UNIVERSITAT POLITÈCNICA DE VALÈNCIA
DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

LOCALIZACIÓN DE ROBOTS MÓVILES DE RECURSOS LIMITADOS BASADA EN FUSIÓN SENSORIAL POR EVENTOS

TESIS DOCTORAL PRESENTADA POR:
Leonardo José Marín Paniagua

DIRIGIDA POR:
Dra. Marina Vallés Miquel
Dr. Ángel Valera Fernández

Valencia, Junio 2014

A Irene.

A mi familia.

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 de 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 esta información es tema central de la tesis en la cual se exponen distintos algoritmos de fusión, desarrollados específicamente para robots móviles con recursos de computación limitados navegando de forma individual o en grupos heterogéneos.

Utilizando modelos dinámicos de robots del tipo diferencial y Ackerman en conjunto con algoritmos de fusión basados en el filtro de Kalman se realiza una estimación local de la postura del robot utilizando sensores inerciales, la cual se actualiza con información global mediante una corrección basada en eventos. Este tipo de corrección da lugar a una nueva familia de filtros de Kalman, la cual permite un ahorro en recursos computacionales y de comunicación durante el proceso de localización pero con una precisión similar a esquemas más complejos de fusión y con evolución acotada del error, siendo este el principal aporte de la presente tesis.

Los algoritmos propuestos con fusión basada en eventos para robots individuales se extienden para el caso de localización cooperativa de grupos de robots heterogéneos, modificando la actualización por eventos para incorporar la información de la postura relativa de distintos robots cercanos entre sí en un filtro de Kalman distribuido. Utilizando la medición de la distancia y ángulo entre los robots, junto con sus posturas transmitidas mediante un sistema de comunicación gestionado por agentes, se realiza la corrección de la estimación local, lo cual permite nuevamente una estimación de la postura con precisión adecuada y error acotado, con costo computacional y de comu-

nicación reducido. Este método además permite realizar una fusión sensorial inteligente, tomando en cuenta únicamente la información relativa más fiable, descartando mediciones poco precisas o procedentes de robots muy alejados.

Los algoritmos propuestos han sido probados extensivamente mediante simulación y en distintas plataformas móviles en las cuales se observa el buen desempeño de los mismos. Se presentan además ejecuciones de larga duración que comprueban la estabilidad y robustez del método en largas distancias. Adicionalmente se analiza, en el caso del método de localización cooperativa, la relación de compromiso entre la covarianza del error de estimación y el uso del ancho de banda al utilizar el algoritmo propuesto.

Se exponen finalmente las posibilidades de ampliación del presente trabajo en áreas como mapeo y localización simultánea, ajuste del método a plataformas omnidireccionales, implementación en distintos grupos heterogéneos de robots y el estudio de distintas definiciones alternativas de eventos y su efecto en el desempeño de la localización.

Palabras clave: Fusión de datos, filtros de Kalman, filtros de Kalman distribuidos, odometría, localización de robots, robots móviles, sistemas basados en eventos, estimación basada en eventos, comunicación basada en eventos, localización cooperativa, sistemas sensoriales, fusión sensorial multirobot, *LEGO NXT*.

Subvenciones: 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.

RESUM

Un dels aspectes essencials a la robòtica mòbil és l'obtenció i processament de la informació relativa a la localització del robot a l'espai de moviment, amb l'objectiu d'emprar-la per a generar els moviments desitjats del robot. Per a aquest fi es busca emprar la major quantitat possible de fonts d'informació per tal que es puga corregir els errors de posició associats a la presència de soroll a les mesures del robot.

La fusió d'aquesta informació és tema central de la tesi a la qual s'exposen diferents algorismes de fusió, desenvolupats específicament per a robots mòbils amb recursos de computació limitats navegant de forma individual o a grups heterogenis.

Emprant models dinàmics de robots del tipus diferencial i Ackerman en conjunt amb algorismes de fusió basats al filtre de Kalman es fa una estimació local de la postura del robot emprant sensors inercials, la qual s'actualitza amb informació global mitjançant una correcció basada en esdeveniments. Aquest tipus de correcció dóna lloc a una nova família de filtres de Kalman, la qual permet un estalvi en recursos computacionals i de comunicació durant el procés de localització però amb una precisió similar a esquemes més complexos de fusió i amb evolució acotada de l'error, sent aquesta la principal contribució de la present tesi.

Els algorismes proposats amb fusió basada en esdeveniments per a robots individuals s'amplien per al cas de localització cooperativa de grups de robots heterogenis, modificant l'actualització per esdeveniments per a incorporar la informació de la postura relativa de distints robots propers entre ells a un filtre de Kalman distribuït. Emprant la mesura de la distància i angle entre els robots, junt amb les seues postures transmeses mitjançant un sistema de comunicació gestionat per agents, es fa la correcció de l'estimació local, la qual cosa permet novament una estimació de la postura amb precisió adequada i error acotat, amb cost computacional i de comunicació reduït.

Aquest mètode a més a més permet fer una fusió sensorial intel·ligent, tenint en compte únicament la informació relativa més fiable, descartant mesures poc precises o procedents de robots molt allunyats.

Els algorismes proposats han estat provats extensivament mitjançant simulació i a distintes plataformes mòbils a les quals s'observa el bon acompliment d'objectius per part d'aquests. Es presenten a més a més execucions de llarga durada que comproven l'estabilitat i robustesa del mètode en llargues distàncies. Addicionalment s'analitza, al cas del mètode de localització cooperativa, la relació de compromís entre la covariància de l'error d'estimació i l'ús d'ample de banda en emprar l'algorisme proposat.

S'exposa finalment les possibilitats d'ampliació del present treball a àrees com mapeig i localització simultània, ajust del mètode a plataformes omnidireccionals, implementació a distints grups heterogenis de robots i l'estudi de distintes definicions alternatives d'esdeveniments i el seu efecte a l'acompliment de la localització.

Paraules clau: Fusió de dades, filtres de Kalman, filtres de Kalman distribuïts, odometria, localització de robots, robots mòbils, sistemes basats en esdeveniments, estimació basada en esdeveniments, comunicació basada en esdeveniments, localització cooperativa, sistemes sensorials, fusió sensorial multirobot, *LEGO NXT*.

Subvencions: Subvencions: Aquest treball ha estat finançat parcialment pel Ministerio de Ciencia e Innovación de España baix els projectes FEDER/CICYT d'investigació DPI2008-06737-C02-01 i DPI2010-20814-C02-02. A més a més s'agraeix el suport financer per part de la Universidad de Costa Rica.

ABSTRACT

Localization, the process of obtaining and processing the information of the robot pose in the movement space, is a fundamental issue in autonomous mobile robots as this information is required to determine the desired robot movements. For this task, all the available sensor sources must be used in order to reduce the error in the pose estimation, as the measurements are subject to noise and nonlinearities.

The formulation of a new sensor fusion framework that improves the localization of mobile robots with limited computational resources, navigating individually in the environment or in assembled heterogeneous groups, is the central and fundamental issue of this thesis.

The proposed algorithms use the dynamic model of an Ackermann and a differential steering mobile robots, along with the Kalman Filter fusion scheme to perform the local pose estimation using the inertial sensors measurements, which is updated with the global sensor information on an event based schedule. This type of correction produces a new kind of Kalman Filters that reduce the resources (execution time and bandwidth) needed to perform the sensor fusion to localize the robot, but with similar performance when compared to the more complex fusion methods and with bounded error evolution, being this the main contribution of this thesis.

The proposed event based algorithms for individual robots are extended to include the multirobot cooperative localization case for heterogeneous groups of mobile robots. This is done by modifying the event based update to include the relative pose information of the various nearby neighbors in a distributed Kalman Filter. The relative measurements of the range and bearing between robots is used along with the transmitted pose estimation of each robot (using agent managed communications) to update the local pose estimation. Once again this approach allows an efficient robot localization, with improved accuracy and bounded error while reducing the use of the platform

bandwidth and computational resources. This method also performs a smart sensor fusion, as it only takes into account the most reliable relative sensor information, discarding inaccurate measurements or the ones delivered from distant robots.

The proposed algorithms have been extensively tested through simulations and by implementation in several mobile platforms, which shows the correct performance of the methods. Also, the several long walk tests performed reflect the stability and robustness of the algorithms when working in long distances. In addition, the tradeoff between the error covariance and the bandwidth usage is analyzed for the proposed cooperative localization method.

Finally, the several extension and continuation possibilities of the present work are exposed in areas such as the simultaneous localization and mapping (SLAM), the extension to omnidirectional platforms, the implementation in different heterogeneous robot groups, the alternative definitions for the event required by the proposed methods and their effect in the localization performance.

Keywords: Sensor fusion, Kalman filtering, distributed Kalman filtering, odometry, robot localization, pose estimation, mobile robots, event based systems, event based estimation, event based communication, cooperative localization, robot sensing systems, multirobot sensor fusion, *LEGO NXT*.

Acknowledgements: This work has been partially funded by FEDER-CICYT projects with references DPI2008-06737-C02-01, DPI2010-20814-C02-02 and DPI2011-28507-C02-01 financed by Ministerio de Ciencia e Innovación (Spain). Also, the financial support from the University of Costa Rica is greatly appreciated.

Agradecimientos

Completar con éxito la amplia investigación requerida para obtener el doctorado sería una tarea imposible de realizar sin contar con la ayuda de múltiples personas e instituciones, que, durante todos estos años de arduo trabajo, me han brindado de una u otra forma su ayuda, apoyo y colaboración desinteresada.

Primeramente y como no podría ser de otra forma, quisiera expresar mi profundo agradecimiento a mis directores de investigación y tesis, Dra. Marina Vallés Miquel y Dr. Ángel Valera Fernández, por toda la ayuda, guía, paciencia, consejos y apoyo absoluto desde el primer día, tanto en los asuntos académicos y administrativos como en los personales. Gracias a sus amplios conocimientos y experiencia me ha sido posible completar mis estudios de posgrado, así como realizar esta tesis y todos los artículos para congresos y revistas que de ella se derivan.

También agradezco a mi tutor, Dr. Pedro Albertos Pérez por todo el apoyo brindado como director de los proyectos de investigación de los cuales formé parte durante el desarrollo de mis estudios doctorales, como becario FPI y como contratado en prácticas. Estos proyectos, financiados por el Ministerio de Ciencia e Innovación (MICINN) de España, con referencias DPI 2008-06737-C02-01 (SIDIRELI) y DPI2011-28507-C02-01 (COBAMI) han sido la principal fuente de financiamiento para mi estancia en Valencia, cubriendo además los gastos de publicación de la mayoría de artículos realizados así como los gastos de asistencia a los congresos en los que tuve participación.

Asimismo, doy un agradecimiento y reconocimiento especial a la Universidad de Costa Rica por el apoyo brindado en mi formación académica, por el financiamiento complementario recibido y la oportunidad ofrecida para mi desarrollo como investigador y docente. Agradezco especialmente a los distintos funcionarios de La Oficina de Asuntos Internacionales y Cooperación Externa, del Sistema de Estudios de Posgrado y de la Escuela de Ingeniería

Eléctrica por la ayuda y el apoyo brindado desde el inicio, el cual ha sido fundamental para realizar esta tesis.

Por otra parte, agradecer a la Universidad Politécnica de Valencia, España, al Departamento de Ingeniería de Sistemas y Automática, DISA, y al Instituto de Automática e Informática Industrial, AI2, por haberme permitido realizar mi investigación en sus instalaciones y por poner a mi disposición los diversos equipos requeridos para la realización de las numerosas pruebas experimentales, así como por brindarme la oportunidad de apoyar como docente en las asignaturas de Automática.

Quiero además mostrar mi agradecimiento a los compañeros del DISA y del AI2 que de una u otra forma han contribuido al desarrollo de esta tesis, en especial a Ángel Soriano por toda la ayuda brindada en la realización de las pruebas experimentales, por elaborar los videos demostrativos y por prestar todo su desarrollo en cuanto al servidor del sensor global (cámara cenital) y su simulador multirobot, los cuales han sido herramientas esenciales para la comprobación de la validez y utilidad de los diversos algoritmos propuestos en la presente tesis.

Por último pero no menos importante, agradecer el apoyo incondicional que durante todo este tiempo me ha brindado mi esposa y mi familia, mis padres, hermano y hermanas, y a mis primos, primas, tíos y tías que estuvieron pendientes de mí, gracias por siempre estar anuentes y por ayudarme en los momentos en que más lo necesité, en especial agradezco a mi esposa Irene, compañera de esta gran aventura por su paciencia, amor y ayuda perpetua que han sido vitales a lo largo de todos mis estudios.

Gracias a todos,

Leonardo Marín Paniagua

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

1 | Introducción

Con el fin de comprender la temática de la presente tesis sobre la localización en robots móviles y la utilización de la fusión sensorial para mejorar la precisión de este proceso, se describe a continuación una visión general de los fundamentos teóricos asociados a la teoría básica de navegación y posicionamiento en robots móviles, así como la justificación y objetivos seguidos en la realización del presente trabajo.

1.1 Localización y navegación de robots móviles

Los robots móviles se diseñan y construyen con un sin fin de aplicaciones, con el objetivo de desempeñar una o varias tareas de forma simultánea. En general los robots deben navegar en el entorno explorando o siguiendo una trayectoria (planificándola o predefinida), deben evitar posibles obstáculos hasta llegar a un punto de destino y comunicarse con robots vecinos para coordinar sus movimientos o mantener una formación deseada. Todas estas tareas deben realizarse de forma autónoma en la mayoría de las ocasiones, para lo cual se implementa un sistema de control en distintos niveles con el fin de establecer las velocidades de las ruedas del robot necesarias para seguir la trayectoria planificada y llegar al punto de destino. Este sistema de control no puede funcionar adecuadamente sin conocer el estado del robot (velocidad, posición), para lo cual se recurre en general a distintos sensores con el fin de determinar el estado y permitir con esto el funcionamiento autónomo del robot.

Debido a esto se puede decir que la localización del robot móvil es esencial, ya que para prácticamente cualquier tarea que el robot móvil deba realizar de forma autónoma, se debe determinar su posición y orientación (*Postura*,*pose*) en el espacio respecto a un sistema de referencia global con cierto nivel de precisión. Este aspecto ha sido ampliamente estudiado en la literatura (visión global en [150] y [121]) ya que es un problema que resulta

desafiante debido a que la información de la velocidad y postura del robot se obtiene de sensores con ruido y no-linealidades, aspectos que si no son tomados en cuenta puede llevar a un crecimiento ilimitado (no acotado) de la incertidumbre de estimación, con lo cual el robot no sabría con certeza su posición y no podría realizar adecuadamente sus tareas.

La localización del robot se puede estudiar de varias formas. Suponiendo que la postura inicial antes de iniciar el movimiento es conocida, la postura actual del robot se puede estimar usando información *local* del movimiento 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*, *navegación por estima* (Dead Reckoning) o *estimación de la posición local* ([150] [121]). La odometría estima la postura del robot en un plano (x y y y la orientación θ, ver figura 1.2) a partir de la postura $L_{k-1} = \begin{bmatrix} x & y & \theta \end{bmatrix}^T$ en el instante anterior $k-1$ y la velocidad del robot en los ejes (x, y) junto con la velocidad angular ω. Para un periodo de muestreo T_s pequeño (cercano a los 50ms aunque esto depende de la plataforma utilizada) se tiene que la postura L_k en el instante actual k está dada por la ecuación (1.1) ([121], [150]).

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

Este procedimiento tiene la ventaja de que el tiempo de respuesta es pequeño (bajo consumo de recursos) y el inconveniente de que el error entre la posición real y la estimada se acumula a lo largo del tiempo (error no acotado). Debido a esto, tras recorrer una cierta distancia la estimación de la posición puede ser muy diferente de la posición real.

En caso de que la postura inicial del robot sea desconocida, pero éste disponga de un sensor (barrido láser [10], sonar [93], infrarrojos [129] o cámara omnidireccional [47]) que pueda determinar la posición *relativa* del robot a puntos de referencia (Landmarks), no preestablecidos, que observa y extrae del entorno, entonces el robot podrá construir un mapa del entorno y obtener su postura absoluta en este mapa de forma simultánea utilizando la fusión de

la información proveniente de los sensores de movimiento y los sensores del entorno. Este procedimiento se conoce como *localización y mapeo simultáneo* (SLAM) [50, 10, 53, 32, 11, 12, 129, 34, 33]. Si el robot dispone previamente de un mapa del entorno, el mismo tipo de sensor puede utilizarse para determinar la postura del robot en el mapa, al comparar las características observadas en el entorno con la información almacenada en memoria. De esta forma el robot se puede *autolocalizar* en el mapa ([86], [95], [40], [129]). Estas estrategias se caracterizan por un consumo elevado de recursos debido a la complejidad computacional de la solución y en algunos casos debido también al procesamiento requerido por el sensor del entorno, principalmente cuando se utilizan cámaras y hay que aplicar métodos de visión por ordenador para extraer las características del entorno. Este factor unido a otros problemas relacionados tales como la asociación de datos (problemas para cerrar el bucle entre el inicio y fin del mapa) y no linealidades hacen que la implementación en tiempo real sobre plataformas experimentales sea un problema desafiante [17].

Otra forma de conocer la posición del robot es usar un sensor complejo como puede ser un sistema de posicionamiento global (GPS [55], cámara cenital [106], sensores basados en radio frecuencia [61], etc.) para conocer su posición *global* absoluta en el entorno sin 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. Además para el caso del GPS se presentan otros problemas como la propagación multicamino producida según el entorno navegado [153]. Otros métodos globales se basan en trilateración o triangulación para estimar la postura del robot, pero requieren poseer información previa (mapa) sobre los Landmarks preestablecidos en el entorno ([23], [129], [49], [69]).

Dado que la información de la postura del robot se necesita en el algoritmo de navegación, el tiempo de respuesta del método de localización resulta muy relevante. 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 la deses-

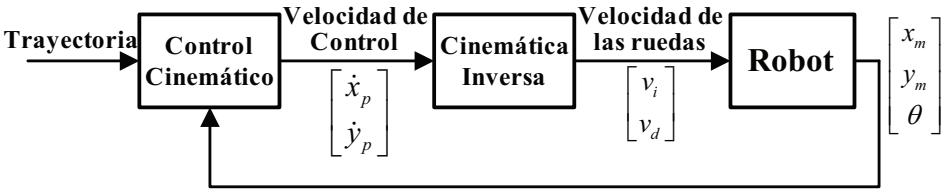


Figura 1.1: Diagrama de bloques del control de posición por Punto Descentralizado.

tabilización del sistema. Debido a esto, la estimación local de la postura se usa habitualmente como fuente principal 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 ([75],[153]). Esta y otras combinaciones entre los métodos de estimación descritos con anterioridad generalmente se implementan mediante un esquema de fusión sensorial basado en alguna versión del filtro de Kalman (de acuerdo a la cantidad de recursos disponibles) como por ejemplo el filtro de Kalman lineal (KF), el extendido (EKF) o el "unscented" (UKF) ([64], [165], [87] and [151]) los cuales se utilizan en gran cantidad de aplicaciones (ver por ejemplo [167] and [122]) además de su uso extendido en localización. Este procedimiento utiliza un modelo del robot móvil para *predecir* su comportamiento y luego, tomando en cuenta la precisión relativa de cada sensor y la de la estimación del modelo, realiza la fusión *actualizando* (*corrigiendo*) esta predicción de forma óptima, incrementando la precisión de la estimación de la postura del robot.

La información sobre la postura (ecuación (1.1)) del robot es utilizada principalmente por el algoritmo de navegación o seguimiento de trayectorias el cual se encarga de generar la referencia en las velocidades de las ruedas del robot. Esta referencia se sigue mediante un control PID [159] que regula la velocidad de los motores. En general se utiliza un algoritmo muy conocido y eficiente para realizar el seguimiento de trayectorias sin obstáculos, el control por punto descentralizado ([150] y [159]). Este algoritmo hace que el robot se mueva siguiendo una trayectoria predefinida, (por ejemplo un cuadrado, un círculo o cualquier función paramétrica en el tiempo. Su diagrama de bloques se muestra en la figura 1.1.

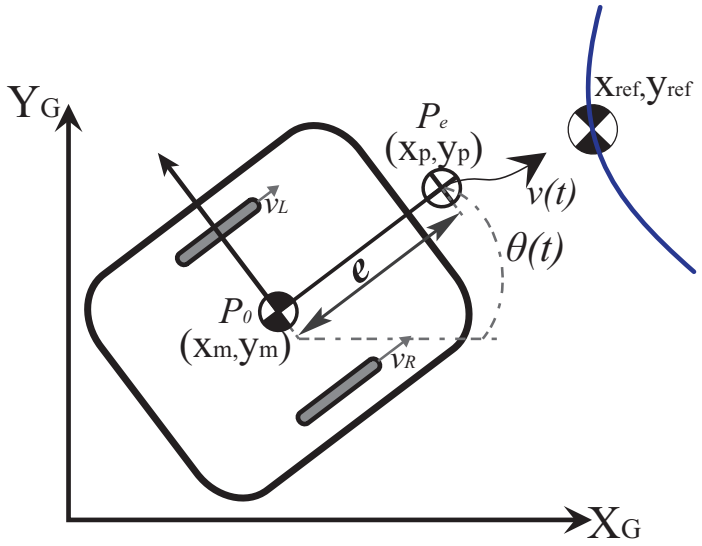


Figura 1.2: Puntos de interés para el control por Punto Descentralizado.

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 de forma 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.

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

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

El funcionamiento de este esquema de control (figura 1.1) es el siguiente: primeramente se obtiene la posición actual del robot (utilizando por ejemplo la ecuación de odometría 1.2) y su diferencia respecto a la posición desea-

da (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. Estas velocidades globales son traducidas por el modelo de la cinemática inversa (ecuación (1.3)) para obtener las velocidades de las ruedas del robot, las cuales son usadas como referencia por el control PID (en el robot). Como la referencia cambia constantemente (en cada instante de muestreo según las funciones paramétricas) esto produce que el error también cambie constantemente lo que produce que el robot se mueva en la trayectoria deseada. El control cinemático utilizado generalmente es un control proporcional con prealimentación de la velocidad tal y como se muestra en la ecuación (1.4).

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

Este esquema básico de navegación requiere una buena precisión en la estimación de la postura del robot para funcionar adecuadamente, por lo que se busca utilizar el método de fusión con menor incertidumbre en la estimación, generalmente esquemas de fusión complejos basados en filtros y modelos no lineales (por ejemplo el UKF y otras técnicas como los filtros de partículas [97], [69]) los cuales requieren un tiempo de ejecución considerable ya que realizan cálculos complejos (por ejemplo inversión o raíz cuadrada de matrices grandes) además de requerir identificar gran cantidad de parámetros para los modelos de robot requeridos. Si el robot a localizar cuenta con *recursos computacionales limitados*, muchos de estos métodos no podrán ser implementados, debido a que, si bien es cierto que la localización es esencial, no es la única tarea relevante que debe realizar el robot; otras tareas dentro del robot como por ejemplo las comunicaciones (supervisión y coordinación), administración de sensores (lectura, calibración, preprocesamiento) y control (navegación y actuación) también son relevantes y requerirán tiempo del procesador. Esto establece un compromiso entre la complejidad de la técnica de localización (cantidad de recursos asignados a la tarea) y su precisión a la hora de estimar la postura utilizando los distintos sensores disponibles.

Considerando estos factores, un buen algoritmo de localización debe ser eficiente desde el punto de vista de recursos utilizados (ancho de banda, tiempo de procesador y consumo de energía [105]), debe incorporar todas las fuentes de información según los sensores disponibles mediante un algoritmo de fusión, e idealmente debe tener un desempeño similar a los esquemas más complejos de manera que provea una estimación de la postura del robot adecuada.

1.2 Justificación

En la actualidad existen muchas plataformas experimentales de amplios recursos computacionales capaces de implementar múltiples métodos de estimación y navegación sin restricciones. Sin embargo, el costo asociado a estas plataformas suele ser muy elevado, por lo que si se desea implementar un grupo de robots o un simple agente para experimentación puede resultar muy costoso. Debido a esto es común la implementación de grupos de robots o agentes simples para experimentación utilizando robots de bajo coste [153] o más accesibles, los cuales, aparte del ahorro económico que suponen, son en general más livianos y fáciles de manipular y mantener. Sin embargo, el ahorro en este tipo de sistemas empotrados conlleva restricciones en la capacidad de cómputo, tamaño de memoria y ancho de banda de comunicación [160] impidiendo la implementación de métodos complejos (computacionalmente costosos) ya que requerirían más memoria de la disponible o bien un tiempo considerable de procesador, lo que impide la ejecución de tareas adicionales.

En estos casos, la práctica común, según los ejemplos que se expondrán en la sección de antecedentes, es la implementación de los métodos en un ordenador externo ya sea mediante simulación únicamente ([118], [40],[34],[145], etc.) o bien como un centro de control externo al robot ([15], [76], [37],[92],[170], etc.) con más recursos pero con la desventaja de agregar peso a la plataforma así como retardos de comunicación no deseados entre el robot y el ordenador. La implementación del algoritmo sobre la unidad de control del robot es menos común, ya que se requieren generalmente unidades de procesamiento potentes y costosas para implementar el método de localización ([73],[141],[79],

[137]) o bien utilizar menos sensores para lograr adaptar la dimensión del filtro a la capacidad de la plataforma ([159]).

Además, la gran mayoría de los métodos realizan un uso constante de la información proveniente de un sensor global, inclusive si el error de estimación local (basado en la unidad inercial y en los encoders) es momentáneamente pequeño o crece con baja velocidad. En el caso de grupos de robots (multirobot) se presenta una situación similar. Una gran parte de los métodos existentes utilizan la información relativa en cada instante de muestreo T_s aún si la estimación local no lo requiere, o bien se hace un uso extensivo del ancho de banda al realizar comunicaciones entre miembros del grupo cada T_s cuando los robots se detectan, pudiendo utilizarse menor cantidad de mensajes para obtener la postura con un error acotado. Estas condiciones indican la posibilidad de mejora en cuanto a los métodos existentes, ya que se puede definir una mejor estrategia en el uso de la información global o relativa utilizando métodos basados en la teoría de control por eventos para determinar cuándo es realmente necesario utilizar esta información para corregir la estimación local y de esta forma consumir menos recursos computacionales y ancho de banda, lo que permitiría la implementación de los algoritmos en robots de recursos limitados.

Debido a estas razones se considera que es necesario el desarrollo e implementación de nuevos algoritmos de fusión, que empleen técnicas del control basado en eventos con el fin de realizar la integración de múltiples sensores de manera eficiente desde el punto de vista computacional y de comunicaciones, y que a su vez produzcan una estimación precisa de la postura de un robot móvil de recursos limitados navegando por su cuenta o bien en un grupo heterogéneo de robots cooperativos. La principal diferencia respecto a los métodos existentes radicará en el uso de la estrategia basada en eventos para dotar a los algoritmos de la capacidad de implementación en los robots de recursos limitados sin necesidad de disminuir la cantidad de sensores o de realizar cálculos previos fuera del robot. Además, en el caso de grupos de robots, los métodos serán distribuidos y con un uso eficiente del ancho de banda respecto a los métodos existentes. Con el fin de desarrollar estos algoritmos, se plantean a continuación los objetivos a desarrollar en el presente trabajo.

1.3 Objetivos

Para la realización de la tesis se estableció como objetivo general el desarrollo de nuevos algoritmos de fusión sensorial para la localización utilizando los principios del control y muestreo basado en eventos, buscando que sean precisos y eficientes computacionalmente así como en el uso del ancho de banda de comunicación, y que además tengan una evolución acotada del error y desempeño adecuado. Para esto se proponen los siguientes objetivos específicos:

- Realizar una revisión del estado del arte de la localización a partir de las técnicas de fusión de datos y localización cooperativa de grupos de robots.
- Obtener modelos dinámicos locales y cinemáticos globales de un robot móvil en configuración diferencial y Ackerman.
- Identificar modelos dinámicos de velocidad y aceleración para los motores típicamente utilizados en robots móviles.
- Realizar un estudio sobre los métodos actuales en estimación de estados y fusión sensorial, con el fin de seleccionar una técnica adecuada para el desarrollo de algoritmos de fusión para la mejora de la localización en plataformas de recursos limitados.
- Desarrollar un algoritmo de fusión que permita la localización de un robot móvil de recursos limitados utilizando la información sensorial local y los modelos obtenidos.
- Desarrollar un algoritmo de fusión para la localización de un robot móvil, utilizando la información local y global junto con los principios del control y muestreo basado en eventos y los modelos obtenidos, buscando reducir la utilización del ancho de banda entre el robot y el sensor global y considerando los casos de navegación en interiores y exteriores.

- Desarrollar un algoritmo de fusión para la localización cooperativa de robots móviles pertenecientes a un grupo heterogéneo, utilizando la información local, global y relativa junto con los principios del control y muestreo basado en eventos y los modelos obtenidos. En este caso se buscará reducir la utilización del ancho de banda entre robots, los cuales comunican entre sí la información de los sensores relativos y su postura para mejorar la precisión de la localización.
- Desarrollar plataformas experimentales de recursos limitados basadas en el LEGO NXT para las configuraciones diferencial y Ackerman en interiores y exteriores. Se realizará la calibración de los distintos sensores a utilizar en caso necesario.
- Comparar los algoritmos desarrollados con los existentes mediante simulación utilizando mediciones provenientes de los sensores utilizados en las plataformas propuestas.
- Observar el desempeño y consumo de ancho de banda de los algoritmos desarrollados para robots individuales al ser implementados en las plataformas experimentales. Realizar la ejecución de pruebas de larga duración y la medición de los tiempos de ejecución de las distintas tareas dentro del robot. Comparar con los resultados obtenidos de las mismas pruebas realizadas con métodos existentes.
- Medición del desempeño y consumo de ancho de banda para el algoritmo cooperativo utilizando una plataforma de simulación con un grupo de robots siguiendo una trayectoria lineal cíclica y con un grupo de robots con evitación de obstáculos Braitenberg en un entorno cerrado. Comparativa con métodos existentes.

1.4 Organización de la tesis

Con la delimitación realizada del tema de la tesis, se organizan los capítulos restantes de la siguiente forma:

Capítulo 2
Se presenta el estado del arte en los temas de fusión sensorial aplicada a localización de robots móviles en interiores y exteriores junto con los antecedentes de la localización cooperativa en grupos de robots y los trabajos relativos a control basado en eventos.

Capítulo 3
Se definen los modelos propuestos para los robots diferencial y Ackerman a utilizar en todos los algoritmos de fusión propuestos. Además se establece un modelo genérico a utilizar en las simulaciones del grupo de robots.

Capítulo 4
Se realiza una descripción de los métodos existentes en cuanto a estimación de estados en sistemas discretos, con el fin de seleccionar el método más adecuado para realizar la fusión sensorial basada en eventos para la localización de un robot móvil de recursos limitados.

Capítulo 5
Se definen los algoritmos de fusión en cascada con actualización basada en el tiempo para la localización de robots móviles de recursos limitados, los cuales son el punto de partida para establecer los métodos basados en eventos.

Capítulo 6
Se establecen los algoritmos de fusión basados en eventos para la mejora de la localización de robots móviles de recursos limitados, utilizando los principios de control y muestreo basados en eventos y tomando como punto de partida los algoritmos en cascada basados en el tiempo.

Capítulo 7

Se describen los algoritmos de fusión para la mejora de la localización en grupos de robots heterogéneos, en donde se establece el algoritmo basado en eventos para la localización cooperativa de robots de recursos limitados.

Capítulo 8

Se describen las plataformas desarrolladas para la prueba de los algoritmos propuestos así como el simulador utilizado para las pruebas en grupos de robots y diversos aspectos de implementación.

Capítulo 9

Se exponen las pruebas realizadas con la plataforma diferencial en interiores.

Capítulo 10

Se muestran las pruebas efectuadas en la plataforma en configuración Ackerman en interiores y exteriores.

Capítulo 11

Se describen los resultados obtenidos con la plataforma de simulación multirobot observando el ahorro en la utilización del ancho de banda obtenida con el método propuesto.

Capítulo 12

Se establecen las conclusiones obtenidas junto con las posibles líneas de continuación de la presente tesis en trabajos futuros.

2 | Antecedentes

Para la realización de esta tesis se ha llevado a cabo una amplia revisión bibliográfica centrándose en las técnicas más recientes en cuanto a fusión sensorial y su aplicación a la localización de robots móviles para plataformas del tipo diferencial y Ackerman así como en el área de grupos de robots con localización cooperativa y distribuida y en la teoría de control basado en eventos. La variedad de trabajos estudiados en estas distintas áreas permite delimitar con claridad la aportación de la presente tesis respecto a los trabajos existentes. Se separan los antecedentes encontrados en tres áreas de énfasis: mejora de la localización mediante fusión sensorial, localización distribuida en grupos de robots y temas afines a la presente tesis que incluye la descripción de métodos relativos a recursos limitados y el control basado en eventos. Finalmente se presenta la justificación para desarrollar la presente tesis basada en los antecedentes establecidos junto con los objetivos principales del presente trabajo, que constituyen los pasos seguidos para la obtención del esquema de fusión sensorial basado en eventos propuesto, así como su implementación y pruebas en el las plataformas desarrolladas. Se incluye además la descripción general de la organización de la tesis.

2.1 Mejora de la localización mediante fusión sensorial

Como se mencionó anteriormente, el filtro de Kalman (KF) y sus variantes son extensamente utilizados para realizar la fusión de los datos provenientes de los distintos sensores del robot con el fin de mejorar la precisión de la estimación de la postura del robot. Estas técnicas son predominantes en los métodos expuestos a continuación.

2.1.1 Estimación inercial

Existen diversos ejemplos que utilizan únicamente la información local obtenida a partir de sensores inerciales. Por ejemplo para robots diferenciales, en [118] se utiliza el EKF para fusionar datos de un sensor de rango láser (para detectar paredes cercanas al robot) y estimar el error producido en la odometría. También en [15], [77], [75] y [76] se utiliza el UKF para combinar la información de un giróscopo, empleándolo para corregir el error en la estimación de la odometría junto con un modelo del error (calculado de forma cinemática). Este esquema da buenos resultados sobre un robot Pioneer 2-DXe [135] controlado desde un ordenador externo al robot (filtros implementados en Matlab ®) aunque las pruebas realizadas son de muy corta duración por lo que el efecto a largo plazo del crecimiento del error no puede ser observado (tenderá a crecer sin medida al no utilizar un sensor global para realizar la corrección de la odometría). De forma similar en [37] se utiliza un giróscopo como sensor adicional a la odometría de los encoders en un EKF, el cual muestra resultados adecuados en un robot Pioneer AT con un ordenador externo ejecutando el método si se realiza una calibración previa del giróscopo.

En esta misma línea, el trabajo presentado en [127] 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 giróscopo. Este caso muestra un buen desempeño pero con la desventaja de que el modelo utilizado es muy complejo (11 estados) lo que incrementa el coste computacional del filtro de Kalman dificultando su implementación dentro del robot. Un esquema de fusión similar se presenta en [128] pero utilizando únicamente la medición del giróscopo en sustitución a la estimación odométrica de la orientación en un robot tipo oruga diferencial. A pesar de que se estima el deslizamiento de la plataforma, se desaprovecha la posibilidad de tomar en cuenta la estimación del ángulo medido por los encoders y el modelo cinemático, perdiendo una fuente de información disponible. Por otra parte, en [173] se emplea el mismo tipo de robot pero con un UKF que estima los parámetros del deslizamiento del robot, con un comportamiento adecuado en las pruebas presentadas a pesar de utilizar únicamente los encoders y no los sensores disponibles de una unidad

inercial (*IMU*). Estos resultados se mejoran en [170] con el robot oruga al utilizar la IMU (giróscopo y acelerómetros) en un EKF tomando en cuenta posibles deslizamientos en las ruedas.

Finalmente en [79] se utiliza un EKF multifrecuencia para combinar las mediciones de una IMU con un sensor de flujo óptico que obtiene el desplazamiento del robot (similar a un ratón láser) con buenos resultados al realizar el mapeo de una tubería a pesar de no contar con navegación autónoma.

2.1.2 Estimación global

2.1.2.1 Navegación en Interiores

Existen diversos ejemplos de navegación en interiores realizando la estimación de la postura utilizando sensores globales con el fin de corregir la deriva de la estimación local en recorridos de larga distancia. Por ejemplo, en [92] se presenta una fusión de un «satélite» ultrasónico (U-SAT: GPS para interiores) con emisores de ultrasonido colocados en el entorno (de posición conocida, generalmente colocados de forma cenital distribuidos en el techo) y un receptor en el robot (ver una descripción completa del sistema en [101]). 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 EKF para determinar la postura del robot. Se muestran resultados adecuados en un robot diferencial de construcción propia, pero con el método implementado en un ordenador externo al robot. Este método se prueba además en un robot móvil Pioneer P3-DX [93] con resultados adecuados utilizando la misma configuración con emisores cenitales, pero ejecutándose nuevamente en un ordenador externo al robot. Un esquema similar se presenta en [96] utilizando el mismo tipo de sensor global basado en localización por ultrasonido pero utilizando el UKF. Se muestran buenos resultados al ser probado en un robot con movimiento restringido mediante una barra de tal forma que solo pueda moverse en un círculo predefinido.

En [27], se diseña y prueba un sistema de localización basado en etiquetas de identificación por radio frecuencia (RFID tags) las cuales se utilizan a

modo de balizas con posición conocida. Al distribuir estas etiquetas en el entorno de movimiento del robot, son detectadas cada vez que el robot se acerca a ellas, leyendo la identificación y posición global, la cual es utilizada para corregir la estimación local. Se observan resultados adecuados en las pruebas de fusión con un EKF en un robot Pioneer P3-DX al compararse con la odometría tradicional.

Otro sistema se puede observar en [159], en el cual se utiliza una cámara cenital a modo de GPS de interior, realizando el procesamiento de las imágenes para obtener el desplazamiento del robot en cada instante de muestreo, fusionando esta medición con los datos de los encoders en un robot Lego Mindstorms RCX (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 sirve como referente en cuanto a plataformas experimentales a utilizar en la presente tesis, utilizando la fusión de información de distintos sensores además de los encoders en la segunda generación del Lego Mindstorms (NXT).

2.1.2.2 Navegación en Exteriores

En el caso de navegación en exteriores existen gran cantidad de ejemplos, la gran mayoría utilizando un robot con configuración Ackerman y navegación autónoma o bien, un coche (vehículo Ackerman) con navegación manual pero adaptado con sensores inerciales (IMU), un GPS y un ordenador portátil para realizar el algoritmo de fusión. Una visión general puede consultarse en [153] para métodos de navegación y posicionamiento en coches, abarcando los aspectos generales de los sistemas y equipos existentes para fusión sensorial y mejora de la localización.

Existen múltiples ejemplos con la configuración Ackerman, la mayoría utilizando la fusión mediante un filtro EKF. Por ejemplo en [114] se utiliza el EKF para fusionar la información de un GPS diferencial (DGPS) con mayor precisión al GPS con las mediciones de los encoders de velocidad y dirección de un vehículo, mejorando de esta forma la estimación de la posición del vehículo. Se mejora aún más la precisión al incorporar un método de

triangulación con balizas locales de posición conocida detectadas mediante un sensor de barrido láser. De forma similar, en [169] emplea el EKF para realizar la fusión de un GPS y una IMU con dos magnetómetros. Ésta es implementada en un ordenador portátil que realiza la corrección global en cada instante de muestreo (frecuencia constante). Se obtiene un buen desempeño en diversas pruebas realizadas en un aparcamiento exterior y en condiciones de falla del GPS (cortes en la señal) al utilizar los sensores restantes para obtener la estimación de la postura del robot mientras se recupera la conexión con los satélites. Una configuración similar se utiliza en [172] pero asistiendo el EKF con una red neuronal entrenada para los casos donde no se tiene señal GPS con buenos resultados en las pruebas en carretera.

En [24] se equipa un vehículo con una IMU de alta precisión y un GPS con dos antenas de recepción como entradas en un modelo completo del vehículo Ackerman (modelo de la bicicleta [168]) cuyos parámetros se ajustan en línea mediante un KF para obtener la postura. De forma similar, en [2] dos GPS con modelo cinemático de tiempo real (RTK-GPS, equivalente a un DGPS) con tres antenas y una IMU se emplean en un EKF con parámetros adaptativos para obtener la postura del robot. La covarianza del sensor se ajusta para mostrar los cortes en el GPS lo cual reduce el error de estimación. En [25] se utiliza una configuración en cascada: primeramente un EKF estima la postura del robot, la cual se utiliza como entrada de un segundo EKF que estima la dinámica del vehículo requerida por el control. Esta configuración se prueba en un tractor Ackerman con un GPS, un giróscopo y encoders. Por último, en [149] utiliza un algoritmo numérico que adapta la estructura del modelo de un vehículo Ackerman para ser utilizado en un KF según la disponibilidad de la señal GPS. Se obtienen resultados adecuados para la estimación de la postura en las extensas pruebas de larga distancia realizadas obteniendo errores bajos durante la falla en la recepción del GPS.

