Ampliación de Robótica

Mapping de balizas de radio PF

Realizado por:

Roberto Morales Entonado

Fernando Román Hidalgo

Andrés Martínez Márquez

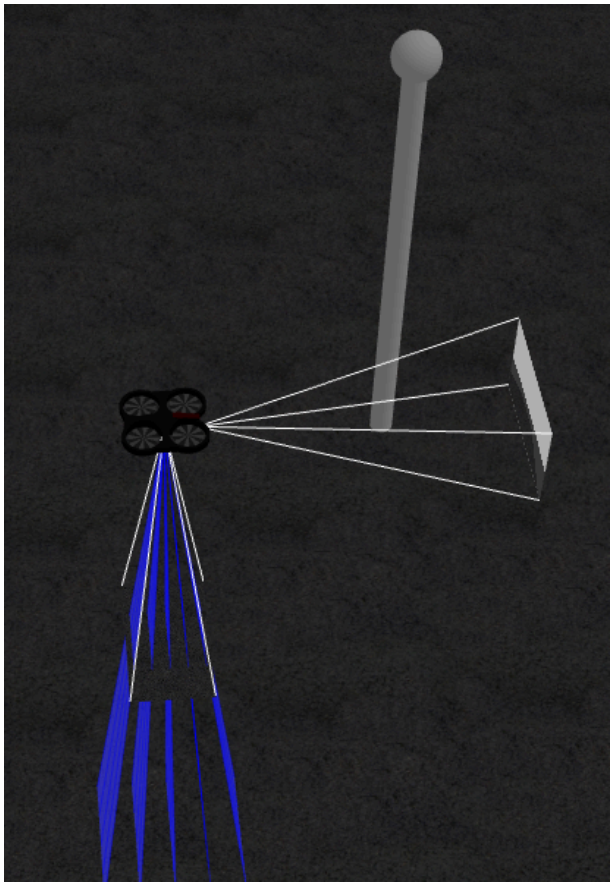
Marcos Ortiz Durán

Índice

1. Introducción

En este trabajo se aborda el desarrollo de técnicas para el mapeo de balizas de radiofrecuencia (RF) en un entorno tridimensional, partiendo de un escenario en el que dichas balizas están ubicadas en posiciones desconocidas. El objetivo principal consiste en diseñar un método que permita estimar la posición 3D de las balizas a partir de las mediciones de distancia obtenidas, teniendo en cuenta condiciones realistas como la presencia de ruido en las señales, limitaciones de alcance y la posibilidad de pérdida de datos debido a errores en la transmisión.

Para realizar esta estimación, se ha optado por emplear el filtro de partículas, una técnica probabilística que permite manejar eficazmente la incertidumbre y las características no lineales del sistema. Este estudio, implementado en *ROS 2*, es fundamental para mejorar la percepción y navegación de sistemas robóticos en entornos complejos, donde la localización precisa de referencias externas es crucial.



2. Entorno de trabajo

El desarrollo del proyecto se ha realizado utilizando *ROS2 Humble* y *Gazebo 11*, herramientas estándar para simulación y control en robótica. Para la simulación del robot aéreo se eligió el paquete *sjtu_drone*, que ofrece un modelo de dron quadrotor compatible con *Gazebo* y *ROS2*. Esta combinación facilitó la integración y validación del método de mapeo basado en filtro de partículas en un entorno simulado realista.

3. Desarrollo del proyecto

3.1. Mapa y visualización

Para crear el entorno de simulación donde poder probar el filtro, se decidió partir del repositorio *sjtu_drone*, que consta de un dron quadrotor con teleoperación completamente listo para usarse desde *Gazebo*. Para mayor comodidad, se utilizó un mundo vacío predeterminado que venía en el repositorio, dejando solamente los elementos básicos como el suelo y la luz.

Además, para poder tener una referencia gráfica de las balizas en el simulador, se creó un archivo *beacon.sdf* que consiste en un palo con una esfera en la punta. Este modelo es utilizado para la representación de las balizas.

Con este modelo 3D se ha configurado un archivo *.launch.py*, que genera sobre el mapa vacío `n` balizas con posiciones aleatorias dentro de un rango predefinido.

Para la visualización de los topics publicados por los filtros de partículas, se ha incluido en el *.launch.py* un apartado en el que se inicializa Rviz2 con una configuración ya guardada previamente que permite ver la posición de cada baliza estimada como un `Pose` y las partículas que han llevado a esa estimación como un `PoseArray`, ambas del mismo color. El archivo de RViz se ha configurado para 5 balizas, pero se podrían añadir cuantas quisiésemos, ya que los nodos están preparados para ser independientes del número de balizas.

De esta forma, ya se tiene un entorno sobre el que probar el filtro de partículas y observar de forma dinámica los resultados.

