Trabajo Fin de Grado
Ingeniería Electrónica, Robótica y Mecatrónica

Localización multisensor de robots aéreos para misiones de inspección en entornos sin GNSS

Autor: José García Villalón

Tutor: José Ramiro Martínez de Dios

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado
Ingeniería Electrónica, Robótica y Mecatrónica

Localización multisensor de robots aéreos para misiones de inspección en entornos sin GNSS

Autor:
José García Villalón

Tutor:
José Ramiro Martínez de Dios
Catedrático

Dpto. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado: Localización multisensor de robots aéreos para misiones de inspección en entornos sin GNSS

Autor: José García Villalón
Tutor: José Ramiro Martínez de Dios

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Agradecer enormemente a mi familia por el apoyo constante en cada paso que doy. A mis compañeros, Jaime, Alfonso y LuisCa, por los ánimos y la fuerza en los días más y menos buenos. Y a mi tutor, Ramiro, por su confianza y tutela.

José García Villalón

Sevilla, 2022

Resumen

El presente trabajo incorpora métodos, algoritmos y elementos capaces de desarrollar el problema de localización para un sistema aéreo libre de GNSS. Un problema que se aborda desde el punto de vista de la aplicación de filtros estadísticos y la fusión sensorial de la percepción provista por distintos sensores. Se enfocará principalmente en el estudio del Filtro Extendido de Kalman, como también en su implementación en entornos como MATLAB o ROS. Esto último permite su uso e incorporación directa en múltiples proyectos y desarrollos futuros.

Abstract

The current proyect incorporates methods, algorithms and elements capable of developing the location problem for a free aerial GNSS system. This problem runs from the point of view of the application of statistical filters and the sensory fusion that different sensors provide. It will focus mainly on the study of the Extended Kalman Filter, as well as its implementation in environments such as MATLAB or ROS. The latter allows its use and direct incorporation in multiple projects and future developments.

Índice

Notación

x_t	Vector de estados en el instante t
A	Matriz del modelo cinemático del sistema
B	Matriz del modelo de actuación
u_t	Vector de actuaciones en el instante t
ε_t, δ_t	Vector con ruido gaussiano en el instante t
μ_t	Estimación del vector de estados en el instante t
Σ_t	Matriz de covarianza de los estados en el instante t
R	Matriz de incertidumbre del modelo A
z_t	Vector de medidas en el instante t
C	Matriz del modelo de observación
Q	Matriz de incertidumbre de las medidas A
K_t	Ganancia de Kalman en el instante t A
$\bar{\mu}_t$	Predicción del vector de estados en el instante t
$\bar{\Sigma}_t$	Matriz de covarianza predicha de los estados en el instante t
I	Matriz identidad
$g(), h()$	Funciones no lineales
∂	Derivada parcial
G_t	Jacobiano del modelo del sistema en el instante t
H_t	Jacobiano del modelo de observación en el instante t
x,y,z	Coordenas de posición en el espacio en un instante
v_x,v_y,v_z	Velocidades en cada eje espacial en un instante
r	Desviación típica de la matriz R
$A^\top$	Traspuesta de A
v_x^IMU	Velocidad en el eje x, según la IMU, en el instante actual
v_x^{k-1}	Velocidad en el eje x en el instante previo
d	Distancia entre dos puntos
q_i	Desviación típica de la matriz Q, para el sensor i
${}^wT_c$	Matriz de transformación del sistema de referencia global al de la cámara
${}^wR_c$	Matriz de rotación del sistema de referencia global al de la cámara
${}^wt_c$	Vector de traslación del sistema de referencia global al de la cámara
f	Distancia focal de la cámara
M,N	Dimensiones de la imagen en píxeles, alto y ancho
w,h	Dimensiones del sensor de la cámara, ancho y alto
(u,v)	Coordenadas de un píxel en la imagen
ρ_x,ρ_y	Dimensiones efectivas del sensor de la cámara
f_x,f_y	Distancia focal efectiva en cada eje de la cámara
K	Matriz de parámetros intrínsecos de la cámara

${}^{w}P$	Punto en el espacio referido al sistema global
p	Punto en la imagen
h_t	Vector de medidas predichas en el instante t

1 Introducción

1.1 Objetivos

El objetivo del trabajo a presentar se describe como la implementación de un algoritmo de localización basado en filtros estadísticos para robots aéreos sin posibilidad de uso de GNSS.

De forma detallada, la localización será necesaria para operaciones de inspección y mantenimiento de entornos de difícil acceso o de riesgo humanitario. Dichos entornos presentan dificultades o imposibilitan el uso de GNSS (*Global Navigation Satellite System*) como recurso para localización del robot. Por ello, será necesario el uso de métodos, sensores y algoritmos capaces de realizar la tarea propuesta partiendo de las limitaciones presentes.

La solución que se pretende alcanzar y que será objeto de estudio vendrá dada por el uso de filtros estadísticos en conjunto con la fusión sensorial de varios sensores disponibles.

En concreto, se hará especial énfasis en la implementación de un Filtro Extendido de Kalman, el cual acondicionará intrínsecamente la fusión de los distintos sensores requeridos para obtener la localización de un quadrotor.

En cuanto a los sensores empleados, se cuenta con una IMU, sensores de rango del tipo balizas radio y una cámara fija. Aunque existe la posibilidad de añadir cualquier otro dispositivo sensorial.

La localización del quadrotor se obtendrá en referencia a un robot terrestre que comandará las operaciones de inspección. Este vehículo terrestre, aunque no estará dentro del problema a abordar, se mantendrá como el sistema de referencia global del robot aéreo.

Por último, una vez implementada la solución propuesta, se pretende evaluar y analizar su bondad, robustez y eficacia en distintas situaciones experimentales propuestas para la localización del robot aéreo.

1.2 Contexto

La idea a partir de la que surgen los objetivos descritos, así como el interés del presente trabajo, tiene como origen un proyecto de mayor envergadura situado en un marco de investigación internacional.

Se trata de **PILOTING** [5], un proyecto de investigación financiado por la *Comisión Europea*. Su nombre es un acrónimo proveniente de: *"PILOTs for robotic INspection and maintenance Grounded on advanced intelligent platforms and prototype applications"*.

Este proyecto nace de la necesidad e importancia de aumentar la eficiencia y la calidad de las actividades de inspección y mantenimiento en infraestructuras civiles, las cuales van deteriorandose gradualmente. Para mantener los niveles de seguridad necesarios en estas infraestructuras envejecidas, es vital que las inspecciones se realicen de forma correcta y exhaustiva.

PILOTING propone la adaptación, integración y demostración de soluciones robóticas, en una plataforma integrada, que será probada y evaluada a gran escala: refinerías, puentes/viaductos y túneles. El objetivo es minimizar los costes, los riesgos y las interrupciones asociadas con las inspecciones y el mantenimiento estructural.

Figura 1.1 Robot aéreo PILOTING [5].

Cuenta con la adaptación de vehículos robotizados y toma de datos. En una primera versión de los robots PILOTING, se tiene: robots aéreos (AEROX, AERO-CAM, VIAD-DRONE, TTDRONE), terrestres (CART, RISING) y crawlers/híbridos (BIKE/HYBRID). Uno de estos robot aéreos puede verse en la Figura 1.1. Además, se han obtenido los primeros conjuntos de datos y se ha definido la configuración de los vehículos robóticos y los sensores de inspección.

Se trata de un proyecto con múltiples desarrollos en funcionalidades de la robótica autónoma. Como por ejemplo, avances en la localización y navegación libre de GNSS para sistemas aéreos, manipulación desde la plataforma RISING, el único vehículo terrestre equipado con un manipulador y por tanto capaz de realizar tareas complejas de manipulación, sistema de navegación para los robots terrestres y los rastreadores, etc.

Es en la primera de estas funcionalidades donde se enmarca el presente trabajo. Se pretende dar una solución a la localización y navegación libre de GNSS para sistemas aéreos mediante los conocimientos y técnicas adquiridos en el grado.

1.3 Estructura

El presente trabajo se ha estructurado entorno a un orden pensado para su entendimiento teórico inicial llegado a la resolución práctica del mismo en instancias finales. Pudiendo verse también como un incremento de detalles mientras más se avanza en la lectura.

- ***Capítulo 1: Introducción***
  Introducción al problema propuesto y su contexto.

- ***Capítulo 2: Estado del arte***
  Base teórica del presente trabajo. Descripción de las técnicas, algoritmos y elementos de interés en la implementación. Fundamentos de filtros estadísticos.

- ***Capítulo 3: Diseño del método***
  Métodos y modelos empleados en el trabajo. Desarrollo de los mismos con enfoque en el problema

descrito. Explicación detallada de los elementos incluidos.

- ***Capítulo 4: Implementación***
  Especificación del uso de las herramientas y software aplicados en el trabajo. Implementación del trabajo desarrollado en distintos entornos de simulación.

- ***Capítulo 5: Experimentos y análisis de resultados***
  Análisis de experimentos realizados en base a las implementaciones descritas. Simulación de escenarios y casos prácticos. Exposición y comprensión de los resultados obtenidos.

- ***Capítulo 6: Conclusiones y desarrollos futuros***
  Valoración y reflexión final del presente trabajo y su realización. Ideas y sugerencias para futuros desarrollos y posibles mejoras.

2 Estado del arte

2.1 Localización de robots

Se entiende la localización de robots como un problema a resolver cada vez que se requiere desarrollar algún tipo de aplicación realizada por robots, en algunas de mayor importancia que en otras. La solución a este tipo de problema viene dada por la implementación de un sistema capaz de definir una posición y orientación del mismo con la mayor precisión posible. Dependiendo del método utilizado y la aplicación, dicha precisión podrá ser de mayor o menor magnitud.

El problema de localización en robots móviles es de mayor importancia aún, debido a la dificultad que ofrecen los efectos dinámicos no lineales del entorno, la libertad de movimiento en el mismo, el modelo del propio robot y su mecánica, y los sensores de los que se sirve. Todos estos fenómenos son los responsables de que localizar al robot de forma exacta en el medio en el que se encuentra no sea posible.

Los métodos para localización pueden variar desde soluciones geométricas, por pura medida sensorial, o mediante estimaciones de localización. En todos ellos es necesario dotar al robot con sensores que le permitan adquirir algún tipo de percepción de su entorno, y con los que poder localizarse en el mismo.

Los métodos de estimación de posición o de localización se basan principalmente en métodos probabilísticos [16], donde se cuenta con la ventaja de poder concretar de forma más acentuada la precisión con la que se tiene la estimación respecto a la localización real. Es decir, es posible conocer, además de la estimación aportada de la localización del robot, la incertidumbre o el posible margen de error que esa estimación pueda contener. Esto es idóneo, en ocasiones donde conocer la localización exacta del robot no es posible.

Dentro de estos métodos de estimación probabilísticos, existen multitud de sistemas de estimación, entendiéndose estos como el conjunto de algoritmos y sensores que contribuyen al funcionamiento global del sistema. Estos métodos o sistemas pueden variar en función de factores como el campo u objetivo en el que vayan a aplicarse, o el grado de complejidad que presente el problema de localización.

Este tipo de métodos, y la robótica probabilística en general, poseen varias ventajas y otorgan soluciones alternativas de interés en contraposición a los métodos más convencionales. El principal punto a tener en cuenta es la posibilidad de tratar el problema de las incertidumbres. Dicho problema no se presenta únicamente en el resultado final o las medidas, también en el modelo del robot, las aproximaciones asumidas o en el propio computo del software que gobierna al robot. La mejor manera de asimilar el conjunto de todo es mediante uso de la probabilidad como base, tanto para el computo como para el punto de como afrontar el problema.

Concretando, un problema de percepción se traduce en un problema de estimación de estados, o lo que es igual, estimación de la localización del robot. Con el uso de la robótica probabilística, existe la posibilidad de integrar los datos que un robot toma de sus sensores en conjunto con las limitaciones y los errores que esos datos puedan contener. Esto nutre de robustez al sistema global, mejorando lo que podrían ofrecer otras técnicas basadas en modelos o de carácter reactivo.

En relación a lo anterior, existen muchos tipos de sensores con distintas características y tipos de medida, que ayudan a obtener información del entorno. Estos se verán posteriormente en mayor profundidad. De esta información que aportan los sensores se sirven los algoritmos de estimación, los cuales suelen ser usualmente filtros bayesianos, aunque estos tienen muchas más utilidades.

Los filtros bayesianos han resultado ser una de las grandes opciones para la resolución de la localización de robots móviles. Tanto que se han derivado una gran cantidad de ellos buscando adaptarse a todo tipo de situaciones en este ámbito. Como punto de partida se tiene el Filtro de Kalman, un filtro óptimo pero con ciertas restricciones de uso, por lo que como se venía diciendo, se han derivado de él, filtros no óptimos pero más adecuados para problemas fuera del alcance del anterior.

2.2 KF y EKF

Dentro de los algoritmos bayesianos, resultan de gran interés y utilidad en la práctica habitual el Filtro de Kalman y derivados, como el Filtro Extendido de Kalman.

Este tipo de algoritmos tienen en común ciertas características, como estar basados en el cálculo probabilístico, ser algoritmos recursivos, y fundamentalmente poder dividirse en dos partes bien diferenciadas; la primera conocida como predicción o actualización de control, y la segunda como actualización de medida.

El término de algoritmo recursivo hace referencia a que para cada iteración del mismo se necesitan los datos de la iteración previa; de otra forma, para la ejecución de la primera parte del algoritmo son necesarios los resultados de la segunda de la iteración previa, como para la ejecución de la segunda son necesarios los resultados de la primera parte actual. Por definición, dicha recursividad requiere de una inicialización del algoritmo.

Por otro lado en un Filtro de Bayes general [16], en la primera parte se define el procesamiento del control. Dicho control o actuación es la aplicada al robot móvil, bien como señal de actuación para su movimiento o bien como medida interna en su dinámica. En la segunda parte se trata de actualizar la creencia de la estimación que proporciona la parte anterior mediante la toma de una medida, y añadir la probabilidad de que la estimación y la medida coincidan. Posteriormente, se seguirá iterando hasta que la probabilidad de un estado sea lo suficientemente segura y confiable, es decir, se produzca una convergencia del algoritmo.

Por tanto, el filtro de Bayes es una forma de estimar el estado de un entorno y de un robot, donde un sistema dinámico modela la interacción del robot con el medio. Estas dinámicas se rigen por dos leyes de probabilidad; la distribución de transición de estado (la manera en la que cambia el estado en el tiempo, por ejemplo, el posible efecto de una actuación) y la distribución de medida (la forma en la que depende la medida del estado).

En cada iteración del filtro, se toman todas las hipótesis de probabilidad posibles, se evalúan, se comparan los resultados y se avanza según autorice la acción de mayor probabilidad. Todas estas hipótesis y cálculos requieren un escenario de Markov, es decir, toman en consecuencia las suposiciones de Markov o de estados completos, donde el estado actual del filtro es independiente de datos pasados y futuros al mismo. Esto en la práctica no es siempre así, aunque se suele tomar como válidas ciertas aproximaciones.

Partiendo del filtro de Bayes se pueden obtener y derivar numerosos algoritmos con características y aplicaciones muy variadas, provenientes muchos de ellos de grandes aproximaciones y consideraciones aplicadas. Para esto último, ha de tenerse en cuenta factores como la eficiencia computacional, la precisión de la aproximación y su facilidad para la implementación.

Una vez entendida la importancia y las posibilidades que brinda un filtro de Bayes a la estimación en robótica, el siguiente paso consiste en aplicar algunas de las aproximaciones mencionadas, ya que este algoritmo descrito se trata de un método teórico no implementable. La primera gran aproximación es considerar las distribuciones de probabilidad del filtro como distribuciones normales multivariable. Con ello se puede hablar de filtros Gaussianos.

Una distribución normal multivariable se caracteriza por dos parámetros, la media y la covarianza. Pa-

rametrizar una gaussiana por su media y covarianza se conoce como "parametrización de momentos", ya que representan respectivamente, el primer y segundo momento de una distribución de probabilidad. Una distribución de probabilidad con estas características es unimodal, es decir, tiene un único máximo (centrado en la media).

A continuación, se describirán dos filtros gaussianos y bayesianos ya mencionados; el Filtro de Kalman, el cual implementa el filtro de Bayes usando parametrización de momentos para un determinado tipo de problemas con dinámicas y funciones de medidas linales, y el Filtro de Kalman extendido, una extensión del anterior para problemas no linales.

2.2.1 Filtro de Kalman (KF)

El Filtro de Kalman [16] se usa como técnica de filtrado y predicción en sistemas gaussianos lineales. Únicamente puede ser implementado para estados continuos, aunque existen derivaciones y otras alternativas para estados discretos. Este filtro usa distribuciones de probabilidad según su parametrización en momentos, siendo estos en tiempo t, media μ_t y covarianza Σ_t.

Además de mantener las suposiciones de Markov, este filtro debe añadir otras tres propiedades para considerar sus distribuciones como gaussianas. La primera de ellas es que las distribuciones deben ser funciones lineales con ruido gaussiano. Este tipo de ruido a su vez, puede modelarse como una distribución gaussiana de media cero. Esta propiedad debe tenerse en cuenta tanto para la función de probabilidad de transición de estados, como para la probabilidad de medida y las probabilidades de inicialización del filtro, siendo este el motivo de que se hable de tres propiedades.

Se define como estado, todos los aspectos del robot y su entorno que afectan al sistema en instantes futuros. Dichos aspectos pueden ser dinámicos o estáticos. Usualmente se toma como estado un vector de variables aleatorias que cumplen con lo anterior. Una variable de estado puede ser, por ejemplo, la posición u orientación del robot.

Si se quiere representar el estado como una función lineal que cumpla con las necesidades del filtro, su estructura debe adecuarse según:

$$x_t = A_t \cdot x_{t-1} + B_t \cdot u_t + \varepsilon_t \tag{2.1}$$

En esta función se distinguen tres componentes principales. El primero de ellos representa la evolución esperada del estado según su modelo (A) y el valor del mismo en el instante anterior. El segundo añade la actuación ejercida en el instante actual. Y el tercero consta de una variable aleatoria que modela el ruido gaussiano mencionado, con media cero y covarianza R_t.

La parametrización de la función anterior quedaría, según lo establecido por el Filtro de Kalman, como:

$$\mu_t = A_t \cdot x_{t-1} + B_t \cdot u_t \tag{2.2}$$

$$\Sigma_t = R_t \tag{2.3}$$

En el caso de el vector de medidas, al igual que el de estados, debe seguir también el modelo de una función lineal. Esta función debe relacionar la medida con el estado, incluyendo además un ruido gaussiano representativo de las incertidumbres que una medida ofrece en el mundo real. De esta forma, la función del vector de medidas se representa como:

$$z_t = C_t \cdot x_t + \delta_t \tag{2.4}$$

El primer término relaciona la medida con el estado según un modelo de observación lineal (C), mientras que el segundo es el encargado de añadir el ruido gaussiano, también de media cero y covarianza Q_t. En este caso, es frecuente modelar el ruido según el error e incertidumbres que introducen los sensores de medida en el sistema; y los cuales suelen incluirse en sus especificaciones y características. La parametrización de esta función toma la siguiente forma:

$$\mu_t = C_t \cdot x_t \tag{2.5}$$

$$\Sigma_t = Q_t \tag{2.6}$$

Por último, la inicialización del filtro se hace directamente en forma paramétrica:

$$\mu_t = \mu_0 \tag{2.7}$$

$$\Sigma_t = \Sigma_0 \tag{2.8}$$

Una vez definidas las funciones lineales que modelan las distribuciones de probabilidad de las que consta el filtro, y luego de haberlo inicializado, se puede conformar el algoritmo que gobierna el Filtro de Kalman.