A pesar del amplio dominio de la configuración Ackerman en exteriores, también existen algunos métodos implementados en otras plataformas. Por ejemplo, en [73] se utiliza la fusión sensorial (local y global) en un robot móvil «Packbot» tipo oruga. Este es un robot avanzado, con tecnología militar y gran capacidad de procesamiento. Se utiliza el filtro de Kalman para estimar la postura basado únicamente en sensores (sin considerar un modelo para

la dinámica del robot sino únicamente las mediciones de los sensores) para fusionar la información de la IMU (acelerómetro, giróscopo, brújula) con un DGPS. 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. De forma similar, en [114] se utiliza un EKF sobre un robot oruga con dos motores por eje, en donde la odometría basada en encoders se corrige mediante una brújula y un DGPS. Se observa un desempeño adecuado y superior del método con el EKF a diferencia de utilizar únicamente el DGPS o los encoders. Finalmente en [141] se utiliza un GPS y un EKF con actualización dinámica de sus parámetros mediante lógica difusa realizando el ajuste según el desempeño en línea del filtro. Se observan resultados adecuados en las pruebas con un robot tipo "rover" avanzado, que incluye un receptor de GPS de antena dual, al realizar trayectorias de gran longitud.

2.2 Localización cooperativa distribuida en grupos de robots

En el caso de localización multirobot cooperativa y distribuida existen gran cantidad de métodos desarrollados para distintos casos los cuales se exponen en esta sección. La idea general en estos métodos es utilizar la información relativa (rango y ángulo [116]) entre robots junto con la postura de los vecinos (recibida mediante comunicaciones) para mejorar la estimación de cada robot siendo además una característica deseable el que los algoritmos sean distribuidos.

En general, los algoritmos de *estimación distribuida* exploran la solución al problema de estimación del estado de un grupo de nodos sensores (constituido por el estado de todos los nodos) sin necesidad de un centro de fusión centralizado ([54], [140]). Estos métodos son extendidos al área de robótica móvil al considerar los robots como nodos sensores en movimiento, de tal forma que cada robot pueda obtener la estimación del estado (velocidades, postura, aceleraciones, etc.) del *grupo* utilizando para esto las comunicacio-

nes entre miembros. De esta forma, el estado del grupo (global obtenido localmente por cada miembro) puede ser utilizado en distintas aplicaciones tales como control de formación, evitación de obstáculos, control cooperativo, seguimiento de objetivo (objeto de interés, líder, etc.), localización segura y/o robusta, entre otras ([44], [119], [48], [66], [163], [123] y [1]).

Siguiendo esta idea, los algoritmos de *localización distribuida* se enfocan en la estimación de la postura sin necesidad de un centro de fusión, de esta forma cada robot obtiene la postura del grupo (conjunto de todas las posturas de los miembros del grupo) utilizando para esto comunicaciones entre vecinos y mediciones de las posturas relativas a estos robots. Si el robot únicamente estima su postura (no la del grupo) en cada instante de muestreo utilizando la información relativa a los vecinos cercanos a él, entonces el problema se considera de *localización cooperativa* y será distribuida si no utiliza información centralizada. La aplicación final del grupo de robots (su objetivo, misión o directiva global) determinará la necesidad de utilizar un método de localización distribuida o de localización distribuida cooperativa para estimar únicamente la postura del grupo o la del robot. Por ejemplo, si el objetivo principal es mantener una formación determinada deberá estimarse el estado del grupo de robots, si por el contrario es un grupo de robots explorando el entorno se puede estimar el estado de cada robot de forma cooperativa. Si la misión del grupo de robots requiere el mantener una formación de forma temporal, puede alternarse entre estrategias de localización o bien complementar la estrategia cooperativa al recibir mediante comunicación de forma regular la postura de los vecinos en formación (estimada independientemente por cada miembro y comunicada a los demás) durante el tiempo en el que se debe mantener la formación. Además, la mayoría de los algoritmos para localización distribuida se adaptan al problema cooperativo al modificar el estado a estimar para contemplar únicamente la postura local del robot y no la de los demás miembros del grupo con lo que la mayoría de algoritmos existentes pueden adaptarse a una condición u otra según lo requiera la misión del grupo.

Las principales estrategias para solucionar la localización distribuida pueden clasificarse en dos grandes áreas: los algoritmos basados en filtros de

fusión y los algoritmos basados en redes de sensores, las cuales se exponen a continuación.

2.2.1 Soluciones basadas en filtros

Las soluciones basadas en filtros de fusión se fundamentan en algún filtro de fusión (KF, EKF, UKF, filtros de partículas, etc. ver [151], [97], [104], [36] y [124]) para incorporar la información relativa en la estimación de la postura local o global e incluir la información de un mapa en el caso de los métodos de SLAM cooperativo [33] o en los métodos con balizas en entorno [26].

Una de las primeras soluciones presentadas para un grupo de robots en constante movimiento se expone en [145] en donde un filtro KF del estado global del grupo se distribuye en forma de filtros de menor dimensión a los miembros del grupo. Cada robot estima su postura y realiza la actualización con la información relativa únicamente cuando los robots se encuentran (se detectan el uno al otro mediante los sensores relativos). Se obtienen resultados adecuados en el caso de un grupo de tres robots, uno de ellos con acceso a información global, y utilizando dos estrategias de actualización: se utiliza la información relativa cada instante de muestreo durante el tiempo que tarda un encuentro entre dos robots, o bien se actualiza en dos instantes (durante el encuentro) previamente definidos. Sin embargo, este método tiene requerimientos elevados en cuanto a comunicaciones, debido a que cada mensaje enviado debe incluir las matrices de covarianza que deben intercambiarse en cada instante de muestreo (cada ciclo con tiempo de muestreo T_s) con los robots del grupo siempre que dos robots se encuentren. Este trabajo se extiende en [113] y [114] para navegación en exteriores y en [117] con restricciones adicionales a la información relativa utilizada (explorando los casos en los que únicamente se utilizan para la solución mediciones de la posición, rango o ángulo).

En [125] se obtiene una planificación sensorial óptima para un grupo de robots con recursos limitados. Es, en cierta forma, una formalización de la actualización en dos tiempos predeterminados utilizada en [145], pero procesando las mediciones de los sensores a una frecuencia óptima, determinada por una optimización convexa previa (offline) de una función de coste que

se basa en la relación entre el ruido del robot y la frecuencia del sensor. Este método muestra buenos resultados en un grupo de robots con formación preestablecida pero no es aplicable en caso de un grupo dinámico, en donde los robots se encuentran de manera aleatoria ya que la frecuencia del sensor relativo es variable con el tiempo.

En lugar de procesar las actualizaciones de forma secuencial, cada vez que los robots se encuentran, la fusión sensorial con información relativa se puede realizar con más de un vecino de forma simultánea, al incrementar de forma dinámica el vector de medición y las matrices correspondientes en el esquema KF tal y como se muestra en [72]. Esta estrategia mejora la precisión de la estimación pero, por otra parte, el costo computacional se incrementa con cada vecino que se agrega a la estimación, por lo que no es un método adecuado para grupos extensos de robots con recursos limitados

Otros casos de algoritmos basados en filtros buscan restringir la información relativa utilizada en el método para que sea anónima, es decir, que no sea necesario conocer la identidad de los vecinos para mejorar la estimación de la postura. Esto evita resolver el problema de correspondencia, el cual define qué mediciones relativas (del sensor de rango y ángulo) corresponden con la información de la postura de los robots vecinos recibida mediante comunicaciones. Como es de esperarse, al utilizar información anónima se producen ambigüedades en la estimación de la postura, ya que se producen múltiples soluciones posibles para cada robot dentro del grupo. En [59, 60] esta ambigüedad se resuelve utilizando un algoritmo de registro múltiple que genera las hipótesis de la postura que se procesan en conjunto por un EKF, el cual incorpora las mediciones de odometría y rechaza las hipótesis que son inconsistentes con la estimación previa de la postura. Se observa un buen desempeño en las pruebas aunque se requiere un tiempo del procesador mayor con cada robot agregado a la estimación, resultando no conveniente en grupos extensos de robots. Este algoritmo de registro múltiple se utiliza también en [38] pero con mediciones anónimas del rango y filtros de partículas con el fin de estimar la postura en tres dimensiones (3D) del robot. Los requerimientos en la capacidad computacional son muy elevados ya que se requiere gran cantidad de filtros de partículas (uno por cada miembro del grupo). Esto impide la implementación del método en cada robot, siendo implementado

en un ordenador externo centralizado. Además, la cantidad de información transmitida es considerable y se actualiza en cada instante de muestreo T_s, haciendo uso extenso del ancho de banda disponible en los robots.

2.2.2 Soluciones basadas en redes de sensores

Las soluciones basadas en el enfoque de localización de redes de sensores requieren un conocimiento previo de la topología de la red (por ejemplo cuando la formación es rígida) para poder estimar la posición o postura de cada nodo. En estos métodos se requiere en general menos información relativa que en las estrategias basadas en filtros de fusión, reduciendo el costo de implementación ya que se utilizan sensores menos complejos. Además, se requiere de un nodo (robot) "*referencia*", que cuente con información precisa de su postura (con información global absoluta) de forma que se pueda acotar el error de estimación de localización en la red y dotarla de información global.

Por ejemplo, en [49], se exponen los fundamentos teóricos para la localización de la red de robots utilizando únicamente información relativa (no se dispone de información local del movimiento). Se plantean y analizan distintos casos de los cuales solo uno se prueba como solucionable (se obtiene la postura de cada nodo de la red de forma única), y es cuando cada robot en la red puede realizar mediciones de rango y ángulo con todos los miembros del grupo (o al menos se detectan dos vecinos por cada robot). En otros casos, como por ejemplo utilizando únicamente mediciones de rango y aún teniendo un robot con información global, el problema no tiene solución (única).

En [29] se utilizan únicamente mediciones de rango (distancia relativa) junto con un método distribuido de optimización por gradiente para estimar la postura del grupo siempre y cuando se cuente con al menos tres robots con información global. Este método ofrece un buen desempeño en grupos extensos pero los robots deben detenerse cada vez que quieran ejecutar el algoritmo para mejorar la estimación de la posición. Se requiere además varias iteraciones del método para converger a una solución, comunicándose cada robot con todos sus vecinos en cada iteración.

2.3 Temas afines

Para la presente tesis resultan relevantes distintos temas relacionados con la localización de robots móviles, los cuales se exponen a continuación.

2.3.1 Recursos limitados

Existen pocos trabajos que toman en cuenta alguna forma de limitación en los recursos de la plataforma o que son diseñados específicamente para utilizar menos ancho de banda. Por ejemplo en [159], los filtros de fusión se limitaron para utilizar solo dos fuentes de información: los encoders y la cámara cenital. De esta forma las dimensiones del filtro son reducidas y se logran implementar en el robot, sin embargo se pierde la posibilidad de aprovechar otros sensores disponibles dentro de la plataforma. Otra forma común de implementar los algoritmos de fusión es realizar parte de los cálculos del filtro fuera de línea previo a la implementación en el robot con el fin de evitar operaciones complejas (inversión o raíz cuadrada de matrices de gran dimensión) dentro de la plataforma de recursos limitados con múltiples sensores ([151], [104] y [36]). Sin embargo, en este caso se pierde la posibilidad de ajustar dinámicamente los parámetros del filtro según las condiciones de funcionamiento o del entorno ya que en este caso se tendría que recalcular la parte del filtro fuera de línea y repetir este proceso en cada cambio de parámetros realizado. Esto no es conveniente y en general se buscan soluciones dinámicas con el cálculo completo del filtro dentro de la plataforma.

En cuanto a métodos con uso eficiente o reducido del ancho de banda se encuentran por ejemplo el expuesto en [158], en donde se establece un método de estimación multirobot de la postura mediante mediciones cuantificadas utilizando únicamente 4 bits por medición. Se obtienen resultados similares al compararse el método propuesto con estimadores de valor real (sin limitar la información de la medición) pero utilizando menos ancho de banda. Sin embargo, los robots deben comunicarse en cada instante de muestreo para garantizar la convergencia del método y evitar que el error en la estimación crezca indefinidamente. Esta limitación se supera en el trabajo presentado por [103] que propone un algoritmo descentralizado para una red dinámi-

ca de robots sin garantía de comunicación entre miembros del grupo. Se consideran casos en donde los mensajes se pierden o no son recibidos entre vecinos, además se resuelve el problema de actualización cíclica ya que no se reutiliza información que ha sido incorporada recientemente en la fusión evitando estimaciones optimistas. Sin embargo, los requerimientos de memoria son importantes y crecen según el tamaño del grupo de robots. Finalmente, en [137] se propone un filtro de partículas basado en agrupamientos con mediciones relativas asíncronas, logrando una reducción considerable en la complejidad de los filtros de partículas y con buen desempeño al utilizar el sensor propuesto en [138] inclusive en caso de pérdida de mensajes. No obstante nuevamente se requiere de comunicación constante entre vecinos en cada instante de muestreo para obtener un error acotado en la estimación.

2.3.2 Control basado en eventos

Los métodos basados en eventos son un área de desarrollo reciente. Surgen en gran medida por la necesidad de limitar el uso del ancho de banda entre el controlador y el proceso en un lazo de control cerrado sobre una red, buscando realizar las comunicaciones cuando una condición se cumple a diferencia de comunicarse en cada instante de muestreo T_s [48]. Por ejemplo, si se realiza la transmisión de las mediciones del sensor del lazo únicamente cuando el error entre el valor actual y el valor anterior de la variable medida supera un nivel predeterminado, se trata de un caso de "medición basada en eventos" [78]. Por otra parte, si la acción de control se aplica únicamente cuando el error en la referencia o en la perturbación supera un nivel preestablecido, se tiene un caso de "control por eventos" o "accionamiento por eventos" ([13], [109]).

Las estrategias de control basado en eventos ([71],[120],[3]) se han extendido para ser utilizadas con éxito en distintas áreas tales como el control cooperativo ([48],[57]), consenso multiagente ([148],[68]) y protocolos de acuerdo en sistemas multiagente [119]. En estos casos, la solución basada en eventos producirá un intercambio de información entre los miembros del grupo (sensores, controladores, robots, etc.) únicamente si el error (en la referencia, perturbación, estimación o respecto al valor anterior de medición) sobrepasa un

nivel predeterminado, lo que reduce la carga de comunicación y el consumo de energía entretanto se logra un comportamiento similar a las estrategias basadas en un tiempo de muestreo constante. Según la aplicación, se puede presentar un compromiso entre la exactitud del método (máximo error permitido) y el nivel del evento utilizado, por ejemplo en el caso de control cooperativo [48] se observa como la precisión del método mejora al utilizar un nivel de evento bajo, lo que tiene el efecto de incrementar en número de comunicaciones entre los miembros del grupo.

Se puede extender la utilización de las estrategias por eventos al área de localización de robots individuales o grupos de robots mediante filtros de fusión, buscando beneficios similares en cuanto al uso de recursos o respecto al uso eficiente del ancho de banda. Sin embargo, el principal problema respecto a la estimación de la postura es que el error de estimación es desconocido para el robot al menos que se posea un sensor global operando en todo momento. En este caso no tendría mucha utilidad ya que se requeriría consultar constantemente el sensor global para determinar si se debe usar o no este sensor global. Un caso similar ocurre en los grupos de robots, ya que se requeriría establecer la comunicación con los vecinos con el fin de obtener el error entre la estimación local del robot y la medición con el sensor relativo y con base a este error decidir si se comunica o no; es decir, se utilizan las comunicaciones para determinar si se deben o no realizar las comunicaciones con los vecinos.

Con esto está claro que utilizar el error de estimación en el caso de la localización para definir el evento con el que se regula el acceso a la información global o relativa no es conveniente y se debe establecer otro método para reflejar la necesidad de establecer comunicaciones para obtener la información relativa o la necesidad de la información global. Algunos métodos existentes ([68],[148]) utilizan el error en la referencia de posición para definir el evento de localización; sin embargo, este error está estrechamente ligado a la navegación y no a localización ya que se puede dar el caso en el que el error de navegación disminuya (al acercarse el robot al punto de destino) pero el error de localización crezca, ya que cuando el robot avanza y estima su postura mediante odometría, el error tiende a acumularse, esto tendría un comportamiento no deseado en localización ya que el evento no refleja la necesidad de corregir la estimación local de la postura. Este problema es

resuelto en la presente tesis definiendo el evento con base a la covarianza del error de estimación local como se expondrá en la sección de algoritmos.

Realizada la descripción del estado del arte de los temas necesarios para el desarrollo de los algoritmos de localización propuestos en la presente tesis, se definen a continuación los modelos desarrollados para los robots diferencial y Ackerman, así como un modelo genérico a utilizar en las simulaciones del grupo de robots.

3 | Modelado cinemático y dinámico de robots móviles

Un componente fundamental de los métodos de localización es la obtención de un modelo adecuado de la plataforma experimental, de tal forma que no sea complejo en extremo (en cuanto a implementación y parametrización) pero que logre caracterizar adecuadamente el comportamiento del robot. En este capítulo se presenta un modelo cinemático y dinámico para las plataformas utilizadas, con un enfoque tomado de la equivalencia dinámica de los sistemas mecánicos a un sistema de partículas, lo que simplifica la obtención del modelo y su implementación en un robot móvil de recursos limitados. Se exponen los casos del robot diferencial (incluyendo el caso utilizado en el sistema multirobot) y del robot con configuración Ackerman.

3.1 Robot móvil 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 que producen las aceleraciones a_R y a_L tal como se muestra en la figura 3.1. Las ruedas son convencionales y se asume que satisfacen la condición de rodamiento puro sin deslizamiento [30]. 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. En el caso del robot diferencial se considera que cumple la condición de movimiento en bajas velocidades. De esta forma se asume que no hay deslizamiento en caso del movimiento del robot en interiores [139]. Se analiza la cinemática y dinámica de este robot para obtener el modelo del robot que se utilizará en los filtros de fusión.

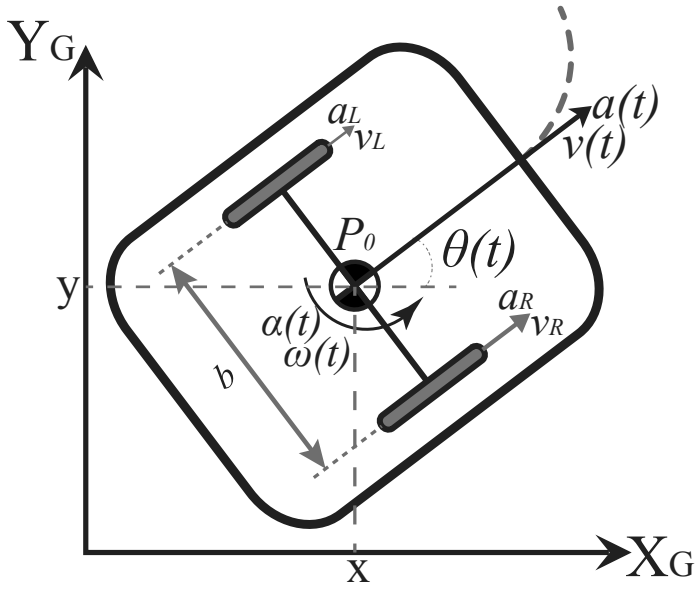


Figura 3.1: Cinemática de un robot en configuración diferencial

3.1.1 Modelo cinemático

Este modelo representa la evolución de las velocidades del robot en un marco inercial fijo. La *postura* del robot se define mediante su posición (x, y) (coordenadas del punto P_0) y el ángulo de orientación θ (ángulo de avance, curso del robot o guiñada) en el marco de referencia *Global* (X_G,Y_G) mostrado en la figura 3.1. Conociendo la velocidad lineal y angular (v y ω) en el marco de referencia *Local* (X_L,Y_L), la velocidad global del robot se define como

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \Rightarrow \begin{array}{c} \dot{x}(t) = v(t)\cos(\theta(t)) \\ \dot{y}(t) = v(t)\sin(\theta(t)) \\ \dot{\theta}(t) = \omega(t) \end{array} \tag{3.1}$$

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 expuestos en [161], [46] y [7]). Sin embargo, es conveniente obtener el resultado exacto de la integral y evaluar el uso de aproximaciones

que se consideren convenientes para simplificar el cálculo en la ejecución del algoritmo sobre el robot.

La solución exacta de (3.1) 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 dos instantes consecutivos de muestreo mostrado en (3.2).

$$T_K \;=\; \{t \in \mathbb{R} \mid kT_s \;\leqslant t \;<\; (k\;+1)\,T_s\;\},\; k \;=\; 0,1,2,\ldots \tag{3.2}$$

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 $t_k \Rightarrow v\left(t_k\right), \omega\left(t_k\right)$ son constantes. De esta forma se plantea la integral en (3.3).

$$\begin{aligned} x\left(t\right) &= x\left(t_k\right) + v\left(t_k\right) \int_{t_k}^{t} \cos\left(\theta\left(\tau\right)\right) d\tau \\ y\left(t\right) &= y\left(t_k\right) + v\left(t_k\right) \int_{t_k}^{t} \sin\left(\theta\left(\tau\right)\right) d\tau \\ \theta\left(t\right) &= \theta\left(t_k\right) + \int_{t_k}^{t} \omega\left(t_k\right) d\tau \end{aligned} \tag{3.3}$$

Se resuelve primeramente para el ángulo de avance del robot θ tal y como se muestra en (3.4)

$$\theta\left(t\right) = \theta\left(t_k\right) + \int_{t_k}^{t} \omega\left(t_k\right) d\tau = \theta\left(t_k\right) + \omega\left(t_k\right) \int_{t_k}^{t} d\tau = \theta\left(t_k\right) + \left(t - t_k\right) \omega\left(t_k\right) \tag{3.4}$$

Se sustituye este resultado en la ecuación (3.3) para (x, y). Cabe destacar que las funciones evaluadas en t_k son constantes $\left(x\left(t_k\right), y\left(t_k\right), \theta\left(t_k\right)\right)$ ya que son las condiciones iniciales (constantes de integración), de esta forma se

procede a integrar (3.3) dando el resultado en (3.5) para x y en (3.6) para y.

$$\begin{aligned}
x(t) &= x(t_k) + v(t_k) \int_{t_k}^{t} \cos\left(\theta(t_k) + (\tau - t_k)\,\omega(t_k)\right) d\tau \\
&= x(t_k) + \frac{v(t_k)}{\omega(t_k)} \left[\sin\left(\theta(t_k) + (\tau - t_k)\,\omega(t_k)\right)\right]_{t_k}^{t} \\
\Rightarrow x(t) &= x(t_k) + \frac{v(t_k)}{\omega(t_k)} \left[\sin\left(\theta(t_k) + (t - t_k)\,\omega(t_k)\right) - \sin\left(\theta(t_k) + (t_k - t_k)\,\omega(t_k)\right)\right] \\
\therefore x(t) &= x(t_k) + \frac{v(t_k)}{\omega(t_k)} \left[-\sin\left(\theta(t_k)\right) + \sin\left(\theta(t_k) + (t - t_k)\,\omega(t_k)\right)\right]
\end{aligned} \tag{3.5}$$

$$\begin{aligned}
y(t) &= y(t_k) + v(t_k) \int_{t_k}^{t} \sin\left(\theta(t_k) + (\tau - t_k)\,\omega(t_k)\right) d\tau \\
&= y(t_k) - \frac{v(t_k)}{\omega(t_k)} \left[\cos\left(\theta(t_k) + (\tau - t_k)\,\omega(t_k)\right)\right]_{t_k}^{t} \\
\Rightarrow y(t) &= y(t_k) - \frac{v(t_k)}{\omega(t_k)} \left[\cos\left(\theta(t_k) + (t - t_k)\,\omega(t_k)\right) - \cos\left(\theta(t_k) + (t_k - t_k)\,\omega(t_k)\right)\right] \\
\therefore y(t) &= y(t_k) + \frac{v(t_k)}{\omega(t_k)} \left[\cos\left(\theta(t_k)\right) - \cos\left(\theta(t_k) + (t - t_k)\,\omega(t_k)\right)\right]
\end{aligned} \tag{3.6}$$

Ambas expresiones (3.5) y (3.6) se pueden simplificar aplicando identidades trigonométricas[1], tal y como se muestra en (3.7):

$$\left.\begin{array}{c} a = \theta(t_k) \\ b = (t - t_k)\,\omega(t_k) \end{array}\right\} \Rightarrow \begin{array}{l} -\sin\left(\theta(t_k)\right) + \sin\left(\theta(t_k) + (t - t_k)\,\omega(t_k)\right) = \sin(a+b) - \sin(a) \\ \cos\left(\theta(t_k)\right) - \cos\left(\theta(t_k) + (t - t_k)\,\omega(t_k)\right) = \cos(a) - \cos(a+b) \end{array}$$

$$\sin(a+b) - \sin(a) = 2\sin(0{,}5\,(a+b-a))\cos(0{,}5\,(a+b+a)) = 2\sin(0{,}5b)\cos(0{,}5\,(2a+b))$$

$$\cos(a) - \cos(a+b) = -2\sin(0{,}5\,(a-a-b))\sin(0{,}5\,(a+b+a)) = -2\sin(-0{,}5b)\sin(0{,}5\,(2a+b)) \tag{3.7}$$

[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 (3.7) a (3.5) y (3.6) se obtiene (3.8).

$$\begin{aligned}
&\Rightarrow x(t) = x(t_k) + 2\frac{v(t_k)}{\omega(t_k)} \sin\left(0{,}5\,(t-t_k)\,\omega(t_k)\right) \cos\left(\theta(t_k) + 0{,}5\,(t-t_k)\,\omega(t_k)\right) \\
&\Rightarrow y(t) = y(t_k) + 2\frac{v(t_k)}{\omega(t_k)} \sin\left(0{,}5\,(t-t_k)\,\omega(t_k)\right) \sin\left(\theta(t_k) + 0{,}5\,(t-t_k)\,\omega(t_k)\right) \\
&\theta(t) = \theta(t_k) + (t-t_k)\,\omega(t_k)
\end{aligned} \tag{3.8}$$

Este es el resultado completo de la integración para cualquier $t \in T_K$. Al evaluar (3.8) 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 (3.9).

$$\begin{aligned}
x(t_{k+1}) &= x(t_k) + 2\frac{v(t_k)}{\omega(t_k)} \sin\left(0{,}5\,(t_{k+1}-t_k)\,\omega(t_k)\right) \cos\left(\theta(t_k) + 0{,}5\,(t_{k+1}-t_k)\,\omega(t_k)\right) \\
y(t_{k+1}) &= y(t_k) + 2\frac{v(t_k)}{\omega(t_k)} \sin\left(0{,}5\,(t_{k+1}-t_k)\,\omega(t_k)\right) \sin\left(\theta(t_k) + 0{,}5\,(t_{k+1}-t_k)\,\omega(t_k)\right) \\
\theta(t_{k+1}) &= \theta(t_k) + (t_{k+1}-t_k)\,\omega(t_k)
\end{aligned} \tag{3.9}$$

Finalmente, sustituyendo el periodo de muestreo $T_s = t_{k+1} - t_k$ en (3.9) 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 en (3.10). Esta ecuación puede considerarse como una *transformación de coordenadas* ya que relaciona el movimiento local del robot (v, ω) con su movimiento global (x, y, θ). Se muestra en esta ecuación la notación a utilizar en los algoritmos de fusión propuestos, tomando en cuenta el instante actual k y el anterior $k-1$.

$$\begin{aligned}
\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} &= \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} \frac{2v_k}{\omega_k} \sin\left(0, 5T_s\omega_k\right) \cos\left(\theta_k + 0, 5T_s\omega_k\right) \\ \frac{2v_k}{\omega_k} \sin\left(0, 5T_s\omega_k\right) \sin\left(\theta_k + 0, 5T_s\omega_k\right) \\ T_s\omega_k \end{bmatrix} = \\
\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} &= \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} \frac{2v_{k-1}}{\omega_{k-1}} \sin\left(0, 5T_s\omega_{k-1}\right) \cos\left(\theta_{k-1} + 0, 5T_s\omega_{k-1}\right) \\ \frac{2v_{k-1}}{\omega_{k-1}} \sin\left(0, 5T_s\omega_{k-1}\right) \sin\left(\theta_{k-1} + 0, 5T_s\omega_{k-1}\right) \\ T_s\omega_{k-1} \end{bmatrix}
\end{aligned} \tag{3.10}$$

En el caso de la mayoría de plataformas con recursos limitados (y en las utilizadas en la presente tesis) se requiere un T_s suficientemente pequeño para la implementación estable del control con fusión sensorial a velocidades intermedias. Con esta condición se puede realizar una aproximación en la integral de la ecuación diferencial (3.10) considerando que $\sin(\omega_k \cdot 0,5T_s) \approx 0,5T_s\omega_k$ tal y como se muestra en (3.11).

$$L_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + T_s \begin{bmatrix} v_{k-1}\cos(\theta_{k-1} + \omega_{k-1} \cdot 0,5T_s) \\ v_{k-1}\sin(\theta_{k-1} + \omega_{k-1} \cdot 0,5T_s) \\ \omega_{k-1} \end{bmatrix} \quad (3.11)$$

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 (3.11) 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 (3.12).

$$L_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + T_s \begin{bmatrix} v_{k-1}\cos(\theta_{k-1}) \\ v_{k-1}\sin(\theta_{k-1}) \\ \omega_{k-1} \end{bmatrix} \quad (3.12)$$

Se observa que esta ecuación es la misma que la (1.1) en la introducción ([121] y [150]). Ambas ecuaciones (3.11) y (3.12) son ampliamente usadas en la literatura sin discriminar las condiciones necesarias para que éstas estimen de forma adecuada la postura del robot, ni se establecen los requerimientos del robot para poder resolver esta ecuación sin superar el tiempo de muestreo T_s. De esta forma, se propone el siguiente criterio de utilización en el presente trabajo: si el robot es de recursos muy limitados, se utilizará la ecuación (3.12) (aproximación simple pero rápida), en cambio si se tiene un poco más de capacidad computacional disponible, se utilizará la ecuación (3.11); finalmente, si se está trabajando con recursos abundantes (por ejemplo en un ordenador para controlar el robot o en las simulaciones) se utilizará el valor exacto dado por (3.10).

Con estos modelos cinemáticos (relacionan velocidades únicamente) se han implementado o simulado la mayoría de los trabajos planteados en los an-

tecedentes. Si bien es cierto que estos producen buenos resultados en las aplicaciones [63], no se está tomando en cuenta la dinámica del robot la cual puede representar de forma más adecuada el comportamiento del robot y mejorar la estimación de la postura en los filtros de fusión.

Para finalizar el modelado cinemático, existe una relación entre v y ω y las velocidades lineales de las ruedas (v_L,v_R obtenidas a partir del cambio en el desplazamiento de los encoders) tal como se muestra en (3.13). 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 colocados sobre 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 (3.10) 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.

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ -1/b & 1/b \end{bmatrix} \begin{bmatrix} v_L \\ v_R \end{bmatrix} \Rightarrow \begin{bmatrix} a \\ \alpha \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ -1/b & 1/b \end{bmatrix} \begin{bmatrix} \dot{v}_L \\ \dot{v}_R \end{bmatrix} \tag{3.13}$$

3.1.2 Modelo dinámico

Este modelo representa la evolución de las aceleraciones lineal y angular del robot (a,α) en términos de las fuerzas que han sido aplicadas por sus motores $F_{L,R}$ y el momento angular (torque de rotación) τ. 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 los modelos obtenidos en [31], [43], [67] [164], [43], 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 [147], [166], [130] y [174]). Tanto en estas estrategias como en las observadas en los antecedentes de la revisión bibliográfica,

se producen modelos muy complejos que tienen varias desventajas. Primeramente, en muchos casos, debido a la complejidad del modelo se tiene una gran dificultad en obtener o identificar la mayoría de los parámetros requeridos de forma experimental o de las especificaciones del fabricante. Además, este proceso puede consumir mucho tiempo ya que debe realizarse de nuevo si se cambia por ejemplo un motor en un robot, o cada vez que se agrega o sustituye un robot en el grupo. Otro aspecto a tomar en cuenta es que al ser en general modelos no lineales de dimensión alta, el cálculo requerido es considerable lo que dificulta su implementación práctica, principalmente porque se requerirá una versión del filtro para sistemas no lineales (extendido EKF o «unscented» UKF) para realizar la fusión sensorial utilizando este modelo. Estas variantes realizarán una inversión o raíz de matrices de gran dimensión, lo que los excluye de la implementación práctica en sistemas de recursos limitados. Debido a esto se consideran que en general, estos modelos aportan poco al esquema de fusión ya que son difíciles de implementar.

La principal complejidad de los modelos descritos en 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 3.2), 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 (de forma similar a la derivada del modelo cinemático (3.1)). Con esto se obtiene la aceleración en los ejes globales la cual es integrada dos veces para obtener el modelo de la postura 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 como solución el utilizar un modelo dinámico *local* para realizar la fusión sensorial (por ejemplo, utilizando un filtro de Kalman) y posteriormente aplicar la transformación de coordenadas (modelo cinemático) para obtener la dinámica *global*. Si bien es cierto esta solución es en apariencia trivial, no lo es, ya que con esto se obtiene una ventaja fundamental, se logra simplificar el cálculo del método de fusión ya que se utilizaría un modelo de menores dimensiones (y en algunos casos lineal) para estimar la dinámica local, incorporando los distintos sensores de la plataforma y con esto estimar la postura del robot mediante (3.11). Esto se expondrá ampliamente en la sección de

algoritmos de fusión así como la incorporación de las mediciones globales en este esquema.

Adicionalmente, existe otro problema 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 muchas plataformas experimentales el valor del par motor no se puede medir, ni su corriente de consumo (que puede ser utilizada para obtener el par), únicamente se dispone de la información de los encoders. Para solucionar este problema lo que comúnmente se hace es obtener un modelo teórico para el par motor con el problema de que en la mayoría de los robots comerciales 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 de implementación sencilla, que no dependa de las fuerzas en las ruedas sino de la aceleración de las mismas. De esta forma se podría utilizar en una gran cantidad de plataformas existentes que no disponen de sensores de fuerza o corriente en cada motor. Las aceleraciones pueden obtenerse de la medición directa por medio de un acelerómetro 3D colocado sobre cada rueda o bien de un modelo que relaciona la acción de control con la aceleración en cada rueda. Con esto se procede de la siguiente forma: se obtendrá primeramente el modelo dinámico con las fuerzas como entradas al utilizar la segunda ley de Newton, posteriormente se determinará el modelo dinámico del movimiento local del robot que dependa de la aceleración en las ruedas del motor (a_L,a_R) mediante un sistema de partículas dinámicamente equivalente en versiones de dos y tres partículas y finalmente se obtendrá el modelo dinámico de las aceleraciones de las ruedas del robot basándose en la información de los encoders y la acción de control del robot.

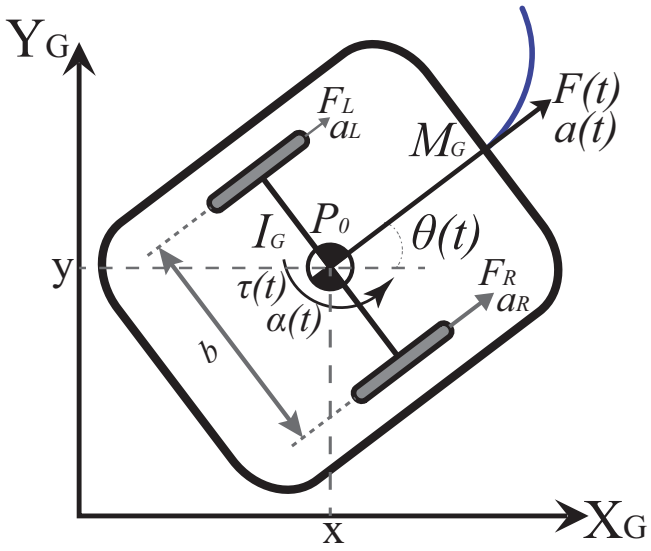


Figura 3.2: Dinámica de un robot en configuración diferencial

3.1.2.1 Dinámica basada en fuerzas, ley de Newton

Si 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 en las ruedas del robot (fuerza resultante del motor menos la fricción) las cuales están separadas una distancia b entre sí tal y como se muestra en la figura 3.3, 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).

Al aplicar la segunda ley de Newton al cuerpo rígido se obtiene la ecuación (3.14).

$$\begin{aligned} \textstyle\sum F &= F_R + F_L = M_G a \\ \textstyle\sum \tau &= \tau_R - \tau_L = 0{,}5b\,(F_R - F_L) = I_G \alpha \end{aligned} \tag{3.14}$$

3.1.2.2 Sistema de partículas dinámicamente equivalente: versión dos partículas

Para poder sustituir las fuerzas en las ruedas por aceleraciones en el modelo (3.14) se necesita conocer la «masa» equivalente que mueve cada una de las ruedas, sin embargo ésta 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 diferencial (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 $R_L + R_R$ como se muestra en la figura 3.3. Esto es posible solamente si se cumplen las siguientes condiciones, ya expuestas en [14] y que en la presente tesis se realiza su extensión para el caso de un robot móvil (ver [14] para las demostraciones en el caso de un brazo manipulador 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*: $M_L R_L^2 + M_R R_R^2 = I_G$

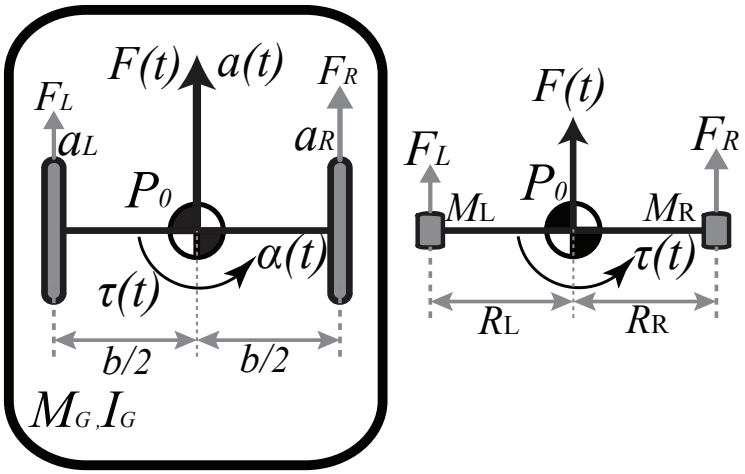


Figura 3.3: Dinámica de un robot en configuración diferencial como un cuerpo rígido y como un sistema de dos partículas equivalente

Con estas condiciones se despejan las masas y radios del sistema equivalente tal y como se muestra en la ecuación (3.15).

$$
\begin{aligned}
& M_L R_L = M_R R_R \Rightarrow M_R = \frac{R_L}{R_R} M_L \\
& M_L R_L^2 + M_R R_R^2 = I_G \Rightarrow M_L R_L^2 + \frac{R_L}{R_R} R_R^2 M_L = I_G \\
& \Rightarrow M_L R_L^2 + M_L R_L R_R = I_G \Rightarrow M_L R_L \left(R_L + R_R\right) = I_G \\
& \Rightarrow M_L = \frac{I_G}{R_L \left(R_L + R_R\right)} \Rightarrow M_R = \frac{R_L}{R_R} M_L = \frac{I_G}{R_R \left(R_L + R_R\right)} \\
& \Rightarrow M_G = M_L + M_R = \frac{I_G}{\left(R_L + R_R\right)} \left(\frac{1}{R_L} + \frac{1}{R_R}\right) = \frac{I_G}{R_L R_R} \\
& \therefore M_L = \frac{I_G}{R_L \left(R_L + R_R\right)}, M_R = \frac{I_G}{R_R \left(R_L + R_R\right)}, R_L R_R = \frac{I_G}{M_G}
\end{aligned}
\tag{3.15}
$$

De la ecuación anterior se observa que las distancias no se pueden asignar arbitrariamente. Al asignar una distancia, la otra se determina de la ecuación (3.15) y con esto las masas quedan definidas. Por simplicidad se pueden asignar para que sean iguales, por lo que $R_L = R_R = R_N$ para que las distancias entre las masas sean iguales entre sí. De esta forma se reduce (3.15) a (3.16).

$$M_L = M_R = \frac{I_G}{2R_N^2}, R_N^2 = \frac{I_G}{M_G} \tag{3.16}$$

Por facilidad de notación esta relación se expresa en función de constantes, ya que las masas M_L y M_R son proporcionales a la masa total M_G, expresado por la constante γ. Se define además un factor λ como la razón entre la masa y el momento de inercia del cuerpo rígido multiplicado por la distancia R_N entre el centro de masa y la partícula (rueda) tal y como se muestra en la ecuación (3.17).

$$\begin{aligned} M_L = M_R &= \gamma M_G \\ R_N^2 &= \frac{I_G}{M_G} \\ \lambda &= \frac{M_G R_N}{I_G} \end{aligned} \tag{3.17}$$

Los parámetros expresados en (3.17) se obtienen para diferentes formas de robots al sustituir el valor correspondiente de I_G en (3.17). Para un anillo delgado en donde $I_G = M_G R^2$ se muestra la deducción de sus parámetros en la ecuación (3.18).

$$\begin{aligned} R_N^2 &= \frac{I_G}{M_G} = \frac{M_G R^2}{M_G} \Rightarrow R_N = R \\ M_L = M_R &= \frac{I_G}{2R_N^2} = \frac{M_G R^2}{2R^2} = 0,5 M_G \\ \lambda &= \frac{M_G R_N}{I_G} = \frac{M_G R}{M_G R^2} = \frac{1}{R} \end{aligned} \tag{3.18}$$

Para un cilindro en donde $I_G = 0{,}5M_GR^2$ se muestra la deducción de sus parámetros en la ecuación (3.19).

$$\begin{aligned} R_N^2 &= \frac{I_G}{M_G} = \frac{0{,}5M_GR^2}{M_G} \Rightarrow R_N^2 = 0{,}5R^2 \Rightarrow R_N = \sqrt{0{,}5}R \\ M_L &= M_R = \frac{I_G}{2R_N^2} = \frac{0{,}5M_GR^2}{2(0{,}5R^2)} = 0,5M_G \\ \lambda &= \frac{M_GR_N}{I_G} = \frac{M_G\sqrt{0{,}5}R}{0{,}5M_GR^2} = \frac{2\sqrt{0{,}5}}{R} = \frac{1}{R\sqrt{0{,}5}} \end{aligned} \tag{3.19}$$

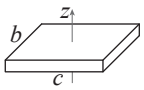
Para un prisma en donde $I_G = \frac{1}{12}\left(b^2 + c^2\right) M_G = \beta M_G, \beta = \frac{1}{12}\left(b^2 + c^2\right)$ se muestra la deducción de sus parámetros en la ecuación (3.20).

$$\begin{aligned} R_N^2 &= \frac{I_G}{M_G} = \frac{\beta M_G}{M_G} \Rightarrow R_N^2 = \beta \Rightarrow R_N = \sqrt{\beta} \\ M_L &= M_R = \frac{I_G}{2R_N^2} = \frac{\beta M_G}{2\beta} = 0,5M_G \\ \lambda &= \frac{M_GR_N}{I_G} = \frac{M_G\sqrt{\beta}}{\beta M_G} = \frac{\sqrt{\beta}}{\beta} \end{aligned} \tag{3.20}$$

Este procedimiento se puede repetir para distintas formas de robot móvil conociendo el valor correspondiente de I_G, inclusive en algunas plataformas experimentales este parámetro puede ser conocido con precisión por lo cual se pueden obtener los valores numéricos de los parámetros en (3.17) al sustituir los valores correspondientes de M_G e I_G. Algunos robots comerciales incluyen un modelo CAD (en Solidworks® o similar) los cuales permiten determinar estos parámetros con precisión ante distintas modificaciones físicas de la plataforma.

