



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



UNIVERSIDAD POLITÉCNICA DE VALENCIA
DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA
MASTER EN AUTOMÁTICA E INFORMÁTICA INDUSTRIAL

TESINA DEL MASTER

**NAVEGACIÓN DE UN ROBOT MÓVIL DE
CONFIGURACIÓN DIFERENCIAL BASADA
EN FUSIÓN SENSORIAL**

Realizada Por:

Leonardo MARÍN PANIAGUA

leomarpa@posgrado.upv.es

Dirigida por:

Dra. Marina VALLÉS MIQUEL

Dr. Ángel VALERA FERNÁNDEZ

Valencia, 19 de julio de 2011

Índice general

1. Introducción	1
1.1. Objetivos	1
1.2. Localización y Navegación mediante seguimiento de trayectorias	2
1.3. Antecedentes	5
2. Descripción de las plataformas	11
2.1. Robot móvil <i>e-puck</i>	11
2.2. Robot móvil LEGO NTX	13
2.3. Sensor Inercial IG500N abordo de un coche	13
3. Teoría del filtro de Kalman	15
3.1. El filtro de Kalman	16
3.2. Filtro de Kalman extendido	18
3.3. Filtro de Kalman Unscented	19
4. Modelos de un robot en configuración diferencial	21
4.1. Modelo cinemático	21
4.2. Modelo dinámico	25
4.3. Modelo dinámico de los motores	32
5. Esquemas de fusión de datos con el filtro de Kalman	39
5.1. Calibración y preprocesamiento de los Sensores	39
5.2. Parámetros del filtro de Kalman	41
5.3. Filtro de Kalman para la velocidad de las ruedas	42
5.4. Filtro de Kalman para la velocidad del robot	43
5.5. Filtro extendido de Kalman para la posición del robot	46
5.6. Filtro UKF para la posición del robot	48

5.7. Sensorización global por eventos	48
5.8. Simulación y comparación de los esquemas de fusión	51
6. Pruebas implementadas de la fusión de datos	56
6.1. Pruebas con el e-puck	56
6.2. Pruebas con el LEGO NTX	58
6.3. Pruebas en el sensor IG500N	65
7. Conclusiones	67
8. Trabajo Futuro	69
Bibliografía	71

Índice de figuras

1.1.	Diagrama de bloques del control de posición por Punto Descentralizado.	4
1.2.	Puntos de interés para el control por Punto Descentralizado.	4
2.1.	Sensores principales del robot <i>e-puck</i>	12
2.2.	Interconexión de componentes en el robot <i>e-puck</i>	12
2.3.	Robot móvil LEGO NXT utilizado	14
2.4.	Sensor inercial IG500N	14
3.1.	Forma de operar del filtro de Kalman.	15
3.2.	Algoritmo del filtro de Kalman.	18
3.3.	Algoritmo del filtro de Kalman Extendido	19
3.4.	Propagación de la media y la covarianza en la UT	20
4.1.	Cinemática de un robot en configuración diferencial	22
4.2.	Dinámica de un robot en configuración diferencial	27
4.3.	Dinámica del robot como sistema de partículas equivalente	28
4.4.	Pruebas realizadas a los motores del LEGO NXT	33
4.5.	Respuesta del modelo local promedio de las ruedas del LEGO NXT	35
4.6.	Respuesta del modelo global de las ruedas del LEGO NXT	36
4.7.	Respuesta del modelo global Promedio de las ruedas del LEGO NXT	38
5.1.	Esquema de fusión de sensores: KF de velocidades y acelerómetros	44
5.2.	Esquema de fusión de sensores: KF de velocidades sin acelerómetros	45
5.3.	Esquema de fusión de sensores: EKF de velocidades y acelerómetros	46
5.4.	Esquema de fusión de sensores: EKF de velocidades sin acelerómetros	47
5.5.	Esquema general Sensorización global por eventos	51
5.6.	Trayectoria cuadrada del LEGO NXT sin implementar el KF	52
5.7.	Trayectoria circular del LEGO NXT sin implementar el KF	53

5.8. Simulación: KF estimando la velocidad lineal	54
5.9. Simulación: KF estimando la velocidad angular	54
5.10. Simulación: sensorización global por eventos	55
6.1. Velocidad rueda derecha e-puck con el KF	57
6.2. Velocidad rueda izquierda e-puck con el KF	58
6.3. E-puck: Trayectoria cuadrada con el KF de velocidades	59
6.4. E-puck: Trayectoria circular con el KF de velocidades	59
6.5. Filtro KF velocidad robot dentro del LEGO NXT, con acelerómetros	61
6.6. Filtro KF velocidad robot dentro del LEGO NXT, sin acelerómetros	62
6.7. Estimación velocidad angular, KF dentro del LEGO	62
6.8. Estimación ángulo de avance, KF dentro del LEGO	63
6.9. Estimación velocidad lineal, KF dentro del LEGO	63
6.10. EKF con acelerómetros dentro del LEGO, posición del robot	64
6.11. Control por eventos simulando la comunicación, KF dentro del LEGO	64
6.12. Prueba IG500N recorrido GPS	66
6.13. Prueba IG500N comparación KF	66

RESUMEN

Uno de los aspectos esenciales en la robótica móvil es la obtención y procesamiento de la información relativa a la localización del robot en el espacio de movimiento con el fin utilizarla para generar los movimientos deseados del robot. Para esto se busca utilizar la mayor cantidad posible de fuentes de información con el fin de corregir los errores de posición asociados a la presencia de ruido en las mediciones del robot. La fusión de toda esta información en una sola medida que pueda ser utilizada en el control de robot es tema central del presente trabajo en el cual se expone la implementación de una fusión de distintos datos provenientes de sensores para mejorar la navegación en robots móviles con recursos de computación limitados. Para ello, se hace una revisión de las técnicas existentes para la fusión de datos, poniendo especial interés en las correspondientes a filtros de Kalman.

Se implementaron y probaron distintos esquemas de fusión de sensores utilizando información proveniente de sensores inerciales comunes de un robot en configuración diferencial (acelerómetros, giroscopios, brújula y encoders). Estas pruebas permitieron obtener el método de fusión de sensores propuesto, el que utiliza un filtro de Kalman junto con la información de un modelo local del robot móvil (modelo dinámico con descomposición por partículas inerciales junto con el modelo cinemático) de un robot móvil diferencial y la información de uno o varios sensores inerciales (según la plataforma). Este método propuesto es muy eficiente en términos de utilización de recursos lo cual permite su implementación en robots con recursos limitados. Además su desempeño es comparable a los esquemas de fusión más complejos que utilizan un modelo no lineal y los filtros de Kalman Extendidos y Unscented tal y como se muestra en los resultados obtenidos.

El esquema propuesto se probó ampliamente en distintas plataformas como el robot *e-puck*, el sensor inercial industrial *IG500* y principalmente utilizando el robot móvil *LEGO NXT* debido a su capacidad de utilizar distintos sensores inerciales, todo esto con el fin de comprobar el buen desempeño del método propuesto.

Los esquemas de fusión sensorial propuestos utilizan información proveniente de mediciones inerciales por lo que se requiere disponer de capacidades de sensorización globales para evitar que la incertidumbre en la posición crezca de forma exponencial y al cabo de cierto tiempo el robot desconozca con certeza donde se encuentra. Para evitar esto se acopla al método local propuesto una corrección basada en eventos utilizando una cámara cenital como fuente de información global. Se agrega a la estimación del filtro la diferencia entre la posición estimada y la medición de la cámara para hacer que el robot corrija su posición y disminuya el error. Esto se realiza compensando el retardo en comunicación debido a la comunicación entre el robot y la cámara. Con esto se mejora la exactitud de la navegación del robot y se muestra el caso implementado en el robot *LEGO NXT* con un buen desempeño de la corrección al simular la comunicación con la cámara.

Se expone finalmente las posibilidades de ampliación del presente trabajo en áreas como localización distribuida y la corrección global en exteriores utilizando sensores GPS o una cámara cenital colocada en un robot helicóptero.

Palabras clave: Fusión de datos, filtros de Kalman, odometría, localización de robots, robots móviles, *LEGO NXT*, *e-puck*, control de trayectorias.

Agradecimientos: Este trabajo ha sido financiado parcialmente por el Ministerio de Ciencia e Innovación de España bajo los proyectos FEDER/CICYT de investigación DPI2008-06737-C02-01 y DPI2010-20814-C02-02. Además se agradece el soporte financiero por parte de la Universidad de Costa Rica.

Capítulo 1

Introducción

Con el fin de comprender el problema de la fusión sensorial para la localización y la navegación de robots móviles en configuración diferencial es importante revisar previamente la teoría básica de navegación y localización de robots móviles junto con la teoría asociada al Filtro de Kalman. Esta se expone a continuación junto con los objetivos principales del presente trabajo, que constituyen los pasos seguidos para la obtención del esquema de fusión de sensores propuesto así como su implementación y pruebas en las plataformas propuestas. Se incluye además un apartado con la investigación bibliográfica realizada sobre los trabajos existentes en fusión sensorial los cuales sirven de referencia para el método propuesto.

1.1. Objetivos

Para la realización del presente trabajo se estableció como objetivo general la mejora de la navegación de un robot móvil de configuración diferencial utilizando la fusión de la información de los distintos sensores locales del robot. Los objetivos específicos son:

- Realizar una revisión del estado del arte de la navegación a partir de la información local de los robots móviles y de las técnicas de fusión de datos basadas en modelos lineales y no lineales, principalmente en la técnica del filtro de Kalman y sus variantes no lineales (Extendido y el «Unscented»).
- Obtener por medios analíticos un modelo cinemático y dinámico no lineal de un robot

móvil en configuración diferencial así como un modelo dinámico de la velocidad y aceleración de las ruedas del robot.

- Implementar la fusión de datos utilizando las variantes no lineales del filtro de Kalman para fusionar la información de los distintos sensores del robot que aportan información relativa de su movimiento (encoders, acelerómetro, giroscopio, brújula digital).
- Implementar una fusión de datos mediante eventos para incorporar la información de un sensor de posición global (por ejemplo la información de una cámara de visión cenital con procesamiento de visión por OpenCV o similar).
- Comparar el comportamiento de la fusión de datos implementada utilizando mediciones «a posteriori» de los sensores del robot en un movimiento dentro de una trayectoria.
- Estudiar la viabilidad de la implementación práctica del mejor algoritmo de fusión desarrollado dentro de un robot móvil con capacidad de cómputo limitada.
- Realizar la implementación del algoritmo de fusión adecuado dentro de los robots en los que sea viable y mostrar su desempeño comparando la estimación del filtro de Kalman con la posición medida mediante un sistema de visión por ordenador para distintas trayectorias.

1.2. Localización y Navegación mediante seguimiento de trayectorias

Un aspecto clave de los robots móviles autónomos es su localización, de tal manera que sepan su posición exacta en el espacio. Sin esta información el robot no es capaz de navegar en el entorno, seguir una trayectoria, ir a una determinada posición o volver al punto de inicio de un movimiento, reduciendo casi todas las capacidades del robot móvil. Este aspecto ha sido ampliamente estudiado en la literatura (para una visión global consultar los capítulos 5 y 6 de [61]) y aunque puede parecer trivial resulta desafiante ya que la información de la posición del robot se obtiene de sensores con ruido y no-linealidades, que si no se tienen en cuenta puede conllevar una gran incertidumbre en la medida de la posición. La localización del robot se puede estudiar de varias formas. Suponiendo que la posición inicial antes de iniciar el movimiento es conocida, la posición actual del robot se puede estimar usando información local del movimiento del robot obtenido a través de distintos sensores de forma que se calcule la distancia recorrida desde el

punto inicial; a esto se le conoce como *odometría* (ver [61] y [51]) o *estimación de la posición local*. La odometría estima la posición del robot en un plano (x y y y el ángulo θ , ver figura 1.2) a partir de la posición inicial $P = [x \ y \ \theta]^T$ y la velocidad del robot en los ejes (x, y) junto con la velocidad de cambio angular ω . Para un periodo de muestreo T_s pequeño (cercano a los 50ms) se tiene que la posición del robot P' está dada por la ecuación (1.1) ([51] y [61]).

$$P' = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + T_s \begin{bmatrix} v_{k-1} \cos(\theta_{k-1}) \\ v_{k-1} \sin(\theta_{k-1}) \\ \omega_{k-1} \end{bmatrix} \quad (1.1)$$

Este procedimiento tiene la ventaja de que el tiempo de respuesta es pequeño y el inconveniente de que un error entre la posición real y la estimada se acumula a lo largo del tiempo. Debido a esto, tras recorrer una cierta distancia la estimación de la posición puede ser muy diferente de la posición real.

Otra forma de conocer la posición del robot es usar un sensor complejo como puede ser un sistema de posicionamiento global (GPS o cámara cenital) para conocer su posición absoluta en el entorno si tener que usar la información local (lo cual evita que el error de posición crezca de forma indefinida). Este procedimiento es conocido como *estimación de posición global* y tiene la desventaja de tener un tiempo de respuesta elevado así como la limitación de trabajar únicamente en interiores (cámara) o exteriores (GPS) dependiendo del sensor utilizado. Dado que la información sobre la posición del robot la necesita el algoritmo de navegación, el tiempo de respuesta del método de localización resulta importante. Si se pierde la información de la localización del robot o se recibe de forma lenta, el algoritmo de control no tendrá actualizados los datos y la acción de control producida será incorrecta lo cual podría provocar que el sistema fuese inestable. Debido a esto, esta estimación local de la posición se usa habitualmente como principal fuente de información para el algoritmo de navegación mientras que el sistema de posicionamiento global se utiliza para corregir la estimación local cuando la información global está disponible (ver por ejemplo el esquema propuesto en [33]).

La información sobre la posición (ecuación (1.1)) del robot es utilizada principalmente por el algoritmo seguimiento de trayectorias el cual se encarga de producir la referencia de las velocidades de las ruedas del robot. Esta referencia se sigue mediante un control PID que regula la velocidad de los motores. Se utilizará un algoritmo muy conocido y eficiente para realizar el seguimiento de trayectorias sin obstáculos en las pruebas de los robots a la hora de obtener los datos de los sensores y realizar las pruebas experimentales. Este algoritmo es el de seguimiento de trayectorias con un control por punto descentralizado ([61] y [66]) el cual se resume a continuación. Este algoritmo se hace que el robot se mueva siguiendo una trayectoria prede-

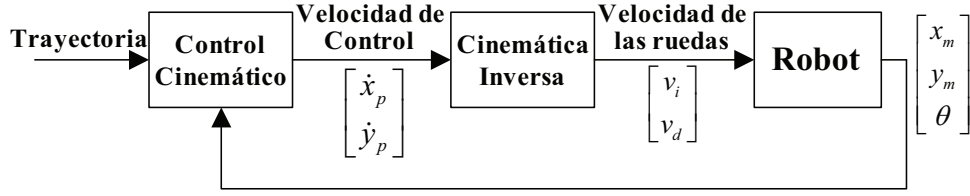


Figura 1.1: Diagrama de bloques del control de posición por Punto Descentralizado.

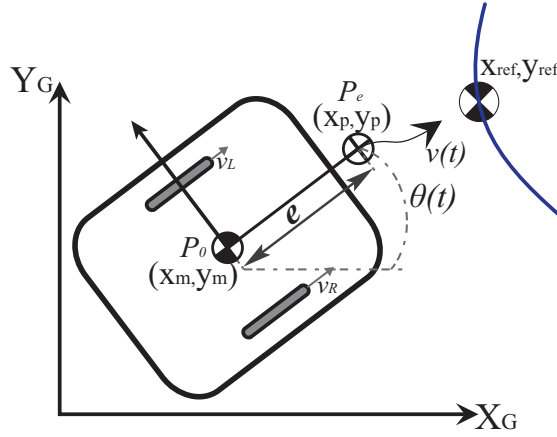


Figura 1.2: Puntos de interés para el control por Punto Descentralizado.

finida, (por ejemplo un cuadrado, un círculo o cualquier función paramétricas en el tiempo). Su diagrama de bloques se muestra en la figura 1.1. Este control se establece a partir de la posición y velocidad de un punto que está separado una distancia e desde el eje de tracción del robot (figura 1.2 y ecuación (1.2)), esto hace que el robot gire con antelación con el fin de que, cuando termine de girar, su origen se encuentre sobre la trayectoria deseada. Al utilizar el punto descentralizado se define la ecuación (1.3) que relaciona las velocidades de las ruedas (v_L, v_R) del robot respecto a la velocidad global del robot.

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} x_m + e \cos(\theta) \\ y_m + e \sin(\theta) \end{bmatrix} \quad (1.2)$$

$$\begin{bmatrix} v_L \\ v_R \end{bmatrix} = \frac{1}{e} \begin{bmatrix} e \cos \theta + 0,5b \sin \theta & e \sin \theta - 0,5b \cos \theta \\ e \cos \theta - 0,5b \sin \theta & e \sin \theta + 0,5b \cos \theta \end{bmatrix} \begin{bmatrix} \dot{x}_p \\ \dot{y}_p \end{bmatrix} \quad (1.3)$$

El funcionamiento de este esquema de control (figura 1.1) es el siguiente: primeramente se obtiene la posición actual del robot (utilizando la ecuación de odometría 1.2) y su diferencia respecto a la posición deseada (la cual está definida por la trayectoria especificada como una función paramétrica del tiempo), este error es usado por el control cinemático para determinar la velocidad necesaria en los ejes globales para alcanzar la posición de referencia. Esta posición

es traducida por el modelo de cinemática inversa (ecuación (1.3)) para obtener las velocidades de las ruedas del *e-puck*, las cuales son enviadas al robot móvil. Como la referencia cambia constantemente (en cada instante de muestreo según las funciones paramétricas) esto produce que el error cambie constantemente por lo que el control implementado hace que el robot se mueva en la trayectoria deseada. El control cinemático implementado es un control proporcional con prealimentación de la velocidad tal y como se muestra en la ecuación (1.4).

$$\begin{bmatrix} \dot{x}_p \\ \dot{y}_p \end{bmatrix} = \begin{bmatrix} k_{v_x} \dot{x}_{ref} \\ k_{v_y} \dot{y}_{ref} \end{bmatrix} + \begin{bmatrix} k_{p_x} & 0 \\ 0 & k_{p_y} \end{bmatrix} \begin{bmatrix} x_{ref} - (x_m + e \cos(\theta)) \\ y_{ref} - (y_m + e \sin(\theta)) \end{bmatrix} \quad (1.4)$$

Este esquema básico es el que se utilizará a la hora de experimentar con el robot para obtener las medidas de sus sensores. Se observa que este esquema supone una única medida o estimación de la velocidad lineal v y angular ω del robot tal y como se establece en la ecuación básica de la odometría ((1.3)). En general esta medida proviene de los encoders que miden el desplazamiento de las ruedas del robot y si su precisión es alta y el nivel de ruido presente es bajo, el error en la estimación de la posición será bajo también (aunque tiende a crecer si no hay información global de la posición).

Sin embargo, es conveniente incorporar medidas provenientes de otros sensores locales como una brújula, un giroscopio o los acelerómetros disponibles en el robot o de un sensor global (cámara o un GPS) con el fin de aumentar la precisión de la medición, disminuir el ruido de medición o corregir la desviación de la variable de su valor real. En este caso surge el problema de cómo realizar la integración de la información de los distintos sensores, tomando en cuenta sus niveles de ruido y nivel de resolución, en una sola medición que pueda ser utilizada por el algoritmo de control, o bien cómo determinar qué información descartar y cual utilizar para realizar el control. Este tema ha sido ampliamente estudiado tal y como se muestra en la siguiente sección de Antecedentes. Estos se centran en las técnicas del filtro de Kalman (KF) el cual es el esquema seleccionado para la implementación de la fusión en los capítulos siguientes.

1.3. Antecedentes

De la investigación bibliográfica realizada se encontraron varios trabajos que sirven de base al desarrollo del esquema de fusión sensorial realizado en el presente trabajo. Se resumen a continuación los aspectos esenciales de esos trabajos y de las distintas ideas que aportan al desarrollo del método de fusión sensorial propuesto. Una de las técnicas más conocidas y eficientes para fusionar datos es el filtro de Kalman (KF), la cual tiene en cuenta la exactitud de cada sensor

para juntar la información de forma óptima. Existen varias versiones del filtro ampliamente estudiadas en la literatura (Normal (KF) y Extendido (EKF) [26], [70], Unscented (UKF) [40]) y todas estas versiones se pueden utilizar en muchos casos prácticos siempre y cuando el robot disponga de suficientes recursos para implementar el filtro.

Por ejemplo, para estimación local (odométrico), en [49] se utiliza la versión extendida y aumentada del filtro para fusionar datos de un sensor de rango laser (para detectar paredes cercanas al robot) y estimar el error producido en la odometría. También en [8], [32], [33] y [34] se usó la versión Unscented del filtro para combinar la información de un giroscopio, en este caso se utiliza el giroscopio a modo de sensor global por lo que se utiliza para corregir el error en la estimación de la odometría. Es la diferencia entre estas mediciones la que es introducida al filtro de Kalman Unscented el cual utiliza un modelo del error (calculado de forma cinemática) para estimar el error de la odometría y compensarlo (sumando la estimación del error a la odometría). Este esquema da buenos resultados sobre un robot Pioneer 2-DXe controlado desde un ordenador externo al robot (los filtros son implementados en Matlab [®]) aunque las pruebas realizadas son bastante sencillas. En esta misma línea el trabajo presentado en [54] muestra un modelo de odometría extendido para considerar los errores de modelado y de calibración de los encoders (por ejemplo ruedas de diámetros distintos entre sí) el cual se integra con la medición de un giroscopio. Este caso muestra un buen desempeño pero con la desventaja de que el modelo desarrollado es muy complejo (11 estados) lo que complica mucho el cálculo del filtro de Kalman y su implementación dentro del robot, además de la utilización de un sistema complejo para supervisar el movimiento del robot (cámaras Hawk Digital System para captura de movimientos en tres dimensiones) hace que su aplicación práctica sea muy costosa.

La fusión sensorial se ha implementado en robots más complejos a los diferenciales, por ejemplo los tipo oruga, en [55] se utiliza la información adicional de un giroscopio para corregir la estimación de la odometría en un robot considerando el deslizamiento. Esta aplicación usa un modelo muy simple de odometría y se realiza la fusión de forma directa (el ángulo del robot se mide con el giroscopio) desaprovechando la posibilidad de tomar en cuenta la estimación del ángulo medido de los encoders y el modelo cinemático. En cambio en [71] se usa un robot del mismo tipo pero usando un UKF para estimar los parámetros del deslizamiento del robot. En este caso la estimación funciona adecuadamente pero no se utilizan muchos de los sensores del robot para mejorar la estimación de la odometría (por ejemplo se utilizan solamente los encoders a pesar de contar con un giroscopio y dos acelerómetros).

Otro caso similar es el mostrado en [31] en donde se utiliza la fusión sensorial (local y global)

en un robot móvil «Packbot» tipo oruga. Este es un robot sumamente avanzado con tecnología militar y gran capacidad de procesamiento. Se utiliza el filtro de Kalman basado únicamente en sensores (sin considerar la dinámica del robot sino únicamente las mediciones de los sensores) para fusionar la información del acelerómetro, del giroscopio, del DGPS (GPS diferencial) y de la brújula con el fin de estimar de manera muy precisa la posición del robot en el espacio así como su orientación para de esta forma seguir una trayectoria predeterminada. Se obtienen resultados muy robustos, inclusive en el caso de pérdida de comunicación con el DGPS, mostrando una superioridad considerable del filtro (implementado en el robot) al usar fuentes de información locales y globales junto con un modelo sencillo (dependiente de sensores exclusivamente). Este artículo muestra un resultado muy sólido sin requerir de un modelo muy complejo de la dinámica del robot, sino que se basa únicamente en los sensores del robot. Esta idea se sigue en el presente trabajo pero implementando un modelo dinámico sencillo para mejorar la estimación basada únicamente en sensores. Otro ejemplo es el caso de los robots omnidireccionales los cuales requieren un modelo dinámico complejo (ver en [50] un modelo completo cinemático y dinámico). Por ejemplo en [29] se implementa una versión adaptativa del UKF en donde se ajustan en línea los parámetros del filtro (covarianza del modelo y de la medición), se muestran buenos resultados para un motor de masa considerable (120Kg) utilizando un modelo dinámico bien definido (el robot es de construcción propia).

Existen otros trabajos que se basan principalmente en la estimación de la posición utilizando sensores globales. En [42] se presenta un esquema muy interesante realizando una fusión de un «satélite» ultrasónico (U-SAT: GPS para interiores) con emisores de ultrasonido colocados en el entorno (de posición conocida) y un receptor en el robot (ver una descripción completa del sistema en [46]). Este sensor global se utiliza junto con un sistema de navegación inercial (INS) compuesto por un giróscopo y encoders, los cuales son incorporados mediante un filtro de Kalman Extendido y un modelo cinemático del robot para determinar la posición del robot. Como el sensor U-SAT es lento en realizar la medición de la posición (tiempo cercano a los 400ms) se utiliza la INS junto con el KF para interpolar los valores de la posición cada 100ms para que de esta forma el algoritmo de navegación pueda funcionar correctamente (actuando cada 100ms). Se muestran resultados adecuados pero mejorables al implementar el método en un ordenador externo al robot y medir los datos de los sensores del robot móvil mediante WiFi. Un esquema similar se presenta en [45] utilizando el mismo tipo de sensor global basado en localización por ultrasonido pero utilizando un filtro UKF. Se muestran buenos resultados al utilizar el filtro con los datos del sensor prefiltrados (se eliminan los valores que están fuera de la desviación estándar de los cuatro emisores de ultrasonido) pero con una configuración de movimiento circular

permanente, es decir, sin implementarlo en un robot móvil sino en un sistema de movimiento fijo. La ventaja del sistema propuesto es la estimación simultánea de la posición, velocidad y aceleración del dispositivo sensor diseñado y un buen desempeño del UKF, sin embargo este trabajo tiene la opción de ampliarse para ser implementado en un robot móvil para realizar seguimiento de trayectorias. Otro ejemplo de estimación con información global se muestra en [59] en donde se usa un GPS y un filtro de Kalman extendido con actualización dinámica de sus parámetros (covarianza a priori del error) los cuales se ajustan según el desempeño en línea del filtro (observando los residuos) para lo cual se utiliza lógica difusa. Se muestran muy buenos resultados al implementar el esquema en un robot avanzado tipo rover con un receptor de GPS incorporado y realizando trayectorias de gran longitud.

Existe gran cantidad de implementaciones de sistemas de estimación global con esquemas de fusión más avanzados, por ejemplo en [21], en este se incorpora con un filtro de Kalman Extendido híbrido, el cual utiliza un modelo cinemático del robot (odometría) también un sensor de rango laser pero esta vez para detectar balizas de posicionamiento, estas son referencias globales de las cuales se conoce su posición real, el filtro se encarga de medir de forma discreta la posición de estas balizas y usar esta información para identificar que baliza se ha detectado (problema de correspondencia) y según esto corregir la información de la odometría. Este método es muy similar a los esquemas SLAM (simultaneous localization and mapping - localización y generación de mapa simultáneo) con filtros de Kalman los cuales debido a su complejidad están fuera del objetivo del presente estudio (para una visión global de este tema y la utilización del EKF para realizar SLAM ver [15]). Además las pruebas mostradas son simulaciones en Matlab lo cual indica un alto coste computacional y la imposibilidad de aplicar el método a robots de recursos limitados. Sin embargo cabe destacar el manejo discreto de las mediciones y el sistema de balizas que puede ser adaptado para ser utilizado en lugar de una cámara cenital para navegación en interiores. Un esquema similar se presenta en [18] pero para la localización de grupos de robots modelándolos como una red de sensores distribuidos (se busca localizar el centro del grupo de robots) y utilizando balizas de posición conocida junto con el EKF y UKF. Se prueba el sistema en una simulación con datos de robots reales en donde se observa una buena localización del grupo para el caso del UKF, mientras que en el EKF el error de la estimación de la posición crece indefinidamente debido a la no linealidad del sistema. Otro estudio similar realizado en [35] muestra un buen desempeño realizando la fusión entre la odometría y un conjunto de sensores ultrasónicos para localizar al robot en base a un mapa conocido (SLAM) pero utilizando un EKF con parámetros adaptativos en línea (covarianza del modelo y de la medición).