Algoritmo

Filtro de Kalman $(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \tag{2.9}$$

$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^\top + R_t \tag{2.10}$$

$$K_t = \bar{\Sigma}_t C_t^\top (C_t \bar{\Sigma}_t C_t^\top + Q_t)^{-1} \tag{2.11}$$

$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \tag{2.12}$$

$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \tag{2.13}$$

return μ_t, Σ_t

El algoritmo del filtro se define para el instante actual, t. El objetivo es el cálculo de los parámetros correspondientes a la media y la covarianza de la distribución de probabilidad del filtro en dicho instante. Esta media puede entenderse como la estimación de las variables aleatorias del vector de estados en el instante t, mientras que la covarianza añade la información sobre las incertidumbres y error que conlleva la estimación ejecutada.

El computo del algoritmo necesita de información previa, como la media y la covarianza calculadas en el instante anterior, el vector de medidas obtenido en la iteración actual y, si la hubiese, la actuación aplicada en el acto.

Como todo filtro bayesiano, se conforma por dos partes bien diferenciadas. La primera correspondiente a la predicción, y la segunda a la actualización de medida.

La predicción se nutre principalmente de la definición paramétrica de la función lineal de transición de estados. Se describe la media predicha como la evolución esperada según el modelo del sistema y la media anterior, acompañada de una posible acción aplicada, por ejemplo, mediante los actuadores del robot. Y como covarianza predicha, esa misma evolución mencionada aplicada esta vez a la matriz de covarianzas previa, más la incertidumbre intrínseca del modelo del sistema aplicado. Esta incertidumbre del modelo del sistema suele usarse como parámetro de ajuste en la estimación, ya que concretar su valor puede suponer una ardua tarea.

En el Filtro de Kalman, se presenta un parámetro característico de la actualización de medida y del propio filtro, llamado Ganancia de Kalman (K_t). Se trata de una matriz dependiente del modelo de observación, la

matriz de covarianza predicha y de la matriz de incertidumbre de las medidas. Se función es ponderar, según la confianza de la predicción y la medida, una fusión sensorial entre los distintos tipos de medidas. Sigue un método similar al de máxima verosimilitud.

Por otro lado, la actualización de medida pretende ajustar la predicción de la estimación propuesta previamente según las medidas y la realimentación del estado que pueda obtener, normalmente mediante uso de sensores. Para ello, se concluye la media estimada como la predicha más la Ganancia de Kalman, por la diferencia entre la medida recibida y la medida esperada. Se denomina medida esperada por ser el valor que de alguna forma se espera medir según el modelo de observación y la media predicha. Esta diferencia que multiplica a la ganancia se denomina *innovación*. En cuanto a la matriz de covarianza estimada, el procedimiento es similar al descrito para la media.

En el caso de no obtenerse medida, no se llevará a cabo la actualización de medida, debido a esa falta de información. En su lugar, se tomará como estimación, en ese instante, las predicciones plantadas en la primera parte del algoritmo.

Este algoritmo se considera bastante eficiente computacionalmente, aunque no es sencillo de paralelizar si se quiere dividir costes de computo. Se trata de un algoritmo óptimo.

2.2.2 Filtro de Kalman Extendido (EKF)

El Filtro de Kalman asume que las funciones utilizadas son lineales, pero esto no siempre puede cumplirse, por ejemplo, al describir una trayectoria circular o medir una distancia.

Una forma de contraponer dicha limitación es mediante el Filtro de Kalman Extendido [16]. Este filtro suaviza dicha suposición y permite el uso de funciones no lineales, las cuales suelen ser las que gobiernen las probabilidades de transición de estados y de medidas. La estructura del vector de estados y medida puede verse de la siguiente forma con el uso de las funciones no lineales g y h:

$$x_t = g(u_t, x_{t-1}) + \varepsilon_t \tag{2.14}$$

$$z_t = h(x_t) + \delta_t \tag{2.15}$$

Esta modificación da lugar a distribuciones no lineales y no gaussianas. Estos efectos no permiten el uso del Filtro de Kalman, ya que incumplen suposiciones importantes para su implementación. La solución que aporta el Filtro Extendido de Kalman es la aproximación gaussiana de la distribución resultante anterior. Por ello, su mayor diferencia con el KF será el uso de distribuciones aproximadas debido a las no linealidades que se presentan.

Aunque el EKF pretende conseguir dichas aproximaciones con gran precisión, esto no es posible al no poder conseguir realizarse cálculos exactos por las no linealidades, lo que añade una aproximación adicional. Por tanto, para aproximar las distribuciones no lineales, el EKF se centrará en el objeto de la linealización de las mismas. Se basa en aproximar la función g, y h, como una función lineal tangente a la misma en la media de la gaussiana que mejor se ajuste a la distribución no lineal.

Una vez se linealizan las funciones remarcadas, se obtiene la equivalencia del EKF con el KF, describiéndose las demás cualidades y características del filtro de forma similar.

Existen numerosas técnicas para linealizar funciones no lineales. En concreto, el EKF opera mediante el método de *expansión de Taylor* (de primer orden). Este método consiste en construir una aproximación lineal de la distribución a tratar mediante el propio valor de la función y su pendiente, donde se considera como pendiente el conjunto de las derivadas parciales de la función respecto a cada variable de estado.

$$g(u_t, x_{t-1}) \simeq g(u_t, \mu_{t-1}) + g'(u_t, \mu_{t-1}) \cdot (x_{t-1} - \mu_{t-1}) \tag{2.16}$$

donde,

$$g'(u_t,\mu_{t-1}) = \frac{\partial g(u_t,\mu_{t-1})}{\partial \mu_{t-1}} = G_t \tag{2.17}$$

En la relación anterior se observa G_t, conocido como Jacobiano de la distribución de transición de estados, el cual depende de u_t y de μ_{t-1}.

En el caso de la función no lineal h se sigue el mismo procedimiento, haciendo uso esta vez de H_t, el Jacobiano de la distribución de medida (dependiente de la predicción $\bar{\mu}_t$).

$$h(x_t) \simeq h(\bar{\mu}_t) + h'(\bar{\mu}_t) \cdot (x_t - \bar{\mu}_t) \tag{2.18}$$

donde,

$$h'(\bar{\mu}_t) = \frac{\partial h(\bar{\mu}_t)}{\partial \bar{\mu}_t} = H_t \tag{2.19}$$

Una vez descritas estas diferencias entre el KF y el EKF ya es posible componer el algoritmo de este último, donde se sustituyen los modelos matriciales A y B en predicción de estado por la función no lineal g, y donde el producto de esa misma predicción por la matriz de observación C se sustituye por h, en la etapa de predicción de medida.

Algoritmo

Filtro de Kalman Extendido $(\mu_{t-1},\Sigma_{t-1},u_t,z_t)$**:**

$$\bar{\mu}_t = g(u_t,\mu_{t-1}) \tag{2.20}$$

$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^\top + R_t \tag{2.21}$$

$$K_t = \bar{\Sigma}_t H_t^\top (H_t \bar{\Sigma}_t H_t^\top + Q_t)^{-1} \tag{2.22}$$

$$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t)) \tag{2.23}$$

$$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t \tag{2.24}$$

return μ_t,Σ_t

Las predicciones lineales se sustituyen por generalizaciones no lineales en el EKF. Esto se denota en el uso de Jacobianos, como G_t y H_t, en lugar de matrices lineales como A_t, B_t y C_t.

Gracias a las aproximaciones y modificaciones ejercidas sobre el KF, el EKF cubre un rango de problemas de estimación de una extensión muy superior a la que el KF es capaz de incorporar. Esto ha hecho que el EKF se convierta en una de las herramientas más populares en la estimación de estados en robótica. En parte, esto también se debe a su simplicidad para la implementación y su eficiencia computacional, todo respaldado por el uso de distribuciones gaussianas multivariable.

A pesar de ello, el EKF también cuenta con importantes limitaciones en determinadas ocasiones debido a esa misma característica, el uso de aproximaciones en transición de estados o el uso de la *expansión de Taylor*. Estos fenómenos ocurren en mayor medida cuanto mayor sea la incertidumbre de la gaussiana que aproxima la distribución no lineal, aunque la linealiación no dependa de ello. El resultado es una distorsión en la estimación que se ve reflejada en una precisión inferior de la misma.

Otro punto en el que la calidad de la estimación puede ser reducida se debe a que se linealiza localmente entorno a la media de la gaussiana. Al no ser una función lineal, dicha media puede variar significativamente, y cambiar el punto de linealización de forma considerable. Si se observan situaciones del estilo, pueden ser paliadas con el uso de una ampliación del EKF, donde se mezclan gaussianas de distintas hipótesis a tener en cuenta, lo que se denomina EKF de multi-hipótesis.

2.3 Sensores

Los sensores son dispositivos electrónicos capaces de dotar con la capacidad de obtener información del medio a los robots que hacen uso de los mismos. Esto se traduce en habilidades tales como detectar objetos a distancia, percibir sonidos, olores e incluso temperatura. En definitiva, su función puede asimilarse a la de los sentidos para los seres humanos, siendo por esto que cuando se posee un conjunto de sensores en un mismo robot [2] se habla de sistema sensorial.

El funcionamiento de un sensor se basa en adquirir información de alguna magnitud física o química, transformando su interpretación de la misma a valores electrónicos capaces de ser procesados o cuantificados por algún microcontrolador, o simplemente ser facilitados para su observación.

Para ello, utilizan los denominados transductores, los cuales convierten las variaciones de una magnitud física en variaciones de una magnitud eléctrica, y un acondicionamiento de la señal apropiado, encargado de modificar la señal eléctrica de forma adecuada a su procesamiento.

Como todo en la robótica, han ido evolucionando y adquiriendo mayor sofisticación y capacidad de precisión en sus medidas, llegando en algunos casos a resultados de un gran nivel. Y siguiendo el efecto de cadena, esto ha desembocado en el uso de robots en nuevas aplicaciones o la mejora de muchas otras, donde previamente por falta de recursos o resultados no se podían utilizar.

Una de las mejoras más significativas en los sensores en las últimas décadas, además del aumento de la precisión de las medidas, es la capacidad de aportar dicha medida en un tiempo casi instantáneo, que junto con la capacidad de procesar dicha información a la par en los sistemas de procesamiento de los robots, hace que las aplicaciones en tiempo real hayan demandado cada vez más el uso de la robótica. Estas aplicaciones en tiempo real tienen un gran interés de uso en robots, ya que en el mundo real, la mayoría de los acontecimientos y procesos ocurren de forma ininterrumpida, como por ejemplo la acción de esquivar un objeto esporádico por un robot móvil.

En la actualidad, existe una gran variedad de sensores en el mercado, con diferentes aplicaciones y usos. Aplicado a la robótica podrían enumerarse una buena cantidad de los mismos, para ello se recurre a los elementos típicos en la materia; estos pueden ser sensores de distancia, de velocidad, de inclinación, de temperatura, de sonido o de luz entre tantos.

Dentro de la robótica, el uso de los sensores se extiende desde su aplicación en sistemas de control, con función de obtener medidas de realimentación, hasta su labor en sistemas de percepción y aprendizaje, donde son la base del propio sistema. Además de estos y muchos otros usos, también pueden extenderse a medios informáticos o a componer sistemas de resolución de problemas de alta complejidad.

Haciendo referencia a la información obtenida de los sensores en robótica, estos pueden clasificarse en dos grupos. Si las señales que reciben estos sensores proceden de señales internas del propio robot, como orientación o carga de batería, se habla de *sensores propioceptivos*, mientras que si las señales proceden del exterior del robot, como la distancia de un objeto, se denominan *sensores exteroseptivos*.

A la hora de tener en cuenta la integración de un sensor en robótica, varias de sus características pueden ser clave en su desempeño. En función de la aplicación a desarrollar por el sensor, la velocidad de operación, la tasa de error, robustez, resolución y sensibilidad pueden marcar la diferencia entre unos resultados satisfactorios o no tan buenos. Aunque estas características no son las únicas a tener en cuenta, puesto que muchos otros factores también deben entrar en consideración, como lo son el coste, el peso o tamaño, los requerimientos computacionales que demanda, o el ruido y efectos de distorsión que puedan introducir.

En cuanto a las posibles aplicaciones que los sensores pueden tener en robótica se pueden enumerar como: sensores de desplazamiento lineal o rotativo (p.e., potenciómetros, encoders), sensores de presencia o proximidad (p.e., finales de carrera, sensores inductivos o capacitivos, sensores de ultrasonido, infrarrojos), sensores de navegación (p.e., giroscopios, GPS, receptores de radiobalizas), sensores de visión artificial (p.e, cámaras), y sensores para muchas otras aplicaciones (p.e., sensores de sonido, presión o temperatura).

En especial, se considerarán los receptores de radiobalizas, la cámara y la IMU.

El primero de ellos, los receptores de radiobalizas, se tratan de sensores de rango enfocados en la navegación en robótica, con el fin de obtener información suficiente para la localización del robot en el entorno.

Estos sensores se componen de emisores de radiofrecuencia codificados, los cuales se ubican en lugares concretos a modo de balizas, marcando así una posición conocida en el entorno. Por otro lado, en el robot móvil se aloja una antena receptora de las señales radio emitidas, y un sistema capaz de decodificarlas y obtener la información de la distancia a la que se encuentra cada emisor radio.

A partir de las medidas de distancia entre emisores y receptor, ya sea por triangulación de las mismas o integradas en un sistema de estimación como los filtros descritos, es posible obtener la posición y localización del robot en el espacio con una precisión considerable. El receptor debe poder detectar señales en cualquier dirección, aunque siempre existen limitaciones de rango máximo o de interferencias. Estos sensores suelen ser bastante económicos y de un tamaño bastante reducido. Una de las bandas típicas de emisión esta entre 300 y 400 MHz.

En el caso de la cámara, su uso como sensor en robótica se ha extendido considerablemente a raíz de la visión artificial en robots.

Su función es la de captar imágenes del entorno, haciendo uso del espectro electromagnético, y posteriormente realizar algún procesamiento de las mismas para adquirir información de ellas. Estas imágenes suelen tomarse en dos dimensiones, trabajando en el espectro visible o infrarrojo. A partir de ellas, y con algoritmos de procesamiento de imágenes es posible captar formas, detectar patrones, eliminar ruidos y distorsiones, compensar la iluminación e incluso realizar representaciones tridimensionales.

De acuerdo a esta última aplicación, se han desarrollado múltiples variantes de la cámara convencional o modelos RGB, para obtener información de profundidad directamente, como las cámaras RGB-D.

Por último, la Unidad de Medición Inercial (IMU) [4] se trata de un conjunto de sensores, del tipo propioceptivos, capaz de medir la aceleración y rotación del robot mediante acelerómetros y giróscopos.

Este tipo de sensores suelen utilizarse en combinación de otros para aumentar la precisión de estimación de magnitudes como la posición o actitud de un robot, normalmente aéreo.

Las IMUs son capaces a partir de sus mediciones de obtener magnitudes tales como la orientación, velocidad, y fuerzas gravitatorias. Dichos reportes son de gran ayuda en aplicaciones de navegación y control. Además de lo mencionado, en aplicaciones con UAVs, la IMU es de gran importancia debido a la necesidad de que estos operen con el máximo control y estabilidad posible. Para ello es necesario una IMU con un buen rendimiento, resistente a vibraciones y estable con las temperaturas.

La integración de todos estos tipos de sensores en robots móviles, como los aéreos, se consigue en muchos de los casos mediante fusión sensorial. La fusión sensorial comprende la integración de las medidas que aportan los sensores o la información que de ellas se obtiene. En general, este procedimiento es realmente útil en problemas de estimación, donde los resultados con fusión sensorial son mucho más precisos que con el uso de algún sensor simplemente. No solo se mejora la precisión de la estimación y reduce los posibles errores cometidos, sino que dota al sistema global de una robustez mayor, al no depender de un solo sensor.

3 Diseño del método

3.1 Introducción

En este capítulo se describirá el trabajo a realizar y los conceptos que se aplicarán al mismo.

Como se ha comentado en el primer capítulo, el objetivo principal a desarrollar se centrará en la implementación de un algoritmo de estimación basado en EKF. Este algoritmo busca ser aplicado en la localización de un robot aéreo de inspección en entornos sin acceso a GNSS.

Partiendo de estas premisas, se buscarán las posibles soluciones y alternativas que pongan en pie el método a desarrollar, con los mejores resultados y asegurando su funcionalidad.

Además, se pretende analizar y caracterizar las distintas secciones del desarrollo, con el fin de obtener sus puntos fuertes y las limitaciones u obstáculos que se presenten.

En una primera instancia, se descompondrán los distintos elementos a tratar entorno a la localización del robot aéreo, y los cuales se van a considerar como objeto de estudio. Continuado por la inclusión del algoritmo EKF en mayor profundidad y las distintas partes que lo componen.

3.2 Localización general

Para localizar un robot aéreo mediante algoritmos de estimación, como el EKF, es necesario proveer al mismo con algún sensor capaz de aportarle información acerca de su estado en su entorno.

En el caso concreto de estudio, se presenta un robot aéreo, concretamente un quadrotor, incapaz de utilizar sensores de tipo GPS (Figura 3.1) debido a su aplicación concreta, inspección de túneles o entornos privados de geolocalización.

El GPS (*Global Positioning System*), es uno de los sensores más utilizados en problemas de navegación con robots móviles. Y ya sea como sensor principal, o como complementario, su función es de gran ayuda en estimadores o algoritmos de localización. El hecho de no poder contar con sus servicios, hace que se deban buscar nuevas alternativas en cuanto al tema sensorial.

La alternativa por la que se ha optado es sin duda la fusión sensorial de varios sensores en conjunto, aprovechando así las propiedades que aporta un filtro como el EKF. Los sensores usados serán una IMU, una cámara, y sensores de rango del tipo balizas radio. También puede mencionarse el uso de sensores del tipo LIDAR, muy comúnmente utilizados en aplicaciones de localización en interiores o problemas de evitación de obstáculos.

El uso del quadrotor, explicado en capítulos previos, venía acompañado de un robot móvil terrestre como centro de mando, al cual este estaba conectado figuradamente. Aprovechando dicho contexto, el problema de localización a resolver vendrá de forma relativa a la posición del robot terrestre mencionado. Es decir,

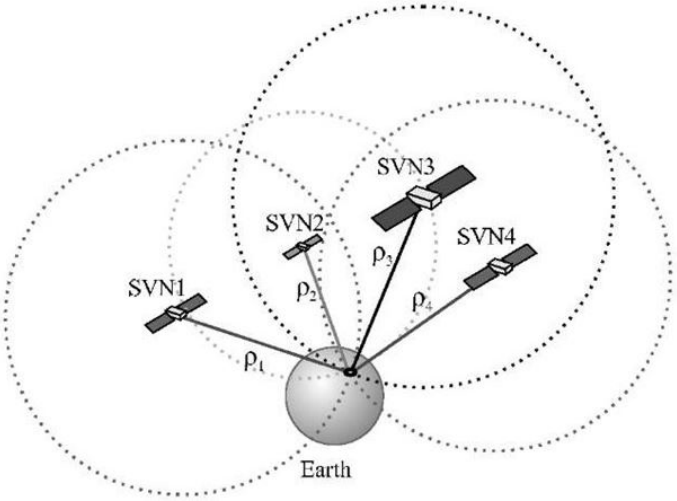

Figura 3.1 Modelo GNSS mediante satélites [9].

se partirá en todo momento de unas coordenadas globales impuestas por el vehículo terrestre y no por el entorno, que a priori, será desconocido para el robot aéreo.