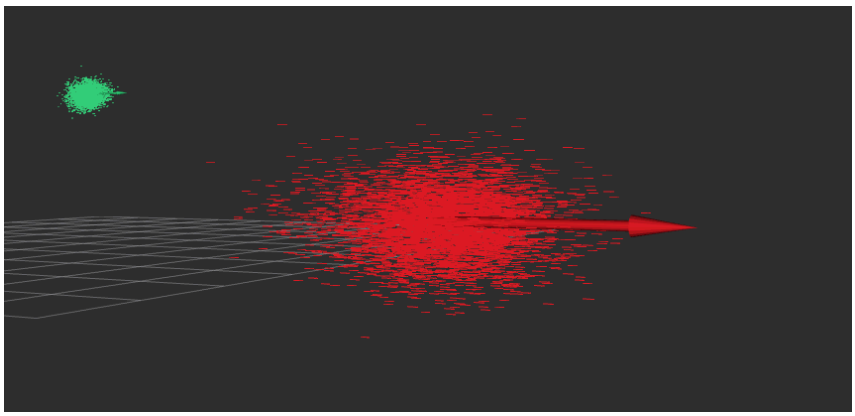


Figura 1: Nube de partículas sobre una baliza

Además, para una mayor precisión en la visualización de resultados, al ejecutar el código del filtro de partículas, este genera archivos *.csv* en los que se guardan los valores de estimación de las balizas y la posición real de estas para, posteriormente, ejecutar el script de python *graficas.py* y poder obtener gráficas de la estimación de las balizas en función del tiempo, además de otros parámetros que serán interesantes en apartados posteriores.

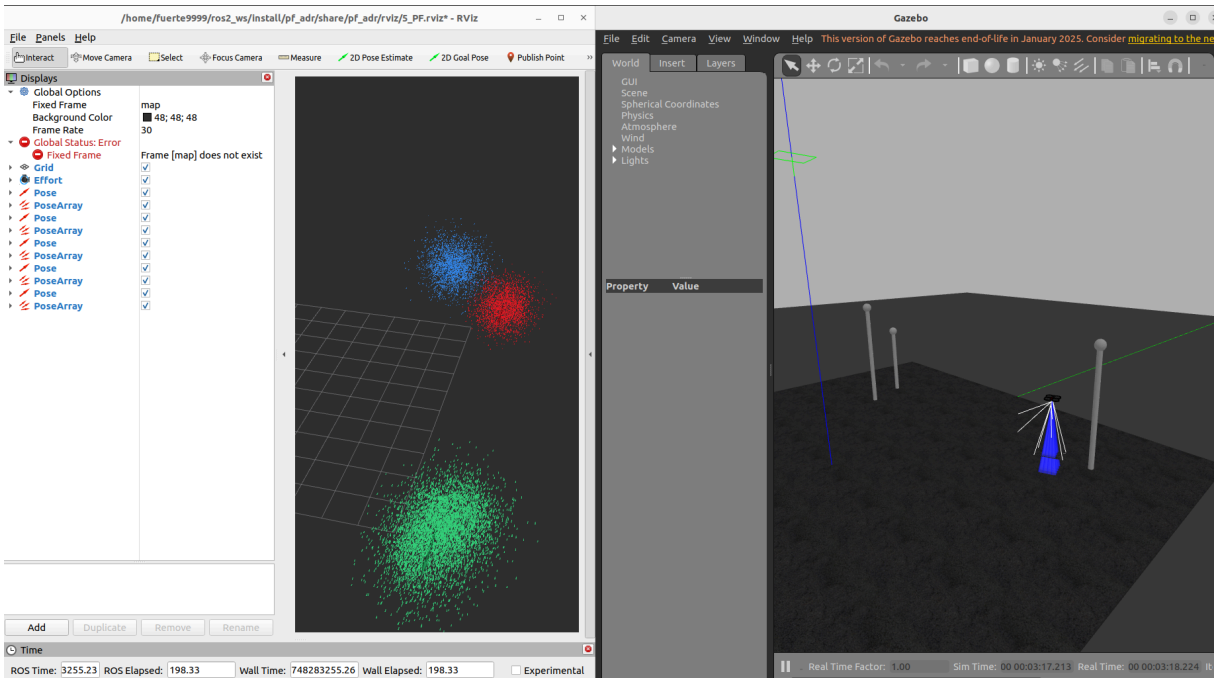


Figura 2: Entorno de trabajo

3.2. Balizas

Para simular el comportamiento de las balizas en este sistema, se ha desarrollado un nodo específico que simula su funcionamiento. El nodo publica una medición de distancia basada en la posición relativa entre el dron y cada baliza fija en una posición conocida. Dicha posición es la misma que en la que se coloca el modelo 3D en la simulación. Este nodo se suscribe al tópico `/simple_drone/odom`, desde donde obtiene la posición del dron, que se supone perfectamente conocida. Con esta información, el nodo calcula la distancia euclídea entre el dron y la ubicación de la baliza y publica este valor en un tópico exclusivo, `/beacon_{id}/distance_to_target`, siendo id el identificador de la baliza que hemos lanzado (dependerá del número de balizas).

Para emular situaciones más realistas, el nodo introduce con cierta periodicidad valores fuera de rango, que simulan outliers. Estos se generan cada n número de mensajes, donde n es una muestra aleatoria tomada dentro de un rango predefinido (entre 90 y 120 mensajes). Se publicará un valor no válido, ya que estará fuera del rango de distancia de la baliza. Además, todos los mensajes que tengan una distancia mayor que 5 metros publicarán una distancia nula, simulando así un posible alcance máximo de una baliza.

Por otro lado, el ruido gaussiano típico de las mediciones reales no se añade en el nodo de la baliza, sino que se incorpora directamente en el nodo del filtro de partículas.

3.3. Filtro de partículas

El filtro de partículas es un algoritmo de estimación basado en muestras (partículas), que representa la distribución de probabilidad del estado a estimar mediante un conjunto de partículas ponderadas con pesos. En este caso, el estado que se desea estimar es la posición de una baliza fija en el espacio tridimensional, representado como:

$$x = (x_b, y_b, z_b)$$

El objetivo final del filtro es obtener una estimación precisa de x a partir de mediciones realizadas por un dron en movimiento.

