UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA SUPERIOR DE TELECOMUNICACIÓN
REDES Y SISTEMAS DE TELECOMUNICACIÓN

PROYECTO FIN DE CARRERA

COMPARACIÓN DE MÉTODOS DE LOCALIZACIÓN DE MÚLTIPLES OBJETIVOS EMPLEANDO REDES DE SENSORES

AUTOR: EL BACHIR EL ACHHAB
TUTOR: JOAQUÍN MÍGUEZ ARENAS
JUNIO DE 2011

Proyecto Fin de Carrera
COMPARACIÓN DE MÉTODOS DE LOCALIZACIÓN DE MÚLTIPLES OBJETIVOS EMPLEANDO REDES DE SENSORES.

Autor
EL BACHIR EL ACHHAB
Tutor
JOAQUÍN MÍGUEZ ARENAS

La defensa del presente Proyecto Fin de Carrera se realizó el día 16 de Junio de 2011; siendo calificada por el siguiente tribunal:

PRESIDENTE: *Ángel Bravo Santos*

SECRETARIO: *David Luengo García*

VOCAL: *Elisa Molanes López*

Habiendo obtenido la siguiente calificación:

Calificación:

Presidente **Secretario** **Vocal**

Leganés, a 16 de Junio de 2011

A mis padres
A mis hermanos

Agradecimientos

Quiero mostrar mi agradecimiento a mis padres y mis hermanos, por todo el apoyo que me han dado durante todos estos años de la carrera, a todos mis amigos y mis compañeros de clase y de prácticas, a Joaquín por ayudarme a hacer este proyecto realidad.

Muchas gracias a todos.

Resumen

En el presente Proyecto Fin de Carrera se compara el rendimiento y la complejidad de varios algoritmos de procesado estadístico de señales aplicados al problema de localización de objetivos estáticos empleando redes de sensores. Se consideran dos escenarios, el primero con un único objetivo y el segundo con múltiples blancos. Los objetivos están úbicados en un área monitorizada por una red de sensores y emiten una señal de radiofrecuencia cuya potencia puede ser medida por los sensores. Estos últimos realizan las mediciones de manera periódica y cada medida es enviada al centro de fusión de datos que es el encargado de aplicar un algoritmo para estimar la posición de los objetivos.

A lo largo del proyecto se han estudiado varios métodos de localización. Se han estudiado métodos de Monte Carlo (MMC) que permiten aproximar funciones de distribución de probabilidad y/o estimar sus momentos. Se ha hecho hincapié en el muestreo enfatizado secuencial con remuestreo. Se han detallado los algoritmos para el caso de un único objetivo y se han mostrado resultados numéricos en ese escenario. Finalmente, se han adaptado los diferentes algoritmos para poder estimar la posición de múltiples blancos y se han presentado también resultados numéricos. Para los dos casos, se ha evaluado el rendimiento y la complejidad de todos los algoritmos estudiados en terminos del error medio absoluto de las estimaciones y el tiempo real de ejecución.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En el presente proyecto se aborda el problema de la localización de objetivos, que recientemente está recibiendo mucha atención tanto en el ámbito civil como el militar. Para ello se estudia el rendimiento y la complejidad de algunos algoritmos clásicos que se emplean para este fin. En concreto se abordan varios métodos que permiten localizar objetivos en un área monitorizada por una red de sensores. En la segunda parte, se estudia la aplicación de estos mismos métodos para la localización de múltiples objetivos.

1.1. Localización de objetivos

Las aplicaciones basadas en la localización están muy demandadas en la actualidad, tanto en el ámbito civil como en el militar, para actividades como el transporte, la vigilancia y la seguridad.

Actualmente, el sistema GPS (*Global Positioning System*) ofrece un buen servicio de localización y navegación en grandes áreas del globo terrestre. Sin embargo, existen zonas o entornos donde las prestaciones del GPS se degradan de forma notable hasta el punto de que se puede perder completamente el servicio de posicionamiento. El problema general es que el receptor de tierra necesita una visión directa de un cierto número de satélites. Esto se convierte en una limitación

en situaciones donde la propagación multitrayecto de la señal es muy acusada, típicamente en interiores de edificios o en entornos urbanos donde existen edificios de gran altura (cañones urbanos). En estos casos, los datos que llegan al receptor dejan de ser útiles para el posicionamiento.

Existen otros sistemas que se emplean en la localización y en la navegación, es decir, aparte de determinar la posición son capaces de determinar la velocidad del objetivo. Estos sistemas se emplean en las aeronaves y están basados en sistemas inerciales, acelerómetros y giróscopos. Desafortunadamente estos sistemas tienen un problema importante y es que sus prestaciones se degradan con el tiempo.

También se destacan otros sistemas como por ejemplo los sensores activos, e.g., el radar o el sónar, que se emplean actualmente para localizar objetivos dentro de una determinada área, pero son sistemas cuyo costes elevados y sólo se emplean en entornos muy específicos.

1.2. Redes de sensores

Un sensor es un dispositivo con capacidad de "sensado", procesamiento y comunicación. Una red de sensores es una colección de nodos sensores conectados entre sí, capaces de adquirir una cierta información acerca de su entorno, procesar dicha información y transmitirla a un centro de fusión de datos. Las redes de sensores inalámbricos se caracterizan por la transmisión de la información mediante radiofrecuencia, por el tamaño reducido de los nodos, el bajo coste económico y una capacidad de procesado limitada.

Las características citadas anteriormente son las que sitúan las redes de sensores como una alternativa a los sistemas de localización mencionados en la sección anterior. Las redes de sensores se pueden emplear en diversas aplicaciones destacando la monitorización de distintos fenómenos meteorológicos como las erupciones volcánicas o los terremotos, la vigilancia y la seguridad. La mayoría de estas aplicaciones requieren un conocimiento previo de las posiciones de los nodos de la propia red. En particular, en las aplicaciones de localización y seguimiento. Se suele asumir por lo tanto que los sensores se encuentran en posición conocida a

priori.

Las redes de sensores son consideradas como una tecnología muy prometedora en la actualidad, por eso se estudia su aplicación en diversos entornos. Se está investigando en las diversas bases de esta tecnología como puede ser las comunicaciones, la capacidad de procesado de los sensores, la tecnología asociada a los elementos sensores, la alimentación, así como en los algoritmos de fusión de datos.

1.3. Algoritmos de localización

Existen varias clases de algoritmos de estimación y optimización que pueden ser aplicados al problema de identificar la posición de un objeto empleando redes de sensores. En este trabajo se estudian los siguientes algoritmos:

- El filtro de partículas que se emplea básicamente para estimar el estado de un sistema que cambia a lo largo del tiempo. Es un método de Monte Carlo secuencial usado ampliamente en comunicaciones [6], seguimiento con sensores activos, visión artificial para el seguimiento de objetos etc...

- El filtro de Kalman extendido es una modificación del filtro de Kalman. Se utiliza fundamentalmente para sistemas dinámicos no lineales. Está basado en linealizar la función de dinámica/observación para poder aplicar la metodología de Kalman.

- El algoritmo de gradiente es un método básico de optimización. Se utiliza para encontrar un extremo local de una función siguiendo la dirección indicada por las derivadas parciales.

- La búsqueda aleatoria acelerada (ARS, "*Accelerated Random Search*") es un algoritmo iterativo de optimización de una función basado en la búsqueda aleatoria en conjunto de medida variable alrededor de la última aproximación disponible.

- El algoritmo de Gibbs es un método para el muestreo de distribuciones de probabilidad mediante la construcción de una cadena de Markov que tiene la distribución deseada como su distribución de equilibrio.

1.4. Objetivos del proyecto

El objetivo del proyecto es realizar una comparación, en términos de rendimiento y complejidad, de los algoritmos mencionados en la Sección 1.3 aplicados a problemas de localización de objetivos estáticos usando redes de sensores. Se abordará esta comparación en problemas de posicionamiento de un único objetivo y entornos con múltiples blancos a localizar. En ambos casos, supondremos que los sensores de la red están ubicados en posiciones conocidas y que propocionan observaciones de potencia de una señal de radio emitida por los blancos de interés.

1.5. Organización de la memoria

Se ha estructurado la memoria del proyecto de la siguiente manera:

En el Capítulo 2 se presenta el modelo de señal que se va a emplear en el proyecto. Se hace hincapié en la ecuación de observaciones tanto para el caso de un único objetivo como para el caso de múltiples objetivos.

El Capítulo 3 está dedicado a una introducción a los métodos de Monte Carlo, ya que, algunos de los algoritmos que se estudian en este proyecto se basan sobre estos métodos, en concreto, el filtro de partículas, el ARS y el muestreo de Gibbs.

El Capítulo 4 está dividido en dos partes. En la primera se detallan todos los algoritmos que se estudian en el caso de posicionamiento de un único objetivo. Se considera que tenemos un único objetivo estático en un área donde se ha desplegado una red de sensores. En la segunda parte se presentan los resultados obtenidos para cada algoritmo mediante la realización de simulaciones. Para cada algoritmo se ha estudiado su rendimiento y su complejidad.

El Capítulo 5, a su vez, está dividido en dos partes. En la primera se adaptan todos los algoritmos para poder emplearlos en el caso de múltiples objetivos. En este caso se considera que tenemos un número conocido de objetivos estáticos en un área monitorizada por la red de sensores. En la segunda parte se presentan tanto los resultados obtenidos de la simulación como el rendimiento y la complejidad de cada uno de los algoritmos estudiados.

La memoria termina con el Capítulo 6, en el cual se presentan las conclusiones obtenidas en todas las fases del mismo.

Capítulo 2

Modelo de la señal

En este capítulo se presentan los diferentes modelos matemáticos de posición y de observaciones que se emplean en los problemas de localización en este proyecto. En primer lugar se considera un sistema con un único objetivo, en el cual se analiza tanto el modelo de posición como el modelo de las observaciones. Posteriormente, en la segunda parte se considera un sistema similar al anterior pero con múltiples objetivos.

2.1. Un único objetivo

Para el modelo de un único objetivo se considera que tenemos un blanco, en una posición desconocida que emite una señal de radiofrecuencia en una zona en la cual se han situado un conjunto de sensores de manera aleatoria en unas posiciones fijas y conocidas. Dichos sensores son capaces de recibir la señal, realizar mediciones de la potencia recibida y enviar dicha información a un centro de fusión de datos que es el encargado de estimar la posición a partir de todas las mediciones de todos los sensores.

Se considera un posicionamiento en dos dimensiones y un espacio acotado con lo cual se denota la posición del blanco como, $\mathbf{x} = [x_1, x_2]^T$ y la posición de los sensores $\mathbf{b}_j = [b_{j,1}, b_{j,2}]^T$, con $j = 1, \ldots, K$, donde K es el número de sensores. A

continuación se presenta el modelo de observaciones para este caso.

2.1.1. Modelo de las observaciones

Para las mediciones se considera un modelo simple. Se asume que los sensores sólo pueden proporcionar información acerca de la potencia recibida, con lo cual todo el peso del algoritmo está en el centro de fusión de datos. Este último tiene como misión recibir la potencia que mide cada sensor y aplicar un algoritmo para estimar la posición.

Cada sensor mide una potencia denotada como y_j, donde el subíndice j indica que la medición proviene del sensor j-ésimo. Para estimar la posición es importante saber la posición de cada sensor, $\mathbf{b}_j = [b_{j,1}, b_{j,2}]^T$. El modelo de las observaciones recibidas en el instante t que se considera para un objetivo situado en la posición $\mathbf{x}$ es

$$y_{j,t} = 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^{\gamma}}\right) + v_{j,t}, \quad v_{j,t} \sim N(0, \sigma_{v_t}^2), \tag{2.1}$$

donde P_0 es la potencia de transmisión, γ es un factor que depende de las pérdidas en el medio y $v_{j,t}$ es el ruido, se asume que es blanco con media cero y varianza $\sigma_{v_t}^2$. Como se ha podido ver en la ecuación anterior, en cada sensor se tiene una observación que es la potencia recibida, es decir tras que el emisor transmite una potencia inicial P_0, el receptor recibe una potencia teniendo en cuenta la distancia $\|\mathbf{x}-\mathbf{b}_j\|$ entre el emisor y el transmisor, un factor γ que depende de las pérdidas en el medio y un ruido $v_{j,t}$.

En el centro de fusión de datos se recopilan todas las observaciones de todos los sensores. Esta información constituye el vector de observaciones recibidas en un instante t, $\mathbf{y}_t = [y_{1,t}, \ldots, y_{K,t}]^T$, con K es el número de sensores. En un periodo de tiempo τ, se obtiene un vector de observaciones $\mathbf{y} = \left[\mathbf{y}_1^T, \ldots, \mathbf{y}_\tau^T\right]^T$, que contiene la informacion de todos los sensores en dicho periodo.

2.2. Múltiples objetivos

Para el caso de múltiples objetivos se considera que tenemos N_b blancos $\mathbf{x}_1, \ldots, \mathbf{x}_{N_b}$. Cada blanco está ubicado en una posición desconocida $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$. Igual que en el apartado anterior, para determinar la posición de cada objetivo se considera que tenemos un conjunto de sensores situados en posiciones conocidas a *priori*. La función de los sensores es recibir la señal de radiofrecuencia emitida por los objetivos, medir la potencia de la señal y enviar dicha potencia al centro de fusión. Este último es el encargado de procesar la información recibida y aplicar un algoritmo para estimar la posición de cada objetivo. En la siguiente sección se presenta el modelo de observación para el caso de múltiples objetivos.

2.2.1. Modelo de las observaciones

Como en el caso de un único objetivo, se asume que los sensores solo pueden proporcionar información acerca de la potencia recibida en el instante t. Cada sensor mide una potencia $y_{j,t}$, donde el subíndice j indica que la potencia proviene del sensor j-ésimo. Para estimar la posición de cada objetivo es importante saber la posición de cada sensor $\mathbf{b}_j = [b_{j,1}, b_{j,2}]^T$. El modelo de observaciones que se considera para el caso de múltiples objetivos, cada uno situado en la posición $\mathbf{x}_i, \quad i = 1, \ldots, N_b$, es

$$y_{j,t} = 10 \log_{10} \left(\sum_{i=1}^{N_b} \frac{P_0}{\|\mathbf{x}_i - \mathbf{b}_j\|^\gamma} \right) + v_{j,t}, \quad v_{j,t} \sim N(0, \sigma_{v_t}^2), j = 1, \ldots, K, \quad (2.2)$$

donde P_0 es la potencia de transmisión, γ es un factor que depende de las pérdidas en el medio y $v_{j,t}$ es el ruido, se asume blanco con media cero y varianza $\sigma_{v_t}^2$. La ecuación es similar al caso de un único objetivo con la salvedad de que en este caso, en cada sensor se recopila la información de todos los blancos, por esta razón aparece el termino de suma $\sum_{j=1}^{N_b} \frac{P_0}{\|\mathbf{x}_i - \mathbf{b}_j\|^\gamma}$.
Igual que en el caso anterior en el centro de fusión de datos se recopilan todas las observaciones en un instante t, $\mathbf{y}_t = [y_{1,t}, \ldots, y_{K,t}]^T$, con K es el número de sensores y en un periodo τ se obtiene un vector con la información de todos los sensores $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_\tau]^T$.

2.3. Planteamiento del problema

Las dos Ecuaciones (2.1) y (2.2) representan observaciones que dependen de la potencia de transmisión, de la distancia entre el objetivo y cada uno de los sensores y de un ruido que se supone blanco con media cero y una varianza conocida. El vector que contiene la posición del objetivo, $\mathbf{x}$, se modela como un vector aleatorio continuo y se expresa su densidad de probabilidad a *priori* como $p(\mathbf{x})$.

El problema de la localización se reduce en la estimación de la posición $\mathbf{x}$ del objetivo dada las observaciones. Para ello, se plantea resolver el problema de optimización

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x} \in \mathbb{R}^2} p(\mathbf{x}|\mathbf{y}) \tag{2.3}$$

$$= \arg \max_{\mathbf{x} \in \mathbb{R}^2} p(\mathbf{y}|\mathbf{x}) p(\mathbf{x}), \tag{2.4}$$

donde $p(\mathbf{x}|\mathbf{y})$ la desidad de probabilidad a *posteriori* condicionada a las observaciones y $p(\mathbf{y}|\mathbf{x})$ es la verosimilitud.

2.4. Conclusiones

En este Capítulo se ha presentado el modelo de la señal que se considera en este proyecto. En primer lugar, se ha detallado el caso de un único objetivo. Donde se considera que en una area monitorizada por un número determinado de sensores solo existe un único objetivo que se pretende estimar su posición. En este modelo, se analizó tanto el modelo de la señal como el modelo de las observaciones. Las observaciones son las medidas de la potencia electromagnética. Estas medidas son realizadas por los sensores del sistema. La potencia recibida por un sensor procedente de un blanco está inversamente proporcional a la distancia euclídea entre el sensor y dicho blanco. En segundo lugar, se ha detallado el caso de múltiples objetivos, es decir, este caso es similar al caso anterior con la salvedad de que ahora en el area existe más de un objetivo que se pretende

estimar sus posiciones. Igual que en el caso anterior se analizó tanto el modelo de la señal como el modelo de las observaciones.

Capítulo 3

Métodos de Monte Carlo

En este capítulo se presenta los métodos de Monte Carlo (MMC) en general, haciendo hincapié en el mustreo enfatizado secuencial y el remuestreo. En primer lugar, se presenta el muestreo exacto, a continucación se detalla el muestreo enfatizado y el muestreo enfatizado con remuestreo, finalmente se presenta el muestreo de Gibbs.

3.1. Métodos de Monte Carlo

Los métodos de Monte Carlo (MMC) son un conjunto de procedimientos basados en simulación que permiten aproximar funciones de distribución de probabilidad y/o estimar sus momentos. Estos presentan varias ventajas. Se puede citar la flexibilidad y las buenas propiedades de convergencia. Otra ventaja que tiene estos métodos es que no tienen una restricción en cuanto a que modelos matemáticos empleados sean lineales o gaussianos.

3.1.1. Muestreo exacto

Nuestro objetivo es estimar momentos de la f.d.p. $p(\mathbf{x}|\mathbf{y})$. Para ello se supone que tenemos un número de muestras aleatorias suficientemente grande de esta

distribución y que estas muestras son independientes e identicamente distribuidas (i.i.d). Nos referimos a menudo a estas muestras como partículas y utilizamos la notación $\Omega = \left\{\mathbf{x}^{(i)}\right\}_{i=1}^{M}$ para referirnos al conjunto de muestras disponibles, donde M es el número de partículas. Una aproximación discreta de la distribución de probabilidad asociada a $p(\mathbf{x}|\mathbf{y})$ es

$$p_M(\mathbf{x}|\mathbf{y}) = \frac{1}{M}\sum_{i=1}^{M}\delta_{\mathbf{x}^{(i)}}(\mathbf{x}), \tag{3.1}$$

donde $\delta_{\mathbf{x}}^{(i)}(\mathbf{x}) = \delta\left(\mathbf{x}^{(i)} - \mathbf{x}\right)$ es la función delta de Dirac.
La esperanza de $f(\mathbf{x})$ con respecto a la distribución a *posteriori* es

$$I(f) = \mathbf{E}_{p(\mathbf{x}|\mathbf{y})}[f(\mathbf{x})] = \int f(\mathbf{x})p(\mathbf{x}|\mathbf{y})d\mathbf{x}. \tag{3.2}$$

A partir de (3.1) una estimación de (3.2) es

$$I_M(f) = \int f(\mathbf{x})p_M(\mathbf{x}|\mathbf{y})d\mathbf{x} = \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{x}^{(i)}), \tag{3.3}$$

bajo ciertas condiciones $I_M(f)$ converge a $I(f)$ [3]

Una desventaja de este método es que, generalmente, no es posible muestrear eficientemente de la distribución a *posteriori*, $p(\mathbf{x}|\mathbf{y})$, en cualquier instante de tiempo. Además, el muestreo exacto no es una opción factible en sistemas dinámicos, ya que, requiere poseer todas las muestras antes de realizar la estimación de la densidad. Afortunadamente existen otras alternativas, como el muestreo enfatizado que se detalla en la siguiente sección. Se puede aplicar el muestreo enfatizado en algunos escenarios donde no es posible utilizar el muestreo exacto.

3.1.2. Muestreo enfatizado

Se aplica el muestreo enfatizado [3] en los casos en que no se pueda muestrar directamente la distribución de interés, en nuestro caso $p(\mathbf{x}|\mathbf{y})$. Se considera una función tentativa, $q(\mathbf{x}|\mathbf{y})$, tal que su dominio contenga al de $p(\mathbf{x}|\mathbf{y})$ ($q(\mathbf{x}|\mathbf{y}) > 0$ siempre que $p(\mathbf{x}|\mathbf{y}) > 0$). Esta función se elige de tal modo que sea lo más parecida posible a $p(\mathbf{x}|\mathbf{y})$, siendo a la vez posible tomar muestras de la misma con facilidad. Se genera un conjunto de partículas $\left\{\mathbf{x}^{(i)}\right\}_{i=1}^{M}$ y a cada partícula se le asigna un peso

$$\tilde{\omega}^{(i)} = \frac{p(\mathbf{x}^{(i)}|\mathbf{y})}{q(\mathbf{x}^{(i)}|\mathbf{y})}. \tag{3.4}$$

Estos pesos se pueden interpretar como una estimación de la densidad de probabilidad en torno a $\mathbf{x}^{(i)}$ y son apropiados para aproximar los momentos de $p(\mathbf{x}|\mathbf{y})$, ya que

$$\begin{aligned}
\mathbf{E}_{q(\mathbf{x})}[\mathbf{x}^{(i)}\tilde{\omega}^{(i)}] &= \int \mathbf{x}^{(i)}\tilde{\omega}^{(i)} q(\mathbf{x}^{(i)}) d\mathbf{x}^{(i)} \\
&= \int \mathbf{x}^{(i)} \frac{p(\mathbf{x}^{(i)}|\mathbf{y})}{q(\mathbf{x}^{(i)})} q(\mathbf{x}^{(i)}) d\mathbf{x}^{(i)} \\
&= \int \mathbf{x}^{(i)} p(\mathbf{x}^{(i)}|\mathbf{y}) d\mathbf{x}^{(i)} \\
&= \mathbf{E}_{p(\mathbf{x}|\mathbf{y})}[\mathbf{x}^{(i)}].
\end{aligned} \tag{3.5}$$

Ahora ya se puede aproximar la distribución con f.d.p. $p(\mathbf{x}|\mathbf{y})$ como

$$p_M(\mathbf{x}) = \sum_{i=1}^{M} \omega^{(i)} \delta_{\mathbf{x}^{(i)}}(\mathbf{x}), \tag{3.6}$$

donde $\omega^{(i)}$ son los pesos normalizados

$$\omega^{(i)} = \frac{\tilde{\omega}^{(i)}}{\sum_{k=1}^{M} \tilde{\omega}^{(k)}}, \quad k = 1, \ldots, M. \tag{3.7}$$

Además se verifica que

$$I_M(f) = \int f(\mathbf{x}) p_M(d\mathbf{x}|\mathbf{y}) = \sum_{i=1}^{M} f(\mathbf{x}^{(i)})\omega^{(i)}, \tag{3.8}$$

bajo ciertas condiciones poco restrictivas converge a $I(f)$ de manera asintótica [3]. Sin embargo, este método no es recursivo.