Por tanto, se presupone conocida la posición del robot terrestre en vista del quadrotor, en el cual se debe aplicar el algoritmo de localización relativa. Para el análisis del problema, se tomará siempre la posición del vehículo terrestre como de coordenadas (0,0,0), y se supondrá que durante la actuación del quadrotor, dichas coordenadas serán estáticas.

Sabiendo esto, es lógico usar el robot terrestre como punto de referencia para colocar las balizas radio, las cuales actuaran como emisoras de señales de rango. Por otro lado, en el quadrotor se encontraría el receptor de dichas señales. De este modo, es fácil obtener cada vez que se realice una recepción de señales radio la distancia a la que se encuentra el emisor del que provengan las mismas.

Para la inclusión de estas medidas en el EKF es acertado suponer que cada emisor envía una señal de radio con una codificación distinta, pudiendo conocerse, además de la distancia a la que se encuentra, una identificación del emisor concreto del que se trata. Esto es de gran ayuda para su implementación en el filtro, ya que cada baliza se encuentra en una posición distinta con el objetivo de enriquecer la información que toma el filtro.

Por otro lado, se cuenta con una IMU (*Inertial Measurement Unit*) a bordo del quadrotor. Este tipo de sensor, compuesto de acelerómetros y giróscopos, es capaz de aportar varias medidas internas de interés, como la aceleración u orientación del robot aéreo. A partir de estos datos, y mediante algún procesamiento, es posible obtener de la IMU (Figura 3.2) las velocidades lineales del quadrotor. Estas últimas serán las medidas a considerar en la localización del robot aéreo mediante estimación.

De forma precisa, serán los incrementos de velocidad relativa del robot, detectados por la IMU, los aportes que se incluirán en la configuración del EKF.

El robot aéreo en base de estudio se dotará además con una cámara fija (Figura 3.4). Con este sensor, y aprovechando el posicionamiento de las balizas de forma dispersa sobre el vehículo terrestre, se detectarán en sus imágenes el posicionamiento de cada una de ellas. Se supone además, que gracias a tecnologías emergentes como *ArUco* [12], el robot sea capaz de identificar, igual que con el uso de las radiofrecuencias, la identificación concreta de cada baliza en las imágenes adquiridas.

ArUco se trata de una tecnología desarrollada recientemente aplicada a identificación de marcadores. Como se observa en la Figura 3.3, se trata de marcadores cuadrados sintéticos compuestos por un borde negro, que

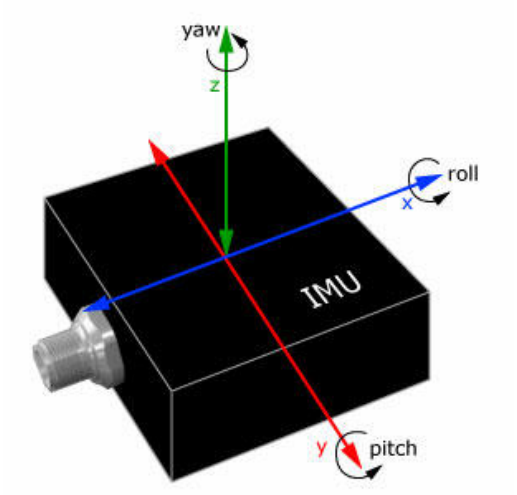

Figura 3.2 IMU [1].

facilita su detección visual, y que contienen en su interior una matriz binaria con la codificación individual de cada uno. Dichos marcadores suelen ser de 4x4, portando por tanto 16 bits de información. Su uso está dirigido principalmente a aplicaciones de detección e identificación en imágenes o calibración y corrección de errores.

Una vez obtenidas imágenes de las balizas en la cámara provista, e identificando individualmente dichos marcadores, esta información medida en píxeles por la cámara es apta para su incorporación al filtro, y con la cual podrá obtenerse la estimación de posición del quadrotor.

Al tratarse de una cámara fija, se puede intuir que la modificación en la orientación del robot aéreo haría que la cámara dejase de enfocar las balizas y obtener información de las mismas. Esto puede llagar a ser una gran limitación en cuanto al uso de este sensor, que se analizará posteriormente.

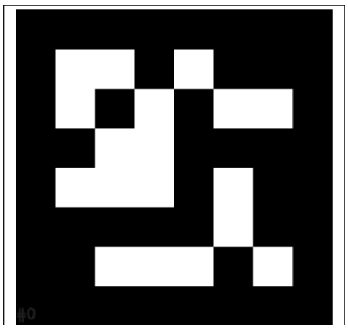

Figura 3.3 ArUco Marker [6].

Figura 3.4 Quadrotor con cámara.

En cuanto al filtro y sus componentes, habiendo ya definidos estos elementos y las suposiciones a tener en cuenta, puede darse paso a su descripción y diseño en detalle.

3.3 EKF - Predicción

La primera parte del filtro es la predicción, o actualización de control. En esta sección se parte de los resultados de la iteración previa del filtro, es decir, la estimación anterior del vector de estados, y la matriz de covarianzas estimada previa.

Partiendo de las expresiones (2.20) y (2.21), se desarrollará el apartado de predicción del filtro implementado. La primera de ellas, correspondiente a la predicción del estado, se describe para una distribución de transición de estados no lineal.

La definición de la distribución de estados en el problema a tratar se centra en la función que determina del modelo del robot aéreo. Por simplicidad, este modelo se toma como lineal, asumiendo los errores que proceden. Por tanto, se usará como modelo el de movimiento rectilíneo uniforme, el cual describirá el movimiento aproximado del quadrotor.

Siguiendo esta tónica se deduce con simplicidad que el vector de estados a considerar lo conformarán únicamente la posición y velocidad del robot en los tres ejes espaciales. De esta forma, con el modelo aplicado, la relación entre estado actual y previo es inmediata.

Por ello, el vector de estados y el modelo del sistema se describen linealmente de acuerdo a la expresión (2.9). Se considera T como el período de ejecución de las iteraciones del algoritmo.

$$x_t = [x, y, z, v_x, v_y, v_z] \tag{3.1}$$

$$A = \begin{bmatrix} 1 & 0 & 0 & T & 0 & 0 \\ 0 & 1 & 0 & 0 & T & 0 \\ 0 & 0 & 1 & 0 & 0 & T \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

Este modelo incorpora de forma matricial la ecuación del MRU para la obtención de la posición a partir de la velocidad. A su vez se asume una velocidad constante e invariante, factor que no se corresponde con la realidad descrita.

Para paliar los errores que se cometen a la hora de definir el modelo del sistema y la incertidumbre que conlleva no conocer con mayor precisión el mismo, se recurre a la matriz de covarianza del modelo. Esta matriz se diseñará de forma que albergue los posibles errores e incertidumbres mencionados, a forma de parámetro de ajuste del modelo y su bondad.

El diseño de la matriz de incertidumbres del modelo se realiza mediante un único parámetro, la desviación típica que se le quiera asignar. Con este parámetro, r, puede representarse fácilmente como una matriz diagonal con autovalores r^2.

$$R = \begin{bmatrix} r^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & r^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & r^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & r^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & r^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & r^2 \end{bmatrix} \tag{3.3}$$

Tanto A como R se tratan de matrices cuadradas de 6 dimensiones, correspondientes a las componentes del vector de estados. Es decir, a modo de ejemplo la primera fila de A se corresponde con la expresión que relaciona la componente inicial del vector de estados y su valor previo. Esto, además, brinda la posibilidad de asignar una cota de incertidumbre variable a cada componente del vector de estados. En función del modelo propuesto, existen versiones donde la suposición de velocidad constante e invariable se trata con mayor peso

que el diseño del MRU para la posición.

Atendiendo a los elementos descritos ya puede ser compuesto el algoritmo del predicción del EKF, haciendo uso únicamente de la predicción en función del modelo. Dicho de otra forma, puede computarse la predicción del vector de estados en base a sus valores previos y el modelo del sistema, con el añadido de las incertidumbres que lo acompañan.

$$\bar{x}_t = A \cdot x_{t-1} \tag{3.4}$$

$$\bar{\Sigma}_t = A \cdot \Sigma_{t-1} \cdot A^\top + R \tag{3.5}$$

Este sería el algoritmo completo para la predicción en ausencia de señales de actuación del tipo u_t, como se describe en la expresión (2.9). Este tipo de señales pueden ser generadas por la aplicación de una fuerza o movimiento en el robot, bien con un actuador propio del robot o externo. También pueden considerarse como actuaciones los movimientos que no quedan contemplados en el normal desarrollo del modelo del sistema, pero que pueden ser medidos. Este último es el caso a desarrollar a continuación.

El quadrotor utilizado posee como sensor una IMU, capaz de obtener diversos datos internos del robot tales como aceleraciones y orientación. A partir de estos datos, mediante algún procesamiento en un sistema de navegación inercial, es posible obtener la velocidad lineal que posee el robot aéreo en cada iteración. La diferencia de esta medida respecto a la velocidad estimada en el instante previo dan lugar al incremento o decremento de velocidad que es capaz de medirse en cada ciclo del algoritmo.

Como todo sensor, los datos aportados por la IMU, además del cómputo intermedio, adquieren una cierta incertidumbre que hace que no pueda darse una medida totalmente exacta. Al incluirse dicha diferencia de velocidad descrita como señal de actuación en predicción, las posibles incertidumbres que conlleva la medida de la IMU se consideran en la propia matriz de incertidumbres del modelo del sistema (R).

Una vez obtenidos estos incrementos o decrementos de velocidad, pueden añadirse como vector u_t, el cual irá precedido por una matriz de actuación B. En este caso, al tratarse de velocidades lineales, tales como las que posee el vector de estados, B coincidirá con la matriz A. Esto se cumple siempre y cuando u_t posea las mismas componentes que x_t, es decir, añadiendo las componentes de la posición como nulos.

$$u_t = [0, 0, 0, v_x^{IMU} - v_x^{k-1}, v_y^{IMU} - v_y^{k-1}, v_z^{IMU} - v_z^{k-1}] \tag{3.6}$$

De esta forma, se adquiere una mayor precisión en la predicción gracias a la información que introduce la IMU a forma de corrección del modelo, como si de actuaciones se tratase.

El modelo completo de predicción se corresponde a las siguientes expresiones.

$$\bar{x}_t = A \cdot x_{t-1} + A \cdot u_t \tag{3.7}$$

$$\bar{\Sigma}_t = A \cdot \Sigma_{t-1} \cdot A^\top + R \tag{3.8}$$

Partiendo de un modelo de predicción de este nivel, las estimaciones generadas, sin contar con la segunda fase del EKF, ofrecen ya la capacidad de dar un resultado bastante acertado en el marco de movimientos que ejecuta un quadrotor. Esto se debe a que la incorporación de la IMU en conjunto con el modelado propuesto dotan de mayor robustez y precisión las predicciones del vector de estados.

3.4 EKF - Actualización

En la segunda parte del algoritmo del EKF, partiendo de los resultados de la fase de predicción desarrollada, se procede a la actualización de medida.

Además de las predicciones obtenidas, deben aportarse para el correcto funcionamiento de esta sección, las medidas provenientes de los sensores externos que posee el robot aéreo. Los sensores de los que se tiene uso son la cámara fija y las balizas radio.

Las balizas radio aportarán medidas de las distancias a las que cada una se encuentra respecto al receptor radio ubicado en el quadrotor. Estas medidas pueden ir acompañadas o no de información direccional (VOR o NDB respectivamente), como se muestra en la Figura 3.5. La relación de estas medidas con las variables de estado son por tanto funciones no lineales, y que no podrán ser tratadas con la estructura del KF de igual forma que la predicción. Se recurre a las expresiones empleadas en el EKF (2.22), (2.23) y (2.24).

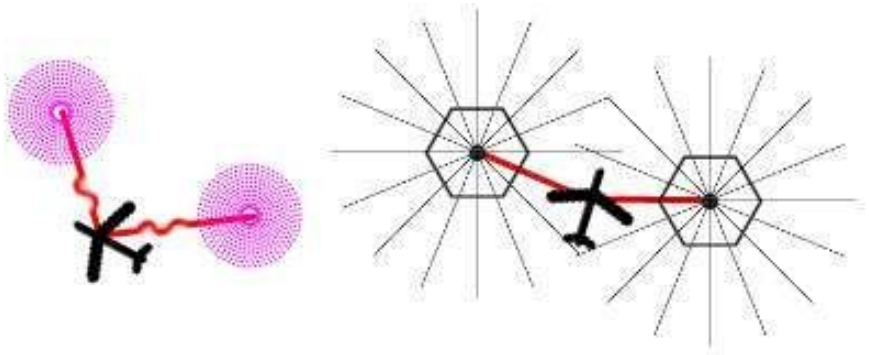

Figura 3.5 Funcionamiento radio balizas tipo: a)NDB, b)VOR [18].

Concretamente, se tiene una cantidad de 4 balizas radio colocadas estratégicamente sobre el robot terrestre, de forma que aporten la mayor información posible en la recepción de sus medidas. Esto se debe a que la riqueza de posicionamiento en todos los ejes espaciales influye en el uso de la fusión sensorial del robot mediante filtros. En el caso de que todas las balizas se encontrasen a la misma altura, la diversidad de medidas en dicho eje no sería de mucho interés por la poca información distinta que añade, lo que se traduce en un incremento de la dispersión de la incertidumbre sobre el eje descrito.

En la práctica, las señales que envían al receptor pueden contener fallos de codificación o verse corrompidas por interferencias y fenómenos externos. Esto sugiere la posibilidad de adaptar el problema, de forma que el filtro sea capaz de procesar cada señal radio recibida de forma individual, dotando de mayor robustez al sistema de estimación. Otro de los motivos por el que podría dejar de obtenerse medidas de este tipo se debe a la posibilidad de salir del rango máximo, donde las señales emitidas no tienen potencia suficiente de propagación, llegando a la misma conclusión previa.

Retomando las expresiones de la actualización, se inicia con el cálculo de la ganancia de Kalman. Este parámetro depende de la matriz de covarianza predicha, la matriz de observación H y la matriz de incertidumbres de esta última, Q. Por tanto, para obtener las dos últimas se tiene el siguiente desarrollo y consideraciones.

La matriz de observación modela la relación entre las medidas tomadas y el vector de estados, mientras que Q, imita a la matriz R, en relación específica a las incertidumbres provenientes de las medidas. A modo de resumen, H portará una expresión por cada medida recibida que la relacionará con las componentes del vector de estados, mientras que Q, almacena las incertidumbres de cada medida en cada autovalor de su diagonal.

Para el caso de las balizas radio, H debe relacionar la medida de la distancia recibida con cada componente de posición y velocidad del vector de estados. Esto da lugar a una relación no lineal que conformará a H como un Jacobiano. Por otro lado, mediante las especificaciones de las balizas puede obtenerse la desviación típica de la medida. Este parámetro se corresponderá, elevado al cuadrado, a un autovalor de la matriz Q,

siendo además indicativo de cómo de dispersas pueden ser las medidas entorno a la magnitud real.

$$d = \sqrt{((x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2)} \tag{3.9}$$

Para el cálculo del Jacobiano, se parte de la expresión de la distancia entre dos coordenadas mediante el módulo. La expresión (3.9) se derivará parcialmente respecto a cada componente del vector, dando como resultado las componentes de cada fila de la matriz de observación. Es decir, se obtiene una fila del Jacobiano por cada medida distinta a considerar.

$$H = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ g_{m1} & g_{m2} & \cdots & g_{mn} \end{bmatrix} \tag{3.10}$$

$$g_{ij} = \frac{\partial f_i}{\partial x_j} \tag{3.11}$$

En las expresiones anteriores se refleja con mayor detalle lo explicado previamente. La matriz H poseerá tantas columnas como componentes tenga el vector de estados, n; por otro lado, las filas dependerán de cuantas medidas se obtengan, m. Cada medida se representa por una función, f_i, capaz de relacionar dicha medida con las componentes del vector de estados. En el caso de las balizas radio la función se trata de la distancia, dada por (3.9).

Como se ha mencionado, no se asegura la recepción de todas las medidas de distancia en cada iteración, es por esto que H y Q serán variables en cada ciclo. Variarán el número de filas, en el caso de H_t, y las dimensiones totales, en el caso de la matriz diagonal Q_t. A modo de ejemplo, si se recibiesen cuatro medidas de distancia de las balizas radio, con una desviación típica de la medida q_1, la matriz de incertidumbre de actualización tendría la siguiente forma.

$$Q_t = \begin{bmatrix} q_1^2 & 0 & 0 & 0 \\ 0 & q_1^2 & 0 & 0 \\ 0 & 0 & q_1^2 & 0 \\ 0 & 0 & 0 & q_1^2 \end{bmatrix} \tag{3.12}$$

Tras describir los modelos de observación y su aplicación con las balizas radio, se pasará a incorporar la cámara fija. Inicialmente se describirá su modelo, en base a la Figura 3.6.

La cámara fija se diseñará en función del modelo de cámara estenopeica o "pinhole". Este es un modelo que describe matemáticamente la relación entre las coordenadas de un punto en el espacio con su representación en dos dimensiones proyectada en la imagen obtenida. Para los siguientes desarrollos se considerará el modelo ideal, libre de toda distorsión o posible error por el modelado.

Para su diseño han de definirse sus parámetros intrínsecos, propios de la cámara. Estos parámetros se han tomado en torno a valores de un modelo genérico, aunque de ser necesario pueden ser modificados para variar sus prestaciones.

Partiendo de un sistema de referencia global, supuesto en el robot terrestre, y considerando la disposición de la cámara en el quadrotor y en relación al mismo, se definen inicialmente los siguientes elementos externos.

El primero de ellos es la matriz de transformación del sistema de referencia global al de la cámara, ${}^wT_c$. Se trata de una matriz compuesta por la matriz de rotación total de los ejes,${}^wR_c$, y el vector de traslación entre los sistemas de referencia,${}^wt_c$.

$${}^wT_c = \begin{bmatrix} [& {}^wR_c &] & [{}^wt_c] \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.13}$$

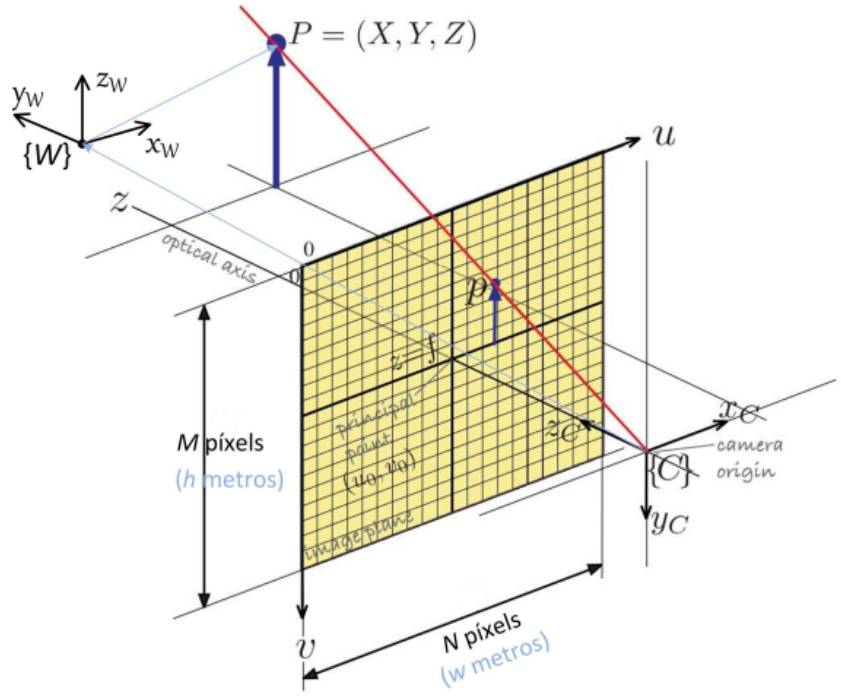

Figura 3.6 Modelo completo de la cámara [17].

