Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

INGENIERÍA SUPERIOR EN TELECOMUNICACIÓN

Proyecto de fin de carrera:

Diseño de secuencias y filtro para localización cooperativa

Autor:

David Aguilar Pérez

Director:

Satyam Dwivedi

Post-doc in KTH School of Electrical Engineering

Department of Signal Processing

Ponente:

Antonio Valdovinos Bardají

Diciembre 2012

Agradecimientos

A mi supervisor, Satyam Dwivedi, por darme la oportunidad de realizar mi proyecto final de carrera en el Departamento de Procesado de Señal de la Escuela de Ingeniería Electrica del Instituto Real de la Tecnología de Estocolmo, y por su excelente guiado y asesoramiento dirante toda la duración del proyecto. También mencionar su disponibilidad y flexibilidad, permitiéndome trabajar a mi manera, incluso trabajar desde España.

A toda la gente de Kista y de Norrtälje que han estado a mi lado durante el desarrollo del proyecto. Han sido realmente muy buenos compañeros de trabajo y han sido de una enorme ayuda y apoyo. También a David Simón, por las muchas horas trabajando juntos durante todo el año.

A Diego y Juan Carlos por hacer que el tramo final del proyecto, y la traducción del mismo, fuera menos tedioso gracias al buen ambiente de trabajo. A Loreto por todo su apoyo y sus buenos consejos. Y en general, a todos mis compañeros de clase y amigos por estar ahí e interesarse por el proyecto.

Finalmente, mi mayor agradecimiento a mis padres, Javier y Pilar, y a mi hermana, Nerea, por darme todas las facilidades y el apoyo necesario para realizar el proyecto en Suecia, con todas las dificultades que ello conllevaba.

Abstract

Diseño de secuencias y filtro para localización cooperativa

En sistemas de localización cooperativa, los dispositivos de una red transmiten pulsos en un determinado orden preestablecido. Cada dispositivo (nodo) realiza mediciones del tiempo de llegada de los distintos pulsos recibidos para localizar a los otros nodos de la red. Este proyecto ha tratado algunos aspectos de la localización cooperativa basada en secuencias.

La primera parte del proyecto define el sistema propuesto y las secuencias de transmisión de pulsos. Después se discuten las restricciones que el mismo sistema impone y se propone un algoritmo de generación de una secuencia que se adapte al número de nodos de una red. Una vez expuesto este algoritmo, se extiende para que sea capaz de generar todas las secuencias posibles para cualquier red.

La segunda parte trata sobre el diseño del filtro que va a procesar las mediciones que los nodos toman, para estimar las distancias entre todos los nodos de la red. Cada nodo ejecuta un filtro de Kalman. Se muestran varias simulaciones para este modelo. Se discute también coste computacional requerido por el filtro, proponiendo diversas formas de reducirlo. Se realiza una comparación entre varios métodos de descomposición de matrices, descomposiciónes QR y Cholesky. Finalmente, se discute la relación del coste computacional con el número de nodos que compone la red.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1 La localización, perspectiva histórica

Desde hace miles de años el ser humano ha tratado de encontrar su posición para así poder guiarse y poder llegar a su destino. La navegación y la localización pronto tomaron un importante papel, tan importante que ahora es incluso considerada una ciencia. La palabra clave es *referencias*. Cada vez que se está localizando o siguiendo un objeto, se hace con respecto a diversas posiciones de referencia.

Los sistemas de localización pueden ser clasificados en dos grupos principales: Sistemas de navegación por localización, y sistemas de navegación a la estima.

- *Navegación por localización*: La posición es facilitada directamente por el sistema de posicionamiento. La velocidad se obtiene mediante la derivación de la misma respecto al tiempo.
- *Navegación a la estima*: La posición es calculada mediante la medición de un conjunto de magnitudes relativas a una conocida posición inicial. Magnitudes que pueden ser medidas son la velocidad o la aceleración.

Ambos tipos de sistemas pueden coexistir en un mismo medio y trabajar de forma conjunta para así poder aprovechar las ventajas de ambos y poder así obtener mejores resultados. (1)

1.1.1 Introducción histórica

El primer sistema de posicionamiento conocido usado por el hombre fue la navegación astronómica. Los astros eran usados como referencias estáticas y, mirando hacia ellos, la posición podía ser calculada. Mediante el uso de herramientas como almanaques, cartas náuticas, sextantes y algunas otras, los barcos podían saber su posición en medio del océano, o ser guiados hacia su destino. Estos principios de navegación astronómica son usados actualmente en algunos sistemas de navegación.

Con la aparición de las radioayudas, diversos sistemas de posicionamiento fueron desarrollados. El primer sistema que empleaba ondas de radio como medio para calcular la posición fue la radiogoniometría. Estaba basado en una antena cuadrada que era capaz de calcular la dirección de la señal entrante debido a su diagrama de radiación. Las referencias eran balizas situadas en tierra firme en posiciones conocidas. Sabiendo la distancia a dos balizas, la posición podía ser calculada. Más adelante, varias versiones de este sistema fueron apareciendo. Algunos empleaban antenas rotatorias, otros usaban entramados de antenas, y algunos otros incluían alguna ayuda electrónica para mejorar la precisión. A día de hoy, este tipo de sistemas se utiliza en sistemas de seguridad.

La siguiente generación de sistemas de transmisión la formaban los sistemas hiperbólicos (1). LORAN (LOng RAnge Navigation) es el sistema que sigue este principio más conocido. Está basado en la medida de la diferencia entre tiempos de llegada de distintas señales procedentes de diversas balizas situadas en posiciones conocidas. La diferencia de tiempos crea lugares geométricos en el espacio (hiperboloides) de posibles posiciones. Mediante la intersección de un mínimo de tres hiperboloides la posición puede ser conocida. Por lo tanto, se necesitan un mínimo de tres balizas de referencia para realizar la localización. El sistema LORAN se sigue utilizando en la actualidad para navegación ya que aun existen balizas situadas en las costas para su utilización.

El sistema de posicionamiento más reciente lo componen los sistemas satelitales. GPS (Global Positioning System), perteneciente a los Estados Unidos es el más conocido. En Europa, un sistema similar (Galileo) está siendo desarrollado, siguiendo los mismos principios (1). Se tratan de sistemas donde la comunicación se produce entre un dispositivo en la superficie terrestre, y diversos satélites, siendo todos estos enlaces descendentes. Se basan en estimación esférica. Cada dispositivo, con cada señal que recibe de cada satélite, crea un lugar geométrico (esfera) de posibles posiciones. De esta forma, mediante la intersección de un mínimo de cuatro esferas, la posición es calculada. Un mínimo de cuatro satélites han de estar visibles en el momento de realizar el posicionamiento, aunque, tal y como están distribuidos los satélites, entre 27 y 28 a lo largo de la órbita geoestacionaria, los satélites con visión directa oscilan entre cinco y ocho.

A día de hoy, existen muchos sistemas de posicionamiento y localización diferentes, cada uno con sus propias propiedades y aplicaciones, todos ellos basados en las referencias. Algunos sistemas emplean referencias estáticas con posiciones fijas y conocidas, llamadas anclajes, y algunos otros se ayudan de referencias móviles con posiciones que no son fijas, conocidas como agentes. Algunos sistemas utilizan sólo anclajes, sólo agentes, o una combinación de ambos. Por ejemplo, anclajes pueden ser los astros en navegación astronómica, las balizas en los sistemas hiperbólicos, o los satélites en GPS o Galileo, mientras que los agentes pueden ser cada dispositivo a posicionar, como puede ser un receptor GPS que se comunique con otros receptores. La figura 1.1 muestra un escenario con varios anclajes y agentes que se comunican entre ellos para

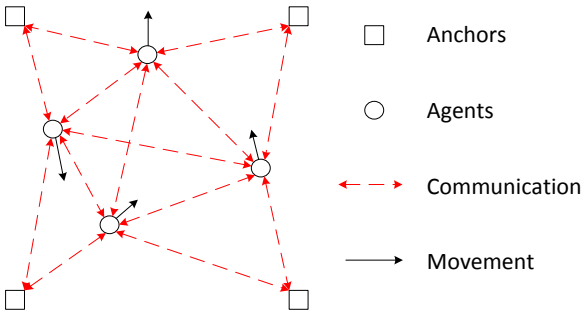

Figura 1.1: Escenario con cuatro anclajes y cuatro agentes

llevar a cabo el posicionamiento.

La comunicación entre agentes para llevar a cabo la localización y el posicionamiento es conocida como *localización cooperativa*, ya que cada agente coopera con los demás compartiendo con ellos algunas medidas realizadas con el mismo. De esta forma, se permite de alguna manera que agentes con posiciones desconocidas tomar y compartir mediciones con otros dispositivos con posiciones también desconocidas (2).

1.2 Localización cooperativa

La principal característica de la localización cooperativa es la existencia de comunicación entre agentes en una red. En esta comunicación, varios datos relevantes relacionados con cada agente son transmitidos a los demás agentes para llevar a cabo el posicionamiento. Existen principalmente dos tipos de sistemas de localiación cooperativa. El primero se compone de sistemas formados por anclajes y agentes donde existen dos tipos de comunicación, anclaje - agente y agente - agente. El segundo lo forman las redes formadas exclusivamente por agentes, donde todos los componentes son dispositivos móviles que se comunican entre ellos. La localización cooperativa presenta varias ventajas comparada con los sistemas de localización estándar. La principal ventaja es su buen funcionamiento en entornos hostiles, donde es imposible obtener información externa. Por ejemplo, en el interior de una cueva o edificio, donde señales externas no pueden penetrar hasta los nodos de la red. Otra ventaja importante es la reducción del consumo de energía. Estas redes de nodos suelen emplearse en escenarios de reducido tamaño, luego las distancias entre los nodos es pequeña y la potencia de transmisión requerida se ve reducida comparada con la necesaria para establecer contacto con un satélite o una baliza.

La localización cooperativa puede ser empleada en diversas aplicaciones. Este tipo de sistemas pueden utilizarse en seguimiento de animales o logística (2). Se pueden emplear para localizar maquinaria, vehículos, mercancías ... También puede emplearse en el seguimiento de las condiciones de almacenaje en grandes almacenes, o en redes de almacenes. Otra situación donde la localización cooperativa puede ejercer un papel

importante es en equipos de rescate, ya sean bomberos, policía, ejército ...

El proceso de localización consiste en dos fases: la fase de medida, durante la cual los agentes toman mediciones itra-nodo o inter-nodo utilizando sensores, y la fase de actualización, durante la cual los agentes, o nodos, infieren sus propias posiciones conociendo su posición previa, y varias mediciones. La precisión durante el posicionamiento depende fuertemente de la calidad de las mediciones que son afectadas por el medio, la topología, propagación multicamino, ruidos, deriva de reloj, etc (3).

En el caso de la localización, la posición de cada agente ha de ser conocida. Una forma de conocerla es midiendo la distancia entre cada nodo y los demás que componen la red, y complementarlo con la medición del ángulo de llegada de cada señal entrante a cada agente. A continuación se introducen diversas técnicas de medición de estas dos magnitudes.

Medición de distancias: potencia de la señal recibida (RSS: Received Signal Strength)

La potencia de la señal recibida en cada agente depende de la potencia inicial de la señal en el agente transmisor, o posición de referencia, P_0, y de la distancia entre los dos agentes, o entre el agente y el anclaje. La potencia de la señal decae cuadráticamente con la distancia. Midiendo esta potencia en el receptor, y conociendo P_0, la distancia puede ser calculada. La ecuación 1.1 permite calcular esa distancia:

$$P(d) = P_0 - 10n_p \log \frac{d}{d_0} \tag{1.1}$$

Siendo n_p el coeficiente de perdidas del medio, y d_0 la distancia de referencia del nodo transmisor donde se ha medido P_0.

Esta técnica no es compleja, pero no es muy precisa y cualquier variación en el medio afecta de forma muy significativa al coeficiente de pérdidas y, por lo tanto, a la potencia recibida. Otro aspecto que modifica la potencia de la señal son las baterías del transmisor. La señal de referencia será menor de la esperada si el transmisor tiene poca batería, y ello influirá a la potencia recibida. Para corregir estos aspectos, se ha de transmitir más información entre nodos para que estos la procesen y el posicionamiento sea correcto.

Medición de distancias: Tiempo de llegada (TOA: Time of arrival)

TOA es una técnica más precisa que RSS para calcular distancias entre dos agentes. Se basa en calcular el tiempo entre que se emite la señal en el transmisor y llega al receptor. La metodología es la siguiente. Un nodo transmite una señal al medio. Los demás nodos de la red reciben esa señal y retransmiten una respuesta a la misma. Para que la reciba el primer nodo. El nodo transmisor durante este tiempo ha estado esperando hasta la llegada de las respuestas. Con un temporizador interno, mide el tiempo que ha

transcurrido desde la transmisión hasta la recepción de las respuestas. Una vez obtenidos estos tiempos, calcula las distancias con la ecuación 1.2.

$$t = \frac{2d - D}{c} \tag{1.2}$$

Donde t es el tiempo que mide el nodo, d es la distancia entre nodos, D es el retardo que se produce entre la recepción del broadcast y el envío de la respuesta en cada nodo, magnitud que es conocida, y c es la velocidad de la propagación de la señal en el medio.

Esta técnica presenta el problema de la propagación multicamino. Generalmente la primera señal recibida es la correspondiente a la trayectoria directa entre nodos, pero puede haber situaciones en las que la velocidad de propagación de la señal en la trayectoria directa sea menor que en alguna otra trayectoria provocada por alguna reflexión. Si este caso ocurre, la señal reflejada llega antes al nodo receptor, provocando una falsa medición. Señales de banda ancha pueden ser empleadas para reducir este aspecto (2; 3; 4). Las señales de banda ancha presentan mejor resolución temporal y entonces, la componente de trayectoria directa puede ser separada de otras contribuciones. A pesar de esto, es muy difícil tratar con señales tempranas multicamino.

Existe una modificación de esta técnica llamada *diferencia de tiempos de llegada (TDOA: Time diference of arrival)*. Se basa en el mismo principio que TOA pero con la variante de que la medida que toman ahora los nodos es la diferencia de tiempos entre las llegadas de las respuestas procedentes de los otros agentes de la red, y a partir de ahí, calcular las distancias.

Medición del ángulo: Ángulo de llegada (AOA: Angle of arrival)

Conocer la distancia entre nodos no es suficiente para saber la posición exacta de los mismos. Con una distancia conocida, d, un agente puede ser situado a lo largo de una circunferecia centrada en su posición y de radio d. Medir el ángulo por el que llegan las señales al dispositivo receptor permite saber la posición exacta en esa circunferencia. Si el receptor está compuesto por un entramado de m sensores, será capaz de obtener el ángulo por el que se recibe la señal debido a su diagrama de radiación. El sistema es similar al aparato auditivo humano. EL cual, mediante dos sensores, una persona puede saber el ángulo por donde le llega cada sonido. Este principio, trasladado a este problema, permite a los dispositivos estimar el ángulo de llegada.

Una breve introducción a la localización cooperativa se ha expuesto. La siguiente sección presenta el estado del arte del sistema que va a ser discutido durante este proyecto.

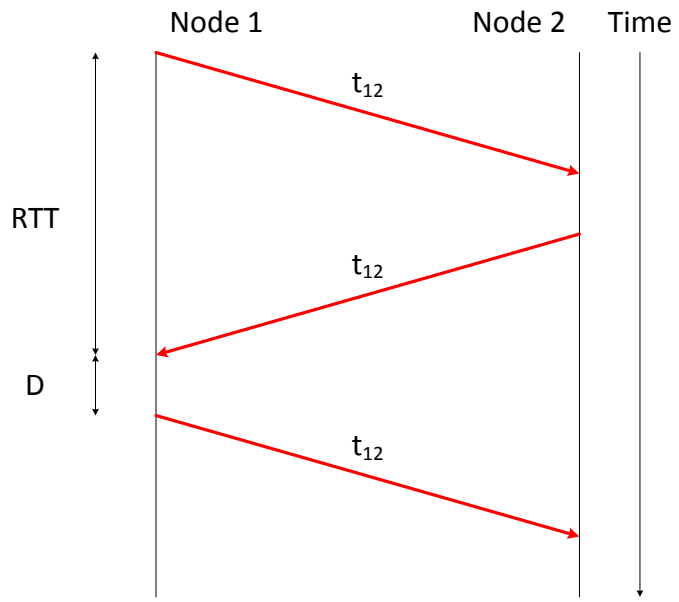

Figura 1.2: Transmisiones de pulsos en un escenario compuesto de dos nodos

1.3 Estado del arte: '3-way ranging'

Uno de sistemas de localización cooperativa mas extendidos es el conocido '3-way ranging'. Este sistema se basa en la técnica de tiempo de llegada (TOA) para calcular las distancias entre dispositivos. '3-way ranging' está diseñado para funcinar en redes de n nodos los cuales están en comunicación directa en un determinado escenario. Esta metodología se basa en la transmisión de pulsos por parte de cada nodo hacia los demás componentes de la red, y esperar a recibir los pulsos de respuesta de los mismos.

Se supone un escenario que contiene dos nodos. Cuando el sistema empieza a funcionar, el nodo etiquetado como nodo 1 transmite un pulso broadcast a la espera que lo reciban los demás componentes de la red, en este caso, el nodo etiquetado como nodo 2. Una vez el segundo nodo recibe el pulso, lo procesa, con un tiempo de procesado conocido por ambos nodos, y envía otro pulso de respuesta. Esta vez, es el nodo 1 el que lo recibe. El nodo 1 mide el tiempo transcurrido entre el envío y la recepción y calcula la distancia entre ambos. De esta forma, el primer nodo ya conoce su distancia con el segundo, pero el segundo no, luego es necesario que el nodo 1 vuelva a transmitir de nuevo otro pulso, tras un tiempo de procesado conocido, para que, mediante el mismo proceso anterior, el segundo nodo también conozca su distancia con el primero. Esto es, tres transmisiones de pulsos son necesarias para un completo conocimiento de cada nodo de las distancias de todos los demás componentes de la red con sí mismo. El nombre de esta técnica proviene de este hecho. La figure 1.2 muestra gráficamente el proceso entre dos nodos.

