Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

Evaluación de técnicas de localización para un sistema RFID

Autor: María Dolores Sancho Martín
Tutor: Eva María Arias de Reyna Domínguez

Dep. Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2015-2016

Trabajo Fin de Grado
Grado en Ingeniería de las Tecnologías de Telecomunicación

Evaluación de técnicas de localización para un sistema RFID

Autor:
María Dolores Sancho Martín

Tutor:
Eva María Arias de Reyna Domínguez
Profesor titular

Dep. de Teoría de la Señal y Comunicaciones
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2015-2016

Agradecimientos

En primer lugar, agradecer a mis padres, mi hermano, abuelos y toda mi familia el apoyo incondicional junto con las palabras de aliento necesarias para alcanzar las metas que me he propuesto a lo largo de mi vida.

También a mi novio por haber sido y ser un gran pilar para llevar a cabo estos años de duro trabajo.

A todos esos amigos con los que he compartido estos años de estudio y que han acabado formando parte de mi vida.

Finalmente, a mi tutora Eva Mª Arias de Reyna Domínguez por su gran esfuerzo y ayuda para la realización de este proyecto.

Resumen

La localización en interiores es objeto de muchos estudios en la actualidad, ya que cada vez se hace más necesaria la integración de tecnologías que lo implementen, tanto en el ámbito industrial como personal. Por ello, se presenta la tecnología RFID como posible solución a dicha problemática.

Este proyecto se basa en el estudio de métodos de estimación de la localización que pueden ser integrados en sistemas RFID. De este modo, se realiza la implementación en Matlab de dichos métodos para el posterior estudio de los resultados obtenidos tras la realización de distintas simulaciones. El análisis muestra, principalmente, la precisión de la estimación frente al número de readers necesarios para obtenerla.

Abstract

Nowadays, many studies are based on indoor location because there are a lot of factories which require the services that it offers. It will also be helpful for mobile phones. Therefore, RFID technology is presented as a possible solution to this problem.

Positioning methods, which can be integrated into RFID systems, are studied in this project and they are implemented by Matlab. Finally, the analysis shows how the reliability of results is subjected to the number of readers.

Índice

ÍNDICE DE FIGURAS

Notación

N	Número de mensajes de petición de localización (queries) transmitidos por un reader en cada posición del tag. En los códigos de Matlab la denominamos como *number_msgs_Tx*
n	Número de queries recibidos en el reader que previamente había enviado N ($0 \leq n \leq N$)
$\leq$	Menor o igual
$\geq$	Mayor o igual
p(d)	Probabilidad de detección de una tag por parte de un reader que está a distancia *d*
$\mathbb{E}(p(d))$	Esperanza de la probabilidad de detección de una tag por parte de un reader que se encuentra a una distancia *d* de la misma
$\mathbb{V}(p(d))$	Varianza de la probabilidad de detección de una tag por parte de un reader que se encuentra a una distancia *d* de la tag

1 INTRODUCCIÓN

Para ver claro, basta con cambiar la dirección de la mirada.

- Antoine de Saint-Exupery -

En la actualidad, las nuevas tecnologías se encuentran en constante avance y crecimiento. Un campo que se está explorando en profundidad hoy en día es el de la localización, el cual está tomando gran relevancia en el ámbito empresarial, como por ejemplo en la parte de producción, logística, industria farmacéutica, textil, etc. Aunque se esté desarrollando dentro de la alta tecnología, también es utilizado en el día a día de las personas, basta con mirar nuestros teléfonos móviles. Por ello, en una sociedad en la que todo está conectado, cada vez es más importante que un dispositivo, objeto, persona o animal pueda ser localizado en cualquier momento. Para tal fin se emplean los sistemas de localización en tiempo real (RTLS - Real Time Location System).

Un tema que aún no está lo suficientemente explotado a nivel mundial es la localización en interiores, debido a que se están realizando estudios e implantaciones al respecto, ya que se desea una buena relación calidad-precio. Con calidad se refiere a la fiabilidad de las estimaciones que realice una tecnología, que sean lo más precisas posible, pero a un precio lo suficientemente asequible.

La tecnología conocida mundialmente para la localización es GPS, sin embargo, dentro de edificios no ofrece resultados suficientemente precisos. Por ello, se encuentran en estudio otras tecnologías para dicho propósito, como es RFID.

La identificación por radiofrecuencia (RFID: Radio Frequency IDentification) es una forma de comunicación inalámbrica que utiliza ondas de radio para identificar y rastrear objetos. El sistema consta de *readers* (o *lectores*) y *tags* (*etiquetas*) que se comunican, realizando un traspaso de información entre ellos. Las *tags* son muy pequeñas y requieren de muy poca potencia para funcionar, por lo que no necesitan baterías para almacenar datos o para realizar el intercambio de información con los lectores. Esto hace que sea fácil y barato aplicar etiquetas a todo aquello que se desee identificar o rastrear [1]. Un ejemplo del aumento del uso de dicha tecnología puede verse en la Figura [1-1], donde se puede ver el aumento exponencial de las ventas de equipos RFID. Además, este aumento es inversamente proporcional al coste de los mismos, como se puede apreciar en la Figura [1-2]. Estas gráficas muestran la gran aceptación que ha tenido esta tecnología en muy pocos años, por lo que dan una visión de futuro próspero para la tecnología RFID en los próximos años.

Por todo ello y por más razones que se explicarán a lo largo del proyecto, se ha decidido realizar este proyecto de evaluación de técnicas de estimación en base a la tecnología RFID.

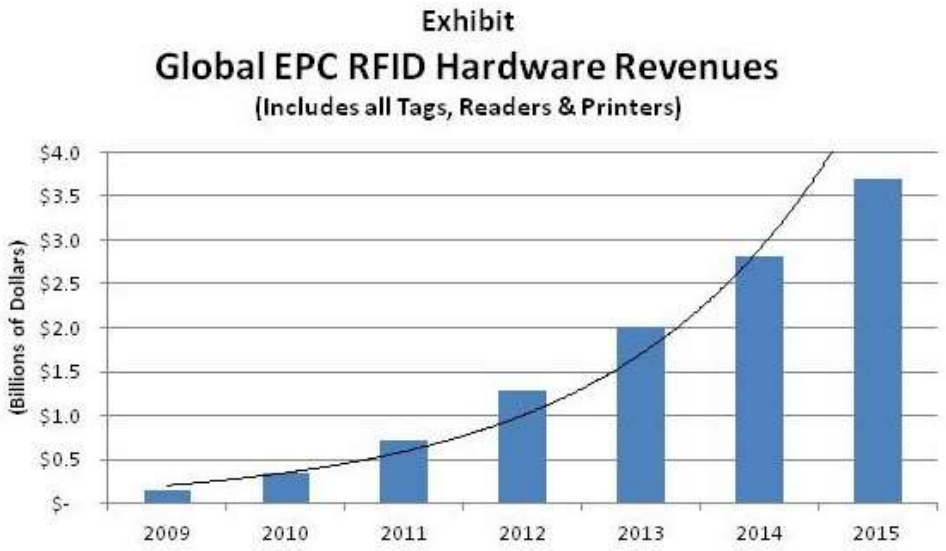

Figura 1-1. Inversión en tecnología RFID.

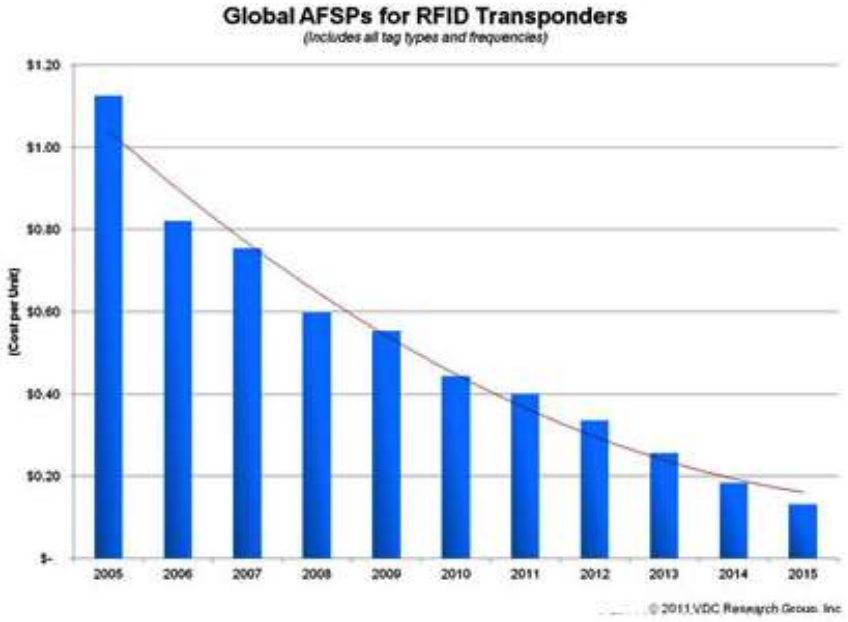

Figura 1-2. Coste etiquetas RFID.

1.1 Objetivo

El objetivo de este proyecto es el estudio de diferentes métodos de estimación de la localización, mostrando la precisión ofrecida en las estimaciones de la tag para cada uno de ellos y ofrecer la mejor alternativa de predisposición de *readers* dentro de una habitación, buscando el equilibrio entre precisión y coste.

1.2 Estructura del proyecto

Se comenzará realizando un breve y simple resumen de las tecnologías que se emplean para la localización, así como sus ventajas e inconvenientes, dejando para el final la tecnología objetivo de este proyecto que es RFID. Se continuará con la descripción de los componentes en los que se basa el sistema RFID y el funcionamiento de los mismos. Tras esto, se explicarán todos los pasos que se han seguido para el diseño e implementación del modelo en lenguaje M para Matlab y se finalizará con un análisis de los resultados obtenidos.

2 ANÁLISIS TECNOLOGÍAS DE LOCALIZACIÓN

La tecnología se alimenta a sí misma.
La tecnología hace posible más tecnología.
- Alvin Toffler -

En este capítulo se realiza un análisis de las distintas tecnologías que se emplean en la localización en interiores. La investigación destinada a dicha localización se origina tras la comprobación de la inexactitud de los sistemas GPS en el posicionamiento de un dispositivo que se encuentre fuera de la LoS (Line of Sight), es decir, al encontrarse en interiores bajo techo, la señal GPS no llega correctamente como para ofrecer una localización fiable. También tuvo lugar debido a que el sistema nombrado puede tener un error de varios metros, por lo que no cubre la necesidad de precisión que requiere la localización en interiores.

Por ello, se analizarán las ventajas y desventajas de distintas tecnologías para la implementación de las mismas en RTLS (Real Time Location Systems) en interiores.

2.1 GPS

La localización mediante GPS es la tecnología por excelencia para dicha tarea en nuestros días. Los localizadores por GPS reciben el soporte de una constelación de hasta 24 satélites, que orbitan por todo el globo terrestre enviando sus señales a todo aquel que lo necesite.

Un receptor de GPS que quiere localizarse dentro del globo terráqueo localiza al menos a cuatro satélites (cuanto mayor sea el número de satélites encontrados, mejor será la estimación de la posición) y de cada uno de ellos obtiene la posición del satélite emisor y el tiempo de envío de cada muestra recibida. Con estos datos, el localizador GPS calcula por triangulación su posición absoluta dentro de la Tierra (latitud, longitud y altitud) gracias a que los satélites emiten en el mismo preciso momento su señal, pero ésta le llega retardada al receptor por razones obvias de distancia [2]. La constelación de satélites de GPS tendría una apariencia como la mostrada en la Figura [2-1].

Este sistema tiene una precisión que puede llegar a ser mejor de 10 metros si se toman en consideración más de cuatro satélites, lo cual es bastante interesante para localizar en el mundo a nivel global, pero no tiene mucho sentido usarlo para localización dentro de un área pequeña, como puede ser un edificio.

Figura 2-1. Constelación de satélites GPS.

Otro inconveniente adicional es que al necesitar línea de visión directa (LoS), hay veces que no se encuentran suficientes satélites al alcance para permitir una localización correcta.

Además, las señales del GPS viajan muchos kilómetros y son bastante tenues, por lo que un receptor GPS en el interior de un edificio lo tendrá muy complicado para encontrar señales procedentes de los satélites y más aún para conseguir que estas señales le sirvan para localizarse. No obstante, el GPS, como su nombre indica, es un sistema de posicionamiento global y no está desarrollado para permitir la localización a nivel local.

Otra última diferencia básica entre lo que ofrece el GPS y lo que debería ser un sistema de localización en interiores es que, en un edificio, la referencia debe ser local, ya que la referencia absoluta que emplea GPS no ofrece unos datos fácilmente legibles como es la altura respecto al suelo, ya que sería más útil, si estamos en un edificio de varias plantas, situarnos en función de la planta en la que estemos.

2.2 GSM

Otra alternativa posible, que además no necesitaría la implantación de hardware adicional, sería la red de telefonía móvil GSM. Existen operadoras de telefonía móvil que ofrecen la opción de localización vía móvil a sus abonados.

Sin embargo, la falta de precisión sitúa a esta tecnología en clara desventaja respecto de otras, ya que los sistemas de localización de este tipo no pueden dar precisiones mayores de 50 metros, por lo que no son funcionales en interiores.

Esto es debido a que la localización con el uso de la red de telefonía móvil se basa en la detección de la célula a la que está conectada el teléfono móvil. Un esquema aproximado de dicho funcionamiento es el mostrado en la Figura [2-2]. En zonas urbanas la precisión es de decenas de metros, sin embargo, en las zonas rurales donde se necesitan menos células para dar servicio a menor población, esta precisión es mucho menor.

Por tanto, ésta es una clara desventaja de la tecnología GSM que hace totalmente inapropiado su uso para la localización en interiores.

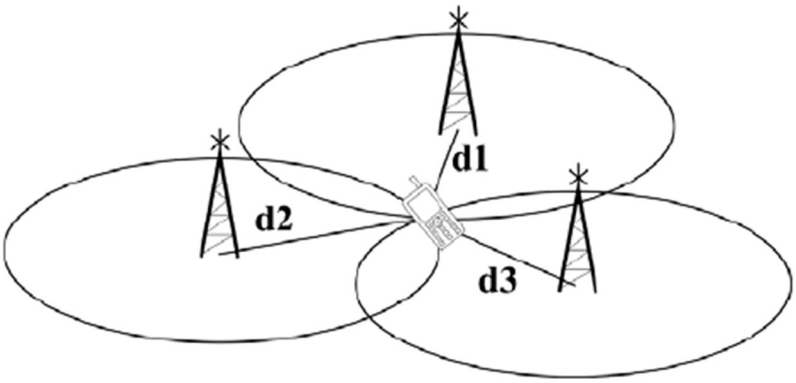

Figura 2-2. Funcionamiento localización mediante GSM.

2.3 Infrarrojos

La localización por infrarrojos se puede desechar en un primer momento para la localización en interiores por ser de corto alcance (unos 2 metros) y porque, además, se requieren enlaces LoS. Por su corto alcance habría que incluir una cantidad enorme de emisores de infrarrojos, y aun así serían imposibles de detectar ciertas localizaciones por el problema de LoS.

Existe un proyecto llamado WIPS (Wireless Indoor Positioning System) que se basa en la existencia de beacons (mensajes cortos para indicar la situación) emitidos vía infrarrojos y SmartBadges que llevan los usuarios del sistema de posicionamiento para localizarse tanto de forma pública como de forma anónima [2].

2.4 Bluetooth

En este caso la localización se basa en la colocación de puntos de posicionamiento bluetooth, pero es el terminal móvil el que calcula la posición mediante triangulación.

El terminal móvil ha de estar conectado a la red wifi para volcar la información y realizar los cómputos en un ordenador para tal fin. Este sistema permite obtener una gran precisión en el cálculo de la posición y su despliegue es rápido, ya que la instalación de las balizas es muy sencilla.

Sin embargo, en comparación con la tecnología que empleamos en este proyecto sigue siendo más cara y el dispositivo a localizar debe tener una batería y ser inteligente, como un teléfono móvil, por lo que no se podría reducir el tamaño de este dispositivo en gran medida.

2.5 Wi-Fi

Las soluciones basadas en esta tecnología calculan el posicionamiento mediante distintos métodos.

Por ejemplo, mediante triangulación. Para ello son necesarios tres o más puntos de acceso que midan la intensidad de la señal WiFi que emite el terminal móvil, y así poder calcular su posición. El terminal necesita tener el WiFi encendido, pero no hace falta estar conectado a ninguna red. Tampoco es necesario tener instalada en el terminal ninguna app. Otra técnica es “fingerprinting”, donde se plantea hacer mapas de medidas de potencia. Estos mapas se almacenan y cuando el target hace sus medidas de potencia, se comparan con los mapas.

Aunque no es la solución que ofrece mayor precisión, permite reaprovechar la infraestructura WiFi ya existente en los edificios y en muchos dispositivos.

Sin embargo, como ocurría con los dispositivos bluetooth, el dispositivo a localizar debe recibir la señal, procesarla y retransmitirla haciendo uso de su batería.

2.6 Wi-Max

La tecnología Wi-Max sigue el estándar 802.16 [3].

Está pensada para la intercomunicación de áreas muy extensas, de hasta 48 kilómetros de radio, y puede llegar a transmitir hasta 70Mbps.

Debido al gran alcance no se puede pensar en establecer un sistema de localización en interiores usando esta tecnología.

2.7 Zig-Bee

Esta tecnología es una buena opción para la localización en interiores, siendo similar a Wi-Fi, y su viabilidad ya ha sido estudiada en algunos trabajos como [4]. Sin embargo, su alcance es más reducido que en Wi-Fi y la señal fluctúa de igual manera cuando cambian las condiciones del entorno, o con el movimiento de personas.

La ventaja principal de esta tecnología es su bajo coste y baja potencia de emisión, pero su bajo ancho de banda hace que su utilidad sea reducida.

2.8 UWB (Ultra Wideband)

La tecnología UWB parece una buena candidata para la localización en interiores. En estos momentos se encuentra en estudio de aplicabilidad y de explotación.

La precisión alcanzada con este sistema es de un metro con lo que quedan patentes sus posibilidades en este ámbito.

Su ventaja principal es su robustez ante obstáculos y cambios del entorno (puertas, paredes, presencia o movimientos de personas, etc.).

2.9 UltraSound

Son sistemas basados en la utilización de nodos emisores y receptores de ultrasonidos que utilizan principalmente el tiempo de vuelo entre ondas ultrasónicas como métrica de localización para determinar la localización del usuario.

Los ultrasonidos se han convertido en una alternativa dentro del mercado RTLS debido a que proporcionan una alta precisión, pero debido a su baja cobertura y a la necesidad de desarrollar una infraestructura propia con un elevado número de nodos fijos en el entorno, puesto que es necesario garantizar una línea de visión directa entre los dispositivos, sus costes resultan muy elevados.

2.10 RFID (Radio Frequency Identification)

Este sistema basa su funcionamiento en la comunicación mediante radiofrecuencia entre dispositivos emisores/receptores (readers) y etiquetas (tags), donde se produce el almacenamiento y posterior recuperación de datos con el propósito de transmitir la identidad de un objeto.

Las etiquetas RFID son unos dispositivos pequeños, similares a una pegatina, que pueden ser adheridas o incorporadas a un producto, animal o persona. Contienen antenas para permitirles recibir y responder a peticiones por radiofrecuencia desde un emisor-receptor RFID. Las etiquetas pasivas no necesitan alimentación eléctrica interna, mientras que las activas sí lo requieren.

Un ejemplo de un sistema de localización que usa RFID es Cricket, un sistema ideado por ingenieros del MIT (Massachusetts Institute of Technology), cuya precisión es 2 centímetros y ha sido empleado en otros proyectos como seguimiento de objetos, control de robots, etc. [5]

Una de las ventajas que presenta esta tecnología es que no necesita línea de visión directa con la etiqueta (al contrario que los infrarrojos y los ultrasonidos). Otra ventaja es que dentro de un emplazamiento se empleará el mismo número de readers para la localización tanto de una como de cientos de tags, suponiendo esto un ahorro respecto a otros sistemas como UWB, ya que el coste de una etiqueta pasiva ronda pocos céntimos de euro. El mayor coste del despliegue reside en los readers y, como se ha comentado, no es necesario en el empleo de gran número de estos. Como contrapartida la cobertura del sistema es limitada, pero esto se puede solucionar utilizando etiquetas activas en vez de pasivas.

2.11 Conclusiones análisis de tecnologías

Así que se toma la tecnología RFID como una buena opción para la implementación de un sistema de localización en interiores ya que cubre los requisitos que se desean, tanto en fiabilidad como en coste.

Para poder verlo de una manera más gráfica, en comparativa con otros métodos tradicionales, la tecnología RFID aporta las ventajas mostradas en la Figura [2-3].

Sistemas tradicionales	**Tecnología RFID**
Escaneo manual del código	Lectura automática
Información fija	Información modificable sin intervención humana
Una sola lectura por acción	Lecturas múltiples por cada acción
Necesidad de visión directa	No requiere visibilidad directa. Legible a través de materiales no metálicos, tales como plástico y pintura
Proceso lento en cada lectura	Automatización de procesos e incremento de las velocidades en los flujos de materiales en producción
Fácil de falsificar	Aporta autentificación y seguridad, reduciendo errores, pasos y falsificaciones

Figura 2-3. Ventajas Tecnología RFID.

3 DESCRIPCIÓN DEL SISTEMA

La ciencia de hoy es la tecnología del mañana.

- Edward Teller -

Como se ha comentado anteriormente, la localización en interiores es un sector el cual se desea explotar hoy en día, debido a la gran demanda de un producto que sea capaz de dar una aproximación de la localización lo suficientemente fiable para su explotación.

Sin embargo, no sólo el factor de exactitud es el que interesa a las grandes compañías para poder desarrollar un producto y que éste tenga éxito. Otro factor muy importante a tener en cuenta es el coste de los productos. Como bien dijo el mismísimo *Henry Ford* – "*El verdadero progreso es el que pone la tecnología al alcance de todos*" – para ello el producto ha de ser barato y tener una buena funcionalidad.

Por este hecho, este proyecto se decanta por la tecnología RFID como posible solución al problema actual que presenta la localización en interiores, ya que es una tecnología de bajo coste, pudiendo adquirir una tag por pocos céntimos de euro, y de reducido tamaño para poder asignar las tags a dispositivos o personas sin que entorpezca a ningún movimiento, por lo tanto, será insignificante su presencia o ausencia. De esta manera, el trabajo se basará en estudiar la fiabilidad de algunas técnicas de localización para su implementación en un sistema RFID para poder solventar la problemática de la localización en interiores.

3.1 Componentes del sistema

Una aproximación de los tres principales componentes de los que consta un sistema de localización RFID sería el que se observa en la Figura [3-1]. Es un esquema muy básico que ayuda en la identificación de dichos componentes que son "*reader*", *"antena*" y "*tag*" (también denominada como "*etiqueta*" o "*transpondedor*" como se indica en la imagen).

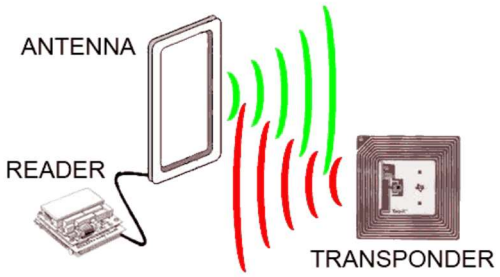

Figura 3-1. Esquema componentes tecnología RFID.

A continuación, se describen estos tres componentes principales y se nombran los que han sido empleados para realizar el estudio [6], en el cual se basa este proyecto.

3.1.1 Readers

Se debe entender como "*readers*" (o en español "*lectores*"), a aquellos dispositivos que tienen la funcionalidad de transceptores, es decir, aquellos que tienen la capacidad tanto de emitir una señal como de recibirla. Estos se conectan a una antena exterior o ellos mismos incorporan una antena que es la encargada de transmitir como tal el mensaje.

Estos dispositivos se encontrarían distribuidos de forma estratégica a lo largo del emplazamiento dentro de un edificio y emiten cada cierto tiempo una o varias señales de petición de localización (también denominada "*queries").* Cada reader recibirá de vuelta un número de queries (o ninguno en algunas situaciones), a partir del cual se obtendrá la posible localización de una *tag* situada dentro del emplazamiento de *readers*. Los readers como tales no son capaces de realizar la triangulación, por lo tanto, estos se encontrarán conectados a un dispositivo de control, bien de forma inalámbrica o cableada, para transmitirle la información recopilada y que este estime la localización.

Para dicho propósito, se ha empleado un Impinj Speedway Reader en [6] para tomar las mediciones oportunas para poder realizar el estudio de esta tecnología. Cabe decir que el emisor está programado para transmitir queries durante un periodo de 30 segundos y que la comunicación entre el reader y la tag sigue el protocolo ISO 18006C.

3.1.2 Antena

Las Antenas RFID son el elemento esencial entre el tag y el lector, ya que es la encargada de transmitir con la potencia necesaria el mensaje que desea transmitir el reader para localizar a la tag. Además, capta la señal de devolución del tag tras haber sido procesada la información por la misma.

Una antena crea un campo de acción tridimensional a su alrededor que se llama "patrón de radiación", que es donde tiene funcionalidad. Las diferencias entre las distintas antenas RFID existentes se resumen en dos características:

- Largo o corto alcance, a escoger en función de la amplitud que se desee leer.
    - Antenas RFID de largo alcance: emplean frecuencias pertenecientes al rango UHF (Ultra High Frequency) y están preparadas para captar tags hasta 14 metros de distancia.
    - Antenas RFID de corto alcance: son antenas preparadas para actuar cerca del producto (menos de 2 metros) tanto en el punto de venta como en líneas de producción. Son rápidas y eficaces en entornos de líquidos y productos densos, los cuales son elementos que tienen gran capacidad de absorción de señales.
- Para alta o baja densidad de campo, a escoger en función de la naturaleza de los productos a leer y de la cantidad a leer al mismo tiempo.

Existe un gran número de tipologías de antenas para distintas aplicaciones, como por ejemplo "antenas de alfombra" para situarlas en el suelo, "antenas de marcos" para colocarlas en las puertas, etc.

En el caso del estudio [6] el modelo empleado ha sido una antena de parche 6dBIC, que dispone de una potencia de 23.5 dBm.

3.1.3 Tag

Las etiquetas o "*tags"* como se han descrito anteriormente, son la forma de empaquetado más común y habitual de los dispositivos RFID. Son autoadhesivas y se caracterizan por su flexibilidad, su "delgadez", la capacidad de poder ser impresas con código humanamente legible en su cara frontal y las capacidades de memoria que dependerán del circuito integrado que lleve incorporado. Constan básicamente de un chip que puede ser tan pequeño que casi sea invisible al ojo humano y de una antena impresa en un material conductor alrededor del chip, como se muestra en la Figura [3-2].

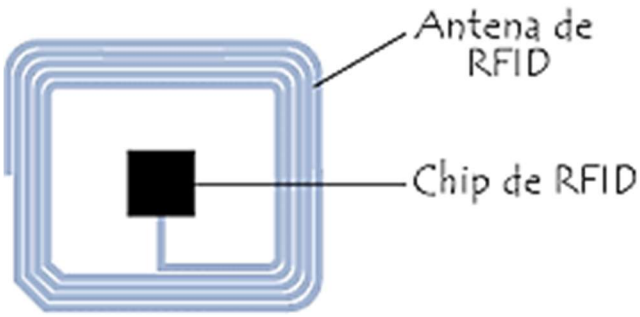

Figura 3-2. Estructura de etiqueta RFID

Los rangos de radiofrecuencias a las que pueden trabajar son las mostradas en la Figura [3-3]. Esto supondrá diferencias en el diseño de las tags debido a que sus antenas deben adaptarse a dichas frecuencias.

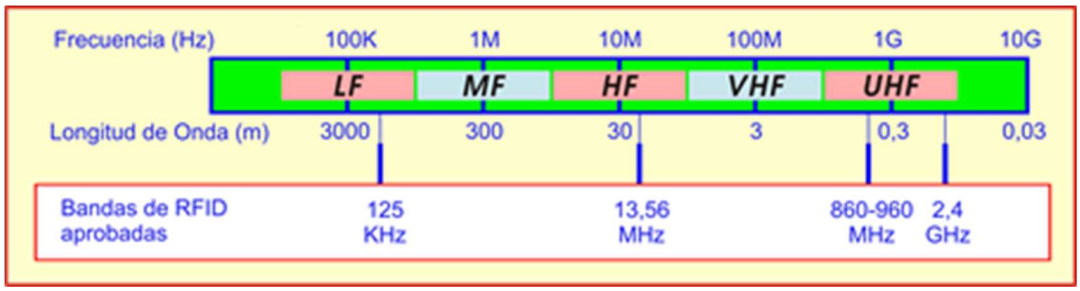

Figura 3-3. Radiofrecuencias funcionamiento RFID Tags.

Por ejemplo, se expone la comparativa entre la frecuencia 13,56 MHz y la frecuencia 868 MHz en la Figura [3-4] y dos tags que las emplean en la Figura [3-5]. Así que se puede deducir que el coste de una tag será inversamente proporcional a la frecuencia que ésta utilice.

Frecuencia	Espesor	Capas	Soldadura	Fabricación
RFID HF a 13.56 MHz	15-19 micras	2	Sí	Alto costo
RFID UHF a 868 MHz	4-9 micras	1	No	Bajo costo

Figura 3-4. Ejemplos de características de etiquetas RFID.

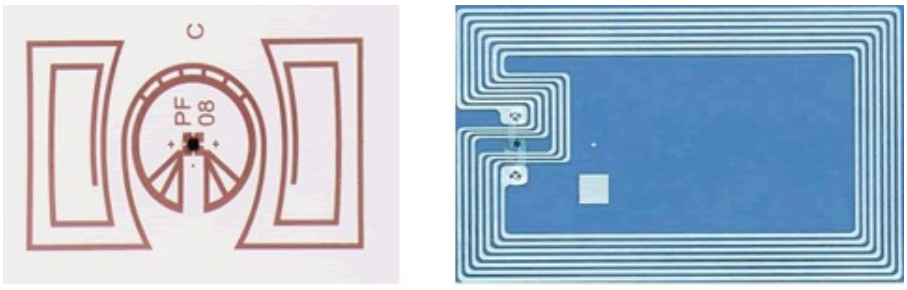

Figura 3-5. Ejemplos de RFID Tags.

En concreto, para el estudio realizado por la universidad de Stony Brook [6] el modelo de etiqueta empleado ha sido Alien Squiggle.