Se resumen los resultados para las formas genéricas del robot en la tabla 3.1 que muestra los parámetros (γ, λ, R_N) del sistema de dos partículas.

Tabla 3.1: Parámetros del sistema de partículas dinámicamente equivalente, caso dos partículas para el robot diferencial

Forma del Robot	Momento de Inercia	γ	R_N	λ
Anillo Delgado equivalente a (3.13) 	$I_G = M_G R^2$	0.5	R	$\frac{1}{R}$
Sólido Cilíndrico 	$I_G = \frac{M_G R^2}{2}$	0.5	$R\sqrt{0{,}5}$	$\frac{1}{R\sqrt{0{,}5}}$
Sólido Rectangular 	$I_G = \beta M_G$ $\beta = \frac{1}{12}\left(b^2 + c^2\right)$	0.5	$\sqrt{\beta}$	$\frac{\sqrt{\beta}}{\beta}$

Con estos parámetros se puede definir el modelo dinámico *local* del robot de la siguiente forma, se aplica primeramente la segunda ley de Newton al sistema de partículas mostrado en la figura 3.3 para obtener (3.21).

$$\begin{aligned} \sum F &= F_R + F_L \Rightarrow M_R a_R + M_L a_L = M_G a \\ \sum \tau &= \tau_R - \tau_L \Rightarrow R_N \left(M_R a_R - M_L a_L\right) = I_G \alpha \end{aligned} \tag{3.21}$$

Se puede afirmar que el sistema dinámico del cuerpo rígido (ecuación (3.14)) y el de partículas (ecuación (3.21)) son dinámicamente equivalentes cuando el sistema de partículas se define utilizando (3.17) y los parámetros de la tabla 3.1. Además de estas condiciones, las fuerzas y pares aplicados a (3.14) y (3.21) deben ser los mismos. En el caso de las fuerzas sí se cumple esta condición pero para que los pares sean equivalentes, las distancias deben cumplir con la condición $b = 2R_N$. Como R_N está definido en la tabla 3.1 y no puede modificarse (para mantener la condición de equivalencia dinámica), la distancia entre las ruedas del robot se deberá ajustar en el diseño de la

plataforma con el fin de utilizar el modelo equivalente para obtener la dinámica del robot. Esto puede realizarse en la etapa de diseño de la plataforma o bien al utilizar una plataforma de estructura configurable. En caso contrario, un modelo similar puede obtenerse al utilizar una equivalencia dinámica mediante tres partículas, caso que se desarrolla en la siguiente sección del presente capítulo.

Por lo tanto, al utilizar $b = 2R_N$, (3.21) es dinámicamente equivalente a (3.14). Con esto y utilizando las masas equivalentes en (3.21) se despejan las aceleraciones locales a y α tal y como se observa en (3.22).

$$\begin{aligned} a &= \gamma \frac{M_G}{M_G} (a_R + a_L) \Rightarrow a = \gamma (a_R + a_L) \\ \alpha &= \gamma \frac{M_G R_N}{I_G} (a_R - a_L) \Rightarrow \alpha = \lambda\gamma (a_R - a_L) \end{aligned} \tag{3.22}$$

Si integramos las aceleraciones de forma discreta (al integrar el modelo cinemático $a = \dot{v}$,$\alpha = \dot{\omega}$) y sustituimos en las entradas (aceleraciones) de (3.22), se obtiene el modelo dinámico *local* tal y como se muestra en (3.23), observándose que esta ecuación es lineal.

$$\begin{bmatrix} v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_{k-1} \\ \omega_{k-1} \end{bmatrix} + \begin{bmatrix} T_s & 0 \\ 0 & T_s \end{bmatrix} \begin{bmatrix} a_{k-1} \\ \alpha_{k-1} \end{bmatrix} \tag{3.23}$$

Sustituyendo (3.23) en (3.11) se obtiene el modelo dinámico global del robot tal y como se muestra en (3.24), siendo posible observar que este modelo es no lineal.

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \\ v_{k-1} \\ \omega_{k-1} \end{bmatrix} + \begin{bmatrix} v_{k-1} T_s \cos(\theta_{k-1} + 0{,}5 T_s \omega_{k-1}) \\ v_{k-1} T_s \sin(\theta_{k-1} + 0{,}5 T_s \omega_{k-1}) \\ T_s \omega_{k-1} \\ T_s a_{k-1} \\ T_s \alpha_{k-1} \end{bmatrix} \tag{3.24}$$

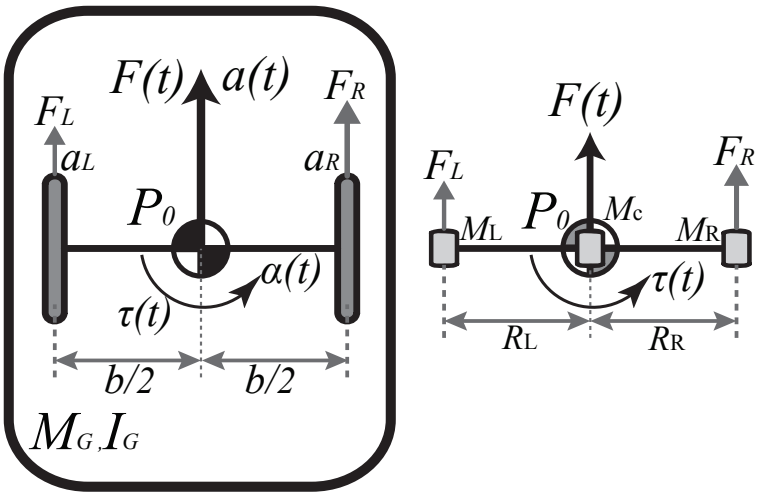


Figura 3.4: Dinámica de un robot en configuración diferencial como un cuerpo rígido y como un sistema de tres partículas equivalente

Los Modelos (3.22), (3.23) y (3.24) se utilizan junto con el método de fusión sensorial para estimar la postura global del robot tal y como se muestra en el capítulo 5.

3.1.2.3 Sistema de partículas dinámicamente equivalente: versión tres partículas

Como se mencionó anteriormente, para eliminar la restricción en la distancia entre las ruedas del robot y dejarla como variable en lugar del valor fijo de $b = 2R_N$ del caso anterior, se procede en esta sección a agregar una partícula adicional (manteniendo las dos masas izquierda y derecha pero agregando una en el centro) y repetir el análisis previo con el fin de determinar las condiciones con R_N variable.

El cuerpo rígido del robot diferencial (masa M_G, momento de inercia I_G) se puede estudiar como un sistema dinámicamente equivalente formado por tres partículas con masa M_L, M_c y M_R unidas por dos barras (conectores)

sin masa de longitud constante R_L y R_R tal y como se muestra en la figura 3.4. Nuevamente, las condiciones expuestas en [14] se extienden para el caso de tres partículas:

- Conservación de la masa: $M_G = M_L + M_c + M_R$
- Conservación del centro de masa: $M_L R_L = M_R R_R$
- Conservación del momento de inercia: $M_L R_L^2 + M_R R_R^2 = I_G$

Con estas condiciones se despejarán las masas del sistema equivalente tal y como se muestra en la ecuación (3.25).

$$
\begin{aligned}
M_G = M_L + M_c + M_R \Rightarrow M_c = M_G - M_L - M_R \\
M_L R_L = M_R R_R \Rightarrow M_R = \frac{R_L}{R_R} M_L \\
M_L R_L^2 + M_R R_R^2 = I_G \\
\Rightarrow M_L R_L^2 + \frac{R_L}{R_R} R_R^2 M_L = I_G \Rightarrow M_L R_L \left(R_L + R_R\right) = I_G \\
\Rightarrow M_L = \frac{I_G}{R_L \left(R_L + R_R\right)} \Rightarrow M_R = \frac{R_L}{R_R} M_L = \frac{I_G}{R_R \left(R_L + R_R\right)} \\
\Rightarrow M_c = M_G - M_L - M_R \\
\Rightarrow M_c = M_G - \left(M_L + M_R\right) = M_G - \left(\frac{I_G}{R_L R_R}\right) \\
\therefore M_L = \frac{I_G}{R_L \left(R_L + R_R\right)}, M_R = \frac{I_G}{R_R \left(R_L + R_R\right)}, M_c = M_G - \left(\frac{I_G}{R_L R_R}\right)
\end{aligned}
\tag{3.25}
$$

De la ecuación anterior se observa que, a diferencia del caso para dos partículas, las distancias R_R y R_L sí se pueden asignar arbitrariamente. Por simplicidad se pueden asignar para que sean iguales entre sí por lo que $R_R = R_L = R_N$; con esto se reduce (3.25) a (3.26) en donde las masas M_L y M_R son proporcionales a la masa total M_G, lo que se expresa mediante la constante γ $(0 \leq \gamma < 1)$, y de la misma forma M_c es proporcional a

M_G expresado por la constante δ ($0 \leq \delta < 1$). Adicionalmente se definen dos constantes, λ_a utilizando γ y δ, y λ_α mediante M_G, R_N e I_G.

$$\begin{aligned} M_L = M_R = \tfrac{I_G}{2R_N^2} = \gamma M_G, \quad & \lambda_a = \tfrac{\gamma}{1-\delta} \\ M_c = M_G - \left(\tfrac{I_G}{R_N^2}\right) = \delta M_G, \quad & \lambda_\alpha = \tfrac{M_G R_N}{I_G} \end{aligned} \tag{3.26}$$

Para los casos presentados en el modelo de dos partículas (anillo, circular y rectangular), se procede con el análisis para obtener las masas utilizando (3.26), considerando que se desean las mediciones sobre las ruedas del robot y que éstas se encuentran en extremos opuestos de su cuerpo. De esta forma se selecciona $R_R = R_L = R_N = 0{,}5b$. Nuevamente cabe destacar que se está escogiendo a conveniencia el $R_N = 0{,}5b$, con el fin de colocar los acelerómetros en la estructura de la plataforma sobre las ruedas del robot. Esta distancia R_N no viene determinada por las condiciones de la ecuación (3.26) a diferencia del sistema con dos partículas en la ecuación (3.16).

Se reescriben los parámetros de (3.26) para las distintas formas de robots al sustituir el valor correspondiente de I_G. Para un anillo delgado en donde $I_G = M_G R^2$ se asigna $R_N = b/2 = R$ y se muestra la deducción de sus parámetros en la ecuación (3.27).

$$\begin{aligned} M_L = M_R &= \frac{I_G}{2R_N^2} = \frac{M_G R^2}{2R^2} = 0,5 M_G, \ \gamma = 0{,}5 \\ M_c = M_G - \left(\frac{I_G}{R_N^2}\right) &= M_G - \left(\frac{M_G R^2}{R^2}\right) = 0, \ \delta = 0 \\ \lambda_\alpha &= \frac{M_G R_N}{I_G} = \frac{M_G R}{M_G R^2} = \frac{1}{R} \\ \lambda_a &= \frac{\gamma}{(1-\delta)} = 0{,}5 \end{aligned} \tag{3.27}$$

Para un cilindro en donde $I_G = 0{,}5 M_G R^2$ se asigna $R_N = b/2 = R$ y se muestra la deducción de sus parámetros en la ecuación (3.28).

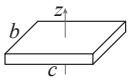
$$\begin{aligned}
M_L = M_R &= \frac{I_G}{2R_N^2} = \frac{0{,}5 M_G R^2}{2R^2} = 0,25 M_G,\ \gamma = 0,25 \\
M_c &= M_G - \left(\frac{I_G}{R_N^2}\right) = M_G - \left(\frac{0{,}5 M_G R^2}{R^2}\right) = 0,5 M_G,\ \delta = 0,5 \\
\lambda_\alpha &= \frac{M_G R_N}{I_G} = \frac{M_G R}{0{,}5 M_G R^2} = \frac{2}{R} \\
\lambda_a &= \frac{\gamma}{(1-\delta)} = \frac{0,25}{(1-0,5)} = 0{,}5
\end{aligned} \tag{3.28}$$

Para un prisma en donde $I_G = \frac{1}{12}\left(b^2 + c^2\right) M_G$ se asigna $R_N = 0{,}5b$ y se muestra la deducción de sus parámetros en la ecuación (3.29).

$$\begin{aligned}
M_L = M_R &= \frac{I_G}{2R_N^2} = \frac{(1/12)\, M_G \left(b^2 + c^2\right)}{2(0{,}5d)^2} = \frac{M_G}{6}\left(1 + (c/b)^2\right) \\
&\Rightarrow \gamma = \frac{1}{6}\left(1 + (c/b)^2\right) \\
M_c &= M_G - \left(\frac{I_G}{R_N^2}\right) = M_G - \left(\frac{M_G}{3}\left(1 + (c/b)^2\right)\right) = M_G\left(1 - \frac{1 + (c/b)^2}{3}\right) \\
&\Rightarrow M_c = \frac{M_G}{3}\left(2 + (c/b)^2\right) \Rightarrow \delta = \left(1 - \frac{1 + (c/b)^2}{3}\right) = (1 - 2\gamma) \\
\lambda_\alpha &= \frac{M_G R_N}{I_G} = \frac{M_G\,(0{,}5d)}{(1/12)\, M_G b^2 \left(1 + (c/b)^2\right)} = \frac{6}{b\left(1 + (c/b)^2\right)} = \frac{1}{b\gamma} \\
\lambda_a &= \frac{\gamma}{(1-\delta)} = \frac{\gamma}{(1-(1-2\gamma))} = 0{,}5
\end{aligned} \tag{3.29}$$

Cabe destacar que el presente análisis se puede repetir para distintas plataformas de forma algebraica si se tiene la ecuación para el I_G o bien de forma numérica sustituyendo los valores correspondientes a M_G, I_G. Se resumen los

Tabla 3.2: Parámetros del sistema de partículas dinámicamente equivalente, caso tres partículas para el robot diferencial

Forma del Robot	Momento de Inercia	γ	δ	λ_a	λ_α
Anillo Delgado 	$I_G = M_G R^2$	0.5	0	0,5	$\frac{1}{R}$
Cilindro Sólido 	$I_G = \frac{M_G R^2}{2}$	0,25	0,5	0,5	$\frac{2}{R}$
Rectángulo Sólido 	$I_G = \beta M_G$ $\beta = \frac{1}{12}\left(b^2+c^2\right)$	$\frac{1}{6}\left(1+(c/b)^2\right)$	$1-2\gamma$	0,5	$\frac{1}{b\gamma}$

resultados para las formas genéricas del robot en la tabla 3.2 que muestra los parámetros del sistema de tres partículas.

Para obtener el modelo dinámico se aplica la segunda ley de Newton al sistema de partículas mostrado en la figura 3.4 para obtener (3.30).

$$\begin{aligned} \sum F &= F_R + F_L = M_L a_L + M_c a_c + M_R a_R \\ \sum \tau &= \tau_R - \tau_L = R_N \left(M_R a_R - M_L a_L\right) \end{aligned} \tag{3.30}$$

Al aplicar el principio de D'Alembert al sistema dinámico (3.14) y al sistema dinámico por partículas (3.30) se obtienen las ecuaciones (3.31) y (3.32) respectivamente.

$$\begin{aligned} \sum F &\Rightarrow M_G a - (F_R + F_L) = 0 \\ \sum \tau &\Rightarrow I_G \alpha - (\tau_R - \tau_L) = 0 \end{aligned} \tag{3.31}$$

$$\begin{aligned} \sum F &\Rightarrow M_R a_R + M_c a_c + M_L a_L - (F_R + F_L) = 0 \\ \sum \tau &\Rightarrow R_N M_R a_R - R_N M_L a_L - (\tau_R - \tau_L) = 0 \end{aligned} \tag{3.32}$$

Al igualar (3.31) y (3.32) se obtiene la ecuación (3.33).

$$\begin{aligned} & M_G a = M_R a_R + M_c a_c + M_L a_L \\ & I_G \alpha = R_N M_R a_R - R_N M_L a_L \Leftrightarrow R_R = R_L = R_N = 0{,}5b \end{aligned} \tag{3.33}$$

Se observa de la ecuación (3.33) que el cuerpo rígido (3.14) y el sistema de partículas (3.30) son dinámicamente equivalentes al asignar $R_R = R_L = R_N = 0{,}5b$ y al utilizar los parámetros de la ecuación (3.26), definidos según la forma del robot de acuerdo con la tabla 3.2. Como se mencionó anteriormente, al utilizar $R_N = 0{,}5b$ se colocarán los acelerómetros para medir (a_R, a_L) en la estructura de la plataforma sobre las ruedas del robot.

Con estas condiciones se resuelve (3.33) para las aceleraciones locales a y α (con $a_c = a$) al sustituir (3.26) en (3.33), tal y como se muestra en la ecuación (3.34)

$$\begin{aligned} M_G a = M_R a_R + M_c a_c + M_L a_L &\Rightarrow M_G a - \delta M_G a = \gamma M_G (a_R + a_L) \\ &\Rightarrow a M_G = \frac{\gamma}{(1-\delta)} M_G (a_R + a_L) \Rightarrow a = \lambda_a (a_R + a_L) \\ I_G \alpha = M_G R_N \gamma (a_R - a_L) & \\ &\Rightarrow \alpha = \frac{M_G R_N}{I_G} \gamma (a_R - a_L) \Rightarrow \alpha = \lambda_\alpha \gamma (a_R - a_L) \\ \therefore a = \lambda_a (a_R + a_L)\,,\ & \alpha = \lambda_\alpha \gamma (a_R - a_L) \end{aligned} \tag{3.34}$$

Estas aceleraciones sirven como entrada a los modelos (3.23) y (3.24) anteriormente descritos, los cuales son utilizados en el proceso de estimación de la postura. Estos modelos (versión tres partículas) se seleccionan para los filtros de localización cooperativa de grupos de robots al ser más generales que la versión de dos partículas, pudiendo utilizarse el mismo modelo para las distintas plataformas que conforman un grupo de robots heterogéneos.

Descritos los modelos para el robot diferencial se procede con la obtención de los modelos para el robot en configuración Ackerman.

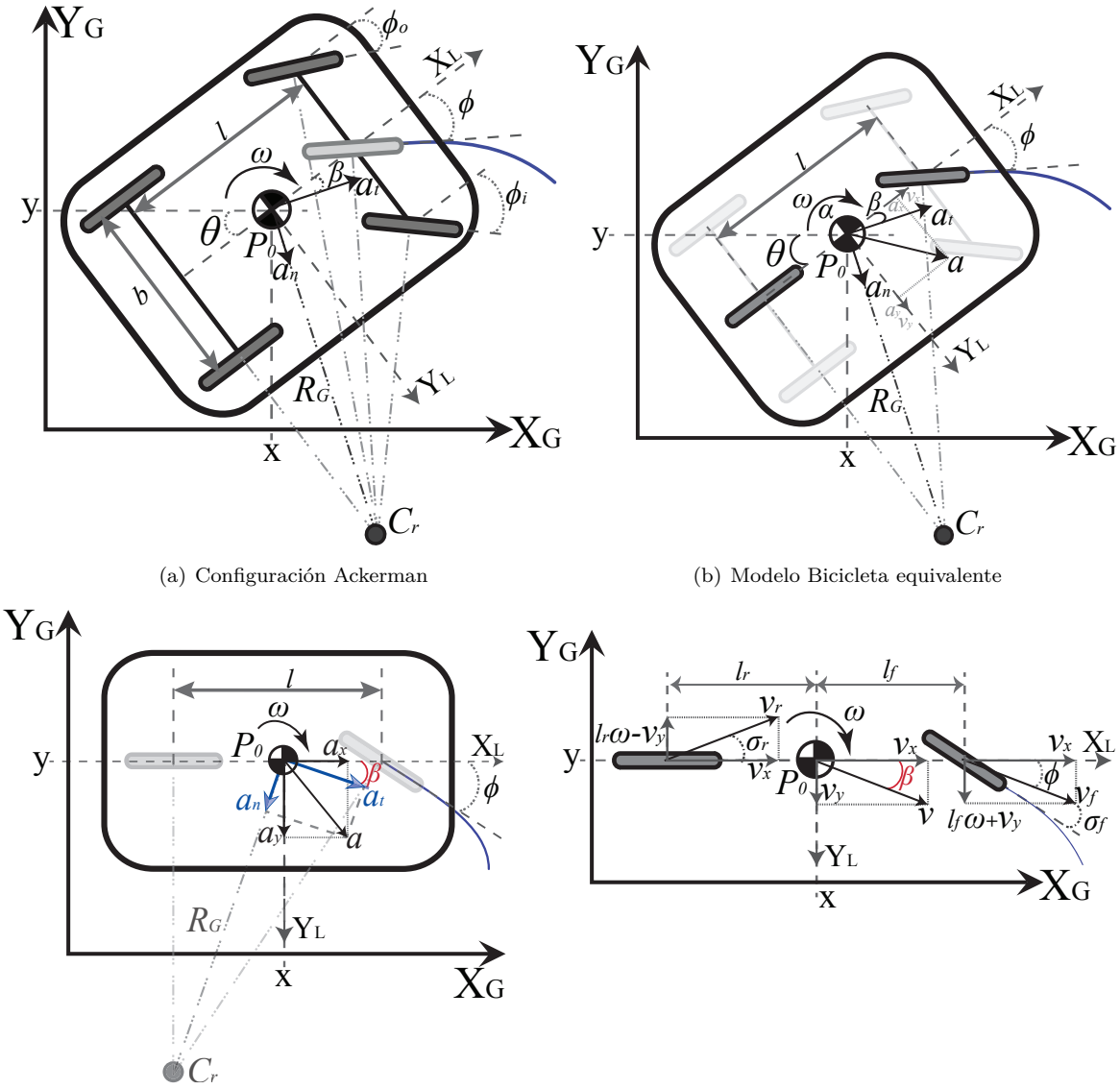
3.2 Robot móvil en configuración Ackerman

Un robot en configuración Ackerman consiste en un cuerpo rígido con centro de masa y gravedad P_0, radio de rotación R_G, masa M_G y un momento de inercia I_G con dos ruedas traseras no deformables y no orientables (fijas) separadas una distancia b entre ellas y a una distancia l de dos ruedas delanteras orientables (también a una distancia b entre ellas) tal y como se muestra en las figuras 3.5 y 3.6. La dirección Ackerman modifica la orientación de las ruedas delanteras de tal forma que, a bajas velocidades, todas las ruedas cumplen con la condición de rodamiento puro sin deslizamiento lateral [168]. Esto se produce debido a que, a baja velocidad, cada rueda sigue una trayectoria curva con radios distintos pero con un centro instantáneo de rotación común C_r (Fig. 3.1). Esta configuración se analiza utilizando el modelo de la bicicleta ([168], [139]) suponiendo que el movimiento se restringe a un plano horizontal. Con esto, se utiliza el promedio ϕ de los ángulos de las ruedas frontales (ϕ_i, ϕ_o) y las ruedas traseras se consideran como una rueda equivalente en donde se aplica la fuerza del motor.

A diferencia del caso diferencial, los de configuración Ackerman pueden ir a una velocidad en la cual se presente deslizamiento, por lo que será considerado en el modelo, a menos que se garantice una velocidad baja en la plataforma según las condiciones establecidas en [139]. Se analiza la cinemática y dinámica de este robot para obtener el modelo del robot que se utilizará en los filtros de fusión.

3.2.1 Modelo cinemático

El modelo cinemático de esta configuración representa la evolución de las velocidades del robot en un marco inercial fijo. La *postura* del robot se define mediante las coordenadas del punto $P_0 = (x, y)$ y el ángulo de avance θ en el marco de referencia *global* (X_G,Y_G) en la figura 3.5. Conociendo las veloci-



(a) Configuración Ackerman

(b) Modelo Bicicleta equivalente

(c) Cinemática del Modelo de la Bicicleta equivalente [168]

Figura 3.5: Cinemática de un robot en configuración Ackerman

dades lineales y angulares (v y ω) en el marco de referencia local (X_L,Y_L),la velocidad *global* del robot se define según la ecuación (3.35).

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix} \tag{3.35}$$

Siguiendo un procedimiento similar al descrito para el caso del robot diferencial, se discretiza e integra recursivamente (3.35) con el tiempo de muestreo T_s para obtener la postura *global* del robot $\boldsymbol{L}$ en (3.36).

$$\boldsymbol{L}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_{k-1} + T_s \begin{bmatrix} \cos\theta_{k-1} & -\sin\theta_{k-1} & 0 \\ \sin\theta_{k-1} & \cos\theta_{k-1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}_{k-1} \tag{3.36}$$

Durante el movimiento de rotación del robot (cuando sigue una trayectoria circular) y debido al deslizamiento del mismo, el robot se encuentra en traslación y en rotación simultáneas. Para describir este comportamiento de la aceleración a se obtienen sus componentes a_x, a_y del centro de gravedad P_0 utilizando las aceleraciones normal y tangencial (a_n,a_t), mostradas en la figura 3.5(c), mediante el procedimiento descrito en [168].

El componente tangencial a_t tiene la misma dirección que la velocidad resultante v del centro de gravedad, la cual se encuentra a un ángulo β del eje de avance X_L tal y como se muestra en la figura 3.5(c). Este ángulo β se conoce como el ángulo de deslizamiento lateral del vehículo. Las aceleraciones tangencial a_t y normal a_n se descomponen según los ejes locales tal y como se muestra en la ecuación (3.37).

$$\begin{aligned} a_t &= \begin{cases} \dot{v}_x \text{ Eje } \mathrm{X}_L \\ \dot{v}_y \text{ Eje } \mathrm{Y}_L \end{cases} \\ a_n &= \begin{cases} -\left(v^2/R\right)\sin\beta = -v\omega \cdot \sin\beta = -v_y\omega \text{ Eje } \mathrm{X}_L \\ \left(v^2/R\right)\cos\beta = v\omega \cdot \cos\beta = v_x\omega \text{ Eje } \mathrm{Y}_L \end{cases} \end{aligned} \tag{3.37}$$

De esta forma, los componentes a_x, a_y de la aceleración a se definen al agrupar en los ejes locales (X_L,Y_L) los términos de las aceleraciones a_n y a_t [168] tal y como se muestra en la ecuación (3.38).

$$\begin{aligned} a_x &= \dot{v_x} - v_y\omega \\ a_y &= \dot{v_y} + v_x\omega \end{aligned} \tag{3.38}$$

Finalmente, se observa en la ecuación (3.39) la relación cinemática entre el ángulo de la dirección Ackerman ϕ y el ángulo de avance del robot ω.

$$\omega = (v_x \tan\phi)/l \tag{3.39}$$

Con esto se procede al análisis de la dinámica del robot.

3.2.2 Modelo dinámico

Este modelo representa la evolución de las aceleración linear y angular del robot (a_x,a_y,α) en términos de las fuerzas que han sido aplicadas por los motores en las ruedas frontal F_f y trasera F_r y el par angular τ. De forma similar al caso diferencial, los modelos en la literatura ([164], [174], [43], [4] y [130]) describen la dinámica en los ejes globales, por lo que nuevamente se requiere aplicar el análisis por partículas dinámicamente equivalentes para obtener los modelos a utilizar en los filtros de fusión.

En este caso se utilizará el caso de tres partículas. El cuerpo rígido del robot Ackerman (masa M_G, momento de inercia I_G) se estudia como un sistema dinámicamente equivalente formado por tres partículas con masa M_r, M_c y M_f unidas por dos barras (conectores) sin masa, de longitud constante R_r y R_f tal y como se muestra en la figura 3.6. Nuevamente, las condiciones expuestas en [14] se deben aplicar:

- Conservación de la masa: $M_G = M_f + M_c + M_r$
- Conservación del centro de masa: $M_f R_f = M_r R_r$
- Conservación del momento de inercia: $M_f R_f^2 + M_r R_r^2 = I_G$

Con estas condiciones se despejarán las masas del sistema equivalente (siguiendo una demostración similar a la ecuación (3.25)) mostradas en la ecuación (3.40).

$$M_f = \frac{I_G}{R_f\left(R_f + R_r\right)}, M_r = \frac{I_G}{R_r\left(R_f + R_r\right)}, M_c = M_G - \left(\frac{I_G}{R_f R_r}\right) \tag{3.40}$$

Las distancias, al ser un modelo de 3 partículas equivalentes, pueden ser asignadas a conveniencia. Por simplicidad se asignan para que sean iguales entre sí por lo que $R_f = R_r = R_N$; con esto se reduce (3.40) a (3.41) en donde las masas M_f y M_r son proporcionales a la masa total M_G, lo que se expresa mediante la constante γ $(0 \leq \gamma < 1)$, y de la misma forma M_c es proporcional a M_G expresado por la constante δ $(0 \leq \delta < 1)$. Adicionalmente se definen dos constantes, λ_a utilizando γ y δ, y λ_α mediante M_G, R_N e I_G.

$$\begin{array}{ll} M_f = M_r = \frac{I_G}{2R_N^2} = \gamma M_G, & \lambda_a = \frac{\gamma}{1-\delta} \\ M_c = M_G - \left(\frac{I_G}{R_N^2}\right) = \delta M_G, & \lambda_\alpha = \frac{M_G R_N}{I_G} \end{array} \tag{3.41}$$

En el caso de los robots de configuración Ackerman, la gran mayoría son de forma rectangular (con longitud c y ancho b), por lo que, para este caso, se utilizará $R_f = l_f = R_r = l_r = 0{,}5l$ según la figura 3.6. De esta forma, se obtienen los parámetros (siguiendo una demostración similar a la ecuación (3.29)) al sustituir $I_G = \frac{1}{12}\left(b^2 + c^2\right) M_G$ en (3.41) asignando $R_N = 0{,}5l = 0{,}5c$. Con esto, se muestran los parámetros en la ecuación (3.42). Esta deducción de los parámetros se puede hacer de forma exacta para otras formas de la plataforma si se posee la ecuación de I_G o bien numéricamente según las características físicas de la misma (M_G, I_G, R_f, R_r).

$$\begin{array}{cc} I_G = \frac{1}{12} M_G\left(b^2 + c^2\right), & \gamma = \frac{1}{6}\left(1 + (b/c)^2\right) \\ \delta = \left(1 - \frac{1+(b/c)^2}{3}\right), & \lambda_\alpha = \frac{6}{c\left(1+(b/c)^2\right)} \\ \lambda_a = 0{,}5, & \lambda_\alpha \gamma = 1/c \end{array} \tag{3.42}$$

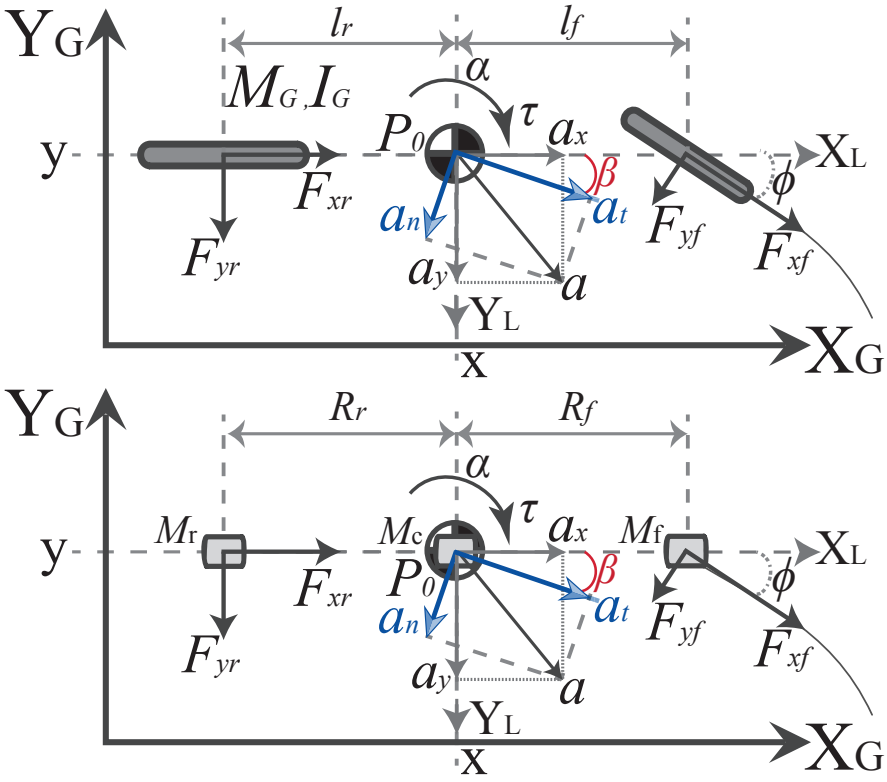


Figura 3.6: Dinámica de un robot en configuración Ackerman como un cuerpo rígido y como un sistema de tres partículas equivalente

Para obtener el modelo dinámico se aplica la segunda ley de Newton al cuerpo rígido y al sistema de partículas mostrado en la figura 3.6 para obtener (3.43) y (3.44).

$$\begin{aligned} \Sigma F_x &= (F_{xf}\cos\phi - F_{yf}\sin\phi) + F_{xr} = M_G a_x \\ \Sigma F_y &= (F_{yf}\cos\phi + F_{xf}\sin\phi) + F_{yr} = M_G a_y \\ \Sigma\tau &= l_f\,(F_{yf}\cos\phi + F_{xf}\sin\phi) - l_r F_{yr} = I_G\alpha \end{aligned} \tag{3.43}$$

$$\begin{aligned} \Sigma F_x &= (F_{xf}\cos\phi - F_{yf}\sin\phi) + F_{xr} = M_f a_{f,x} + M_c a_{c,x} + M_r a_{r,x} \\ \Sigma F_y &= (F_{yf}\cos\phi + F_{xf}\sin\phi) + F_{yr} = M_f a_{f,y} + M_c a_{c,y} + M_r a_{r,y} \\ \Sigma\tau &= R_f\,(F_{yf}\cos\phi + F_{xf}\sin\phi) - R_r F_{yr} = R_f M_f a_{f,y} - R_r M_r a_{r,y} \end{aligned} \tag{3.44}$$

Aplicando el principio de D'Alembert a (3.43) y (3.44) se obtienen (3.45) y (3.46) respectivamente.

$$\begin{aligned} M_G a_x - (F_{xf}\cos\phi - F_{yf}\sin\phi + F_{xr}) &= 0 \\ M_G a_y - (F_{yf}\cos\phi + F_{xf}\sin\phi + F_{yr}) &= 0 \\ I_G\alpha - (l_f F_{yf}\cos\phi + l_f F_{xf}\sin\phi - l_r F_{yr}) &= 0 \end{aligned} \tag{3.45}$$

$$\begin{aligned} M_f a_{f,x} + M_c a_{c,x} + M_r a_{r,x} - (F_{xf}\cos\phi - F_{yf}\sin\phi + F_{xr}) &= 0 \\ M_f a_{f,y} + M_c a_{c,y} + M_r a_{r,y} - (F_{yf}\cos\phi + F_{xf}\sin\phi + F_{yr}) &= 0 \\ (R_f M_f a_{f,y} - R_r M_r a_{r,y}) - (R_f F_{yf}\cos\phi + R_f F_{xf}\sin\phi - R_r F_{yr}) &= 0 \end{aligned} \tag{3.46}$$

Al igualar (3.45) y (3.46) se obtiene (3.47).

$$\begin{aligned} M_G a_x &= M_f a_{f,x} + M_c a_{c,x} + M_r a_{r,x} \\ M_G a_y &= M_f a_{f,y} + M_c a_{c,y} + M_r a_{r,y} \\ I_G\alpha &= (R_f M_f a_{f,y} - R_r M_r a_{r,y}) \Leftrightarrow R_f = l_f, R_r = l_r \end{aligned} \tag{3.47}$$

Se observa de la ecuación (3.47) que el cuerpo rígido y el sistema de partículas son dinámicamente equivalentes al utilizar los parámetros de la ecuación

(3.42) y cuando $R_f = l_f, R_r = l_r$. Como se mencionó anteriormente se utilizará $R_f = l_f = R_r = l_r = 0{,}5l$ para el sistema equivalente del robot Ackerman (esta es la distancia a la que se colocarán los acelerómetros, en el centro de los ejes de la plataforma). Utilizando estas condiciones se resuelve para las aceleraciones locales a_x, a_y y α (con $a_c = a$) tal y como se muestra en la ecuación (3.48). Éstas resultan ser similares al caso diferencial (ecuación (3.34)) pero con deslizamiento.

$$\begin{aligned}
&M_G a_x – \delta M_G a_x = \gamma M_G \left(a_{f,x} + a_{r,x}\right) \Rightarrow a_x = \frac{\gamma}{1–\delta}\left(a_{f,x} + a_{r,x}\right) = \lambda_a \left(a_{f,x} + a_{r,x}\right) \\
&M_G a_y – \delta M_G a_y = \gamma M_G \left(a_{f,y} + a_{r,y}\right) \Rightarrow a_y = \frac{\gamma}{1–\delta}\left(a_{f,y} + a_{r,y}\right) = \lambda_a \left(a_{f,y} + a_{r,y}\right) \\
&\alpha = \frac{\gamma M_G}{I_G}\left(R_f a_{f,y} – R_r a_{r,y}\right) = \gamma \frac{M_G}{I_G} R_N \left(a_{f,y} – a_{r,y}\right) = \lambda_\alpha \gamma \left(a_{f,y} – a_{r,y}\right) \\
&\therefore a_x = \lambda_a \left(a_{f,x} + a_{r,x}\right),\ a_y = \lambda_a \left(a_{f,y} + a_{r,y}\right),\ \alpha = \lambda_\alpha \gamma \left(a_{f,y} – a_{r,y}\right)
\end{aligned} \tag{3.48}$$

Ahora bien, estas aceleraciones (3.48) se sustituyen en (3.38) para tomar en cuenta los componentes de la aceleración normal, tal y como se muestra en la ecuación (3.49).

$$\begin{aligned}
\dot{v}_x &= v_y \omega + \lambda_a \left(a_{f,x} + a_{r,x}\right) = v_y \omega + \lambda_a u_1 \\
\dot{v}_y &= -v_x \omega + \lambda_a \left(a_{f,y} + a_{r,y}\right) = -v_x \omega + \lambda_a u_2 \\
\dot{\omega} &= \lambda_\alpha \gamma \left(a_{f,y} – a_{r,y}\right) = \lambda_\alpha \gamma u_3
\end{aligned} \tag{3.49}$$

La ecuación (3.49) corresponde al modelo dinámico local en el que se han definido las entradas u_1, u_2, u_3 como los respectivos términos de la aceleraciones de entrada en las ecuaciones de $v_x, v_y, \dot{\omega}$ para facilitar la notación. Se observa que el modelo (3.49) del robot Ackerman con deslizamiento, corresponde a un modelo no lineal, con lo que resulta conveniente aproximar la solución de la integral discreta mediante el método numérico de Euler, pudiendo utilizarse otros métodos numéricos o resolver la integral exacta (siguiendo un procedimiento similar a la cinemática diferencial) si se desea más precisión o se disponen de más recursos en la plataforma. De esta forma, obteniendo

la discretización mediante el método de Euler, el modelo dinámico del robot se obtiene en (3.50).

$$\begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}_k = \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}_{k-1} + T_s \begin{bmatrix} v_y\omega + \lambda_a u_1 \\ -v_x\omega + \lambda_a u_2 \\ \lambda_\alpha \gamma u_3 \end{bmatrix}_{k-1} \tag{3.50}$$

Al sustituir (3.50) como entradas de (3.36) se obtiene el modelo dinámico *global* del robot tal y como se muestra en (3.51).

$$\begin{bmatrix} x \\ y \\ \theta \\ v_x \\ v_y \\ \omega \end{bmatrix}_k = \begin{bmatrix} x \\ y \\ \theta \\ v_x \\ v_y \\ \omega \end{bmatrix}_{k-1} + T_s \begin{bmatrix} v_x \cos\theta - v_y \sin\theta \\ v_x \sin\theta + v_y \cos\theta \\ \omega \\ v_y\omega + \lambda_a u_1 \\ -v_x\omega + \lambda_a u_2 \\ \lambda_\alpha \gamma u_3 \end{bmatrix}_{k-1} \tag{3.51}$$

Por último, en caso de que el robot viaje a baja velocidad, se puede aprovechar la condición de no deslizamiento $v_y \approx 0$ ([139], [168]) para obtener los modelos local y global simplificados, tal y como se muestra en (3.52) y (3.53).

$$\begin{bmatrix} v_x \\ \omega \end{bmatrix}_k = \begin{bmatrix} v_x \\ \omega \end{bmatrix}_{k-1} + T_s \begin{bmatrix} \lambda_a u_1 \\ \lambda_\alpha \gamma u_3 \end{bmatrix}_{k-1} \tag{3.52}$$

$$\begin{bmatrix} x \\ y \\ \theta \\ v_x \\ \omega \end{bmatrix}_k = \begin{bmatrix} x \\ y \\ \theta \\ v_x \\ \omega \end{bmatrix}_{k-1} + T_s \begin{bmatrix} v_x \cos\theta \\ v_x \sin\theta \\ \omega \\ \lambda_a u_1 \\ \lambda_\alpha \gamma u_3 \end{bmatrix}_{k-1} \tag{3.53}$$

Los modelos (3.36), (3.48) y de (3.50) a (3.53) serán utilizados en conjunto con el método de fusión sensorial para estimar la postura del robot, tal y como se mostrará en el capítulo 5.