3.1.3. Muestreo enfatizado secuencial

Supongamos a continuación que la posición del objetivo va cambiando con el tiempo. En lugar de un vector fijo, $\mathbf{x}$, tenemos entonces una secuencia de posiciones $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t$ y una secuencia de observaciones asociada $\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_t$, con una fdp a *priori* $p(\mathbf{x}_0)$ para la posición inicial del objetivo.
La fdp de interés es, ahora, $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$, y se pretende emplear el muestreo enfatizado de manera secuencial, i.e., en vez de generar partículas $\mathbf{x}_{0:t}^{(i)} = \{\mathbf{x}_0^{(i)}, \ldots, \mathbf{x}_t^{(i)}\}$ de la trayectoria completa queremos que en el instante n sólo sea necesario generar muestras de $\mathbf{x}_n$ condicionadas a $\mathbf{y}_{1:n}$. Para ello escogemos una fdp tentativa que se pueda factorizar como

$$q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = q(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \mathbf{y}_{1:t}) q(\mathbf{x}_{0:t-1}|\mathbf{y}_{1:t-1}). \tag{3.9}$$

El primer factor obtenido se corresponde con la función tentativa en el instante t y el segundo factor es la función hasta el instante previo. Es decir, que las muestras actuales se obtienen agregando las muestras ya existentes con las nuevas. Recursivamente se obtiene:

$$q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = q(\mathbf{x}_0) \prod_{k=1}^{t} q(\mathbf{x}_k|\mathbf{x}_{1:k-1}, \mathbf{y}_{1:k}). \tag{3.10}$$

Se supone que $q(\mathbf{x}_0) = p(\mathbf{x}_0)$ conocida. Para un conjunto de partículas $\left\{\mathbf{x}_{0:t}^{(i)}\right\}_{i=1}^{M}$

los pesos cumplen que

$$\omega_t^{(i)} \propto \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})p(\mathbf{x}_{1:t-1}^{(i)}|\mathbf{y}_{1:t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_{1:t}^{(i)})q(\mathbf{x}_{0:t-1}^{(i)}|\mathbf{y}_{1:t-1}^{(i)})} \propto \omega_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_{1:t}^{(i)})}, \quad (3.11)$$

i.e. se puede calcular recursivamente.
Con las partículas generadas según $\mathbf{x}_t^{(i)} \sim q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{y}_t)$ y sus correspondientes pesos en (3.11) la distribución a *posteriori* se aproxima

$$p_M(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \sum_{i=1}^{M} \omega_t^{(i)} \delta_{\mathbf{x}_{0:t}^{(i)}}(\mathbf{x}_{0:t}). \quad (3.12)$$

Hay que tener en cuenta de que la elección de una buena función tentativa es fundamental para el buen funcionamiento de este método. A la hora de elegir la función tentativa, es conveniente elegir una que minimice la varianza de los pesos y debe estar condicionada a $\mathbf{x}_{0:t-1}^{(i)}$ y a $\mathbf{y}_{1:t}$. Esta varianza verifica que [3]

$$var_{q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_{1:t})}[\tilde{\omega}_t^{(i)}] = (\tilde{\omega}_{t-1}^{(i)})^2 \left[\int \frac{(p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}))^2}{q(\mathbf{x}_t|\mathbf{y}_{1:t},\mathbf{x}_{0:t-1}^{(i)})} d\mathbf{x}_t - p^2(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}) \right]. \quad (3.13)$$

La función tentativa óptima es $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)},\mathbf{y}_{1:t})_{opt} = p(\mathbf{x}_t|\mathbf{y}_t,\mathbf{x}_{t-1}^{(i)})$ [3]. Para esta función óptima, la ecuación de actualización de los pesos utilizando (3.11) se reduce a

$$\tilde{\omega}_t^{(i)} = \omega_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}). \quad (3.14)$$

Utilizar la función tentativa óptima tiene un inconveniente, es que hay que evaluar la siguiente expresión, que generalmente no tiene una solución analítica:

$$p(\mathbf{y}_t|\mathbf{x}_{0:t-1}^{(i)}) = \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})d\mathbf{x}_t. \quad (3.15)$$

Por lo tanto, suele ser necesario recurrir a funciones tentativas alternativas. Para este problema, hay una solución sencilla que consiste en escoger como función tentativa la densidad a *priori* $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. Lo que se obtiene con esto es una simplificación del cálculo de los pesos. Con esta elección las ecuaciones (3.10) y (3.11) tienen la siguiente forma:

$$q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = p(\mathbf{x}_{0:t}) = p(\mathbf{x}_0)\prod_{k=1}^{t} p(\mathbf{x}_k|\mathbf{x}_{k-1}), \tag{3.16}$$

$$\omega_t^{(i)} \propto \omega_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)} = \omega_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)}). \tag{3.17}$$

La expresión para el cálculo de los pesos en (3.17) se ha simplificada considerablemente, a cambio de despreciar la información de las observaciones $\mathbf{y}_{1:t}$. Usando este método se puede obtener una buena aproximación, pero tiene el inconveniente de que necesita un número de muestras elevado.

En cualquier caso, e independientemente de la función tentativa utilizada el muestreo enfatizado secuencial "*Sequential Importance Sampling*"(SIS) tiene el problema conocido como degeneración de los pesos [3] debido al aumento de la varianza de los pesos. Para solucionar este problema se suele usar el muestreo enfatizado secuencial con remuestreo "*Sequential Importance Resampling*"(SIR).

3.1.4. Muestreo enfatizado secuencial con remuestreo

Con el muestreo enfatizado secuencial se puede producir el problema de degeneración de pesos, que consiste en que, con el paso de tiempo sólo unas pocas partículas tienen unos pesos asociados altos, mientras el resto de las partículas tienen pesos asociados cercanos a cero. Estas últimas no aportan prácticamente nada a la estimación de la densidad de interés, por lo que se pierde un gran esfuerzo computacional por su actualización. La degeneración de los pesos puede medirse cuantitativamente mediante el parámetro [3]

$$M_{eff} = \frac{M}{1 + var(\tilde{\omega}_t^{(i)})}. \tag{3.18}$$

El parámetro M_{eff} nos da el tamaño efectivo de la muestra, es decir, el número de muestras necesario, empleando el muestreo exacto, para conseguir una aproximación de la misma calidad. Como no es posible calcular M_{eff} de forma exacta se emplea la aproximación

$$\hat{M}_{eff} = \frac{1}{\sum_{i=1}^{M} (\omega_t^{(i)})^2}, \tag{3.19}$$

donde $\omega_t^{(i)}$ son los pesos normalizados. El valor de $\hat{M}_{eff}$ siempre es menor o igual que el valor de M y, cuanto menor sea su valor, mayor es la degeneración de los pesos.

Para evitar el fenómeno de la degeneración de pesos se emplea el remuestreo, que consiste en eliminar las partículas con peso muy pequeño y replicar las partículas con mayor peso [3]. El objetivo es que el algoritmo siempre tenga partículas con pesos aceptables para realizar una buena estimación.

Hay varias técnicas de remuestreo. En este trabajo se emplea el remuestreo multinomial [5]. Este puede ser empleado en cada iteración del algoritmo o sólo cuando el parámetro $\hat{M}_{eff}$ sea tan bajo que lo haga necesario. A menudo se opta por la última opción, es decir, se realiza el remuestreo sólo cuando $\hat{M}_{eff}$ es inferior a un cierto umbral de remuestreo M_{umb}.

El remuestreo multinomial, consiste en que se parte de una distribución discreta $Prob(j = l) = \omega_t^{(l)}$, $l = 1, \ldots, M$, de la que se muestrea M veces, obteniendo un conjunto de índices $\{j^{(i)}\}_{i=1}^{M}$. Se toman las partículas asociadas a estos índices $\{\mathbf{x}_{0:t}^{j^{(i)}}\}_{i=1}^{M}$ y se le asigna a cada una un peso $1/M$. Estas partículas, junto con sus pesos, $\{\mathbf{x}_{0:t}^{j^{(i)}}, \frac{1}{M}\}_{i=1}^{M}$, proporcionan una estimación de distribución asociada a la función de densidad a *posteriori* $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. La aproximación resultante es

$$p_M(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{1}{M} \sum_{i=1}^{M} \delta_{\mathbf{x}_{0:t}^{j^{(i)}}}(\mathbf{x}_{0:t}). \tag{3.20}$$

El remuestreo disminuye la varianza de los pesos, pero la selección de partículas con pesos muy elevados hace que la muestra $\{\mathbf{x}_{0:t}^{(i)}, \omega_t^{(i)}\}_{i=1}^M$ después de cada paso del remuestreo contenga muestras repetidas muchas veces. Esto lleva a una pérdida de diversidad que se suele denominar empobrecimiento muestral.

Inicialización:

1. Muestrear $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \quad i = 1, \ldots, M.$
2. Fijar $\omega_0^{(i)} = \frac{1}{M} \quad \text{para} \;\; i = 1, \ldots, M.$

Para cada t:

1. Actualizar los pesos: $\tilde{\omega}_t^{(i)} = \tilde{\omega}_{t-1}^{(i)} p(\mathbf{y}_t | \mathbf{x}_t^{(i)})$.
2. Normalizar los pesos: $\omega_t^{(i)} = \frac{\hat{\omega}_t^{(i)}}{\sum_{k=1}^M \hat{\omega}_{t-1}^{(k)}}$.
3. Cáluclar: $\hat{\mathbf{x}}_t = \sum_{i=1}^M \omega_t^{(i)} \mathbf{x}_t^{(i)}$.
4. Calcular : $M_{eff} = \frac{1}{\sum_{i=1}^M (\omega_t^{(i)})^2}$.
   a) Si $M_{eff} < M_{umb}$ se remuestrea:
   - Mustrear M veces el índice $j \in \{1, \ldots, M\}$ de acuerdo a la distribución discreta $Prob(j = l) = \omega_t^{(i)}$ siendo $l = 1, \ldots, M$; generando $\{j^{(i)}\}_{i=1}^M$.
   - Asignar $\mathbf{x}_t^{(i)} = \mathbf{x}_t^{j^{(i)}}$.
   - Asignar $\omega_t^{(i)} = 1/M$.
5. $\mathbf{x}_t^{(i)} = \mathbf{x}_{t-1}^{(i)} + \mathbf{u}_t^{(i)}$.
6. $t = t + 1$, ir al paso 1.

Tabla 3.1: Pseudocódigo del algorimto SIR

3.1.5. Muestreo de Gibbs

El muestreo de Gibbs es un algoritmo ampliamente usado para generar una secuencia de muestras de una distribución de probabilidad determinada. Es un caso especial del algoritmo Metropolis-Hastings [2]. El algoritmo de Gibbs permite simular una cadena de Markov con una distribución de equilibrio que coincide con la deseada. Cada valor de la cadena se obtiene a través de un proceso iterativo que sólo requiere generar muestras de distribuciones que en la mayoria de los casos tienen una forma sencilla.

Sea $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_k]^T$ el vector de variables de interes y sean $\pi(\theta_1|\theta_2, \ldots, \theta_k)$, $\pi(\theta_2|\theta_1, \theta_3, \ldots, \theta_k)$,...,$\pi(\theta_k|\theta_1, \ldots, \theta_{k-1})$ las distribuciones condicionales *completas* obtenidas a partir de la distribución final $\pi(\boldsymbol{\theta})$, se dice *completa* porque todos los componentes de $\boldsymbol{\theta}$ están presentes en la distribución condicional. Entonces, dado un punto inicial $\boldsymbol{\theta}^{(0)} = [\theta_k^{(0)}, \ldots, \theta_k^{(0)}]^T$, se procede a generar muestras $\boldsymbol{\theta}^{(i+1)}$ a partir de $\boldsymbol{\theta}^{(i)}$ de la siguiente manera:

- Generar $\theta_1^{(i+1)}$ a partir de $\pi(\theta_1|\theta_2^{(i)}, \ldots, \theta_k^{(i)})$,
- Generar $\theta_2^{(i+1)}$ a partir de $\pi(\theta_2|\theta_1^{(i+1)}, \theta_3^{(i)}, \ldots, \theta_k^{(i)})$,
- ...
- Generar $\theta_k^{(i+1)}$ a partir de $\pi(\theta_k|\theta_1^{(i+1)}, \ldots, \theta_{k-1}^{(i+1)})$,

con lo que se obtiene $\boldsymbol{\theta}^{(i+1)} = [\theta_1^{(i+1)}, \ldots, \theta_k^{(i+1)}]^T$. La sucesión que se obtiene es una realización de una cadena de Markov con la distribución final como distribución estacionaria.

Este proceso exige que sea posible muestrear a partir de las distribuciones condicionales *completas*. Para ello, en este trabajo se emplea el método aceptación-rechazo ("*Rejection Sampling* "). El muestreo suele ser trivial si el vector $\boldsymbol{\theta}$ se descompone de forma que todas esas distribuciones condicionales sean univariantes. Por otra parte, no es necesario seguir el orden seguido anteriormente para generar las muestras, el resultado es válido cualquiera que sea el orden.

Hasta ahora no existen resultados teóricos de fácil utilización en aplicaciones concretas, sobre cuándo se puede considerar que ha convergido la cadena de Markov. En especial, sigue siendo un tema de investigación atractivo la búsqueda de métodos que sugieran automáticamente el número de muestras de las etapas iniciales de la cadena que deben ser desechadas, e incluso si es más eficiente utilizar una o varias cadenas independientes para monitorizar esa converegencia.

3.2. Conclusiones

En este capítulo se ha realizado una breve descripción de algunos de los métodos de Monte Carlo más conocidos. Se ha descrito la teoría de muestreo exacto que nos ha servido como primera aproximación o caso ideal. Debido a las limitaciones prácticas que presenta se ha pasado al muestreo enfatizado. Este, a su vez, presenta problemas de coste computacional de cara a realizar un uso secuencial, ya que dicho coste crece con el tiempo. Para resolver este problema se ha introducido el algoritmo SIS que opera secuencialmente. Sin embargo, este método presenta el problema de degeneración de pesos de las partículas con el paso de tiempo. A continuación, se ha introducido el algoritmo SIR que resuelve el problema de degeneración de los pesos utilizando el remuestreo. Finalmente se ha presentado el muestreo de Gibbs que permite simular una cadena de Markov con una ditribución de equilibrio que coincide con la de interés.

Capítulo 4

Posicionamiento de un único objetivo

Este capítulo está dividido en dos partes. En la primera, se detallan todos los algoritmos de posicionamiento que se han estudiado. La segunda parte está dedicada a la evaluación numérica de los algoritmos. Para ello, se han simulado varios ejemplos mediante MATLAB. Se presenta, para cada método, un ejemplo de estimación y los errores cometidos.

4.1. Algoritmos

4.1.1. Filtro de partículas

Se ha empleado el algoritmo SIR descrito en la Sección 3.1.4. Se ha considerado la ecuación de estado

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{u}_t, \quad \mathbf{u}_t \sim N(0, \sigma_{u_t}^2), \tag{4.1}$$

donde $\mathbf{x}_t$ es la posición del objetivo en el instante t y $\mathbf{u}_t$ es ruido blanco de media cero y varianza $\sigma_{u_t}^2$. La densidad de probabilidad (fdp) inicial $p(\mathbf{x}_0)$ es conocida,

y viene caracterizada por una ecuación de transición markoviana, caracterizada a su vez por una fdp condicionada $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ también conocida. En este trabajo se aborda el posicionamiento estático, i.e., la posición verdadera del blanco es $\mathbf{x} = \mathbf{x}_t$ para $\forall t$. Por ello, escogemos $\sigma_{u_t}^2 << 1$ de modo que la variación entre $\mathbf{x}_t$ y $\mathbf{x}_{t-1}$ es suficientemente pequeña.
La ecuación de observaciones está descrita en la Sección 2.1.1,

$$y_{j,t} = 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}_t - \mathbf{b}_j\|^\gamma}\right) + v_{j,t}, \quad v_{j,t} \sim N(0, \sigma_{v_t}^2). \tag{4.2}$$

Las observaciones son condicionalmente independientes entre sí. Así, suponemos que la verosimilitud $p(\mathbf{y}_t|\mathbf{x}_t)$ es conocida. A continuación se describe el funcionamiento del filtro.

Inicialmente se genera un conjunto de partículas $\{\mathbf{x}_0^{(i)}, i = 1, \ldots, M\}$, muestreando a partir de $p(\mathbf{x}_0)$. La estructura de la partícula i-ésima es $\mathbf{x}_t^{(i)} = \left[x_{t,1}^{(i)}, x_{t,2}^{(i)}\right]^T$, que es la misma que la del objetivo. Además, se inicializan todos los pesos $\{\omega_0^{(i)}, i = 1, \ldots, M\}$ a un valor fijo $1/M$. La posición de los sensores $\{\mathbf{b}_j, i = 1, \ldots, K\}$ es conocida para el algoritmo.

El modelo para la actualización de las partículas a cada iteración del algoritmo se basa en la ecuación de estado (Ecuación 4.1), es decir, se actualizan las partículas, $\mathbf{x}_t^{(i)}$ a partir de la información en el instante anterior $\mathbf{x}_{t-1}^{(i)}$

$$\mathbf{x}_t^{(i)} = \mathbf{x}_{t-1}^{(i)} + \mathbf{u}_t^{(i)}, \tag{4.3}$$

donde $\mathbf{u}_t^{(i)}$ es una muestra de la variable aleatoria compleja $N(0, \sigma_u^2)$ y representa las fuerzas no medibles que afectan al estado actual [1].
A partir de las observaciones se calculan los pesos de cada partícula. Para ello, hay que obtener la verosimilitud de cada una de ellas,

$$p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) = \prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}_t^{(i)}), \tag{4.4}$$

donde $p(y_{j,t}|\mathbf{x}_t^{(i)})$ tiene la forma

$$p(y_{j,t}|\mathbf{x}_t^{(i)}) = \frac{1}{\sqrt{2\pi\sigma_v^2}} e^{-\frac{1}{\sigma_v^2}(y_{j,t}-m_{j,t}^{(i)})^2}, \tag{4.5}$$

donde

$$m_{j,t}^{(i)} = 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}_t^{(i)} - \mathbf{b}_j\|^\gamma}\right), \tag{4.6}$$

es la potencia recibida en la posición $\mathbf{b}_j$ supuesto que $\mathbf{x}_t = \mathbf{x}_t^{(i)}$.
El peso sin normalizar de la partícula i-ésima es

$$\tilde{\omega}_t^{(i)} = \tilde{\omega}_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}), \tag{4.7}$$

que se normaliza como

$$\omega^{(i)} = \frac{\tilde{\omega}^{(i)}}{\sum_{k=1}^{M} \tilde{\omega}^{(k)}}. \tag{4.8}$$

Después de obtener los pesos normalizados, se estudia la necesidad de realizar un remuestreo de partículas, para ello se calcula el valor de $\hat{M}_{eff}$, como indica la Ecuación (3.19). El umbral elegido es $M_{umb} = 2M/3$ [1].

Después de obtener los pesos normalizados y realizar el remuestreo si es necesario, se procede a realizar la estimación de la posición del objetivo. Tenemos que

$$\mathbf{x}_t^{MMSE} = E[\mathbf{x}_t|\mathbf{y}_t] = \int p(\mathbf{x}_t|\mathbf{y}_t)\mathbf{x}_t d\mathbf{x}_t \approx \int p_M(\mathbf{x}_t|\mathbf{y}_t)\mathbf{x}_t d\mathbf{x}_t, \tag{4.9}$$

se puede calcular $\int p_M(\mathbf{x}_t|y_t)\mathbf{x}_t d\mathbf{x}_t$ de la siguiente manera:

$$\int p_M(\mathbf{x}_t|\mathbf{y}_t)\mathbf{x}_t d\mathbf{x}_t = \sum_{i=1}^{M} \omega_t^{(i)} \int \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)})\mathbf{x}_t d\mathbf{x}_t = \sum_{i=1}^{M} \omega_t^{(i)}\mathbf{x}_t^{(i)} \approx \mathbf{x}_t^{MMSE}. \tag{4.10}$$

está estimación es una aproximación del estimador de error cuadrático medio mínimo ("*Minimum Mean Square Error* ", MMSE) [1]. De este modo la estimación de la posición es

$$\hat{\mathbf{x}}_t = \sum_{i=1}^{M} \omega_t^{(i)} \mathbf{x}_t^{(i)}. \tag{4.11}$$

En la Tabla 4.1 se resume el algoritmo del filtro de partículas.

4.1.2. Filtro de Kalman extendido

El filtro de Kalman extendido (EKF,"*Extended Kalman Filter* ") es una variación del filtro de Kalman que se utiliza cuando la ecuación de estado y/o la ecuación de observación son no lineales. En nuestro caso sólo la ecuación de observación es no lineal. Para manejarla se realiza una linealización local de la función de observaciones utilizando el desarollo de Taylor. El EKF se basa en aplicar el filtro de Kalman al modelo lineal resultante de está aproximación. Antes de detallar el EKF, se presenta el filtro de Kalman.

Filtro de Kalman

El filtro de Kalman (KF, "*Kalman Filter* ") es un conjunto de ecuaciones recursivas que permiten calcular la fdp exacta. Así, el objetivo del filtro de Kalman es calcular la fdp a *posteriori* de un sistema dinámico, cuyo ecuación de estado se detalla a continuación. El sistema se representa con un modelo lineal y el ruido del sistema y de las observaciones tiene una distribución normal con media cero y varianza conocida. Así, las ecuaciones de estado y de observaciones tienen la forma

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A}\mathbf{x}_{t-1} + \mathbf{u}_t, \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{v}_t, \end{aligned} \tag{4.12}$$

Inicialización:

1. Muestrear $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0), \quad i = 1, \ldots, M.$
2. Fijar $\omega_0^{(i)} = \frac{1}{M} \quad$ para $\quad i = 1, \ldots, M.$

Para cada t:

1. Actualizar los pesos: $\tilde{\omega}_t^{(i)} = \tilde{\omega}_{t-1}^{(i)} e^{-\frac{1}{\sigma_v^2} \sum_{j=1}^{K} (y_{j,t} - m_{j,t}^{(i)})^2}$.
2. Normalizar los pesos: $\omega_t^{(i)} = \frac{\tilde{\omega}_t^{(i)}}{\sum_{k=1}^{M} \tilde{\omega}_{t-1}^{(k)}}$.
3. Estimación: $\hat{\mathbf{x}}_t = \sum_{i=1}^{M} \omega_t^{(i)} \mathbf{x}_t^{(i)}$.
4. Estudiar la necesidad del remuestreo:
   a) Tamaño efectivo de la muestra $M_{eff} = \frac{1}{\sum_{i=1}^{M} (\omega_t^{(i)})^2}$
   b) Si $M_{eff} < M_{umb}$ se remuestrea:
   - Muestrear M veces el índice $j \in \{1, \ldots, M\}$ de acuerdo a la distribución discreta $Prob(j = l) = \omega_t^{(i)}$ siendo $l = 1, \ldots, M$; generando $\{j^{(i)}\}_{i=1}^{M}$.
   - Asignar $\mathbf{x}_t^{(i)} = \mathbf{x}_t^{j^{(i)}}$.
   - Asignar $\omega_t^{(i)} = 1/M$.
5. $\mathbf{x}_t^{(i)} = \mathbf{x}_{t-1}^{(i)} + \mathbf{u}_t^{(i)}$, donde $\mathbf{u}_t^{(i)} = N(0, \sigma_u^2)$.
6. $t = t + 1$, ir al paso 1.