En la fase inicial, llamada inicialización, se generan N partículas distribuidas en el espacio 3D donde se cree que podría estar la baliza. En esta etapa, se asignan pesos uniformes a todas las partículas:

$$w_i = \frac{1}{N}, \quad \text{para } i = 1, 2, ..., N$$

En nuestras implementaciones, las partículas se distribuyen en una esfera centrada en el origen $(0, 0, 0)$ o un cubo centrado en $(0, 0, 0.5)$, debido a la total incertidumbre sobre la posición inicial de la baliza.

Una vez tenemos las partículas inicializadas, pasaremos a la fase de predicción. Normalmente, esta fase utiliza un modelo de movimiento para propagar las partículas, sin embargo, dado que la baliza es estática, nuestro modelo de movimiento es nulo:

$$x_i^t = x_i^{t-1}\}$$

Por tanto, se omite la fase de predicción en este caso.

Pasada la fase de predicción, entramos en la fase de actualización/observación. En la fase de observación, se utilizan las mediciones (distancia a la baliza y posición del dron) para ajustar los pesos de cada partícula. Se calcula la distancia estimada desde la posición del dron x_{dron} a cada partícula x_i:

$$\hat{z}_i = |x_i - x_{\text{dron}}|$$

Luego, se compara con la distancia medida por el sensor, z, y se asigna un peso a cada partícula en función de un modelo gaussiano:

$$w_i = \exp\left(\frac{-(z-\hat{z}_i)^2}{2\sigma^2}\right)$$

Donde σ representa la desviación estándar del ruido del sensor. Posteriormente, los pesos se normalizan:

$$w_i = \frac{w_i}{\sum_{\{j=1\}}^{N} w_j}$$

Por último, para evitar la degeneración del filtro (donde unas pocas partículas concentran casi todo el peso), se realiza un re-muestreo. Este consiste en seleccionar N nuevas partículas de la población actual con probabilidad proporcional a sus pesos. En nuestras implementaciones, se ha utilizado el método de remuestreo multinomial, basado en la función de distribución acumulativa. A las nuevas partículas se les añade un pequeño ruido gaussiano para mantener la diversidad:

$$x_i = x_i + N(0, {\sigma_{\text{ruido}}}^2)$$

Finalmente, los pesos se reinicializan de forma uniforme:

$$w_i = \frac{1}{N}$$

Este enfoque permite al dron estimar de forma robusta la posición de una baliza fija en un entorno 3D, incluso en presencia de ruido e incertidumbre en las mediciones.

3.3.1. Filtro de partículas básico

Cada filtro se suscribe a un topic individual que contiene las mediciones de distancia a la baliza correspondiente. Por tanto, para un sistema con `n` balizas, existirán `n` topics del tipo `beacon_{id}/distance_to_target`, y `n` instancias de filtros de partículas, cada una asociada exclusivamente a uno de estos topics. Las posiciones estimadas se publicarán en los topics `/pf/beacon_{id}/estimate` y las partículas de la estimación se publicarán en `/pf/beacon_{id}/particles`.

Se ha elegido esta estructura porque tener un filtro de partículas por cada baliza favorece la independencia y el paralelismo, permitiendo que cada nodo calcule su estimación sin necesidad de compartir información o sincronizarse con otros nodos. Además, permite que el sistema se mantenga robusto frente a errores en una de las fuentes de medición.

El diseño del PF solamente toma datos de la odometría del dron, que se supone perfectamente conocida (aunque se le añade un pequeño ruido) y de la distancia a la baliza, que se obtiene de la distancia estimada a la baliza específica que se desea localizar.

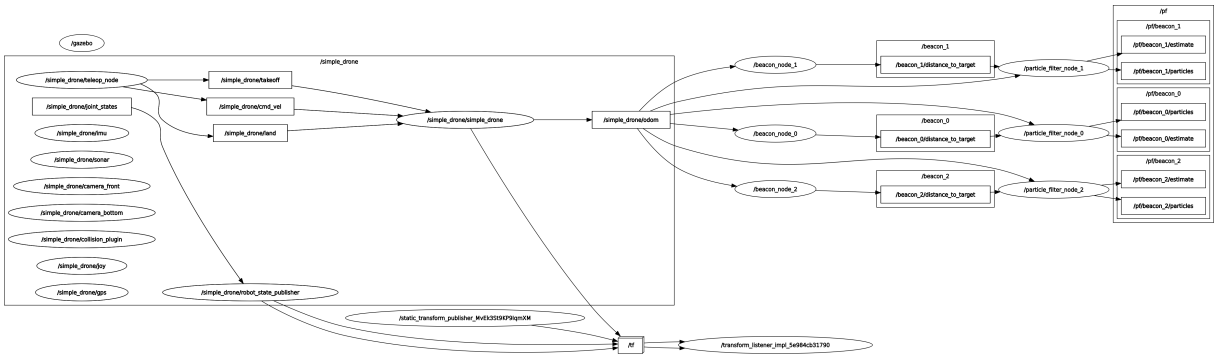


Figura 3: Nodos usados en el filtro sencillo

En cuanto al tratamiento de los datos, cada filtro incorpora un mecanismo de detección y rechazo de mediciones no válidas y outliers, consistente en que si la medida está fuera del rango de envío de la baliza, esta se rechaza. Esta estrategia permite considerar un rango máximo de envío y que el filtro mantenga una distribución de partículas coherente, sin que la estimación se vea afectada mediciones incoherentes.