En esta misma línea de investigación de localización mediante sensores globales, se pueden utilizar cámaras para observar características distintivas del entorno y fusionar esta información con la odometría del robot para disminuir la incertidumbre en la posición del robot. Este esquema se utiliza en [17] para observar una trayectoria marcada en el suelo y, con un filtro extendido de Kalman, fusionar la estimación del segmento de trayectoria con la información de la odometría (basada en encoders). Sin embargo en este caso solo se muestran simulaciones y el autor menciona que se puede mejorar la estimación utilizando un filtro UKF. Un esquema similar se aplica en [44] pero utilizando la información experimental de un «robot» que en realidad es un cortacéspedes con sensores inerciales (encoders, giroscopio) junto con dos cámaras para detectar marcas en el terreno y un DGPS (utilizado únicamente para supervisar el seguimiento de la trayectoria). Se muestra un buen resultado fusionando la información con un EKF junto con un modelo linealizado del robot y la información de las cámaras que miden el ángulo relativo a las marcas de posición conocida. Finalmente, en [66] se usa una cámara cenital como GPS de interior y se fusiona con datos de los encoders sobre un robot LEGO de primera generación. Se observan muy buenos resultados a pesar de la gran cantidad de ruido presente en la medición de los encoders y del retardo de comunicación entre la cámara y el LEGO. Este desarrollo es de los principales referentes del presente trabajo ya que se intentará mejorar la estimación de la posición con la fusión de información de varios sensores además de los encoders y la utilización de la nueva versión del Lego con mejores prestaciones.

En todos estos ejemplos hay una característica común, que es el hecho de que la mayoría de los algoritmos implementados para realizar la fusión de sensores son muy costosos computacionalmente (ya que necesitan la inversión y manipulación de matrices grandes para realizar la estimación con el filtro de Kalman) y en la mayor parte de los casos no se implementan dentro del procesador del robot. Esto significa que se implementan como una simulación o en un computador externo, con el problema de los retardos en la comunicación entre el computador controlador y el robot y el hecho de que estos métodos dependen exclusivamente de su ejecución fuera del propio robot en un ordenador con gran capacidad de cómputo comparado a un robot de recursos limitados. Debido a esto es necesario implementar un nuevo método de fusión basado en la técnica del filtro de Kalman que permite una integración de múltiples sensores de forma rápida, fiable y simple junto con un filtrado de ruido y estimación óptima de estado. Esto con el fin de permitir la implementación del algoritmo dentro del robot sin ejecutar la estimación de la posición y el control fuera de este.

El método propuesto en las secciones siguientes mejora la estimación de la posición del robot

y es capaz de ejecutarse dentro del robot móvil considerando la fusión de al menos dos fuentes de información, un modelo dinámico del movimiento del robot expresado de forma sencilla (para que sea calculado de forma rápida) y la fusión de los sensores locales del robot, de donde para obtener las coordenadas globales de la posición se aplicará la odometría utilizando como entradas la fusión local (modelo y sensores) de las velocidades lineal y angular. Esto mantiene el modelo del robot lineal permitiendo reducir las dimensiones de las matrices requeridas en el proceso del filtro de Kalman lo que facilita su implementación. Se expone a continuación la descripción de las plataformas a utilizar en el presente trabajo.

Capítulo 2

Descripción de las plataformas

A continuación se describen las plataformas utilizadas para la prueba de los algoritmos de fusión aunque estos pueden implementarse en más plataformas adaptando las ecuaciones que incorporan las mediciones y los parámetros de los modelos descritos en los capítulos siguientes.

2.1. Robot móvil *e-puck*

El *e-puck* [3] (figura 2.1) es un robot móvil de ruedas diferenciales desarrollado por el Dr. Francesco Mondada y Michael Bonani en el 2006 en la *EPFL*, (*Federal Swiss Institute of Technology in Lausanne*) ([51] y [52]) y es un robot de hardware y software abierto. Consta de dos ruedas motorizadas con la capacidad de girar en direcciones opuestas para hacer cambiar la dirección del robot. Además cuenta con múltiples sensores, de los cuales son de gran utilidad los sensores infrarrojos para medir la distancia del *e-puck* a los objetos cercanos. Además se dispone directamente de las señales de activación de los motores las cuales se pueden utilizar en lugar de los encoders (físicamente no disponibles en el *e-puck*). El cerebro del *e-puck* es un procesador dsPIC el cual funciona como un micro-controlador. Es en donde se programa la rutina de control del robot y se almacena el firmware del mismo. Es fabricado por la compañía *Microchip* y permite un procesamiento eficiente de señales. En la figura 2.2 se pueden observar la interconexión de los componentes electrónicos del *e-puck*.

Para programar el dsPIC del *e-puck* se cuenta con un compilador, el *MPLABC30*. Este realiza la compilación los programas del *e-puck* escritos en código *C*. Además se tienen múltiples librerías para el manejo de los sensores y actuadores del robot así como para el modulo bluetooth

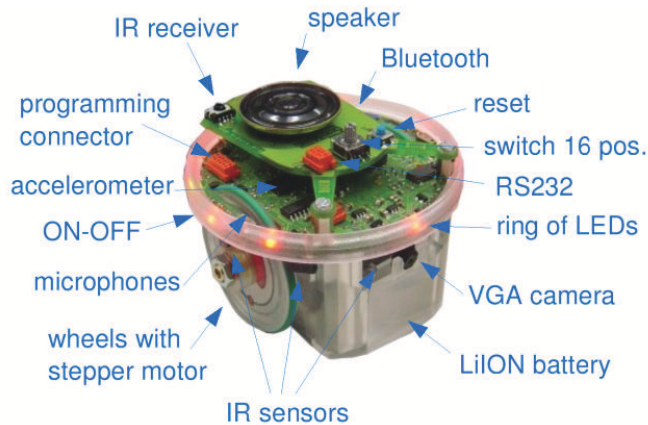


Figura 2.1: Sensores principales del robot *e-puck* [51].

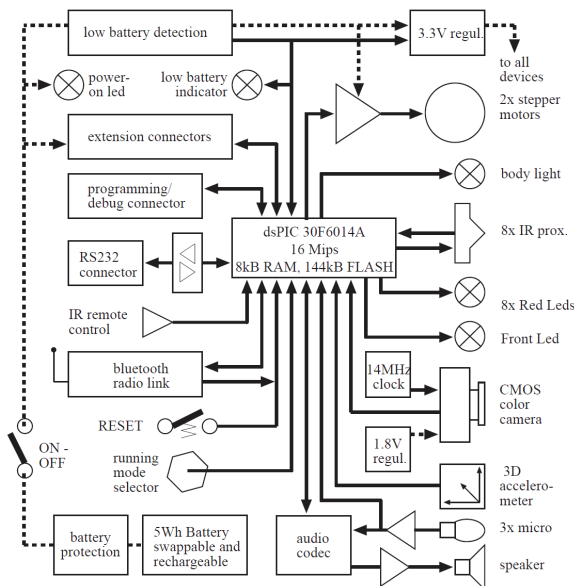


Figura 2.2: Interconexión de componentes en el robot *e-puck* [52].

que permite comunicar el *e-puck* con el ordenador para programar el dsPIC o bien registrar la evolución de las variables de interés (por ejemplo la posición), este es el principal medio para supervisar lo que sucede dentro del robot, por lo que se diseñó un programa que se ejecuta dentro del robot para enviar la información de la posición del *e-puck* al ordenador. De esta forma se consigue obtener la estimación de la odometría dentro del robot para así compararla con una medición global (por ejemplo una cámara cenital). Debido a la limitación sensorial del robot y a su baja capacidad computacional, únicamente se podrá implementar en este el filtro de Kalman más sencillo, sin embargo, este produce buenos resultados tal y como se muestra en el capítulo siguiente.

2.2. Robot móvil LEGO NTX

LEGO Mindstorms [®]NXT es una plataforma de robot móvil de bajo coste. Resulta de la evolución de la primera versión desarrollada en 1998 en colaboración con LEGO y el MIT que incorpora la nueva unidad de control NXT, basada en un microcontrolador ARM7 de 32-bits, con 256 Kbytes FLASH y 64 Kbytes de RAM. Cuenta además con un puerto USB 2.0 y un dispositivo inalámbrico Bluetooth clase II, V2.0. La nueva unidad de control tiene 4 entradas (una de ellas proporciona una expansión IEC 61158 Tipo 4/EN 50 170 para usos futuros) y 3 salidas analógicas. La nueva versión de LEGO proporciona 4 tipos de sensores: contacto, luz, sonido y distancia. Pero, además de estos sensores propios de LEGO, en la actualidad se pueden comprar una gran variedad de distintos tipos de sensores como, por ejemplo, cámaras para aplicaciones de visión, brújula, acelerómetros, giróscopos, buscadores de infrarrojos, etc. [2]. Otros componentes LEGO muy importantes son los actuadores, los cuales consisten en motores de corriente continua que incorporan encoders integrados y de buena resolución. Se puede encontrar una descripción más detallada de los motores de LEGO Mindstorms NXT en [2]. Existen múltiples opciones para la programación del robot y distintos lenguajes, de los cuales se escoge el *RobotC* debido a que permite un control de bajo nivel sobre los motores, permitiendo implementar un control de bajo nivel diseñado a partir del modelo obtenido en 4.7, pudiendo ser por ejemplo un control PID o un control bifrecuencia. El robot LEGO NXT usado en las pruebas se muestra en la figura 2.3. Los sensores usados para el esquema de fusión son dos acelerómetros puestos encima de cada rueda para el modelo dinámico ((4.22) y (4.23)), un giróscopo, un compás magnético y los encoders de las ruedas.

2.3. Sensor Inercial IG500N abordo de un coche

El sensor IG500N [1] es un sensor desarrollado por SBG Systems para la medición de las variables asociadas al movimiento de vehículos en el espacio. Consta de dos componentes principales, la unidad inercial (INS) que consta de un magnetómetro 3D (sensor de campo magnético), un giroscopio 3D, un acelerómetro 3D y un termómetro para la calibración de la INS. Además consta con un receptor GPS (con antena externa) y un altímetro barométrico. Además cuenta con un filtro de Kalman implementado en el sensor que permite estimar la posición del sensor usando los datos del GPS y de la INS. Su esquema de funcionamiento se muestra en 2.4. Si bien es cierto este sensor no es un robot en sí mismo, se utilizará en un automóvil con el fin de mos-

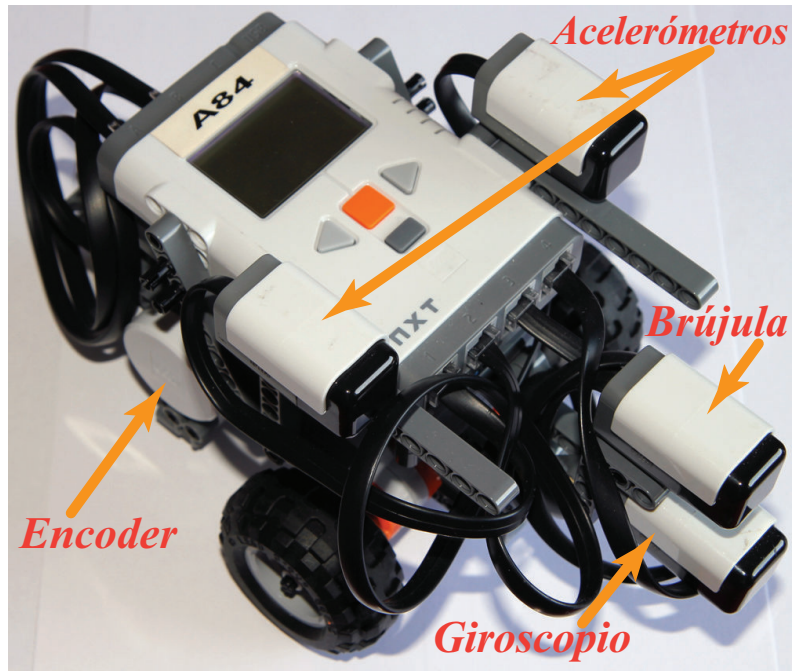


Figura 2.3: Robot móvil LEGO NXT utilizado

trar la aplicación del filtro del Kalman desarrollado al utilizar sensores únicamente inerciales sin mediciones precisas de velocidad proveniente de los encoders pero utilizando información global del GPS de estas velocidades. Este sensor puede utilizarse para incorporar medición local y global a un robot de construcción propia o adaptarlo a un robot comercial comunicando su interfaz (RS-232 y USB) con la del robot. Vistas las plataformas a utilizar se procede a exponer un resumen de la teoría del KF junto con su representación algorítmica que será implementada en un robot móvil.

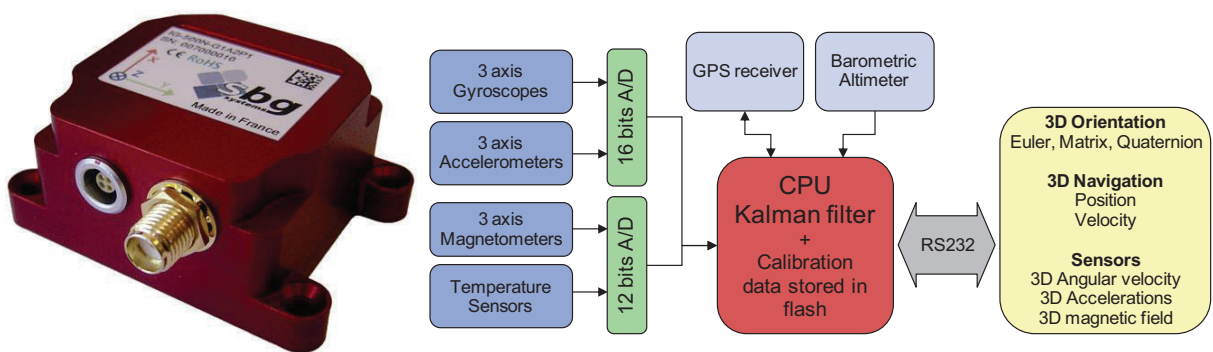


Figura 2.4: Sensor inercial IG500N

Capítulo 3

Teoría del filtro de Kalman

Como es bien sabido, el KF proporciona una herramienta computacional para estimar el estado de un proceso, de una forma que minimiza la media cuadrática del error del estado estimado. Hay distintas variaciones del filtro y desde su origen ha sido estudiado ampliamente y se ha modificado para mejorar su desempeño, generando nuevas versiones para distintas aplicaciones prácticas (para una amplia visión global y últimos avances consultar [12], [47], [20] y [53]). A pesar de la gran variedad de filtros con los que se cuentan hoy en día, todos basan su funcionamiento en el algoritmo fundamental propuesto originalmente por Kalman [41]. Además debido a la complejidad de ciertos filtros, es conveniente centrar el estudio en aquellos que pueden ser implementados dentro del robot. Este es el caso del algoritmo original (filtro de Kalman KF) y del algoritmo extendido (EKF) para sistemas no lineales, los cuales son bastante sencillos y robustos si se diseñan adecuadamente. Se expone además un filtro avanzado basado en la transformada Unscented para sistemas no lineales que servirá de medida del desempeño de los filtros KF y EKF desarrollados ya que debido a su complejidad no se implementará dentro del robot.

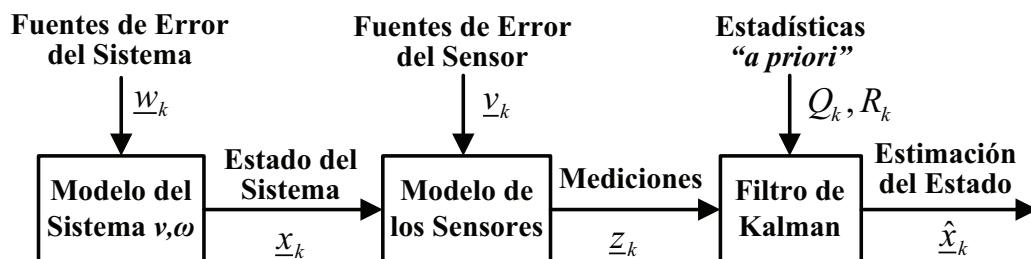


Figura 3.1: Forma de operar del filtro de Kalman. [26]

3.1. El filtro de Kalman

Esta es la formulación estándar (forma original expuesta por [41]) para un sistema lineal. Se resume brevemente su formulación matemática y su presentación en forma de algoritmo ([70] y [26]) el cual se utilizará para implementar la fusión sensorial dentro del robot. Consideremos el sistema lineal, invariante con el tiempo, discreto con ruido mostrado en la ecuación (3.1).

$$\begin{aligned}x_k &= Ax_{k-1} + Bu_{k-1} + w_{k-1} \\z_k &= Hx_k + v_k\end{aligned}\tag{3.1}$$

Donde $x_k \in \mathfrak{R}^n$ es el vector de estado, $u_k \in \mathfrak{R}^u$ es el vector de entrada y $z_k \in \mathfrak{R}^m$ es el vector de medidas, todos con una distribución de probabilidad independiente, blanca y normal con media cero (gaussiana). Los términos w_k y v_k representan ruidos de procesos y medidas en el instante k (no es necesario que estos ruidos tengan distribución gaussiana para que el filtro KF funcione de forma óptima [62]). Los valores exactos de w_k y v_k en el instante k son normalmente desconocidos pero el desarrollo del KF se realiza asumiendo que se posee cierto conocimiento sobre sus matrices de covarianza, Q_k y R_k (respectivamente), formalmente estas son variantes en el tiempo pero para este trabajo se utilizarán invariantes en el tiempo. Puede ser que se conozcan sus valores siendo este el caso menos común para Q aunque para R se pueden obtener de forma experimental algunos de sus valores. Si este no es el caso se deberán ajustar los valores de estas matrices como parámetro de diseño del filtro (caso más general, la forma de ajustar estos parámetros se explica en el capítulo 5).

El objetivo del KF es estimar el estado x_k de (3.1) basado en el conocimiento de las dinámicas del sistema (modelo lineal), las características de la perturbación (matrices de covarianza) y la disponibilidad de medidas ruidosas z_k . Este filtro minimizará los efectos de w_k y v_k en la estimación del estado. La forma de trabajar del KF se muestra en la figura 3.1. Definiendo $\hat{x}_k^- \in \mathfrak{R}^n$ como el estado estimado *a priori* en el instante k (el cual se obtiene del conocimiento del proceso antes del instante de tiempo k) y $\hat{x}_k \in \mathfrak{R}^n$ como la estimación del estado *a posteriori* (en el instante k usando la medida z_k). Los errores estimados *a priori* y *a posteriori* se definen en (3.2).

$$e_k^- = x_k - \hat{x}_k^- \quad e_k = x_k - \hat{x}_k\tag{3.2}$$

La estimación *a priori* y *a posteriori* de la covarianza del error se define en (3.3) (donde E representa el valor esperado).

$$P_k^- = E [e_k^- e_k^{-T}] \quad P_k = E [e_k e_k^T]\tag{3.3}$$

Para obtener las ecuaciones del filtro de Kalman se inicia definiendo la ecuación (3.4) que calcula la estimación del estado *a posteriori* $\hat{x}_k \in \mathfrak{R}^n$ como una combinación lineal de la

estimación *a priori* \hat{x}_k^- y el error estimado, el cual es la diferencia entre la medida actual z_k y la predicción de la medida $H\hat{x}_k^-$.

$$\hat{x}_k = \hat{x}_k^- + K (z_k - H \cdot \hat{x}_k^-) \quad (3.4)$$

El término $z_k - H \cdot \hat{x}_k^-$ se denomina *innovación* o *residuo* el cual refleja la discrepancia entre las mediciones y las predicciones para actualizar el estado estimado *a priori*. Aquí la matriz de ganancias K se elige para minimizar la covarianza del error *a posteriori* P_k y se determina por una aproximación probabilística usando las covarianzas de error y ruido. Únicamente para entender mejor la aplicación del KF, se esquematiza el guión para la derivación de la ganancia de Kalman (un desarrollo completo se encuentra por ejemplo en [62] y [11]). La minimización se realiza sustituyendo (3.4) en e_k (3.2) y entonces el resultado se sustituye en P_k (3.3). Tal como se ha mencionado anteriormente, minimizando (3.3) con respecto a K (derivando respecto a K e igualando a cero para despejar K) se obtiene la ganancia óptima del filtro K tal y como se muestra en (3.5).

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1} \quad (3.5)$$

Esta ecuación (3.5) necesita el valor de la covarianza *a priori* P_k^- . Este valor se obtiene a partir de la definición de (3.3), el cual requiere del conocimiento de la variable del sistema x_k . Mediante manipulaciones simples, se obtiene la covarianza en (3.6).

$$P_k^- = A P_{k-1} A^T + Q \quad (3.6)$$

El valor de la covarianza *a posteriori*, P_k se obtiene también a partir de su definición en (3.3) como se muestra en (3.7).

$$P_k = (I - K_k H) P_k^- \quad (3.7)$$

Con las ecuaciones (3.4) a (3.7) el KF se implementa en forma de algoritmo (figura 3.2). Este se puede dividir en dos estados [70]:

- Actualización del tiempo (o Predictor): Calcula la siguiente estimación del estado *a priori* utilizando la estimación previa del estado y el valor actual de la entrada (requiere normalmente una suposición inicial al empezar los cálculos); esto se usa para calcular la covarianza *a priori*.
- Actualización de medidas (o Corrector): Utiliza la medida actual para refinar el resultado dado por la etapa anterior para obtener una estimación *a posteriori* mejorada.

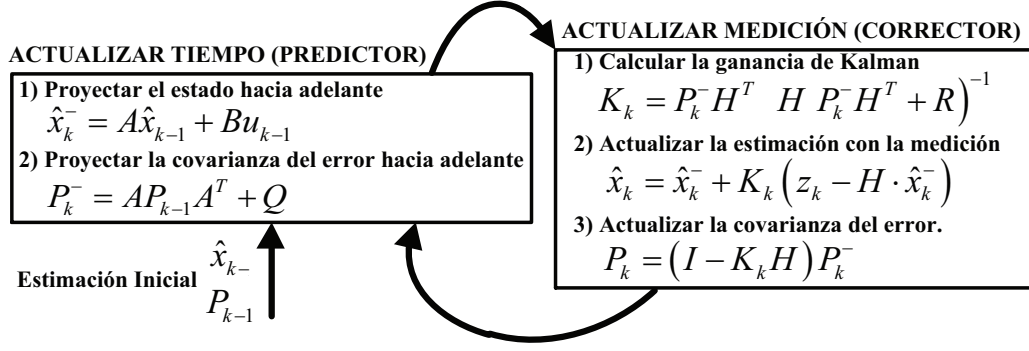


Figura 3.2: Algoritmo del filtro de Kalman [70].

3.2. Filtro de Kalman extendido

Este filtro extiende la formulación estándar para sistemas no lineales ([70] y [26]). Considerando el sistema no lineal, invariante con el tiempo, discreto con ruido mostrado en (3.8).

$$x_k = f(x_{k-1}, u_{k-1}, w_{k-1}), \quad z_k = h(x_k, v_k) \quad (3.8)$$

Donde w_k y v_k son desconocidos. La función no lineal f relaciona el estado en el instante anterior $k - 1$ con el estado actual en el instante k . La función no lineal h relaciona el estado x_k con las mediciones z_k . El estado y las mediciones en el instante inicial pueden ser aproximados considerando la respuesta del sistema ante la entrada anterior sin la presencia de ruido tal y como se muestra en (3.9).

$$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0), \quad \hat{z}_k = h(\hat{x}_k^-, 0) \quad (3.9)$$

Siendo \hat{x}_k la estimación del estado *a posteriori* en el instante de tiempo anterior $k - 1$. Para poder utilizar las ecuaciones del filtro ya desarrolladas para el KF lineal, se linealiza (3.8) alrededor del punto de funcionamiento obteniendo (3.10).

$$\begin{aligned} x_k &\approx \hat{x}_k^- + A(x_{k-1} - \hat{x}_{k-1}) + Ww_{k-1} \\ z_k &\approx \hat{z}_k^- + H(x_k - \hat{x}_k^-) + Vv_k \end{aligned} \quad (3.10)$$

Donde \hat{x}_k^-, \hat{z}_k^- son las estimaciones a partir de (3.9). Las matrices del sistema de la ecuación (3.10) se calculan en cada instante k utilizando (3.11)

$$A_k = \frac{\partial f}{\partial x} |_{\hat{x}_{k-1}, u_{k-1}, 0} \quad W_k = \frac{\partial f}{\partial w} |_{\hat{x}_{k-1}, u_{k-1}, 0} \quad (3.11)$$

$$H_k = \frac{\partial h}{\partial x} |_{\hat{x}_k^-, 0} \quad V_k = \frac{\partial h}{\partial v} |_{\hat{x}_k^-, 0}$$

La ganancia del EKF se calcula de manera similar al caso del KF (ver desarrollo en [62] y [11]), y se puede escribir también de forma algorítmica tal como se muestra en la figura 3.3.

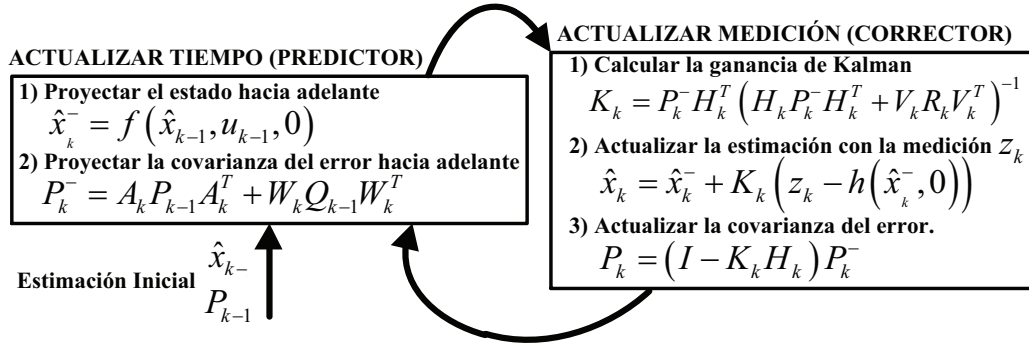


Figura 3.3: Algoritmo del filtro de Kalman Extendido [70].

