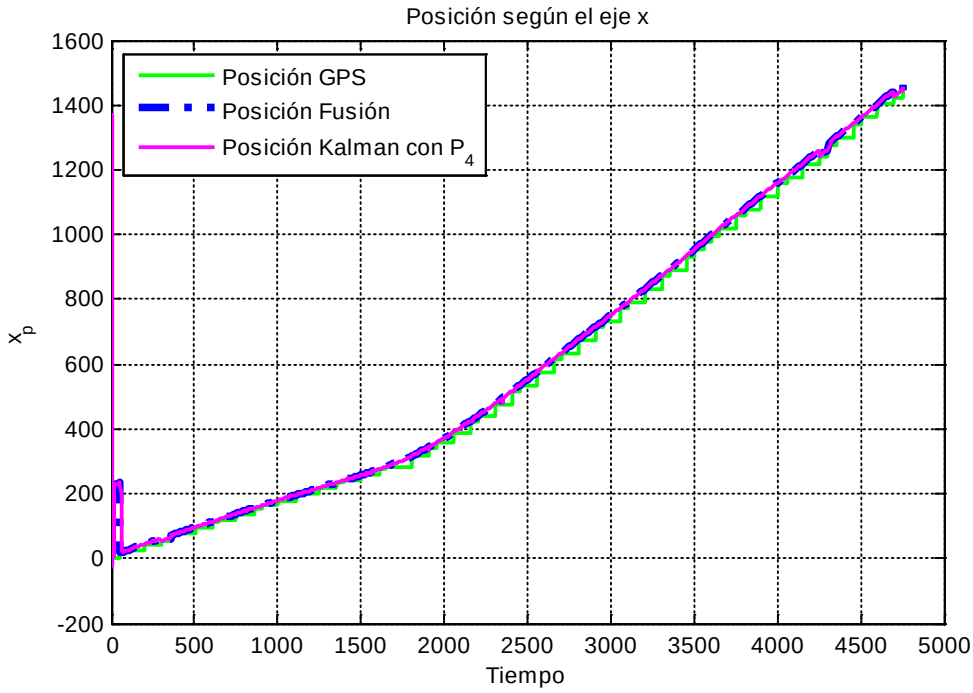

Figura 7.15: Posición en el eje x – P_4 – 50Hz

- **Matriz de covarianzas negativas con valores pequeños:**

$$P_5(0) = 7 * \begin{bmatrix} 1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & 1 \end{bmatrix}$$

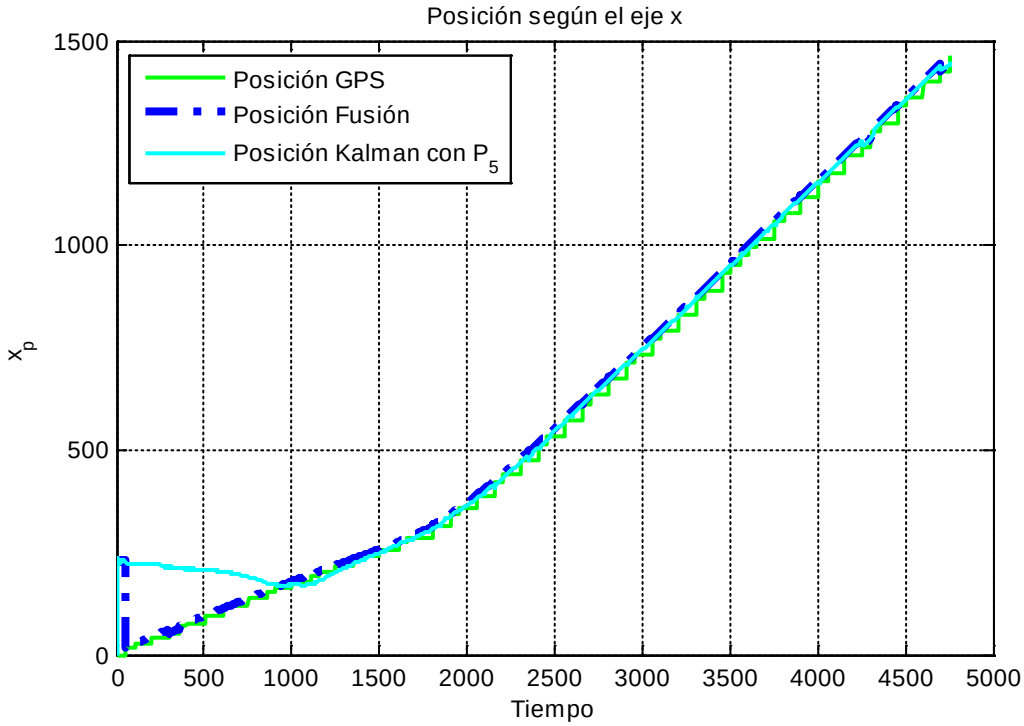

Figura 7.16: Posición en el eje x – P_5 – 50Hz

Como se puede comprobar en estas gráficas, el valor de la matriz de covarianza inicial afecta a la rapidez con la que el filtro se establece en el entorno de la medida. Los mejores resultados se obtienen para una matriz diagonal de valores elevados (matriz P_2) ya que, aunque cuando la matriz es completa de números elevados también se posiciona en los alrededores de la medida rápidamente, aparece una oscilación inicial brusca.

Se deduce que valores pequeños para las matrices de covarianza no benefician la estabilización de las estimaciones, así como que las matrices con covarianzas negativas dan resultados muy parecidos con las matrices de covarianza diagonales. Por último, se muestra un ejemplo con una matriz de covarianzas negativas (recordar que la diagonal está formada por elementos al cuadrado que no pueden tomar valores negativos) con valores muy elevados, a la espera de obtener resultados parecidos a los de P_2, como se muestra en la Figura 7.17.

- **Matriz de covarianzas negativas con valores grandes:**

$$P_6(0) = 500 * \begin{bmatrix} 1 & -1 & -1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & -1 & -1 & 1 \end{bmatrix}$$

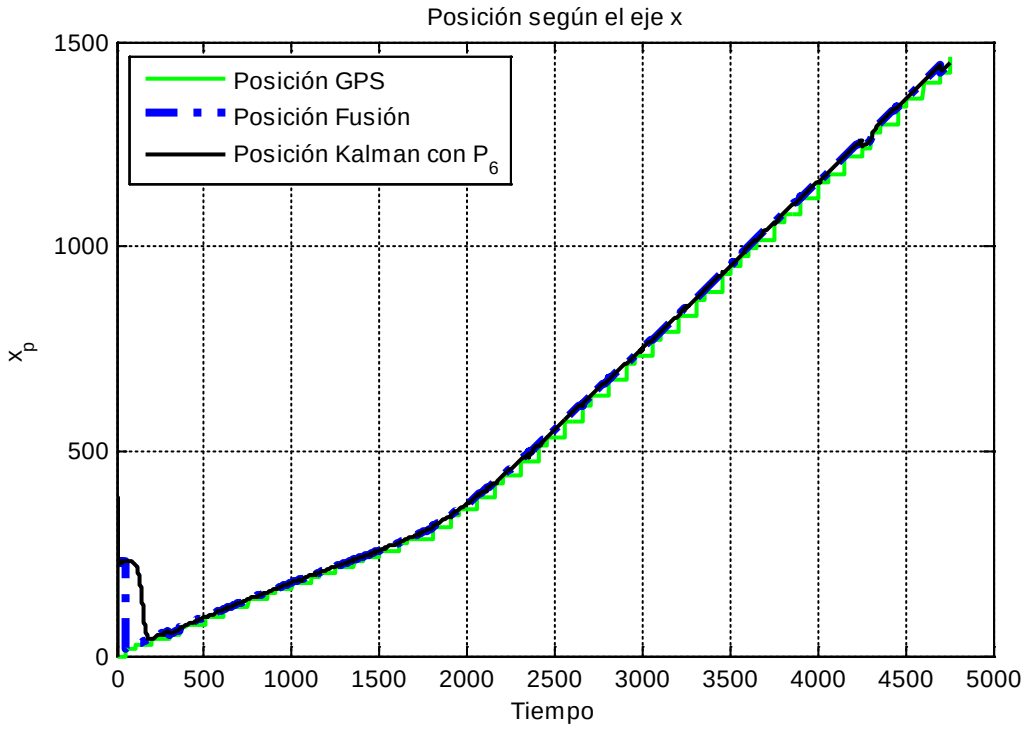

Figura 7.17: Posición en el eje x – P_6 – 50Hz

7.4.4 Efecto de los vectores de ruido W y Z

Por último, se mostrará en este apartado el efecto que tienen los vectores que modelan el ruido del sistema sobre la solución proporcionada por el filtro.

Como se comentó en el apartado 7.2, existen desviaciones intencionadas en las medidas proporcionadas por los sensores. Por otro lado, el modelo dinámico realizado del sistema puede contener algún tipo de desviación por algún agente externo. Estas grandes desviaciones no se corresponden con el concepto de ruido blanco que maneja el filtro, por lo que no es capaz de eliminarlas ni de determinar su valor. En el caso de los sensores, tampoco puede ser solventada con ajustes en la calibración.

Los vectores de ruido añaden al filtrado una desviación constante que contempla estos fallos. Por desconocer el valor de las desviaciones intencionadas del modelo y las medidas, para la implementación del filtro se ha supuesto que son de valor pequeño. Sin embargo, se realiza a continuación una prueba dando valores grandes a estos parámetros, para comprobar gráficamente el efecto que tienen.

Por facilidad en la representación, se ha tomado también una única variable de estado para la observación del efecto, siendo esta representativa del resto de variables a las que se le aplica el filtrado Kalman. La variable de estado será, también en este caso, la posición según el eje x.

Se observará el efecto de la modificación del valor de los vectores por separado, y a continuación el efecto de un aumento en ambos. Se utilizará la matriz $P(0) = P_1$ y se filtrarán los datos obtenidos a 50Hz. Siendo los nuevos valores:

$$W = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}; \qquad Z = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}$$

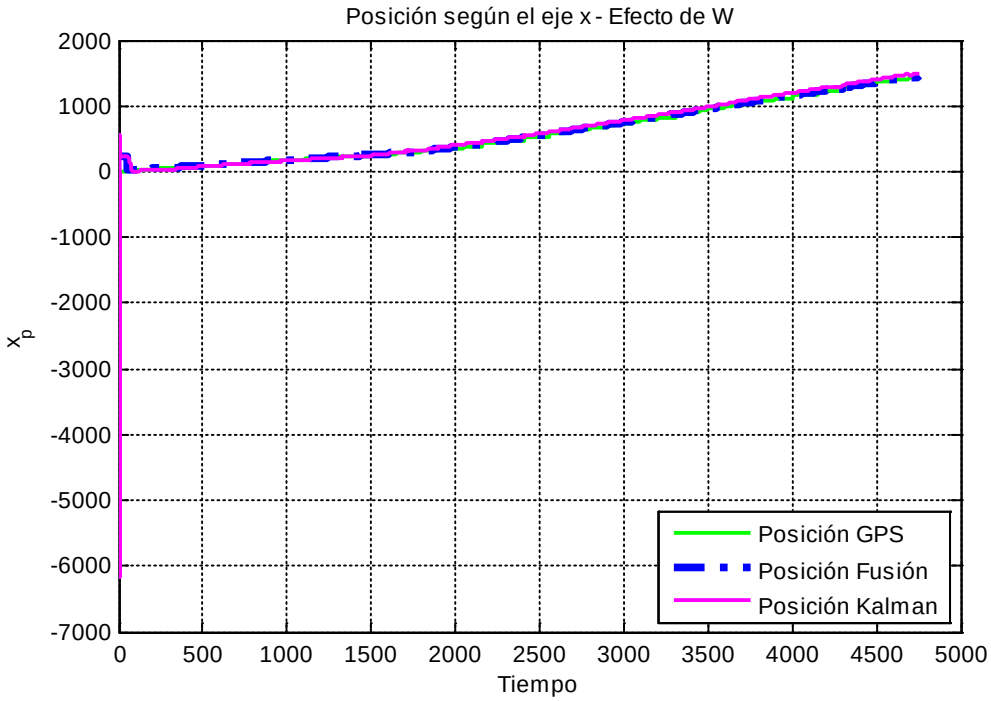

Figura 7.18: Posición en el eje x – Efecto de W – 50Hz

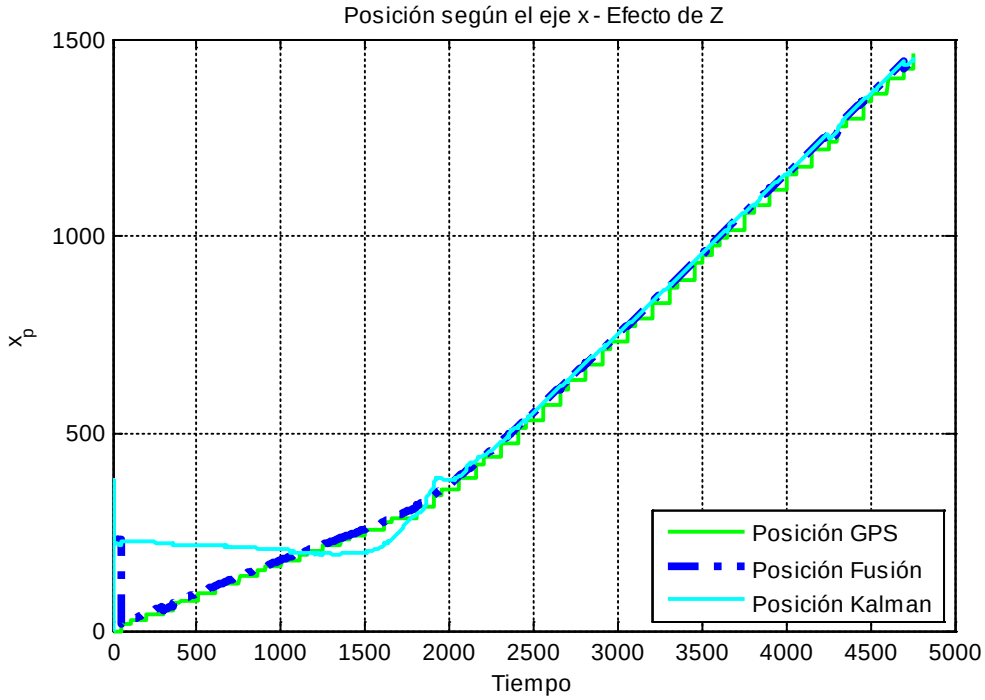

Figura 7.19: Posición en el eje x – Efecto de Z – 50Hz

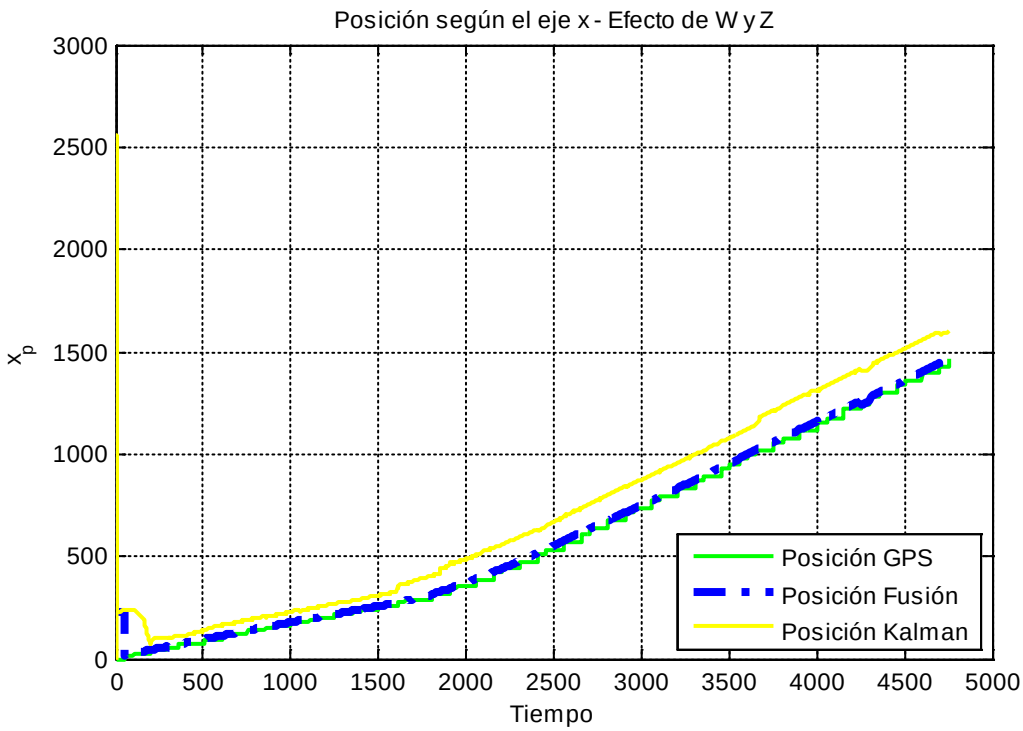

Figura 7.20: Posición en el eje x – Efecto de W y Z – 50Hz

El efecto de W se hace notar sobre el modelo dinámico del sistema. Además de introducir una gran oscilación al comienzo de la estimación, cosa que puede seguir achacándose a errores en los valores iniciales, se puede observar que el resultado del filtro avanza casi paralelo a los datos proporcionados por las medidas.

El efecto de Z en la estimación se hace notar menos, aunque atrasa la capacidad de estabilización del filtro. Sin embargo, cuando se representan ambos errores en conjunto, puede observase como la estimación del filtrado Kalman avanza paralelo a las medidas, incluso dando la impresión de divergir de ellas. Esto es porque el filtro asume esos errores como constantes, y por ello decide que su estimación debe estar desplazada sobre la medida.

8. FILTRO DE KALMAN EXTENDIDO

Esta sección se centrará en explicar de dónde surge y en qué consiste el filtro de Kalman Extendido, cuáles son sus diferencias respecto al filtro lineal y qué ventajas e inconvenientes presenta su utilización.

8.1 Introducción

Cuando realizamos un filtrado hay dos ecuaciones que proporcionan información sobre el sistema bajo estudio: la ecuación de la medida u observación y la ecuación que modela el sistema dinámico. La ecuación de la observación describe cómo han sido tomadas las medidas por los sensores, mientras que la ecuación de modelado describe cómo se espera que el sistema evolucione con el tiempo. En el filtro de Kalman asumimos que estas ecuaciones son lineales, pero existen situaciones en las que no lo son. Reescribiéndolas, según [11], de un modo no lineal, las ecuaciones quedan de la forma:

- **Ecuación de la observación:** si Y_k es la medida, X_k las variables de estado, n_k el ruido de las medidas y g representa una matriz de ecuaciones:

$$Y_k = g(X_k, n_k)$$

- **Ecuación del sistema dinámico:** si X_{k-1} es el estado estimado en el instante $k-1$, a_{kp} es un ruido dinámico aleatorio, X_{kp} es el estado predicho en el instante actual y f es una matriz de ecuaciones del modelo dinámico:

$$X_{kp} = f\left(X_{k-1}, a_{kp}\right)$$

El filtro de Kalman Extendido se aplica cuando la dinámica del sistema o la observación están modeladas por ecuaciones no lineales. Las ecuaciones pueden ser transformadas en lineales realizando una aproximación por serie de Taylor:

$$x(t+\Delta t) = x(t) + \Delta t\dot{x}(t) + \frac{(\Delta t)^2}{2!}\ddot{x}(t) + \frac{(\Delta t)^3}{3!}\dddot{x}(t) + \cdots$$

Cuando Δt o $\ddot{x}(t)$ son muy pequeños, todos los términos de la serie, excepto los dos primeros, pueden despreciarse. Si se aplica esta aproximación a las ecuaciones descritas anteriormente, las ecuaciones en las que se basa un filtro de Kalman Extendido son:

- **Ecuación del sistema dinámico:** La ecuación predice el estado actual del sistema dinámico en base a la siguiente ecuación [12]:

$$X_{kp} = \tilde{x}_k + \frac{\partial f}{\partial x}\left(X_{k-1} - X_{(k-1)p}\right) + \frac{\partial f}{\partial a}a_{kp}$$

  El término $\tilde{x}_k$ representa la última estimación del estado, es decir, la salida del filtro en la iteración anterior.

  El segundo término está basado en la linealización del sistema dinámico entorno al valor conocido del estado correspondiente a la salida del filtro en la última

iteración. La derivada de la función en ese punto es multiplicada por el último incremento conocido del vector de estado, resultado de la diferencia entre el valor de la salida del filtro y la predicción en la iteración anterior, considerada esa diferencia como una señal de error.

La derivada parcial por la que está multiplicada ese error se conoce como Jacobiano. El Jacobiano de un conjunto de funciones de varias variables es la derivada parcial de cada función respecto a cada una de las variables, expresando este resultado en forma matricial. Es importante señalar que, en general, el Jacobiano es dependiente del tiempo, esto es, la forma de la derivada cambia dependiendo de la posición en la que se encuentre en la curva no lineal.

$$\frac{\partial f}{\partial x} = \frac{\partial(f_1, f_2, \dots, f_i)}{\partial(x_1, x_2, \dots, x_j)} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_j} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_i}{\partial x_1} & \cdots & \frac{\partial f_i}{\partial x_j} \end{bmatrix}$$

El tercer término representa el efecto del ruido del sistema dinámico. El valor a_t puede ser considerado como otro incremento del error, y está multiplicado por su respectivo Jacobiano.

- **Ecuación de la medida:**

$$Y_k = \tilde{y}_k + \frac{\partial g}{\partial x}(X_k - X_{kp}) + \frac{\partial g}{\partial n} n_k$$

Donde el término $\tilde{y}_k$ se calcula utilizando la predicción del estado de la iteración actual:

$$\tilde{y}_k = g(X_{kp}, 0)$$

El segundo y tercer término no pueden calcularse, ya que incluyen el vector de estado y el ruido de la medida, que son actualmente desconocidos, aunque sí pueden calcularse sus Jacobianos.

El filtro de Kalman Extendido también se basa, como en el caso del filtro de Kalman lineal, en un método iterativo de corrección – predicción, pero con unas ecuaciones ligeramente modificadas. Las partes del filtro lineal donde se utilizan la matriz de la predicción y de la medida deben ser modificadas para usar matrices no lineales, que serán funciones de g y f, así como de sus respectivos Jacobianos.

Las ecuaciones mostradas sirven para ilustrar de dónde deriva el filtro Extendido, no siendo utilizadas directamente cuando se realiza una implementación del mismo. A continuación, se detallan las ecuaciones que sí serán implementadas, donde aparecerá el efecto que el Jacobiano tiene también sobre las matrices de covarianza [13].