Finalmente, cabe destacar que, manteniendo un número constante de partículas en cada filtro, se consigue asegurar una carga computacional predecible y homogénea, que puede ser medida para así saber los requerimientos computacionales que serían necesarios para implementar este algoritmo en un ordenador de a bordo.

3.3.2. Filtro de partículas inteligente

Para reducir el coste computacional del filtro de partículas, se ha planteado un sistema inteligente de redistribución de partículas. Este se basa en repartir un número total fijo de partículas entre las balizas que el dron está detectando activamente en cada instante. La idea principal es evitar asignar partículas a balizas que no pueden ser detectadas por el dron por distancia, lo que permitiría operar con mayor eficiencia sin comprometer la precisión.

Para implementar esta lógica, se ha desarrollado un nodo específico encargado de monitorizar la actividad de las balizas. Este nodo recibe periódicamente información sobre qué balizas están siendo detectadas. A partir de esta información, calcula cuántas balizas están activas y determina una repartición equitativa de las partículas disponibles entre las balizas activas.

Una baliza se considera activa si el sistema recibe al menos una medición válida de distancia dentro de un periodo de tiempo predefinido.

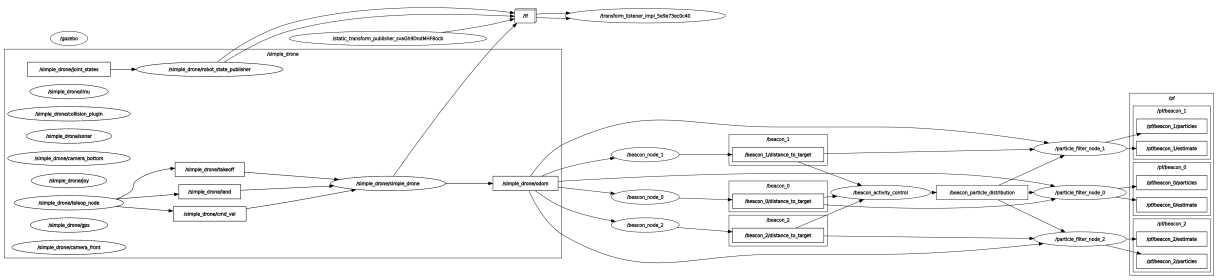


Figura 4: Nodos usados en el filtro inteligente

Por ejemplo, si el sistema dispone de 3000 partículas totales y en un momento determinado se detectan 3 balizas, el nodo asigna 1000 partículas a cada una. Si en el siguiente instante solo hay 2 balizas activas, el sistema redistribuye automáticamente las partículas, asignando 1500 a cada una y desactivando el filtro de la baliza ausente.

Cada nodo de filtro de partículas recibe esta información mediante un mensaje que indica el número exacto de partículas que debe manejar. Cuando una baliza deja de estar activa, el nodo correspondiente pone a cero sus partículas, deteniendo el procesamiento hasta que se reciba una nueva asignación. Por el contrario, si una baliza inactiva vuelve a estar presente, su filtro se reinicia con el nuevo número de partículas alrededor de la última posición estimada y se incorpora nuevamente al sistema.

Este mecanismo permite simular de forma efectiva un entorno con recursos limitados, y ha demostrado mejorar tanto la precisión como la rapidez del sistema. Al concentrar las partículas en las balizas que realmente están siendo detectadas, el filtro puede converger más rápidamente a una estimación fiable de la posición, evitando el desperdicio de partículas en filtros que no aportan información en ese momento.

En resumen, esta estrategia permite utilizar varios filtros de partículas de forma eficiente con restricciones de cómputo, adaptando dinámicamente la asignación de recursos según la percepción actual del dron.

3.3.3. Filtro de partículas + EKF

Como mejora final en el diseño del sistema de estimación, se ha utilizado una estructura híbrida que combina el filtro de partículas con el filtro de Kalman extendido (EKF). Para ello, se ha desarrollado una versión del filtro de partículas básico, que incorpora un método para evaluar la gausianidad de su distribución de partículas.

Una vez que se determina que las partículas están suficientemente concentradas y organizadas en torno a un único punto (el sistema ha convergido), el filtro estima la media y la matriz de covarianza de la nube de partículas y, con estos dos parámetros,

construiremos una estimación gaussiana del estado, pudiendo así inicializar el filtro de Kalman extendido.

Para la implementación del EKF se ha creado un nuevo nodo, `ekf_filter`, que recibe la inicialización desde el nodo del filtro de partículas. Este nodo ejecuta un EKF convencional, permitiendo reducir el cómputo una vez la estimación del filtro de partículas es lo suficientemente buena. Al igual que en las implementaciones anteriores, para cada baliza se crea una instancia del `ekf_filter` correspondiente, de forma que cada EKF trabaja de forma aislada con los datos de su propia baliza.

Este enfoque presenta importantes ventajas. Por un lado, tenemos el filtro de partículas, que nos proporciona robustez frente a distribuciones que inicialmente eran multimodales, tenían un gran ruido o no teníamos una buena estimación inicial. Por otro lado, una vez alcanzada una fase de estimación razonable, la transición al EKF permite reducir drásticamente la carga computacional, al reemplazar la simulación de miles de partículas por un modelo paramétrico mucho más eficiente.