3.3. Filtro de Kalman Unscented

Este filtro es una formulación para sistemas no lineales presentada originalmente por [40] sin usar el paso de linealización del EKF para el modelo (3.8). Mantiene la estructura de predicción-corrección del filtro de Kalman (figuras 3.2 y 3.3) pero usando un esquema de predicción diferente. En lugar de estimar la covarianza del error P_k^- utilizando (3.10) emplea la transformación Unscented (UT) [40] que calcula un conjunto de puntos muestra (*Puntos Sigma*) para representar de forma más precisa la media y la varianza del estado (suponiendo que tiene una distribución Gaussiana). Propagando estos puntos a través del sistema no-lineal (3.8) la media y covarianza posterior se obtiene para cualquier no-linealidad presente en el sistema de forma precisa ([39], [36], [37], [40], [38] y [68]), dando un mejor resultado al compararlo con la aproximación del EKF tal y como se muestra en la figura 3.4. Estos valores se utilizan para obtener la predicción del estado y de su covarianza en una estimación *a priori*. A pesar de que el UKF es muy preciso, es muy costoso en términos de capacidad de cálculo, principalmente porque este filtro requiere el cálculo de la raíz cuadrada de la matriz P_k^- . Esto excluye a este filtro de ser implementado en un robot con recursos limitados en el cual el hecho de calcular esta operación puede llevar mucho tiempo, además de que en general no existe soporte de cálculo matricial dentro del robot por lo que también se tendría que implementar. Debido a esto, el UKF se usará como medida de desempeño para comparar los resultados obtenidos a partir del KF y del EKF en la próxima sección usando los algoritmos UKF implementados en [30].

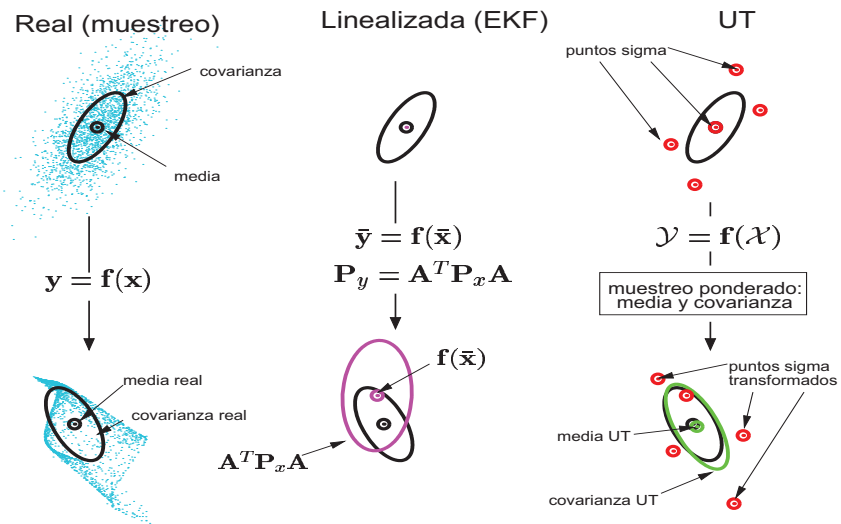


Figura 3.4: Propagación de la media y la covarianza en la Transformada Unscented [68].

Capítulo 4

Modelos de un robot en configuración diferencial

Un robot en configuración diferencial consiste en un cuerpo rígido de masa M_G y un momento de inercia I_G con dos ruedas no deformables y no orientables (fijas) separadas una distancia b y que son movidas por dos motores que aplican dos fuerzas lineales F_R y F_L tal como se muestra en las figuras 4.1 y 4.3. Las ruedas son convencionales y se asumirá que satisfacen la condición de rodamiento puro sin deslizamiento [13]. El robot también tiene la restricción de moverse únicamente en un plano horizontal y su centro de masa, denominado P_0 , corresponde también al centro de gravedad y al eje de rotación. Se analiza la cinemática y dinámica de este robot para obtener el modelo del robot que se utilizará en los filtros de Kalman. Además se muestra el modelado de los motores de robot LEGO NXT en el cual se probarán la mayoría de filtros desarrollados.

4.1. Modelo cinemático

Este modelo representa la evolución de las velocidades del robot en un marco inercial fijo. La *posición* del robot se define mediante las coordenadas del punto $P_0 = (x, y)$ y el ángulo de avance θ (curso del robot o guiñada) en el marco de referencia global (figura 4.1). Conociendo las velocidades lineales y angulares locales (v y ω) la velocidad global del robot se define mediante (4.1).

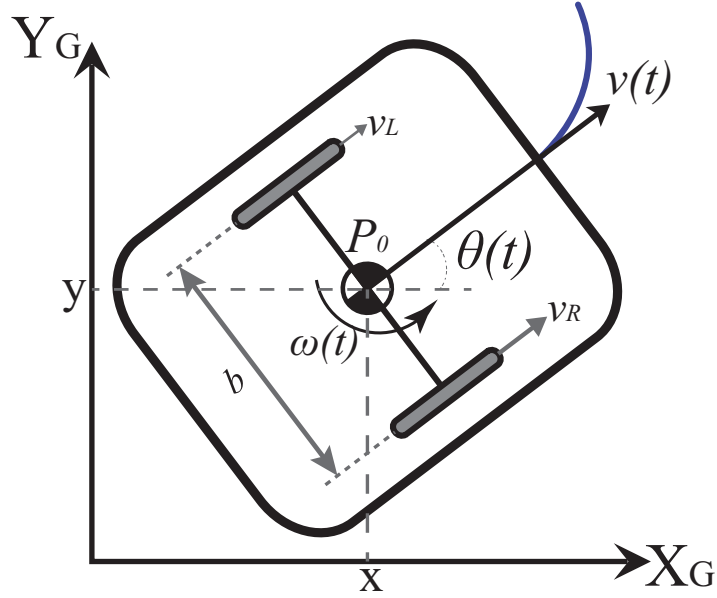


Figura 4.1: Cinemática de un robot en configuración diferencial

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \Rightarrow \begin{aligned} \dot{x}(t) &= v(t) \cos(\theta(t)) \\ \dot{y}(t) &= v(t) \sin(\theta(t)) \\ \dot{\theta}(t) &= \omega(t) \end{aligned} \quad (4.1)$$

Existen en la literatura muchas formas de resolver esta ecuación diferencial, ya sea por integración directa o por aproximaciones numéricas (ver por ejemplo los modelos utilizados en [67], [23] y [5]). Sin embargo es conveniente realizar la demostración de la integral con el fin de demostrar cual es el resultado exacto y sobre este evaluar el uso de alguna aproximación que se considere conveniente para simplificar el cálculo en la ejecución del algoritmo sobre el robot. La solución exacta de (4.1) se calcula de forma discreta para un periodo de muestreo $T_s = t_{k+1} - t_k$ de forma que se obtenga una ecuación recursiva que pueda implementarse dentro del robot. De esta forma se define el intervalo de tiempo entre 2 instantes consecutivos de muestreo mostrado en (4.2).

$$T_K = \{t \in \mathbb{R} \mid kT_s \leq t < (k+1)T_s\}, \quad k = 0, 1, 2, \dots \quad (4.2)$$

Para simplificar la deducción de las ecuaciones se define $t_k = kT_s$, $t_{k+1} = (k+1)T_s$. Se considera que las velocidades (v, ω) son constantes en el intervalo de integración (no cambian mientras se realiza el muestreo en T_s , lo que es equivalente a decir que se tiene un retenedor de orden cero (ZOH) en estas entradas, adquiriendo el valor en $t_k \Rightarrow v(t_k), \omega(t_k)$ son constantes.

De esta forma se plantea la integral en (4.3).

$$\begin{aligned}x(t) &= x(t_k) + v(t_k) \int_{t_k}^t \cos(\theta(\tau)) d\tau \\y(t) &= y(t_k) + v(t_k) \int_{t_k}^t \sin(\theta(\tau)) d\tau \\ \theta(t) &= \theta(t_k) + \int_{t_k}^t \omega(t_k) d\tau\end{aligned}\tag{4.3}$$

Se resuelve primeramente para el ángulo de avance del robot θ tal y como se muestra en (4.4)

$$\theta(t) = \theta(t_k) + \int_{t_k}^t \omega(t_k) d\tau = \theta(t_k) + \omega(t_k) \int_{t_k}^t d\tau = \theta(t_k) + (t - t_k) \omega(t_k)\tag{4.4}$$

Se sustituye este resultado en las ecuación (4.3) para (x, y) . Cabe destacar que las funciones evaluadas en t_k son constantes $(x(t_k), y(t_k), \theta(t_k))$ ya que son las condiciones iniciales (constantes de integración), de esta forma se procede a integrar (4.3) dando el resultado en (4.5) para x y en (4.6) para y .

$$\begin{aligned}x(t) &= x(t_k) + v(t_k) \int_{t_k}^t \cos(\theta(t_k) + (\tau - t_k) \omega(t_k)) d\tau = x(t_k) + \frac{v(t_k)}{\omega(t_k)} [\sin(\theta(t_k) + (\tau - t_k) \omega(t_k))]_{t_k}^t \\ \Rightarrow x(t) &= x(t_k) + \frac{v(t_k)}{\omega(t_k)} [\sin(\theta(t_k) + (t - t_k) \omega(t_k)) - \sin(\theta(t_k) + (t_k - t_k) \omega(t_k))] \\ \therefore x(t) &= x(t_k) + \frac{v(t_k)}{\omega(t_k)} [-\sin(\theta(t_k)) + \sin(\theta(t_k) + (t - t_k) \omega(t_k))]\end{aligned}\tag{4.5}$$

$$\begin{aligned}y(t) &= y(t_k) + v(t_k) \int_{t_k}^t \sin(\theta(t_k) + (\tau - t_k) \omega(t_k)) d\tau = y(t_k) - \frac{v(t_k)}{\omega(t_k)} [\cos(\theta(t_k) + (\tau - t_k) \omega(t_k))]_{t_k}^t \\ \Rightarrow y(t) &= y(t_k) - \frac{v(t_k)}{\omega(t_k)} [\cos(\theta(t_k) + (t - t_k) \omega(t_k)) - \cos(\theta(t_k) + (t_k - t_k) \omega(t_k))] \\ \therefore y(t) &= y(t_k) + \frac{v(t_k)}{\omega(t_k)} [\cos(\theta(t_k)) - \cos(\theta(t_k) + (t - t_k) \omega(t_k))]\end{aligned}\tag{4.6}$$

Ambas expresiones (4.5) y (4.6) se pueden simplificar aplicando identidades trigonométricas¹, tal y como se muestra en (4.7):

$$\begin{aligned}\left. \begin{aligned}a &= \theta(t_k) \\ b &= (t - t_k) \omega(t_k)\end{aligned} \right\} \Rightarrow \begin{aligned}-\sin(\theta(t_k)) + \sin(\theta(t_k) + (t - t_k) \omega(t_k)) &= \sin(a + b) - \sin(a) \\ \cos(\theta(t_k)) - \cos(\theta(t_k) + (t - t_k) \omega(t_k)) &= \cos(a) - \cos(a + b)\end{aligned} \\ \sin(a + b) - \sin(a) &= 2 \sin(0,5(a + b - a)) \cos(0,5(a + b + a)) = 2 \sin(0,5b) \cos(0,5(2a + b)) \\ \cos(a) - \cos(a + b) &= -2 \sin(0,5(a - a - b)) \sin(0,5(a + b + a)) = -2 \sin(-0,5b) \sin(0,5(2a + b))\end{aligned}\tag{4.7}$$

¹ $\sin u - \sin v = 2 \cos(0,5u + 0,5v) \sin(0,5u - 0,5v)$,
 $\cos u - \cos v = -2 \sin(0,5u + 0,5v) \sin(0,5u - 0,5v)$, $\sin(-u) = -\sin(u)$

Al aplicar esta simplificación (4.7) a (4.5) y (4.6) se obtiene (4.8).

$$\begin{aligned}
\Rightarrow x(t) &= x(t_k) + 2 \frac{v(t_k)}{\omega(t_k)} \sin(0,5(t-t_k)\omega(t_k)) \cos(\theta(t_k) + 0,5(t-t_k)\omega(t_k)) \\
\Rightarrow y(t) &= y(t_k) + 2 \frac{v(t_k)}{\omega(t_k)} \sin(0,5(t-t_k)\omega(t_k)) \sin(\theta(t_k) + 0,5(t-t_k)\omega(t_k)) \\
\theta(t) &= \theta(t_k) + (t-t_k)\omega(t_k)
\end{aligned} \tag{4.8}$$

Este es el resultado completo de la integración para cualquier $t \in T_K$. Al evaluar (4.8) en $t = t_{k+1}$ se obtiene el modelo discreto en el periodo de muestreo $T_s = t_{k+1} - t_k$ tal y como se muestra en (4.9).

$$\begin{aligned}
\Rightarrow x(t_{k+1}) &= x(t_k) + 2 \frac{v(t_k)}{\omega(t_k)} \sin(0,5(t_{k+1}-t_k)\omega(t_k)) \cos(\theta(t_k) + 0,5(t_{k+1}-t_k)\omega(t_k)) \\
\Rightarrow y(t_{k+1}) &= y(t_k) + 2 \frac{v(t_k)}{\omega(t_k)} \sin(0,5(t_{k+1}-t_k)\omega(t_k)) \sin(\theta(t_k) + 0,5(t_{k+1}-t_k)\omega(t_k)) \\
\Rightarrow \theta(t_{k+1}) &= \theta(t_k) + (t_{k+1}-t_k)\omega(t_k)
\end{aligned} \tag{4.9}$$

Finalmente, sustituyendo el periodo de muestreo $T_s = t_{k+1} - t_k$ en (4.9) y simplificando la notación al eliminar la indicación del tiempo t al agregar un subíndice a las variables (por ejemplo cambiando $x(t_k) = x_k$) se obtiene el modelo cinemático del robot (la ecuación de la odometría exacta) en (4.10). Esta ecuación funciona como una *TRANSFORMACIÓN DE COORDENADAS* ya que relaciona el movimiento local del robot (v, ω) con su movimiento global (x, y, θ) (se muestra además en la notación correspondiente al KF).

$$\begin{aligned}
\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} &= \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} \frac{2v_k}{\omega_k} \sin(0,5T_s\omega_k) \cos(\theta_k + 0,5T_s\omega_k) \\ \frac{2v_k}{\omega_k} \sin(0,5T_s\omega_k) \sin(\theta_k + 0,5T_s\omega_k) \\ T_s\omega_k \end{bmatrix} = \\
\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} &= \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} \frac{2v_{k-1}}{\omega_{k-1}} \sin(0,5T_s\omega_{k-1}) \cos(\theta_{k-1} + 0,5T_s\omega_{k-1}) \\ \frac{2v_{k-1}}{\omega_{k-1}} \sin(0,5T_s\omega_{k-1}) \sin(\theta_{k-1} + 0,5T_s\omega_{k-1}) \\ T_s\omega_{k-1} \end{bmatrix}
\end{aligned} \tag{4.10}$$

En el caso de las plataformas propuestas se tiene un T_s suficientemente pequeño (50ms) para la implementación del control con fusión sensorial. En este caso se puede realizar una aproximación en la integral de la ecuación diferencial (4.10) considerando que $\sin(\omega_k \cdot 0,5T_s) \approx 0,5T_s\omega_k$ tal y como se muestra en (4.11).

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + T_s \begin{bmatrix} v_{k-1} \cos(\theta_{k-1} + \omega_{k-1} \cdot 0,5T_s) \\ v_{k-1} \sin(\theta_{k-1} + \omega_{k-1} \cdot 0,5T_s) \\ \omega_{k-1} \end{bmatrix} \tag{4.11}$$

Si además se considera que el robot gira a una velocidad tal que en un tiempo T_s el incremento de θ_k es muy pequeño entonces la ecuación (4.11) se puede simplificar aún más ya que en estas

condiciones $\theta_k + \omega_k \cdot 0,5T_s \approx \theta_k$, esto se muestra en (4.12).

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + T_s \begin{bmatrix} v_{k-1} \cos(\theta_{k-1}) \\ v_{k-1} \sin(\theta_{k-1}) \\ \omega_{k-1} \end{bmatrix} \quad (4.12)$$

Se observa que esta ecuación es la misma que la (1.1) en la introducción ([51] y [61]). Ambas ecuaciones (4.11) y (4.12) son ampliamente usadas en la literatura sin discriminar las condiciones necesarias para que estas estimen de forma adecuada el valor de la posición del robot ni los requerimientos del robot para poder resolver a tiempo esta ecuación durante el ciclo de control. En el presente trabajo se sigue el siguiente criterio, si el robot es de recursos muy limitados (por ejemplo el *e-puck*) se utilizará la ecuación (4.12) (aproximación simple y aproximada pero rápida), en cambio si se tiene un poco más de capacidad computacional (por ejemplo el *LEGO*) se utilizará la ecuación (4.11), y si se está trabajando en un ordenador para controlar el robot o en las simulaciones, se utilizará el valor exacto dado por (4.10). Con estos modelos cinemáticos, los cuales relacionan velocidades únicamente, se han implementado la mayoría de los trabajos planteados en los antecedentes. Si bien es cierto que estos producen buenos resultados en las aplicaciones, no se está tomando en cuenta la dinámica del robot. Esta puede mejorar el desempeño del control si es tomada en consideración por el filtro de Kalman, ya que producirá una estimación más cercana a la real mejorando el funcionamiento del control del robot.

Para finalizar el modelado cinemático, existe una relación entre v y ω y las velocidades de las ruedas (obtenidas a partir del cambio en el desplazamiento de los encoders) tal como se muestra en (4.13). Derivando esta ecuación, se obtiene una relación cinemática entre las aceleraciones lineales y angulares a y α del robot y las aceleraciones de las ruedas izquierda a_L y derecha a_R . Usando dos acelerómetros puestos encima de cada rueda, o un modelo de la aceleración de cada rueda, esta relación dará las aceleraciones del robot que se pueden integrar para obtener las velocidades y usarlas como entrada en (4.10) para obtener la posición global del robot. Este procedimiento se puede mejorar usando la dinámica del robot para obtener las aceleraciones en lugar de usar la cinemática tal y como se expresa en la siguiente sección.

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ -1/b & 1/b \end{bmatrix} \begin{bmatrix} v_L \\ v_R \end{bmatrix} \Rightarrow \begin{bmatrix} a \\ \alpha \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ -1/b & 1/b \end{bmatrix} \begin{bmatrix} \dot{v}_L \\ \dot{v}_R \end{bmatrix} \quad (4.13)$$

4.2. Modelo dinámico

Este modelo representa la evolución de las aceleraciones del robot a y α en términos de las fuerzas que le han sido aplicadas por sus motores. Existen gran cantidad de desarrollos en la

literatura para obtener el modelo dinámico del robot de configuración diferencial, siendo el procedimiento general aplicar la segunda Ley de Newton para obtener a y α en el centro de masas del robot de forma local y utilizarla junto con el modelo cinemático para obtener la aceleración del robot en los ejes globales (ver por ejemplo [14], [22], [28] [69] y [4]). Otro método es utilizar directamente la formulación de Lagrange para obtener el modelo directamente en las coordenadas globales de avance (ver por ejemplo [60], [25], [57] y [72]). Sin embargo para ambos métodos y en los ejemplos vistos en la revisión bibliográfica, se producen modelos muy complejos que tienen varias desventajas. Primero, son complejos en exceso, siendo difícil de obtener muchos de los parámetros requeridos de forma experimental, además de que, en los robots estudiados, el fabricante no indica muchos de estos. Otro aspecto a tomar en cuenta es que debido a que en general son modelos no lineales de dimensión alta, el cálculo requerido es considerable lo que dificulta su implementación práctica, principalmente porque se requerirá un filtro extendido o «unscented» para hacer la fusión sensorial utilizando este modelo, lo que los excluye de la implementación práctica (principalmente en el caso del filtro «unscented»). Debido a esto se consideran que en general, estos modelos aportan poco al esquema de fusión ya que son difíciles de implementar.

La complejidad principal de modelos de la literatura radica en que son modelos que describen el movimiento del robot en el marco de referencia *global* (ejes X_G, Y_G , figura 4.2), para lo cual transforman la aceleración *local* del robot (lineal a y angular α) mediante el cálculo del seno y coseno del ángulo de avance del robot θ . Con esto se obtiene la aceleración en los ejes globales en donde es integrada dos veces para obtener el modelo de la posición del robot (x, y, θ) . Esta transformación de coordenadas es la principal fuente de no linealidad del modelo (excluyendo los más complejos que estiman el deslizamiento en las ruedas del robot). Ante este problema se plantea la siguiente solución, utilizar el modelo dinámico *local* para realizar la fusión sensorial (filtro de Kalman) y posteriormente aplicar la transformación de coordenadas para obtener la dinámica *global*. Si bien es cierto esta solución es en aparente trivial, no lo es ya que con esto se obtiene una ventaja fundamental, el lograr simplificar el cálculo del filtro de Kalman para la fusión sensorial tal y como se verá en la sección siguiente.

A parte de este problema existe otro de índole práctico, la obtención de un modelo dinámico local no es siempre sencilla en cuanto este modelo requiere conocer la fuerza que aplican los motores para calcular la aceleración del robot (a y α). Esta dificultad no es sencilla de resolver ya que en el caso de las plataformas estudiadas el valor del par motor no se puede medir, únicamente se dispone de la información de los encoders. Lo que comúnmente se hace en este caso

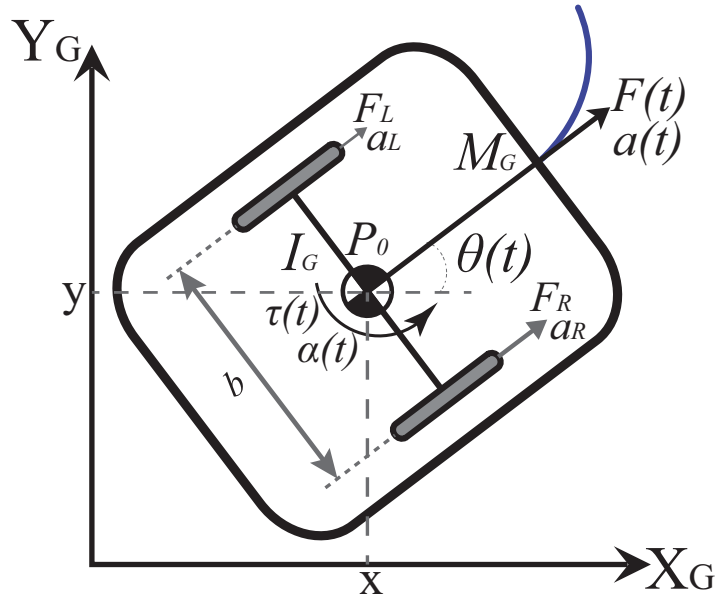


Figura 4.2: Dinámica de un robot en configuración diferencial

es obtener un modelo teórico para el par motor con el problema de que en los robots estudiados se desconocen los parámetros necesarios para este modelo (momento de inercia y coeficiente de amortiguamiento). Se pueden estimar estos parámetros pero no con la precisión adecuada (requiere modificación mecánica de las plataformas para acoplar un medidor de fuerza o par) y con el problema de que estos parámetros varían entre dos motores de un mismo fabricante.

Para solucionar este problema se obtendrá un modelo dinámico que no dependa de las fuerzas en las ruedas sino de la aceleración de las mismas. Para esto se divide el problema en dos partes, la primera es el de obtener un modelo dinámico del movimiento local del robot que dependa de la aceleración en las ruedas del motor (a_L, a_R) el cual se calcula a continuación y el segundo es el de estimar un modelo para esta aceleración basándose en la información de los encoders y la acción de control del robot el cual se muestra en la sección siguiente. Para el primer modelo se considera que al cuerpo rígido del robot de masa M_G y momento de inercia I_G se le aplican dos fuerzas lineales (F_L, F_R) generadas por las ruedas del robot (fuerza resultante del motor menos la fricción) las cuales están separadas una distancia $b = 2R_N$ entre sí tal y como se muestra en la figura 4.3. Estas fuerzas con signo variable son perpendiculares al eje de las ruedas y generan una fuerza resultante de traslación (F) y un par de rotación (τ) sobre el centro de masa considerado como P_0 . Si el centro de masa es distinto al del punto P_0 pero desplazado en la dirección de F , las fuerzas pueden trasladarse sobre sus ejes hasta coincidir con este punto (el análisis es válido para ambos casos).

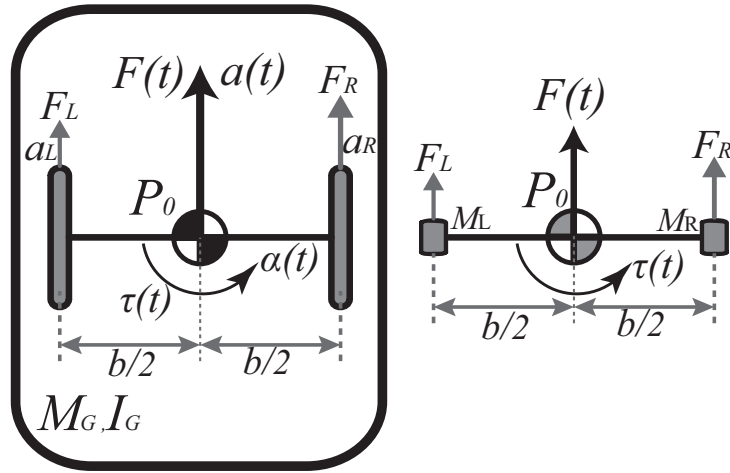


Figura 4.3: Dinámica de un robot en configuración diferencial como un cuerpo rígido y como un sistema de partículas equivalente

Al aplicar la segunda ley de Newton al cuerpo rígido se obtiene la ecuación (4.14).

$$\begin{aligned} \sum F &= F_R + F_L = M_G a \\ \sum \tau &= \tau_R - \tau_L = R_N (F_R - F_L) = I_G \alpha \end{aligned} \quad (4.14)$$

Para poder cambiar las fuerzas de las ruedas a aceleraciones en este modelo (4.14) necesitamos conocer la «masa» que mueve cada una de estas, sin embargo esta es variable ya que por ejemplo si ambas ruedas se mueven a la misma velocidad puede decirse que cada una desplaza la mitad de la masa del robot (asumiendo una densidad constante) pero esto no es así en el caso de que una rueda esté frenada y la otra esté aplicando fuerza o en el caso de fuerzas con signo distinto. Para poder realizar este cambio de fuerzas a aceleraciones se puede estudiar el cuerpo rígido del robot (masa M_G , momento de inercia I_G) como un sistema de partículas dinámicamente equivalente formado por dos partículas de masa M_L y M_R unidas por una barra (conector) sin masa de longitud constante b como se muestra en la figura 4.3. Para hacer esto se deben cumplir las siguientes condiciones (ver [7] para una amplia demostración y la extensión al caso 3D):

- Conservación de la masa: *La suma de las masas de las partículas debe ser igual a la masa del cuerpo rígido:* $M_G = M_L + M_R$
- Conservación del centro de masa: *Las masas deben estar alineadas con el centro de masas y una a cada lado de este centro:* $M_L R_L = M_R R_R$
- Conservación del momento de inercia: *La suma de los momentos de inercia de las partículas debe ser igual al momento de inercia del cuerpo rígido:* $M_L R_L^2 + M_R R_R^2 = I_G$

Con estas condiciones se despejan las masas y radios del sistema equivalente tal y como se muestra en la ecuación (4.15).

$$\begin{aligned}
M_G &= M_L + M_R \Rightarrow M_R = M_G - M_L \\
M_L R_L &= M_R R_R \Rightarrow M_R = \frac{M_L R_L}{R_R} \\
M_L R_L^2 + M_R R_R^2 &= I_G \Rightarrow M_L R_L^2 + \frac{M_L R_L}{R_R} R_R^2 = I_G \Rightarrow M_L R_L^2 + M_L R_L R_R = I_G \\
\therefore M_L &= \frac{I_G}{R_L (R_L + R_R)}
\end{aligned} \tag{4.15}$$

Se pueden definir las masas que se desean o bien las distancias deseadas entre las partículas. En este caso se asigna $R_L = R_R = R_N = b/2$ para que las distancias entre las masas sean iguales entre sí. De esta forma se reduce (4.15) a (4.16).

$$M_L = \frac{I_G}{2R_N^2}, M_R = M_G - M_L \tag{4.16}$$

Según la forma geométrica del robot se sustituyen el valor correspondiente al momento de inercia del robot I_G y el valor de la distancia entre el centro de masas y las ruedas R_N . Por facilidad de notación esta relación se expresa en función de constantes y se define un factor λ tal y como se define en la ecuación (4.17).

$$\begin{aligned}
M_L &= \frac{I_G}{2R_N^2} = c_L M_G \\
M_R &= M_G - M_L = c_R M_G \\
\lambda &= \frac{M_G R_N}{I_G}
\end{aligned} \tag{4.17}$$

De esta forma las masas M_L y M_R son proporcionales a la masa total M_G expresado mediante las constantes c_L y c_R , además la constante λ se define como la razón entre la masa y el momento de inercia del cuerpo rígido multiplicado por la distancia entre el centro de masas y la partícula. Los parámetros expresados en (4.17) se obtienen para diferentes formas de robots. Para un anillo delgado en donde $I_G = M_G R^2$ se asigna $R_N = R$ y muestra la deducción de sus parámetros en la ecuación (4.18).

$$\begin{aligned}
M_L &= \frac{M_G R^2}{2R^2} = c_L M_G \Rightarrow c_L = 0,5 \\
M_R &= M_G - M_L = c_R M_G \Rightarrow c_R = 0,5 \\
\lambda &= \frac{M_G R_N}{I_G} = \frac{M_G R}{M_G R^2} = \frac{1}{R}
\end{aligned} \tag{4.18}$$

Para un cilindro en donde $I_G = 0,5 M_G R^2$ se asigna $R_N = R$ y muestra la deducción de sus

parámetros en la ecuación (4.19).