- **Predicción del estado:** $X_{kp} = f(X_{k-1}, 0)$
- **Predicción de la matriz de covarianza:** $P_{kp} = \left(\frac{\partial f}{\partial x}\right) P_{k-1} \left(\frac{\partial f}{\partial x}\right)^T + \left(\frac{\partial f}{\partial a}\right) Q \left(\frac{\partial f}{\partial a}\right)^T$

- **Obtención de la medida:** Y_k
- **Cálculo de la Ganancia de Kalman:**

$$K = P_{kp}\left(\frac{\partial g}{\partial x}\right)^T \left[\left(\frac{\partial g}{\partial x}\right) P_{kp} \left(\frac{\partial g}{\partial x}\right)^T + \left(\frac{\partial g}{\partial n}\right) R \left(\frac{\partial g}{\partial n}\right)^T\right]^{-1}$$

- **Estimación del estado actual:** $X_k = X_{kp} + K\left[Y_k - g\left(X_{kp}, 0\right)\right]$
- **Covarianza de la estimación actual:** $P_k = \left[I - K\left(\frac{\partial g}{\partial x}\right)\right] P_{kp}$
- **Creación de un estado anterior para comenzar con la siguiente iteración.**

8.2 Idoneidad y limitaciones

A pesar de que el filtro de Kalman Extendido surgió como una alternativa al filtro de Kalman para sistemas no lineales, pronto aparecieron una serie de problemas derivados, en su mayoría, de la utilización y cálculo de la matriz Jacobiana. Ésta, al manejar términos correspondientes a derivadas parciales, es fuente de errores al provocar singularidades en el proceso de cálculo, por lo que los resultados obtenidos, aunque en apariencia puedan tener una forma o valores correctos, no lo son.

Por otro lado, según [14], el hecho de aproximar las ecuaciones no lineales por unas que sí lo son mediante una aproximación por serie de Taylor induce también muchos errores, ya que en la mayoría de los casos prácticos se desprecian los términos de orden mayor o igual que dos de dicha serie por tomar valores muy pequeños.

Es importante afirmar que el filtro Extendido no es óptimo. No obstante, es implementado en base a un conjunto de aproximaciones. Por ello, las matrices de covarianza no representan realmente las covarianzas de los estados estimados.

Al contrario de lo que ocurre con el filtro de Kalman lineal, el Extendido puede divergir si las sucesivas linealizaciones no son una buena aproximación del modelo lineal a lo largo del dominio.

9. APLICACIÓN DEL FILTRO DE KALMAN EXTENDIDO

Una vez estudiado en qué se basa el filtro de Kalman Extendido, en esta sección se llevará a cabo un experimento real como el realizado en el apartado 7. Aunque los datos manejados serán los mismos para después poder comparar los resultados obtenidos, será necesario realizar una serie de modificaciones a las ecuaciones que modelan el sistema para poder implementar el filtro Extendido. Una vez vistos los cambios necesarios, se inicializarán las matrices necesarias para el funcionamiento del filtro y se analizarán los resultados obtenidos.

9.1 Modelo dinámico

Por tratarse de un movimiento rectilíneo y uniforme, es claro que la ecuación que modela el comportamiento del sistema no presenta términos no lineales. Sin embargo, cuando se describió el modelo lineal, se habló de que existían unas aceleraciones aleatorias que procedían de defectos en la rodadura, viento, etc., que no podían ser medidos e iban a sumarse al sistema como una matriz de ruido constante. En el modelo dinámico no lineal que se define aquí, basado en [15], se modelarán estas aceleraciones aleatorias mediante un modelo de ruido similar al desarrollado para el filtro lineal. Siendo las variables de estado del sistema la posición en los tres ejes coordenados (x, y, z), así como las tres velocidades correspondientes a cada uno de estos ejes, las ecuaciones que definen la posición y velocidad en el movimiento rectilíneo y uniforme del vehículo son:

$$x(k) = x(k-1) + v_x * t\,; \qquad y(k) = y(k-1) + v_y * t\,; \qquad z(k) = z(k-1) + v_z * t\,;$$

$$v_x = v_x\,; \qquad v_y = v_y\,; \qquad v_z = v_z\,;$$

Expresando estas ecuaciones en forma matricial:

$$f(X_{kp}, a_{kp}) = \begin{bmatrix} x(kp) = x(k-1) + v_x * t \\ y(kp) = y(k-1) + v_y * t \\ z(kp) = z(k-1) + v_z * t \\ v_x(kp) = v_x(k-1) \\ v_y(kp) = v_y(k-1) \\ v_z(kp) = v_z(k-1) \end{bmatrix}$$

Para usar este modelo en un filtro Extendido es necesario calcular la matriz Jacobiana. Las derivadas de las ecuaciones del modelo dinámico respecto de las variables de estado son:

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial (x,y,z,\dot{x},\dot{y},\dot{z})} = \begin{bmatrix} 1 & 0 & 0 & t & 0 & 0 \\ 0 & 1 & 0 & 0 & t & 0 \\ 0 & 0 & 1 & 0 & 0 & t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Se puede comprobar que el Jacobiano es constante, por tratarse de un modelo lineal.

9.2 Modelo utilizado en la medida

Para exponer un ejemplo de filtro Kalman Extendido, se partirá de los mismos registros utilizados en el filtrado Kalman lineal, transformando los datos de posición del vehículo para expresarlos en coordenadas polares. Aunque esto no es del todo cierto en nuestro caso, ya que los sensores utilizados no operan en este sistema de coordenadas, existen sensores que sí trabajan en coordenadas polares, como pueden ser el radar o el sonar.

Para adaptar las medidas obtenidas al ejemplo se realizará un cambio de coordenadas al sistema polar antes de introducir las medidas en el filtro. Por tanto, la no linealidad del modelo estará, según [16], en la relación entre la observación y el modelo dinámico.

El vector de la medida tomará la forma:

$$Y_k = \begin{bmatrix} r_k \\ \theta_k \\ z_k \end{bmatrix}$$

Las ecuaciones de la medida para este modelo son, en función de las variables de estado:

$$g(X_k, n_k) = \begin{bmatrix} r_t = \sqrt{x_k^2 + y_k^2 + z_k^2} \\ \theta_k = \tan^{-1}\dfrac{x_k}{y_k} \\ z_k = z_k \end{bmatrix}$$

La alteración del numerador y el denominador del argumento del arco tangente en el cálculo de θ_k se debe a que los ejes de la IMU están cambiados de orden respecto a los ejes cartesianos a los que se refiere la transformación conocida, donde y_k estaría en el numerador y x_k en el denominador.

Omitiendo la dependencia con el tiempo por claridad en la expresión, a pesar de que esta matriz debe ser calculada en cada iteración, las derivadas parciales de g respecto de las variables de estado están calculadas en función de los valores de la predicción actual:

$$\frac{\partial g}{\partial x} = \frac{\partial g}{\partial (x, y, z, \dot{x}, \dot{y}, \dot{z})} = \begin{bmatrix} \dfrac{x}{\sqrt{x^2 + y^2}} & \dfrac{y}{\sqrt{x^2 + y^2}} & 0 & 0 & 0 & 0 \\ \dfrac{y}{x^2 + y^2} & \dfrac{-x}{x^2 + y^2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

9.3 Inicialización de las matrices

La inicialización de las matrices del filtro Extendido será análoga a la presentada en el apartado 7.2 para el caso del filtro lineal. El motivo es buscar la mayor similitud entre ambos experimentos para después poder realizar una comparación de los resultados obtenidos. Por ello, el modelo de ruido aquí presentado también se corresponderá con el del caso lineal, ya que, por la falta de datos para realizar un modelo de ruido, ha sido supuesto un ruido constante, es decir, las matrices W y Z se corresponden con las del filtro lineal.

Mientras que en el caso lineal distinguíamos entre matrices que permanecían constantes durante todo el filtrado y aquellas que eran modificadas en cada iteración, en este caso, al tener definido un modelo de ruido diferente, las matrices Q y R tomarán valores constantes. Las matrices necesarias serán, por tanto:

- **X(0):** Válida únicamente para la primera iteración. Por ser el inicio del movimiento y estar realizado a nivel del mar, todas las variables de estado serán inicializadas a cero.

$$X(0) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- **P(0):** Válida únicamente para la primera iteración, se tomará del mismo valor que en el caso lineal, aunque en el apartado 9.4 se realizará el mismo estudio que ya fue presentado en el apartado 7.4, dándole los mismos valores a esta matriz y comprobando la rapidez de convergencia del filtro según su valor.

$$P(0) = \begin{bmatrix} 7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 \end{bmatrix}$$

- **Q:** Matriz de covarianza que refleja la covarianza del ruido del modelo dinámico. El valor 0.85 en la componente de v_z se ha tomado porque al suponer cero la velocidad proporcionada por el GPS, la curva aparece mal condicionada.

$$Q = 0.85 * \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- **R:** Matriz de la covarianza del ruido de la medida.

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- **W:** Vector de ruido del modelo dinámico que corrige la predicción.

$$W = \begin{bmatrix} 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \end{bmatrix}$$

- **Z:** Vector de ruido de la medida que corrige las desviaciones de sus valores introducidas por los sensores.

$$Z = \begin{bmatrix} 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \\ 0.01 \end{bmatrix}$$

9.4 Análisis de los resultados obtenidos

En esta sección se muestran los resultados gráficos obtenidos tras la implementación del filtro de Kalman Extendido detallado en el Apéndice D en el caso práctico bajo estudio.

Se mostrarán todos los resultados de posición y velocidad para una muestra de datos tomada a 100Hz y otra a 50Hz. Estas muestras serán las mismas que fueron utilizadas en el apartado 7.4 para la implementación del filtro Kalman lineal, con el objetivo de comparar posteriormente los resultados. Después, se realizará un estudio sobre cómo afectan los cambios en la matriz de covarianza inicial del filtro P(0) y, finalmente, se comprobará el efecto que tienen los vectores de ruido sobre los resultados.

9.4.1 Entrada de datos a 100Hz en tramo 1

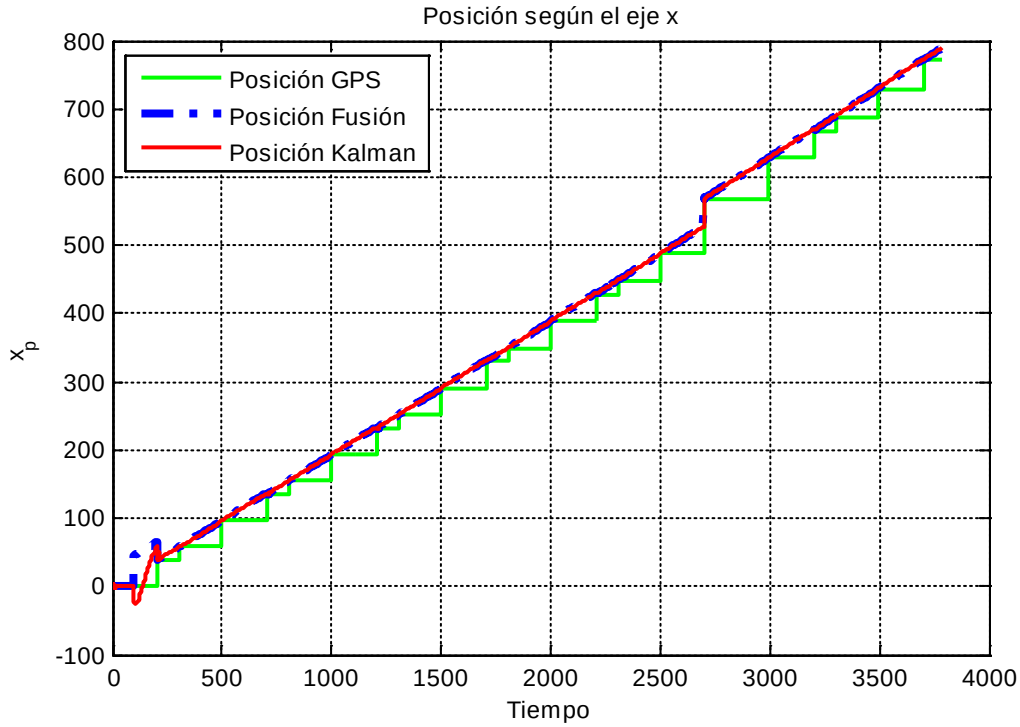

Figura 9.1: Posición en el eje x – filtro extendido – 100Hz

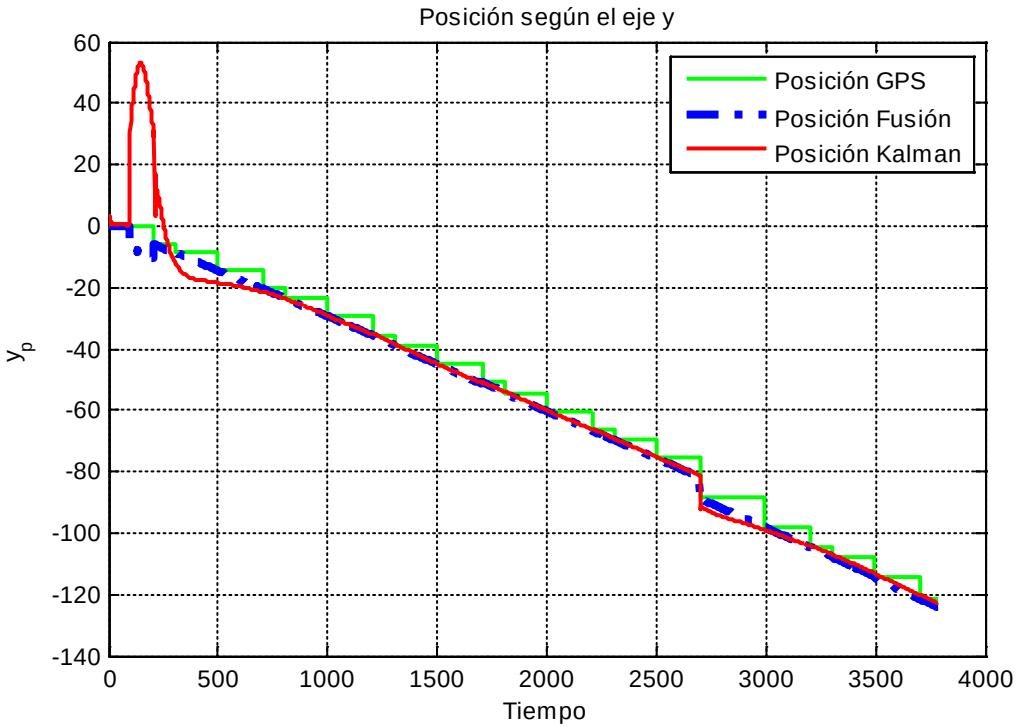

Figura 9.2: Posición en el eje y – filtro extendido – 100Hz

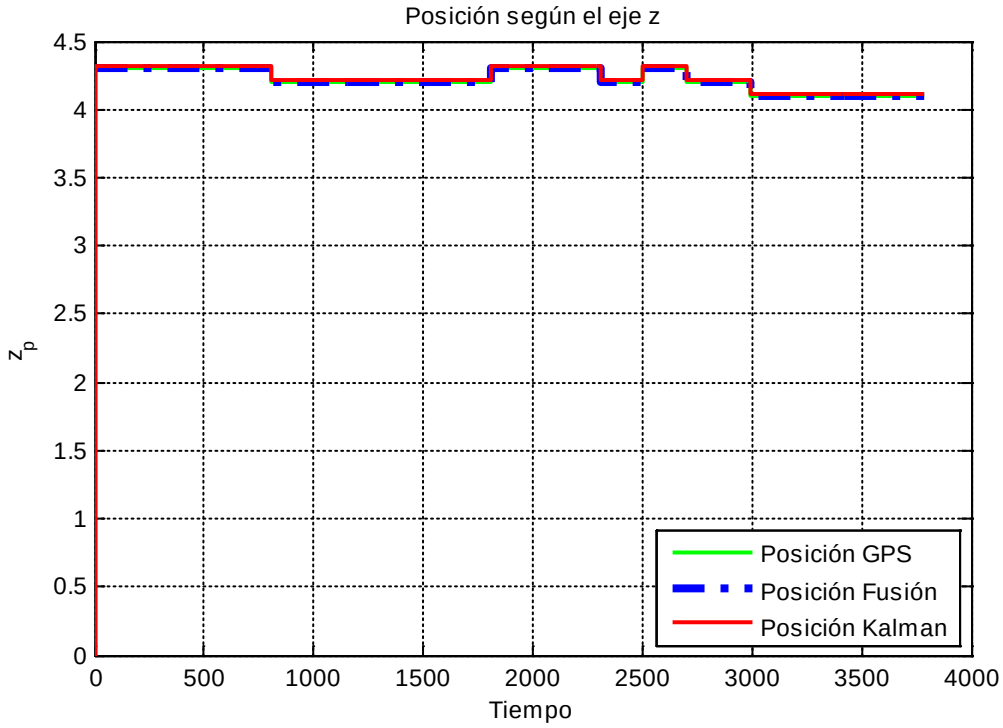

Figura 9.3: Posición en el eje z – filtro extendido – 100Hz

Aunque se estaba a la espera de unos malos resultados o de que incluso la estimación pudiese divergir, podemos comprobar que las estimaciones de la posición se encuentran en el entorno de la medida, salvo por pequeñas oscilaciones al inicio.

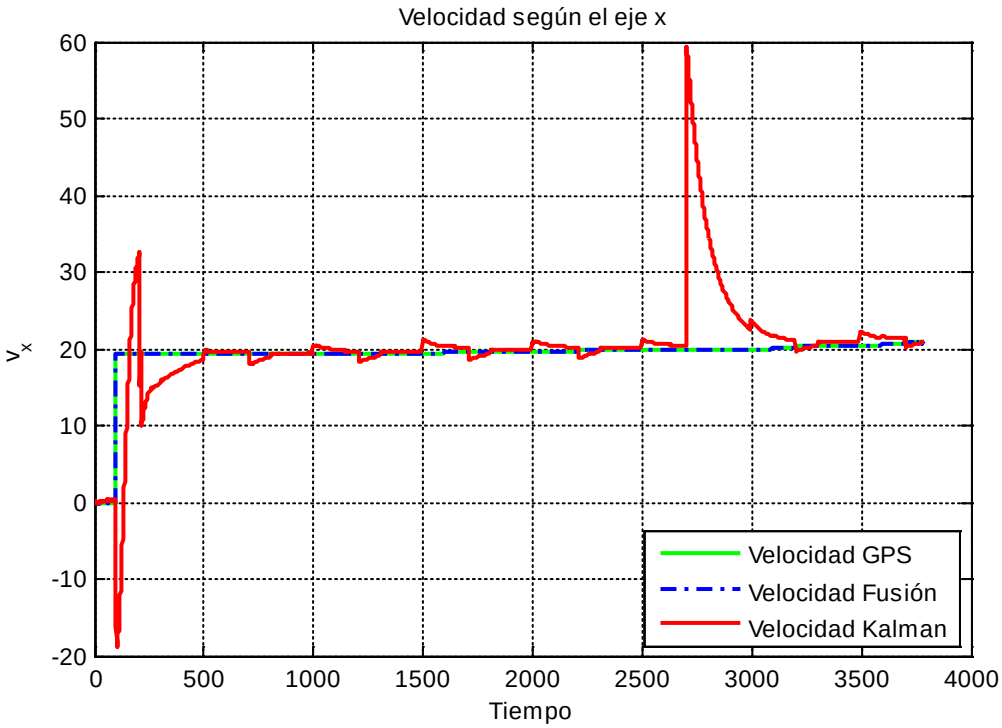

Figura 9.4: Velocidad en el eje x – filtro extendido – 100Hz

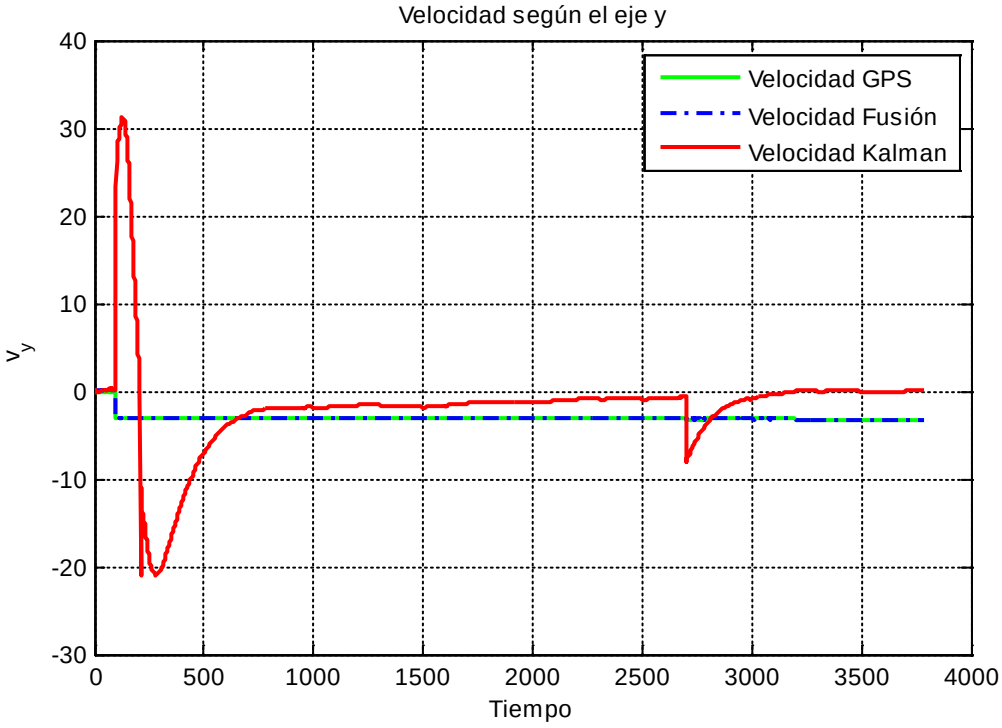

Figura 9.5: Velocidad en el eje y – filtro extendido – 100Hz

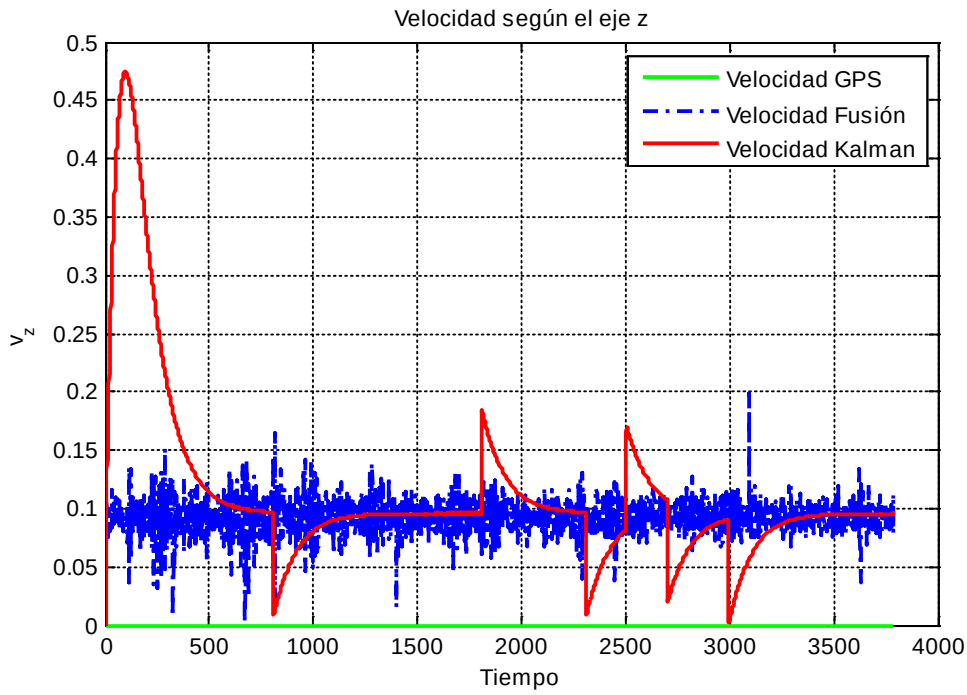

Figura 9.6: Velocidad en el eje z – filtro extendido – 100Hz

En el caso de la velocidad los resultados no son tan favorables, mostrando mucha inestabilidad, presumiblemente por desajustes en las muestras tomadas, que hacen la estimación del filtro poco fiable, aunque su evolución tienda a acercarse a los valores medidos.