De la expresión anterior se deducen las dimensiones de los elementos, donde ${}^{w}R_{c}$ debe ser una matriz cuadrada $3x3$ que incluya las rotaciones necesarias, y ${}^{w}t_{c}$ un vector columna $3x1$ que aporte el desplazamiento entre los sistemas descritos. De esta forma, la relación entre la cámara y el robot terrestre queda definida.

En cuanto a los parámetros intrínsecos de la cámara, será necesario conocer la distancia focal (f), la resolución en píxeles de la imagen resultante (NxM, ancho por alto), el tamaño del sensor (wxh), el punto principal o central de la imagen (u_0,v_0) y el "skew" (s), aunque este último parámetro se tomará como nulo.

Una vez definidos dichos parámetros, es posible obtener las dimensiones efectivas del sensor en m/pix, y de ahí, la distancia focal efectiva en píxel. Con esta última característica puede formarse la matriz K, tal como en (3.16).

$$\rho_x = w/N; \;\; \rho_y = h/M \tag{3.14}$$

$$f_x = f/\rho_x; \;\; f_y = f/\rho_y \tag{3.15}$$

$$K = \begin{bmatrix} f_x & s \cdot f_x & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.16}$$

Esta matriz incluye como información la parametrización y caracterización de la cámara, por ello, recibe el nombre de matriz de parámetros intrínsecos.

La medida que aporta la cámara utilizada, y que se usará como dato en la actualización de la medida, son las coordenadas en píxeles de puntos en el espacio captados en la imagen obtenida. Estos puntos no serán otros que las propias balizas radio alojadas en el vehículo terrestre, que mediante técnicas de procesamiento de imagen y tecnología de índole similar a *ArUcos* pueden ser identificadas individualmente en cada *frame* analizado.

Este supuesto tiene como condicionante obtener una medida en función del sistema local de la cámara y no de forma global. Por ello, una de las medidas requeridas para el modelado es invertir la matriz de

transformación de sistemas referencias. A partir de ${}^{w}T_{c}$, calculando su inversa, se obtiene ${}^{c}T_{w}$. Esta matriz muestra la transformación de la cámara al sistema global, con sus respectivas submatrices ${}^{c}R_{w}$ y ${}^{c}t_{w}$

Por último, la posición exacta de cada baliza se presupone conocida, al estar ubicadas sobre el robot terrestre de forma fija. Es decir, se conocen sus coordenadas en relación a la referencia global que marca el robot terrestre. Con estos puntos y los elementos definidos es posible determinar el modelo completo de la cámara empleada, que además se corresponderá con las funciones no lineales que relacionan las medidas aportadas con el vector de estados del filtro.

$$p = K \cdot [{}^{c}R_{w} \;\; {}^{c}t_{w}] \cdot {}^{w}P \tag{3.17}$$

Este modelo [3] proporciona los puntos en píxeles de la imagen en coordenadas homogéneas. Lo que atribuye la posterior obtención de la coordenada X e Y mediante normalización con la tercera componente. Las coordenadas del punto en el espacio también deben ser homogéneas, siendo necesario añadir una cuarta componente unidad.

A partir de este modelo pueden relacionarse las coordenadas X e Y en píxeles de la baliza que se muestra en una imagen (las medidas recibidas) con las coordenadas X, Y y Z en metros de la baliza en el entorno, supuestamente conocidas. Estos dos elementos pueden verse de forma gráfica en la Figura 3.6, donde p se encuentra contenido en el plano de la imagen.

Ha de tenerse en cuenta que al igual que ocurría con la recepción de medidas de distancia con las balizas radio, si una de ellas sale del campo de visión de la cámara, esa medida en píxeles no puede ser tomada. Por ello, también se debe priorizar la inclusión selectiva de únicamente los puntos visibles en la imagen. Esto se traduce en añadir a la matriz Jacobiana H_t, además de las medidas radio recibidas, filas según los puntos captados en cámara. Además, también deberá tenerse en cuenta el aumento de una dimensión de la matriz Q_t por cada punto dentro del FOV (*Fill Of View*) de la cámara.

Es de fácil apreciación, debido al dinamismo y las dimensiones de los elementos de cálculo de la parte de actualización del filtro, además de los posibles cálculos simbólicos que pueden emplearse para los Jacobianos, considerar dicha parte del algoritmo la de mayor carga computacional.

Concretamente, suponiendo obtener las medidas en distancia de las cuatro balizas empleadas y las coordenadas X e Y de cada una de ellas, en caso de verse todas incluidas en el FOV, las dimensiones de la matriz Q_t alcanzaría ser de 12x12. Para el resto de elementos los ordenes de magnitud rondarían valores similares, lo que en conjunto supone un mayor coste de cómputo.

Retomando la estructura de diseño del filtro, una vez definidos los elementos y sus dimensiones en cada iteración según las medidas obtenidas, puede integrarse el conjunto según rige el algoritmo del EKF. En base a las expresiones (2.22)-(2.24), y los elementos empleados, se obtienen el siguiente diseño de la parte de actualización en el EKF.

$$K_t = \bar{\Sigma}_t \cdot H_t^{\top} \cdot (H_t \cdot \bar{\Sigma}_t \cdot H_t^{\top} + Q_t)^{-1} \tag{3.18}$$

$$x_t = \bar{x}_t + K_t \cdot (z_t - h_t) \tag{3.19}$$

$$\Sigma_t = (I - K_t \cdot H_t) \cdot \bar{\Sigma}_t \tag{3.20}$$

En la expresión (3.19), z_t se trata del vector que contiene las medidas recibidas en cada instante, mientras que h_t es un vector que contiene el valor de cada magnitud a medir según el modelo de cada medida evaluado con el valor predicho del vector de estados. Es decir, el valor que se esperaría obtener en la medida si la predicción fuese correcta. Esta diferencia, como se indico con anterioridad, recibe el nombre de *innovación*.

En caso de obtener una innovación nula, lo que designa una predicción sin error, el valor de la estimación del vector de estados coincidiría con la del valor predicho. Esta situación, puede darse también en la situación donde no se reciba ninguna medida que pueda realizar el papel de actualizador del filtro. Aunque al contrario que la primera situación, la incertidumbre respecto a la estimación y la desviación típica de la

misma adquirirían valores cada vez más elevados.

Este último es así debido a que no llegaría a entrarse en la fase de actualización, asignando los valores predichos de la media y covarianza como los resultantes de la estimación. Al hacer esto, no se tendría ninguna realimentación sobre la bondad de la estimación, y por pura definición, la matriz de covarianza iría incrementando su valor iterativamente, hasta que pueda recibirse alguna medida y evaluarse la fase de actualización.

3.5 Conclusiones

Durante el desarrollo y diseño del Filtro Extendido de Kalman, han podido conocerse en mayor profundidad los elementos y factores que caracterizan este tipo de algoritmos, así como también, se ha manejado de primera mano la teoría y los fundamentos que construyen los cimientos de su planteamiento.

Uno de los enfoques que se ha pretendido visualizar muestra como ha sido posible la adaptación de un algoritmo con limitaciones prácticas como el KF, a otro, que según las necesidades se ha modificado de forma adecuada al problema, como en este caso el EKF ante no linealidades.

Además del análisis del filtro, otro de los grandes retos en el diseño parte del modelado y adquisición de medidas de los sensores incorporados en el sistema de estimación. Tanto su diseño como inclusión en el conjunto sirven de gran ejemplo ante uno de los objetivos del trabajo, como es la fusión sensorial en robots móviles.

Finalmente, visualizando el sistema de estimación de forma global, se denota una buena robustez ante variaciones de medidas y en la precisión sensorial, además de obtener un algoritmo relativamente sencillo y eficaz en relación a los costes computacionales y sus limitaciones.

4 Implementación

4.1 Introducción

A continuación se expondrá la implementación realizada del trabajo al completo en los distintos softwares empleados (Figura 4.1). Se describe como se ha desarrollado la obtención de medidas y su funcionamiento, así como su inclusión en MATLAB y ROS.

Dicha implementación se ha dividido en dos partes bien diferenciadas: una primera en MATLAB, con el objetivo de afianzar los conceptos aplicados y diseñar los métodos y algoritmos con una mayor versatilidad; y una segunda en ROS, un *middleware* muy usado en desarrollos de software robótico, con el objetivo de acercar en mayor medida un uso práctico del trabajo. Este último software se apoyará en el uso de Gazebo, un simulador 3D multi-robot con dinámicas.

Figura 4.1 Software empleado: a) Matlab, b) ROS, c) Gazebo.

4.2 Simulación de medidas

En una primera instancia, se describirá el uso de medidas simuladas. Esto se debe al propio diseño del sistema de estimación, que depende sustancialmente de la recepción de medidas. Aunque a efectos prácticos estas medidas se obtienen de los propios sensores, para su implementación y en ausencia de banco de medidas, éstas habrán de simularse.

El algoritmo del filtro maximiza sus prestaciones si en cada ciclo de ejecución se han recibido medidas, puesto que aumentaría la precisión de la fase de actualización de medidas mientras más información reciba.

Esto no es siempre posible, bien por fallos de recepción de medidas o no poder tomarlas por otro tipo de motivos, o bien por diferencias en los períodos de ejecución del algoritmo y de recepción de medidas de

cada sensor.

En caso de no simular las medidas, su integración en el filtro vendrá dado de forma inmediata, ya que la robustez con la que se ha diseñado el algoritmo lo permite. Es decir, se utilizan las medidas disponibles en cada iteración acometida.

Para simular la aleatoriedad de recepción descrita, se ha optado por la generación aleatoria de bancos de medida con las características propias que brinda cada sensor.

En el caso de la IMU, las medidas de velocidad se han obtenido a partir del *Ground truth*, es decir, de la trayectoria que realmente sigue el quadrotor. Sabiendo la trayectoria descrita y el tiempo que tarda en recorrerse, todo ello definido para las simulaciones, se puede obtener la velocidad lineal con la que se sigue la trayectoria. A esa velocidad, se le ha aplicado un ruido gaussiano con la finalidad de aumentar la similitud en mayor medida con lo que se obtendría de la propia IMU.

De esta forma, se ha simulado una medida de velocidad lineal en cada eje espacial para cada iteración del algoritmo, y la cual posee un pequeño ruido propicio de la recepción de un sensor como la IMU.

Para la adquisición de las medidas de las balizas radio se ha partido desde el mismo punto, la trayectoria impuesta por el *Ground truth*. A partir de la misma, se ha calculado la distancia a cada baliza en cada iteración. Además, esta distancia se ha modificado de igual forma con el añadido de ruido gaussiano, simulando la incertidumbre causada por este tipo de sensores de rango.

Posteriormente, partiendo de las características propias de estos sensores, se ha simulado la posibilidad de no recibir medida por motivos varios, como por ejemplo una interferencia de las señales radio. A cada baliza individualmente se le ha asignado una probabilidad de que ocurra este fenómeno, y como es un banco de medidas simuladas debe recibirse algún tipo de medida, por lo que se ha optado por la toma de distancias negativas. Una vez se reciba en el algoritmo una distancia negativa se ignorará, tomándose como no recibida.

Se ha utilizado del mismo modo el procedimiento descrito para las medidas de balizas radio cuando el robot aéreo se encuentra fuera de rango de recepción, de nuevo se tendrá que definir la distancia máxima de recepción. Se consigue así, añadir medidas de cada baliza independientemente de las demás, y tomando en cuenta factores prácticos y aleatorios dentro de la simulación y en cada ejecución del filtro.

La simulación de las medidas de la cámara vendrá dada por una metodología un tanto distinta. Primeramente, debe construirse una imagen partiendo de la posición del robot durante la trayectoria. Dicha imagen contendrá los puntos simbólicos de las balizas contenidas de el FOV de la cámara. Todo ello implementado según se ha descrito en el capítulo previo. Llegados a este punto se habría simulado la recepción de información de una cámara

Una vez se obtiene la imagen generada, se tendrá que procesar la información que porta. Este análisis en una implementación física tendría que darse mediante procesamiento de imágenes y el uso de los mencionados *ArUcos Markers*, para poder identificar y detectar las balizas individuales que están contenidas en cada *frame*.

En la simulación propuesta se parte de que las balizas contenidas en la imagen recibida son inmediatamente identificables. Por tanto, para obtener información de dicha imagen, simplemente habrá que obtener las coordenadas x e y de cada baliza en píxeles.

Al tratarse de una cámara fija, no siempre podrán verse todas las balizas en una imagen. Esto hace que solo se añada como medida las coordenadas de las que se encuentran contenidas en ella en una iteración dada del algoritmo. Es decir, solo se obtendrá información de la cámara si durante la trayectoria que sigue el quadrotor se divisan balizas en el FOV de la cámara.

4.3 Esquema en MATLAB

MATLAB [8] es un sistema de computo numérico con lenguaje de programación propio. Cuenta con un entorno de programación y desarrollo muy versátil y propicio para fines educativos.

Todo ello hace que sea una opción adecuada para el cómputo de algoritmos iterativos y que pueden contar con cierto coste y complejidad. Además, presenta gran facilidad para la representación gráfica, y otro tipo de cálculos como el simbólico.

El desarrollo del presente trabajo usando este software se ha incluido en un único *script*. En él se ejecuta el algoritmo diseñado para una trayectoria dada, además de varios parámetros y efectos prácticos de diseño. Estos se definen para su desarrollo en MATLAB en la siguiente tabla.

Tabla 4.1 Valores de parámetros en MATLAB.

Definición	Notación	Valor
Período de ejecución	T	1 s
Desviación típica de medida en balizas radio	q_1	0.02 m
Desviación típica de medida en IMU	q_2	0.03 m/s
Desviación típica de medida en cámara	q_3	2 pix
Distancia máxima de alcance de las balizas	d_{max}	15 m
Probabilidad de recibir medida de una baliza	$Prob$	75 %
Distancia focal	f	0.0042 m
Ancho de la imagen	N	1500 pix
Alto de la imagen	M	1000 pix
Ancho del sensor de imagen	w	0.00496 m
Alto del sensor de imagen	h	0.00352 m
Desviación típica de incertidumbres del modelo del sistema	r	0.03 m
Posición baliza 1	$B1$	(0.2,1.0,0.5) m
Posición baliza 2	$B2$	(0.0,0.5,0.8) m
Posición baliza 3	$B3$	(1.5,0.2,1.0) m
Posición baliza 4	$B4$	(1.0,0.0,1.8) m
Posición inicial del robot	X_0	(1.0,1.0,1.0) m

Una vez definidos los parámetros necesarios para la implementación del filtro y la simulación de medidas, es posible ejecutar el *script* para una trayectoria dada. Esta trayectoria se ha definido como un conjunto de puntos por los que el robot aéreo pasa cada período T, definido en la Tabla 4.1.

Por otro lado, como bien se explica en capítulos anteriores, las balizas se sitúan sobre el robot terrestre, partiendo de que su posición es el origen de coordenadas. Su distribución sobre el mismo se ha asignado de forma que aporten la mayor información posible en los distintos ejes espaciales, colocándose a distintas alturas y distancias entre sí.

A continuación, se presentará un diagrama (Figura 4.2) que refleje la estructura del algoritmo que implementa el EKF y los demás elementos aplicados en el *script* desarrollado.

Por último, para el cálculo de los Jacobianos se ha usado cálculo simbólico, el cual es de gran utilidad en la derivación de expresiones complejas. Par ello, se han tenido que definir como variables simbólicas aquellas que representan las componentes del vector de estados. En la fase de actualización se toman las expresiones necesarias y se sustituyen dichas variables simbólicas por las predichas en la fase previa.

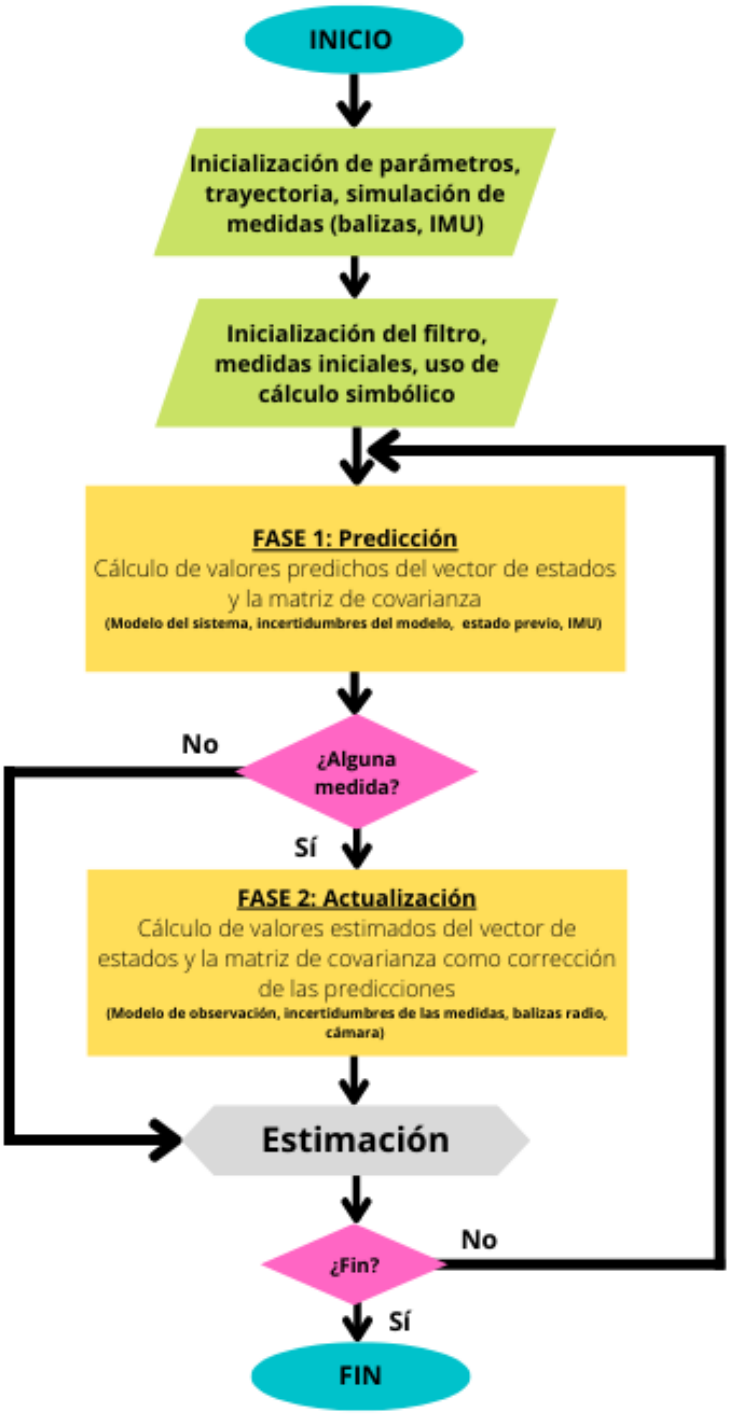

Figura 4.2 Diagrama de flujo del EKF implementado en MATLAB.