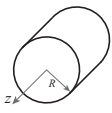
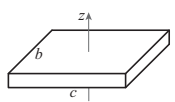
$$\begin{aligned}
 M_L &= \frac{0,5M_G R^2}{2R^2} = c_L M_G \Rightarrow c_L = 0,25 \\
 M_R &= M_G - M_L = c_R M_G \Rightarrow c_R = 0,75 \\
 \lambda &= \frac{M_G R_N}{I_G} = \frac{M_G R}{0,5M_G R^2} = \frac{2}{R}
 \end{aligned} \tag{4.19}$$

Para un prisma en donde $I_G = \frac{M_G}{12} (b^2 + c^2)$ se asigna $R_N = 0,5b$ y muestra la deducción de sus parámetros en la ecuación (4.20).

$$\begin{aligned}
 M_L &= \frac{(1/12) M_G (b^2 + c^2)}{2(0,5b)^2} = \frac{(1/12) M_G b^2 (1 + (c/b)^2)}{2(0,5b)^2} = c_L M_G \Rightarrow c_L = \frac{1 + (c/b)^2}{6} \\
 M_R &= M_G - M_L = M_G - c_L M_G = c_R M_G \Rightarrow c_R = 1 - c_L \\
 \lambda &= \frac{M_G (0,5b)}{(1/12) M_G b^2 (1 + (c/b)^2)} = \frac{(0,5)}{(1/12) b (1 + (c/b)^2)} = \frac{6}{b (1 + (c/b)^2)} = \frac{1}{bc_L}
 \end{aligned} \tag{4.20}$$

Se resumen estos resultados en el cuadro 4.1. Cabe destacar que se considera que las ruedas del robot están colocadas en el borde de la figura (de forma que la distancia entre ruedas es R para las formas de anillo y cilindro y b para la forma rectangular).

Cuadro 4.1: Parámetros del sistema de partículas dinámicamente equivalente

Forma del Robot	Momento de Inercia	c_L	c_R	λ
Anillo delgado, equivalente a (4.13) 	$I_G = M_G R^2$	0.5	0.5	$\frac{1}{R}$
Cilindro Sólido 	$\frac{1}{2} M_G R^2$	0.25	0.75	$\frac{2}{R}$
Rectángulo Sólido 	$\frac{M_G}{12} (b^2 + c^2)$	$\frac{1}{6} (1 + (c/b)^2)$	$1 - c_L$	$\frac{1}{bc_L}$

Se observa que las constantes c_L y c_R son distintas entre sí ya que se seleccionó al inicio del desarrollo que las distancias entre las ruedas del robot y el centro de masa fueran simétricas

entre sí (al deducir la ecuación (4.16)) por lo que las masas de las partículas son distintas. Esto no es ningún inconveniente ya que lo que se busca obtener al final es una ecuación entre las aceleraciones de cada rueda (partículas) y la global del robot (cuerpo rígido). Con este desarrollo se puede definir el modelo dinámico *local* del robot de la siguiente forma, se aplica inicialmente la segunda ley de Newton al sistema de partículas mostrado en la figura 4.3 para obtener (4.21).

$$\begin{aligned}\sum F &= F_R + F_L \Rightarrow M_R a_R + M_L a_L = M_G a \\ \sum \tau &= \tau_R - \tau_L \Rightarrow R_N (M_R a_R - M_L a_L) = I_G \alpha\end{aligned}\quad (4.21)$$

Se puede afirmar que el sistema dinámico del cuerpo rígido (ecuación (4.14)) y el de partículas (ecuación (4.21)) son dinámicamente equivalentes cuando el sistema de partículas se define utilizando (4.17) y los parámetros del cuadro 4.1. Sustituyendo M_L y M_R de (4.17) en (4.21) y despejando a y α se obtiene el modelo dinámico del robot tal y como se muestra en (4.22).

$$\begin{aligned}M_G a &= M_G (c_R a_R + c_L a_L) = \frac{M_G}{M_G} (c_R a_R + c_L a_L) \Rightarrow a = c_R a_R + c_L a_L \\ I_G \alpha &= M_G R_N (c_R a_R - c_L a_L) \Rightarrow \alpha = \frac{M_G R_N}{I_G} (c_R a_R - c_L a_L) = \lambda (c_R a_R - c_L a_L) \\ a &= \frac{M_G}{M_G} (c_R a_R + c_L a_L) \Rightarrow a = c_R a_R + c_L a_L \\ \alpha &= \frac{M_G R_N}{I_G} (c_R a_R - c_L a_L) \Rightarrow \alpha = \lambda (c_R a_R - c_L a_L)\end{aligned}\quad (4.22)$$

Este modelo dinámico es lineal respecto a las aceleraciones de las ruedas del robot, lo que facilita la obtención del esquema de fusión sensorial en el siguiente capítulo. Además las aceleraciones son fácilmente medibles ya sea colocando un acelerómetro sobre cada rueda o bien mediante la identificación de un modelo de forma experimental para las aceleraciones de las ruedas que tenga como entrada la acción de control del robot (esto se realiza en la sección siguiente).

Si integramos las aceleraciones de forma discreta (se integra el modelo cinemático $a = \dot{v}, \alpha = \dot{\omega}$) y sustituimos en las entradas (aceleraciones) del modelo dinámico (4.22) se obtiene el modelo *local* del robot (dinámica y cinemática) tal y como se muestra en (4.23), se observa que esta ecuación es lineal.

$$\begin{bmatrix} v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{k-1} \\ \omega_{k-1} \end{bmatrix} + \begin{bmatrix} T_s & 0 \\ 0 & T_s \end{bmatrix} \begin{bmatrix} a_{k-1} \\ \alpha_{k-1} \end{bmatrix}, \quad \begin{aligned} a_{k-1} &= c_R a_{R,k-1} + c_L a_{L,k-1} \\ \alpha_{k-1} &= \lambda (c_R a_{R,k-1} - c_L a_{L,k-1}) \end{aligned}\quad (4.23)$$

Sustituyendo (4.23) en (4.11) se obtiene el modelo global del robot (cinemática y dinámica) tal

y como se muestra en (4.24).

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \\ v_{k-1} \\ \omega_{k-1} \end{bmatrix} + \begin{bmatrix} v_{k-1}T_s \cos(\theta_{k-1} + \omega_{k-1} \cdot 0,5T_s) \\ v_{k-1}T_s \sin(\theta_{k-1} + \omega_{k-1} \cdot 0,5T_s) \\ T_s\omega_{k-1} \\ T_s a_{k-1} \\ T_s \alpha_{k-1} \end{bmatrix}, \quad \begin{matrix} a_{k-1} = c_{R}a_{R,k-1} + c_{L}a_{L,k-1} \\ \alpha_{k-1} = \lambda(c_{R}a_{R,k-1} - c_{L}a_{L,k-1}) \end{matrix} \quad (4.24)$$

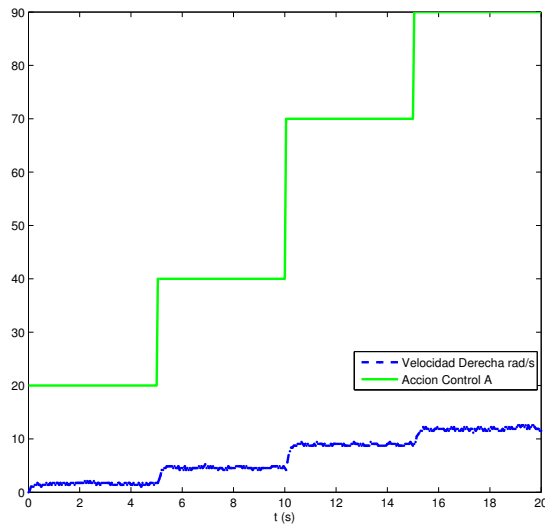
Los Modelos (4.11), (4.12), (4.23) y (4.24) se utilizan junto con el filtro de Kalman para estimar la posición global del robot tal y como se muestra en las siguientes secciones.

4.3. Modelo dinámico de los motores

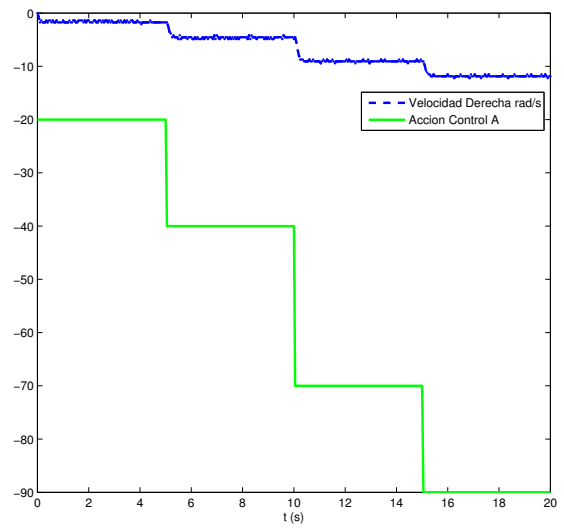
Se puede identificar un modelo para las velocidades de las ruedas del motor y derivar este modelo para obtener una aproximación a la aceleración de cada rueda. Estos modelos se puede utilizar para el diseño del control PID de bajo nivel (regulación de la velocidad de los motores) y ser utilizado como entrada al modelo dinámico local de la ecuación (4.23) o global en la ecuación (4.24).

El siguiente procedimiento se puede aplicar a cualquier robot del tipo diferencial, aunque se muestra únicamente el del robot LEGO NXT en el cual se probarán la mayoría de filtros. Se realiza la identificación del modelo de velocidad angular de los motores del robot al introducir un escalón en la acción de control (voltaje PWM especificado en % así un valor de +100 indica que se aplica el voltaje máximo posible al motor) para obtener la respuesta dinámica del motor como una función de transferencia de bucle abierto de primer orden para la velocidad. Se realizan dos pruebas con escalones positivos y negativos para cada motor (cuatro pruebas en total), estas se observan en la figura 4.4.

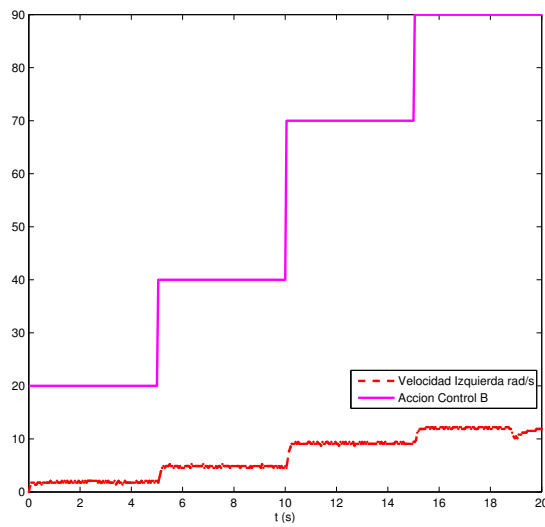
En total se deben identificar ocho funciones de transferencia para cada motor (dieciséis en total), para facilitar la tarea se utiliza el «System Identification Toolbox» de Matlab [®]. Primeramente se separan las pruebas completas en cuatro secciones cada una (una por escalón) y se identifica un modelo de primer orden sin tiempo muerto para cada una de estas (modelos locales). Los parámetros del modelo se muestran en el cuadro 4.2 en donde se calculan los modelos con los parámetros promedio de cada rueda.



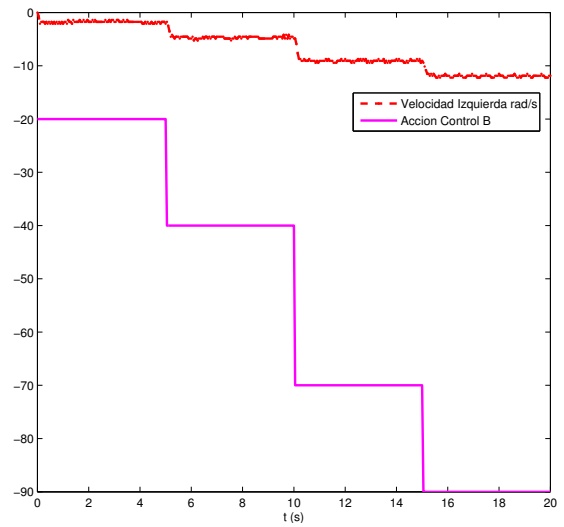
(a) Rueda Derecha +



(b) Rueda Derecha -



(c) Rueda Izquierda +



(d) Rueda Izquierda -

Figura 4.4: Pruebas realizadas a los motores del LEGO NXT

Cuadro 4.2: Modelos Locales para los motores del LEGO NXT

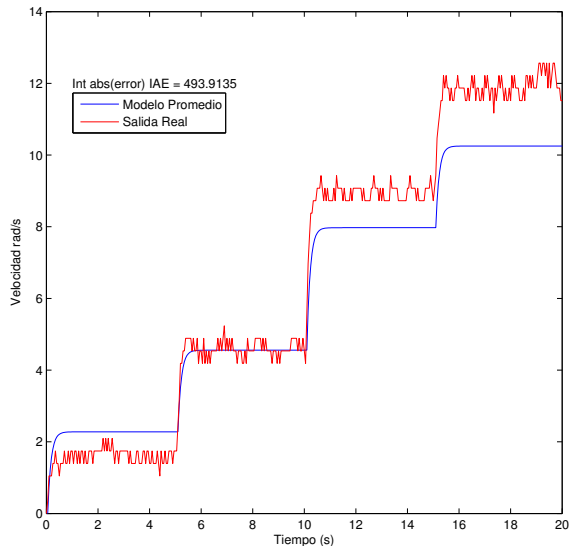
Escalón de Entrada	Motor Rueda Derecha (A)		Motor Rueda Izquierda (B)	
	Ganancia K_p	Cte. Tiempo τ	Ganancia K_p	Cte. Tiempo τ
20	0.0809	0.1561	0.0918	0.0534
40	0.1146	0.1452	0.1203	0.1062
70	0.1275	0.1342	0.1308	0.1003
90	0.1311	0.1359	0.1339	0.1080
-20	0.0809	0.0530	0.0876	0.0481
-40	0.1158	0.1567	0.1177	0.1374
-70	0.1290	0.1333	0.1296	0.1044
-90	0.1314	0.1120	0.1324	0.1077
PROMEDIO para	0.1139	0.1283	0.1180	0.0957
cada rueda (LOCAL)	$G_{pMA} = \frac{0.1139}{0.1283 s + 1}$		$G_{pMB} = \frac{0.118}{0.0957 s + 1}$	

Para observar el comportamiento del modelo promedio (LOCAL) se grafica su respuesta ante la entrada de prueba (serie de escalones) y se observa su desempeño al compararlo con la salida del motor real. Para esto se calcula la integral del valor absoluto del error (IAE) cuyo valor indica la cercanía de la estimación del modelo a la salida real (entre menor el IAE mejor la estimación). Las pruebas para el modelo se muestran en la figura 4.5.

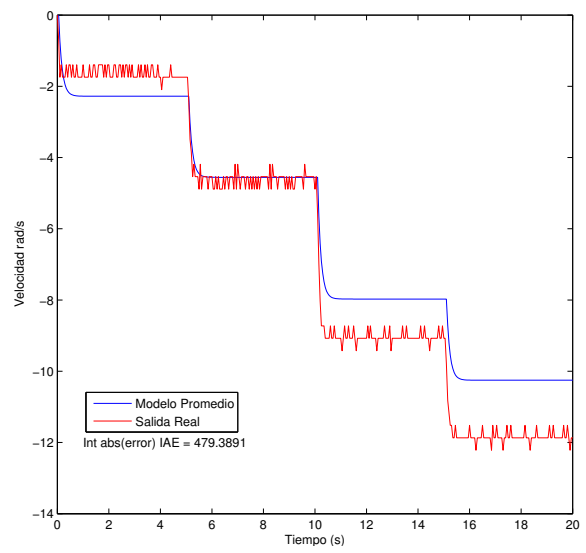
Se observa que los modelos promedio no aproximan de forma adecuada a todas las pruebas realizadas, ya que sus salidas difieren de forma considerable en velocidades mayores a 5rad/s. Para mejorar la estimación se identifica un modelo global para cada rueda utilizando todas las entradas a la hora de hacer la identificación (pruebas para todos los escalones positivos y negativos en conjunto, no por separado, función «pem» en Matlab). Los resultados de la identificación se resumen en el cuadro 4.3 en donde se calculan los modelos promedio de cada rueda a partir de los modelos globales.

Se observa que ambos modelos son muy similares entre sí (entre motores y entre pruebas). Para observar el comportamiento del modelo promedio (GLOBAL de cada rueda) se grafica su respuesta ante la entrada de prueba (serie de escalones) y se observa su desempeño al compararlo con la salida del motor real. Nuevamente se calcula el valor de IAE para poder comparar con los modelos promedio locales. Las pruebas para el modelo se muestran en la figura 4.6.

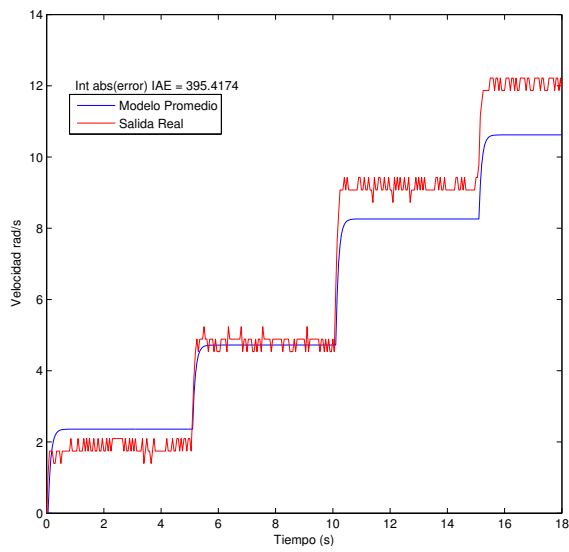
Se observa que el modelo promedio global de cada rueda se desempeña mejor que el promedio local ya que el valor de IAE es casi la mitad que en el caso local.



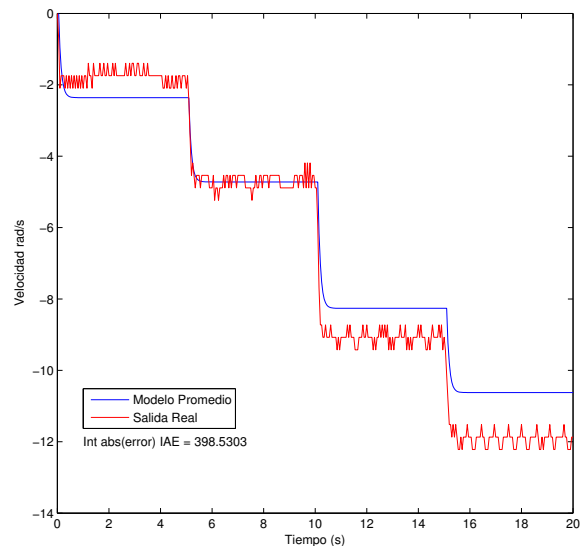
(a) Rueda Derecha +



(b) Rueda Derecha -

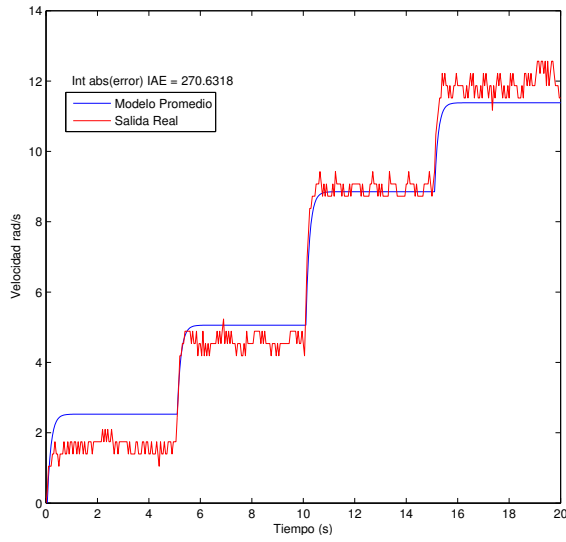


(c) Rueda Izquierda +

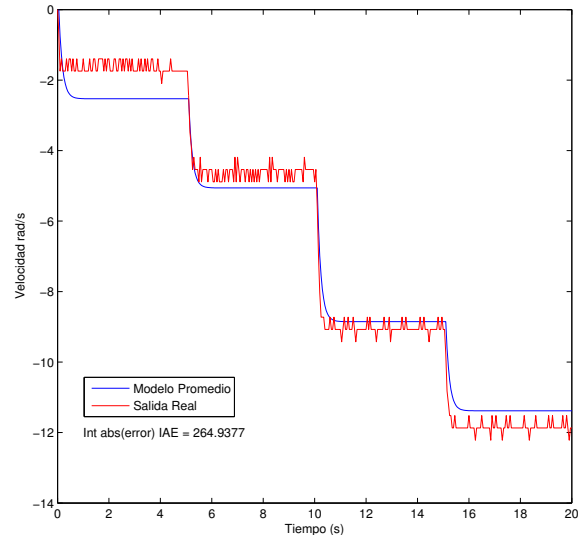


(d) Rueda Izquierda -

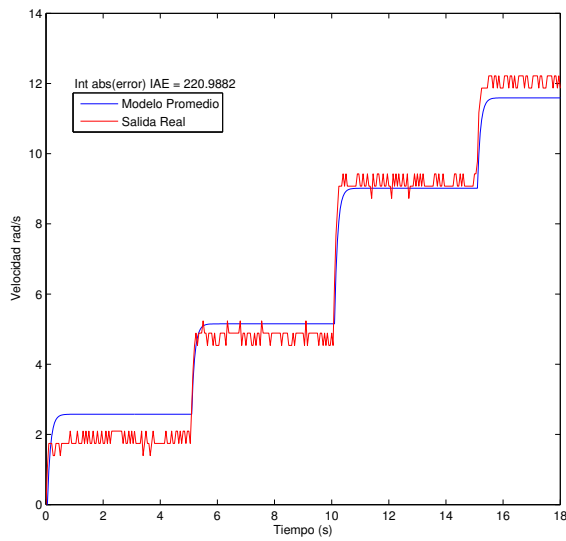
Figura 4.5: Respuesta del modelo local promedio de las ruedas del LEGO NXT



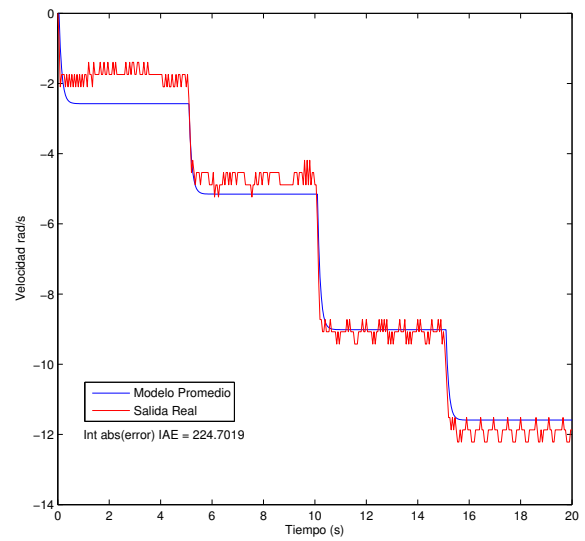
(a) Rueda Derecha +



(b) Rueda Derecha -



(c) Rueda Izquierda +



(d) Rueda Izquierda -

Figura 4.6: Respuesta del modelo global de las ruedas del LEGO NXT

Cuadro 4.3: Modelos Globales para los motores del LEGO NXT

Escalón de Entrada	Motor Rueda Derecha (A)		Motor Rueda Izquierda (B)	
	Ganancia K_p	Cte. Tiempo τ	Ganancia K_p	Cte. Tiempo τ
20, 40, 70, 90	0.1261	0.1513	0.1297	0.1062
-20, -40, -70, -90	0.1269	0.1307	0.1279	0.1059
PROMEDIO para	0.1265	0.1410	0.1288	0.1061
cada rueda (GLOBAL)	$G_{gMA} = \frac{0.1265}{0.141 s + 1}$		$G_{gMB} = \frac{0.1288}{0.1061 s + 1}$	

Para simplificar aún más el diseño del controlador se obtiene el modelo GLOBAL promedio calculando el promedio de los modelos de cada rueda, esto se muestra en la ecuación (4.25). Su desempeño en las pruebas con los datos de los motores se muestra en la figura 4.7.

$$G_{gMA} = \frac{0,1276}{0,1235 s + 1} \quad (4.25)$$

Se observa que el modelo global para ambas ruedas funciona muy similar a los modelos globales para cada rueda por separado, se mejora un poco el IAE en la rueda derecha y empeora en la izquierda, sin embargo esta variación es pequeña por lo que se considera aceptable con el fin de realizar un único diseño del control para las ruedas de los motores. Se utilizará por lo tanto el modelo global promedio de ambas ruedas (ecuación (4.25)) para el diseño del controlador PID que regulará la velocidad de los motores y para la estimación de la aceleración de cada rueda que actúa como entrada del modelo dinámico (4.23) y (4.24). Para determinar la aceleración de las ruedas se deriva la ecuación (4.25) para obtener (4.26).

