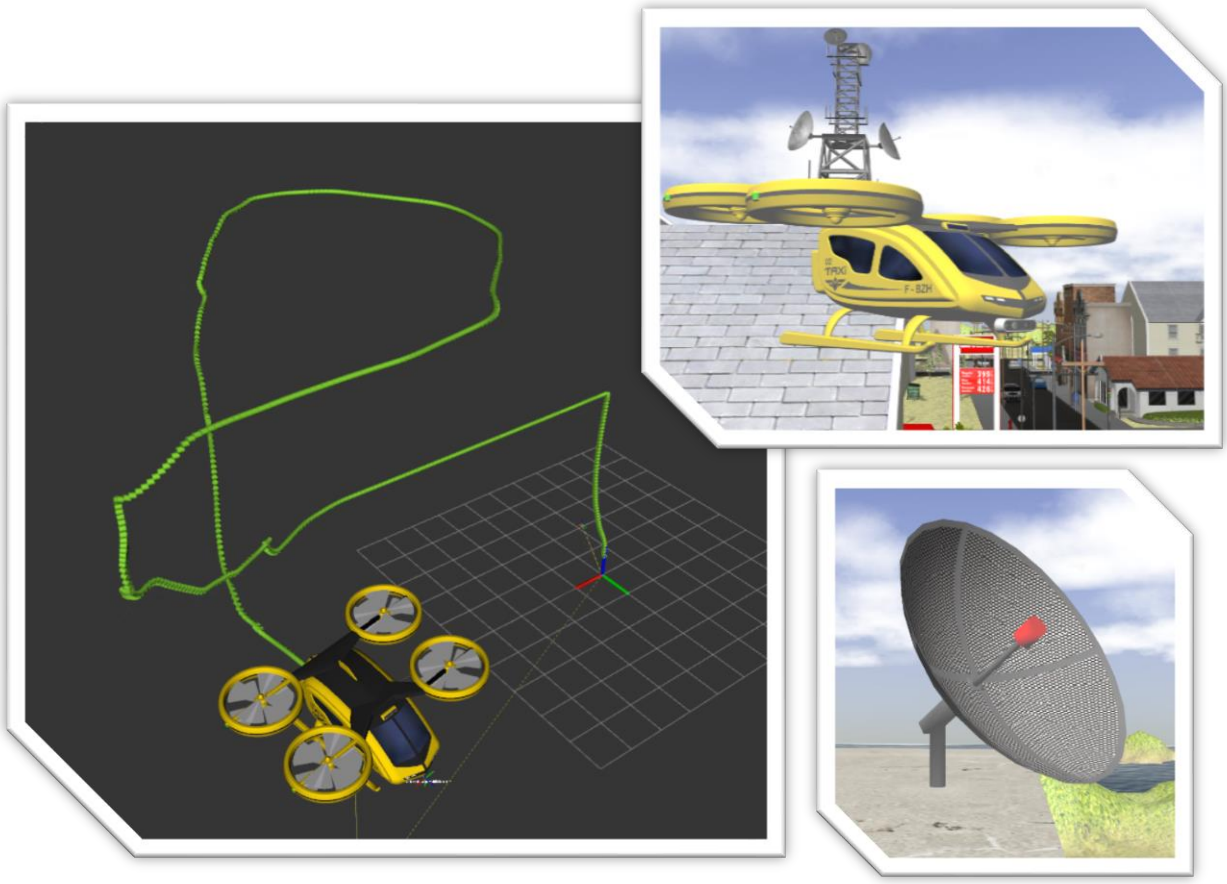
MEMORIA PROYECTO:
BETA-BOT LOCALIZATION

Memoria de trabajo desarrollado y resultados obtenidos en simulación de la localización de un robot aéreo tipo quadrotor mediante fusión sensorial enfocado a movilidad aérea urbana

Ampliación de Robótica – 4ºGIERM

Curso 2022/2023



AUTORES:

SERGIO LEÓN DONCEL

ARTURO RENATO ÚBEDA QUILÓN

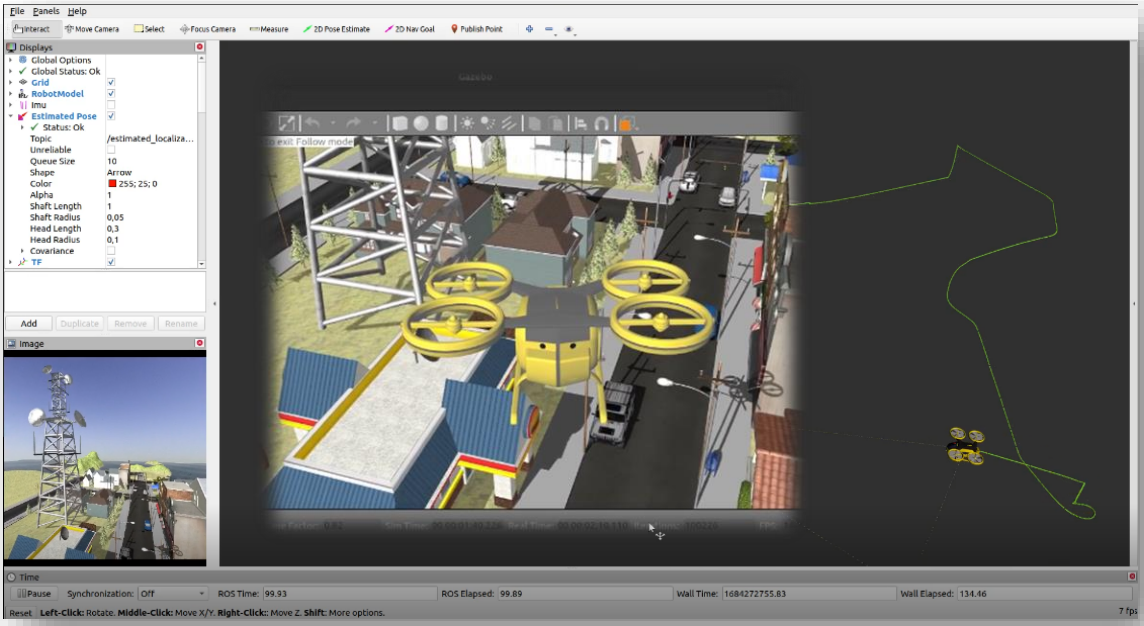
ÁLVARO GARCÍA LORA

CONTENIDOS

0. INTRODUCCIÓN

En la presente memoria se recogen las explicaciones relativas al proyecto de curso de la asignatura que versa sobre la localización de un robot aéreo tipo quadrotor. Para ello se usarán técnicas de fusión sensorial cuya base teórica se ha estudiado en la asignatura.

Ampliaremos los conceptos vistos y los llevaremos a la práctica desarrollando en ROS un entorno de trabajo. Usaremos tanto Gazebo como RViz para simulación y desarrollo de pruebas experimentales. Para la extracción y representación de resultados se trabajará en Matlab.



El enfoque que planteamos consiste en la aplicación hacia la movilidad aérea urbana, en la cuál es de suma importancia contar con una buena localización de los vehículos en las ciudades, con vistas a que estos sean cada vez más autónomos a la vez que seguros.

Partimos como base de un repositorio desarrollado por la comunidad [1], del cual usaremos algunos de los paquetes existentes como dependencias externas de los propios. Para ello es necesaria una adaptación de los mismos para lograr la compatibilidad en el entorno de trabajo.

Añadimos al proyecto la capacidad del control del vehículo en simulación mediante mandos de control de PS3/PS4 o XBOX. Esto resulta muy útil de cara al modo en control manual, tanto para realizar pruebas de desarrollo como para programar experimentos idénticos que requieren múltiples replicaciones para la extracción de resultados fiables. Incidiendo en esta última idea, se ha trabajado en métodos que posibilitan la grabación de las señales de control que se envían al vehículo, logrando de esta forma la replicación exacta de pruebas para comparar las distintas técnicas implementadas bajo las mismas condiciones.

Progresivamente incluiremos al sistema y entorno distintos sensores típicamente usados en esta área, analizando la influencia del uso de la información que proporcionan en la localización del robot aéreo. Aprenderemos de forma práctica en simulación a incluir y manejar los distintos tipos de sensores y datos que aportan.

Por tanto, el objetivo del proyecto no es únicamente la integración de múltiples sensores para conseguir una buena localización sino analizar las aportaciones de cada sensor, influencia en los parámetros de los mismos y comparar diferentes posibilidades de trabajo bajo mismas condiciones.

1. SENSORES USADOS

1.1. GNSS

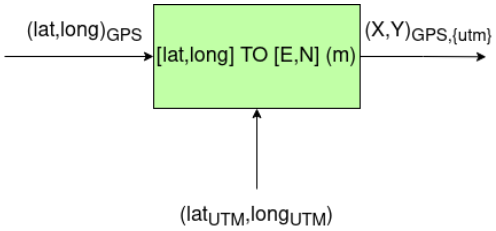
El sistema de referencia UTM es un sistema de coordenadas basado en el mapamundi plano de Mercator. Este sistema divide el mapamundi en 60 husos o regiones verticales, llamadas **zonas UTM (longitudinales)**, que a su vez distinguen entre zona norte y zona sur.

Los paquetes de acondicionamiento de sensores GNSS, como el que hemos implementado, tienen que convertir las lecturas del sensor [latitud, longitud, altitud] en [x,y,z] del robot respecto del marco de referencia fijo del mundo llamado "world" en nuestro caso. Para ello, toman un marco de referencia por encima de "world", llamado "utm". Seguimos los pasos que se muestran:

1. A partir de la latitud y la longitud absolutas del GNSS, determinamos en qué zona UTM estamos. El origen de esta zona, donde situaremos el **frame "utm"**, tendrá por longitud la del meridiano que se encuentra a 500 km al oeste del meridiano central de la zona, y por latitud la del ecuador (0º).

2. Dada la latitud y la longitud del GNSS, y usando como parámetro la longitud del meridiano que pasa por el origen de nuestra zona UTM, se pueden calcular las coordenadas UTM en metros, la posición X, Y del sensor GNSS respecto del frame "utm", respecto del origen de la subzona UTM mediante el **datum geodésico WGS84** (modelo matemático de la Tierra)

3. Conocida la posición (X, Y) del sensor respecto del marco de referencia "utm", trasladamos ahora las expresiones al marco de referencia "world".



1.2. BARÓMETRO

Empleado para medir la presión originada por el peso de la columna de aire de la atmósfera que tenemos sobre el sensor. Dado que la presión depende de la altura de dicha columna de aire, puede ser empleada para estimar la altitud a la que está ubicado el sensor respecto al nivel del mar.