Tabla 4.1: Pseudocódigo del filtro de partículas.

donde la matriz $\mathbf{A}$ relaciona el estado en el instante de tiempo $t-1$, con el estado en el instante t. La matriz $\mathbf{C}$ relaciona el estado $\mathbf{x}_t$ con la observación $\mathbf{y}_t$. Las variables aleatorias $\mathbf{u}_t$ y $\mathbf{v}_t$ representan el error del proceso y de la observación, respectivamente, con distribución de probabilidad normal con media cero y matriz de covarianza $\mathbf{Q}$ y $\mathbf{R}$. Por simplicidad, se supone que las cuatro matrices $\mathbf{A}$, $\mathbf{C}$, $\mathbf{Q}$ y $\mathbf{R}$ son constantes.
Las ecuaciones de predicción del filtro de Kalman son

$$\begin{aligned} \bar{\mathbf{x}}_{t|t-1} &= \mathbf{A}_t\bar{\mathbf{x}}_{t-1} + \mathbf{u}_t. \\ \bar{\mathbf{P}}_t &= \mathbf{A}_t\mathbf{P}_{t-1}\mathbf{A}_t^T + \mathbf{Q}_t. \end{aligned} \tag{4.13}$$

donde $\mathbf{P}_{t-1}$ es la covarianza de la f.d.p. $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$. $\bar{\mathbf{P}}_t$ es la covarianza de la f.d.p $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$.
Las ecuaciones de actualización del filtro de Kalman son

$$\mathbf{K}_t = \bar{\mathbf{P}}_t\mathbf{C}_t^T(\mathbf{C}_t\bar{\mathbf{P}}_t\mathbf{C}_t^T + \mathbf{R}_t)^{-1}, \tag{4.14}$$

$$\bar{\mathbf{x}}_t = \bar{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{C}_t\bar{\mathbf{x}}_{t|t-1}), \tag{4.15}$$

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{C}_t)\bar{\mathbf{P}}_t, \tag{4.16}$$

donde $\mathbf{P}_t$ es la covarianza de la f.d.p. $p(\mathbf{x}_t|\mathbf{y}_t)$. La segunda etapa se empieza con el cálculo de la ganancia de Kalman, $\mathbf{K}_t$, mediante (4.14). Se selecciona $\mathbf{K}_t$, de tal manera que se minimice la covarianza del error de la nueva estimación del estado. A continuación se genera una nueva estimación del estado mediante la Ecuación (4.15), incorporando una nueva observación. Finalmente, se obtiene una nueva estimación de la matriz de covarianza del error usando la Ecuación (4.16). En la Tabla 4.2 se resume el algoritmo de Kalman.

Filtro de Kalman extendido

Como se ha mencionado anteriormente, el filtro de Kalman extendido es una variante del filtro de Kalman visto en la sección anterior. Está pensado espe-

Etapa de predicción

1. $\bar{\mathbf{x}}_{t|t-1} = \mathbf{A}_t \mathbf{x}_{t-1} + \mathbf{u}_t$.
2. $\bar{\mathbf{P}}_t = \mathbf{A}_t \mathbf{P}_{t-1} \mathbf{A}_t^T + \mathbf{Q}_t$.

Etapa de actualización:

3. Cómputo de la ganancia de Kalman.
   $\mathbf{K}_t = \bar{\mathbf{P}}_t \mathbf{C}_t^T (\mathbf{C}_t \bar{\mathbf{P}}_t \mathbf{C}_t^T + \mathbf{R}_t)^{-1}$.
4. Actualización del estado con la medida $\mathbf{y}_t$.
   $\mathbf{x}_t = \bar{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{C}_t \bar{\mathbf{x}}_t)$.
5. Actualización de la covarianza del error.
   $\mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{C}_t)\bar{\mathbf{P}}_t$
6. Devuelve $\mathbf{x}_t$, $\mathbf{P}_t$

Tabla 4.2: Algoritmo del filtro de Kalman.

cialmente para los casos en que la ecuación de estado y/o observaciones son no lineales. Estas últimas, en general tienen la forma

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + \mathbf{u}_t, \tag{4.17}$$

$$\mathbf{y}_t = h(\mathbf{x}_t) + \mathbf{v}_t. \tag{4.18}$$

El filtro de Kalman extendido se basa en aplicar el filtro de Kalman al modelo lineal resultante de una linealización local de las funciones f y h utilizando el desarollo de Taylor. En nuestro caso las ecuaciones de estado y de observaciones son respectivamente,

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{u}_t, \tag{4.19}$$

$$y_{j,t} = 10 \log_{10} \left(\frac{P_0}{\|\mathbf{x} - \mathbf{b}_j\|^\gamma} \right) + v_{j,t}, \quad v_{j,t} \sim N(0, \sigma_{v_t}^2), \tag{4.20}$$

de manera que la ecuación de estado es lineal pero la ecuación de observación no es lineal. Las ecuaciones de predicción y de estimación son similares a las del filtro de Kalman, así que, las ecuaciones de predicción tienen la forma

$$\bar{\mathbf{x}}_{t|t-1} = \bar{\mathbf{x}}_{t-1} + \mathbf{u}_t, \tag{4.21}$$

$$\bar{\mathbf{P}}_t = \mathbf{A}_t \mathbf{P}_{t-1} \mathbf{A}_t^T + \mathbf{Q}_t, \tag{4.22}$$

la etapa de predicción es igual que en el KF estático.
En las ecuaciones de actualización se empieza con el cálculo de la ganancia de Kalman, a continuación se actualiza la estimación del estado y la covarianza del error. Las ecuaciones de actualización tienen la forma

$$\mathbf{K}_t = \bar{\mathbf{P}}_t \mathbf{H}_t^T (\mathbf{H}_t \bar{\mathbf{P}}_t \mathbf{H}_t^T + \mathbf{R}_t)^{-1} \tag{4.23}$$

$$\bar{\mathbf{x}}_t = \bar{\mathbf{x}}_{t|t-1} + \mathbf{K}_t (\mathbf{y}_t - h(\bar{\mathbf{x}}_{t|t-1})) \tag{4.24}$$

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \bar{\mathbf{P}}_t, \tag{4.25}$$

donde la matriz $\mathbf{H}_t$ es la jacobiana de la función de observación respecto del estado del proceso se calcula de la siguiente manera:

$$\mathbf{H}_t = \begin{bmatrix} \frac{\partial h_1}{\partial x_{t,1}} & \frac{\partial h_1}{\partial x_{t,2}} \\ \vdots & \vdots \\ \frac{\partial h_K}{\partial x_{t,1}} & \frac{\partial h_K}{\partial x_{t,2}} \end{bmatrix}_{K \times 2} . \tag{4.26}$$

En la Tabla 4.3 se resume el filtro de Kalman extendido.

4.1.3. Algoritmo de gradiente

El algoritmo de gradiente es un método análitico básico de optimización pero es bastante eficiente, es un método iterativo que a partir de un iterante inicial va calculando sucesivos iterantes que se van acercando a la solución exacta de una

Etapa de predicción

1. $\bar{\mathbf{x}}_{t|t-1} = \mathbf{x}_{t-1} + \mathbf{u}_t$.
2. $\bar{\mathbf{P}}_t = \mathbf{G}_t \mathbf{P}_{t-1} \mathbf{G}_t^T + \mathbf{Q}_t$.

Etapa de actualización:

3. Cómputo de la ganancia de Kalman.
   $\mathbf{K}_t = \bar{\mathbf{P}}_t \mathbf{H}_t^T (\mathbf{H}_t \bar{\mathbf{P}}_t \mathbf{H}_t^T + \mathbf{R}_t)^{-1}$.
4. Actualización del estado con la medida $\mathbf{y}_t$.
   $\mathbf{x}_t = \bar{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - h(\bar{\mathbf{x}}_{t|t-1}))$.
5. Actualización de la covarianza del error.
   $\mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t)\bar{\mathbf{P}}_t$
6. Devuelve $\bar{\mathbf{x}}_t$, $\mathbf{P}_t$

Tabla 4.3: Algoritmo del filtro de Kalman extendido.

función. Para encontrar esa solución se sigue la dirección de las derivadas parciales de la función en cuestión. En la Tabla 4.4 se presenta la iteración canónica del método del gradiente. En la Figura 4.1 se presenta el esquema de iteración del método del gradiente.

Se predente estimar la posición $\mathbf{x} = [x_1, x_2]^T$ de un objetivo, empleando el modelo de observaciones descrito en la Sección 2.1.1. La ecuación de observaciones tiene la forma

$$y_{j,t} = 10 \log_{10} \left(\frac{P_0}{\|\mathbf{x} - \mathbf{b}_j\|^\gamma} \right) + v_{j,t},\ v_{j,t} \sim N(0, \sigma_v^2), j = 1, \ldots, K,$$

donde $y_{j,t}$ es la observación que proviene del sensor j-ésimo en el instante t. Así, en el mismo instante se obtiene un conjunto de observaciones de todos los sensores $\mathbf{y}_t = [y_{1,t}, \ldots, y_{K,t}]$, K es el número de sensores. En un periodo τ se obtiene el vector de observaciones de todos los sensores $\mathbf{y} = \left[\mathbf{y}_1^T, \ldots, \mathbf{y}_\tau^T\right]^T$.

Inicialización

1. Seleccionar un punto inicial $\mathbf{x}^0 \in \mathbb{R}^2$. Fijar $k = 1$ y $\mu << 1$.

Iteración

1. Calcular el gradiente $d^k = -\nabla f(\mathbf{x})$.
2. Actualizar la posición: $\mathbf{x}^k = \mathbf{x}^{k-1} + \mu d^k$.
3. $k = k + 1$ y volver a 1.

Tabla 4.4: Algoritmo de gradiente.

El estimador de máxima verosimilitud (ML) de $\mathbf{x}$ es

$$\hat{\mathbf{x}}_{ML} = \arg \max_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}), \tag{4.27}$$

suponiendo que el ruido es independiente tenemos que

$$\hat{\mathbf{x}}_{ML} = \arg \max_{\mathbf{x}} \prod_{t=1}^{\tau} \prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}) \tag{4.28}$$

donde

$$p(y_{j,t}|\mathbf{x}) \sim N\left(y_{j,t}|10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^{\gamma}}\right), v_j^2\right). \tag{4.29}$$

Aplicando el método de gradiente tenemos,

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mu \nabla_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}), \tag{4.30}$$

con $p(\mathbf{y}|\mathbf{x}) = \prod_{t=1}^{\tau} \prod_{j=1}^{K} p(y_{j,t}|\mathbf{x})$. Entonces hay que calcular el gradiente de la verosimilitud $p(\mathbf{y}|\mathbf{x})$ que tiene la forma

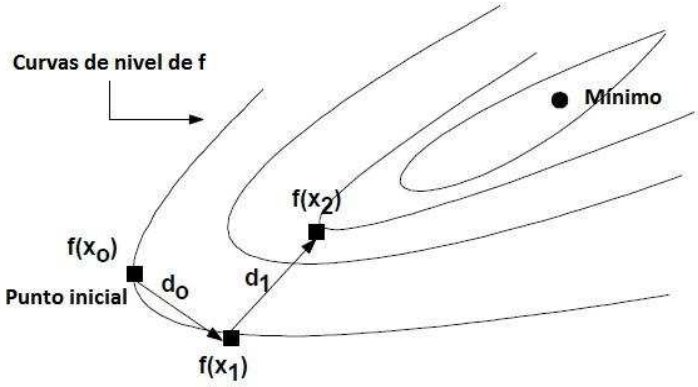


Figura 4.1: Esquema de iteración del método de gradiente.

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \prod_{t=1}^{\tau}\prod_{j=1}^{K}\frac{1}{\sqrt{2\pi\sigma_v^2}}e^{-\frac{1}{\sigma_v^2}(y_{j,t}-m_j)^2} \\ &= \frac{1}{(\sqrt{2\pi\sigma_v^2})^{K\tau}}e^{-\frac{1}{\sigma_v^2}\sum_{t=1}^{\tau}\sum_{j=1}^{K}(y_{j,t}-m_j)^2}, \end{aligned} \tag{4.31}$$

donde $m_j = 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^\gamma}\right)$. Si ponemos que

$$f(\mathbf{x}) = \sum_{t=1}^{\tau}\sum_{j=1}^{K}\left(y_{j,t} - 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^\gamma}\right)\right)^2, \tag{4.32}$$

el calculo del estimador $\hat{\mathbf{x}}_{ML}$ se reduce a

$$\hat{\mathbf{x}}_{ML} = \arg\min_{\mathbf{x}} f(\mathbf{x}). \tag{4.33}$$

Para ello, se aplica

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \mu\nabla f(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^n}, \tag{4.34}$$

con

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f(\mathbf{x})}{\partial x_1}, \frac{\partial f(\mathbf{x})}{\partial x_2}\right]^T . \tag{4.35}$$

Las derivadas parciales de la función $f(\mathbf{x})$ (4.32) tienen la forma

$$\begin{aligned} \frac{\partial f(\mathbf{x})}{\partial x_i} &= \sum_{t=1}^{\tau}\sum_{j=1}^{K}\left(y_{j,t} - 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^\gamma}\right)\right) S \\ S &= \frac{20\gamma}{\log_e(10)\|\mathbf{x}-\mathbf{b}_j\|^2}(x_i - b_{j,i}) \quad i = 1, 2. \end{aligned} \tag{4.36}$$

En la Tabla 4.4 se resume el algoritmo.

4.1.4. Búsqueda aleatoria acelerada

La búsqueda aleatoria acelerada ("*Accelerated Random Search*", ARS) es un algoritmo de optimización global [7]. Se basa en realizar una serie de búsquedas aleatorias sobre conjuntos de medidas variables, el mayor de los cuales contiene la solución. Es un algoritmo iterativo. La búsqueda se realiza repetitivamente, de manera local, en conjuntos de búsqueda cada vez más pequeños alrededor de la última solución disponible, hasta que esta última se ve mejorada, instante en el cual se vuelve a buscar sobre el conjunto entero.

El objetivo es estimar la posición de un blanco que está situado en $\mathbf{x} = [x_1, x_2]^T$. Aplicamos el algoritmo ARS para calcular el estimador de máxima versomilitud (ML)

$$\hat{\mathbf{x}}_{ML} = \arg\min_{\mathbf{x}} f(\mathbf{x}), \tag{4.37}$$

donde

$$f(\mathbf{x}) = \sum_{t=1}^{\tau}\sum_{j=1}^{K}\left(y_{j,t} - 10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^\gamma}\right)\right)^2, \tag{4.38}$$

es una suma de residuos cuadráticos. Como mostramos en la Sección 4.1.3,

$p(\mathbf{y}|\mathbf{x}) \propto e^{-\frac{1}{2\sigma_v^2} f(\mathbf{x})}$. Los parámetros del algoritmo son R_{max} si este parámetro es grande el dominio de la búsqueda es más grande y por lo tanto el conjunto de búsqueda es más amplio, R_{min} determina la precisión es decir cuanto más pequeño la precisión es mejor y c es el parámetro de contracción la velocidad de búsqueda depende mucho de este parámetro cuanto más grande sea la velocidad es menor. En la Tabla 4.5 se resume el algoritmo ARS.

- Parámetros del algoritmo: radio máximo R_{max}, radio mínimo R_{min} y factor de contracción $c > 1$.
- Algoritmo
  1. Inicialización
     - Se elige un conjunto inicial $[a_1(0), b_1(0)] \times [a_2(0), b_2(0)]$
     - Se genera la estimación inicial $\mathbf{x}^{(0)}$ a partir de una distribución uniforme, $x_1^{(0)} \sim U[a_1(0), b_1(0)]$ y $x_2^{(0)} \sim U[a_2(0), b_2(0)]$.
     - Se calcula el coste $f^{(0)} = f(\mathbf{x}^{(0)})$.
     - Se elige un radio inicial como $r^{(0)} = R_{max}$
  2. Iteración en $n + 1$
     - Se genera un punto $\widetilde{\mathbf{x}} = [\widetilde{x}_1, \widetilde{x}_2]^T$ a partir de una distribución uniforme de la siguiente manera: $\widetilde{x}_1 \sim U[x_1^{(n)} - r^{(n)}, x_1^{(n)} + r^{(n)}]$ y $\widetilde{x}_2 \sim U[x_2^{(n)} - r^{(n)}, x_2^{(n)} + r^{(n)}]$
     - Se calcula el coste de $\widetilde{\mathbf{x}}$ como $\widetilde{f} = f(\widetilde{\mathbf{x}})$
       - Si $\widetilde{f} < f^{(n)}$ entonces: $\mathbf{x}^{(n+1)} = \widetilde{\mathbf{x}}$, $f^{(n+1)} = \widetilde{f}$ y $r^{(n+1)} = R_{max}$
       - Si $\widetilde{f} \geq f^{(n)}$ entonces: $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)}$, $f^{(n+1)} = f^{(n)}$ y $r^{(n+1)} = \frac{r^{(n)}}{c}$.
       - Si $r^{(n+1)} < R_{min}$ entonces $r^{(n+1)} = R_{max}$

Tabla 4.5: Algoritmo de búsqueda aleatoria acelerada (ARS).

4.1.5. Muestreo de Gibbs

El muestreo de Gibbs está descrito en la Sección 3.1.5. Requiere la capacidad de muestrear de las distribuciones condicionales de las variables de interés, lo que no siempre es sencillo. Como herramienta para el muestreo, utilizamos el método de aceptación/rechazo [8], que se describe a continuación.

Método de aceptación/rechazo

El objetivo es generar una variable aleatoria X con función de densidad $r(x)$. Para ello se puede emplear varios métodos, pero en este trabajo se ha optado por emplear el método de aceptación/rechazo que consiste en generar una variable aleatoria Y con función de densidad tentativa $g(y)$, la cual se puede generar fácilmente. Entonces, se genera un valor de Y con densidad $g(y)$ y se genera una variable aleatoria uniforme $U \sim U(0,1)$. Se acepta la muestra Y, si $U \leq \frac{r(Y)}{cg(Y)}$ con c es una constante tal que $r(y)/g(y) \leq c$ para todo y. Sino la muestra es rechazada. Se puede resumir los pasos del método en lo siguiente:

1. Generar Y con densidad g
2. Generar una variable aleatoria U uniforme $U[0,1]$
3. Si $U \leq \frac{r(Y)}{cg(Y)}$ se acepta la muestra Y de lo contrario volver al paso 1.

Mientras más cerca se encuentra r de g más rápidamente se obtendrá la cantidad deseada de valores aleatorios de X. Usualmente se toma la densidad uniforme como la densidad $g(y)$ y en ese caso el método de aceptación rechazo es llamado "*hit and miss* ".

Aplicación al escenario

El objetivo es determinar la posición $\mathbf{x} = [x_1, x_2]^T$ a partir de una colección de muestras de la distribución a *posteriori*, con fdp $p(\mathbf{x}|\mathbf{y})$. Se genera una secuencia de muestras $x_1^{(i)}$ y $x_2^{(i)}$, empleando Gibbs [2], y después se calcula la media,

rechazando un número n de las muestras de las etapas iniciales para asegurar la convergencia. Las muestras se generan a partir de las densidades de probabilidad condicionales. Para obtener la muestra $\mathbf{x}^{(i)}$, se generan $x_1^{(i)} \sim p(x_1|x_2^{(i-1)}, \mathbf{y})$ y $x_2^{(i)} \sim p(x_2|x_1^{(i)}, \mathbf{y})$. Este procedimiento se itera hasta obtener un número N de muestras. En la Tabla 4.6 se resume el algoritmo de Gibbs.

1. Seleccionar un punto inicial $\mathbf{x}^{(0)} = [x_1^{(0)}, x_2^{(0)}]^T$.
2. Para $i = 1, \ldots, N$
   - $x_1^{(i)} \sim p(x_1|x_2^{(i-1)}, \mathbf{y})$
   - $x_2^{(i)} \sim p(x_2|x_1^{(i)}, \mathbf{y})$
3. Para la estimación, se calcula la media rechazando las n muestras de las etapas iniciales:

$$\hat{\mathbf{x}} = \frac{1}{N-n} \sum_{i=n}^{N} \mathbf{x}^{(i)}.$$

Tabla 4.6: Muestreo de Gibbs

Para generar las muestras se emplea el método de aceptación/rechazo [8], porque no es fácil muestrear a partir de la densidad de probabilidad $p(x_i|x_j, \mathbf{y})$. Si empleamos el teorema de Bayes tenemos,

$$p(x_1|x_2, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(x_1)}{p(\mathbf{y}|x_2)}, \tag{4.39}$$

y como $p(\mathbf{y}|x_2)$ no depende de x_1, entonces, es constante en función de x_1, con lo cual se puede emplear la aproximación

$$p(x_1|x_2, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{x})p(x_1), \tag{4.40}$$

ahora la función donde vamos a muestrear es $r(x) = p(\mathbf{y}|\mathbf{x})p(x_1)$, si cogemos la

función tentativa como $g(x) = p(x_1)$. La condición de aceptación se reduce a

$$\frac{p(\mathbf{y}|\mathbf{x})p(x_1)}{cp(x_1)} = \frac{p(\mathbf{y}|\mathbf{x})}{c} \geq U \tag{4.41}$$

con U es una variable aleatoria uniforme que se genera de $U(0,1)$. Tenemos,

$$y_{j,t}|x_i \sim N\left(10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^{\gamma}}\right), v_j^2\right), \tag{4.42}$$

entonces,

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \prod_{t=1}^{\tau}\prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}) \\ &= \prod_{t=1}^{\tau}\prod_{j=1}^{K} \frac{1}{\sqrt{2\pi\sigma_v^2}} e^{-\frac{1}{\sigma_v^2}(y_{j,t}-m_j)^2}, \\ &= \frac{1}{\left(\sqrt{2\pi\sigma_v^2}\right)^{K\tau}} e^{-\frac{1}{\sigma_v^2}f(\mathbf{x})}, \end{aligned} \tag{4.43}$$

la función $f(\mathbf{x})$ es la función de residuos cuadráticos (véase la Ecuación 4.32). Si ponemos $c = \frac{1}{\left(\sqrt{2\pi\sigma_v^2}\right)^K}$. Ahora, la condición de aceptación es

$$e^{-\frac{1}{\sigma_v^2}f(\mathbf{x})} \geq U. \tag{4.44}$$

En la Tabla 4.7 se resume el algoritmo.

4.2. Resultados

En está sección se analiza el rendimiento de los algoritmos presentados en este capítulo. Para ello se han simulado casos prácticos mediante MATLAB implementando los diferentes algoritmos. Se presenta para cada algoritmo un ejemplo de simulación con la estimación de la posición y la trayectoria que lleva a está estimación. Para comparar los diferentes algoritmos se presentan gráficas en las

1. Generar una muestra x_i, $i = 1, 2$ de la densidad $p(x)$
2. Generar una variable aleatoria U de $U[0, 1]$
3. Si $U \leq e^{-\frac{1}{\sigma_v^2} f(\mathbf{x})}$, $\mathbf{x} = [x_1, x_2]^T$ elegir la muestra x_i si no volver a 1

Y así, sucesivamente hasta generar todas las muestras.

Tabla 4.7: Método de aceptación/rechazo.

que se muestra el error de la estimación y como evoluciona ese error en función del tiempo.