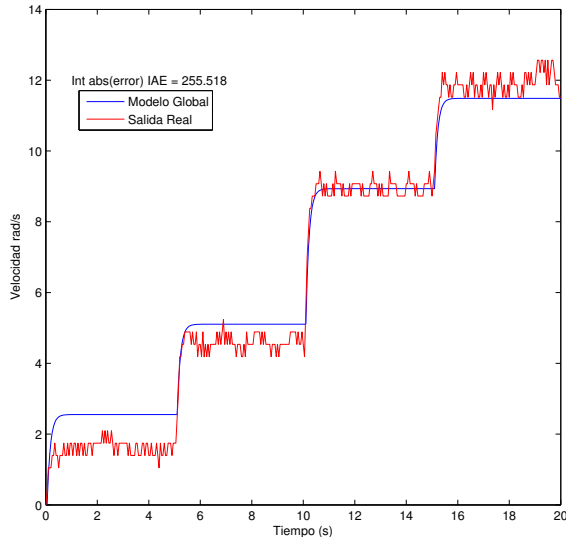
$$G_{gAcc} = \frac{a_{ccRueda}}{V_{control}} = \frac{0,1276 s}{0,1235 s + 1} \quad (4.26)$$

Este modelo relaciona la acción de control $V_{control}$ con la aceleración de la rueda $a_{ccRueda}$. Se discretiza (4.26) con un periodo de muestreo $T_s = 50ms$ con lo que se obtiene (4.27). Esta se escribe en forma de ecuación en diferencias en (4.28) para facilitar su implementación dentro del robot.

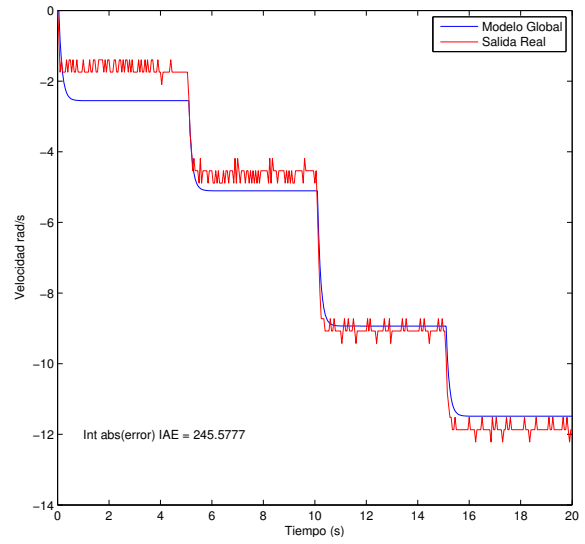
$$G_{gAcc}(z) = \frac{1,033z - 1,033}{z - 0,6671} = \frac{1,033 - 1,033z^{-1}}{1 - 0,6671z^{-1}} \quad (4.27)$$

$$a_{R,k} = 1,033V_{c,k} - 1,033V_{c,k-1} + 0,6671a_{R,k-1} \quad (4.28)$$

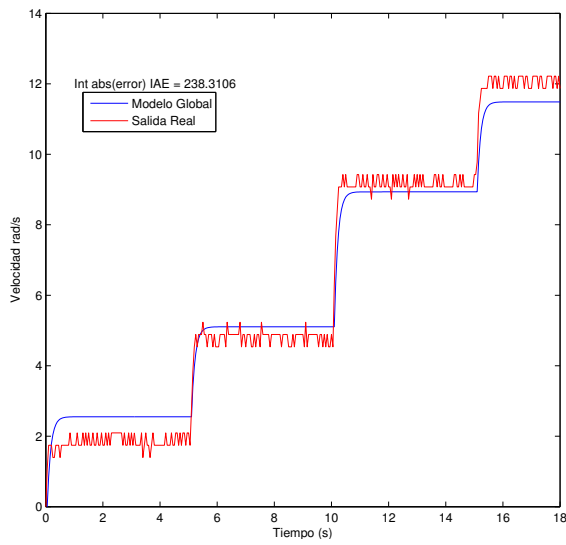
Realizado el modelado del robot se procede al diseño de los filtros de Kalman para fusión sensorial y el de los controladores PID de bajo nivel.



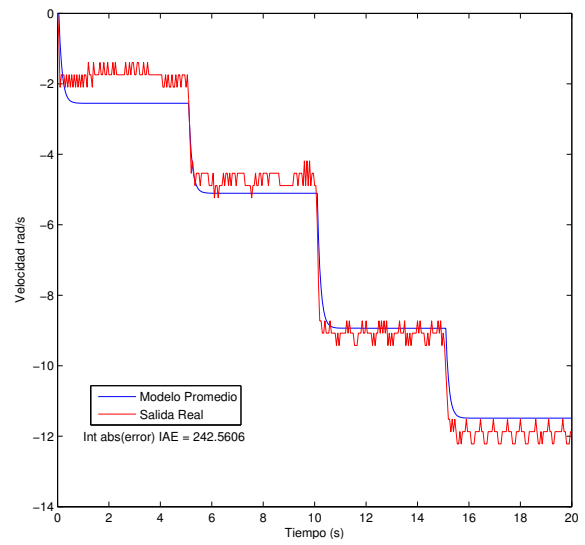
(a) Rueda Derecha +



(b) Rueda Derecha -



(c) Rueda Izquierda +



(d) Rueda Izquierda -

Figura 4.7: Respuesta del modelo global Promedio de las ruedas del LEGO NXT

Capítulo 5

Esquemas de fusión de datos con el filtro de Kalman

Utilizando la información de los sensores disponibles de las distintas plataformas, junto con los modelos desarrollados y los algoritmos del filtro de Kalman se desarrollan a continuación distintos esquemas de fusión, los cuales se podrán adaptar a las plataformas según sus sensores disponibles y capacidad de cómputo. Se expone a continuación una breve descripción de la calibración y preprocesamiento requerido en los sensores así como la forma en la que se seleccionan los valores de Q y R para los KF. Con esto se procede a explicar los esquemas de fusión implementados así como las pruebas simuladas de los mismos utilizando la información proveniente de los sensores reales.

5.1. Calibración y preprocesamiento de los Sensores

Los sensores de la mayoría de plataformas utilizadas proveen sus mediciones en unidades que pueden ser distintas a las del *S.I.* (sistema internacional de medidas) y por esta razón como primer paso antes de utilizarlas en el KF deben ser transformadas a este sistema. Además se debe remover cualquier «bias» (nivel que desplaza la señal del cero) para lo cual se realiza una prueba inicial con los sensores de la plataforma midiendo un mismo valor un cierto tiempo, con esto se puede determinar este nivel (promedio) para cada sensor para poder restarlo y obtener el valor de la variable que se desea medir. Este procedimiento debe realizarse siempre que se cambie un sensor porque los valores cambian aunque se tengan sensores de un mismo fabricante. Con

las señales calibradas se procede al preprocesamiento de los sensores. Esto es simplemente transformar la señal a la variable que deseamos utilizar en el esquema de fusión, esto se indica a continuación para los sensores que lo requieren:

- **Giroscopio:** En este caso el sensor provee una medición de la velocidad angular ω , por lo que puede utilizarse directamente o bien integrarla para obtener la medición del ángulo de avance del robot θ .
- **Brújula:** La mayoría de estos sensores (también conocidos como magnetómetros) dan una medida absoluta de la dirección del norte magnético, esta medida en general va entre los 0 y los 360 grados por lo que no se puede incorporar de forma directa como una medida de θ . Para esto se transforma esta señal en continua detectando el cruce por cero y acumulando el ángulo medido según su signo. Se lleva además un contador de los giros de 360° completos que se han dado y un indicador del sentido de giro (horario y antihorario). Con esto se tiene la medida de θ la cual puede derivarse para obtener ω según se necesite.
- **Encoders:** Estos sensores tienen como salida un contador que indica la cantidad de incrementos realizados por la rueda (este incremento es un factor de una vuelta de 360°, para un e-puck se tienen 1000 incrementos por vuelta y para un lego 360 pulsos por vuelta). Restando el valor anterior al valor actual del contador se obtiene el desplazamiento angular de las ruedas para un periodo de muestreo, este se transforma a desplazamiento lineal ($\Delta s_L, \Delta s_R$) (con un factor que depende del radio de las ruedas del robot y la cantidad de incrementos por vuelta). Finalmente al dividir este por el tiempo de muestreo (T_s) se obtiene la velocidad lineal de cada rueda (v_L, v_R). Ahora bien estas velocidades se relacionan con la (v, ω) del robot con la ecuación (5.1). Finalmente para los encoders se deben determinar el radio de ruedas y separación entre estas de forma precisa para mejorar la exactitud en las mediciones de v y ω , para esto se siguió el procedimiento expuesto en [51] en el ejemplo de odometría aunque existen otros métodos como por ejemplo [5] y [23] pero se selecciona [51] debido a sencillez y precisión.

$$\begin{bmatrix} v_{enc} \\ \omega_{enc} \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ -1/b & 1/b \end{bmatrix} \begin{bmatrix} v_L \\ v_R \end{bmatrix} \quad (5.1)$$

Con esto se tienen las señales acondicionadas para la fusión sensorial para la cual se deben escoger los valores de las matrices Q y R del filtro, estos se escogieron siguiendo el procedimiento descrito a continuación.

5.2. Parámetros del filtro de Kalman

Para implementar el algoritmo de filtro de Kalman es necesario conocer los valores de la matriz de covarianza del proceso Q y la de las mediciones R las cuales se utilizarán invariantes en el tiempo. La matriz Q define la precisión del modelo para describir el proceso real, los términos diagonales son la varianza de cada estado y los términos fuera de la diagonal son las covarianzas entre los estados. En el presente trabajo se elige una matriz Q diagonal por lo que se asume que no hay correlación entre los estados. Los términos de esta diagonal son modificados según se necesite para mejorar la estimación asignándolos primeramente del mismo orden de magnitud de los valores de la diagonal de R , según el desempeño del filtro se puede incrementar este valor (haciendo que el filtro tome menos en cuenta la estimación del modelo) o disminuirlo (para que el filtro tome más en cuenta lo que estima el modelo que lo que dicen los sensores). La matriz R se puede obtener mediante la medición de la varianza y covarianza de los sensores utilizados, por ejemplo al realizar una prueba midiendo el mismo valor en los sensores durante mucho tiempo y utilizando esta información para calcular la varianza del sensor y su covarianza con otros sensores. Sin embargo esta medición da valores que al ser utilizados directamente en el filtro pueden no producir un buen desempeño del mismo. De esta forma, el procedimiento más común es ajustar los valores de R de forma manual para mejorar la estimación del estado. Este ajuste se realiza considerando que los valores pequeños en la diagonal de R indican que el sensor correspondiente tiene una alta resolución y el filtro basará su estimación mayormente en esta medida (el modelo y los demás sensores aportan menos información a la estimación). Un valor grande en la diagonal de esta matriz hace que el filtro base su estimación utilizando más la predicción del modelo y/o la medición de otros sensores (con valores más bajos en los elementos de R).

Como se mencionó anteriormente se asume que la varianza de las mediciones son independientes por lo que la matriz R es diagonal. Si bien es cierto que algunas mediciones se obtienen de sensores distintos por lo que su varianza no está relacionada (su covarianza es cero) para otras mediciones esto no sucede. Por ejemplo si se determina la velocidad lineal y angular a partir de una misma fuente de información como lo son los encoders entonces estas medidas estarán relacionadas entre sí (la covarianza de estos es distinta de cero, por lo que los términos de R son distintos de cero fuera de la diagonal en estas medidas). Sin embargo debido a que se cuenta con sensores adicionales para la velocidad angular y como la velocidad lineal se puede estimar con el uso de los acelerómetros y el modelo dinámico, se puede realizar una simplificación y

establecer que la covarianza entre v y ω obtenida a partir de los encoders es cero. De esta forma se obtiene una matriz R diagonal que es más fácil de manipular en el algoritmo del KF. Además si se ajustan correctamente los parámetros de la matriz R esta simplificación es válida ya que el filtro calculará su estimación a partir de distintas fuentes de información por lo que no se basará únicamente en los encoders haciendo que el posible error en la aproximación de esta matriz sea pequeño. Si en el robot no se cuentan con más sensores que los encoders y se desea realizar una implementación tomando en cuenta estas covarianzas de las velocidades a partir de los encoders se puede consultar el trabajo realizado por [19]. Por último, el filtro necesita la estimación del estado en el instante inicial $\hat{x}_{k-1,ini}$ y la covarianza de error inicial $P_{k-1,ini}$. El estado inicial es normalmente conocido porque el robot comienza a moverse desde el reposo con lo que las velocidades en el instante inicial son cero. Además se deberá conocer el punto del cual el robot inicia su movimiento o asumir que se inicia del origen y estimar la posición desde el inicio del movimiento. La matriz de covarianza del error inicial se define como una matriz diagonal con los términos mayor que cero. Estos términos indica la exactitud de los estados inicial y debe ser cuidadosamente elegido para hacer el filtro converja aunque si el estado es bien conocido en el instante inicial, entonces los valores de esta matriz son bajos.

Con esto se tienen todos los elementos necesarios para la fusión sensorial con el KF. Si bien es cierto la forma seleccionada para ajustar Q y R requiere realizar múltiples pruebas para obtener un buen desempeño del filtro, el esfuerzo merece la pena ya que como se mostrará en el capítulo de implementación se logran conseguir resultados muy buenos en la fusión sensorial. A continuación se describen los esquemas utilizados en las distintas plataformas en donde se muestran los valores utilizados para los parámetros del filtro así como que modelo y algoritmo de KF se utiliza para su cálculo.

5.3. Filtro de Kalman para la velocidad de las ruedas

Si se tienen plataformas muy limitadas en donde no hay más sensores que los encoders, se puede aprovechar parte del desarrollo para implementar un filtro básico KF utilizando únicamente un sensor, por lo que si bien es cierto no hay fusión entre sensores distintos, el filtro si fusiona la información entre el modelo y los encoders. Si además la plataforma no tiene muchos recursos computacionales se deberá reducir al máximo el cálculo asociado al filtro. Una solución básica que puede mejorar el desempeño de la odometría es estimar las velocidades de las ruedas del motor antes de usar (5.1) junto con para la ecuación de odometría (4.12). Para esto se plantea un

modelo muy sencillo $\omega(t) = \dot{\theta}(t)$ para cada rueda, de esta forma se toma $x_1(t) = \theta(t)$, $x_2(t) = \omega(t)$ por lo que el modelo lineal de la velocidad de la rueda se muestra en (5.2).

$$\dot{x}(t) = Ax(t) + w(t), \quad A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad (5.2)$$

Este modelo se discretiza utilizando (5.3). Además como solo se dispone de los encoders, la ecuación de la medición se muestra en (5.4)

$$x_k = e^{A \cdot T_s} x_{k-1} + w_{k-1} \quad (5.3)$$

$$z_k = Hx_{k-1} + v_{k-1}, \quad H = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (5.4)$$

Los parámetros del filtro KF (figura 3.2) se muestran (5.5) en para el robot e-puck y en (5.6) para el robot LEGO.

$$R_R = 0,1; R_L = 0,1$$

$$Q = \begin{bmatrix} 0,1 & 0 \\ 0 & 3 \end{bmatrix}; x_{k-1,ini} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; P_{k-1,ini} = 300 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (5.5)$$

$$R_R = 2,5; R_L = 1,5$$

$$Q = \begin{bmatrix} 0,1 & 0 \\ 0 & 10 \end{bmatrix}; x_{k-1,ini} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; P_{k-1,ini} = 10 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (5.6)$$

5.4. Filtro de Kalman para la velocidad del robot

Este método se aplica cuando se tienen disponibles más sensores en el robot (como es el caso del LEGO) y es la principal aportación del presente trabajo, ya que se utiliza el filtro KF lineal (figura 3.2) junto con el modelo local del robot (4.23) para hacer la fusión sensorial y estimar v y ω . Este esquema permite fusionar los distintos sensores sin necesidad de utilizar un modelo complejo que requiere una inversión de matrices grandes para el cálculo de la ganancia del filtro y por lo tanto de la estimación del estado. Esto ahorra recursos computacionales y como se muestra en las secciones siguientes tiene un desempeño comparable a filtros más complejos. Para determinar la posición global se utilizan las velocidades estimadas en la ecuación (4.11).

El esquema de funcionamiento se muestra en la figura 5.1 para el caso de utilizar la medición de acelerómetros como entrada al modelo dinámico. Se toman la información de los sensores y se preprocesa. Las aceleraciones medidas se utilizan en el modelo local (ecuaciones (4.22)

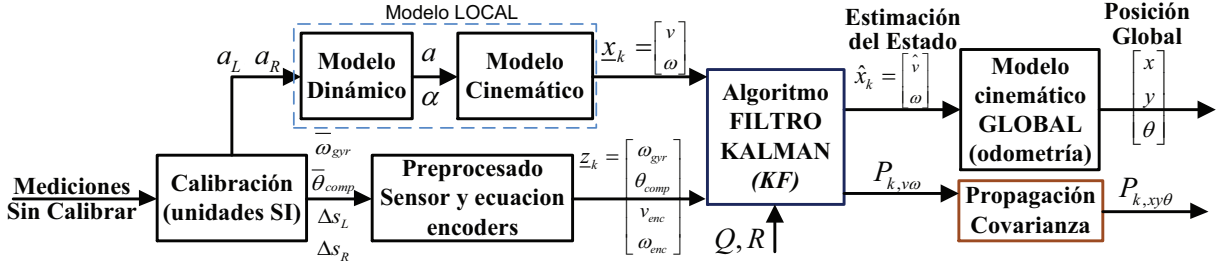


Figura 5.1: Esquema de fusión con el KF para las velocidades del robot, con acelerómetros

y (4.23)). Para el LEGO NXT los parámetros de (4.22) son $c_R = 0,353$, $c_L = 0,647$ y $\lambda = 0,02386$. Las mediciones restantes se preprocesan para obtener las mediciones de v y de ω o θ según se requiera. Esto se introduce al algoritmo del KF junto con las matrices Q y R . La salida es la estimación de v, ω que se usan en la ecuación (4.11) para obtener la posición del robot.

Las aceleraciones se miden utilizando dos acelerómetros colocados encima de cada rueda del robot. Existen varios estudios sobre el uso de acelerómetros como única fuente de información inercial para observar el movimiento (velocidad o posición) del robot (ver por ejemplo [16], [63], [65], [56] y [58]), pero ninguno de estos se contempla con el modelo dinámico local propuesto ni se utiliza en un esquema fusión sensorial. Sin embargo de estos trabajos surge la idea de incorporar más de un acelerómetro en el robot (en este caso 2) y utilizar sus medidas como entrada al modelo dinámico. Para esta versión del filtro se tiene que la ecuación de medidas mostrada en la ecuación (5.7). Esta ecuación implementa la fusión de la información de los sensores, v_{enc} y ω_{enc} son la velocidad lineal y angular obtenidas de los encoders, ω_{gyr} y ω_{cmp} son las velocidades angulares medidas mediante el giróscopo y el compás. Las matrices de covarianzas de este esquema R y Q se muestran en la ecuación (5.8).

$$\tilde{z}_k = \begin{bmatrix} v_{enc} \\ \omega_{enc} \\ \omega_{gyr} \\ \omega_{cmp} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} + v \quad (5.7)$$

$$Q_{KF} = \begin{bmatrix} 5,2 & 0 \\ 0 & 5,2 \end{bmatrix}, R = \begin{bmatrix} 4,5 & 0 & 0 & 0 \\ 0 & 0,0152 & 0 & 0 \\ 0 & 0 & 0,0232 & 0 \\ 0 & 0 & 0 & 0,0883 \end{bmatrix} \quad (5.8)$$

En caso de no disponer de estos se puede utilizar el modelo dinámico de los motores del robot para obtener la aceleración en función de la acción de control tal y como se obtuvo en la

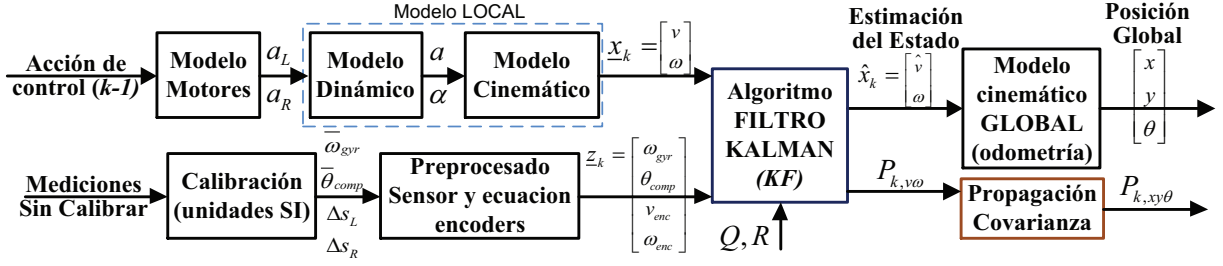


Figura 5.2: Esquema de fusión con el KF para las velocidades del robot, sin acelerómetros

ecuación (4.28). En este caso el esquema cambia levemente ya que se utilizan menos sensores, este se muestra en la figura 5.2. La ecuación de las medidas es la misma que en el caso anterior (5.7) y las matrices de covarianzas de este esquema R y Q se muestran en la ecuación (5.9). Los valores de Q son menores ya que el modelo no tiene tanto ruido comparado con el caso anterior donde estos valores se asocian a los acelerómetros.

$$Q_{KF} = \begin{bmatrix} 0,02 & 0 \\ 0 & 0,02 \end{bmatrix}, R = \begin{bmatrix} 0,5 & 0 & 0 & 0 \\ 0 & 0,0152 & 0 & 0 \\ 0 & 0 & 0,0232 & 0 \\ 0 & 0 & 0 & 0,0883 \end{bmatrix} \quad (5.9)$$

Al utilizar este esquema de fusión se tiene la ventaja de un bajo coste computacional porque utiliza matrices de tamaño pequeño para obtener la Ganancia de Kalman en (3.5) pero con la desventaja de que la incertidumbre en la medida de v y ω no se propaga hasta x, y, θ . Sin embargo esto se puede solucionar usando la matriz de covarianzas de v y ω de una forma similar a la propuesta en [6]. Esto se muestra en la ecuación recursiva (5.10) donde Px_k es la covarianza en x, y, θ en el instante k , $P_{k,v\omega}$ es la covarianza de v y ω (obtenido a partir del KF) y ∇F_u es el operador gradiente aplicado a la ecuación de la odometría (4.11) respecto a las entradas v y ω .

$$\nabla F_u = \begin{bmatrix} \alpha c & -0,5v_k T_s \beta c \\ \beta c & 0,5v_k T_s \alpha c \\ 0 & T_s \end{bmatrix}, \quad \begin{aligned} \alpha c &= T_s \cos(\theta_k + \omega_k \cdot 0,5T_s) \\ \beta c &= T_s \sin(\theta_k + \omega_k \cdot 0,5T_s) \end{aligned} \quad (5.10)$$

$$Px_k = Px_{k-1} + \nabla F_u P_{k,v\omega} \nabla F_u^T + Q$$

El esquema de fusión propuesto se prueba con datos obtenidos del robot móvil en configuración diferencial LEGO y se compara con el EKF y UKF.

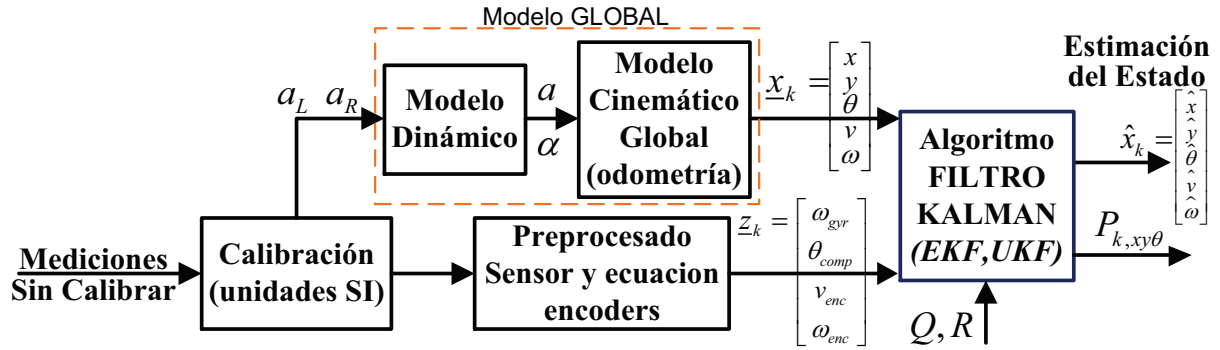


Figura 5.3: Esquema de fusión con el EKF para las velocidades del robot, con acelerómetros

5.5. Filtro extendido de Kalman para la posición del robot

Este método se aplica cuando se tienen disponibles más sensores en el robot y también más recursos computacionales (este método funciona en el LEGO al optimizar el cálculo de la ganancia del filtro). En este caso se utiliza el EKF (figura 3.3) junto con el modelo global del robot (4.24) linealizado utilizando (3.11) para hacer la fusión sensorial y estimar directamente la posición del robot x, y, θ . Este esquema fusiona los distintos sensores pero utilizando un modelo complejo que requiere una inversión de matrices grande para el cálculo de la ganancia del filtro y por lo tanto de la estimación del estado. Esto gasta mucho más recursos computacionales que el esquema anterior pero tiene la ventaja de que propaga la incertidumbre de forma directa a la posición.

El esquema de funcionamiento se muestra en la figura 5.3 para el caso de utilizar la medición de acelerómetros como entrada al modelo dinámico. Se toman la información de los sensores y se preprocesa. Las aceleraciones medidas se utilizan en el modelo global (ecuaciones (4.22) y (4.24) linealizado con (3.11)). Para el LEGO NXT los parámetros de (4.22) son $c_R = 0,353$, $c_L = 0,647$ y $\lambda = 0,02386$. Las mediciones restantes se preprocesan para obtener las mediciones de v y de ω o θ según se requiera. Esto se introduce al algoritmo del KF junto con las matrices Q y R . La salida es la estimación de x, y, θ, v, ω .

Las aceleraciones se miden utilizando dos acelerómetros colocados encima de cada rueda del robot igual que en el caso anterior. Para esta versión del filtro se tiene que la ecuación de medidas mostrada en la ecuación (5.11). Esta ecuación implementa la fusión de la información de los sensores, v_{enc} y ω_{enc} son la velocidad lineal y angular obtenidas de los encoders, ω_{gyr} y ω_{cmp} son las velocidades angulares medidas mediante el giróscopo y el compás. Las matrices

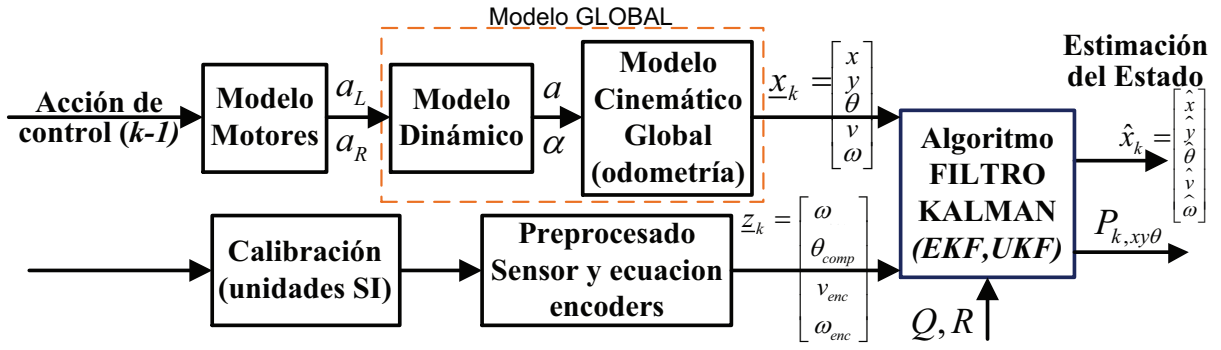


Figura 5.4: Esquema de fusión con el EKF para las velocidades del robot, sin acelerómetros

de covarianzas de este esquema R y Q se muestran en la ecuación (5.12).

$$z_k = \begin{bmatrix} v_{enc} \\ \omega_{enc} \\ \omega_{gir} \\ \omega_{bru} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \\ \omega \\ v \end{bmatrix} + v \quad (5.11)$$

$$Q_{EKF} = \begin{bmatrix} 0,0002 & 0 & 0 & 0 & 0 \\ 0 & 0,0002 & 0 & 0 & 0 \\ 0 & 0 & 0,000 & 0 & 0 \\ 0 & 0 & 0 & 0,3 & 0 \\ 0 & 0 & 0 & 0 & 0,8 \end{bmatrix}, R = \begin{bmatrix} 5,1 & 0 & 0 & 0 \\ 0 & 0,0152 & 0 & 0 \\ 0 & 0 & 0,0232 & 0 \\ 0 & 0 & 0 & 0,0883 \end{bmatrix} \quad (5.12)$$

En caso de no disponer de estos se puede utilizar el modelo dinámico de los motores del robot para obtener la aceleración en función de la acción de control tal y como se obtuvo en la ecuación (4.28). En este caso el esquema cambia levemente ya que se utilizan menos sensores, este se muestra en la figura 5.4. La ecuación de las medidas es la misma que en el caso anterior (5.7) y las matrices de covarianzas de este esquema R y Q se muestran en la ecuación (5.9). Los valores de Q son menores ya que el modelo no tiene tanto ruido comparado con el caso anterior donde estos valores se asocian a los acelerómetros.

$$Q_{EKF} = \begin{bmatrix} 0,0002 & 0 & 0 & 0 & 0 \\ 0 & 0,0002 & 0 & 0 & 0 \\ 0 & 0 & 0,000 & 0 & 0 \\ 0 & 0 & 0 & 0,03 & 0 \\ 0 & 0 & 0 & 0 & 0,08 \end{bmatrix} R = \begin{bmatrix} 0,5 & 0 & 0 & 0 \\ 0 & 0,0152 & 0 & 0 \\ 0 & 0 & 0,0232 & 0 \\ 0 & 0 & 0 & 0,0883 \end{bmatrix} \quad (5.13)$$

El esquema de fusión propuesto se prueba con datos obtenidos del robot móvil en configuración diferencial LEGO y se compara con el KF y UKF.