D es el restardo de procesado, t_{12} es el tiempo de propagación entre nodos, y RTT

(Round Trip Time) es el tiempo entre la transmisión y la recepción en cada nodo.

Este procedimiento puede ser trasladado a escenarios con un mayor número de nodos. La metodología es la misma que en el escenario con dos nodos para cada enlace directo entre cada pareja de agentes. En un escenario con tres nodos, el proceso debe ser realizado tres veces, dado que el número de enlaces directos entre nodos aumenta hasta tres. Con cuatro dispositivos, los enlaces directos suben a seis, y así sucesivamente. El número de veces que se ha de realizar el '3-way ranging' depende del número de nodos, y puede ser calculado mediante la ecuación 1.3:

$$c_2^n = \frac{n!}{2!\,(n-2)} = \frac{n\,(n-1)}{2} \tag{1.3}$$

De este modo, como son necesarios tres tranmisiones para tener conocimiento completo de las distancias por cada enlace, el número total de transmisiones necesarias es:

$$N_{tx} = 3c_2^n = \frac{3}{2}n\,(n-1) \tag{1.4}$$

Realizando '3-way ranging' solamente la distancia es calculada. Para tener un completo conocimiento de la posición de los demás dispositivos del escenario se requiere dotar a los mismos con alguna característica extra, como la ya mencionada anteriormente, un entramado de antenas para permitir medir el ángulo de llegada de las señales. Conociendo el ángulo y la distancia la posición ya es conocida. Otra forma de solucionar esta cuestión es dotar a los nodos de cierta capacidad de procesado de señal y de este modo ser capaz de realizar un seguimiento de cada nodo.

Una forma de obtener todas las distacias se propone en (5). Más adelante, diversos aspectos de este método van a ser discutidos.

1.4 Plan del proyecto

Durante los siguientes capítulos se propone una mejora de este método de posicionamiento. El primer capítulo trata sobre la secuencia y su aplicación al sistema. Después, un algoritmo basado en varias restricciones dadas por el mismo sistema es propuesto para generar una secuencia dependiendo del tamaño de la red. Una vez obtenido este algoritmo, se propone otro, basado en el mismo principio, para calcular todas las posibles secuencias para cada número de dispositivos. El siguiente capítulo habla sobre el procesado de señal necesario en cada dispositivo. Se introduce la implementación de un modelo adaptado al problema del filtro de Kalman en cada nodo de la red. Varias simulaciones han sido realizadas para escenarios de tres, cuatro, y cinco nodos, mostrando los resultados en diversas gráficas. El posterior capítulo. La parte final de este proyecto trata sobre la complejidad computacional que soporta cada

dispositivo, y se discuten maneras eficientes de la implementación del filtro de Kalman en cada uno de ellos. Por último, se realiza un estudio sobre el incremento de la carga computacional con respecto al tamaño de la red.

Capítulo 2

Localización cooperativa programada

2.1 El concepto

En el capítulo anterior se ha introducido una idea básica de posicionamiento basada en la transmisión de tres pulsos por par de dispositivos en una determinada red de nodos. Este sistema trabaja correctamente tanto en entornos interiores como en entornos exteriores. La simplicidad es la principal característica del sistema, esta simplicidad tiene como consecuencia que otros aspectos del mismo puedan ser mejorables. En este caso, mediante el procedimiento del '3-way ranging', cada nodo es capaz solamente de conocer la distancia entre sí mismo y los demás, desconociendo por completo las distancias existentes entre los demás dispositivos. Este aspecto puede ser mejorado de tal forma que cada nodo tenga un conocimiento completo de todas las distancias asociadas a todos los enlaces directos entre los dispositivos de la red, dotando a cada nodo de una mayor información del escenario. Otro aspecto a mejorar es la reducción del número de emisiones de pulsos con el fin de obtener una reducción del gasto de energía. La transmision de tres pulsos por enlace directo es inviable en redes con un alto número de componentes, con lo cual es importante que este número de transmisiones se vea reducido considerablemente para redes de gran tamaño. Para satisfacer estos dos aspectos, se propone fijar una secuencia de transmisiones de pulsos programada adaptada al número de dispositivos, esto es, fijar un orden de transmisión de señales por parte de cada nodo. Esta secuencia de transmisión es conocida por todos ellos y, aplicando cierto procesado de señal, se obtienen los resultados. Al dotar a cada dispositivo de un procesamiento de señal, la simplicidad se ve reducida, pero por contra, mejores resultados son obtenidos y, como se verá más adelante, el número de transmisiones totales se verá disminuido considerablemente en redes grandes.

La figura 2.1 muestra un diagrama temporal básico de un escenario compuesto por tres nodos. La secuencia empleada en esta figura es $\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{2}, \mathbf{1}, \mathbf{3}$. Esta secuencia se

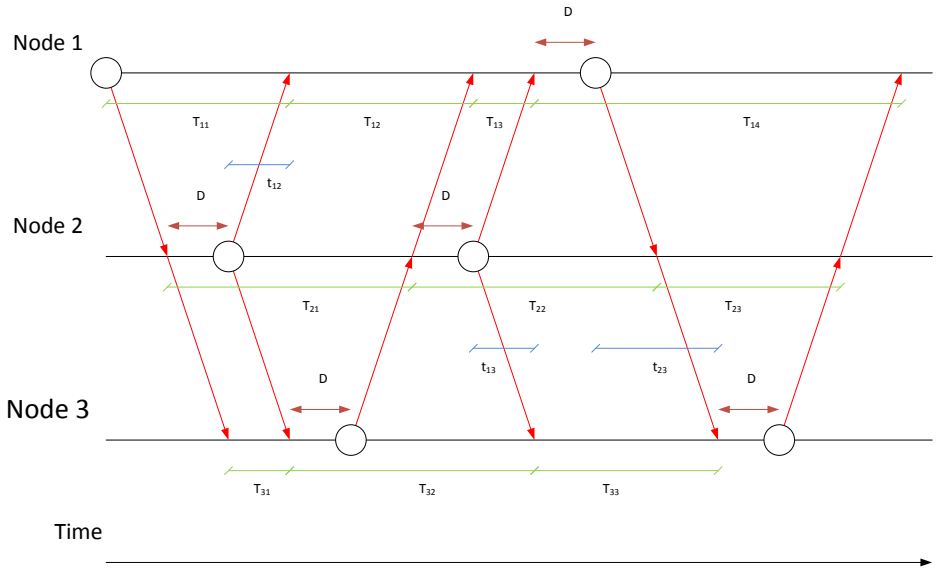

Figura 2.1: Diagrama temporal de un escenario compuesto por tres nodos

repite cíclicamente. En este caso, los nodos ya no miden el tiempo de propagación de los pulsos en cada enlace, sino que miden la diferencia de tiempo entre llegada de los diferentes pulsos procedentes de cada nodo. Esta diferencia de tiempos se muestra en la figura 2.1 como T_{ij}, donde i es el número del nodo en la secuencia, y j es el número de medida durante la misma. En este ejemplo, $i, j = 1, 2, 3$. Otro aspecto a tener en cuenta es el tiempo de procesado en cada nodo, dado que tiene que ser incluido en el procesado de señal de cada dispositivo para evitar resultados erróneos. Esa configuración propuesta va a ser usada durante toda la memoria.

Una vez medidas estas diferencias de tiempos de llegada de los distintos pulsos procedentes de los demás nodos, cierto procesamiento de datos ha de llevarse a cabo para que cada dispositivo conozca los tiempos de propagación de los pulsos en la totalidad de los enlaces directos, que es en lo que se basa el '3-way ranging'. De las medidas tomadas en este sistema, se obtienen distintos sistemas de ecuaciones, teniendo como incógnitas estos tiempos de propagación. La tabla 3.1 contiene los sistemas de ecuaciones correspondientes a cada nodo, en el escenario expuesto anteriormente. Pueden ser deducidas observando la figura 2.1 con detenimiento. Las incógnitas t_{ij} corresponden a los tiempos de propagación entre los pulsos transmitidos por el nodo i y el nodo j, D corresponde al tiempo de procesado en cada nodo.

Estos sistemas pueden ser reescritos en forma matricial como:

$$\mathbf{T}_m = \mathbf{C}_m \mathbf{t} + \mathbf{D}_m \tag{2.1}$$

Node 1	**Node 2**	**Node 3**
$T_{11} = 2t_{12} + D$	$T_{12} = 2t_{23} + 2D$	$T_{12} = t_{12} + t_{23} - t_{13} + D$
$T_{21} = t_{23} + t_{13} - t_{12} + D$	$T_{22} = t_{13} + t_{12} - t_{23} + D$	$T_{23} = 2t_{13} + 2D$
$T_{31} = 2t_{13} + 2D$	$T_{32} = t_{13} + t_{23} - t_{12} + D$	$T_{33} = 2t_{23} + 2D$

Tabla 2.1: Sistemas de ecuaciones en cada nodo

Donde $\mathbf{T}_m$ es el vector de medición en el nodo m, $\mathbf{C}_m$ es la matriz de medición asociada al nodo m, $\mathbf{D}_m$ es el tiempo de procesado en el nodo m, y $\mathbf{t} = [t_{12}, t_{13}, t_{23}]^T$ es el vector de incógnitas.

Por lo tanto, en el nodo 1:

$$\mathbf{T}_1 = \begin{bmatrix} T_{11} \\ T_{12} \\ T_{13} \end{bmatrix} \quad \mathbf{C}_1 = \begin{bmatrix} 2 & 0 & 0 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \end{bmatrix} \quad \mathbf{T}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \tag{2.2}$$

En el nodo 2:

$$\mathbf{T}_2 = \begin{bmatrix} T_{21} \\ T_{22} \\ T_{23} \end{bmatrix} \quad \mathbf{C}_2 = \begin{bmatrix} 0 & 0 & 2 \\ 2 & 0 & 0 \\ -1 & 1 & 1 \end{bmatrix} \quad \mathbf{T}_2 = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix} \tag{2.3}$$

En el nodo 3:

$$\mathbf{T}_3 = \begin{bmatrix} T_{31} \\ T_{32} \\ T_{33} \end{bmatrix} \quad \mathbf{C}_3 = \begin{bmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & -1 \end{bmatrix} \quad \mathbf{T}_3 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \tag{2.4}$$

Mediante este set de ecuaciones el sistema está definido para tres nodos. Este concepto puede ser extendido a escenarios con m número de nodos, incrementándose la complejidad computacional en cada uno. Como el tamaño de la red aumenta, lo mismo sucede con la secuencia de transmisión de pulsos y, por lo tanto, el número de mediciones y ecuaciones.

El concepto está ya prácticamente definido. Para tener una completa definición del mismo es necesario saber diseñar una secuencia de transmisión de pulsos. El siguiente subapartado trata sobre este aspecto.

2.2 La secuencia

Crear la secuencia es un aspecto muy importante del sistema, ya que no todas son válidas. Como puede observarse en la tabla 3.1, se obtiene un sistema lineal para cada dispositivo, por lo tanto, la secuencia ha de ser diseñada de forma que estos sistemas sean compatibles y determinados, esto es, que las ecuaciones obtenidas no sean linealmente

dependientes entre sí. Si existiese alguna dependencia lineal, el sistema tendría infinitas soluciones y no se alcanzaría la solución final.

Otra cuestión a tener en cuenta es el número de transmisiones por parte de cada nodo a lo largo de la secuencia. Estas transmisiones han de estar distribuidas uniformemente a lo largo de la misma. El motivo es que todos los nodos han de obtener su sistema compatible determinado. Si las transmisiones no se distribuyen uniformemente algunos nodos obtienen sistemas sobredimensionados, y otros, incompletos. Tomando como ejemplo la secuencia $\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{2}, \mathbf{1}, \mathbf{4}, \mathbf{1}$, el nodo 3 transmite una vez y recibe siete pulsos, obteniendo así un sistema sobredimensionado, mientras que el nodo 1 transmite tres veces y recibe 5 pulsos, obteniendo un sistema incompleto.

Por lo tanto, el objetivo es generar una secuencia con las transmisiones de pulsos por parte de los nodos que estén uniformemente distribuidas a lo largo de la secuencia, y evitar dependencias lineales entre ecuaciones.

Evitar dependencias lineales

Primero, es necesario conocer cómo se generan las ecuaciones en cada nodo. Una ecuación se genera cada vez que un nodo recibe un pulso. Tomando como ejemplo la secuencia anterior $\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{2}, \mathbf{1}, \mathbf{3}$, el nodo 1 genera ecuaciones midiendo la diferencia de tiempo entre la llegada de los pulsos en los intervalos $1-2, 2-3, 3-2$ y $2-3$ con una transmisión en medio. Destacar que, al contar el número de ecuaciones en el nodo i, este nodo es transparente, ya que las transmisiones no generan ecuaciones. En este ejemplo, $1-2-3-\mathbf{2}-\mathbf{1}-\mathbf{3}$, los números en negrita generan una ecuación, siendo la transmisión del nodo 1 invisible.

Puede observarse en esta secuencia que no exisite repetición de pares de nodos en la misma, es decir, ninguna subsecuencia de longitud 2 se repite en la secuencia principal. Si existiera alguna repetición de pares se generarían sistemas de ecuaciones con ecuaciones dobles y, por lo tanto, linealmente dependientes. De este modo, para evitar la aparición de dependencias lineales, basta con evitar la repetición de pares en la secuencia principal. Tomando como ejemplo la secuencia $\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{2}, \mathbf{3}, \mathbf{1}$ puede verse como la subsecuencia $2-3$ aparece dos veces. Al obtener el sistema de ecuaciones en el nodo 2 se observa lo siguiente:

$$\begin{aligned} T_{21} &= 2t_{23} + 2D \\ T_{22} &= 2t_{23} + 2D \\ T_{23} &= 2t_{12} + D \end{aligned}$$

Se aprecia que las dos primeras ecuaciones del sistema son la misma ecuación, llevando al sistema a ser incompleto, compatible indeterminado. Esto se produce debido

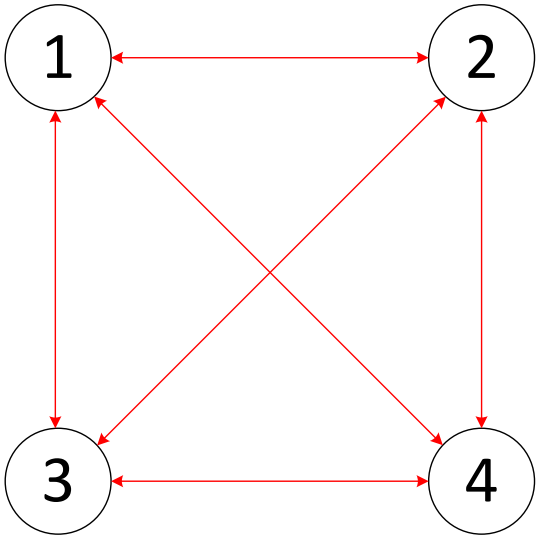

Figura 2.2: Conexiones en un escenario formado por cuatro dispositivos

a que ambas ecuaciones son producidas por la misma subsecuencia. Por lo tanto, se concluye que para evitar la aparición de estas dependencias, se debe evitar la repetición de subsecuencias dentro de una secuencia principal, pudiéndose repetir en cualquier repetición cíclica de la misma.

Tras esta conclusión, y sabiendo que las emisiones de pulsos han de estar uniformemente distribuidas, el siguiente paso es conocer el número mínimo de transmisiones de pulsos con el fin de obtener una secuencia óptima para un determinado tamaño de red.

2.3 Mínimo número de transmisiones de pulsos

Antes de entrar al problma del cálculo del número de transmisiones mínimas necesarias, se requiere saber cuantas incógnitas t_{ij} forman los diferentes sistemas de ecuaciones en cada dispositivo, esto es, el número de interconexiones directas entre los mismos. Este número se obtiene mediante la ecuación 2.5

$$N_{int} = c_2^n = \frac{n(n-1)}{2} \tag{2.5}$$

En la figura 2.2 se representa un escenario formado por cuatro nodos con sus seis correspondientes interconexiones.

El número de ecuaciones debe ser igual al número de enlaces directos. Contando el número de ecuaciones mediante el método introducido en el subapartado anterior, se obtienen los siguientes resultados almacenados en la tabla 2.2. También se realiza una comparación entre la localización cooperativa programada y el '3-way ranging' en cuanto

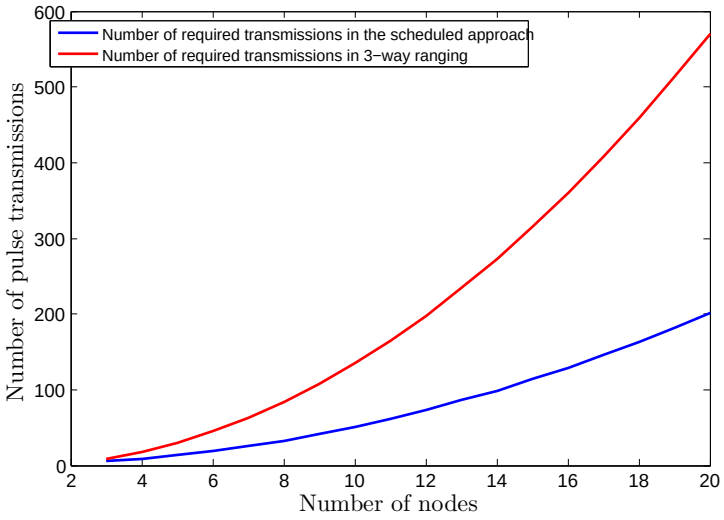

Figura 2.3: Comparación del número de transmisiones de pulsos entre '3-way ranging' y localización cooperativa programada

al número de transmisiones de pulsos. Se puede apreciar una reducción de las mismas considerable, siendo esta reducción mayor para redes grandes, redes donde el gasto de energía es mucho mayor.

Number of nodes	**3 - way ranging**	**Scheduled approach**
2	3	3
3	9	6
4	18	9
5	30	14
6	45	19
7	63	26
8	84	33
9	108	42
...	...	...
n	$3c_2^n$	$c_2^n + \lceil \frac{n}{2} \rceil + 1$

Tabla 2.2: Number of required transmissions against number of nodes

Por lo tanto, el número mínimo de emisiones de pulsos requerido puede expresarse mediante la ecuación 2.6

$$N_{tx} = c_2^n + \left\lceil \frac{n}{2} \right\rceil + 1 \tag{2.6}$$

La figura 2.3 muestra una comparación gráfica entre el número de transmisiones necesarios en cada uno de los dos sistemas de localización. Se observa que este número crece exponencialmente con el número de nodos en ambos sistemas. También se observa que, a mayor tamaño de la red de dispositivos, mayor es el ahorro de energía relativo.