Debemos tener en cuenta que la presión varía continuamente debido a las condiciones climatológicas por lo que no proporciona una medición absoluta de la altitud con precisión.

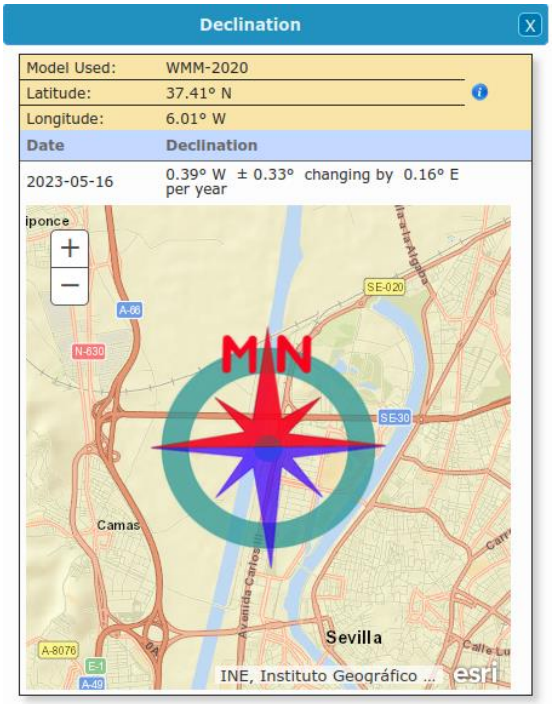
El plugin implementado usa como referencia el QNH: presión establecida en el altímetro de manera que el sensor nos indica la altura sobre el nivel del mar.

1.3. IMU (ACELERÓMETRO Y GIRÓSCOPO)

La IMU integrada en nuestro proyecto es de 6 ejes y combina tanto un acelerómetro como un giróscopo ambos de 3 ejes. Mientras que el giróscopo es un sensor incremental, y por tanto sólo lo usaremos en predicción, el acelerómetro será empleado en predicción/actualización en los distintos filtros EKF que implementaremos.

1.4. MAGNETÓMETRO

Un acelerómetro sólo puede estimar de forma absoluta roll y pitch comparando sus lecturas con la aceleración de la gravedad, pero no puede estimar yaw. El magnetómetro de 3 ejes integrado puede estimar de forma absoluta yaw a través de sus lecturas del campo magnético alrededor del sensor.



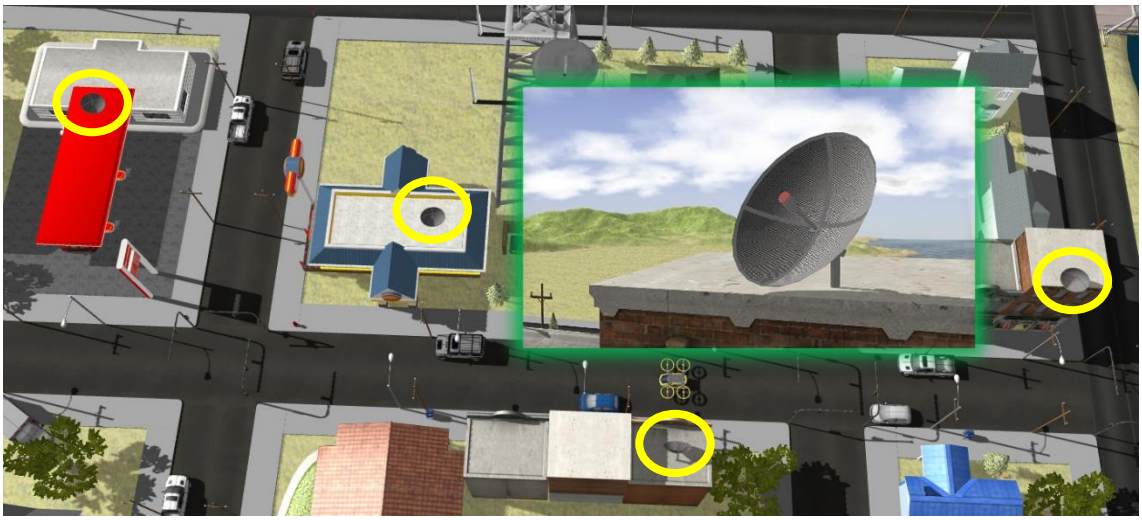
En ausencia de perturbaciones externas, el magnetómetro estimará el ángulo formado en yaw del eje X del sensor respecto del norte magnético. Sin embargo, dado que según la norma REP103 en ROS se trabaja en marcos de referencia ENU (X = Este, Y = Norte, Z = Up), el magnetómetro debe medir 0º cuando mire hacia el este. Por tanto, debe introducirse un offset de 90º.

Otro parámetro a tener en cuenta es la declinación magnética: la diferencia entre el norte geográfico y el norte magnético que detectan los magnetómetros. Esta diferencia depende del punto de la superficie terrestre donde se encuentre el sensor y va cambiando con el tiempo. Al tratarse de una simulación, nuestro mundo no cuenta con declinación magnética, por lo que no hemos necesitado tenerlo en cuenta.

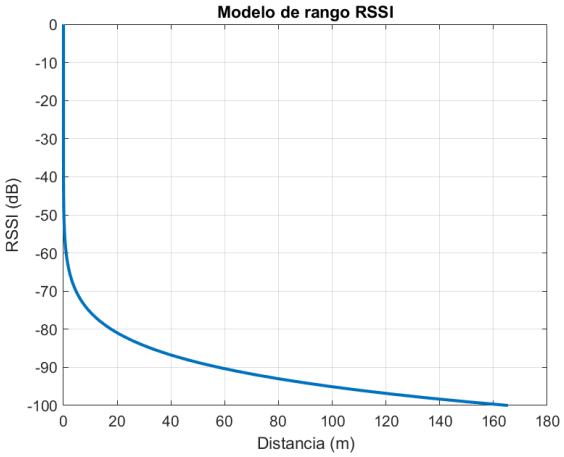
Declinación magnética en la ETSI

1.5. BALIZAS

Se utilizan 4 antenas colocadas estratégicamente en las terrazas de los edificios a lo largo de toda la ciudad, como balizas activas de radioemisión con alcance de hasta 100 m a su alrededor. Aunque a priori este número de balizas puede parecer insuficiente para obtener la precisión deseada, como se observará más adelante en los resultados extraídos de los experimentos, aporta más cantidad de información añadir otro tipo de sensores, a la vez que resulta más económico.



Las balizas incorporan un transmisor que envía señal con una determinada potencia que a su vez será recibida por el receptor incorporado en nuestro quadrotor. La señal recibida tendrá una energía menor debida a las pérdidas por el camino. Gracias a la escala RSSI (indicador de fuerza de señal recibida) se puede determinar la distancia entre ambos.

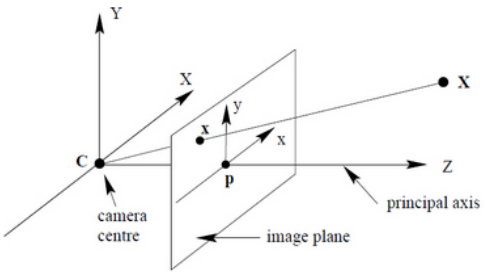


Para hacer una aproximación, se ha asumido un modelo de propagación en espacio libre [12] del modo: RSSI(D) = a log(D)+b, siendo a y b los coeficientes del modelo que se han ajustado experimentalmente.

Dado que no se encontraba desarrollado dicho modelo por la comunidad, se implementa dentro de un plugin propio trabajado bajo la API de Ignition Gazebo a bajo nivel.

1.6. CÁMARA

La cámara estéreo empleada sigue el modelo de pinole, un modelo de formación de la imagen que nos permite calcular dónde quedan proyectados los puntos de un escenario tridimensional sobre el plano de la imagen de una cámara que está capturando dicho escenario:



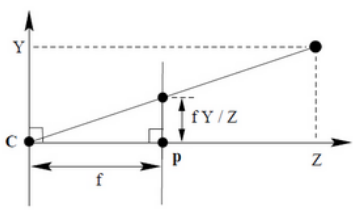
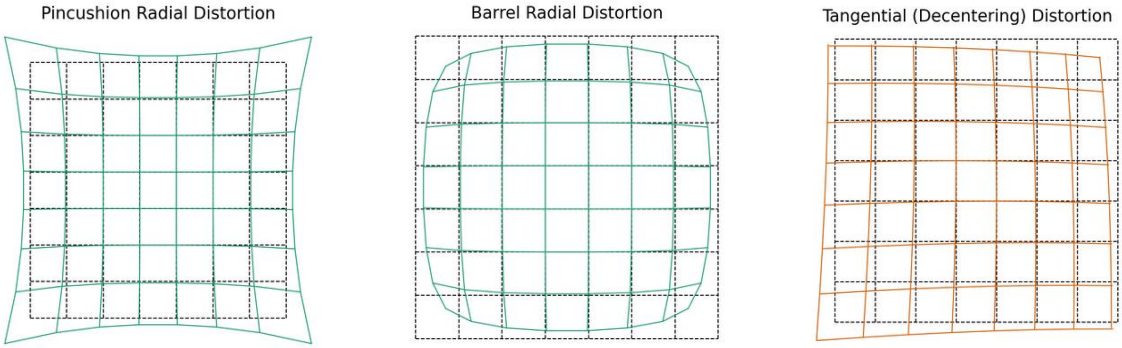


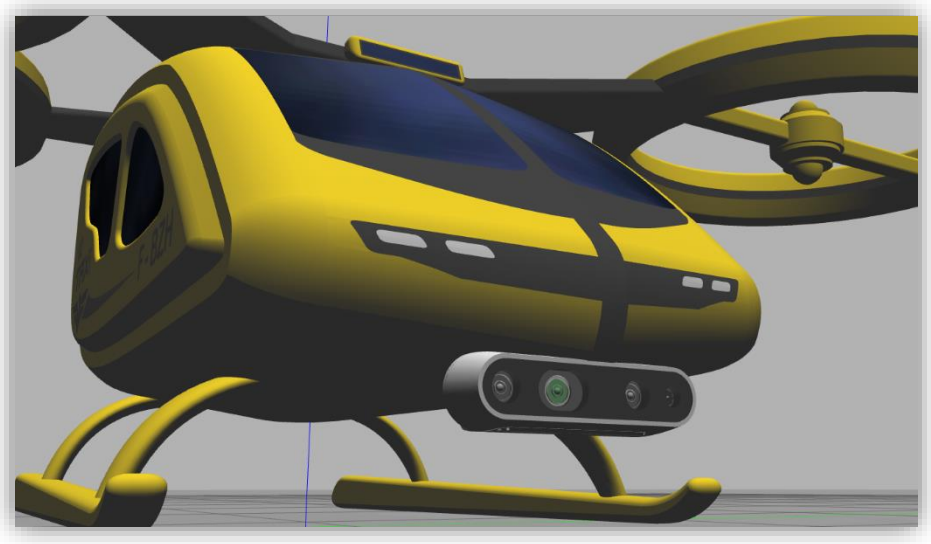
Figure 1. Pinhole

La distorsión se puede modelar como la suma de dos componentes: una radial, que hace que las líneas rectas aparezcan curvas en los bordes de la imagen, y otra tangencial, que ocurre cuando la lente de la cámara no está perfectamente paralela al plano de la imagen, y que repercute en una distorsión en forma de "comba" en la imagen:



Cuando usamos un par estéreo, aparecen parámetros adicionales. Uno de los incluidos ha sido el baseline (distancia entre cámaras). Es especialmente importante que, para realizar odometría visual, las imágenes se tomen en el mismo instante de tiempo. Muchos pares estéreo emplean una sincronización hardware para la toma de imágenes.