Si nos referimos a la estructura y comunicación entre los nodos, de partida se lanza un filtro de partículas y un EKF asociado a cada baliza. El EKF inicialmente permanecerá inactivo, a la espera de recibir una estimación inicial. Cuando el filtro de partículas ha superado el umbral de gausianidad establecido, se publica la estimación inicial y se activa una flag, que evita que se sigan actualizando las partículas del filtro. Cuando el EKF haya recibido la estimación inicial, comenzará su estimación. De esta forma conseguimos que nunca estén los dos filtros funcionando al mismo tiempo, reduciendo el gasto computacional lo máximo posible.

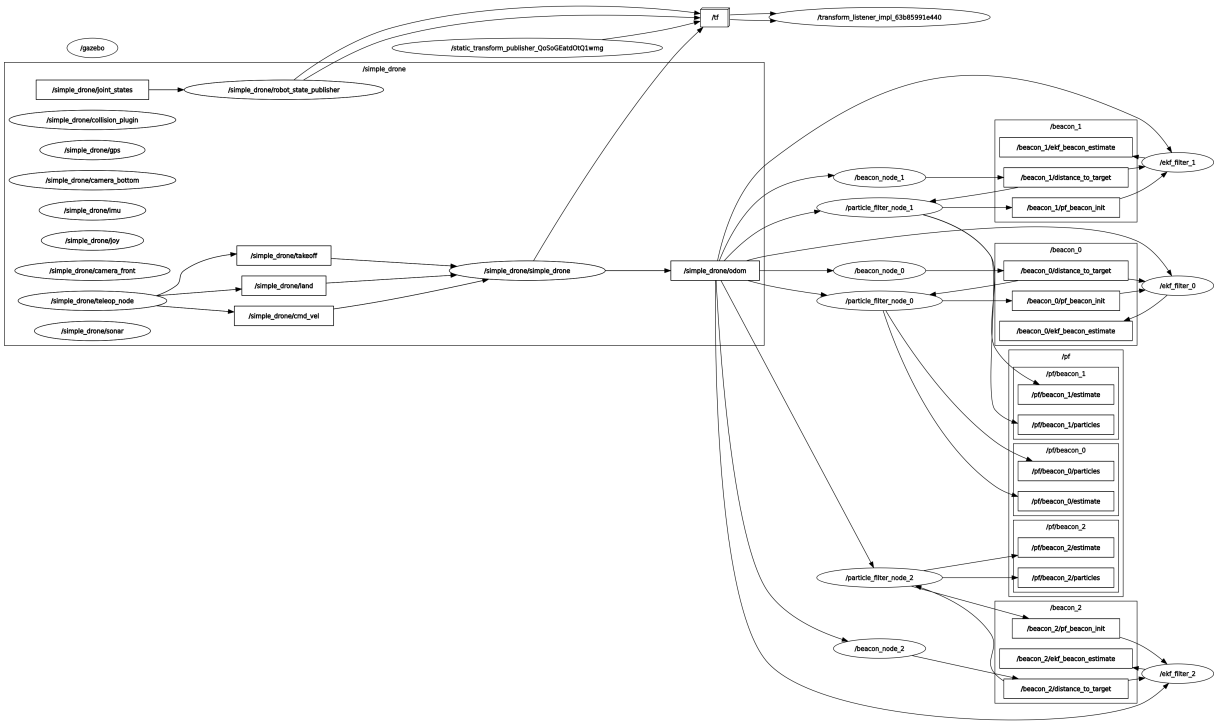


Figura 5: Nodos usados en el filtro de partículas + EKF

4. Fase de pruebas y discusión de resultados

Al realizar 3 filtros diferentes durante la realización del trabajo, podemos realizar numerosos experimentos. Para poder diferenciar entre los 3, se denomina el filtro de partículas básico como «1», el «inteligente» como «2» y finalmente el filtro de partículas junto al EKF es el número 3.

4.1. Tabla de resultados

Filtro	Nº Partículas	sigma	ruido std	errores en x	errores en y	errores en z	Gráfica
				0 | 1 | 2	0 | 1 | 2	0 | 1 | 2	
1	5000	0.3	0.3	0.13 | 0.02 | 0.18	0.09 | 0.12 | 0.09	0.2 | 0.2 | 0.16	6
1	2000	0.3	0.3	0.28 | 0.18 | 0.01	0.24 | 0.07 | 0.03	0.08 | 0.21 | 0.15	7
1	2000	0.6	0.3	0.1 | 0.43 | 0.65	1.76 | 1.22 | 1.43	1.55 | 0.36 | 1	8
1	2000	0.15	0.3	0.1 | 0.05 | 0.1	0.12 | 0.1 | 0.08	0.25 | 0.2 | 0.15	9
1	2000	0.3	0.15	0.16 | 0.03 | 0.08	0.21 | 0.32 | 0.25	0.04 | 0.06 | 0.08	10
1	2000	0.3	0.6	0.75 | 0.55 | 0.40	0.50 | 0.65 | 1.10	0.6 | 0.50 | 0.22	11
2	15000	0.3	0.3	0.05 | 0.44 | 0.22	0.03 | 0.2 | 0.05	0.15 | 0.26 | 0.01	12
2	6000	0.3	0.3	0.20 | 0.14 | 0.02	0.02 | 0.20 | 0.08	0.025 | 0.4 | 0.33	13
2	15000	0.6	0.3	0.10 | 0.25 | 0.15	0.9 | 0.05 | 0.95	0.70 | 0.65 | 0.55	-
2	15000	0.15	0.3	0.10 | 0.05 | 0.15	0.20 | 0.15 | 0.20	0.25 | 0.15 | 0.35	-
3	5000	0.3	0.3	0.02 | 0.02 | 0.01	0.03 | 0.012 | 0.01	0.025 | 0.03 | 0.01	14

Tabla 1: Parámetros

4.2. Filtro de partículas

En este experimento se puede observar como al disminuir el número de partículas, el filtro tarda más en converger y lo hace con un mayor error.