9.4.2 Entrada de datos a 50 Hz en tramo 2

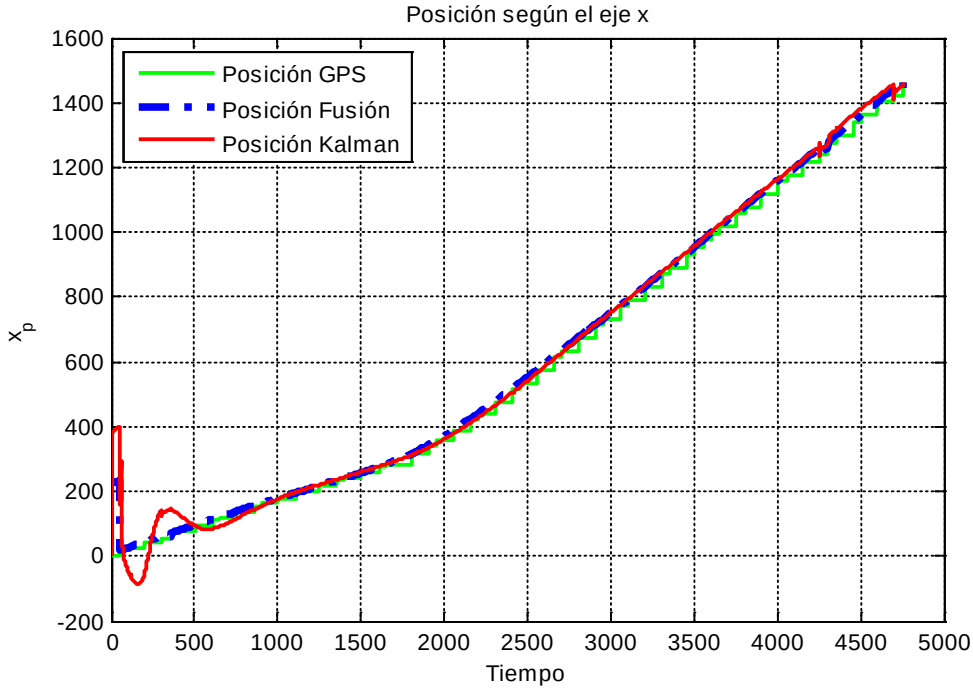

Figura 9.7: Posición en el eje x – filtro extendido – 50Hz

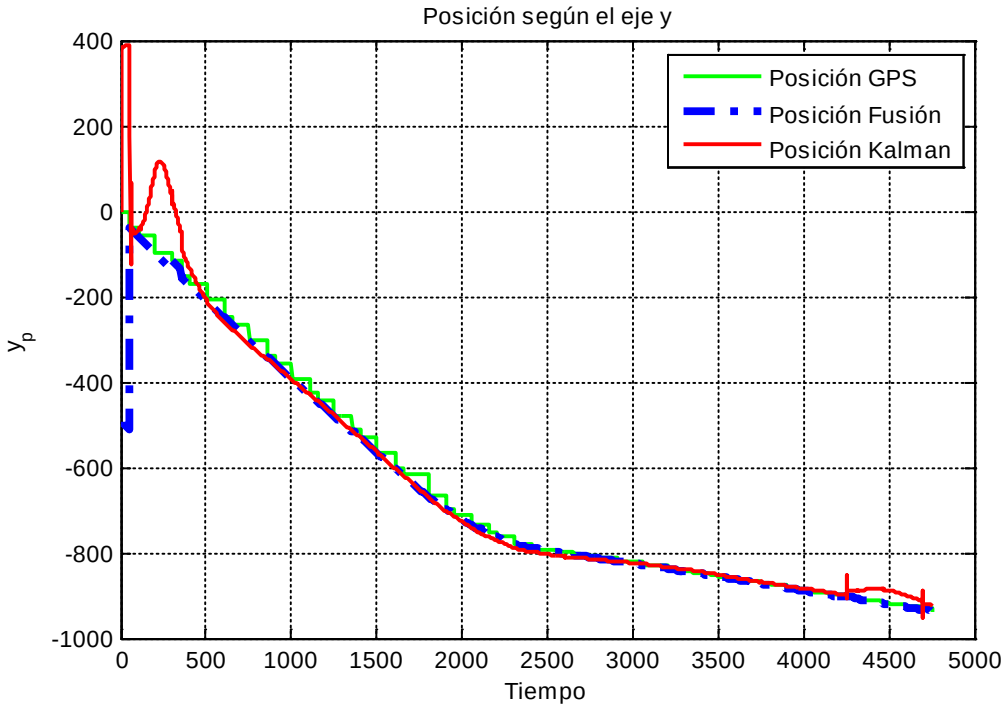

Figura 9.8: Posición en el eje y – filtro extendido – 50Hz

Observando los resultados, las muestras tomadas a 50Hz no ralentizan demasiado la estabilización de la estimación, la cual se ajusta de una manera más o menos correcta a las medidas. La componente y, al ser la que se encuentra afectada por el ángulo θ, presenta mayor inestabilidad. La fusión presenta también grandes oscilaciones al principio provocadas por errores en la recogida de los datos.

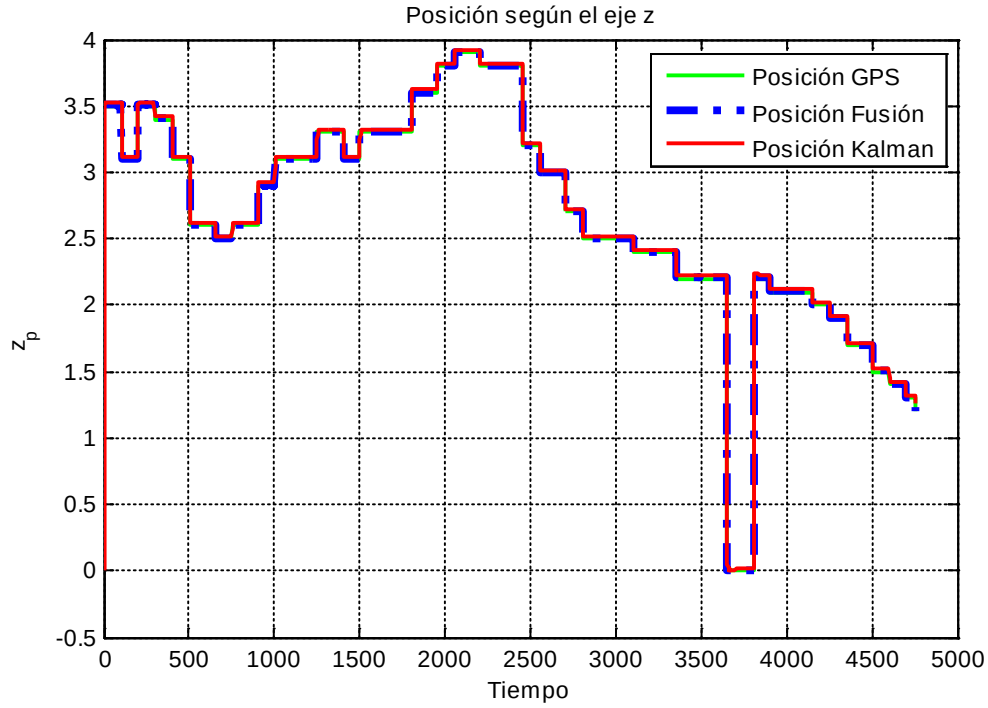

Figura 9.9: Posición en el eje z – filtro extendido – 50Hz

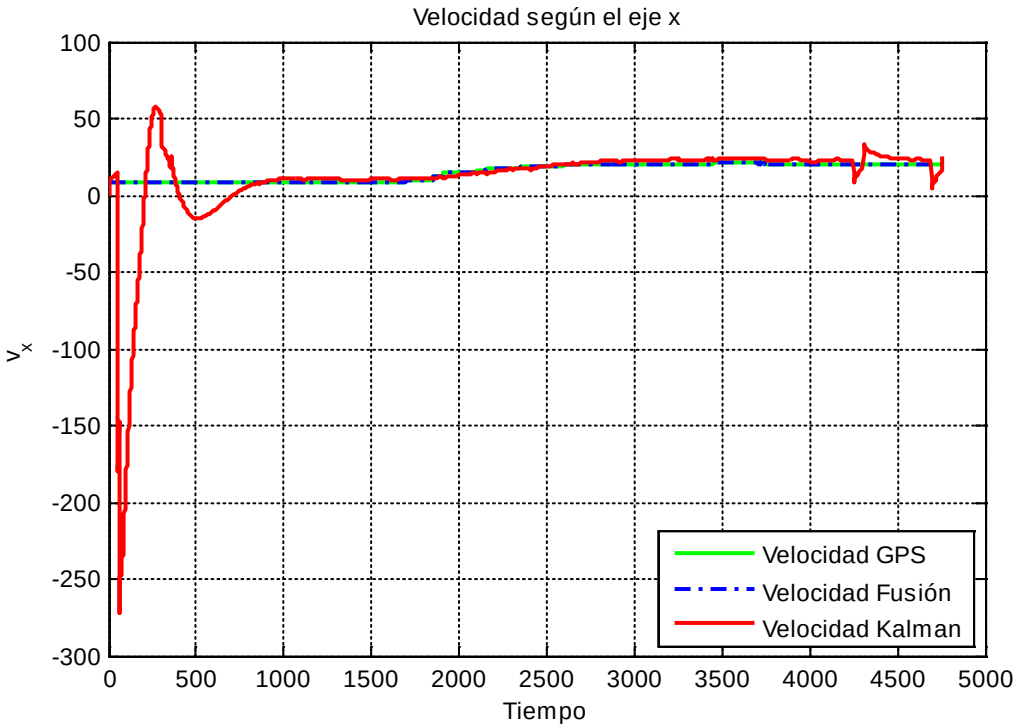

Figura 9.10: Velocidad en el eje x – filtro extendido – 50Hz

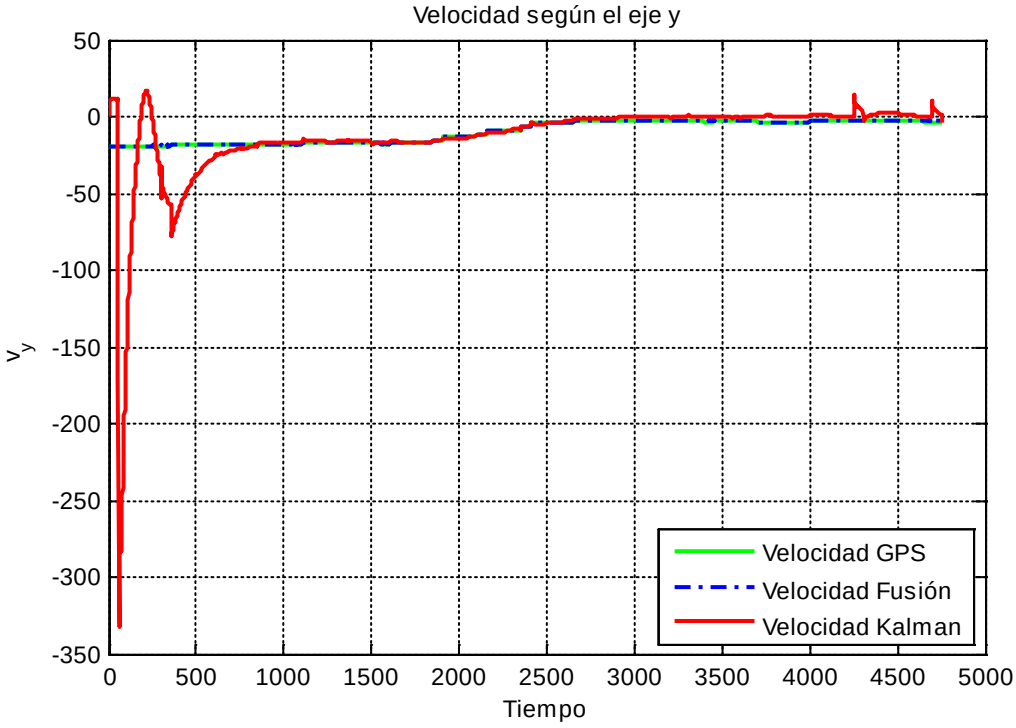

Figura 9.11: Velocidad en el eje y – filtro extendido – 50Hz

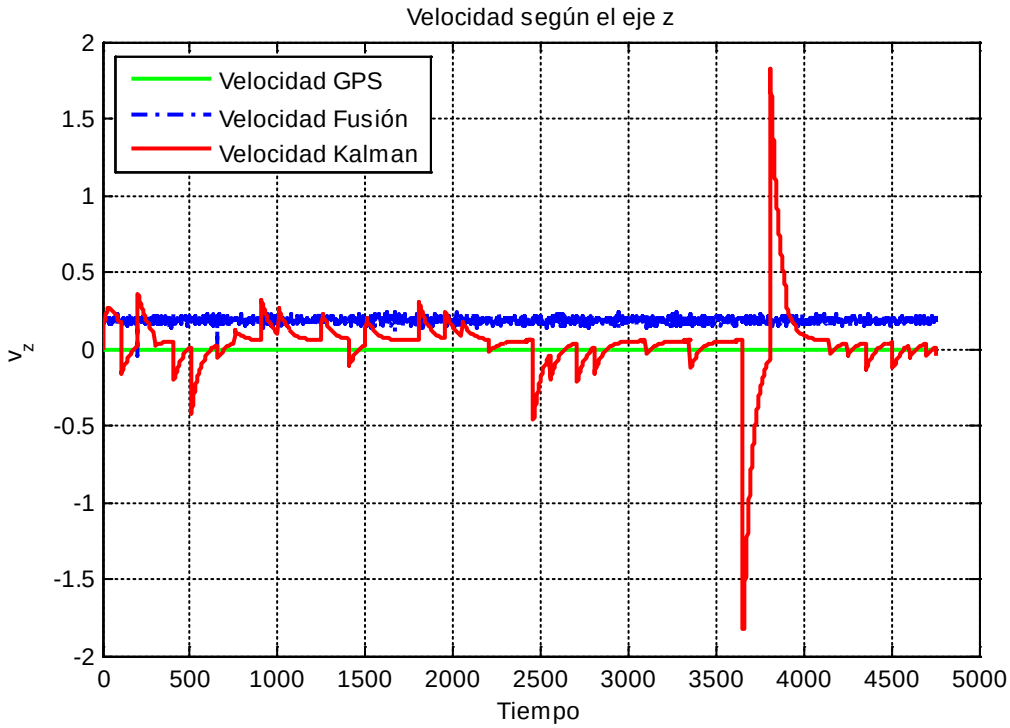

Figura 9.12: Velocidad en el eje z – filtro extendido – 50Hz

Las velocidades, como en el caso de muestras a 100 Hz, dan peores resultados que las posiciones. Sin embargo, en este caso, a pesar de lo que pueda parecer en las gráficas, las escalan indican que las estimaciones difieren más de las medidas y con valores más grandes. Esto es porque se dispone de la mitad de información entre dos entradas de GPS, con lo que es más difícil interpolar el comportamiento del filtro entre ellas.

Es importante remarcar aquí que, como se predijo, los resultados obtenidos con el filtro Extendido son peores que los del filtro lineal. Esto es porque el cálculo del Jacobiano y las aproximaciones tomadas tanto en la linealización como en las matrices de covarianza provocan la pérdida de mucha información.

9.4.3 Inicialización con distintas matrices de covarianza iniciales

Se estudiará el comportamiento del filtro con una matriz de covarianza inicial diagonal de pequeño tamaño, como se definió en el apartado 9.3, para después estudiar el comportamiento tras realizar la inicialización con cinco matrices más. Estas matrices serán las mismas que las utilizadas en el estudio realizado en el apartado 7.4.3 para el filtrado lineal.

Por mayor claridad en el tiempo de convergencia de las matrices se realizará la implementación del filtro para los datos obtenidos a una frecuencia de 50Hz. Por otro lado, se ha decidido realizar el experimento con una única variable de estado, siendo esta representativa del comportamiento del filtro con el resto de variables. A la vista de los resultados obtenidos en el apartado 9.4.2, por las oscilaciones que presenta al comienzo del filtrado, se ha decidido que esa variable sea la posición según el eje x.