En el anexo **??** se prueba esta expresión de número mínimo de transmisiones.

2.4 Generador de secuencias: una secuencia

En este instante, las características necesarias para generar una secuencia válida son conocidas. El siguiente paso es crear un generador de secuencias. Si el tamaño de la red es pequeño, hacerlo manualmente no resulta muy tedioso, pero crear una secuencia para un número grande de dispositivos es inviable hacerlo de forma manual. En esta sección se propone un algoritmo para generar una secuencia válida y óptima para un tamaño de la red dado. Tres son las restricciones a tener en cuenta: El número mínimo de transmisiones, evitar dependencias lineales, esto es, evitar repetir subsecuencias de longitud dos, y la uniformidad en el número de transmisiones de cada nodo.

Restricción 1: Mínimo número de transmisiones

El número mínimo de transmisiones es el primer valor a ser calculado. Este valor puede ser calculado mediante la ecuación 2.6. Este valor indicará el final del algoritmo.

Restricción 2: Evitar dependencias lineales

Como se ha explicado anteriormente, han de evitarse las repeticiones de las susbsecuencias de longitud dos a lo largo de la secuencia principal. Si el nodo j transmite justo después del nodo i, el nodo j ya no debe volver a transmitir después del nodo i hasta que no se termine la secuencia y comience de nuevo. Una forma de tratar con este problema es definir una matriz auxiliar, llamada *checkmatrix*. La checkmatrix es una matriz $n \times n$, donde n es el número de nodos de la red. Las filas i de la checkmatrix se asocian a los nodos que acaban justo de transmitir en un determinado instante, mientras que las columnas j se asocian a los nodos a transmitir en ese mismo instante. La checkmatrix está compuesta únicamente por valores 1 y 0. Un 0 en una determinada posición indica que esa subsecuencia i, j no ha sido utilizada previamente, luego el nodo j puede ser seleccionado para ser el próximo a transmitir en la secuencia principal. Sin embargo, el 1 indica lo contrario, que en el caso de seleccionar el nodo j, se crearían dependencias. Por ejemplo, en un escenario $n = 4$, si el nodo que acaba de transmitir es el 2, y la checkmatrix presenta este aspecto:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & \mathbf{0} & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

Tan solo el nodo 3 podría transmitir ya que $a_{23} = 0$, y a_{21}, a_{22} and $a_{24} = 1$. Esto significa que las subsecuencias $\mathbf{2-1}$ y $\mathbf{2-4}$ han aparecido previamente.

Al inicializar, la checkmatrix es la matriz identidad y se va actualizando guardando 1s en las posiciones adecuadas. La diagonal se inicializa con 1s debido a que no está permitido que un nodo transmita dos veces seguidas ya que también se crean dependencias lineales.

Restricción 3: Uniformidad en las transmisiones

La uniformidad puede conseguirse seleccionando los nodos posibles que hayan transmitido un menor número de veces hasta un determinado instante. Dependiendo de la cantidad de 1s que contenga la columna asociada a cada nodo en la checkmatrix, puede observarse cuántas veces ha participado ese nodo en la secuencia. La idea es, si existe más de un nodo disponible para transmitir en un determinado instante, seleccionar aquel que menos veces haya participado en las transmisiones, esto es, aquel con menor número de $1s$ en la columna asociada. De esta forma, se garantiza la uniformidad. Si, una vez tenida en cuenta esta restricción, todavía existe más de un nodo disponible, la elección es aleatoria, ya que no hay ningún otro requerimiento.

El algoritmo

La primera parte del algoritmo es la inicialización de cada elemento. Como se ha dicho anteriormente, la checkmatrix ha de inicializarse como la matriz identidad de tamaño $n \times n$. La secuencia también tiene que inicializarse, en este caso, con el primer nodo a transmitir. Por lo tanto la secuencia inicial está formada por una transmisión por parte del nodo 1. Después, el número óptimo de transmisiones ha de ser calculado. Mediante la ecuación 2.6 se obtiene ese valor.

Una vez conocido esto, el algoritmo comienza. El bucle ha de realizarse $N_{tx} - 1$ veces (la primera transmisión ya se ha definido) y cada vez que se completa una iteración, un nuevo elemento es añadido a la secuencia principal. Lo primero que se hace en cada iteración es buscar los nodos disponibles para transmitir en ese instante, mirando la posición de los 0s en la fila asociada al nodo que ha transmitido previamente. Una vez hecho esto la posición de estos 0s se guardan como posibles nodos a transmitir. Si solo existe un nodo disponible, se selecciona y se actualiza la checkmatrix y se pasa a la siguiente iteración. Si hay más de un nodo disponible, se cuenta el número de 1s de cada columna donde estaba situado cada 0 anterior. La columna con menos 1s es la seleccionada ya que es la asociada al nodo que menos veces ha sido seleccionado. Para ello, se crea una submatriz a partir de la checkmatrix, con las columnas asociadas a los nodos disponibles para transmitir. Sumando todas las columnas de esta submatriz, se selecciona la que menor suma se obtenga. Si hay varias columnas asociadas a cada nodo que suman el mismo número, la elección del nodo se hace de forma arbitraria. Al final de cada iteración, el nodo seleccionado ha de ser guardado como último nodo que ha transmitido, para así proceder con la siguiente iteración del algoritmo.

La tabla 2.3 guarda varias secuancias generadas mediante este algoritmo propuesto para diferentes tamaños de redes.

Nodes	**Transmissions**	**Schedule**
3	6	1 2 3 2 1 3
4	9	1 3 4 2 1 2 4 3 2
5	14	1 2 1 3 4 5 1 4 2 3 5 2 4 1
6	19	1 2 1 3 4 5 6 1 4 2 3 5 1 6 2 4 3 6 5
7	26	1 2 1 3 4 5 6 7 1 4 2 3 5 7 6 1 5 2 4 3 6 2 7 3 1 6

Tabla 2.3: Secuencias generadas para un determinado número de nodos

La tabla 2.4 muestra un resumen del algoritmo:

Algoritmo para generar una secuencia	
1.	Calcular la longitud de la secuencia
2.	Inicializar la checkmatrix como matriz identidad y la secuencia como 1
3.	Buscar en la checkmatrix los nodos disponibles a transmitir
4.	Seleccionar el nodo con menor número de transmisiones previas, escogiendo la columna con menos 1s
5.	Comprobar que la fila asociada al nodo seleccionado no está compuesta por un vector de 1s
6.	Añadir el número de nodo seleccionado a la secuencia
7.	Actualizar la checkmatrix
8.	Colocar el nodo seleccionado como nodo que ha realizado la última transmisión, y repetir desde el paso 3 hasta completar la longitud de la secuencia

Tabla 2.4: Algoritmo para generar una secuencia

Algorithm 1 Construcción de la secuencia

1: **Initialize:** $n = 1, m = 1, N_{tx} = 1, g_{11} = 1, \mathcal{T} = \{1\}$
2: **while** $(N_{tx} \leq P)$ **do**
3: $\quad s_j = \sum_i g_{ij} \quad j = 1, 2, 3, \cdots, N; j \neq m$
4: $\quad m = \arg\min\{s_j\}$
5: $\quad g_{nm} = 1,\ \mathcal{T} = \{\mathcal{T}, m\},\ N_{tx} = N_{tx} + 1,\ n = m$
6: **end while**

2.5 Generador de secuencias: todas las posibles secuencias

En la sección anterior se han expuesto las condiciones para crear una secuencia válida para llevar a cabo la localización cooperativa programada. En este punto, se ha propuesto un algoritmo para generar esta secuencia. El siguiente paso es extender ese algoritmo para generar todas las posibles secuencias dado un determinado número de nodos. A primera vista, puede observarse que la cantidad de secuencias crece realmente

rápido con el tamaño de la red. La tabla 2.5 muestra una aproximación de la cantidad de secuencias que se obtendrían dependiendo del número de nodos.

$$N_{schd} \leq n^{(N_{tx}-1)} \tag{2.7}$$

La ecuación 2.7 devuelve el número total de combinaciones que se deberían probar empleando el método de fuerza bruta, es decir, probar todas las posibles combinaciones. n es el número de componentes en la red y N_{tx} es el número de transmisiones necesarias en la secuencia, su longitud.

Nodes	**Transmissions**	**Combinations to prove**
3	6	243
4	9	65536
5	14	$1{,}22 \times 10^{9}$
6	19	$1{,}01 \times 10^{14}$

Tabla 2.5: Número de combinaciones a probar con respecto al número de nodos

Nodes	**Number of schedules**
2	1
3	6
4	498
5	438144

Tabla 2.6: Número de secuencias válidas según el número de nodos

La tabla 2.6 indica el número de secuencias válidas para cada número de nodos.

La idea de este nuevo algoritmo es escanear las posibilidades en un diagrama de tipo árbol. Inicializar la secuencia con el 1, e ir añadiendo posibles nodos. La idea puede verse gráficamente en la figura 2.5 para un ejemplo de un escenario con 3 nodos. Cada círculo representa un nodo, las líneas negras las ramas del árbol y la línea roja, el proceso de escaneo.

Primero, el algoritmo añade la subsecuencia $2, 1, 3, 2, 3$ al 1 inicial, uno a uno. Cuando alcanza el número óptimo de transmisiones, retrocede una posición y prueba con el 1 en la última posición. Esta rama también llega a su final, luego esta vez retrocede dos veces y prueba con la siguiente rama superior. Mediante este proceso se escanean las posibilidades.

Primero, se inicializa la secuencai a 1 y se crea la checkmatrix identidad asociada. En este caso, cada rama va a tener su propia checkmatrix asociada, cada vez que se crean nuevos niveles en el árbol, la checkmatrix se divide en tantas ramas como sean necesarias. Una vez inicializado todo, se ejecuta una función recursiva que escanee el árbol, y añada el nodo oportuno a la secuencia.

El primer deber de esta función es realizar una comprobación de la checkmatrix asociada a la rama que se está ejecutando, para buscar nodos posibles nodos. Si no hay

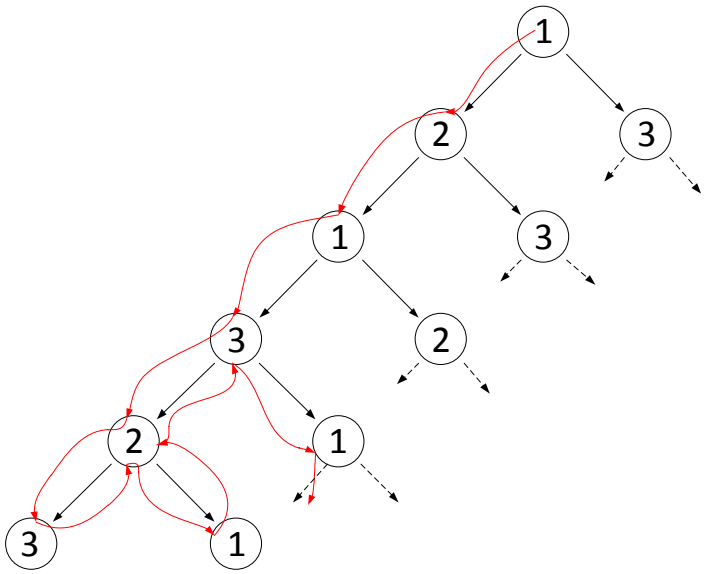

Figura 2.4: Árbol del algoritmo

nodos posibles, o la longitud de la secuencia es la óptima, se alcanza el final de la rama y la función termina sin añadir ningún nodo. Si hay nodos posibles, se añade el primer nodo disponible y la checkmatrix se actualiza. Se ha creado en estos momentos una nueva rama, con su checkmatrix actualizada. En este momento la función se llama a sí misma para continuar añadiendo nodos.

Viendo el ejemplo de la figura, la primera llamada a la función devuelve como nodos posibles $\{2, 3\}$. Entonces, añadirá el nodo 2 a la secuencia, actualizará la checkmatrix y volverá a llamarse a sí misma. Ahora devolverá como posibles nodos $\{1, 3\}$, añadirá el 1 a la secuencia, actualizará de nuevo la checkmatrix, y se volverá a llamar. Este proceso continúa hasa la última rama. En la última rama del ejemplo, el nodo 2 ha transmitido y devuelve $\{3, 1\}$. Seleccionará el nodo 3 y, como se ha llegado al final, retornará a la función madre, y esta función madre añadirá el siguiente nodo al añadido anteriormente y continuará el proceso. Siguiendo este proceso al final se obtienen todas las secuencias posibles.

Con esta función, se obtienen muchas secuencias, pero no todas son válidas ya que no se ha tenido en cuenta la propiedad de la uniformidad. Para seleccionar las secuencias correctas, se cuenta cuántas veces ha participado cada nodo. Si la diferencia entre transmisiones del más activo al menos activo es mayor que 1, no se alcanza uniformidad en las mismas, y la secuencia no es válida. Se ha de tener en cuenta el hecho de que el nodo 1 puede participar una vez más ya que la primera transmisión genera ecuación.

Con todo esto, todas las posibles secuencias son obtenidas. Ha de tenerse muy en cuenta a la hora de ejecutar este algoritmo que para redes mayores de 5 nodos, su tiempo de ejecución se hace muy largo dado que la cantidad de secuencias es enorme.

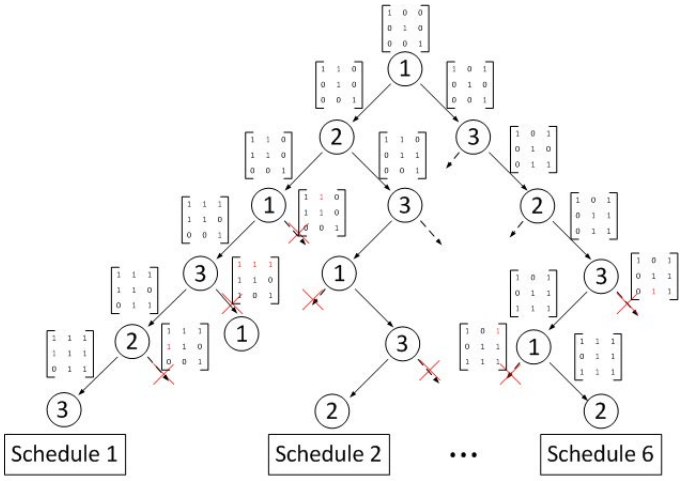

Figura 2.5: Árbol del algoritmo

2.6 Tiempo de procesado

El último aspecto a definir en el problema de la localización cooperativa programada es el tiempo de procesado de los pulsos en cada dispositivo. Este valor es conocido por todos los miembros de la red y ha de cumplir:

$$D > t_{max} \tag{2.8}$$

Donde t_{max} es el tiempo de propagación del pulso si dos nodos se encuentran situados en los puntos más alejados del escenario. Si la ecuación 2.8 no se cumple, los pulsos podrían recibirse en un orden distinto al de la secuencia y, por lo tanto, el sistema no funcionaría adecuadamente. Con lo cual, es muy importante diseñar este parámetro conforme al escenario donde va a ser empleado el sistema.

2.7 Resumen

En este capítulo se ha expuesto un nuevo sistema de localización cooperativa basada en la transmisión de pulsos en orden programado. Se ha demostrado que, aunque la complejidad aumenta comparado con el sistema '3-way ranging', la cantidad de transmisiones necesarias disminuye considerablemente, y con ellas, la energía gastada. Además, se consigue un conocimiento completo por parte de todos los dispositivos de la red de las distancias entre ellos. Después, se han propuesto algoritmos para generar estas secuencias de trasnmisión de pulsos, basadas en las restricciones que el propio sistema impone. Se han propuesto algoritmos para generar una secuencia, o todas las posibles secuencias. Al final, el problema del tiempo de procesado por parte de cada nodo se ha introducido como un aspecto importante para el correcto funcionamiento del sistema.

Capítulo 3

El filtro de Kalman y las secuencias

Una vez se tiene generada la secuencia de transmisión de pulsos por parte de los nodos de la red, se necesita de un algoritmo capaz de procesar las mediciones realizadas en cada uno de ellos. Como el sistema está funcionando en un entrono ruidoso y cambiante, se requiere que el algoritmo sea adaptativo. Este algoritmo adaptativo debe ser capaz de calcular las distancias a pesar del movimiento de los dispositivos y de las perturbaciones. El algoritmo escogido es el filtro de Kalman. Este filtro se basa en dos principales ecuaciones.

- Ecuación de procesado:

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{m}_k \tag{3.1}$$

  Donde $\mathbf{x}$ es el vector de estado del escenario, $\mathbf{F}$ es la matriz de transiciones entre estados, y $\mathbf{m}$ es el ruido de procesado, el cual es un ruido blanco gaussiano de media cero.

- Ecuación de medición

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{n}_k \tag{3.2}$$

  Donde $\mathbf{x}_k$ es el vector de estado, $\mathbf{H}$ es la matriz de medición, y $\mathbf{n}$ es el ruido de medición, cuyas características se exponen más adelante.