4.4 Esquema en ROS

ROS (*Robot Operating System*) [14] se trata de un conjunto de bibliotecas de software y herramientas que ayudan a crear múltiples y potentes aplicaciones en robótica. Contiene desde controladores hasta algoritmos innovadores para desarrolladores. En definitiva, todo lo necesario para operar en cualquier proyecto de robótica, siendo además de código abierto.

El funcionamiento interno de ROS se descompone en distintas estructuras y elementos, relacionados entre sí. Estas estructuras y elementos funcionales pueden clasificarse en cuatro subconjuntos: nodos, tópicos, mensajes y servicios [7].

Los nodos son la unidad base en ROS, encargados de manejar dispositivos o algoritmos informáticos. Cada nodo se ocupa de una tarea separada, y se comunican entre sí mediante los tópicos o servicios. La distribución del software se organiza en paquetes, donde cada uno se encarga de un tipo de tarea y puede contener uno o varios nodos.

Mientras tanto, los tópicos se definen como flujos de datos que se utilizan para intercambiar información entre los nodos. Estos tópicos se usan para enviar mensajes frecuentes de un mismo tipo, por ejemplo, la lectura de un sensor o la velocidad de referencia de un motor. Cada *topic* posee un nombre único y un tipo de mensaje definido. De esta forma, un nodo puede suscribirse a un tópico y recibir mensajes, o simplemente publicar sus propios mensajes en uno de ellos.

Por último, un servicio ofrece un tipo de comunicación similar a la de cliente-servidor entre dos nodos. Se diferencia de los tópicos, en que se permite comunicación bidireccional.

Al no implementar ROS en un robot físico, se ha recurrido a Gazebo [13], un simulador 3D multi-robot capaz de dar soporte a múltiples proyectos de ROS. Provee de entornos realistas y multitud de paquetes de simulación, que albergan desde simples sensores hasta modelos complejos de robots y sus dinámicas.

Por ello, la implementación en ROS, se ha construido con ayuda del simulador Gazebo. Concretamente, se ha realizado sobre un paquete de simulación de un quadrotor en un entorno 3D [10]. Este paquete brinda los modelos y dinámicas de un quadrotor en un entorno abierto tridimensional, que junto con un paquete de control por teleoperación, facilitan la odometría y trayectoria seguida por el robot aéreo. Véase en la Figura 4.3

Figura 4.3 Simulador con quadrotor.

A continuación se indican los valores empleados en la implementación del algoritmo en ROS. Solo se indican los que difieren de la Tabla 4.1.

Tabla 4.2 Valores de parámetros en ROS.

Definición	Notación	Valor
Período de ejecución	T	50 ms
Desviación típica de incertidumbres del modelo del sistema	r	0.3 m
Posición inicial del robot	X_0	(0.0,0.0,0.0) m

Cada nodo tiene asociado un *script* con la programación del mismo. Éstos pueden estar escritos en *Python* o $C++$. Para la implementación del filtro, se ha diseñado un nodo encargado de ejecutar el algoritmo completo, donde para la programación de su *script* se ha optado por utilizar *Python*. Dentro de dicho *script* se ha programado el algoritmo con ayuda de dos librerías del lenguaje usado, como son *NumPy* [11], para el cálculo numérico, y *SymPy* [15], para el cálculo simbólico.

Una vez ejecutado el nodo, y con ayuda de la herramienta *Rviz* (propia de ROS), podemos visualizar el entorno de simulación del quadrotor. Gracias a esta herramienta se visualiza durante la teleoperación, la posición de las balizas y de la estimación en cada instante. Estos parámetros se corresponden a las flechas rojas de la Figura 4.4, donde la flecha bajo el quadrotor se corresponderá a la estimación de su posición en dicho instante.

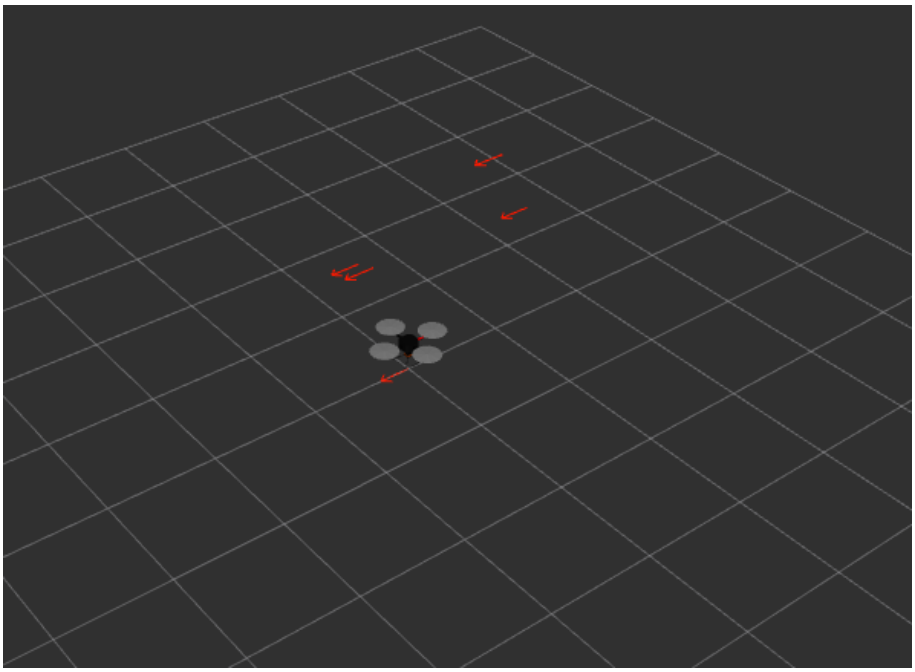

Figura 4.4 Quadrotor en Rviz.

Por último, se mostrará un diagrama donde se expondrán los distintos nodos y *topics* empleados (Figura 4.5). Se han remarcado de distinto color los elementos a destacar en la red de nodos compuesta.

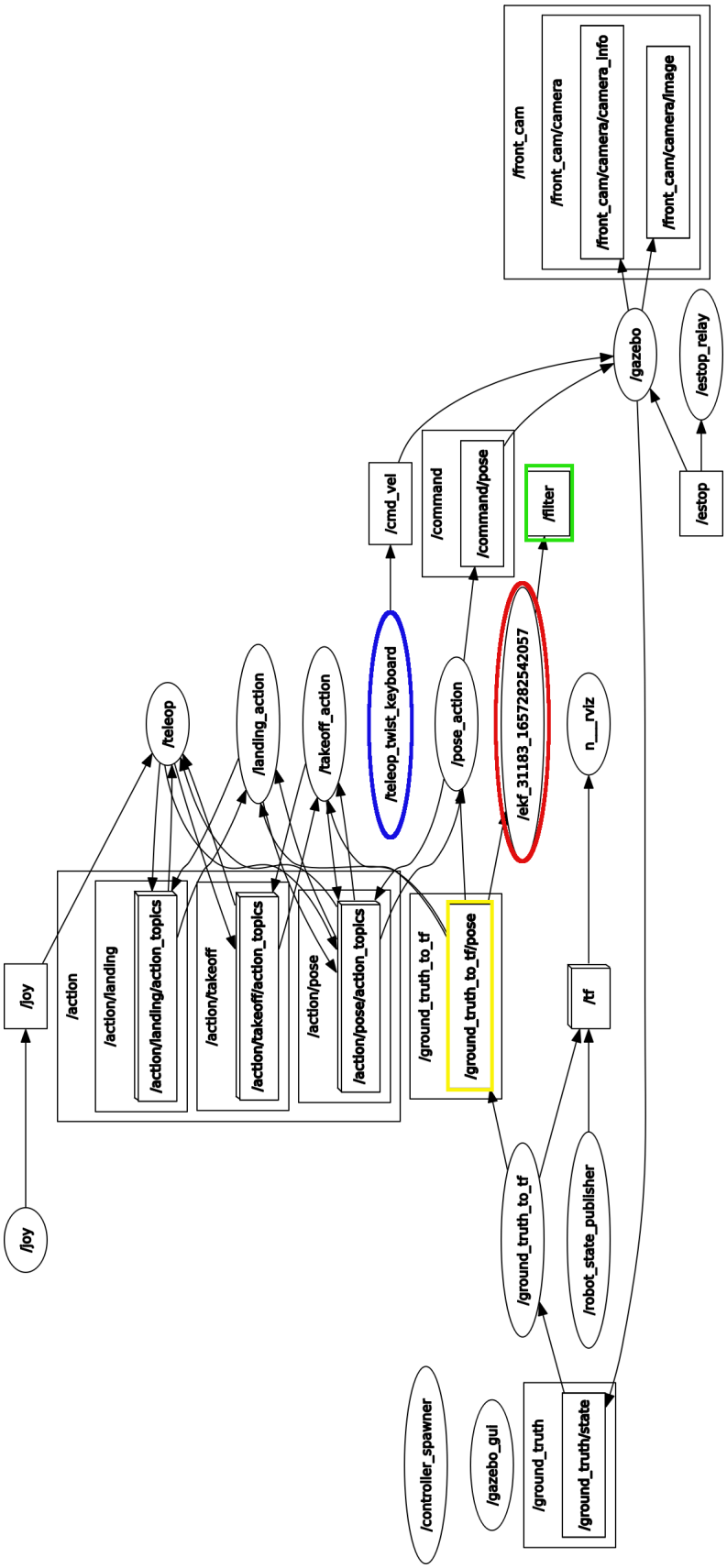

Figura 4.5 Grafo de ROS.

De color azul se muestra el nodo proveniente del paquete de teleoperación, con el que mediante atajos de teclado se puede ejercer una actuación sobre el robot aéreo. Esta actuación surte efecto a través del *topic* $/cmd\_vel$. Luego, de color rojo se encuentra el nodo desarrollado con la implementación del EKF, mientras que el *topic* marcado de verde será la información que se envíe a la simulación, como la posición de las balizas o la estimación en forma de flecha ($/filter$), es decir, donde publica el nodo del EKF.

El resto de nodos formarán parte de la ejecución de ROS y Gazebo, así como de la simulación del quadrotor empleada, inclusive el *topic* remarcado en amarillo. Este último, contiene la información de la posición real (*Ground truth*) del robot aéreo, por lo que el nodo desarrollado debe estar suscrito y recibiendo información de éste ($/ground\_truth\_to\_tf/pose$).

La estructura del nodo desarrollado, entendiéndose como el algoritmo interno de éste, se corresponderá a la implementada en MATLAB, pudiendo hacer referencia a la Figura 4.2

4.5 Conclusiones

La implementación del algoritmo completo en MATLAB demuestra el potencial de este software.

Es posible diseñar un sistema complejo relativamente rápido y de manera sencilla. Contando con un potente motor de cálculo y una precisión bastante buena. Es una gran herramienta educativa y de desarrollo para proyectos de robótica en fase inicial o en investigación.

Por otro lado, ROS es el software o herramienta por excelencia en cualquier proyecto que se desee desarrollar en robótica.

ROS permite aplicar directamente un proyecto a la práctica, por lo que el desarrollo de aplicaciones adquiere una seriedad y madurez de alto nivel. Podría destacarse la flexibilidad y potencia que ofrece su estructura y elementos funcionales. Pudiéndose concebir una extensa red de nodos para cada tarea a realizar, los cuales se comunican constantemente entre sí mediante *topics* con el envío de mensajes de distintos tipos, o mediante comunicaciones por servicios.

En conclusión, ambas implementaciones tienen numerosos beneficios y ventajas según la plataforma y los objetivos que se planteen. Si lo que se busca es la investigación entorno a un método, o el estudio de alguno a nivel teórico, MATLAB ofrece una mayor sencillez y fácilidad de programación. Por otro lado, si se quiere llevar el proyecto a la práctica o se trata de una implementación de un mayor nivel y complejidad, ROS permite este tipo de desarrollos.

5 Experimentos y análisis de resultados

5.1 Introducción

El objetivo de este capítulo será la puesta a prueba de los algoritmos implementados, tanto en MATLAB como en ROS. Además, se pretende recoger los resultados más simbólicos y determinantes para el análisis y comprensión del problema tratado.

La principal diferencia en la implementación efectuada viene dada por la trayectoria seguida por el quadrotor. En MATLAB, dicha trayectoria se impondrá desde un inicio, determinando exactamente su recorrido y la forma en que se recorre. Por otro lado, en ROS/Gazebo la trayectoria será fruto de la teleoperación del robot aéreo en el el entorno de simulación por el propio desarrollador.

Con los siguientes experimentos y simulaciones se pretende validar la robustez y bondad del filtro, además de la fusión sensorial desarrollada. Otro de los puntos claves a analizar será la calidad de la estimación efectuada, y de si se ha conseguido obtener una localización aceptable del quadrotor.

Para ello, se expondrá la siguiente estructura de experimentos y análisis.

- **EKF sin sensores**
  - Pruebas en MATLAB
  - Pruebas en ROS
- **EKF con IMU**
  - Pruebas en MATLAB
  - Pruebas en ROS
- **EKF con IMU y sensor de rango**
  - Pruebas en MATLAB
  - Pruebas en ROS
- **EKF con IMU y cámara**
  - Pruebas en MATLAB
  - Pruebas en ROS
- **EKF con todos los sensores**
  - Pruebas en MATLAB
  - Pruebas en ROS

Se sigue un orden creciente en el uso y adición de los sensores empleados, de forma que podrá analizarse paso a paso los beneficios y ventajas de la fusión sensorial.

5.2 Sin sensores

Para una primera toma de contacto con el algoritmo se implementará su uso en ausencia de sensores. Esto se traduce en ignorar la fase de actualización de medidas, ya que no se dispone de sensores que aporten información. Aunque no es la naturaleza propia del algoritmo y el uso del filtro, en ocasiones, debe realizarse la estimación sin toma de datos externos, normalmente puede ocurrir solo de forma temporal.

5.2.1 Pruebas en MATLAB

Lo primero antes de realizar las pruebas con el algoritmo es definir una trayectoria a seguir. Dicho recorrido se ha establecido de tal forma que se maximice el uso de los sensores durante toda la simulación. Se trata de una trayectoria de ida y vuelta por distintos caminos y con desplazamiento en cada uno de los tres ejes. De esta forma, se consigue aprovechar al máximo el algoritmo y analizar todo su potencial.

En esta primera prueba se utiliza el algoritmo en ausencia de cualquier sensor. Es decir, su funcionamiento se basará únicamente en la predicción de la primera fase del filtro y los valores iniciales.

El vector de estados esta compuesto por las coordenadas tridimensionales de la posición del quadrotor y las componentes de su velocidad. Además, el modelo del sistema sugiere un MRU para la evolución de la posición, y una velocidad invariable. En este punto, surgen dos posibles caminos, que la velocidad se inicialice a cero o a un valor constante.

En el primero de los casos, la estimación de estados, y con ello la localización del robot aéreo, puede verse en el siguiente gráfico.

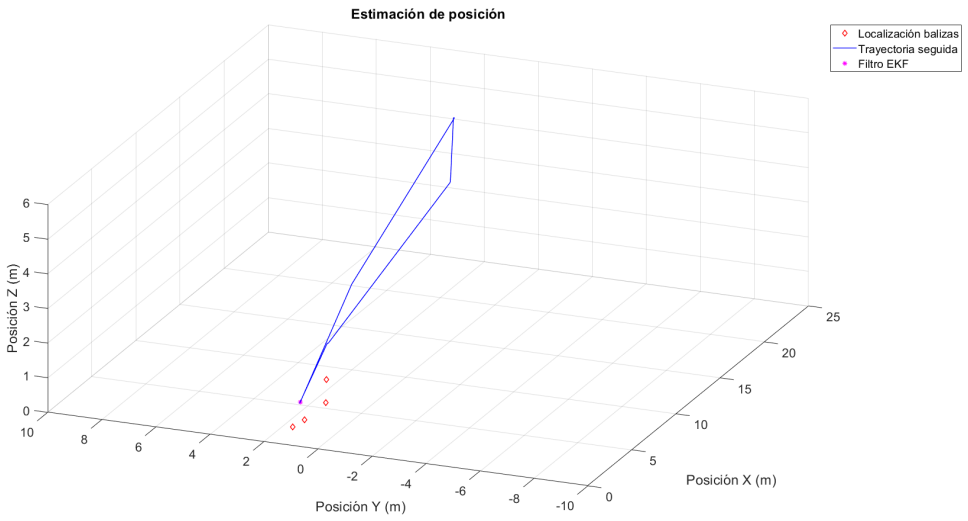

Figura 5.1 Localización sin sensores 1.

Al mantenerse la velocidad nula, la predicción aunque esté basada en un MRU, tenderá a mantener la posición constante también. Por tanto, la estimación siempre será la inicialización del filtro.

Además, observando la matriz de incertidumbre de la estimación, se aprecia como al no actualizar en ningún momento, el determinante de la misma (método para evaluar la magnitud de la incertidumbre) va creciendo.

Por otro lado, si se analiza aplicando una velocidad no nula en la inicialización, la estimación de la posición muestra la siguiente forma.

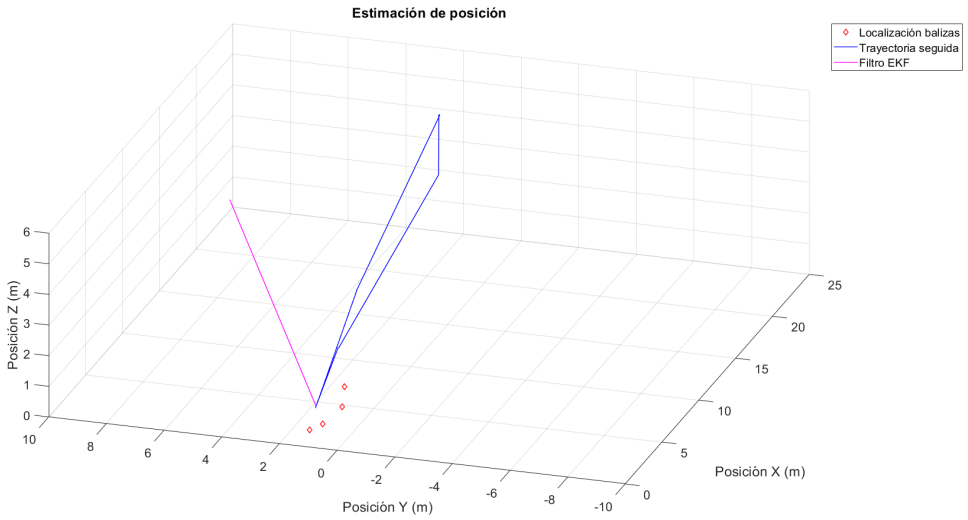

Figura 5.2 Localización sin sensores 2.

En este ejemplo, la velocidad lineal se ha supuesto de igual valor y constante para las componentes de cada eje. Por ello, la trayectoria que se estima se corresponde con una recta equidistante a cada plano. Al suponer una velocidad constante, esta no variará y por tanto marcará las posiciones que seguirá la estimación.

Como puede observarse en la Figura 5.3, el error cometido, simplemente prediciendo el estado del quadrotor en función del modelo lineal impuesto y el estado previo, es de una magnitud totalmente inadmisible y fuera de credibilidad.