En nuestro proyecto, los parámetros de la cámara tales como la resolución de la imagen o los parámetros de distorsión se han introducido mediante una adaptación propia de un plugin de Gazebo.

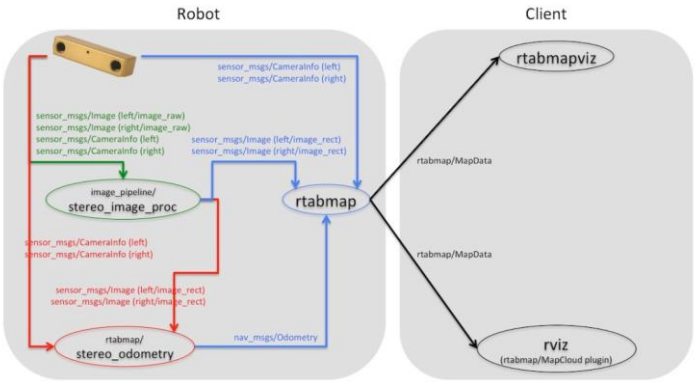


Cámara implementada en nuestro robot aéreo

Para poder realizar odometría visual a partir de un par estéreo, se deben resolver los siguientes pasos:

- Detección y correspondencia de puntos característicos de la imagen (features). En esta parte, algoritmos muy comunes suelen ser ORB, SURF, SIFT o FAST.
- Detección y eliminación de outliers. Una vez se encuentran features en las dos imágenes y su correspondencia en ambas imágenes, se emplean algoritmos que detectan emparejamientos erróneos (outliers) y los eliminan. Un ejemplo de algoritmo común sería RANSAC.
- Geometría de múltiples vistas para el cálculo de las coordenadas 3D de los features a partir de varias imágenes planas, donde se han ubicado dichos features. Típicamente, se pueden usar algoritmos de geometría epipolar.

Mostramos implementación en ROS de esta funcionalidad:



- Stereo_image_proc: procesamiento previo de las imágenes, retirando la distorsión. Es necesario especificarle los parámetros de distorsión.
- Nodo rtabmap/stereo_odometry: responsable de publicar /rtabmap/odom (VO – Visual Odometry), además de publicar la transformación odom -> base_link. Es el nodo que implementa la odometría visual, con todos los pasos mencionados anteriormente.
- Nodo rtabmap: responsable de publicar la transformación map -> odom (si se pretende hacer localización pura con la cámara, sigue la norma REP105).

2. IMPLEMENTACIÓN EKF. DESARROLLO EN FASES

La filosofía de trabajo a seguir consistirá en incluir de forma progresiva los distintos sensores con los que se trabajará, comparando y analizando sus aportaciones en los resultados de la estimación de la pose del robot.

Para ello se plantean distintas fases. Se ha desarrollado un nodo de inicialización común a todas que será usado para inicializar el vector de estados en el filtro.

2.1. INICIALIZACIÓN

Aprovechando la información de los sensores GNSS, barómetro, magnetómetro, acelerómetro y balizas repartidas en diferentes puntos del mapa, se ha implementado un nodo que sirva para inicializar las componentes del vector de estados en las futuras fases del filtro EKF.

Las suposiciones que impondremos sobre el sistema serán:

- Inicialmente el robot aéreo estará estacionado de forma estática. Por tanto, las velocidades y aceleraciones tanto lineales como angulares son nulas en el instante inicial.
- La pose correspondiente al estacionamiento será tal que el vehículo se encuentre en el plano del suelo del mapa o bien en un plano paralelo a éste (por ejemplo, en la azotea de algún edificio).

La información que aportan los sensores usados en la inicialización se tratará de la siguiente forma:

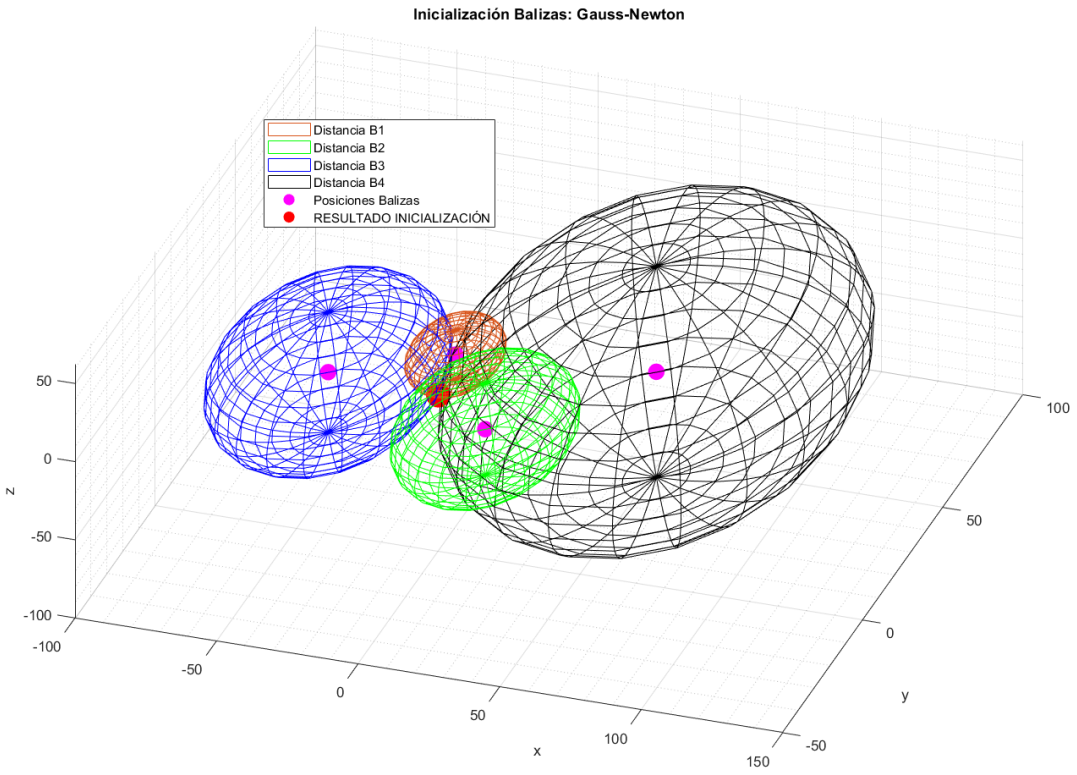
- GNSS: Usaremos los datos de las coordenadas ‘x’ e ‘y’ que aporta. Tomamos un número inicial de muestras y nos quedamos con su promedio.

- Barómetro: Usaremos la posición en ‘z’ disponible de su lectura. Nos quedamos igualmente con la media de un numero inicial de muestras tomadas.

- Balizas: Implementamos el algoritmo iterativo de Gauss-Newton basado en mínimos cuadrados para conocer cuál es la posición inicial en ‘x’, ‘y’ y ‘z’ en base a las posiciones de las balizas y las distancias a las que se encuentra en robot. Es importante destacar que dichas distancias presentan una cierta incertidumbre, de ahí la necesidad de usar este método.

  La idea principal es hallar cual es el punto que hace mínima la distancia simultánea a todas las superficies esféricas virtuales que se pueden formar con centro en las coordenadas en el espacio de cada baliza y con radio aquella distancia a la que cada baliza cree estar del robot. Visto de otra forma, la función a minimizar es la del error cuadrático total, suma de los errores al cuadrado de cada baliza individual, entendiendo dicho error como la distancia publicada por cada baliza y la calculada como la diferencia entre las coordenadas de la baliza en cuestión y las coordenadas del robot, las cuales se van calculando en cada iteración.

  Para asegurar una buena convergencia hacia la solución es necesario unos valores iniciales de los parámetros a hallar en el método (las coordenadas del robot) con los cuales comentar a iterar. Para ello se usan los correspondientes a las medidas iniciales del GNSS y Barómetro disponibles. Recalcamos que esto únicamente sirve como punto de inicio para iterar, es preferible de cara a la convergencia dar algún punto conocido que probar suerte o inventarse un punto inicial.



- Magnetómetro: Inicialización de yaw a partir de varias medidas iniciales.

- Acelerómetro: Inicialización de pitch y roll a partir de varias medidas iniciales comparando con el vector de gravedad.

La siguiente imagen muestra un ejemplo de los resultados obtenidos por el nodo de inicialización en una de las diversas pruebas lanzadas. El quadrotro se sitúa en el eje de referencia global del mundo, por lo tanto, todos los valores deben ser nulos.

Como puede apreciarse esto no es así, debido a la incertidumbre, pero está más próximo a la realidad que la medida cruda de los sensores.

```
---------------RESULTADOS--------------

INICIALIZACION GPS + BAROMETRO
x: 0.732337
y: 0.697055
z: 0.889524

INICIALIZACION BALIZAS
x: -0.613435
y: -0.181117
z: 0.964006

INICIALIZACION ORIENTACIÓN
roll: 0.00550674
pitch: 0.00208329
yaw: -0.0126809
```

2.2. FASE 1: IMU Y GNSS+BARÓMETRO

Los sensores usados en esta primera fase de desarrollo serán GNSS, barómetro IMU (acelerómetro, giróscopo y magnetómetro).

Realmente no tenemos modelos del sistema ni sensores no lineales en esta fase, se trata de un KF, pero lo implementamos como EKF por unificar con las próximas fases. Los resultados deben ser los mismos ya que las ecuaciones que intervienen son lineales.