- **Matriz $P_2(0)$:**

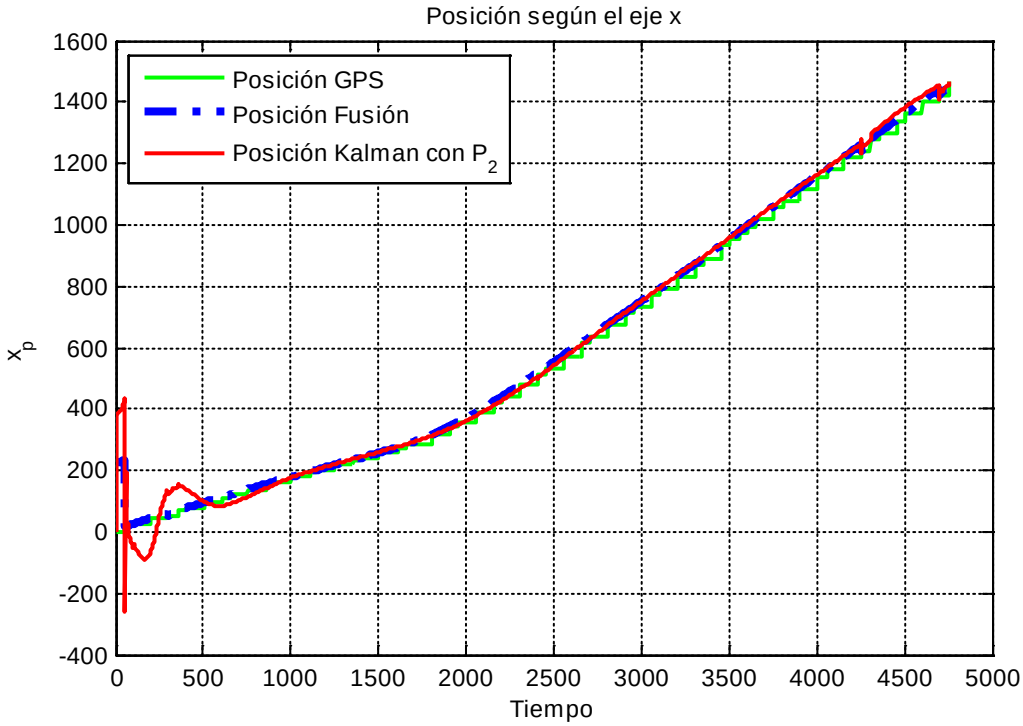

Figura 9.13: Posición en el eje x – P_2 – 50Hz

- **Matriz $P_3(0)$:**

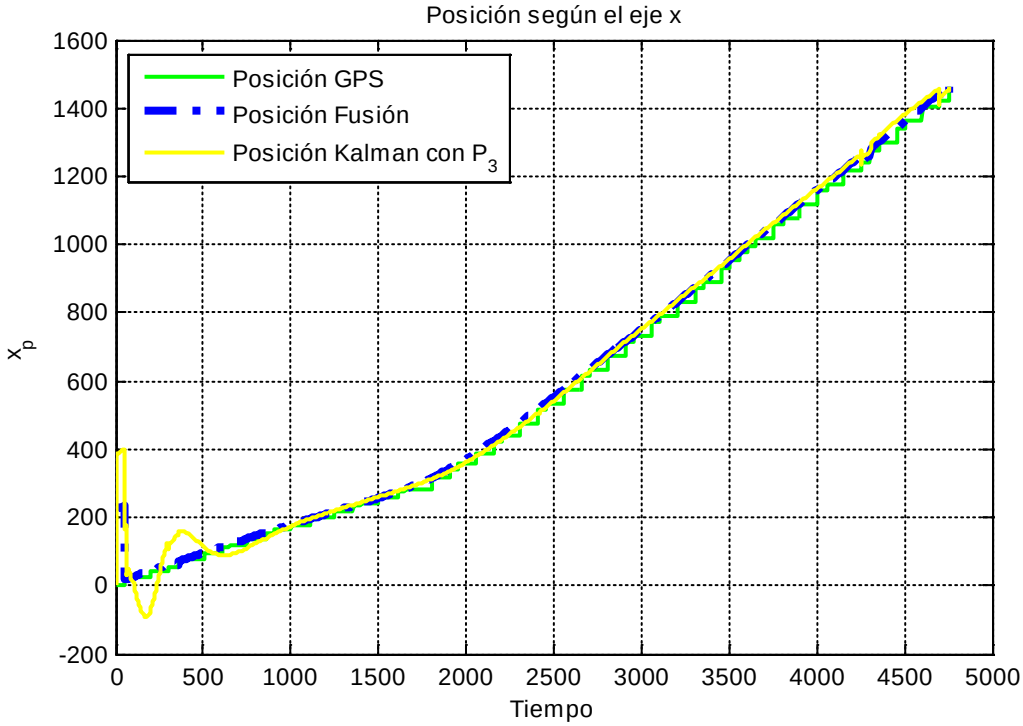

Figura 9.14: Posición en el eje x – P_3 – 50Hz

- **Matriz $\boldsymbol{P_4(0)}$:**

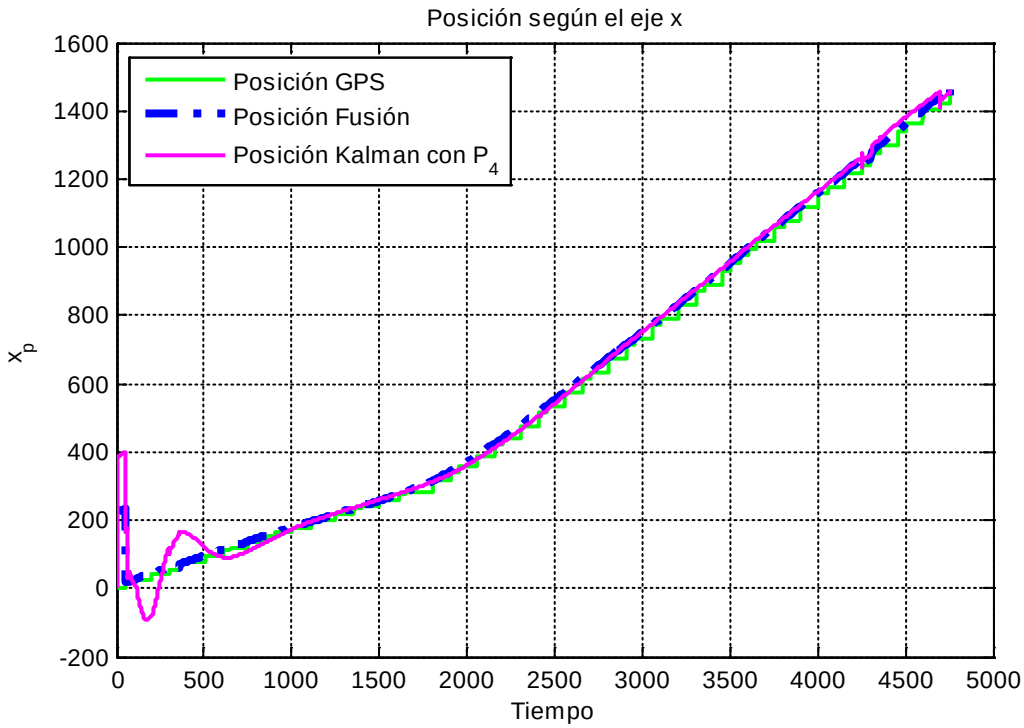

Figura 9.15: Posición en el eje x – P_4 – 50Hz

- **Matriz $\boldsymbol{P_5(0)}$:**

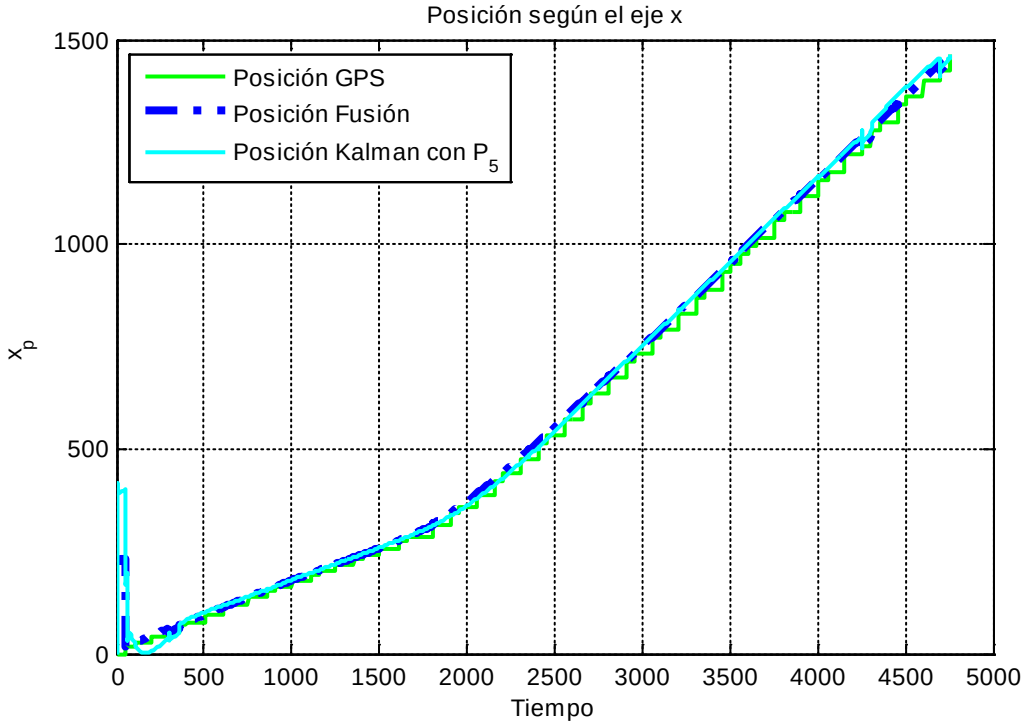

Figura 9.16: Posición en el eje x – P_5 – 50Hz

- **Matriz $\boldsymbol{P_6(0)}$:**

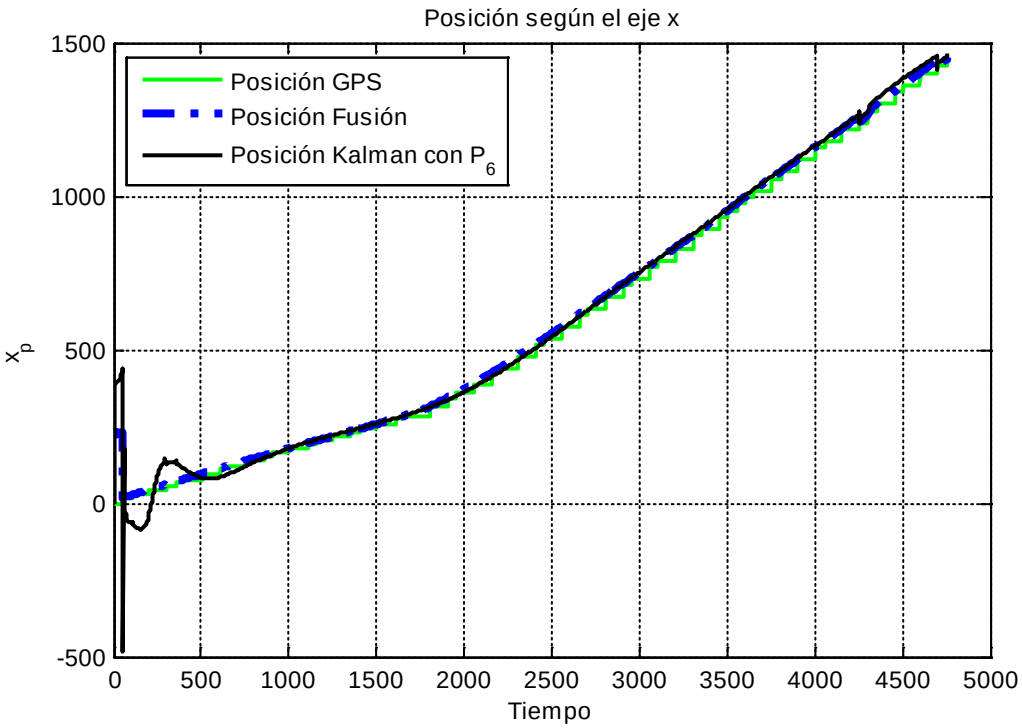

Figura 9.17: Posición en el eje x – P_6 – 50Hz

Como se observa en las gráficas, en este caso el valor de la matriz de covarianza inicial casi no afecta a la rapidez con la que el filtro se establece en el entorno de la medida. Los resultados son bastante parecidos entre las distintas matrices que han sido probadas, llegando a concluir que es mejor una matriz de covarianza inicial con valores pequeños, porque minimiza la oscilación que se produce al comienzo de la estimación.

Las matriz con covarianza negativa y valores elevados es en este caso la que peor resultado da, como se puede apreciar en la figura de arriba de este párrafo. Paradójicamente, es esa misma matriz completa de covarianzas negativas para valores pequeños la que proporciona mejores resultados (matriz P_5). No se demuestra aquí por tanto que el hecho de que la matriz de covarianza inicial sea diagonal beneficie a la estabilización de la estimación.

9.4.4 Efecto de los vectores de ruido W y Z

Se realizará en este apartado un estudio de las implicaciones que tienen los vectores de ruido W y Z en la solución del filtrado extendido, análogo al ya realizado en el apartado 7.4.4 para el caso de filtrado lineal.

Se ha tomado como variable de estado la posición según el eje x. Se observará el efecto de la modificación del valor de los vectores por separado, y a continuación el efecto de un aumento en ambos. La matriz $P(0) = P_1$ y se filtrarán los datos obtenidos a 50Hz. Siendo los nuevos valores los ya utilizados en el caso lineal:

$$W = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}; \qquad Z = \begin{bmatrix} 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \end{bmatrix}$$

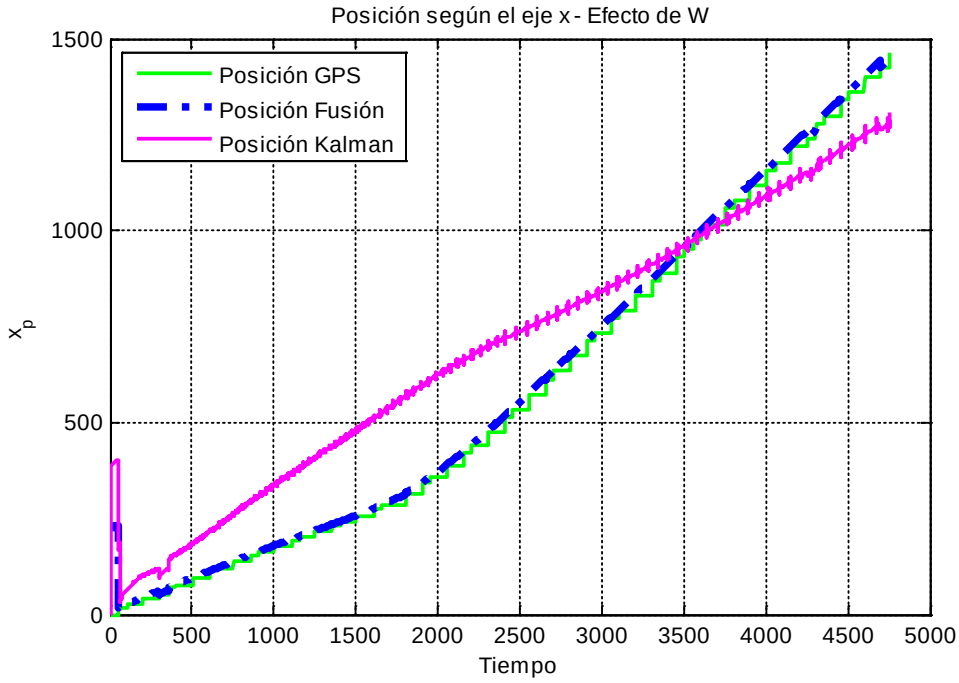

Figura 9.18: Posición en el eje x – Efecto de W – 50Hz

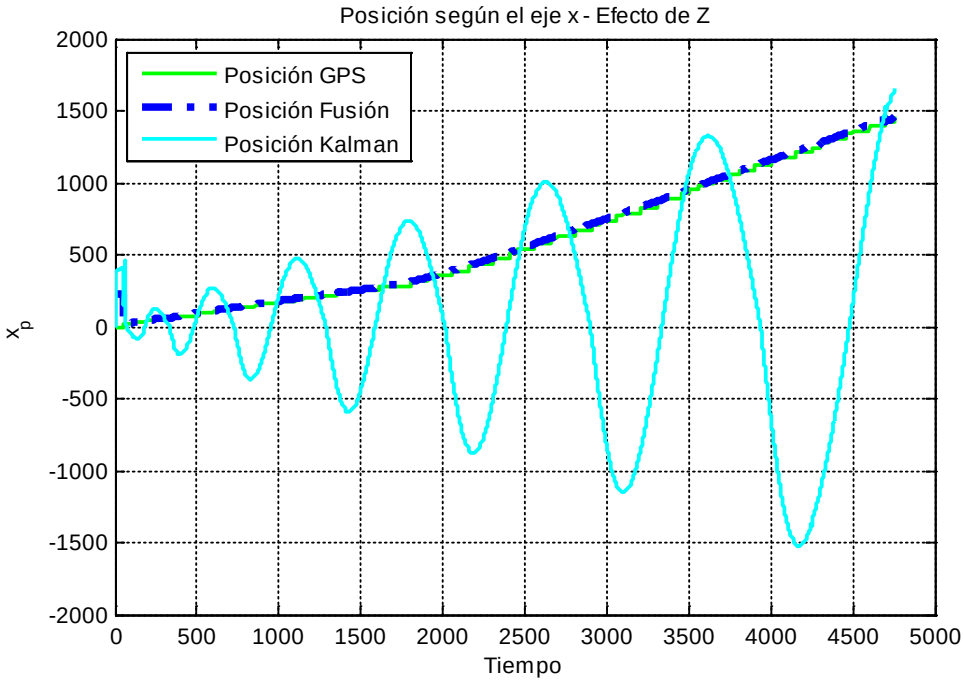

Figura 9.19: Posición en el eje x – Efecto de Z – 50Hz

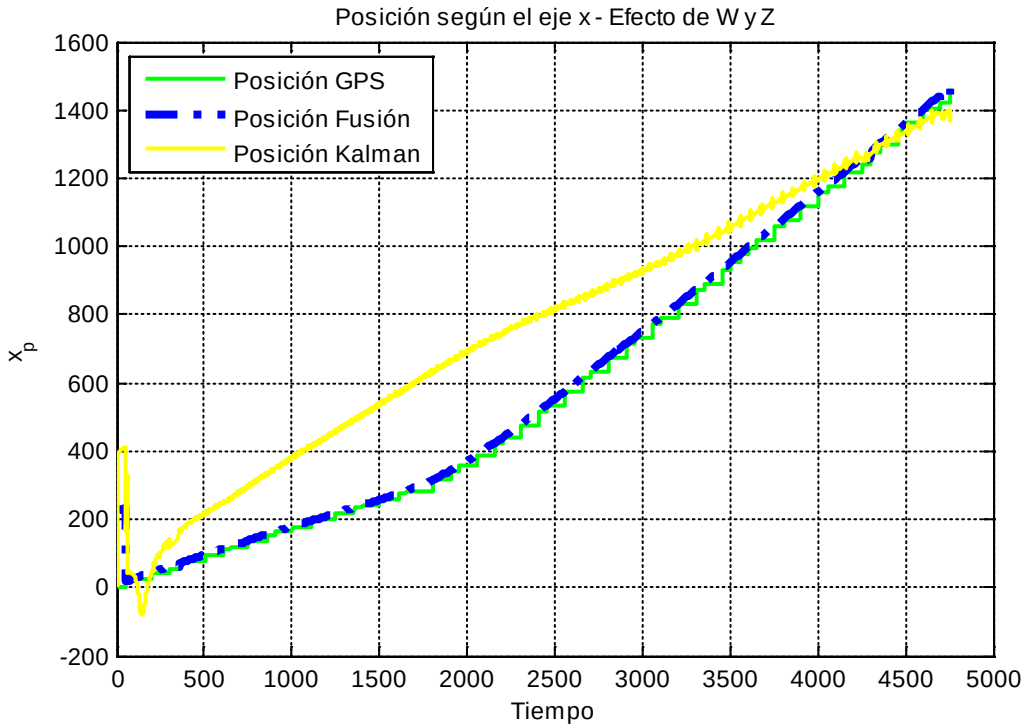

Figura 9.20: Posición en el eje x – Efecto de W y Z – 50Hz

Las matrices de ruido sí tienen, como se puede observar en las figuras, una gran importancia en la implementación de un filtro Extendido. Esto es debido a que el modelo de ruido adoptado para el problema que se esté estudiando se ve afectado por el cálculo de su Jacobiano, así como también es aproximado al caso lineal por una serie de Taylor. Todos estos factores hacen que cualquier variación de sus valores provoque una reacción del filtro a peor.

La susceptibilidad del filtro extendido se ve aumentada en el caso de que el ruido únicamente afecte a la medida (Z). El filtro no es capaz de controlar esa desviación y diverge de un modo no amortiguado.

Cuando el ruido afecta a la predicción (W), o a una combinación de ambos, la estimación tampoco es fiable debido a que no sigue en ningún momento lo que dictan las muestras obtenidas. Por ello, es muy importante a la hora de implementar un filtro Extendido que no solo el modelo sea acertado y la linealización lo más precisa posible, sino que el modelo de ruido con el que se trabaje sea también una buena estimación de la realidad, para evitar que el filtro no sea capaz de decidir a quién hacer más caso, si a unas medidas erróneas y con desviaciones o a un modelo que tras la linealización dista bastante de la realidad que se quiere modelar.

10. COMPARATIVA DE LOS RESULTADOS OBTENIDOS

En esta sección se realizará una comparación entre los resultados obtenidos mediante la implementación del Filtro de Kalman lineal y los proporcionados por la función que ofrece el programa Matlab, que realiza un filtrado Kalman equivalente a este. El código utilizado aparece descrito en el Apéndice E. La utilización, por tanto, de esta función predeterminada es posible debido a que el modelo del movimiento del experimento es lineal.

En base a los estudios realizados en el apartado 7.4, se ha optado por utilizar aquí unas matrices de ruido W y Z de pequeño valor, debido a la imposibilidad de realizar en este caso un modelo acertado de ruido.

Las matrices de covarianza Q y R serán las correspondientes a cada filtro, es decir, el filtro lineal funcionará con unas matrices de la covarianza que cambiarán en cada iteración, mientras que el filtro realizado mediante la función operará con matrices de covarianza constantes durante todo el filtrado, e iguales a matrices identidades de dimensión 6x6.

Por otro lado, dado que se comprobó que la convergencia del filtro era más rápida en el caso de que la matriz P(0) fuese diagonal y tomara valores muy grandes, será la que se utilice en esta comparación, buscando así optimizar el filtrado.

Por último, especificar que esta comparativa ha sido realizada en base a las muestras obtenidas cada 100Hz, dado que esta es una frecuencia normal a la que suele proporcionar valores la IMU. Se ha eliminado la representación de la componente de la medida obtenida del GPS por claridad en las figuras.