3.3 Modelo dinámico de los motores

En caso de no disponer de acelerómetros en la plataforma, se puede identificar un modelo para las velocidades de las ruedas del motor y calcular su derivada para obtener una aproximación de la aceleración lineal o angular de cada rueda. Estos modelos pueden ser utilizados como entrada a los modelos basados en el sistema de partículas ((3.22),(3.34) y (3.52)) los cuales a su vez funcionan como entrada del modelo dinámico local (3.23) o global (3.24) para el robot diferencial o en el modelo del caso sin deslizamiento del robot Ackerman (3.53). Este tipo de modelos son comunes y suelen obtenerse para realizar el diseño del controlador PID de regulación de velocidad de los motores. En general se utilizan una serie de escalones positivos y negativos en la acción de control con el fin de obtener la función de transferencia de bucle abierto del motor del robot. A manera de ejemplo se utilizará el modelo de velocidad de la ecuación (3.54) el cual es obtenido para los motores DC de un robot diferencial construido utilizando la plataforma LEGO MINDSTORMS NXT, tal y como se detalla en el apéndice A.

$$G_{gMA} = \frac{0{,}1276}{0{,}1235\ s + 1} \tag{3.54}$$

Para determinar la aceleración de las ruedas se deriva la ecuación (3.54) para obtener (3.55).

$$G_{gAcc} = \frac{a_{ccRueda}}{V_{control}} = \frac{0{,}1276\ s}{0{,}1235\ s + 1} \tag{3.55}$$

Este modelo relaciona la acción de control $V_{control}$ con la aceleración de la rueda $a_{ccRueda}$. Se discretiza (3.55) con un periodo de muestreo $T_s = 50ms$ con lo que se obtiene (3.56). Ésta se escribe en forma de ecuación en diferencias en (3.57) lo que facilita su implementación dentro del robot.

$$G_{gAcc}(z) = \frac{1{,}033z - 1{,}033}{z - 0{,}6671} = \frac{1{,}033 - 1{,}033z^{-1}}{1 - 0{,}6671z^{-1}} \tag{3.56}$$

$$a_{R,k} = 1{,}033V_{c,k} - 1{,}033V_{c,k-1} + 0{,}6671a_{R,k-1} \tag{3.57}$$

Con los modelos propuestos para las distintas plataformas se procede en el siguiente capítulo con la selección del método de fusión a utilizar en la definición de los algoritmos de fusión basados en eventos del capítulo 5.

3.4 Conclusiones del Capítulo

En el presente capítulo se obtuvieron diversos modelos cinemáticos y dinámicos para un robot móvil en configuración diferencial y Ackerman:

- Se obtuvieron tres versiones del modelo cinemático del robot diferencial (ecuaciones (3.10), (3.11) y (3.12)), las cuales se utilizan de acuerdo a los recursos computacionales disponibles en la unidad de control del robot.

- Se desarrollaron los modelos dinámicos basados en sistemas de partículas dinámicamente equivalentes para un robot móvil en configuración diferencial (ecuaciones (3.22),(3.34)) y Ackerman (ecuaciones (3.48),(3.49)), de aplicación sencilla a sistemas de recursos limitados, ya que permiten obtener la dinámica del robot al utilizar únicamente acelerómetros colocados en la posición adecuada sobre la plataforma, o bien mediante un modelo dinámico identificado del movimiento de los motores del robot (ecuaciones (3.55),(3.57)).

- Los parámetros de los modelos obtenidos (tablas 3.1 y 3.2, ecuación (3.42)) pueden ser obtenidos de forma exacta para múltiples plataformas móviles si se conocen con precisión los valores de M_G (masa), I_G (momento de inercia) y la distancia R_N (entre las ruedas del robot) según su forma geométrica. Estos parámetros también pueden determinarse experimentalmente, o bien mediante un modelo CAD del robot (en Solidworks® o similar). Esto resulta conveniente en caso de tener que realizar modificaciones físicas a la plataforma o cuando su forma física es compleja.

- Por último, los modelos dinámicos obtenidos se utilizan como entrada de los modelos cinemáticos, con lo que se obtienen los modelos de la postura del robot a utilizar en la fusión sensorial (ecuaciones (3.23), (3.24), (3.50), (3.51), (3.52) y (3.53)). Estos modelos son fundamentales en el desarrollo del método de fusión por eventos para plataformas de recursos limitados, ya que permiten la estimación en cascada de la postura del robot, tal y como se expone en el siguiente capítulo.

4 | Estimación de Estados y Fusión Sensorial

En el presente capítulo se realiza una descripción de los métodos existentes en cuanto a estimación de estados en sistemas discretos, con el fin de seleccionar un método adecuado para realizar la fusión sensorial para la mejora de la localización de un robot móvil de recursos limitados. Se expone primeramente la notación básica utilizada en el capítulo así como una introducción al problema de estimación. Posteriormente se describen las técnicas más relevantes en las categorías de observadores de estados y filtros de estimación, tanto para sistemas lineales como no lineales. Con base en esta descripción se procede a seleccionar la técnica de estimación que se utilizará en el capítulo siguiente para definir los algoritmos de fusión basados en eventos.

4.1 Introducción y notación

Con el fin de controlar un proceso dinámico observable y controlable, con estado x_k que es afectado por el ruido w_k, se requiere obtener las variables de interés en el mismo al observar su salida y_k utilizando un sensor, sujeto al ruido de medición n_k, que obtiene la medición z_k de los estados a controlar. De esta forma se establece el bucle o lazo de control mostrado en la figura 4.1 en donde el controlador buscará que el error e_k, calculado como la diferencia entre la referencia deseada r_k y z_k, sea nulo: $e_k = r_k - z_k = 0$.

En el caso de no poder obtener la medición directa o indirecta de todos los estados a controlar, se modifica el lazo de control con el fin de incluir un estimador, tal y como se muestra en la figura 4.2. En general, los métodos de estimación buscan obtener el valor de todos los estados de un sistema observable [151] o de ciertos estados de interés sin posibilidad de sensorización, al utilizar las mediciones de las entradas u_k y salidas del sistema medidas a través de uno o varios sensores, z_k. Además de las aplicaciones en control,

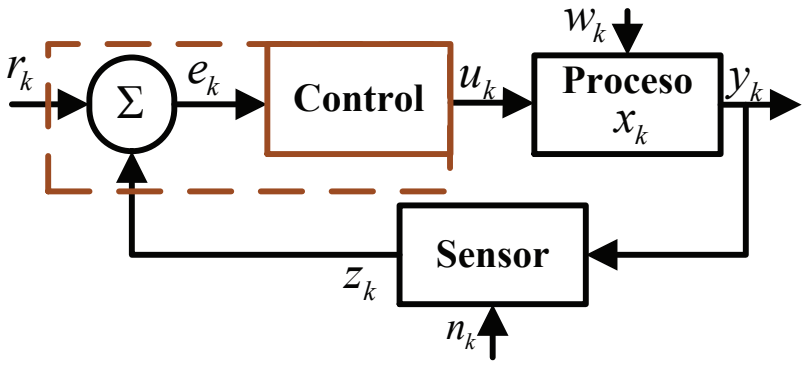


Figura 4.1: Lazo de control realimentado

los estimadores pueden ser utilizados en tareas de identificación de sistemas, así como en el monitoreo y detección de fallos en los mismos.

Para realizar la estimación se requiere conocer un modelo del sistema, aunque algunos de estos métodos requieren además información sobre las propiedades estadísticas de los ruidos w_k y n_k. La estimación del estado se considera óptima en caso de que el error de estimación $(x_k - \hat{x}_k)$ sea acotado o cero. El estado estimado $\hat{x}_k$ es utilizado en el controlador para minimizar e_k mediante la ley de control que se diseña según el proceso y estado a controlar.

Con el fin de simplificar la notación en el presente capítulo, los modelos de los procesos a utilizar en los algoritmos se definen como *lineales* cuando pueden escribirse tal y como se muestra en la ecuación (4.1), con el vector de entrada $\boldsymbol{u}_k \in \Re^u$, vector de medición $\boldsymbol{z}_k \in \Re^m$ y vector de estados $\boldsymbol{x}_k \in \Re^n$ además de la matriz de estados A_k, de entradas B_k y de medición H_k (constantes en caso de sistemas invariantes en el tiempo). Por el contrario, si el modelo se puede representar mediante la ecuación (4.2) se considera un modelo *no lineal*, en donde una función no lineal $\boldsymbol{f}$ se requiere para relacionar el estado x_k en el instante actual k con el estado en el instante anterior $k-1$, y la función no lineal $\boldsymbol{h}$ que relaciona x_k con z_k. En ambas ecuaciones se definen

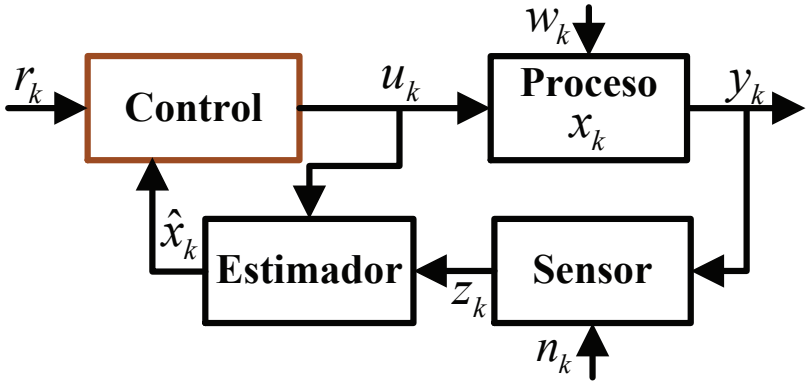


Figura 4.2: Lazo de control realimentado con estimador del estado

los términos $\boldsymbol{w}_k$, $\boldsymbol{n}_k$ correspondientes al ruido del proceso y de la medición respectivamente [151].

$$\begin{aligned} \mathbf{x_k} &= \mathbf{A_{k-1}}\mathbf{x}_{k-1} + \mathbf{B_{k-1}}\mathbf{u}_{k-1} + \mathbf{w}_{k-1} \\ \mathbf{z}_k &= \mathbf{H_k}\mathbf{x}_k + \mathbf{n}_k \end{aligned} \tag{4.1}$$

$$\begin{aligned} \mathbf{x}_k &= f\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}\right) \\ \mathbf{z}_k &= h\left(\mathbf{x}_k, \mathbf{n}_k\right) \end{aligned} \tag{4.2}$$

Se exponen a continuación distintos métodos de estimación existentes así como la elección del método a utilizar en la localización de robots móviles de recursos limitados.

4.2 Observadores de estado

4.2.1 Sistemas Lineales

Este tipo de estimadores requieren únicamente información del modelo del sistema observable para obtener $\hat{x}_k$, éste se considera un modelo no estocástico del sistema, sin ruidos en el proceso o en la medición, por lo que se

tendría por ejemplo $z_k = y_k = Cx_k$ en el caso lineal invariante en el tiempo (LTI), siendo **C** la matriz de salida del modelo del sistema. De esta forma, se puede obtener la estimación del estado utilizando el modelo preciso del sistema lineal al agregar un factor de corrección que tome en cuenta el error de estimación a partir de la salida medida del sistema y_k y la salida del estimador $\hat{y}_k = C\hat{x}_k$, ponderado mediante una matriz de ganancia K_o tal y como se muestra en la ecuación (4.3). Esta ecuación es conocida como observador lineal de estado completo u observador de Luenberger. El desempeño del observador se establece según la elección de K_o, la cual se realiza buscando que la dinámica del error de estimación sea más rápida que la del sistema mediante técnicas de ubicación de polos, de forma que $\hat{x}_k$ alcance el valor real de x_k.

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_{k-1} + K_o\left(y_{k-1} - C\hat{x}_k\right) \tag{4.3}$$

Cabe destacar que la matriz de ganancias K_o es constante en la mayoría de diseños y calculada fuera de línea, por lo que, en general, no se pueden ajustar dinámicamente en tiempo real, pero, por otra parte, esto disminuye en gran medida el consumo de recursos computacionales (memoria y tiempo de cómputo). Para realizar la fusión de distintos sensores, con el fin de estimar el estado completo, se modifican las dimensiones correspondientes en la salida, matriz C y ganancia K_o. Sin embargo, se observa que en el diseño de este tipo de observadores no se toma en cuenta la precisión de cada sensor durante el diseño o bien para el cálculo de la ganancia del observador. Esto impediría, por ejemplo, dar más importancia en la fusión sensorial a uno u otro sensor según sus características físicas.

4.2.2 Sistemas No Lineales

Para el caso de sistemas no lineales, se recurre a técnicas de Lyapunov para sintetizar, según el modelo del sistema, un observador estable de forma que se garantice que el error de estimación decaiga asintóticamente a cero [82, 22], o bien se utilizan observadores predefinidos (de los cuales se ha probado previamente su estabilidad) si el modelo cumple con alguna estructura

previamente estudiada (ver [21] para una amplia descripción de estos métodos). Con la síntesis de los observadores no lineales se obtienen un conjunto de ganancias K_o (las cuales suelen ser no lineales) para incorporar el error de estimación $(y_{k-1} - h(\hat{x}_{k-1}))$ al modelo no lineal tal y como se muestra en la ecuación (4.4), la cual se conoce como observador no lineal de estado completo.

$$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}\right) + K_o\left(k, y_{k-1} - h(\hat{x}_{k-1})\right) \tag{4.4}$$

Ejemplos de síntesis de observadores no lineales se pueden encontrar en [82, 81, 65] con aplicaciones en estimación de la velocidad de un vehículo y para localización y navegación inercial. Debido a que el diseño de observadores está basado fuertemente en el modelo del sistema, se requiere en general un modelo muy preciso del mismo siendo por lo general complejo y difícil de parametrizar, principalmente en los casos de robots móviles navegando en el entorno. Además, al requerir la utilización de técnicas de Lyapunov para obtener la ganancia, se requiere un tiempo considerable en el diseño del observador el cual además es poco flexible, ya que no se puede modificar fácilmente en caso de alguna variación en el modelo o bien al agregar o retirar un sensor, requiriendo repetir el diseño en estos casos.

Debido a esto, son técnicas de difícil aplicación a fusión sensorial además de que no consideran ruidos en el proceso y medición durante el diseño y síntesis del observador, por lo que se requiere comprobar la robustez del observador en presencia de estas perturbaciones posterior a su diseño, pudiendo variar según el comportamiento del sistema en el transcurso del tiempo. Aun así, una vez superada la fase de diseño y tras obtener las ganancias del observador (fuera de línea mediante Lyapunov), pueden obtenerse buenos resultados en sistemas con niveles bajos de ruido, con la ventaja de tener una utilización de recursos reducida, siempre que K_o sea constante o sin operaciones complejas.

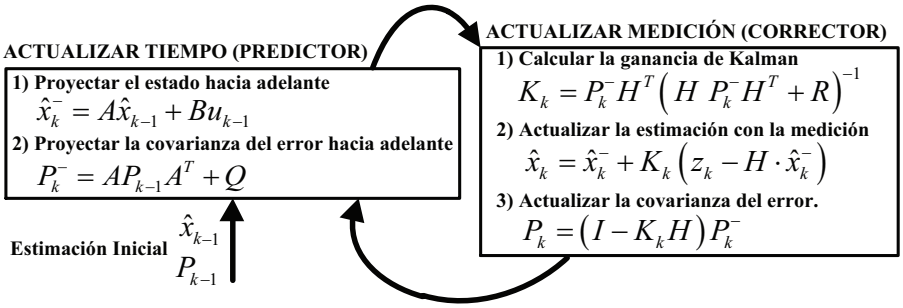


Figura 4.3: Estructura Predicción/Corrección en el Filtro de Kalman [165]

4.3 Filtros para la estimación de estados

Los filtros aplicados a la estimación de estados son observadores que toman en cuenta la presencia del ruido en el proceso w_k y medición n_k al considerar un modelo estocástico del sistema (por ejemplo, las ecuaciones (4.1) y (4.2)). De esta forma, los filtros de estimación requieren, además de diseñar la ganancia K_k para incorporar el error de estimación, de un método para administrar la forma en la que se agrega la información asociada a (w_k, n_k) en el observador (mediante K_k), así como para estimar el efecto de (w_k, n_k) en el sistema conforme éste evoluciona en el tiempo. La forma en la que se realizan estas tareas difiere según el filtro de estimación, pero en general se recurren a las herramientas provistas en la teoría de probabilidad y estadística para describir las propiedades de (w_k, n_k) y a las técnicas de optimización para obtener K_k de tal forma que se minimice un índice de desempeño [151].

Además, con el fin de obtener algoritmos computacionalmente eficientes, los filtros se expresan como métodos recursivos para aprovechar la información estimada en instantes anteriores, por lo que normalmente se dividen en las etapas de *predicción* (a *priori*) en la que se utiliza el modelo para actualizar la estimación en el tiempo (desde el instante anterior al actual) y la etapa de corrección (a *posteriori*) que incorpora la medición de los sensores en la estimación tal y como se muestra en la figura 4.3 [151, 21].

4.3.1 Filtros de estimación para sistemas lineales

Para el desarrollo de los filtros de estimación es necesario conocer las propiedades estadísticas (valor esperado, varianza, covarianza, función de densidad de probabilidad *pdf*, etc.) de una variable aleatoria X, éstas se resumen convenientemente en el apéndice B (un desarrollo más extenso puede consultarse en [151]). Entre éstas propiedades conviene destacar la evolución en el tiempo de la media $\bar{x}_k$ y la covarianza del estado x_k, P_k, de un sistema lineal [151]. La media se obtiene al obtener el valor esperado del modelo (4.1) tal y como se muestra en la ecuación (4.5), en donde se considera que w_k tiene media cero.

$$\begin{aligned} x_k &= A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \\ \Rightarrow E(x_k) = \bar{x}_k &= A_{k-1}\bar{x}_{k-1} + B_{k-1}u_{k-1} \end{aligned} \tag{4.5}$$

Con este resultado se puede obtener la evolución de la covarianza P_k de x_k según se muestra en la ecuación (4.6), al utilizar la definición de covarianza (ecuación (B.5), apéndice B) y considerando que el ruido w_k es gaussiano y blanco con covarianza Q_k.

$$\begin{aligned} P_k &= E\left[(x_k - \bar{x}_k)(x_k - \bar{x}_k)^T\right] \\ &= A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1} \end{aligned} \tag{4.6}$$

4.3.1.1 Filtro de Kalman (KF):

En el caso de sistemas lineales, el filtro de estimación por excelencia es el filtro de Kalman (*KF*,figura 4.3) [91, 64, 165, 151], el cual busca estimar el estado x_k del sistema lineal (modelo (4.1)) considerando que tanto w_k como n_k (ruido del proceso y medición) se pueden caracterizar mediante una distribución de probabilidad independiente, blanca y normal (Gaussiana) con media cero [151].

Para realizar la estimación, la información de la que se dispone corresponde a las matrices de covarianza Q_k para w_k y R_k para n_k, las cuales son variantes en el tiempo pero asumibles como constantes en ciertas condiciones, además de las matrices A, B y H del modelo (4.1) (*LTI*) junto con las

entradas u_k y mediciones z_k realizadas. Con esto, el filtro debe obtener la estimación del estado $\hat{x}$ junto con la covarianza P_k del error de esta estimación $\epsilon_{x,k} = (x_k - \hat{x}_k)$. Para un sistema lineal [151], se inicia el algoritmo con la actualización temporal (o predicción a *priori*), al calcular la evolución en el tiempo de $\hat{x}_k$ y P_k. Ésta se define recursivamente mediante la ecuación (4.7) en la que se ha utilizado el modelo (4.1) para definir $\hat{x}_k^-$ y para P_k^- se emplea la ecuación (4.6) pero utilizando $\hat{x}$ en lugar de $\bar{x}$ (ver apéndice B y [151]). Cabe destacar que el superíndice "$^-$" indica la estimación a *priori* de la variable.

$$\begin{aligned} \hat{x}_k^- &= A\hat{x}_{k-1} + Bu_{k-1} + w_{k-1} \\ P_k^- &= E\left[\left(x_k - \hat{x}_k^-\right)\left(x_k - \hat{x}_k^-\right)^T\right] = AP_{k-1}A^T + Q_k \end{aligned} \tag{4.7}$$

Para incorporar la información de la medición de los sensores en la etapa a *posteriori*, el filtro de Kalman estimará $\hat{x}_k$ según la ecuación recursiva (4.8), en la cual $\hat{x}_k^-$ se obtiene de la actualización temporal realizada con la ecuación (4.7).

$$\begin{aligned} \hat{x}_k &= \hat{x}_k^- + K_k\left[z_k - H\hat{x}_k^-\right] \\ z_k &= H_k x_k + n_k \end{aligned} \tag{4.8}$$

La obtención de la ganancia del filtro de Kalman K_k se realiza según un criterio basado en el error de estimación $\epsilon_{x,k} = (x_k - \hat{x}_k)$. Su valor esperado $E(\epsilon_{x,k})$ se obtiene en la ecuación (4.9) utilizando la ecuación (4.8).

$$\begin{aligned} E(\epsilon_{x,k}) &= E(x_k - \hat{x}_k) = E(x_k - \hat{x}_{k-1} - K_k[z_k - H\hat{x}_{k-1}]) \\ &= E(\epsilon_{x,k-1} - K_k[Hx_k + n_k - H\hat{x}_{k-1}]) \\ &= (I - K_k H)E(\epsilon_{x,k-1}) - K_k E(n_k) \end{aligned} \tag{4.9}$$

En concreto, el filtro de Kalman obtiene una K_k que minimiza la suma de las varianzas de los errores de estimación $\epsilon_{x,k}$ para cada estado $(x_1, \ldots, x_n)$ en el instante k. De esta forma, se establece el índice de coste en la ecuación

(4.10) en el que se sustituye la definición de $P_k = E\left(\epsilon_{x,k}\epsilon_{x,k}^T\right)$.

$$\begin{aligned} J_k &= E\left(\left[x_{1,k} - \hat{x}_{1,k}\right]^2\right) + \ldots + E\left(\left[x_{n,k} - \hat{x}_{n,k}\right]^2\right) \\ &= E\left(\epsilon_{x_1,k}^2 + \ldots + \epsilon_{x_n,k}^2\right) = E\left(\epsilon_{x,k}^T\epsilon_{x,k}\right) \\ &= E\left(\mathrm{Tr}\left[\epsilon_{x,k}\epsilon_{x,k}^T\right]\right) = \mathrm{Tr}\, P_k \end{aligned} \tag{4.10}$$

Para proceder con la optimización se desarrolla primeramente la covarianza del error de estimación P_k según su definición y sustituyendo $(\epsilon_{x,k})$ de (4.9) tal como se muestra en la ecuación (4.11).

$$\begin{aligned} P_k &= E\left(\epsilon_{x,k}\epsilon_{x,k}^T\right) \\ &= E\left\{\left[\left(I - K_kH\right)\epsilon_{x,k-1} - K_kn_k\right]\left[\left(I - K_kH\right)\epsilon_{x,k-1} - K_kn_k\right]^T\right\} \\ &= \left(I - K_kH\right)E\left(\epsilon_{x,k-1}\epsilon_{x,k-1}^T\right)\left(I - K_kH\right)^T \\ &\quad - K_kE\left(n_k\epsilon_{x,k-1}^T\right)\left(I - K_kH\right) - \left(I - K_kH\right)E\left(\epsilon_{x,k-1}n_k^T\right)K_k^T \\ &\quad + K_kE\left(n_kn_k^T\right)K_k^T \end{aligned} \tag{4.11}$$

Debido a que el ruido en el instante k no puede afectar la estimación en el instante anterior $k-1$, $\epsilon_{x,k-1}$ y n_k son independientes (no hay correlación entre ambos términos). De esta forma $E\left(n_k\epsilon_{x,k-1}^T\right) = E\left(n_k\right)E\left(\epsilon_{x,k-1}\right)$ y $E\left(\epsilon_{x,k-1}n_k^T\right) = E\left(\epsilon_{x,k-1}\right)E\left(n_k\right)$. Además, ambos términos son iguales a cero debido a que el valor esperado de n_k es $E\left(n_k\right) = 0$ (ruido blanco normal con media cero). De esta forma, la ecuación (4.11) se reduce a (4.12) en donde además se sustituye la definición [151] de la matriz de covarianza $R_k = E\left(n_kn_k^T\right)$.

$$P_k = E\left(\epsilon_{x,k}\epsilon_{x,k}^T\right) = \left(I - K_kH\right)P_{k-1}\left(I - K_kH\right)^T + K_kR_kK_k^T \tag{4.12}$$

Se obtiene la ganancia del filtro de Kalman al sustituir (4.12) en (4.10). Con esto se deriva el índice J_k y se iguala a cero para despejar la K_k que minimiza J_k. Al ser P_k una matriz de covarianzas tiene como propiedad ser simétrica y definida positiva [151], por lo que se puede aplicar la regla de la cadena junto

con la propiedad $\frac{\partial}{\partial X}\,\mathrm{Tr}\left[XYX^T\right] = 2XY$ (si Y es simétrica) para obtener el índice $\dot{J}_k$ mostrado en la ecuación (4.13).

$$\begin{aligned}
\dot{J}_k = \frac{\partial J_k}{\partial K_k} &= 2\left(I - K_k H\right) P_{k-1}\left(-H^T\right) + 2K_k R_k = 0 \\
&= -2P_{k-1}H^T + 2K_k H P_{k-1} H^T + 2K_k R_k = 0 \\
&= -2P_{k-1}H^T + 2K_k\left(H P_{k-1} H^T + R_k\right) = 0 \\
&\Rightarrow K_k = P_{k-1}H^T\left(H P_{k-1} H^T + R_k\right)^{-1} \\
&\therefore K_k = P_k^- H^T\left(H P_k^- H^T + R_k\right)^{-1}
\end{aligned} \tag{4.13}$$

Al obtener la segunda derivada $\ddot{J}_k$ respecto a K_k se comprueba que $\ddot{J}_k = M_k = \left(H P_{k-1} H^T + R_k\right) > 0$ al ser M_k definida positiva, por lo que la K_k obtenida hace que J_k sea mínimo. Es posible simplificar la ecuación (4.12) con el valor de la ganancia obtenido, obteniéndose una versión más compacta [151] para calcular P_k, tal y como se muestra en la ecuación (4.14).

$$\begin{aligned}
P_k &= \left(I - K_k H\right) P_{k-1}\left(I - K_k H\right)^T + K_k R_k K_k^T \\
&= \left(I - K_k H\right) P_{k-1} \\
\therefore P_k &= \left(I - K_k H\right) P_k^-
\end{aligned} \tag{4.14}$$

En resumen, el filtro de Kalman utiliza las ecuaciones (4.7),(4.8), (4.12) (o (4.14)) y (4.13) para realizar la estimación del estado tomando en cuenta la influencia del ruido en el sistema (w_k, n_k) de forma que se minimiza la covarianza del error de estimación P_k. El procedimiento es el mismo en caso de tener un sistema lineal variante en el tiempo sustituyendo en estas ecuaciones las correspondientes matrices $(A, B, H)_k$.

Por otra parte, en ciertos sistemas en donde se puede considerar que las matrices (A_k, H_k, Q_k, R_k) son invariantes en el tiempo, el cálculo de K_k puede realizarse fuera de línea con el fin de obtener la ganancia en régimen permanente y ahorrar recursos computacionales al permitir la implementación del filtro en sistemas con recursos muy limitados, ya que no se requeriría el cálculo de matrices inversas (ecuación (4.13)) dentro de la unidad de control. Esto, sin embargo, puede no ser conveniente en el caso de fusión sensorial

en donde puede requerirse ajustar dinámicamente, durante el tiempo de ejecución del algoritmo, los valores de R_k según el tipo de sensor o la cantidad disponible de los mismos; con lo que al depender K_k de R_k no puede realizarse su cálculo fuera de línea, requiriendo obtener la inversa dentro de la unidad de control.

El filtro de Kalman es la solución óptima en caso de que (w_k, n_k) se puedan caracterizar mediante una distribución de probabilidad independiente, blanca y normal (Gaussiana) con media cero [151]. Si (w_k, n_k) cumplen las condiciones anteriores pero no son ruidos Gaussianos, el filtro de Kalman es la mejor solución lineal posible en este problema (el KF es el filtro óptimo lineal para realizar la combinación de mediciones) aunque podría darse el caso de poder obtener un filtro no lineal con mejores resultados. Además, en el caso donde alguno de los ruidos (w_k, n_k) sea coloreado o no independientes (correlacionados entre sí), el filtro puede modificarse para tomar en cuenta estas condiciones durante el diseño. Finalmente, existen aspectos a tomar en cuenta al implementar este filtro, tales como la correcta inicialización de (x_k, P_x) en el instante inicial, además de la precisión asociada a las variables y al modelo. Debido a esto, las matrices (Q_R, R_R) pueden ajustarse posteriormente, después de realizar el diseño inicial, para mejorar el desempeño del filtro (de forma que se puedan considerar errores de modelado o inicialización) además de asignar la suficiente precisión a cada variable dentro de la unidad de control.

Cabe destacar que, a pesar de que el KF es óptimo en cuanto a minimizar error de estimación, no siempre es una solución conveniente para todos los sistemas. Debido a esto, existen distintas variantes del KF adaptados según cada situación (tipo de ruido, modelo parcial o desconocido, configuración del bucle de control, sistemas continuos, restricciones en el estado etc. [151]) y tipo de unidad de control (recursos disponibles, precisión, memoria, procesador, comunicaciones, etc.). Estas variantes se obtienen siguiendo el mismo procedimiento expuesto en el caso del KF por el cual se conocen como filtros de varianza mínima, debido a que minimizan la traza de la matriz de covarianzas P_k establecida mediante un índice de coste similar al de la ecuación (4.10).

4.3.1.2 Filtro H_∞

Existen también otros tipos de filtros para sistemas lineales que utilizan otros índices de coste en lugar del de mínima varianza (ecuación (4.10)) aunque en general son obtenidos mediante un procedimiento similar al del KF manteniendo la estructura de predicción/corrección (figura 4.3). Un ejemplo es el filtro conocido como H_∞, el cual busca minimizar el peor caso del error de estimación (optimización *minmax*) y cuya obtención se puede realizar mediante multiplicadores de Lagrange y teoría de juegos [151]. Los filtros H_∞ son diseñados para considerar tanto los posibles errores de modelado (variaciones en las matrices (A, B, H)) como los errores en la caracterización del ruido del proceso y la medida (errores en las matrices (Q, R)). De esta forma se consideran filtros robustos a estas variaciones que son tomadas en cuenta desde su diseño.

El filtro H_∞ en particular, no realiza suposiciones en las características de (w_k, n_k) por lo que es más general que el KF y aplicable en distintos tipos de ruido (incluso sin información de los mismos). Sin embargo, su implementación puede ser compleja computacionalmente en algunos casos de fusión sensorial ya que pueden requerirse múltiples inversiones de matrices en tiempo real, requiriendo más tiempo de procesador y memoria que el KF. Además el desempeño de este filtro es muy sensible al ajuste de sus parámetros, requiriendo mayor tiempo en el diseño de los mismos con el fin de garantizar un comportamiento adecuado. Finalmente, cabe mencionar que existen distintas combinaciones entre el KF/H_∞ que pueden establecerse con el fin de obtener los beneficios de ambas estrategias, al utilizar un índice de coste combinado entre minimizar P_k y minimizar el peor error de estimación o bien mediante una combinación lineal de las ganancias de ambos algoritmos, con el respectivo incremento en la complejidad computacional que conlleva ejecutar dos filtros de forma simultánea.

En el caso de sistemas no lineales, los filtros lineales descritos pueden adaptarse con el fin de solucionar el problema de estimación tal y como se describe a continuación.

4.3.2 Filtros de estimación para sistemas no lineales

A diferencia del caso lineal, la teoría asociada a los filtros en sistemas no lineales es compleja y aun en desarrollo, existiendo muy pocas soluciones exactas al problema de estimación que solo pueden aplicarse en un rango muy limitado de sistemas [8], por lo que es un área de constante desarrollo donde actualmente predominan soluciones aproximadas y subóptimas pero aplicables a gran cantidad de procesos [151, 124, 104, 110].

Para sistemas no lineales (ecuación (4.2)) no se tiene una solución exacta y general (aplicable a múltiples sistemas [8]) que describa la transformación de la densidad *pdf* a través del sistema no lineal o la evolución de la covarianza del error de estimación P_k, tal como se realiza en los sistemas lineales (ecuación (4.7)). De esta forma, para describir la evolución de P_k en el tiempo así como su modificación debido a la incorporación de la medición de los sensores, se recurre en la mayoría de los casos a aproximaciones, ya sea mediante linealizaciones en el modelo y adaptaciones de los filtros lineales, o bien mediante aproximaciones de la función de densidad de probabilidad *pdf* del sistema no lineal, de las cuales se puede obtener y actualizar P_k (ver descripción detallada en [151, 8]).

Se exponen a continuación los métodos más relevantes en filtros de estimación no lineal, los cuales consisten en las extensiones de filtros lineales (Kalman y H_∞ [151]) y en los métodos de transformación de la densidad *pdf* del estado no lineal, tanto asumiendo distribución gaussiana (filtros con transformadas Uncented [87, 90] y Cubature [8, 9]) como sin restricciones en la *pdf* (filtros de partículas [97]).

4.3.2.1 Filtro Extendido de Kalman (EKF):

Una de las primeras formas en las que se puede afrontar la estimación de un sistema no lineal es utilizar una aproximación de su modelo al linealizarlo mediante series de Taylor, con el fin de utilizar el mismo desarrollo realizado para el filtro de Kalman lineal con sus respectivas ecuaciones. Por ejemplo, para una función vectorial no lineal $f(\cdot)$ aplicada a la variable escalar x que tiene como punto de operación nominal $\bar{x}$ (punto de linealización), al definir

$\tilde{x} = (x - \bar{x})$ su expansión en series de Taylor viene dada [151] por la ecuación (4.15).

$$f(x) = f(\bar{x}) + \left.\frac{\partial f}{\partial x}\right|_{\bar{x}} \tilde{x} + \frac{1}{2!}\left.\frac{\partial^2 f}{\partial x^2}\right|_{\bar{x}} \tilde{x}^2 + \frac{1}{3!}\left.\frac{\partial^3 f}{\partial x^3}\right|_{\bar{x}} \tilde{x}^3 + \cdots \tag{4.15}$$

Si se aplica la función a un vector de dimensión n, su expansión se define como se muestra en la ecuación (4.16).

$$\begin{aligned} f(x) = f(\bar{x}) &+ \left(\tilde{x}_1 \frac{\partial}{\partial x_1} + \cdots + \tilde{x}_n \frac{\partial}{\partial x_n}\right) f\Big|_{\bar{x}} \\ &+ \frac{1}{2!}\left(\tilde{x}_1 \frac{\partial}{\partial x_1} + \cdots + \tilde{x}_n \frac{\partial}{\partial x_n}\right)^2 f\Big|_{\bar{x}} \\ &+ \frac{1}{3!}\left(\tilde{x}_1 \frac{\partial}{\partial x_1} + \cdots + \tilde{x}_n \frac{\partial}{\partial x_n}\right)^3 f\Big|_{\bar{x}} + \cdots \end{aligned} \tag{4.16}$$

Al definir el operador $D^k_{\tilde{x}} f$ se puede simplificar la ecuación (4.16) tal y como se muestra en la ecuación (4.17).

$$\begin{aligned} &\because D^k_{\tilde{x}} f = \left(\sum_{i=1}^{n} \tilde{x}_i \frac{\partial}{\partial x_i}\right)^k f(x)\Big|_{\bar{x}} \\ &\therefore f(x) = f(\bar{x}) + D^1_{\tilde{x}} f + \frac{1}{2!} D^2_{\tilde{x}} f + \frac{1}{3!} D^3_{\tilde{x}} f + \cdots \end{aligned} \tag{4.17}$$

En general la aproximación se reduce al considerar únicamente el primer término (primer momento o primer polinomio) $D^1_{\tilde{x}} f$, ya que en muchos casos las derivadas de orden superior tienen valores numéricos pequeños. Este caso se produce además cuando x se encuentra cerca de su punto de operación $\tilde{x}$. Considerando estas situaciones, se realiza la linealización de $f(x)$ utilizando únicamente el primer polinomio de (4.17) según se muestra en la ecuación (4.18).

$$f(x) \approx f(\bar{x}) + D^1_{\tilde{x}} f \tag{4.18}$$

Con esto se procede a realizar la linealización del modelo (4.2) alrededor del punto de operación $\hat{x}$ con $w_k = 0$ y $n_k = 0$, con lo que el modelo se reduce a (4.19).

$$\begin{aligned} x_k &= f\left(x_{k-1}, u_{k-1}, 0\right) \\ z_k &= h\left(x_k, 0\right) \end{aligned} \tag{4.19}$$

Al aplicar la relación (4.18) a (4.19) se obtiene la linealización del modelo en la ecuación (4.20) alrededor del punto de operación $\hat{x}$.

$$\begin{aligned} x_k &\approx f_{k-1}\left(\hat{x}_{k-1}, u_{k-1}, 0\right) + A_{k-1}\left(x_{k-1} - \hat{x}_{k-1}\right) + W_{k-1} w_{k-1} \\ z_k &\approx h\left(\hat{x}_k, 0\right) + H_k\left(x_k - \hat{x}_k\right) + N_k n_k \end{aligned} \tag{4.20}$$

En donde las matrices del sistema se obtienen en cada instante k utilizando la ecuación (4.21).

$$\begin{aligned} \mathbf{A_k} &= \left.\frac{\partial f}{\partial x}\right|_{\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0} \\ \mathbf{H_k} &= \left.\frac{\partial h}{\partial x}\right|_{\hat{\mathbf{x}}_k, 0} \\ \mathbf{W_k} &= \left.\frac{\partial f}{\partial w}\right|_{\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0} \\ \mathbf{N_k} &= \left.\frac{\partial h}{\partial n}\right|_{\hat{\mathbf{x}}_k, 0} \end{aligned} \tag{4.21}$$

De esta forma se pueden utilizar las mismas ecuaciones para la ganancia obtenidas en el caso lineal. El filtro extendido de Kalman se muestra en la figura 4.4.

Al igual que para el caso lineal, se pueden obtener versiones adicionales del EKF para el caso de sistemas continuos, o realizando modificaciones adicionales para tomar en cuenta distintos tipos de ruido. De la misma forma, se puede obtener una versión del H_∞ para sistemas no lineales [132] utilizando un procedimiento similar al EKF, por lo que se emplean las mismas ecuacio-

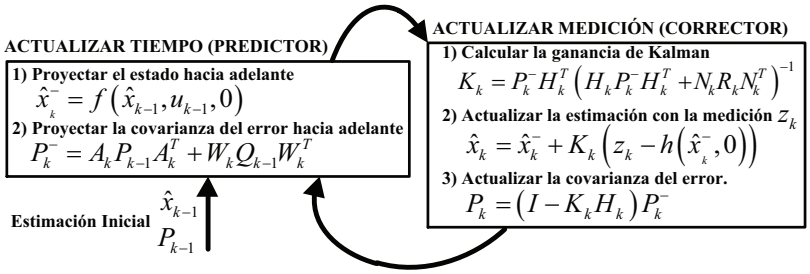


Figura 4.4: Algoritmo del filtro de Kalman Extendido [151, 165].

nes del H_∞ lineal junto con el modelo linealizado y las matrices de (4.21). Cabe destacar además que existen versiones tanto del EKF como del H_∞ que utilizan una linealización de segundo orden (hasta el segundo polinomio de la aproximación de Taylor (ecuación (4.17))), aportando un poco más de precisión a la estimación a cambio de un incremento en el coste computacional del filtro, al requerir el cálculo de múltiples matrices inversas. Si se requiere más precisión, se recurre a los métodos de aproximación de la *pdf* los cuales se describen a continuación.

4.3.2.2 Filtros con Aproximaciones de la Densidad pdf*:*

Debido a que el EKF utiliza una linealización de primer orden, la estimación mediante este método puede estar sujeta a errores considerables en sistemas con no linealidades considerables (o bruscas), que no son representadas de forma adecuada mediante el primer término de la aproximación por series de Taylor del sistema no lineal (ecuación (4.18)). De esta forma, para representar adecuadamente un sistema altamente no lineal, puede requerirse calcular varios términos adicionales de la aproximación de Taylor para obtener una precisión adecuada, lo que puede suponer un alto coste computacional. Debido a esto, surgen filtros de estimación que a diferencia del EKF toman en consideración una aproximación más precisa del sistema y su covarianza asociada (del error de estimación), sin requerir el uso de las aproximaciones del modelo mediante series de Taylor.