Para adaptar estas ecuaciones al problema, lo primero es definir qué representa cada una de las variables de las mismas en el sistema propuesto. La variable a estimar es la distancia entre todos los nodos de la red, por lo tanto, el vector de estado estará compuesto por este conjunto de valores, $\mathbf{x} = [d_{12}, d_{13}, \ldots, d_{1N}, d_{23}, d_{24}, \ldots, d_{N-1,N}]$, donde N es el número de nodos presentes en el sistema. Este vector de distancias está altamente relacionado con los tiempos de propagación de los pulsos entre dispositivos. El tamaño de este vector de estado depende

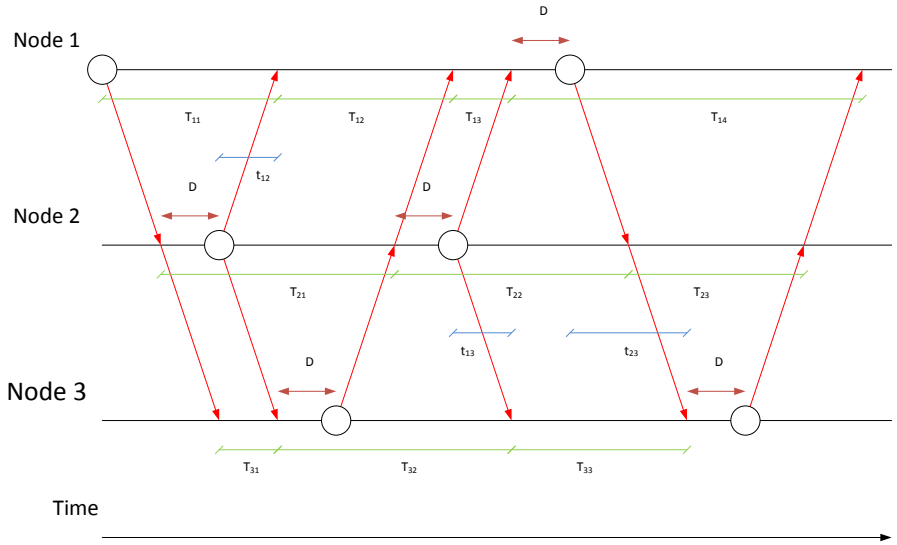

Figura 3.1: Diagrama temporal de transmisiones para tres nodos con secuencia 1-2-3-2-1-3

del tamaño de la red, y es igual al número de enlaces directos entre ellos. Este tamaño M puede calcularse mediante la expresion 3.3.

$$M = \frac{n(n-1)}{2} \tag{3.3}$$

El vector de medición $\mathbf{z}$ es el tiempo transcurrido entre la recepción de dos pulsos en cada nodo $\mathbf{z}_i = [T_{i1}, T_{i2}, \ldots T_{iN}]$. Donde el subíndice i es el número de nodo en la secuencia, y N es el número total de ecuaciones. Este número puede calcularse mediante la ecuación 2.6.

$$N = 1 + \frac{n(n-1)}{2} + \left\lceil \frac{n}{2} \right\rceil$$

Como no hay dependencias entre el presente estado y el próximo, la matriz de transiciones entre estados es la matriz identidad del tamaño acorde al tamaño del vector de estado, $\mathbf{F}_M = \mathbf{I}_{M \times M}$, donde M es el tamaño del vector de estado $\mathbf{x}$. La matriz de mediciones se extrae del sistema, ya que depende de la secuencia. Esta matriz $\mathbf{H}$ coincide con la matriz $\mathbf{C_m}$ en la ecuación 2.1.

Para ilustrar el problema, se han realizado varias simulaciones con distinto número de nodos.

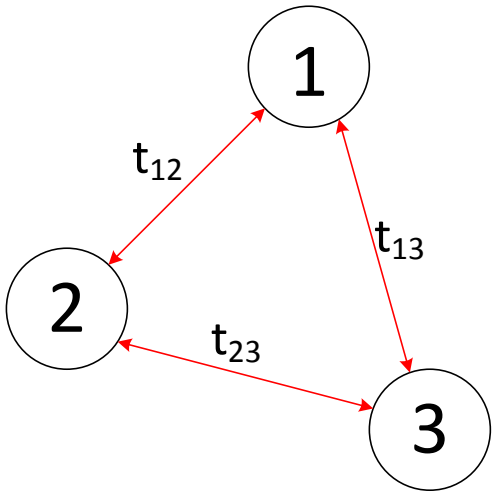

Figura 3.2: Escenario con tres nodos

3.1 Red de 3 nodos

En un escenario con tres nodos, tres son los rangos a ser calculados, por lo tanto, el vector de estado es $\mathbf{x} = [t_{12}, t_{13}, t_{23}]$. Se han escogido tiempos para el vector de estado en lugar de distancias por simplicidad. Para transformar el vector de tiempos en distancias tan solo hay que multiplicarlo por la velocidad de propagación de los pulsos, la velocidad de la luz en este caso. El vector de medición es $\mathbf{z}_i = [T_{i1}, T_{i2}, T_{i3}]$. Como puede observarse, en este sistema, $M = N = 3$. La figura 3.2 muestra un escenario estándar con tres nodos.

Mirando la figura 3.1, las ecuaciones de cada nodo son obtenidas. Las ecuaciones se almacenan en la tabla 3.1.

Node 1	**Node 2**	**Node 3**
$T_{11} = 2t_{12} + D$	$T_{21} = 2t_{23} + 2D$	$T_{31} = t_{12} - t_{13} + t_{23} + D$
$T_{12} = -t_{12} + t_{13} + t_{23} + D$	$T_{22} = 2t_{12} + 2D$	$T_{32} = t_{23} + 2D$
$T_{13} = t_{12} - t_{13} + t_{23} + D$	$T_{23} = -t_{12} + t_{13} + t_{23} + D$	$T_{33} = t_{12} + t_{13} - t_{23} + D$

Tabla 3.1: Conjunto de ecuaciones en cada nodo

Donde D es el tiempo de procesado en cada dispositivo, el cual es conocido por toda la red. Si se escriben las ecuaciones de forma más general:

$$\mathbf{T}_i = \mathbf{H}_i\mathbf{t} + \mathbf{D}_i + \mathbf{n}_i \tag{3.4}$$

Donde $\mathbf{D}_i$ es el vector que contiene todos los tiempos de procesado en el nodo i, y $\mathbf{n}_i$ es el ruido de medición en el nodo i.

Mirando a la tabla 3.1, las matrices de medición y los vectores de tiempos de procesado pueden obtenerse.

$$\mathbf{H}_1 = \begin{bmatrix} 2 & 0 & 0 \\ -1 & 1 & 1 \\ 1 & -1 & 1 \end{bmatrix} \quad \mathbf{D}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\mathbf{H}_2 = \begin{bmatrix} 0 & 0 & 2 \\ 2 & 0 & 0 \\ -1 & 1 & 1 \end{bmatrix} \quad \mathbf{D}_1 = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}$$

$$\mathbf{H}_1 = \begin{bmatrix} 1 & -1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & -1 \end{bmatrix} \quad \mathbf{D}_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

El sistema está ya definido. El siguiente paso es generar el escenario donde se va a realizar la simulación. Será un escenario 2D de 1000×1000 metros con tres dispositivos colocados de forma aleatoria en él. Una vez el escenario ha sido generado, un movimiento aleatorio es aplicado a cada uno de los nodos. La forma de generar este movimiento aleatorio es aplicar una aceleración aleatoria en una dirección aleatoria a cada uno, partiendo de una posición inicial estática, o con una velocidad inicial prefijada.

Dada la posición inicial del nodo, la velocidad inicial, la aceleración, el tiempo de muestreo, el tiempo total de la simulación y el tiempo que el nodo está en reposo antes de que el movimiento comience, el movimiento total del nodo puede ser calculado. Como puede verse en la figura 3.3, cada nodo parte de una posición inicial y se mueve a lo largo del escenario aleatoriamente.

Ahora que el escenario está completamente creado, se requiere generar las mediciones en cada nodo. EL tiempo de propagación de los pulsos es necesario que sea conocido para generar las mediciones, por lo tanto, es necesario calcular la distancia euclídea entre nodos y posteriormente dividir por la velocidad de propagación. Después, estos tiempos de propagación obtenidos se multiplican por la matriz de mediciones $\mathbf{H}$ y después, se le añaden los tiempos de procesado. De esta forma, se han creado las mediciones en cada nodo, mediciones libres de ruido.

$$\mathbf{z}_{i_{clean}} = \mathbf{H}_i\mathbf{t} + \mathbf{D}_i \tag{3.5}$$

Donde el subíndice i es el número del nodo en la secuencia.

El siguiente paso es definir los parámetros del filtro. Como la red se compone de tres nodos y son necesarias tres mediciones, el tamaño del vector de estado $\mathbf{x}$, M, es igual al vector de medición $\mathbf{z}$, N, tres. La matriz de medición considerado es la matriz de secuenciación $\mathbf{H}_i$ definida anteriormente y la matriz de transicion de estados será la

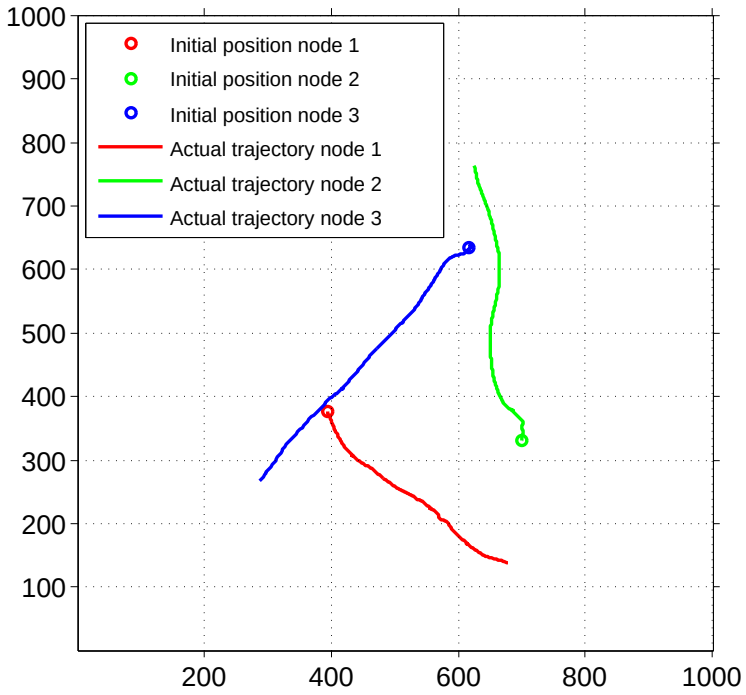

Figura 3.3: Movimiento de tres nodos

matriz identidad de tamaño M. El ruido de procesado es gaussiano de media 0. El ruido de medición es algo más complejo. Es también de distribución gaussiana de media 0, pero la desviación típica aumenta de forma exponencial con la distancia. El modelo escogido es $n(t) = \mathcal{N}(0, \sigma_0 e^{\frac{t}{2}})$, donde t es el tiempo de propagación del pulso entre los nodos. (6)

Una vez se ha generado el ruido, las medidas ruidosas pueden generarse añadiendo el ruido de medición a las medidas limpias previamente calculadas, y crear el vector $\mathbf{z}_i$.

Antes de poner en marcha el filtro, es necesario inicializar el vector de estado y la matriz de covarianzas del error de predicción.

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathbf{P}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

En este punto, el filtro está listo para ser ejecutado en los nodos. Un aspecto importante a tener en cuenta es que el ruido de medición no es estacionario, por lo tanto, la matriz de covarianzas del ruido de medición es necesario calcularla en cada iteración del filtro, tiene que ser actualizada cada vez que las distancias cambian. Esto implica que dentro del bucle del filtro, una parte se dedicará a actualizar esta matriz antes de ejecutar las ecuaciones del filtro en sí. Después de actualizar la matriz de covarianzas del ruido de medición, se computan las ecuaciones del filtro:

$$\begin{aligned}
\mathbf{x}_k^+ &= \mathbf{F}_k\mathbf{x}_k && (3.6)\\
\mathbf{P}_k^+ &= \mathbf{F}_k\mathbf{P}_k\mathbf{F}_k + \mathbf{Q}_M && (3.7)\\
\mathbf{K}_k &= \mathbf{F}_k\mathbf{P}_k^+\mathbf{H}^T\left[\mathbf{H}\mathbf{P}_k^+\mathbf{H}^T + \mathbf{Q}_N\right]^{-1} && (3.8)\\
\mathbf{x}_{k+1} &= \mathbf{x}_k^+ + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}\mathbf{x}_k^+ - \mathbf{d}) && (3.9)\\
\mathbf{P}_{k+1} &= \left[\mathbf{I} - \mathbf{K}_k\mathbf{H}\right]\mathbf{P}_k^+ + \mathbf{Q}_M && (3.10)
\end{aligned}$$

Las ecuaciones 3.6 y 3.10 se utilizan para computar el filtro cada N mediciones, esto es, cada vez que la secuencia termina. Lo recomendable es diseñar un filtro que se actualice cada vez que un pulso llega al nodo, es decir, en cada medida tomada. Teniendo en cuenta este hecho, puede rediseñarse el filtro obteniéndose el siguiente conjunto de ecuaciones:

$$\begin{aligned}
\mathbf{h} &= \mathbf{H}_j && (3.11)\\
\mathbf{x}_k^+ &= \mathbf{F}_k\mathbf{x}_k && (3.12)\\
\mathbf{P}_k^+ &= \mathbf{F}_k\mathbf{P}_k\mathbf{F}_k + \mathbf{Q}_M && (3.13)\\
\mathbf{K}_k &= \mathbf{F}_k\mathbf{P}_k^+\mathbf{h}^T\left[\mathbf{h}\mathbf{P}_k^+\mathbf{h}^T + \mathbf{Q}_N\right]^{-1} && (3.14)\\
\alpha_k &= z_k - \mathbf{h}\mathbf{x}_k^+ - \mathbf{d} && (3.15)\\
\mathbf{x}_{k+1} &= \mathbf{z}_k - \mathbf{K}_k\alpha && (3.16)\\
\mathbf{P}_{k+1} &= \left[\mathbf{I} - \mathbf{K}_k\mathbf{H}\right]\mathbf{P}_k^+ + \mathbf{Q}_M && (3.17)
\end{aligned}$$

Donde $\mathbf{h}$ es el vector fila j de $\mathbf{H}$ el cual corresponde con la medición $\mathbf{T}_{ij}$ siendo i el nodo correspondiente. α es la innovación escalar relacionada con esta medición. La ganancia de kalman $\mathbf{K}$ se mantiene como una matriz $M \times N$ pero solo adaptada a una medición, no a toda la secuencia. Esta segunda versión es ejecutada durante 250 segundos y actualizada cada segundo. Los resultados se muestran en la figura 3.4.

En la figura 3.4, se muestra la distancia real y la distancia estimada en la misma gráfica. Puede verse que el tiempo de convergencia es relativamente bajo, y que las distancias son seguidas correctamente. Ejecutando el filtro en cada uno de los tres nodos, cada uno con su propia matriz de mediciones y sus propias medidas, se obtiene la figura 3.5. Se observa que cada nodo obtiene el mismo resultado, y que se obtiene un completo conocimiento de las distancias.

Llegado a este punto, es conveniente transformar estas distancias calculadas a posiciones con el fin de comparar las trayectorias reales con las estimadas representando ambas en una misma figura. La posición ha de ser calculada en cada escalón de tiempo, de forma que es necesario incluir código extra dentro del bucle del filtro una vez el vector distancias ha sido actualizado. Estas distancias actualizadas se guardan en un

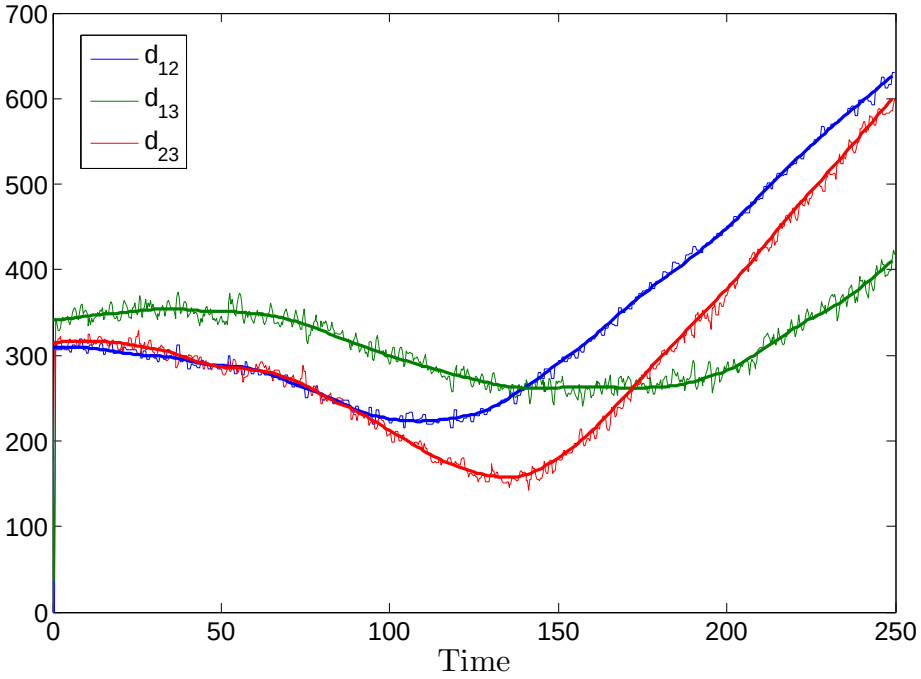

Figura 3.4: Evolución de las distancias en 250 s medidas en el nodo 1

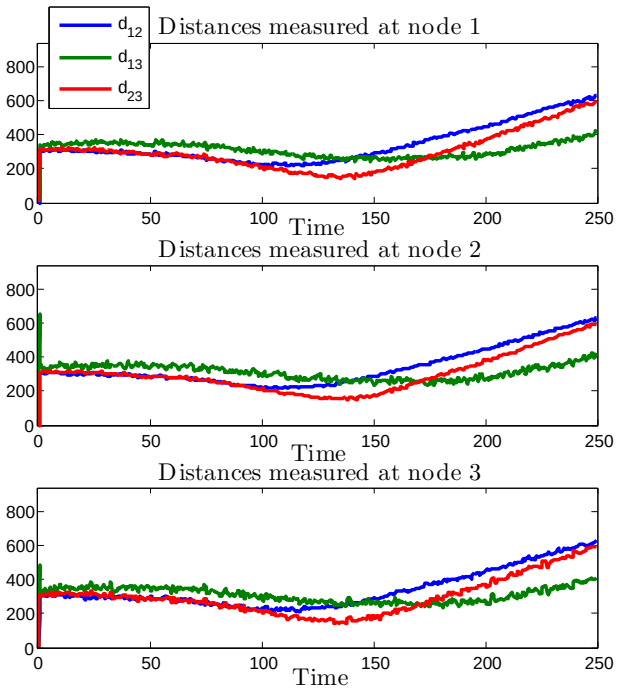

Figura 3.5: Evolución de las distancias en 250 s

vector columna de tamaño M. Tratando este vector columna con algunas funciones, las posiciones a partir de las distancias pueden ser calculadas. Estas funciones son *squareform*, *cmdscale* y *procrustes*.