3.2 Estructura del sistema

En el apartado anterior se ha mostrado, a modo de introducción, un esquema donde se distinguen tres componentes principales en un sistema RFID. Ahora se presenta en la Figura [3-6] otro esquema de la estructura que tendría un sistema de esta tecnología.

Figura 3-6. Funcionamiento general básico tecnología sistema RFID.

Se observa un componente añadido a la derecha de la imagen, una unidad de control como sería un ordenador. Este componente es esencial en el sistema ya que actúa como el cerebro del mismo, siendo el que recopila toda la información que le llega desde distintos readers y muestra, tras realizar un considerable cómputo de algoritmos programados, una estimación de la posición de una o varias tags.

En este trabajo, el sistema empleado es el que se ha tomado para un experimento llevado a cabo por personal del Departamento de Ingeniería Eléctrica e Informática y del Centro de Excelencia en Tecnología Inalámbrica e Información de Stony Brook, NY, USA [6].

3.2.1 Mapa

Se denomina mapa a la zona a cubrir donde se desea realizar la localización. En él se distribuyen una serie de readers equiespaciados a lo largo de una superficie cuadrada. Para el caso en el que se basa este proyecto, se realiza la aproximación de considerar el mapa en 2D siendo visto desde arriba, pese a que su implementación experimental en la vida real fuera en 3D.

Para el experimento [6] en cuestión, se ha considerado un mapa de 48x48 metros, donde la distancia entre los readers es de 12 metros. Sin embargo, en este proyecto se toman como ejemplos mapas de diferentes tamaños y distancia entre readers variable, así se puede analizar qué ocurre en distintas ocasiones y cuál sería la combinación más óptima.

3.2.2 Disposición antenas

Cada reader dispone de tres antenas. Cada una tiene una visibilidad de 120º, dispuestas de la forma que se indica en una ilustración de [6] que podemos ver en la Figura [3-7], tendríamos con estas tres antenas una visibilidad de 360º alrededor de cada reader.

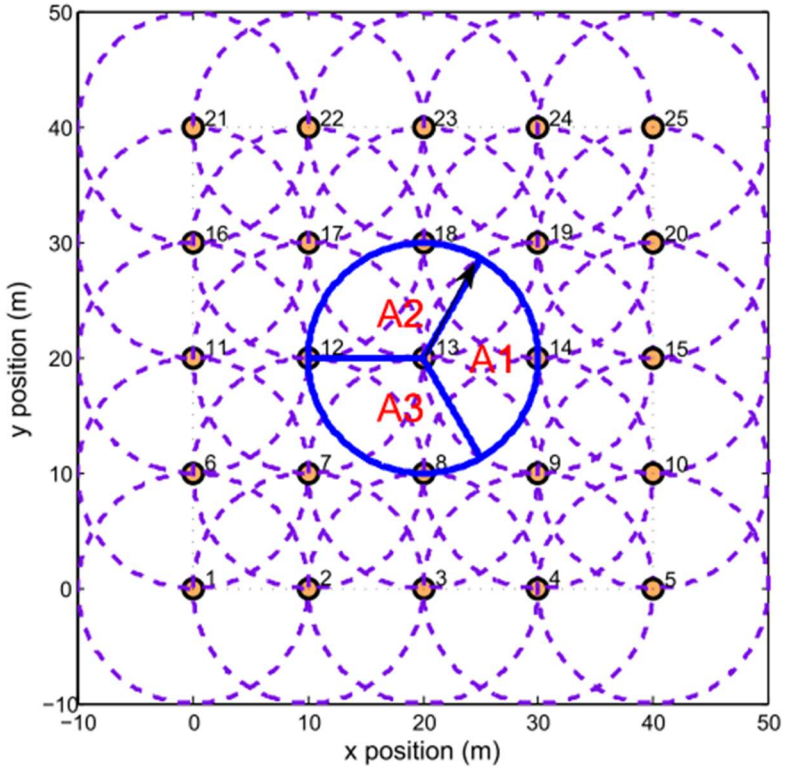

Figura 3-7. Red de readers RFID. Cada reader dispone de tres antenas con un campo de visión de 120º.

Los readers están desplegados en un mapa de 40x40m con una distancia entre ellos de 10m y cada uno está representado por un pequeño círculo, mientras que la circunferencia punteada alrededor del mismo indica su zona de cobertura. En el centro podemos ver que toma a un reader como ejemplo: la circunferencia indica el área de cobertura del reader 13, gracias a las tres antenas (A1, A2 y A3) de las que dispone.

3.2.3 Unidad de control

En este caso, el dispositivo encargado de realizar los cálculos matemáticos para la estimación de la posición de la tag sería el ordenador desde el cual se está realizando este proyecto, ya que los parámetros que supuestamente serían de entrada en un sistema real son los que se le proporcionan al programa para que los ejecute.

3.3 Funcionamiento del sistema

En este apartado se describe, en términos sencillos, las distintas fases en las que se compone el funcionamiento del sistema de localización.

3.3.1 Emisión mensaje de petición de localización

Una vez situados los readers en las posiciones oportunas dentro del mapa, emiten de forma simultánea mensajes de petición de localización a través de sus tres antenas. El número de mensajes transmitidos por cada reader es prefijado a un número considerablemente grande pero sin que suponga errores en los cálculos, siendo 100 como máximo. Un buen ejemplo es la emisión de 50 mensajes.

3.3.2 Recepción mensaje de petición de localización en el reader

El propósito de los mensajes emitidos es que lleguen hasta una o varias tags, éstas al verse alteradas por la señal electromagnética, reciben y procesan el mensaje incluyendo en el mismo la información propia (la que está contenida en cada tag). Una vez hecho esto, retransmiten la señal regresando a cada reader un porcentaje de los mensajes que había emitido.

Este porcentaje variará en función de las condiciones que se estén dando en dicho momento. Al tener una tag y un reader, la gran variable será la distancia entre los mismo. Si la tag está cerca del reader (el cual tiene una posición fija), el porcentaje de mensajes recibidos de vuelta será alto. En cambio, si la tag se encuentra a gran distancia del reader, puede ser que este porcentaje sea bajo o inclusive cero si la señal no tiene la potencia suficiente para dicho alcance.

Como comentario, recordar que la recepción del mensaje emitido desde la tag se recibe a través de la misma antena que fue emitido ya que las antenas empleadas son dispositivos transpondedores (capaces tanto de emitir como de recibir), por lo cual el cálculo de la distancia es sencillo.

3.3.3 Procesamiento de la información

Una vez se haya recopilado toda la información posible al haber emitido y recibido los mensajes, es decir, el porcentaje de recepción de los mensajes de respuesta de cada reader que es lo que nos interesa, y teniendo las coordenadas que ocupa cada reader dentro del mapa, se introduce en la unidad de control.

Este se encarga de realizar una estimación de la posición de la tag en función de estos dos parámetros: posiciones de los readers y número de respuestas recibidas desde la tag, haciendo uso de varios métodos de estimación de la localización.

4 DISEÑO DEL SISTEMA

Los que se enamoran de la práctica sin la teoría
son como los pilotos sin timón ni brújula,
que nunca podrán saber a dónde van.
- Leonardo Da Vinci -

Este capítulo se centra, una vez descrito en qué consiste el sistema de localización RFID, en cómo se ha llevado a cabo el desarrollo del entorno de trabajo y las distintas metodologías de estimación de la situación de una tag respecto a un conjunto de readers.

4.1 Puesta a punto

El objetivo de esta sección es obtener todos los parámetros necesarios para el desarrollo de los distintos métodos de aproximación a la localización. Para ello, se emplea como modelo el perteneciente al experimento [6] llevado a cabo en la Universidad de Stony Brook en Nueva York, el cual define un ámbito óptimo experimental donde se podrán realizar las simulaciones. Así que se ha adaptado este entorno a la programación en la que se basa Matlab.

El principal parámetro a calcular es el número de queries que llegan de vuelta hasta el reader que las había transmitido, tras haber llegado hasta una tag, ser procesada por esta y retransmitida. A este número le denominaremos "*n*" y será como máximo el número total de solicitudes de localización que el reader ha emitido en un primer momento, el cual será "*N"*. Como se puede entender, "*n"* puede hacerse igual a cero, esto ocurre cuando la distancia existente entre el reader y la tag es mayor del alcance que tienen las antenas propias del reader. Por lo tanto, desde un principio se ha de tener en cuenta que la distancia es el factor más importante para realizar el cálculo de "*n"*.

Por otro lado, en este apartado también se explicará cómo se ha realizado la distribución de los readers a lo largo del mapa. Estas localizaciones serán conocidas y controladas dentro del diseño del sistema ya que, gracias a estas y a las "*n"* propias de cada reader para una prueba de estimación de la posición, serán las que nos permitan calcular dicha aproximación de posición de la tag.

4.1.1 Cálculo de parámetro "n" para un reader

A continuación, se explican los pasos que se han seguido para poder realizar el cómputo de una variable "*n"* específicamente perteneciente a un reader tras la realización de una simulación. Como ya se ha escrito anteriormente, esta variable está ligada fuertemente a la distancia existente entre un reader y una tag, por lo tanto, ha de tenerse presente para los próximos cálculos.

A modo de introducción, los pasos generales seguidos en esta sección son los siguientes:

1. Para cada pareja tag-reader, se calcula la distancia "*d*" existente entre ellos.
2. A partir de "*d*" se calcula la esperanza y varianza de la probabilidad de detección según el modelo seguido en el estudio [6].
3. Con la esperanza y la varianza, se obtienen los valores numéricos de "α" y "β", los cuales dependen de "*d*".
4. Además, se halla la probabilidad de detección de una tag por parte de un reader que se encuentra a una distancia "*d", "p(d)".*
5. Así pues, gracias a "*p(d)",* se genera el valor de "*n*" como una realización de una variable aleatoria.

4.1.1.1 Media o esperanza de probabilidad de detección

En primer lugar, siguiendo el método propuesto en el experimento [6], para el modelado de la esperanza de la probabilidad de detección de una tag a una distancia "*d"* del reader se asume que tiene la forma:

$$\mathbb{E}\big(p(d)\big) = \frac{1}{1 + e^{a(d-d_0)}} \tag{4–1}$$

En el modelo [6] se ha escogido esta ecuación porque tiene carácter decreciente cuanto mayor sea "d". La probabilidad "*p(d)"* se asume como una variable aleatoria, cuya distribución sigue la distribución Beta. Como se puede observar, esta ecuación depende de tres parámetros que se han obtenido gracias a los experimentos llevados a cabo en Stony Brook [6]:

- a: es un parámetro que se ha tomado del estudio [6] como $\hat{a} = 0.8471$ para modelar las observaciones tomando las distancias en la Figura [4-1]. Además, ha de cumplirse que $a > 0$.
- d_0: es la distancia desde el reader a la que la probabilidad de detección es igual a 1/2. Este parámetro ha resultado de [6] como $\hat{d}_0 = 5.2972$ al tomar las distancias en la Figura [4-1]. También tiene que cumplirse que $d_0 > 0$.
- d: es la distancia existente entre un reader y una tag. Esta será una variable se introducirá a modo de parámetro cada vez que se desee obtener la esperanza.

Como se ha nombrado anteriormente para la estimación de parámetros, se comienza el experimento situando una tag a ciertas distancias del reader en cuestión. Para verlas de forma gráfica se ha realizado el código que se encuentra en "*Anexos –Puesta a punto – Cálculo de "n" – Media probabilidad detección – regla_inic_dist "*. En la Figura [4-1] se ve el resultado gráfico que da *regla_inic_dist*.

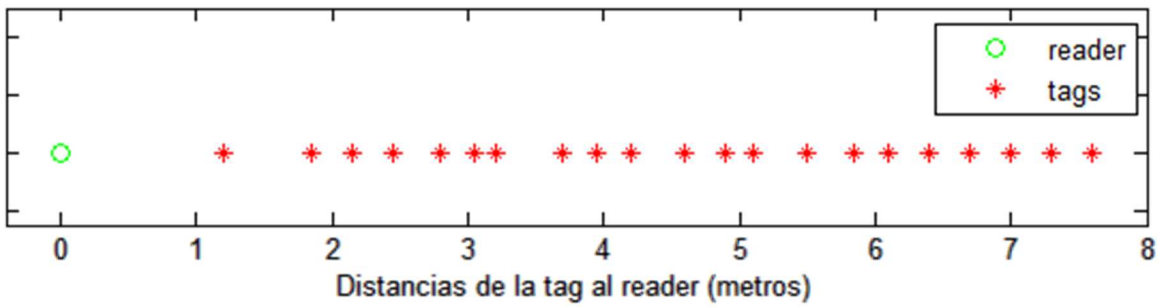

Figura 4-1. Distancias experimentales entre tag y reader.

En el estudio de [6] se presenta la gráfica de la Figura [4-3], donde se pueden ver, en una nube de puntos, los valores experimentales tomados para realizar el modelo. Además, se ha realizado el polinomio de ajuste de sexto grados de dichos puntos. Para ilustrar que la Ecuación (4-1) se ha implementado correctamente en Matlab, se obtiene la esperanza tomando las mismas distancias de la Figura [4-1].

Esto puede verse en la Figura [4-2], la cual se asemeja considerablemente a la presentada por el modelo [6]. El código implementado para calcular la esperanza puede consultarse en "*Anexos –Puesta a punto – Cálculo de "n" – Media probabilidad detección – mean_p"*.

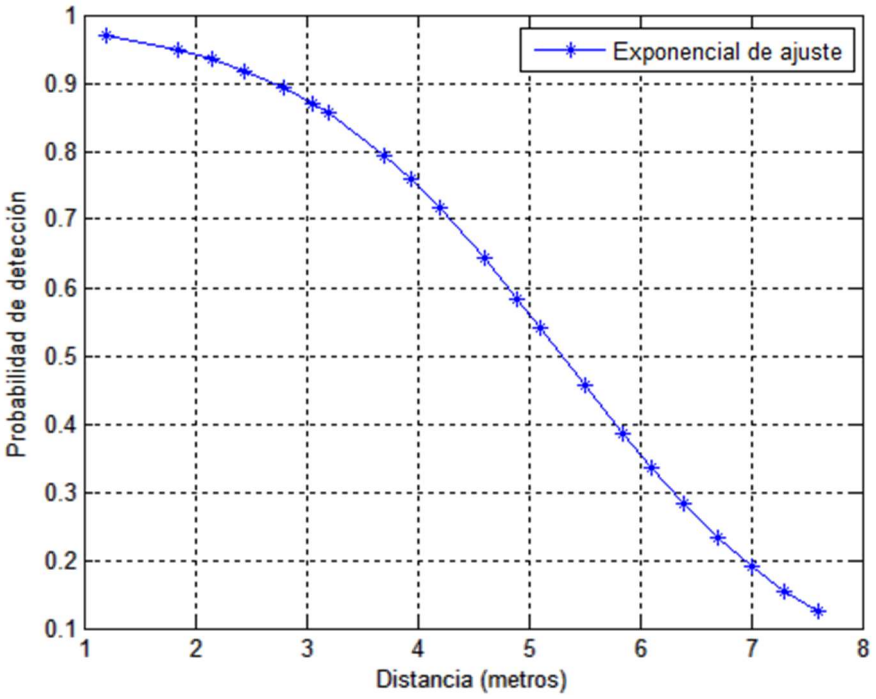

Figura 4-2. Esperanza obtenida para diferentes distancias entre tag y reader, *mean_p.*

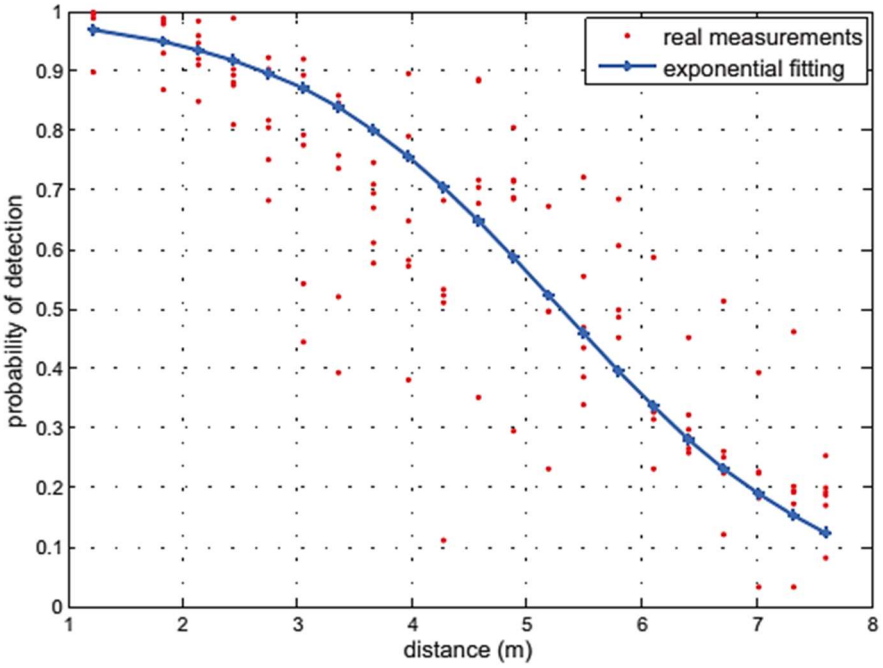

Figura 4-3. Exponencial de ajuste de los valores experimentales obtenidos. [6]

4.1.1.2 Varianza

En este apartado se quiere modelar la varianza de los valores de "*p(d)"* en función de la distancia. Para ello, se toman los valores experimentales del estudio [6] y se realiza el ajuste de los mismos con un polinomio. En la Figura [4-5] se observan los valores experimentales aportados por el documento [6] y el polinomio de sexto grado que los ajusta. En este proyecto se ha realizado la implementación de la varianza en Matlab y se puede consultar en "*Anexos –Puesta a punto – Cálculo de "n" – Varianza – var_p"*. Gracias a esta función se muestra por pantalla la Figura [4-4], la cual es muy semejante a la Figura [4-5] aportada por [6].

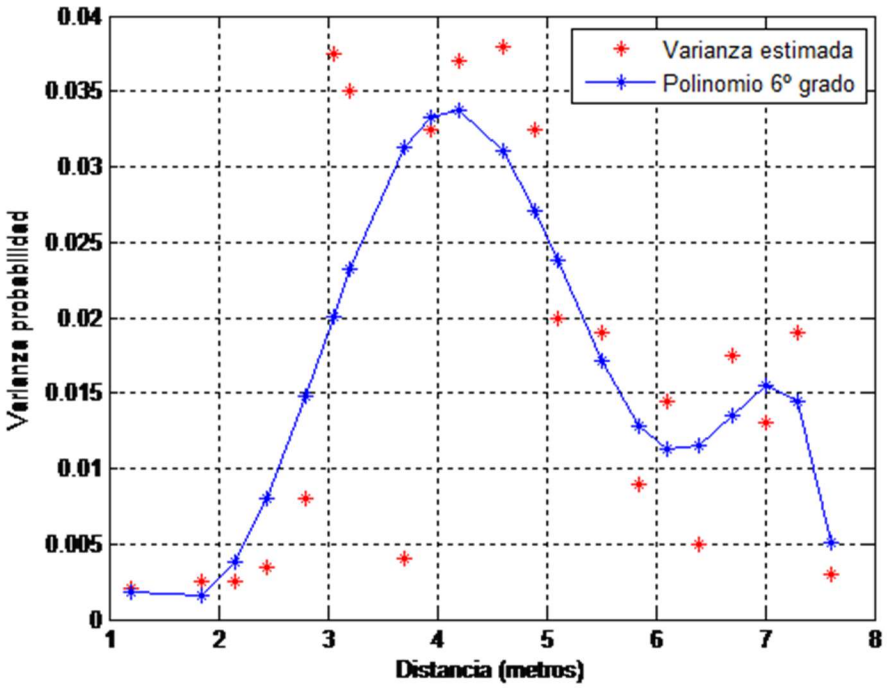

Figura 4-4. Valores varianza tomados de [6] y polinomio de ajuste de sexto grado, *var_p*.

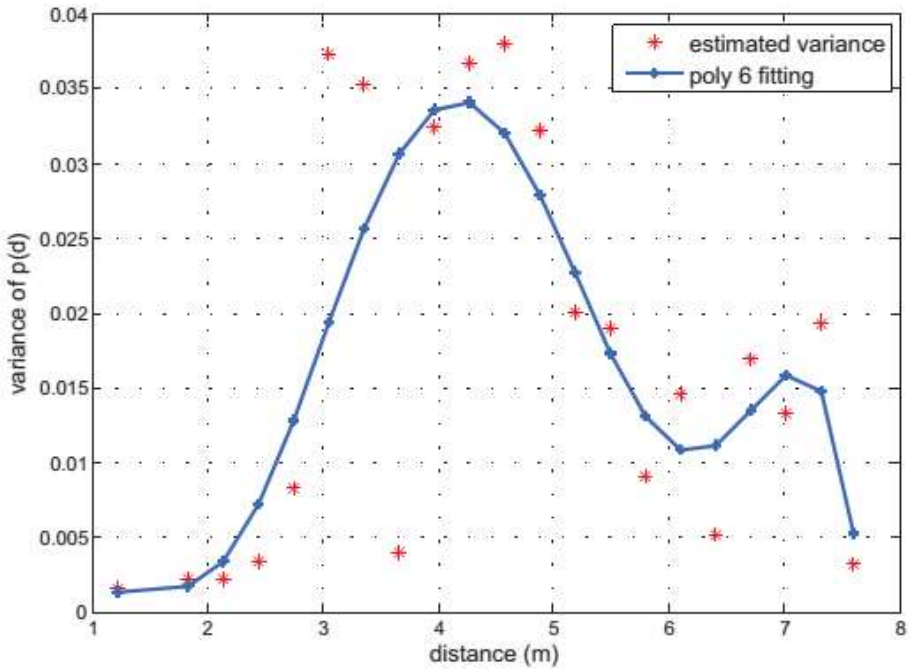

Figura 4-5. Valores promediados de la varianza y polinomio de ajuste de sexto grado. [6]

Por otra parte, se ha de tener en cuenta que las distancias entre tag y reader, tomadas para la obtención de la esperanza y la varianza en el esperimento [6], se encuentran dentro del rango 1.2 metros < "*d"* < 7.6 metros. Esto quiere decir que el modelo [6] no está diseñado para trabajar por debajo o por encima de este rango, por lo que presenta problemas de cálculos para esas distancias. El principal problema reside en el cálculo de la varianza, ya que el polinomio de sexto grado provoca que la varianza sea negativa fuera de estos rangos, lo cual no puede darse. Así que, para dar solución a esta problemática y hacer el código más robusto, se ha procedido a realizar dos truncamientos en la función de la varianza:

- Para "*d"* < 1.2 metros: la varianza se iguala a cero, asumiendo que no habrá equivocación en los datos al estar a tan corta distancia el reader de una tag.
- Para "*d"* > 7.6 metros: se toma la varianza con el valor cero por el hecho de que están tan alejados que ni si quiera se lee ningún query de vuelta, al no tener la potencia suficiente la señal para realizar el camino de ida y vuelta entre el reader y la tag.

Concluyendo, la función *var_p* recibe como parámetro de entrada la distancia existente entre reader y tag "*d"* y devuelve como salida la varianza de la probabilidad de detección para dicha distancia, *"V"*.

4.1.1.3 Parámetros α y β

A continuación, se desarrolla el procedimiento llevado a cabo para calcular los parámetros alpha y beta, los cuales son necesarios en la obtención de la probabilidad de detección "*p(d)"* de la siguiente sección.

Se ha de tener en cuenta que *p(d)* es una variable aleatoria que sigue la distribución Beta. Anteriormente se ha asumido que la esperanza de dicha variable era (4-1). Además, dado que la esperanza de una variable aleatoria Beta viene dada por:

$$\mathbb{E}\big(p(d)\big) = \frac{\alpha(d)}{\alpha(d) + \beta(d)} \tag{4–2}$$

Se puede decir que:

$$\mathbb{E}(p(d)) = \frac{1}{1 + e^{\alpha(d-d_0)}} = \frac{\alpha(d)}{\alpha(d) + \beta(d)} \tag{4–3}$$

Esta información se obtiene de [6]. Para continuar, se ha de tener también en cuenta la definición de la varianza de una variable *X* que sigue una distribución Beta, la cual se obtiene de [7]:

$$V[X] = \frac{ab}{(a + b + 1)(a + b)^2} \tag{4–4}$$

Entonces, haciendo tres cambios de variables tales como: $X = p(d)$, $a = \alpha$, $b = \beta$, se obtienen para este caso específico:

$$\mathbb{V}(p(d)) = \frac{\alpha(d)\beta(d)}{(\alpha(d) + \beta(d) + 1)(\alpha(d) + \beta(d))^2} \tag{4–5}$$

Ahora se despejan las variables $\alpha(d)$ y $\beta(d)$ de las ecuaciones (4-2) y (4-5). Para mejor visualización de los cálculos, no se ha especificado la dependencia de "α" y "β" con "d".

$$\mathbb{E} = \frac{\alpha}{\alpha + \beta} \rightarrow \mathbb{E}\alpha + \mathbb{E}\beta = \alpha \rightarrow \mathbb{E}\beta = \alpha - \mathbb{E}\alpha = \alpha(1 - \mathbb{E}) \rightarrow \beta = \frac{\alpha(1 - \mathbb{E})}{\mathbb{E}} \tag{4–6}$$

$$\mathbb{V} = \frac{\alpha\beta}{(\alpha+\beta+1)(\alpha+\beta)^2} = \frac{\alpha\left(\frac{\alpha(1-\mathbb{E})}{\mathbb{E}}\right)}{\left(\alpha+\frac{\alpha(1-\mathbb{E})}{\mathbb{E}}+1\right)\left(\alpha+\frac{\alpha(1-\mathbb{E})}{\mathbb{E}}\right)^2} \tag{4–7}$$

$$= \frac{\alpha^2\left(\frac{1-\mathbb{E}}{\mathbb{E}}\right)}{\left(\alpha+\frac{\alpha(1-\mathbb{E})}{\mathbb{E}}+1\right)\left(\alpha^2+2\alpha^2\frac{(1-\mathbb{E})}{\mathbb{E}}+\alpha^2\frac{(1-\mathbb{E})^2}{\mathbb{E}^2}\right)} = \frac{\alpha^2\left(\frac{1-\mathbb{E}}{\mathbb{E}}\right)}{\left(\frac{\alpha+\mathbb{E}}{\mathbb{E}}\right)\left(\frac{\alpha^2}{\mathbb{E}^2}\right)}$$

$$= \frac{\alpha^2\left(\frac{1-\mathbb{E}}{\mathbb{E}}\right)}{\frac{\alpha^3+\mathbb{E}\alpha^2}{\mathbb{E}^3}} = \frac{\alpha^2\left(\frac{1-\mathbb{E}}{\mathbb{E}}\right)}{\alpha^2\left(\frac{\alpha+\mathbb{E}}{\mathbb{E}^3}\right)} = \frac{\mathbb{E}^2(1-\mathbb{E})}{\alpha+\mathbb{E}} \rightarrow \mathbb{V}\alpha+\mathbb{V}\mathbb{E} = \mathbb{E}^2-\mathbb{E}^3 \rightarrow$$

$$\rightarrow \alpha = \frac{\mathbb{E}^2-\mathbb{E}^3-\mathbb{E}\mathbb{V}}{\mathbb{V}}$$

Una vez obtenidos estos resultados de manera algebraica, se pasa a la implementación de los mismos a código Matlab. A esta función se le denomina "*parametros"*, la cual se puede consultar en "*Anexos – Puesta a punto – Cálculo de "n" – Alpha y beta – parametros"*, que recibe como entrada la esperanza y varianza de *"p(d)"*. Como "$\mathbb{E}$" y "$\mathbb{V}$" ya se han obtenido a partir de un valor de distancia "*d",* se están obteniendo los parámetros de salida alpha (*"α"*) y beta (*"β"*) en función de esta misma.

4.1.1.4 Probabilidad de detección de un reader

Ahora se va a obtener la probabilidad *"p(d)"* tan nombrada anteriormente que permitirá la obtención de "*n"*.

Como se sabe que la distribución de dicha variable sigue una distribución beta [6], lo hacemos de la manera que se presenta en la función "*distbeta"* ("*Anexos –Puesta a punto – Cálculo de "n" – Probabilidad detección"*). A esta función se le introduce como entrada la distancia entre el reader y la tag, "*d",* y como salida la probabilidad *"p(d)"* que se le nombra como "*prob"*. Además, ofrece como salida otros parámetros que les son necesarios a funciones que llama a esta, como son *"alpha"*, *"beta"* y "*d"*.

Para asegurar que "*prob"* siempre obtiene un valor coherente, cuando "*d"* se encuentre fuera de los límites tomados por el experimento [6], se han puesto dos condiciones en el código al respecto:

- Si "*d"* < 1.2 metros: "*prob"* = 1 ya que la tag y el reader están muy juntos y se asegura que existirá detección de la tag a dicha distancia
- Si "*d"* > 7.6 metros: "*prob"* = 0 ya que la tag y el reader están tan alejados que no se podrá detectar.

4.1.1.5 Parámetro "n"

Finalmente se calcula "*n",* que es el propósito de esta sección y se realiza mediante la función "*n_acks"*. Según [6], este parámetro está modelado por una distribución binomial. Así pues, "*n"* se calcula mediante la función "*n_acks",* que se puede consultar en "*Anexos –Puesta a punto – Cálculo de "n" – Parámetro "n" – n_acks*.

Gracias a la "*p(d)"* obtenida anteriormente y al número total de peticiones que han sido mandadas por el reader (*"number_msgs_Tx"* en el código o "*N"* como se toma en este proyecto)*",* es posible calcular "*n"* simplemente implementando la función de Matlab *binornd*.

4.1.2 Distribución de los readers en el mapa

En esta sección se lleva a cabo la asignación de un "*n"* a cada reader para una tag en concreto. Los lectores se encuentran distribuidos en un mapa al que se le han asignado unas dimensiones específicas ("*x"* e *"y"*).

Para ello, se ha creado la función "*mapa_rdrs_acks"* ("*Anexos –Puesta a punto – Distribución readers en el mapa" – mapa_rdrs_acks"*), a la que se le pasa como parámetros de entrada las dimensiones "*x"* e "*y"*, "*dist"* que es la distancia de separación entre readers, *"number_msgs_Tx"* y las coordenadas de una tag genérica ("*xtag*"," ytag").