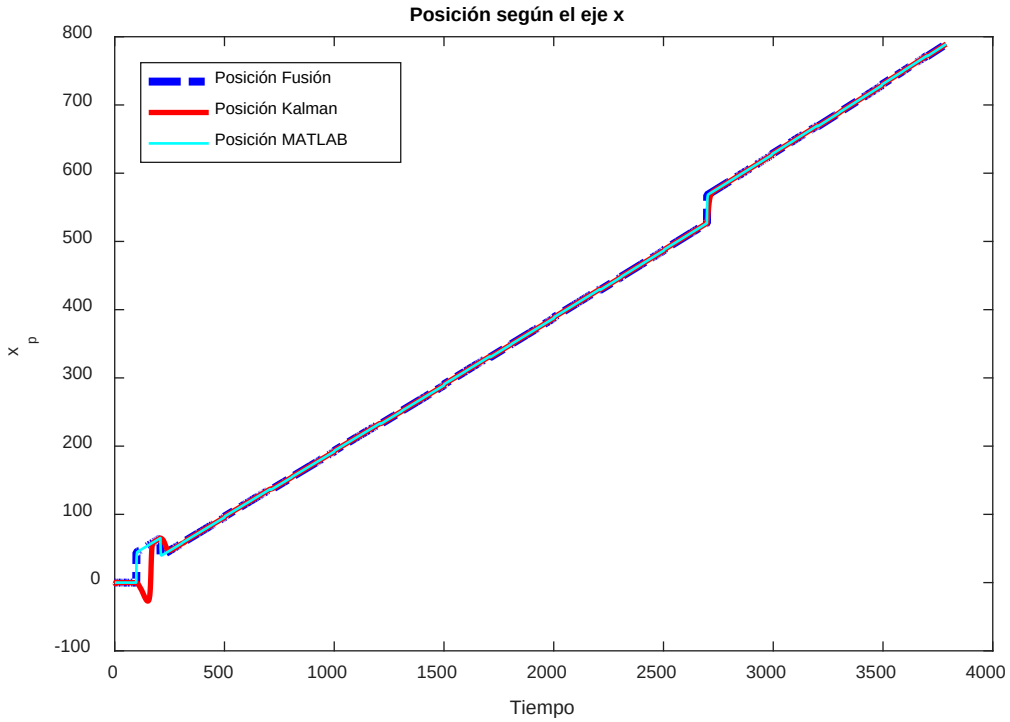

Figura 10.1: Comparativa de la posición en dirección x

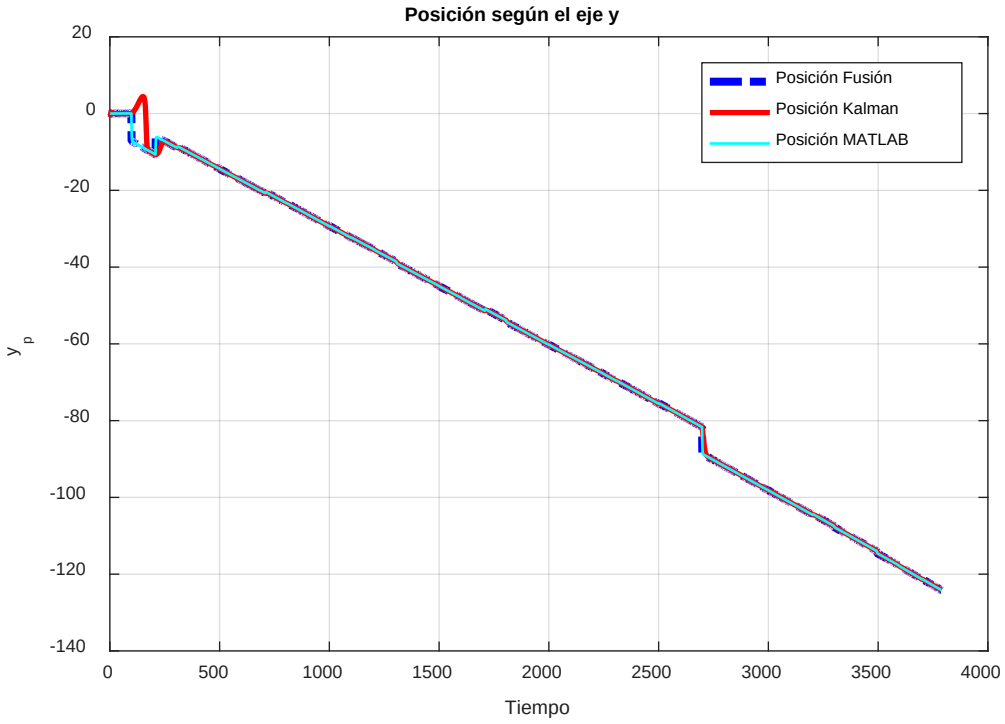

Figura 10.2: Comparativa de la posición en dirección y

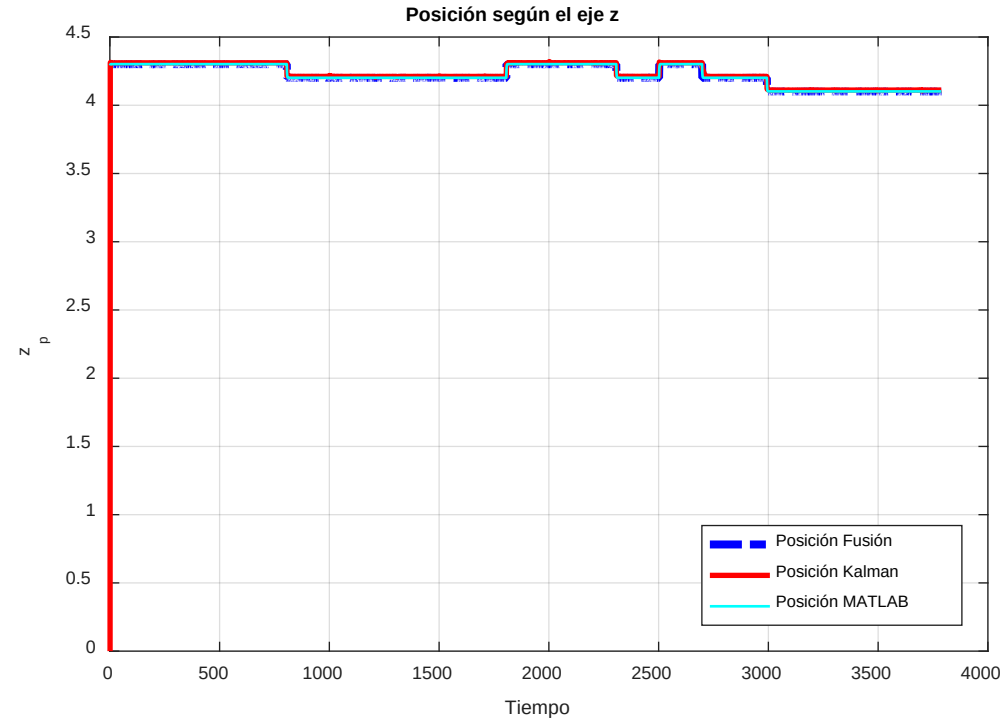

Figura 10.3: Comparativa de la posición en dirección z

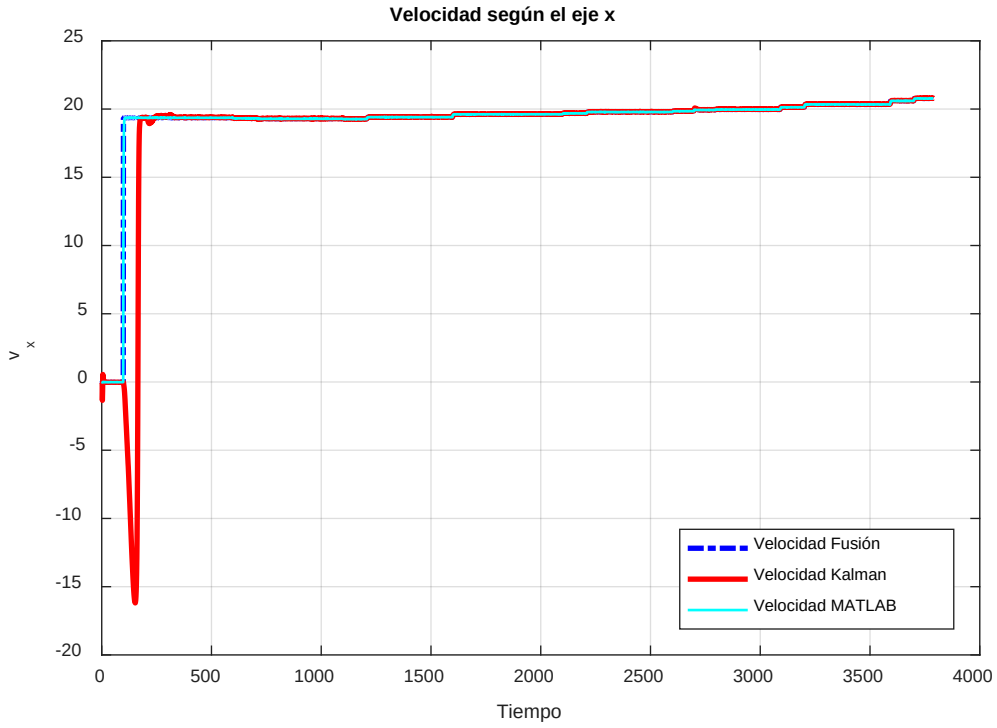

Figura 10.4: Comparativa de la velocidad en dirección x

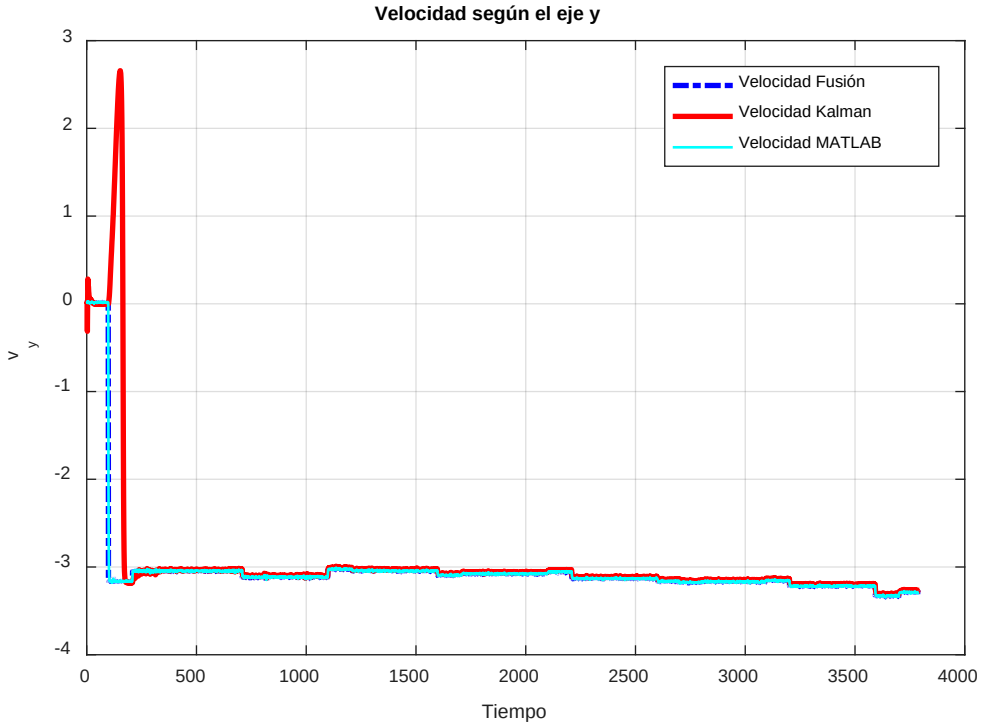

Figura 10.5: Comparativa de la velocidad en dirección y

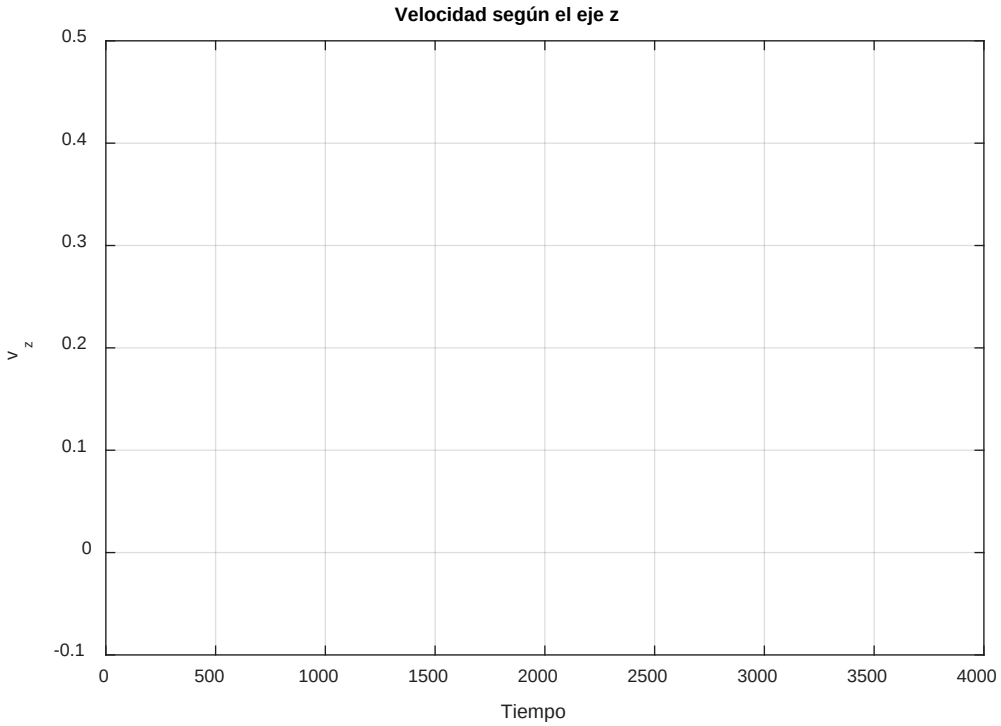

Figura 10.6: Comparativa de la velocidad en dirección z

Como se puede observar en las seis representaciones, la función de Matlab sigue a los valores obtenidos de las medidas tras la fusión de datos. Esto tiene sentido, porque como se comentó, tanto los sensores como el modelo tienen un error muy bajo. El filtrado simplemente elimina el ruido blanco de la muestra que posee, y no añade ninguna desviación adicional. Por otro lado, el modelo realizado en base al movimiento rectilíneo y uniforme constituye una representación bastante fiel de la realidad. Se verá más adelante la complejidad que tendría modelar el comportamiento de un UAV en el aire.

El filtrado realizado por el programa elaborado en esta memoria se adapta bastante bien al proporcionado por el programa Matlab, por lo que se puede admitir esta prueba como comprobación de la fiabilidad de la función desarrollada, detallada en el Apéndice C.

Las oscilaciones al comienzo de la estimación en las gráficas se deben a la inicialización realizada de las matrices del filtro X(0) y P(0). Un estudio previo del comportamiento del sistema que va a ser ensayado contribuiría a minimizar el error de aproximación y facilitar la rápida convergencia del filtro. Por ejemplo, en este caso, se conocía previamente que la altura de la carretera por la que circulaba el vehículo se encontraba a una altura constante, pequeña sobre el nivel del mar, por lo que podría haberse ajustado el valor de la variable de estado z al entorno de 5m en el vector inicial de estado.

Para terminar esta comparación, destacar que, aunque puede parecer a simple vista que la componente z de la velocidad difiere mucho tanto de la medida como del resultado proporcionado por Matlab, nada más lejos de la realidad. Es preciso prestar atención a la escala para percibir que el error es de menos de 0.1m/s, que incluso está dentro del rango de error que poseen los sensores.

11. OTROS FILTROS

En esta sección se analizará una extensión del filtro de Kalman como alternativa al filtro de Kalman Extendido utilizado para sistemas no lineales. Se trata del filtro de Kalman Unscented, denominado en forma abreviada como UKF por sus siglas en inglés, Unscented Kalman Filter. Se expondrán los fundamentos de esta extensión y se comparará con el Kalman Extendido, describiendo como sería la implementación de la transformación Unscented para un filtrado en coordenadas polares, que es el mismo sistema que el utilizado en este trabajo.

Además, para complementar el estudio del filtrado Kalman, se dará otra visión del mismo bajo la perspectiva de su condición de filtro adaptativo. [14], [17] y [18].

11.1. La transformación Unscented.

El filtrado UKF está basado en la transformación Unscented, que permite calcular los primeros momentos de la densidad de distribución de probabilidad de una variable aleatoria $y = f(x)$ que resulta de aplicar una transformación no-lineal f a una variable aleatoria x de estadística conocida.

El método se basa en la idea intuitiva de que resulta más fácil aproximar una distribución gaussiana que aproximar una transformación no-lineal arbitraria.

Un conjunto de puntos, llamados puntos sigma, se eligen de forma tal que la media y la covarianza del conjunto de puntos coincida con la esperanza y covarianza de la variable aleatoria x. Luego, la transformación no-lineal f se aplica a cada punto sigma, para generar una nube de puntos transformados. De la estadística de los puntos transformados, se estima la esperanza y covarianza de la variable aleatoria $y = f(x)$.

Si bien el método es similar a los métodos de muestreo Monte Carlo, existe una diferencia fundamental que es que las muestras no se toman al azar, sino de acuerdo a un algoritmo determinista. De esta forma se logra capturar información de alto orden de la distribución de probabilidad, tomando sólo un número pequeño de puntos.

Aclarar aquí que los métodos de Monte Carlo son una amplia clase de algoritmos computacionales que se basan en repetidos muestreos aleatorios para obtener resultados numéricos. Su idea esencial es usar la aleatoriedad para resolver problemas que podrían ser deterministas en principio. A menudo, se utilizan en problemas físicos y matemáticos y son más útiles cuando es difícil o imposible utilizar otros enfoques. Los métodos de Monte Carlo se utilizan principalmente en problemas de optimización, integración numérica y generación de sorteos a partir de una distribución de probabilidad.

Para ver el fundamento de la transformación Unscented, supóngase que la variable aleatoria vectorial x, de dimensión L, tiene esperanza $\mathbb{E}[x] = \bar{x}$ (vector columna de L componentes) y covarianza $\mathbb{C}ov\ (x, x) = P_x$ (matriz LxL). Para calcular, aproximadamente, la esperanza $\mathbb{E}[y] = \bar{y}$ y la covarianza $\mathbb{C}ov\ (y, y) = P_y$ de la variable aleatoria $y = f(x)$, se

comienza construyendo una matriz $\mathcal{X}$ de $2L+1$ vectores columna $\mathcal{X}_i$ de acuerdo con lo siguiente:

$$\mathcal{X}_0 = \bar{x}$$

$$\mathcal{X}_i = \bar{x} + \left(\sqrt{(L+\lambda)\,P_x}\,\right)_i \qquad i = 1, \ldots, L$$

$$\mathcal{X}_i = \bar{x} - \left(\sqrt{(L+\lambda)\,P_x}\,\right)_{i-L} \qquad i = L+1, \ldots, 2L$$

Donde $\lambda = \alpha^2\ (L+\kappa) - \beta$ es un parámetro de escala. La constante α determina la dispersión de los puntos sigma alrededor de $\bar{x}$ y, habitualmente, se elige igual a una pequeña constante positiva, del orden de 1 x $10^{-4} \leq \alpha \leq 1$. La constante κ es un parámetro de escala secundario que habitualmente se toma igual a 0 o a $3 - L$ y β, es usada para incorporar el conocimiento que se tenga de antemano acerca de la distribución de x, siendo $\beta = 2$ el óptimo para distribuciones gaussianas. $\left(\sqrt{(L+\lambda)\,P_x}\,\right)_i$ es la i-ésima columna de la matriz raíz cuadrada de $(L+\lambda)\,P_x$.

El método genera estimaciones de $\bar{y}$ y P_y correctas hasta el tercer orden de la esperanza y covarianza del desarrollo multi-variable de Taylor de la función f, supuesta analítica en todo valor posible de x, en torno a $\bar{x}$, cuando la distribución de x es simétrica. Para distribuciones no simétricas, genera estimaciones correctas hasta el segundo orden, dependiendo la exactitud de los momentos de tercer y mayor orden, y de la elección de los parámetros α y β.

A continuación, se presenta un ejemplo para clarificar la transformación Unscented:

Consideremos un vehículo sumergible que detecta obstáculos mediante el uso de un sonar. Los datos de posición del sonar son obtenidos en coordenadas polares, es decir distancia ρ y ángulo φ, respecto de un sistema de referencia solidario a él. Las coordenadas polares $\mathrm{X} = \begin{bmatrix} \rho \\ \varphi \end{bmatrix}$ deben de ser convertidas a coordenadas cartesianas $\mathrm{Y} = \begin{bmatrix} x \\ y \end{bmatrix}$ antes de ser procesadas. El sónar tiene una buena precisión en la medida de distancias (σ_ρ= 2 cm de desviación estándar), a cambio de una mala precisión en la medida angular (σ_φ= 15º de desviación estándar).

Supóngase que la ubicación real de un punto donde se encuentra un obstáculo es $X_0 = \begin{bmatrix} \rho_0 \\ \varphi_0 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{\pi}{2} \end{bmatrix}$, entonces $\mathrm{Y} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = f\left(\begin{bmatrix} \rho_0 \\ \varphi_0 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Para apreciar los errores que pueden ser introducidos por linealización, en la estimación de los dos primeros momentos $\mathbb{E}[y]$ y $\mathbb{C}ov\ (y, y)$ de la variable vectorial aleatoria $\mathrm{Y} = \begin{bmatrix} x \\ y \end{bmatrix}$ resultante de la transformación $\mathrm{X} = \begin{bmatrix} \rho \\ \varphi \end{bmatrix}$, se estiman estos momentos mediante tres métodos distintos:

- **Muestreo Monte Carlo:** Genera un conjunto de N= 10^6 puntos aleatorios instanciados de una densidad de distribución gaussiana con media $\mathbb{E}[x] = X_0 = \begin{bmatrix} \rho_0 \\ \varphi_0 \end{bmatrix}$ y covarianza $\mathbb{C}ov\ (X, X)$ = $\begin{bmatrix} {\sigma_\rho}^2 & 0 \\ 0 & {\sigma_\varphi}^2 \end{bmatrix}$. Luego, se transforman los puntos X_i para generar la nube de puntos transformados, Y_i, y se estiman sus momentos $\mathbb{E}[y]$ y $\mathbb{C}ov\ (y, y)$.

$$\mathbb{E}[y] \approx \bar{Y}^{(MC)} \triangleq \frac{1}{N}\sum_{i=1}^{N} X_i,$$

$$\mathbb{C}ov\,(y,y) \approx \mathrm{P_y}^{(MC)} \triangleq \frac{1}{N}\sum_{i=1}^{N}(Y_i - \bar{Y}^{(MC)})(Y_i - \bar{Y}^{(MC)})^T$$

- **Transformación Unscented:** Genera un conjunto de puntos sigma (en este ejemplo cinco), los cuales se transforman a través del cambio de coordenadas y se estiman los momentos $\mathbb{E}[y] \approx \bar{Y}^{(UT)}$ y $\mathbb{C}ov\,(y,y) \approx \mathrm{P_y}^{(UT)}$. Se eligieron los siguientes valores para los parámetros de transformación: $\alpha = 1, \kappa = 0$ y $\beta = 2$.

- **Linealización análoga a la aplicada por el filtro Extendido:**

$$\mathbb{E}[y] \approx \bar{Y}^{(LIN)} = Y_0 = f\,(\mathcal{X}_0)$$

$$\mathbb{C}ov\,(y,y) \approx \mathrm{P_y}^{(LIN)} = \Im f\,(\mathcal{X}_0)\,P_x\,[(\Im f\,(\mathcal{X}_0))^T]$$

Donde $\Im f\,(\mathcal{X}_0)$ es el jacobiano de la transformación.

En la Figura 11.1 se reproducen los puntos del muestro de Monte Carlo junto con los cinco puntos sigma en el plano (ρ, φ) a la izquierda y su transformación al plano (x, y) a la derecha.

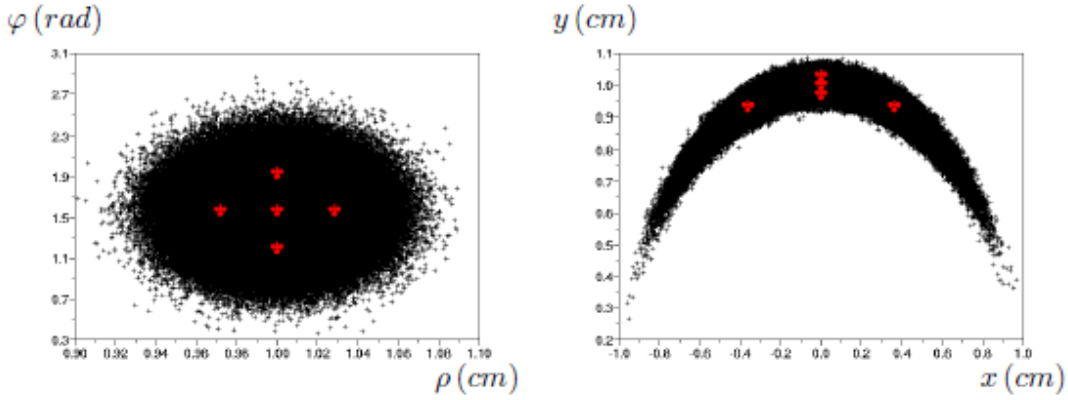

Figura 11.1: Puntos en el plano de la medida y en el transformado

En la Figura 11.2 se representa el plano (x, y) con las estimaciones de $\overline{\mathrm{Y}}$ (esperanza $\mathbb{E}[\mathrm{y}]$), indicadas por puntos y los contornos 1σ , que se definen como el lugar geométrico donde $\{y : (Y - \bar{Y})^T\,\mathrm{P_y}\,(Y - \bar{Y}) = 1\}$, que se desprenden de las estimaciones de $\mathrm{P_y}$ (matriz de covarianza $\mathbb{C}ov\,(y,y)$) para cada uno de los tres métodos.

Si consideramos que las estimaciones $\overline{\mathrm{Y}}^{(\mathrm{MC})}$ y $\mathrm{P_y}^{(\mathrm{MC})}$ por muestreo Monte Carlo son muy buenas estimaciones de los momentos desconocidos $\mathbb{E}[\mathrm{y}]$ y $\mathbb{C}\mathrm{ov}\,(\mathrm{y},\mathrm{y})$, lo cual es debido a la enorme cantidad de puntos tomados para el muestreo, se puede concluir que la estimación de la estadística de Y por linealización es claramente sesgada e inconsistente. Sesgada porque $\overline{\mathrm{Y}}^{(\mathrm{LIN})} \neq \mathbb{E}[\mathrm{y}]$ e inconsistente porque $\mathrm{P_y}^{(\mathrm{LIN})} < \mathbb{C}\mathrm{ov}\,(\mathrm{y},\mathrm{y})$, subestimando la incertidumbre real, lo cual se manifiesta en que la elipse 1σ no es suficientemente ancha en la dirección radial.