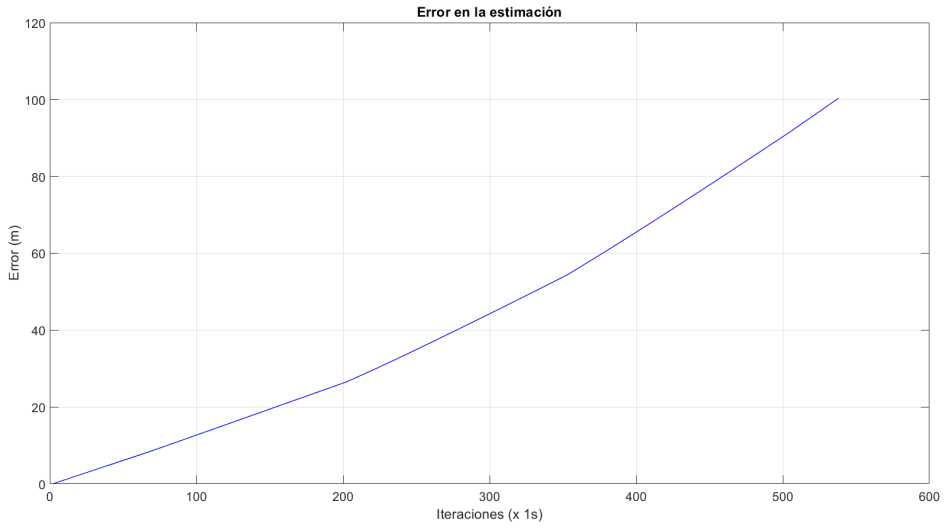

Figura 5.3 Error de estimación sin sensores.

Al no tener ningún mecanismo de corrección de estimación, como la actualización, hace que los errores alcancen cada vez valores mayores. Se remarca la importancia de la obtención de información externa a través de los sensores del robot.

No puede determinarse ninguna utilidad ni ventaja del uso del EKF en ausencia de sensores, aunque si es importante la capacidad del filtro de dar una predicción en intervalos pequeños sin medidas, pero siempre

en constante corrección por la actualización de estas.

5.2.2 Pruebas en ROS

En el uso de ROS como software, se expondrán los resultados obtenidos y las diferencias con respecto a la implementación en MATLAB. Es decir, se dedicará a un análisis más exhaustivo el uso de MATLAB, ya que se cuenta con la posibilidad de recorrer una trayectoria determinada, mientras que la seguida en la simulación de ROS se obtendrá por teloperación.

Suponiendo el caso de no contar con sensores en el robot aéreo, el resultado no dista demasiado del ya descrito en MATLAB. Se recibe una estimación de la localización anclada a la inicialización propuesta.

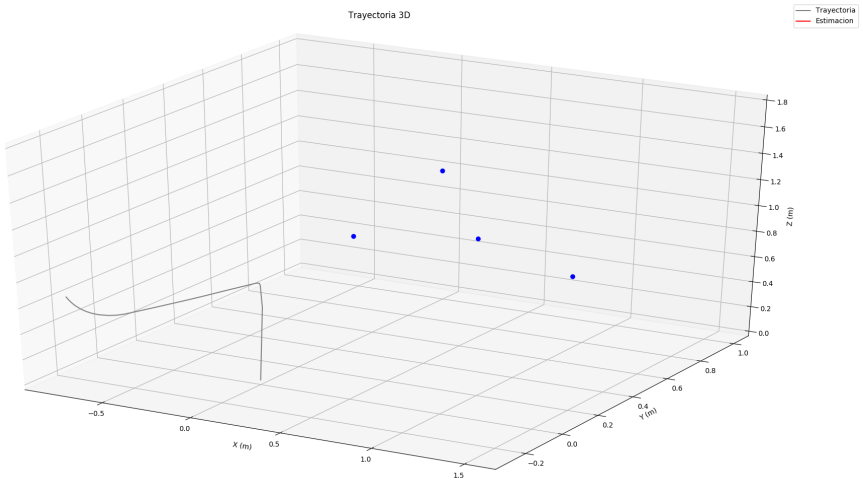

Figura 5.4 Localización sin sensores.

5.3 Solo IMU

En la siguiente prueba, se va a incluir como único sensor la IMU. De esta forma, podrán medirse los incrementos y decrementos de velocidad lineal en cada eje que sufre el robot aéreo durante su trayectoria.

5.3.1 Pruebas en MATLAB

Usando la misma trayectoria que en la prueba anterior, se comparará las modificaciones que sufre la estimación resultante con el hecho de incluir la IMU como sensor. Dicho resultado se visualiza en la Figura 5.5.

La mejora efectuada sobre la estimación de la posición es indudablemente mayor. Se observa como se intenta seguir el recorrido provisto de forma aproximada. Aun así, los errores siguen siendo bastante elevados.

Esto último se debe a que la información que obtiene la IMU proviene de medidas internas, es decir, no se reflejan datos percibidos del entorno que ayuden a su localización con mayor precisión.

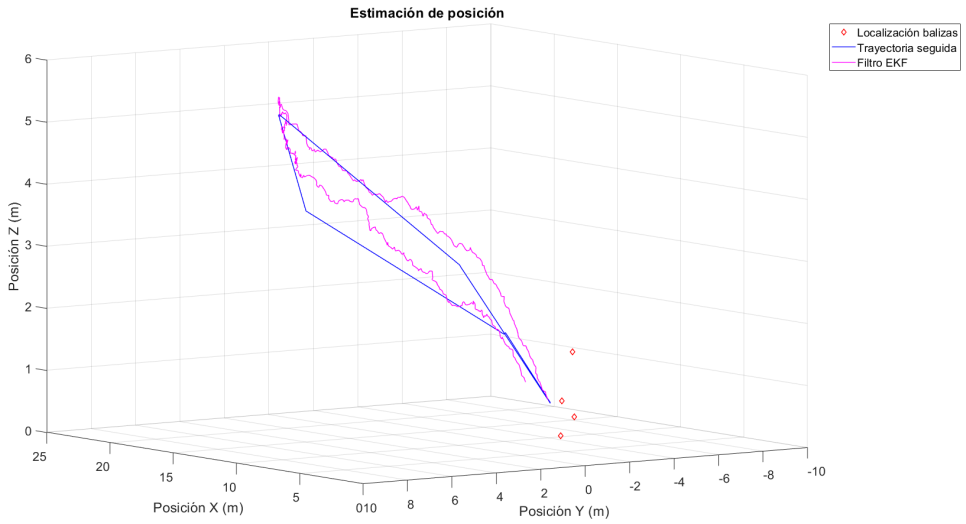

Figura 5.5 Localización con IMU.

Dichos errores pueden verse en valor absoluto en la Figura 5.6, donde se comprueba como su magnitud aumenta en función del número de iteraciones. Esto da la pista de que el error se sigue acumulando, y que la mejora que ofrece la IMU, en este caso, va en la precisión de la predicción, dando mayor información en la evolución de los estados.

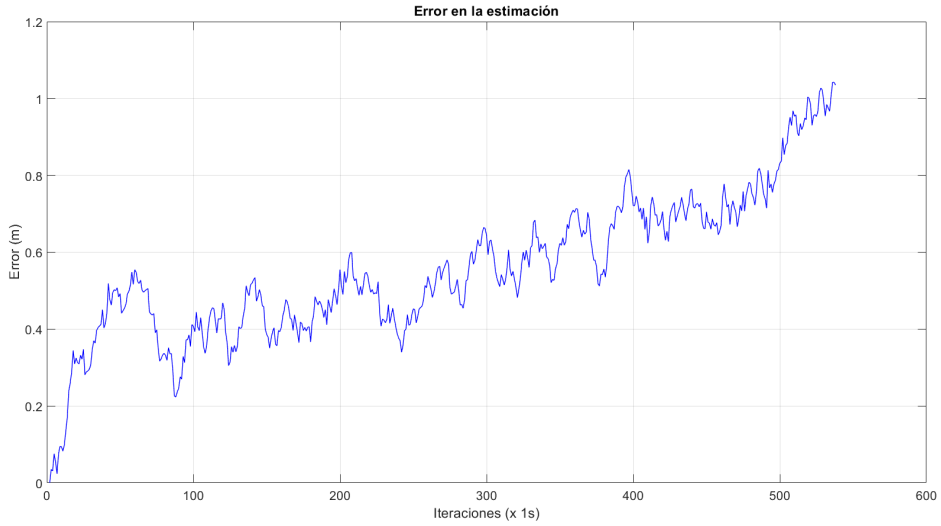

Figura 5.6 Error de estimación con IMU.

Esta conclusión puede obtenerse también observando la magnitud de la matriz de incertidumbre en la estimación. No ha varia demasiado con respecto al uso o no de la IMU. Esto en parte se debe a que la IMU se aplica en predicción y no se toma como una actualización externa.

5.3.2 Pruebas en ROS

En el caso de ROS, las velocidades se han tomado directamente de la odometría del robot, con el consecuente error gaussiano aplicado para la simulación de las medidas de la IMU.

Posiblemente al seguirse una trayectoria más alejada de los movimientos rectilíneos empleados en MATLAB, los incrementos y decrementos de velocidad aportan valores más dispares, dando como resultado un mayor incremento del error al visto anteriormente.

Esto puede comprobarse en la siguiente figura, donde la estimación difiere en cada instante cada vez más acentuadamente.

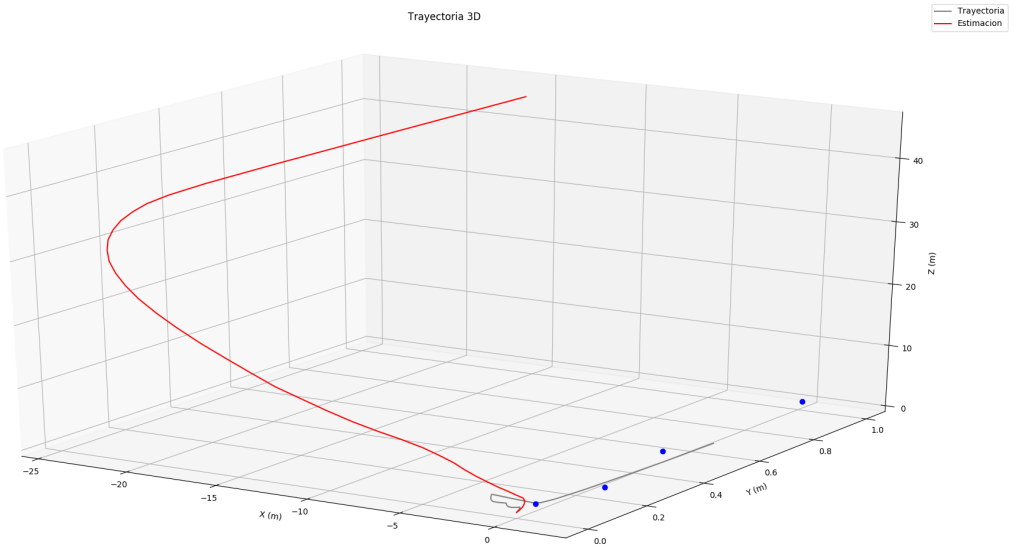

Figura 5.7 Localización con IMU.

5.4 IMU y sensor de rango

La siguiente prueba a realizar consiste en la inclusión de las balizas radio como sensores de rango. Con este experimento podrá evaluarse la influencia de un sensor de rango como el aplicado sumado al sistema de predicción ya descrito. Será la primera toma de datos externos mediante un sensor.

5.4.1 Pruebas en MATLAB

En este caso, se añaden medidas de la distancia a la que se encuentran las balizas, siguiendo el procedimiento descrito sobre la simulación de medidas. Gracias a estos datos se obtiene información externa que se incluye en el cómputo del EKF. Con dichas medidas es posible ejecutar la fase de actualización y corregir las incertidumbres generadas en la predicción.

En la siguiente figura se representa la localización estimada del robot con la ayuda del los sensores de rango.

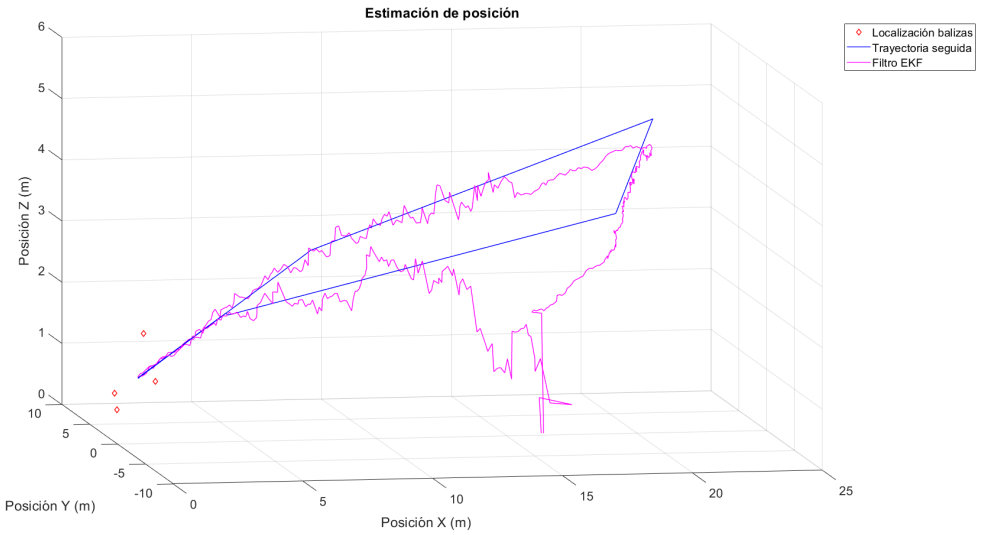

Figura 5.8 Localización con IMU y balizas.

Salvo en uno de los tramos estimados, se consigue una mayor precisión y acercamiento de la estimación a la trayectoria real. Se ha observado un peor comportamiento en las transiciones de la ausencia al uso de sensores externos, hasta que gracias a la información que aportan corrigen la estimación dada.

Debido a la trayectoria elegida, durante el tramo más alejado de las balizas, estas salen del rango máximo permitido para la recepción de las señales radio. Esto obliga al algoritmo a ejecutar dicho tramo solo con la IMU, incrementando notoriamente los errores cometidos y la incertidumbre de la estimación.

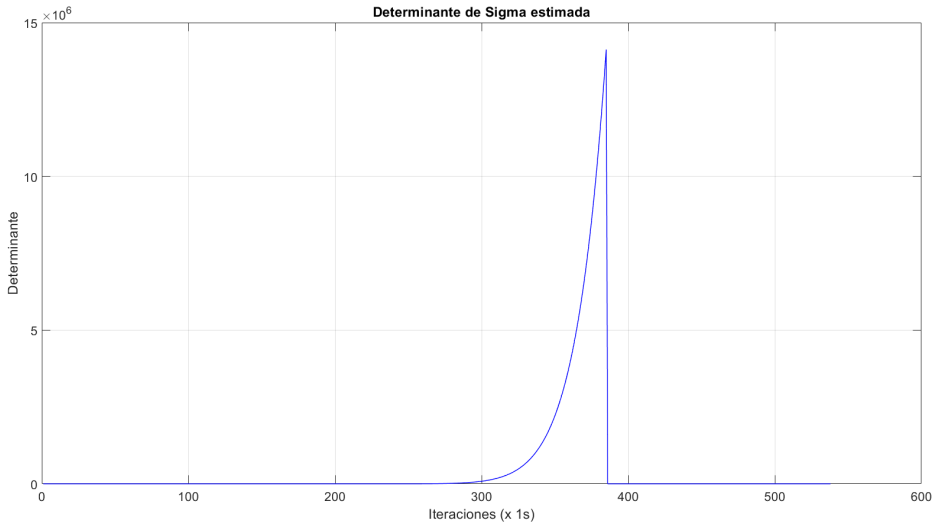

Figura 5.9 Incertidumbre de la estimación con IMU y balizas.

Gracias a la Figura 5.9 se puede comprender mejor el fenómeno observado. Durante el tramo fuera de rango la incertidumbre de la estimación crece exponencialmente, acumulando suficiente error como para desviar temporalmente la estimación que venía contenida entorno al recorrido correcto.

Por tanto, la siguiente gráfica representativa del error cometido describe perfectamente la bondad de la estimación en cada tramo de la trayectoria. En la zona donde el quadrotor se encuentra fuera de rango estos

errores se disparan, mientras que en la zona de recepción de medidas de distancia, el error es proporcional a la distancia media de las balizas. Es decir, a mayor distancia del conjunto de balizas, la precisión se ve linealmente decrementada.

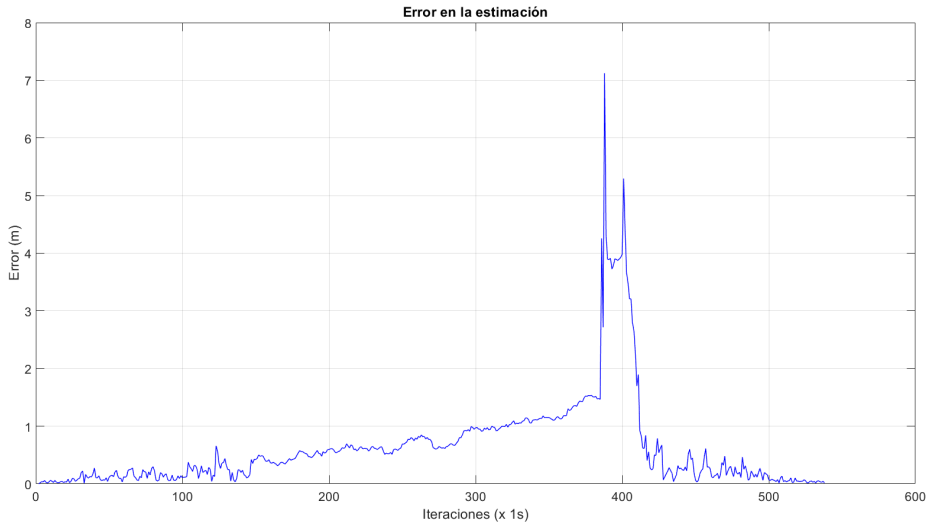

Figura 5.10 Error de estimación con IMU y balizas.

En relación a las distancias medidas de cada baliza, estas quedan reflejadas en la Figura 5.11. Se observa con facilidad como el robot aéreo se aleja de cada una de ellas en una primera instancia, luego sale de sus rangos de alcance, y posteriormente vuelve al punto inicial. Durante las zonas de recepción de medida se encuentran también los fallos de recepción aleatorios ya descritos.

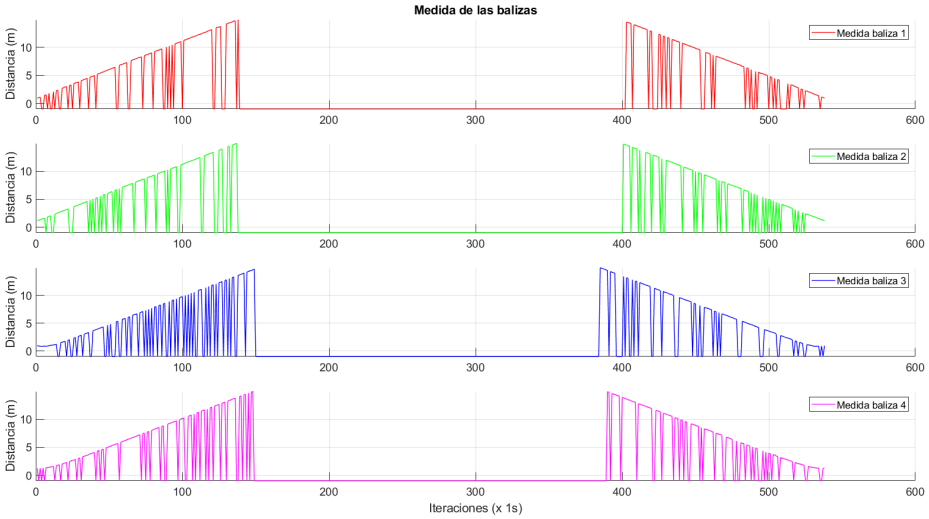

Figura 5.11 Medidas de distancias a balizas.