Esta función emite como salida dos parámetros:

- *mapa_acks:* matriz que tiene como tamaño las dimensiones que se le ha asignado al mapa y donde se ha sustituido, en cada coordenada donde se encontraría un reader, el valor "*n"* correspondiente a ese reader para la posición dada de una tag.
- *mapa_lectores:* es similar a la anterior conceptualmente, pero esta matriz tiene menor dimensión al incluir únicamente los valores "*n*" en cada reader, es decir, sólo muestra los readers, siendo sus dimensiones "*x/dist"* y "*y/dist"*. Esta matriz es para poder visualizar el número de "*n"* que recibe cada reader por el WorkSpace. Como ejemplo para su entendimiento se expone el siguiente:

Ejemplo 4–1. *Para un mapa de dimensiones 10x10 metros y una separación entre readers de 5 metros, donde se sitúa una tag en la posición (3,7) metros y los readers emiten 100 solicitudes de localización lo que se puede ver en el WorkSpace es lo siguiente*

Mapa representante del número de acks que recibe cada reader:

mapa_lectores =

66	91	0
87	98	7
0	1	0

4.2 Métodos de estimación empleados

Las metodologías que se han seguido para la estimación de la posición de una tag dentro del mapa tienen, como principales parámetros, la distribución de readers dentro del propio mapa y el número de queries que ha recibido cada uno, "*n"*.

Se emplean cuatro métodos de estimación distintos, como son el "método de asociación", "método del centroide", "método del centroide ponderado" y "aproximación al método de máxima verosimilitud". Se comienza explicando el método más simple, a nivel conceptual y de desarrollo, para terminar con el que ofrece mayor complejidad.

4.2.1 Método de asociación

Este método es el más básico a la hora de estimar la posición de una tag ya que basa su funcionamiento en asociar como localización de la tag las coordenadas del reader que tenga el mayor número de respuestas a sus peticiones, "*n".* Con esto se presupone que ese sería el reader más cercano a la tag y se asignan las coordenadas del mismo como las coordenadas de la tag estimada.

4.2.1.1 Estimación de la posición de la tag

La implementación de este método de estimación de una tag se puede ver en "*Anexos – Métodos de estimación – Método de asociación – Estimación posición de la tag – aproxtag_1metodo"*.

La función "*aproxtag_1metodo"* recibe como entrada únicamente el parámetro "*mapa_acks"*; gracias a este, se computa el índice o coordenadas del mapa donde se encuentra el mayor "*n"*, así que la tag estimada se sitúa en las mismas coordenadas de dicho reader.

Como salida, la función "*aproxtag_1metodo"* ofrece las coordenadas de la tag estimada (*"xaprox_tag","yaprox_tag"*). También ofece a modo de traspaso para la función que llama a ésta, las coordenadas de los readers que reciben queries (*"x_rdrsRx","* y*_rdrsRx"*), es decir, donde "*n" > 0,* resultando *"x_rdrsRx" y "y_rdrsRx"* vectores (ya que no interesa que sólo lea un reader, sino varios para poder estimar la posición correctamente).

Para hacer el código más robusto y coherente, se ha tenido en cuenta la situación de que el máximo "*n"* se de en varios readers a la vez, lo cual ocasionaría un problema de cómputo. Para solventar dicha problemática, pero haciendo que la estimación sea lo más simple posible, para tal ocasión se toma el primer reader que se encuentra con la máxima "*n",* es decir, el que se encuentra más próximo a la posición (0,0) metros.

4.2.1.2 Representación gráfica

Mediante la función "*primer_metodo"* lo que se realiza básicamente es la representación gráfica del resultado obtenido en "*aproxtag_1metodo"* y devuelve como salida el error cuadrático (square error - *se*) cometido en la estimación de la posición de la tag respecto de la posición de la tag real "*se1metodo"*. Esta función se puede consultar en "*Anexos – Métodos de estimación – Método de asociación – Representación gráfica – primer_metodo"*.

Para ello, se dibujan en un mapa o grid con las dimensiones que tiene la variable "*mapa_acks":*

- Readers: la posición de todos los readers desplegados en el mapa se representan mediante círculos verdes.
- Readers que han recibido ack/s: todos aquellos readers en los que "*n" > 0* se identifican en el mapa dibujando una cruz superpuesta a los círculos de los readers.
- Tag real: la posición de la tag real se representa mediante un cuadrado, así ofrecerá de una forma visual al lector el error de la estimación de la tag que se ha producido al poder visualizar ambas.
- Tag estimada: se representa mediante un asterisco la posición de la tag que se ha estimado en el método.

Para poder realizar estos cálculos, la función "*primer_metodo"* requiere como parámetros de entrada el mapa donde se distribuyen los "*n"* ("*mapa_acks"*), *"dist_rdrs"* que es la distancia entre readers y las coordenadas de la tag real (*"xtag_real","ytag_real"*) para poder realizar el error cuadrático.

A continuación, se muestra un ejemplo [4-2] para mejor visualización de lo descrito.

Ejemplo 4–2. *Para un mapa de dimensiones 10x10 metros y una separación entre readers de 5 metros, donde se sitúa una tag en la posición (3,7) metros y los readers emiten 100 queries, la estimación mediante asociación resulta como en la Figura [4-6], que sigue el mapa_acks del Ejemplo [4-1]*

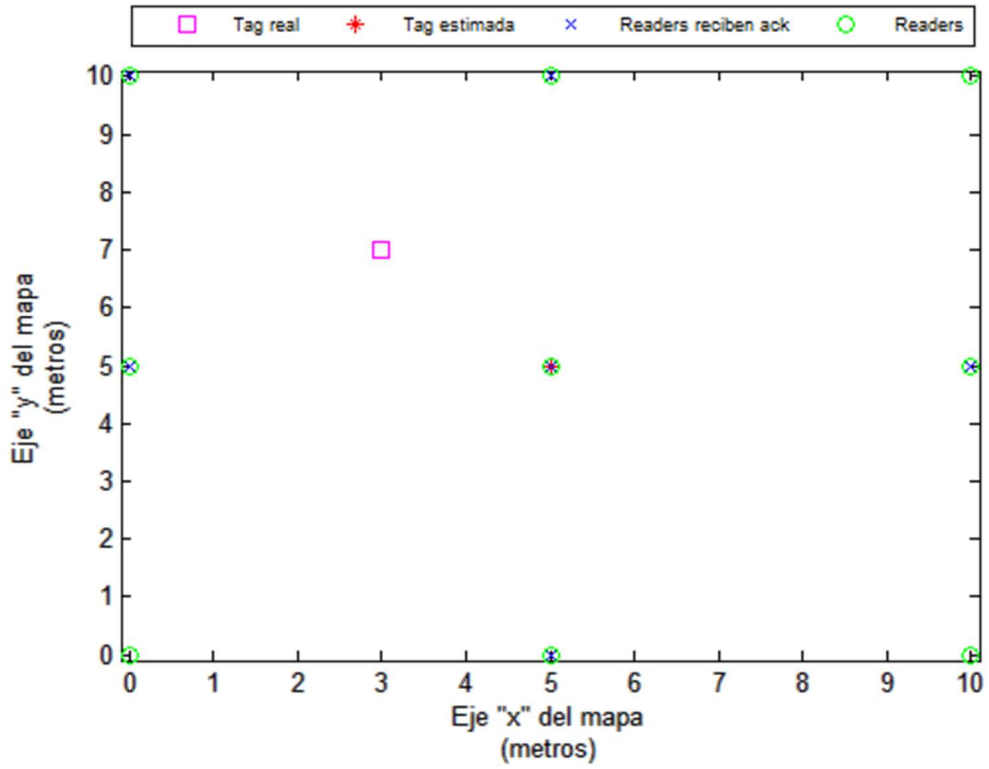

Figura 4-6. Ejemplo método de asociación.

4.2.2 Método del centroide

El segundo método que tratamos es algo más complejo que el anterior, pero se puede considerar que sus estimaciones son más exactas. Se basa en tener en cuenta sólo los readers que han recibido alguna petición de vuelta desde la tag, es decir, todos aquellos en los que "*n*" > 0 y se hace la media de todas las posiciones de los mismos, situando la tag estimada justo en el centro de estos lectores.

4.2.2.1 Estimación de la posición de la tag

Considerando que a la posición de una tag estimada se le denota con el vector $\underline{\hat{z}} = [\hat{x}\ \hat{y}]$, dentro de un mapa constituido por coordenadas *x* e *y,* la fórmula de estimación de la posición de la tag [8] es:

$$\underline{\hat{z}} = \frac{1}{L}\sum_{j=1}^{L} \underline{Z_j} \tag{4–8}$$

Donde "L" es el número de readers que han recibido acks y el vector "$\underline{Z_j}$" constituye las coordenadas correspondientes a cada reader, $\underline{Z_j} = [x_j\ y_j]$. Se denota como "*j*" a un reader en concreto.

La implementación del mismo se puede ver en "*Anexos – Métodos de estimación – Método del centroide – Estimación posición de la tag – aproxtag_2metodo*".

La función "*aproxtag_2metodo*" recibe como entrada el parámetro "*mapa_acks*"; gracias a este, se extrae las coordenadas de los readers que leen los acks o respuestas, situando la tag estimada justo en el centro de ellos siguiendo la fórmula (4-8).

Como salida, la función "*aproxtag_2metodo*" ofrece las coordenadas de la tag estimada (*"xaprox_tag","yaprox_tag"*) y las coordenadas de los readers que reciben queries (*"x_rdrsRx","*y*_rdrsRx"*), es decir, donde "*n*" > *0.*

4.2.2.2 Representación gráfica

Mediante la función *"segundo_metodo"* lo que se realiza es la representación gráfica del resultado obtenido en "*aproxtag_2metodo"* y devuelve como salida el error cuadrático (square error - *se*) cometido en la estimación de la posición de la tag respecto de la posición de la tag real, *"se2metodo"*. Esta función se puede consultar en "*Anexos – Métodos de estimación – Método del centroide – Representación gráfica –segundo_metodo"*.

Para ello, como en el método anterior, se dibujan en un mapa la distribución de los readers mediante círculos y cuáles de estos han leído algún query de vuelta, señalándolos con una cruz. Además, se representan tanto la tag real como la estimada, para ver cuán buena ha sido la estimación de la tag en dicho método.

Para poder realizar estos cálculos, la función "*segundo_metodo"* requiere como parámetros de entrada el mapa donde se distribuyen los "*n"* ("*mapa_acks"*), *"dist_rdrs"* que es la distancia entre readers y las coordenadas de la tag real (*"xtag_real","ytag_real"*) para poder calcular el error cuadrático.

Ejemplo 4–3. *Para un mapa de dimensiones 10x10 metros y una separación entre readers de 5 metros, donde se sitúa una tag en la posición (3,7) metros y los readers emiten 100 queries, la estimación mediante el centroide resulta como en la Figura [4-7], que sigue el mapa_acks del Ejemplo [4-1]*

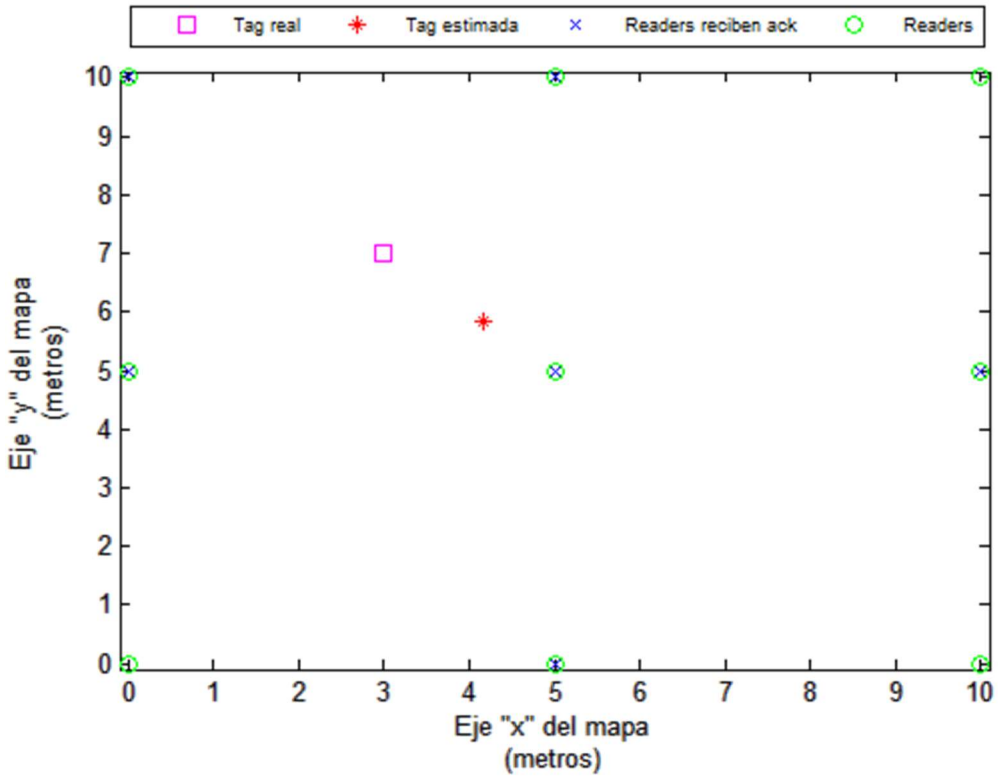

Figura 4-7. Ejemplo método del centroide.

4.2.3 Método del centroide ponderado

Este tercer método se puede considerar como una mejora del método del centroide. Al igual que el anterior, se toman las coordenadas de aquellos readers en los que "*n" > 0* aunque también el propio "*n"* de cada uno de ellos. Entonces, el método consiste en realizar una media de dichas posiciones considerando una ponderación para cada una, a mayor número de acks recibidos en un reader, mayor relevancia tendrá dicha posición.

4.2.3.1 Estimación de la posición de la tag

Considerando que a la posición de una tag estimada se le denota con el vector $\underline{\hat{z}} = [\hat{x}\ \hat{y}]$, dentro de un mapa constituido por coordenadas *x* e *y,* la fórmula de estimación de la posición de la tag [8] es:

$$\underline{\hat{z}} = \frac{\sum_{j=1}^{L}(w_j \underline{Z_j})}{\sum_{j=1}^{L} w_j} \tag{4–9}$$

Donde "L" es el número de readers que han recibido acks y el vector "$\underline{Z_j}$" constituye las coordenadas de dichos readers, $\underline{Z_j} = [x_j\ y_j]$. Se denota como "*j"* a un reader en concreto. La modificación puede apreciarse por el término "w_j", que representa los pesos, es decir, la importancia de cada reader dentro de la suma. Estos pesos, que son asignados a cada reader, se definen en la siguiente ecuación (4-10):

$$w_j = \frac{n_j}{N} \tag{4–10}$$

En esta ecuación se identifica a "n_j" como el número "*n"* que se comentaba anteriormente y a "*N"* como el número total de queries emitidos por cada reader.

La implementación de este método se puede consultar en "*Anexos – Métodos de estimación – Método del centroide ponderado – Estimación posición de la tag – aproxtag_3metodo"*.

La función "*aproxtag_3metodo"* recibe como entrada el parámetro "*mapa_acks"* y, además, *"number_msgs_Tx"* que es equivalente a "*N"* de la ecuación (4-10).

Como salida, la función "*aproxtag_3metodo"* ofrece las coordenadas de la tag estimada (*"xaprox_tag","yaprox_tag"*) y las coordenadas de los readers que reciben queries (*"x_rdrsRx","* y*_rdrsRx"*), es decir, donde "*n" > 0.*

4.2.3.2 Representación gráfica

La función "*tercer_metodo"* realiza la representación gráfica del resultado obtenido en "*aproxtag_3metodo"* y devuelve como salida el error cuadrático (square error - *se*) cometido en la estimación de la posición de la tag respecto de la posición de la tag real *"se3metodo"*. Esta función se puede consultar en "*Anexos – Métodos de estimación – Método del centroide – Representación gráfica – tercer_metodo"*.

Para ello, se dibujan en un mapa de dimensiones iguales que las que tiene la variable "*mapa_acks",* como en los métodos anteriores, una malla de readers equiespaciados, cúales de ellos reciben queries de vuelta y ambas tag, tanto la real como la estimada, para poder ver el error que se ha cometido en la estimación de la tag.

Para poder realizar estos cálculos, la función "*tercer_metodo"* requiere como parámetros de entrada el mapa donde se distribuyen los "*n"* ("*mapa_acks"*)*, "dist_rdrs",* las coordenadas de la tag real (*"xtag_real","ytag_real"*) y *"number_msgs_Tx"* nombrado anteriormente para pasárselo a la función "*aproxtag_3metodo"*.

Para mostrar un ejemplo visual al lector se expone el Ejemplo [4-4].

Ejemplo 4–4. *Para un mapa de dimensiones 10x10 metros y una separación entre readers de 5 metros, donde se sitúa una tag en la posición (3,7) metros y los readers emiten 100 queries, la estimación mediante el centroide ponderado resulta como en la Figura [4-8], que sigue el mapa_acks del Ejemplo [4-1]*

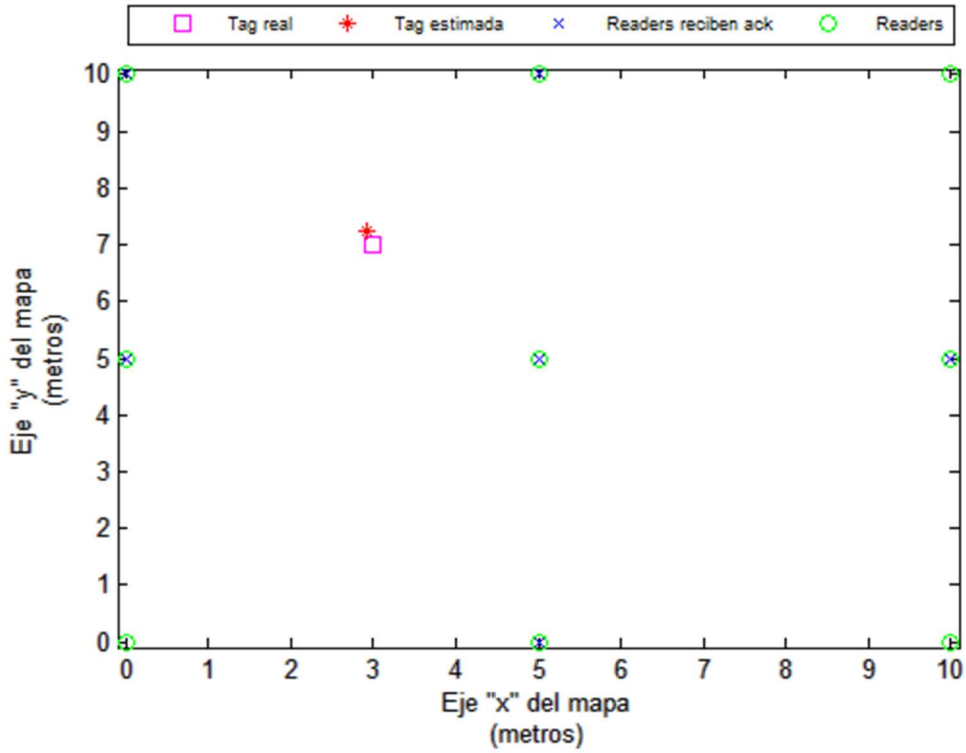

Figura 4-8. Ejemplo método del centroide ponderado.

4.2.4 Aproximación al método de máxima verosimilitud

El método de máxima verosimilitud es el último que se expone en este proyecto y también el más complejo de los cuatro que se tratan. Muchos procedimientos estadísticos suponen que los datos siguen algún tipo de modelo matemático que se define mediante una ecuación, en la que se desconoce alguno de sus parámetros, siendo éstos calculados o estimados a partir de la información obtenida en un estudio bien diseñado para tal fin [9]. En este caso, los parámetros a estimar son las coordenadas de una tag dentro de un mapa.

Así pues, la evaluación por la aproximación al método de máxima verosimilitud procura encontrar los valores más probables de la posición de la tag para un conjunto de datos, maximizando el valor de lo que se conoce como la "función de verosimilitud". Dicha función se basa en la función de densidad de la probabilidad (fdp) para una distribución dada. [10]

Se le denomina aproximación porque, pese a que el código en Matlab esté diseñado para hallar el método de máxima verosimilitud, debido a las limitaciones computacionales, no resulta el valor exacto de máxima verosimilitud con la precisión tomada en este trabajo, aunque es válido como una aproximación. Esta aproximación considera que el mapa no es continuo tal y como ocurre en la vida real, sino que es un grid en el cual se toman posibles posiciones para situar la tag. Tras recorrer todo el grid y realizar los cálculos oportunos, se asigna como localización de la tag estimada aquella en la que el valor de la función de verosimilitud sea el mayor de todo el grid.

Siguiendo el estudio [6], la función de verosimilitud viene dada por la ecuación (4-11), la cual proporciona la probabilidad de que se de el vector de queries recibidos en todos los readers "$\underline{n}$" cuando se sitúa el punto de pueba en una posición del grid "$\underline{z}$".

$$P(\underline{n}|\underline{z}) = \prod_{j=1}^{M} P(n_j|\underline{z}) \tag{4–11}$$

Esta función toma los distintos factores "$P(n|\underline{z})$" de la expresión (4-12), la cual sigue una distribución Beta-binomial. Esto es posible ya que (4-12) ofrece como resultado la probabilidad de que se de para un reader "*j*" cierto valor de "n_j" (es decir, queries recibidos), cuando la posición de pueba de estimación de la tag ("$\underline{z}$") se encuentra a una distancia "d_j" de dicho reader "*j"*. De esta forma, (4-11) va recorriendo todos los readers, que son "*M"* en total, y calculando dicha probabilidad. El cálculo de dicha "d_j" se realiza haciendo una llamada a la función "*calc_distancia",* creada para este proyecto, que se puede consultar en "*Anexos – Métodos de estimación – Método de máxima verosimilitud – Estimación posición de la tag"*

$$P(n_j|d_j) = \binom{N}{n} \frac{B(n+\alpha(d), N-n+\beta(d))}{B(\alpha(d),\beta(d))} \tag{4–12}$$

En la ecuación (4-12) se ha empleado "*p*" en lugar de "*p(d)*" por simplicidad y se toma a *"B(.,.)"* como la función Beta definida en la ecuación (4-13) [11].

$$B(v,w) = \int_0^1 t^{v-1}(1-t)^{w-1}\,dt \tag{4–13}$$

En la Figura [4-9] se muestra a modo de ejemplo la toma de 3 puntos de prueba de posición de la tag en el grid y se muestran con líneas de distinta forma para cada uno, las distancias de los mismos a cada reader del mapa. Es decir, todas las distancias que se calcularían para una posición de prueba en la ecuación (4-11). Recordemos que el número de puntos de prueba depende de una variable denominada "*precision*", siendo mayor el número de puntos que se tomen dentro del grid cuanto menor sea la precisión que se exigida.

Ejemplo 4–5. *Para un mapa de dimensiones 4x4 metros, una separación entre readers de 2 metros y una precisión de 0.5 metros en la toma de puntos de prueba en el grid, se dibujan en la Figura [4-9] las distancias entre los readers del mapa y tres puntos de prueba del grid (cuadrado, rombo y estrella)*

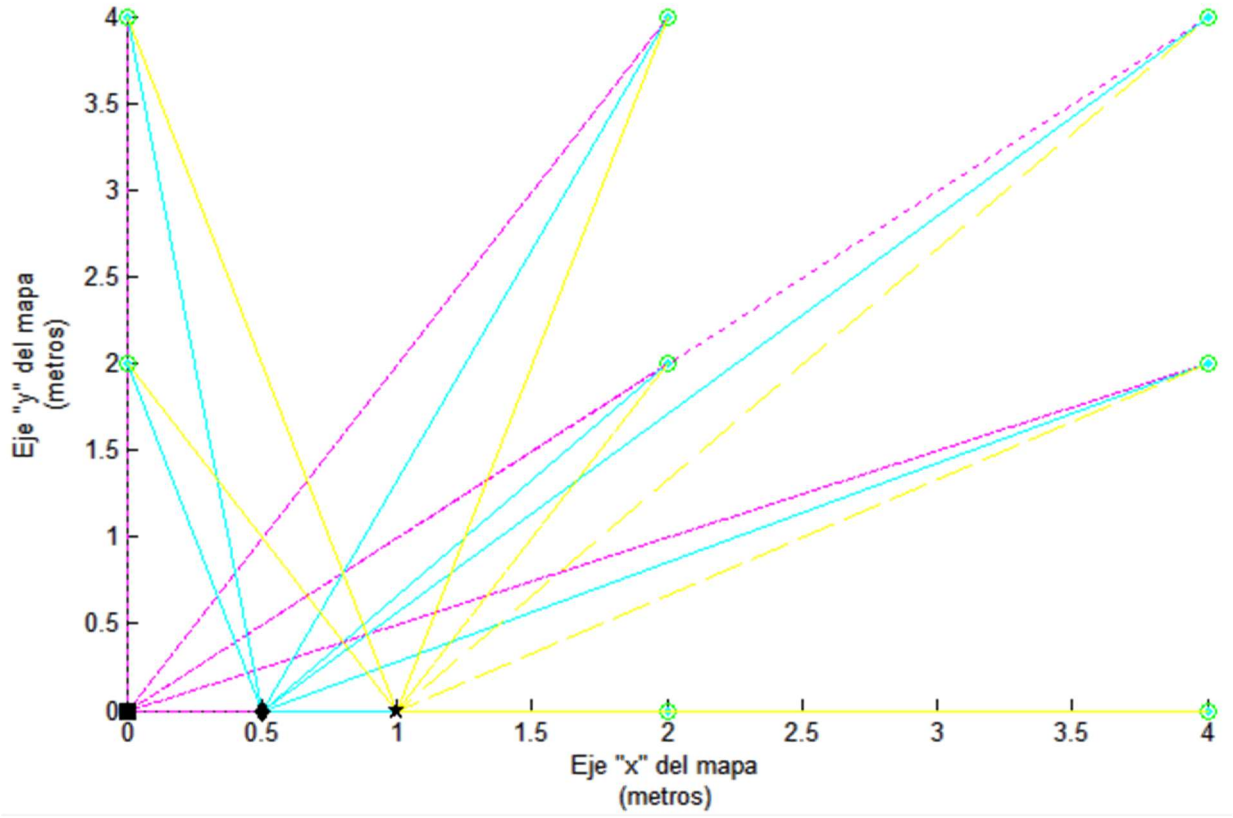

Figura 4-9. Distancias entre 3 puntos de prueba y readers.

La expresión de distribución "$P(n_j|d_j)$" dará como resultado valores más próximos a 1 cuanto más parecida sea la distancia entre el reader *j* y la posición de prueba a la distancia real entre el reader *j* y la tag, al ser coherentes los valores "n_j" para dichas distancias.

Para obtener el valor del punto de prueba que se le asigna como posición estimada a una tag, el procedimiento se basa en escoger el punto de prueba donde la función de verosimilitud toma su mayor valor, suponiendo esta posición como la mejor aproximación.

4.2.4.1 Estimación de la posición de la tag

Todo el procedimiento comentado anteriormente se desarrolla en lenguaje M en las funciones que se encuentran en "*Anexos – Métodos de estimación – Método de máxima verosimilitud – Estimación posición de la tag"*. En este apartado se hace uso de cuatro funciones:

- *aproxtag_4metodo:* el principal cometido de esta función es asignar como posición estimada de la tag las coordenadas donde se da el máximo valor de la función de verosimilitud. Para ello, hace llamadas a la función *estimacionmaxveros* aportándole los posibles puntos de prueba. Esto se hace dentro de un bucle *for,* donde se recorre el mapa desde la posición (0,0) hasta el final del mismo dando saltos de valor *precision* entre las coordenadas a tomar. Para el ejemplo (4-5) se ha tomado una *precision* de 0.5 y, como se puede ver en la Figura [4-9], los puntos de prueba se han tomado en las posiciones (0,0) metros, (0.5,0) metros y (1,0) metros.

  La función "*aproxtag_4metodo*" recibe como entrada el parámetro "*mapa*" que es el que representa las "n_j*"* de cada reader*,* la distancia entre los readers "*dist_rdrs",* "number_msgs_Tx" y "*precision"* que se desea tener en cuenta para la estimación de la tag.

  Como salida, la función ofrece las coordenadas de la tag estimada ("*xtag_aprox_4metodo*", "*ytag_aprox_4metodo*"), las coordenadas de los readers que reciben queries ("x_rdrsRx","y_rdrsRx") y *"emv_matriz"* que es la matriz que representa el mapa y donde se ha sustituido en cada posible punto de prueba el valor de la función de verosimilitud para el mismo.

- *estimacionmaxveros:* esta función es la codificación de la ecuación (4-11), donde los "$P(n_j|d_j)$" se calculan haciendo llamadas a la función "*betabinfunc".* Como salida, ofrece el parámetro "*emv"* a modo de resultado de la ecuación (4-11) para cada punto de prueba*.* Además, como parámetros de entrada se le pasan "*dist_rdrs", "number_msgs_Tx"* y "*mapa_acks"* ya que los necesita para la llamada a "*betabinfunc".*

- *betabinfunc:* aquí se encuentra codificada la ecuación (4-12). Devuelve como parámetro de salida "*betabinf",* que es "$P(n_j|d_j)$" para un punto de prueba dado. Las coordenadas de este punto se piden como parámetros de entrada ("*xtag_aprox", "ytag_aprox"*), además de la distancia entre readers "*dist_rdrs",* el número "*number_msgs_Tx"* y el "*mapa_acks*" del que tomar la "n_j*"* de cada reader.

  Su funcionamiento consiste en ir tomando las diferentes distancias "d_j*"* existentes entre la posición de estimación y cada reader gracias a la función *"calc_distancia"*como se dijo anteriormente. Para asegurar que el código sea robusto y de resultados coherentes, como el estudio [6] se ciñe a un rango de distancias 1.2 < "d_j*"* < 7.6 metros, se han formulado las siguientes condiciones:

  - Si "d_j*"* < 1.2 metros y "n_j*"* es igual a "*number_msgs_Tx",* entonces la probabilidad de que la posición de estimación esté muy próxima a la tag real es igual a 1.
  - Si por el contrario "d_j*"* < 1.2 metros y "n_j*"* es distinto de "*number_msgs_Tx",* entonces la probabilidad de que la posición de estimación esté muy próxima a la tag real es igual a 0, siendo muy improbable.
  - Otra situación "d_j*"* > 7.6 metros y "n_j*"* igual a 0*,* donde el factor j-ésimo es igual a 1, por lo que no contribuirá en la multiplicación de la ecuación (4-11) al ser el 1 un elemento neutro. Esto ocurrirá para toda una serie de puntos de prueba con "d_j*"* > 7.6 m.

- Si "d_j" > 7.6 metros y "n_j" distinto de 0, entonces la probabilidad de que la posición de estimación esté muy próxima a la tag real es igual a 0, ya que esta situación no presenta una coherencia realista en su resultado.