En cambio, la transformación Unscented genera una estimación $\overline{\mathrm{Y}}^{(\mathrm{UT})}$ más próxima a $\mathbb{E}[\mathrm{y}]$ y, además, consistente, al menos para el caso en que la distribución de X es gaussiana.

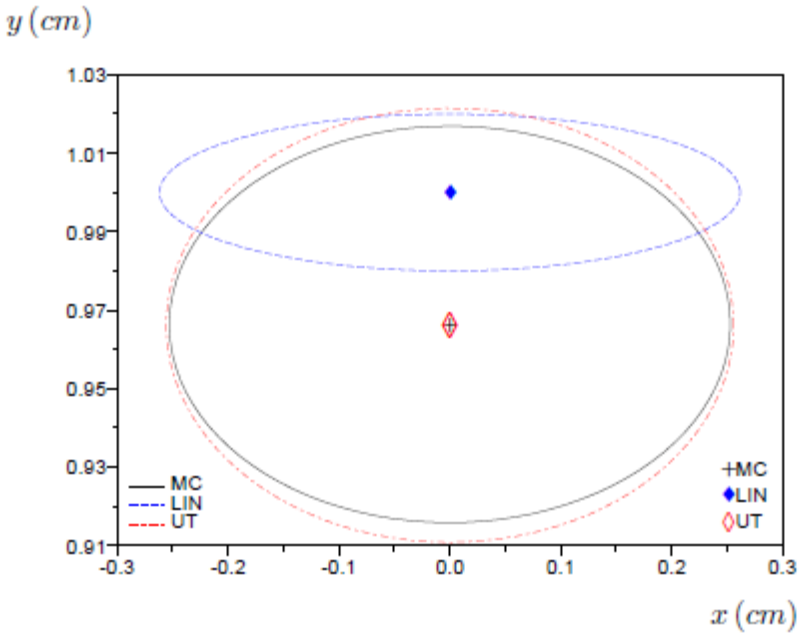

Figura 11.2: Medias ("+" MC, "◊" LIN y "◊" UT) y contornos 1σ estimados para $[x, y]^T$

Dadas estas propiedades de estimación superiores de la transformación Unscented sobre la linealización tradicionalmente empleada en el contexto del filtro de Kalman Extendido, y considerando la simplicidad con que se implementa, esto es, sin requerir el cálculo de la matriz Jacobiana, esta transformación se ha vuelto una alternativa atractiva a la linealización para aplicaciones de filtrado no lineal.

11.2. Filtro de Kalman Unscented

Esta extensión del filtro de Kalman es el resultado de incorporar la transformación Unscented al filtro de Kalman Extendido, para mejorar las aproximaciones que se hacen de la $\mathbb{E}[\mathrm{y}]$ y $\mathbb{C}\mathrm{ov}\,(\mathrm{y},\mathrm{y})$ que resulta de propagar una variable aleatoria, supuesta gaussiana, a través de una transformación no lineal.

Al igual que el filtro Extendido, el filtro Unscented es recursivo, por lo que en realidad se aproxima la distribución de probabilidad real del estado x_k, en cada instante k, por una distribución gaussiana.

Como el filtro Unscented no requiere del cálculo de matrices Jacobianas, presenta una complejidad de cálculo numérico comparable a la del filtro Extendido y, además, genera estimaciones de $\mathbb{E}[x_k \,|y_{1:k}]$ de mayor orden. Es por ello que se ha vuelto una herramienta corriente para abordar el problema de estimación de estado, en tiempo real, de un sistema dinámico no-lineal.

11.3. El filtro de Kalman como ejemplo de filtro adaptativo.

El objetivo de esta sección es introducir las nociones básicas relacionadas con el filtrado de Kalman como uno de los algoritmos de filtros adaptivos.

Un filtro adaptativo es un sistema con un filtro lineal que tiene una función de transferencia controlada por parámetros variables y un medio para ajustar esos parámetros de acuerdo con un algoritmo de optimización. Debido a la complejidad de los algoritmos de optimización, casi todos los filtros adaptativos son filtros digitales. Los filtros adaptables son necesarios para algunas aplicaciones debido a que algunos parámetros de la operación de procesamiento deseada no se conocen de antemano o están cambiando. El filtro adaptativo de lazo cerrado utiliza la retroalimentación en forma de señal de error para refinar su función de transferencia.

En términos generales, el proceso adaptativo de bucle cerrado implica el uso de una función de coste, que es un criterio para el rendimiento óptimo del filtro, para alimentar un algoritmo, que determina cómo modificar la función de transferencia del filtro para minimizar el coste en la siguiente iteración. La función de coste más común es el cuadrado medio de la señal de error.

El filtrado Kalman, en este sentido, es un ejemplo claro de filtro adaptativo. En él, la solución es óptima por cuanto el filtro combina toda la información observada y el conocimiento previo acerca del comportamiento del sistema para producir una estimación del estado de tal manera que el error es minimizado estadísticamente. El término recursivo significa que el filtro recalcula la solución cada vez que una nueva observación o medida ruidosa es incorporada en el sistema.

El filtro de Kalman es el principal algoritmo para estimar sistemas dinámicos representados en la forma de espacio estado. En esta representación, el sistema es descrito por un conjunto de variables denominadas de estado. El estado contiene toda la información relativa al sistema a un cierto punto en el tiempo. Esta información debe permitir la inferencia del comportamiento pasado del sistema, presente o futuro, dependiendo si la problemática a encarar por parte del filtro de Kalman es el alisado, el filtrado o la predicción respectivamente.

Las ecuaciones que se utilizan para derivar el filtro de Kalman se pueden dividir en dos grupos: las que actualizan el tiempo o ecuaciones de predicción y las que actualizan los datos observados o ecuaciones de actualización. Las del primer grupo son responsables de la proyección del estado al momento $n + 1$ tomando como referencia el estado en el momento n y de la actualización intermedia de la matriz de covarianza del estado. El segundo grupo de ecuaciones son responsables de la retroalimentación, es decir, incorporan nueva información dentro de la estimación anterior con lo cual se llega a una estimación mejorada del estado.

Las ecuaciones que actualizan el tiempo pueden también ser pensadas como ecuaciones de pronóstico, mientras que las ecuaciones que incorporan nueva información pueden considerarse como ecuaciones de corrección. Efectivamente, el algoritmo de estimación final puede definirse como un algoritmo de pronóstico-corrección para resolver numerosos problemas. Así, el filtro de Kalman funciona por medio de un mecanismo de proyección y

corrección al pronosticar el nuevo estado y su incertidumbre y corregir la proyección con la nueva medida. Este ciclo se muestra en la Figura 11.3.

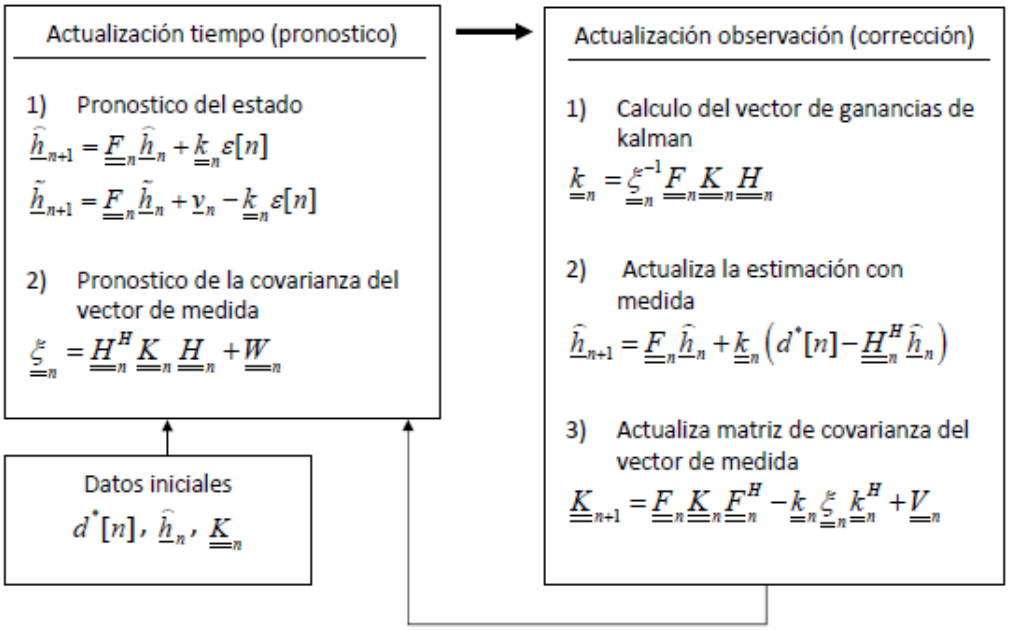

Figura 11.3: Esquema completo del concepto general del filtrado de Kalman.

11.4. Filtro Schmidt-Kalman.

El Filtro Schmidt-Kalman es una modificación del filtro Kalman para reducir la dimensionalidad de la estimación del estado, mientras que todavía se consideran los efectos del estado adicional en el cálculo de la matriz de covarianza y las ganancias de Kalman. Una aplicación común es dar cuenta de los efectos de parámetros molestos tales como los sesgos de los sensores sin aumentar la dimensionalidad de la estimación del estado. Esto asegura que la matriz de covarianza representará con precisión la distribución de los errores.

La ventaja principal de utilizar el filtro Schmidt-Kalman en lugar de aumentar la dimensionalidad del espacio de estado es la reducción de la complejidad computacional. Esto puede permitir el uso de filtros en sistemas en tiempo real. Otro uso de Schmidt-Kalman es cuando los sesgos residuales son inobservables, es decir, el efecto del sesgo no se puede separar de la medición. En este caso, Schmidt-Kalman es una forma robusta de no tratar de estimar el valor del sesgo, sino sólo de hacer un seguimiento del efecto del sesgo en la verdadera distribución de errores.

Para su uso en sistemas no lineales, los modelos de observación y de transición de estado pueden linealizarse en torno a la media actual y la estimación de covarianza en un método análogo al filtro Kalman Extendido.

12. ECUACIONES PARA MODELOS UAV

En este apartado se realiza el modelado del sistema de un UAV quadrotor al objeto de presentar una ejemplo de la extensibilidad del modelado presentado en este trabajo a sistemas dinámicos más complejos y susceptibles también de utilizar diferentes variantes del filtrado de Kalman para su implementación [19], [20] y [21].

12.1. Descripción del UAV

A tal fin se ha usado un helicóptero de pequeña escala con cuatro motores coplanarios, denominado quadrotor. El movimiento del quadrotor se debe a la diferencia de velocidad a la que giren los cuatro rotores, cuyos componentes son un motor eléctrico y unas hélices. Para lograr movimiento hacia adelante, la velocidad del rotor trasero debe ser aumentada y, simultáneamente, la velocidad del rotor delantero debe ser disminuida. El desplazamiento lateral se ejecuta con el mismo procedimiento, pero usando los rotores de la derecha y de la izquierda según lo antes expuesto. El movimiento de guiñada (yaw), se obtiene a partir de la diferencia en el par de torsión entre cada par de rotores, es decir, se aceleran los dos rotores con sentido horario mientras se desaceleran los rotores con sentido anti-horario, y viceversa.

12.2. Modelado del UAV

Se desarrolla a continuación el modelado basado en leyes físicas que describan la posición y orientación del helicóptero quadrotor. El modelo dinámico del helicóptero se presenta bajo la formulación matemática de Newton-Euler. Se supondrá al vehículo como un cuerpo rígido en el espacio, sujeto a una fuerza principal (empuje) y tres momentos (pares).

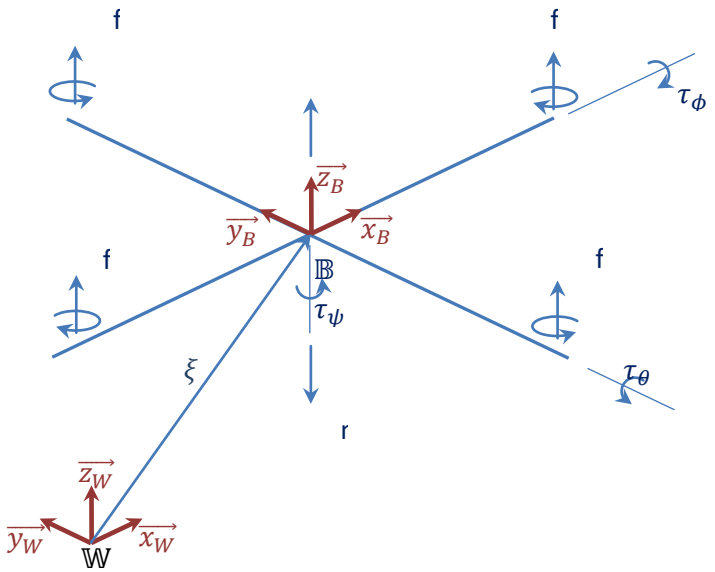

Figura 12.1: Modelo dinámico del UAV

Analizando de forma más detallada lo comentado respecto al movimiento en la descripción de un UAV, el par para generar un movimiento de balanceo o de roll (ángulo ϕ) se realiza mediante un desequilibrio entre las fuerzas f2 y f4 (ver figura 12.1). Para el movimiento de cabeceo o de pitch (ángulo θ), el desequilibrio se realizará entre las fuerzas f1 y f3. El movimiento en el ángulo de guiñada o de yaw (ángulo ψ) se realizará por el desequilibrio entre los conjuntos de fuerzas (f1, f3) y (f2, f4). Este movimiento será posible ya que los rotores 1 y 3 giran en sentido contrario a los rotores 2 y 4. Finalmente, el empuje total, que hará que el helicóptero se desplace perpendicularmente al plano de los rotores, se obtendrá como suma de las cuatro fuerzas que ejercen los rotores.

Se realizarán algunas hipótesis para la simplificación del modelo:

- Se desprecia el efecto de los momentos causados por un cuerpo rígido sobre las dinámicas traslacionales y el efecto suelo.
- Supondremos que el centro de masa es coincidente con el origen del sistema de coordenadas fijo del helicóptero y la estructura del helicóptero simétrica, obteniendo así una matriz de inercia diagonal.

A continuación, y antes de obtener el modelo del helicóptero, se mostrará cómo se estima la posición y orientación del vehículo respecto a un sistema de coordenadas de referencia inercial denominado $\mathbb{W}$.

Se considera que sobre el vehículo se encuentra definido un sistema de coordenadas ligado con origen en su centro de masas, como puede observarse en la Figura 12.1. El sistema de referencia $\{\mathbb{B}\} = \{\overrightarrow{x_B}, \overrightarrow{y_B}, \overrightarrow{z_B}\}$, donde el eje $\overrightarrow{x_B}$ es la dirección normal de ataque del helicóptero, $\overrightarrow{y_B}$ es ortogonal a $\overrightarrow{x_B}$ y es positivo hacia babor en el plano horizontal, mientras que $\overrightarrow{z_B}$ está orientado en sentido ascendiente y ortogonal al plano $\overrightarrow{x_B}$ O $\overrightarrow{y_B}$. El sistema de coordenadas inercial $\{\mathbb{W}\} = \{\overrightarrow{x_W}, \overrightarrow{y_W}, \overrightarrow{z_W}\}$ se considerará, en principio, fijo con respecto a la Tierra.

Se designa $\xi = [x, y, z]^T$ como la posición del centro de masas con respecto al sistema inercial $\mathbb{W}$. Por otro lado, la rotación del helicóptero vendrá dada por una matriz de rotación ${}^{W}R_B$, que representa la matriz de rotación que rota los ejes del sistema $\{\mathbb{W}\}$ para hacerlos coincidentes con los ejes del sistema $\{\mathbb{B}\}$.

La orientación de un cuerpo rígido puede ser obtenida utilizando diversos métodos. Se utilizará la convención-xyz (giro alrededor de x, y', z") muy utilizada para aplicaciones de ingeniería aeroespacial y que se denomina de Tait-Bryan. Los ángulos de Tait-Bryan son tres ángulos, usados para describir una rotación general en el espacio Euclídeo tridimensional a través de tres rotaciones sucesivas en tornos de ejes del sistema móvil en el cual están definidos. Los ángulos de Tait-Bryan para describir la orientación de un helicóptero serán denominados:

$$\eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}$$

La configuración de la rotación de un cuerpo rígido en el espacio se produce mediante tres rotaciones sucesivas:

1. Rotación según $\vec{x}$ de ϕ: el primer giro es el correspondiente al ángulo de roll o de balanceo, ϕ, y se realiza alrededor del eje $\vec{x}$.

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix}$$

2. Rotación según $\vec{y}$ de θ: el segundo giro se realiza alrededor del eje $\vec{y}$ a partir del nuevo eje y_B, con el ángulo pitch o ángulo de cabeceo, θ para dejar el ángulo z_B en su posición final.

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix}$$

3. Rotación según $\vec{z}$ de ψ: el tercer giro y última rotación corresponde al ángulo de guiñada o yaw, ψ, se realiza alrededor del eje $\vec{z}$ a partir del nuevo eje z_B, para llevar el helicóptero a su posición final.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix}$$

El resultado global de estos giros se muestra en la siguiente matriz:

$$^{W}R_B = \begin{bmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\theta & \cos\psi\sin\theta\cos\phi + \sin\psi\sin\theta \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\theta & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\theta \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}$$

La matriz de rotación inversa $^{B}R_W$ es la traspuesta de $^{W}R_B$, ya que dicha matriz es ortonormal. A partir de la relación entre la derivada de la matriz $^{W}R_B$ y una cierta matriz anti-simétrica se podrán obtener las ecuaciones cinemáticas de rotación del vehículo que establecen las relaciones entre las velocidades angulares.

La variación de los ángulos de Tait-Bryan es diferente a la velocidad angular del vehículo en el sistema de coordenadas de dicho cuerpo rígido, $\omega = [p, q, r]^T$, las cuales pueden ser medidas directamente a través de una unidad de medida inercial, IMU. La relación entre la velocidad angular en el sistema fijado al cuerpo y la variación en el tiempo de los ángulos de Tait-Bryan, en función de la derivada de los ángulos:

$$\omega = W\,(\eta)\,\dot{\eta}$$

$$\omega = W\,(R)\,\dot{\eta}$$

Si por el contrario se tiene la velocidad y se quiere obtener la derivada de los ángulos:

$$\dot{\eta} = W^{-1}\,(\eta)\,\omega$$

$$\dot{\eta} = W^{-1}\,(R)\,\omega$$

La matriz W se puede expresar tanto en función de los ángulos como en función de los elementos de la matriz de rotación cuyas definiciones son:

$$W(\eta) = \begin{bmatrix} 1 & 0 & \sin\theta \\ 0 & \cos\phi & \sin\phi\ \cos\theta \\ 0 & -\sin\phi & \cos\phi\ \sin\theta \end{bmatrix}, \quad W(\eta) = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & -\sin\phi\ sec\ \theta & \cos\phi\sec\theta \end{bmatrix}$$

$$W(R) = \begin{bmatrix} 1 & 0 & r_{31} \\ 0 & \dfrac{r_{33}}{\sqrt{{r_{32}}^2 + {r_{33}}^2}} & r_{32} \\ 0 & \dfrac{-r_{32}}{\sqrt{{r_{32}}^2 + {r_{33}}^2}} & r_{33} \end{bmatrix}, \quad W(R) = \begin{bmatrix} 1 & \dfrac{-r_{31}\,r_{32}}{{r_{32}}^2 + {r_{33}}^2} & \dfrac{-r_{31}\,r_{33}}{{r_{32}}^2 + {r_{33}}^2} \\ 0 & \dfrac{r_{33}}{\sqrt{{r_{32}}^2 + {r_{33}}^2}} & \dfrac{-r_{32}}{\sqrt{{r_{32}}^2 + {r_{33}}^2}} \\ 0 & \dfrac{r_{32}}{\sqrt{{r_{32}}^2 + {r_{33}}^2}} & \dfrac{r_{33}}{\sqrt{{r_{32}}^2 + {r_{33}}^2}} \end{bmatrix}$$

El movimiento rotacional del helicóptero viene dado por las componentes de las velocidades angulares en los tres ejes: velocidad angular de balanceo (p), velocidad angular de cabeceo (q), y velocidad angular de guiñada (r), sobre los ejes $\overrightarrow{x_B}$, $\overrightarrow{y_B}$ y $\overrightarrow{z_B}$, respectivamente. Estas velocidades rotacionales son debidas a los pares ejercidos sobre el sistema ligado al cuerpo del helicóptero producidas por las fuerzas externas, las cuales definen los diferentes momentos en los tres ejes: momento de balanceo (τ_ϕ), momento de cabeceo (τ_θ), y momento de guiñada (τ_ψ), sobre los ejes $\overrightarrow{x_B}$, $\overrightarrow{y_B}$ y $\overrightarrow{z_B}$, respectivamente.

El movimiento de traslación viene dado por las componentes de la velocidad ${}^W v_B = v$ en los tres ejes inerciales con relación a la velocidad absoluta del helicóptero. Las velocidades v y V están relacionadas por la expresión ${}^W v_B = {}^W R_B\ {}^W v_B$ o en notificación simplificada:

$$v = {}^W R_B\, V$$

12.3. Formulación por Newton-Euler

Se obtendrá a continuación de forma sucinta las ecuaciones dinámicas del helicóptero. A efectos de nomenclatura se usará v en lugar de ${}^W v_B$, V en lugar ${}^W v_B{}^{(B)}$ y ω en lugar de ${}^W \omega_B{}^{(B)}$.

Las ecuaciones dinámicas de un cuerpo rígido sujeto a fuerzas externas aplicadas al centro de masa y expresadas en el sistema de coordenadas ligado al cuerpo se pueden obtener a través de la formulación de Newton-Euler como sigue:

$$\begin{bmatrix} mI & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} \dot{V} \\ \dot{\omega} \end{bmatrix} + \begin{bmatrix} \omega \times m\, V \\ \omega \times J\, \omega \end{bmatrix} = \begin{bmatrix} F + F_d \\ \tau + \tau_d \end{bmatrix}$$

Donde J es la matriz de inercia, ${}^W J_B{}^{(B)}$, I es la matriz identidad, m es la masa total del helicóptero, F son las fuerzas aplicadas, F_d son fuerzas que actúan como perturbaciones, τ son los pares aplicados y τ_d son pares que actúan como perturbaciones.