Con el fin de observar claramente la diferencia entre los planteamientos, se puede utilizar el promedio y la covarianza de una transformación (función o modelo) no lineal $y = h(x)$. Para esto se obtiene su aproximación en series de Taylor al utilizar la ecuación (4.17), tal y como se muestra en la ecuación (4.22), recordando que $\tilde{x} = (x - \bar{x})$.

$$\begin{aligned} y &= h(x) \\ &= h(\bar{x}) + D^1_{\tilde{x}}h + \frac{1}{2!}D^2_{\tilde{x}}h + \frac{1}{3!}D^3_{\tilde{x}}h + \cdots \end{aligned} \tag{4.22}$$

De esta forma, se obtiene la media al obtener el valor esperado de (4.22) tal y como se muestra en la ecuación (4.23).

$$\begin{aligned} \bar{y} &= h(\bar{x}) + E\left[D^1_{\tilde{x}}h + \frac{1}{2!}D^2_{\tilde{x}}h + \frac{1}{3!}D^3_{\tilde{x}}h + \cdots\right] \\ &= h(\bar{x}) + E\left(D^1_{\tilde{x}}h\right) + \frac{1}{2!}E\left(D^2_{\tilde{x}}h\right) + \frac{1}{3!}E\left(D^3_{\tilde{x}}h\right) + E[\cdots] \end{aligned} \tag{4.23}$$

Esta ecuación se reduce al considerar el hecho de, por definición [151], para una variable aleatoria con *pdf* simétrica y media cero, todos sus momentos impares tienen un valor esperado nulo, por lo que (4.23) se reduce a (4.24).

$$\bar{y} = h(\bar{x}) + \frac{1}{2!}E\left(D^2_{\tilde{x}}h\right) + \frac{1}{4!}E\left(D^4_{\tilde{x}}h\right) + E[\cdots] \tag{4.24}$$

Con $\bar{y}$ se puede obtener la respectiva covarianza P_y , para esto primeramente se obtiene $(y - \bar{y})$ tal y como se muestra en la ecuación (4.25).

$$\begin{aligned} y - \bar{y} &= \left[h(\bar{x}) + D^1_{\tilde{x}}h + \frac{1}{2!}D^2_{\tilde{x}}h + \frac{1}{3!}D^3_{\tilde{x}}h + \cdots\right] \\ &\quad - \left[h(\bar{x}) + \frac{1}{2!}E\left(D^2_{\tilde{x}}h\right) + \frac{1}{4!}E\left(D^4_{\tilde{x}}h\right) + E[\cdots]\right] \\ \Rightarrow y - \bar{y} &= \left[D^1_{\tilde{x}}h + \frac{1}{2!}D^2_{\tilde{x}}h + \frac{1}{3!}D^3_{\tilde{x}}h + \cdots\right] \\ &\quad - \left[\frac{1}{2!}E\left(D^2_{\tilde{x}}h\right) + \frac{1}{4!}E\left(D^4_{\tilde{x}}h\right) + \cdots\right] \end{aligned} \tag{4.25}$$

Utilizando la definición de covarianza (ecuación (B.5), apéndice B) se obtiene P_y según se muestra en la ecuación (4.26)

$$\begin{aligned} P_y &= E\left[(y-\bar{y})\,(y-\bar{y})^T\right] \\ &= E\left[D^1_{\tilde{x}}h\Big(D^1_{\tilde{x}}h\Big)^T\right] \\ &+ E\left[\frac{1}{3!}D^1_{\tilde{x}}h\Big(D^3_{\tilde{x}}h\Big)^T + \frac{1}{2!2!}D^2_{\tilde{x}}h\Big(D^2_{\tilde{x}}h\Big)^T + \frac{1}{3!}D^3_{\tilde{x}}h\Big(D^1_{\tilde{x}}h\Big)^T\right] \\ &+ E\left[\frac{1}{2!}D^2_{\tilde{x}}h\right]E\left[\frac{1}{2!}D^2_{\tilde{x}}h\right]^T + \cdots \end{aligned} \tag{4.26}$$

El primer término $E\left[D^1_{\tilde{x}}h(D^1_{\tilde{x}}h)^T\right]$ puede reescribirse al sustituir la definición del operador en la ecuación (4.17) tal y como se muestra en la ecuación (4.27).

$$\begin{aligned} E\left[D^1_{\tilde{x}}h\Big(D^1_{\tilde{x}}h\Big)^T\right] &= E\left[\left(\sum_{i=1}^{n}\tilde{x}_i\,\frac{\partial h}{\partial x_i}\Big|_{x=\bar{x}}\right)\left(\sum_{j=1}^{n}\tilde{x}_j\,\frac{\partial h}{\partial x_j}\Big|_{x=\bar{x}}\right)^T\right] \\ &= E\left[\sum_{i=1}^{n}\tilde{x}_i\,\frac{\partial h}{\partial x_i}\Big|_{x=\bar{x}}\,\frac{\partial h^T}{\partial x_j}\Big|_{x=\bar{x}}\,\tilde{x}_j\right] \\ &= \sum_{i,j}H_iE\left[\tilde{x}_i\tilde{x}_j\right]H_j^T = \sum_{i,j}H_iP_{ij}H_j^T = HPH^T \end{aligned} \tag{4.27}$$

De esta forma, al sustituir la ecuación (4.27) en (4.26) se obtiene la expresión para P_y, según se describe en la ecuación (4.28).

$$\begin{aligned} P_y = HPH^T &+ E\left[\frac{1}{3!}D^1_{\tilde{x}}h\Big(D^3_{\tilde{x}}h\Big)^T + \frac{1}{2!2!}D^2_{\tilde{x}}h\Big(D^2_{\tilde{x}}h\Big)^T + \frac{1}{3!}D^3_{\tilde{x}}h\Big(D^1_{\tilde{x}}h\Big)^T\right] \\ &+ E\left[\frac{1}{2!}D^2_{\tilde{x}}h\right]E\left[\frac{1}{2!}D^2_{\tilde{x}}h\right]^T + \cdots \end{aligned} \tag{4.28}$$

Al comparar la evolución de la covarianza P_y con la definida en el EKF (ecuación (4.7) junto con la aproximación de (4.21)) en donde para el estado se definía $P_k = A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1}$ (utilizando W_k igual a la matriz identidad en la ecuación (4.21)), se observa que el EKF únicamente utiliza el primer término de la aproximación de P_y (obtenida mediante series de Taylor), dejando una gran cantidad de términos sin considerar a la hora de realizar el cálculo. Nuevamente, el uso de la linealización de primer orden tanto del estado como de la covarianza puede no ser adecuada en sistemas altamente no lineales donde deberían utilizarse varios términos adicionales de la aproximación. Esto sin embargo tiene el gran inconveniente de producir un incremento considerable en el coste computacional del filtro (por ejemplo en el EKF de segundo orden), al requerir calcular múltiples matrices inversas según en número de polinomios utilizados en la aproximación de Taylor.

Por esta razón surgen nuevos métodos que buscan formas alternativas de obtener una precisión adecuada (sobre el tercer polinomio de la aproximación de Taylor) pero que no requieran utilizar el procedimiento de linealización aplicado al modelo (procedimiento seguido para obtener la ecuación (4.28)). Bajo la premisa de que es más sencillo aproximar la densidad *pdf* de una función no lineal $y = h(x)$ que realizar su aproximación mediante series de Taylor con precisión adecuada, surge el método denominado como transformada *Unscented* (UT) [88, 89, 87, 90, 151, 162]. Para realizar la aproximación de la función de densidad *pdf* se utilizan un conjunto de puntos, ya que es menos costoso realizar la transformación no lineal y sobre un punto x (un estado del sistema por ejemplo) que sobre toda la función *pdf*.

De esta forma, se debe realizar una selección de puntos, denominados puntos *sigma* $\chi_{(i)}$, a los cuales se les aplica la función no lineal $y = h(\chi_{(i)})$ (modelo del sistema) para, a partir de su transformación, aproximar la media y la covarianza de la densidad *pdf*. Ante este planteamiento surge la cuestión de cómo seleccionar los puntos *sigma*. En el caso de la transformada *Unscented*, se eligen de forma que la media y covarianza compuesta (obtenida utilizando los puntos $\chi_{(i)}$) sean iguales a la $\bar{x}$ y P iniciales del vector de estados x (vector columna, a priori) de dimensión n (orden del sistema, cantidad de estados independientes; no confundir con el ruido de medición n_k). De esta forma,

la UT define $2n$ puntos sigma a partir de x con media $\bar{x}$ y covarianza P conocidas, según la ecuación (4.29) [87, 151].

$$\begin{aligned} \chi_{(i)} &= \bar{x} + \tilde{\mathrm{P}}_{(i)} \qquad && i = 1, \cdots n, n+1, \cdots, 2n \\ \tilde{\mathrm{P}}_{(i)} &= \left(\sqrt{nP}\right)_{(i)} && i = 1, \cdots, n \\ \tilde{\mathrm{P}}_{(n+i)} &= -\left(\sqrt{nP}\right)_{(i)} && i = 1, \cdots, n \end{aligned} \tag{4.29}$$

En donde $\sqrt{nP}_{(i)}$ es la *i-ésima* columna de la raíz cuadrada matricial de nP, definida de tal forma que $\left(\sqrt{nP}\right)^T \sqrt{nP} = nP$. De esta forma, al aplicar la función no lineal y a los puntos sigma, se utiliza el resultado (los $\chi_{(i)}$ transformados) para realizar una combinación lineal ponderada, que permite obtener una buena estimación de la media $\bar{y}_u$ y covarianza $P_{y,u}$ real de la función no lineal y tal y como se define en la ecuación (4.30).

$$\begin{aligned} y_{(i)} &= h\left(\chi_{(i)}\right), \ i = 1, \cdots, 2n \\ \bar{y}_u &= \sum_{i=1}^{2n} W_{(i)} y_{(i)} = \frac{1}{2n} \sum_{i=1}^{2n} y_{(i)} \\ P_{y,u} &= \sum_{i=1}^{2n} W_{(i)} \left(y_{(i)} - \bar{y}_u\right)\left(y_{(i)} - \bar{y}_u\right)^T \\ &= \frac{1}{2n} \sum_{i=1}^{2n} \left(y_{(i)} - \bar{y}_u\right)\left(y_{(i)} - \bar{y}_u\right)^T \end{aligned} \tag{4.30}$$

En esta ecuación se ha definido el factor de ponderación $W_{(i)}$ como una constante $W_{(i)} = 1/(2n)$ para $i = 1, \cdots, 2n$, por lo que los puntos sigma se han ponderado igualitariamente para obtener $\bar{y}_u$ y $P_{y,u}$.

Al sustituir la aproximación en series de Taylor de la ecuación (4.23) en (4.30) se puede seguir un procedimiento similar al realizado para obtener las ecuaciones de la media y covarianza con la aproximación en series de Taylor (ecuaciones (4.24) y (4.28)). Este desarrollo demuestra que al utilizar la UT (ecuaciones (4.29) y (4.30)) se logra aproximar $\bar{y}_u$ y $P_{y,u}$ con una precisión de hasta el tercer polinomio de la expansión en series de Taylor (ver desarrollo

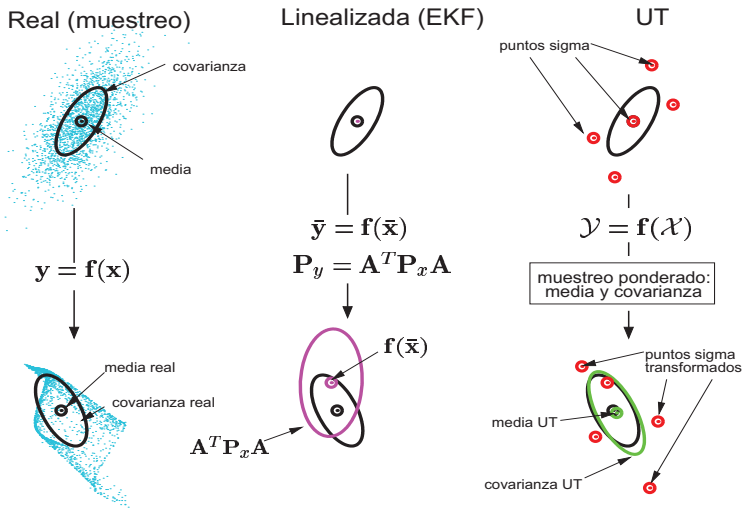


Figura 4.5: Propagación de la media y la covarianza en la Transformada Unscented [162].

en [151]). Con esto, al comparar con la aproximación que realiza el EKF, que solo tiene precisión de hasta el primer término de la serie de Taylor, se observa un incremento considerable en la precisión al utilizar la UT, sin la desventaja de requerir calcular múltiples matrices inversas ni tener que realizar la linealización mediante series de Taylor. Sin embargo, la UT tiene la desventaja de requerir el cálculo de la raíz cuadrada matricial de nP, la matriz de covarianza del error de estimación. Esta $\sqrt{nP}$ puede requerir recursos computacionales considerables dependiendo del orden del sistema, por lo que puede no ser implementable en determinadas unidades de control.

Una comparación entre los métodos de linealización de primer orden (utilizado por el EKF) y la transformada *Unscented* se muestra en las figuras 4.5 y 4.6. La figura 4.5 resume la propagación de la media y la covarianza para el método de linealización y para la UT. En la figura 4.6 se muestra la obtención de la media $\bar{y}$ y covarianza P_y de una transformación de coordenadas no lineal (polar a cartesiana, $y_1 = rcos\theta$,$y_2 1 = rsin\theta$) obtenida a partir de

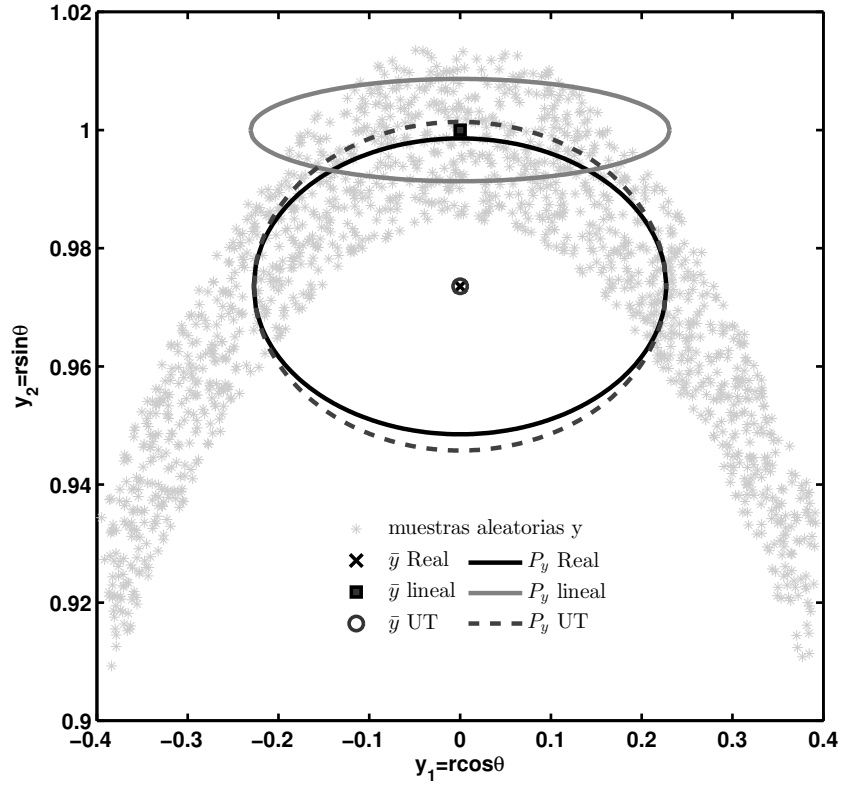


Figura 4.6: Comparación entre el método de linealización y la Transformada Unscented para la obtención de $\bar{y}$ y P_y de una transformación de coordenadas [90, 151].

una prueba con 1500 puntos distribuidos uniformemente con $\sigma_r = 0{,}015$ y $\sigma_\theta = 0{,}4$. De esta figura se observa claramente como tanto la aproximación de primer orden como la UT para $\bar{y}_1$ son iguales a la ($\bar{y}_1$) real (obtenida a partir de las definiciones aplicadas al modelo [151])). Sin embargo, esto no ocurre para $\bar{y}_2$ en donde la aproximación lineal tiene una diferencia porcentual de 2,72 % muy superior al de la aproximación realizada con la UT con un 0,0024 %. En cuanto a P_y ambas aproximaciones tienen una amplitud similar en y_1 pero no en y_2 donde la aproximación lineal no se aproxima a la real, caso contrario a la obtenida con la UT.

Aunque éste no es un ejemplo de estimación mediante filtros, muestra muy claramente el error esperado al utilizar una u otra técnica de aproximación (linealización o UT) en sistemas altamente no lineales. Para el caso de filtros, depende de otros factores, como el tiempo de muestreo aparte del tipo de no linealidad del sistema. Debido a esto, tanto filtros basados en linealización como en la UT podrían tener comportamientos similares en determinadas aplicaciones, por lo que se podría determinar la utilización de una u otra estrategia según la conveniencia de calcular la raíz matricial para la UT que es computacionalmente costosa.

A pesar de esto, la UT resulta útil en casos altamente no lineales y en donde la capacidad de cómputo no es limitada, con lo que se ha utilizado para definir un filtro de estimación no lineal con la misma estructura de predicción/corrección del KF, conocido como el filtro de Kalman Unscented (UKF [90, 151]). De esta forma, para el modelo no lineal de la ecuación (4.2), con $\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1})$ y $\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{n}_k)$, al considerar que (w_k, n_k) se caracterizan mediante una distribución de probabilidad independiente, blanca y normal (Gaussiana) con media cero, con matrices de covarianza Q_k y R_k, se define el UKF a partir de la UT al considerar, en lugar de la media $\bar{x}$, el estado estimado $\hat{x}$, tal y como se describe a continuación.

Partiendo del estado y covarianza en el instante inicial $k = 0$, $\hat{x}_{k-1} = \hat{x}_0 = E[x_o]$ y $P_{k-1} = E[(x_o - \hat{x}_0)(x_o - \hat{x}_0)^T]$, y considerando los factores de ponderación $W_{(i)} = 1/(2n)$ para $i = 1, \cdots, 2n$, se ejecuta el UKF en dos etapas: actualización temporal y corrección con la medición. La actualización temporal corresponde a siete pasos. Los primeros tres corresponden a la actualización temporal del estado (al ser $f(\cdot)$ no lineal). Estos consisten en: generar los puntos sigma $\hat{x}^{(i)}_{k-1}$ en la ecuación (4.31), propagar los puntos sigma a través del modelo no lineal dada la entrada u_{k-1} según describe la ecuación (4.32) y finalmente obtener el estado y su covarianza actualizadas en el tiempo de acuerdo con la ecuación (4.33).

$$\begin{aligned} \hat{x}^{(i)}_{k-1} &= \hat{x}_{k-1} + \widetilde{\mathrm{P}}_{(i)} \qquad i = 1, \cdots n, n+1, \cdots, 2n \\ \widetilde{\mathrm{P}}_{(i)} &= \left(\sqrt{nP_{k-1}}\right)_{(i)} \qquad i = 1, \cdots, n \\ \widetilde{\mathrm{P}}_{(n+i)} &= -\left(\sqrt{nP_{k-1}}\right)_{(i)} \qquad i = 1, \cdots, n \end{aligned} \tag{4.31}$$

$$\hat{x}_k^{(i)} = f\left(\hat{x}_{k-1}^{(i)}, u_{k-1}\right), \qquad i = 1, \cdots, 2n \tag{4.32}$$

$$\begin{aligned} \hat{x}_k^- &= \sum_{i=1}^{2n} W_{(i)} \hat{x}_k^{(i)} \\ P_k^- &= \sum_{i=1}^{2n} W_{(i)} \left(\hat{x}_k^{(i)} - \hat{x}_k\right)\left(\hat{x}_k^{(i)} - \hat{x}_k\right)^T + Q_{k-1} \end{aligned} \tag{4.33}$$

Los segundos tres pasos son los correspondientes a la actualización temporal de la medición (al ser $h(\cdot)$ no lineal), estos consisten en: generar los puntos sigma $\hat{x}_k^{(i)}$ en la ecuación (4.34), propagar los puntos sigma a través del modelo no lineal de la medición según la ecuación (4.35) y finalmente obtener la estimación de la medición y su covarianza actualizadas en el tiempo de acuerdo con la ecuación (4.36)

$$\begin{aligned} \hat{x}_k^{(i)} &= \hat{x}_k + \widetilde{\mathrm{P}}_{(i)} && i = 1, \cdots n, n+1, \cdots, 2n \\ \widetilde{\mathrm{P}}_{(i)} &= \left(\sqrt{nP_k}\right)_{(i)} && i = 1, \cdots, n \\ \widetilde{\mathrm{P}}_{(n+i)} &= -\left(\sqrt{nP_k}\right)_{(i)} && i = 1, \cdots, n \end{aligned} \tag{4.34}$$

$$\hat{z}_k^{(i)} = h\left(\hat{x}_k^{(i)}\right), \qquad i = 1, \cdots, 2n \tag{4.35}$$

$$\begin{aligned} \hat{z}_k &= \sum_{i=1}^{2n} W_{(i)} \hat{z}_k^{(i)} \\ P_z &= \sum_{i=1}^{2n} W_{(i)} \left(\hat{z}_k^{(i)} - \hat{z}_k\right)\left(\hat{z}_k^{(i)} - \hat{z}_k\right)^T + R_{k-1} \end{aligned} \tag{4.36}$$

El último paso de la actualización temporal corresponde al cálculo de la estimación de la covarianza cruzada entre $\hat{x}_k$ y $\hat{z}_k$ según la ecuación (4.37) la cual se necesita para calcular la ganancia del filtro *UKF*.

$$P_{xz} = \sum_{i=1}^{2n} W_{(i)} \left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)\left(\hat{z}_k^{(i)} - \hat{z}_k\right)^T \tag{4.37}$$

La corrección con la medición consta de dos pasos: obtener la ganancia del filtro K_k en la ecuación (4.38) y finalmente obtener el estado estimado $\hat{x}_k$ y la covarianza P_k actualizadas utilizando la medición z_k tal como se muestra en la ecuación (4.39).

$$K_k = P_{xz}(P_z)^{-1} \tag{4.38}$$

$$\begin{aligned} \hat{x}_k &= \hat{x}_k^- + K_k\left(z_k - \hat{z}_k\right) \\ P_k &= P_k^- - K_k P_z K_k^T \end{aligned} \tag{4.39}$$

De esta forma, al ejecutar el filtro UKF con las ecuaciones de la (4.31) a la (4.39), se logra obtener el estado del sistema $\hat{x}_k$ y su covarianza P_x. Esta versión del UKF requiere el cálculo de dos raíces cuadradas matriciales en (4.31) y (4.34) junto con una matriz inversa en (4.39), todas dependientes del orden del sistema n (cantidad de estados a estimar) lo que supone un coste computacional muy elevado al comparar con los métodos de linealización.

En algunos casos, el coste puede reducirse al omitir la ecuación (4.34) y aprovechar los mismos puntos sigma generados con la ecuación (4.31) lo que simplifica la ejecución del filtro pero disminuye su precisión. El desarrollo presentado asume que el ruido entra al proceso y a la medición de forma lineal. Si este no es el caso o bien, si se desea estimar la forma en la que el ruido interactúa con el sistema, se realiza la estimación del estado ampliado mostrado en la ecuación (4.40) y se inicializa el filtro según la ecuación (4.41). De esta forma se utilizan las mismas ecuaciones propuestas pero eliminando Q_{k-1} de (4.33) y R_k de (4.36) ya que son estimadas dentro del filtro. Este UKF es más general pero incrementa el coste computacional ya que aumenta la dimensión del filtro y por consiguiente la dificultad en calcular las raíces matriciales y la inversa.

$$x_k^{(a)} = \begin{bmatrix} x & w & n \end{bmatrix}_k^T \tag{4.40}$$

$$\begin{aligned} \hat{x}_0^{(a)} &= \begin{bmatrix} E(x_0) & 0 & 0 \end{bmatrix}_k^T \\ P_0^{(}a) &= \begin{bmatrix} (x_o - \hat{x}_0)(x_o - \hat{x}_0)^T & 0 & 0 \\ 0 & Q_0 & 0 \\ 0 & 0 & R_0 \end{bmatrix} \end{aligned} \tag{4.41}$$

Cabe destacar que existen distintas variantes de la transformación Unscented que permiten adecuar el filtro a distintas características, principalmente para mejorar la estabilidad numérica en algunos sistemas. Entre éstas sobresale una modificación al asignar los puntos sigma, en la cual se propone asignar un punto adicional en $\hat{x}_{k-1}^{(i)} = \hat{x}_{k-1}$ según la ecuación (4.42), siendo κ una constante definida con cualquier valor real (que no anule el denominador) con un valor recomendado en $\kappa = 3-n$ para ruidos gaussianos. Esta modificación logra minimizar el error hasta el cuarto polinomio de Taylor en algunos sistemas. Utilizando la ecuación (4.42) junto con los factores de ponderación definidos en la ecuación (4.43) se utilizan las mismas ecuaciones definidas para el UKF tradicional pero realizando las sumatorias con el índice $i = 0$ para abarcar los $2n + 1$ puntos requeridos en esta versión.

$$\begin{aligned} \hat{x}_{k-1}^{(0)} &= \hat{x}_{k-1} && i = 1, \cdots n, n+1, \cdots, 2n \\ \hat{x}_{k-1}^{(i)} &= \hat{x}_{k-1} + \widetilde{\mathrm{P}}_{(i)} \\ \widetilde{\mathrm{P}}_{(i)} &= \left(\sqrt{(n+\kappa)\, P_{k-1}}\right)_{(i)} && i = 1, \cdots, n \\ \widetilde{\mathrm{P}}_{(n+i)} &= -\left(\sqrt{(n+\kappa)\, P_{k-1}}\right)_{(i)} && i = 1, \cdots, n \end{aligned} \tag{4.42}$$

$$\begin{aligned} W_{(0)} &= \kappa/(n+\kappa) \\ W_{(i)} &= 1/2\,(n+\kappa) \quad i = 1, \cdots, 2n \end{aligned} \tag{4.43}$$

Por último, es conveniente mencionar un desarrollo reciente que busca aprovechar nuevas técnicas de integración numérica conocido como el filtro de Kalman *Cubature* (CKF) [8, 9, 132], con aplicación en problemas fusión sensorial y en SLAM [131]. Este filtro tiene un procedimiento muy similar al UKF pero es deducido al utilizar las definiciones del valor esperado y covarianza (ecuaciones (B.1b) y (B.4), apéndice B) expresados como la integral

dependiente de la densidad *pdf*, para definir la covarianza y el valor esperado de la estimación (ecuación (4.33) del UKF) o su medición (ecuación (4.36) del UKF). Estas integrales son aproximadas mediante la utilización de las reglas "Cubature" con lo que cada ecuación se aproxima mediante una suma ponderada (de forma muy similar a las del UKF).

La principal diferencia respecto al UKF radica en la asignación de los puntos "Cubature" utilizados para realizar la aproximación (equivalentes a los sigma en el UKF) y en la consideración del tipo de *pdf* utilizada, que para el caso del CKF solo se requiere que sea una densidad simétrica, no es necesario que sea Gaussiana (Normal). Los métodos "Cubature" utilizados para obtener el CKF garantizan una mayor precisión y estabilidad numérica que el UKF y el EKF, además de ser aplicables a casos más generales que distribuciones normales. A pesar de sus ventajas, tanto el CKF como el UKF requieren el cálculo de la raíz cuadrada matricial $\sqrt{P}$ que es costosa computacionalmente.

4.3.2.3 Filtros de partículas PF*:*

Los filtros de partículas (métodos de filtrado/filtrado secuencial Monte Carlo) son métodos de estimación exhaustiva (por fuerza bruta) que son adecuados para problemas donde los filtros de Kalman no proveen una solución adecuada (sistemas altamente no lineales) [51, 157, 97, 151]. A diferencia del EKF, UKF y CKF, que son filtros que aproximan la solución del problema de estimación no lineal al considerar únicamente un tipo de distribución *pdf* (Gaussiana, simétrica, etc.), los filtros de partículas son más generales, ya que buscan aproximar numéricamente toda la *pdf* sin asunciones previas y sin estimar solamente su media o covarianza.

De esta forma, se utilizan una gran cantidad de puntos (partículas, estados posibles) que son propagados a través del sistema no lineal y cuyo resultado es utilizado para aproximar la *pdf* total; a partir de la cual se obtiene el valor estimado del estado y su respectiva covarianza. Debido a que cada partícula tiene una probabilidad asociada que es modificada según la medición realizada, ésta se utiliza para seleccionar qué partículas son las más cercanas al valor real del estado. Al eliminar las partículas con baja probabilidad

de ocurrencia se realiza un *remuestreo*, duplicando las partículas con mayor probabilidad hasta alcanzar el número de partículas inicial y poder proceder con la siguiente ejecución del método en el instante $k+1$. De esta forma, al cabo de cierto tiempo se tendrán un gran número de partículas cercanas al valor real de la variable, tal como se muestra en el ejemplo de la figura 4.7. Teóricamente el filtro de partículas tenderá a la solución exacta si el número de partículas tiende al infinito.

Los Filtros de Partículas producen en general estimaciones muy precisas, pueden resolver problemas más complejos y relevantes en localización tales como el robot secuestrado o localización con punto inicial desconocido (localización global con incertidumbre inicial) [150, 6, 115] además de problemas de localización con mapa desconocido y SLAM (mapeo y localización simultánea) [50, 157, 53, 11]. Sin embargo, este tipo de soluciones requieren de gran capacidad computacional para operar (memoria y tiempo de cálculo) y estimar la probabilidad, a priori, de un conjunto de estados iniciales (partículas).

En el ejemplo de la figura 4.7 se muestra el caso de localización de un robot móvil en un entorno con mapa conocido. En este caso, cada partícula del filtro equivaldría a una posible postura (x, y, θ) del robot en el espacio. El método inicia al generar un conjunto de estas soluciones y las reparte en el mapa (figura 4.7(a)). Conforme el robot avanza dentro del entorno y utiliza la información de los sensores, el filtro de partículas va eliminando partículas erróneas (con baja probabilidad de ocurrencia) y va agregando nuevas partículas, cercanas a las de mejor probabilidad (figura 4.7(b)). Con esto, al cabo de cierto tiempo, se habrán eliminado la mayoría de partículas erróneas mientras que las correctas se habrán agrupado cerca de la postura real del robot (figura 4.7(c)).

Descritos los métodos actuales de estimación y fusión sensorial se procede a justificar la elección de los algoritmos a utilizar en la presente tesis.

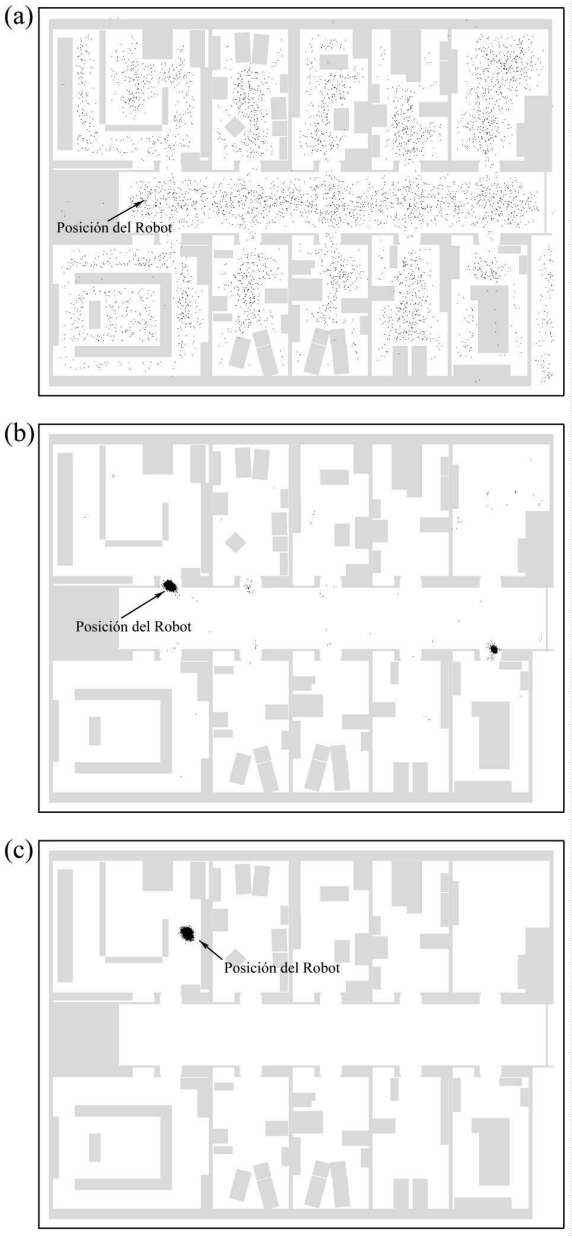


Figura 4.7: Funcionamiento del Filtro de Partículas en aplicaciones de localización [157].

4.4 Selección de algoritmos para la fusión sensorial

Después del resumen realizado de los métodos existes en estimación de estados y fusión sensorial, se debe seleccionar una técnica que permita el desarrollo de los distintos algoritmos de fusión utilizando los principios del control y muestreo basado en eventos aplicados a robots de recursos limitados. En el presente capítulo se describieron dos grandes grupos de técnicas de estimación, los *observadores* y los *filtros* de estimación, y para ambos grupos se distinguen los métodos lineales y no lineales.

En cuanto a los observadores se puede apreciar que en algunos casos son de utilidad, ya que pueden requerir menos recursos (memoria y tiempo de ejecución) que las soluciones basadas en filtros [81]. Sin embargo, son métodos que no toman en cuenta ruidos en el proceso y medición durante el diseño y síntesis del observador, lo que implica realizar un estudio adicional para comprobar que el método es robusto ante este tipo de perturbaciones. Además requieren modelos muy precisos del sistema, difíciles de obtener y parametrizar. En el caso no lineal, se requiere el uso de las técnicas de Lyapunov para obtener la ganancia del observador, lo cual conlleva un tiempo considerable en el diseño. Para el caso específico de fusión sensorial no son métodos flexibles en cuanto a implementación y adaptabilidad ante alguna variación en el esquema, como es el caso de tener que agregar o retirar un sensor, dar más importancia a un sensor respecto a otro en la fusión de acuerdo a su precisión o variar su aportación según el desempeño del sistema.

Por otra parte, los filtros de estimación son métodos basados en modelos estocásticos que permiten tomar en cuenta, desde la etapa de diseño, la incorporación de la información disponible sobre el ruido del proceso y de la medición para la obtención de la estimación del estado. Tienen la ventaja de permitir la incorporación de índices de desempeño a través de la ganancia del método, de forma que se garantice, por ejemplo, un error de estimación mínimo o el error mínimo en el peor caso. Además, existen múltiples alternativas tanto para el caso de sistemas lineales como no lineales que son adecuadas para realizar la fusión sensorial en múltiples situaciones, ya que permiten incorporar o retirar sensores fácilmente durante la ejecución del algoritmo, además de posibilitar la asignación de una importancia relativa a cada sensor

de forma independiente según su precisión (a través de la matriz R_k). Entre los métodos no lineales, las opciones estudiadas constan de distintos niveles de precisión, siguiendo como norma el incremento de la misma conforme se incrementan los recursos computacionales disponibles para ejecutar el filtro, de la misma forma en que su generalidad aumenta desde métodos poco generales y que consumen pocos recursos hasta métodos muy generales y costosos computacionalmente.

Tomando en cuenta que el objetivo a conseguir en la presente tesis es la integración eficiente de múltiples sensores para mejorar la localización de un robot móvil de recursos limitados, resulta muy conveniente la elección de las técnicas de fusión sensorial basadas en filtros de estimación. Éstas, a diferencia de los observadores, toman en cuenta la precisión relativa a cada sensor, además de tener un diseño más simple comparado a las técnicas de Lyapunov para el caso de modelos no lineales (por ejemplo, los obtenidos en el capítulo 3 en las ecuaciones (3.24),(3.50) y (3.51)).

Según el análisis presentado y la comparación entre metodologías, muchos de los filtros no lineales requieren la ejecución de una o varias operaciones matriciales complejas (inversa y raíz cuadrada) que dependen de la cantidad de estados n a estimar en el sistema así como el número de sensores disponibles (dimensión de la matriz R), por lo que muchos de estos filtros sufren del problema de dimensionalidad, ya que conforme aumenta n (o el número de partículas en el caso de los métodos generales), la complejidad del filtro aumentará de forma exponencial en muchos casos. Debido a esto, su implementación en sistemas de recursos limitados no es trivial, ya que se requieren de diversas estrategias con el fin de ajustar la ejecución del filtro para utilizar la memoria y tiempo de ejecución disponibles, pero sin influir negativamente en su desempeño. A pesar de estos ajustes, estrategias como el UKF, CKF y los filtros de partículas pueden ser sumamente costosas computacionalmente para ser implementadas en algunas plataformas.

Una comparación entre los filtros de estimación para sistemas no lineales se presenta en el trabajo realizado por [97], en donde se realiza una comparación entre el EKF, UKF y los filtros de partículas PF (al limitar la *pdf* a ser únicamente Gaussiana, *GPF* y en su versión estándar con el remuestreo,

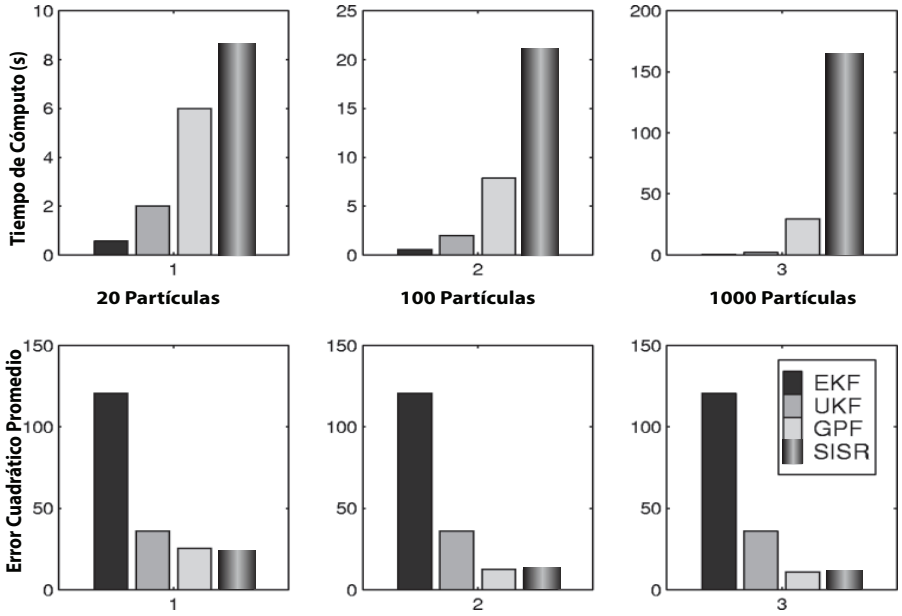


Figura 4.8: Comparación del EKF, UKF y del Filtro de Partículas en la estimación de un sistema altamente no lineal, desempeño y tiempos de ejecución (implementación realizada en Matlab®, ordenador con un procesador Intel Pentium III 450MHz) [97].

SISR) aplicados a un sistema altamente no lineal, según se muestra en la figura 4.8. En esta figura se observa claramente el diseño de compromiso entre los distintos tipos de filtros de estimación. Mientras que el EKF posee el mayor índice de error, también es el filtro menos demandante en cuanto a tiempo de cómputo, a diferencia del UKF y los PF. Además, se observa en el caso de los PF como mejoran su desempeño al incrementar el número de partículas utilizadas para realizar la estimación del estado, lo cual requiere a su vez un mayor tiempo de cómputo para obtener la solución. A pesar de que éste no es un ejemplo de localización o fusión sensorial, ejemplifica claramente tanto la precisión esperada de las distintas técnicas de filtrado para estimación así como el costo asociado a cada una de ellas.

En el caso de localización, se puede mejorar la precisión de técnicas como el EKF al utilizar la fusión de los distintos sensores disponibles en la plataforma, sin necesidad de aumentar el coste computacional en exceso al utilizar el

UKF, CKF o los PF. Por esta razón, se seleccionan de entre los filtros de estimación, las técnicas del KF para los modelos lineales (es el filtro óptimo en el caso lineal y solo requiere el cálculo de una matriz inversa) y el EKF para los modelos no lineales, al proveer las ventajas de implementación descritas con anterioridad y no requerir el cálculo de una raíz cuadrada matricial o el cálculo de múltiples soluciones (partículas) de los métodos alternativos.

Sin embargo, como se mencionó anteriormente, a mayor número de estados o sensores, se incrementa el coste computacional de obtener una matriz inversa, por lo que se debe ajustar el KF y el EKF para permitir su ejecución en plataformas de recursos limitados. Se descarta ajustar los filtros para utilizar una matriz de ganancia constante, sin requerir su cálculo en línea. Esto debido a que, a pesar de requerir menos recursos para su ejecución al no tener que calcular matrices inversas durante la ejecución del algoritmo dentro de la plataforma, en el caso de fusión sensorial se tendría una gran desventaja ya que no se podrían ajustar el aporte de los sensores en línea (a través de la matriz R_k), limitando grandemente las posibles aplicaciones de los algoritmos.