- *distbeta:* esta función ya ha sido presentada en la sección "*Puesta a punto",* dentro de este mismo capítulo, en la subsección "*Probabilidad de detección de un reader".*

4.2.4.2 Representación gráfica

La función "*cuarto_metodo"* realiza la representación gráfica del resultado obtenido en "*aproxtag_4metodo"* y devuelve como salida el error cuadrático (square error - *se*) cometido en la estimación de la posición de la tag respecto de la posición de la tag real *"se4metodo",* además del mapa donde se incluyen los valores de la función de verosimilitud *"emv_matriz"*. Esta función se puede consultar en "*Anexos – Métodos de estimación – Método de máxima verosimilitud – Representación gráfica – cuarto_metodo"*.

Como en los métodos anteriores, lo que se representa son los readers distribuidos en un mapa y cuáles de estos readers reciben queries de vuelta. Además, se señalan las posiciones tanto de la tag real como de la estimada para ver el error que se ha comentido en la predicción de la posición de la tag.

Para poder realizar estos cálculos, la función "cuarto*_metodo"* requiere como parámetros de entrada el mapa donde se distribuyen los "*n"* ("*mapa_acks"*), *"dist_rdrs",* las coordenadas de la tag real (*"xtag_real", "ytag_real"*), el *"number_msgs_Tx"* y la "*precision"* que se desea en las medidas.

Para mostrar un ejemplo visual al lector se expone el Ejemplo [4-6].

Ejemplo 4–6. *Para un mapa de dimensiones 10x10 metros y una separación entre readers de 5 metros, donde se sitúa una tag en la posición (3,7) metros y los readers emiten 100 solicitudes de localización la estimación mediante la aproximación al método de máxima verosimilitud resulta como en la Figura [4-10], que sigue el mapa_acks del Ejemplo [4-1]*

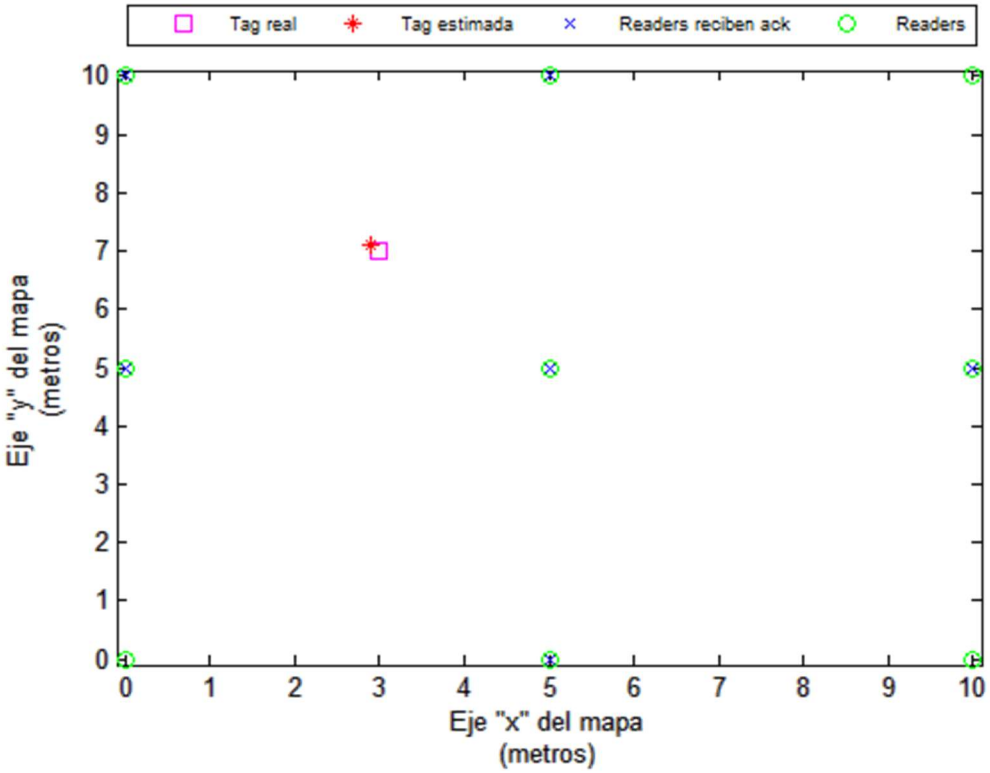

Figura 4-10. Ejemplo aproximación al método de máxima verosimilitud.

4.2.5 Comparativa métodos de estimación

Una vez expuestos los cuatro métodos de estimación, se procede a agruparlos para poder tener una mejor visualización de los mismos y poder hacer una comparativa rápida de los resultados que han dado cada uno.

4.2.5.1 Representación gráfica

Como se puede ver en "*Anexos – Métodos de estimación – Representación todos los métodos de estimación de la posición – metodos"*, "*métodos"* es la función encargada de llamar a los cuatro métodos de estimación de la localización, proporcionándoles los parámetros que necesiten y ofrecerles un título dentro de una misma figura. Esto es posible porque, dentro de cada método, antes de la representación de cada uno se ha añadido el comando *subplot*, asignándole así a cada uno una posición en la figura. Para mejor entendimiento, se expone el ejemplo (4-7).

En cuanto a los parámetros de salida, esta función devuelve todos los errores cuadráticos (square errors - *se*) proporcionados por los distintos métodos (*"se1metodo", "se2metodo", "se3metodo"* y *"se4metodo"*), ya que se hará uso de ellos posteriormente. Además, ofrece como salida la matriz de estimación de máxima verosimilitud para su representación en otra función superior a ésta.

Ejemplo 4–7. *Para un mapa de dimensiones 10x10 metros y una separación entre readers de 5 metros, donde se sitúa una tag en la posición (3,7) metros, los readers emiten 100 queries y se exige una precisión de 0.1metros, la estimación de cada método resulta como en la Figura [4-11], que sigue el mapa_acks del Ejemplo [4-1].*

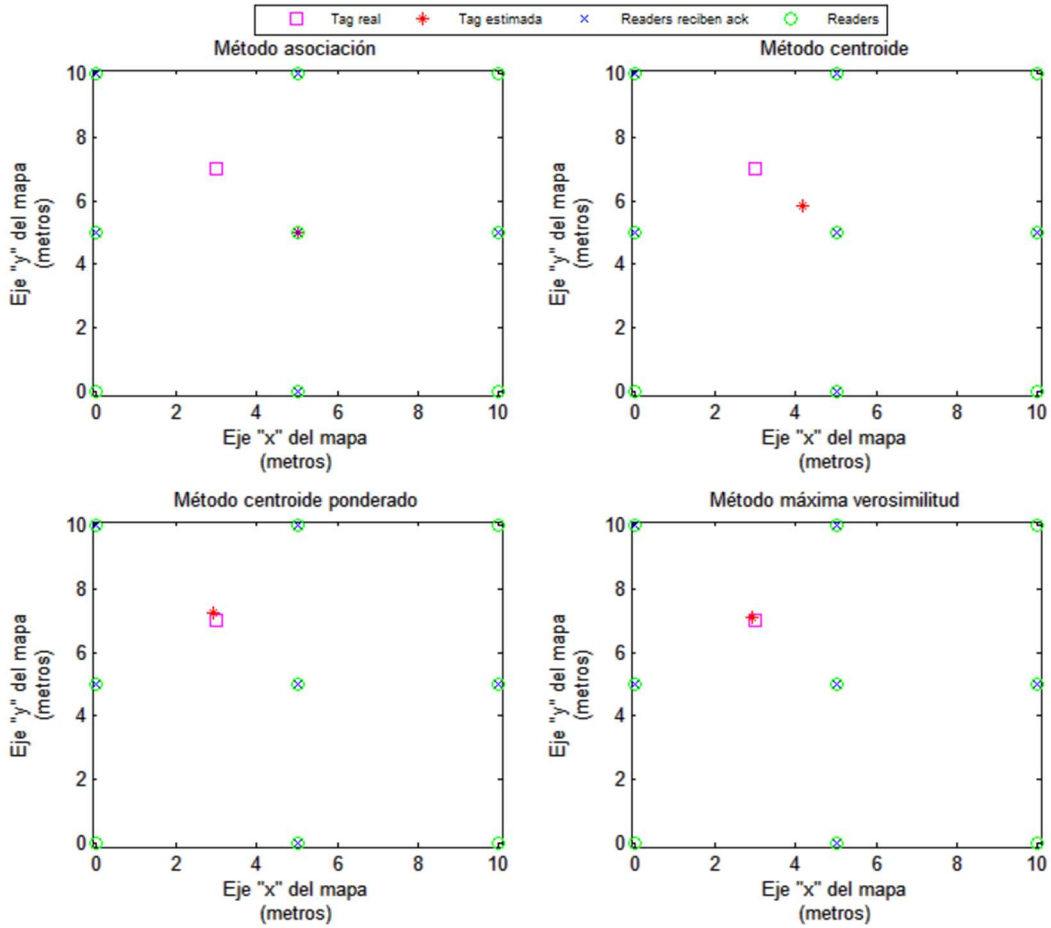

Figura 4-11. Comparativa visual métodos de estimación.

4.3 Variación de condiciones de entorno y errores de estimación

Para poder valorar realmente la fiabilidad de los métodos de estimación empleados, se someten a varias pruebas donde varían los parámetros que reciben como entrada.

4.3.1 Variación distancia entre readers

Para la modificación de los parámetros de entrada de la función "*métodos"* se ha creado la función "*test_dimensionfija*", la cual se puede ver en "*Anexos –Variación condiciones de entorno y errores de estimación de localización – Distancia entre readers variable".* Ésta recibe como parámetros de entrada los que pueden ser modificados para especificar las condiciones deseadas:

- *xmap, ymap:* ancho y largo del mapa o habitación donde queremos desarrollar la simulación.
- *number_msgs_Tx:* también conocido como "*N"* en este documento. Es el número total de queries emitidos por cada reader.
- *precision:* se especifica para la utilización en la aproximación al método de máxima verosimilitud vista en la sección anterior. Simboliza la precisión que deseamos alcanzar en la estimación de la tag dentro de dicho método.
- *dist_rdrs:* este es el más importante y el que sufre más modificación, ya que en una llamada a la función "*test_dimensionfija"* la función "*métodos"* se llevará a cabo tantas veces como valores tenga este parámetro, una iteración del mismo para cada distancia entre readers.

Otro parámetro importante es la posición de la tag real, la cual no se pasa como parámetro de entrada ya que se desea que sea tomada de forma aleatoria en cada llamada a esta función. En el código se le denominan "*xtag_real"* y *"ytag_real"* a las coordenadas donde se sitúa dicha tag dentro del mapa. Se emplea para poder hallar el error que se ha cometido en la predicción de la tag estimada en cada método, los cuales se proporcionan como parámetros de salida.

Pese a que se tomen valores aleatorios, también pueden prefijarse a un lugar específico fácilmente.

El objetivo de "*test_dimensionfija*" es que el usuario pueda realizar un estudio visual de los resultados obtenidos para diferentes distancias entre readers para un mapa con unas dimensiones fijas.

4.3.2 Errores de estimación en los distintos métodos

Para medir la fiabilidad de los métodos expuestos en el proyecto y como principal objetivo del mismo, se procede a mostrar gráficamente el error de estimación de la posición de la tag que se ha producido en cada método. Para tal fin se emplea la raíz del error cuadrático medio (RMSE: Root Mean Square Error), el cual se basa en la fórmula (4-14), descomponiendo el error cuadrático en la ecuación (4-15).

$$RMSE = \sqrt{\frac{1}{M}\sum_{k=1}^{M} SE_k} \tag{4–14}$$

$$SE_k = \left(||\hat{\underline{z}}_k - \underline{z}_k||\right)^2 \tag{4–15}$$

Para realizar el cálculo de este RMSE, se han empleado dos sencillas funciones "*square_error"* y *"rmse"* que pueden consultarse en "*Anexos –Variación condiciones de entorno y errores de estimación de localización – Errores de estimación".* En la primera, se implementa ecuación (4-15) en lenguaje M. La función "*square_error"* es llamada dentro de cada función que implementa a un método de estimación, es por ello que todas ellas lo tenían como parámetro de salida.

Por otro lado, el parámetro de entrada de la función "*rmse"* es el error cuadrático "*se".* Para ver dónde se emplea la llamada a dicha función se explica en la siguiente sección "*RMSE para distintos métodos y distancias entre readers".*

4.3.3 RMSE para distintos métodos y distancias entre readers

Para poder realizar el valor medio dentro del RMSE se debe tener varios valores a los que realizar la media aritmética, es decir, tener varias estimaciones de tags dentro de un mismo método con las mismas condiciones de entorno. Para hacer esto posible, se ha creado la función "*resultados_test",* la cual es la encargada de realizar un número de iteraciones ("*iteraciones"*), prefijado antes de la simulación, de la función "*test_dimensionfija*", lo que supone la toma de tantas posiciones aleatorias distintas de la tag real como el número "*iteraciones"*.

Así pues, en cada iteración se obtienen los vectores de los errores cuadráticos de cada método de estimación, correspondiendo cada término del vector a una distancia entre readers. Tras la obtención de los mismos, se agrupan en matrices propias de cada método, resultándonos cuatro matrices como la mostrada para el primer método (método de asociación que devuelve *se1*) en la Figura (4-12). Las filas presentan el *se* obtenido en una iteración y en las columnas el valor propio del mismo para cada distancia entre readers. Esta matriz se le pasa como parámetro a la función *"rmse"* que le calcula la media y raíz, obteniendo una matriz fila donde se han promediado los valores para las diferentes distancias entre readers, siendo este el parámetro a analizar en las gráficas mostradas en el siguiente capítulo *Resultados.*

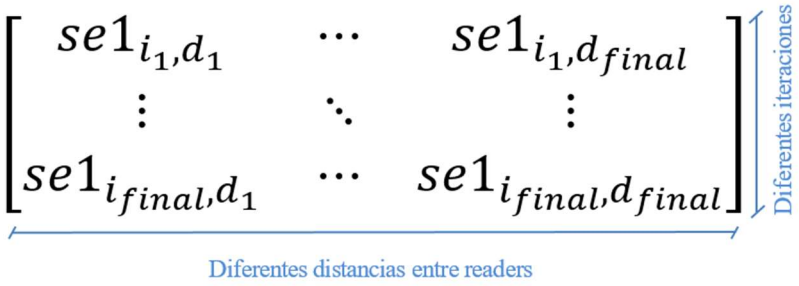

Figura 4-12. Matriz "square error" para primer método en *resultados_test*.

5 RESULTADOS

El éxito es el resultado de la perfección, trabajo duro,
aprender del fracaso, lealtad y persistencia.

- Colin Powell -

En el presente capítulo se dispondrán los resultados finales del diseño expuesto anteriormente. De este modo, se realizará un análisis de las situaciones óptimas en las que los resultados obtenidos son lo suficientemente exactos y puedan desarrollarse favorablemente en la vida real.

Para tal propósito, se van a realizar las observaciones en base a los resultados mostrados por la función "*resultados_test",* definida en el capítulo anterior de este proyecto.

Antes de realizar una simulación de dicha función, es necesario prefijar los parámetros que definirán el entorno de trabajo. Éstos son cinco y de ellos dependerá el tiempo empleado en cada simulación. A continuación, se comentan estos parámetros:

- El número de iteraciones que se desee realizar del modelo, denominado como "*iteraciones"* dentro de "*resultados_test".* Cada iteración lo que supone es un cambio en la posición de la tag, que se le da de forma aleatoria. Así que, como es lógico, a mayor número de iteraciones, más veces se deberá repetir el cómputo de los métodos para todas las posibles distancias entre readers, tardando más la obtención de los resultados. Sin embargo, a mayor número de pruebas realizadas, mejor serán los resultados al obtener un mayor muestrario de resultados al que realizar la media aritmética. Por ejemplo, 100 posiciones distintas de la tag dentro del mapa es un número suficientemente grande del que obtener resultados fiables para el análisis.
- Las dimensiones del mapa ("*xmap","ymap"*) donde se sitúen los readers. A mayor extensión, mayor carga computacional.
- La precisión tomada para analizar la aproximación al método de máxima verosimilitud, denominada "*precision",* supondrá mayor carga de trabajo al programa cuanto menor sea esta. Esto es debido a que si se desea que los resultados sean más exactos, dicho método realizará más operaciones para el cálculo.
- El número de diferentes distancias entre readers también supondrá tiempo añadido por cada elemento de la matriz "*dist_readers".* Cabe decir que cuanto mayor sea la distancia entre readers, menor será el número de readers distribuidos en el mapa, agilizando los tiempos de simulación.
- Por último, el parámetro "*N",* o como se muestra en el código *"number_msgs_Tx".* Este número debe ser lo suficientemente grande para que haya un gran número de muestras a tener en cuenta, pero sin pasar el límite de 100 queries, ya que el propio Matlab es incapaz de realizar el cómputo debido a sus limitaciones numéricas. Un número óptimo es $N = 50$ queries.

Así pues, en este capítulo se mostrarán los resultados obtenidos en combinaciones diferentes de estos parámetros y se analizarán cuáles son las situaciones más óptimas.

Las gráficas resultantes pueden clasificarse en tres, que son las que mejor muestran una visión de la situación:

- Mapas de dimensiones prefijadas, mostrando los resultados de los distintos métodos en función de la distancia entre readers.
- Valores de la función de verosimilitud de la aproximación al método de máxima verosimilitud, para comprobar los resultados de dicho método.
- RMSE producido en los distintos métodos de localización en función de la distancia entre readers

Estos tres tipos serán presentados y analizados en cada ejemplo de los que se exponen a continuación.

5.1 Primer ejemplo

A modo de primer ejemplo, se toman unos parámetros que requieren de poco tiempo de cómputo para una primera presentación.

Ejemplo 5–1. *La tag es situada únicamente en 3 posiciones distintas en un pequeño mapa de dimensiones 8x8 metros. Las separaciones entre readers tomadas son 2, 4 y 8 metros. Los readers emiten 20 queries y se exige una precisión de 1 metro.*

Para empezar, se muestran los mapas de las Figuras [5-1], [5-2] y [5-3], los cuales se diferencian en las distancias que separan los readers. Éstas son un ejemplo para una tag en concreto, situada aleatoriamente en (2.742, 6.656) metros. En las gráficas se observan cuatro mapas, representantes de los resultados de cada método de estimación. En la leyenda se denotan los cuatro elementos principales a distinguir en ellos.

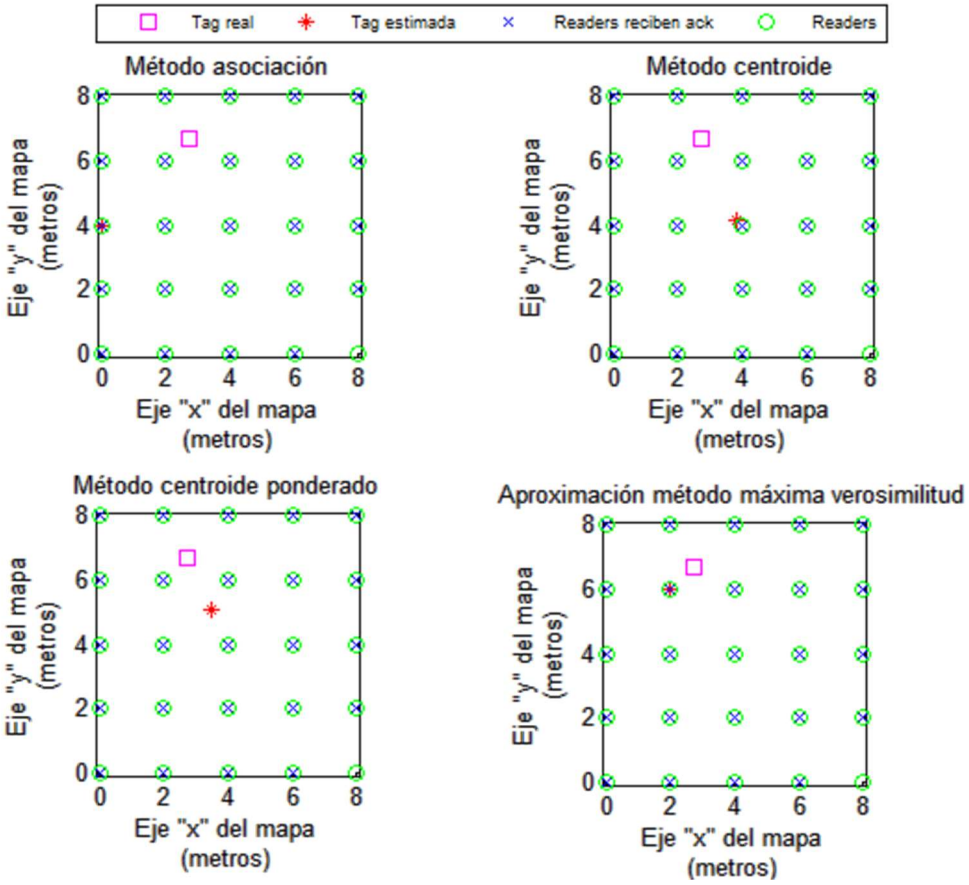

Figura 5-1. Primer ejemplo. Mapas, distancia entre readers 2 metros.

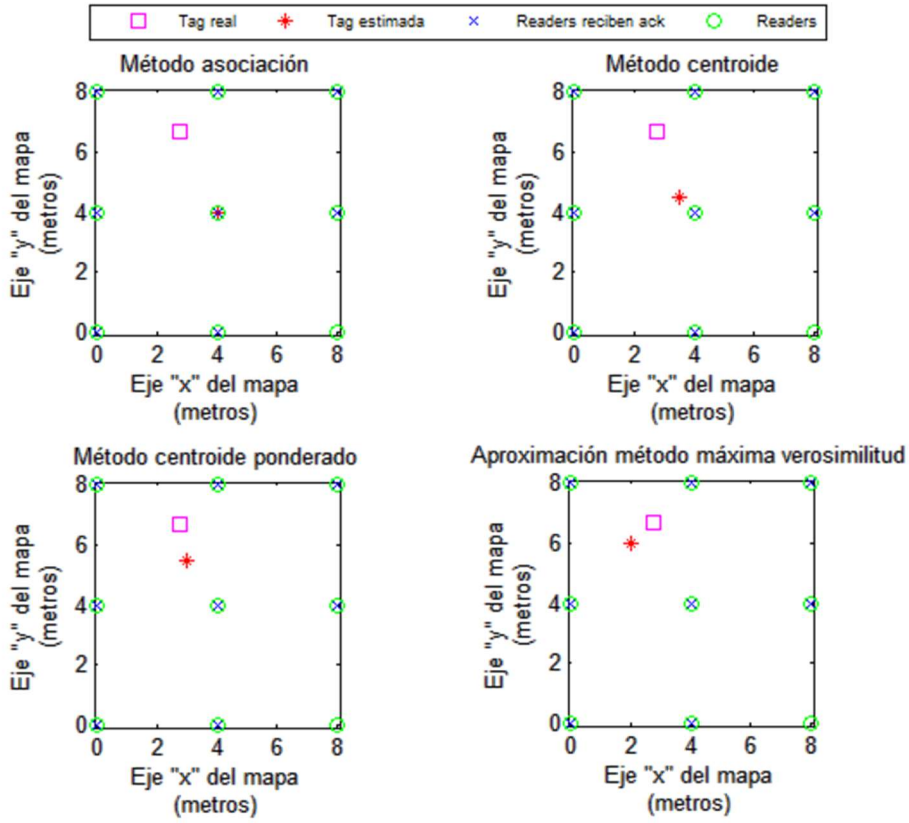

Figura 5-2. Primer ejemplo. Mapas, distancia entre readers 4 metros.

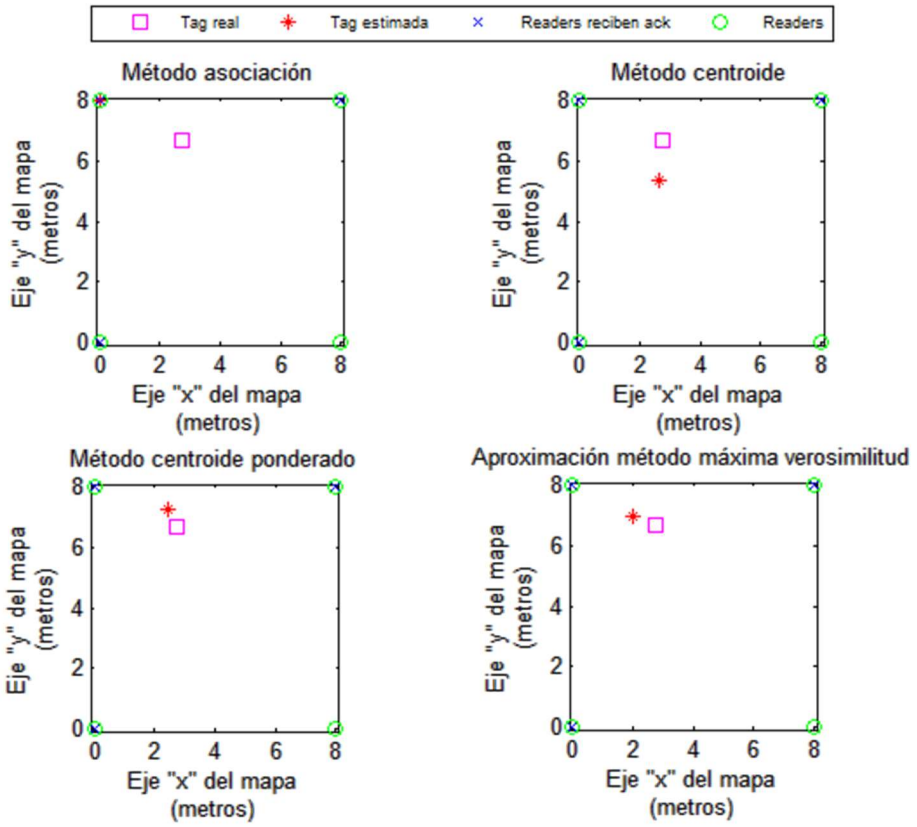

Figura 5-3. Primer ejemplo. Mapas, distancia entre readers 8 metros.

Para poder visualizar cómo ha procedido el método de máxima verosimilitud para estimar la mejor posición aproximada de la tag, se presentan las gráficas correspondientes a cada figura anterior, en función de la separación entre readers, de los valores que toma la función de verosimilitud.

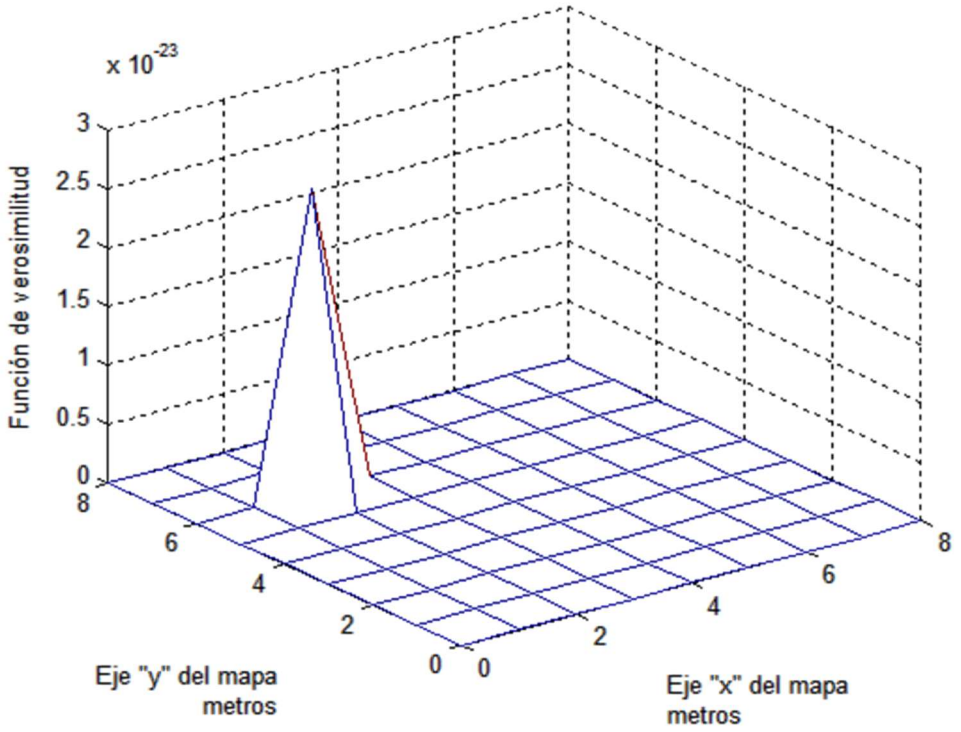

Figura 5-4. Primer ejemplo. Función verosimilitud, distancia entre readers 2 metros.

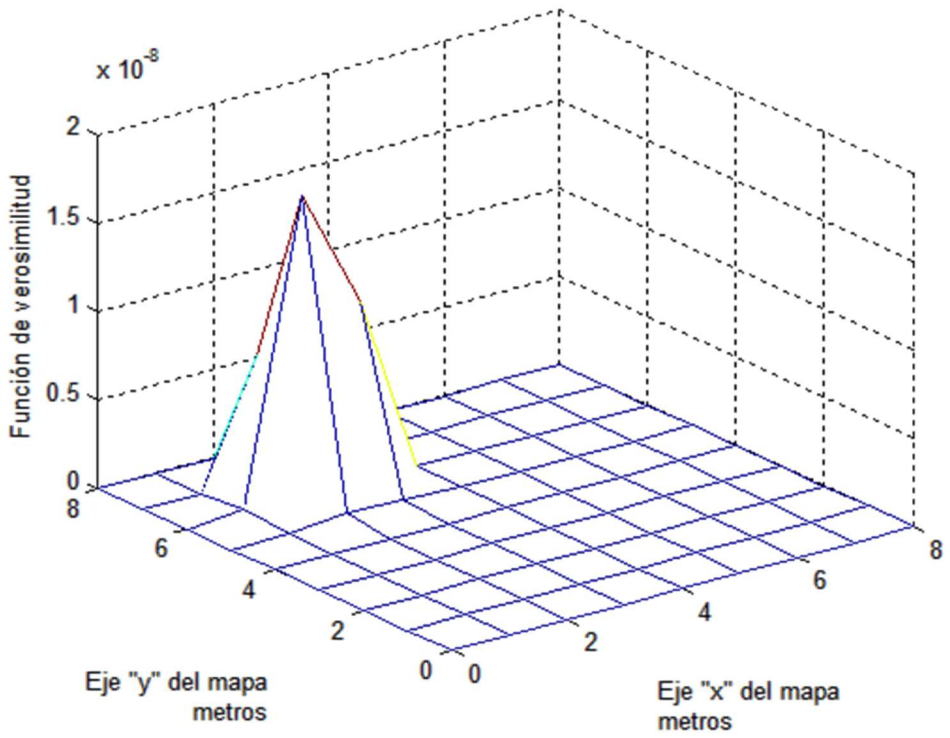

Figura 5-5. Primer ejemplo. Función verosimilitud, distancia entre readers 4 metros.