- Predicción:
  - Giróscopo: A partir de medidas de velocidades angulares podemos obtener orientación mediante integración.
  - Acelerómetro: A partir de medidas de aceleraciones lineales podemos conseguir las velocidades lineales mediante integración y la posición mediante doble integración.

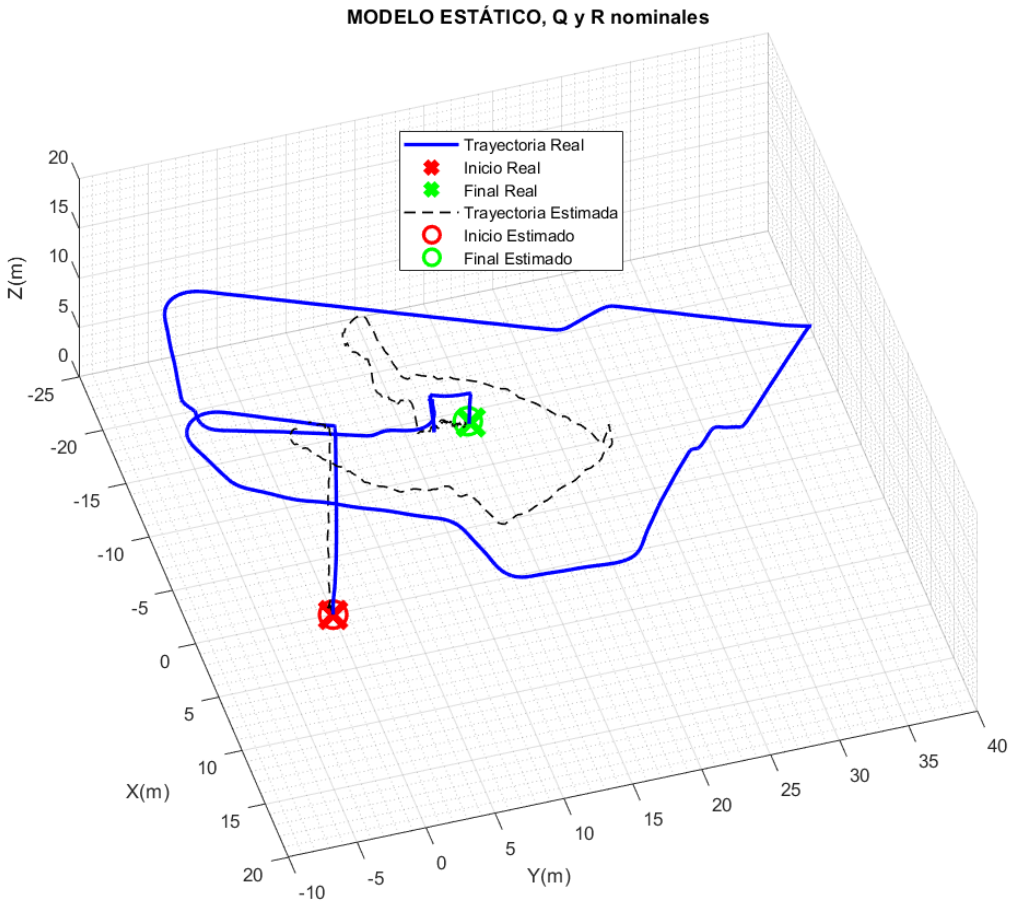
- Actualización:
  - GNSS: Aporta posición x e y de forma absoluta.
  - Barómetro: Aporta altura en z de forma absoluta.
  - Acelerómetro: Comparando las mediciones con el vector de gravedad obtenemos medidas absolutas de roll y pitch.
  - Magnetómetro: A partir de sus mediciones podemos extraer de forma absoluta la orientación en yaw.

En esta primera fase se ha decidido plantear distintos modelos para el sistema, los cuales serán usados en la etapa de predicción. Concretamente, se han lanzado experimentos con los modelos estático, MRU y MRUA. A continuación, vemos cada uno de ellos con más detalle y los resultados obtenidos en simulación. Todos los resultados experimentales recogidos en los siguientes apartados han sido tomados de multiples replicaciones para el recorrido del robot aéreo a lo largo de una misma trayectoria.

2.2.1. MODELO ESTÁTICO

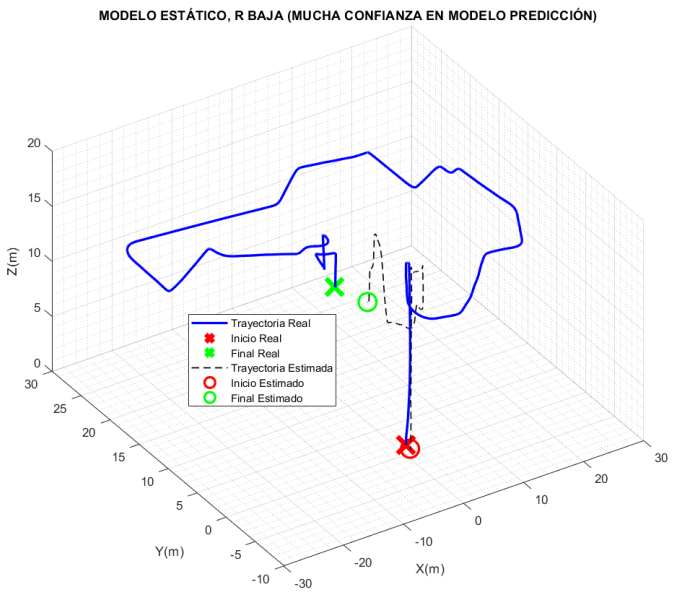
Empecemos por el caso más simple de todos: la pose predicha del robot de un instante al siguiente es la misma, permanece estático, aunque sabemos que esto realmente no ocurre en general a no ser que esté quieto realmente. De esta forma no usamos acelerómetro ni giróscopo para predecir, sino que la corrección de los errores se hará en la etapa de actualización a partir de las medidas de GNSS, barómetro y sensores de la IMU.

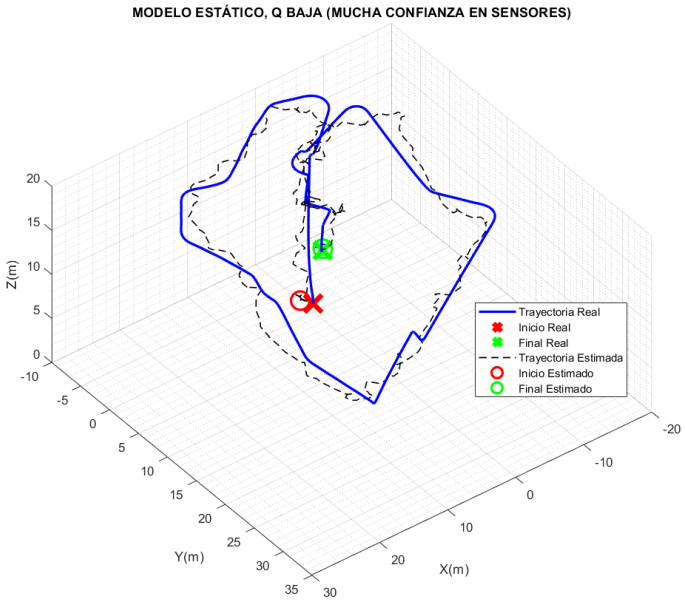
A continuación, se muestran las trayectorias estimadas frente al ground truth y los resultados estadísticos tras varias replicaciones.



	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	4.78	1.28	15.79	3.3
	Y [m]	5.97	0.54	21.87	0.4
	Z [m]	0.15	0.01	1.22	0.07
	Distancia [m]	8.45	1.15	23.36	0.49
ORIENTACIÓN	Roll [º]	3.03	0.38	33.46	1.21
	Pitch [º]	3.24	0.53	65.81	20.59
	Yaw [º]	2.4	0.3	34.43	1.43
	Ángulo [º]	6.3	0.84	69.54	21.58

Se ha variado la confianza sobre modelo y sensores alterando las matrices R (matriz de covarianzas del modelo de predicción) y Q (matriz de covarianzas de los sensores) con el fin de mejorar la estimación frente a la trayectoria real sin mejorar el modelo de predicción. Se muestran los resultados obtenidos:

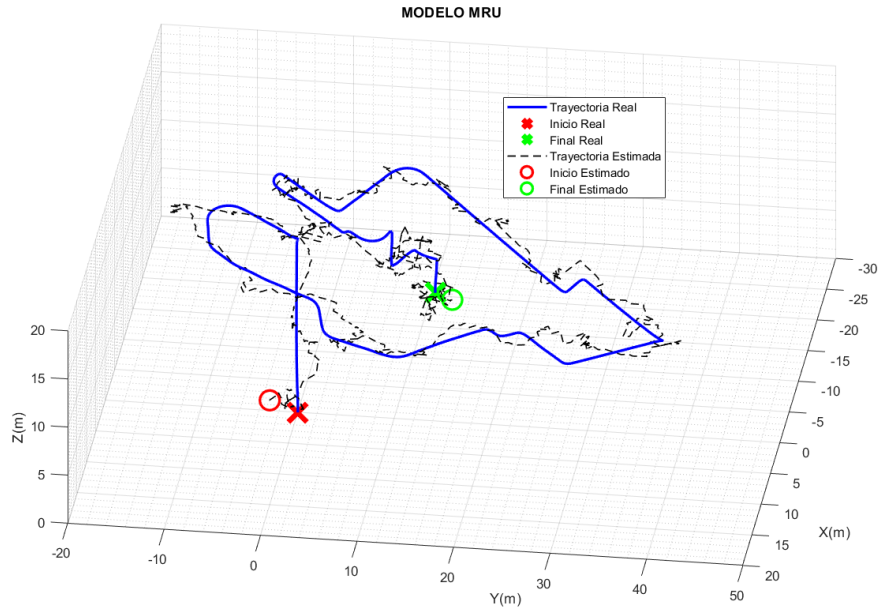




2.2.2. MODELO MRU

Usamos en predicción un modelo que tienen en cuenta velocidades de un instante al siguiente. Las medidas del giróscopo son usadas para hallar la orientación e incluimos en el vector de estados la velocidad lineal del vehículo para la predicción de la posición. La velocidad instantánea la suponemos constante en el modelo. Conseguimos estimar las velocidades lineales a partir de medidas actuales y anteriores de GNSS y barómetro, ya que no disponemos de ningún sensor que mida directamente velocidad lineal.