El número mínimo de sensores para poder realizar una estimación de la posición en el plano es $K = 3$ sensores, ya que, si no hay al menos tres sensores con cobertura sobre el objetivo, se produce una indeterminación. Supongamos que los sensores pudiesen medir sin ruido; en el caso de emplear un único sensor la zona en la que puede estar el objetivo es una circunferencia. Si se emplean $K = 2$ sensores el objetivo puede estar en los dos puntos en los que se cortan las circunferencias determinadas por cada sensor individualmente. En el caso de que se disponga de $K = 3$ sensores sólo existe un punto en el plano en el que puede estar situado el objetivo, la intersección de las tres circunferencias determinadas por los tres sensores.

Para medir las prestaciones de cada algoritmo se ha optado por calcular el error absoluto medio. Para ello, se ejecuta el algoritmo en N_t simulaciones independientes. En cada ejecución se calcula el error de la estimación, que es la distancia entre la posición real y la estimada. Finalmente, se hace una media de todos los errores. Cuantitativamente, se tiene que el error medio es

$$E = \frac{1}{N_t} \sum_{i=1}^{N_t} \left\| \mathbf{x} - \mathbf{x}_{est}^i \right\| \tag{4.45}$$

donde $\mathbf{x}$ es la posición real del objetivo y $\mathbf{x}_{est}^{i}$ la posición estimada en la simulación i-ésima.

4.2.1. Filtro de partículas

La descripción general del algoritmo se presenta en la Sección 4.1.1. Los parámetros utilizados en la implementación del algoritmo se presentan en la Tabla 4.8.

Parámetros	**Valor**
Desviación típica de la ecuación de estado σ_u	3×10^{-6}
Desviación típica de la ecuación de observación σ_v	1
Constante de propagación γ	2
Potencia transmisión P_0	10mW

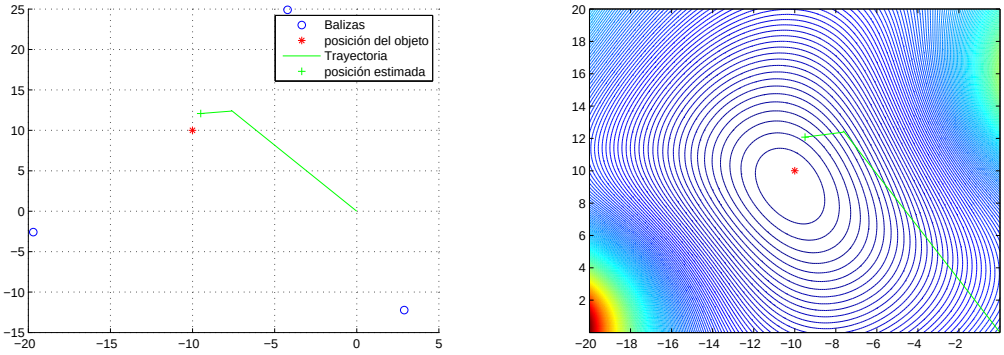
Tabla 4.8: Parámetros de simulación del filtro de partículas.

Se ha elegido una desviación típica de la ecuación de estado, σ_u, muy pequeña, ya que en nuestro caso el objetivo no está en movimiento. La desviación típica de la ecuación de observación, σ_v, introduce errores en las observaciones. Este parámetro caracteriza tanto la calidad de la medida del sensor como los posibles ruidos ambientales. A continuación se presenta un ejemplo de simulación en el que se han empleado $M = 100$ partículas y $K = 3$ sensores.

En la Figura 4.2(a) se representa una estimación de la posición empleando el filtro de partículas con un sistema compuesto por $M = 100$ partículas y $K = 3$ sensores, es decir, el mínimo número de sensores para poder realizar una estimación sin ambiguïedad.

En la Figura 4.2(b) se representan las curvas de nivel de la función $f(\mathbf{x})$ de residuos cuadráticos (véase la Ecuación (4.32)) donde se ve claramente la trayectoria de las estimaciones (en color verde).

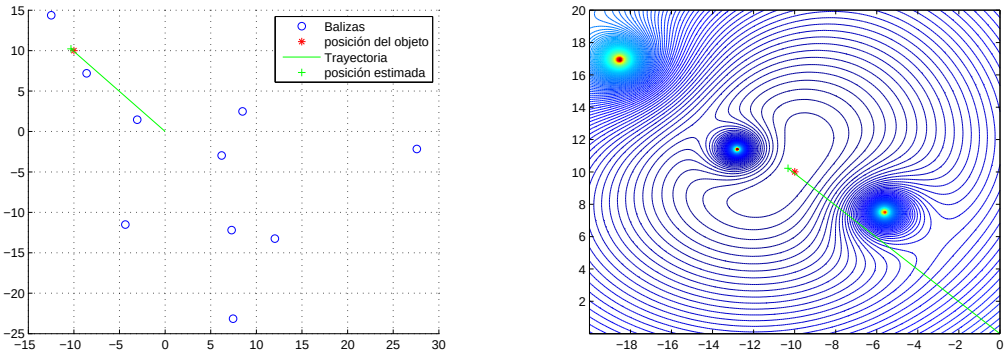
En la Figura 4.3 se representa tanto la estimación como las curvas de nivel



(a) Estimación de la posición (b) Curvas de nivel

Figura 4.2: Estimación de la posición empleando el filtro de partículas con $K = 3$ sensores y $M = 100$ partículas.

de un ejemplo de estimación donde se ha empleado el filtro de partículas con un sistema compuesto por $M = 100$ partículas y $K = 10$ sensores. Como se puede ver en esta figura, se mejora considerablemente la estimación.

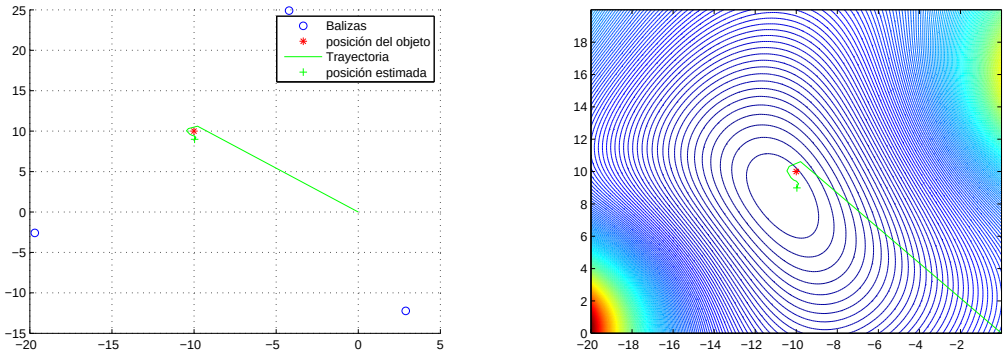


(a) Estimación de la posición (b) Curvas de nivel

Figura 4.3: Estimación de la posición empleando el filtro de partículas con $M = 100$ partículas y $K = 10$ sensores.

Para ver la mejoría del algoritmo en función del número de partículas, en la Figura 4.4, se presenta un ejemplo de simulación con mayor número de partículas y el mismo número de sensores que en el primer ejemplo ($K = 3$ y $M = 100$).

Se han empleado $M = 500$ partículas y $K = 3$ sensores. Como se puede ver, la estimación se ha mejorado un poco respecto al caso donde se han empleado sólo $M = 100$ partículas pero no ha mejorado mucho respecto al caso donde se han empleado $K = 10$ sensores y $M = 100$ partículas. Esto es debido a que cuanto más sensores hay en el recinto, más observaciones tendremos y por lo tanto más información, lo que implica una mejoría en las prestaciones.



(a) Estimación de la posición

(b) Curvas de nivel

Figura 4.4: Estimación de la posición empleando el filtro de partículas con $K = 3$ sensores y $M = 500$ partículas.

En la Figura 4.5 se presenta la evolución del error medio en función del número de partículas. Se observa como el error se decrementa rapidamente con el aumento del número de partículas.

En la Figura 4.6 se presenta la evolución del error en función del tiempo real de ejecución empleando $M = 200$ partículas. Puesto que la posición del objetivo es fija, se consigue reducir el error de estimación con el paso del tiempo.

4.2.2. Filtro de Kalman extendido

Se presentan los resultados de las simulaciones del filtro de Kalman extendido. La descprición general del algoritmo se presenta en la Sección 4.1.2. En la Tabla 4.9 se muestran los parametros utilizados en las simulaciones.

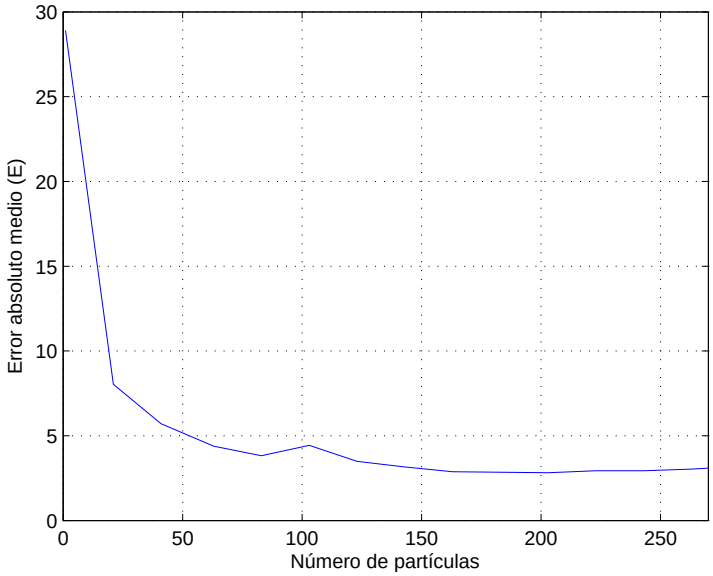


Figura 4.5: Error medio de la estimación empleando el filtro de partículas en función del número de partículas. El número de sensores es $K = 3$ y se han realizado $N_t = 50$ simulaciones independientes.

Como se ve en la Tabla 4.9, igual que en el caso del filtro de partículas, se ha elegido una desviación típica de la ecuación de estado, σ_u, muy pequeña, ya que se considera que el objetivo no está en movimiento. La desviación típica de la ecuación de observación, σ_v, es la misma que se eligió en el caso del filtro de partículas. Este parámetro introduce errores en las observaciones y caracteriza tanto la calidad de la medida como los posibles ruidos ambientales.

En la Figura 4.7 se presenta un ejemplo de la simulación el filtro de Kalman extendido. En la Figura 4.7(a) se representa una estimación de la posición empleando el filtro de Kalman extendido con un sistema compuesto por $K = 3$ sensores, mientras que en la Figura 4.7(b) se representa una gráfica de las curvas de nivel donde se aprecia claramente la convergencia de la secuencia de estimaciones.

En la Figura 4.8 se muestra la evolución del error medio de la estimación en función del número de iteraciones. En está gráfica podemos ver claramente cuantas iteraciones del algoritmo hacen falta para hacer una estimación con un

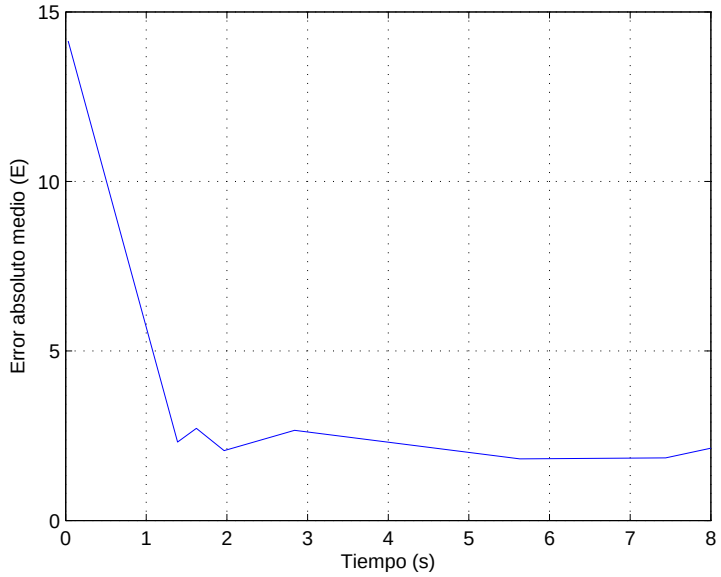


Figura 4.6: Error medio de la estimación empleando el filtro de partículas en función del tiempo real de ejecución. El número de sensores es $K = 3$, se han realizado $N_t = 50$ simulaciones independientes y se han empleado $M = 200$ partículas.

error dado.

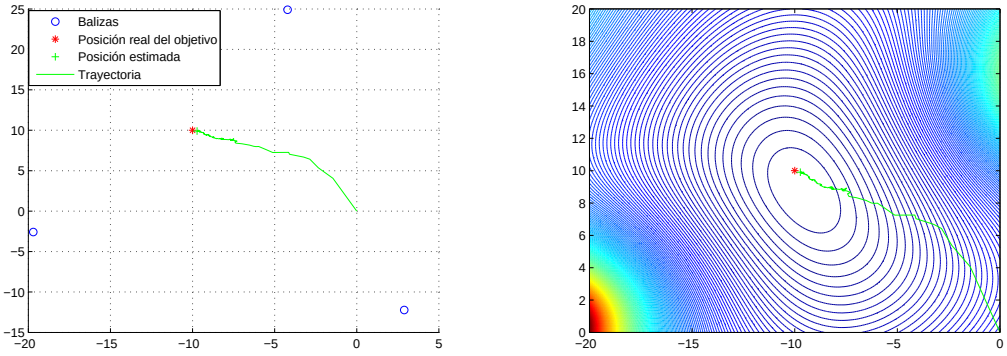
En la Figura 4.9 se aprecia la evolución del error medio de la estimación en función del tiempo real de ejecución del algoritmo. Con el paso del tiempo el error decrece, ya que la posición del objetivo es fija.

4.2.3. Algoritmo de gradiente

Se presentan los resultados de las simulaciones empleando el algoritmo de gradiente descrito en la Sección 4.1.3. En este caso para evitar caer en el problema de la convergencia a algún mínimo local, se ha optado por acudir a un procedimiento simple, que consiste en ejecutar el algoritmo N_r veces, con inicializaciones diferentes, y quedarse con el punto que proporciona el menor valor de la función de coste (función $f(\mathbf{x})$, Ecuación (4.32)).

Parámetros	Valor
Desviación típica de la ecuación de estado σ_u	3×10^{-6}
Desviación típica de la ecuación de observación σ_v	1
Exponente de propagación γ	2
Potencia de transmisión P_0	10mW

Tabla 4.9: Parámetros de simulación del filtro de Kalman extendido



(a) Estimación de la posición

(b) Curvas de nivel

Figura 4.7: Estimación de la posición empleando el filtro de Kalman extendido con un sistema compuesto por $K = 3$ sensores.

Es un algoritmo lento, que requiere un número de iteraciones alto en comparación con el EKF por ejemplo. Se presentan ejemplos de la simulación de este algoritmo, en la Tabla 4.10 se muestran los parámetros de la simulación.

En la Figura 4.10 se presenta un ejemplo de estimación de la posición de un objetivo empleando el algoritmo de gradiente con un sistema compuesto por $K = 3$ sensores. En la Figura 4.10(a) se presenta la estimación de la posición y la trayectoria de convergencia del algoritmo, mientras que en la Figura 4.10(b) se muestran las curvas de nivel de la función de coste.

En la Figura 4.11 se presenta la evolución del error medio cometido en la estimación de la posición de un objetivo empleando el algoritmo de gradiente. El sistema está compuesto por el mínimo número de sensores para poder hacer

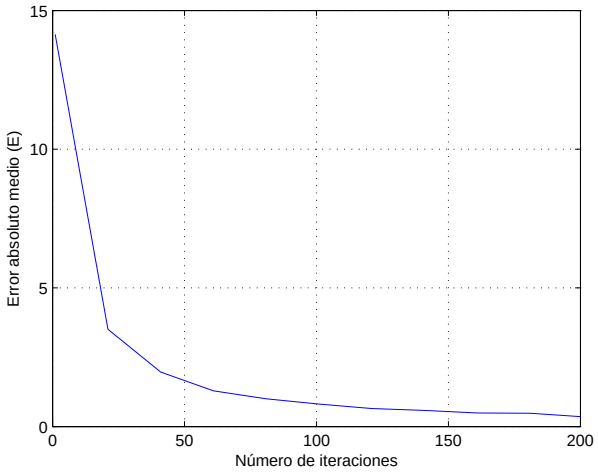


Figura 4.8: Error medio de la estimación empleando el algoritmo del filtro de Kalman extendido en función del número de iteraciones. El número de sensores es $K = 3$ y se han realizado $N_t = 50$ simulaciones independientes.

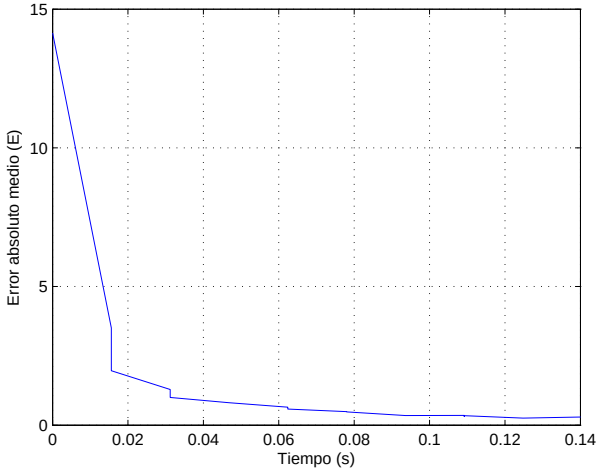
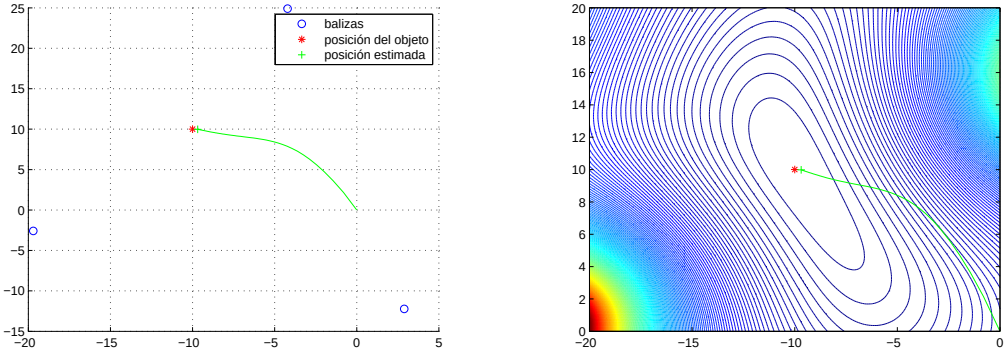


Figura 4.9: Error medio de la estimación empleando el algoritmo del filtro de Kalman extendido en función del tiempo de ejecución. El número de sensores es $K = 3$ y se han realizado $N_t = 50$ simulaciones independientes.

una estimación sin ambiguïdad, es decir sólo $K = 3$ sensores. Como se puede apreciar en la Figura 4.11, con un número pequeño de iteraciones el error medio es grande. Esto es debido a que el algoritmo no llega a converger a la solución

Parámetros	Valor
Desviación típica de la ecuación de observación σ_v	1
Exponente de propagación γ	2
Potencia de transmisión P_0	10mW
Paso de adaptación μ	10^{-4}
Número de repeticiones N_r	2

Tabla 4.10: Parámetros de simulación del algoritmo de gradiente.



(a) Estimación de la posición

(b) Curvas de nivel

Figura 4.10: Estimación de la posición empleando el algoritmo de gradiente con un sistema compuesto por $K = 3$ sensores.

deseada. Con número elevado de iteraciones, el algoritmo converge con un error menor. En la figura se aprecia claramente que con el aumento del número de iteraciones el error se decrementa, hasta llegar a una situación estacionaria.

Para ver el rendimiento del algoritmo, y más concretamente su complejidad, se presenta una gráfica en la Figura 4.12 donde se presenta el error medio de la estimación en función del tiempo de ejecución.

Se puede observar en las dos figuras que el algoritmo de gradiente es un método lento en comparación con el EKF por ejemplo, es decir que converge lentamente a la solución exacta.

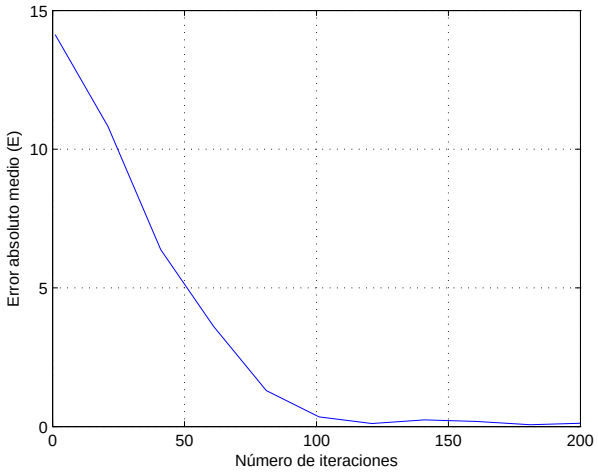


Figura 4.11: Error medio de la estimación empleando el algoritmo de gradiente en función del número de iteraciones. El número de sensores es $K = 3$ y se han realizado $N_t = 50$ simulaciones independientes.

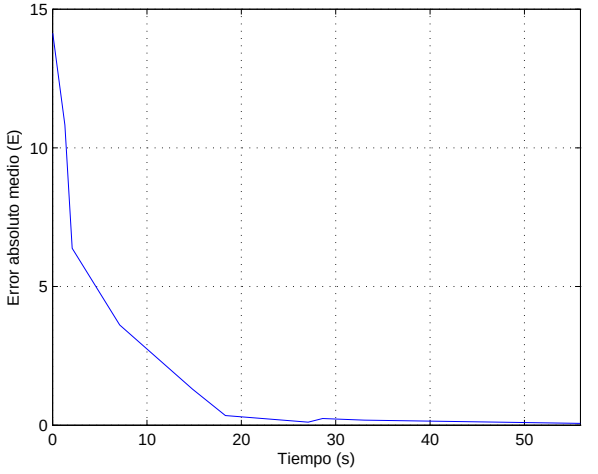


Figura 4.12: Error medio de la estimación empleando el algoritmo de gradiente en función del tiempo real de ejecución. El número de sensores es $K = 3$

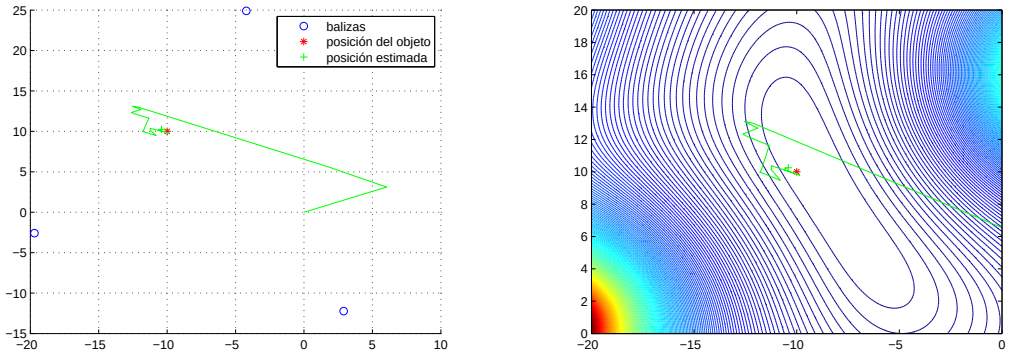
4.2.4. Búsqueda aleatoria acelerada

La descripción general del algoritmo se presenta en la Sección 4.1.4. En la Tabla 4.11 se presentan los parámetros utilizados en las simulaciónes del algoritmo.