- Squareform: Transforma el vector distancias de tamaño M en una matriz $\mathbf{S}$ de tamaño $M \times M$ donde cada elemento s_{ij} de $\mathbf{S}$ corresponde a d_{ij}, el cual es la distancia entre los nodos i y j.

$$\mathbf{d} = \begin{bmatrix} d_{12} \\ d_{13} \\ d_{23} \end{bmatrix}; \quad \mathbf{S} = \begin{bmatrix} 0 & d_{12} & d_{13} \\ d_{12} & 0 & d_{23} \\ d_{13} & d_{23} & 0 \end{bmatrix}$$

- Cmdscale: Parte de la matriz $\mathbf{S}$ y calcula una distribución espacial relativa en un espacio p dimensional usando la posición $\mathbf{0}$ como referencia.
- Procrustes: Traslada, rota, refleja y escala las coordenadas obtenidas mediante *cmdscale* para ajustar las mismas al problema.

Después de ejecutar estas tres funciones, la posición estimada queda determinada. En la figura 3.6 pueden observarse las estimaciones en el escenario generado, partiendo de las distancias obtenidas.

Algunos puntos no se ajustan a la trayectoria real, eso es debido al tiempo de convergencia del filtro. Al inicio, antes de que el filtro converja, no es capaz de estimar la posición y no devuelve resultados válidos.

3.2 Escenarios de 4 y 5 nodos

3.2.1 4 nodos

En un escenario de cuatro nodos, la secuencia $\mathbf{1-2-3-4-2-1-4-3-1}$ se ha obtenido aplicando el algoritmo propuesto, consiguiendo el diagrama temporal expuesto en la figura 3.7. La siguiente 6×6 matriz de secuenciación y 6×1 vector de tiempos de procesado son los obtenidos:

$$\mathbf{H} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 1 \end{bmatrix}; \quad \mathbf{D} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

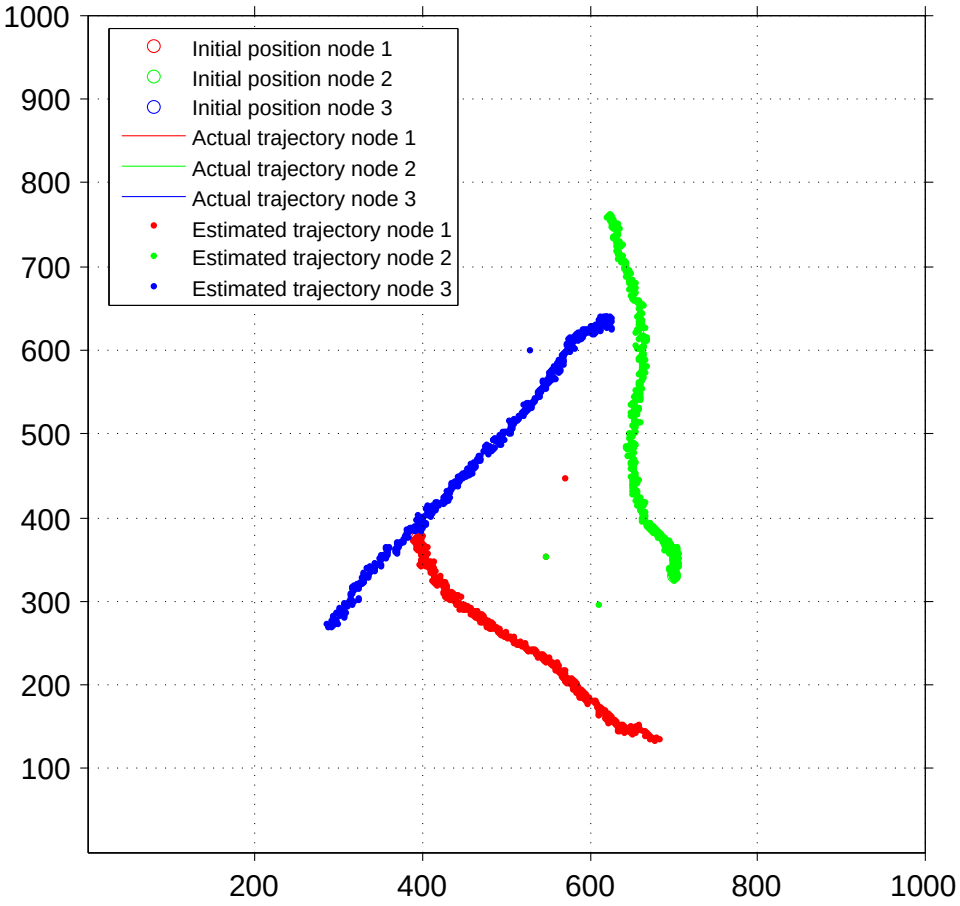

Figura 3.6: Movimiento estimado de los tres nodos

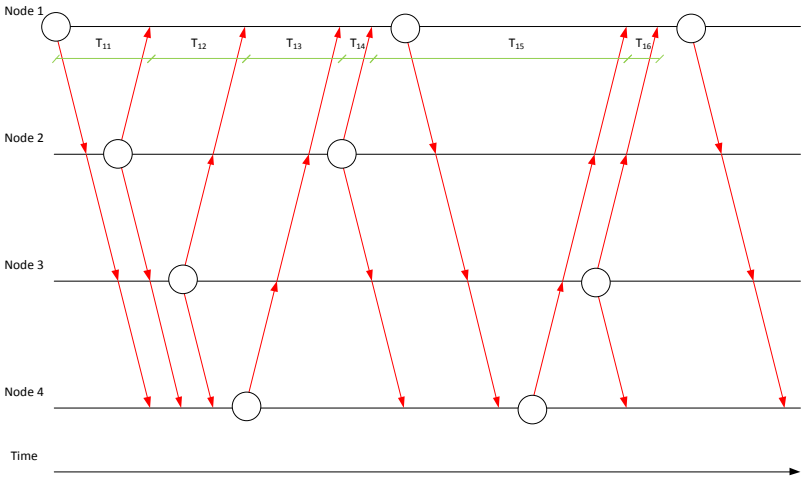

Figura 3.7: Diagrama temporal de una red de cuatro nodos

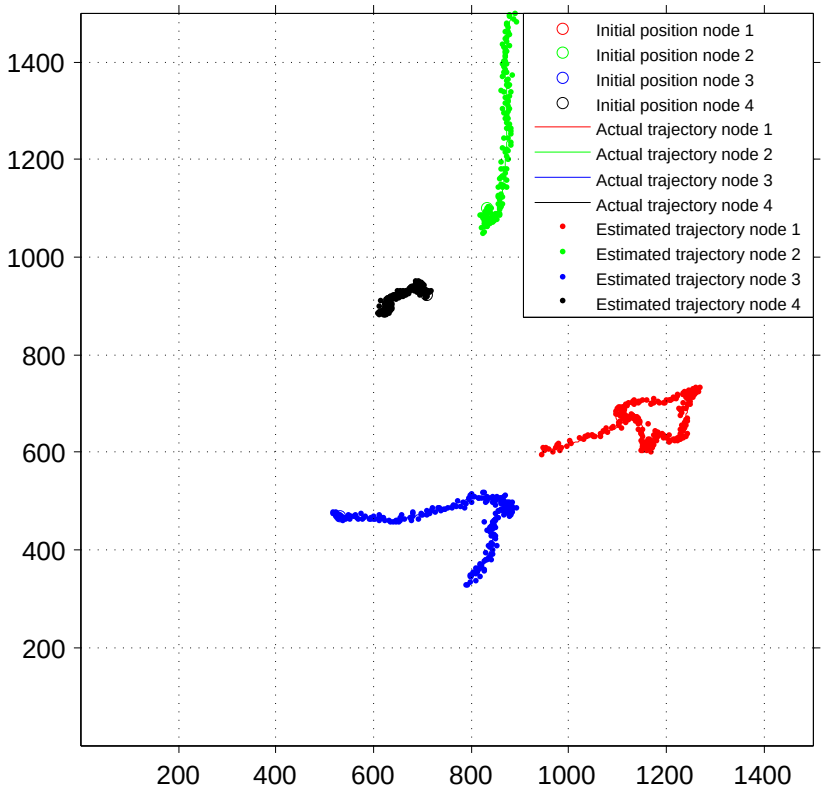

Figura 3.8: Movimiento estimado de los cuatro nodos

Los resultados después de ejecutar el filtro se observan en las figuras 3.8 y 3.9.

3.2.2 5 nodos

La secuencia calculada es $\mathbf{1-2-3-4-5-1-3-5-2-4-1-4-2-5}$, obteniendo el diagrama temporal mostrado en la figura 3.10. La siguiente 10×10 matriz de secuenciación y 10×1 vector de tiempos de procesado son los obtenidos:

$$\mathbf{H} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}; \quad \mathbf{D} = \begin{bmatrix} 2 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

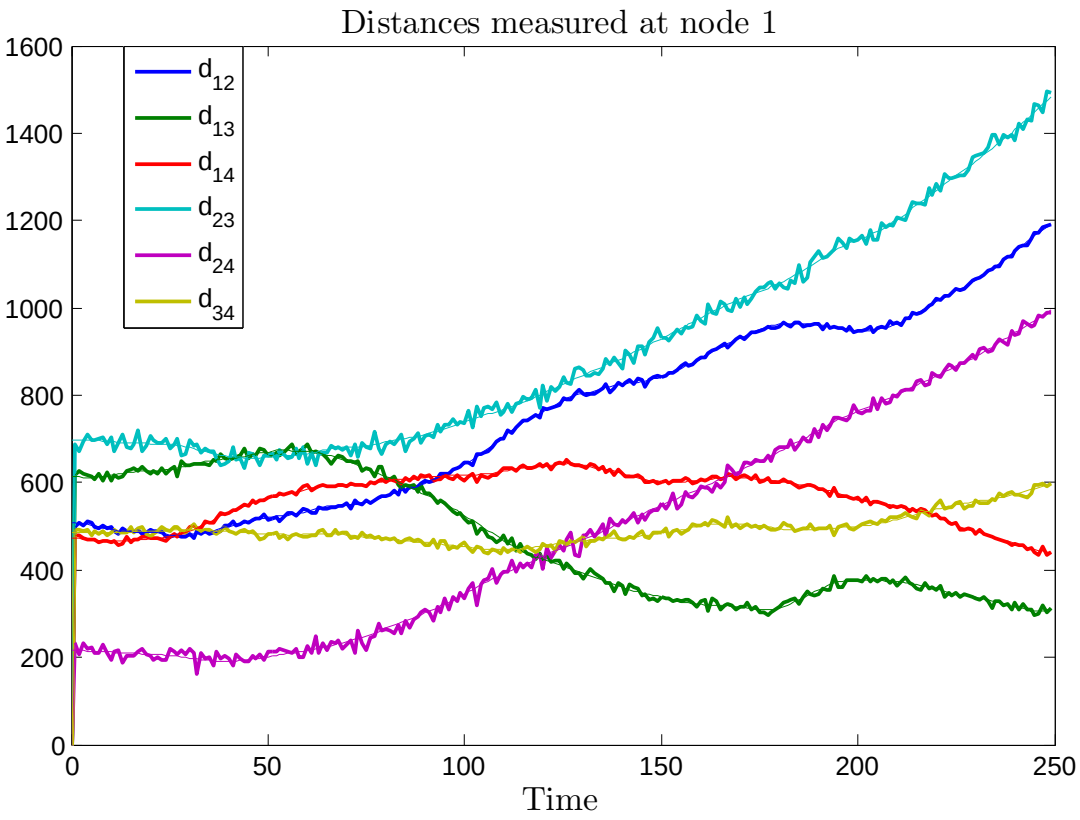

Figura 3.9: Distancias estimadas de los cuatro nodos

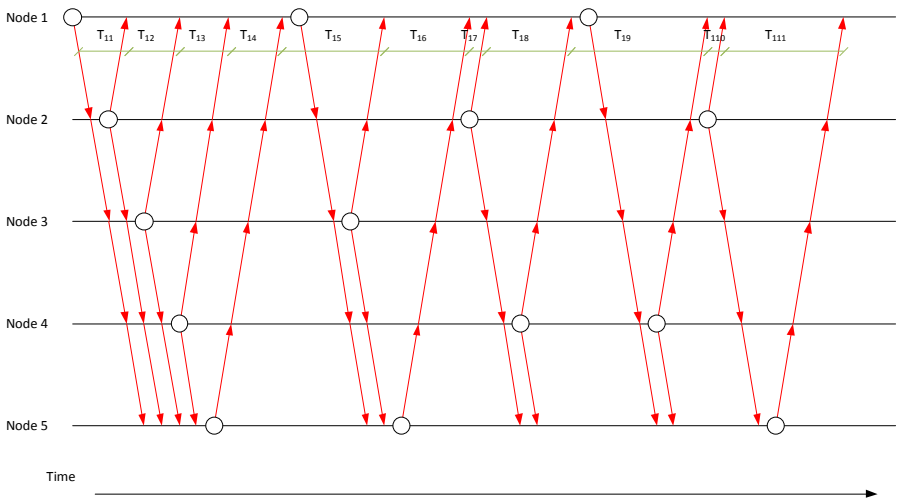

Figura 3.10: Diagrama temporal de una red de cinco nodos

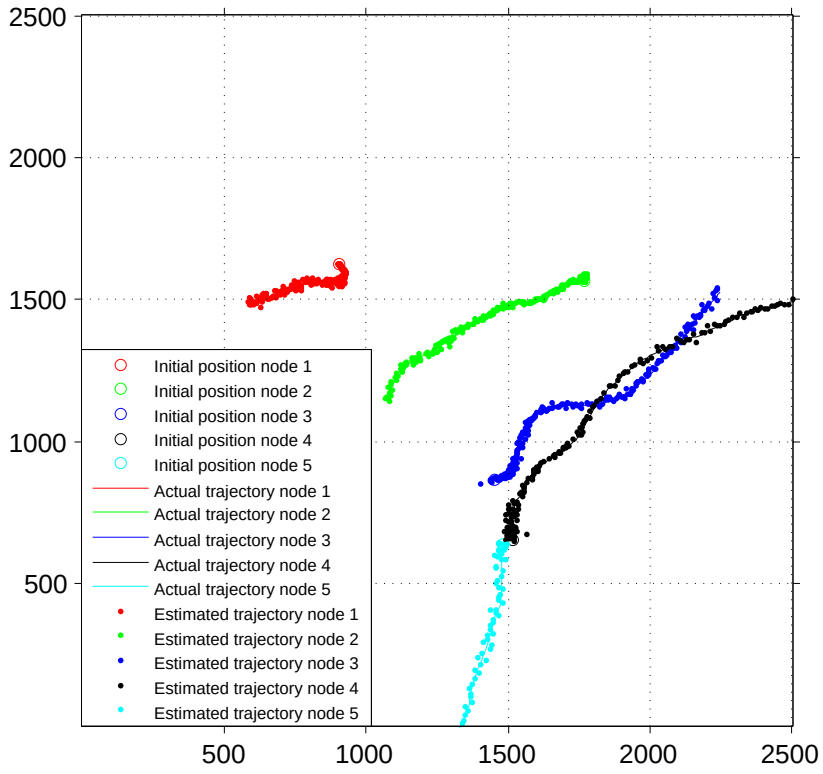

Figura 3.11: Movimiento estimado de los cuatro nodos

Los resultados obtenidos se muestran en las figuras 3.11, y 3.12.

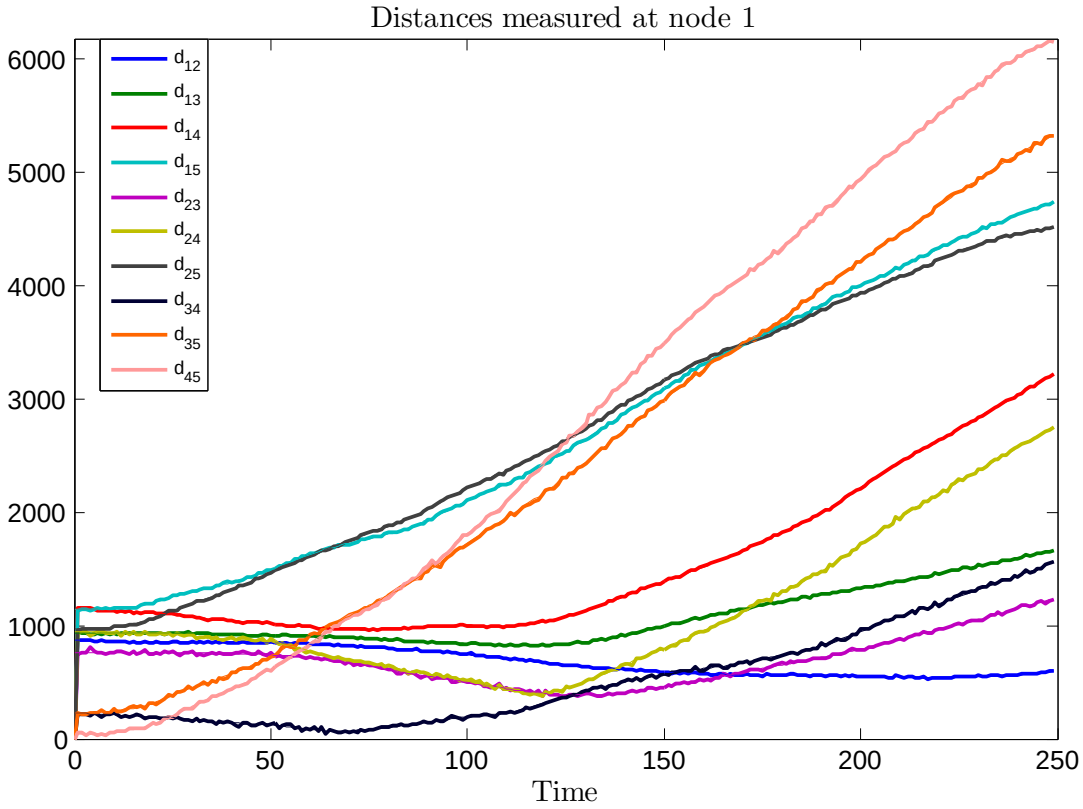

Figura 3.12: Distancias estimadas de los cinco nodos

Capítulo 4

Complejidad computacional en cada nodo

El filtro de Kalman es un algoritmo empleado en muchos sistemas de procesado de señal que se caracteriza por realizar estimaciones que minimizan el error cuadrático medio (MMSE) del mismo. El MMSE se define como la esperanza del cuadrado de la diferencia entre el valor real de una variable aleatoria desconocida y el valor estimado de la misma, la cual es generalmente una función de un varable aleatoria conocida.

$$MSE = E\left[\left|X - \hat{X}\right|^2\right]$$

Donde X es una realización de una variable aleatoria desconocida y $\hat{X}$ es el valor predicho de X.