Según las suposiciones realizadas al inicio del capítulo, la matriz de inercia supuesta diagonal, tiene la siguiente forma:

$$\mathrm{J} = \begin{bmatrix} \mathrm{I}_{xx} & 0 & 0 \\ 0 & \mathrm{I}_{yy} & 0 \\ 0 & 0 & \mathrm{I}_{zz} \end{bmatrix}$$

Si se considera el vector de estado $[\xi\ \mathcal{v}\ \eta\ \omega\ ]^T$, donde $\xi = [x, y, z]^T$ y $\mathcal{v}\ \in \mathbb{R}^2$ representa la posición y la velocidad lineal en $\{\ \mathbb{W}\ \}$, $\eta = [\phi\ \theta\ \psi\ ]^T$ y $\omega\ \in \mathbb{R}^3$ la orientación y la velocidad angular expresada en $\{\ \mathbb{B}\ \}$, pudiéndose escribir las ecuaciones de movimiento de un cuerpo rígido como sigue:

$$\begin{gathered} \dot{\xi} = \mathrm{v} \\ \mathrm{m}\dot{\mathrm{v}} = {}^{W}R_B\ \mathrm{F_b} \\ {}^{W}\dot{R_B} = {}^{W}R_B \mathrm{S}\ (\omega) \\ \mathrm{J}\dot{\omega} = -\omega\ \mathrm{x}\ \mathrm{J}\ \omega + \tau_{\mathrm{b}} \end{gathered}$$

Donde $\mathrm{S}\ (\omega) = ({}^{W}R_B)^T\ {}^{W}\dot{R_B}$

El helicóptero es un sistema mecánico sub-actuado con 6 grados de libertad y sólo 4 actuadores, que corresponden a la fuerza principal y a los tres momentos actuantes producidos por las cuatro hélices.

Las fuerzas y pares externos aplicados al cuerpo del helicóptero, $\mathrm{F_B} \in \mathbb{B}$ y $\tau_b \in \mathbb{B}$ respectivamente, consisten en su propio peso, en el vector de fuerzas aerodinámicas, en el empuje y en los pares desarrollados por los cuatro motores.

$$\begin{cases} {}^{W}R_B\ \mathrm{F_B} = -mg \cdot \mathrm{E_3} + {}^{W}R_{B\ \mathrm{E3}} \left(\sum_{i=1}^{4} b\ \Omega_i^2 \right) + \mathrm{A_T} \\ \tau_{\ B} = -\sum_{i=1}^{4} J_R\ (\ \omega\ \mathrm{x}\ \mathrm{E_3})\ \ \Omega_i + \tau_{\ a} + \mathrm{A_R} \end{cases}$$

Donde $\mathrm{A_T} = \left[A_x\ A_y\ \ A_z\right]^T$ y $\mathrm{A_R} = \left[A_p\ A_q\ \ A_r\right]^T$ son las fuerzas y pares aerodinámicos que actúan sobre el helicóptero, b es el coeficiente de empuje aplicado por los rotores, J_R es el momento de inercia rotacional del rotor alrededor de su eje, Ω_i es la velocidad de giro del i-ésimo rotor, el par τ_a es el vector de pares de control aplicados al helicóptero.

La fuerza principal o entrada de control, $\mathrm{U_1}$, se relaciona con los rotores mediante:

$$U_{1=} \left(\sum_{i=1}^{4} f_i \right) = \left(\sum_{i=1}^{4} b\ \Omega_i^2 \right)$$

Siguiendo el proceso matemático con el siguiente vector de estados:

$$\zeta = \begin{bmatrix} \xi \\ v \\ \eta \\ \omega \end{bmatrix}$$

Expresado según las componentes de cada uno como:

$$\zeta = \begin{bmatrix} x \\ y \\ z \\ u_0 \\ v_0 \\ w_0 \\ \phi \\ \theta \\ \psi \\ p \\ q \\ r \end{bmatrix}$$

Introduciendo además las expresiones de relación entre velocidad angular y derivadas de los ángulos, la relación entre v y V, la expresión de τ_a y la expresión que relaciona el empuje total con las velocidades de los rotores, se consigue expresar la ecuación diferencial no lineal de forma compacta con la siguiente expresión:

$$\dot{\zeta} = \mathrm{f}\,(\zeta) + \sum_{i=1}^{4} \mathrm{g}_i\ (\zeta)\ U_i$$

Siendo:

$$U_2 = \tau_{\phi_a}, U_3 = \tau_{\theta_a}, U_4 = \tau_{\psi_a},$$

Donde:

$$\zeta = \begin{bmatrix} u_0 \\ v_0 \\ w_0 \\ \frac{A_x}{m} \\ \frac{A_y}{m} \\ -g + \frac{A_z}{m} \\ p + q \sin\phi \tan\theta + r\ \cos\phi \tan\theta \\ q\ \cos\phi - r\ \sin\ \phi \\ q\ \sin\phi \sec\theta + r\ \cos\phi \sec\theta \\ \frac{(I_{yy} - I_{zz})}{I_{xx}}\ qr - \frac{J_R\Omega}{I_{xx}}\ q + \frac{A_p}{I_{xx}} \\ \frac{(I_{zz} - I_{xx})}{I_{yy}}\ pr - \frac{J_R\Omega}{I_{yy}}\ p + \frac{A_q}{I_{yy}} \\ \frac{(I_{xx} - I_{yy})}{I_{zz}}\ pq + \frac{A_r}{I_{zz}} \end{bmatrix}$$

$$\mathbf{g_1}(\zeta) = \begin{bmatrix} 0 & 0 & 0 & g_1^1 & g_1^2 & g_1^3 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$

$$\mathbf{g_1}(\zeta) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{l}{I_{xx}} & 0 & 0 \end{bmatrix}^T$$

$$\mathbf{g_1}(\zeta) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{l}{I_{yy}} & 0 \end{bmatrix}^T$$

$$\mathbf{g_1}(\zeta) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}^T$$

Con:

$$g_1^1 = \frac{1}{m} (\cos\psi \sin\theta \cos\phi + \sin\psi \sin\phi$$

$$g_2^1 = \frac{1}{m} (\sin\psi \sin\theta \cos\phi - \cos\psi \sin\phi)$$

$$g_3^1 = \frac{1}{m} (\cos\theta \cos\phi)$$

El modelo matemático obtenido puede asumirse suficientemente preciso en la presentación de todos los movimientos funcionales de un vehículo aéreo autónomo. Sin embargo, no es adecuado para el diseño de control, porque éste depende de fuerzas y momentos aerodinámicos, como puede verse en $\mathrm{f}(\zeta)$, los cuales son desconocidos ante la presencia de vientos y turbulencias imprevisibles, y de efectos giroscópicos que se consideran desconocidos debido a que, en principio, no se tiene acceso a las velocidades de los motores. En consecuencia, estos términos serán despreciados durante la fase de diseño del control y serán considerados como perturbaciones externas.

13. CONCLUSIONES Y LÍNEAS FUTURAS

Con este estudio se ha intentado mostrar las ventajas que supone la implementación de un filtro Kalman en la estimación de un estado. Se ha comprobado que se trata de una herramienta muy potente que no solo es capaz de manejar más de una variable al mismo tiempo, sino que la estimación que ofrece se acerca bastante a lo observado mediante la experiencia.

El desarrollo de un modelo de ruido para los sensores mejoraría su capacidad de estimación, ya que como se ha dicho esos errores no son constantes en la realidad y representan un modelo altamente no lineal.

Con el objetivo de aplicar un filtrado Kalman a estos estados no lineales, propios de la navegación de cualquier vehículo, aparece la alternativa del filtro de Kalman Extendido. Si bien se ha comprobado que el proceso que utiliza este filtro para aproximar la no linealidad a estados lineales no es del todo exacto e introduce numerosos errores, se presenta como una fuente de investigación alternativa en el desarrollo de estos estimadores matemáticos.

Observando el proceso de obtención de datos realizado con los sensores, se deduce que cuanto mayor sea la calidad de estos y mejor calibrados estén, más fiables serán los datos extraídos de ellos. Además, el hecho de perder las señales del GPS hace que la señal medida difiera de la tendencia seguida hasta el momento, haciendo oscilar la estimación del filtro y, en algunos casos, debido a la inestabilidad de este, provocar la divergencia de la solución.

Este fenómeno debe ser controlado, mejorando la calibración y capacidades de los sensores, pero también trabajando en la estabilidad del filtro, ya que existen aplicaciones prácticas en las que esta situación es frecuente, como puede ser por ejemplo durante la inmersión de un submarino. En este caso la señal de GPS recibida se perderá durante días, siendo la IMU la encargada de determinar la trayectoria seguida, con el consiguiente error derivado de la integración numérica que realiza. En el momento en el que el submarino emerja, actualizará su posición con la que reciba de la señal de GPS, dotando esto de total sentido al método aquí utilizado para fusionar los datos recibidos de ambos sensores.

A pesar de que el programa Matlab dispone de una función que realiza una implementación del filtrado Kalman, la cual es útil en numerosas aplicaciones prácticas que no requieren gran complejidad ni rigor matemático, esta función no es válida en el momento en que el sistema adquiere mayor dificultad de modelado, como en el caso del movimiento de un UAV, o las medidas recogidas necesitan ser tratadas y adaptadas al estado que se desea estimar.

Dado que, como se ha mencionado, la navegación de cualquier tipo de vehículo es normalmente un problema de estimación de un estado altamente no lineal, es necesaria la creación de un filtro que, a la vista de los resultados y limitaciones mostrados en este estudio que ofrece el filtro de Kalman Extendido, haga posible la estimación de estos estados. Nace, por tanto, la idea del filtro de Kalman Unscented, que presenta numerosas ventajas ya que aproxima una distribución gaussiana en lugar de aproximar una transformación no-lineal arbitraria, como es el caso del filtro Extendido. Se propone aquí como una posible línea futura la

programación de un filtrado Unscented para comprobar qué tipo de mejora supone y qué limitaciones presenta.

Otra línea de investigación muy importante se abre en el caso de aplicar un filtrado Kalman en tiempo real, esto es, a medida que se van recogiendo datos mediante los sensores, el filtro devuelve directamente su estimación, para así poder ir corrigiendo el estado automáticamente. Para ello, sería necesario implementar, la adquisición, el proceso y el filtrado en un lenguaje de programación adecuado para la plataforma donde vaya a ser ejecutado. Además de la dificultad de implementar todas las operaciones matriciales mediante codificación, lejos de la comodidad y las facilidades de Matlab, en tiempo real se pueden dar situaciones no previstas que hagan que el programa reaccionar ante las situaciones excepcionales que puedan producirse sin abortar ni comprometer la salida ya que esta puede estar siendo utilizada como señal de control para el vehículo. En caso de que la recogida de datos se hiciese a alta frecuencia, que es como se comprobó que se obtenían mejores resultados por el pequeño intervalo de tiempo entre estimaciones, sería necesaria una optimización del programa para que no hubiese demoras en las estimaciones.

Una de estas optimizaciones podría venir de la correcta definición de las matrices de covarianza Q y R, ya que quizás pueden tomar valores constantes y no recalcularse en cada iteración si son correctamente definidas y adaptadas al modelo cuyo estado se quiera estimar.

Por último, una vez realizadas las implementaciones del filtro Kalman para que estimen el estado en tiempo real, y adaptando el modelo dinámico del sistema a las complejas ecuaciones que implica el movimiento de un UAV, sería interesante volcar el programa a un microcontrolador Arduino e instalarlo en la aeronave, junto con los sensores, para que realice la estimación y mejora de las medidas obtenidas *in situ* y realimente directamente a los controladores y estabilizadores encargados de gobernar el vuelo de la misma, garantizando así una mejora en su capacidad de posicionamiento y permitiendo, por tanto, su aplicación a situaciones que requieren una mayor precisión.

Apéndice A: Programa Arduino

Este apéndice incluye el programa de Arduino utilizado para la obtención de los datos durante los experimentos realizados.

```
/*Incluir funciones propias del GPS*/
#include <Adafruit_GPS.h>
#define  mySerial Serial1
Adafruit_GPS GPS(&mySerial);
#define  GPSECHO  false
boolean usingInterrupt = false;


/*Incluir funciones propias de la IMU*/
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM303_U.h>
#include <Adafruit_BMP085_U.h>
#include <Adafruit_10DOF.h>


/*Asignar denominaciones unicas a los sensores*/
Adafruit_10DOF                dof   = Adafruit_10DOF ();
Adafruit_LSM303_Accel_Unified accel = Adafruit_LSM303_Accel_Unified (30301);
Adafruit_LSM303_Mag_Unified   mag   = Adafruit_LSM303_Mag_Unified (30302);
Adafruit_BMP085_Unified       bmp   = Adafruit_BMP085_Unified (18001);


/*Update this with the correct SLP for accurate altitude measurements*/
float seaLevelPressure = SENSORS_PRESSURE_SEALEVELHPA;



void initSensors(){
 if(!accel.begin()){
   /*There was a problem detecting the LSM303 ... check your connections*/
   Serial.println(F("Ooops, no LSM303 detected ... Check your wiring!"));
   while(1);}
 if(!mag.begin()){
   /*There was a problem detecting the LSM303 ... check your connections*/
   Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
   while(1);}
 if(!bmp.begin()){
   /*There was a problem detecting the BMP180 ... check your connections*/
   Serial.println("Ooops, no BMP180 detected ... Check your wiring!");
   while(1);}}
```

```
void setup(){
 /*Lineas del GPS*/
 Serial.begin(115200);

 //9600 NMEA is the default baud rate for Adafruit MTK GPS's- some use 4800
 GPS.begin(9600);
 mySerial.begin(9600);

 //RMC (recommended minimum) and GGA (fix data) including altitude
 GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);

 //Set the update rate to 1Hz
 GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);

 //Request updates on antenna status, comment out to keep quiet
 GPS.sendCommand(PGCMD_ANTENNA);

 //Ask for firmware version
 mySerial.println(PMTK_Q_RELEASE);

 /*Lineas de la IMU - PitchRollHeadingTemperature*/
 Serial.println(F("IMU + GPS")); Serial.println("");

 /*Initialise the sensors*/
 initSensors();

 /*Initialise the sensor*/
 if(!accel.begin()){
  /*There was a problem detecting the ADXL345 ... check your connections*/
  Serial.println("Ooops, no LSM303 detected ... Check your wiring!");
  while(1);}}



uint32_t timer            = millis();
uint32_t timer2           = millis();
uint32_t timerP           = millis();
float latitudeDegreesP    = 0;
float longitudeDegreesP = 0;
float speedP              = 0;
float altitudeP           = 0;
float angleP              = 0;
uint8_t fixqualityP        = 0;
uint8_t satellitesP       = 0;
boolean fixP              = false;
int flag                  = 0;
```

```
void loop(){
  //If millis() or timer wraps around, we'll just reset it
        if (timer > millis())    timer   = millis();
        if (timer2 > millis()) timer2 = millis();

        if (millis()-timer > 975){
                if (! usingInterrupt) {
                        //Read data from the GPS in the 'main loop'
                        char c = GPS.read();
                        //If you want to debug, this is a good time to do it!
                        if (GPSECHO)
                        if (c) Serial.print(c);}

                if (GPS.newNMEAreceived()){
                        if (!GPS.parse(GPS.lastNMEA()))
                                //This also sets the newNMEAreceived() flag to false
                                return;
                                //If we fail parsing a sentence we just wait for another

   timer = millis();
                                timerP = timer; //1
                                fixP = GPS.fix; //2
                                fixqualityP = GPS.fixquality; //3
                                satellitesP = GPS.satellites; //4
                                latitudeDegreesP = GPS.latitudeDegrees; //5
                                longitudeDegreesP = GPS.longitudeDegrees; //6
                                altitudeP = GPS.altitude; //7
                                speedP = GPS.speed; //8
                                angleP = GPS.angle;}} //9

  if (millis()-timer2 > 19){
                sensors_event_t accel_event;
                sensors_event_t mag_event;
                sensors_event_t bmp_event;
                sensors_vec_t   orientation;

                Serial.print(timer); Serial.print("\t"); //1
                Serial.print((int)fixP); Serial.print("\t"); //2
                Serial.print((int)fixqualityP); Serial.print("\t"); //3
                Serial.print((int)satellitesP); Serial.print("\t"); //4
                Serial.print(latitudeDegreesP,7); Serial.print("\t"); //5
                Serial.print(longitudeDegreesP,7); Serial.print("\t"); //6
                Serial.print(altitudeP); Serial.print("\t"); //7
                Serial.print(speedP); Serial.print("\t"); //8
                Serial.print(angleP); Serial.print("\t"); //9

                timer2 = millis();             //Reset the timer
                Serial.print(timer2);  Serial.print("\t"); //10
```

```
/*Calculate the heading using the magnetometer*/
mag.getEvent(&mag_event);

if (dof.magGetOrientation(SENSOR_AXIS_Z, &mag_event, &orientation)){
        /*'Orientation' should have valid heading data now*/
        //Serial.print(F("Heading:"));
        Serial.print(orientation.heading); Serial.print("\t");} //11
        //Serial.print(F(";"));
else {
        orientation.heading = 0.00;
        Serial.print(orientation.heading); Serial.print("\t");} //11


accel.getEvent(&accel_event);

if (dof.accelGetOrientation(&accel_event, &orientation)){
        /*'Orientation' should have valid roll and pitch fields*/
        //Serial.print(F("Pitch:"));
        Serial.print(orientation.pitch); Serial.print("\t"); //12
        //Serial.print(F("; "));
        //Serial.print(F("Roll:"));
        Serial.print(orientation.roll); Serial.print("\t");} //13
        //Serial.print(F(";"));
else {
        orientation.pitch = 0.00;
        orientation.roll    = 0.00;
        Serial.print(orientation.pitch); Serial.print("\t"); //12
        Serial.print(orientation.roll); Serial.print("\t");} //13

sensors_event_t event;
accel.getEvent(&event);


/*Display the results (acceleration is measured in m/s^2)*/
Serial.print(event.acceleration.x); Serial.print("\t"); //14
Serial.print(event.acceleration.y); Serial.print("\t"); //15
Serial.print(event.acceleration.z); Serial.print("\n");}} //16
```

Apéndice B: Tratamiento de medidas

Este apéndice incluye las funciones programadas en Matlab que realizan el tratamiento de las medidas obtenidas de los sensores.

1 Latitud, longitud y altitud a coordenadas x, y, z.

```
function [x,y,z] = Lat_long_alt_to_xyz (Lat,Long,Alt,lat_0,long_0)

%Conversión de grados a radianes
Latitud_GPS_rad  = Lat*(pi/180);
Longitud_GPS_rad = Long*(pi/180);

%Condiciones iniciales
R = 6371e3;    %Radio de la Tierra en metros

%CALCULO DE LA POSICIÓN RESPECTO A LA POSICIÓN INICIAL
%Calculo de la diferencia entre la posición actual y el origen
dif_lat  = Latitud_GPS_rad-lat_0;
dif_long = Longitud_GPS_rad-long_0;

%Calculo de la distancia entre la posición actual y el origen
a = sin(dif_lat/2)^2 +...
    cos(lat_0)*cos(Latitud_GPS_rad)*sin(dif_long/2)^2;
c = 2*atan2(sqrt(a),sqrt(1-a));
d = R*c;

%Calculo del bearing (ángulo en sentido horario desde la posición
origen entre la dirección del vector distancia y el norte
geográfico)
theta = atan2(sin(dif_long)*cos(Latitud_GPS_rad),...
    cos(lat_0)*sin(Latitud_GPS_rad) -
sin(lat_0)*cos(Latitud_GPS_rad)*cos(dif_long));

%Calculo de las posiciones en x,y,z
x = d*sin(theta);
y = d*cos(theta);
z = Alt;

end
```

2 Conversión de un vector en función para poder ser integrado

```
function Y = integral(x, n)

%Se forma un vector con los inputs de la función, el cual
posteriormente sale definido como función
x = n(1);
y = n(2);
z = n(3);
Y = [x y z];

End
```

3 Rotación de las aceleraciones de la IMU

```
function A = rotacion_acel_IMU (Pitch,Roll,Yaw,Acel_x,Acel_y,Acel_z)

%Conversión a radianes
pitch = Pitch*(pi/180);
roll  = Roll*(pi/180);
yaw   = Yaw*(pi/180);

%Matrices de rotación
R_z_yaw   = [cos(yaw) -sin(yaw) 0; sin(yaw) cos(yaw) 0; 0 0 1];
R_y_roll  = [cos(roll) 0 sin(roll); 0 1 0; -sin(roll) 0 cos(roll)];
R_x_pitch = [1 0 0; 0 cos(pitch) -sin(pitch); 0 sin(pitch)
cos(pitch)];

%Matriz de rotación total
R = R_z_yaw*R_y_roll*R_x_pitch;

%Aceleraciones giradas
A = R*[Acel_x; Acel_y; Acel_z];

end
```