	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	1.28	0.12	5.19	0.84
	Y [m]	1.34	0.16	5.54	0.65
	Z [m]	0.16	0	1.22	0.12
	Distancia [m]	2.08	0.2	6.46	0.82
ORIENTACIÓN	Roll [º]	3.36	0.17	34.58	0.72
	Pitch [º]	3.76	0.23	67.28	11.7
	Yaw [º]	1.44	0.06	20.01	6.9
	Ángulo [º]	6.37	0.36	69.19	12.69

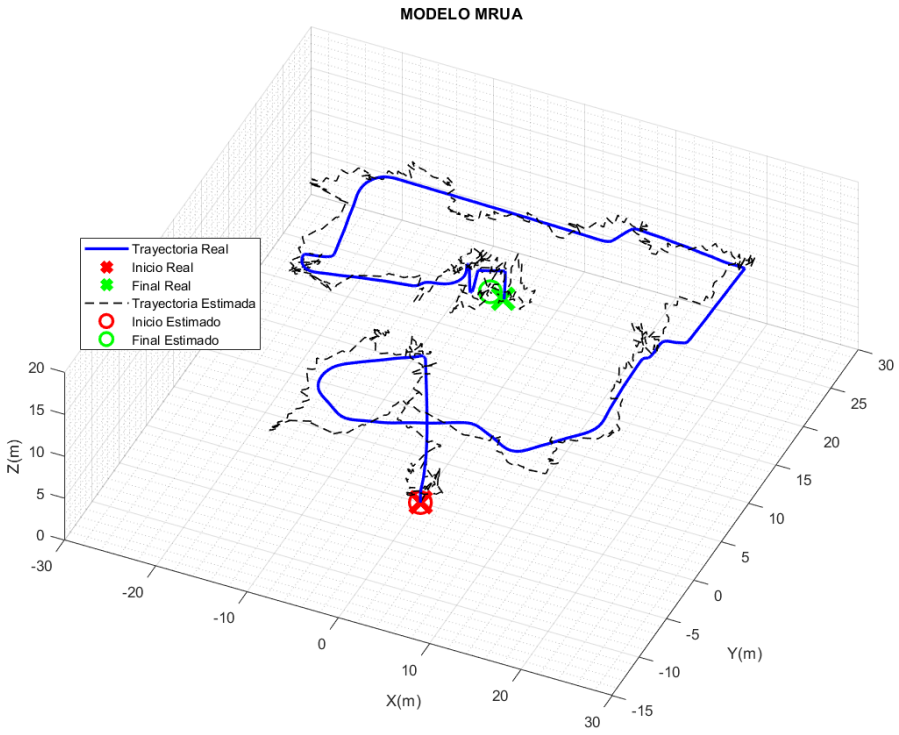


2.2.3. MODELO MRUA

Incluimos en el modelo para predicción aceleraciones respecto al modelo anterior. Usamos las medidas del acelerómetro para la parte correspondiente a la posición. Como disponemos directamente de esas medidas no necesitamos estimarlas.

En esta ocasión las velocidades lineales predichas en el vector de estados son variables gracias a incluir las medidas del acelerómetro.

	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	1.39	0.12	5.88	1.12
	Y [m]	1.26	0.11	5.17	0.69
	Z [m]	0.19	0.01	1.42	0.06
	Distancia [m]	2.11	0.11	6.72	0.63
ORIENTACIÓN	Roll [º]	2.94	0.63	45.09	22.13
	Pitch [º]	3.18	0.65	69.71	6.09
	Yaw [º]	1.27	0.17	27.18	14.05
	Ángulo [º]	5.52	1.04	77.93	12.32



A la vista de los resultados obtenido podemos ver que existe una mejora al usar alguno de los modelos alternativos al estático. Sin embargo, no se aprecian grandes cambios entre el rectilíneo uniforme y el uniformemente acelerado. Es por ello por lo que en los futuros apartados se usará el MRU por simplicidad.

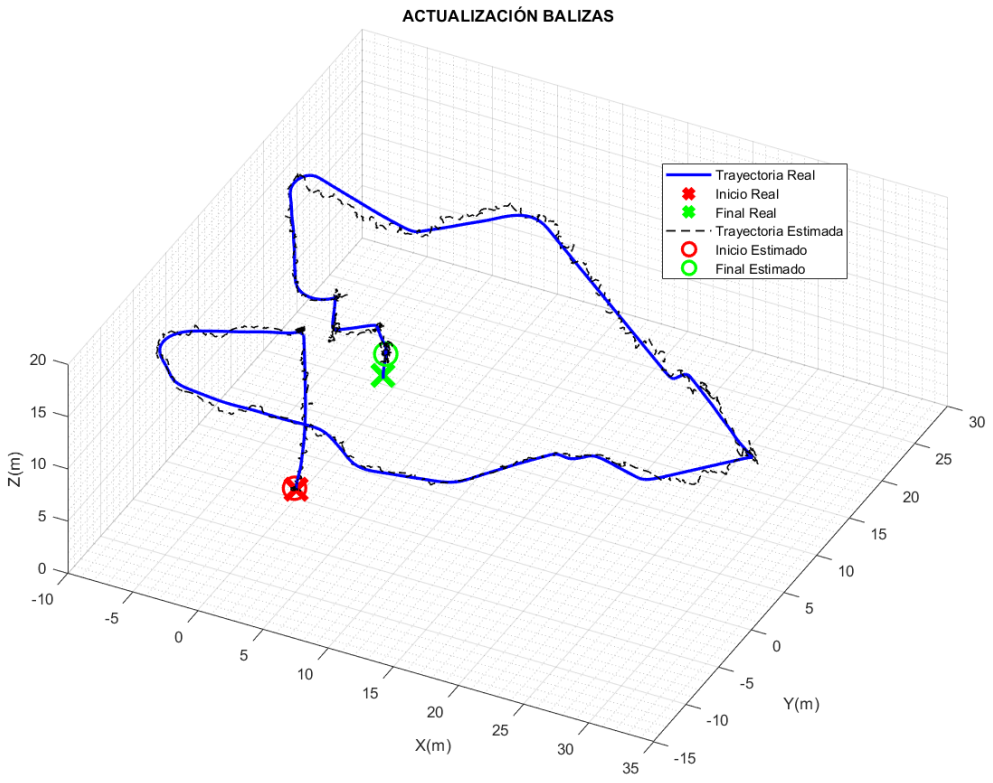
2.3. FASE 2: IMU Y BALIZAS

Los sensores usados en esta fase serán los contenidos en la IMU, como en la fase anterior, pero usaremos una serie de balizas dispuestas en el mapa en lugar del GNSS + barómetro.

La diferencia respecto a la primera fase reside en la actualización. En predicción no hay cambios, tomamos como se ha comentado el modelo MRU.

En este caso, dado que el modelo de las balizas es no lineal, el filtro implementado si que se trata de un EKF en toda regla.

- Predicción: Mismo modelo que en Fase 1.
    - Giróscopo.
    - Acelerómetro.
- Actualización: Cambiamos GNSS+barómetro por balizas.
    - Balizas: Hallar posición de forma absoluta a partir de distancias medidas al robot y coordenadas de las balizas.
    - Acelerómetro
    - Magnetómetro

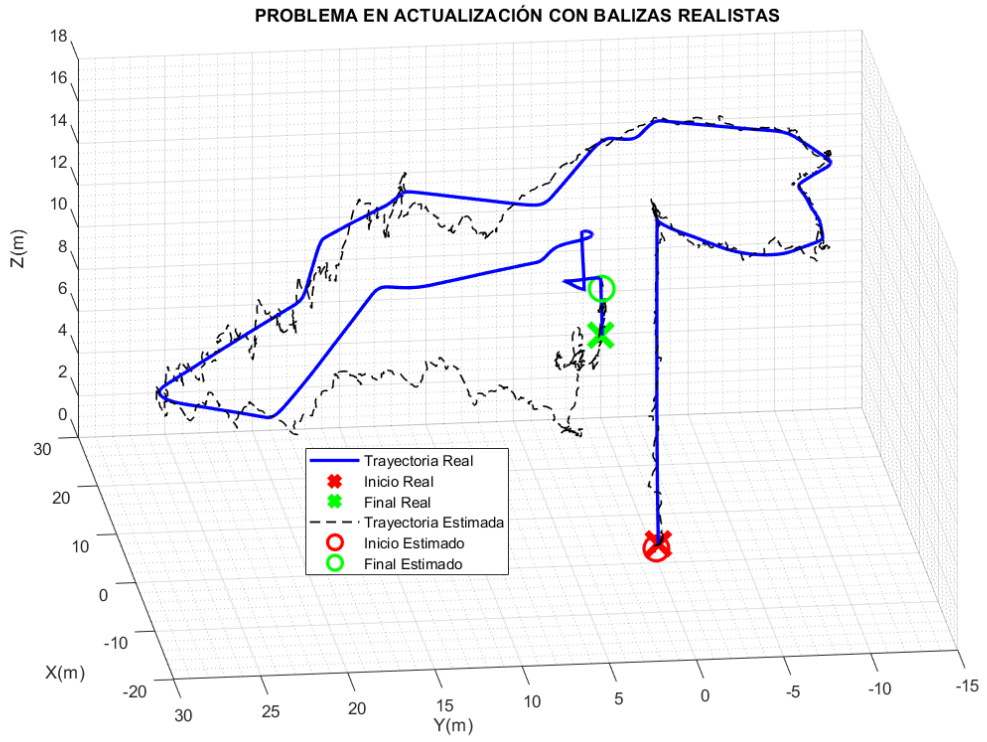


	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	0.17	0.04	1.03	0.22
	Y [m]	0.2	0.03	1.11	0.22
	Z [m]	0.72	0.47	5	2.49
	Distancia [m]	0.83	0.47	5.09	2.5
ORIENTACIÓN	Roll [º]	3.35	0.45	47.15	9.06
	Pitch [º]	3.41	0.45	81.24	3.25
	Yaw [º]	1.31	0.15	141.11	8.63
	Ángulo [º]	6.16	0.77	142.07	9.71

En comparación con el EKF desarrollado en la fase 1, la actualización mediante balizas de la fase 2 consigue unos márgenes de ruido sensiblemente inferiores a los obtenidos en la fase 1. Esto se debe a la elevada incertidumbre del GPS y al tiempo de actualización. Las balizas emiten medidas a una mayor frecuencia que los datos recibidos del GPS. En consecuencia, en situaciones con cambios de movimientos bruscos, al tener una actualización más lenta se cometen más errores en estimación respecto a la trayectoria real.

Es importante comentar la existencia de un problema que ocurre en algunas replicaciones.