5.6. Filtro UKF para la posición del robot

Como se mencionó anteriormente el UKF se usará como medida de desempeño para comparar los resultados obtenidos a partir del KF y del EKF para lo cual se utilizan los algoritmos UKF en MATLAB desarrollados por [30] (disponibles en línea). Es decir, este algoritmo no se implementará en las plataformas, únicamente se usara en las pruebas simuladas de la sección siguiente. Para este filtro se sigue el mismo esquema que en el EKF de la figura 5.3 pero utilizando la ecuación de medidas mostrada en (5.14) y cambiando las matrices Q y R del filtro por las mostradas en la ecuación (5.15).

$$\hat{z}_k = \begin{bmatrix} v_{enc} \\ \theta_{enc} \\ \theta_{gir} \\ \theta_{bru} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \\ \omega \\ v \end{bmatrix} + v \quad (5.14)$$

$$Q_{UKF} = \begin{bmatrix} 0,0002 & 0 & 0 & 0 & 0 \\ 0 & 0,0002 & 0 & 0 & 0 \\ 0 & 0 & 0,0002 & 0 & 0 \\ 0 & 0 & 0 & 0,1 & 0 \\ 0 & 0 & 0 & 0 & 0,1 \end{bmatrix}, R = \begin{bmatrix} 5,1 & 0 & 0 & 0 \\ 0 & 0,0138 & 0 & 0 \\ 0 & 0 & 0,0332 & 0 \\ 0 & 0 & 0 & 0,0883 \end{bmatrix} \quad (5.15)$$

5.7. Sensorización global por eventos

Debido a que los esquemas presentados hasta el momento basan su funcionamiento únicamente en la información local del robot, la incertidumbre en la posición puede ser pequeña al inicio del movimiento pero tenderá a aumentar conforme lo hace la distancia recorrida por el robot para todos los filtros. Esto indica la necesidad de contar con una fuente de información global como una cámara cenital para corregir la estimación local cada cierto tiempo.

Para realizar esta actualización el robot móvil deberá comunicarse mediante bluetooth a un servidor que administra el acceso a la cámara cenital y realiza el procesamiento de la imagen para

determinar la posición real del robot. Al ser un dispositivo móvil de recursos limitados se presentan 2 problemas, el primero es el retardo en la comunicación entre el robot y la cámara ya que si se solicita a la cámara la posición en un instante determinado ésta llegará con un cierto retardo. Esto se compensa determinando la velocidad a la que se mueve el robot y de esta forma trasladar la posición recibida de la cámara al instante actual (conociendo el tiempo que tarda la petición al servidor).

El otro problema presente es el de cómo realizar la incorporación de la información de la cámara a la estimación de la posición. Para esto existen distintas soluciones, por ejemplo se podría incrementar el esquema del EKF para incorporar a la cámara como un sensor adicional, pero esto incrementa la matriz de mediciones por lo que el cálculo de la inversa requerida para la ganancia del filtro crece de dimensión lo cual podría no ser implementable dentro del robot. Si suponemos que esto no es problema, aun se tiene la dificultad de hacer que la información de la cámara esté disponible en el robot cada periodo de muestreo lo cual no es posible debido a los retardos de comunicación. En este caso la información de la cámara llegará al robot únicamente en ciertos instantes de tiempo (con un periodo distinto al de muestreo) lo cual requiere modificar la matriz R para que sea variante en el tiempo (valor muy alto si la medición no está disponible y valor de la covarianza de la cámara en caso contrario).

Otros esquemas son mucho más complejos requiriendo retroceder la estimación en el tiempo para que coincida con el instante en donde se obtuvo la medición de la cámara, actualizar el estado en este instante y luego adelantar la estimación al instante de tiempo actual (ver [9] y la sección 10.5 de [62]). Estas soluciones se consideran muy complejas para ser implementadas en un robot de recursos limitados debido al alto coste computacional requerido en estos esquemas con R variante en el tiempo o con mediciones con retardos en el tiempo en el los filtros KF.

Además de estas dificultades, se tiene el problema de que el robot es un dispositivo operado por baterías, por lo que si se realiza una actualización con una frecuencia alta de su posición utilizando la cámara se agotarían muy rápido las baterías ya que se harían muchas llamadas al servidor, además de limitar la ejecución de otras tareas dentro del robot (tales como el cálculo del KF, el control de motores y control de trayectorias) por lo que podría darse un mal desempeño.

Tomando en cuenta todos estos factores se propone a continuación un esquema en el cual la actualización de la posición se hace directamente sobre la estimación del estado lo cual omite

la varianza de la medición de la cámara a cambio de ser un procedimiento más sencillo con posibilidad de ser implementado. Este esquema es similar al propuesto por [12] (capítulo 10) pero con la gran diferencia de no requerir una medición con un periodo de muestreo igual al del filtro KF implementado.

La solución consiste en que el robot realice el menor número posible de llamadas a la cámara para actualizar su posición para de esta forma ahorrar batería y tiempo de procesador además reducir la incertidumbre de la estimación. Para esto se define la actualización por eventos, aprovechando que los filtros KF calculan la covarianza del error en la posición, se puede utilizar esta información para determinar cuándo este error supera un nivel máximo permitido. A partir de la covarianza del error se calculan los elipsoides de incertidumbre (con el intervalo de error $3 - \sigma$ equivalente al 98 % de probabilidad) mediante las ecuaciones propuestas en [64] (sección 6.5). Si se toma el área de estos elipsoides y se dividen por el área del robot se tendría un indicador normalizado del momento en el cual el error en la posición del robot es igual de grande que el tamaño del robot. Solamente cuando este supera un nivel determinado (por ejemplo 1.6 indicando que el área del error es 1.6 veces el de robot) se realiza la llamada a la cámara para actualizar la posición. Esta forma es mucho más eficiente que la actualización por tiempo.

Una vez que se genera el evento el robot sigue el comportamiento mostrado en la figura 5.5. Al inicio el robot se encuentra en una trayectoria real medida por la cámara que es distinta a la trayectoria estimada por el esquema de fusión KF y distinta además a la trayectoria deseada. En este instante k_0 se produce el evento por lo que el robot llama a la cámara la cual determina que la posición real del robot es P_{cam} . Esta posición llega tarde al robot en el instante k_1 debido al retardo de comunicación. Una vez llega al robot este traslada la posición de la cámara utilizando las velocidades del robot y el tiempo que tardó la información en llegar desde que se realiza la llamada (medido con un temporizador). A este tiempo se le resta el tiempo que tardó la cámara en procesar la imagen que se considera como un valor promedio (medio a priori) y se divide entre dos para obtener el tiempo que tardó la información en llegar al robot desde que se obtuvo la imagen de la posición del robot. Con la posición trasladada se le resta la posición estimada por el KF en el momento k_0 (almacenada en memoria) y con esto se obtiene el desplazamiento en los ejes globales para hacer que la estimación del KF coincida con la medición de la cámara ($\Delta X, \Delta Y$). Estos valores se suman a la estimación de la posición del esquema de fusión KF en el instante k_1 . Al actualizar la posición se genera un error entre la posición del robot y la posición deseada, por lo que el algoritmo de seguimiento de trayectorias generará las acciones de control para hacer que el robot siga realmente a la trayectoria deseada y que esta coincida

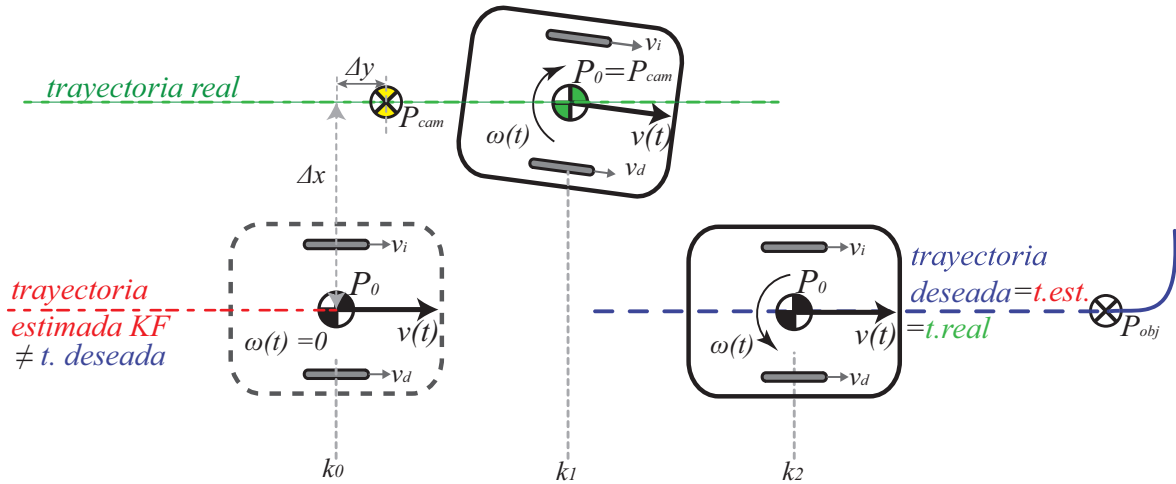


Figura 5.5: Esquema general de la sensorización global por eventos

con la medida por la cámara y por la estimada en el filtro KF, lo cual se consigue en el instante k_2 . Cabe destacar que este método funcionará adecuadamente si el tiempo de retardo no es muy grande (esto dependerá de la plataforma utilizada) y los valores de ΔX y ΔY no son muy grandes (ya que la traslación de la posición de la cámara entre instantes de tiempo se realiza de forma lineal). Descritas todos los esquemas planteados se procede con la simulación de estas, previo a la implementación en las plataformas.

5.8. Simulación y comparación de los esquemas de fusión

Se implementan todos los esquemas de fusión propuestos en MATLAB y se utilizan como entradas los datos reales obtenidos del robot LEGO NXT cuando el robot sigue una trayectoria como referencia cuadrada (80cm lado) y circular (80cm diámetro) utilizando únicamente la información de los encoders (sin calibración). Para esto se utilizó un control por punto descentralizado (expuesto en la introducción y en [48]) para el control de trayectoria y un controlador P para el accionamiento de los motores. La trayectoria real seguida por el robot se mide mediante una cámara cenital y se almacenan dentro del robot todos los datos de los sensores sin preprocesar (este se realiza en los algoritmos MATLAB). Tal como era de esperar, las medidas de los encoders no son suficientemente exactas y el robot estima y sigue una trayectoria distinta de la deseada (esta es la trayectoria medida por la cámara). Usando el KF, el EKF y el UKF con la fusión de sensores propuesta y la configuración de acelerómetros dual, la estimación de la posición del robot mejora en gran medida y está más cerca del valor real. Esto se muestra en

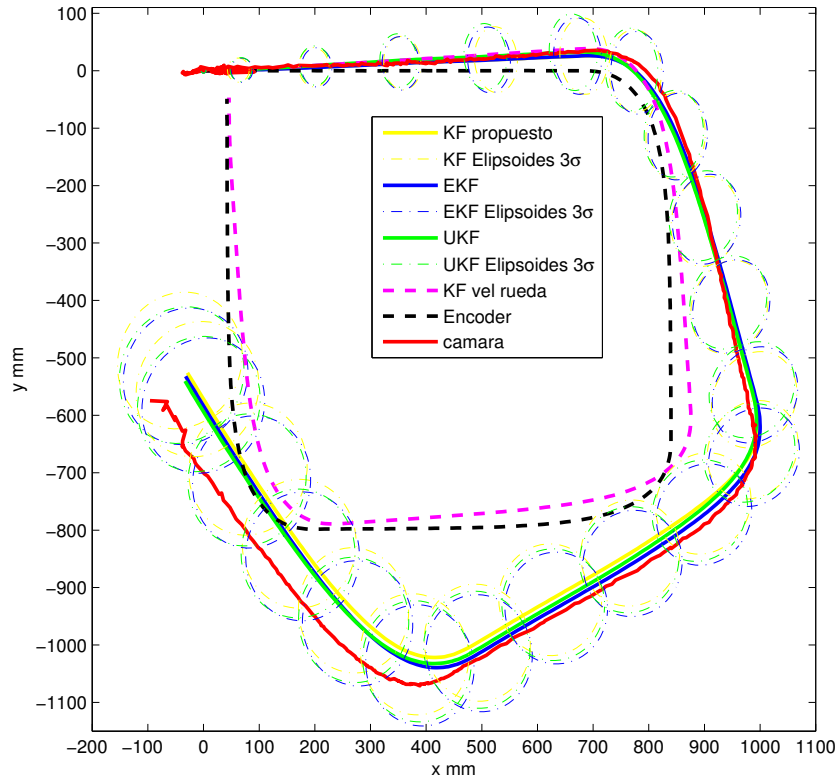


Figura 5.6: Simulación: Trayectoria cuadrada seguida por el robot móvil LEGO NXT sin implementar el KF, comparación de los esquemas de fusión

las figuras 5.6 y 5.7 junto con la estimación basada únicamente en los encoders, con el KF de las ruedas del robot e indicando los elipsoides del intervalo de error $3 - \sigma$ obtenidos a partir de la matriz de covarianzas del error P_k (para el KF se utiliza la ecuación (5.10)) para el KF de velocidad del robot y el EKF y UKF de la posición. Se observa como el esquema de fusión de datos propuesto con el KF de velocidades del Robot funciona de manera adecuada, estimando la posición del robot de manera similar a como lo hacen los filtros EKF y UKF pero usando menos recursos de computación del robot móvil. Se observa además que en el caso del robot LEGO el filtro de velocidades de las ruedas no es suficiente para una buena estimación de la posición del robot.

Para observar la estimación de las velocidades del robot por el KF se muestran las figuras 5.8 y 5.9 en donde se observa el buen funcionamiento del filtro KF de velocidades del robot realizando una estimación que toma en cuenta los distintos sensores del robot y el modelo definido.

En cuanto a la sensorización global por eventos se utilizan los datos del robot pero utilizando un KF de velocidad dentro del robot para estimar su posición (tomado del capítulo siguiente

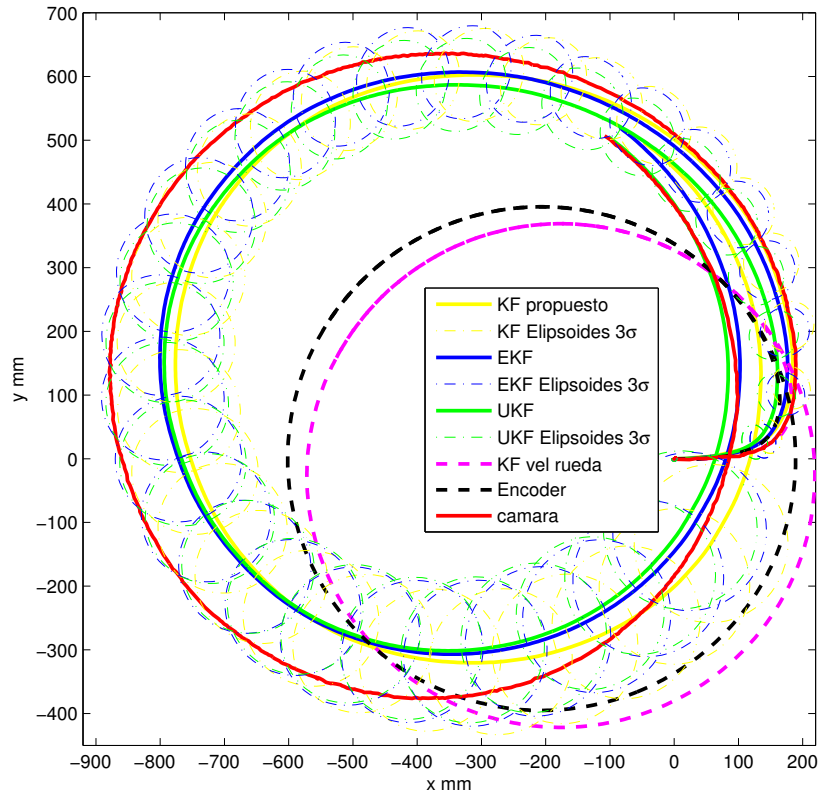


Figura 5.7: Simulación: Trayectoria circular seguida por el robot móvil LEGO NXT sin implementar el KF, comparación de los esquemas de fusión

para simular la actualización por eventos) junto con los datos de la cámara cenital que midió la trayectoria recorrida. En la figura 5.10(a) se muestra el caso para la trayectoria cuadrada al actualizar la posición del robot agregando diferencia entre la estimación del KF y la medición de la cámara cada cierto tiempo (tal y como se sugiere en el capítulo 10 de [12], esto para los casos de actualizar la posición compensando un retardo de comunicación de 0,4s haciendo llamadas excesivas (cada segundo) a la cámara o bien sin compensar el retardo en comunicación y llamando menos veces a la cámara (cada 7segundos). Se muestra que en ambos casos no se tiene una buena estimación de la posición por lo que no es adecuada la actualización por tiempo. En cambio para el método propuesto de actualización de la posición por eventos este funciona de mejor forma tal y como se muestra en la figura 5.10(b) en la cual se observa un buen comportamiento de la estimación de la posición evitando que su incertidumbre crezca de forma indefinida. Con los buenos resultados obtenidos en las simulaciones se procede a implementar los distintos algoritmos en las plataformas propuestas y observar así su comportamiento.

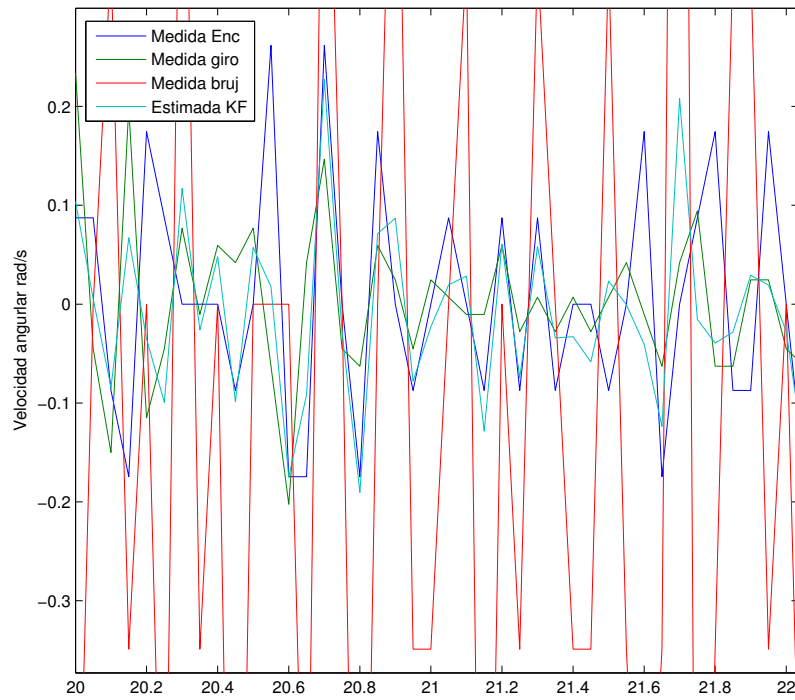


Figura 5.8: Simulación: KF estimando la velocidad lineal del robot LEGO

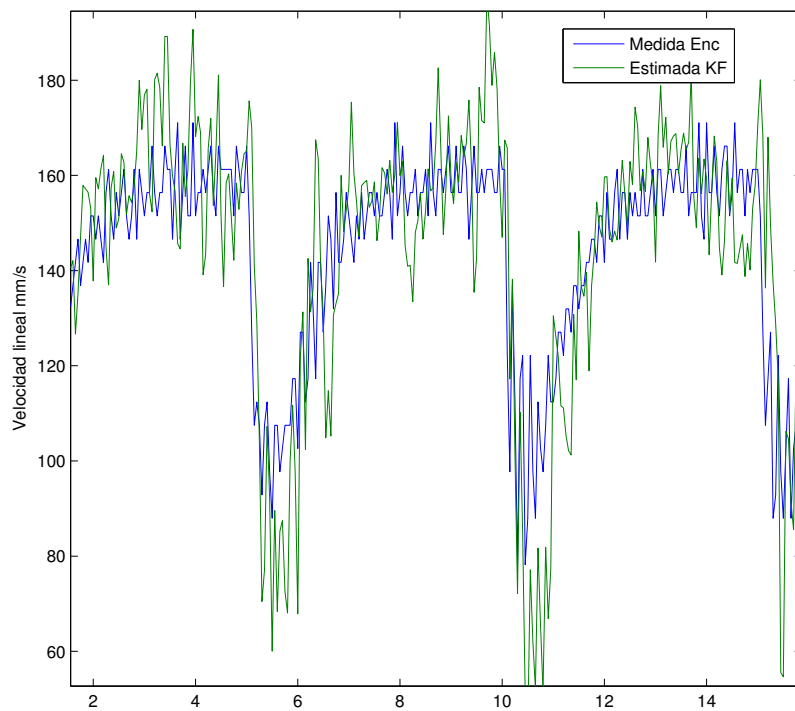
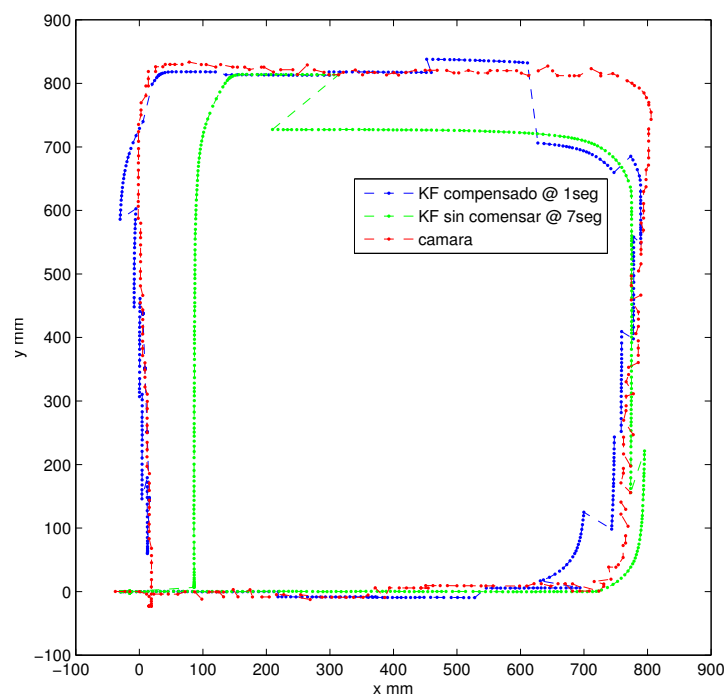
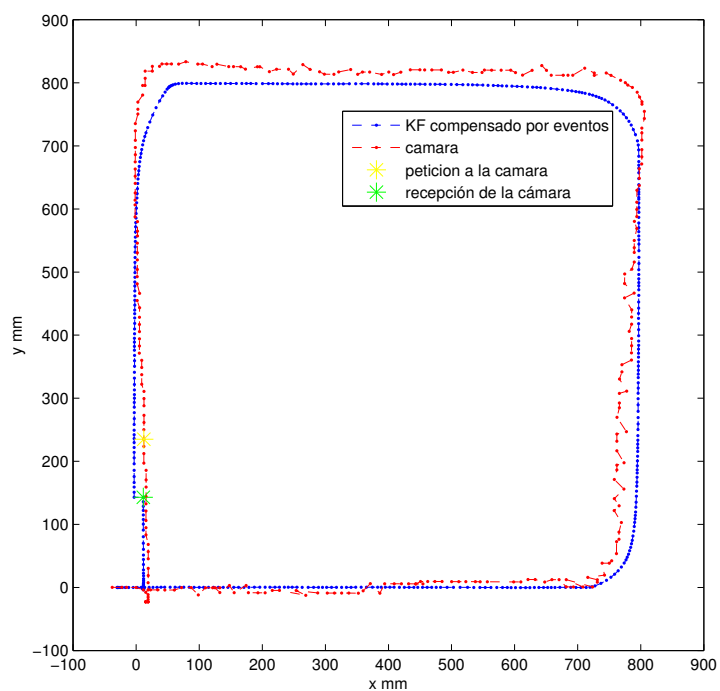


Figura 5.9: Simulación: KF estimando la velocidad angular del robot LEGO



(a) Por Tiempo



(b) Por Eventos

Figura 5.10: Simulación: sensorización global por eventos, actualización de la posición con el KF y la cámara cenital

Capítulo 6

Pruebas implementadas de la fusión de datos

Se exponen a continuación los principales resultados obtenidos de implementar los esquemas de fusión propuestos en las distintas plataformas. En general los tiempos de muestreo son de $50ms$ para los robots y de $100ms$ para el sensor IG500N. Además para obtener los algoritmos de Kalman se utilizó la «Symbolic Math Toolbox» de MATLAB, definiendo las matrices del filtro con términos simbólicos y calculando el resultado de cada paso del filtro para obtener las operaciones del KF separadas en un conjunto de operaciones de una línea en lugar de usar algebra de matrices.