En conclusión, aunque las zonas fuera del alcance de las balizas generan una falta de corrección y actualización de la estimación importante, el objetivo de su inclusión era analizar dicha desventaja o limitación. En condiciones prácticas de uso, se usarían balizas con rangos que sobrepasen las zonas de actuación previstas según la tarea a realizar.

Las ventajas que ofrece el uso de este tipo de sensor en aplicaciones de inspección y mantenimiento como es el caso, también son numerosas. A un buen alcance como se ve en la Figura 5.10, el error cometido es bastante aceptable según las operaciones a ejecutar. Por otro lado, su uso no se ve afectado por condiciones de luminosidad, lo que es de gran ayuda en la aplicación requerida.

5.4.2 Pruebas en ROS

Para ROS, el uso de las balizas aporta una gran mejora, debido a que sus medidas son independientes del tipo de trayectoria seguida. Aunque puede verse afectada por los fallos aleatorios o las zonas donde no alcanza el rango máximo de la señal, la estimación dada por la actualización que proporcionan las balizas puede considerarse aceptable.

En la gráfica que se muestra a continuación, se observa como gracias a la información externa que aporta la medida de distancia a cada baliza, la estimación de la posición se va reconduciendo constantemente entorno al recorrido real. En la Figura 5.13 se muestra otro punto de vista de la trayectoria y estimación obtenidas.

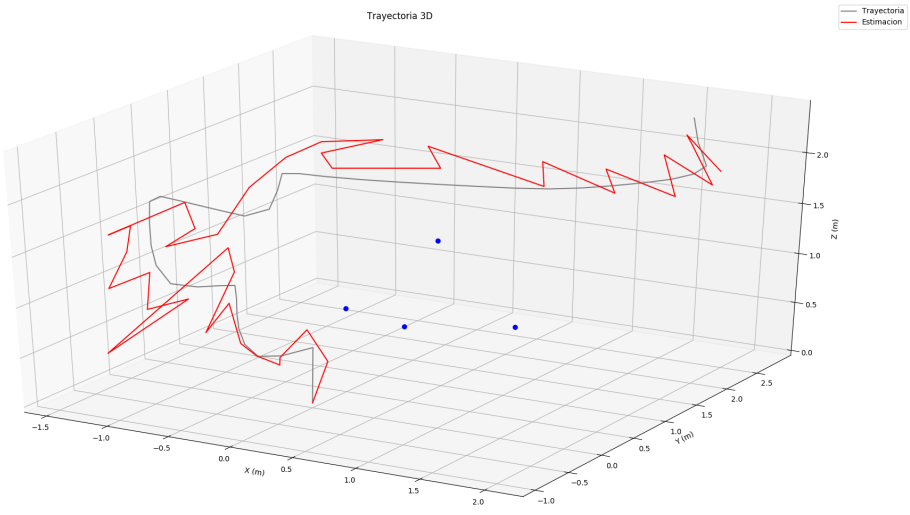

Figura 5.12 Localización mediante balizas 1.

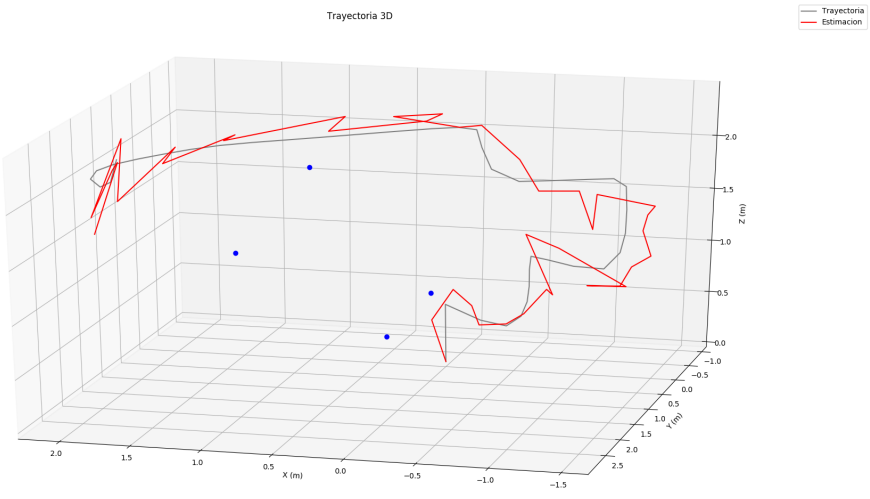

Figura 5.13 Localización mediante balizas 2.

5.5 IMU y cámara

A continuación se usará la cámara en conjunto con la IMU, en vez de los sensores de rango. Al igual que estos, la cámara solo añadirá una nueva medida de una baliza en la imagen si dicha baliza se enmarca en el FOV de ese instante. Al igual que los sensores de rango, la cámara aporta información del entorno, lo que podrá emplearse para la actualización del filtro.

5.5.1 Pruebas en MATLAB

En el uso de la cámara, solo las balizas contenidas en el FOV de la imagen añadirán medidas al cómputo del EKF. Al ser una cámara fija, y dada la trayectoria propuesta, se ha colocado con una orientación determinada en el quadrotor a fin de obtener el mayor número de medidas durante el recorrido.

El resultado de la estimación de la localización mediante la recepción de imágenes cada iteración (1 segundo), queda reflejado en la siguiente gráfica.

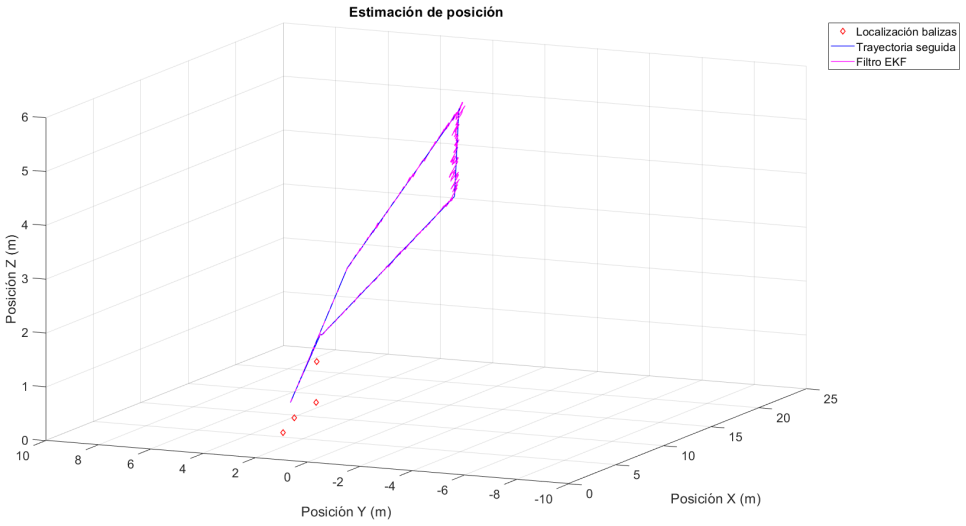

Figura 5.14 Localización con IMU y cámara.

La estimación efectuada con ayuda de la cámara obtiene una precisión bastante buena con respecto a la trayectoria real. Aunque conlleva un coste computacional mucho mayor, además de los procesamientos de imágenes que se deben implementar en la práctica, la bondad obtenida en el resultado hacen que sea recomendable su uso.

Dicho resultado depende también en gran medida del error en píxeles que introduzca la propia cámara. Con los parámetros elegidos, el error cometido para la estimación de posición dada se observan en mejor grado en la siguiente figura.

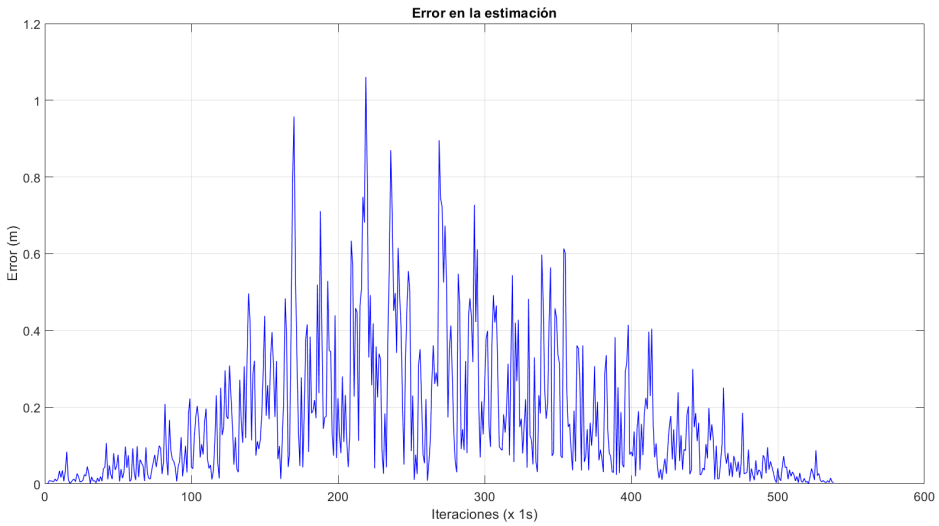

Figura 5.15 Error de estimación con IMU y cámara.

Además de reducir el orden del error cometido con respecto a las demás pruebas, también se observa relación entre la magnitud del mismo y la distancia a la que se encuentra el robot de las balizas. En el caso de la cámara, esto ocurre debido a que el FOV comprende una relación de puntos del entorno por píxel mucho mayor, es decir, mientras más lejos se encuentre el robot, más puntos del entorno se almacenarán en un mismo píxel, reduciendo poco a poco la precisión.

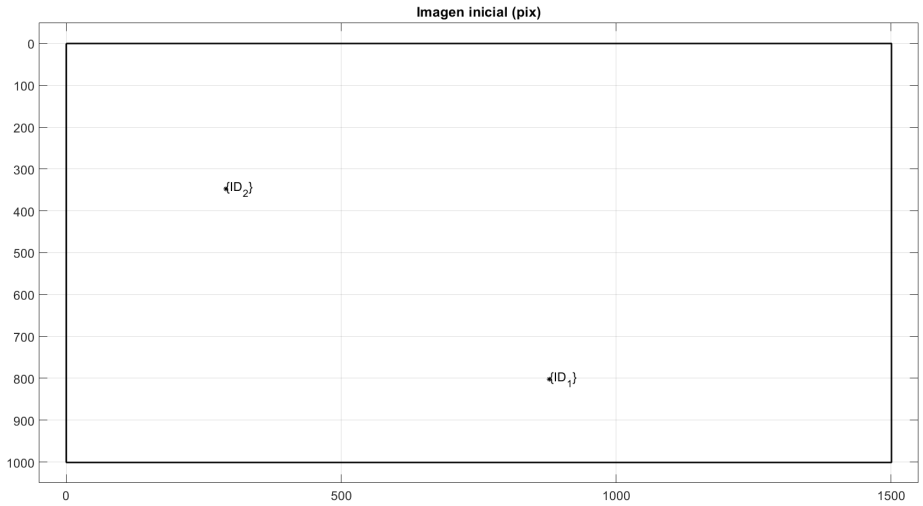

Figura 5.16 Imagen inicial de la cámara.

En la Figura 5.16 puede verse la representación de la imagen inicial obtenida por la cámara del quadrotor. En dicho instante, solo dos balizas se incluyen el FOV. Esto irá evolucionando de forma que a la mayor distancia, todas las balizas estarán incluidas, aunque se irán concentrando en una parte de la imagen con la ampliación del FOV, afirmando el fenómeno explicado.

Debido a la gran precisión obtenida y la capacidad de actualización que brinda la cámara, al observar la magnitud de la incertidumbre de la estimación se concluye como prácticamente nula (Figura 5.17).

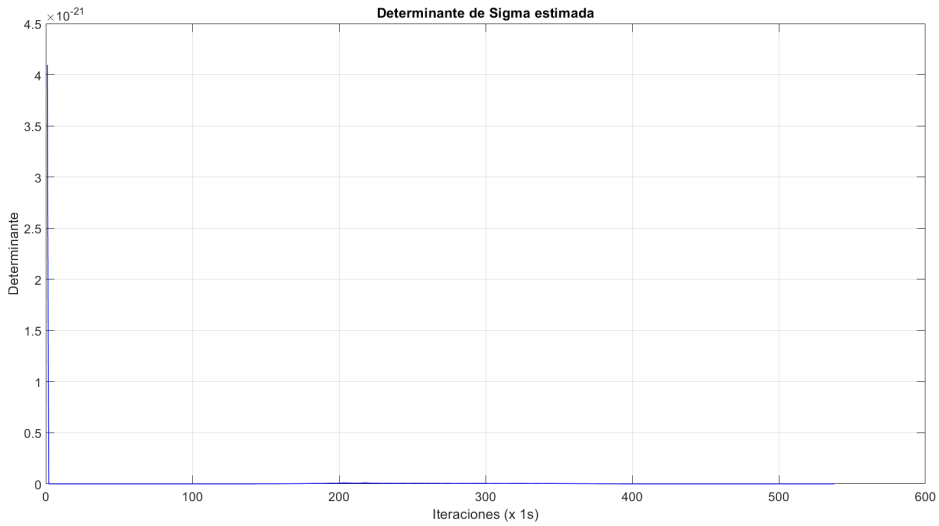

Figura 5.17 Incertidumbre de la estimación con IMU y cámara.

5.5.2 Pruebas en ROS

En contraposición a las pruebas realizadas con la cámara en MATLAB, en ROS no es posible obtener el mismo tipo de resultados.

Esto se debe principalmente, a que la trayectoria elegida en MATLAB aprovecha en todo momento el FOV de la cámara para captar las balizas en las imágenes. Por otro lado, en ROS al seguir una trayectoria por teleoperación, es muy complicado mantener dicha condición.

Dicho de otro modo, en la práctica, más aún si la trayectoria es aleatoria, contar con una cámara fija para detección de balizas no aporta demasiada ventaja. Una buena solución es utilizar una cámara móvil capaz de eludir los cambios de orientación del quadrotor.

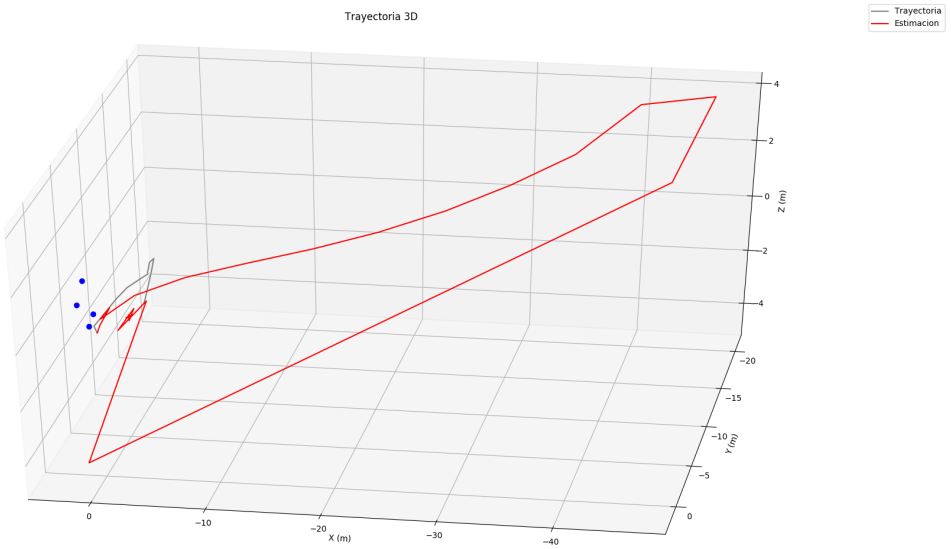

Figura 5.18 Localización mediante cámara.

El fenómeno descrito se aprecia con claridad en la Figura 5.18, donde al dejar de obtener medidas en el FOV de la cámara, esta deja de actuar hasta no volver a enmarcar alguna baliza en las imágenes recibidas. Como resultado, una vez la cámara es incapaz de captar las balizas, la estimación se dispara, corrigiéndose una vez se capta alguna de nuevo.

5.6 Todos los sensores

Por último, se analizará el uso al completo de todos los sensores disponibles. Dicho de otra forma, se comprobará el resultado de la fusión sensorial al completo. Para ello se tiene la IMU, añadida como mejora de la predicción, y las balizas y cámara, como sensores externos para la actualización del filtro.

5.6.1 Pruebas en MATLAB

Como última prueba a realizar en MATLAB, se incluirán los sensores de rango al experimento previo, con el objetivo de completar la fusión sensorial propuesta.

Al aplicar el EKF con el uso de todos estos sensores, la estimación de la localización queda de la siguiente forma.

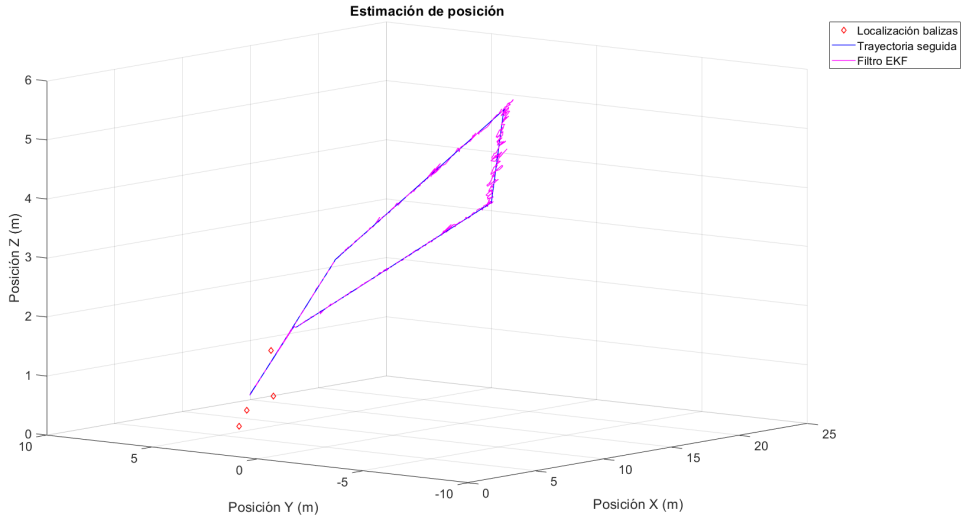

Figura 5.19 Localización con todos los sensores.

Al igual que en la prueba anterior, donde también se usaba la cámara, se obtiene una estimación de la posición con una gran precisión. Puede diferenciarse también las zonas donde entran y salen de acción las recepciones de las balizas radio, como una pequeña perturbación.

Observando la Figura 5.20, pueden obtenerse alguna conclusión extra de la combinación de estos dos sensores externos. Por ejemplo las pequeñas perturbaciones en la incertidumbre de la estimación al incluir o no señales radio. Esto viene de la comparación con la prueba anterior, a grandes rasgos, dicha incertidumbre sigue siendo considerada nula debido a su reducido orden de magnitud.