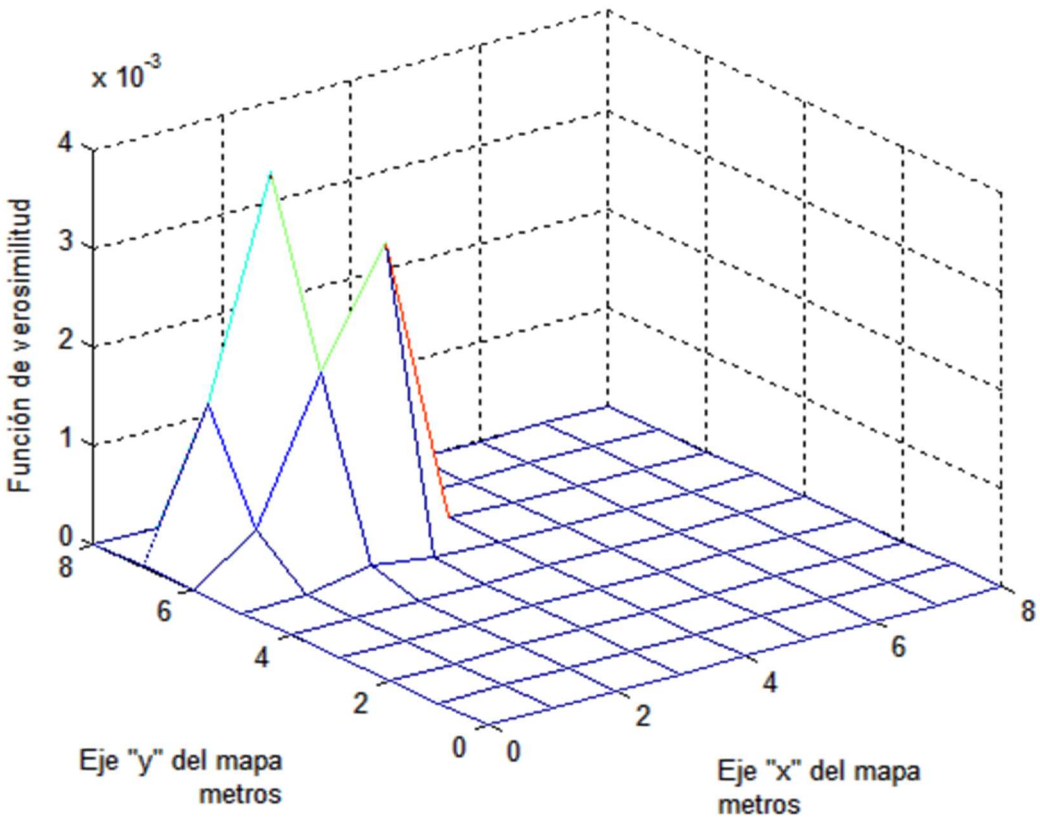

Figura 5-6. Primer ejemplo. Función verosimilitud, distancia entre readers 8 metros.

La Figura [5-4] presenta los valores tomados por la función de verosimilitud correspondiente al mapa de la Figura [5-1], con una separación entre readers de 2 metros. El grid está dividido en recuadros de 1x1 metro, ya que es la precisión que se ha tomado. Lo que muestra es una "pirámide" que tiene su pico en la posición (2, 6) metros. Como se puede ver en la Figura [5-1], la estimación de la tag se encuentra efectivamente en dichas coordenadas.

Respecto a la figura [5-5], correspondiente a la Figura [5-2], con la separación entre readers de 4 metros, ya no se obtiene una pirámide, sino que los valores cercanos al pico anterior también toman cierta relevancia. Esto es debido a que, a mayor separación entre readers, éstos reciben menos queries o ninguno, lo que hace que la función de verosimilitud no muestre un máximo que destaque tanto como anteriormente para menos distancia entre readers.

En cuanto a la Figura [5-6] correspondiente a una separación entre readers de 8 metros de la Figura [5-3], se observan varios picos considerables. Esto ocurre, como se ha explicado anteriormente, por decrecer el número de readers, que en este caso sólo se han utilizado 4, uno en cada esquina del mapa. Como el rango de distancias entre tag y reader del experimento [6] llega hasta los 7.6 metros, es probable que con sólo 4 readers en un grid de 8x8 metros alguno no reciba queries de vuelta. Como se observa en la Figura [5-3], el reader situado en la esquina inferior derecha no recibe ningún query, mientras que los demás sí. De este modo, los cálculos pueden oscilar de mayor manera al tener pocos readers que hayan leído.

Así se puede ver que los resultados dependerán en gran medida de la separación entre readers.

5.2 Segundo ejemplo

Se seguirá con un ejemplo donde el mapa es el mismo pero se toman más posiciones de tag para tener mayor número de pruebas realizadas, además se toma una mejor precisión y mayor número de queries transmitidos por cada reader, para así mejorar los resultados.

Ejemplo 5–2. *La tag es situada en 100 posiciones distintas en un mapa de dimensiones 8x8 metros. Las separaciones entre readers tomadas son 2, 4 y 8 metros. Los readers emiten 50 queries y se exige una precisión de 0.5 metros.*

Como anteriormente, se toma como muestra una tag para representar los mapas de las Figuras [5-7], [5-8] y [5-9], los cuales se diferencian en las distancias que separan los readers.

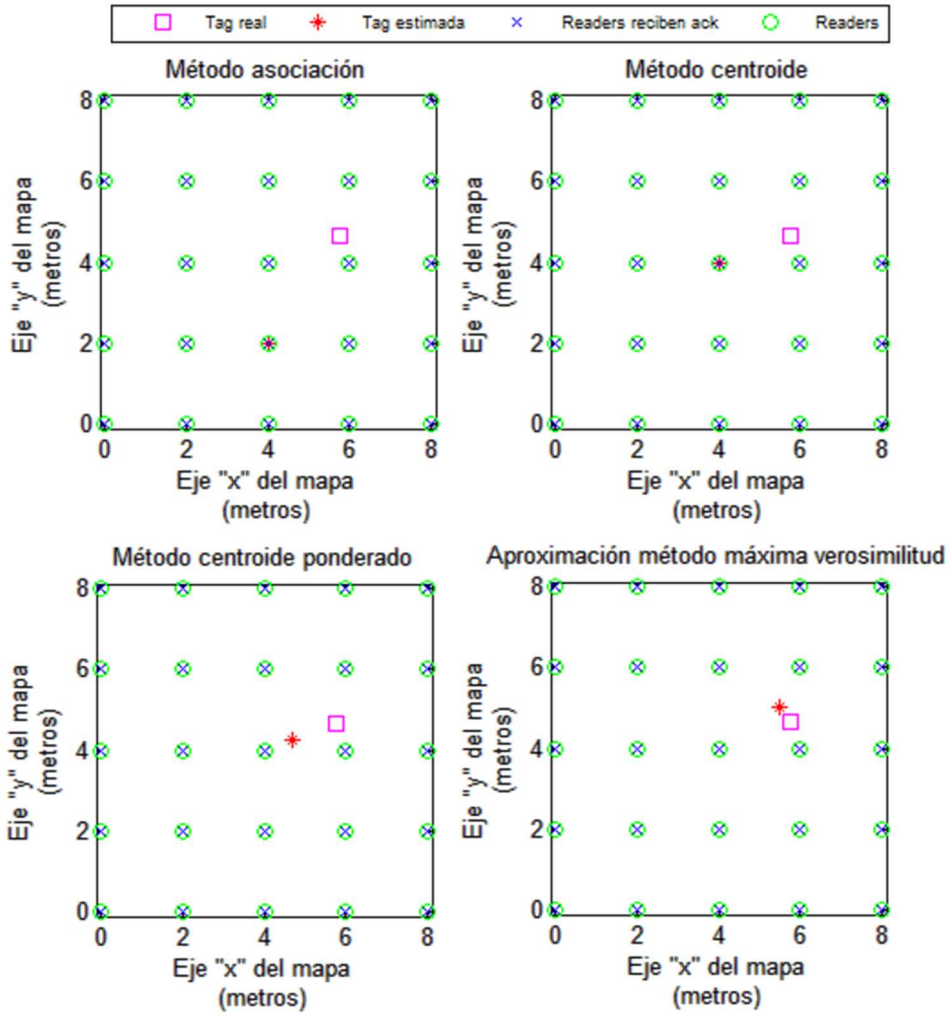

Figura 5-7. Segundo ejemplo. Mapas, distancia entre readers 2 metros.

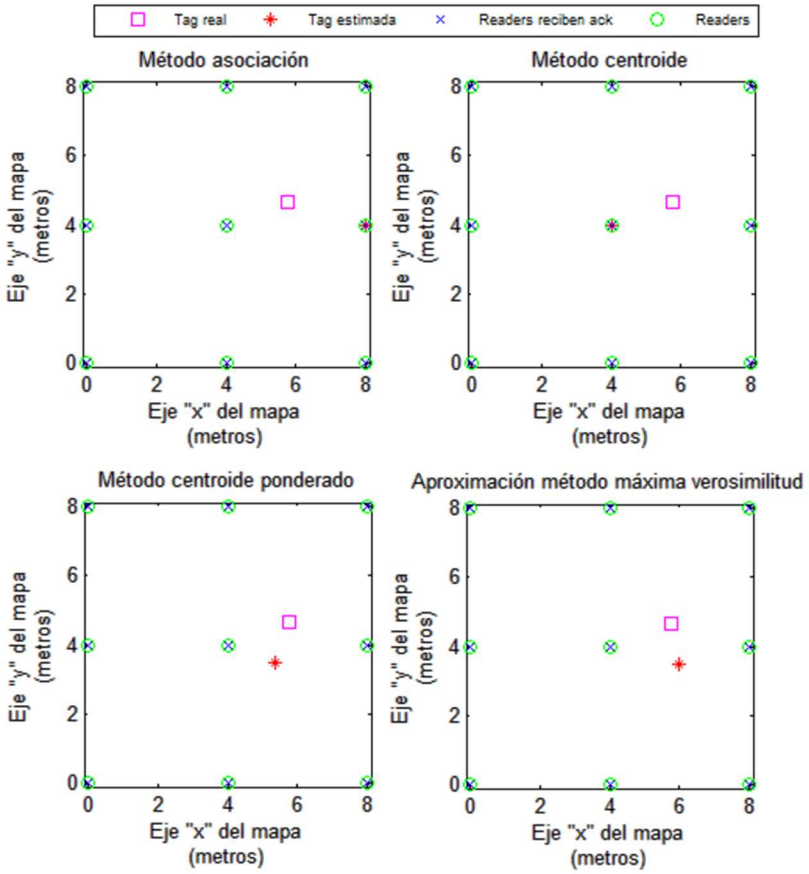

Figura 5-8. Segundo ejemplo. Mapas, distancia entre readers 4 metros.

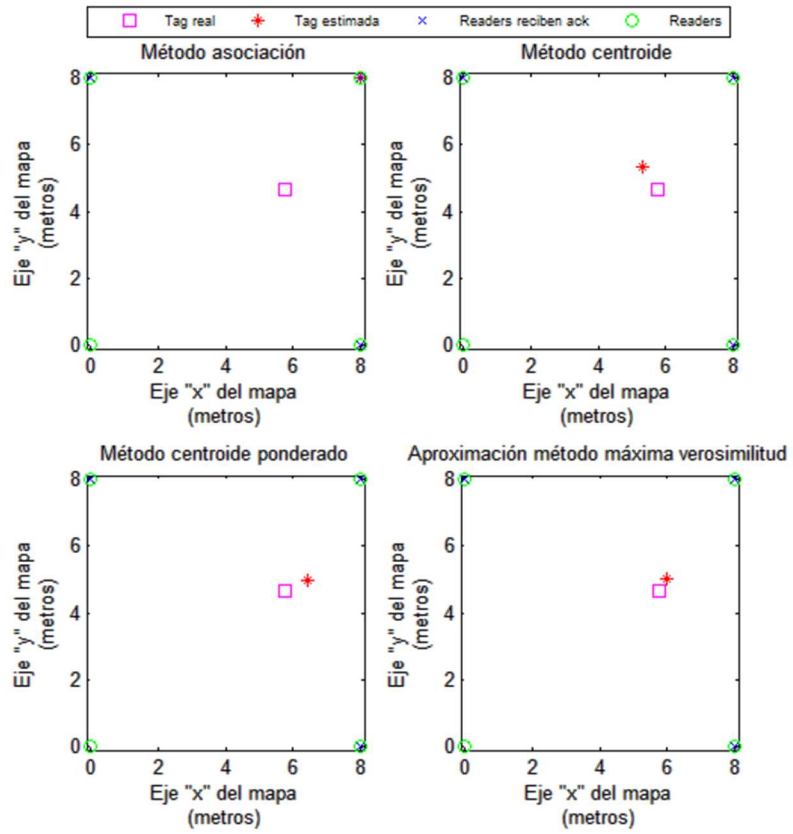

Figura 5-9. Segundo ejemplo. Mapas, distancia entre readers 8 metros.

En este caso, la precisión que se ha tomado para la aproximación al método de máxima verosimilitud es de 0.5 metros. Por este hecho, en las gráficas de los valores de la función de verosimilitud que se muestran en las Figuras [5-10], [5-11] y [5-12], pueden verse más picos que en el ejemplo anterior al estar más juntos los puntos de prueba y ser un grid que consta de cuadros de 0.5x0.5 metros.

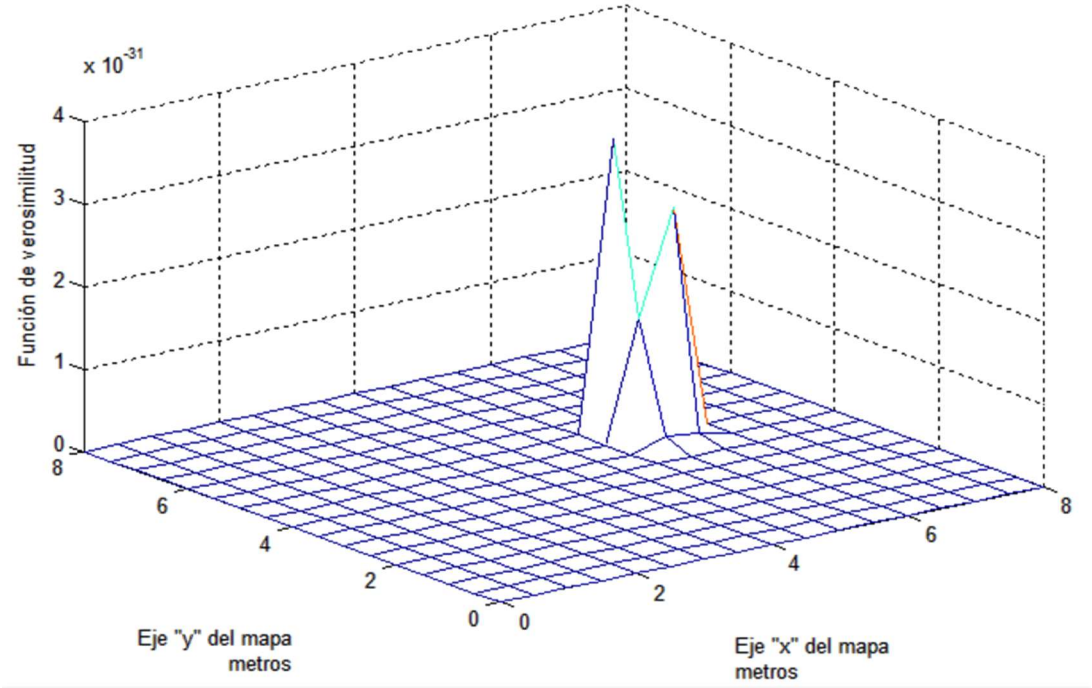

Figura 5-10. Segundo ejemplo. Función verosimilitud, distancia entre readers 2 metros

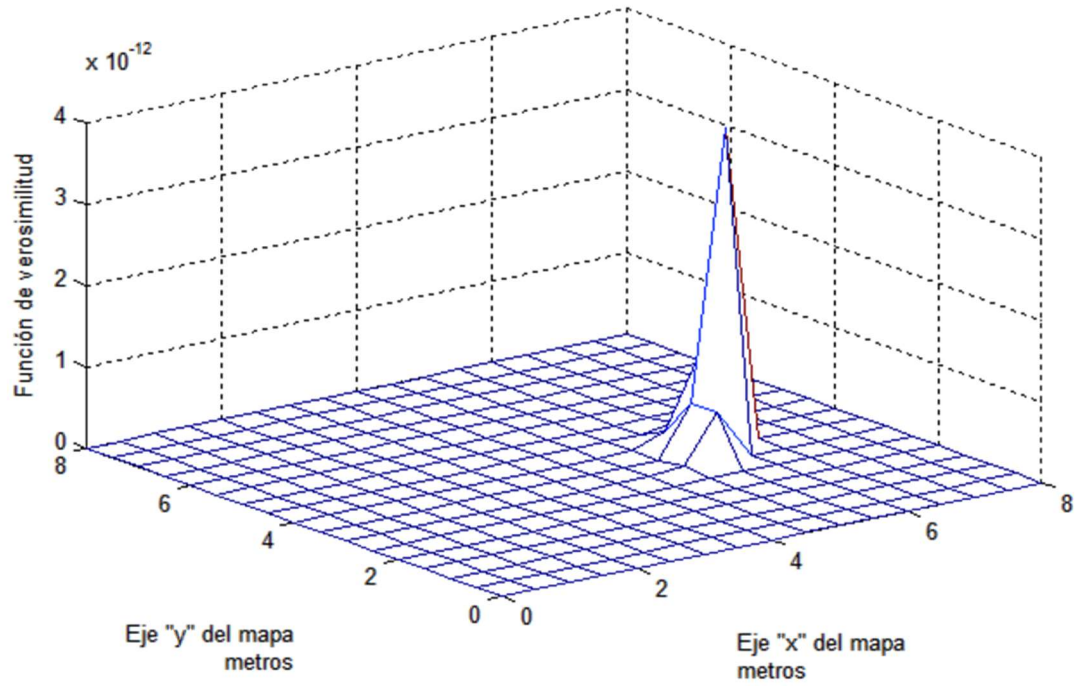

Figura 5-11. Segundo ejemplo. Función verosimilitud, distancia entre readers 4 metros.

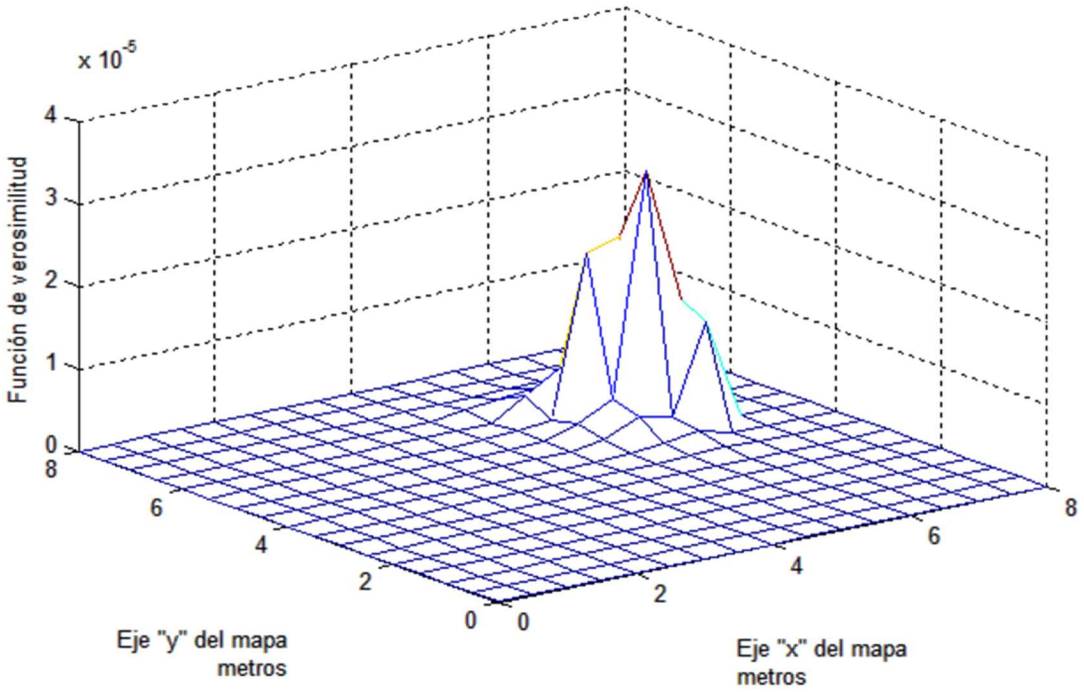

Figura 5-12. Segundo ejemplo. Función verosimilitud, distancia entre readers 8 metros.

En este ejemplo, al contar con más muestras al haber tomado más iteraciones que en el ejemplo anterior, se puede representar el RMSE (raíz del error cuadrático medio) para ver los errores que se han producido en cada método en este ejemplo. En el primer ejemplo esta gráfica no se mostró ya que no se disponían de muestras suficientes a las que poder realizar la media aritmética. Así pues, se muestra en la Figura [5-13] el RMSE en metros producido en función de la distancia entre readers.

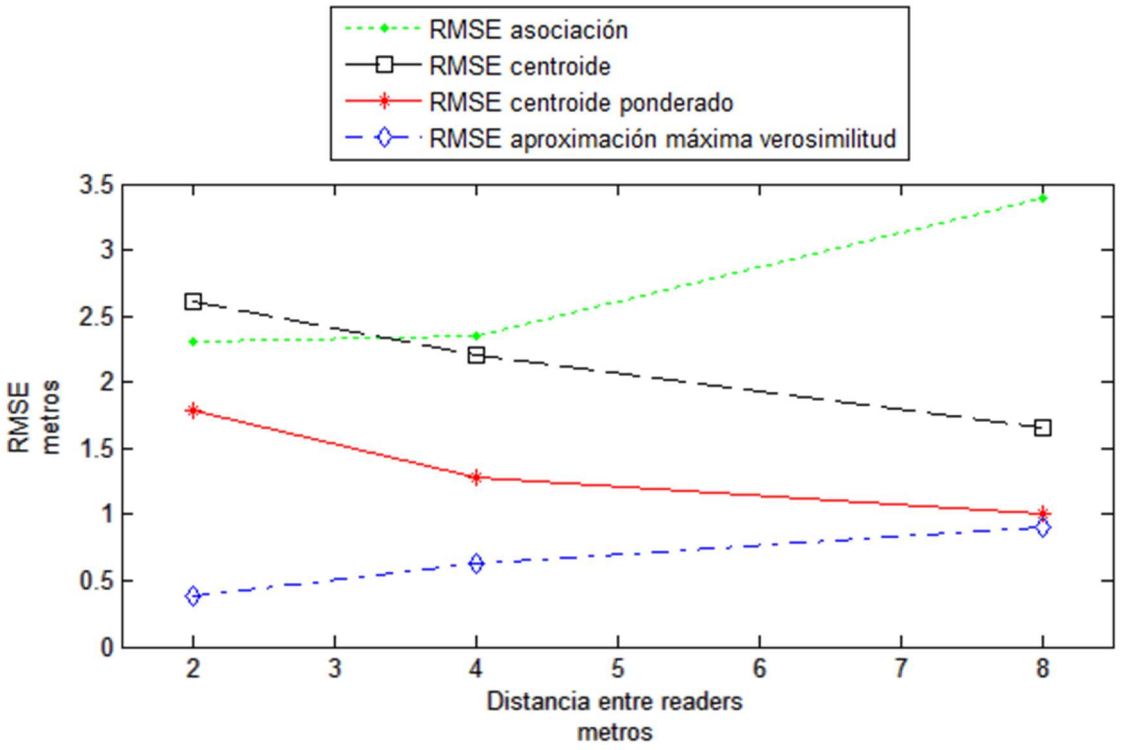

Figura 5-13. RMSE segundo ejemplo.

En términos generales se puede ver que el método que presenta mayor RMSE es el método de asociación, mientras que la aproximación al método de máxima verosimilitud es el que obtiene valores más exactos, siendo el que da menor RMSE. Como era de esperar, entre ambas líneas se encuentran los métodos del centroide y del centroide ponderado, ofreciendo este último mejores resultados.

Analizando cada método, el método de asociación presenta una curva creciente cuanto mayor es la distancia entre readers, esto es así ya que al estar más alejados los readers, menos son las posibles posiciones que tome la tag para este método, pudiéndose alejar cada vez más de la posición real de la tag.

En cuanto a ambos métodos del centroide, se observa un comportamiento quizás distinto al esperado a priori. En lugar de crecer el error a mayor distancia entre readers, se produce un decrecimiento; esto es debido a que, como se ha comentado anteriormente, a mayor separación de los readers menor es el número de readers que reciben queries. Por este motivo, en el caso de una distancia entre readers de 2 metros, el número de readers que reciben lecturas es mayor, por lo que la tag estimada se sitúa en el centro de estos. Para comprobar los queries recibidos por cada reader, se muestra en el Workspace de Matlab la matriz representante de dichas lecturas:

>> Mapa representante del número de acks que recibe cada reader:

mapa_lectores =

27	30	49	96	60
16	85	99	100	64
52	94	94	100	100
44	97	77	82	73
15	30	23	53	39

En cambio, para 8 metros de separación entre readers, únicamente reciben queries tres readers, desviándose la estimación de la tag hacia el cuadrante donde se encuentra la tag real, como se puede ver en la Figura []. Se comprueba:

>> Mapa representante del número de acks que recibe cada reader:

mapa_lectores =

36	74
0	46

Algo similar ocurre con el centroide ponderado, al haber menos readers, habrá algún reader que reciba un número de queries considerable, la ponderación de este reader hará que se acerque la tag estimada más a él. Si en su caso hay mayor concentración de readers, los más próximos a la tag recibirán más o menos el mismo número de queries de vuelta, por lo que tendrán ponderaciones similares y la tag estimada se situará aproximadamente en el centro de estos.

Así que, basándonos en este ejemplo, se puede decir que el método de máxima verosimilitud es el que muestra mejores resultados al presentar menor RMSE.

Como apunte importante, cabe decir que, aunque la precisión del grid se haya tomado de 0.5 metros, realmente será de la mitad de este el RMSE obtenido. Como se puede ver de manera gráfica en la Figura [5-14] para dos posibles tags, realmente la precisión que se tiene es la mitad de la precisión que se da, ya que si se encuentra la tag real entre dos posibles puntos de prueba, en teoría se tomaría el más cercano, que estará como máximo a 0.25 metros. De este modo, no se debe confundir la variable *precision* con los valores cercanos a la tag que puede tomar la tag estimada.

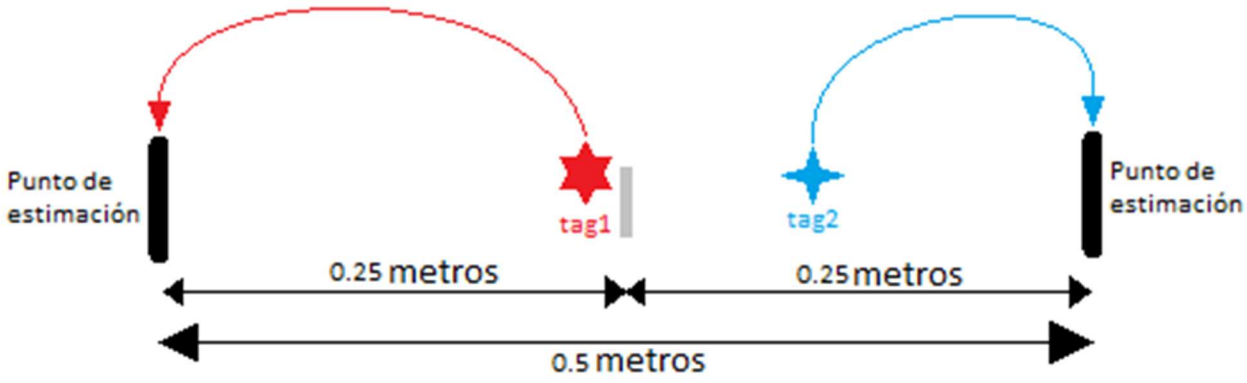

Figura 5-14. Interpretación precisión.

5.3 Tercer ejemplo

Ahora se verá, con un pequeño ejemplo, la gráfica de la función de verosimilitud más claramente al mejorar la precisión.

Ejemplo 5–3. *La tag se sitúa en 10 posiciones diferentes dentro de un grid de dimensiones 5x5 metros. Sólo se toma una separación entre readers de 5 metros. Los readers emiten 100 queries y se exige una precisión de 0.1 metros.*

Al haber exigido una precisión de 0.1 metros, se presupone que la estimación resultante será más exacta. Esto a su vez, requiere de un mayor número de repeticiones de los cálculos, por lo que el tiempo de simulación es mayor cuanto más fina sea la precisión.

Se toma como ejemplo una tag situada en (3.563, 1.815) metros y se obtienen las Figuras [5-15] y [5-16].

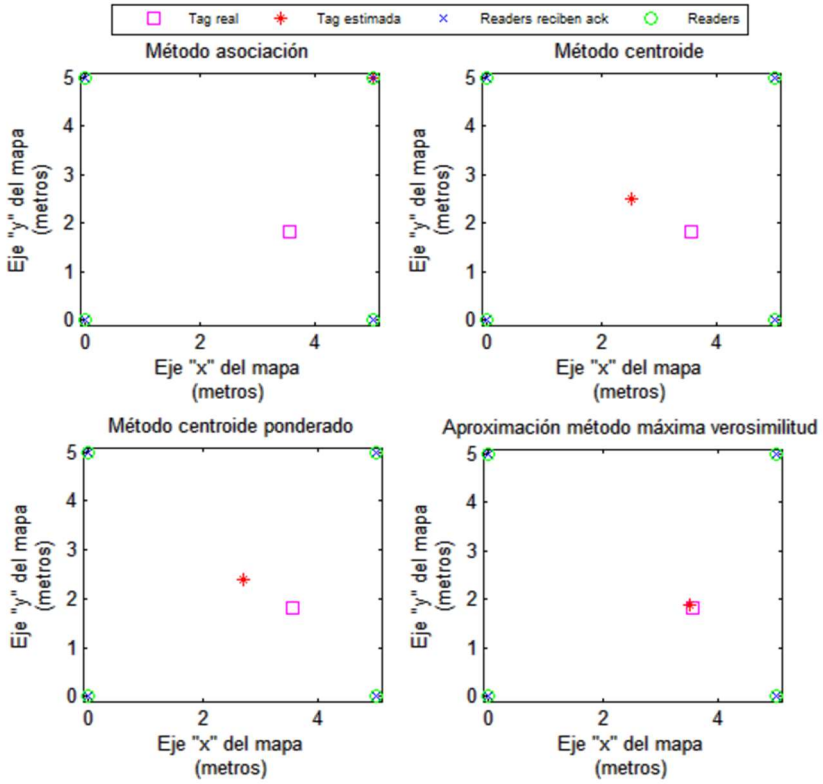

Figura 5-15. Tercer ejemplo. Mapas, distancia entre readers 5 metros.