Debido al bajo número de balizas (solamente estamos usando 4 balizas distribuidas por la ciudad), errores existentes en las distancias que proporcionan al robot y la limitación en alcance de las mismas, vemos como se comete un error en altura con simetría respecto a la trayectoria real. Además, la información que proporcionan las balizas en altura es más pobre que la obtenida en x e y. Esta cuestión supone también una limitación realista puesto que, para distribuir los sensores en un mayor rango de alturas, necesitaríamos de edificios con gran diferencia de cotas, lo cual no siempre se da en ciudades. Por tanto, el problema se debe al uso y colocación realista de las balizas. Este tipo de fallos ocurren aproximadamente en el 20 % de las pruebas.



Las opciones posibles que ofrecemos para lidiar con este problema sería por un lado incluir una mayor cantidad de balizas en el mapa, distribuidas en zonas de tal forma que se consiga una mayor cantidad de información. Esta solución sería en la práctica costosa.

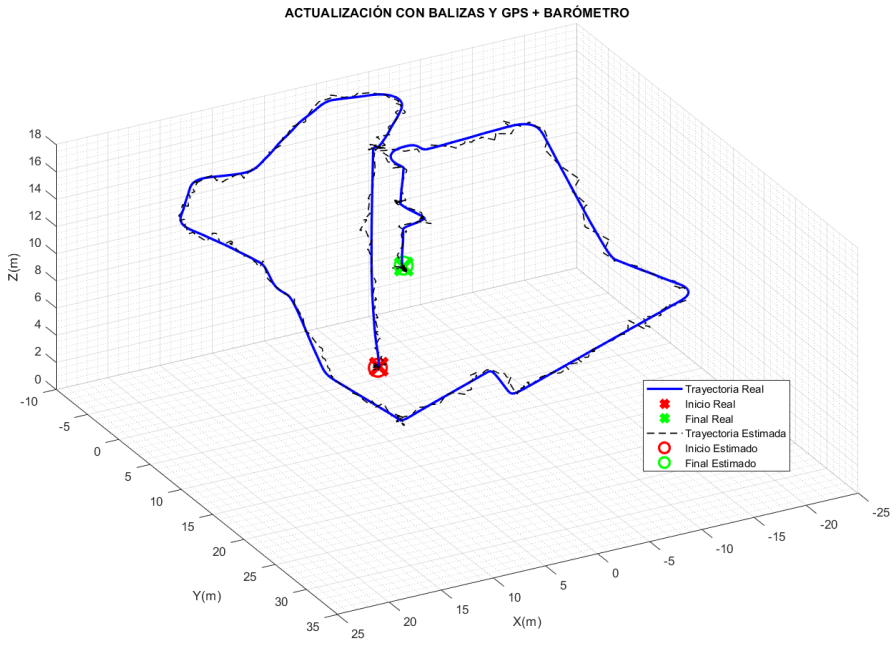
Por otro lado, se plantea la posibilidad de incluir los sensores GNSS y barómetro con los que ya cuenta el sistema y que en esta fase no han sido usados. Es previsible que incluyendo una mayor cantidad de medidas de distintos sensores en actualización se consiga evitar el problema por completo, logrando solucionar esta cuestión.

2.4. FASE 3: COMBINACIÓN GNSS+BARÓMETRO CON BALIZAS

Esta fase supone la integración de las dos anteriores con el propósito de estudiar la mejora que resulta al integrar varios métodos de actualización. Por tanto, los sensores a usar serán tanto GNSS+Barómetro como balizas, además de los incluidos en la IMU.

- Predicción: Mismo modelo que en Fase 1 y Fase 2.
  - Giróscopo
  - Acelerómetro
- Actualización: Todo lo visto hasta ahora.
  - Balizas
  - GNSS
  - Barómetro
  - Acelerómetro
  - Magnetómetro

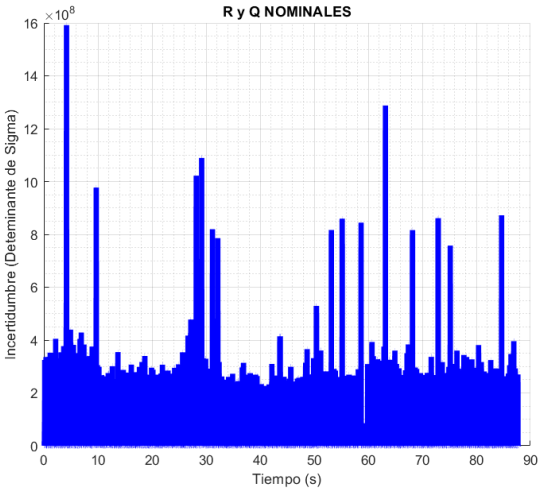
En resultados vemos que nos hemos desecho del problema que teníamos en algunas replicaciones de la fase de desarrollo anterior.

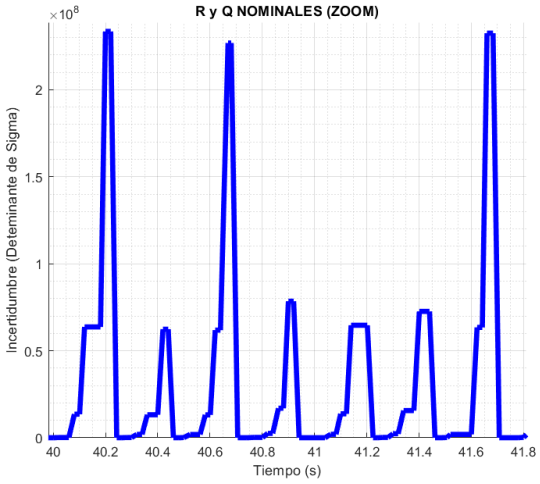


En cuanto a los errores cometidos observar una disminución comparando la tabla de resultados actual con la mostrada en la fase 2. De esta forma corroboramos la mejora que supone una mayor la fusión sensorial.

	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	0.3	0.03	1.65	0.22
	Y [m]	0.27	0.02	1.44	0.24
	Z [m]	0.15	0.01	1.13	0.04
	Distancia [m]	0.51	0.02	2.02	0.3
ORIENTACIÓN	Roll [º]	3.33	0.29	35.37	2.76
	Pitch [º]	3.57	0.41	66.28	10.37
	Yaw [º]	1.53	0.17	46.69	3.02
	Ángulo [º]	6.33	0.62	67.51	8.19

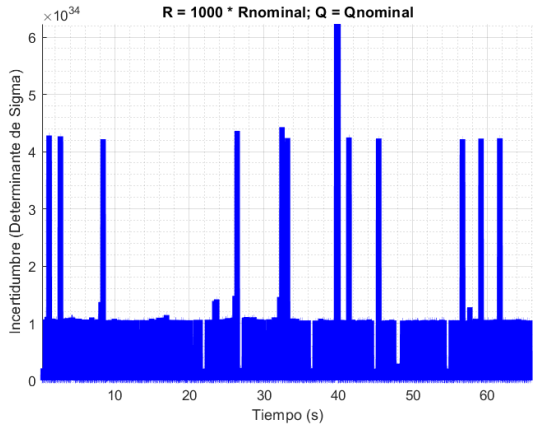
Presentamos en este punto una métrica adicional a las vistas. Como resultado de la estimación, el filtro proporciona la incertidumbre estimada. Tomaremos el determinante de esta matriz de covarianzas que ofrece el filtro (determinante de sigma). Esta métrica puede interpretarse como una medida de la confianza que el EKF tiene en su estimación.

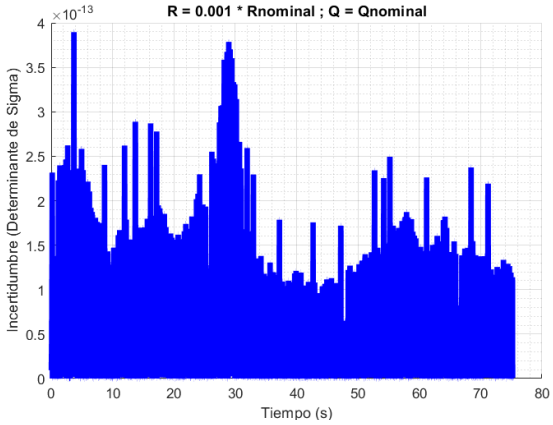


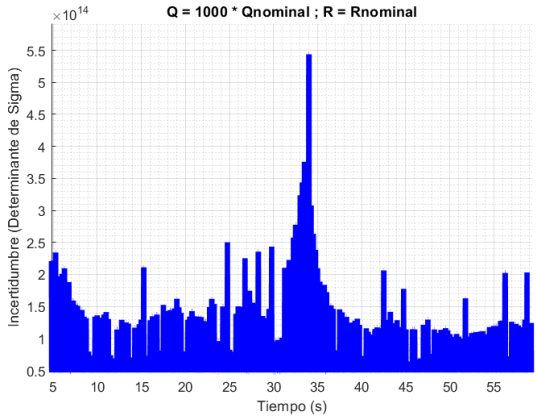


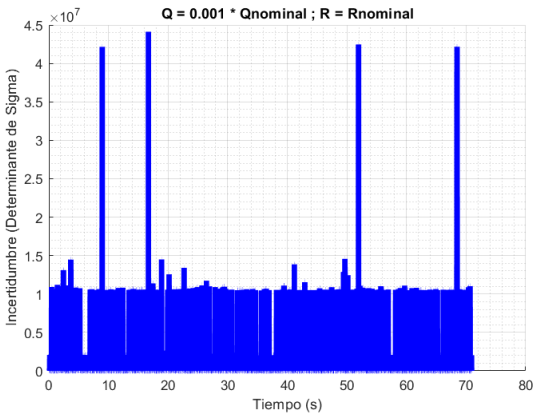
En la figura ampliada, se aprecia una forma dentada. Esto se debe a que cada vez que se ejecuta la predicción, el EKF confía menos en su estimación. En su etapa de actualización, el EKF contrasta la información de sus sensores absolutos con la estimación de la pose para corregirla y se genera como resultado la forma dentada comentada.

Comparamos a continuación diferentes situaciones con valores distintos de R y Q:









- Si disminuimos la confianza en los sensores de actualización, la confianza que el filtro tiene en su estimación disminuye (la incertidumbre aumenta varios órdenes de magnitud). En caso contrario, al disminuir Q (aumentar confianza), la incertidumbre disminuye.
- Si disminuimos la confianza en el modelo de predicción (aumentamos R), la confianza que el filtro tiene en su estimación disminuye (la incertidumbre aumenta varios órdenes de magnitud). En caso contrario, la confianza en la estimación aumenta mucho.

2.5. FASE 4: ODOMETRÍA VISUAL

Incluimos en esta fase una cámara para tener disponible odometria visual.