Apéndice C: Filtro de Kalman Lineal

Este apéndice incluye la función programada en Matlab que implementa un filtro de Kalman lineal. Utiliza para ello las funciones definidas en el Apéndice B. En el ejemplo aquí mostrado los datos se recibían a 100Hz, por lo que el intervalo de filtrado era de 0.01s. En los ejemplos realizados a 50 Hz, la variable t debe tomar el valor 0.02s.

```
%FILTRO DE KALMAN LINEAL
clear all; clc; format long;

%Vamos a implementar el filtro para un movimiento MRU
%El resultado será una aproximación óptima a la posición y
velocidades reales del cuerpo en 3D (coordenadas x,y,z)

%INICIALIZACIÓN DE VARIABLES Y OBTENCIÓN DE DATOS
%Inicialización para la fusión de datos
t         = 0.01;  %Intervalo de integración. Tiempo de duración de
un ciclo
Vz_GPS    = 0;     %Movimiento en el plano
T_GPS_ant = 0;
x_GPS_ant = 0;
y_GPS_ant = 0;

%Inicialización de las matrices del filtro
X0    = zeros(6,1);     %Posición inicial y velocidades en los ejes
x, y, z para entrada al filtro Kalman
P0    = 7*eye(6);       %Matriz de la covarianza inicial del filtro
Kalman
A     = [1 0 0 t 0 0; 0 1 0 0 t 0; 0 0 1 0 0 t; 0 0 0 1 0 0; 0 0 0 0
1 0; 0 0 0 0 0 1];
W     = 0.01*ones(6,1); %Matriz de ruido del modelo dinámico. Viento
y desperfectos de rodadura de la carretera
H     = eye(6);
C     = eye(6);
Z     = 0.01*ones(6,1); %Matriz de ruido del sensor. Aleatorio,
"disponibilidad condicional" del uso militar
I     = eye(6);
Sol   = [];             %Matriz que incluye las soluciones de
posición y velocidad de GPS, de la fusión y de Kalman
X_ant = X0;
P_ant = P0;
X_Kal = X0;
P_Kal = P0;

%Inicialización matrices covarianza ruido
sum_Q     = zeros(6);
sum_R     = zeros(6);
sum_med_Q = zeros(6,1);
sum_med_R = zeros(6,1);

%Cargar el archivo de texto (separado con tabulación, no con ;)
datos = load('1_v_100_1.txt');
[m,n] = size(datos);
```

```
for i = 1:m
    %OPERACIÓN DE FUSIÓN
    %Extraer los datos del archivo de texto
    Tiempo_GPS    = datos(i,1);
    Latitud_GPS   = datos(i,5);
    Longitud_GPS  = datos(i,6);
    Altitud_GPS   = datos(i,7);
    Velocidad_GPS = datos(i,8);
    Angulo_GPS    = datos(i,9);
    Tiempo_IMU    = datos(i,10);
    Heading_IMU   = datos(i,11);
    Pitch_IMU     = datos(i,12);
    Roll_IMU      = datos(i,13);
    Acel_x_IMU    = datos(i,14);
    Acel_y_IMU    = datos(i,15);
    Acel_z_IMU    = datos(i,16);

    %Convertir coordenadas geográficas a cartesianas de GPS
    if i == 1
        Lat_0  = Latitud_GPS*(pi/180);
        Long_0 = Longitud_GPS*(pi/180);
    end
    [x_GPS,y_GPS,z_GPS] = Lat_long_alt_to_xyz
(Latitud_GPS,Longitud_GPS,Altitud_GPS,Lat_0,Long_0);

    %Rotación de ejes de la IMU
    Yaw          = Angulo_GPS*pi/180;      %Ángulo de guiñada del
móvil respecto al norte geográfico (radianes)
    Acel_rot_IMU = rotacion_acel_IMU (Pitch_IMU, Roll_IMU, Yaw,
Acel_x_IMU, Acel_y_IMU, Acel_z_IMU);

    %Integrar aceleraciones y velocidades IMU
    Vel_xyz_IMU = quadv(@(x)integral(x, Acel_rot_IMU),0,t);
    Pos_xyz_IMU = quadv(@(x)integral(x, Vel_xyz_IMU),0,t);

    %Obtención de componentes de velocidad en los ejes del
movimiento
    Vx_GPS = 0.514*Velocidad_GPS*sin(Angulo_GPS*pi/180); %m/s
    Vy_GPS = 0.514*Velocidad_GPS*cos(Angulo_GPS*pi/180); %m/s

    %Suma de velocidades de GPS e IMU (Fusión de datos)
    Vx_FUS = Vx_GPS + Vel_xyz_IMU(1);
    Vy_FUS = Vy_GPS + Vel_xyz_IMU(2);
    Vz_FUS = Vz_GPS + Vel_xyz_IMU(3);

    %Suma de posiciones de GPS e IMU (Fusión de datos)
    if x_GPS == x_GPS_ant && y_GPS == y_GPS_ant
        Tiempo_GPS = T_GPS_ant;
    end
    x_FUS     = x_GPS + Pos_xyz_IMU(1) + Vx_FUS*(Tiempo_IMU -
Tiempo_GPS)/1000; %Tiempo en segundos
    y_FUS     = y_GPS + Pos_xyz_IMU(2) + Vy_FUS*(Tiempo_IMU -
Tiempo_GPS)/1000; %Tiempo en segundos
    z_FUS     = z_GPS + Pos_xyz_IMU(3);
    T_GPS_ant = Tiempo_GPS;
    x_GPS_ant = x_GPS;
    y_GPS_ant = y_GPS;

    %Vector de estado de la fusión
    Med_FUS = [x_FUS; y_FUS; z_FUS; Vx_FUS; Vy_FUS; Vz_FUS];
```

```
    %IMPLEMENTACIÓN DEL FILTRO KALMAN
    %Predicción de Kalman basada en modelo dinámico: matriz de
estado
    X_p = A*X_ant + W;

    if i>=2; %Es necesaria una medida anterior y un estado anterior
para calcular las matrices Q y R
        %Calculo de la matriz de covarianza de la medida
        sum_med_Q = sum_med_Q + (X_p - Pre_Kal_ant);
        media_Q   = sum_med_Q./i;
        sum_Q     = sum_Q + (X_p - Pre_Kal_ant - media_Q)*(X_p -
Pre_Kal_ant - media_Q)';
        Q         = sum_Q./(i-1);

        %Predicción de Kalman basada en modelo dinámico: matriz de
covarianza
        P_p = A*P_ant*A' + Q;

        %Calculo de la matriz de covarianza de la medida
        sum_med_R = sum_med_R + (Med_FUS - Med_FUS_ant);
        media_R   = sum_med_R./i;
        sum_R     = sum_R + (Med_FUS - Med_FUS_ant -
media_R)*(Med_FUS - Med_FUS_ant - media_R)';
        R         = sum_R./(i-1);

        %Calculo de la ganancia de Kalman
        K = (P_p*H)/(H*P_p*H' + R);

        %Calculo de la medida
        Y = C*Med_FUS + Z;

        %Estimación de Kalman del estado actual
        X_Kal = X_p + K*(Y - H*X_p);
        P_Kal = (I - K*H)*P_p;
    end

    %Actualización de los estados anteriores para el siguiente bucle
    X_ant       = X_Kal;
    P_ant       = P_Kal;
    Pre_Kal_ant = X_p;
    Med_FUS_ant = Med_FUS;

    %Vector solución de posiciones de GPS, FUS y Kalman
    Sol = [Sol; x_GPS y_GPS z_GPS x_FUS y_FUS z_FUS X_Kal(1)
X_Kal(2) X_Kal(3)...
        Vx_GPS Vy_GPS Vz_GPS Vx_FUS Vy_FUS Vz_FUS X_Kal(4) X_Kal(5)
X_Kal(6)];

end
```

Apéndice D: Filtro de Kalman Extendido

Este apéndice incluye la función programada en Matlab que implementa un filtro de Kalman Extendido. Utiliza para ello las funciones definidas en el Apéndice B. En el ejemplo aquí mostrado los datos se recibían a 50Hz, por lo que el intervalo de filtrado era de 0.02s. En los ejemplos realizados a 100 Hz, la variable t debe tomar el valor 0.01s.

```
%FILTRO DE KALMAN EXTENDIDO
clear all; clc; format long;

%Vamos a implementar el filtro para un movimiento MRU
%El resultado será una aproximación óptima a la posición y
velocidades reales del cuerpo en 3D (coordenadas x,y,z)

%INICIALIZACIÓN DE VARIABLES Y OBTENCIÓN DE DATOS
%Inicialización fusión de datos
t         = 0.02;  %Intervalo de integración. Tiempo de duración de
un ciclo
Vz_GPS    = 0;     %Movimiento en el plano
T_GPS_ant = 0;
x_GPS_ant = 0;
y_GPS_ant = 0;

%Inicialización matrices del filtro
X0    = zeros(6,1);     %Posición inicial y velocidades en los ejes
x, y, z para entrada al filtro Kalman (6x6) [x_KAL y_KAL z_KAL
Vx_KAL Vy_KAL Vz_KAL]
P0    = 7*eye(6);       %Matriz de la covarianza inicial del filtro
Kalman (6x6)
A     = [1 0 0 t 0 0; 0 1 0 0 t 0; 0 0 1 0 0 t; 0 0 0 1 0 0; 0 0 0 0
1 0; 0 0 0 0 0 1];
W     = 0.01*ones(6,1); %Matriz de ruido del modelo dinámico. Viento
y desperfectos de rodadura de la carretera
Z     = 0.01*ones(3,1); %Matriz de ruido del sensor. Aleatorio,
"disponibilidad condicional" uso militar
I     = eye(6);
Sol   = [];             %Matriz que incluye las soluciones de
posición en cartesianas de GPS, de la fusión y de Kalman
X_ant = X0;
P_ant = P0;
X_Kal = X0;
P_Kal = P0;

%Inicialización matrices covarianza ruido
Q      = eye(6);
Q(6,6) = 0.85;
R      = eye(3);

%Cargar el archivo de texto (separado con tabulación, no con ;)
datos = load('2_v_50_1.txt');
[m,n] = size(datos);
```

```
for i = 1:m
    %OPERACIÓN DE FUSIÓN
    %Extraer los datos del archivo de texto
    Tiempo_GPS    = datos(i,1);
    Latitud_GPS   = datos(i,5);
    Longitud_GPS  = datos(i,6);
    Altitud_GPS   = datos(i,7);
    Velocidad_GPS = datos(i,8);
    Angulo_GPS    = datos(i,9);
    Tiempo_IMU    = datos(i,10);
    Heading_IMU   = datos(i,11);
    Pitch_IMU     = datos(i,12);
    Roll_IMU      = datos(i,13);
    Acel_x_IMU    = datos(i,14);
    Acel_y_IMU    = datos(i,15);
    Acel_z_IMU    = datos(i,16);

    %Convertir coordenadas geográficas a cartesianas de GPS
    if i == 1
        Lat_0  = Latitud_GPS*(pi/180);
        Long_0 = Longitud_GPS*(pi/180);
    end
    [x_GPS,y_GPS,z_GPS] = Lat_long_alt_to_xyz
(Latitud_GPS,Longitud_GPS,Altitud_GPS,Lat_0,Long_0);

    %Rotación de ejes de la IMU para hacer coindicir direcciones GPS
con IMU
    Yaw          = Angulo_GPS*pi/180;      %Ángulo de guiñada del
móvil respecto al norte geográfico (radianes)
    Acel_rot_IMU = rotacion_acel_IMU (Pitch_IMU, Roll_IMU, Yaw,
Acel_x_IMU, Acel_y_IMU, Acel_z_IMU);

    %Integrar aceleraciones y velocidades IMU
    Vel_xyz_IMU = quadv(@(x)integral(x, Acel_rot_IMU),0,t);
    Pos_xyz_IMU = quadv(@(x)integral(x, Vel_xyz_IMU),0,t);

    %Obtención de componentes de velocidad en los ejes del
movimiento
    Vx_GPS = 0.514*Velocidad_GPS*sin(Angulo_GPS*pi/180); %m/s
    Vy_GPS = 0.514*Velocidad_GPS*cos(Angulo_GPS*pi/180); %m/s

    %Suma de velocidades de GPS e IMU (Fusión de datos)
    Vx_FUS = Vx_GPS + Vel_xyz_IMU(1);
    Vy_FUS = Vy_GPS + Vel_xyz_IMU(2);
    Vz_FUS = Vz_GPS + Vel_xyz_IMU(3);

    %Suma de posiciones de GPS e IMU (Fusión de datos)
    if x_GPS == x_GPS_ant && y_GPS == y_GPS_ant
        Tiempo_GPS = T_GPS_ant;
    end
    x_FUS     = x_GPS + Pos_xyz_IMU(1) + Vx_FUS*(Tiempo_IMU -
Tiempo_GPS)/1000; %Tiempo en segundos
    y_FUS     = y_GPS + Pos_xyz_IMU(2) + Vy_FUS*(Tiempo_IMU -
Tiempo_GPS)/1000; %Tiempo en segundos
    z_FUS     = z_GPS + Pos_xyz_IMU(3);
    T_GPS_ant = Tiempo_GPS;
    x_GPS_ant = x_GPS;
    y_GPS_ant = y_GPS;
```

```
    %Conversión de coordenadas cartesianas a cilíndricas
    r_FUS = sqrt(x_FUS^2 + y_FUS^2 + z_FUS^2);
    theta_FUS = atan2(x_FUS, y_FUS);

    %Vector de estado de la fusión
    Med_FUS = [r_FUS; theta_FUS; z_FUS];


    %IMPLEMENTACIÓN DEL FILTRO KALMAN
    %Predicción de Kalman basada en modelo dinámico: matriz de
estado
    X_p = A*X_ant + W;

    if i>=2
        %Predicción de Kalman basada en modelo dinámico: matriz de
covarianza
        P_p = A*P_ant*A' + Q;

        %Calculo de la ganancia de Kalman
        J = [X_p(1)/sqrt(X_p(1)^2+X_p(2)^2)
X_p(2)/sqrt(X_p(1)^2+X_p(2)^2) 0 0 0 0;...
            X_p(2)/(X_p(1)^2+X_p(2)^2) -X_p(1)/(X_p(1)^2+X_p(2)^2) 0
0 0 0; 0 0 1 0 0 0];
        K = (P_p*J')/(J*P_p*J' + R);

        %Calculo de la medida
        Y = Med_FUS + Z;

        %Estimación de Kalman del estado actual
        g = zeros(3,1);
        %El cálculo del arco tangente si el denominador es 0 puede
dar problemas
        if X_p(1)==0 && X_p(2)==0
            g(2) = 0;
        else
            g(2) = atan2(X_p(1), X_p(2));
        end
        g(1)  = sqrt(X_p(1)^2 + X_p(2)^2 + X_p(3)^2);
        g(3)  = X_p(3);
        X_Kal = X_p + K*(Y - g);
        P_Kal = (I - K*J)*P_p;
    end

    %Actualización de los estados anteriores para el siguiente bucle
    X_ant       = X_Kal;
    P_ant       = P_Kal;
    Pre_Kal_ant = X_p;
    Med_FUS_ant = Med_FUS;

    %Vector solución de posiciones de GPS, FUS y Kalman
    Sol = [Sol; x_GPS y_GPS z_GPS x_FUS y_FUS z_FUS X_Kal(1)
X_Kal(2) X_Kal(3)...
        Vx_GPS Vy_GPS Vz_GPS Vx_FUS Vy_FUS Vz_FUS X_Kal(4) X_Kal(5)
X_Kal(6)];

end
```

Apéndice E: Filtro de Kalman Matlab

En este apéndice está incluido el código que implementa la función de la que dispone el programa Matlab para aplicar un filtrado Kalman sobre la fusión de datos realizada en esta memoria.

```
%FILTRO DE KALMAN LINEAL MATLAB
clear all; clc; format long;


%INICIALIZACIÓN DE VARIABLES Y OBTENCIÓN DE DATOS
%Inicialización fusión de datos
t         = 0.01;  %Intervalo de integración. Tiempo de duración de
un ciclo
Vz_GPS    = 0;     %Movimiento en el plano
T_GPS_ant = 0;
x_GPS_ant = 0;
y_GPS_ant = 0;

%Inicialización matrices del filtro
X0    = zeros(6,1);
P0    = 500*eye(6);
A     = [1 0 0 t 0 0; 0 1 0 0 t 0; 0 0 1 0 0 t; 0 0 0 1 0 0; 0 0 0 0
1 0; 0 0 0 0 0 1];
C     = eye(6);
Sol   = [];

%Inicialización matrices covarianza ruido
Q = eye(6);
R = eye(6);

%Inicialización parámetros del filtro
H = dsp.KalmanFilter('StateTransitionMatrix',A,
'ControlInputPort',false,...
    'MeasurementMatrix',C, 'ProcessNoiseCovariance',Q,
'MeasurementNoiseCovariance',R,...
    'InitialStateEstimate',X0, 'InitialErrorCovarianceEstimate',P0);

%Cargar el archivo de texto
datos = load('1_v_100_1.txt');
[m,n] = size(datos);


for i = 1:m
    %OPERACIÓN DE FUSIÓN
    %Extraer los datos del archivo de texto
    Tiempo_GPS    = datos(i,1);
    Latitud_GPS   = datos(i,5);
    Longitud_GPS  = datos(i,6);
    Altitud_GPS   = datos(i,7);
    Velocidad_GPS = datos(i,8);
    Angulo_GPS    = datos(i,9);
    Tiempo_IMU    = datos(i,10);
    Heading_IMU   = datos(i,11);
    Pitch_IMU     = datos(i,12);
    Roll_IMU      = datos(i,13);
```

```
    Acel_x_IMU    = datos(i,14);
    Acel_y_IMU    = datos(i,15);
    Acel_z_IMU    = datos(i,16);

    %Convertir coordenadas geográficas a cartesianas de GPS
    if i == 1
        Lat_0  = Latitud_GPS*(pi/180);
        Long_0 = Longitud_GPS*(pi/180);
    end
    [x_GPS,y_GPS,z_GPS] = Lat_long_alt_to_xyz
(Latitud_GPS,Longitud_GPS,Altitud_GPS,Lat_0,Long_0);

    %Rotación de ejes de la IMU para hacer coindicir direcciones GPS
con IMU
    Yaw          = Angulo_GPS*pi/180;      %Ángulo de guiñada del
móvil respecto al norte geográfico
    Acel_rot_IMU = rotacion_acel_IMU (Pitch_IMU, Roll_IMU, Yaw,
Acel_x_IMU, Acel_y_IMU, Acel_z_IMU);

    %Integrar aceleraciones y velocidades IMU
    Vel_xyz_IMU = quadv(@(x)integral(x, Acel_rot_IMU),0,t);
    Pos_xyz_IMU = quadv(@(x)integral(x, Vel_xyz_IMU),0,t);

    %Obtención de componentes de velocidad en los ejes del
movimiento
    Vx_GPS = 0.514*Velocidad_GPS*sin(Angulo_GPS*pi/180);
    Vy_GPS = 0.514*Velocidad_GPS*cos(Angulo_GPS*pi/180);

    %Suma de velocidades de GPS e IMU (Fusión de datos)
    Vx_FUS = Vx_GPS + Vel_xyz_IMU(1);
    Vy_FUS = Vy_GPS + Vel_xyz_IMU(2);
    Vz_FUS = Vz_GPS + Vel_xyz_IMU(3);

    %Suma de posiciones de GPS e IMU (Fusión de datos)
    if x_GPS == x_GPS_ant && y_GPS == y_GPS_ant
        Tiempo_GPS = T_GPS_ant;
    end
    x_FUS     = x_GPS + Pos_xyz_IMU(1) + Vx_FUS*(Tiempo_IMU -
Tiempo_GPS)/1000;
    y_FUS     = y_GPS + Pos_xyz_IMU(2) + Vy_FUS*(Tiempo_IMU -
Tiempo_GPS)/1000;
    z_FUS     = z_GPS + Pos_xyz_IMU(3);
    T_GPS_ant = Tiempo_GPS;
    x_GPS_ant = x_GPS;
    y_GPS_ant = y_GPS;

    %Vector de estado de la fusión
    Med_FUS = [x_FUS; y_FUS; z_FUS; Vx_FUS; Vy_FUS; Vz_FUS];

    %Filtro Kalman Matlab
    Kal = step(H, Med_FUS);
    Sol = [Sol; Kal(1) Kal(2) Kal(3) Kal(4) Kal(5) Kal(6)];

end
```

BIBLIOGRAFÍA

[1] José Luis Carretero Rodríguez. *Estudio de integración de sensores en UAVs.* Departamento de Ingeniería Electrónica, Universidad de Sevilla, ETSI, Sevilla, España, 2016.

[2] Manuel Bravo Escudero. *Aplicación Electrónica para UAV: integración de IMU y GPS.* Departamento de Ingeniería Electrónica, Universidad de Sevilla, ETSI, Sevilla, España, 2015.

[3] Belén Arronte Arroyuelos. *Sensor Fusion of differential GPS and Inertial Measuring Unit to measure state of a test vehicle*. Escuela Técnica Superior de Ingeniería (ICAI). Universidad Pontificia Comillas, Madrid, España, 2007.

[4] National Centers for Environmental Information. *Magnetic Field Calculators*. https://www.ngdc.noaa.gov/geomag-web/ NOAA, Internet.

[5] Métodos matemáticos. *Cuadratura y derivación numérica*. Dpto. Matemática Aplicada II. GIA. Universidad de Sevilla, 2012.

[6] Vara Rodríguez, David. *Sistemas para determinar la posición y orientación de herramientas quirúrgicas en operaciones de cirugía laparoscópica*. Escuela de Ingenierías Industriales. Universidad de Valladolid, 2014.

[7] Bostanci, Erkan. *Motion model transitions in GPS-IMU sensor fusion for user tracking in augmented reality*, 2015.

[8] Mohinder S. Grewal, Angus P. Andrews. *Kalman Filtering. Theory and Practice Using MATLAB.* John Wiley & Sons, Inc., tercera edición, 2008.

[9] Michel van Biezen. iLectureOnline, http://www.ilectureonline.com/lectures/subject/SPECIAL%20TOPICS/ ,2016

[10] Conor Lynch, Michael J. OMahony, Ted Scully. *Simplified method to derive the Kalman Filter covariance matrices to predict wind speeds from a NWP model.* Elsevier Ltd., 62:676-685, 2004.

[11] Clemson university. Lecture notes, http://cecas.clemson.edu/~ahoover/ece854/lecture-notes/, Nonlinear filtering, 2016.

[12] Ribeiro, Maria Isabel. *Kalman and Extended Kalman Filters: Concept, Derivation and Properties*. Institute for Systems and Robotics. Instituto Superior Tecnico, Lisboa.

[13] Clemson university. Lecture notes, http://cecas.clemson.edu/~ahoover/ece854/lecture-notes/, Extended Kalman filter, 2016.

[14] Eric A. Wan,Rudolph van der Merwe. *The Unscented Kalman Filter for Nonlinear Estimation*. Oregon Graduate Institute of Science & Technology, Oregon, 2000.

[15] Clemson university. Lecture notes, http://cecas.clemson.edu/~ahoover/ece854/lecture-notes/, Extended Kalman filter, 2016.

[16] Quijano, Jorge. *Estimation of the position of a moving target using the Extended Kalman Filter*. Centro de Estudios Científicos, Chile, 2006.

[17] Alejandro Pascual. *EKF y UKF: dos extensiones del filtro de Kalman para sistemas no lineales aplicadas al control de un péndulo invertido*. Facultad de Ingeniería, UDELAR, 2006.

[18] Varios autores. *Procesamiento digital de señales*. FCEFyN, Universidad Nacional de Córdoba http://www.dsp.efn.unc.edu.ar/documentos/Filtros_adaptivos.pdf

[19] Mayorga Rodríguez, Rodrigo Alberto. *Sistema de Navegación para Vehículos Aéreos Cuadricópteros*. Escola Politècnica Superior de Castelldefels. Universitat Politècnica de Catalunya, 2009.

[20] Carlos Soria, Francisco Rossomando, Angel Veca, Pedro Campillo. *Modelado y Diseño del Control de un Tricóptero UAV*. Instituto de Automática , CONICET, UNSJ.

[21] García Rodríguez, Ramón Andrés. *Herramienta para la simulación integrada de subsistemas en un equipo quadrotor*. ETSI, Universidad de Sevilla.