Se puede observar en la Figura [5-16] que el grid se ha dividido en cuadros de 0.1 metros, por lo que se han renombrado las medidas de los ejes, tomando decímetros en lugar de metros. Esta curva es mucho más suave que las que se han visto en los ejemplos anteriores. En el caso de que las variables empleadas en el estudio fueran continuas como ocurre en la vida real, se formaría una campana uniforme. Sin embargo, no es posible implementar variables continuas en un sistema digital, pero se pueden realizar buenas aproximaciones de las mismas tomando un gran número de muestras. Esto es posible tomando un grid más fino gracias a tomar una precisión más ajustada.

En el caso de este trabajo la mayor precisión que se toma es la de 0.1 metros, mientras que la peor es la de 1 metro, para obtener unos resultados que sean válidos para localización en interiores.

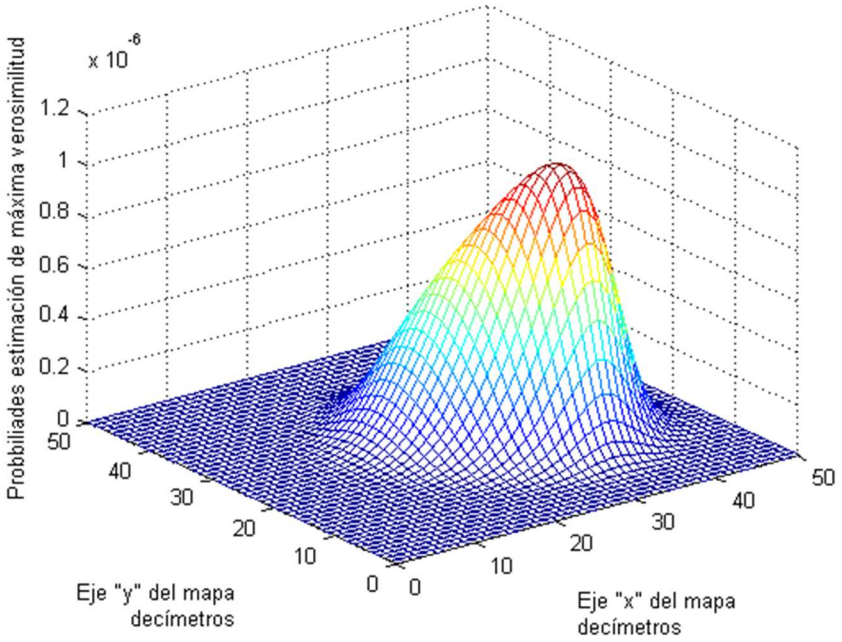

Figura 5-16. Tercer ejemplo. Función verosimilitud, distancia entre readers 5 metros.

Se ha realizado para dicha precisión la toma de 10 muestras, situando la tag real en 10 posiciones aleatorias en el mapa, y se ha representado el error producido por el método de aproximación a la máxima verosimilitud en la estimación de la tag, obteniendo así unos resultados preliminares de lo que resultaría el RMSE en esta situación dada una precisión de 0.1 metros.

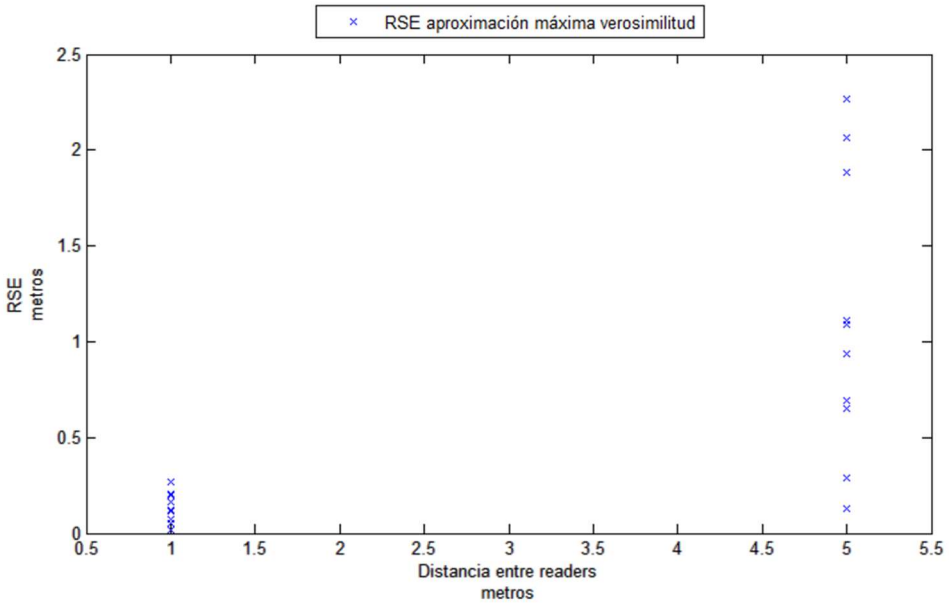

Figura 5-17. RSE tercer ejemplo.

Se observa en la Figura [5-17] que, pese a que uno puede pensar que si se exige tal precisión el RSE (raíz del error cuadrático) será mucho menor, ciertamente no es así para la separación de 5 metros en la mayoría de las muestras. Esto es debido a que influye en gran medida el número de readers que se encuentren en el grid. Para este caso, al ser un mapa de 5x5 metros, debido a esta separación, sólo se disponen de 4 readers cuando la separación es de 5 metros entre ellos, como se puede ver en la Figura [5-15]. Como ya se ha comentado, al tener un pequeño número de lectores, los cálculos sólo se pueden basar en la posición de estos cuatro y en el número de queries que le llega de vuelta a cada uno, haciéndolos más inexactos que si se tiene un gran número de readers. Sin embargo, se puede ver que para una separación de 1 metro entre readers el error producido, en la mayoría de las muestras, sí ronda el valor de la precisión que se pedía.

Con esto se demuestra que, pese a que se exija una precisión, ésta puede que no se cumpla para cualquier separación entre readers, sino que la exactitud de los resultados está fuertemente ligada a la separación entre readers.

5.4 Cuarto ejemplo

Para los últimos ejemplos de este capítulo se ha considerado un mapa de mayores dimensiones que pueda asemejarse mejor a lo que ocurriría en la vida real. Además, se ha tomado una precisión considerablemente buena y un número suficiente de diferentes distancias entre readers. En este caso en particular, se ha tenido en cuenta un gran número de muestras de tag para obtener unos buenos resultados de la gráfica RMSE.

Ejemplo 5–4. *La tag es situada en 120 posiciones distintas en un mapa de dimensiones 24x24 metros. Las separaciones entre readers tomadas son 2, 3, 4, 6 y 8 metros. Los readers emiten 50 queries y se exige una precisión de 0.5 metros.*

Como en los otros ejemplos, se toma como muestra una tag para representar los mapas de las Figuras [5-18], [5-19], [5-20], [5-21] y [5-22], correspondientes a las distintas distribuciones de los readers según la distancia de separación entre ellos.

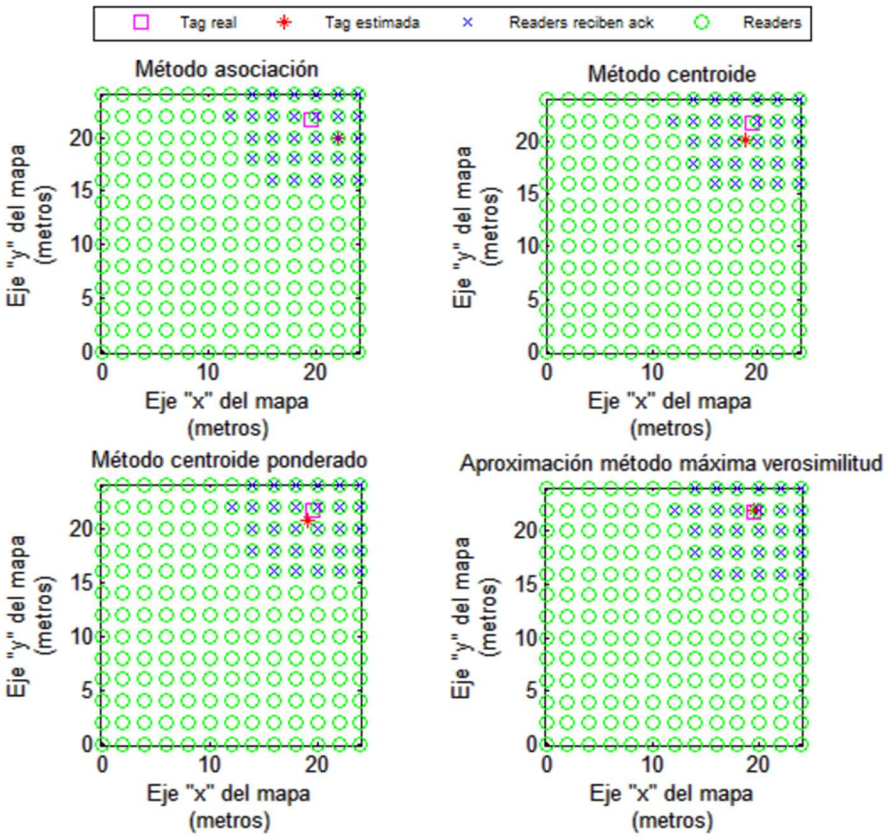

Figura 5-18. Cuarto ejemplo. Mapas, distancia entre readers 2 metros.

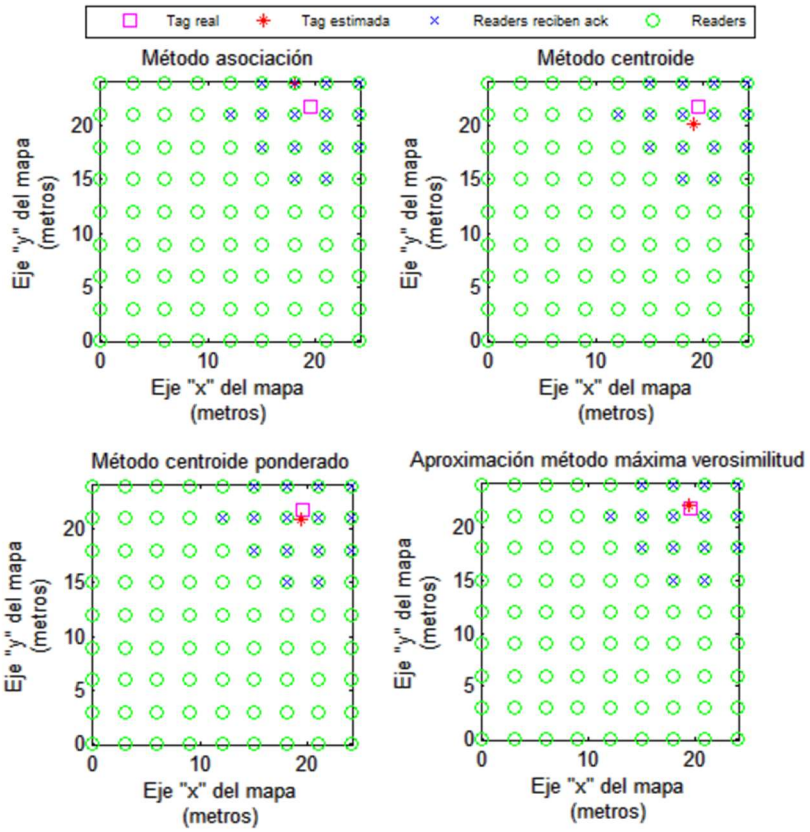

Figura 5-19. Cuarto ejemplo. Mapas, distancia entre readers 3 metros.

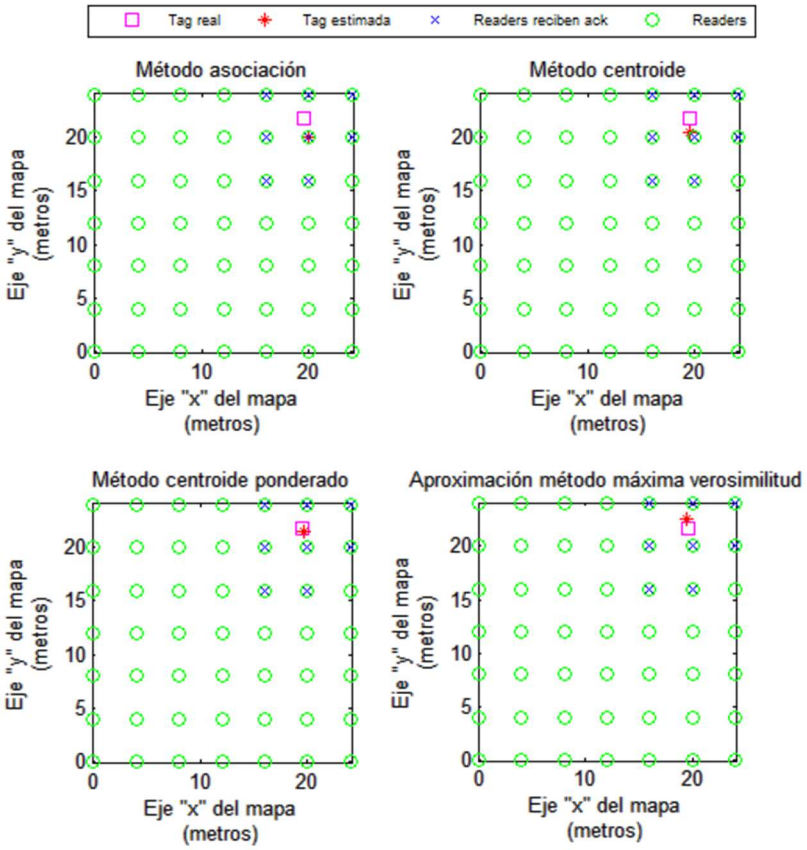

Figura 5-20. Cuarto ejemplo. Mapas, distancia entre readers 4 metros.

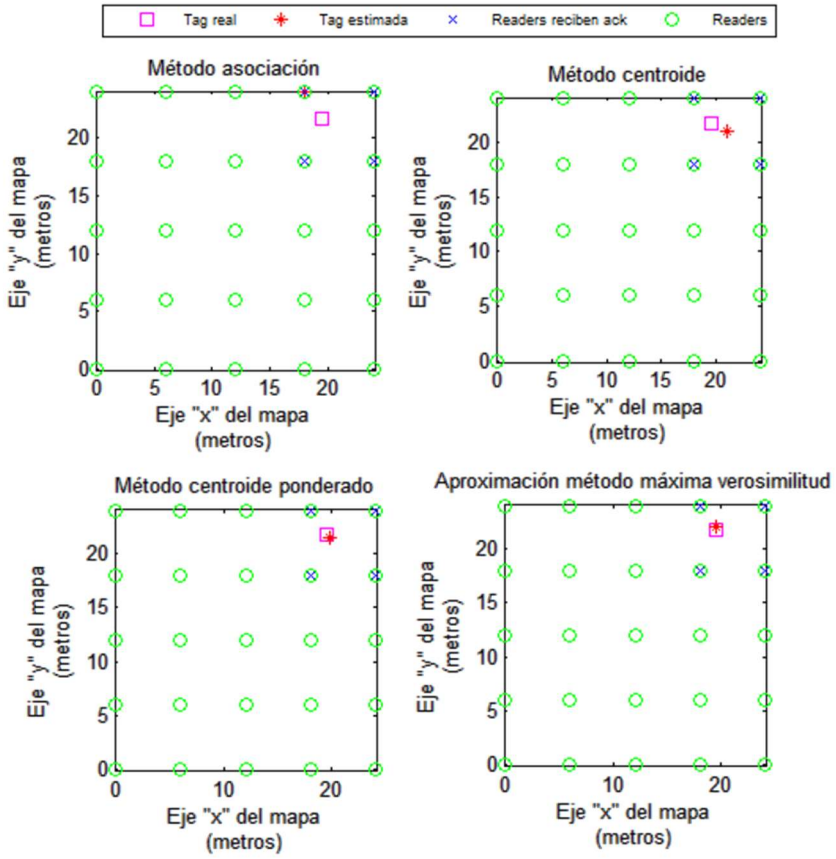

Figura 5-21. Cuarto ejemplo. Mapas, distancia entre readers 6 metros.

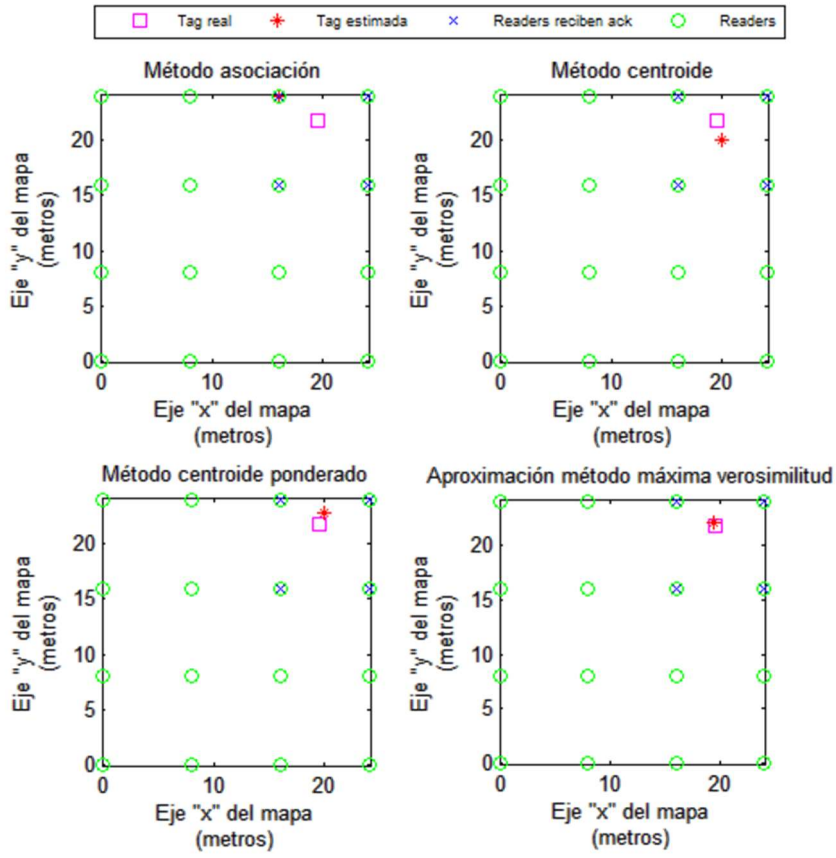

Figura 5-22. Cuarto ejemplo. Mapas, distancia entre readers 8 metros.

Por su parte, las figuras de la función de verosimilitud serán: para una separación de 4 metros la Figura [5-23], para una separación de 6 metros la Figura [5-24] y para una separación de 8 metros la Figura [5-25]. En los casos en los que las distancias entre readers es de 2 o 3 metros, las gráficas de la función de verosimilitud son tan picudas que no son facilmente interpretables, por lo que no se incluyen en este documento. Sin embargo, comentar que para estas dos últimas distancias será para las que se obtengan mejores resultados de la estimación de la tag, ya que apenas se producirán fallos al no tener muchos resultados simulares cercanos al máximo que lo distorsionen, lo cual se comprobará más adelante en la gráfica del RMSE.

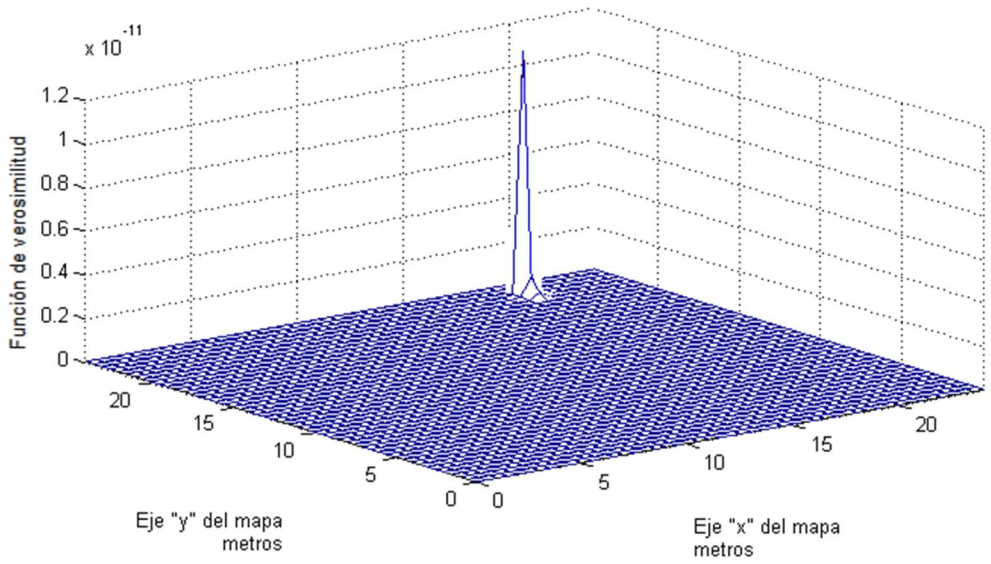

Figura 5-23. Cuarto ejemplo. Función verosimilitud, distancia entre readers 4 metros.

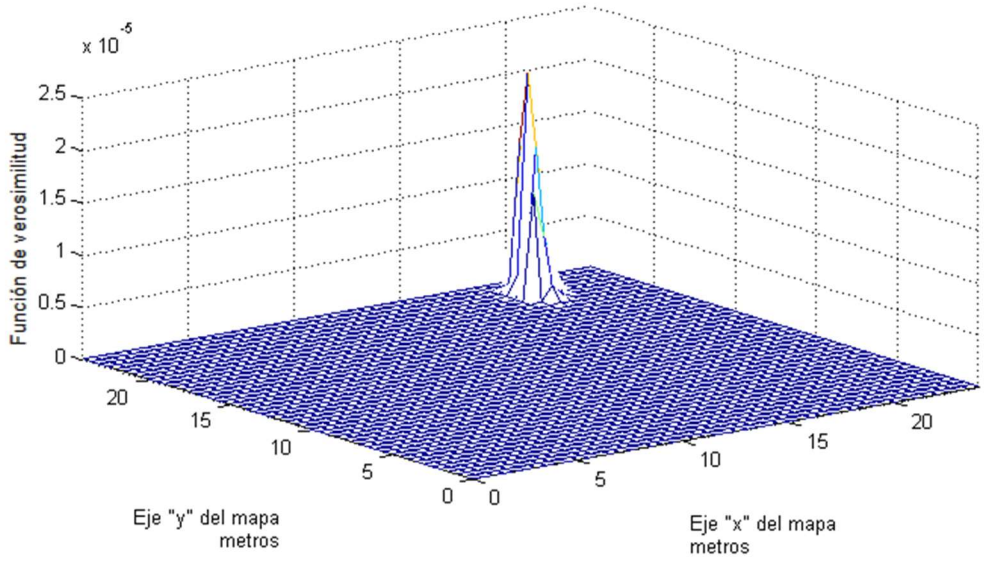

Figura 5-24. Cuarto ejemplo. Función verosimilitud, distancia entre readers 6 metros.

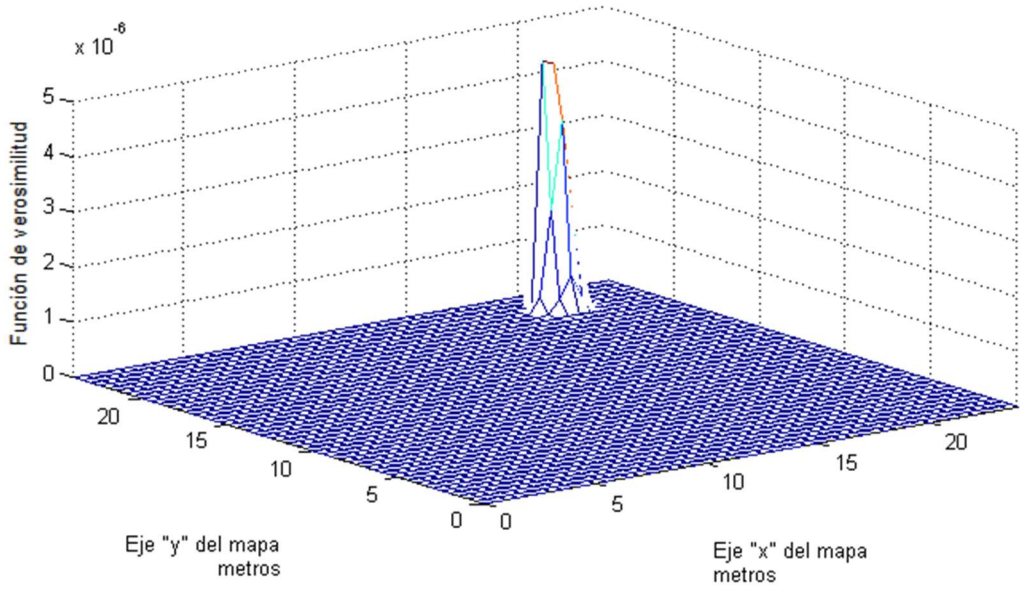

Figura 5-25. Cuarto ejemplo. Función verosimilitud, distancia entre readers 8 metros.

Finalmente, se muestra en la Figura [5-26] el RMSE en metros producido en función de la distancia entre readers.

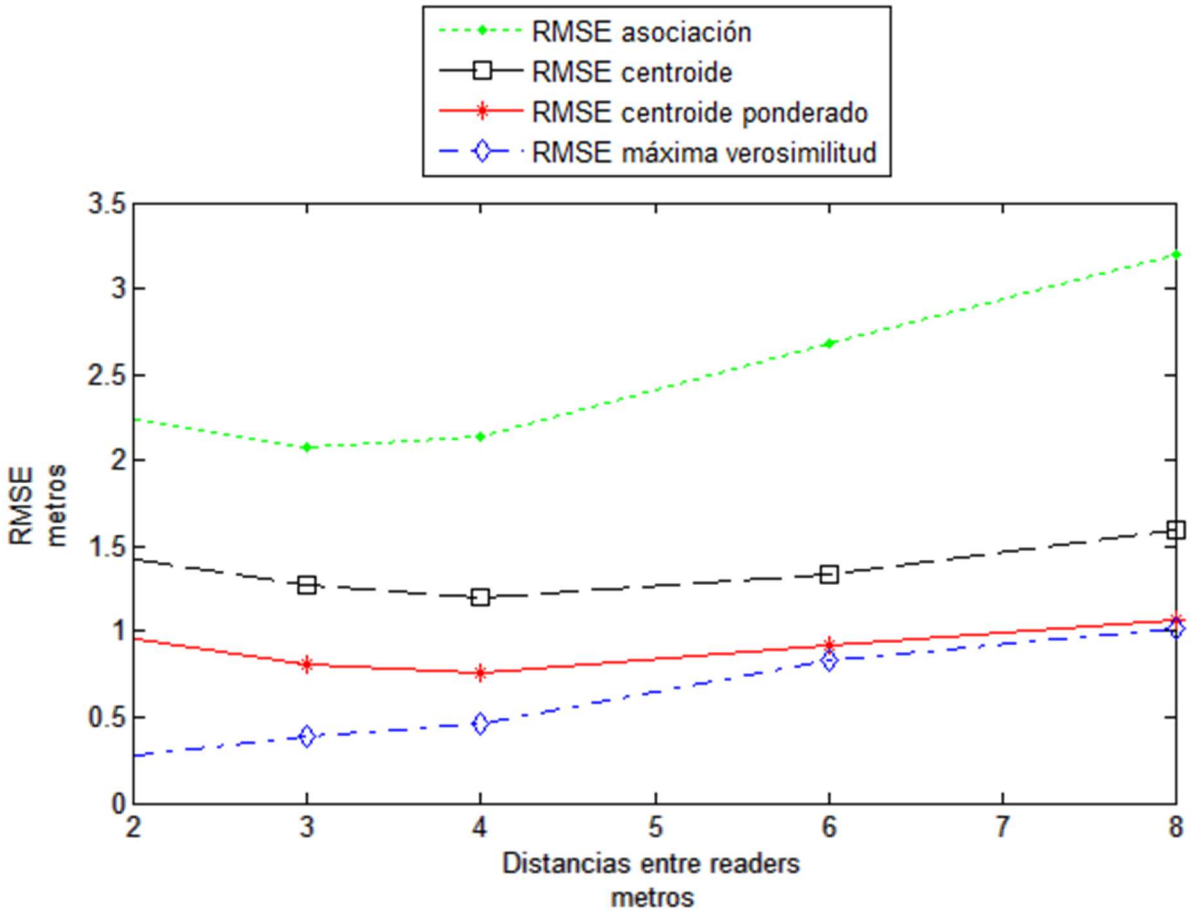

Figura 5-26. RMSE cuarto ejemplo.