Por esta razón, en la presente tesis se prefiere realizar el ajuste de los algoritmos elegidos para evitar el cálculo de matrices inversas de dimensión alta, de manera que la ganancia pueda calcularse en línea en las plataformas de recursos limitados. El ajuste del KF y EKF se realizará mediante el uso de una combinación de los modelos obtenidos en el capítulo 3 junto con la incorporación de técnicas basadas en eventos, tal y como se describe en el siguiente capítulo. Finalmente, cabe mencionar que el filtro UKF será utilizado como validación del esquema básico de fusión sensorial mediante simulaciones externas a las plataformas de recursos limitados, tal y como se describe en los capítulos 5 y 6.

4.5 Conclusiones del Capítulo

En el presente capítulo se estudiaron los métodos existes en cuanto a estimación de estados y fusión sensorial, clasificados en *observadores* y *filtros* de estimación, ambos para sistemas lineales y no lineales:

- Los observadores proveen en ciertos casos una solución menos demandante en cuanto a recursos computacionales (memoria y tiempo de ejecución), pero no toman en cuenta en su diseño el ruido del proceso o el de medición. Además, requieren modelos complejos del sistema y para sistemas no lineales requieren el uso de técnicas de Lyapunov para obtener la ganancia del observador, siendo estas técnicas costosas en cuanto a tiempo de diseño, proporcionando soluciones muy específicas y poco flexibles ante variaciones en el esquema de fusión.

- Los filtros de estimación son métodos basados en modelos estocásticos que toman en cuenta, desde la etapa de diseño, la incorporación de la información disponible sobre el ruido del proceso y de la medición para estimar el estado. Permiten la incorporación de índices de desempeño a través de la ganancia del método (garantizando por ejemplo, un error de estimación mínimo o el error mínimo en el peor caso) y son métodos flexibles, permitiendo agregar o retirar dinámicamente sensores durante la ejecución del algoritmo, además de posibilitar la asignación de una importancia relativa a cada sensor de forma independiente según su precisión (a través de la matriz R_k). Por esta razón se seleccionaron estas técnicas para el desarrollo de los distintos algoritmos de fusión basada en eventos.

- De los múltiples algoritmos disponibles se eligieron el filtro de Kalman (KF) y el filtro de Kalman Extendido (EKF), los cuales sirven como base para el desarrollo de los algoritmos de fusión basada en eventos al no consumir recursos en exceso (comparados con el UKF, CKF y los PF), y al poder incrementar su precisión mediante la fusión de múltiples sensores en el caso de localización.

- No son convenientes para el caso de localización las estrategias con ganancia constante, al limitar grandemente las posibles aplicaciones de los métodos. Por esta razón, se prefiere obtener la ganancia en cada instante de muestreo, ajustando los algoritmos elegidos para evitar el cálculo de matrices inversas de dimensión alta, de manera que puedan utilizarse en plataformas de recursos limitados.

5 | Fusión Sensorial en Cascada para Localización

En el presente capítulo se exponen diversos algoritmos de fusión sensorial para la localización de robots móviles, utilizando los filtros de estimación seleccionados en el capítulo 4. Los algoritmos desarrollados se denotan como algoritmos basados en el *tiempo* al realizar la ejecución completa del filtro de estimación (etapas de predicción y corrección) en cada instante de tiempo k según el tiempo de muestreo regular T_s.

Los algoritmos basados en el tiempo se establecen con el fin de mejorar la precisión de la localización de un robot móvil al utilizar los distintos sensores disponibles en la plataforma en cada T_s. Se definen diversos algoritmos que sirven para realizar la comparación entre estrategias de corrección, ajustando la cantidad de sensores utilizados según los recursos disponibles; con estas modificaciones se propone un método que utiliza toda la información sensorial local disponible mediante una combinación de los modelos obtenidos en el capítulo 3 y una actualización global en cascada.

Para simplificar la notación en el capítulo se omiten los superíndices a priori "$-$", con lo que las ecuaciones de corrección de los filtros se escriben utilizando la convención algorítmica, en la que la variable a la izquierda de la igualdad es actualizada a partir de su valor previamente existente en la parte derecha de la ecuación. A continuación se exponen los métodos existentes (medición completa) junto con los algoritmos propuestos (medición seccionada/en cascada) para plataformas de recursos limitados.

5.1 Medición completa

Estos métodos se entienden como "tradicionales" en el sentido de que utilizan el modelo global tanto en sus versiones cinemáticas (ecuaciones (3.11) y (3.36)) como dinámicas (ecuaciones (3.24) y (3.51)) junto con la entrada $\boldsymbol{u}_k \in \Re^u$ y la totalidad de las mediciones disponibles $\boldsymbol{z}_k \in \Re^m$ obtenidas de los distintos sensores de la plataforma experimental para estimar el estado $\boldsymbol{x}_k \in \Re^n$. La mayoría de los métodos de fusión aplicada a la localización estudiados en los antecedentes del capítulo 2, pueden considerarse dentro de esta categoría.

La descripción de estos algoritmos se realiza al definir un vector de medición $\boldsymbol{z}_k = H_k x_k$ conformado por las mediciones útiles a la hora de obtener la localización de un robot móvil. De esta forma, se definen los vectores z_k en las ecuaciones (5.1) y (5.2) para los casos de navegación en interiores y exteriores respectivamente. En estas ecuaciones se considera que las mediciones disponibles en la plataforma experimental están constituidas por las velocidades lineales $v_{x,enc}$, $v_{y,enc}$ y angular ω_{enc} obtenidas de los encoders; además de ω_{gyr} y ω_{comp} de un giróscopo y una brújula respectivamente, junto con las aceleraciones (u_1, u_2, u_3) obtenidas mediante dos acelerómetros 3D colocados según el modelo de partículas obtenido en capítulo 3 (modelos (3.23),(3.50), figuras 3.4,3.6), donde u_1 es la suma de las aceleraciones en el eje x local, u_2 la suma para el eje y y u_3 la diferencia de las aceleraciones en el eje y (según la ecuación (3.49)). Como información global se tiene la postura del robot obtenida de una cámara cenital $L_{GM} = (x, y, \theta)_{GM}$ en el caso de navegación en interiores, o bien $L_{GPS} = (x, y, \theta)_{GPS}$ obtenida de un sensor GPS en el caso de navegación en exteriores.

Cabe destacar que en el caso de no considerar deslizamiento (por ejemplo para los modelos del robot diferencial, ecuación (3.24)) se reduce el vector z_k a 9 mediciones, ya que se considera que tanto la velocidad como la aceleración lateral (eje y local) son nulas, con lo que $v_{y,enc} = 0$ y $u_2 = 0$, obteniéndose las ecuaciones (5.3) y (5.4).

$$\boldsymbol{z}_{k,i} = H_k x_k = \begin{bmatrix} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & u_1 & u_2 & u_3 & x_{GM} & y_{GM} & \theta_{GM} \end{bmatrix}^T \quad (5.1)$$

$$\boldsymbol{z}_{k,e} = \left[\begin{array}{cccccccccccc} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & u_1 & u_2 & u_3 & x_{GPS} & y_{GPS} & \theta_{GPS} \end{array} \right]^T \quad (5.2)$$

$$\boldsymbol{z}_{k,i} = \left[\begin{array}{cccccccccc} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & u_1 & u_3 & x_{GM} & y_{GM} & \theta_{GM} \end{array} \right]^T \quad (5.3)$$

$$\boldsymbol{z}_{k,e} = \left[\begin{array}{cccccccccc} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & u_1 & u_3 & x_{GPS} & y_{GPS} & \theta_{GPS} \end{array} \right]^T \quad (5.4)$$

Al utilizar el z_k definido (ecuaciones de (5.1) a (5.4)) junto con los modelos cinemático (ecuaciones (3.11) y (3.36)) o dinámico (ecuaciones (3.24) y (3.51)) en un filtro de estimación, por ejemplo un EKF o UKF, se logra obtener la postura estimada del robot móvil $\hat{\boldsymbol{x}}_k = [x, y, \theta]_k$ y la covarianza del correspondiente error de estimación P_k, mediante la fusión de la información de los distintos sensores en cada instante de muestreo k. Cabe destacar que esta definición de z_k implica que los modelos obtenidos consideran que la entrada u_{k-1} es obtenida a partir del modelo de velocidades de las ruedas dependiente de la acción de control de la ecuación (3.57) para el caso diferencial, y de las ecuaciones (3.57), (3.39) y la integral de (3.54) para el caso de una plataforma Ackerman; aunque también se puede considerar un modelo sin entrada (al incorporar las aceleraciones a través de z_k).

Para realizar la estimación se requiere el estado x_0 y covarianza P_0 inicial (postura del robot al inicio del movimiento y la covarianza del error para este estado) junto con las matrices Q_k del ruido del proceso y R_k del ruido de medición, las cuales pueden ajustarse inicialmente como matrices diagonales con las covarianzas σ^2 individuales de cada sensor para R_k, y la confianza en el modelo obtenido expresada mediante una covarianza inicial en cada estado para Q_k. La fusión la realiza el EKF a través de la matriz H_k obtenida mediante la linealización de la ecuación de medición del modelo en el caso no lineal o bien mediante la definición lineal de las ecuaciones (5.1) a (5.4). Esta matriz establece una ecuación por cada estado, asignándole sus correspondientes mediciones, las cuales son incorporadas a la estimación mediante la ganancia K_k que depende de la precisión de cada sensor, definida en la matriz R_k. En el caso del UKF se sigue un procedimiento similar para la fusión pero utilizando la propagación de los puntos sigma para obtener $\hat{z}_k$ (misma dimensión que el z_k de las ecuaciones (5.1) a (5.4)).

Se reescriben el EKF y UKF de forma algorítmica ante el planteamiento realizado de la corrección basada en el tiempo, tal y como se muestra en el Algoritmo 1 para el EKF y en el Algoritmo 2 para el UKF. Estos algoritmos servirán para comparar entre las estrategias desarrolladas a lo largo del capítulo al tener teóricamente la mejor precisión ya que disponen de la información de los sensores en cada instante k.

Cabe destacar que la principal desventaja del planteamiento basado en el tiempo radica en el coste computacional, que es sumamente elevado al tener que realizar la inversión y/o raíz cuadrada matricial de dimensiones altas (en línea), por ejemplo, la inversa de una matriz 11×11 o 9×9 según las mediciones en el caso de las ecuaciones (5.1) a la (5.4) ya que, para obtener K_k, se realiza una inversión de una matriz de las mismas dimensiones que R en el EKF o de la suma de dimensiones de P_k, Q_k y R_k en el caso del UKF. Esto impide la implementación directa de este tipo de algoritmos en sistemas de recursos limitados. Además no se ha tomado en cuenta los posibles retardos de comunicación que pueden darse a la hora de obtener la medición global desde una cámara o GPS lo que podría limitar la utilización de esta información en cada instante k. Aun así, son métodos útiles en sistemas sin limitaciones de recursos y con gran disponibilidad de ancho de banda y velocidades de transmisión elevadas.

Como notación resumida, el EKF del Algoritmo 1 puede denotarse como *EKF5s11m* si utiliza un modelo de 5 estados (ecuación (3.24)) y 11 mediciones (ecuación (5.1) o (5.2)) utilizadas en cada instante k, de la misma forma, *EKF5s9m* indica 5 estados pero 9 mediciones (ecuaciones (5.3) o (5.4)). Esta notación se utilizará en las comparaciones de las pruebas experimentales debido a que da una indicación clara de la capacidad computacional requerida para calcular el algoritmo; ya que, conforme más mediciones se utilicen en los algoritmos basados en el tiempo, mayor será la dimensión de la inversa que se debe calcular para obtener la matriz de ganancia del filtro. Con esto se puede denotar el UKF del Algoritmo 2 como *UKF5s11m* (modelo (3.24), medición (5.1) o (5.2)) o como el *UKF6s9m* (modelo (3.51), medición (5.3) o (5.4)) según se requiera.

Algoritmo 1: Algoritmo EKF recursivo, corrección basada en el tiempo

Entrada: u_{k-1},$\hat{x}_{k-1}$,P_{k-1}

Medición: z_k=$\left[v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ u_1\ u_2\ u_3\ \{x\ y\ \theta\}_{GM/GPS} \right]^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(del modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,R_k

Salida: $\hat{x}_k$,P_k

Inicialización: $\hat{x}_0$,P_0

Para *el instante actual* k **hacer**

Predicción:

$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$

$A_k = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1},u_{k-1},0}$, $W_k = \left.\frac{\partial f}{\partial w}\right|_{\hat{x}_{k-1},u_{k-1},0}$

$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$

Corrección:

$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_k,0}$, $N_k = \left.\frac{\partial h}{\partial n}\right|_{\hat{x}_k,0}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k \left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

fin

Algoritmo 2: Algoritmo UKF recursivo, corrección basada en el tiempo

Entrada: u_{k-1},$\hat{x}_{k-1}$,P_{k-1}

Medición: z_k=$\left[v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ u_1\ u_2\ u_3\ \{x\ y\ \theta\}_{GM/GPS}\right]^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(del modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,R_k,n,κ

Salida: $\hat{x}_k$,P_k

Inicialización: Ampliar $\hat{x}_0^{(a)} = E\left[x, w, n\right]_k^T$ y $P_0^{(a)} = \text{diag}\{P_0, Q_0, R_0\}$, Definir Pesos: $W_{(0)} = \kappa/(n+\kappa)$,$W_{(i)} = 1/2\,(n+\kappa), i = 1, \cdots, 2n$

Para *el instante actual* k **hacer**

Predicción:

Obtener puntos sigma $\hat{x}_{k-1}^{(i)}$:

$\hat{x}_{k-1}^{(0)} = \hat{x}_{k-1}$

$$\hat{x}_{k-1}^{(i)} = \hat{x}_{k-1} + \widetilde{\mathrm{P}}_{(i)} \begin{cases} \widetilde{\mathrm{P}}_{(i)} = \left(\sqrt{(n+\kappa)\,P_{k-1}}\right)_{(i)}, i = 1, \cdots, n \\ \widetilde{\mathrm{P}}_{(i)} = -\left(\sqrt{(n+\kappa)\,P_{k-1}}\right)_{(i)}, i = n+1, \cdots, 2n \end{cases}$$

$$\hat{x}_k^{(i)} = f\left(\hat{x}_{k-1}^{(i)}, u_{k-1}\right), i = 0, \cdots, 2n$$

$$\hat{x}_k = \sum_{i=0}^{2n} W_{(i)} \hat{x}_k^{(i)}$$

$$P_k = \sum_{i=0}^{2n} W_{(i)} \left(\hat{x}_k^{(i)} - \hat{x}_k\right)\left(\hat{x}_k^{(i)} - \hat{x}_k\right)^T$$

Obtener Puntos Sigma $x_k^{(i)}$

$$\hat{z}_k^{(i)} = h\left(\hat{x}_k^{(i)}\right), i = 0, \cdots, 2n$$

$$\hat{z}_k = \sum_{i=0}^{2n} W_{(i)} \hat{z}_k^{(i)}$$

$$P_z = \sum_{i=0}^{2n} W_{(i)} \left(\hat{z}_k^{(i)} - \hat{z}_k\right)\left(\hat{z}_k^{(i)} - \hat{z}_k\right)^T$$

$$P_{xz} = \sum_{i=0}^{2n} W_{(i)} \left(\hat{x}_k^{(i)} - \hat{x}_k^-\right)\left(\hat{z}_k^{(i)} - \hat{z}_k\right)^T$$

Corrección:

$$K_k = P_{xz}(P_z)^{-1}$$

$$\hat{x}_k = \hat{x}_k + K_k\left(z_k - \hat{z}_k\right)$$

$$P_k = P_k - K_k P_z K_k^T$$

fin

5.2 Medición Seccionada

Con el fin de adaptar los algoritmos de fusión para las plataformas de recursos limitados se proponen en la presente tesis diversas modificaciones en el vector de medición z_k de los métodos de medición completa (ecuaciones (5.1) a la (5.4)). La idea principal es seccionar el vector z_k en distintas componentes, de forma que se pueda reducir la dimensión de la matriz inversa a calcular en línea, según se expone a continuación.

5.2.1 Asignación de Entradas

La primera reducción del vector de medición z_k se realiza utilizando el planteamiento de los modelos dinámicos globales no lineales, obtenidos por la descomposición de partículas (diferencial en la ecuación (3.24) y Ackerman en la ecuación (3.51)), en los cuales se puede utilizar como entrada al modelo las aceleraciones obtenidas de los acelerómetros del robot. De esta forma se define el vector de entrada según la ecuación (5.5) para el caso general y mediante la ecuación (5.6) si no hay deslizamiento en el eje y local de la plataforma. Cabe destacar que al utilizar las aceleraciones como entradas al modelo, la covarianza de los acelerómetros quedará representada mediante la covarianza del ruido del proceso en la matriz Q_k, incrementando el término correspondiente a las aceleraciones lineales (u_1, u_2) y angular (u_3). Este mismo esquema es útil en caso de no disponer de acelerómetros, en donde se requiere utilizar el modelo de la aceleración de las ruedas dependiente de la acción de control (ecuación (3.57), (3.57), (3.39) y la integral de (3.54)).

$$\boldsymbol{u}_k = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T \tag{5.5}$$

$$\boldsymbol{u}_k = \begin{bmatrix} u_1 & u_3 \end{bmatrix}^T \tag{5.6}$$

De esta forma se reduce el vector de medición z_k tal y como se muestra en las ecuaciones (5.7) y (5.8) para el caso general y (5.9) y (5.10) para el caso sin deslizamiento.

$$\boldsymbol{z}_{k,i} = \begin{bmatrix} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & x_{GM} & y_{GM} & \theta_{GM} \end{bmatrix}^T \tag{5.7}$$

$$\boldsymbol{z}_{k,e} = \begin{bmatrix} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & x_{GPS} & y_{GPS} & \theta_{GPS} \end{bmatrix}^T \tag{5.8}$$

$$\boldsymbol{z}_{k,i} = \begin{bmatrix} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & x_{GM} & y_{GM} & \theta_{GM} \end{bmatrix}^T \tag{5.9}$$

$$\boldsymbol{z}_{k,e} = \begin{bmatrix} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & x_{GPS} & y_{GPS} & \theta_{GPS} \end{bmatrix}^T \tag{5.10}$$

Utilizando el nuevo vector z_k y u_k se reescriben los algoritmos del EKF y UKF según se muestra en los algoritmos 3 y 4.

A diferencia de los algoritmos con medición completa, el esquema por asignación de entradas no permite la utilización simultánea de un modelo para las aceleraciones de los motores junto con las mediciones provenientes de los acelerómetros. Sin embargo, se logra reducir z_k de tal forma que la inversa de la matriz requerida para el cálculo de la ganancia del filtro tiene dimensión 7×7 u 8×8 en el caso con deslizamiento. Esto supone un ahorro considerable en el costo computacional del algoritmo. Sin embargo el costo es aún elevado para implementarlo en plataformas de recursos limitados por lo que se procede con una nueva división del vector z_k en la siguiente sección.

A partir de este punto no se modifican más versiones del UKF ya que, aunque se reduzcan los requerimientos para el cálculo de la inversa, siempre se deben realizar el cálculo de dos raíces cuadradas matriciales que requieren gran cantidad de recursos computacionales (ya que la dimensión de esta raíz es la suma de las dimensiones de P_k,Q_k,R_k). Por esta razón se utilizará el Algoritmo 4 únicamente para validar los algoritmos propuestos mediante simulación.

Algoritmo 3: Algoritmo EKF recursivo con asignación de entradas, corrección basada en el tiempo

Entrada: $u_{k-1} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T$,$\hat{x}_{k-1}$,$P_{k-1}$

Medición: z_k=$\begin{bmatrix} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} & \{x & y & \theta\}_{GM/GPS} \end{bmatrix}^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(del modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,R_k

Salida: $\hat{x}_k$,P_k

Inicialización: $\hat{x}_0$,P_0

Para *el instante actual* k **hacer**

Predicción:

$$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$$
$$A_k = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1},u_{k-1},0}$$
$$W_k = \left.\frac{\partial f}{\partial w}\right|_{\hat{x}_{k-1},u_{k-1},0}$$
$$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$$

Corrección:

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_k,0}$$
$$N_k = \left.\frac{\partial h}{\partial n}\right|_{\hat{x}_k,0}$$
$$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$$
$$\hat{x}_k = \hat{x}_k + K_k \left[z_k - h\left(\hat{x}_k, 0\right)\right]$$
$$P_k = \left(I - K_k H_k\right) P_k$$

fin

Algoritmo 4: Algoritmo UKF recursivo con asignación de entradas, corrección basada en el tiempo

Entrada: $u_{k-1} = [\ u_1 \quad u_2 \quad u_3\ ]^T$,$\hat{x}_{k-1}$,$P_{k-1}$

Medición: $z_k = \left[v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ \{x\ y\ \theta\}_{GM/GPS}\right]^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(del modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,R_k,n,κ

Salida: $\hat{x}_k$,P_k

Inicialización: Ampliar $\hat{x}_0^{(a)} = E\left[x, w, n\right]_k^T$ y $P_0^{(a)} = \text{diag}\{P_0, Q_0, R_0\}$, Definir Pesos: $W_{(0)} = \kappa/(n+\kappa)$,$W_{(i)} = 1/2\,(n+\kappa), i = 1, \cdots, 2n$

Para *el instante actual* k **hacer**

Predicción:

Obtener puntos sigma $\hat{x}_{k-1}^{(i)}$: $\hat{x}_{k-1}^{(0)} = \hat{x}_{k-1}$

$$\hat{x}_{k-1}^{(i)} = \hat{x}_{k-1} + \widetilde{\mathrm{P}}_{(i)} \begin{cases} \widetilde{\mathrm{P}}_{(i)} = \left(\sqrt{(n+\kappa)\,P_{k-1}}\right)_{(i)}, i = 1, \cdots, n \\ \widetilde{\mathrm{P}}_{(i)} = -\left(\sqrt{(n+\kappa)\,P_{k-1}}\right)_{(i)}, i = n+1, \cdots, 2n \end{cases}$$

$$\hat{x}_k^{(i)} = f\left(\hat{x}_{k-1}^{(i)}, u_{k-1}\right), i = 0, \cdots, 2n$$

$$\hat{x}_k = \sum_{i=0}^{2n} W_{(i)} \hat{x}_k^{(i)}$$

$$P_k = \sum_{i=0}^{2n} W_{(i)} \left(\hat{x}_k^{(i)} - \hat{x}_k\right)\left(\hat{x}_k^{(i)} - \hat{x}_k\right)^T$$

Obtener Puntos Sigma $x_k^{(i)}$

$$\hat{z}_k^{(i)} = h\left(\hat{x}_k^{(i)}\right), i = 0, \cdots, 2n$$

$$\hat{z}_k = \sum_{i=0}^{2n} W_{(i)} \hat{z}_k^{(i)}$$

$$P_z = \sum_{i=0}^{2n} W_{(i)} \left(\hat{z}_k^{(i)} - \hat{z}_k\right)\left(\hat{z}_k^{(i)} - \hat{z}_k\right)^T$$

$$P_{xz} = \sum_{i=0}^{2n} W_{(i)} \left(\hat{x}_k^{(i)} - \hat{x}_k^{-}\right)\left(\hat{z}_k^{(i)} - \hat{z}_k\right)^T$$

Corrección:

$$K_k = P_{xz}(P_z)^{-1}$$

$$\hat{x}_k = \hat{x}_k + K_k\,(z_k - \hat{z}_k)$$

$$P_k = P_k - K_k P_z K_k^T$$

fin

5.2.2 Filtro en Cascada: Fusión Local Estado Completo - Corrección Global

Tomando como punto de partida la reducción anterior con las entradas asignadas (ecuación (5.5)) en el modelo no lineal global (postura) para el robot Ackerman (ecuación (3.51), modelo con 6 estados), se realiza una segunda reducción del vector de medición z_k al separar las mediciones en locales (encoders, giróscopo y brújula) de las globales (cámara, GPS) según se muestra en la ecuación (5.11).

$$
\begin{aligned}
\boldsymbol{x}_k &= \left[\begin{array}{cccccc} x & y & \theta & v_x & v_y & \omega \end{array}\right]_k^T = \left[\begin{array}{cc} x_{k,p} & x_{k,v} \end{array}\right]^T \\
\boldsymbol{x}_{k,p} &= \left[\begin{array}{ccc} x & y & \theta \end{array}\right]_k^T \\
\boldsymbol{x}_{k,v} &= \left[\begin{array}{ccc} v_x & v_y & \omega \end{array}\right]_k^T \\
\boldsymbol{u}_k &= \left[\begin{array}{ccc} u_1 & u_2 & u_3 \end{array}\right]_k^T \\
z_k &= \left[\left\{\begin{array}{ccc} x & y & \theta \end{array}\right\}_{GM/GPS} \begin{array}{ccccc} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{array}\right]_k^T \\
&= \left[\begin{array}{cc} z_{k,p} & z_{k,v} \end{array}\right]^T \\
\boldsymbol{z}_{k,p} &= \boldsymbol{L}_{GM/GPS} = \left[\begin{array}{ccc} x_k & y_k & \theta_k \end{array}\right]_{GM/GPS}^T \\
\boldsymbol{z}_{k,v} &= \left[\begin{array}{ccccc} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{array}\right]_k^T
\end{aligned}
\tag{5.11}
$$

Utilizando la ecuación (5.11) se podría realizar la fusión sensorial en dos etapas: en la primera se realizaría la fusión de la información local (velocidades) empleando únicamente $z_{k,v}$ junto con el modelo no lineal del robot (3.51) en un algoritmo EKF local, que estimaría el estado completo x_k y en la segunda etapa (en cascada) se utilizaría la información global $z_{k,p}$ para actualizar únicamente la postura en el estado $x_{k,p}$ determinada mediante la primera etapa. De esta forma se realizarían dos inversiones matriciales, la local de dimensión 5×5 y la global de dimensión 3×3, las cuales tienen un menor coste computacional que el cálculo de la matriz inversa 8×8 para el vector de medición completo z_k (Algoritmos 1 al 4). Este método de filtros *en cascada* se propone en la presente tesis como estrategia fundamental para

el desarrollo de los algoritmos de fusión para recursos limitados, incluyendo los basados en eventos. Debido a esto se describe en detalle esta estrategia a continuación, con el fin de observar la diferencia respecto al método con medición completa z_k (ecuación (5.11)) en cuanto a estimación del estado $\hat{x}_k$ y covarianza P_k.

El desarrollo del método en cascada se realiza al expresar las ecuaciones de predicción y corrección del EKF con medición completa respecto a los componentes de velocidad y postura, de forma que se puedan deducir los filtros local y global al eliminar los términos correspondientes según las definiciones de la ecuación (5.11).

5.2.2.1 EKF en componentes: Predicción

Considerando el modelo no lineal de la postura del robot Ackerman (ecuación (3.51)) se procede a su linealización según la ecuación (5.12).

$$\begin{aligned}
\begin{bmatrix} x \\ y \\ \theta \\ v_x \\ v_y \\ \omega \end{bmatrix}_k &= \begin{bmatrix} x \\ y \\ \theta \\ v_x \\ v_y \\ \omega \end{bmatrix}_{k-1} + T_s \begin{bmatrix} v_x\cos\theta - v_y\sin\theta \\ v_x\sin\theta + v_y\cos\theta \\ \omega \\ v_y\omega + \lambda_a u_1 \\ -v_x\omega + \lambda_a u_2 \\ \lambda_\alpha \gamma u_3 \end{bmatrix}_{k-1} \\
&= \begin{bmatrix} x \\ y \\ \theta \\ v_x \\ v_y \\ \omega \end{bmatrix}_{k-1} + \begin{bmatrix} v_x\,\mathrm{cs} - v_y\,\mathrm{sn} \\ v_x\,\mathrm{sn} + v_y\,\mathrm{cs} \\ T_s\omega \\ T_s v_y\omega + T_s\lambda_a u_1 \\ -T_s v_x\omega + T_s\lambda_a u_2 \\ T_s\lambda_\alpha\gamma u_3 \end{bmatrix}_{k-1}, \begin{array}{l} \mathrm{cs} = T_s\cos\theta \\ \mathrm{sn} = T_s\sin\theta \end{array} \\
\Rightarrow A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\substack{x_{k-1} \\ u_{k-1}}} &= \begin{bmatrix} 1 & 0 & -v_x\,\mathrm{sn} - v_y\,\mathrm{cs} & \mathrm{cs} & -\mathrm{sn} & 0 \\ 0 & 1 & v_x\,\mathrm{cs} - v_y\,\mathrm{sn} & \mathrm{sn} & \mathrm{cs} & 0 \\ 0 & 0 & 1 & 0 & 0 & T_s \\ 0 & 0 & 0 & 1 & T_s\omega & T_s v_y \\ 0 & 0 & 0 & -T_s\omega & 1 & -T_s v_x \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{\substack{x_{k-1} \\ u_{k-1}}}
\end{aligned} \tag{5.12}$$

Utilizando la ecuación (5.12), se separa A_k en 4 componentes: A_p con el modelo de la postura, A_{pv} con la influencia de la velocidad en la postura, A_{vp} con la influencia de la postura en la velocidad (nula en el modelo linealizado (5.12)) y A_v con el modelo de la velocidad; tal y como se muestra en la ecuación (5.13).

$$A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{x_{k-1},u_{k-1}} = \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix}_{k-1}, A_p = \begin{bmatrix} 1 & 0 & -v_x \operatorname{sn} - v_y \operatorname{cs} \\ 0 & 1 & v_x \operatorname{cs} - v_y \operatorname{sn} \\ 0 & 0 & 1 \end{bmatrix} \quad (5.13)$$

Se definen y seccionan las matrices de covarianza del error P_k y covarianza del ruido del proceso Q_k para el modelo (3.51), según se muestran en las ecuaciones (5.14) y (5.15) respectivamente.

$$P_{k-1} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} & P_{16} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} & P_{26} \\ P_{31} & P_{32} & P_{33} & P_{34} & P_{35} & P_{36} \\ P_{41} & P_{42} & P_{43} & P_{44} & P_{45} & P_{46} \\ P_{51} & P_{52} & P_{53} & P_{54} & P_{55} & P_{56} \\ P_{61} & P_{62} & P_{63} & P_{64} & P_{65} & P_{66} \end{bmatrix}_{k-1} = \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix}_{k-1} \quad (5.14)$$

$$Q_{k-1} = \begin{bmatrix} Q_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & Q_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & Q_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & Q_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & Q_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & Q_{66} \end{bmatrix}_{k-1} = \begin{bmatrix} Q_p & Q_{pv} \\ Q_{vp} & Q_v \end{bmatrix}_{k-1} \quad (5.15)$$

El valor exacto de la matriz Q_k puede determinarse de forma precisa para sistemas lineales a partir de la matriz de covarianza del proceso continua Qc (la cual pocas veces es conocida de forma exacta), aunque puede resultar un proceso complejo en el caso de sistemas no lineales de dimensiones altas. Esto debido a que se requiere el cálculo de la integral $Q_{k-1} = \int_{t_{k-1}}^{t_k} e^{A(t_k-\tau)} Q_c(\tau)\, e^{A^T(t_k-\tau)} d\tau$ [28, 64, 171, 151], por lo que en estos

casos se suele determinar mediante distintas aproximaciones. Por ejemplo, a partir de la matriz continua Qc se puede realizar [64, 151] la aproximación $Q_k = WQcW^T Ts$, con W obtenida de la linealización del modelo del sistema (obtenida por ejemplo para el caso del EKF en el algoritmo 3). Aun así, aunque se tenga su valor exacto, es común realizar el ajuste de esta matriz a partir de una matriz diagonal inicial con el fin de ajustar el desempeño del filtro, para lo que en general se suele aumentar su valor para mejorar la estimación [28, 64, 171, 151]. Éste es el enfoque seguido en la presente tesis, utilizar una matriz Q_k diagonal y ajustarla de acuerdo con la plataforma experimental según se describe con más detalle en el capítulo 8.

El EKF realiza la predicción de la covarianza del error P_k^- según su definición (Algoritmo 3) al utilizar las matrices A_k, P_k y Q_k definidas en las ecuaciones (5.13), (5.14) y (5.15) junto con W_k que es igual a la matriz identidad 6×6, tal y como se muestra en la ecuación (5.16) (se simplifica la notación al omitir el subíndice $k-1$).

$$
\begin{aligned}
P_k &= A_{k-1} P_{k-1} A_{k-1}^T + Q_{k-1} \\
&= \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix} \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix}^T + \begin{bmatrix} Q_p & Q_{pv} \\ Q_{vp} & Q_v \end{bmatrix} \\
&= \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix} \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} \begin{bmatrix} A_p^T & A_{vp}^T \\ A_{pv}^T & A_v^T \end{bmatrix} + \begin{bmatrix} Q_p & Q_{pv} \\ Q_{vp} & Q_v \end{bmatrix} \\
&= \begin{bmatrix} A_p P_p + A_{pv} P_{vp} & A_p P_{pv} + A_{pv} P_v \\ A_{vp} P_p + A_v P_{vp} & A_{vp} P_{pv} + A_v P_v \end{bmatrix} \begin{bmatrix} A_p^T & A_{vp}^T \\ A_{pv}^T & A_v^T \end{bmatrix} + \begin{bmatrix} Q_p & Q_{pv} \\ Q_{vp} & Q_v \end{bmatrix} \qquad (5.16) \\
\because \quad & \Omega_p = (A_p P_p + A_{pv} P_{vp}), \Omega_{pv} = (A_p P_{pv} + A_{pv} P_v), \\
& \Omega_{vp} = (A_{vp} P_p + A_v P_{vp}), \Omega_v = (A_{vp} P_{pv} + A_v P_v) \\
\Rightarrow P_k &= \begin{bmatrix} \Omega_p & \Omega_{pv} \\ \Omega_{vp} & \Omega_v \end{bmatrix} \begin{bmatrix} A_p^T & A_{vp}^T \\ A_{pv}^T & A_v^T \end{bmatrix} + \begin{bmatrix} Q_p & Q_{pv} \\ Q_{vp} & Q_v \end{bmatrix} \\
\therefore P_k &= \begin{bmatrix} \Omega_p A_p^T + \Omega_{pv} A_{pv}^T + Q_p & \Omega_p A_{vp}^T + \Omega_{pv} A_v^T + Q_{pv} \\ \Omega_{vp} A_p^T + \Omega_v A_{pv}^T + Q_{vp} & \Omega_{vp} A_{vp}^T + \Omega_v A_v^T + Q_v \end{bmatrix}
\end{aligned}
$$

Se puede considerar que $A_{vp} = 0_{3\times3}$, ya que según el modelo (3.51) es una matriz nula (ecuación (5.12)). De esta forma, se logra simplificar la ecuación (5.16) en (5.17), en donde se podrían eliminar además los términos $Q_{vp} = Q_{pv} = 0_{3\times3}$ cuando Q_k es diagonal (ecuación (5.15)).

$$\begin{aligned} &\because A_{vp} = 0_{3\times3} \\ \Rightarrow \Omega_p &= \left(A_p P_p + A_{pv} P_{vp}\right), \Omega_{pv} = \left(A_p P_{pv} + A_{pv} P_v\right), \\ \Omega_{vp} &= \left(A_v P_{vp}\right), \Omega_v = \left(A_v P_v\right) \\ \therefore P_k &= \begin{bmatrix} \Omega_p A_p^T + \Omega_{pv} A_{pv}^T + Q_p & \Omega_{pv} A_v^T + Q_{pv} \\ \Omega_{vp} A_p^T + \Omega_v A_{pv}^T + Q_{vp} & \Omega_v A_v^T + Q_v \end{bmatrix} = \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix}_k \end{aligned} \tag{5.17}$$

Se observa de la ecuación (5.17) el proceso de actualización en el tiempo para la covarianza del error, la cual actualiza P_{k-1} mediante A_{k-1} y Q_{k-1} para obtener P_k^- que es utilizada en la etapa de corrección del EKF. Para ahorrar recursos computacionales se pueden observar la evolución de los componentes de P_k^- en el tiempo al realizar la implementación del filtro, con el fin de determinar si alguno es nulo y de esta forma evitar su cálculo en tiempo real. Cabe destacar la influencia directa de cada término de Q_{k-1} en los componentes de la matriz de covarianza del error P_k, lo que permite incrementar o disminuir su valor según se defina Q_{k-1}; por ejemplo, en el caso de que requerir aumentar P_k para modificar el comportamiento del filtro bastará con realizar el correspondiente incremento en Q_{k-1}.

Concluida la etapa de predicción por componentes se procede a continuación con el desarrollo de la corrección del EKF (solo ha sido necesario expresar P_k^- ya que para x_k^- se utiliza directamente el modelo no lineal, según la definición en el algoritmo 3).