6.1. Pruebas con el e-puck

Se implementa el filtro de Kalman de Velocidad de las ruedas para mejorar la estimación de la velocidad y por lo tanto el seguimiento de la trayectoria deseada. En las siguientes figuras se muestra el resultado del filtro ejecutándose en el robot al medir las velocidades de las ruedas y compararlas con la estimación del filtro. Cabe destacar que los motores del *e-puck* son del tipo paso a paso («stepper») con 1000 pasos por 360° por lo que la odometría calibrada es mucho más precisa que en el caso del LEGO ya que tiene mayor resolución que este. De esta forma se obtiene el buen desempeño mostrado en las pruebas. La trayectoria seguida por el *e-puck* real en la prueba inicial se muestra en la figura 6.3(a) (se inicia en la esquina inferior izquierda) y en la figura 6.3(b) se muestra la trayectoria estimada por la cámara y el módulo de odometría

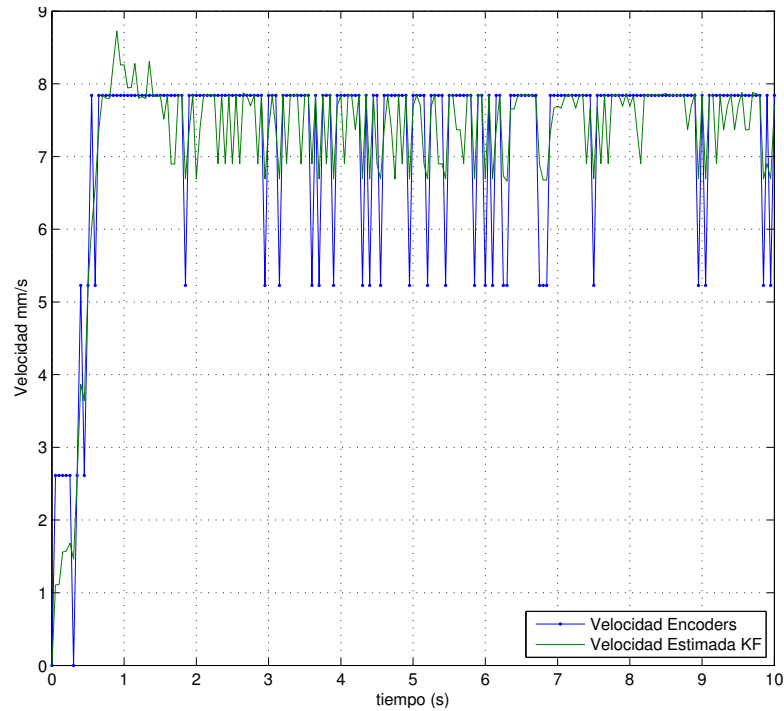


Figura 6.1: Velocidad rueda derecha e-puck con el KF

junto con la trayectoria de referencia que debía seguir el robot, para una segunda prueba con los parámetros de la odometría ajustados para un mejor desempeño. Observando en la figura 6.3(b) la trayectoria estimada por la odometría y la referencia (ambas calculadas internamente por el *e-puck*) se aprecia un buen comportamiento del algoritmo de control logrando seguir la trayectoria cuadrada (los giros en las esquinas son debidos al control por punto descentralizado). Sin embargo se muestra una diferencia entre la trayectoria estimada por el *e-puck* (odometría) y la medida mediante la cámara, principalmente en la última parte del trayecto. Esta no se debe propiamente al algoritmo de control sino al error de estimación de la posición del robot. De esto se concluye que el algoritmo de KF para las velocidades de las ruedas no es suficiente para tener un error cero en el posicionamiento del robot pero si mejora el comportamiento del robot. Para el caso de una referencia circular, la trayectoria seguida por el *e-puck* real se muestra en la figura 6.4(a) (se inicia en el centro del círculo) y en la figura 6.4(b) se muestra la trayectoria estimada por la cámara y el módulo de odometría junto con la trayectoria de referencia que debía seguir el robot. Se observa un buen comportamiento del algoritmo de control en la figura 6.4(b) logrando seguir la trayectoria circular. Además la prueba real mostrada en la figura 6.4(a) es muy similar al comportamiento descrito por la odometría. Esto se observa también en la figura 6.4(b) donde se aprecia que la diferencia entre la posición estimada y la real es menor que para

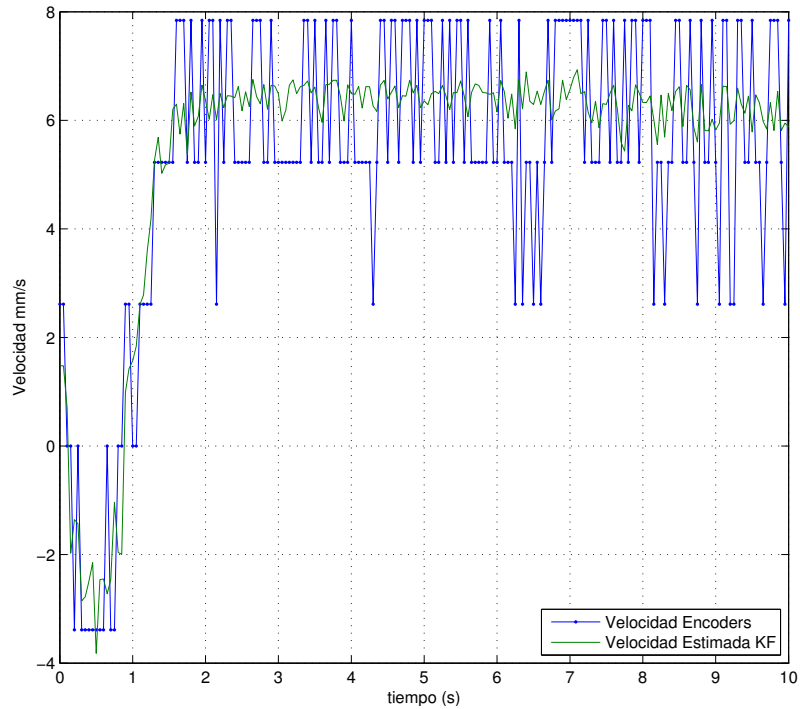


Figura 6.2: Velocidad rueda izquierda e-puck con el KF

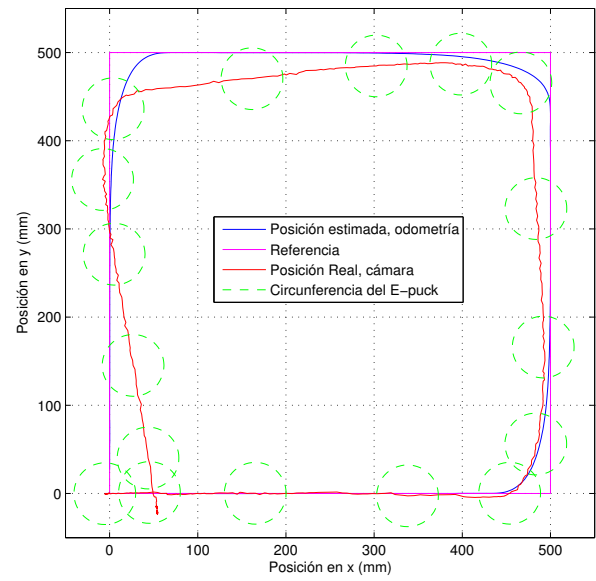
la prueba del cuadrado. Esto se debe a que en la trayectoria circular no hay cambios bruscos en la referencia a seguir lo que mejora el comportamiento del control. Finalmente se observa que el algoritmo de KF para las velocidades de las ruedas no es suficiente para tener un error cero en el posicionamiento del robot pero si mejora el comportamiento del robot y en este caso en particular el error en la posición es muy pequeño.

6.2. Pruebas con el LEGO NTX

Para el LEGO, al tener más recursos computacionales y contar con los sensores adicionales se lograron implementar los esquemas de fusión para la velocidad del robot (KF) y para la posición del robot (EKF). Para el EKF fue necesario simplificar la forma de calcular las operaciones del EKF utilizando múltiples variables intermedias para tener más líneas de código pero simple, de esta forma se consiguió implementar dentro del robot. Para mostrar el funcionamiento de los métodos de fusión, se establece que el robot siga una trayectoria de referencia pero esta vez utilizando la información del esquema de fusión con los filtros de Kalman para obtener la posición del robot que se utiliza en el control por punto descentralizado [48] y con un control tipo P para



(a) imagen compuesta

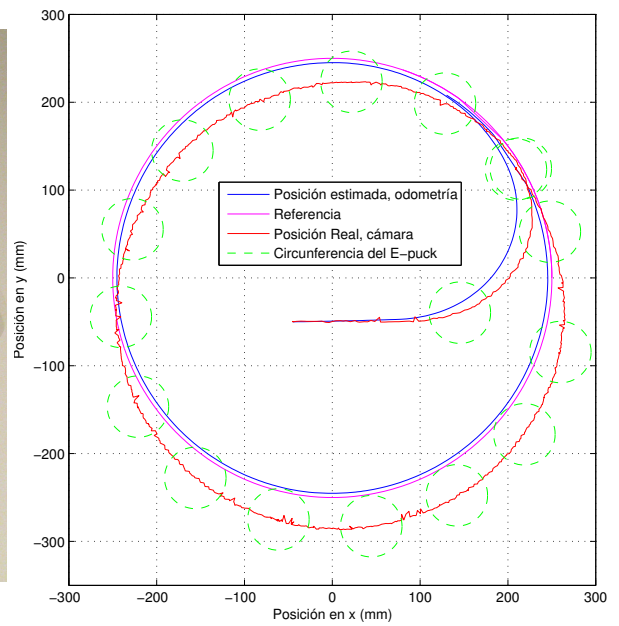


(b) Medición cámara cenital

Figura 6.3: E-puck: Trayectoria cuadrada con el KF de velocidades



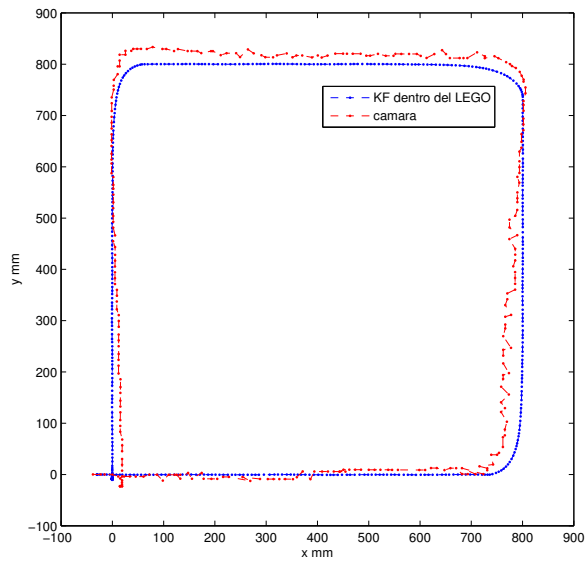
(a) imagen compuesta



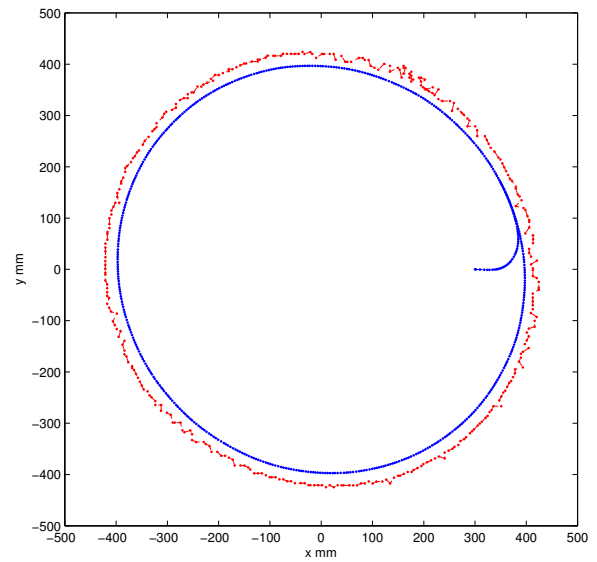
(b) Medición cámara cenital

Figura 6.4: E-puck: Trayectoria circular con el KF de velocidades

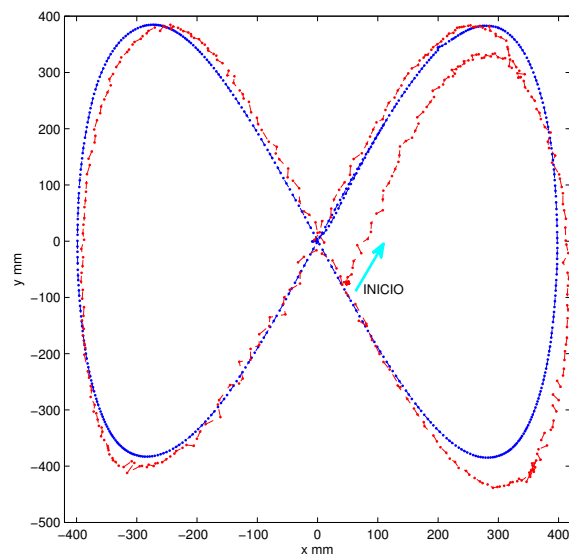
el accionamiento de los motores. La trayectoria real seguida por el robot se mide mediante una cámara cenital. En la figura 6.5 se muestra el comportamiento del KF para la velocidad del robot utilizando los acelerómetros para distintas trayectorias de referencia. En la figura 6.6 se repite la prueba de la trayectoria circular pero utilizando el modelo dinámico de las ruedas del robot (sin utilizar los acelerómetros) donde se observa un desempeño similar al caso con acelerómetros. En las figuras 6.7, 6.8 y 6.9 se muestra el funcionamiento interno del filtro (trayectoria cuadrada sin acelerómetros) en donde se observa la forma de la estimación del filtro para la velocidad angular, el ángulo de avance del robot y la velocidad lineal del robot. Por último se muestra en la figura 6.10 el filtro EKF implementado en el robot para una trayectoria circular. En todas las pruebas realizadas se observa como el funcionamiento del control de trayectoria mejora en gran medida al utilizar la estimación de la información provista por los esquemas de fusión de datos propuesto en lugar de utilizar únicamente la información de los encoder. Aún en los casos más complejos en donde el control de trayectoria toma cierto tiempo en converger se observa que al final se sigue adecuadamente la trayectoria coincidiendo la estimación de la posición del KF con la trayectoria medida por la cámara. Se logró además de estas pruebas implementar el control por eventos dentro del robot LEGO pero simulando el retardo de comunicación con la cámara y agregando una diferencia arbitraria (y de forma gradual) al final del recorrido para observar cómo se comporta el control de trayectorias. Se observa en la figura 6.11 que cuando se genera el evento el robot actúa correctamente moviéndose a partir de la posición actualizada y en la dirección correcta, esto hace que el robot se mueva «abriendo» más el cuadrado ya que en este caso se supuso que el robot en el momento del evento seguía un cuadrado más «cerrado». La implementación de la comunicación con la cámara queda fuera del propósito del presente trabajo y puede implementarse como trabajo futuro.



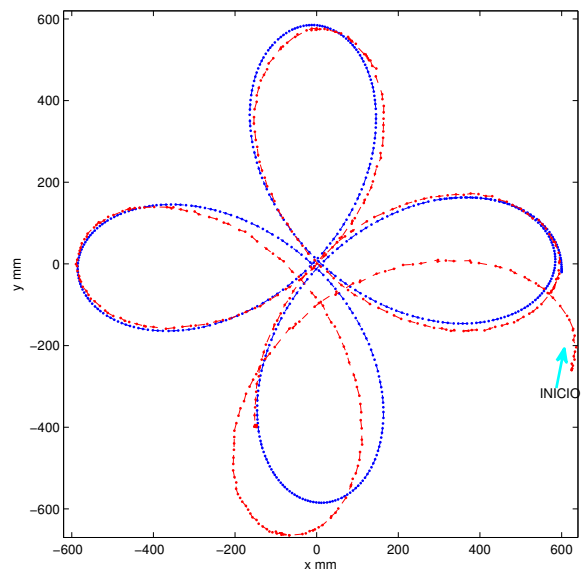
(a) Cuadrado



(b) Círculo



(c) Lemniscata



(d) Rosa Polar

Figura 6.5: Pruebas con el filtro KF velocidad robot implementado en el robot móvil LEGO NXT, con acelerómetros

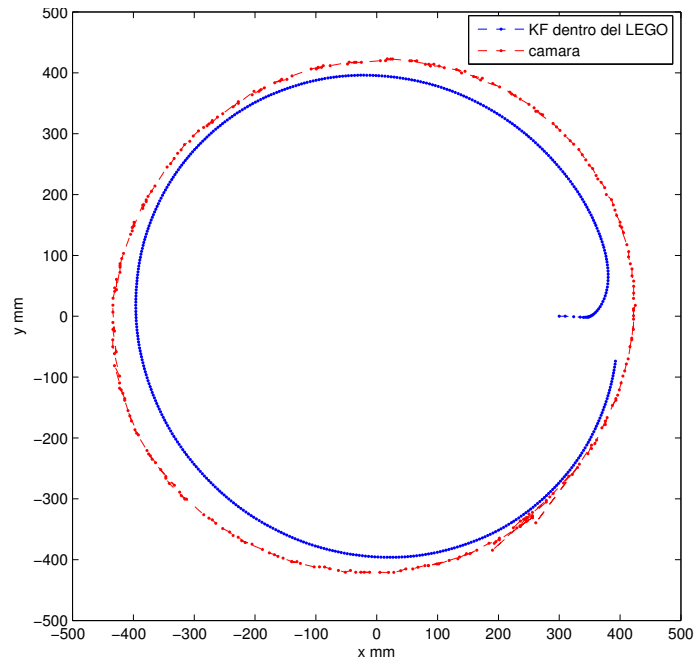


Figura 6.6: Pruebas con el filtro KF velocidad robot implementado en el robot móvil LEGO NXT, trayectoria circular sin acelerómetros

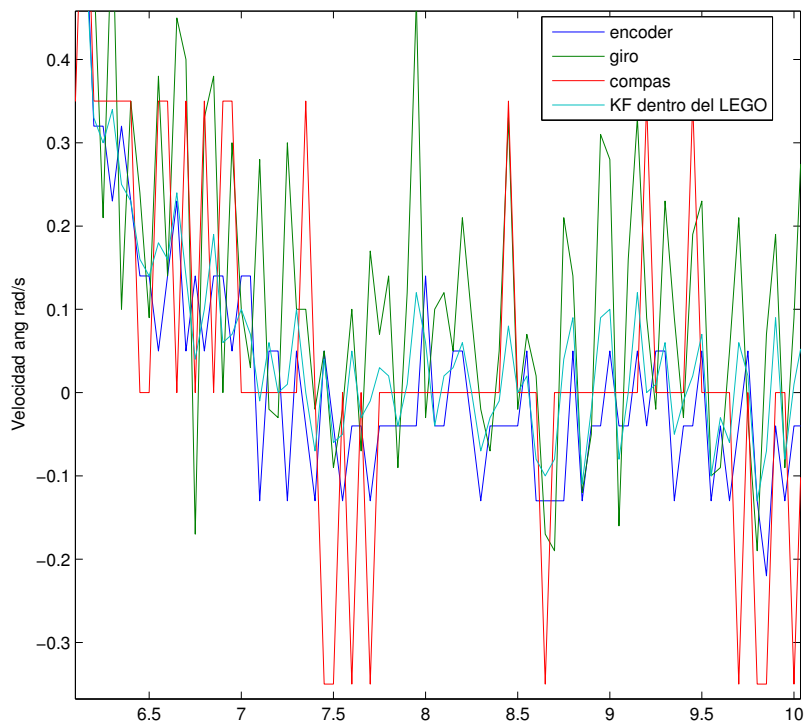


Figura 6.7: Estimación velocidad angular, KF dentro del LEGO

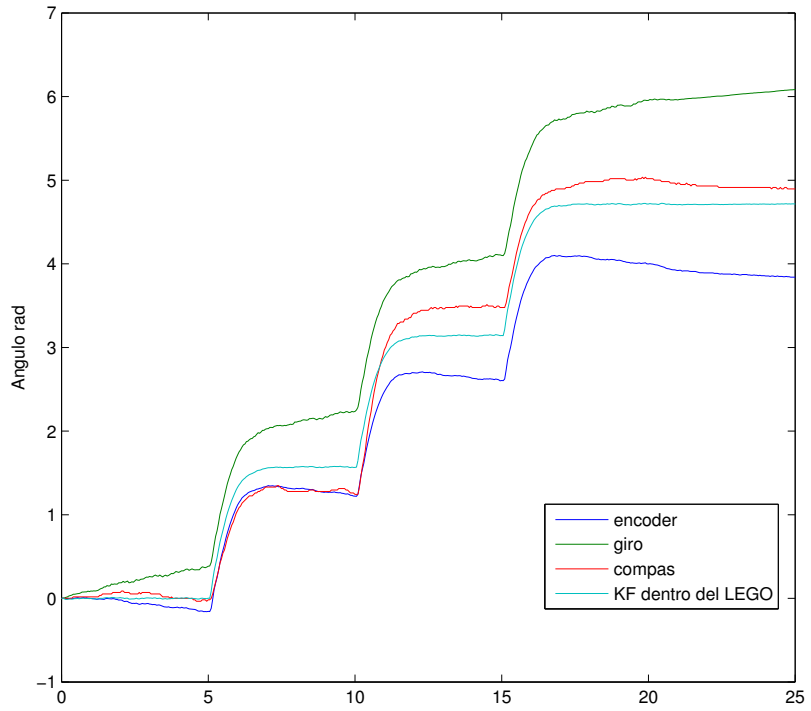


Figura 6.8: Estimación ángulo de avance, KF dentro del LEGO

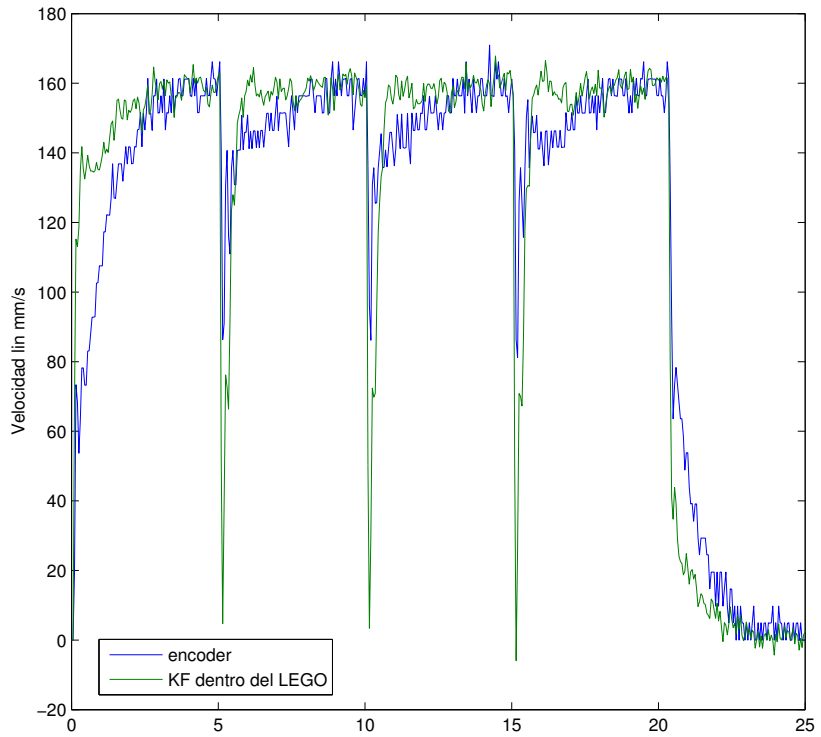


Figura 6.9: Estimación velocidad lineal, KF dentro del LEGO

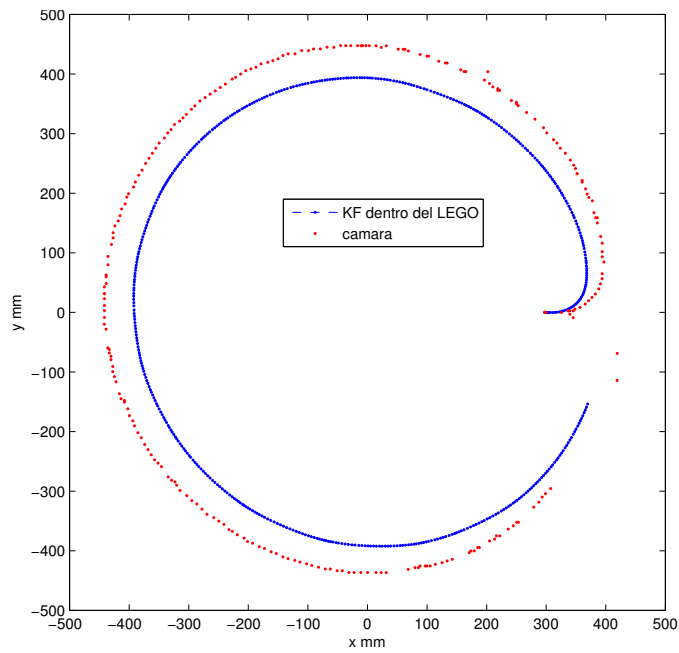


Figura 6.10: Pruebas con el filtro EKF posición del robot implementado en el robot móvil LEGO NXT, trayectoria circular con acelerómetros

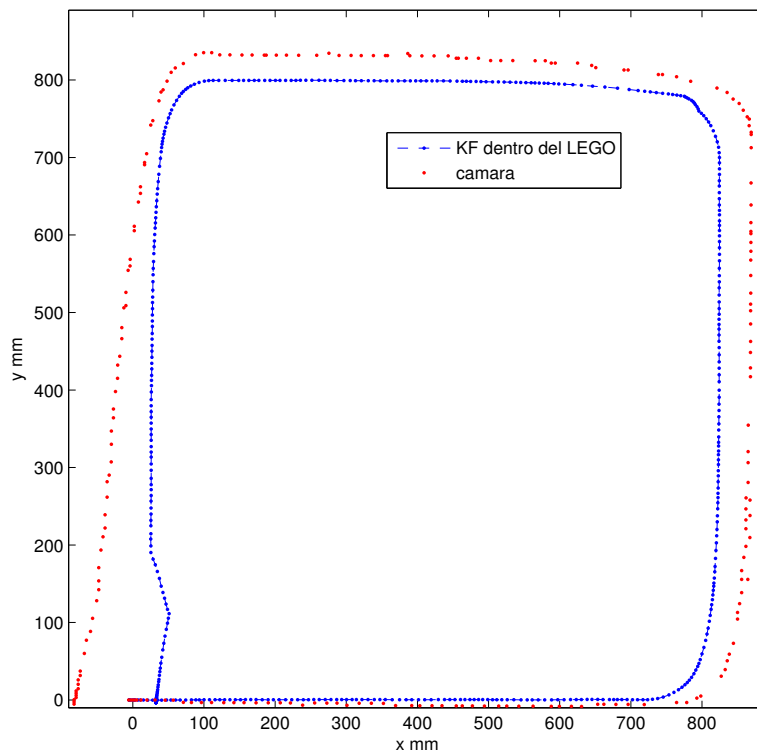


Figura 6.11: Control Por eventos simulando la comunicacion, KF dentro del LEGO

6.3. Pruebas en el sensor IG500N

Se utiliza el software de acceso al sensor desarrollado en [27] para obtener las variables calibradas del sensor pero sin tratar por el KF del fabricante. Específicamente se utiliza la información de las velocidades medidas por el sensor (integradas del acelerómetro en la INS y la estimada por el GPS) y el algoritmo del KF de velocidades para estimar la posición de un vehículo en movimiento en recorridos exteriores. El algoritmo de velocidades del KF no se modifica, únicamente se adapta para tomar en cuenta los sensores disponibles en el sensor, de esta forma en la ecuación (6.1) se muestra la matriz de mediciones utilizada para este sensor junto con los valores de las matrices Q y R utilizadas.

$$\begin{aligned} \hat{z}_k = \begin{bmatrix} v_{gps} \\ \omega_{gps} \\ \omega_{gyr} \\ \omega_{cmp} \end{bmatrix} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} + v \\ Q &= \begin{bmatrix} 0,5 & 0 \\ 0 & 0,5 \end{bmatrix}, R = \begin{bmatrix} 0,05 & 0 & 0 & 0 \\ 0 & 0,05 & 0 & 0 \\ 0 & 0 & 0,5 & 0 \\ 0 & 0 & 0 & 15 \end{bmatrix} \end{aligned} \quad (6.1)$$

Los valores de R son bajos para el GPS y altos para la unidad inercial ya que al no tener una medición local de la velocidad utilizando encoders la velocidad lineal no se puede medir con la precisión necesaria para un buen funcionamiento. En la figura 6.12 se muestra uno de los recorridos realizados, en esta se muestra la posición GPS medida en un mapa de Google Maps del robot. En la figura 6.13 se muestra el mismo recorrido pero mostrando el funcionamiento del KF velocidades implementado en MATLAB. Se observa un buen comportamiento al utilizar la información global de las velocidades lo cual ejemplifica la aplicabilidad del método desarrollado a distintas plataformas. En este caso particular si se desea utilizar con menos frecuencia la información del GPS se deberá adaptar unos encoders al vehículo que se desee monitorizar para mejorar la estimación de la velocidad lineal basada en el comportamiento inercial, además de considerar la utilización de un esquema por eventos similar al mostrado para el robot LEGO.