El filtro de Kalman emplea el concepto de *estado* del sistema. El estado es el conjunto de variables que definen el sistema donde el filtro está siendo ejecutado . Contiene toda la información y, dado el estado actual y un conjunto de valores de entrada, es posible estimar estados futuros. Un aspecto importante es que, como el filtro es ejecutado de forma iterativa, tan solo es necesario almacenar el estado actual, no todos los estados previos. Este hecho implica un menor uso de memoria y una mayor idoneidad para dispositivos digitales.

4.1 Filtro

4.1.1 Establecimiento del problema

Se denota el estado de un sistema dinamico como $\mathbf{x}(n)$, el cual es un vector columna M-dimensional que contiene el conjunto de M variables que definen el sitema en su totalidad. El índice k denota el tiempo donde el estado es observado. Se define el vector de medición, o medidas, como $\mathbf{z}(n)$, el cual es un vector columna N-dimensional donde las

medidas son almacenadas. Una vez definidos estos dos vectores, el problema del filtrado puede ser expuesto. El problema se describe mediante dos ecuaciones principales, la ecuación de proceso, que computa el estado siguiente basándose en el comportamiento del sistema, y la ecuación de medida, que emplea información externa para calcular el estado siguiente. Estas ecuaciones son:

- Ecuación de proceso:

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{m}_k \tag{4.1}$$

Donde $\mathbf{F}_k$ es la matriz de transición de estados entre el instante k y $k+1$ conocida, de tamaño $M \times M$, y el vector columna de tamaño M $\mathbf{m}_k$ es el ruido de proceso que se modela como un ruido blanco Gaussiano.

- Ecuación de medida:

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{n}_k \tag{4.2}$$

Donde $\mathbf{H}_k$ es la matriz que relaciona el estado con las medidas externas, y el vector columna de tamaño N $\mathbf{n}_k$ es el ruido de de medidas, el cual es modelado como un proceso Gaussiano con varianza variable.

4.1.2 Innovaciones

Se conoce como innovación a la diferencia entre las observaciones reales, o medidas del sistema, con las predichas o estimadas. Se define el vector $\hat{\mathbf{z}}_k$ como las estimaciones de las medidas en el instante k, dadas las observaciones previas $\mathbf{z}$ desde 1 hasta $k-1$. El proceso de innovación se define mediante la ecuación 4.3.

$$\alpha_k = \mathbf{z}_k - \mathbf{z}_k^+, \ k = 1, 2, \ldots \tag{4.3}$$

El siguiente paso es determinar la matriz de correlación de este proceso de innovación. Definiendo $\mathbf{x}_k^+$ y $\mathbf{n}_k^+$ como el estado predicho y el error de medición predicho dadas las $k-1$ observaciones previas, la siguiente expresión denota la estimación MMSE del vector de observaciones.

$$\mathbf{z}_k^+ = \mathbf{H}_k \mathbf{x}_k^+ + \mathbf{n}_k^+$$

Como $\mathbf{n}_k^+$ es ortogonal a todas las observaciones pasadas, el término puede omitirse. Por lo tanto, la expresion puede reescribirse como:

$$\mathbf{z}_k^+ = \mathbf{H}_k \mathbf{x}_k^+ \tag{4.4}$$

Y 4.3 puede ser reescrita como:

$$\alpha_k = \mathbf{z}_k - \mathbf{H}_k \mathbf{x}_k^+ \tag{4.5}$$

Empleando la ecuación de medición, la expresión del proceso de innovación queda:

$$\alpha_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{n}_k - \mathbf{H}_k \mathbf{x}_k^+$$

$$\alpha_k = \mathbf{H}_k \varepsilon_{k,k-1} + \mathbf{n}_k \tag{4.6}$$

Donde $\varepsilon_{k,k-1}$ es el error de predicción de estado y es igual a:

$$\varepsilon_{k,k-1} = \mathbf{x}_k - \mathbf{x}_k^+ \tag{4.7}$$

Ahora, la matriz de correlación del proceso de innovación, por definición es:

$$\mathbf{\Sigma}_k = E[\alpha_k \alpha_k^H] \tag{4.8}$$

Introduciendo 4.6, en 4.8, se obtiene la ecuación:

$$\mathbf{\Sigma}_k = \mathbf{H}_k \mathbf{P}_{k,k-1} \mathbf{H}_k^H + \mathbf{Q}_N \tag{4.9}$$

Donde $\mathbf{P}_{k,k-1}$ es la matriz de correlación del error de predicción de estado definida como:

$$\mathbf{P}_{k,k-1} = E[\varepsilon_{k,k-1} \varepsilon_{k,k-1}^H] \tag{4.10}$$

Este error de predicción de estado se calcula adaptativamente y el proceso se expone más adelante.

4.1.3 Estimación del estado

La estimación del estado puede calcularse mediante combinaciones lineales de los procesos de innovación desde el instante $k = 1$ hasta $k = k$.

$$\mathbf{x}_k^+ = \sum_{j=1}^{k} \mathbf{B}_{ij} \alpha_k \tag{4.11}$$

Siendo $\mathbf{B}_{ij}$ un conjunto de matrices $M \times N$. El principio de ortogonalidad dice que los procesos de innovación y los errores de predicción de estado son ortogonales, por lo tanto:

$$E[\varepsilon_{ik} \alpha_m^H] = \mathbf{0} \tag{4.12}$$

Sustituyendo 4.7 y 4.11 en 4.12, puede obtenerse una expresión del conjunto de matrices $\mathbf{B}_{ik}$:

$$\begin{aligned} E[\varepsilon_{in}\alpha_m^H] &= E[(\mathbf{x}_i - \mathbf{x}_i^+\alpha_m^H] \\ &= E\left[(\mathbf{x}_i - \sum_{j=1}^{k}\mathbf{B}_{ij}\alpha_j)\alpha_m^H\right] \\ &= E[\mathbf{x}_i\alpha_m^H] - \mathbf{B}_{im}\mathbf{\Sigma}_m \end{aligned}$$

$$\mathbf{B}_{im} = E[\mathbf{x}_i\alpha_m^H]\mathbf{\Sigma}_m^{-1} \tag{4.13}$$

Una vez se tiene esta expresión para $\mathbf{B}_{im}$, la estimación del estado se obtiene siguiendo el siguiente procedimiento:

$$\begin{aligned} \mathbf{x}_i^+ &= \sum_{j=1}^{k} E[\mathbf{x}_i\alpha_j^H]\mathbf{\Sigma}_j^{-1}\alpha_j \\ &= \sum_{j=1}^{k-1} E[\mathbf{x}_i\alpha_j^H]\mathbf{\Sigma}_j^{-1}\alpha_j + E[\mathbf{x}_i\alpha_k^H]\mathbf{\Sigma}_k^{-1}\alpha_k \end{aligned}$$

Para $i = k+1$ la expresión se escribe como:

$$\mathbf{x}_{k+1}^+ = \sum_{j=1}^{k-1} E[\mathbf{x}_{k+1}\alpha_j^H]\mathbf{\Sigma}_j^{-1}\alpha_j + E[\mathbf{x}_{k+1}\alpha_k^H]\mathbf{\Sigma}_k^{-1}\alpha_k \tag{4.14}$$

Ahora, empleando la ecuación de proceso:

$$\begin{aligned} E[\mathbf{x}_{k+1}\alpha^H j] &= E\left[(\mathbf{F}_k\mathbf{x}_k + \mathbf{m}_k)\,\alpha_j^H\right] \\ E[\mathbf{x}_{k+1}\alpha_j^H] &= \mathbf{F}_k E[\mathbf{x}_k\alpha_j^H] \end{aligned} \tag{4.15}$$

Sustituyendo 4.15 en 4.14:

$$\begin{aligned} \mathbf{x}_{k+1}^+ &= \sum_{j=1}^{k-1} \mathbf{F}_k E[\mathbf{x}_k\alpha_j^H]\mathbf{\Sigma}_j^{-1}\alpha_j + E[\mathbf{x}_{k+1}\alpha_k^H]\mathbf{\Sigma}_k^{-1}\alpha_k \\ \mathbf{x}_{k+1}^+ &= \mathbf{F}_k\mathbf{x}_k^+ + E[\mathbf{x}_{k+1}\alpha_k^H]\mathbf{\Sigma}_k^{-1}\alpha_k \end{aligned} \tag{4.16}$$

De esta forma, la ecuación final de actualización de estados es:

$$\mathbf{x}_{k+1}^+ = \mathbf{F}_k\mathbf{x}_k^+ + \mathbf{K}_k\alpha_k \tag{4.17}$$

$$\mathbf{K}_k = E[\mathbf{x}_{k+1}\alpha_k^H]\mathbf{\Sigma}_k^{-1} \tag{4.18}$$

Donde $\mathbf{K}_k$ es una matriz $M \times N$ conocida como *ganancia Kalman*.

4.1.4 Ganancia Kalman

La ganancia Kalman se expresa como:

$$\mathbf{K}_k = E[\mathbf{x}_{k+1}\alpha_k^H]\mathbf{\Sigma}_k^{-1}$$

Expandiendo la esperanza:

$$\begin{aligned} E[\mathbf{x}_{k+1}\alpha_k^H] &= \mathbf{F}_k E[\mathbf{x}_k\alpha_k^H] \\ &= \mathbf{F}_k E[\mathbf{x}_k(\mathbf{H}_k\varepsilon_{k,k-1} + \mathbf{n}_k)^H] \\ &= \mathbf{F}_k E[\mathbf{x}_k\varepsilon_{k,k-1}^H]\mathbf{H}_k^H \\ &= \mathbf{F}_k E[\mathbf{x}_k\left(\mathbf{x}_k - \mathbf{x}_k^+\right)^H]\mathbf{H}_k^H \end{aligned}$$

Como $\mathbf{x}_k^+$ y $\varepsilon_{k,k-1}$ están incorreladas, la esperanza puede escribirse como:

$$E[\mathbf{x}_{k+1}\alpha_k^H] = \mathbf{F}_k E[\varepsilon_{k,k-1}\varepsilon_{k,k-1}^H]\mathbf{H}_k^H$$

$$E[\mathbf{x}_{k+1}\alpha_k^H] = \mathbf{F}_k\mathbf{P}_{k,k-1}\mathbf{H}_k^H \tag{4.19}$$

De esta forma, la expresión de la ganancia Kalman es:

$$\mathbf{K}_k = \mathbf{F}_k\mathbf{P}_{k,k-1}\mathbf{H}_k^H\mathbf{\Sigma}_k^{-1} \tag{4.20}$$

4.1.5 Matriz de correlación del error de predicción de estado

El problema ahora reside en computar la matriz de correlación del error de predicción de estado. Este cálculo se hace de forma adaptativa. El proceso parte de las siguientes ecuaciones:

$$\begin{aligned} \varepsilon_{k,k-1} &= \mathbf{x}_k - \mathbf{x}_k^+ \\ \varepsilon_{k+1,k} &= \mathbf{x}_{k+1} - \mathbf{x}_{k+1}^+ \\ \mathbf{x}_{k+1} &= \mathbf{F}_k\mathbf{x}_k + \mathbf{m}_K \\ \mathbf{x}_{k+1}^+ &= \mathbf{F}_k\mathbf{x}_k^+ + \mathbf{K}_k\alpha_k \end{aligned}$$

Ahora, expandiendo $\varepsilon_{k+1,k}$:

$$\begin{aligned}\varepsilon_{k+1,k} &= \mathbf{F}_k\mathbf{x}_k + \mathbf{m}_k - \mathbf{F}_k\mathbf{x}_k^+ \\ &+ \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k^+) \\ &= \mathbf{F}_k\mathbf{x}_k + \mathbf{m}_k - \mathbf{F}_k\mathbf{x}_k^+) \\ &+ \mathbf{K}_k(\mathbf{H}_k\mathbf{x}_k - \mathbf{H}_k\mathbf{x}_k^+))\end{aligned}$$

$$\varepsilon_{k+1,k} = (\mathbf{F}_k - \mathbf{K}_k\mathbf{H}_k)\,\varepsilon_{k,k-1} - \mathbf{K}_k\mathbf{n}_k + \mathbf{m}_k \tag{4.21}$$

Y calculando la matriz del error de predicción de estado mediante la definición:

$$\begin{aligned}\mathbf{P}_{k+1,k} &= E[\varepsilon_{k+1,k}\varepsilon_{k+1,k}^H] \\ &= (\mathbf{F}_k - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k,k-1}(\mathbf{F}_k^H - \mathbf{K}_k^H\mathbf{H}_k^H) \\ &+ \mathbf{K}_k\mathbf{Q}_N\mathbf{K}_k^H + \mathbf{Q}_M\end{aligned}$$

Finalmente la ecuación que permite calcular esta matriz de forma adaptativa es:

$$\mathbf{P}_{k+1,k} = \mathbf{F}_k\mathbf{P}_k\mathbf{F}_k^H + \mathbf{Q}_M \tag{4.22}$$

$$\mathbf{P}_k = \mathbf{P}_{k,k-1} - \mathbf{F}_k\mathbf{K}_k\mathbf{H}_k\mathbf{P}_{k,k-1} \tag{4.23}$$

4.1.6 Condiciones iniciales

Para obtener las condiciones iniciales hay que observar el caso $k = 0$. La expresión es:

$$\mathbf{x}_1^+ = \mathbf{F}_0\mathbf{x}_0^+$$

Como $\mathbf{x}_0^+$ es la estimación sin ningún dato previo, es igual a cero, por eso la primera condición inicial es:

$$\mathbf{x}_0 = \mathbf{0} \tag{4.24}$$

La otra variable que ha de ser inicializada es la matriz de correlación del error de predicción de estados. Siguiendo el procedimiento anterior se deduce que:

$$\begin{aligned}\mathbf{P}_{1,0} &= E[\varepsilon_{1,0}\varepsilon_{1,0}^H)] \\ &= E[\left(\mathbf{x}_1 - \mathbf{x}_0^+\right)\left(\mathbf{x}_1 - \mathbf{x}_0^+\right)^H] \\ &= E[\mathbf{x}_1\mathbf{x}_1^H]\end{aligned}$$

$$\mathbf{P}_{1,0} = E[\mathbf{x}_1 \mathbf{x}_1^H] \tag{4.25}$$

4.1.7 Resumen

La tabla 4.1 guarda todas las medidas, ecuaciones, parámetros y condiciones iniciales necesarias para ejecutar el filtro.

Kalman filter
Inputs:
$\mathbf{z}_k$
Known parameters:
State transition matrix: $\mathbf{F}_k$
Measurement matrix: $\mathbf{H}_k$
Correlation matrix of process noise vector: $\mathbf{Q}_M$
Correlation matrix of measurement noise vector: $\mathbf{Q}_N$
Initial conditions:
$\mathbf{x}_0 = \mathbf{0}$
$\mathbf{P}_{1,0} = E[\mathbf{x}_1 \mathbf{x}_1^H]$
Computation loop:
Time update:
$\mathbf{x}_k^+ = \mathbf{F}_k \mathbf{x}_k$
$\mathbf{P}_k^+ = \mathbf{F}_k \mathbf{P}_k \mathbf{F}_k^H + \mathbf{Q}_M$
Measurement update:
$\mathbf{K}_k = \mathbf{F}_k \mathbf{P}_k^+ \mathbf{H}_k^H \left[\mathbf{H}_k \mathbf{P}_k^+ \mathbf{H}_k^+ + \mathbf{Q}_N\right]^{-1}$
$\alpha_k = \mathbf{z}_k - \mathbf{H}_k \mathbf{x}_k$
$\mathbf{x}_{k+1} = \mathbf{x}_k^+ + \mathbf{K}_k \alpha_k$
$\mathbf{P}_{k+1} = [\mathbf{I} - \mathbf{K}_k \mathbf{H}_k] \mathbf{P}_k^+ + \mathbf{Q}_M$

Tabla 4.1: El filtro de Kalman

4.2 Complejidad del filtro de Kalman

Una vez introducido el filtro de Kalman, es conveniente analizar su complejidad computacional calculando el número de multiplicaciones y sumas. Como es un filtro adaptativo, no tiene un número fijo de multiplicaciones y sumas, ya que el bucle se está ejecutando continuamente. Para medir la complejidad se tendrá en cuenta una iteración del filtro. Primero, dos variables deben ser definidas. M denota el tamaño del vector de estados, y N el tamaño del vector de medidas. M se relaciona con el número de nodos mediante la expresión:

$$M = \frac{N_{nodes}!}{2!(N_{nodes} - 2)!} \tag{4.26}$$

La metodología es la misma que la empleada en el anexo **??**. Tomar una ecuación del bucle y analizarla paso por paso. Las tablas desde 4.2 hasta 4.8 almacenan los resultados.