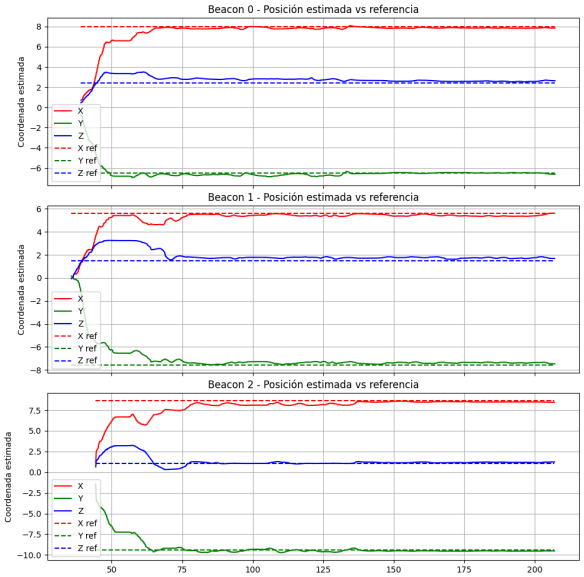


Figura 6: 5000 partículas por filtro

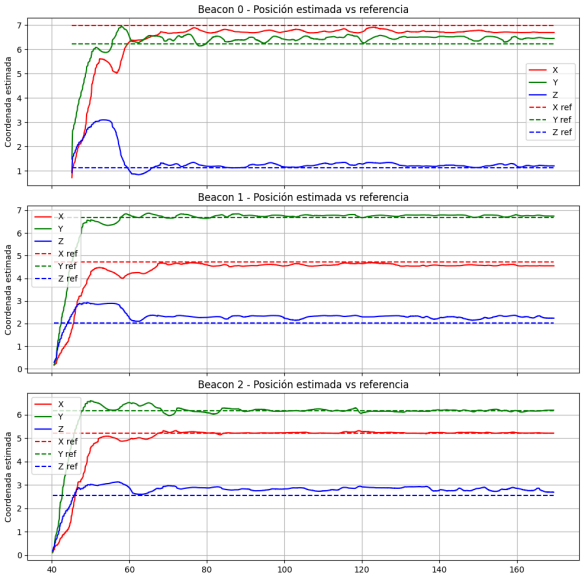


Figura 7: 2000 partículas por filtro

En este otro experimento se observa que al variar el parámetro sigma (dispersión de las partículas en la inicialización del filtro) el tiempo de convergencia varía. Cuando sigma es muy alta, se genera mucho error que el filtro tarda en corregir, y cuando es muy baja se le fuerza al filtro a converger en un sitio erróneo y, al tener que corregir esa posición errónea, también tarda más. Por lo tanto lo ideal es elegir una sigma intermedia en función del entorno donde se encuentre el robot.

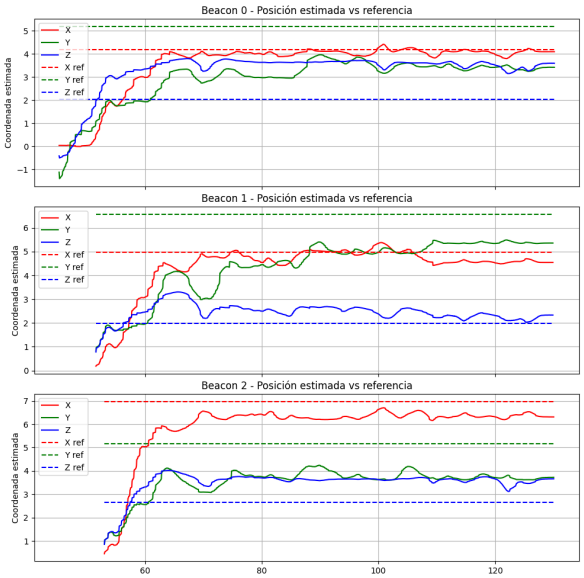


Figura 8: Sigma alta

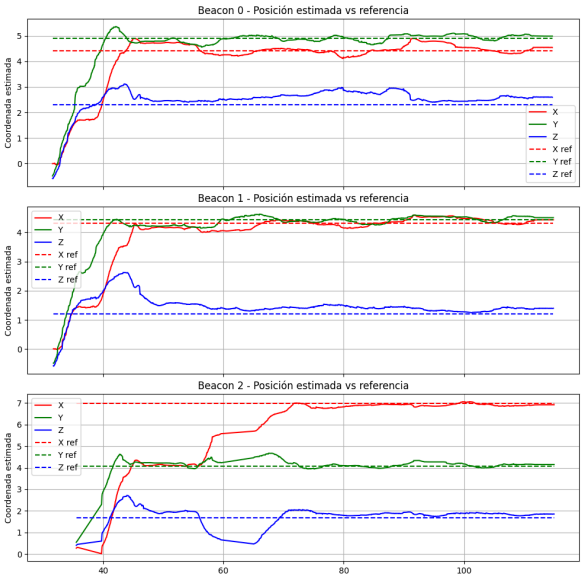


Figura 9: Sigma baja

Al aumentar el ruido de la distancia que nos envía el nodo del beacon, aumenta el tiempo de convergencia y baja la precisión de la estimación ya que las medidas recibidas se alejan de la realidad. Aún así, el filtro es capaz de obviar el ruido y conseguir una estimación aceptable.