En la Figura 4.13 se presenta un ejemplo de simulación donde se ha empleado el algoritmo de búsqueda aleatoria acelerada con un sistema compuesto por $K = 3$ sensores y los parámetros presentados en la tabla.

Parámetros	**Valor**
Desviación típica de la ecuación de observación σ_v	1
Exponente de propagación γ	2
Potencia de transmisión P_0	10mW
Número de iteraciones N	100

Tabla 4.11: Parámetros de simulación del algoritmo búsqueda aleatoria acelerada.



(a) Estimación de la posición

(b) Curvas de nivel

Figura 4.13: Estimación de la posición empleando el algoritmo de la búsqueda aleatoria acelerada con sistema compuesto por $K = 3$ sensores.

En la Figura 4.13 se presentan dos gráficas. En la Figura 4.13(a) se muestra un ejemplo de simulación con la trayectoria de convergencia del algoritmo, mientras que en la Figura 4.13(b) se presentan las curvas de nivel de la función de coste, con la trayectoria de las estimaciones superpuestas.

Para estudiar el rendimiento del algoritmo, en la Figura 4.14 se presenta la evolución del error medio cometido en la estimación de la posición de un objetivo empleando el algoritmo de búsqueda aleatoria acelerada con un sistema compuesto por $K = 3$ sensores. Como se puede ver en la figura, el error se

decrementa rápidamente con el aumento del número de iteraciónes hasta llegar a un punto estacionario.

En la Figura 4.15 se presenta la evolución del error medio cometido en función del tiempo real de ejecución empleando el algoritmo de búsqueda aleatoria acelerada (ARS) con un sistema compuesto por $K = 3$ sensores.

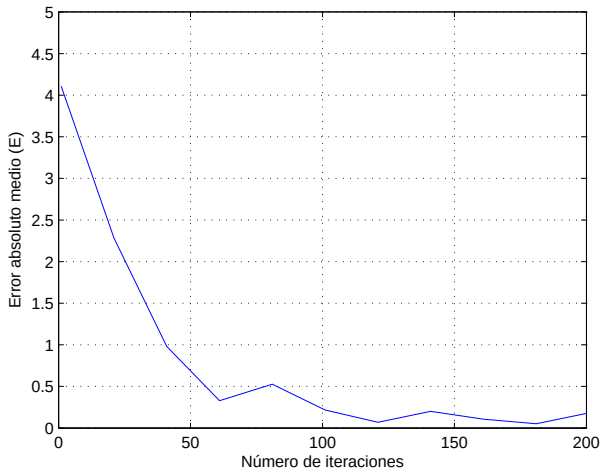


Figura 4.14: Error medio de la estimación empleando el algoritmo de búsqueda aleatoria acelerada (ARS) en función del número de iteraciones. El número de sensores es $K = 3$ y se han realizado $N_t = 50$ simulaciones independientes.

4.2.5. Muestreo de Gibbs

La descripción del algoritmo se presenta en la Sección 4.1.5. Con este algoritmo se ha hecho dificil realizar una estimación con un error comparable al de los métodos anteriores debido a que la tasa de aceptación del método de aceptación/rechazo es muy baja, es decir, que se aceptan muy pocas muestras. Se ha encontrado que está tasa depende fuertemente de la desviación típica de la ecuación de observación σ_v.

La condición de aceptación de una muestra $x_i \sim p(x_i|\mathbf{y}, x_j)$, $i, j \in 1, 2$, $i \neq j$, es $E_a = e^{-\frac{1}{\sigma_v^2} f(\mathbf{x})} \geq U$, donde U es una variable aleatoria $U(0, 1)$. Obviamente, cuanto mayor, es la varianza del ruido observacional, mayor es la tasa de acepta-

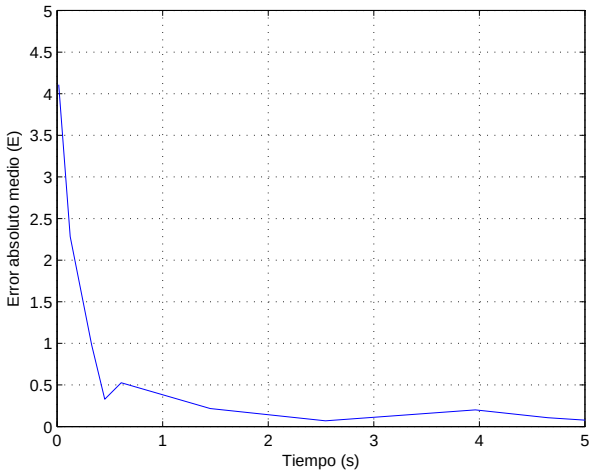


Figura 4.15: Error medio de la estimación empleando el algoritmo de búsqueda aleatoria acelerada (ARS) en función del tiempo de ejecución. El número de sensores es $K = 3$ y se han realizado $N_t = 50$ simulaciones independientes.

ción pero también es mayor la cota inferior del error de estimación. Si la varianza de ruido es pequeña, la cota del error de estimación también es pequeña pero la tasa de aceptación se vuelve tan baja que el algoritmo deja de ser práctico.

Para ver la influencia de este parámetro se presenta una gráfica de la tasa de aceptación en función del mismo. En la Figura 4.16 se presenta la evolución de la tasa de aceptación en función de la desviación típica de la ecuación de observación σ_v. Como se puede ver en la figura, la tasa de aceptación es muy baja para valores de σ_v pequeños, mientras que está tasa sube un poco con valores mayores de σ_v.

4.2.6. Comparación

Se procede a comparar los algoritmos estudiados en este capítulo. En primer lugar se muestran las trayectorias de las estimaciones generadas por cada uno de los algoritmos. Para comparar el rendimiento y la complejidad de los algoritmos se presentan unas gráficas donde se presentan el error absoluto medio y el tiempo real de ejecución respectivamente.

En la Figure 4.17 se presentan gráficas de la convergencia de cada uno de

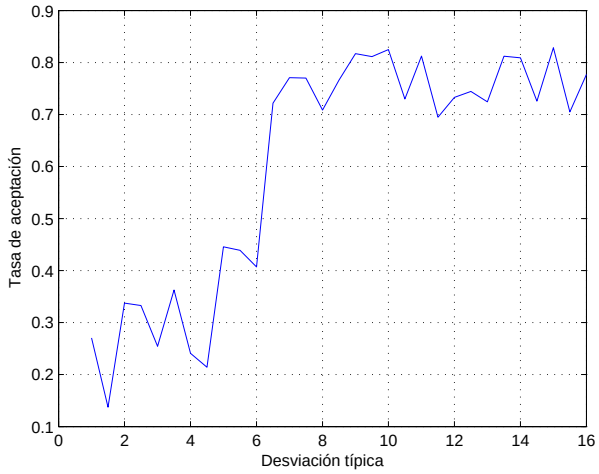
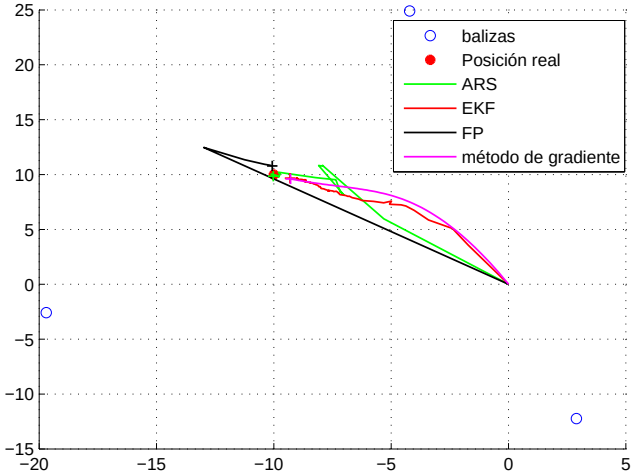


Figura 4.16: Tasa de aceptación en función de la desviación típica de la ecuación de observación σ_v.

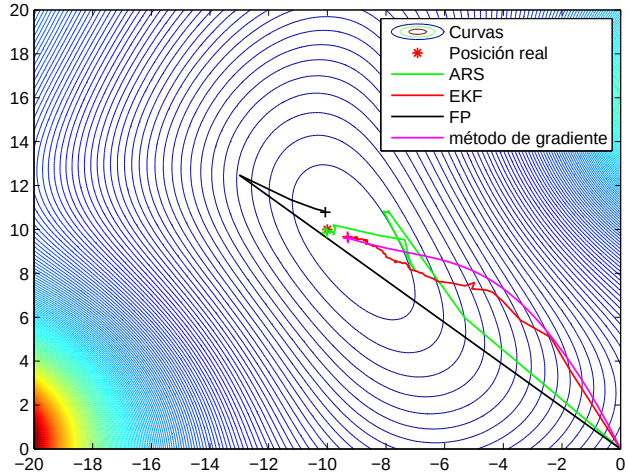
los algoritmos empleando un sistema compuesto por $K = 3$ sensores. Para la estimación se ha utilizado la misma posición y las mismas observaciones. En la Figura 4.17(a) se presenta la convergencia de los algoritmos: búsqueda aleatoria acelerada (ARS), filtro de Kalman extendido (EKF), filtro de partículas (FP) y método de gradiente. En la Figura 4.17(b) se presentan las curvas de nivel con las distintas trayectorias superpuestas.

En la Figura 4.18 se presenta una gráfica, donde se muestra el error absoluto medio cometido en estimar la posición de un blanco empleando un sistema compuesto por $K = 3$ sensores. Como se puede ver en la figura todos los algoritmos convergen a la solución deseada. Los algoritmos FP y EKF convergen considerablemente más rápido que el ARS y el método del gradiente. Al emplear más partículas el rendimiento del filtro de partículas se mejora.

A continuación se procede a comparar la complejidad de los algoritmos. Se presenta una Figura 4.19, donde se presenta el tiempo real de ejecución de los algoritmos estudiados. En la Figura 4.19(a) se muestra el tiempo de ejecución de FP, EKF y el ARS mientras que en la Figura 4.19(b) se muestra el tiempo de ejecución de los tres anteriores más el tiempo de ejecución del método de gradiente. Como se puede ver en las figuras, el algoritmo de EKF presenta el menor



(a) Convergencia de los algoritmos



(b) Curvas de nivel del primer objetivo

Figura 4.17: Convergencia de los algoritmos estudiados. La posición final es el que está marcada con el simbolo +

tiempo de ejecución. El método de gradiente presenta un tiempo de ejecucióm más grande que todos los algoritmos estudiados. Mientras que el ARS y el FP presentan un tiempo de ejecución comprendido entre el tiempo de ejcución del EKF y del método de gradiente. Como es lógico cuando se emplea más partículas

el tiempo real de ejecución del filtro de partículas se aumenta considerablemente.

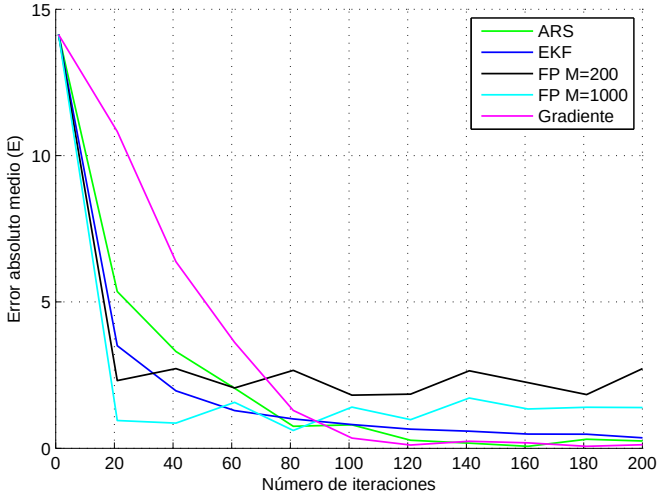
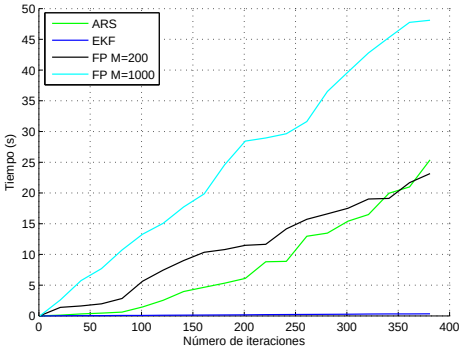
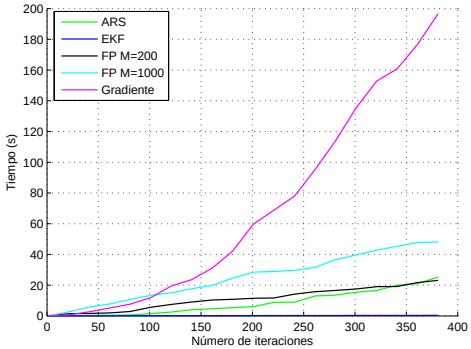


Figura 4.18: Comparación del error absoluto medio cometido al estimar la posición de un blanco con un sistema compuesto por $K = 3$ sensores.



(a) Tiempo de ejecución de: FP, EKF y ARS



(b) Tiempo de ejecución de: FP, EKF, ARS y Gradiente

Figura 4.19: Comparación del tiempo de ejecución al estimar la posición de un blanco con un sistema compuesto por $K = 3$ sensores.

4.3. Conclusiones

En este capítulo se han presentado los algoritmos que se estudian en este trabajo aplicados al problema de posicionamiento estático. En primer lugar se ha presentado el filtro de partículas, a continuación se ha discutido el filtro de Kalman extendido; finalmente se han presentado el método de gradiente y el algoritmo de búsqueda aleatorio acelerada. El primero es un método de búsqueda local mientras el segundo es un método de búsqueda global. Finalmente se ha discutido el muestreo de Gibbs.

La segunda parte del capítulo está dedicada a la comparación del rendimiento de los distintos métodos. Para presentar los resultados se han estudiado varios escenarios, siendo el más sencillo el que está compuesto de tres sensores (es el número mínimo para realizar una estimación sin ambiguïdad). En el caso de que se disponga de tres sensores sólo existe un punto en el plano en el que puede estar situado el objetivo, la intersección de las tres circunferencias determinadas por los tres sensores. Esto no ocurre cuando se emplea un sólo sensor o dos sensores. En el primer caso la zona donde puede estar el objetivo es una circunferencia, mientras que en el segundo caso el objetivo puede estar en los dos puntos en los que cortan las circunferencias determinadas por cada sensor individualmente.

Para todos los algoritmos se han presentado gráficas donde se muestran ejemplos de simulación. Todos los algoritmos convergen a la solución deseada. Cabe destacar que la convergencia de cada algoritmo depende de algunos parámetros. Para el filtro de partículas la convergencia depende del número de partículas. Cuantas más partículas se emplean la convergencia es mejor y, por lo tanto, la estimación se mejora. Pero al emplear más partículas el coste computacional crece considerablemente y el tiempo de ejecución crece también. El EKF y el ARS dependen del número de iteraciones, cuantas más iteraciones se emplean mejor es la estimación. El método de gradiente también depende del número de iteraciones, especialmente por la posibilidad de que converja a un mínimo local. Como se ha explicado en la Sección 4.2.5, con el muetreo de Gibbs no se ha podido realizar estimaciones aceptables.

Para comparar el rendimiento de los diferentes algoritmos. se ha comparado

el error absoluto medio. El EKF y el FP convergen rápidamente. El ARS y el método de gradiente convergen lentamente. El FP con pocas partículas presenta un error mayor pero cuando se aumenta el número de partículas este error decrece considerablemente pero con un tiempo de ejecución mayor. Lo que se refiere a tiempo de ejecución, EKF presenta el tiempo de ejecución más pequeño de todos. El método de gradiente tiene un tiempo ejecución más grande.

Capítulo 5

Posicionamiento de múltiples objetivos

En este capítulo se presentan los diferentes algoritmos que se han estudiado para estimar la posición de múltiples objetivos. Para ello, se adaptan los algoritmos vistos en el Capítulo 4 y se considera el modelo de señal descrito en la Sección 2.2 con múltiples blancos. En primer lugar, se presentan los algoritmos del Capítulo 4 modificados para la localización de múltiples objetivos. En segundo lugar, se procede a estudiar el rendimiento de cada uno de los algoritmos estudiados.

5.1. Algoritmos

5.1.1. Filtro de partículas

El algoritmo es el mismo que el descrito en la Sección 4.1.1, con la salvedad de que ahora tenemos un sistema compuesto por múltiples objetivos $\mathbf{x}_{1,t}, \ldots, \mathbf{x}_{N_b,t}$, donde N_b es el número de objetivos y cada blanco está ubicado en la posición $\mathbf{x}_{i,t}$ en el instante t, que tiene la forma $\mathbf{x}_{i,t} = [x_{i,1,t}, x_{i,2,t}]^T$. La ecuación de estado

tiene la forma

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{u}_t, \quad \mathbf{u}_t \sim N(0, \sigma_{u_t}^2). \tag{5.1}$$

El modelo de observaciones, descrito en la Sección 2.2, es

$$y_{j,t} = 10 \log_{10} \left(\sum_{i=1}^{N_b} \frac{P_0}{\|\mathbf{x}_{i,t} - \mathbf{b}_j\|^\gamma} \right) + v_{j,t}, \quad v_{j,t} \sim N(0, \sigma_{v_t}^2), \tag{5.2}$$

donde $y_{j,t}$ es la observación recogida en el sensor j-ésimo, ubicado en la posición $\mathbf{b}_j$ en el instante t. Hay en total K sensores, de modo que $j = 1, \ldots, K$.

Suponemos que $\mathbf{x}_t$ contiene la posición de todos los objetivos, es decir que $\mathbf{x}_t = [\mathbf{x}_{1,t}^T, \ldots, \mathbf{x}_{N_b,t}^T]^T$. Cada partícula tiene la misma forma, $\mathbf{x}_t^{(i)} = [\mathbf{x}_{1,t}^{(i)T}, \ldots, \mathbf{x}_{N_b,t}^{(i)T}]^T$. Se puede escribir la verosimilitud de cada partícula, $p(\mathbf{y}_t|\mathbf{x}_t^{(i)})$, como

$$p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) = \prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}_t^{(i)}), \tag{5.3}$$

y la densidad de probabilidad condicional $p(y_{j,t}|\mathbf{x}_t^{(i)})$ tiene la forma

$$p(y_{j,t}|\mathbf{x}_t^{(i)}) = \frac{1}{\sqrt{2\pi\sigma_v^2}} e^{-\frac{1}{\sigma_v^2}(y_{j,t} - m_{j,t}^{(i)})^2}, \tag{5.4}$$

donde $\{m_{j,t}^{(i)}, j = 1, \ldots, K\}$, es la potencia media recibida en la posición $\mathbf{b}_j$ supuesto que $\mathbf{x}_t = \mathbf{x}_t^{(i)}$, i.e.,

$$m_{j,t}^{(i)} = 10 \log_{10} \left(\sum_{l=1}^{N_b} \frac{P_0}{\|\mathbf{x}_{l,t}^{(i)} - \mathbf{b}_j\|^\gamma} \right). \tag{5.5}$$

Sustituyendo (5.4) y (5.5) en (5.3) se obtiene la verosimilitud de cada partícula,

$$p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) = \prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}_t^{(i)}) = \frac{1}{(\sqrt{2\pi\sigma_v^2})^K} e^{-\frac{1}{\sigma_v^2}\sum_{j=1}^{K}(y_{j,t} - m_{j,t}^{(i)})^2}. \tag{5.6}$$

De forma análoga al problema con un único objetivo, el peso sin normalizar de la partícula $\mathbf{x}_t^{(i)}$ es

$$\tilde{\omega}_t^{(i)} = \tilde{\omega}_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}), \tag{5.7}$$

que se normaliza como

$$\omega_t^{(i)} = \frac{\tilde{\omega}_t^{(i)}}{\sum_{k=1}^{M} \tilde{\omega}_t^{(k)}} \tag{5.8}$$

Como se ha podido observar en las ecuaciones anteriores, el procedimiento es similar al caso de un único blanco. En el siguiente paso se requiere evaluar la necesidad de hacer un remuestreo. Para ello, se calcula el valor de $\hat{M}_{eff}$, como indica la Ecuación (3.19). Se realiza el remuestreo sólo cuando el valor de $\hat{M}_{eff}$ es inferior a un umbral M_{umb} (se ha empleado $M_{umb} = 2M/3$). Finalmente se calcula el estimador (MMSE),

$$\hat{\mathbf{x}}_t = \sum_{i=1}^{M} \omega_t^{(i)} \mathbf{x}_t^{(i)}. \tag{5.9}$$

conjuntamente para todos los blancos. En la Tabla 5.1 se presenta un resumen del algoritmo.

5.1.2. Filtro de Kalman extendido

El filtro de Kalman extendido es una variante del filtro de Kalman que está pensada para casos en que las ecuaciones de estado y/o observaciones son no lineales. De hecho, este algoritmo se basa en aplicar el filtro de Kalman al modelo lineal resultante de una linealización local de las funciones f y h de la Ecuación (4.18) utilizando el desarollo de Taylor. La descripción del algoritmo está detallada en la Sección 4.1.2.

Inicialización:

1. Muestrear $\mathbf{x}_o^{(i)} \sim p(\mathbf{x}_0), \quad i = 1, \ldots, M.$
2. Fijar $\omega_0^{(i)} = \frac{1}{M} \quad \text{para} \;\; i = 1, \ldots, M.$

Para cada t:

1. Muestrear $\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t | \mathbf{x}_{t-1}^{(i)})$.
2. Actualizar los pesos: $\tilde{\omega}_t^{(i)} = \tilde{\omega}_{t-1}^{(i)} \frac{1}{(\sqrt{2\pi\sigma_v^2})^K} e^{-\frac{1}{\sigma_v^2} \sum_{j=1}^{K} (y_{j,t} - m_{j,t}^{(i)})^2}$
3. Normalizar los pesos: $\omega_t^{(i)} = \frac{\tilde{\omega}_t^{(i)}}{\sum_{k=1}^{M} \tilde{\omega}_{t-1}^{(k)}}$.
4. Estimación: $\hat{\mathbf{x}}_t = \sum_{i=1}^{M} \omega_t^{(i)} \mathbf{x}_t^{(i)}$.
5. Calcular el tamaño efectivo de la muestra $M_{eff} = \frac{1}{\sum_{i=1}^{M} (\omega_t^{(i)})^2}$.

   a) Si $M_{eff} < M_{umb}$ se remuestrea:

   - Muestrear M veces el índice $j \in \{1, \ldots, M\}$ de acuerdo a la distribución discreta $Prob(j = l) = \omega_t^{(i)}$ siendo $l = 1, \ldots, M$; generando $\{j^{(i)}\}_{i=1}^{M}$.
   - Asignar $\mathbf{x}_{0:t}^{(i)} = \mathbf{x}_{0:t}^{j^{(i)}}$.
   - Asignar $\omega_t^{(i)} = 1/M$.

6. $t = t + 1$, ir al paso 2.

Tabla 5.1: Pseudocódigo del filtro de partículas para el caso de múltiples objetivos.