5.2.2.2 EKF en componentes: Corrección

Para el modelo (3.51) se definen y seccionan las matrices de medición H_k y la de covarianza del ruido en la medición R_k, según se define en las ecuaciones (5.18) y (5.19) respectivamente. Cabe destacar que para el caso del vector z_k definido en la ecuación (5.11), H_k es una matriz lineal.

$$z_k = H_k x_k \Rightarrow H_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_k = \begin{bmatrix} H_p & H_{pv} \\ H_{vp} & H_v \end{bmatrix}_k \tag{5.18}$$

$$R_k = \begin{bmatrix} R_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & R_{22} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & R_{33} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & R_{44} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & R_{55} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & R_{66} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & R_{77} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & R_{88} \end{bmatrix}_k = \begin{bmatrix} R_p & R_{pv} \\ R_{vp} & R_v \end{bmatrix}_k \tag{5.19}$$

Se procede con el cálculo de la matriz inversa requerida para obtener la ganancia del filtro K_k según se describe en la ecuación (5.20), en donde se utilizan las matrices H_k y R_k según las ecuaciones (5.18) y (5.19), junto con

la matriz P_k obtenida en el resultado de la ecuación (5.17) (se simplifica la notación al omitir el subíndice k).

$$
\begin{aligned}
& HP_k^- H^T + R_k = \\
& = \begin{bmatrix} H_p & H_{pv} \\ H_{vp} & H_v \end{bmatrix} \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} \begin{bmatrix} H_p & H_{pv} \\ H_{vp} & H_v \end{bmatrix}^T + \begin{bmatrix} R_p & R_{pv} \\ R_{vp} & R_v \end{bmatrix} \\
& = \begin{bmatrix} H_pP_p + H_{pv}P_{vp} & H_pP_{pv} + H_{pv}P_v \\ H_{vp}P_p + H_vP_{vp} & H_{vp}P_{pv} + H_vP_v \end{bmatrix} \begin{bmatrix} H_p^T & H_{vp}^T \\ H_{pv}^T & H_v^T \end{bmatrix} + \begin{bmatrix} R_p & R_{pv} \\ R_{vp} & R_v \end{bmatrix} \\
& \quad \because C_1 = (H_pP_p + H_{pv}P_{vp}), C_2 = (H_pP_{pv} + H_{pv}P_v), \\
& \quad C_3 = (H_{vp}P_p + H_vP_{vp}), C_4 = (H_{vp}P_{pv} + H_vP_v) \\
& \Rightarrow HP_k^- H^T + R_k = \\
& = \begin{bmatrix} C_1 & C_2 \\ C_3 & C_4 \end{bmatrix} \begin{bmatrix} H_p^T & H_{vp}^T \\ H_{pv}^T & H_v^T \end{bmatrix} + \begin{bmatrix} R_p & R_{pv} \\ R_{vp} & R_v \end{bmatrix} \\
& = \begin{bmatrix} C_1H_p^T + C_2H_{pv}^T + R_p & C_1H_{vp}^T + C_2H_v^T + R_{pv} \\ C_3H_p^T + C_4H_{pv}^T + R_{pv} & C_3H_{vp}^T + C_4H_v^T + R_v \end{bmatrix} = \begin{bmatrix} M_p & M_{pv} \\ M_{vp} & M_v \end{bmatrix}
\end{aligned}
\tag{5.20}
$$

Al considerar que H_{vp}, R_{vp}, H_{pv} y R_{pv} son matrices nulas según sus definiciones en las ecuaciones (5.18) y (5.19), se simplifica la ecuación (5.20) en (5.21).

$$
\begin{aligned}
& \because H_{vp} = R_{vp} = 0_{5\times3}, H_{pv} = 0_{3\times3}, R_{pv} = 0_{3\times5} \\
& \Rightarrow C_1 = (H_pP_p), C_2 = (H_pP_{pv}), C_3 = (H_vP_{vp}), C_4 = (H_vP_v) \\
& \Rightarrow HP_k^- H^T + R_k = \begin{bmatrix} C_1H_p^T + R_p & C_2H_v^T \\ C_3H_p^T & C_4H_v^T + R_v \end{bmatrix} \\
& \therefore HP_k^- H^T + R_k = \begin{bmatrix} H_pP_pH_p^T + R_p & H_pP_{pv}H_v^T \\ H_vP_{vp}H_p^T & H_vP_vH_v^T + R_v \end{bmatrix} = \begin{bmatrix} M_p & M_{pv} \\ M_{vp} & M_v \end{bmatrix}
\end{aligned}
\tag{5.21}
$$

Se obtiene la inversa de la ecuación (5.21) en la ecuación (5.22) mediante la fórmula de inversión de matrices en bloque (ver fórmulas en [133] para las operaciones básicas con matrices en bloque).

$$\begin{aligned}\left(HP_k^- H^T + R_k\right)^{-1} &= \begin{bmatrix} M_p^{-1} + M_p^{-1}M_{pv}S_vM_{vp}M_p^{-1} & -M_p^{-1}M_{pv}S_v \\ -S_vM_{vp}M_p^{-1} & S_v \end{bmatrix} \\ &= \begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix}, \\ S_v &= \left(M_v - M_{vp}M_p^{-1}M_{pv}\right)^{-1}\end{aligned} \tag{5.22}$$

Se procede con el cálculo de la ganancia K_k al utilizar el resultado de la ecuación (5.22) según la definición en el algoritmo 3) tal y como se muestra en la ecuación (5.23)

$$\begin{aligned}K_k &= P_kH^T\left(HP_k^- H^T + R_k\right)^{-1} \\ &= \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix}\begin{bmatrix} H_p & H_{pv} \\ H_{vp} & H_v \end{bmatrix}^T\begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix} \\ &= \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix}\begin{bmatrix} H_p^T & H_{vp}^T \\ H_{pv}^T & H_v^T \end{bmatrix}\begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix} \\ &= \begin{bmatrix} P_pH_p^T + P_{pv}H_{pv}^T & P_pH_{vp}^T + P_{pv}H_v^T \\ P_{vp}H_p^T + P_vH_{pv}^T & P_{vp}H_{vp}^T + P_vH_v^T \end{bmatrix}\begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix} \\ &\because D_1 = \left(P_pH_p^T + P_{pv}H_{pv}^T\right), D_2 = \left(P_pH_{vp}^T + P_{pv}H_v^T\right), \\ &\quad D_3 = \left(P_{vp}H_p^T + P_vH_{pv}^T\right), D_4 = \left(P_{vp}H_{vp}^T + P_vH_v^T\right) \\ \therefore K_k &= \begin{bmatrix} D_1 & D_2 \\ D_3 & D_4 \end{bmatrix}\begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix} = \begin{bmatrix} D_1N_p + D_2N_{vp} & D_1N_{pv} + D_2N_v \\ D_3N_p + D_4N_{vp} & D_3N_{pv} + D_4N_v \end{bmatrix} \\ &= \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix}\end{aligned} \tag{5.23}$$

Recordando nuevamente que H_{vp} y H_{pv} son matrices nulas, se realiza la simplificación de la ecuación (5.23) para obtener la ganancia en componentes según se muestra en la ecuación (5.24).

$$
\begin{aligned}
&\because H_{vp} = 0_{5\times 3}, H_{pv} = 0_{3\times 3} \\
&\Rightarrow D_1 = \left(P_p H_p^T\right), D_2 = \left(P_{pv} H_v^T\right), D_3 = \left(P_{vp} H_p^T\right), D_4 = \left(P_v H_v^T\right) \\
&\therefore K_k = \begin{bmatrix} P_p H_p^T N_p + P_{pv} H_v^T N_{vp} & P_p H_p^T N_{pv} + P_{pv} H_v^T N_v \\ P_{vp} H_p^T N_p + P_v H_v^T N_{vp} & P_{vp} H_p^T N_{pv} + P_v H_v^T N_v \end{bmatrix} \\
&\qquad = \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix}
\end{aligned}
\tag{5.24}
$$

Con la ganancia K_k obtenida en la ecuación (5.24) y la definición del estado y la medición realizadas en la ecuación (5.11), se realiza la corrección del estado x_k según se describe en la ecuación (5.25), se ha tomado en cuenta que H_{vp} y H_{pv} son matrices nulas.

$$
\begin{aligned}
\because H_k \hat{x}_k &= \begin{bmatrix} H_p & H_{pv} \\ H_{vp} & H_v \end{bmatrix} \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} = \begin{bmatrix} H_p x_{k,p} + H_{pv} \hat{x}_{k,v} \\ H_{vp} x_{k,p} + H_v \hat{x}_{k,v} \end{bmatrix} = \begin{bmatrix} H_p \hat{x}_{k,p} \\ H_v \hat{x}_{k,v} \end{bmatrix} \\
\Rightarrow \hat{x}_k &= \hat{x}_k + K_k \left(z_k - \hat{z}_k\right) = \hat{x}_k + K_k \left(z_k - H_k \hat{x}_k\right) = \hat{x}_k + K_k \Delta_{k,p,v} \\
&= \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \begin{bmatrix} z_{k,p} - H_p \hat{x}_{k,p} \\ z_{k,v} - H_v \hat{x}_{k,v} \end{bmatrix} \\
&= \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \begin{bmatrix} \Delta_{k,p} \\ \Delta_{k,v} \end{bmatrix} \\
\therefore \hat{x}_k &= \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p \Delta_{k,p} + K_{pv} \Delta_{k,v} \\ K_{vp} \Delta_{k,p} + K_v \Delta_{k,v} \end{bmatrix}
\end{aligned}
\tag{5.25}
$$

Se procede con la actualización de la covarianza del error de estimación P_k tal y como se muestra en la ecuación (5.26) al utilizar las matrices H_k y K_k según las ecuaciones (5.18) y (5.24), junto con la matriz P_k obtenida de (5.17) (se simplifica la notación al omitir el subíndice k).

$$
\begin{aligned}
P_k &= (I - K_k H) P_k = P_k - K_k H P_k \\
&= \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} - \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \begin{bmatrix} H_p & H_{pv} \\ H_{vp} & H_v \end{bmatrix} \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} \\
&= \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} - \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \begin{bmatrix} H_p P_p + H_{pv} P_{vp} & H_p P_{pv} + H_{pv} P_v \\ H_{vp} P_p + H_v P_{vp} & H_{vp} P_{pv} + H_v P_v \end{bmatrix} \\
&\because \Sigma_p = (H_p P_p + H_{pv} P_{vp}), \Sigma_{pv} = (H_p P_{pv} + H_{pv} P_v), \\
&\quad \Sigma_{vp} = (H_{vp} P_p + H_v P_{vp}), \Sigma_v = (H_{vp} P_{pv} + H_v P_v) \\
\Rightarrow P_k &= \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} - \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \begin{bmatrix} \Sigma_p & \Sigma_{pv} \\ \Sigma_{vp} & \Sigma_v \end{bmatrix} \\
&= \begin{bmatrix} P_p & P_{pv} \\ P_{vp} & P_v \end{bmatrix} - \begin{bmatrix} K_p \Sigma_p + K_{pv} \Sigma_{vp} & K_p \Sigma_{pv} + K_{pv} \Sigma_v \\ K_{vp} \Sigma_p + K_v \Sigma_{vp} & K_{vp} \Sigma_{pv} + K_v \Sigma_v \end{bmatrix} \\
\therefore P_k &= \begin{bmatrix} P_p - K_p \Sigma_p - K_{pv} \Sigma_{vp} & P_{pv} - K_p \Sigma_{pv} - K_{pv} \Sigma_v \\ P_{vp} - K_{vp} \Sigma_p - K_v \Sigma_{vp} & P_v - K_{vp} \Sigma_{pv} - K_v \Sigma_v \end{bmatrix}
\end{aligned}
\tag{5.26}
$$

Se realiza la simplificación de la ecuación (5.23) al sustituir las matrices nulas H_{vp}, R_{vp}, H_{pv} y R_{pv}, con lo que se obtiene la ganancia en componentes según se muestra en la ecuación (5.27).

$$
\begin{aligned}
&\because H_{vp} = R_{vp} = 0_{5\times 3}, H_{pv} = 0_{3\times 3}, R_{pv} = 0_{3\times 5} \\
&\Rightarrow \Sigma_p = (H_p P_p), \Sigma_{pv} = (H_p P_{pv}), \Sigma_{vp} = (H_v P_{vp}), \Sigma_v = (H_v P_v) \\
&\therefore P_k = \begin{bmatrix} P_p - K_p H_p P_p - K_{pv} H_v P_{vp} & P_{pv} - K_p H_p P_{pv} - K_{pv} H_v P_v \\ P_{vp} - K_{vp} H_p P_p - K_v H_v P_{vp} & P_v - K_{vp} H_p P_{pv} - K_v H_v P_v \end{bmatrix}
\end{aligned}
\tag{5.27}
$$

Con las ecuaciones obtenidas para $\hat{x}_k$ y P_k concluye la obtención del EKF por componentes el cual permite analizar la influencia entre los términos de postura y velocidad además de obtener los filtros local y global según se describe a continuación. Se resume los resultados obtenidos para la descomposición en componentes del filtro EKF/KF en el caso general según la ecuación (5.28) y para el caso simplificado específico al modelo Ackerman (ecuación (3.51)) que considera las respectivas matrices nulas en la ecuación (5.29).

EKF en Componentes, versión general

Predicción,$P_k = [P_p \;\; P_{pv}; P_{vp} \;\; P_v]$,$Q_k = [Q_p \;\; Q_{pv}; Q_{vp} \;\; Q_v]$:

$$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$$

$$A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\substack{x_{k-1} \\ u_{k-1}}} = \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix}, \quad \begin{array}{l} \Omega_p = \left(A_p P_p + A_{pv} P_{vp}\right), \Omega_{pv} = \left(A_p P_{pv} + A_{pv} P_v\right) \\ \Omega_{vp} = \left(A_{vp} P_p + A_v P_{vp}\right), \Omega_v = \left(A_{vp} P_{pv} + A_v P_v\right) \end{array}$$

$$P_k = A_{k-1} P_{k-1} A_{k-1}^T + Q_{k-1} = \begin{bmatrix} \Omega_p A_p^T + \Omega_{pv} A_{pv}^T + Q_p & \Omega_p A_{vp}^T + \Omega_{pv} A_v^T + Q_{pv} \\ \Omega_{vp} A_p^T + \Omega_v A_{pv}^T + Q_{vp} & \Omega_{vp} A_{vp}^T + \Omega_v A_v^T + Q_v \end{bmatrix}$$

Corrección,$H_k = [H_p \;\; H_{pv}; H_{vp} \;\; H_v]$,$R_k = [R_p \;\; R_{pv}; R_{vp} \;\; R_v]$:

$$\begin{array}{l} C_1 = \left(H_p P_p + H_{pv} P_{vp}\right), C_2 = \left(H_p P_{pv} + H_{pv} P_v\right), \\ C_3 = \left(H_{vp} P_p + H_v P_{vp}\right), C_4 = \left(H_{vp} P_{pv} + H_v P_v\right) \end{array}$$

$$H P_k^- H^T + R_k = \begin{bmatrix} C_1 H_p^T + C_2 H_{pv}^T + R_p & C_1 H_{vp}^T + C_2 H_v^T + R_{pv} \\ C_3 H_p^T + C_4 H_{pv}^T + R_{pv} & C_3 H_{vp}^T + C_4 H_v^T + R_v \end{bmatrix} = \begin{bmatrix} M_p & M_{pv} \\ M_{vp} & M_v \end{bmatrix}$$

$$\left(H P_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} M_p^{-1} + M_p^{-1} M_{pv} S_v M_{vp} M_p^{-1} & -M_p^{-1} M_{pv} S_v \\ -S_v M_{vp} M_p^{-1} & S_v \end{bmatrix} = \begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix}$$

$$S_v = \left(M_v - M_{vp} M_p^{-1} M_{pv}\right)^{-1}$$

$$\begin{array}{l} D_1 = \left(P_p H_p^T + P_{pv} H_{pv}^T\right), D_2 = \left(P_p H_{vp}^T + P_{pv} H_v^T\right), \\ D_3 = \left(P_{vp} H_p^T + P_v H_{pv}^T\right), D_4 = \left(P_{vp} H_{vp}^T + P_v H_v^T\right) \end{array}$$

$$K_k = P_k H^T \left(H P_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} D_1 N_p + D_2 N_{vp} & D_1 N_{pv} + D_2 N_v \\ D_3 N_p + D_4 N_{vp} & D_3 N_{pv} + D_4 N_v \end{bmatrix} = \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix}$$

$$\hat{x}_k = \hat{x}_k + K_k\left(z_k - \hat{z}_k\right) = \hat{x}_k + K_k \Delta_{k,p,v} = \hat{x}_k + K_k \begin{bmatrix} \Delta_{k,p} & \Delta_{k,v} \end{bmatrix}^T$$

$$\hat{x}_k = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p \Delta_{k,p} + K_{pv} \Delta_{k,v} \\ K_{vp} \Delta_{k,p} + K_v \Delta_{k,v} \end{bmatrix}$$

$$P_k = P_k - K_k H P_k$$

$$\begin{array}{l} \Sigma_p = \left(H_p P_p + H_{pv} P_{vp}\right), \Sigma_{pv} = \left(H_p P_{pv} + H_{pv} P_v\right), \\ \Sigma_{vp} = \left(H_{vp} P_p + H_v P_{vp}\right), \Sigma_v = \left(H_{vp} P_{pv} + H_v P_v\right) \end{array}$$

$$P_k = \begin{bmatrix} P_p - K_p \Sigma_p - K_{pv} \Sigma_{vp} & P_{pv} - K_p \Sigma_{pv} - K_{pv} \Sigma_v \\ P_{vp} - K_{vp} \Sigma_p - K_v \Sigma_{vp} & P_v - K_{vp} \Sigma_{pv} - K_v \Sigma_v \end{bmatrix}$$

(5.28)

EKF en Componentes, versión simplificada para el modelo (3.51)

$\because A_{vp} = 0_{3\times3}, H_{vp} = R_{vp} = 0_{5\times3}, H_{pv} = 0_{3\times3}, R_{pv} = 0_{3\times5}$

Predicción,$P_k = [P_p \;\; P_{pv}; P_{vp} \;\; P_v]$,$Q_k = [Q_p \;\; Q_{pv}; Q_{vp} \;\; Q_v]$:

$$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$$

$$A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{x_{k-1},u_{k-1}} = \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix}$$

$$\Omega_p = \left(A_p P_p + A_{pv} P_{vp}\right), \Omega_{pv} = \left(A_p P_{pv} + A_{pv} P_v\right), \Omega_{vp} = \left(A_v P_{vp}\right), \Omega_v = \left(A_v P_v\right)$$

$$P_k = A_{k-1} P_{k-1} A_{k-1}^T + Q_{k-1} = \begin{bmatrix} \Omega_p A_p^T + \Omega_{pv} A_{pv}^T + Q_p & \Omega_{pv} A_v^T + Q_{pv} \\ \Omega_{vp} A_p^T + \Omega_v A_{pv}^T + Q_{vp} & \Omega_v A_v^T + Q_v \end{bmatrix}$$

Corrección,$H_k = [H_p \;\; H_{pv}; H_{vp} \;\; H_v]$,$R_k = [R_p \;\; R_{pv}; R_{vp} \;\; R_v]$:

$$C_1 = \left(H_p P_p\right), C_2 = \left(H_p P_{pv}\right), C_3 = \left(H_v P_{vp}\right), C_4 = \left(H_v P_v\right)$$

$$H P_k^- H^T + R_k = \begin{bmatrix} H_p P_p H_p^T + R_p & H_p P_{pv} H_v^T \\ H_v P_{vp} H_p^T & H_v P_v H_v^T + R_v \end{bmatrix} = \begin{bmatrix} M_p & M_{pv} \\ M_{vp} & M_v \end{bmatrix}$$

$$\left(H P_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} M_p^{-1} + M_p^{-1} M_{pv} S_v M_{vp} M_p^{-1} & -M_p^{-1} M_{pv} S_v \\ -S_v M_{vp} M_p^{-1} & S_v \end{bmatrix} = \begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix}$$

$$S_v = \left(M_v - M_{vp} M_p^{-1} M_{pv}\right)^{-1}$$

$$D_1 = \left(P_p H_p^T\right), D_2 = \left(P_{pv} H_v^T\right), D_3 = \left(P_{vp} H_p^T\right), D_4 = \left(P_v H_v^T\right)$$

$$K_k = P_k H^T \left(H P_k^- H^T + R_k\right)^{-1}$$

$$= \begin{bmatrix} P_p H_p^T N_p + P_{pv} H_v^T N_{vp} & P_p H_p^T N_{pv} + P_{pv} H_v^T N_v \\ P_{vp} H_p^T N_p + P_v H_v^T N_{vp} & P_{vp} H_p^T N_{pv} + P_v H_v^T N_v \end{bmatrix} = \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix}$$

$$\hat{x}_k = \hat{x}_k + K_k\left(z_k - \hat{z}_k\right) = \hat{x}_k + K_k\left(z_k - H_k \hat{x}_k\right) = \hat{x}_k + K_k \begin{bmatrix} \Delta_{k,p} & \Delta_{k,v} \end{bmatrix}^T$$

$$\hat{x}_k = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p \Delta_{k,p} + K_{pv} \Delta_{k,v} \\ K_{vp} \Delta_{k,p} + K_v \Delta_{k,v} \end{bmatrix}, \begin{bmatrix} \Delta_{k,p} \\ \Delta_{k,v} \end{bmatrix} = \begin{bmatrix} z_{k,p} - H_p \hat{x}_{k,p} \\ z_{k,v} - H_v \hat{x}_{k,v} \end{bmatrix}$$

$$P_k = P_k - K_k H P_k$$

$$\Sigma_p = \left(H_p P_p\right), \Sigma_{pv} = \left(H_p P_{pv}\right), \Sigma_{vp} = \left(H_v P_{vp}\right), \Sigma_v = \left(H_v P_v\right)$$

$$P_k = \begin{bmatrix} P_p - K_p H_p P_p - K_{pv} H_v P_{vp} & P_{pv} - K_p H_p P_{pv} - K_{pv} H_v P_v \\ P_{vp} - K_{vp} H_p P_p - K_v H_v P_{vp} & P_v - K_{vp} H_p P_{pv} - K_v H_v P_v \end{bmatrix}$$

(5.29)

5.2.2.3 Filtro EKF local: velocidades

El siguiente paso para determinar el filtro en cascada es la obtención del filtro local, el cual se deduce utilizando el filtro EKF en componentes de la ecuación (5.29) empleando únicamente los sensores locales (velocidades) para actualizar el estado y la covarianza. En cuanto a la etapa de predicción, el modelo (3.51) a utilizar en el filtro local es el mismo que para el EKF seccionado, por lo que la etapa de predicción es la misma en ambos filtros .

De esta forma, al considerar que no se realiza la actualización de la postura (sin medición global $z_{k,p}$) se remplaza la matriz H_p por una matriz nula $0_{3\times3}$ indicando al filtro que no se disponen de los sensores globales. De esta forma, se obtiene la ecuación (5.30) con el filtro local, en donde m corresponde a la dimensión de z_k (8 mediciones, ecuación (5.11)) y la dimensión de la medición global $z_{k,p}$ es 3×1. En adelante y para simplificar la notación, las matrices nulas se indican sin subíndice, pero cabe destacar que sus dimensiones son acordes al orden del sistema n, de la medición, m y de las medidas globales (3×1), según se definió en las ecuaciones (5.11), (5.18) y (5.19).

EKF local para el modelo (3.51)

$\because H_p = H_{pv} = A_{vp} = 0_{3\times3}, H_{vp} = R_{vp} = 0_{5\times3}, R_{pv} = 0_{3\times5}$

Predicción: igual que el EKF en componentes, ecuación (5.29)

Corrección:

$$
\begin{aligned}
&HP_k^- H^T + R_k = \begin{bmatrix} R_p & 0_{3\times m-3} \\ 0_{m-3\times3} & H_v P_v H_v^T + R_v \end{bmatrix} = \begin{bmatrix} M_p & M_{pv} \\ M_{vp} & M_v \end{bmatrix} = \begin{bmatrix} M_p & 0 \\ 0 & M_v \end{bmatrix} \\
&\left(HP_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} M_p^{-1} & 0 \\ 0 & S_v \end{bmatrix} = \begin{bmatrix} M_p^{-1} & 0 \\ 0 & M_v^{-1} \end{bmatrix} = \begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix} \\
&K_k = P_k H^T \left(HP_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} 0 & P_{pv} H_v^T N_v \\ 0 & P_v H_v^T N_v \end{bmatrix} = \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \\
&\hat{x}_k = \hat{x}_k + K_k \left(z_k - H_k \hat{x}_k\right) = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_{pv}\Delta_{k,v} \\ K_v \Delta_{k,v} \end{bmatrix}, \Delta_{k,v} = z_{k,v} - H_v \hat{x}_{k,v} \\
&P_k = P_k - K_k H P_k = \begin{bmatrix} P_p - K_{pv} H_v P_{vp} & P_{pv} - K_{pv} H_v P_v \\ P_{vp} - K_v H_v P_{vp} & P_v - K_v H_v P_v \end{bmatrix}
\end{aligned}
\tag{5.30}
$$

Se observa del filtro local (ecuación (5.30)) que la medición de las velocidades $z_{k,v}$ actualiza e influye en todos los estados y covarianzas del modelo según K_k. Por último cabe destacar que se puede obtener el mismo resultado de la ecuación (5.30) al deducir el filtro local utilizando $z_k = z_{k,v}$ con $H_k = \begin{bmatrix} H_{vp} & H_v \end{bmatrix}$ repitiendo el proceso realizado para el EKF en componentes.

5.2.2.4 Filtro EKF global: postura

El último paso para determinar el filtro en cascada es la obtención del filtro global, el cual se deduce utilizando el filtro EKF en componentes de la ecuación (5.29) empleando únicamente los sensores globales (postura) para actualizar el estado y la covarianza. De la misma forma que en los casos anteriores, la etapa de predicción no cambia ya que se utiliza el mismo modelo (3.51). De esta forma, al considerar que no se realiza la actualización de la velocidad (sin medición local $z_{k,v}$) se remplaza la matriz H_v por una matriz nula $0_{m-3\times 3}$ indicando al filtro que no se disponen de los sensores locales. De esta forma, se obtiene la ecuación (5.31) con el filtro global.

EKF global para el modelo (3.51)

$\because H_{pv} = A_{vp} = 0_{3\times 3}, H_v = H_{vp} = R_{vp} = 0_{5\times 3}, R_{pv} = 0_{3\times 5}$

Predicción: igual que el EKF en componentes, ecuación (5.29)

Corrección:

$$
\begin{aligned}
&HP_k^- H^T + R_k = \begin{bmatrix} H_p P_p H_p^T + R_p & 0_{3\times m-3} \\ 0_{m-3\times 3} & R_v \end{bmatrix} = \begin{bmatrix} M_p & M_{pv} \\ M_{vp} & M_v \end{bmatrix} = \begin{bmatrix} M_p & 0 \\ 0 & M_v \end{bmatrix} \\
&\left(HP_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} M_p^{-1} & 0 \\ 0 & S_v \end{bmatrix} = \begin{bmatrix} M_p^{-1} & 0 \\ 0 & M_v^{-1} \end{bmatrix} = \begin{bmatrix} N_p & N_{pv} \\ N_{vp} & N_v \end{bmatrix} \\
&K_k = P_k H^T \left(HP_k^- H^T + R_k\right)^{-1} = \begin{bmatrix} P_p H_p^T N_p & 0 \\ P_{vp} H_p^T N_p & 0 \end{bmatrix} = \begin{bmatrix} K_p & K_{pv} \\ K_{vp} & K_v \end{bmatrix} \\
&\hat{x}_k = \hat{x}_k + K_k \left(z_k - H_k \hat{x}_k\right) = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p \Delta_{k,p} \\ K_{vp} \Delta_{k,p} \end{bmatrix}, \Delta_{k,p} = z_{k,p} - H_p \hat{x}_{k,p} \\
&P_k = P_k - K_k H P_k = \begin{bmatrix} P_p - K_p H_p P_p & P_{pv} - K_p H_p P_{pv} \\ P_{vp} - K_{vp} H_p P_p & P_v - K_{vp} H_p P_{pv} \end{bmatrix}
\end{aligned}
\tag{5.31}
$$

Se observa en el filtro local de la ecuación (5.31) que la medición de la postura $z_{k,p}$ actualiza e influye en todos los estados y covarianzas del modelo según el valor de la ganancia K_k.

5.2.2.5 *Filtro EKF en Cascada: predicción local, corrección global*

Utilizando el filtro local (5.30) y global (5.31) se plantea el filtro en cascada al utilizar el filtro local en sus etapas de predicción y corrección junto con la etapa de corrección del filtro global tal y como se muestra en la ecuación (5.32). El filtro en cascada se denomina como tal debido a que la salida del filtro local $(\hat{x}_k, P_k)$ actualizada con $z_{k,v}$ es utilizada como entrada del filtro global (conexión en cascada), cuya salida $(\hat{x}_k, P_k)$ es actualizada mediante $z_{k,p}$. Este planteamiento permite reducir el coste computacional, al calcular una matriz inversa 5×5 para el filtro local (solo utiliza M_v^{-1}) y una inversa de 3×3 para el filtro global (solo requiere M_p^{-1}) en lugar de una matriz 8×8 para el filtro completo (ecuación (5.32)).

Al comparar la etapa de corrección del filtro EKF en cascada (ecuación (5.32)) con la del filtro EKF de estado y medición completa con entradas asignadas (ecuación (5.29), en componentes) se observa que se realizan las mismas actualizaciones sobre el estado estimado $\hat{x}_k$ y covarianza P_k en ambos filtros. En el caso del filtro completo se realizan las actualizaciones en conjunto a través de cada término de la ganancia, por lo que en $\hat{x}_k$ se sumarán de forma simultánea los correspondientes términos de velocidad $(\Delta_{k,v})$ y postura $(\Delta_{k,p})$. De la misma forma en P_k se suman simultáneamente los términos locales (P_v, P_{vp}) y globales (P_p, P_{pv}). Por el contrario el filtro en cascada realizará primeramente las correcciones con los términos de velocidad (locales) y posteriormente con los términos de postura (globales) tanto en $\hat{x}_k$ como en P_k, siendo la corrección total la misma que la realizada para el filtro completo.

EKF en Cascada, modelo (3.51)

$\because A_{vp} = 0_{3\times3}, H_{vp} = R_{vp} = 0_{5\times3}, H_{pv} = 0_{3\times3}, R_{pv} = 0_{3\times5}$

Predicción Local:

$$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$$

$$A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{x_{k-1},u_{k-1}} = \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix}, \begin{array}{l} \Omega_p = (A_p P_p + A_{pv} P_{vp}), \Omega_v = (A_v P_v) \\ \Omega_{pv} = (A_p P_{pv} + A_{pv} P_v), \Omega_{vp} = (A_v P_{vp}) \end{array}$$

$$P_k = A_{k-1} P_{k-1} A_{k-1}^T + Q_{k-1} = \begin{bmatrix} \Omega_p A_p^T + \Omega_{pv} A_{pv}^T + Q_p & \Omega_{pv} A_v^T + Q_{pv} \\ \Omega_{vp} A_p^T + \Omega_v A_{pv}^T + Q_{vp} & \Omega_v A_v^T + Q_v \end{bmatrix}$$

Corrección Local: $\because H_p = 0_{3\times3}$

$$\begin{aligned} M_v^{-1} &= \left(H_v P_v H_v^T + R_v\right)^{-1} \\ K_k &= \begin{bmatrix} 0 & P_{pv} H_v^T N_v \\ 0 & P_v H_v^T N_v \end{bmatrix} = \begin{bmatrix} 0 & P_{pv} H_v^T M_v^{-1} \\ 0 & P_v H_v^T M_v^{-1} \end{bmatrix} = \begin{bmatrix} 0 & K_{pv} \\ 0 & K_v \end{bmatrix} \\ \hat{x}_k &= \hat{x}_k + K_k\left(z_k - H_k \hat{x}_k\right) = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_{pv}\Delta_{k,v} \\ K_v \Delta_{k,v} \end{bmatrix}, \Delta_{k,v} = z_{k,v} - H_v \hat{x}_{k,v} \\ P_k &= P_k - K_k H P_k = \begin{bmatrix} P_p - K_{pv} H_v P_{vp} & P_{pv} - K_{pv} H_v P_v \\ P_{vp} - K_v H_v P_{vp} & P_v - K_v H_v P_v \end{bmatrix} \end{aligned} \tag{5.32}$$

Corrección Global: $\because H_v = 0_{5\times3}$

$$\begin{aligned} M_p^{-1} &= \left(H_p P_p H_p^T + R_p\right)^{-1} \\ K_k &= \begin{bmatrix} P_p H_p^T N_p & 0 \\ P_{vp} H_p^T N_p & 0 \end{bmatrix} = \begin{bmatrix} P_p H_p^T M_p^{-1} & 0 \\ P_{vp} H_p^T M_p^{-1} & 0 \end{bmatrix} = \begin{bmatrix} K_p & 0 \\ K_{vp} & 0 \end{bmatrix} \\ \hat{x}_k &= \hat{x}_k + K_k\left(z_k - H_k \hat{x}_k\right) = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p \Delta_{k,p} \\ K_{vp}\Delta_{k,p} \end{bmatrix}, \Delta_{k,p} = z_{k,p} - H_p \hat{x}_{k,p} \\ P_k &= P_k - K_k H P_k = \begin{bmatrix} P_p - K_p H_p P_p & P_{pv} - K_p H_p P_{pv} \\ P_{vp} - K_{vp} H_p P_p & P_v - K_{vp} H_p P_{pv} \end{bmatrix} \end{aligned}$$

Sin embargo, se puede observar como los términos de las ganancias K_k del filtro completo (ecuación (5.29)) y del filtro en cascada (ecuación (5.32), suma de la ganancia local y global) son distintos. En particular, al utilizar la configuración en cascada no se toman en cuenta los factores de covarianza cruzada (términos dependientes de N_{vp}, N_{pv}, recuadro continuo en la ecuación (5.29)) en la obtención de cada componente de K_k, por lo que al no sumar estos términos adicionales, la K_k calculada en cascada podría ser distinta a la calculada para la medición completa.

Dependiendo de los valores de los términos de la inversa (en partícular para el caso de localización) se distinguen en la práctica dos casos principales, al poder obtener una ganancia del filtro en cascada de *menor* o *mayor* magnitud a la ganancia del filtro completo. En el caso de obtener una ganancia de menor magnitud (filtro en cascada) implicaría que el filtro estimará el estado $\hat{x}_k$ dando mayor importancia al modelo que a la medición, caso contrario a cuando la ganancia de magnitud alta (filtro completo) en el que se le da mayor peso al factor $\Delta_{k,p,v}$ (ecuación (5.29)) que corrige la predicción del modelo. Esto implica que ante el caso de un error de modelado, el filtro de ganancia mayor funcionará mejor que el de ganancia menor, al darle más importancia a la medición.

Debido a que calcular dos matrices inversas de orden reducido consume menos recursos que el cálculo de una matriz inversa de orden superior, el filtro en cascada es una opción atractiva para implementar la fusión sensorial de forma que se mejore la localización de las plataformas de recursos limitados. Para poder utilizar esta estrategia se debe limitar el impacto de la omisión de las covarianza cruzada en el desempeño. Esto se consigue al incrementar o disminuir la *magnitud* de la ganancia calculada por los filtros en cascada; lo cual, como se observa en la ecuación (5.29) para K_k, se puede conseguir al elevar o disminuir el valor asignado a los términos de Q_k (la magnitud de K_k es proporcional a la de Q_k). Por ejemplo, al incrementar los términos de Q_k aumentan también los términos de P_k que a su vez producirían el incremento deseado de la ganancia obtenida en cascada. Mediante este procedimiento, y con una elección adecuada de Q_k, el filtro en cascada producirá un resultado similar o idéntico al filtro de medición completa, requiriendo un menor coste computacional.

Cabe destacar que la magnitud del incremento o decremento de Q_k puede ajustarse durante la etapa de implementación del filtro según se requiera mejorar el desempeño del algoritmo. En el caso de tener una matriz Q_k diagonal se puede realizar un incremento o decremento de mayor magnitud con el fin de que los términos (P_p, P_v) compensen la falta de modificación de los términos (P_{vp}, P_{pv}) (ya que Q_{vp} y Q_{pv} son nulas en este caso) de forma que se obtenga una ganancia y desempeño similar al filtro de medición completa.

5.2.3 Filtro EKF en Cascada reducido: predicción local, corrección global de la postura

Una versión simplificada del filtro en cascada puede derivarse al considerar la actualización de la postura como un proceso sin influencia en la estimación de la velocidad, al no tener en el modelo una relación que determine la velocidad local a partir de la postura global. De esta forma, $\hat{x}_k$ se puede actualizar mediante la ecuación (5.33) utilizando la salida del filtro local junto con la actualización global de la postura.

$$\hat{x}_k = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_p\Delta_{k,p} + K_{pv}\Delta_{k,v} \\ K_v\Delta_{k,v} \end{bmatrix} \tag{5.33}$$

Este planteamiento puede suponer una ventaja computacional adicional al requerir menos operaciones que el filtro en cascada (ecuación (5.32)), ya que se actualizaría la postura mediante la información global $z_{k,p}$ al calcular K_p pero no se actualizaría la velocidad ya que no se calcularía K_{vp}. De esta forma se simplifican los respectivos términos en P_k al considerar K_{vp} como una matriz nula.

Como simplificación adicional se puede considerar que la actualización en la postura no tiene una influencia considerable sobre la covarianza cruzada P_{pv}, esto debido a que el error de estimación asociado a A_{pv} depende en gran medida a las mediciones locales (velocidad) y en menor medida de las mediciones globales. De esta forma, el aporte en la disminución de P_{pv} asociado a la actualización la postura podría considerarse poco significativo, con lo que se podría omitir el cálculo del término $K_pH_pP_{pv}$ en la ecuación

(5.32). Esto produce un filtro menos optimista al considerar una P_{pv} mayor al filtro en cascada. De esta forma la corrección completa (local y global) de la covarianza P_k del filtro en cascada simplificado se expresa mediante la ecuación (5.34). Las ecuaciones del filtro cascada reducido se obtienen al utilizar las ecuaciones (5.33) y (5.34) en la ecuación (5.32) (K_{vp} nula, $K_pH_pP_{pv}$ omitido), con lo que se obtiene la ecuación (5.35).

$$P_k = \begin{bmatrix} P_p - K_pH_pP_p - K_{pv}H_vP_{vp} & P_{pv} - K_{pv}H_vP_v \\ P_{vp} - K_vH_vP_{vp} & P_v - K_vH_vP_v \end{bmatrix} \tag{5.34}$$

EKF en Cascada reducido, modelo (3.51)

$\because A_{vp} = 0_{3\times3}, H_{vp} = R_{vp} = 0_{5\times3}, H_{pv} = 0_{3\times3}, R_{pv} = 0_{3\times5}$

Predicción Local:

$$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$$

$$A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{x_{k-1},u_{k-1}} = \begin{bmatrix} A_p & A_{pv} \\ A_{vp} & A_v \end{bmatrix}, \Omega_p = (A_pP_p + A_{pv}P_{vp}), \Omega_v = (A_vP_v), \Omega_{pv} = (A_pP_{pv} + A_{pv}P_v), \Omega_{vp} = (A_vP_{vp})$$

$$P_k = A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1} = \begin{bmatrix} \Omega_pA_p^T + \Omega_{pv}A_{pv}^T + Q_p & \Omega_{pv}A_v^T + Q_{pv} \\ \Omega_{vp}A_p^T + \Omega_vA_{pv}^T + Q_{vp} & \Omega_vA_v^T + Q_v \end{bmatrix}$$

Corrección Local: $\because H_p = 0_{3\times3}$

$$\begin{aligned} M_v^{-1} &= \left(H_vP_vH_v^T + R_v\right)^{-1} \\ K_{k,v} &= \begin{bmatrix} 0 & P_{pv}H_v^TN_v \\ 0 & P_vH_v^TN_v \end{bmatrix} = \begin{bmatrix} 0 & P_{pv}H_v^TM_v^{-1} \\ 0 & P_vH_v^TM_v^{-1} \end{bmatrix} = \begin{bmatrix} 0 & K_{pv} \\ 0 & K_v \end{bmatrix} \\ \hat{x}_k &= \hat{x}_k + K_k\left(z_k - H_k\hat{x}_k\right) = \begin{bmatrix} \hat{x}_{k,p} \\ \hat{x}_{k,v} \end{bmatrix} + \begin{bmatrix} K_{pv}\Delta_{k,v} \\ K_v\Delta_{k,v} \end{bmatrix}, \Delta_{k,v} = z_{k,v} - H_v\hat{x}_{k,v} \\ P_k &= P_k - K_kHP_k = \begin{bmatrix} P_p - K_{pv}H_vP_{vp} & P_{pv} - K_{pv}H_vP_v \\ P_{vp} - K_vH_vP_{vp} & P_v - K_vH_vP_v \end{bmatrix} \end{aligned} \tag{5.35}$$

Corrección Global: $\because H_v = 0_{5\times3}, K_{vp} = 0_{3\times3}$

$$K_{k,p} = P_pH_p^TN_p = P_pH_p^TM_p^{-1} = P_pH_p^T\left(H_pP_pH_p^T + R_p\right)^{-1}$$

$$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right)$$

$$P_{k,p} = P_{k,p} - K_{k,p}H_{k,p}P_{k,p}$$

Los filtros en cascada propuestos pueden ser utilizados con otros modelos no lineales de la postura del robot (por ejemplo para el robot diferencial, ecuación (3.24)) al ajustar las dimensiones de los componentes de las matrices del filtro así como los vectores x_k,u_k y z_k; siempre que se considere la separación local y global de z_k. El procedimiento general del filtro en cascada (ecuación (5.32)) y cascada reducido (ecuación (5.35)) se resume en los algoritmos 5 y 6 respectivamente, válidos si H_{pv} y H_{vp} son matrices nulas.

Algoritmo 5: Algoritmo EKF recursivo en cascada, asignación de entradas, predicción local y corrección global basada en el tiempo

Entrada: $u_{k-1} = [\ u_1 \quad u_2 \quad u_3\ ]^T$,$\hat{x}_{k-1}$,$P_{k-1}$

Medición: $z_k = [\ z_{k,p} \quad z_{k,v}\ ]^T$, $z_{k,v} = [\ v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ ]_k^T$

$z_{k,p} = L_{GM/GPS} = [\ x_k\ y_k\ \theta_k\ ]_{GM/GPS}^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(del modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,R_k

Salida: $\hat{x}_k$,P_k

Inicialización: $\hat{x}_0$,P_0

Para *el instante actual* k **hacer**

Predicción:

$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$

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

$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$

$H_k = \frac{\partial h}{\partial x}\Big|_{\hat{x}_k,0} = [H_p \ \ H_{pv}; H_{vp} \ \ H_v]_k = [H_p \ \ 0; 0 \ \ H_v]_k$, $N_k = \frac{\partial h}{\partial n}\Big|_{\hat{x}_k,0}$

Corrección Local: utilizar H_p nula para actualizar con $z_{k,v}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k \left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

Corrección Global: utilizar H_v nula para actualizar con $z_{k,p}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k \left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

fin

Algoritmo 6: Algoritmo EKF recursivo en cascada–reducido, asignación de entradas, predicción local y corrección global basada en el tiempo

Entrada: $u_{k-1} = [\ u_1 \quad u_2 \quad u_3\ ]^T$,$\hat{x}_{k-1}$,$P_{k-1}$

Medición: $z_k = [\ z_{k,p} \quad z_{k,v}\ ]^T,\ z_{k,v} = [\ v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ ]_k^T$

$z_{k,p} = L_{GM/GPS} = [\ x_k\ y_k\ \theta_k\ ]_{GM/GPS}^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(del modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,

$R_k = [R_p\ \ R_{pv}; R_{vp}\ \ R_v]_k$

Salida: $\hat{x}_k = [\ x_{k,p} \quad x_{k,v}\ ]^T$,$P_k = [P_p\ \ P_{pv}; P_{vp}\ \ P_v]_k$

Inicialización: $\hat{x}_0$,P_0

Para *el instante actual* k **hacer**

Predicción:

$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$

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

$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$

$H_k = \frac{\partial h}{\partial x}\Big|_{\hat{x}_k,0} = [H_p\ \ H_{pv}; H_{vp}\ \ H_v]_k = [H_p\ \ 0; 0\ \ H_v]_k$, $N_k = \frac{\partial h}{\partial n}\Big|_{\hat{x}_k,0}$

Corrección Local: utilizar H_p nula para actualizar con $z_{k,v}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k\left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

Corrección Global: actualizar únicamente $(\hat{x}_{k,p}, P_{k,p})$ con $z_{k,p}$

$K_{k,p} = P_{k,p} H_{k,p}^T \Big(H_{k,p} P_{k,p} H_{k,p}^T + R_{k,p}\Big)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right)$

$P_{k,p} = P_{k,p} - K_{k,p} H_{k,p} P_{k,p}$

fin

5.2.4 Filtro en Cascada con Modelos en Cascada: Fusión Local Estado Reducido - Modelo Postura - Corrección Global

Un esquema de fusión adicional puede establecerse a partir del desarrollo realizado para el filtro en cascada reducido, con el fin de disminuir aún más el costo computacional del filtro. Para esto se secciona el estado x_k de acuerdo a sus estados locales $x_{k,v}$ (velocidad) y globales $x_{k,p}$ (postura) de la misma forma que se secciona la medición en sensores de velocidad $z_{k,v}$ y postura global $z_{k,p}$ según se muestra en la ecuaciones (5.36) para el caso con deslizamiento y (5.37) sin deslizamiento. En estas ecuaciones se ha definido además los vectores de entrada $u_{k,Aks}$ para la plataforma Ackerman con deslizamiento, $u_{k,Akns}$ del Ackerman sin deslizamiento y $u_{k,dif}$ para el robot diferencial sin deslizamiento.

Con ésta asignación, el procedimiento a seguir es primeramente emplear el filtro local para realizar la fusión sensorial utilizando u_k, $x_{k,v}$ y $z_{k,v}$ junto con los modelos locales de velocidad (ecuaciones (3.50), (3.52) para el robot Ackerman y (3.23) para el robot diferencial). Posteriormente se utiliza la salida del filtro local como entrada del modelo cinemático global (ecuaciones (3.11) y (3.36) para el robot diferencial y Ackerman respectivamente) para obtener la postura global. Finalmente se actualiza la postura utilizando $z_{k,p}$ y la corrección global del filtro en cascada reducido de la ecuación (5.35).