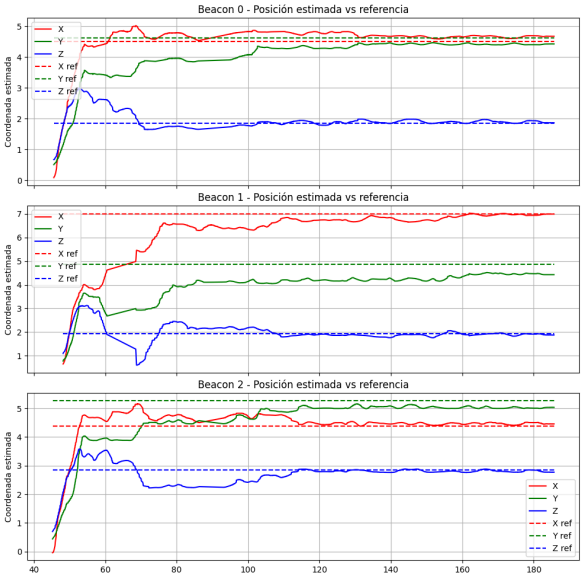


Figura 10: Ruido bajo

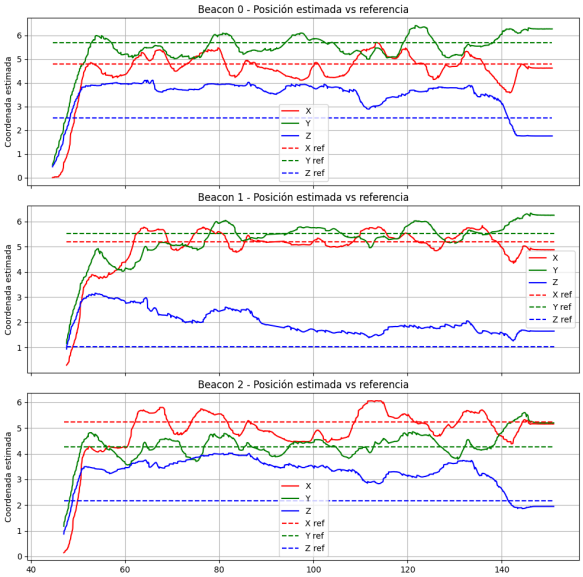


Figura 11: Ruido alto

4.3. Filtro de partículas inteligente

En este experimento estamos usando las mismas partículas totales que en el experimento anterior, pero estas se distribuyen equitativamente entre los filtros activos, como se puede observar en la 4ª gráfica de las figuras. De esta forma el filtro es capaz de converger con una mayor velocidad y una mejor precisión, sobre todo para el caso en el que haya un menor número de partículas totales.

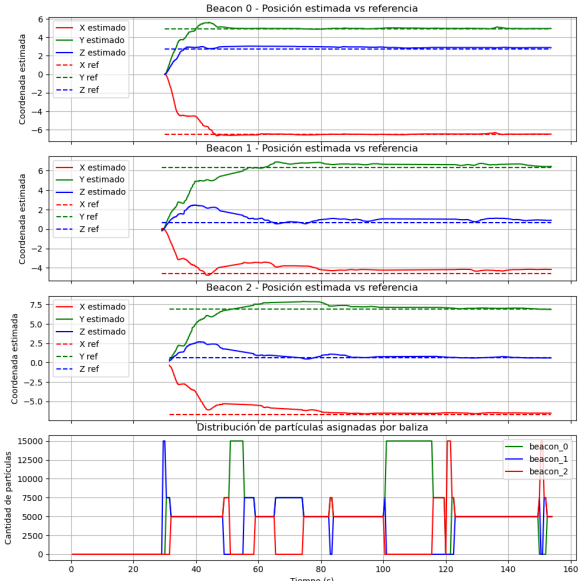


Figura 12: 15000 partículas totales

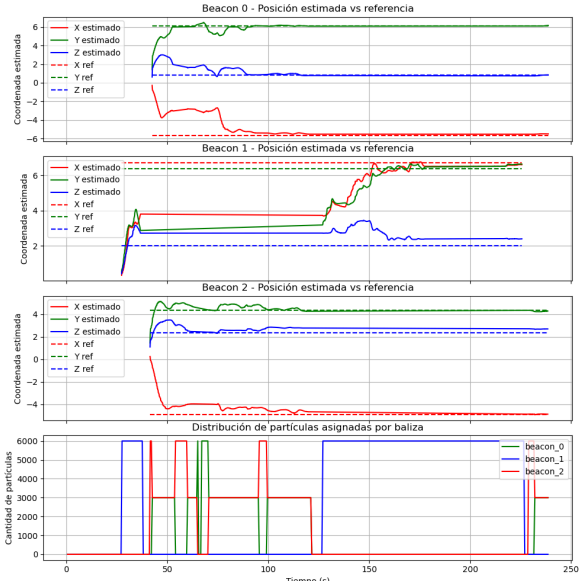


Figura 13: 6000 partículas totales

4.4. Filtro de partículas + Filtro EKF

En esta gráfica se puede apreciar que cuando la estimación del filtro de partículas se vuelve gaussiana y lo suficientemente precisa, se activa el EKF, reduciendo muchísimo el coste computacional y mejorando enormemente la precisión.