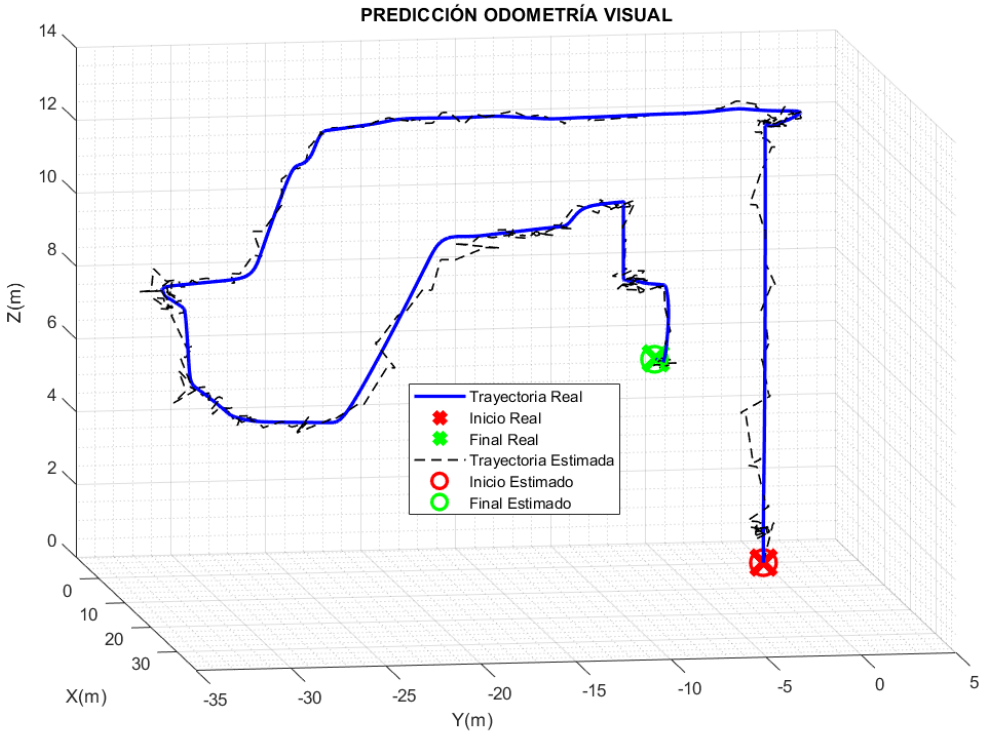
Por tanto, la diferencia respecto a la fase anterior será la parte de predicción mientras que la actualización contará con todo lo disponible hasta ahora visto.

- Predicción: Odometria visual por cámara en lugar de IMU.
  - Cámara: en esta fase aporta todas las componentes del vector de estados en predicción.

- Actualización: Mismo que Fase 3.
  - Balizas
  - GNSS
  - Barómetro
  - Acelerómetro
  - Magnetómetro

A la vista de los resultados vemos que esta fase desarrollada con los datos obtenidos de las imágenes de la cámara supone un buen método para la parte de predicción. Sin embargo, no se observa una mejoría respecto a la fase anterior.

	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	0.4	0.01	1.99	0.28
	Y [m]	0.39	0	2.31	0.59
	Z [m]	0.16	0	1.15	0.04
	Distancia [m]	0.67	0.01	2.59	0.33
ORIENTACIÓN	Roll [º]	4.47	0.32	111.56	89.38
	Pitch [º]	5.08	0.09	73.51	4.49
	Yaw [º]	3.09	0.62	85.15	49.2
	Ángulo [º]	9.16	0.41	126.23	62.98



Los resultados obtenidos empleando únicamente odometría visual muestran un correcto seguimiento de la trayectoria real. Es necesario comentar que la odometría visual se implementa a través de nodos externos que realizan su propia estimación de la pose, sin atender a la fusión sensorial realizada por el EKF, de tal manera que la deriva cometida por la odometría visual se mantiene en el tiempo.

Para evitar esta situación y hacer que la etapa de actualización del EKF elimine el drift de la odometría visual, esta es reseteada cada vez que se ejecuta la actualización del filtro a la pose obtenida tras esta etapa. De esta manera, el drift es eliminado, y en la próxima iteración, la odometría visual no partirá de una posición con deriva. Este efecto puede apreciarse en el despegue inicial del quadrotor: se dan varios ciclos de predicción de la odometría visual donde se acumula deriva y con la llegada de la actualización reducimos significativamente el error, para partir en la próxima predicción desde una pose muy cercana al ground truth.

Otra problemática apreciable en la gráfica y que puede ocurrir en general es que la odometría visual se pierda. Para corregir esta situación, hemos introducido un modelo de predicción de respaldo: un supervisor detecta si la odometría visual se pierde (deja de enviar medidas), y entonces entra en predicción el MRU con la IMU, hasta que se vuelvan a obtener lecturas de la odometría visual, en cuyo caso se restaura esta fuente de información.

3. LOCALIZACIÓN COMPLETA. INTEGRACIÓN FASES

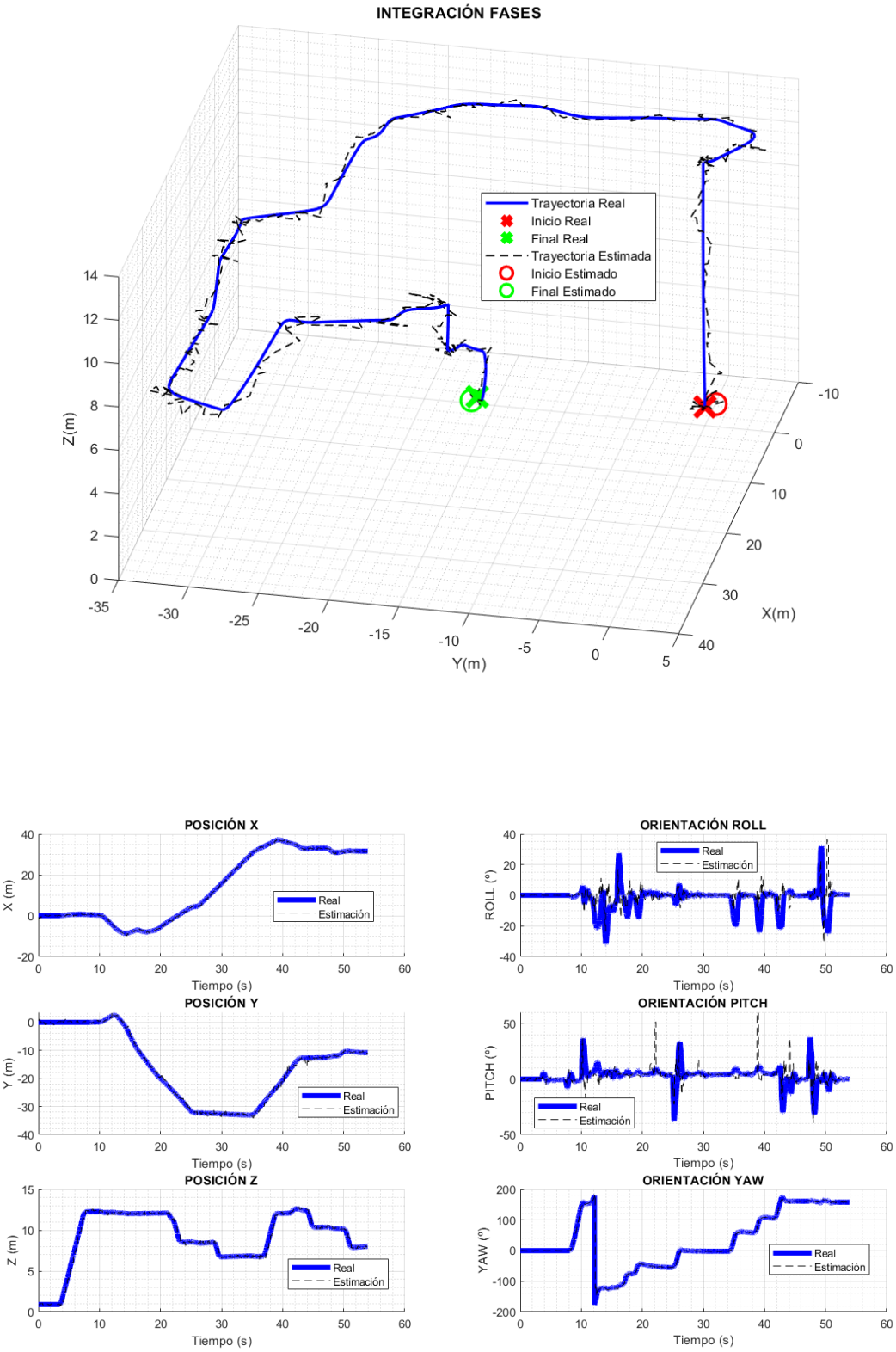
El objetivo de este apartado es el de tener todos los sensores y fases anteriores en una versión final de localización que integre los mejores resultados obtenidos y la mayor cantidad de información por parte de los sensores para lograr la mayor fusión sensorial con los sensores disponibles.

En este enfoque más general, usamos la odometría visual para la predicción de la posición, y el giróscopo de la IMU para la predicción de la orientación. De esta manera, cada uno de los dos sensores proporciona componentes del vector de estados diferentes y, de esta forma, evitamos emplear técnicas de fusión dentro de la etapa de predicción (como puede ser el empleo del Método de Máxima Verosimilitud, planteado durante la elaboración de este trabajo).

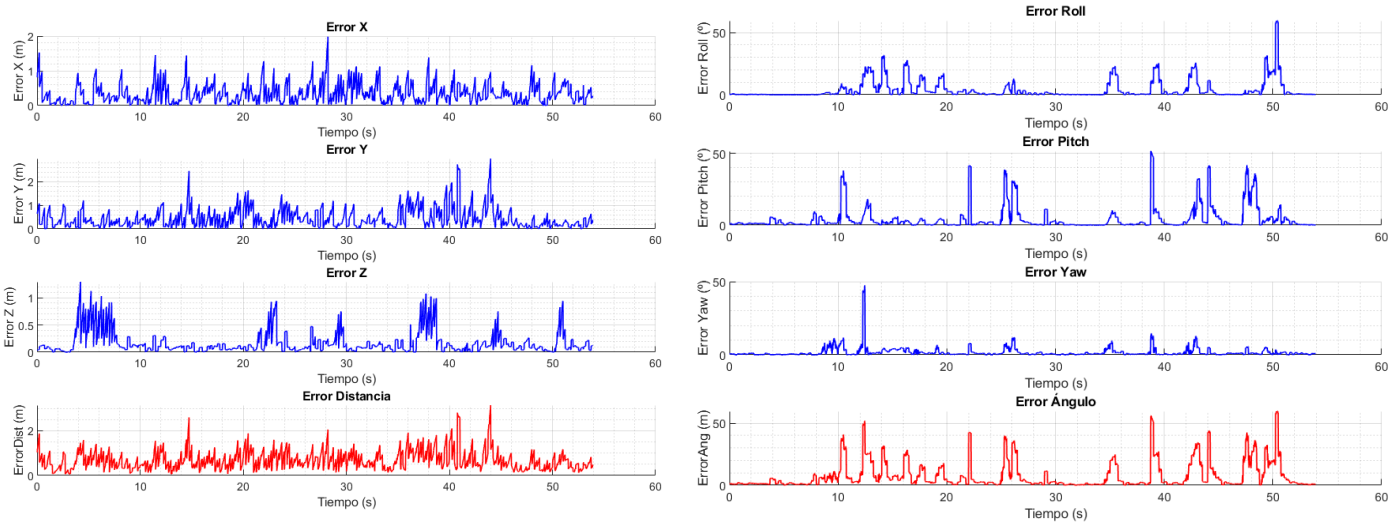
- Predicción:
  - Cámara
  - Giróscopo
  - Acelerómetro
- Actualización:
  - Balizas
  - GNSS
  - Barómetro
  - Acelerómetro
  - Magnetómetro

En este caso daremos un estudio más minucioso al ser la fase de integración. Mostramos las evoluciones temporales de las componentes de la pose del robot y de los errores respecto al ground truth.