Se emplea el mismo modelo que se ha empleado en el filtro de partículas. La ecuación de estado y de observaciones tienen la forma

$$\begin{aligned} \mathbf{x}_t &= \mathbf{x}_{t-1} + \mathbf{u}_t, \quad \mathbf{u}_t \sim N(0, \sigma_{u_t}^2) && (5.10) \\ y_{j,t} &= h(\mathbf{x}_t) + v_{j,t}, \quad v_{j,t} \sim N(0, \sigma_{v_t}^2), && (5.11) \end{aligned}$$

donde

$$h(\mathbf{x}_t) = 10 \log_{10} \left(\sum_{i=1}^{N_b} \frac{P_0}{\|\mathbf{x}_{i,t} - \mathbf{b}_j\|^\gamma} \right). \quad (5.12)$$

Las ecuaciones de predicción y de actualización son similares al caso de un único objetivo,

$$\begin{aligned} \bar{\mathbf{x}}_{t|t-1} &= \bar{\mathbf{x}}_{t-1} + \mathbf{u}_t, && (5.13) \\ \bar{\mathbf{P}}_t &= \mathbf{A}_t \mathbf{P}_{t-1} \mathbf{A}_t^T + \mathbf{Q}_t, && (5.14) \end{aligned}$$

salvo que la dimensión de los vectores y matrices se ha cambiado al ser $\mathbf{x}_t$ de dimensión $2N_b \times 1$. Las ecuaciones de actualización tienen la forma

$$\begin{aligned} \mathbf{K}_t &= \bar{\mathbf{P}}_t \mathbf{H}_t^T (\mathbf{H}_t \bar{\mathbf{P}}_t \mathbf{H}_t^T + \mathbf{R}_t)^{-1}, && (5.15) \\ \bar{\mathbf{x}}_t &= \bar{\mathbf{x}}_{t|t-1} + \mathbf{K}_t (\mathbf{y}_t - h(\bar{\mathbf{x}}_{t|t-1})), && (5.16) \\ \mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \bar{\mathbf{P}}_t, && (5.17) \end{aligned}$$

donde la matriz $\mathbf{H}_t$ es la jacobiana de la función de observación respecto del estado del proceso y se calcula como

$$\mathbf{H}_t = \begin{bmatrix} \frac{\partial h_1}{\partial x_{1,1,t}} & \frac{\partial h_1}{\partial x_{1,2,t}} & \cdots & \frac{\partial h_1}{\partial x_{N_b,1,t}} & \frac{\partial h_1}{\partial x_{N_b,2,t}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{\partial h_K}{\partial x_{1,1,t}} & \frac{\partial h_K}{\partial x_{1,2,t}} & \cdots & \frac{\partial h_K}{\partial x_{N_b,1,t}} & \frac{\partial h_K}{\partial x_{N_b,2,t}} \end{bmatrix}_{K \times 2N_b}. \quad (5.18)$$

5.1.3. Algoritmo de gradiente

Pretendemos estimar la posición de N_b objetivos estáticos empleando una serie de observaciones de la forma

$$y_{j,t} = 10\log_{10}\left(\sum_{i=1}^{N_b}\frac{P_0}{\|\mathbf{x}_{i,t}-\mathbf{b}_j\|^\gamma}\right)+v_j,\quad v_j\sim N(0,\sigma_{v_t}^2), \tag{5.19}$$

donde $y_{j,t}$ es la observación recogida en el sensor j-ésimo, ubicado en la posición $\mathbf{b}_j$, en el instante t. En el mismo instante tenemos la observación de todos los sensores $\mathbf{y}_t = \{y_{1,t},\ldots,y_{K,t}\}$. En un periodo τ tenemos todas las observaciones recogidas en ese periodo $\mathbf{y} = \left[\mathbf{y}_1^T,\ldots,\mathbf{y}_\tau^T\right]^T$. Entonces para estimar la posición de los objetivos calculamos numéricamente el estimador ML de $\mathbf{x}$,

$$\hat{\mathbf{x}}_{ML} = \arg\max_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}) \tag{5.20}$$

$$= \arg\max_{\mathbf{x}} \prod_{t=1}^{\tau}\prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}) \tag{5.21}$$

con

$$p(y_{j,t}|\mathbf{x}) = N\left(y_{j,t}|10\log_{10}\left(\frac{P_0}{\|\mathbf{x}-\mathbf{b}_j\|^\gamma}\right),v_j^2\right). \tag{5.22}$$

Aplicamos el método de búsqueda de gradiente,

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mu\nabla_{\mathbf{x}}p(\mathbf{x}|\mathbf{y}), \tag{5.23}$$

donde $\nabla_{\mathbf{x}}p(\mathbf{x}|\mathbf{y}) = [\nabla_{\mathbf{x}_1}p(\mathbf{x}|\mathbf{y})^T,\ldots,\nabla_{\mathbf{x}_{N_b}}p(\mathbf{x}|\mathbf{y})^T]^T$.
Puesto que podemos escribir la función de verosimilitud como

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{(2\pi\sigma_v^2)^{\tau K/2}}\exp\left[-\frac{1}{2\sigma_v}f_b(\mathbf{x})\right] \tag{5.24}$$

donde

$$f_b(\mathbf{x}) = \sum_{t=1}^{\tau}\sum_{j=1}^{K}\left(y_{j,t}-10\log_{10}\left(\sum_{i=1}^{N_b}\frac{P_0}{\|\mathbf{x}_i-\mathbf{b}_j\|^\gamma}\right)\right)^2 \tag{5.25}$$

es una función de residuos cuadráticos, entonces el cálculo del estimador (ML) se reduce a resolver el problema

$$\hat{\mathbf{x}}_{ML} = \arg \min_{\mathbf{x}} f_b(\mathbf{x}). \tag{5.26}$$

Para ello aplicamos de nuevo una búsqueda por gradiente,

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \mu \nabla f_b(\mathbf{x})|_{\mathbf{x}=\mathbf{x}^n}, \tag{5.27}$$

donde

$$\nabla f_b(\mathbf{x}) = \left[\frac{\partial f_b(\mathbf{x})}{\partial x_{1,1}}, \frac{\partial f_b(\mathbf{x})}{\partial x_{1,2}}, \ldots, \frac{\partial f_b(\mathbf{x})}{\partial x_{N_b,1}}, \frac{\partial f_b(\mathbf{x})}{\partial x_{N_b,2}} \right]^T. \tag{5.28}$$

Ahora calculamos explícitamente el gradiente de la función $f_b(\mathbf{x})$. Sea $d_i = \|\mathbf{x}_i - \mathbf{b}_j\|$. Aplicando la regla de la cadena obtenemos

$$\frac{\partial f_b(\mathbf{x})}{\partial x_{i,l}} = \frac{\partial f_b(\mathbf{x})}{\partial d_i} \frac{\partial d_i}{\partial x_{i,l}} \quad con \quad l = 1, 2. \tag{5.29}$$

Es inmediato calcular

$$\frac{\partial d_i}{\partial x_{i,l}} = \frac{(x_{i,l} - b_{j,l})}{\|\mathbf{x}_i - \mathbf{b}_j\|} \tag{5.30}$$

y

$$\frac{\partial f_b(\mathbf{x})}{\partial d_i} = 2 \sum_{t=1}^{\tau} \sum_{j=1}^{K} \left(y_j - 10 \log_{10} \left[\sum_{i=1}^{N_b} \frac{P_0}{d_i^\gamma} \right] \right) P, \tag{5.31}$$

donde

$$\begin{aligned} P &= \frac{\partial}{\partial d_i} \left(y_{j,t} - 10 \log_{10} \left[\sum_{i=1}^{N_b} \frac{P_0}{d_i^\gamma} \right] \right) \\ &= \frac{10\gamma P_0}{d_i^{\gamma+1} \log_e 10} \left(\sum_{i=1}^{N_b} \frac{P_0}{d_i^\gamma} \right)^{-1}. \end{aligned} \tag{5.32}$$

Sustituyendo (5.30) y (5.31) en (5.29) obtenemos

$$\frac{f_b(\mathbf{x})}{\partial x_{i,l}} = 2\sum_{t=1}^{\tau}\sum_{j=1}^{K}\left[\left(y_j - 10\log_{10}\left[\sum_{i=1}^{N_b}\frac{P_0}{d_i^{\gamma}}\right]\right)\left(\frac{(x_{i,l}-b_{j,l})}{\|\mathbf{x}_i-\mathbf{b}_j\|}\right)P\right], l=1,2, \quad (5.33)$$

de manera que tenemos todas las derivadas parciales necesarias para calcular el gradiente de la función $f_b(\mathbf{x})$. Entonces para estimar la posición sólo hace falta aplicar el algoritmo que se resume en la Tabla 5.2.

Incialización

1. Seleccionar un punto inicial $\mathbf{x}^0 \in \mathbb{R}^{2N_b}$, donde N_b es el número de objetivos. Fijar $k = 0$ y $\mu << 1$.

Iteración

1. Calcular el gradiente $d^k = -\nabla f_b(\mathbf{x})$.
2. Actualizar la posición: $\mathbf{x}^k = \mathbf{x}^{k-1} + \mu d^k$.
3. $k = k + 1$ y volver a 1.

Tabla 5.2: Algoritmo de gradiente para la estimación de múltiples objetivos.

5.1.4. Búsqueda aleatoria acelerada

Como está descrito en la Sección 4.1.4, la búsqueda aleatoria acelerada (ARS) es un algoritmo iterativo de optimización global [7].

El procedimiento para el caso de múltiples objetivos es similar al de un único objetivo, con la dificultad de que el espacio de búsqueda es mucho mayor. El problema a resolver es el mismo que en la Sección 4.1.4, i.e., el cálculo del estimador de máxima verosimilitud

$$\hat{\mathbf{x}}_{ML} = \arg \min_{\mathbf{x}} f_b(\mathbf{x}) \tag{5.34}$$

donde $f_b(\mathbf{x})$ es la función de residuos cuadráticos de la Ecuación (5.25). Los parámetros del algoritmo son R_{max} si este parámetro es grande el espacio de la búsqueda es más amplio, R_{min} determina la precisión es decir cuanto más pequeño la precisión es mejor y c es el parámetro de contracción la velocidad de búsqueda depende mucho de este parámetro cuanto más grande sea, la velocidad es menor. Para este caso el conjunto de búsqueda es mas grande y es proporcional al número de blancos N_b. El algoritmo ARS para múltiples blancos se resume en la Tabla 5.3

5.1.5. Muestreo de Gibbs

La descripción general del muestreo de Gibbs está presentada en la Sección 3.1.5. Igual que en el caso de un único objetivo para muestrear utilizamos el método de aceptación/rechazo que está descrito en la Sección 4.1.5.

Pretendemos estimar la posición $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_{N_b}]^T$ de N_b objetivos estáticos, donde $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$ es la posición del objetivo i-ésimo. Para ello, generamos una secuencia de muestras $x_{1,1}^{(j)}$,$x_{1,2}^{(j)}$,..., $x_{N_b,1}^{(j)}$,$x_{N_b,2}^{(j)}$ empleando Gibbs [2]. Después de generar N muestras se calcula la media de todas ellas rechazando n muestras de las etapas iniciales. Las muestras se generan a partir de las densidades de probabilidad condicionales.

Se generan los componentes de cada muestra por separado. Para generar la muestra $\mathbf{x}^{(j)}$, se generan los componentes de la misma $\mathbf{x}_1^{(j)}$,...,$\mathbf{x}_{N_b}^{(j)}$. En primer lugar se procede a generar los componentes de la muestra $\mathbf{x}_1^{(j)}$. Para ello, se genera primero $x_{1,1}^{(j)} \sim p(x_{1,1}|x_{1,2}^{(j-1)}, x_{2,1}^{(j-1)}, \ldots, x_{N_b,1}^{(j-1)}, x_{N_b,2}^{(j-1)}, \mathbf{y})$ y luego $x_{1,2}^{(j)} \sim p(x_{1,2}|x_{1,1}^{(j)}, x_{2,1}^{(j-1)}, \ldots, x_{N_b,1}^{(j-1)}, x_{N_b,2}^{(j-1)}, \mathbf{y})$ y así, sucesivamente hasta llegar al componente N_b del vector $\mathbf{x}^{(j)}$ que se genera a su vez de la misma forma, se genera primero $x_{N_b,1}^{(j)} \sim p(x_{N_b,1}|\mathbf{x}_1^{(j)}, \ldots, x_{N_b-1,1}^{(j)}, x_{N_b,2}^{(j-1)}, \mathbf{y})$ y luego $x_{N_b,2}^{(j)} \sim p(x_{N_b,2}|\mathbf{x}_1^{(j)}, \ldots, x_{N_b,1}^{(j)}, \mathbf{y})$. Esto se hace para todo $j = 1, \ldots, M$. En la Tabla 5.4 se resume el muestreo de Gibbs para el caso de múltiples objetivos.

- Parámetros del algoritmo: Radio máximo R_{max}, radio mínimo R_{min} y factor de contracción $c > 1$.
- Algoritmo

1. Inicialización
   - Para cada objetivo se elige un conjunto inicial $[a_{i,1}(0), b_{i,1}(0)] \times [a_{i,2}(0), b_{i,2}(0)]$, $i = 1, \ldots, N_b$
   - Se generan los componentes del punto inicial de todos los objetivos $\mathbf{x}_i^{(0)}$ a partir de distribuciones uniformes, $x_{i,1}^{(0)} \sim U[a_{i,1}(0), b_{i,1}(0)]$ y $x_{i,2}^{(0)} \sim U[a_{i,2}(0), b_{i,2}(0)]$.
   - Se calcula el coste $f^{(0)} = f(\mathbf{x}^{(0)})$, con $\mathbf{x}^{(0)} = [\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_{N_b}^{(0)}]^T$.
   - Se elige el radio inicial $r_i^{(0)} = R_{max}$.
2. Iteración en $n + 1$
   - Para cada objetivo se genera un punto $\widetilde{\mathbf{x}}_i = [\widetilde{x}_{i,1}, \widetilde{x}_{i,2}]^T$ a partir de una distribución uniforme, $\widetilde{x}_{i,1} \sim U[x_{i,1}^{(n)} - r_i^{(n)}, x_{i,1}^{(n)} + r_i^{(n)}]$ y $\widetilde{x}_{i,2} \sim U[x_{i,2}^{(n)} - r_i^{(n)}, x_{i,2}^{(n)} + r_i^{(n)}]$.
   - Se calcula el coste de $\widetilde{\mathbf{x}}$, $\widetilde{f}_b = f_b(\widetilde{\mathbf{x}})$.
     - Si $\widetilde{f}_b < f_b^{(n)}$ entonces: $\mathbf{x}^{(n+1)} = \widetilde{\mathbf{x}}$, $f_b^{(n+1)} = \widetilde{f}_b$ y $r_i^{(n+1)} = R_{max}$.
     - Si $\widetilde{f} \geq f_b^{(n)}$ entonces: $\mathbf{x}^{(n+1)} = \mathbf{x}^n$, $f_b^{(n+1)} = f_b^{(n)}$,$r_i^{(n+1)} = \frac{r_i^{(n)}}{c}$.
     - Si $r_i^{(n+1)} < R_{min}$ entonces $r_i^{(n+1)} = R_{max}$.

Tabla 5.3: Algoritmo de búsqueda aleatoria acelerada (ARS) para el caso de múltiples objetivos.

Como en el caso de un único objetivo para generar las muestras se emplea el método de aceptación/rechazo, ya que no es posible muestrear directamente de la densidad de probabilidad $p(x_{1,1}|x_{1,2}^{(j-1)}, x_{2,1}^{(j-1)}, \ldots, x_{N_b,1}^{(j-1)}, x_{N_b,2}^{(j-1)}, \mathbf{y})$.
Alpicando el teorema de Bayes tenemos

1. Seleccionar un punto inicial $\mathbf{x}^{(0)} = [\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_{N_b}^{(0)}]^T$, con $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$
2. Para $j = 1, \ldots, M$
   - $x_{1,1}^{(j)} \sim p(x_{1,1}|x_{1,2}^{(j-1)}, \mathbf{x}_2^{(j-1)}, \ldots, \mathbf{x}_{N_b}^{(j-1)}, \mathbf{y})$
   - $x_{1,2}^{(j)} \sim p(x_{1,2}|x_{1,1}^{(j)}, \mathbf{x}_2^{(j-1)}, \ldots, \mathbf{x}_{N_b}^{(j-1)}, \mathbf{y})$
   - $\ldots$
   - $x_{N_b,1}^{(j)} \sim p(x_{N_b,1}|\mathbf{x}_1^{(j)}, \ldots, \mathbf{x}_{N_b-1,1}^{(j)}, x_{N_b,2}^{(j-1)}, \mathbf{y})$
   - $x_{N_b,2}^{(j)} \sim p(x_{N_b,2}|\mathbf{x}_1^{(j)}, \ldots, \mathbf{x}_{N_b-1,1}^{(j)}, x_{N_b,1}^{(j)}, \mathbf{y})$
3. Para la estimación se calcula la media descartando las n muestras de las etapas iniciales,

$$\hat{\mathbf{x}}_{ML} = \frac{1}{N-n}\sum_{j=n}^{N}\mathbf{x}^{(j)}$$

Tabla 5.4: Muestreo de Gibbs para el caso de múltiples objetivos.

$$p(x_{1,1}|x_{1,2}^{(j-1)}, \mathbf{x}_2^{(j-1)}, \ldots, \mathbf{x}_{N_b}^{(j-1)}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(x_{1,1})}{p(\mathbf{y}|x_{1,2}^{(j-1)}, \mathbf{x}_2^{(j-1)}, \ldots, \mathbf{x}_{N_b}^{(j-1)})}, \tag{5.35}$$

la densidad de probabilidad $p(\mathbf{y}|x_{1,2}^{(j-1)}, \mathbf{x}_2^{(j-1)}, \ldots, \mathbf{x}_M^{(j-1)})$ no depende de $x_{1,1}$, con lo cual,

$$p(x_{1,1}|x_{1,2}^{(j-1)}, \mathbf{x}_2^{(j-1)}, \ldots, \mathbf{x}_M^{(j-1)}, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{x})p(x_{1,1}), \tag{5.36}$$

con esto podemos aplicar el método de aceptación/rechazo, poniendo como función de muestreo $r(x) = p(\mathbf{y}|\mathbf{x})p(x_{1,1})$. Ahora si ponemos como función tentativa $g(x) = p(x_{1,1})$, la condición de acpetación se reduce a

$$\frac{p(\mathbf{y}|\mathbf{x})p(x_{1,1})}{cp(x_{1,1})} = \frac{p(\mathbf{y}|\mathbf{x})}{c} \geq U, \tag{5.37}$$

donde U es una variable aleatoria uniforme se genera de $U[0,1]$ y c es una cota superior. A continuación se detalla el cálculo de $p(\mathbf{y}|\mathbf{x})$. Tenemos

$$y_{j,t}|\mathbf{x}_i \sim N\left(10\log_{10}\left(\sum_{i=1}^{N_b}\frac{P_0}{\|\mathbf{x}_i-\mathbf{b}_j\|^\gamma}\right), v_j^2\right), \tag{5.38}$$

si ponemos

$$m_j = 10\log_{10}\left(\sum_{i=1}^{N_b}\frac{P_0}{\|\mathbf{x}_i-\mathbf{b}_j\|^\gamma}\right),$$

entonces

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \prod_{t=1}^{\tau}\prod_{j=1}^{K} p(y_{j,t}|\mathbf{x}) \\ &= \prod_{t=1}^{\tau}\prod_{j=1}^{K}\frac{1}{\sqrt{2\pi\sigma_v^2}}e^{-\frac{1}{\sigma_v^2}(y_{j,t}-m_j)^2} \\ &= \frac{1}{\left(\sqrt{2\pi\sigma_v^2}\right)^{\tau K}}e^{-\frac{1}{\sigma_v^2}f_b(\mathbf{x})}. \end{aligned} \tag{5.39}$$

donde $f_b(\mathbf{x})$ es la función de residuos cuadráticos (véase la Ecuación (5.25)) ahora sólo falta escoger la constante c. Para simplificar los cálculos se elige como $c = \frac{1}{\left(\sqrt{2\pi\sigma_v^2}\right)^{\tau K}}$. Así, la condición de aceptación se reduce a

$$e^{-\frac{1}{\sigma_v^2}f_b(\mathbf{x})} \geq U. \tag{5.40}$$

En la Tabla 5.5 se resume el algoritmo.

1. Generar una muestra $x_{h,i}$, $h = 1, 2$, $i = 1, \ldots, N_b$ de la densidad $p(\mathbf{x})$
2. Generar una variable aleatoria U de $U[0, 1]$
3. Si $U \leq e^{-\frac{1}{\sigma_v^2} f_b(\mathbf{x})}$, $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_{N_b}]^T$ elegir la muestra $x_{h,i}$ si no volver a 1

Y así, sucesivamente hasta generar todas las muestras.

Tabla 5.5: Método de aceptación/rechazo para el caso de múltiples objetivos.

5.2. Resultados

En está sección se presentan los resultados de simulación de los algoritmos estudiados en este capítulo. Para ello, se han simulado casos prácticos mediante MATLAB implementando los diferentes algoritmos. Se presentan ejemplos de simulación donde se estima la posición de un número fijo de objetivos. Estos objetivos están situados en un recinto donde se ha posicionado un número determinado de sensores en posiciones conocidas a *priori*.

Los sensores son los responsables de recibir la señal emitida por cada objetivo y enviar está información al centro de fusión de datos. Este último es el encargado de recibir las observaciones de todos los sensores y aplicar un algoritmo para estimar la posición de cada objetivo.

A la hora de evaluar los algoritmos se hace uso del error absoluto cometido en la estimación. Este error nos permite evaluar de forma muy intuitiva el comportamiento de los algoritmos. Si $\mathbf{x}(i) = \{\mathbf{x}_1(i), \ldots, \mathbf{x}_{N_b}(i)\}$ es la posición de los objetivos de la i-ésima simulación, con $\mathbf{x}_j(i)$ es la posición del objetivo j en la simulación i-ésima. Y $\hat{\mathbf{x}}(i) = \{\hat{\mathbf{x}}_1(i), \ldots, \hat{\mathbf{x}}_{N_b}(i)\}$ es la estimación de la posición de todos los objetivos, el error absoluto de la estimación es

$$e(i) = \frac{1}{N_b} \sum_{j=1}^{N_b} \|\mathbf{x}_j(i) - \hat{\mathbf{x}}_j(i)\|, \tag{5.41}$$

con N_b es el número de objetivos. Sin embargo, nos interesa ver el comportamiento medio de los algoritmos propuestos. Para ello, se procede a calcular el error absoluto medio. Para un número N_t de simulaciones independientes sobre condiciones semejantes este error se calcula de la siguiente manera:

$$E = \frac{1}{N_t} \sum_{i=1}^{N_t} e(i). \tag{5.42}$$

Para presentar las gráficas de las curvas de nivel, se ha optado por presentar dichas curvas de cada objetivo por separado. Para ello se ha fijado la posición de todos los objetivos en su posición real menos el objetivo en cuestión.

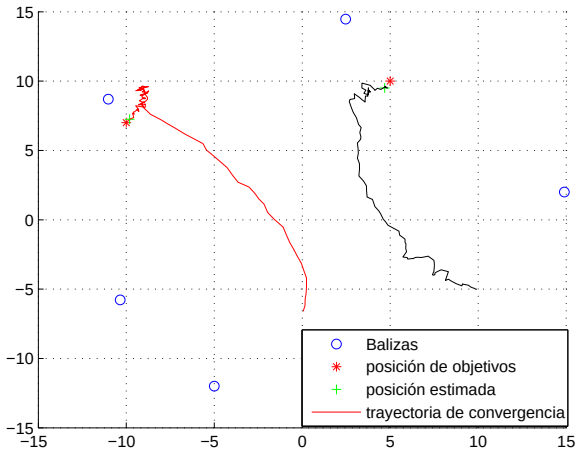
5.2.1. Filtro de partículas

La descripción del algoritmo se presenta en la Sección 5.1.1. Los parámetros utilizados en la implementación del algoritmo se presentan en la Tabla 5.6. Se ha elegido una desviación típica de la ecuación de estado σ_u muy pequeña. Porque en este trabajo se aborda el problema de posicionamiento estático, es decir que, se considera que los objetivos no están en movimiento. Sólo interesa estimar la posición.