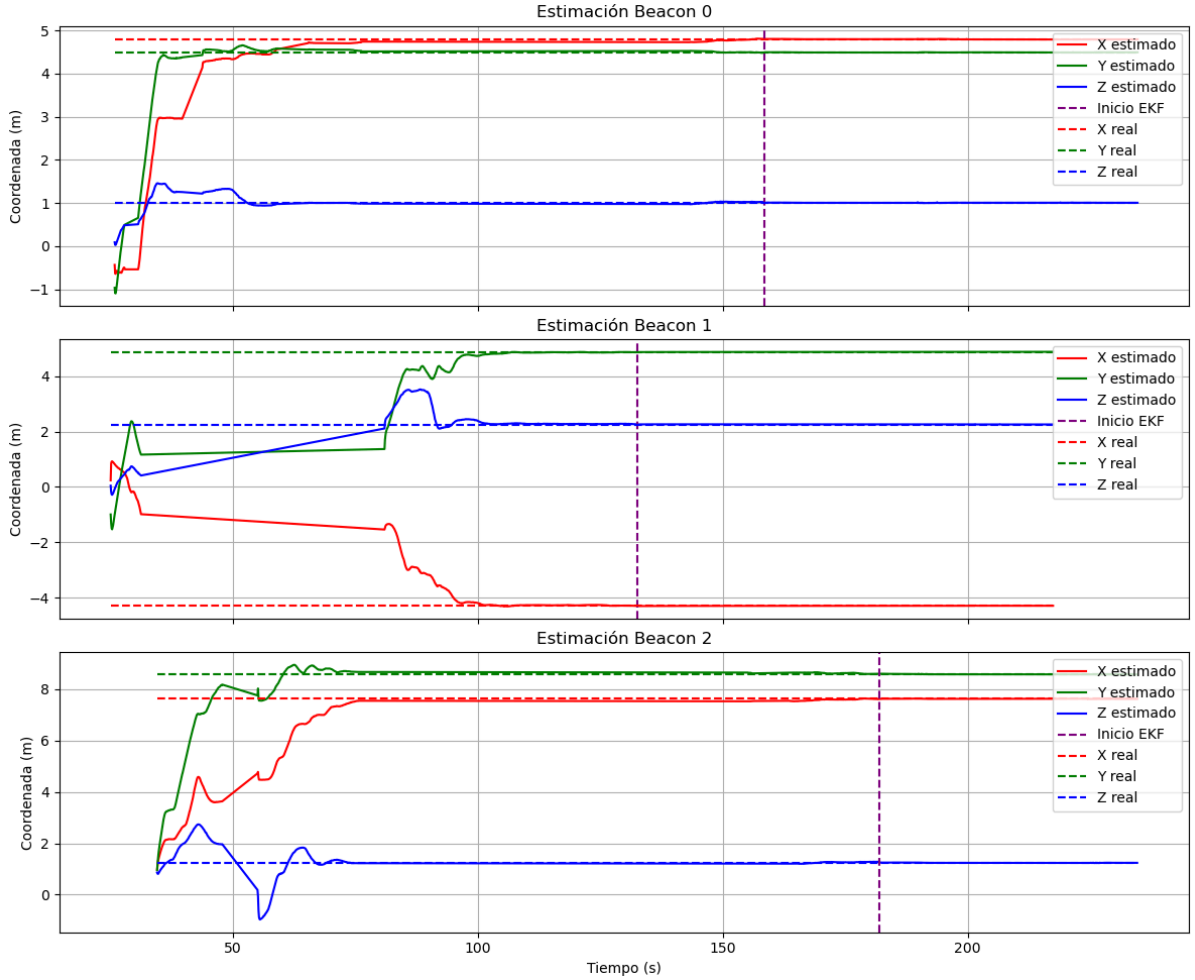


Figura 14: Gráfica de posición con ambos filtros en funcionamiento

5. Conclusión

En este trabajo se ha desarrollado un sistema de mapeo tridimensional de balizas de radiofrecuencia en un entorno simulado, empleando técnicas basadas en filtros de partículas dentro del marco de ROS 2. Se han implementado y comparado tres variantes del sistema: un filtro de partículas básico, una versión con redistribución dinámica de partículas y un enfoque híbrido que combina filtro de partículas con filtro de Kalman extendido (EKF).

Los resultados experimentales han demostrado que la variante con redistribución dinámica mejora significativamente la eficiencia del sistema al asignar los recursos computacionales de forma adaptativa, concentrando partículas en las balizas activas. Esta optimización reduce tanto el error medio como el tiempo de convergencia en comparación con el enfoque estándar.

Por su parte, la variante híbrida mostró el mejor rendimiento global, combinando la robustez de los filtros de partículas con la precisión y suavidad del EKF. Esta integración permitió obtener estimaciones más estables y precisas, especialmente tras la detección inicial de las balizas.

En conjunto, los resultados validan la utilidad de técnicas probabilísticas avanzadas para la localización pasiva de emisores RF en entornos 3D, abriendo la puerta a futuras aplicaciones en búsqueda y rescate, navegación autónoma y monitoreo ambiental.