Variable	Size
$\mathbf{z}_k$	$(N \times 1)$
$\mathbf{x}_k$	$(M \times 1)$
$\mathbf{F}_k$	$(M \times M)$
$\mathbf{H}_k$	$(N \times M)$
$\mathbf{Q}_M$	$(M \times M)$
$\mathbf{Q}_N$	$(N \times N)$
$\mathbf{K}_k$	$(M \times N)$
α_k	$(N \times 1)$
$\mathbf{x}_k^+$	$(M \times 1)$
$\mathbf{P}_k$	$(M \times M)$
$\mathbf{P}_k^+$	$(M \times M)$

Tabla 4.2: Tamaño de las variables del filtro de Kalman

$\mathbf{x}_k^+ = \mathbf{F}_k\mathbf{x}_k$			
Step	**Sizes**	**Products**	**Additions**
$\mathbf{x}_k^+ = \mathbf{F}_k\mathbf{x}_k$	$(M \times M)(M \times 1)$	M^2	$M(M-1)$
Total	–	$\mathbf{M^2}$	$\mathbf{M^2 - M}$

Tabla 4.3: Complejidad de la estimación del estado

$\mathbf{P}_k^+ = \mathbf{F}_k\mathbf{P}_k\mathbf{F}_k^H + \mathbf{Q}_M$			
Step	**Sizes**	**Products**	**Additions**
$\mathbf{F}_k\mathbf{P}_k$	$(M \times M)(M \times M)$	M^3	$M^2(M-1)$
$[\mathbf{F}_k\mathbf{P}_k]\mathbf{F}_k^H$	$(M \times M)(M \times M)$	M^3	$M^2(M-1)$
$[\cdots] + \mathbf{Q}_M$	$(M \times M)(M \times M)$	–	M^2
Total	–	$\mathbf{2M^3}$	$\mathbf{2M^3 - M}$

Tabla 4.4: Complejidad de la matriz de correlación del error de predicción

$\mathbf{K}_k = \mathbf{F}_k\mathbf{P}_k^+\mathbf{H}_k^H[\mathbf{H}_k\mathbf{P}_k^+\mathbf{H}_k^H + \mathbf{Q}_N]^{-1}$			
Step	**Sizes**	**Products**	**Additions**
$\mathbf{H}_k\mathbf{P}_k^+$	$(N \times M)(M \times M)$	NM^2	$NM(M-1)$
$[\mathbf{H}_k\mathbf{P}_k^+]\mathbf{H}_k^H$	$(N \times M)(M \times N)$	N^2M	$N^2(M-1)$
$[\cdots] + \mathbf{Q}_N$	$(N \times N)(N \times N)$	–	N^2
$[\cdots]^{-1}$	$(N \times N)$	$mults_{inv}$	$adds_{inv}$
$\mathbf{F}_k\mathbf{P}_k^+$	$(M \times M)(M \times M)$	M^3	$M^2(M-1)$
$[\mathbf{F}_k\mathbf{P}_k^+]\mathbf{H}_k^H$	$(M \times M)(M \times M)$	M^2N	$MN(N-1)$
$[\cdots][\cdots]^{-1}$	$(M \times N)(N \times N)$	MN^2	$MN(N-1)$
Total	–	$\mathbf{M^3 + 2M^2N}$ $\mathbf{+2MN^2}$ $\mathbf{+mults}_{inv}$	$\mathbf{M^3 + 2M^2 + 2MN^2}$ $\mathbf{-M^2 - 3MN}$ $\mathbf{+adds}_{inv}$

Tabla 4.5: Complejidad de la ganancia Kalman

$\alpha_k = \mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k^+ - \mathbf{d}_k$			
Step	**Sizes**	**Products**	**Additions**
$\mathbf{H}_k\mathbf{x}_k^+$	$(N \times M)(M \times 1)$	NM	$N(M-1)$
$\mathbf{z}_k - [\mathbf{H}_k\mathbf{x}_k^+]$	$(N \times 1) - (N \times 1)$	–	N
$\alpha_k = [\mathbf{z}_k - \mathbf{H}_k\mathbf{x}_k^+] - \mathbf{d}_k$	$(N \times 1) - (N \times 1)$	–	N
Total	–	**NM**	**NM + M**

Tabla 4.6: Complejidad de las innovaciones

La ecuación de las innovaciones ha sido adaptada a este problema de localización cooperativa, restando el retardo de procesado d.

$\mathbf{x}_{k+1} = \mathbf{x}_k^+ + \mathbf{K}_k\alpha_k$			
Step	**Sizes**	**Products**	**Additions**
$\mathbf{K}_k\alpha_k$	$(M \times N)(N \times 1)$	MN	$M(N-1)$
$\mathbf{x}_k^+ + \mathbf{K}_k\alpha_k$	$(M \times 1) + (M \times 1)$	–	M
Total	–	**MN**	**MN**

Tabla 4.7: Complejidad de la estimación del estado actualizado

$\mathbf{P}_{k+1} = [\mathbf{I} - \mathbf{K}_k\mathbf{H_k}]\mathbf{P}_k^+ + \mathbf{Q}_M$			
Step	**Sizes**	**Products**	**Additions**
$\mathbf{K}_k\mathbf{H_k}$	$(M \times N)(N \times M)$	M^2N	$M^2(N-1)$
$\mathbf{I} - \mathbf{K}_k\mathbf{H_k}$	$(M \times M) + (M \times M)$	–	M^2
$[\mathbf{I} - \mathbf{K}_k\mathbf{H_k}]\mathbf{P}_k^+$	$(M \times M)(M \times M)$	M^3	$M^2(M-1)$
$[\mathbf{I} - \mathbf{K}_k\mathbf{H_k}]\mathbf{P}_k^+ + \mathbf{Q}_M$	$(M \times M) + (M \times M)$	–	M^2
Total	–	$\mathbf{M^3 + M^2N}$	$\mathbf{M^3 + M^2N}$

Tabla 4.8: Complejidad de la matriz de correlación del error de predicción actualizada

Teniendo todos estos resultados, se puede calcular la complejidad global de una iteración del filtro de Kalman sumando las contribuciones de cada ecuación. Un aspecto importante a tener en cuenta es que la mayor contribución al coste computacional viene dado por la ganancia Kalman y la matriz de correlación del error de predicción de estado. Más importante es implementar un algoritmo eficiente de inversión de matrices dado que la matriz a invertir el muy grande, y la inversión de matrices es conocida por ser muy costosa computacionalmente. La tabla 4.9 almacena el resultado total de una iteración.

4.3 Inversión de la matriz de la ganancia Kalman

Calcular la ganancia Kalman requiere una inversión de una matriz de tamaño $N \times N$ por iteración. La inversión de matrices es una operación con un alto coste computacional y difícil de implementar en hardware. El número de operaciones necesarias crece exponencialemnte con el tamaño de la matriz. Hablando sobre costes computacionales, la inversión de matrices suele ser siempre el punto crítico del sistema.

Complexity of 1 iteration of the Kalman filter	
Products	**Additions**
$4M^3 + M^2(1+3N)$ $+M(2N^2+2N)$ $+mults_{inv}$	$4M^3 + M^2(N+2)$ $+M(2N^2-N-1)$ $+adds_{inv}$

Tabla 4.9: Complejidad de una iteración del filtro de Kalman

Resolver una inversión empleando la fuerza bruta es inviable. La mejor solución es descomponer la matriz de tal forma que las matrices obtenidas tras la descomposición sean de fácil inversión, como por ejemplo, matrices triangulares.

4.3.1 Sustitución hacia adelante y hacia atrás

La sustitución hacia adelante y hacia atrás son dos algoritmos usados para invertir matrices triangulares. Ambos son equivalentes y su coste computacional es el mismo. En este caso, se analiza la sustitución hacia atrás.

$$\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1N} \\ 0 & u_{22} & u_{23} & \cdots & u_{2N} \\ 0 & 0 & u_{33} & \cdots & u_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{NN} \end{bmatrix}$$

Se busca una matriz $\mathbf{W}$:

$$\mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & \cdots & w_{1N} \\ 0 & w_{22} & w_{23} & \cdots & w_{2N} \\ 0 & 0 & w_{33} & \cdots & w_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & w_{NN} \end{bmatrix}$$

para resolver esta ecuación:

$$\mathbf{UW} = \mathbf{I} \tag{4.27}$$

Todos estos coeficientes pueden obtenerse de forma iterativa mediante las siguientes ecuaciones:

$$w_{ij} = \frac{1}{u_{ij}} \qquad i = j \tag{4.28}$$

$$w_{ij} = \frac{-1}{u_{ii}} \left(\sum_{k=i+1}^{j} u_{ik} w_{kj} \right) \qquad i \neq j \tag{4.29}$$

4.3.2 Complejidad de la sustitución hacia atrás

Como puede verse en las ecuaciones anteriores, la cantidad de operaciones depende directamente de la diferencia entre los dos subíndices i y j. A mayor diferencia, mayor es la ecuación. Por ejemplo, mirando la primera columna:

$$\begin{aligned}
w_{11} &= \frac{1}{u_{11}} \\
w_{12} &= \frac{-1}{u_{11}} (u_{12} w_{22}) \\
w_{13} &= \frac{-1}{u_{11}} (u_{12} w_{23} + u_{13} w_{33}) \\
w_{14} &= \frac{-1}{u_{11}} (u_{12} w_{24} + u_{13} w_{34} + u_{14} w_{44}) \\
&\cdots \\
w_{1N} &= \frac{-1}{u_{11}} (u_{12} w_{2N} + u_{13} w_{3N} + \cdots + u_{1N} w_{NN})
\end{aligned}$$

Si se almacena todo esto en una tabla, tabla 4.10, para cada caso, el resultado puede verse de una forma más clara.

Index difference	$\times$	$+$	$/$	$\surd$	**Number of elements**
0	0	0	1	0	N
1	1	1	1	0	N-1
2	2	2	1	0	N-2
3	3	3	1	0	N-3
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
N	N	N	1	0	1

Tabla 4.10: Complejidad de obtener cada elemento de la inversa dependiendo de la diferencia entre subíndices

Donde la última columna de la tabla indica el número de elementos a ser calculados con la diferencia de índices. De la tabla 4.10 pueden deducirse expresiones cerradas para calcular las multiplicaciones, sumas y divisiones. Para sumas y multiplicaciones la expresión es la misma:

$$\begin{aligned}
N_{mults} = N_{adds} &= 0N + 1(N-1) + 2(N-2) + 3(N-3) + \cdots + (N-1)\,(N-(N-1)) \\
&= \sum_{k=0}^{N-1} k(N-k)
\end{aligned}$$

$$= \frac{1}{6}N^3 - \frac{1}{6}N$$

$$N_{mults} = N_{adds} = \frac{1}{6}N^3 - \frac{1}{6}N \tag{4.30}$$

La expresión para las divisiones es:

$$\begin{aligned} N_{divs} = &= 1N + 1(N-1) + 1(N-2) + 1(N-3) + \cdots + 1\,(N-(N-1)) \\ &= \sum_{k=0}^{N-1}(N-k) \\ &= \frac{1}{2}N^2 - \frac{1}{2}N \end{aligned}$$

En este punto, el coste computacional de este método es conocido, y se muestra en la tabla 4.11.

Backward substitution	Number of operations
Multiplications	$\frac{1}{6}N^3 - \frac{1}{6}N$
Additions	$\frac{1}{6}N^3 - \frac{1}{6}N$
Divisions	$\frac{1}{2}N^2 + \frac{1}{2}N$

Tabla 4.11: Complejidad de la sustitución hacia atrás

4.3.3 Descomposición e inversión

Se han tenido en cuenta dos tipos de descomposición de matrices para llevar a cabo la inversión de la matriz de la ganancia Kalman.

4.3.3.1 Inversión mediante la descomposición QR

El primer método de descomposición es la descomposición QR. El anexo **??** detalla este método. El procedimiento para invertir la matriz es el siguiente.

$$\begin{aligned} \mathbf{A} &= \mathbf{QR} \\ \mathbf{A}^{-1} &= [\mathbf{QR}]^{-1} = \mathbf{R}^{-1}\mathbf{Q}^{-1} = \mathbf{R}^{-1}\mathbf{Q}^T \end{aligned}$$

Se reduce la complejidad dado que $\mathbf{Q}^{-1} = \mathbf{Q}^T$. Trasponer una matriz no requiere operación alguna. La única matriz a invertir es $\mathbf{R}$. Como es una matriz $N \times N$, el coste total de la inversión puede calcualrse sumando los costes de la sustitución hacia atrás, el coste del algoritmo empleado para descomponer la matriz, y el coste de una multiplicación de matrices. Las tablas 4.12, 4.13 y 4.14 almacenan todos los resultados.

Inversion with Gram Schmidt	Number of operations	Order
Multiplications	$\frac{13}{6}N^3 - \frac{1}{6}N$	$\mathcal{O}\left(\frac{13}{6}N^3\right)$
Additions	$\frac{13}{6}N^3 - \frac{3}{2}N^2 - \frac{2}{3}N$	$\mathcal{O}\left(\frac{13}{6}N^3\right)$
Divisions	$\frac{3}{2}N^2 + \frac{1}{2}N$	$\mathcal{O}\left(\frac{3}{2}N^2\right)$
Square roots	N	$\mathcal{O}(N)$

Tabla 4.12: Complejidad de obtener la inversa mediante el uso de la ortogonalización de Gram-Schmidt

Inversion with Householder	Number of operations	Order
Multiplications	$\frac{5}{2}N^3 + \frac{5}{2}N - 8$	$\mathcal{O}\left(\frac{5}{2}N^3\right)$
Additions	$\frac{5}{2}N^3 + N^2 - \frac{23}{6}N - 3$	$\mathcal{O}\left(\frac{5}{2}N^3\right)$
Divisions	$N^2 + N - 1$	$\mathcal{O}\left(N^2\right)$
Square roots	$2N - 2$	$\mathcal{O}(2N)$

Tabla 4.13: Complejidad de obtener la inversa mediante el uso de la transformación de Householder

4.3.3.2 Inversión con la descomposición de Cholesky

El segundo método de descomposición propuesto es la descomposición de Cholesky. Este algoritmo descompone matrices simétricas en dos matrices triangulares, una inferior y su traspuesta, triangular superior.

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

Como la matriz a invertir en la ganancia Kalman es simétrica, este algoritmo es válido. Los coeficientes l_{ij} pueden obtenerse iterativamente empleando las siguientes expresiones:

$$l_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2} \tag{4.31}$$

$$l_{ij} = \frac{1}{l_{jj}}\left(a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk}\right) \qquad i \geq j \tag{4.32}$$

Como puede verse, aparece una raíz cuadrada. Este hecho puede llevar a cabo

Inversion with Givens rotations	Number of operations	Order
Multiplications	$\frac{1}{2}N^5 + \frac{2}{3}N^3 - \frac{7}{6}N$	$\mathcal{O}\left(\frac{1}{2}N^5\right)$
Additions	$\frac{1}{2}N^5 - N^4 + \frac{5}{3}N^3 - \frac{1}{2}N^2 - \frac{1}{6}N$	$\mathcal{O}\left(\frac{1}{2}N^5\right)$
Divisions	$\frac{3}{2}N^2 - \frac{1}{2}N$	$\mathcal{O}\left(\frac{3}{2}N^2\right)$
Square roots	$\frac{1}{2}N^2 - \frac{1}{2}N$	$\mathcal{O}\left(\frac{1}{2}N^2\right)$

Tabla 4.14: Complejidad de obtener la inversa mediante el uso de las rotaciones de Givens

la aparición de números complejos si $a_{jj} \leq \sum_{k=1}^{j-1} l_{jk}^2$. Modificando el algoritmo adecuadamente, la raíz cuadrada puede eliminarse.

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T \tag{4.33}$$

Donde $\mathbf{L}$ es una matriz triangular inferior cuya diagonal está compuesta por 1s, y $\mathbf{D}$ es una matriz diagonal con elementos positivos. Empleando este algoritmo, la forma de obtener la inversa es:

$$\mathbf{A}^{-1} = \mathbf{L}^{-T}\mathbf{D}^{-1}\mathbf{L}^{-1} \tag{4.34}$$

Donde el superíndice $-T$ denota la inversa de la traspuesta. Todos los coeficientes de $\mathbf{L}$ y $\mathbf{D}$ pueden calcularse como:

$$d_{jj} = a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 d_{kk} \tag{4.35}$$

$$l_{ij} = \frac{1}{d_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} d_{kk} \right) \qquad i \geq j \tag{4.36}$$

4.3.3.3 Complejidad de la descomposición de Cholesky

Primero se analiza la diagonal.

Element	$\times$	$+$
$d_{11} = a_{11}$	-	-
$d_{22} = a_{22} - l_{21}^2 d_{11}$	2	1
$d_{33} = a_{33} - l_{31}^2 d_{11} - l_{32}^2 d_{22}$	4	2
$\cdots$	$\cdots$	$\cdots$
$d_{kk} = a_{kk} - l_{k1}^2 d_{11} - \cdots - l_{k,k-1}^2 d_{k-1,k-1}$	2(k-1)	k-1

Tabla 4.15: Complejidad de obtener los elementos de $\mathbf{D}$

Juntando todo en expresiones cerradas:

$$N_{mults} = \sum_{k=1}^{N} 2(k-1) \tag{4.37}$$

$$N_{adds} = \sum_{k=1}^{N} (k-1) \tag{4.38}$$

Mirando ahora al resto de elementos, el número de operaciones aumenta con el índice de fila (ver tabla 4.17):

Obtaining diagonal elements of D	Number of operations
Multiplications	$N^2 - N$
Additions	$\frac{1}{2}N^2 - \frac{1}{2}N$
Divisions	0
Square roots	0

Tabla 4.16: Complejidad de obtener los elementos de **D**

Element	$\times$	$+$	$/$	$\surd$	**Times**
$l_{12} = \frac{1}{d_{11}}a_{12}$	0	0	1	0	N-1
$l_{23} = \frac{1}{d_{22}}(a_{23} - l_{12}l_{13}d_{11})$	2	1	1	0	N-2
$l_{34} = \frac{1}{d_{33}}(a_{34} - l_{14}l_{13}d_{11} - l_{24}l_{23}d_{22})$	4	2	1	0	N-3
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$l_{k,x}$	2(k-1)	k-1	1	0	N-k

Tabla 4.17: Complejidad de obtener los elementos de **L**

Juntando todo en expresiones cerradas:

$$\begin{aligned} N_{mults} &= \sum_{k=1}^{N-1} 2(k-1)(N-k) \\ &= \frac{1}{3}N^3 + N^2 + \frac{2}{3}N \\ N_{adds} &= \sum_{k=1}^{N-1} (k-1)(N-k) \\ &= \frac{1}{6}N^3 + \frac{1}{2}N^2 + \frac{1}{3}N \\ N_{divs} &= \sum_{k=1}^{N-1} (N-k) \\ &= \frac{1}{2}N^2 - \frac{1}{2}N \end{aligned}$$

Obtaining elements of L	Number of operations
Multiplications	$\frac{1}{3}N^3 + N^2 + \frac{2}{3}N$
Additions	$\frac{1}{6}N^3 + \frac{1}{2}N^2 + \frac{1}{3}N$
Divisions	$\frac{1}{2}N^2 - \frac{1}{2}N$
Square roots	0

Tabla 4.18: Complejidad de obtener los elementos de **L**

La tabla 4.19 contiene la complejidad de todo el algoritmo, sumando ambos procesos:

Finalmente, para obtener el coste total de la inversión empleando este método, ha de sumarse la complejidad de la descomposición, dos multiplicaciones de matrices, una sustitución hacia atrás, una sustitución hacia adelante y N divisiones escalares. La tabla 4.20 muestra todos los resultados.