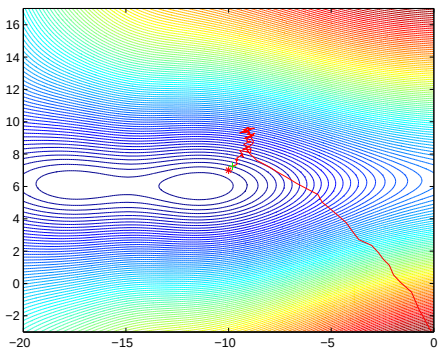
Parámetros	**Valor**
Desviación típica de la ecuación de estado σ_u	3×10^{-6}
Desviación típica de la ecuación de observación σ_v	1
Exponente de propagación γ	2
Potencia de transmisión P_0	10mW

Tabla 5.6: Parámetros de simulación del filtro de partículas para el caso de múltiples objetivos.

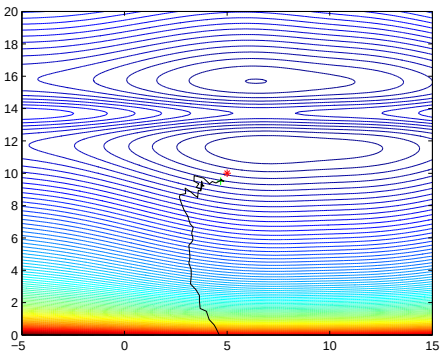
En la Figura 5.1 se presenta un ejemplo de simulación empleando el filtro de partículas. Con un sistema compuesto por $N_b = 2$ objetivos y $K = 5$ sensores. En la Figura 5.1(a), se representa una estimación de la posición de dos objetivos y las trayectorias de convergencia que han llevado a esa estimación. En las Figuras 5.1(b) y 5.1(c), se representa la trayectoria de convergencia y las curvas de nivel de la función de residuos cuadráticos (véase la Ecuación (5.25)) donde se ven claramente las trayectorias de convergencia.



(a) Estimación de la posición de los dos objetivos



(b) Curvas de nivel del primer objetivo



(c) Curvas de nivel del segundo objetivo

Figura 5.1: Estimación de la posición de $N_b = 2$ objetivos empleando el filtro de partículas con $K = 5$ sensores y $M = 1000$ partículas.

Para ver el comportamiento del algoritmo, en especial cuando se aumenta el número de objetivos, se presenta la Tabla 5.7, donde se muestra el error absoluto medio de la estimación y el número de blancos correspondiente. Para ello, se ha empleado un sistema igual que el anterior, es decir, un sistema compuesto por $N_b = 5$ sensores y el filtro de partículas con $M = 1000$ partículas.

Como se puede ver en la Tabla 5.7. El error absoluto de la estimación crece con el aumento del número de objetivos. En el caso de que el número de objetivos sea igual al número de sensores el error se dispara (el número de sensores vuelve a ser insuficiente para monitorizar todos los objetivos con presición).

Para ver el efecto del número de sensores, en la Tabla 5.8 se presenta un ejemplo donde se ha duplicado el número de sensores respecto al ejemplo anterior y con el máximo número de objetivos.

Número de objetivos	**Error absoluto medio**
2	1.2075
3	2.9938
4	6.6567
5	7.4934

Tabla 5.7: Error absoluto medio de la estimación empleando el filtro de partículas con $M = 1000$ partículas y un sistema compuesto por $K = 5$ sensores.

Número de objetivos	**Error absoluto medio**
5	3.7209
6	5.4013
7	5.4284
8	8.2949

Tabla 5.8: Error absoluto medio de la estimación empleando el filtro de Partículas con $M = 1000$ partículas y un sistema compuesto por $K = 10$ sensores.

Como se puede ver en la Tabla 5.8, el error se ha mejorado con el aumento del número de sensores respecto al caso anterior donde se ha empleado $K = 5$

sensores. Esto es lógico, ya que, en este caso tenemos un número suficiente de sensores para cubrir toda la zona lo que implica que haya más observaciones y por lo tanto el algoritmo funciona mejor. A continuación se estudia el rendimiento del algoritmo.

En la Figura 5.2 se presenta una gráfica donde se muestra la evolución del error absoluto medio en función del filtro de partículas. Empleando un sistema compuesto por $K = 5$ sensores y $N_b = 2$ objetivos. Como se puede ver el error decrece con el aumento del número de partículas hasta llegar a una situación estacionaria.

En la Figura 5.3 se presenta la evolución del error de estimacón en función del tiempo real de ejecución del algoritmo. Puesto que los objetivos no están en movimiento, el error decrece con el paso del tiempo.

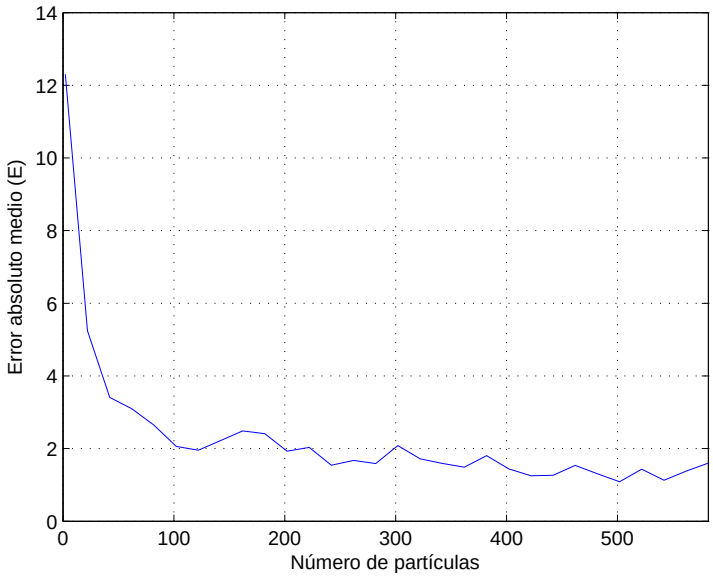


Figura 5.2: Error medio de la estimación empleando filtro de partículas en función del número de partículas. El número de objetivos es $N_b = 2$ y de los sensores es $K = 5$. Se han realizado $N_t = 50$ simulaciones independientes.

5.2.2. Filtro de Kalman extendido

La descripción del algoritmo se presenta en la Sección 5.1.2. En la Tabla 5.9 se presentan los parámetros utilizados en la implementación del filtro de Kalman extendido. Igual que en el caso del filtro de partículas, para la ecuación de estado se ha elegido una desviación típica σ_u muy pequeña, ya que el objetivo es, en realidad, estático.

Parámetros	**Valor**
Desviación típica de la ecuación de estado σ_u	3×10^{-6}
Desviación típica de la ecuación de observación σ_v	1
Exponente de propagación γ	2
Potencia de transmisión P_0	10mW

Tabla 5.9: Parámetros de simulación del filtro de Kalman extendido.

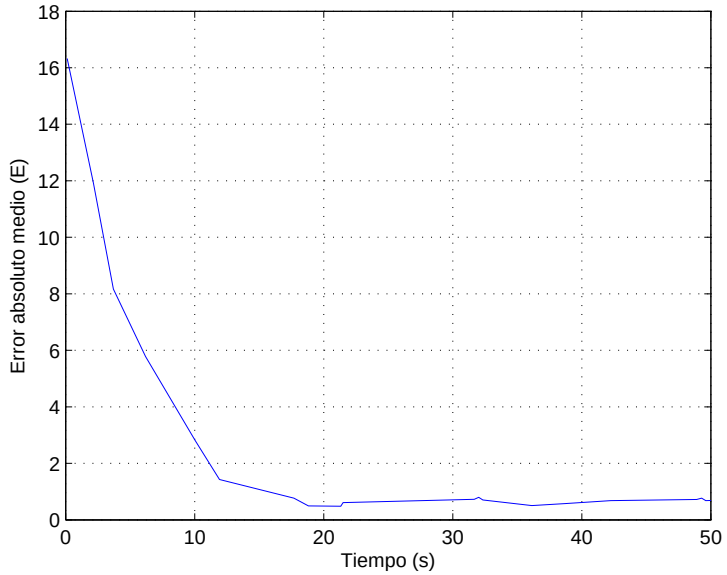
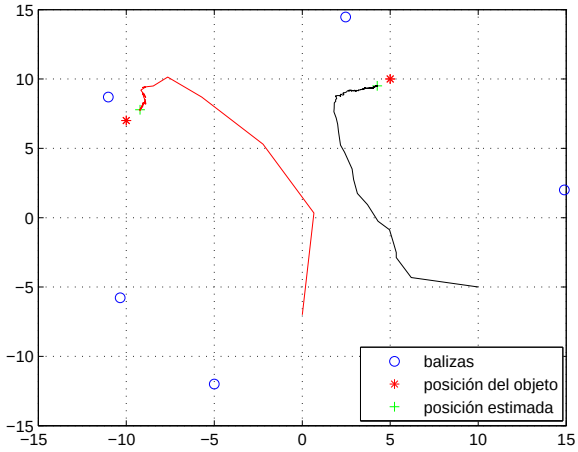
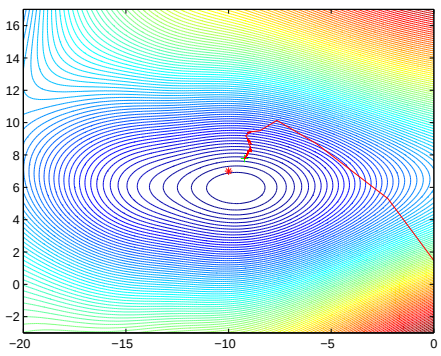


Figura 5.3: Error medio de la estimación empleando filtro de partículas en función del tiempo real de ejecución. El número de objetivos es $N_b = 2$ y de los sensores es $K = 5$. Se han realizado $N_t = 50$ simulaciones independientes y se han empleado $M = 200$ partículas.

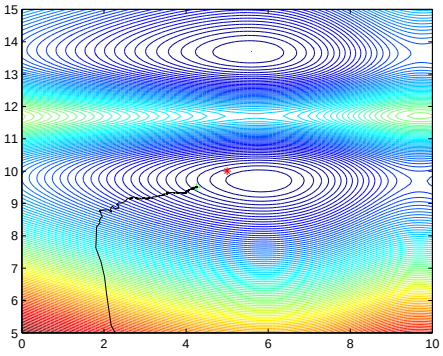
En la Figura 5.4 se presenta un ejemplo de estimación donde se ha empleado el filtro de Kalman extendido para estimar la posición de $N_b = 2$ objetivos con un sistema compuesto por $K = 5$ sensores. En la Figura 5.4(a) se presenta una gráfica donde se aprecia claramente la trayectoria de convergencia del algoritmo y la estimación de la posición de los dos objetivos. En las Figuras 5.4(b) y 5.4(c) se presenta las curvas de nivel de la función de residuos cuadráticos (véase la Ecuación (5.25)).



(a) Estimación de la posición de los dos objetivos



(b) Curvas de nivel del primer objetivo



(c) Curvas de nivel del segundo objetivo

Figura 5.4: Estimación de la posición de $N_b = 2$ objetivos empleando el filtro de Kalman extendido con un sistema compuesto de $K = 5$ sensores.

Para ver el comportamiento del filtro de Kalman extendido cuando se aumenta

el número de objetivos se presenta la Tabla 5.10. Donde se muestra el error medio de la estimación y el número de objetivos empleados para realizar dicha simulación, el número de sensores utilizado es $K = 5$. Como se puede ver en la tabla, si conservamos el número de sensores y aumentamos el número de objetivos, el error absoluto medio aumenta.

Número de objetivos	**Error absoluto medio**
2	0.1878
3	0.2647
4	6.3700
5	7.2484

Tabla 5.10: Error absoluto medio de la estimación empleando el filtro de Kalman extendido y un sistema compuesto por $K = 5$ sensores.

Para estudiar el efecto del aumento del número de sensores en la estimación se presenta una Tabla 5.11 donde se muestra el error absoluto medio de la estimación. En estas simulaciones se ha empleado un sistema compuesto por $K = 10$ sensores.

Número de objetivos	**Error absoluto medio**
5	1.8493
6	3.6368
7	3.6670
8	4.0616

Tabla 5.11: Error absoluto medio de la estimación empleando el filtro de Kalman extendido y un sistema compuesto por $K = 10$ sensores.

Para estudiar el rendimiento del algoritmo, se presenta una Figura 5.5 donde se muestra la evolución del error absoluto medio en función del número de iteraciones del algorimto. Como se ve en la gráfica el algoritmo converge rapidamente a un valor y a partir de un número determinado de iteraciones el error no se decrementa. Para realizar estas simulaciones se ha empleado un sistema compuesto

por $K = 5$ sensores y $N_b = 2$ blancos.

Para estudiar la complejidad del algoritmo, se presenta una Figura 5.6 donde se muestra la evolución del error absoluto medio en función del tiempo.

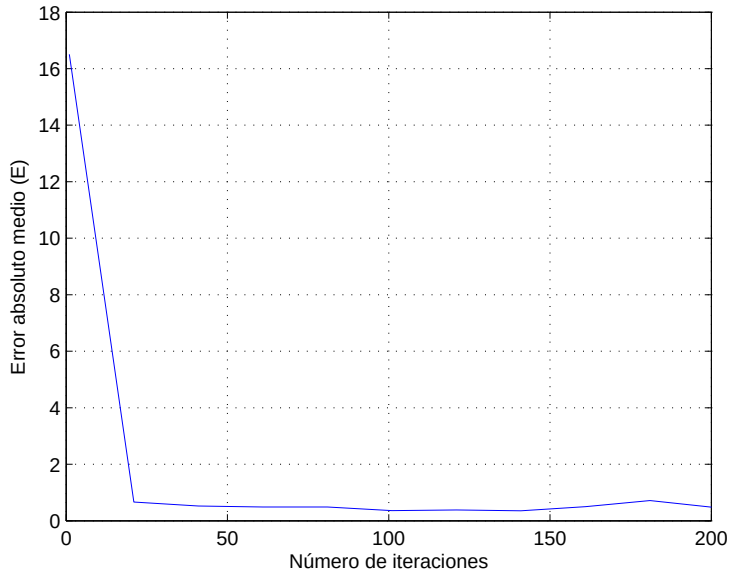


Figura 5.5: Error medio de la estimación empleando filtro de Kalman extendido en función del número de iteraciones. El número de objetivos es $N_b = 2$ y de los sensores es $K = 5$. Se han realizado $N_t = 50$ simulaciones independientes.

5.2.3. Algoritmo de gradiente

La descripción general del algoritmo se presenta en la Sección 5.1.4. Los parámetros utilizados en la implementación del algoritmo se presentan en la Tabla 5.13. A continuación, se procede a presentar los resultados de las simulaciones.

En la Figura 5.7 se presenta un ejemplo de simulación del algoritmo donde se estima la posición de $N_b = 2$ objetivos con un sistema compuesto por $K = 5$ de sensores. En la Figura 5.7(a) se muestra la trayectoria de convergencia de los dos objetivos. En las Figuras 5.7(b) y 5.7(c) se presentan respectivamente la trayectoria de convergencia y las curvas de nivel de la función de residuos cuadráticos (Ecuación (5.25)).

Para ver el comportamiento del algoritmo, en especial cuando se aumenta el número de objetivos, se presenta la Tabla 5.11 donde se muestra el error absoluto medio y el número de objetivos correspondiente. Como se puede observar si mantenemos el número de sensores fijo y aumentamos el número de objetivos, el error crece rápidamente.

Para ver el efecto del aumento de número de sensores, en la Tabla 5.13 se presenta el error absoluto medio con el número de objetivos correspondiente. Como se puede observar el error decrece con el aumento del número de sensores.

En la Figura 5.8 se muestra el error medio absoluto en función del número de iteraciones. Como se ve en la gráfica, el algoritmo converge a la solución deseada. Cuando el algoritmo converge el error no derece más aunque el número de iteraciones aumente.

En la Figura 5.9 se presenta la evolución del error absoluto medio en función del tiempo real de ejecución del algoritmo.

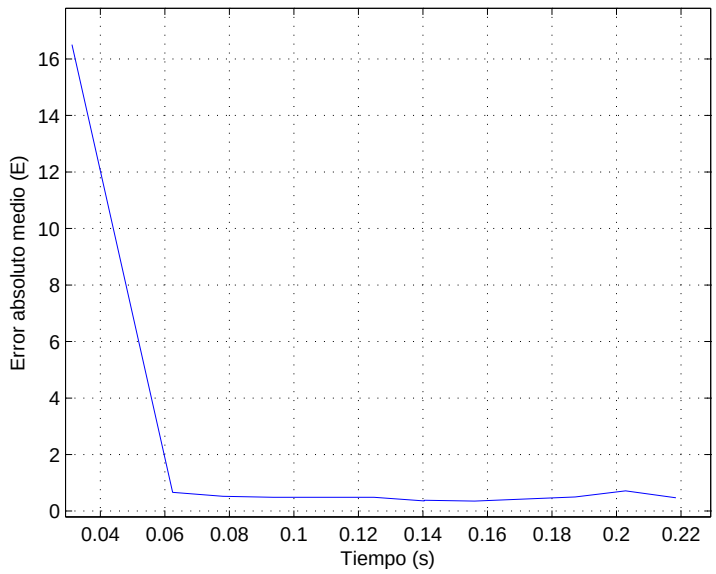


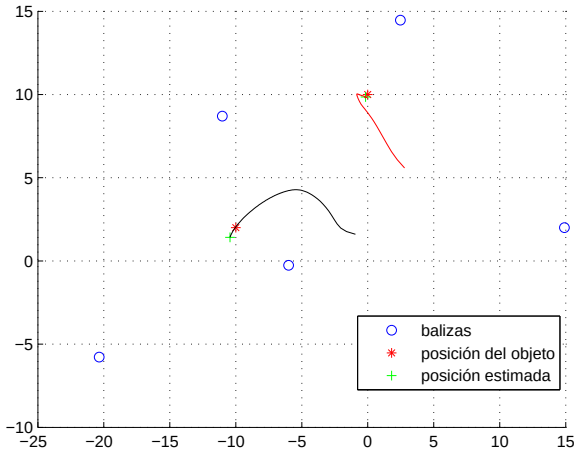
Figura 5.6: Error medio de la estimación empleando filtro de Kalman extendido en función del tiempo de ejecución. El número de objetivos es $N_b = 2$ y de los sensores es $K = 5$. Se han realizado $N_t = 50$ simulaciones independientes.

Número de objetivos	Error absoluto medio
2	0.3215
3	0.3869
4	6.0380
5	7.0640

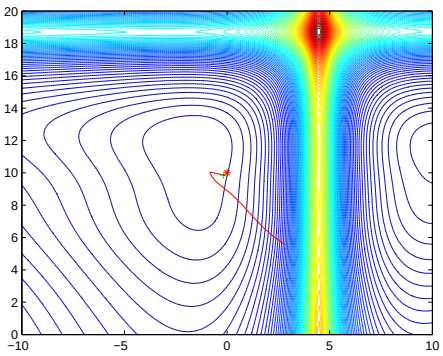
Tabla 5.12: Error absoluto medio de la estimación empleando el algoritmo de gradiente con un sistema compuesto por $K = 5$ sensores.

Número de objetivos	Error absoluto medio
5	2.8717
6	4.3858
7	6.1003
8	6.2508

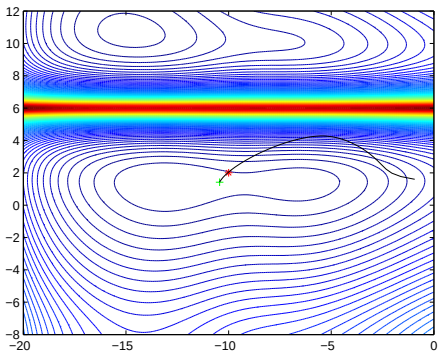
Tabla 5.13: Error absoluto medio de la estimación empleando el algoritmo de gradiente con un sistema compuesto por $K = 10$ sensores.



(a) Estimación de la posición de los dos objetivos



(b) Curvas de nivel del primer objetivo



(c) Curvas de nivel del segundo objetivo

Figura 5.7: Estimación de la posición de $N_b = 2$ objetivos empleando el algoritmo de gradiente con un sistema compuesto por $K = 5$ sensores.

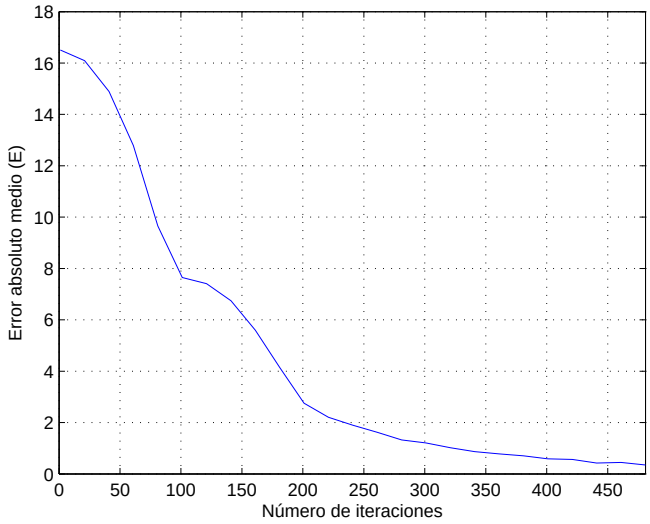


Figura 5.8: Error absoluto medio de la estimación empleando el algoritmo de gradiente en función del número de iteraciones. El número de objetivos es $N_b = 2$ y de los sensores es $K = 5$. Se han realizado $N_t = 50$ simulaciones independientes.

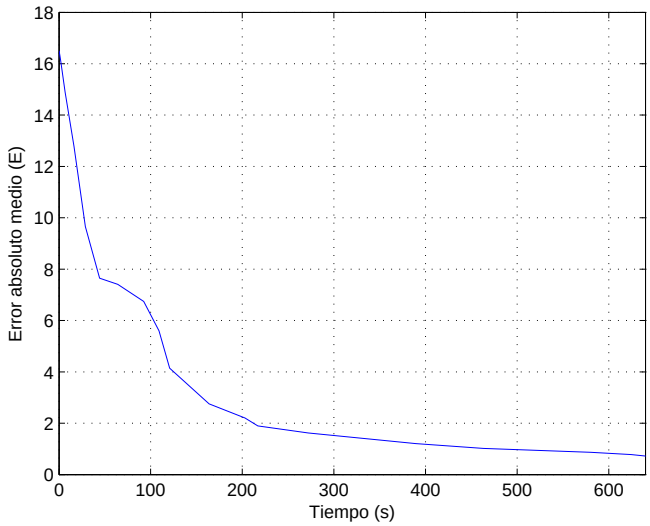


Figura 5.9: Error absoluto medio de la estimación empleando el algoritmo de gradiente en función del tiempo de ejecución. El número de objetivos es $N_b = 2$ y de los sensores es $K = 5$. Se han realizado $N_t = 50$ simulaciones independientes.