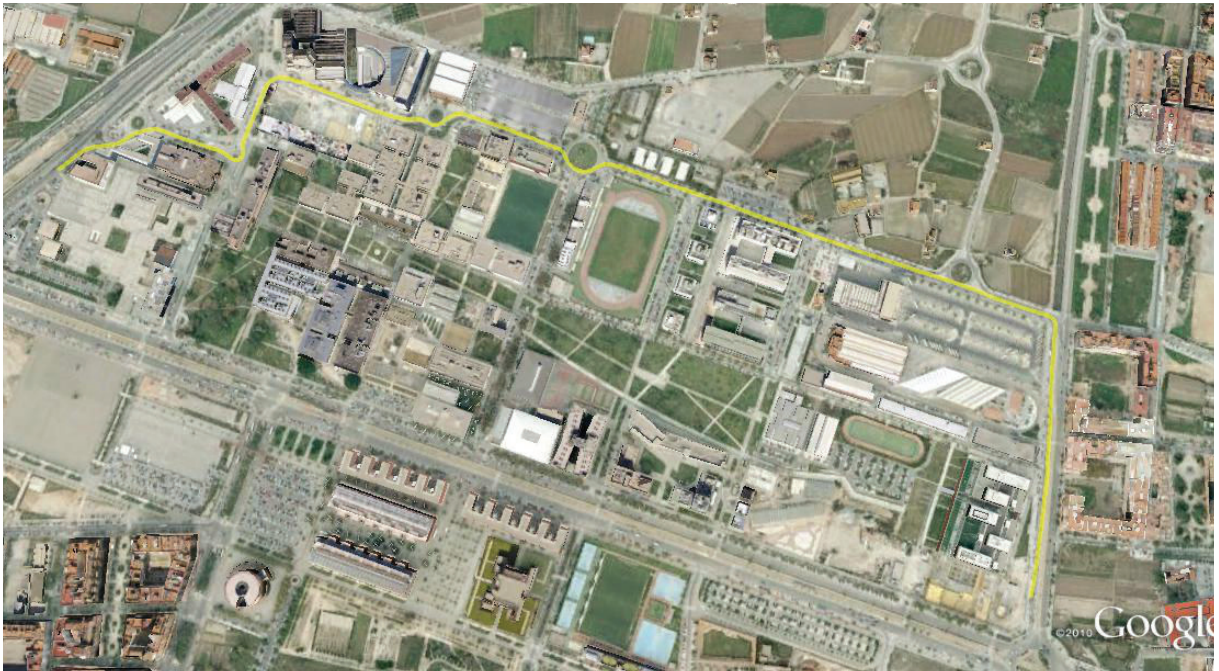


Figura 6.12: Prueba con el IG500N, recorrido GPS visto en Google Maps

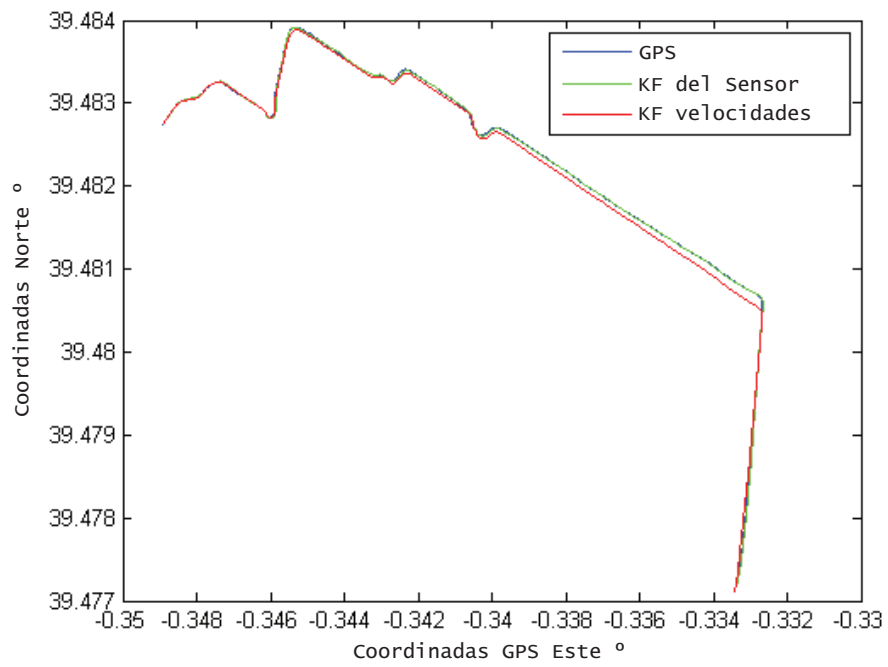


Figura 6.13: Prueba con el IG500N, comparación entre el KF del sensor y el KF de velocidades

Capítulo 7

Conclusiones

Se ha presentado múltiples esquemas de fusión sensorial basada en la utilización de filtros de Kalman los cuales según los resultados obtenidos funcionan de manera adecuada y eficiente, permitiendo implementarse en distintas plataformas y adaptar el esquema de fusión según los recursos computacionales disponibles en la plataforma utilizada.

La principal aportación del presente trabajo es el esquema de fusión basado en el filtro KF junto con las velocidades del robot ya que al utilizar el modelo del robot obtenido (como si fuera un sistema de partículas) junto con la información de los distintos sensores, se logra obtener un desempeño muy similar a las estrategias EKF y UKF más complejas (figuras 5.6 y 5.7) pero utilizando menos recursos computacionales. Esto se debe a que el KF tiene un modelo de dos estados por lo que el cálculo del filtro es más rápido que para los 5 estados utilizados en EKF o el UKF por lo que para el cálculo de la ganancia del KF la inversa tiene muchos menos términos que se deben agregar que al utilizar el EKF o UKF. Además si en los filtros EKF o UKF se agrega una medición sobre la posición además de la medición de la velocidad (que es la misma del KF) se deberán invertir matrices más grandes que en el KF lo que aumenta en gran medida la carga computacional del algoritmo.

De igual importancia es el modelo dinámico obtenido mediante la descomposición del robot como un sistema de partículas, ya que permite obtener la dinámica del robot al relacionar las aceleraciones de las ruedas con la aceleración local del robot. Esto sin necesidad de medir la fuerza o el torque de los motores lo que facilita su aplicación práctica ya que las aceleraciones de las ruedas pueden ser medidas con un acelerómetro o bien estimadas mediante el modelo dinámico de las ruedas.

Para disminuir el error de predicción observado en los resultados es necesario utilizar un sistema de posicionamiento global como un GPS o una cámara cenital para corregir la estimación de la posición global obtenida a partir de los sensores locales con la del sensor de información global. La actualización del valor se puede realizar de forma síncrona mediante intervalos regulares de tiempo o de manera asíncrona, tal como se plantea en el presente trabajo, bajo demanda del robot cuando el error de predicción normalizado exceda un determinado umbral. La ventaja de usar este segundo método es que se consigue de esta manera una mejor estimación de la posición y uso más óptimo de los recursos de comunicación (ya que únicamente se transmite en caso de ser necesario, reduciendo así el tráfico). Esto conllevará además un ahorro en el consumo de baterías, necesario para las comunicaciones inalámbricas, muy importante en sistemas con alimentación autónoma como son los robots móviles.

Tal y como se mostró en las pruebas implementadas si es posible ejecutar los esquemas de fusión propuestos dentro de las plataformas utilizadas. Esto es muy relevante ya que tal y como se mostró en los antecedentes, los métodos existentes en la literatura son muy complejos por lo que en general se implementan en un ordenador fuera del robot o simplemente se realizan simulaciones. En cambio los esquemas de fusión propuestos son eficientes en cuanto al uso de recursos y por lo tanto implementables dentro del robot lo cual evita los problemas asociados al retardo de comunicación que se tiene si el control del robot se ejecuta en un ordenador externo a este.

Capítulo 8

Trabajo Futuro

Se pueden implementar distintas mejoras y ampliaciones a los métodos propuestos con el fin de continuar el estudio presentado. Por ejemplo se puede aplicar los esquemas de fusión sensorial a un problema de localización en exteriores realizando la adaptación de un robot volador tipo helicóptero o flotador dirigido el cual, si es equipado con una unidad de procesamiento de alta capacidad, podría medir su posición mediante GPS y corregir la posición de robots móviles en tierra al medir su posición relativa utilizando una cámara apuntada hacia el plano de movimiento de los robots. Para esto se deben compensar las perturbaciones del viento y cambios atmosféricos que afectan al robot volador. Este robot podría ser el guía en ejercicios de coordinación de robots corrigiendo la posición de los robots terrestres y asignando los objetivos de control al grupo. Entre los principales problemas estaría la necesidad de una gran capacidad de cómputo para poder procesar con buena precisión las imágenes de la cámara, la necesidad de tomar en cuenta la altura y movimiento de la cámara en la obtención de la posición de los vehículos terrestres y el diseño mecánico para poder levantar el peso de los sensores y del robot aunque para solucionar esto se puede modificar algún robot comercial como por ejemplo el AR-Drone dotándolo de un GPS y de más capacidad de cómputo.

Los esquemas de fusión también pueden ser utilizados en un método de localización distribuida, por ejemplo si un robot actualizó su posición con la información global, este puede actualizar la posición de otros robots (o grupos de robots) si se conoce la distancia entre estos. Por ejemplo, si un robot LEGO actualizó su posición con la cámara cenital, podría informar a otros robots de menos recursos como son los e-puck comunicándoles su posición corregida. Si los e-pucks reciben esta información y pueden medir su distancia relativa al LEGO entonces pueden corre-

gir su estimación local. Si la comunicación con más de un e-puck vía bluetooth no es eficiente para muchos robots, se puede modificar el esquema al comunicar únicamente un robot LEGO con un e-puck vía bluetooth para corregir su posición y luego este e-puck se comunicará con otros robots e-puck para corregir sus posiciones. Esto requiere aumentar la capacidad de comunicación del e-puck utilizando zigbee para comunicarse entre ellos o bien utilizando la tarjeta «range and bearing board» para los e-puck la cual mide la distancia relativa entre dos e-puck y permite la comunicación por IR.

Finalmente se puede ampliar los esquemas de fusión al considerar la utilización de sensores avanzados como una cámara para realizar localización por flujo óptico, con la opción de usarlo a modo de sensor de cambio de velocidad de la imagen del suelo o bien con la cámara apuntando al entorno (ver métodos actuales en [24]). La dificultad principal es que hay que desarrollar o bien el sensor a modo de «ratón óptico» para obtener el flujo óptico del movimiento del suelo, aunque en este caso se cuenta con algunos ejemplos en la literatura (ver por ejemplo el desarrollo global del sensor presentado en [10] o en [43]) o bien usar librerías de procesamiento de visión (por ejemplo OpenCV) para el caso del movimiento del entorno, sin embargo esto requiere mucho procesamiento por lo que se debe utilizar una plataforma con alta capacidad de cómputo.

Bibliografía

- [1] *IG-500N GPS aided AHRS: User Manual Revision: 9 – 19 November, 2009 SBG Systems.*
- [2] *Información en línea del LEGO NXT (2011), disponible en: <http://www.mindsensors.com/> y <http://www.hitechnic.com/> motores del LEGO NXT: <http://www.philohome.com/motors/motorcomp.htm>.*
- [3] *Sitio web del E-puck. [Online]. Disponible: <http://www.e-puck.org/>, 2010.*
- [4] Albagul, A., W. Martono y R. Muhida: *Dynamic Modelling and Adaptive Traction Control for Mobile Robots*. International Journal of Advanced Robotic Systems, 1.3:149 – 154, 2005. http://www.intechopen.com/books/show/title/cutting_edge_robotics.
- [5] Antonelli, G., S. Chiaverini y G. Fusco: *A calibration method for odometry of mobile robots based on the least-squares technique: theory and experimental validation*. IEEE Transactions on Robotics, 21.5:994 – 1004, 2005.
- [6] Arras, K. O.: *An Introduction To Error Propagation: Derivation, Meaning and Examples of Equation $C_y = F_x C_x F_x^T$* . Informe técnico., Autonomous Systems Lab, Institute of Robotic Systems, Swiss Federal Institute of Technology Lausanne (EPFL), 1998.
- [7] Attia, H. A.: *Dynamic model of multi-rigid-body systems based on particle dynamics with recursive approach*. Journal of Applied Mathematics, 2005:365–382, 10.1155/JAM.2005.365 2005.
- [8] Azizi, F. y N. Houshangi: *Sensor integration for mobile robot position determination*. Systems, Man and Cybernetics, 2003. IEEE International Conference on, 2:1136 – 1140, 2003.
- [9] Bar-Shalom, Y.: *Update with Out-of-Sequence Measurements in Tracking: Exact Solution*. IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS VOL. 38, NO. 3 JULY 2002, 38.3:769–778, 2002.

- [10] Bell, S.: *High-Precision Robot Odometry Using an Array of Optical Mice*. Informe técnico., Oklahoma Christian University, 2011. http://botsnlinux.net/school_projects.php.
- [11] Bozic, S.: *Digital and Kalman filtering: an introduction to discrete-time filtering and optimum linear estimation*. Halsted Press, 1994.
- [12] Brown, R. G. y P. Y. C. Hwang: *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises and Solutions*. John Wiley & Sons, 3rd edition ed., 1997.
- [13] Campion, G., G. Bastin y B. Dandrea-Novel: *Structural properties and classification of kinematic and dynamic models of wheeled mobile robots*. Robotics and Automation, IEEE Transactions on, 12(1):47–62, feb 1996.
- [14] Carona, R., A. P. Aguiar y J. Gaspar: *Control of Unicycle Type Robots: Tracking, Path Following and Point Stabilization*. En *IV Jornadas de Engenharia Electrónica e Telecomunicações e de Computadores*, pp180-185 November 2008, Lisbon, Portugal, 2008.
- [15] Cetto, J. A. y A. Sanfeliu: *Environment Learning for Indoor Mobile Robots, A Stochastic State Estimation Approach to Simultaneous Localization and Map Building*. Springer, 2006.
- [16] Chee, W.: *Yaw Rate Estimation Using Two 1-Axis Accelerometers*. Proceedings of the American Control Conference, págs. 423 – 428, 2005.
- [17] Chen, R., H. Zhao y B. Xiao: *Self-localization of mobile robot based on monocular and extended kalman filter*. Electronic Measurement Instruments, 2009. ICEMI '09. 9th International Conference on, págs. 2–450 –2–454, 2009.
- [18] Chisci, G. B. L., F. Chiti, R. Fantacci y S. Menci: *Localization of a Swarm of Mobile Agents via Unscented Kalman Filtering*. IEEE International Conference on Communications, 2009. ICC '09., págs. 1–5, 2009.
- [19] Chong, K. S. y L. Kleeman: *Accurate Odometry and Error Modelling for a Mobile Robot*. IEEE International Conference on Robotics and Automation, págs. 2783–2788, 1997.
- [20] Chui, C. K. y G. Chen: *Kalman Filtering with Real-Time Applications*. Springer, 4th edition ed., 2009.
- [21] Cong, T. H., Y. J. Kim y M. T. Lim: *Hybrid Extended Kalman Filter-based localization with a highly accurate odometry model of a mobile robot*. En *Control, Automation and Systems, 2008. ICCAS 2008. International Conference on*, págs. 738 –743, oct 2008.
- [22] Cruz, C. D. L. y R. Carelli: *Dynamic model based formation control and obstacle avoidan-*

- ce of multi-robot systems*. Robotica, Cambridge University Press, 26.3:345–356, 2008.
- [23] Das, C.: *Calibration of Kinematic Odometric Parameters for Differential Drive Mobile Robots: An Overview*. Informe técnico., RMSLab, Robotics and neuro-Mechanical Systems Laboratory, University of Arizona, 2010.
- [24] DeSouza, G. N. y A. C. Kak: *Vision for Mobile Robot Navigation: A Survey*. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, NO. 2., 24 NO. 2:237–267, 2002.
- [25] Glenn D. White, R. M. B. y V. N. Krovi: *Dynamic redundancy resolution in a nonholonomic wheeled mobile manipulator*. Robotica, Cambridge University Press, 25.2:147–156, 2007.
- [26] Grewal, M. S. y A. P. Andrews.: *Kalman Filtering: Theory and Practice Using Matlab*. John Wiley & Sons, 2001.
- [27] Groh, B.: *Analysing of advanced sensors and development of a user interface with additional filter options*. Informe técnico., Institute of Control Systems and Industrial Computing, Polytechnic University of Valencia Department of Sensor Technology Friedrich-Alexander University Erlangen-Nuremberg, 2011.
- [28] Guerrero, R. V.: *DINÁMICA Y CONTROL DE SILLAS DE RUEDAS ROBÓTICAS*, *Avances en Ingeniería Electrónica*, cap. 11, págs. 151–164. UAM-UdG (México, DF), 2010.
- [29] Han, Q. S. J. Q. J.: *An Adaptive UKF Algorithm and Its Application in Mobile Robot Control*. Proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics, págs. 1117–1122, 2006.
- [30] Hartikainen, J. y S. Särkkä: *Optimal filtering with Kalman filters and smoothers – a Manual for Matlab toolbox EKF/UKF*. Department of Biomedical Engineering and Computational Science, Helsinki University of Technology, Espoo, Finland, Available: <http://www.lce.hut.fi/research/mm/ekfukf/>, 1.2 ed., Feb 2008.
- [31] Hogg, R. W., A. L. Rankin, S. I. Roumeliotis, M. C. McHenry, D. M. Helmick, C. F. Bergh y L. Matthies: *Algorithms and Sensors for Small Robot Path Following*. En *IEEE International Conference on Robotics and Automation*, Washington D.C., 2002.
- [32] Houshangi, N. y F. Azizi: *Mobile robot position determination using data from gyro and odometry*. Electrical and Computer Engineering, 2004. Canadian Conference on, 2:719–722, 2004.
- [33] Houshangi, N. y F. Azizi: *Accurate mobile robot position determination using unscented*

- Kalman filter*. Canadian Conference on Electrical and Computer Engineering, págs. 846 – 851, 2005.
- [34] Houshangi, N. y F. Azizi: *Mobile Robot Position Determination Using Data Integration of Odometry and Gyroscope*. En *World Automation Congress (WAC), Budapest, Hungary, 2006*.
- [35] Jetto, L., S. Longhi y G. Venturini: *Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots*. *IEEE Transactions on Robotics and Automation*, 15.2:219 – 229, 1999.
- [36] Julier, S. y J. Uhlmann: *A General Method for Approximating Nonlinear Transformations of Probability Distributions*. Informe técnico., 1996.
- [37] Julier, S. y J. Uhlmann: *A new extension of the Kalman filter to nonlinear systems*. En *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, Orlando, FL, 1997*.
- [38] Julier, S. y J. Uhlmann: *Unscented filtering and nonlinear estimation*. *Proceedings of the IEEE*, 92:401 – 422, 2004.
- [39] Julier, S., J. Uhlmann y H. Durrant-Whyte: *A new approach for filtering nonlinear systems*. *American Control Conference, 1995*. *Proceedings of the*, 3:1628 –1632, 1995.
- [40] Julier, S., J. Uhlmann y H. Durrant-Whyte: *A new method for the nonlinear transformation of means and covariances in filters and estimators*. *Automatic Control, IEEE Transactions on*, 45:477 –482, mar 2000.
- [41] Kalman, R. E.: *A New Approach to Linear Filtering and Prediction Problems*. *Transactions of the ASME–Journal of Basic Engineering*, 82:35–45, 1960. <http://www.cs.unc.edu/~welch/kalman/kalmanPaper.html>.
- [42] Kim, J., Y. Kim y S. Kim: *An accurate localization for mobile robot using extended Kalman filter and sensor fusion*. En *Neural Networks, 2008. IJCNN 2008. IEEE International Joint Conference on*, págs. 2928 –2933, june 2008.
- [43] Kim, S. y S. Lee: *Optical Mouse Array Position Calibration for Mobile Robot Velocity Estimation*. *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*, págs. 1167 –1172, 2008.
- [44] Kiriy, E. y M. Buehler: *Three-state Extended Kalman Filter for Mobile Robot Localization*. Informe técnico TR-CIM 05.06, McGill University, Montreal, Canada, April 2002.
- [45] Ko, S. il, J. suk Choi y B. hoon Kim: *Performance Enhancement of Indoor Mobile Localization System using Unscented Kalman Filter*. *SICE-ICASE, 2006*. *International Joint*

- Conference, Bexco, Busan, Korea, págs. 1355 – 1360, 2006.
- [46] Lee, J. M., D. H. Lee, H. An, N. Huh, M. K. Kim y M. H. Lee: *Ultrasonic satellite system for the positioning of mobile robots*. Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE, 1:448 – 453, 2004.
- [47] Liggins, M. E., D. L. Hall y J. Llinas (eds.): *Handbook of Multisensor Data Fusion: Theory and Practice*. CRC Press, Taylor & Francis Group, second edition ed., 2009. <http://www.crcpress.com/product/isbn/9781420053081>.
- [48] Marín, L., M. Vallés, Ángel Valera y P. Albertos: *Control de trayectorias en el robot móvil e-puck*. XXXI Jornadas Automatica JAEN, 2010.
- [49] Martinelli, A. y R. Siegwart: *Estimating the odometry error of a mobile robot during navigation*. European Conference on Mobile Robots (ECMR 2003), 2003.
- [50] Martínez, V. F. M., G. Gil-Gómez y A. G. Cerezo: *Modelado cinemático y dinámico de un robot móvil omni-direccional*. XXIV Jornadas Automatica JAEN, 2003.
- [51] Michel, O., F. Rohrer, N. Heiniger y wikibooks contributors: *Cyberbotics' Robot Curriculum*. Cyberbotics Ltd. and Wikibooks contributors, http://en.wikibooks.org/wiki/Cyberbotics%27_Robot_Curriculum, 2009.
- [52] Mondada, F., M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klapotocz, S. Magnenat, J. C. Zufferey, D. Floreano y A. Martinoli: *The e-puck, a Robot Designed for Education in Engineering*. Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, 1:59–65, 2009.
- [53] Moreno, V.M. y A. Pigazo (eds.): *Kalman Filter: Recent Advances and Applications*. In-Tech, 2009. http://www.intechopen.com/books/show/title/kalman_filter_recent_adavnces_and_applications.
- [54] Myung, H., H. K. Lee, K. Choi, S. W. Bang, Y. B. Lee y S. R. Kim: *Constrained Kialman Filter for Mobile Robot Localization with Gyroscope*. Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems Beijing, China, págs. 442 – 447, 2006.
- [55] Nagatani, K., D. Endo y K. Yoshida: *Improvement of the Odometry Accuracy of a Crawler Vehicle with Consideration of Slippage*. IEEE International Conference on Robotics and Automation, págs. 2752–2757, 2007.
- [56] Nevalainen, E.: *Accelerometer configurations for a gyroscope free inertial navigation system*. Informe técnico., HELSINKI UNIVERSITY OF TECHNOLOGY, Faculty of Infor-

- mation and Natural Sciences, Department of Mathematics and Systems Analysis, 2008. www.sal.tkk.fi/publications/pdf-files/enev08.pdf.
- [57] Noga, S.: *Kinematics and dynamics of some selected two-wheeled mobile robots*. Archives of Civil and Mechanical Engineering, 6:55–70, 2006. www.acme.pwr.wroc.pl/abstract.php?p_id=112.
- [58] Piedrahita, G. A. y D. M. Guayacundo: *Evaluation of Accelerometers as Inertial Navigation System for Mobile Robots*. IEEE 3rd Latin American Robotics Symposium, LARS '06., 84 - 90, 2006.
- [59] Reina, G., A. Vargas, K. Nagatani y K. Yoshida: *Adaptive Kalman Filtering for GPS-based Mobile Robot Localization*. En *Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on*, págs. 1 –6, 2007.
- [60] Sarkar, N., X. Yun y R. V. Kumar: *Control of Mechanical Systems with Rolling Constraints: Application To Dynamic Control of Mobile Robots*. Informe técnico., University of Pennsylvania, 1992. http://repository.upenn.edu/cis_reports/504/.
- [61] Siegwart, R. y I. Nourbakhsh: *Introduction to autonomous mobile robots*. The MIT Press Cambridge, Massachusetts London, England, 2004.
- [62] Simon, D.: *Optimal State Estimation: Kalman, H_∞ , and Nonlinear Approaches*. John Wiley & Sons, 2006.
- [63] Tan, C. W. y S. Park: *Design of Accelerometer-Based Inertial Navigation Systems*. IEEE Transactions on Instrumentation and Measurement, 54.6:2520 – 2530, 2005. 10.1109/TIM.2005.858129.
- [64] Tornero, J. y L. Armesto: *Control Óptimo*. Universidad Politécnica de Valencia, 2007.
- [65] Trimpe, S. y R. D'Andrea: *Accelerometer-based tilt estimation of a rigid body with only rotational degrees of freedom*. IEEE International Conference on Robotics and Automation (ICRA), págs. 2630 – 2636, 2010.
- [66] Valera Ángel, M. Weiss, M. Vallés y J. L. Diez: *BLUETOOTH-NETWORKED TRAJECTORY CONTROL OF AUTONOMOUS VEHICLES*. Eight IFAC Symposium on Cost Oriented Automation, 8, 2007.
- [67] Villa, M. V., E. A. Bricaire y R. O. Guerrero: *Discrete-Time Modeling and Path-Tracking for a Wheeled Mobile Robot*. Computación y Sistemas, 13.2:142–160, 2009.
- [68] Wan, E. y R. Van Der Merwe: *The unscented Kalman filter for nonlinear estimation*. En *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000*.

AS-SPCC. The IEEE 2000, 2000.

- [69] Ward, C. C. y K. Iagnemma: *A Dynamic-Model-Based Wheel Slip Detector for Mobile Robots on Outdoor Terrain*. IEEE TRANSACTIONS ON ROBOTICS, 24.4:821 – 831, 2008.
- [70] Welch, G. y G. Bishop: *An Introduction to the Kalman Filter*, 2007. <http://www.cs.unc.edu/~welch/kalman/>.
- [71] Zhou, B., Y. Peng y J. Han: *UKF Based Estimation and Tracking Control of Nonholonomic Mobile Robots with Slipping*. Proceedings of the 2007 IEEE International Conference on Robotics and Biomimetics, págs. 2058 – 2063, 2007.
- [72] Zohar, I., A. Ailon y R. Rabinovici: *Mobile robot characterized by dynamic and kinematic equations and actuator dynamics: Trajectory tracking and related application*. Robotics and Autonomous Systems, 59.6:343 – 353, 2011.