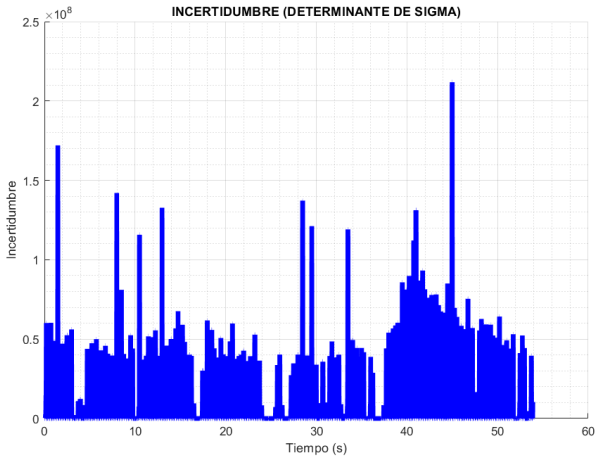
	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
POSICIÓN	X [m]	0.34	0.02	1.91	0.33
	Y [m]	0.4	0.1	2.44	0.75
	Z [m]	0.17	0.02	1.25	0.25
	Distancia [m]	0.64	0.09	2.76	0.72
ORIENTACIÓN	Roll [º]	4.26	0.68	72.45	60.98
	Pitch [º]	4.02	0.5	58.34	24.15
	Yaw [º]	2.94	1.62	68.83	47.33
	Ángulo [º]	8.02	1.27	89.97	48.6



Como puede verse, la estimación de posición queda solapada bajo la real, no cometiendo errores mayores a los 3 metros y en general, con una desviación típica del orden de los centímetros.



En las gráficas anteriores se muestra la evolución temporal de los errores en posición y orientación respecto a la trayectoria real.



	COMPONENTES POSE COMPLETA	MEDIA DE ERRORES MEDIOS	DESV.TÍP DE ERRORES MEDIOS	MEDIA DE ERRORES MÁXIMOS	DESV.TÍP DE ERRORES MÁXIMOS
Errores orientación FASE 4					
ORIENTACIÓN	Roll [º]	4.47	0.32	111.56	89.38
	Pitch [º]	5.08	0.09	73.51	4.49
	Yaw [º]	3.09	0.62	85.15	49.2
	Ángulo [º]	9.16	0.41	126.23	62.98
Errores orientación INTEGRACIÓN					
ORIENTACIÓN	Roll [º]	4.26	0.68	72.45	60.98
	Pitch [º]	4.02	0.5	58.34	24.15
	Yaw [º]	2.94	1.62	68.83	47.33
	Ángulo [º]	8.02	1.27	89.97	48.6

También vemos el determinante de sigma, ya explicado este concepto en detalle en la fase 3. En la izquierda hacemos una comparativa entre errores cometidos en orientación respecto a la fase 4.

En vista de los resultados obtenidos, podemos afirmar que la IMU realiza una mejor estimación de la orientación que la odometría visual, a la que vamos a encargar por tanto sólo la estimación de la posición.

Debemos tener especial cuidado a la hora de incluir varios sensores en predicción, dado que ambos sensores funcionan a tasas diferentes. En actualización, esto no es un problema tan crítico, dado que la actualización de la pose entre dos instantes de tiempo puede hacerse de forma razonablemente precisa tomando simplemente la última medida de cada sensor absoluto porque las medidas son independientes entre sí.

En el contexto de los modelos de predicción, la estimación en el instante siguiente se calcula incrementalmente a partir de la pose en el instante anterior. Hemos elegido como tasa de la etapa de predicción la frecuencia a la que llega la odometría visual, que es más lenta que la IMU.

Mientras llega una nueva medida visual, la IMU debe acumular sus sucesivas predicciones para ser consistente con la estimación entre los instantes de tiempo donde se ejecuta la predicción completa.

4. CONCLUSIONES

En este trabajo, se han explorado las posibilidades del Filtro de Kalman Extendido, un filtro donde la inclusión de sensores de diferente naturaleza plantea una serie de retos que se pueden superar gracias a la flexibilidad de este estimador.

La integración de diferentes sensores nos conduce potencialmente a métodos de localización más precisos, pero también implica la aparición de estructuras matemáticas y computacionales más complejas: el filtro escala en dificultad con el número de sensores a fusionar.

Afortunadamente, el EKF permite ampliar su funcionamiento base con todo un ecosistema de módulos adicionales que extienden su funcionalidad con labores tales como sincronización de sensores, comprobación/supervisión de la estimación, valoración de la salud de los sensores (detección de outliers y frecuencia con la que aparecen), optimización del cómputo (aumento de dimensiones de matrices a mayor número sensores usados), filtrado previo de los sensores, etc.

Además, la integración e inclusión de nuevos sensores puede implicar la necesidad de volver a ajustar la confianza en todos los sensores para asegurar unos niveles de error similares o mejores a los de una situación anterior con una menor cantidad de estos dispositivos.

También hemos comprobado que no siempre las estimaciones mejoran al añadir más sensores: en general, se dispone de mayor información y eso debería repercutir en un seguimiento más preciso de la trayectoria real, pero en la práctica, la integración de nuevos sensores puede degradar el funcionamiento del sistema debido a una mayor cantidad y fuentes de ruidos.

Por todo ello no nos podemos quedar en la práctica con una única versión de filtro como la mejor de todas. Esto depende de la aplicación concreta, los errores que se consideran aceptables, la cantidad y tipo de sensores de los que se dispone… Hemos desarrollado las distintas posibilidades para conocer y estudiar las repercusiones que tiene cada integración añadida y combinaciones implementadas.

5. AMPLIACIONES FUTURAS

La naturaleza abierta de este proyecto abre la puerta a toda una serie de líneas de trabajo futuras que pueden ampliar el alcance del mismo:

- Cierre de la estructura de navegación de un robot autónomo: mientras este proyecto está enfocado al estudio de métodos de localización de taxis aéreos en el contexto de la movilidad urbana, para la resolución de esta tarea será necesario también el desarrollo de controladores de vuelo y de planificadores en tres dimensiones para el trazado de la trayectoria aérea deseada y así trasladar pasajeros o mercancías de un lugar a otro de las ciudades del futuro.

- En este mismo contexto, será necesario contar con una planificación no sólo global, sino también local. Los taxis autónomos del futuro no sólo tendrán que calcular la mejor ruta para trasladarse, sino que tendrán que adaptarse de forma correcta a las normas de tráfico y ofrecer una navegación respetuosa con la seguridad de peatones, vehículos y edificios. De esta manera, será imprescindible desarrollar labores de conducción autónoma, para lo que se puede aprovechar la visión estéreo de la que dispone el robot.

- La inclusión de otros sensores puede permitir mejorar la estimación de la pose del robot siempre que se haga un correcto acondicionamiento y procesamiento de estos. Este proyecto puede ampliar su alcance abordando la localización mediante sensores Lidar 3D. La estimación también puede mejorar aumentando la densidad de balizas en la ciudad, lo que además permite no sólo a nuestro robot, sino a otros robots destinados a otros proyectos que también puedan enriquecer su localización.

- Coordinación de flotas de drones: el trabajo con flotas de robots aéreos puede emplearse en labores de mapeo y levantamiento topográfico, vigilancia y seguridad, inspección de infraestructuras, distribución y entrega eficientes, etc.

- Resolución del problema de mapeo del entorno, como hemos comentado: el estudio del entorno por parte del quadrotor es imprescindible para su correcta navegación. De esta forma, se resolvería el problema de localización y el de mapeo.

- Otra posibilidad sería la ejecución de algoritmos de SLAM, una línea de trabajo que permite resolver los problemas de localización y mapeo conjuntamente.

- Implementación dual y comparativa con Filtro Extendido de Información (EIF), ya que podría permitir un menor coste computacional y una red de procesamiento distribuida.

6. REFERENCIAS

[1] “Hector-quadrotor: hector_quadrotor ported to ROS Noetic & Gazebo 11”: https://github.com/RAFALAMAO/hector-quadrotor-noetic

[2] “Wiki ROS”: http://wiki.ros.org/

[3] Sebastian Thrun, Dieter Fox, Wolfram Burgard, “Probabilistic Robotics” (1999-2000)

[4] “Pinhole Camera Model”: https://hedivision.github.io/Pinhole.html

[5] “Camera Modeling: Exploring Distortion and Distortion Models, Part I”: https://www.tangramvision.com/blog/camera-modeling-exploring-distortion-and-distortion-models-part-i

[6] Peter Corke, “Robotics, Vision & Control” (2011-2017)

[7] “Outdoor Global Localization · Robot & Chisel”: https://www.robotandchisel.com/2020/05/01/outdoor-navigation/

[8] “UTM Zone Characteristics”: https://www.e-education.psu.edu/natureofgeoinfo/book/export/html/1696

[9] “Altimeter Pressure Settings”: https://www.skybrary.aero/articles/altimeter-pressure-settings

[10] “Norma REP 103 ROS”: https://www.ros.org/reps/rep-0103.html

[11] “Magnetic Field Calculators”: https://www.ngdc.noaa.gov/geomag/calculators/magcalc.shtml?useFullSite=true#declination

[12] J. Ramiro Martínez-De Dios, Aníbal Ollero: “On-Line RSSI-range model learning for target localization and tracking” (2017)