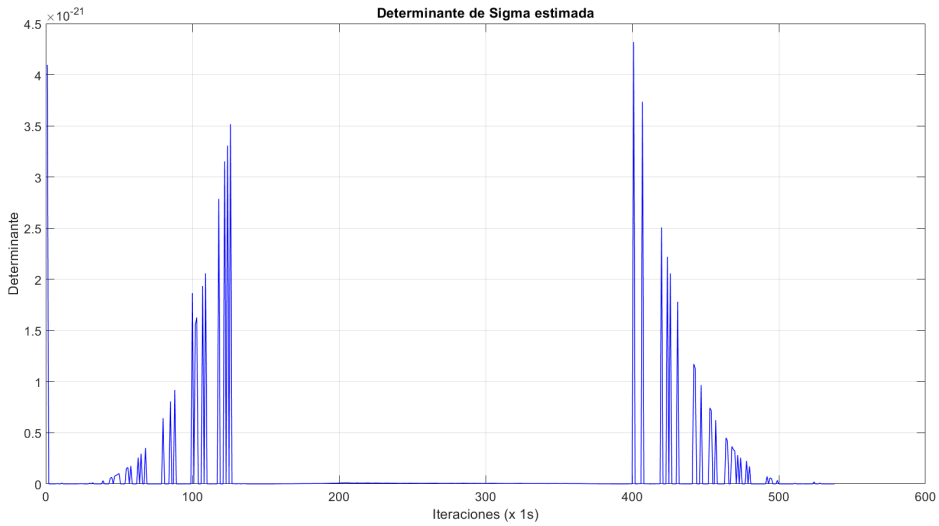

Figura 5.20 Incertidumbre de la estimación con todos los sensores.

Este leve cambio con la inclusión o no de las balizas es debido a la diferencia de las incertidumbres que introduce cada tipo de sensor. En términos generales, los dos centímetros de desviación típica de las balizas afectan en mayor medida a la incertidumbre total que los dos píxeles de desviación típica que posee la cámara.

Por otro lado, a fin de obtener el error absoluto cometido en la estimación, se observa la Figura 5.21. En comparación con el resto de pruebas se trata de la estimación con menor error cometido. En la escala propuesta de la trayectoria los errores no superan los 80 centímetros respecto al recorrido real.

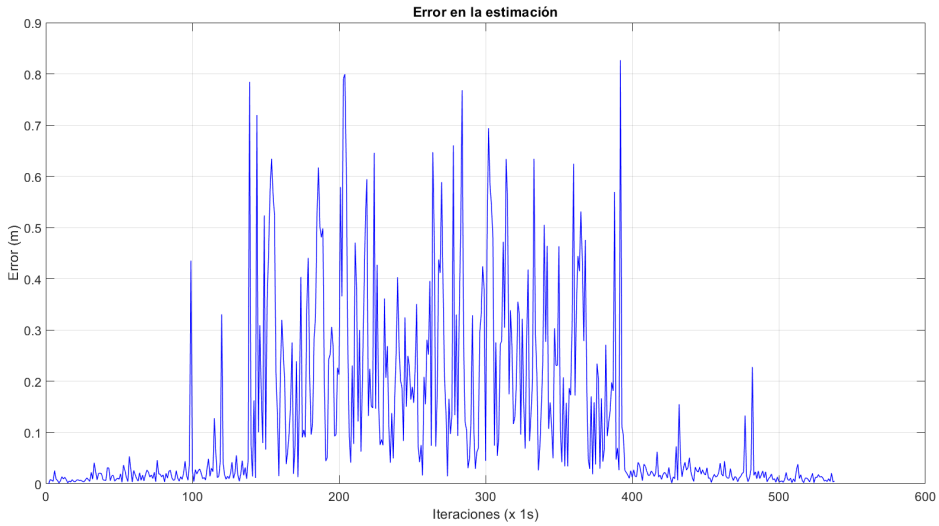

Figura 5.21 Error de estimación con todos los sensores.

Además, dichos máximos ocurren en el tramo donde solo actúa la cámara como sensor de actualización. En contraparte, en los tramos donde la fusión sensorial con todos los sensores es efectiva, el error promedio es considerablemente menor, no superando los 5 o 6 centímetros.

En definitiva, la fusión sensorial, además de mejorar la bondad y precisión de la estimación generada por el EKF, aporta una gran robustez al algoritmo completo. No solo mejora las prestaciones que aporta el uso de cada sensor por separado, sino que, a efectos prácticos, el uso de varios sensores mantienen al filtro constantemente en toma de información externa, que corrige y actualiza las predicciones propuestas para la estimación.

Dicho de otra forma, la fusión sensorial no solo opaca las carencias que puede tener un sensor en un escenario o momento determinado, sino que también potencia la información obtenida cuando se cuenta con varios de ellos.

5.6.2 Pruebas en ROS

Finalmente, se implementará el algoritmo con todo el conjunto de sensores disponible. Como ya se ha visto, el uso de la cámara no supone una mejora de tal nivel como la probada en MATLAB.

Aun así, los resultados con la fusión sensorial aplicada mejoran los descritos hasta ahora con el uso de ROS. Esto se muestra en la siguiente gráfica.

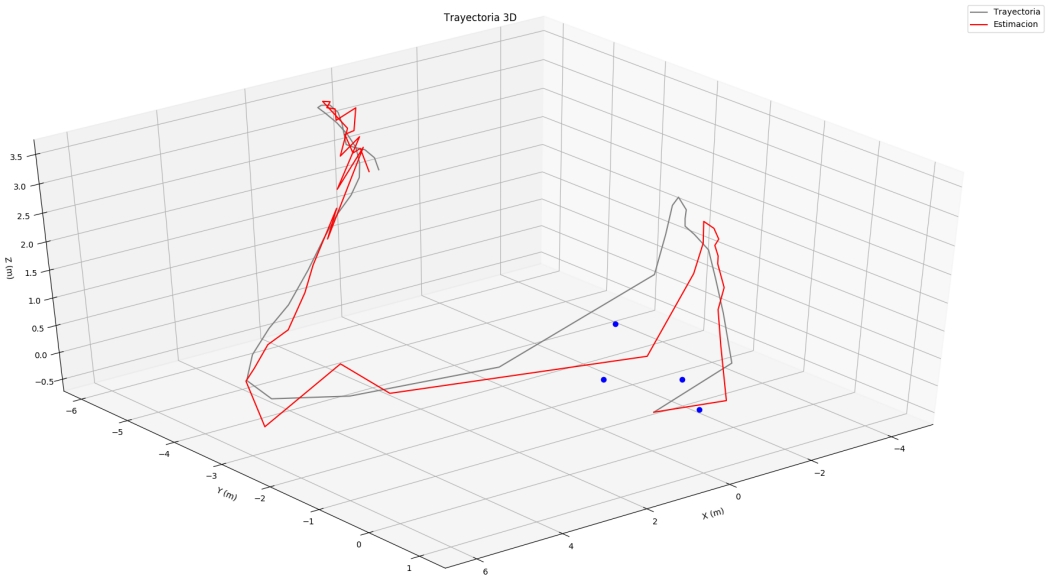

Figura 5.22 Localización con todos los sensores 1.

Se concluye un peor comportamiento que el obtenido en MATLAB, aunque puede considerarse una estimación aceptable. Además de las complicaciones en el uso de la cámara, al realizarse de forma tele-operada, sumado a los tiempos de computo y de ejecución del conjunto de la implementación, puede llevar a inducir mayor número de errores.

La trayectoria y estimación resultantes desde otro punto de vista se muestran en la Figura 5.23.

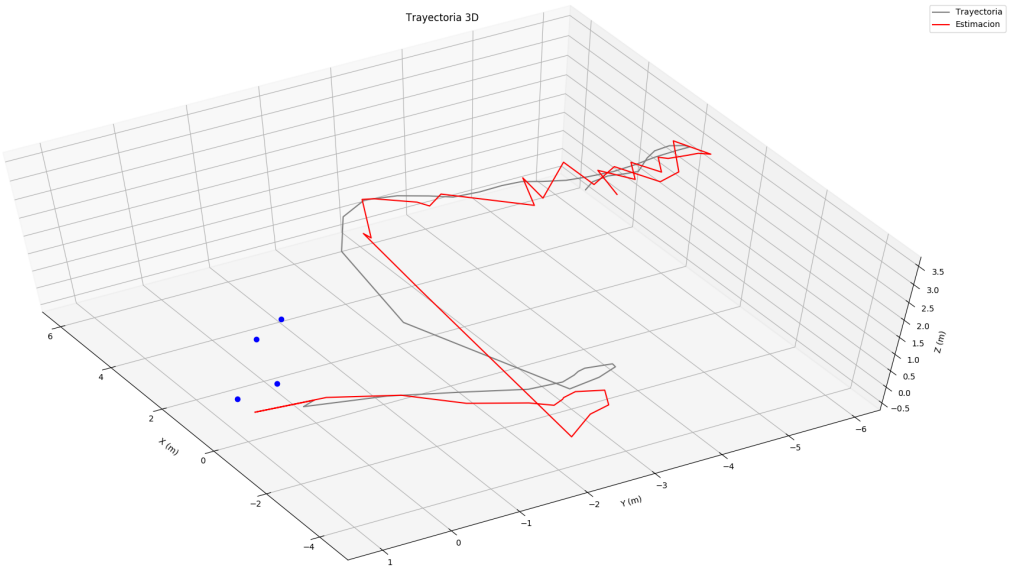

Figura 5.23 Localización con todos los sensores 2.

5.7 Comparativa de resultados

Como última instancia, se realizará una comparativa cuantitativa que refleje los resultados y análisis ejecutados previamente. De tal forma, que se puedan entender y comprender estas pruebas y experimentos de forma numérica (pruebas en MATLAB).

Para ello, se muestran en la Tabla 5.1 datos numéricos referentes a los errores y características provenientes de cada experimento realizado, facilitando la visualización de los efectos de la fusión sensorial. Como ya se ha visto durante las pruebas, todas han sido realizadas bajo las mismas condiciones y una misma trayectoria de vuelo.

Puede observarse como a medida que se incluyen sensores en la implementación del EKF, la media de los errores cometidos en la estimación va desplomándose hasta valores de poco más de algunos centímetros. Esto refuerza en gran medida la afirmación del gran potencial y efecto positivo que aporta la fusión sensorial.

También se ha destacado el análisis de resultados en las zonas donde es efectivo el uso de los sensores de rango, viéndose un mejor resultado que en el análisis de la trayectoria completa.

Además de esto, se puntualizan las perturbaciones sufridas en las salidas y entradas del alcance de estos sensores, lo que conlleva a la obtención de picos de error en la estimación. Esto puede comprobarse al ver que los errores máximos coinciden tanto a nivel global como concretando en el rango de las balizas.

Por otro lado, la magnitud de la incertidumbre de las estimaciones de cada prueba también se encuentra directamente relacionada con el uso de sensores externos. Éstos, durante la fase de actualización, reducen dicha magnitud e incluso la llevan a converger a valores muy pequeños. Se destaca la precisión del uso de la cámara en este caso.

En definitiva, la aplicación de la fusión sensorial es capaz de mejorar en gran medida los resultados obtenidos. Además de ello, el uso de sensores externos, que aporten información del entorno, ayuda a aumentar considerablemente la precisión de las estimaciones y reducir sus incertidumbres.

Tabla 5.1 Comparativa de las pruebas del EKF.

	Sin sensores	Solo IMU	IMU+balizas	IMU+cámara	Todos los sensores
Número de sensores	0	1	2	2	3
Error máximo en la trayectoria global	$100.34m$	$1.04m$	$7.12m$	$1.06m$	$0.83m$
Error medio en la trayectoria global	$42.63m$	$0.55m$	$0.62m$	$0.17m$	$0.13m$
Error cuadrático máximo en la trayectoria global	$1.01 \cdot 10^4 m^2$	$1.09m^2$	$50.73m^2$	$1.12m^2$	$0.68m^2$
Error cuadrático medio en la trayectoria global	$2.66 \cdot 10^3 m^2$	$0.34m^2$	$1.07m^2$	$0.06m^2$	$0.05m^2$
Error máximo en rango de balizas	–	–	$7.12m$	–	$0.83m$
Error medio en rango de balizas	–	–	$0.48m$	–	$0.04m$
Error cuadrático máximo en rango de balizas	–	–	$50.73m^2$	–	$0.68m^2$
Error cuadrático medio en rango de balizas	–	–	$1.32m^2$	–	$0.01m^2$
Determinante máximo de la matriz de incertidumbre de la estimación	–	–	$1.41 \cdot 10^7 *$	$4.10 \cdot 10^{-21}$	$4.32 \cdot 10^{-21}$
Determinante medio de la matriz de incertidumbre de la estimación	–	–	$5.07 \cdot 10^5 *$	$9.81 \cdot 10^{-24}$	$1.05 \cdot 10^{-22}$

$*$ Valores obtenidos por salir del rango máximo, ver Figura 5.9.

Resumiendo los resultados obtenidos, con solo una IMU y balizas radio se es capaz de estimar la posición de un quadrotor dentro del rango de alcance con un error medio no superior a $48cm$. En sustitución de las balizas por una cámara, dicho error medio puede reducirse a no más de $17cm$. Finalmente en su combinación, y en referencia a los beneficios de la fusión sensorial mediante un EKF, dicho error queda acotado a unos $4cm$ de media, unos resultados bastante buenos para la aplicación requerida.

6 Conclusiones y desarrollos futuros

A lo largo del desarrollo del trabajo realizado, se ha profundizado en gran medida en el estudio de los filtros estadísticos, con el fin de su aplicación en problemas de estimación de localización en robots móviles.

Además de ello, y con enfoque en la implementación del Filtro Extendido de Kalman, se han estudiado en gran medida numerosos sensores aplicables al trabajo. Todo ello, con el objetivo de lograr una fusión sensorial de todas las partes.

Tras el estudio y profundización en estos temas mencionados, se abordó la implementación y validación de los conocimientos adquiridos, con el fin de poner en práctica todos los elementos que componen el algoritmo de estimación. Para ello, no solo ha sido necesario diseñar el algoritmo correspondiente al filtro, sino que se ha tenido que afrontar la adición de los modelos de los sensores aplicados y la simulación de medidas de los mismos.

Por otro lado, no solo se ha realizado una única implementación de todo el conjunto del método, sino que se ha desarrollado una versión para MATLAB y otra para ROS/Gazebo. Teniendo que desarrollarse en dos lenguajes de programación distintos (lenguaje de MATLAB y *Python* en el caso de ROS).

Dicha implementación multi-plataforma hace que el diseño del algoritmo deba ser bastante general y exportable, de forma que pueda replicarse en distintos entornos. Esto junto con la propia naturaleza de los filtros estadísticos, convierten el resultado final en algoritmos bastante robustos e independientes del entorno de desarrollo.

Sortear todo este tipo de problemas, y las complicaciones que han podido producirse a lo largo del desarrollo, hacen que se haya profundizado y obtenido numerosos conocimientos tanto en el área de diseño del método, como en el uso de las plataformas de implementación.

En cuanto a las conclusiones finales obtenidas, se remarcan el gran potencial y utilidad de los filtros estadísticos en problemas de localización, las grandes ventajas y beneficios descritos de la fusión sensorial, y la gran versatilidad y aplicación de las herramientas y softwares empleados. Dentro de esta última, destacar las posibilidades que ofrece una plataforma como ROS, donde pueden construirse proyectos de elevada envergadura y aplicable a todo tipo de proyectos robóticos de última generación.

Atendiendo a las posibles aplicaciones del presente trabajo o futuras mejoras implementables, se cuenta con las siguientes partes.

La aplicación del trabajo realizado puede ser prácticamente inmediata, gracias a su desarrollo en ROS, que brida la posibilidad de su uso en robots directamente. Aplicable a cualquier problema que necesite de estimación de la localización en robots móviles con uso de sensores. Únicamente debe tenerse en cuenta la sustitución de las medidas simuladas por las que aporten los sensores físicos que se posean.

El algoritmo se ejecuta a tiempo real, ya que en ROS, la trayectoria empleada se construye por teleoperación

del quadrotor, incluso teniendo que simularse medidas. Todo esto contribuye a la afirmación previa, aunque es probable poder conseguir una mayor optimización del método aplicado.

Para las futuras mejoras, uno de los caminos principales de enfoque es la adición de nuevos sensores que incrementen las prestaciones y robustez de las estimaciones. Un claro ejemplo sería añadir un sensor del tipo LIDAR, el cual sería de gran utilidad en la inspección de este tipo de entornos.

Por último, sería de gran interés la mejora de la cámara, ya que como se ha expuesto, sus prestaciones pueden mejoran indudablemente el rendimiento global. Por ello, con el empleo de una cámara móvil, con algún tipo de algoritmo de detección, que dirija su FOV a la zona donde se encuentre una mayor cantidad de balizas, infundiría que los resultados se viesen enormemente mejorados.

Índice de Figuras

Índice de Tablas

Bibliografía

[1] Pacific Coastal and Marine Science Center, *Inertial measurement unit (imu)*, *https://www.usgs.gov/centers/pcmsc/science/inertial-measurement-unit-imu*, 2020.

[2] Andreas Obed Llanes Cornejo, *Sistema de sensores en robótica*, *https://porprofesionalmic.files.wordpress.com/2015/09/investigacion-documental-sistema-sensores-robotica.pdf*, 2015.

[3] A. De San Bernabe, J.R. Martinez-de Dios, and A. Ollero, *Efficient integration of rssi for tracking using wireless camera networks*, Information Fusion **36** (2017), 296–312.

[4] Embention, *Cómo elegir la imu adecuada para un uav*, *https://www.embention.com/es/news/la-imu-adecuada-para-un-uav/*, 2020.

[5] Comisión Europea, *Piloting*, *https://piloting-project.eu/noticias/4th-edition-of-piloting-projects-newsletter/*, 2021.

[6] Andras Lasso, Tamas Ungi, Gabor Fichtinger, and Mark Asselin, *Towards webcam-based tracking for interventional navigation*, *https://www.researchgate.net/figure/A-typical-ArUco-library-marker_fig3_323864883*, 2018.

[7] J.R. Martinez-de Dios, *Transparencias de control y programación de robots*, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, 2021.

[8] MathWorks, *Matlab*, *https://es.mathworks.com/products/matlab.html?s_tid=hp_products_matlab*, 2022.

[9] mobacommunity, *Modelo gnss mediante satélites*, *https://mobacommunity.com/modules/boonex/blogs/blogs.php?action=show_member_post&postUri=GNSS-Positioning-for-Machine-Control-More-Satellites-%C3%A2-Higher-Precision&skin=uni*, 2015.

[10] Basavaraj Navalgund, *hector-quadrotor*, *https://github.com/basavarajnavalgund/hector-quadrotor*, 2020.

[11] NumPy, *Numpy documentation*, *https://numpy.org/doc/stable/*, 2022.

[12] OpenCV, *Detection of aruco markers*, *https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html*, 2022.

[13] Open Robotics, *Gazebo*, *https://gazebosim.org/home*, 2021.

[14] _____, *Ros*, *https://www.ros.org/*, 2021.

[15] SymPy, *Welcome to sympy's documentation!*, *https://docs.sympy.org/latest/index.html*, 2021.

[16] Sebastian Thrun, Wolfram Burgard, and Dieter Fox, *Probabilistic robotics*, MIT Press, 2005.

[17] Manuel Vargas Villanueva, *Transparencias de sistemas de percepción*, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, 2021.

[18] FlightGear wiki, *Es/radio balizas*, *https://wiki.flightgear.org/Es/Radio_balizas*, 2016.