Analizando la Figura [5-26], donde se han obtenido curvas más suaves que la de anteriores ejemplos al realizar más iteraciones, se puede decir que según la técnica de estimación de la tag empleada:

- Método de asociación: en dicha curva se puede observar un leve decrecimiento de unos dos decímetros del error que se produce cuando los readers están a dos y tres metros de distancia. Sin embargo, a partir de los 3 metros crece de forma monótona del RMSE. Esto es debido a que, a tan corta distancia entre readers como son los 2 metros, varios readers leen el número máximo de queries emitidos, siendo asignado como mejor posición entre ellos el que se encuentra más próximo a (0,0) metros, como ya se comentó en la descripción del método de asociación en este documento. Entonces, para este ejemplo se puede concluir que para emplear dicho método la mejor separación entre readers a tomar es la de 3 metros. Esta técnica se puede considerar como la que peor estimación de la tag ofrece, pudiendo estar el error entre 2 y 3 metros.
- Método del centroide: en este método también ocurre un decrecimiento al comienzo de la curva, que esta vez se alarga hasta la separación de 4 metros entre readers. Las razones de ello se proporcionaron en el segundo ejemplo de este mismo capítulo. Aún así, tras los 4 metros de distancia se comporta de una forma lógica, siendo peor la estimación de la tag conforme más se separan los readers. En cuanto a este método, ofrece una mejora considerable en sus estimaciones de la tag respecto al método anterior.
- Método del centroide ponderado: en esta curva ocurre lo mismo que en el método anterior, la única diferencia es que se produce a valores menores de RMSE. Ésta sería la segunda mejor técnica que aquí se presenta, rondando un error en la estimación de la tag de 1 metro, siendo muy parecido el resultado al ofrecido por el siguiente método para una distancia entre readers de 6 y 8 metros.
- Método aproximado a la máxima verosimilitud: la curva de esta técnica es la única que presenta una monotonía creciente en todo su recorrido. Esta es la única que cumple que con lo que se podría presuponer a priori, que a mayor distancia entre readers, peor serán las estimaciones de la tag tomadas. Como se puede observar, ésta es la técnica que presenta menor error en sus estimaciones, no siendo mayor de 1 metro para una separación entre readers de 8 metros en este ejemplo y llegando a ser de 0.25 metros para una distancia entre readers de 2 metros.

5.5 Quinto ejemplo

Para este último ejemplo, se ha tomado un mapa con las mismas características que el ejemplo anterior, con la excepción de que la precisión se ajusta aún más. Por este hecho, debido al tiempo de simulación que es necesario para tomar los resultados del mismo, se ha tomado una pequeña muestra de posiciones distintas de la tag para poder realizar una comparativa con las gráficas anteriores.

Ejemplo 5–5. *La tag es situada en 7 posiciones distintas en un mapa de dimensiones 24x24 metros. Las separaciones entre readers tomadas son 2, 3, 4, 6 y 8 metros. Los readers emiten 50 queries y se exige una precisión de 0.1 metros.*

Se toma como muestra una tag para representar los nuevos mapas de las Figuras [5-27], [5-28], [5-29], [5-30] y [5-31] correspondientes a las distintas distribuciones de los readers según la distancia de separación entre estos.

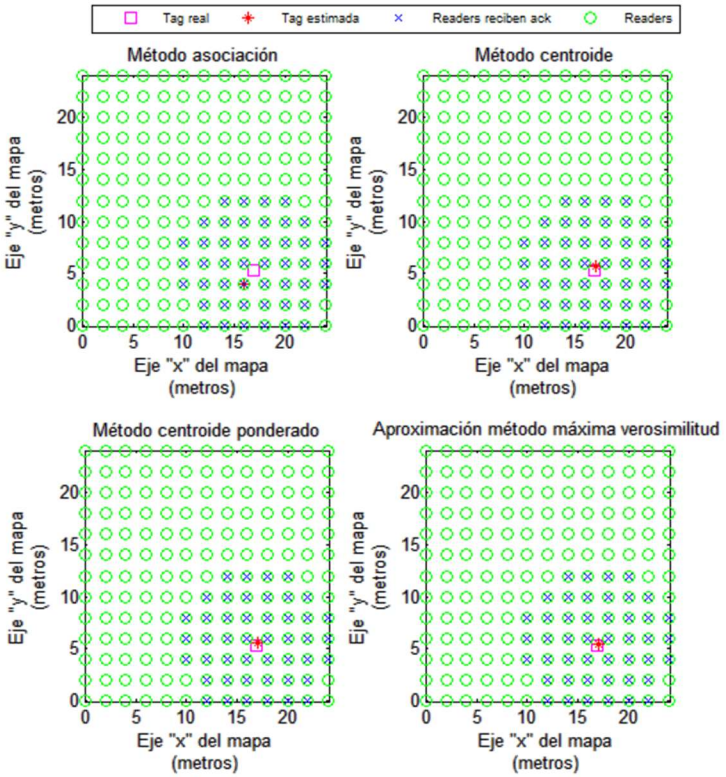

Figura 5-27. Quinto ejemplo. Mapas, distancia entre readers 2 metros.

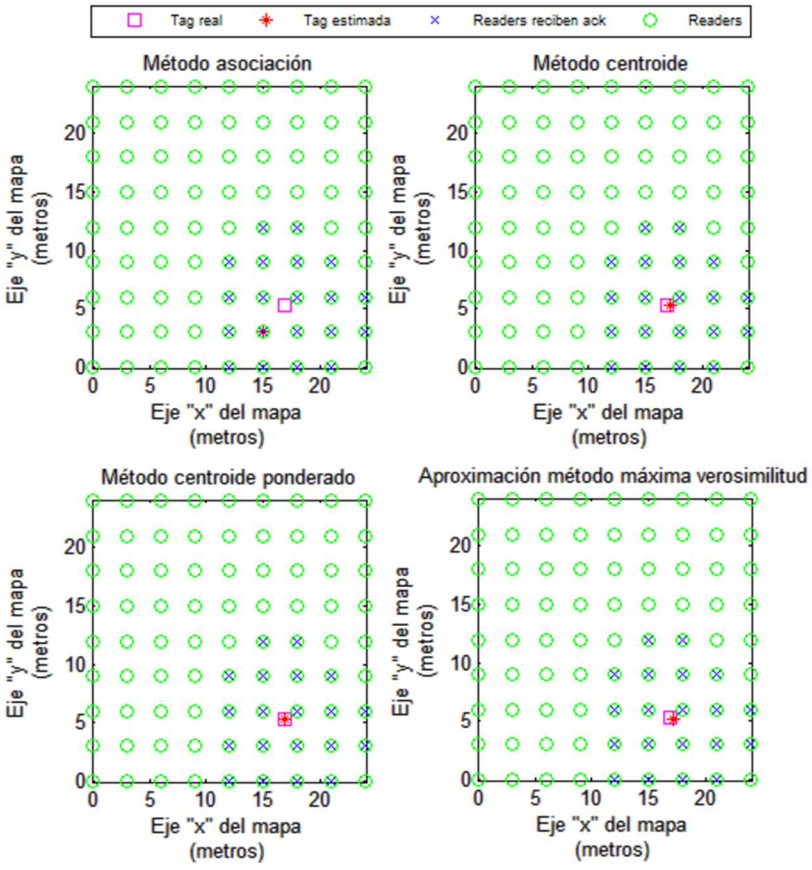

Figura 5-28. Quinto ejemplo. Mapas, distancia entre readers 3 metros.

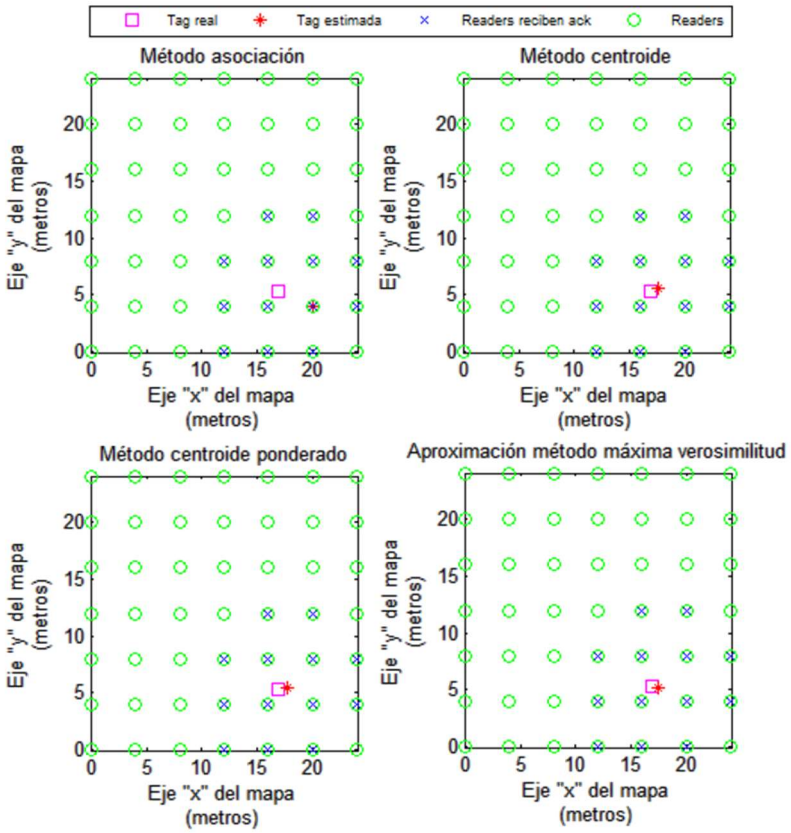

Figura 5-29. Quinto ejemplo. Mapas, distancia entre readers 4 metros.

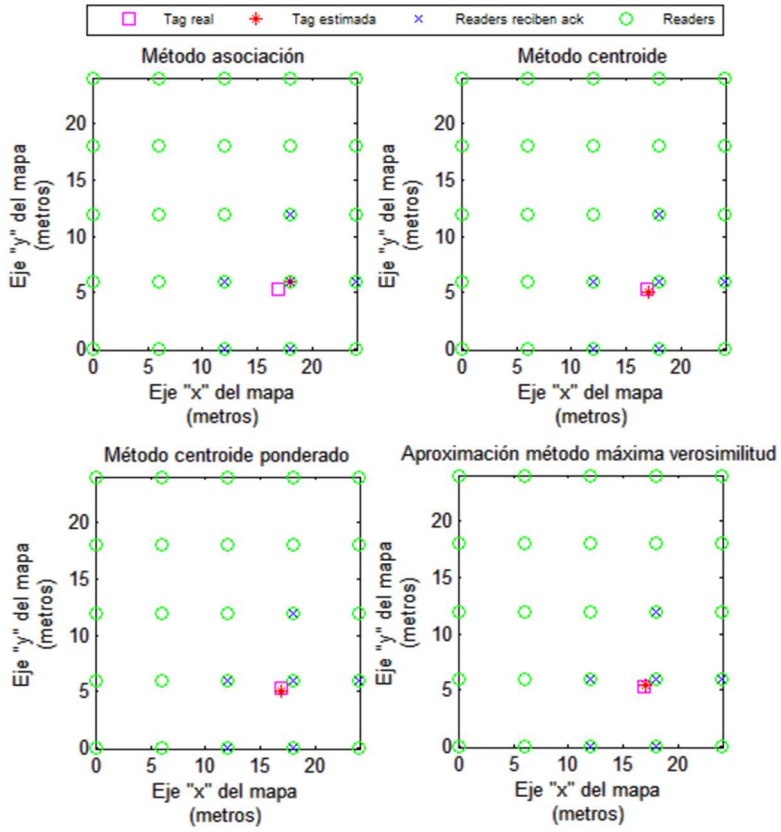

Figura 5-30. Quinto ejemplo. Mapas, distancia entre readers 6 metros.

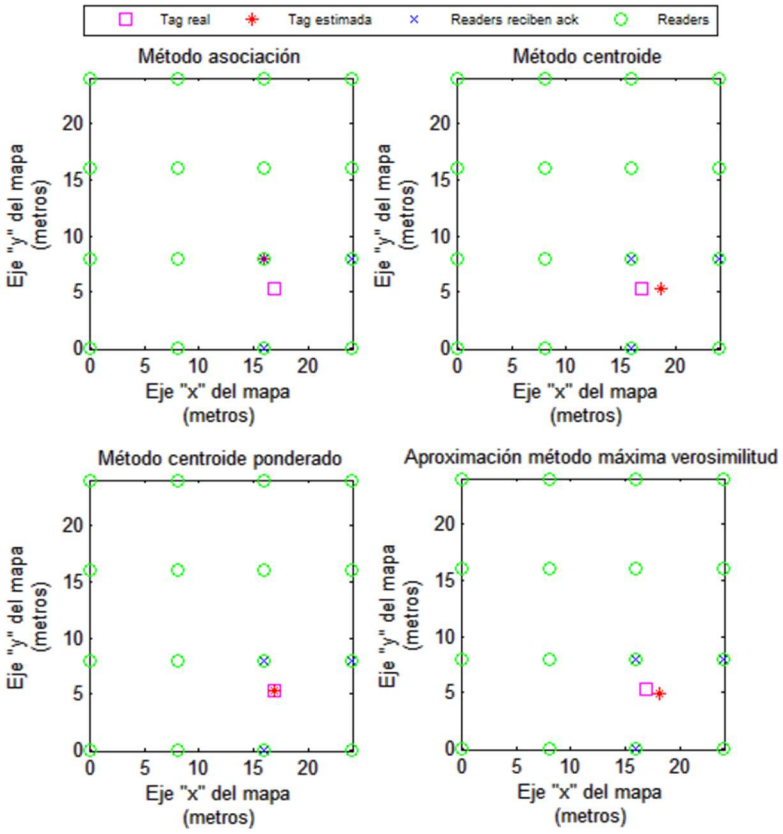

Figura 5-31. Quinto ejemplo. Mapas, distancia entre readers 8 metros.

Como se puede ver en los mapas, por ejemplo en la Figura [5-31], el método del centroide ponderado ha estimado exactamente la posición de la tag, mientras que la aproximación al método de máxima verosimilitud, pese a que se le exige una precisión de 0.1 metros, no lo ha conseguido. Estos resultados son correctos debido a que es simplemente una muestra aislada, pero no es lo que ocurriría en el mayor porcentaje de los casos si se tomaran muchas más muestras de la simulación.

Para esta muestra de ejemplo, las figuras de la función de verosimilitud serán: para una separación de 4 metros la Figura [5-32], para una separación de 6 metros la Figura [5-33] y para una separación de 8 metros la Figura [5-34].

Al igual que ocurría en el Ejemplo [5-4], la función de verosimilitud para las distancias entre readers de 2 y 3 metros no se muestran. En la Figura [5-32] se ve que dicho valor es realmente picudo, por ello, para distancias menores, lo será aún más.

Como se puede ver, a mayor número de readers, menos picudo será el resultado y mayor puede ser el error producido en la estimación.

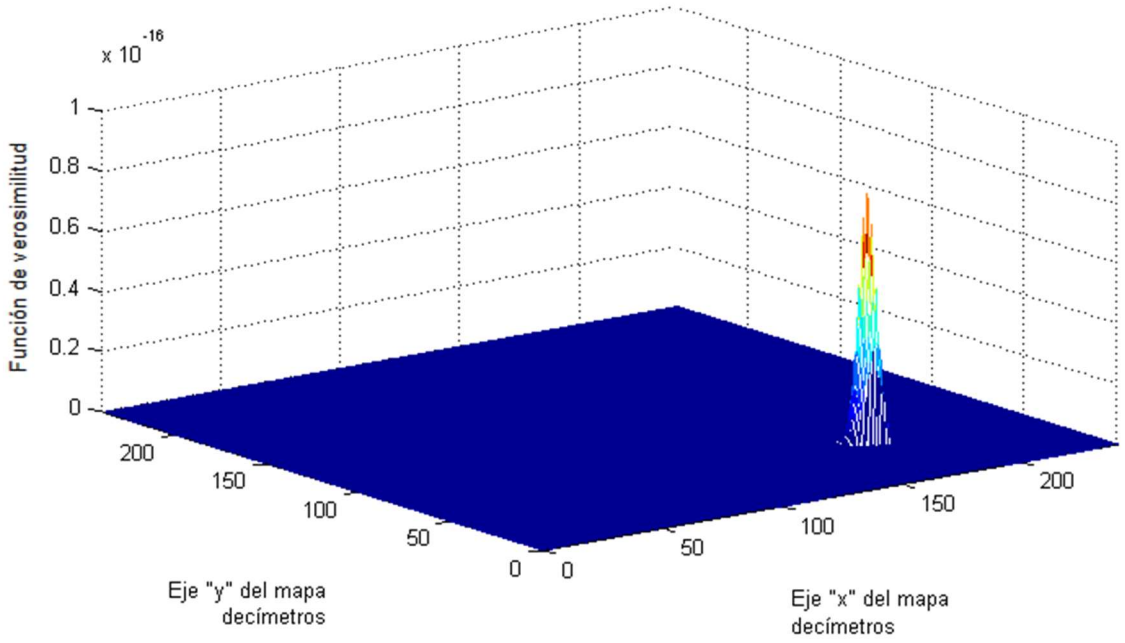

Figura 5-32. Quinto ejemplo. Función verosimilitud, distancia entre readers 4 metros

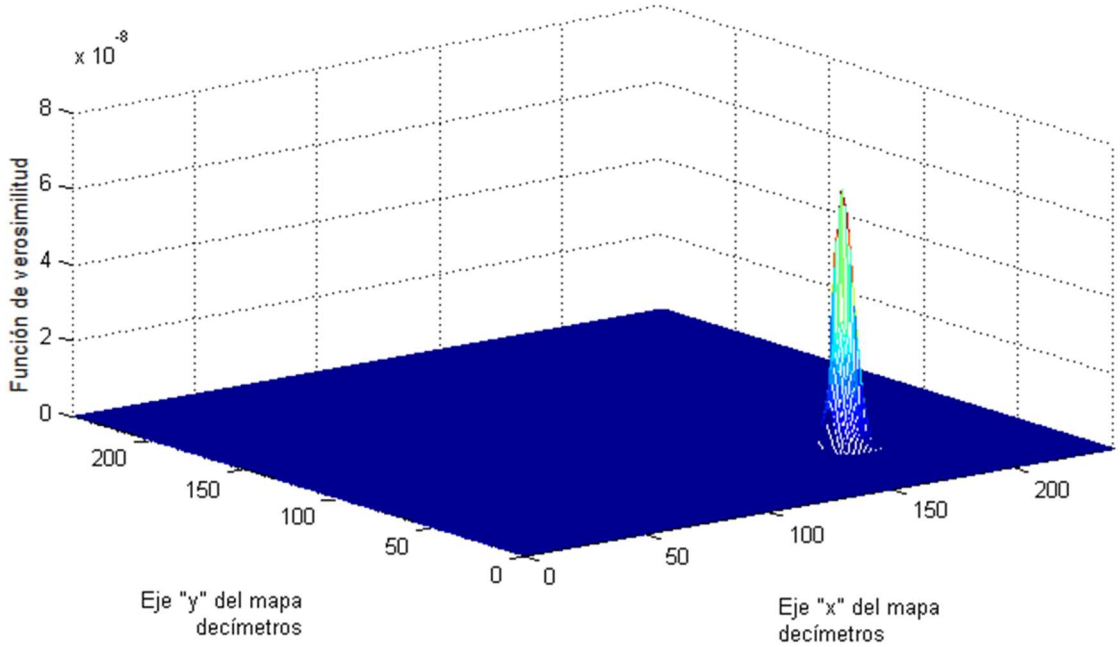

Figura 5-33. Quinto ejemplo. Función verosimilitud, distancia entre readers 6 metros.

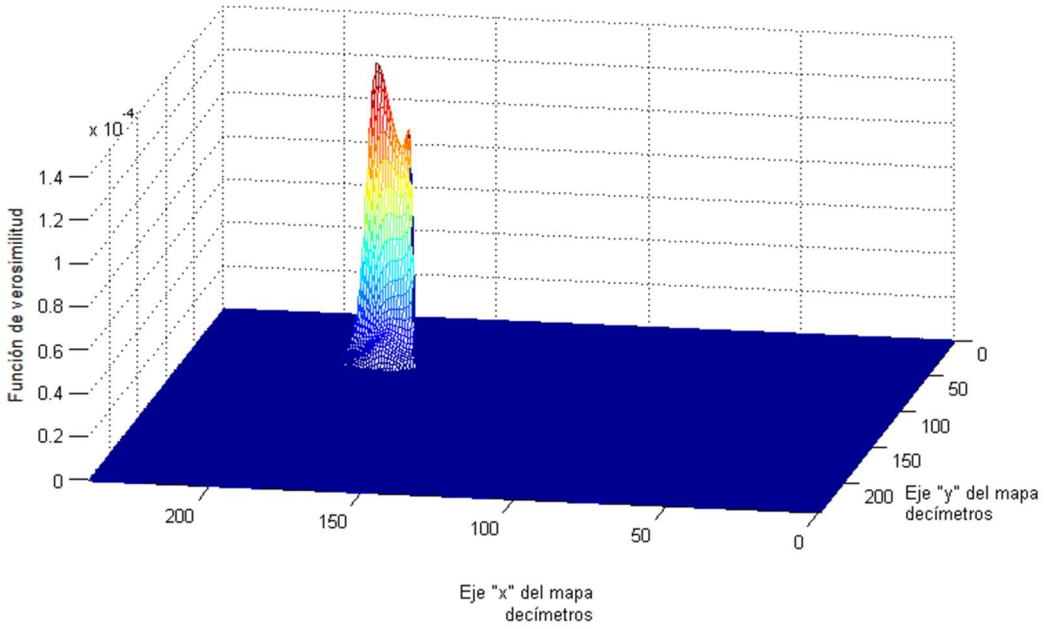

Figura 5-34. Quinto ejemplo. Función verosimilitud, distancia entre readers 8 metros.

Se ha de decir que la gráfica de la Figura [5-34] ha sido girada para una mejor visualización de la misma.

A continuación, se muestra en la Figura [5-35], al igual que en el ejemplo [5-3], el error de estimación de la tag cometido por el método aproximado a la máxima verosimilitud para las distintas distancias entre readers. Éstos son valores puntuales ya que se muestran a modo de nube de puntos para las 7 pruebas.

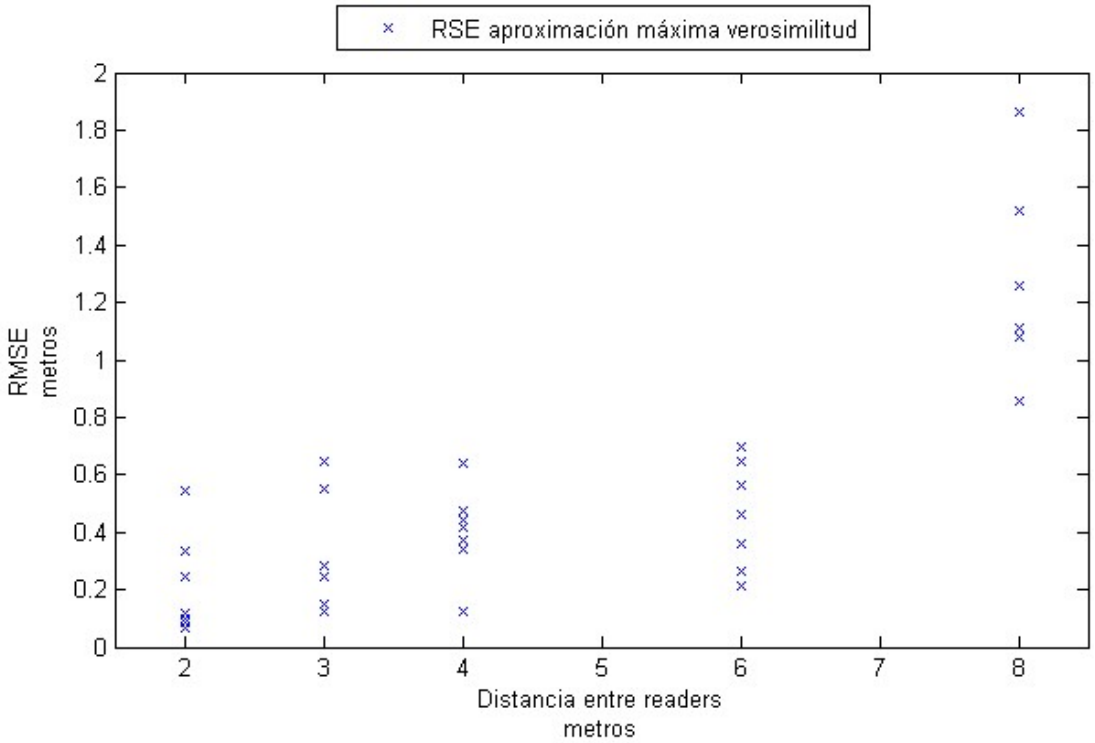

Figura 5-35. RSE quinto ejemplo.

Para finalizar, se va a realizar una comparativa entre realizaciones individuales del quinto ejemplo, que se puede ver en la gráfica [5-35], con la del ejemplo anterior que hacía la media de un mayor número de iteraciones, Figura [5-26]. Así se pueden obtener unos resultados preliminares de lo que resultaría el RMSE para estas condiciones dada una precisión de 0.1 metros y compararlas con el RMSE obtenido al dada una precisión de 0.5 metros. El análisis se va a realizar en función de las distancias entre readers:

- Separación entre readers de 2 metros: en la gráfica del cuarto ejemplo se podía ver que la media se encontraba en 0.25 metros, que es correcta para la precisión que se le exigía. En cambio, en la gráfica de este último ejemplo, la mayor concentración de cruces se encuentra en 0.1 metros, lo cual también es lo correcto y lo que se esperaba.
- Separación entre readers de 3 metros: cuando la distancia entre readers aumenta 1 metro, el RMSE también aumenta a 0.4 metros, como se puede ver en la Figura [5-26]. Esto también ocurre en la figura de este ejemplo, aunque en este caso se ve que el reparto de cruces se extiende desde los 0.1 metros hasta los 0.7, siendo más amplio este rango de expansión.
- Separación entre readers de 4 metros: en cuanto a esta distancia ocurre algo similar al anterior, donde los RMSE en ambas gráficas suben unos decímetros. Sin embargo, en las muestras de la gráfica [5-35] se puede observar que la expansión de los errores de las muestras tomadas es igual que para 3 metros pudiendo darse el caso en que el RMSE alcance 0.7 metros.
- Separación entre readers de 6 metros: por su parte en esta ocasión si se ve una crecida más considerable del RMSE en la Figura [5-26], siendo de 0.7 metros el error medio. Sin embargo, en las muestras individuales tomadas en este ejemplo el máximo error es ese mismo, 0.7 metros, estándo los demás por debajo de este y tomando incluso sólo 0.2 metros de RMSE.
- Separación entre readers de 8 metros: en cambio, aquí ocurre justo lo contrario que para 6 metros de distancia entre readers. En la gráfica [5-26] el valor medio del RMSE se encuentra sobre 1 metro, mientras que en la Figura [5-35] este será el menor valor del error tomado, pudiendo incluso alcanzar un error de 1.85 metros en la estimación de la tag.

Con esto se puede concluir que la precisión exigida antes de realizar las simulaciones no siempre se cumplirá, ya que la buena aproximación de la tag estimada a la tag real vendrá también condicionada en gran medida por la distancia de separación existente entre readers.

6 CONCLUSIONES

Nadie triunfa sin esfuerzo. Aquellos que trinfan deben su éxito a la perseverancia.

- Ramana Maharshi -

Para finalizar este proyecto, tras haber analizado los resultados obtenidos, se pueden sacar unas conclusiones generales a modo de resumen. Éstas son el principal objetivo de este trabajo ya que, aunque alguna pueda parecer que se puede obtener usando la lógica racional, se han podido constatar mediante los distintos resultados. Además, por lógica podíamos intuir los resultados cualitativamente, mientras que los resultados de las simulaciones permiten obtener conclusiones cuantitativas.

- Como primera conclusión, para un mapa de unas dimensiones dadas, la distancia existente entre los readers influye directamente en la precisión con la que se estima la posición de una tag. Esto es debido a que, a mayor distancia entre lectores, menor es el número de readers empleados para cubrir todo el área, suponiendo un mayor error en la estimación.
- Otro punto a tener en cuenta es que a mayor número de readers, el coste del sistema se elevará, ya que estos suponen un gran porcentaje del coste de un sistema RFID, siendo las tags los dispositivos más baratos. Por este hecho, hay que encontrar el equilibrio deseado entre exactitud de los resultados y número de readers a emplear.
- Hay que tener en cuenta que los cálculos empleados en este proyecto se han basado en el estudio experimental [6], por lo que bajo diferentes condiciones se obtendrán distintos resultados a los mostrados en este proyecto. Lo que habría que realizar es la adecuación del modelo de [6], o bien, obtener un modelo particularizado para el escenario concreto en que se situará el sistema, además de determinar la precisión con que seamos capaces de determinar ese modelo.
- Para finalizar, decir que en este proyecto se han ofrecido cuatro métodos de estimación de la localización. Se ha comprobado que el más simple, es decir, el método de estimación, es el que sufre mayor error en las estimaciones, pero también será el más simple de implementar y el más rápido para la realización de los cálculos. En contrapartida, la aproximación al método de máxima verosimilitud ha mostrado ser el que ofrece resultados más precisos, pero también el que requiere de mayor tiempo en los cálculos. Por su parte, el método del centroide ponderado también ofrece buenos resultados y no requiere de tanto tiempo como el último nombrado, siendo el método del centroide algo más inexacto. Así pues, según la aplicación que se desee de la localización en interiores, el presupuesto para ello y la precisión de los datos que se desea, se escogerá entre alguno de estos métodos, no teniendo que ser el que ofrezca la mayor exactitud necesariamente.

REFERENCIAS

[1] «RFID Technology PRIMER,» IMPINJ, [En línea]. Available: http://www.impinj.com/resources/about-rfid/.

[2] L. Díaz-Ambrona Tabernilla, Sistema de Localización en Interiores, F. Pérez Costoya, Ed., Madrid.

[3] I. S. 802.16, «Air interface for fixed broadband wireless access systems,» 2004.

[4] S. Tadakamadla, «Indoor Local Positioning System for Zigbee, Based on RSSI,» 2007.

[5] A. Ruber Royo, Implementación y Desarrollo de un Sistema de Localización en Interiores mediante RFID, Zaragoza.

[6] L. Geng, M. F. Bugallo, A. Athalye y P. M. Djuric, «Real Time Indoor Tracking of Tagged Objects with a Network of RFID Readers,» de 20th European Signal Processing Conference (EUSIPCO 2012), Bucharest, Romania, 2012.

[7] «Distribución Beta,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Distribuci%C3%B3n_beta.

[8] J. Blumenthal, R. Grossmann, F. Golatowski y D. Timmermann, «Weighted Centroid Localization in Zigbee-based,» 2007.

[9] «Bioestadística,» SEH-LELHA, [En línea]. Available: http://www.seh-lelha.org/maxverosim.htm.

[10] «El Método de Máxima Verosimilitud,» ReliabilityWeb. A culture of Reliability, [En línea]. Available: http://reliabilityweb.com/sp/articles/entry/el-metodo-de-maxima-verosimilitud/.

[11] «Beta function,» MathWorks, [En línea]. Available: http://es.mathworks.com/help/matlab/ref/beta.html.

[12] «RFID,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/RFID.

[13] J. Gómez Cadenas, «Método de máxima verosimilitud,» de Curso de Estadística, TAE, 2005.

[14] «Etiquetas RFID,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Etiqueta_RFID.