5.2.4. Búsqueda aleatoria acelerada

En está sección se procede a presentar los resultados de la simulación del algoritmo de búsqueda aleatoria acelerada. La descripción del algoritmo se presenta en la Sección 5.1.4. Los parámetros utilizados para implementar el algoritmo se presentan en la Tabla 5.14.

Se empieza por presentar las gráficas donde se muestra la convergencia del algoritmo y las curvas de nivel de la función de residuos cuadráticos. Para mostrar la convergencia del algoritmo se ha empleado un sistema compuesto por $K = 5$ sensores y $N_b = 2$ objetivo.

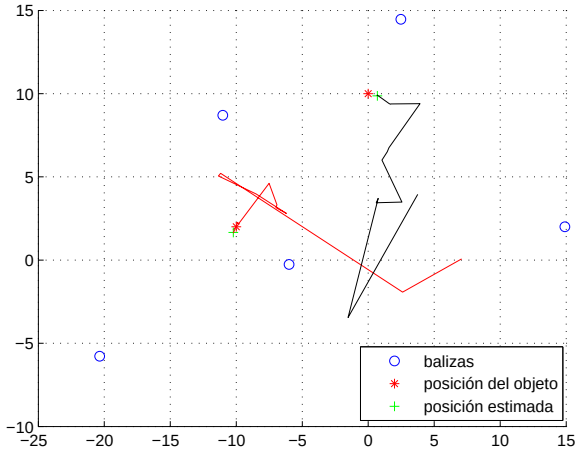
En la Figura 5.10 se presentan tres gráficas. En la primera, Figura 5.10(a), se muestra como el algorimo converge a las posiciones reales. En las Figuras 5.10(b) y 5.10(c) se ve como converge el algoritmo a la posición del primer y del segundo objetivo respectivamente, se ven también las curvas de nivel de la función de residuos cuadráticos.

Parámetros	**Valor**
Desviación típica de la ecuación de observación σ_v	1
Exponente de propagación γ	2
Potencia de transmisión P_0	10mW
Número de sensores K	5

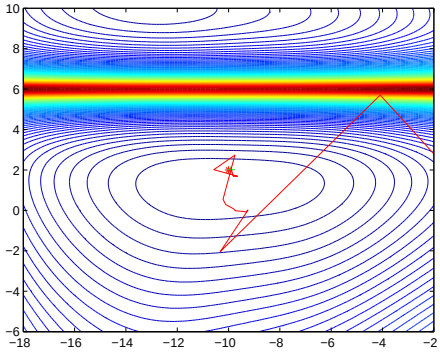
Tabla 5.14: Parámetros de simulación del algoritmo búsqueda aleatoria acelerada.

Como se ha hecho con los algoritmos anteriores, a continuación en la Tabla 5.15 se presenta el comportamiento del algoritmo con el aumento del número de objetivos manteniendo el número de sensores fijo ($K = 5$ sensores). Para ello se muestra el error absoluto medio y el número de objetivos correspondiente.

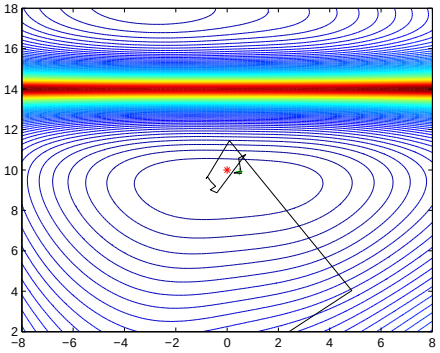
Para ver el comportamiento del algoritmo cuando se aumenta el número de sensores, se presenta la Tabla 5.16, donde se muestra el error absoluto medio y el número de objetivos correspondiente. En este ejemplo se ha empleado el doble del número de sensores que en el ejemplo anterior, es decir, $K = 10$.



(a) Estimación de la posición de los dos objetivos



(b) Curvas de nivel del primer objetivo



(c) Curvas de nivel del segundo objetivo

Figura 5.10: Estimación de la posición de dos objetivos empleando el algoritmo ARS con un sistema compuesto por $K = 5$ sensores y $N_b = 2$ objetivos.

Para estudiar el rendimiento del algoritmo, se presenta la Figura 5.11, donde se muestra la evolución del error absoluto medio en función del número de iteraciones del algoritmo. Como se puede ver en la figura el algoritmo ARS converge a la solución deseada con un error aceptable.

Para estudiar la complejidad del algoritmo, se presenta una Figura 5.12, donde se muestra la evolución del error absoluto medio en función del tiempo real de ejecución del algoritmo.

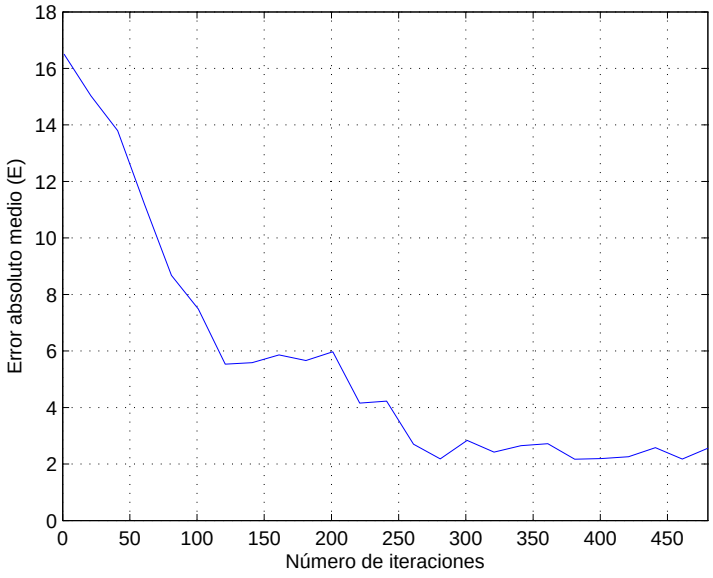


Figura 5.11: Error medio de la estimación empleando el algoritmo ARS en función del número de iteraciones. Se han realizado $N_t = 50$ simulaciones independientes.

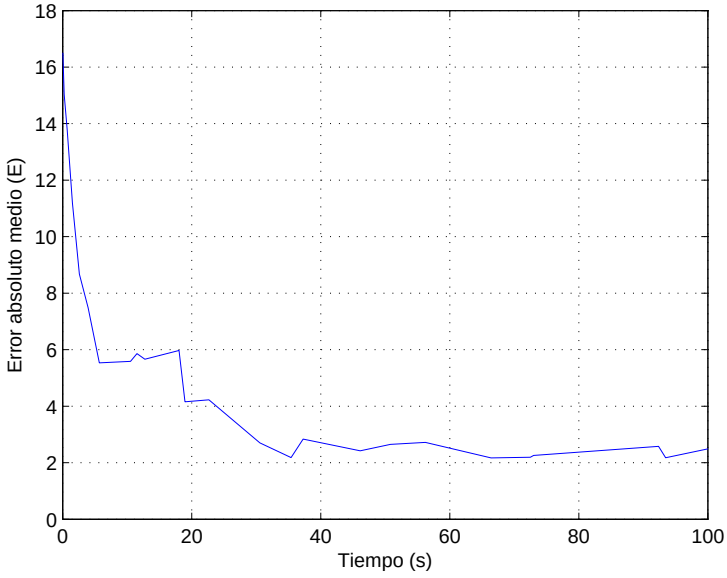


Figura 5.12: Error medio de la estimación empleando el algoritmo ARS en función del tiempo real de ejecución. Se han realizado $N_t = 50$ simulaciones independientes.

Número de objetivos	**Error absoluto medio**
2	2.3707
3	4.8924
4	7.9394
5	10.4301

Tabla 5.15: Error absoluto medio de la estimación empleando el algoritmo ARS con un sistema compuesto por $K = 5$ sensores.

Número de objetivos	**Error absoluto medio**
5	8.1686
6	8.4594
7	12.1938
8	12.7624

Tabla 5.16: Error absoluto medio de la estimación empleando el algoritmo ARS con un sistema compuesto por $K = 10$ sensores.

5.3. Comparación

En este apartado se comparan la complejidad y el rendimiento de los algoritmos estudiados en este capítulo. En primer lugar, se presenta una figura donde se muestra un ejemplo de estimación de la posición de dos objetivos empleando un sistema compuesto por $K = 5$ sensores y las trayectorias que han llevado a está estimación. En segundo lugar, se compara el error absoluto medio de la estimación y el tiempo real de ejecución de los algoritmos.

En la Figura 5.13 se presentan gráficas donde se muestra un ejemplo de estimación empleando un sistema compuesto por $K = 5$ sensores. Para realizar las estimaciones se ha utilizado la misma posición y las mismas observaciones. Como se observa en la figura todos los algoritmos llegan a una situación estacionaria. En la Figura 5.13(a) se presenta la convergencia de los algoritmos: búsqueda aleatoria acelerada (ARS), filtro de Kalman extendido (EKF), filtro de partículas (FP) y método de gradiente. En las Figuras 5.13(b) y 5.13(c) se presentan las curvas de nivel con las distintas trayectorias superpuestas.

En la Figura 5.14 se compara el error absoluto medio de todos los algoritmos estudiados. Como se puede ver en la figura, los algoritmos ARS y el método de gradiente convergen más lentamente que el EKF y el FP, pero con el paso de tiempo llegan a una situación estacionaria. El EKF y FP convergen considerablemente más rápido que los otros métodos. Cuando el EKF, el FP y el método de gradiente llegan a la situación estacionaria el error es similar. El ARS tiene un error un poco mayor que los tres anteriores al cabo del mismo tiempo.

En la Figura 5.15 se compara el tiempo real de ejecución de los algoritmos: ARS, EKF, FP y método de gradiente. Con muchas iteraciones (más de 250) el algoritmo de gradiente presenta un tiempo de ejecución más grande. El FP con muchas partículas, en este caso $M = 1000$, presenta un tiempo de ejecución mayor que el ARS y el EKF. Este último presenta el menor tiempo de ejecución de todos los algoritmos estudiados. Además, como se ha mencionado anteriormente, el ARS y el gradiente son dos algoritmos lentos comparados con el FP y el EKF. Es decir que llegan a la situación estacionaria más lentamente.

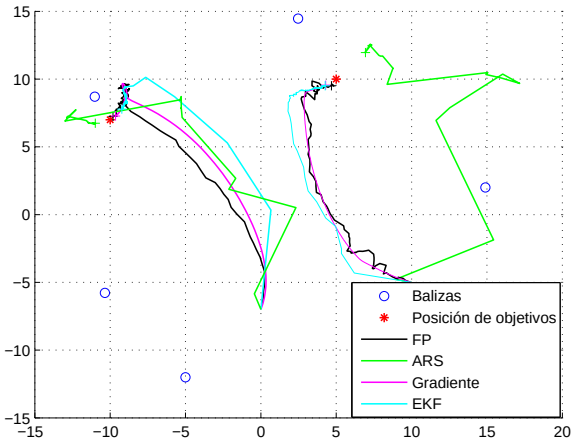
5.4. Conclusiones

En este capítulo se han presentado los diferentes algoritmos que se han estudiado para estimar la posición de múltiples objetivos. Para ello, se han adaptado los algoritmos vistos en el capítulo anterior. A continuación, se han presentado los resultados.

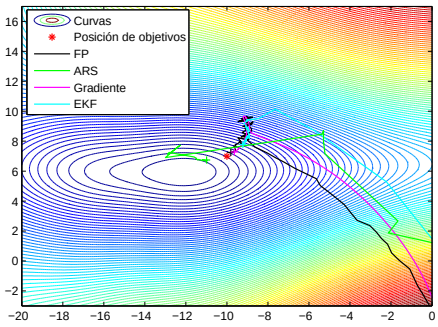
Para cada algoritmo se ha presentado un ejemplo de estimación y las trayectorias de convergencia que han llevado a está estimación. Se ha observado que todos los algoritmos convergen a la posición real de los objetivos y cuantas más observaciones se obtienen mejor es la estimación. El FP depende del número de partículas. Si se emplea un número elevado de partículas la estimación se mejora pero el coste computacional aumenta.

Se han presentado tablas donde se muestra el error absoluto medio de estimación y el número de objetivos correspondientes. Para todos los algoritmos se ha observado que con el aumento del número de objetivos, y manteniendo el número de sensores fijo, el error de estimación crece considerablemente. Esto es lógico ya que con el aumento del número de objetivos las observaciones que se reciben vuelven a ser insuficientes para realizar una estimación sin ambigüedad. En el caso en que se aumenta el número de sensores la estimación se mejora porque se obtienen más observaciones.

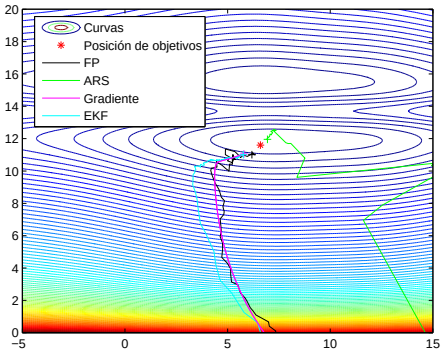
Se ha comparado el error absoluto medio de estimación y el tiempo real de ejecución. El ARS y el método de gradiente convergen lentamente comparado con el EKF y el FP. Este último depende del número de partículas, es decir, el error decrece considerablemente con el aumento del número de partículas. Pero el coste computacional crece y el tiempo real de ejecución crece también. Puesto que la posición de los objetivos es fija, con el paso de tiempo el error de estimación decrece. El método de gradiente presenta el tiempo real de ejecución más grande de todos los algoritmos estudiados.



(a) Estimación de la posición de los dos objetivos



(b) Curvas de nivel del primer objetivo



(c) Curvas de nivel del segundo objetivo

Figura 5.13: Estimación de la posición de $N_b = 2$ objetivos empleando: FP, ARS, método de gradiente y EKF en un escenario con $K = 5$ sensores.

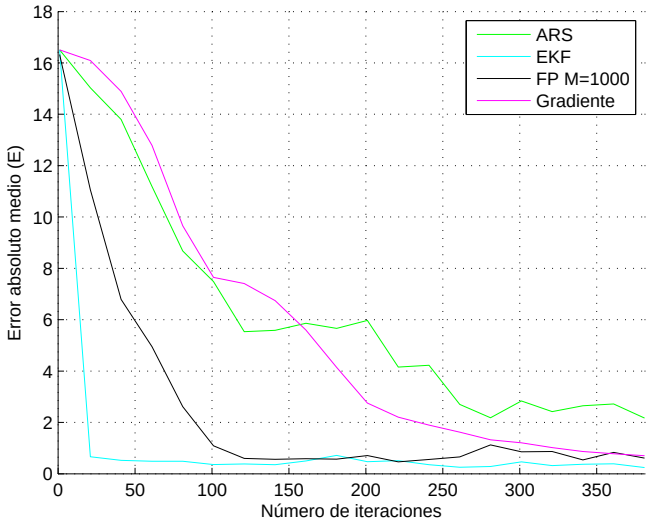
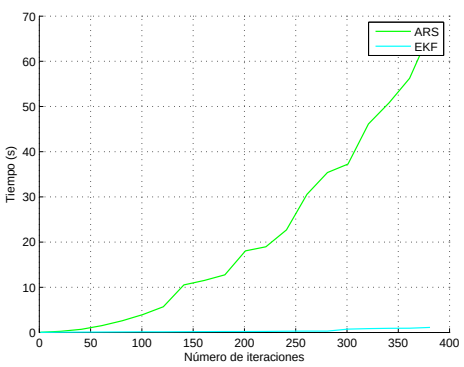
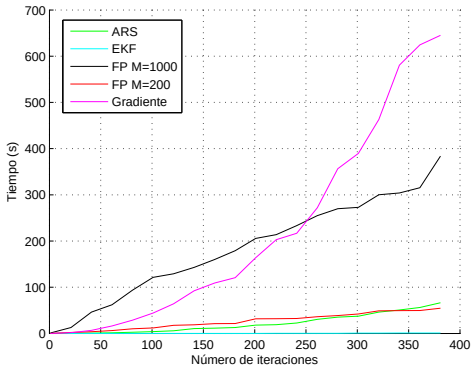


Figura 5.14: Comparación del error absoluto medio cometido al estimar la posición de $N_b = 2$ blancos con un sistema compuesto por $K = 5$ sensores.



(a) Tiempo de ejecución de: EKF y ARS



(b) Tiempo de ejecución de: FP, EKF, ARS y Gradiente

Figura 5.15: Comparación del tiempo de ejecución al estimar la posición de $N_b = 2$ blancos con un sistema compuesto por $K = 5$ sensores.

Capítulo 6

Conclusiones

El objetivo de este proyecto ha sido realizar una comparación en terminos de rendimiento y complejidad de los diferentes algoritmos presentados en la Sección 1.3. Los algoritmos estudiados son: el filtro de partículas, el filtro de Kalman extendido, el método de gradiente, la búsqueda aleatoria acelerada y el muestreo de Gibbs. Se han empleado estos algoritmos para estimar la posición de un único objetivo y múltiples objetivos. Se considera que la posición de los objetivos es fija. Los sensores están ubicados en posiciones conocidas y proporcionan observaciones de potencia de una señal emitida por los blancos de interés.

En el Capítulo 2 se ha detallado el escenario donde se han realizado las estimaciones. En primer lugar, se ha detallado el caso de un único objetivo. Se considera que en un área monitorizada por un número determinado de sensores existe un único blanco. La potencia recibida por un sensor procedente del blanco es inversamente proporcional a la distancia euclídea entre el sensor y dicho blanco. A continuación se ha detallado el caso de múltiples objetivos, en el que se considera que existen múltiples objetivos en posiciones fijas dentro del área monitorizada.

En el Capítulo 3 se ha hecho una breve descripción de algunos métodos de Monte Carlo más conocidos empezando por el muestreo exacto y el muestreo enfatizado. A partir de este último, se ha introducido el algoritmo SIS que opera secuencialmente. Sin embargo, este método presenta el problema de la degene-

ración de pesos de las partículas con el paso de tiempo. A continuación, se ha introducido el algoritmo SIR que resuelve el problema de degeneración de los pesos utilizando pasos de remuestreo.

En el Capítulo 4 se han presentado y evaluado los algoritmos que se emplean para estimar la posición de un único objetivo. Para ello, se han implementado los diferentes algoritmos en Matlab. Para todos los algoritmos se han mostrado gráficas donde se muestran ejemplos de simulación. Se ha observado que todos los algoritmos convergen a la posición real del objetivo. A continuación, se ha comparado el rendimiento y la complejidad de los diferentes algoritmos estudiados. Para ello, se han presentado gráficas del error absoluto medio y el tiempo real de ejecución. Se ha observado que el EKF y FP convergen más rápido que el ARS y el método de gradiente. Este último presenta el mayor tiempo de ejecución.

En el Capítulo 5 se han adaptado los algoritmos para poder estimar la posición de múltiples objetivos. Para cada algoritmo se ha presentado un ejemplo de estimación y las trayectorias de convergencia que han llevado a esa estimación. Se ha observado que todos los algoritmos convergen a la posición real de los objetivos y cuantas más observaciones se obtienen mejor es la estimación.

El rendimiento y el coste del FP dependen del número de partículas. Si se emplea un número elevado de partículas la estimación se mejora pero el coste computacional aumenta. Se han mostrado tablas con el error absoluto medio de estimación y el número de objetivos correspondiente. Para todos los algoritmos se ha observado que con el aumento del número de objetivos y manteniendo el número de sensores fijo, el error de estimación crece considerablemente. Esto es lógico ya que con el aumento del número de objetivos las observaciones que se reciben vuelven a ser insuficientes para realizar una estimación sin ambigüedad. En el caso en que se aumenta el número de sensores la estimación se mejora.

Igual que en el caso de un único objetivo, el ARS y el método de gradiente convergen lentamente comparado con el EKF y el FP. Puesto que la posición de los objetivos es fija, con el paso de tiempo, el error de estimación de todos los algoritmos decrece hasta llegar a la situación estacionaria. El método de gradiente presenta un tiempo real de ejecución más grande de todos los algoritmos

estudiados.

Apéndice A

Presupuesto del proyecto

A continuación se muestra los costes de material de este proyecto y las fases en la que se ha dividido el mismo. Finalmente, se adjunta el presupuesto total del proyecto calculado en base a dicha información.

A.1. Costes de material

En la Tabla A.1 se presenta el coste de material necesario para realizar este trabajo.

Concepto	**Importe**
Ordenador de gama media	1,300 euros
Lisencia Matlab	2,300 euros
Local	120 euros/mes
Documentación	200 euros

Tabla A.1: Costes de material.

El coste total de material es 3,920 euros

A.2. Planificación

En la Tabla A.2 se presenta un resumen de la planificación del proyecto y la duración del mismo. Para la planificación se consideran jornadas de 8 horas. El proyecto se puede dividir en 4 fases:

- Fase 1: Estudiar los algoritmos.
- Fase 2: Algoritmos para el caso de un único objetivo.
- Fase 3: Algoritmos para el caso de múltiples objetivos.
- Fase 4: Redactar la memoria.

Tarea	Descripción de la tarea	Duración
A	Estudiar los algoritmos	12 jornadas
B	Implementar los algoritmos de la Fase 2	22 jornadas
C	Simulaciones y sacar resultados de la Fase 2	40 jornadas
D	Implementar los algoritmos de la Fase 3	20 jornadas
E	Simulaciones y sacar los resultados de las Fase 3	46 jornadas
F	Redacción y revisión de la memoria	56 jornadas

Tabla A.2: Planificación temporal del proyecto.

Según la planificación la duración del proyecto es de 196 jornadas de 8 horas de trabajo. Si se consideran 22 jornadas de trabajo al mes, la duración total del proyecto es 8 meses y 20 jornadas de trabajo.

A.3. Coste total

Si consideramos que el coste del personal es 40 euros/hora es decir 320 euros/jornada. El presupuesto total es el mostrado en la Tabla A.3.

Concepto	**Importe**
Costes personal	62,720 euros
Costes material	3,920 euros
Local	1080
Base imponible	67,720 euros
I.V.A. (16 %)	10,835, 2 euros
TOTAL	78,555, 2 euros

Tabla A.3: Presupuesto Total

Bibliografía

[1] F. Gustafsson, F. Gunnarsson, N. Bergman,U. Forssell, J. Jansson, R. Karlsson, and Per-Johan Nordlund. "Particle Filters for Positioning, Navigation, and Tracking ".*IEEE Transactions Signal Processing*, 50(2):425-437, February 2002.

[2] W.J. Fitzgerald "Markov chain Monte Carlo methods with applications to signal processing ". *Signal Processing* , 81 3-18 (2001).

[3] A. Doucet, S. Godsill and C. Andreu "On sequential Monte Carlo sampling methods for Bayesian filtering ". *Statistics and Computing* 10, 197-208 (2000).

[4] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp "A Tutorial on Particle Filters for Online ". *IEEE Transactions on signal processing,* Vol. 50, No. 2, February 2002

[5] R. Douc, O. Cappé, and,E., Maulines "Comparison of Resampling Schemes for Particle Filtering ". *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis* pages 64-69 (2005).

[6] T. Bertozzi, D. Le Ruyet, G. Rigal and H. Vu-Thien "On Particle Filtering for digital communications"

[7] M. J. Appel, R. LaBarre and D. Radulovic "On Accelerated Random Search"*SIAM Journal on Optimization* Volume 14 Issue 3, 2003

[8] C. P. Rober and G. Casella "Monte Carlo Statistical Method "*Springer, 1999*