Complexity of Cholesky	Number of operations
Multiplications	$\frac{1}{3}N^3 + 2N^2 + \frac{5}{3}N$
Additions	$\frac{1}{6}N^3 + N^2 - \frac{1}{6}N$
Divisions	$\frac{1}{2}N^2 - \frac{1}{2}N$
Square roots	0

Tabla 4.19: Complejdad de la descomposición de Cholesky

Complexity of inversion with Cholesky	Number of operations	Order
Multiplications	$\frac{8}{3}N^3 + 2N^2 + \frac{4}{3}N$	$\mathcal{O}\left(\frac{8}{2}N^3\right)$
Additions	$\frac{5}{2}N^3 - N^2 - \frac{1}{2}N$	$\mathcal{O}\left(\frac{5}{2}N^3\right)$
Divisions	$\frac{3}{2}N^2 + \frac{3}{2}N$	$\mathcal{O}\left(\frac{3}{2}N^2\right)$
Square roots	0	$\mathcal{O}(0)$

Tabla 4.20: Complejidad de la inversión usando la descomposición de Cholesky

4.4 Coste computacional total del filtro

La tabla 4.21 congrega la complejidad de todo el filtro. En general, el tamaño del vector de estados M es mucho mayor que el número de medidas N. Al calcular el orden de complejidad se ha tenido en cuenta este hecho.

En el caso $M = N$, el orden de complejidad cambia dado que N toma más importancia. La tabla 4.22 guarda los resultados.

Total complexity of Kalman filter, $(M > N)$		
KF + Gram Schmidt	Number of operations	Order
Multiplications	$\frac{13}{6}N^3 + 2MN^2 + \left(3M^2 + 2M - \frac{1}{6}\right)N + 4M^3 + M^2$	$\mathcal{O}\left(4M^3\right)$
Additions	$\frac{13}{6}N^3 + \left(2M - \frac{3}{2}\right)N^2 + \left(M^2 - M - \frac{2}{3}\right)N + 4M^3 + 2M^2 - M$	$\mathcal{O}\left(4M^3\right)$
Divisions	$\frac{3}{2}N^2 + \frac{1}{2}N$	$\mathcal{O}\left(2N^2\right)$
Square roots	N	$\mathcal{O}(N)$
KF + Householder	Number of operations	Order
Multiplications	$\frac{5}{2}N^3 + 2MN^2 + \left(3M^2 + 2M + \frac{5}{2}\right)N + 4M^3 + M^2 - 8$	$\mathcal{O}\left(4M^3\right)$
Additions	$\frac{5}{2}N^3 + (2M+1)N^2 + \left(M^2 - M - \frac{23}{6}\right)N + 4M^3 + 2M^2 - M - 3$	$\mathcal{O}\left(4M^3\right)$
Divisions	$N^2 + N - 1$	$\mathcal{O}\left(N^2\right)$
Square roots	$2N - 2$	$\mathcal{O}(2N)$
KF + G.Rotations	Number of operations	Order
Multiplications	$\frac{1}{2}N^5 + \frac{2}{3}N^3 + 2MN^2 + \left(3M^2 + 2M - \frac{7}{6}\right)N + 4M^3 + M^2$	$\mathcal{O}\left(\frac{1}{2}N^5\right)$
Additions	$\frac{1}{2}N^5 - N^4 + \frac{5}{3}N^3 + \left(2M - \frac{1}{2}\right)N^2 +$ $+ \left(M^2 - M - \frac{1}{6}\right)N + 4M^3 + 2M^2 - M$	$\mathcal{O}\left(\frac{1}{2}N^5\right)$
Divisions	$\frac{3}{2}N^2 - \frac{1}{2}N$	$\mathcal{O}\left(\frac{3}{2}N^2\right)$
Square roots	$\frac{1}{2}N^2 - \frac{1}{2}N$	$\mathcal{O}\left(\frac{1}{2}N^2\right)$
KF + Cholesky	Number of operations	Order
Multiplications	$\frac{8}{3}N^3 + (2M+2)N^2 + \left(3M^2 + 2M + \frac{4}{3}\right)N + 4M^3 + M^2$	$\mathcal{O}\left(M^3\right)$
Additions	$\frac{5}{2}N^3 + (2M-1)N^2 + \left(M^2 - M - \frac{1}{2}\right)N + 4M^3 + 2M^2 - M$	$\mathcal{O}\left(4M^3\right)$
Divisions	$\frac{3}{2}N^2 + \frac{3}{2}N$	$\mathcal{O}\left(\frac{3}{2}N^2\right)$
Square roots	0	$\mathcal{O}(0)$

Tabla 4.21: Coste computacional total del filtro de Kalman

Order of complexity of Kalman filter, $(M = N)$	
KF + Gram Schmidt	Order
Multiplications	$\mathcal{O}\left(\frac{67}{6}M^3\right)$
Additions	$\mathcal{O}\left(\frac{55}{6}M^3\right)$
Divisions	$\mathcal{O}\left(\frac{3}{2}M^2\right)$
Square roots	$\mathcal{O}(M)$
KF + Householder	Order
Multiplications	$\mathcal{O}\left(\frac{23}{2}M^3\right)$
Additions	$\mathcal{O}\left(\frac{19}{2}M^3\right)$
Divisions	$\mathcal{O}\left(M^2\right)$
Square roots	$\mathcal{O}(2M)$
KF + G.Rotations	Order
Multiplications	$\mathcal{O}\left(\frac{1}{2}M^5\right)$
Additions	$\mathcal{O}\left(\frac{1}{2}M^5\right)$
Divisions	$\mathcal{O}\left(\frac{3}{2}M^2\right)$
Square roots	$\mathcal{O}\left(\frac{1}{2}M^2\right)$
KF + Cholesky	Order
Multiplications	$\mathcal{O}\left(\frac{35}{3}M^3\right)$
Additions	$\mathcal{O}\left(\frac{35}{3}M^3\right)$
Divisions	$\mathcal{O}\left(\frac{3}{2}M^2\right)$
Square roots	$\mathcal{O}(0)$

Tabla 4.22: Orden de complejidad cuando $M = N$

Orders	KF + Gram Schmidt	KF + Householder	KF + Cholesky
Multiplications	$\mathcal{O}$ (11.16 N^3)	$\mathcal{O}$ (11.5 N^3)	$\mathcal{O}$ (11.66 N^3)
Additions	$\mathcal{O}$ (9.16 N^3)	$\mathcal{O}$ (9.5 N^3)	$\mathcal{O}$ (11.66 N^3)
Divisions	$\mathcal{O}$ (1.5 N^2)	$\mathcal{O}$ (N^2)	$\mathcal{O}$ (1.5 N^3)
Square roots	$\mathcal{O}$ (N)	$\mathcal{O}$ (2 N)	$\mathcal{O}$ (0.5 N)
FLOPs	$\mathcal{O}$ (20.33 N^3)	$\mathcal{O}$ (21 N^3)	$\mathcal{O}$ (23.33 N^3)

Tabla 4.23: Orden de operaciones con el número de medidas

Method	Order
KF + Gram-Schmidt	$\mathcal{O}$ (2.54 n^6)
KF + Householder	$\mathcal{O}$ (2.63 n^6)
KF + Cholesky	$\mathcal{O}$ (2.92 n^6)

Tabla 4.24: Orden de operaciones con el número de nodos

4.5 Complejidad del sistema con el aumento de nodos en la red

Finalmente, una vez que la complejidad del filtro es conocida, es hora de discutir cómo afecta el número de dispositivos de la red a la misma. Lo primero, se van a considerar las operaciones como operaciones de punto flotante (flops). La tabla 4.23 muestra el número de flops requerido para cada caso en términos del número de medidas tomadas por cada nodo.

En la tabla 4.23 no se han tenido en cuenta las rotaciones de Givens. El número de mediciones es igual al número de enlaces directos entre nodos, luego se relacionan mediante la ecuación 2.5.

$$N = c_2^n = \frac{n(n-1)}{2}$$

La tabla 4.24 muestra el orden de flops en términos del número de nodos n, empleando la ecuación 2.5 en las expresiones en la tabla 4.23.

Puede verse que el coste computacional aumenta con un orden de 6 con el número de nodos, esto significa que añadir un nuevo nodo a una red pequeña tiene menos impacto que añadirlo a una red grande. Esta complejidad creciente tiene un efecto en la tasa de actualización. Las expresiones de la tabla 4.24 son útiles para calcular el máximo tamaño de una red en concreto. Dependiendo de la tasa de actualización de la posición deseada, y de la velocidad de procesado de los nodos, este número puede ser calculado.

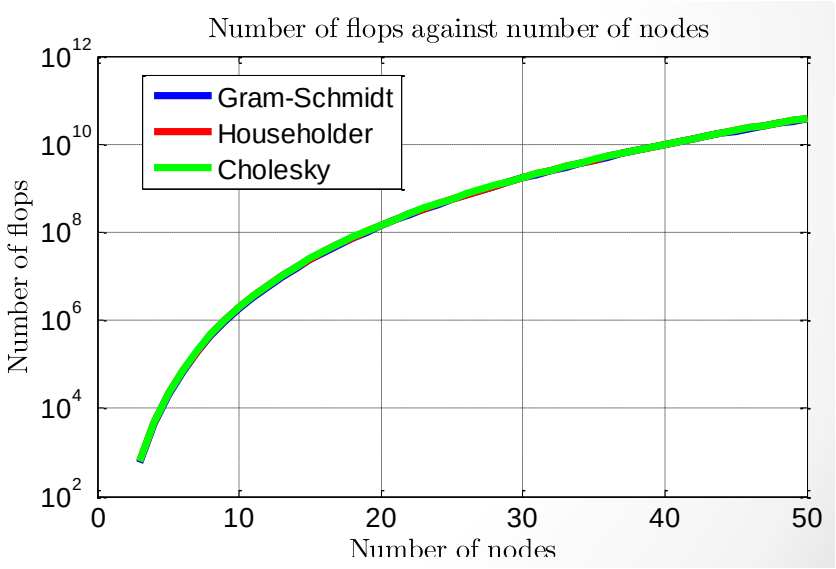

Figura 4.1: Complejidad con el aumento de los nodos en una red

Capítulo 5

Conclusiones

Varias conclusiones pueden sacarse una vez realizado el proyecto. El principal objetivo era diseñar un sistema de localización cooperativa en una red de dispositivos basada en una secuencia de transmisión de pulsos con el objetivo de mejorar el sistema existente *3-way ranging*. El primer aspecto a ser mejorado era la cantidad de distancias conocidas por cada nodo, esto es, que cada nodo no solo fuera capaz de medir la distancia entre él mismo y los demás sino también conocer las demás distancias entre los demás nodos entre sí. Otro aspecto secundario, pero también importante, a mejorar, era la cantidad de transmisiones de pulsos necesarios. Ambos aspectos han sido mejorados mediante este nuevo sistema. Las ventajas de esta aproximación son significantes por lo tanto este método es apto para ser desarrollado.

Una vez definido el sistema, el problema del diseño de secuencias aparece. Una secuencia válida debe satisfacer diversas restricciones que el propio sistema impone. Después de estudiar todas estas restricciones, se ha desarrollado un algoritmo. Este algoritmo ha de ser simple y eficiente debido a que la longitud de la secuencia aumenta factorialmente con el número de nodos, y este número puede llegar a ser muy grande. Una vez diseñado este algoritmo, puede concluirse que la idea de la checkmatrix es una buena herramienta que hace que el algoritmo sea eficiente, reduce el número de iteraciones comparado con la fuerza bruta. Otra buena característica de la checkmatrix es que muestra cómo se va formando la secuencia de una forma clara si se ejecuta el algoritmo paso a paso. El segundo algoritmo propuesto se basa en el primero. La simplicidad aquí toma mayor peso dado el gran número de posibles secuencias válidas para redes grandes. Es conveniente guardar en un fichero lo obtenido para usar los datos en el futuro, ya que ejecutar este algoritmo cada vez lleva un tiempo excesivo.

Después del tema de las secuencias, el siguiente paso era diseñar un filtro para procesar todos los datos medidos en cada nodo. Se ha escogido el filtro de Kalman debido a su capacidad de seguimiento. La idea inicial era diseñar un filtro de Kalman capaz de calcular rangos en la red. Puede concluirse que el filtro de Kalman es una herramienta válida para este sistema y funciona perfectamente. Después de probar que el

filtro elegido era el correcto, la idea era reducir el coste computacional en cada nodo, esto es, diseñar el filtro minimizando el número de operaciones. Después de analizar el filtro paso a paso, la conclusión es que la mayor contribución al coste computacional la aporta la inversión de la matriz de la ganancia Kalman. Para reducir las operaciones, varias formas de descomposición de matrices se han estudiado, siendo la descomposición QR la estudiada con más detalle. La conclusión obtenida es que la forma más óptima de invertir la matriz es emplear la descomposición QR mediante el método de la ortogonalización de Gram-Schmidt. La transformación de Householder y la descomposición de Cholesky son algoritmos más estables pero dada la gran estabilidad del problema, este aspecto puede ser descartado. Las rotaciones de Givens se descartaron enseguida debido a su alto coste computacional. Finalmente, para invertir la matriz, se ha escogido descomponerla mediante la ortogonalización de Gram-Schmidt e invertir mediante la sustitución hacia atrás.

Después de calcular la complejidad en cada nodo, se ha discutido cómo afecta el tamaño de la red a la misma. Después de discutirlo, la conclsión es que el número de operaciones en cada nodo se relaciona con el número de nodos a la sexta potencia.

Una vez estudiada esta propuesta de secuencias de transmisión de pulsos, otra mejora fue propuesta. En vez de estimar rangos, estimar la posición en un escenario bidimensional. Para poder hacerlo, el filtro de Kalman fue modificado dado que esta propuesta requiere de un filtrado no lineal, de este modo, el filtro propuesto fue el filtro de Kalman extendido (EKF). El EKF emplea derivadas y Jacobianos para linearizar las no linealidades. El EKF debería poder estimar las posiciones, pero después de realizar varias simulaciones, la conclusión obtenida es que la linearización de las medidas introduce errores severos en el sistema y no funciona correctamente. Esto es, el EKF no es válido para este sistema.

5.1 Líneas futuras

La primera sugerencia es estudiar otro tipo de filtro para lograr estimar la posición. Como el EKF no es válido se propone el estudio del Unscented Kalman filter (UKF), que en lugar de emplear linearizaciones, trabaja con estadística. El UKF se comporta mucho mejor con las no linealidades.

Otra sugerencia es desarrollar un protocolo de comunicaciones entre nodos. En vez de transmitir solo pulsos, enviar también, identificadores, errores u otro tipo de datos para añadir robustez al sistema.

También puede estudiarse la forma de juntar varias redes de forma automática. Un ejemplo sería nombrar un nodo maestro en cada red. Otra forma sería realizar un recálculo de la secuencia si se detectan nuevos nodos en el sistema.

La última sugerencia es mejorar el sistema para mejorar su funcionamiento en entornos adversos, que sea capaz de lidiar con propagaciones multicamino, caídas de

enlaces, caídas de nodos, desvanecimientos, etc.

Referencias

[1] G. de Tecnologías de las Comunicaciones, *Introducción a los sistemas de radionavegación*. De las notas del curso 18117, Sistemas de radionavegación, Universitdad de Zaragoza.

[2] N. Patwari, J. N. Ash, S. Kyperountas, A. O. H. III, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing magazine*, vol. 22, pp. 54–69, jul. 2005.

[3] M. Z. Win, A. Conti, S. Mazuelas, Y. Shen, W. M. Gifford, D. Dardari, and M. Chiani, "Network localization and navigation via cooperation," *IEEE Communications magazine*, vol. 49, pp. 56–62, may. 2011.

[4] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative localization in wireless networks," *Proceedings of the IEEE*, vol. 97, pp. 427–450, feb. 2009.

[5] S. Dwivedi, A. D. Angelis, and P. Händel, "Scheduled uwb pulse transmission for cooperative localization," *IEEE ICUWB*, 2012.

[6] J.-O. Nilsson, A. D. Angelis, I. Skog, P. Carbone, and P. Händel, "Signal processing issues in indoor positioning by ultra wide band radio aided inertial navigation," *17th European Signal Processing Conference (EUSIPCO 2009), Glasgow*, aug. 2009.

[7] S. Haykin, *Adaptive Filter Theory*. Prentice Hall, 4 ed., 2001.

[8] P. G. Kaminski, A. E. Bryson, and S. F. Schmidt, "Discrete square root filtering: A survey of current techniques," *Automatic control, IEEE Transactions*, vol. 16, pp. 727–736, dec. 1971.

[9] H. W. Sorenson, "Least-squares estimation: from gauss to kalman," *Spectrum,IEEE*, vol. 7, pp. 63–68, jul. 1970.

[10] W. Gander, "Algorithms for qr-decomposition," tech. rep., Eidgenoessische Technische Hochschule, 1980.

[11] T. Lacey, *Tutorial: The Kalman filter*. De las notas del curso CS7322 en Georgia Tech.

[12] Y. T. Chan, A. G. C. Hu, and J. B. Plant, "" A Kalman Filter Based Tracking Scheme with Input Estimation "," *IEEE Transactions on aerospace and electronic systems*, vol. AES-15, pp. 237–244, mar. 1979.

[13] A. El-Amawy and K. R. Dharmarajan, "Parallel vlsi algorithm for stable inversion of dense matrices," *Computers and Digital Techniques, IEE Proceedings*, vol. 136, pp. 575–580, nov. 1989.