[15] A. Athalye, V. Savic, M. Bolic y P. M. Djuric, «Novel Semi-Passive RFID System,» IEEE Sensors Journal, vol. 13, nº 2, pp. 528 - 537, Febrero 2013.

[16] «Soluciones de Trazabilidad y RFID,» Dipole, [En línea]. Available: http://www.dipolerfid.es/Productos/Lectores-RFID/Portatiles-PDA.aspx.

[17] «Error cuadrático medio,» Wikipedia, [En línea]. Available: https://es.wikipedia.org/wiki/Error_cuadr%C3%A1tico_medio.

[18] S. Gupta, Order Statistics from the Gamma Distribution, 1962.

[19] A. Papoulis y S. U. Pillai, Probability, random variables, and stochastic processes, 4th ed., 2002.

1 ANEXOS: CÓDIGO MATLAB

1.1 Puesta a punto

1.1.1 Cálculo de "n"

1.1.1.1 Media de probabilidad de detección

```
function E = mean_p(d)

% Valores experimentales
a = 0.8471;
d0 = 5.2972;

% Como el experimento no tiene en cuenta las distancias que están por
%debajo de 1.2 metros ni por encima de 7.6, así que los tomamos de la %siguiente
manera ya que sabemos que deben darnos esos valores. De no tomar %esta precaución,
la media nos daría negativa, lo cual provoca a posteriori
%que la alpha y beta sean negativas, cosa que no puede ocurrir
if d < 1.2
    E = 1;
elseif d > 7.6
    E = 0;
else
    % Fórmula de la esperanza para de la detección de la petición enviada por %el
reader
    E = 1./(1+exp(a*(d-d0)));
end
% Para representación de la exponencial de ajuste
figure(1)
plot(d,E,'-*'),grid on
xlabel('Distancia (metros)')
ylabel('Probabilidad de detección')
legend('Exponencial de ajuste')

end



function d = regla_inic_dist

%Distancias de una tag respecto a un reader, sacadas de un experimento realizado
d = [1.2, 1.85, 2.15, 2.45, 2.8, 3.05, 3.2, 3.7, 3.95, 4.2, 4.6, 4.9, 5.1, 5.5,
5.85, 6.1, 6.4, 6.7, 7, 7.3, 7.6];
%Altura para la gráfica, ya que queremos ver las distancias simplemente en una
barra horizontal a modo de regla
h = zeros(length(d));
%Eje de abscisas en metros, ya que el reader se sitúa en el origen lo añadimos
xmetros = [0, d];
%Se representa en una gráfica tanto el reader en el origen (0,0) como a las
distintas posiciones de la tag
plot(xmetros(1),h(1),'go',xmetros(2:length(xmetros)),h,'*r');
%Se define el límite desde -0.4 para que sea visible el reader y hasta 8 ya que la
última distancia posible de la tag no supera los 8 metros. Así veremos todas las
posiciones al completo
xlim([-0.4,8]),shg
ylim([-0.25,0.5])
legend('reader','tags','Location','NorthEast')
xlabel('Distancias de la tag al reader (metros)')
end
```

1.1.1.2 Varianza

```
%Devuelve los valores estimados de la varianza
function V = var_p(d)

%Varianza estimada según la distancia de la tag al reader
%Distancia del tag al reader
varesx = [1.2, 1.85, 2.15, 2.45, 2.8, 3.05, 3.2, 3.7, 3.95, 4.2, 4.6, 4.9, 5.1,
5.5, 5.85, 6.1, 6.4, 6.7, 7, 7.3, 7.6];


%Varianza estimada
varesy = [0.002, 0.0025, 0.0025, 0.0035, 0.008, 0.0375,0.035, 0.004, 0.0325, 0.037,
0.038, 0.0325, 0.02, 0.019, 0.009, 0.0145, 0.005, 0.0175, 0.013, 0.019, 0.003];


%Polinomio de ajuste de sexto grado
ci = polyfit(varesx,varesy,6);


% Como el experimento no tiene en cuenta las distancias que están por
%debajo de 0.9151 metros ni por encima de 7.682 los tomamos de la siguiente
%manera ya que sabemos que deben darnos esos valores. De no tomar esta
%precaución, la varianza nos daría negativa, lo cual provoca a posteriori
%que la alpha y beta sean negativas, cosa que no puede ocurrir

if (d < 1.2) || (d > 7.6)
    V = 0;

else
    %Obtener resultados del polinomio para una distancia dada
    V = polyval(ci,d);

end


% Representar puntos estimados en el experimento de la varianza
plot(varesx,varesy,'r*'),hold on


% Representar polinomio de sexto grado
plot(varesx,V,'-*'),grid on
xlabel('Distancia (metros)')
ylabel('Varianza p(d)')
legend('Valores experimentales varianza','Polinomio 6° grado')
hold off

end
```

1.1.1.3 Alpha y beta

```
% Obtención de los parámetros necesarios para una distribución beta
function [alpha,beta] = parametros(E,V)


% Calcular alpha y beta en función de los valores de la esperanza y la varianza
alpha = (E^2 - E^3 - E*V) / V;
beta = (E^3 - 2*(E^2) + E*(1+V) - V) / V;

end
```

1.1.1.4 Probabilidad detección

```
function [prob,d,alpha,beta] = distbeta(d)

% Obtenemos las matrices fila tanto de la media como esperanza en función de
%la distancia (21 distancias distintas), implementadas en otros ficheros
E = mean_p(d);
V = var_p(d);

% Obtención de los parámetros requeridos para una distribución beta
[alpha,beta] = parametros(E,V);

% Probabilidad de que le llegue de vuelta al reader la petición de tag en
%función de la distancia a la que esté la tag
% Para asegurarnos de que, aunque esté por debajo de los
%límites de detección o por encima, de que da un valor coherente lo forzamos

if (d > 1.2) && (d < 7.6)
    prob = betarnd(alpha,beta);

elseif d <= 1.2
    prob = 1;

elseif d >= 7.6
    prob = 0;

end

end
```

1.1.1.5 Parámetro “n”

```
function n = n_acks(number_msgs_Tx,d)

%Obtener la probabilidad de enviarse un mensaje desde un Tx a una tag a 21
%distancias (d) distintas comprendidas entre 0 y 8 metros
[p,~,~,~] = distbeta(d);


num = binornd(number_msgs_Tx,p);


% Redondea hacia abajo porque no es válido que nos llegue la mitad de un
%mensaje, sólo son válidos los mensajes completos recibidos
n = floor(num);


for i=1:length(d)
    fprintf('Para una distancia %1.2f metros llegarán %d de los %d
transmitidos\n',d(i),n(i),number_msgs_Tx);
end

end
```

1.1.2 Distribución readers en el mapa

```
function [mapa_acks,mapa_lectores] =
mapa_rdrs_acks(x,y,dist,number_msgs_Tx,xtag,ytag)

%Matriz correspondiente al mapa completo (en metros) del lugar donde se
%desea localizar a la tag
mapa_acks = zeros(x+1,y+1);

%Número de readers dentro del mapa en dos dimensiones(+1 porque en (0,0)
%hay otro reader)
xrdrs = (x/dist)+1;
yrdrs = (y/dist)+1;

%Inicialización
k = 0;

%Matriz donde se representan solo los lectores como puntos y que cada uno
%incluye el numero de acks que reciben desde la tag
mapa_lectores = zeros(xrdrs,yrdrs);

%Sitúo a los readers poniendo en el mapa el número de mensajes recibidos en
%cada reader desde la tag
for i = 0:xrdrs-1
    for j = 0:yrdrs-1

        %Calcula las distancias entre cada reader y la tag
        d = sqrt(((i*dist)-xtag)^2+((j*dist)-ytag)^2);

        aux = n_acks(number_msgs_Tx,d);

        if aux >= 0
            mapa_acks((i*dist)+1,(j*dist)+1) = aux;
        end
    end
end

for i = 0:xrdrs-1
    for j = yrdrs-1:-1:0
        mapa_lectores(i+1,j+1) = mapa_acks((i*dist)+1,(k*dist)+1);
        k = k+1;
    end

    k = 0;
end

disp('Mapa representante del número de acks que recibe cada reader:')
mapa_lectores = mapa_lectores'

end
```

1.2 Métodos de estimación

1.2.1 Método de asociación

1.2.1.1 Estimación posición de la tag

```
function [xaprox_tag,yaprox_tag,x_rdrsRx,y_rdrsRx] =
aproxtag_1metodo(mapa_acks)

%Inicializamos variables
x_mapa = 0;
y_mapa = 0;

%Devuelve los índices del mapa donde está el/los reader/s que ha/n recibido
más
%mensajes
if max(max(mapa_acks))~=0
    [x_mapa,y_mapa] = find((mapa_acks==max(max(mapa_acks))));
end

% Como las matrices en matlab comienzan por 1 y no por 0(que queremos que
%sea la posición del primer reader (0,0) le restamos 1. Así tenemos las
%posiciones de los readers que han recibido alguna notificación
x_rdrsmaxacks = x_mapa-1;
y_rdrsmaxacks = y_mapa-1;

% Escojo como tag estimada la posición del primer reader (comenzando desde
%la coordenada (0,0)metros) que encontramos que tiene el número mayor de
%lecturas que se han recibido en los readers desde la tag
xaprox_tag = x_rdrsmaxacks(1);
yaprox_tag = y_rdrsmaxacks(1);

%Para poder dibujar los readers lectores:
%Devuelve los índices del mapa donde los readers han recibido más de un
%mensaje
[x_mapa_rdrs,y_mapa_rdrs] = find(mapa_acks>0);

%Como las matrices en matlab comienzan por 1 y no por 0(que queremos que
%sea la primera posición, 0 metros) le restamos 1 para que nos de los
%metros donde se posiciona el reader más cercano a la tag
x_rdrsRx = x_mapa_rdrs-1;
y_rdrsRx = y_mapa_rdrs-1;

end
```

1.2.1.2 Representación

```
function se1metodo = primer_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real)

[xtag_aprox,ytag_aprox,x_rdrsRx,y_rdrsRx] = aproxtag_1metodo(mapa_acks);


%Representamos tanto readers, readers que han leido acks, la posicion real
%de la tag y la posición estimada de la tag
subplot(2,2,1)


% Parámetros readers dentro del mapa donde estarán situados
estratégicamente
[xmap,ymap] = size(mapa_acks);
xpos = 0:dist_rdrs:xmap;
ypos = 0:dist_rdrs:ymap;
% Para hacer una matriz cuadrada con las dimensiones del mapa
[xpos_map,ypos_map] = meshgrid(xpos,ypos);


% Dibuja readers dentro del mapa
% Dibuja la tag real donde se sitúa
% Dibuja los readers que reciben mensajes desde la tag
% Dibujo posición aproximada donde se encuetra la tag
plot(xtag_real,ytag_real,'ms',xtag_aprox,ytag_aprox,'*r',x_rdrsRx,y_rdrsRx,
'xb',xpos_map,ypos_map,'go')


% Para mejor visualización del mapa 2D
% Limito los ejes del mapa
xlim([0-0.1 xmap-0.9])
ylim([0-0.1 ymap-0.9])

% Doy nombre a los ejes
xlabel({'Eje "x" del mapa','(metros)'})
ylabel({'Eje "y" del mapa','(metros)'})

% Se escribe una leyenda con los distintos componentes que aparecen en el
mapa
leyenda = legend('Tag real','Tag estimada','Readers reciben
ack','Readers','Location','northoutside','Orientation','horizontal');
set(leyenda,'FontSize',7);


se1metodo = square_error(xtag_real,ytag_real,xtag_aprox,ytag_aprox);


end
```

1.2.2 Método del centroide

1.2.2.1 Estimación posición de la tag

```
function [xtag_aprox,ytag_aprox,x_rdrsRx,y_rdrsRx] =
aproxtag_2metodo(mapa_acks)


%Inicializamos variables
x_mapa = 0;
y_mapa = 0;
xdist = 0;
ydist = 0;


%Devuelve los índices del mapa donde los readers han recibido más de un
%mensaje
[x_mapa,y_mapa] = find(mapa_acks>0);


% Como las matrices en matlab comienzan por 1 y no por 0(que queremos que
%sea la posición del primer reader (0,0) le restamos 1. Así tenemos las
%posiciones de los readers que han recibido alguna notificación
x_rdrsRx = x_mapa-1;
y_rdrsRx = y_mapa-1;


%Suma de ambas coordenadas de los readers que reciben mensajes de la tag
for i = 1:length(x_rdrsRx)
   xdist = xdist+x_rdrsRx(i);
   ydist = ydist+y_rdrsRx(i);
end


%En esta segunda aproximacion la tag se situa a la distancia media entre
%todos los nodos que reciben mensajes desde la tag
xtag_aprox = xdist/length(x_rdrsRx);
ytag_aprox = ydist/length(y_rdrsRx);


end
```

1.2.2.2 Representación

```
function se2metodo =
segundo_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real)

[xtag_aprox,ytag_aprox,x_rdrsRx,y_rdrsRx] = aproxtag_2metodo(mapa_acks);

%Representamos tanto readers, readers que han leido acks, la posicion real
%de la tag y la posición estimada de la tag
subplot(2,2,2)

% Parámetros readers dentro del mapa donde estarán situados
estratégicamente
[xmap,ymap] = size(mapa_acks);
xpos = 0:dist_rdrs:xmap;
ypos = 0:dist_rdrs:ymap;

% Para hacer una matriz cuadrada con las dimensiones del mapa
[xpos_map,ypos_map] = meshgrid(xpos,ypos);

% Dibuja readers dentro del mapa
% Dibuja la tag real donde se sitúa
% Dibuja los readers que reciben mensajes desde la tag
% Dibujo posición aproximada donde se encuetra la tag
plot(xtag_real,ytag_real,'ms',xtag_aprox,ytag_aprox,'*r',x_rdrsRx,y_rdrsRx,
'xb',xpos_map,ypos_map,'go')

% Para mejor visualización del mapa 2D
% Limito los ejes del mapa
xlim([0-0.1 xmap-0.9])
ylim([0-0.1 ymap-0.9])

% Doy nombre a los ejes
xlabel({'Eje "x" del mapa','(metros)'})
ylabel({'Eje "y" del mapa','(metros)'})

% Escribo una leyenda con los distintos componentes que aparecen en el mapa
leyenda = legend('Tag real','Tag estimada','Readers reciben
ack','Readers','Location','northoutside','Orientation','horizontal');
set(leyenda,'FontSize',7);


se2metodo = square_error(xtag_real,ytag_real,xtag_aprox,ytag_aprox);


end
```

1.2.3 Método del centroide ponderado

1.2.3.1 Estimación posición de la tag

```
function [xtag_aprox,ytag_aprox,x_rdrsRx,y_rdrsRx] =
aproxtag_3metodo(mapa_acks,number_msgs_Tx)


%Inicializamos variables
xdist = 0;
ydist = 0;
sum_pesos = 0;


%Devuelve los índices del mapa donde los readers han recibido algún mensaje
[x_mapa,y_mapa] = find(mapa_acks>0);


% Como las matrices en matlab comienzan por 1 y no por 0(que queremos que
%sea la posición del primer reader (0,0) le restamos 1. Así tenemos las
%posiciones de los readers que han recibido alguna notificación
x_rdrsRx = x_mapa-1;
y_rdrsRx = y_mapa-1;


%Suma de ambas coordenadas de los readers que reciben mensajes de la tag
for i = 1:length(x_rdrsRx)
   peso = (mapa_acks(x_mapa(i),y_mapa(i)))/number_msgs_Tx;
   xdist = xdist + peso*(x_rdrsRx(i));
   ydist = ydist + peso*(y_rdrsRx(i));
   sum_pesos = sum_pesos+peso;
end

%En esta segunda aproximacion la tag se situa a la distancia media entre
%todos los nodos que reciben mensajes desde la tag
xtag_aprox = xdist/sum_pesos;
ytag_aprox = ydist/sum_pesos;

end
```

1.2.3.2 Representación

```
function se3metodo =
tercer_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real,number_msgs_Tx)

[xtag_aprox,ytag_aprox,x_rdrsRx,y_rdrsRx] =
aproxtag_3metodo(mapa_acks,number_msgs_Tx);


%Representamos tanto readers, readers que han leido acks, la posicion real
%de la tag y la posición estimada de la tag
subplot(2,2,3)


% Parámetros readers dentro del mapa donde estarán situados
estratégicamente
[xmap,ymap] = size(mapa_acks);
xpos = 0:dist_rdrs:xmap;
ypos = 0:dist_rdrs:ymap;
% Para hacer una matriz cuadrada con las dimensiones del mapa
[xpos_map,ypos_map] = meshgrid(xpos,ypos);


% Dibuja readers dentro del mapa
% Dibuja la tag real donde se sitúa
% Dibuja los readers que reciben mensajes desde la tag
% Dibujo posición aproximada donde se encuetra la tag
plot(xtag_real,ytag_real,'ms',xtag_aprox,ytag_aprox,'*r',x_rdrsRx,y_rdrsRx,
'xb',xpos_map,ypos_map,'go')


% Para mejor visualización del mapa 2D
% Limito los ejes del mapa
xlim([0-0.1 xmap-0.9])
ylim([0-0.1 ymap-0.9])
% Doy nombre a los ejes
xlabel({'Eje "x" del mapa','(metros)'})
ylabel({'Eje "y" del mapa','(metros)'})
% Escribo una leyenda con los distintos componentes que aparecen en el mapa
leyenda = legend('Tag real','Tag estimada','Readers reciben
ack','Readers','Location','northoutside','Orientation','horizontal');
set(leyenda,'FontSize',7);


se3metodo = square_error(xtag_real,ytag_real,xtag_aprox,ytag_aprox);


end
```

1.2.4 Método de máxima verosimilitud

1.2.4.1 Estimación posición de la tag

```
function
[xtag_aprox_4metodo,ytag_aprox_4metodo,x_rdrsRx,y_rdrsRx,emv_matriz] =
aproxtag_4metodo(mapa,dist_rdrs,number_msgs_Tx,precision)
% Inicializacion
m = 1;
k = 1;
[xmapa,ymapa] = size(mapa);
% Devuelve los índices del mapa donde los readers han recibido más de un
%mensaje
[x_mapa,y_mapa] = find(mapa > 0)
% Como las matrices en matlab comienzan por 1 y no por 0(que queremos que
%sea la posición del primer reader (0,0) le restamos 1. Así tenemos las
%posiciones de los readers que han recibido alguna notificación
x_rdrsRx = x_mapa-1;
y_rdrsRx = y_mapa-1;
% Va obteniendo las probabilidades de que la tag real este en distintas
%posiciones supuestas, recorriendo el mapa según la precisión que se desee
%mediante la función "estimacionmaxveros" que sigue una función beta-
binomial
for i = 1:precision:xmapa
    for j = 1:precision:ymapa
        emv_matriz(k,m) = estimacionmaxveros(i-1,j-
1,dist_rdrs,number_msgs_Tx,mapa);
        m = m+1;
    end
    m = 1;
    k = k+1;
end
% Escogemos el valor mayor de probabilidad, que así es como se define el
%metodo de maxima verosimilitud
[xtag_aprox,ytag_aprox] = find(emv_matriz==max(max(emv_matriz)));
xtag_aprox_4metodo = (xtag_aprox-1)*precision;
ytag_aprox_4metodo = (ytag_aprox-1)*precision;
end
```

```
function emv =
estimacionmaxveros(x_puntoestim,y_puntoestim,dist_rdrs,number_msgs_Tx,mapa_
acks)
% Inicialización
emv = 1;
% Obtengo el valor de las probabilidades de cada posición estimada a cada
%reader para incluirlas posteriormente en el metodo de máxima verosimilitud
betabinf =
betabinfunc(x_puntoestim,y_puntoestim,dist_rdrs,number_msgs_Tx,mapa_acks);
[xsize_betabinf,ysize_betabinf] = size(betabinf);
% Multiplicación para método de máxima verosimilitud (emv: estimación
%máxima verosimilitud) como indica su definición
for k = 1:xsize_betabinf
    for j = 1:ysize_betabinf
            emv = betabinf(k,j)*emv;
    end
end
end
```

```
function betabinf =
betabinfunc(xtag_aprox,ytag_aprox,dist_rdrs,number_msgs_Tx,mapa_acks)

%Inicialización
aux = 0;
[xmap,ymap] = size(mapa_acks);
xmap = xmap-1;
ymap = ymap-1;
prob = zeros(xmap+1,ymap+1);
betabinf = zeros(((xmap)/dist_rdrs)+1,((ymap)/dist_rdrs)+1);

for i = 1:dist_rdrs:xmap+1
    for j = 1:dist_rdrs:ymap+1

        d = calc_distancia(xtag_aprox,ytag_aprox,i-1,j-1);

        [~,~,alpha,bet] = distbeta(d);

        n = mapa_acks(i,j);
        p =
(factorial(number_msgs_Tx)/(factorial(n)*factorial(number_msgs_Tx-
n)))*(beta(n+alpha,number_msgs_Tx-n+bet)/(beta(alpha,bet)));

        % Se modificará el valor de "p" en los casos extremos, donde la
        % distancia es menor que 1.2metros entre la tag y el reader o bien
        % superior a 7.6 metros
        if (d < 1.2) && (n == number_msgs_Tx)
            p = 1;
        elseif (d < 1.2) && (n ~= number_msgs_Tx)
            p = 0;
        end

        if (d > 7.6) && (n == 0)
            p = 1;
        elseif (d > 7.6) && (n ~= 0)
            p = 0;
        end

        if p > 0
            prob(i,j) = p;
        end
    end
end
%Para ver las probabilidades tal y como se verían en la gráfica si el (0,0)
%esta en la esquina inferior izquierda, solo redistribuye "prob" en
%"betabinf"
for m = 0:(xmap)/dist_rdrs
    for l = (ymap)/dist_rdrs:-1:0
        betabinf(m+1,l+1) = prob((m*dist_rdrs)+1,(aux*dist_rdrs)+1);
        aux = aux+1;
    end
    aux = 0;
end

% disp('Probabilidad RMSE:')
betabinf = betabinf';
end
```

```
function distancia=calc_distancia(x1,y1,x2,y2)

% Calcula la distancia entre dos coordenadas del mapa
distancia = sqrt((x1-x2)^2+(y1-y2)^2);

end
```

1.2.4.2 Representación

```
function [se4metodo,emv_matriz] =
cuarto_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real,number_msgs_Tx,precis
ion)

[xtag_aprox,ytag_aprox,x_rdrsRx,y_rdrsRx,emv_matriz] =
aproxtag_4metodo(mapa_acks,dist_rdrs,number_msgs_Tx,precision);

%Representamos tanto readers, readers que han leido acks, la posicion real
%de la tag y la posición estimada de la tag
subplot(2,2,4)

% Parámetros readers dentro del mapa donde estarán situados
estratégicamente
[xmap,ymap] = size(mapa_acks);
xpos = 0:dist_rdrs:xmap;
ypos = 0:dist_rdrs:ymap;
% Para hacer una matriz cuadrada con las dimensiones del mapa
[xpos_map,ypos_map] = meshgrid(xpos,ypos);

% Dibuja readers dentro del mapa
% Dibuja la tag real donde se sitúa
% Dibuja los readers que reciben mensajes desde la tag
% Dibujo posición aproximada donde se encuetra la tag
plot(xtag_real,ytag_real,'ms',xtag_aprox,ytag_aprox,'*r',x_rdrsRx,y_rdrsRx,
'xb',xpos_map,ypos_map,'go')

% Para mejor visualización del mapa
% Limito los ejes del mapa
xlim([0-0.1 xmap-0.9])
ylim([0-0.1 ymap-0.9])
% Doy nombre a los ejes
xlabel({'Eje "x" del mapa','(metros)'})
ylabel({'Eje "y" del mapa','(metros)'})
% Escribo una leyenda con los distintos componentes que aparecen en el mapa
leyenda = legend('Tag real','Tag estimada','Readers reciben
ack','Readers','Location','northoutside','Orientation','horizontal');
set(leyenda,'FontSize',7);

se4metodo = square_error(xtag_real,ytag_real,xtag_aprox,ytag_aprox);

end
```

1.2.4.3 Gráfica estimación de máxima verosimilitud

```
function graficaemv(emv)

% Para obtener el tamaño del mapa
[xmapa,ymapa] = size(emv);

% Obtengo los dos vectores que corresponden a la malla del mapa
[x,y] = meshgrid(0:xmapa-1,0:ymapa-1);

% Representación visual mediante una gráfica en 3D de las
%distintas probabilidades de que se de una posición
mesh(y,x,emv)
xlim([0 xmapa-1])
ylim([0 ymapa-1])
xlabel({'Eje "x" del mapa','decímetros'})
ylabel({'Eje "y" del mapa','decímetros'})
zlabel('Probbiliades estimación de máxima verosimilitud')

end
```

1.2.5 Representación todos los métodos de estimación de la posición

```
function [se1metodo,se2metodo,se3metodo,se4metodo,emv_matriz] =
metodos(xmap,ymap,dist_rdrs,precision,xtag_real,ytag_real,number_msgs_Tx)

fprintf('La tag está situada en:(%d,%d)\n',xtag_real,ytag_real)

%Llamamos sólo una vez al algoritmo que calcula el número de acks que
%recibe cada reader porque si lo hacemos dentro de cada metodo, al llamar a
%una binornd, saldrán numeros distintos
mapa_acks =
mapa_rdrs_acks(xmap,ymap,dist_rdrs,number_msgs_Tx,xtag_real,ytag_real);


se1metodo = primer_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real);
title('Método asociación')

se2metodo = segundo_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real);
title('Método centroide')

se3metodo =
tercer_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real,number_msgs_Tx);
title('Método centroide ponderado')

[se4metodo,emv_matriz] =
cuarto_metodo(mapa_acks,dist_rdrs,xtag_real,ytag_real,number_msgs_Tx,precis
ion);
title('Método máxima verosimilitud')

end
```

1.3 Variación condiciones de entorno y errores de estimación de localización

1.3.1 Distancia entre readers variable

```
function [se1metodo,se2metodo,se3metodo,se4metodo,j] =
test_dimensionfija(precision,xmap,ymap,dist_rdrs,number_msgs_Tx)

%Para introducir la posición exacta de la tag
% xtag_real = input('Introduzca coordenada x de la tag: ');
% ytag_real = input('Introduzca coordenada y de la tag: ');
xtag_real = xmap*rand;
ytag_real = ymap*rand;

%Si alguna de las coordenadas de la tarjeta se sale del mapa, la volverá a
%pedir
while (xtag_real > xmap || xtag_real < 0)
xtag_real = input('Por favor, introduzca correctamente la coordenada ''x''
de la tag: ');

end

while (ytag_real > ymap || ytag_real < 0)
ytag_real = input('Por favor, introduzca correctamente la coordenada ''y''
de la tag: ');
end

for i=1:length(dist_rdrs)

    %Para dibujar la emv en otra figura y no se solapen
    j = i+length(dist_rdrs);

    % Para garantizar que la distancia mínima entre readers sea de 1 metro
como
    %el desarrollo llevado a cabo en el experimento
    if dist_rdrs(i) < 1
        disp('La distancia mínima entre readers ha de ser de 1 metro')
    else
        figure(i)
        [se1metodo(i),se2metodo(i),se3metodo(i),se4metodo(i),emv_matriz] =
metodos(xmap,ymap,dist_rdrs(i),precision,xtag_real,ytag_real,number_msgs_Tx
);
    end
    % Dibujar en 3D los valores de emv que se da en el método de máxima
    % verosimilitud para las distintas distancias entre readers
    figure(j)
    graficaemv(emv_matriz)
end
end
```

1.3.2 Errores de estimación

```
function se = square_error(xtag_real,ytag_real,xtag_estimada,ytag_estimada)
% Realiza el calculo del error cometido en la estimacion de una tag
%respecto de la tag real, en metros
e = calc_distancia(xtag_real,ytag_real,xtag_estimada,ytag_estimada);
% Eleva este error al cuadrado para que sea el error cuadrático, obteniendo
%metros al cuadrado
se = e.^2;
end
```

```
function rmse_metodo = rmse(se_metodo)
% Realiza la media y de los errores cuadráticos y calcula su raiz para un
% mejor entendimiento (así resultan metros y no metros al cuadrado)
rmse_metodo = sqrt(mean(se_metodo));
end
```

1.3.3 RMSE para distintos métodos y distancias entre readers

```
function resultados_test
close all
% Inicialización
% Precisión en las medidas estimadas que se desea (en metros)
precision = 0.1;
% Largo y alto del mapa donde se sitúan los readers (tener en cuenta que en
%la posición (0,0) hay un reader) metros
xmap = 24;
ymap = 24;
% Distancia minima entre readers, pensados para que se cubra toda la
%superficie, hasta las esquinas
dist_rdrs = [2 3 4 6 8];
% Número de mensajes de localización emitidos por cada reader
number_msgs_Tx = 50;
% Número de veces que queremos realizar el test (tomando distintas
posiciones aleatorias de la tag) para tomar una media de los
%errores cuadráticos medio producidos, para ver en la curva claramente cuál
%es el mejor método y qué situación es la óptima
iteraciones = 20;

% Este for lo que hace es realizar un número de iteraciones del test, es
%decir, tomar distintas posiciones de la tag real (posiciones aleatorias)
%en cada iteración
for i = 1:iteraciones
    [se1metodo,se2metodo,se3metodo,se4metodo,k] =
test_dimensionfija(precision,xmap,ymap,dist_rdrs,number_msgs_Tx);

    for j = 1:length(se1metodo)
        se1metodo_iter(i,j) = se1metodo(j);
        se2metodo_iter(i,j) = se2metodo(j);
        se3metodo_iter(i,j) = se3metodo(j);
        se4metodo_iter(i,j) = se4metodo(j);
    end
end
% Hace la media de los errores al cuadrado y su raíz para que sean raiz del
%error cuadrático medio, lo cual es más legible al representar los errores
%que se han producido en las estimaciones viéndolos en metros y no en
%metros al cuadrado. Las distintas columnas presentan las distintas
%distancias que se han tomado para las simulaciones
rmse1met = rmse(se1metodo_iter);
rmse2met = rmse(se2metodo_iter);
rmse3met = rmse(se3metodo_iter);
rmse4met = rmse(se4metodo_iter);

% Representación de las curvas de RMSE para los distintos métodos en
%función de la distancia entre readers
figure(k+1)
plot(dist_rdrs,rmse1met,'x--g',dist_rdrs,rmse2met,'x--
k',dist_rdrs,rmse3met,'x--r',dist_rdrs,rmse4met,'x--b')
legend('RMSE asociación','RMSE centroide','RMSE centroide ponderado','RMSE
máxima verosimilitud','Location','northoutside')
xlim([dist_rdrs(1)-0.5 dist_rdrs(length(dist_rdrs))+0.5])
end
```