La principal ventaja de este planteamiento es el número reducido de operaciones a efectuar (comparado con los filtros de las ecuaciones (5.32) y (5.35)) además de poder utilizar el filtro KF lineal cuando se utilizan los modelos sin deslizamiento, ya que son modelos lineales (ecuaciones (3.23) y (3.52)). En

estos casos se reduce aún más el costo computacional y de implementación al no requerirse la linealización del modelo.

$$\begin{aligned}
\boldsymbol{x}_{k,p} &= \left[\begin{array}{ccc} x & y & \theta \end{array}\right]_k^T, \quad \boldsymbol{x}_{k,v} = \left[\begin{array}{cc} v_x & v_y \quad \omega \end{array}\right]_k^T \\
\boldsymbol{u}_{k,Aks} &= \left[\begin{array}{ccc} u_1 & u_2 & u_3 \end{array}\right]_k^T \\
\boldsymbol{z}_{k,p} &= H_{k,p} x_{k,p} = \boldsymbol{L}_{GM/GPS} = \left[\begin{array}{ccc} x_k & y_k & \theta_k \end{array}\right]_{GM/GPS}^T \\
\boldsymbol{z}_{k,v} &= H_{k,v} x_{k,v} = \left[\begin{array}{ccccc} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{array}\right]_k^T
\end{aligned} \tag{5.36}$$

$$\begin{aligned}
\boldsymbol{x}_{k,p} &= \left[\begin{array}{ccc} x & y & \theta \end{array}\right]_k^T, \quad \boldsymbol{x}_{k,v} = \left[\begin{array}{cc} v_x & \omega \end{array}\right]_k^T \\
\boldsymbol{u}_{k,Akns} &= \left[\begin{array}{cc} u_1 & u_3 \end{array}\right]_k^T, \boldsymbol{u}_{k,dif} = \left[\begin{array}{cc} a & \alpha \end{array}\right]_k^T \\
\boldsymbol{z}_{k,p} &= H_{k,p} x_{k,p} = \boldsymbol{L}_{GM/GPS} = \left[\begin{array}{ccc} x_k & y_k & \theta_k \end{array}\right]_{GM/GPS}^T \\
\boldsymbol{z}_{k,v} &= H_{k,v} x_{k,v} = \left[\begin{array}{cccc} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{array}\right]_k^T
\end{aligned} \tag{5.37}$$

El planteamiento de este filtro se puede realizar a partir del filtro en cascada reducido (5.35) al considerar las mismas definiciones para Q_k (ecuación (5.15)), H_k (ecuación (5.18)) y R_k (ecuación (5.19)) según se describe a continuación para el caso con deslizamiento de la ecuación (5.36). El filtro local de velocidad se obtiene al sustituir $A_p = A_{pv} = A_{vp} = 0_{3\times 3}$ en las etapas de predicción y corrección local y al considerar un único término Q_v según se muestra en la ecuación (5.38). Esto debido a que solamente se utiliza el modelo local no lineal (ecuación (3.50)) para realizar la fusión de velocidad con lo que solo se requiere el cálculo de A_v (cuyo valor es el mismo al obtenido en las ecuaciones (5.12) y (5.13)).

Se aprecia al comparar este resultado con la ecuación (5.35) que no se consideran las covarianzas cruzadas (P_{pv}, P_{vp}) al no realizar la fusión con el modelo de estado completo incluyendo los términos de postura. En el caso de los términos de velocidad se observa cómo se obtienen los mismos resul-

tados en la predicción y corrección al obtener el mismo estado y covarianza en ambas ecuaciones.

$$
\begin{aligned}
&\because A_{vp}=A_{pv}=A_p=0_{3\times3}, H_{vp}=R_{vp}=0_{5\times3}, H_{pv}=0_{3\times3}, R_{pv}=0_{3\times5}\\
&\text{Predicción Local:}\\
&\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)\\
&A_{k-1}=\left.\frac{\partial f}{\partial x}\right|_{x_{k-1},u_{k-1}}=\begin{bmatrix}0 & 0\\ 0 & A_v\end{bmatrix}, A_v=\begin{bmatrix}1 & T_s\omega & T_s v_y\\ -T_s\omega & 1 & -T_s v_x\\ 0 & 0 & 1\end{bmatrix}_{\substack{x_{k-1}\\ u_{k-1}}}\\
&\quad \Omega_p=0 \quad \Omega_v=(A_v P_v) \quad \Omega_{pv}=0 \quad \Omega_{vp}=(A_v P_{vp})\\
&P_k=A_{k-1}P_{k-1}A_{k-1}^T+Q_{k-1}=\begin{bmatrix}0 & 0\\ 0 & \Omega_v A_v^T+Q_v\end{bmatrix}=\begin{bmatrix}P_p & P_{pv}\\ P_{vp} & P_v\end{bmatrix}\\
&\text{Corrección Local: } \because H_p=0_{3\times3}\\
&\quad M_v^{-1}=\left(H_v P_v H_v^T+R_v\right)^{-1}\\
&K_{k,v}=\begin{bmatrix}0 & P_{pv}H_v^T M_v^{-1}\\ 0 & P_v H_v^T M_v^{-1}\end{bmatrix}=\begin{bmatrix}0 & 0\\ 0 & K_v\end{bmatrix}\\
&\hat{x}_{k,v}=\hat{x}_{k,v}+K_k\left(z_{k,v}-H_{k,v}\hat{x}_{k,v}\right)\\
&P_{k,v}=P_{k,v}-K_{k,v}H_{k,v}P_{k,v}
\end{aligned}
\tag{5.38}
$$

Con la estimación de la velocidad $x_{k,v}$ de la ecuación (5.38) en la que se han incorporado las mediciones de los sensores locales, se procede a utilizar el modelo cinemático global (3.36) para obtener la postura $\hat{x}_{k,p}=f_p\left(\hat{x}_{k,p},\hat{x}_{k,v}\right)$ al utilizar $x_{k,v}$ como entrada, según se muestra en la ecuación (5.39).

$$
\begin{aligned}
x_{k,p}&=\begin{bmatrix} x & y & \theta \end{bmatrix}_k^T, \quad x_{k,v}=\begin{bmatrix} v_x & v_y & \omega \end{bmatrix}_k^T\\
\hat{x}_{k,p}&=\hat{x}_{k-1,p}+T_s\begin{bmatrix}\cos\theta_{k-1} & -\sin\theta_{k-1} & 0\\ \sin\theta_{k-1} & \cos\theta_{k-1} & 0\\ 0 & 0 & 1\end{bmatrix}\hat{x}_{k,v}=f_p\left(\hat{x}_{k-1,p},\hat{x}_{k,v}\right)
\end{aligned}
\tag{5.39}
$$

Hasta este punto se ha realizado la localización del robot utilizando únicamente la información local, lo cual ha requerido un menor número de operaciones que el filtro en cascada, aunque se ha utilizado una inversión de las mismas dimensiones (5×5). Para actualizar la postura utilizando la información global $z_{k,p}$ se utiliza la etapa de corrección global del filtro en cascada reducido (5.35), la cual requiere el valor de $P_{k,p}$ para el cálculo de la ganancia del filtro. Sin embargo, a diferencia del filtro cascada reducido, el término $P_{k,p}$ no es calculado por el filtro local de la ecuación (5.38) que solo obtiene $P_{k,v}$.

Para determinar $P_{k,p}$ se propone utilizar su valor en el instante anterior junto con la propagación de la covarianza $P_{k,v}$ a través del modelo cinemático global que relaciona la velocidad con la postura. Para esto primeramente se linealiza el modelo global respecto a $(\hat{x}_{k,p}, \hat{x}_{k,v})$ tal y como se muestra en la ecuación (5.40), con lo que se obtiene el mismo resultado que para el A_p y A_{pv} del filtro en cascada resumido (ecuación (5.38)) al ser el modelo dinámico global (3.51) deducido mediante el cinemático global (3.36).

$$\begin{aligned} \mathrm{cs} = T_s \cos\theta, \mathrm{sn} = T_s \sin\theta \\ A_{k,p} = \left.\frac{\partial f}{\partial x_p}\right|_{\substack{x_{k-1} \\ u_{k-1}}} = \left[\begin{array}{ccc} 1 & 0 & -v_x\,\mathrm{sn} - v_y\,\mathrm{cs} \\ 0 & 1 & v_x\,\mathrm{cs} - v_y\,\mathrm{sn} \\ 0 & 0 & 1 \end{array}\right]_{\substack{x_{k-1} \\ u_{k-1}}} \\ A_{k,pv} = \left.\frac{\partial f}{\partial x_v}\right|_{\substack{x_{k-1} \\ u_{k-1}}} = \left[\begin{array}{ccc} \mathrm{cs} & -\mathrm{sn} & 0 \\ \mathrm{sn} & \mathrm{cs} & 0 \\ 0 & 0 & T_s \end{array}\right]_{\substack{x_{k-1} \\ u_{k-1}}} \end{aligned} \tag{5.40}$$

A partir de la ecuación de predicción de la covarianza del error P_k del filtro en cascada reducido (ecuación (5.35)) se observa que la covarianza del error de estimación de la postura P_p se puede determinar mediante la ecuación (5.41).

$$\begin{aligned} \Omega_p = (A_p P_p + A_{pv} P_{vp}) \, , \Omega_{pv} = (A_p P_{pv} + A_{pv} P_v) \, , \\ P_p = \Omega_p A_p^T + \Omega_{pv} A_{pv}^T + Q_p \\ \Rightarrow P_p = (A_p P_p + A_{pv} P_{vp}) \, A_p^T + (A_p P_{pv} + A_{pv} P_v) \, A_{pv}^T + Q_p \\ \Rightarrow P_p = A_p P_p A_p^T + A_{pv} P_{vp} A_p^T + A_p P_{pv} A_{pv}^T + A_{pv} P_v A_{pv}^T + Q_p \end{aligned} \tag{5.41}$$

El filtro local de la ecuación (5.38) no realiza la estimación de los términos (P_{pv}, P_{vp}), los cuales no se requieren en el cálculo de $x_{k,p}$ realizado en la ecuación (5.39). De esta forma, al no disponer de estos términos para el cálculo de P_p se debe incorporar su influencia mediante el término Q_p, incrementando su magnitud para obtener un valor de P_p cercano al obtenido mediante la ecuación (5.41). De esta forma al eliminar los términos (P_{pv}, P_{vp}) se simplifica la ecuación (5.41) en (5.42) con la que se obtiene la P_p requerida. Se observa que P_p depende de la propagación de la covarianza del instante anterior al actual mediante A_p, de la propagación de la covarianza asociada a la velocidad P_v mediante A_{pv} y la matriz de covarianza del ruido del proceso (postura) Q_p. Como aproximación adicional se puede omitir la propagación de $P_{k-1,p}$ en A_p, al considerar que ésta se aproxima mediante el valor de $P_{k-1,p}$, evitando el cálculo de A_p con lo que se requieren menos operaciones para implementar el filtro. Esta aproximación supone un mayor aporte de P_v en P_p y su precisión puede mejorarse con el respectivo ajuste de valor de Q_p.

$$\begin{aligned} P_{k,p} &= A_{k,p} P_{k-1,p} A_{k,p}^T + A_{k,pv} P_{k,v} A_{k,pv}^T + Q_{k,p} \\ P_{k,p} &\approx P_{k-1,p} + A_{k,pv} P_{k,v} A_{k,pv}^T + Q_{k,p} \end{aligned} \tag{5.42}$$

Obtenido el valor de P_p se procede a actualizar la postura utilizando la información global $z_{k,p}$ mediante la etapa de corrección global del filtro en cascada reducido (5.35), tal y como se muestra en la ecuación (5.43), en donde H_p es la matriz identidad 3×3 (según su definición en las ecuaciones (5.18) y (5.36)).

$$
\begin{aligned}
K_{k,p} &= P_p H_p^T \left(H_p P_p H_p^T + R_p \right)^{-1} \\
\hat{x}_{k,p} &= \hat{x}_{k,p} + K_{k,p} \left(z_{k,p} - H_{k,p} \hat{x}_{k,p} \right) \\
P_{k,p} &= P_{k,p} - K_{k,p} H_{k,p} P_{k,p}
\end{aligned}
\tag{5.43}
$$

Con este resultado se ha obtenido la postura actualizada mediante las etapas de predicción y corrección local (velocidad, ecuación (5.38)), modelo de la postura en cascada (salida del modelo de velocidad utilizado como entrada al modelo de postura) y su covarianza del error (ecuaciones (5.39), (5.40) y (5.42)) y actualización global (ecuación (5.43)), procedimiento que se resume en la ecuación (5.44).

Cabe destacar que al utilizar el filtro de fusión de la ecuación (5.44) con los modelos local y global en cascada, se está considerando una aproximación al problema con el fin de reducir el número de operaciones requeridas, debido a las simplificaciones realizadas al no utilizar las covarianzas cruzadas entre el proceso local (velocidades) y el global (postura) para determinar x_k y P_k, a diferencia del filtro EKF completo (ecuación (5.29)) y el en cascada (ecuación (5.32)). A pesar las simplificaciones realizadas se puede obtener un desempeño y precisión adecuadas para el caso de localización con un ajuste adecuado de (Q_p, Q_v).

Conviene señalar además que, aunque el método de modelos en cascada (5.44) realiza un menor número de operaciones, requiere el cálculo de dos matrices inversas de las mismas dimensiones que para el filtro en cascada (5.32) y cascada reducido (5.35), por lo que en el caso de modelos de velocidad con deslizamiento (ecuación (3.50), no lineal) puede no suponer una ventaja considerable que justifique su utilización. Sin embargo, al considerar el caso de los modelos de velocidad sin deslizamiento (ecuación (3.23) para el robot diferencial y (3.52) para el robot Ackerman) se consigue con el método de modelos en cascada una ventaja adicional, ya que al ser modelos lineales se puede utilizar el filtro de Kalman lineal para implementar el filtro local. De esta forma no se realiza la linealización del modelo para la fusión local, requiriendo menos recursos para su implementación. De esta forma, se obtiene la variante lineal del filtro de la ecuación (5.44) al sustituir el EKF de la

etapa de predicción y corrección local por un KF, tal y como se muestra en la ecuación (5.45).

EKF en Cascada con Modelo local (3.51)/**global**(3.36) **en cascada**

Predicción Local:

$$\hat{x}_{k,v} = f_v\left(\hat{x}_{k-1,v}, u_{k-1}, 0\right)$$

$$A_{k-1,v} = \left.\frac{\partial f_v}{\partial x}\right|_{x_{k-1},u_{k-1}} = \left[\begin{array}{ccc} 1 & T_s\omega & T_s v_y \\ -T_s\omega & 1 & -T_s v_x \\ 0 & 0 & 1 \end{array}\right]_{\substack{x_{k-1} \\ u_{k-1}}}$$

$$P_{k,v} = A_{k-1,v} P_{k-1,v} A_{k-1,v}^T + Q_{k-1,v}$$

Corrección Local:

$$K_{k,v} = P_{k,v} H_{k,v}^T \left(H_{k,v} P_{k,v} H_{k,v}^T + R_{k,v}\right)^{-1}$$

$$\hat{x}_{k,v} = \hat{x}_{k,v} + K_{k,v}\left(z_{k,v} - H_{k,v}\hat{x}_{k,v}\right)$$

$$P_{k,v} = P_{k,v} - K_{k,v} H_{k,v} P_{k,v}$$

Modelo Global, obtener postura $\hat{x}_{k,p} = f_p\left(\hat{x}_{k,p}, \hat{x}_{k,v}\right)$:

$$\hat{x}_{k,p} = \hat{x}_{k-1,p} + T_s \left[\begin{array}{ccc} \cos\theta_{k-1} & -\sin\theta_{k-1} & 0 \\ \sin\theta_{k-1} & \cos\theta_{k-1} & 0 \\ 0 & 0 & 1 \end{array}\right] \hat{x}_{k,v} = f_p\left(\hat{x}_{k-1,p}, \hat{x}_{k,v}\right)$$

Covarianza P_p: propagar P_v a través del modelo global:

$\mathrm{cs} = T_s \cos\theta, \mathrm{sn} = T_s \sin\theta$

$$A_{k,p} = \left.\frac{\partial f}{\partial x_p}\right|_{\substack{x_{k-1} \\ u_{k-1}}} = \left[\begin{array}{ccc} 1 & 0 & -v_x\,\mathrm{sn} - v_y\,\mathrm{cs} \\ 0 & 1 & v_x\,\mathrm{cs} - v_y\,\mathrm{sn} \\ 0 & 0 & 1 \end{array}\right]_{\substack{x_{k-1} \\ u_{k-1}}}, A_{k,pv} = \left.\frac{\partial f}{\partial x_v}\right|_{\substack{x_{k-1} \\ u_{k-1}}} = \left[\begin{array}{ccc} \mathrm{cs} & -\mathrm{sn} & 0 \\ \mathrm{sn} & \mathrm{cs} & 0 \\ 0 & 0 & T_s \end{array}\right]_{\substack{x_{k-1} \\ u_{k-1}}}$$

$$P_{k,p} = A_{k,p} P_{k-1,p} A_{k,p}^T + A_{k,pv} P_{k,v} A_{k,pv}^T + Q_{k,p} \approx P_{k-1,p} + A_{k,pv} P_{k,v} A_{k,pv}^T + Q_{k,p}$$

Corrección Global:

$$\begin{aligned} K_{k,p} &= P_p H_p^T \left(H_p P_p H_p^T + R_p\right)^{-1} \\ \hat{x}_{k,p} &= \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right) \\ P_{k,p} &= P_{k,p} - K_{k,p} H_{k,p} P_{k,p} \end{aligned} \tag{5.44}$$

KF en Cascada con Modelos local/global en cascada

Predicción Local:

$$\hat{x}_{k,v} = A_{k-1,v}\hat{x}_{k-1,v} + B_{k-1,v}u_{k-1}$$

$$P_{k,v}{=}A_{k-1,v}P_{k-1,v}A_{k-1,v}^T{+}Q_{k-1,v}$$

Corrección Local:

$$K_{k,v}{=}\ P_{k,v}H_{k,v}^T\Big(H_{k,v}P_{k,v}H_{k,v}^T + R_{k,v}\Big)^{-1}$$

$$\hat{x}_{k,v}{=}\hat{x}_{k,v}{+}K_{k,v}\left(z_{k,v}{-}H_{k,v}\hat{x}_{k,v}\right)$$

$$P_{k,v} = P_{k,v} - K_{k,v}H_{k,v}P_{k,v}$$

Modelo Global, obtener postura $\hat{x}_{k,p}$:

$$\hat{x}_{k,p}{=}\ f_p\left(\hat{x}_{k-1,p}, \hat{x}_{k,v}\right) \tag{5.45}$$

Covarianza P_p: propagar P_v a través del modelo global:

$$A_{k,p}{=}\frac{\partial f}{\partial x_p}\bigg|_{\substack{x_{k-1}\\u_{k-1}}},\ A_{k,pv}{=}\frac{\partial f}{\partial x_v}\bigg|_{\substack{x_{k-1}\\u_{k-1}}}$$

$$\begin{aligned}P_{k,p}&{=}A_{k,p}P_{k-1,p}A_{k,p}^T{+}A_{k,pv}P_{k,v}A_{k,pv}^T{+}Q_{k,p}\\&\approx P_{k-1,p}{+}A_{k,pv}P_{k,v}A_{k,pv}^T{+}Q_{k,p}\end{aligned}$$

Corrección Global:

$$K_{k,p} = P_pH_p^T\Big(H_pP_pH_p^T + R_p\Big)^{-1}$$

$$\hat{x}_{k,p} = \hat{x}_{k,p}{+}K_{k,p}\left(z_{k,p}{-}H_{k,p}\hat{x}_{k,p}\right)$$

$$P_{k,p} = P_{k,p} - K_{k,p}H_{k,p}P_{k,p}$$

Se resume el procedimiento general de los algoritmos EKF y KF con modelos en cascada en los algoritmos 7 y 8 respectivamente, válidos si H_{pv} y H_{vp} son matrices nulas.

Obtenidos los filtros de fusión con actualización basada en el tiempo y las variantes propuestas con un uso reducido de recursos computacionales se procede a exponer los métodos basados en eventos en el capítulo siguiente, los cuales buscan una reducción mayor en el uso de recursos en la estimación de la postura del robot y son desarrollados a partir de los métodos en cascada previamente descritos.

Algoritmo 7: Algoritmo EKF recursivo en cascada con modelos local/global en cascada, asignación de entradas, predicción local de velocidades, modelo global de postura y corrección global basada en el tiempo

Entrada: $u_{k-1} = u_{Aks} = [u_1 \ u_2 \ u_3]^T$, $\hat{x}_{k-1,v}$,$P_{k-1,v}$, $\hat{x}_{k-1,p}$,$P_{k-1,p}$

Medición: $z_{k,v} = [\ v_{x,enc} \ v_{y,enc} \ \omega_{enc} \ \omega_{gyr} \ \omega_{comp}\ ]_k^T$,
$z_{k,p} = [\ x_k \ y_k \ \theta_k\ ]_{GM/GPS}^T$

Datos: $f_v(\cdot)$ *y* $h_v(\cdot)$ *del modelo no lineal local* (3.50),$Q_{k,v}$,$R_{k,v}$,
$f_p(\cdot)$ *y* $h_p(\cdot)$ *del modelo no lineal global* (3.36),$Q_{k,p}$,$R_{k,p}$

Salida: $\hat{x}_{k,v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}_k^T$,$\hat{x}_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T$,$P_{k,p}$, $P_{k,v}$

Inicialización: $\hat{x}_{0,v}$,$P_{0,v}$,$\hat{x}_{0,p}$,$P_{0,p}$

Para *el instante actual* k **hacer**

Predicción Local:

$\hat{x}_{k,v} = f_v\left(\hat{x}_{k-1,v}, u_{k-1}, 0\right)$

$A_{k-1,v} = \frac{\partial f_v}{\partial x_v}\Big|_{\substack{x_{k-1} \\ u_{k-1}}}$, $W_{k-1,v} = \frac{\partial f_v}{\partial w_v}\Big|_{\hat{x}_{k-1,v},u_{k-1},0}$

$P_{k,v} = A_{k-1,v}P_{k-1,v}A_{k-1,v}^T + W_{k-1,v}Q_{k-1,v}W_{k-1,v}^T$

$H_{k,v} = \frac{\partial h_v}{\partial x_v}\Big|_{\hat{x}_k,0}$, $N_{k,v} = \frac{\partial h_v}{\partial n_v}\Big|_{\hat{x}_k,0}$

Corrección Local:

$K_{k,v} = P_{k,v}H_{k,v}^T\Big(H_{k,v}P_{k,v}H_{k,v}^T + N_{k,v}R_{k,v}N_{k,v}^T\Big)^{-1}$

$\hat{x}_{k,v} = \hat{x}_{k,v} + K_{k,v}\left[z_{k,v} - h_{k,v}\left(\hat{x}_{k,v}, 0\right)\right]$

$P_{k,v} = \left(I - K_{k,v}H_{k,v}\right)P_{k,v}$

Modelo Global, Covarianza Global:

$\hat{x}_{k,p} = f_p\left(\hat{x}_{k-1,p}, \hat{x}_{k,v}\right)$

$A_{k,p} = \frac{\partial f_p}{\partial x_p}\Big|_{\substack{x_{k-1} \\ u_{k-1}}}$,$A_{k,pv} = \frac{\partial f_p}{\partial x_v}\Big|_{\substack{x_{k-1} \\ u_{k-1}}}$

$P_{k,p} = A_{k,p}P_{k-1,p}A_{k,p}^T + A_{k,pv}P_{k,v}A_{k,pv}^T + Q_{k,p}$

Corrección Global: $H_{k,p} = \frac{\partial h_p}{\partial x_p}\Big|_{\hat{x}_k,0}$=Identidad 3×3 según ecuación (5.36)

$K_{k,p} = P_{k,p}H_{k,p}^T\Big(H_{k,p}P_{k,p}H_{k,p}^T + R_{k,p}\Big)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right)$

$P_{k,p} = P_{k,p} - K_{k,p}H_{k,p}P_{k,p}$

fin

Algoritmo 8: Algoritmo KF recursivo en cascada con modelos local/global en cascada, asignación de entradas, predicción local de velocidades, modelo global de postura y corrección global basada en el tiempo

Entrada: u_{k-1}=u_{Akns}=$[u_1\ u_3]^T$=u_{dif} =$[a\ \alpha]^T$,$\hat{x}_{k-1,v}$,$P_{k-1,v}$,$\hat{x}_{k-1,p}$,$P_{k-1,p}$

Medición: $z_{k,v} = [\ v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ ]_k^T$,

$z_{k,p} = [\ x_k\ y_k\ \theta_k\ ]_{GM/GPS}^T$

Datos: A_k *y* B_k *del modelo lineal local* (3.23) *(o* (3.52)*)*,$Q_{k,v}$,$R_{k,v}$,

$f_p(\cdot)$ *y* $h_p(\cdot)$ *del modelo no lineal global* (3.11) *(o* (3.12)*)*,$Q_{k,p}$,$R_{k,p}$

Salida: $\hat{x}_{k,v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}_k^T$,$\hat{x}_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T$,$P_{k,p}$, $P_{k,v}$

Inicialización: $\hat{x}_{0,v}$,$P_{0,v}$,$\hat{x}_{0,p}$,$P_{0,p}$

Para *el instante actual* k **hacer**

Predicción Local:

$\hat{x}_{k,v} = A_{k-1,v}\hat{x}_{k-1,v} + B_{k-1,v}u_{k-1}$

$P_{k,v} = A_{k-1,v}P_{k-1,v}A_{k-1,v}^T + W_{k-1,v}Q_{k-1,v}W_{k-1,v}^T$

Corrección Local:

$K_{k,v} = P_{k,v}H_{k,v}^T\Big(H_{k,v}P_{k,v}H_{k,v}^T + N_{k,v}R_{k,v}N_{k,v}^T\Big)^{-1}$

$\hat{x}_{k,v} = \hat{x}_{k,v} + K_{k,v}\left(z_{k,v} - H_{k,v}\hat{x}_{k,v}\right)$

$P_{k,v} = \left(I - K_{k,v}H_{k,v}\right)P_{k,v}$

Modelo Global, Covarianza Global:

$\hat{x}_{k,p} = f_p\left(\hat{x}_{k-1,p}, \hat{x}_{k,v}\right)$

$A_{k,p} = \frac{\partial f_p}{\partial x_p}\Big|_{\substack{x_{k-1}\\u_{k-1}}}$, $A_{k,pv} = \frac{\partial f_p}{\partial x_v}\Big|_{\substack{x_{k-1}\\u_{k-1}}}$

$P_{k,p} = A_{k,p}P_{k-1,p}A_{k,p}^T + A_{k,pv}P_{k,v}A_{k,pv}^T + Q_{k,p}$

$\approx P_{k-1,p} + A_{k,pv}P_{k,v}A_{k,pv}^T + Q_{k,p}$

Corrección Global: $H_{k,p} = \frac{\partial h_p}{\partial x_p}\Big|_{\hat{x}_k,0}$=Identidad 3×3 según ecuación (5.37)

$K_{k,p} = P_{k,p}H_{k,p}^T\Big(H_{k,p}P_{k,p}H_{k,p}^T + R_{k,p}\Big)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right)$

$P_{k,p} = P_{k,p} - K_{k,p}H_{k,p}P_{k,p}$

fin

5.3 Conclusiones del Capítulo

En el presente capítulo se realizó la propuesta de distintos filtros de fusión con el fin de mejorar la estimación de la postura de un robot de recursos limitados. Se establecieron los algoritmos basados en el *tiempo*, al realizar la corrección del filtro con un tiempo de muestreo regular (T_s constante):

- Se describieron primeramente los métodos «tradicionales» (Algoritmos 1 y 2), de los cuales se observó que tienen un coste computacional muy elevado en el caso de fusión sensorial, ya que requieren realizar la inversión y/o raíz cuadrada matricial de dimensiones altas para obtener la ganancia del filtro, lo que impide la implementación directa de este tipo de algoritmos en sistemas de recursos limitados y deben ser modificados para este fin. Además, suponen que no hay problemas de disponibilidad en la medición de los sensores, considerándola disponible en cada instante de muestreo.

- Como primera modificación se realizó la asignación de las aceleraciones como entradas al modelo utilizado en la fusión sensorial (Algoritmos 3 y 4), con lo que se reducen las dimensiones de la matriz a invertir, pero no lo suficiente como para permitir su implementación en plataformas de recursos limitados.

- Como segunda modificación se realizó la división del vector de medición z_k en las mediciones globales $z_{k,p}$ (postura desde un GPS, cámara, etc.) y las mediciones locales $z_{k,v}$ (velocidades lineal y angular, encoders, giróscopo y brújula). Con esta división de z_k se realizó el desarrollo del filtro EKF en componentes locales y globales en sus etapas de predicción y corrección, tanto en su versión general (ecuación (5.28)) como reducida (ecuación (5.29)).

- Empleando el EKF en componentes se obtuvo el filtro EKF en cascada (Algoritmo 5), el cual realiza la fusión sensorial para obtener la postura en dos etapas: la fusión local empleando $z_{k,v}$ para obtener la estimación local del estado, y la fusión global en cascada, utilizando $z_{k,p}$ para corregir la estimación local. Este filtro tiene un menor coste computacional al realizar dos inversiones matriciales (local y global) de dimensiones menores a cuando se utiliza el vector de medición completo. Además, este filtro obtiene un resultado similar al de medición completa, al realizar un ajuste adecuado de la matriz covarianza del ruido del proceso Q_k, pero con un menor coste computacional.

- Con el filtro EKF en cascada se deduce el EKF en cascada reducido (Algoritmo 6), en el cual el filtro global únicamente actualiza la postura mediante $z_{k,p}$, con lo que se reduce el número de operaciones matriciales requeridas en la implementación del filtro.

- Se utilizó el filtro cascada reducido para obtener el filtro con modelos en cascada (algoritmos 7 y 8), el cual resulta de utilidad principalmente en el caso de los modelos sin deslizamiento, ya que se puede realizar la predicción mediante un KF (algoritmo 8) en lugar del EKF (algoritmo 7), reduciendo aún más las operaciones requeridas en la implementación por lo que es más adecuado para plataformas de recursos limitados.

- Finalmente cabe destacar que en ambos filtros en cascada (cascada reducido y modelos en cascada) Q_k puede ser ajustada para mejorar el desempeño del filtro y obtener resultados similares al filtro en cascada o al de medición completa, pero requiriendo menos recursos en su implementación.

6 | Fusión Sensorial por Eventos en Localización

En el presente capítulo se expone uno de los aportes principales de la presente tesis, el cual corresponde al desarrollo de los distintos algoritmos de fusión sensorial basada en *eventos* para la localización de un robot móvil de recursos limitados. Se obtienen los algoritmos al utilizar como punto de partida los filtros basados en el tiempo del capítulo 5 y los principios del control y muestreo basado en eventos, de forma que se define la etapa de corrección del filtro en términos de la ocurrencia de un evento durante la ejecución del algoritmo, esto con el fin de reducir la utilización de los recursos computacionales y el ancho de banda de la plataforma al incorporar la información de sensores globales. Estos algoritmos se extenderán al caso de localización cooperativa en el capítulo siguiente.

Nuevamente se omiten los superíndices a priori "$^-$" para simplificar la notación en el capítulo, con lo que las ecuaciones de corrección de los filtros se escriben utilizando la convención algorítmica, en la que la variable a la izquierda de la igualdad es actualizada a partir de su valor previamente existente en la parte derecha de la ecuación.

Se describe a continuación la incorporación de los métodos basados en eventos para adaptar los filtros de fusión expuestos anteriormente ante distintas condiciones de disponibilidad de los sensores y retardos de comunicación, así como para reducir la utilización del ancho de banda de la plataforma de recursos limitados. Para lo cual, se exponen primeramente los problemas asociados al muestreo de sensores que son resueltos al emplear las estrategias basadas en eventos, posteriormente se procede a definir la metodología de fusión basada en eventos especificando distintos tipos de eventos válidos para el caso de localización, y finalmente se realiza la modificación de los algoritmos previamente definidos con la estrategia basada en eventos.

6.1 Problemas relativos a la medición

Los esquemas de fusión con actualización basada en el tiempo presentados en la sección anterior suponen que la medida de todos los sensores está disponible en cada instante k, de forma que si alguna medición no puede realizarse o es obtenida con un retardo no se tendrá un funcionamiento correcto de estos algoritmos.

En el caso de la localización de plataformas móviles de recursos limitados se pueden presentar ambos problemas, ya que dependiendo de la frecuencia de medición de cada sensor y del medio de comunicación utilizado, se pueden obtener medidas con retraso que pertenecen a instantes anteriores y que no pueden incorporarse mediante los métodos previamente definidos; o bien, no recibir la medida del todo debido a errores de medición o por la no disponibilidad del sensor.

Por ejemplo, si el robot debe comunicarse con un sensor externo (como una cámara cenital o un robot vecino) para obtener las mediciones requeridas en la fusión, la frecuencia de muestreo (medición) estará sujeta al ancho de banda disponible para comunicarse con dicho sensor, además de existir un retardo de comunicación inherente al medio utilizado (por ejemplo, bluetooth, Wi-Fi, etc.). Otro ejemplo es el uso de sensores avanzados (GPS, sensor de barrido láser, cámaras y cámaras 3D) los cuales requieren un tiempo considerable de cómputo para obtener la medición. Por esta razón la frecuencia de estos sensores será menor a los sensores restantes de la plataforma y no pueden ser incorporados en todo instante k en el que estén disponibles los sensores locales de alta frecuencia (encoders, giróscopos, etc.). Además, en algunas situaciones se pueden presentar errores de disponibilidad como por ejemplo un número de satélites reducido o nulo en el campo de visión de un sensor GPS, o bien un cambio en la luminosidad del ambiente detectado con una cámara, etc.

En consecuencia, a la hora de realizar la implementación de los algoritmos sobre plataformas de recursos limitados se debe tomar en cuenta, además de la complejidad computacional del filtro de fusión, la frecuencia (disponibilidad) de las mediciones, el ancho de banda disponible y los retardos de

medición debidos a la comunicación (con sensores o robots cercanos). Además, en el caso específico de robots móviles, se tiene el problema adicional de ser plataformas que funcionan mediante baterías, por lo que si las frecuencias de medición y fusión son altas se producirá un consumo mayor de energía, disminuyendo el tiempo de operación del robot.

Existen soluciones parciales desarrolladas para incorporar la solución de algunos de estos aspectos dentro de los algoritmos de fusión. Por ejemplo, si solo existe un retardo conocido en alguna de las mediciones, esta puede incorporarse al retroceder la estimación actual al instante en donde se realizó la medición, realizar la fusión y luego volver a propagar en el tiempo al realizar la fusión para todos los instantes, desde el de medición hasta el instante actual [18, 151]. Sin embargo este método requiere gran cantidad de memoria en la plataforma para almacenar todos los estados y mediciones intermedias (entre el instante de medición y el actual), además de requerir la definición de ecuaciones para la variación de la dimensión y valores en la matriz R_k de los filtros (para decidir cuándo se fusionan las mediciones con retardo).

Para el caso específico de localización, si el retardo se da en la medición de la postura $z_{k,p}$ (global), esta puede trasladarse hasta el instante actual k utilizando la velocidad y el tiempo de retardo medido, al considerar un desplazamiento lineal. Este método podría solventar la presencia del retardo en la medición en los algoritmos definidos (actualización por tiempo, Algoritmos del 5 al 8) siempre y cuando el ancho de banda disponible en la plataforma no esté limitado, o bien si se administra adecuadamente al limitar las frecuencias de actualización de los sensores en el caso de recursos limitados.

En el caso de que varias de las frecuencias de medición sean distintas pero constantes y conocidas, se puede recurrir a métodos de fusión sensorial multifrecuencia [156, 45, 10, 11, 136] los cuales se diseñan para incorporar en la fusión las mediciones según la frecuencia de cada sensor al utilizar retenedores multifrecuencia (por ejemplo un ZOH a la entrada y salida del filtro), obteniendo la postura del robot de forma adecuada e incorporando en algunos casos los retardos existentes en las mediciones. Sin embargo, estos métodos podrían no funcionar adecuadamente al ser implementados en

plataformas en donde el ancho de banda es limitado, ya que no todas las mediciones se podrían obtener e incorporar a la frecuencia adecuada para cada sensor. Además, podrían presentar dificultades cuando alguno de los sensores tiene una frecuencia de muestreo variable, caso que puede presentarse en el uso de los sensores complejos ya que, según los recursos disponibles en la plataforma en el instante k, pueden determinar la medición con mayor o menor frecuencia. Por otra parte, los retenedores multifrecuencia requieren gran cantidad de memoria al requerir un historial de los valores de cada señal a utilizar en la fusión de acuerdo al orden del retenedor, por lo que no son adecuados en el caso de recursos limitados.

Con el fin de incorporar las mediciones con retardos o con frecuencia variable a la vez que se limita el uso del ancho de banda de la plataforma para el método de localización, se puede utilizar la teoría de control y muestreo basado en eventos para modificar los filtros de fusión expuestos anteriormente. Este enfoque permite realizar la fusión de sensores con frecuencia de medición variable o bien realizar la fusión cuando el ancho de banda esté disponible en la plataforma, según se expone a continuación.

6.2 Fusión Sensorial basada en eventos para localización

Como se describió en el capítulo de antecedentes, la idea principal de los métodos de control y muestreo basados en eventos es realizar el muestreo del sensor o aplicar la acción de control, únicamente si un *evento* $\boldsymbol{R_A}$ definido en función del error (en la referencia, perturbación, o respecto al valor anterior de medición) supera un nivel límite $\boldsymbol{R_{A,lim}}$ predeterminado, por lo que se reduce el ancho de banda y consumo de energía en la plataforma a la vez que se obtiene un funcionamiento similar a las estrategias con un tiempo de muestreo constante.

El ahorro en el ancho de banda y comunicaciones depende de la frecuencia con la que el evento supere el valor límite, de forma que son métodos de compromiso entre el desempeño de la solución y el uso del ancho de banda para el caso de plataformas sin limitaciones de recursos [48]. Sin embargo,

para plataformas de recursos limitados, este compromiso es una ventaja que permite ajustar el uso del ancho de banda para que, dentro de las limitaciones inherentes al uso del medio de comunicación, se pueda implementar el esquema de fusión de múltiples sensores. El enfoque basado en eventos permite además definir esquemas de control que incorporen sensores con frecuencias variables y retardos en la medición, al considerar ambas situaciones como un evento que condiciona la realización o no de la medición y por lo tanto cuando es posible realizar el control de la plataforma.

En cuanto a filtros de fusión sensorial existen diversos ejemplos que utilizan los principios del control basado en eventos para realizar la fusión. En general, estos métodos realizarán el filtrado y fusión sensorial únicamente si el evento, definido en función del cambio en el valor de la medición del sensor o en función del error en el lazo de control, supera su valor límite, en cuyo caso se procede a realizar la estimación correspondiente [156, 152, 78] con buenos resultados en sistemas distribuidos [57, 119].

6.2.1 Corrección global por eventos

Para el caso específico de fusión sensorial para la mejora de la localización de un robot de recursos limitados, partiendo de los resultados obtenidos con las variantes del filtro en cascada se puede observar que *no* existen problemas de acceso, uso de ancho de banda, retardos o frecuencias de muestreo variable en el tiempo para la sección de los filtros que utiliza la información local (velocidades), que pueden ser ejecutados sin inconvenientes en cada instante k. Sin embargo, estos problemas *si* pueden presentarse en la sección de corrección de los filtros con información global (postura) en donde se puede requerir el acceso a sensores utilizando el medio de comunicación.

De esta forma se propone en la presente tesis utilizar el esquema de fusión basado en eventos únicamente para la sección de corrección global del filtro de localización, ejecutándola cuando $\boldsymbol{R_A} > \boldsymbol{R_{A,lim}}$, esto a diferencia de los métodos existentes que ejecutan la totalidad del filtro al cumplirse la condición del evento. Debido a esto se denotarán los algoritmos propuestos como filtros con corrección o actualización global basada en eventos.

En consecuencia, las mediciones $z_{k,v}$ serán utilizadas en cada instante k por el filtro local que estimará la postura del robot utilizando únicamente esta información. Por el contrario, la corrección del filtro global se realizará si se cumple con la condición del evento, en cuyo caso se obtendrá $z_{k,p}$ utilizando el medio de comunicación. Debido a que esta comunicación introduce un retardo en la obtención de $z_{k,p}$, se ejecutará únicamente el filtro local hasta que se haya recibido su valor. Una vez recibida la medición $z_{k,p}$ se compensará en su valor el tiempo retardo, con lo que se procede a realizar la etapa de corrección del filtro; la cual, además de corregir la estimación de la postura y la covarianza del error con la información global, restablecerá el valor de R_A, desactivando la corrección global hasta que se vuelva a cumplir con la condición $R_A > R_{A,lim}$. En la figura 6.1 se muestra el lazo de control realimentado modificado para incorporar la solución propuesta, la línea discontinua indica que el acceso a la información no se realiza en cada instante k, sino que depende del evento R_A.

Con este planteamiento, se puede conseguir una regulación en el uso del ancho de banda de la plataforma, con lo que se podría implementar el filtro en plataformas de recursos de comunicación limitados. Además se tendrá un funcionamiento adecuado ante la variación de la frecuencia de muestreo o del retardo de comunicación de los sensores globales ya que se ejecutará el filtro local hasta que se haya recibido $z_{k,p}$, con lo que el control de navegación del robot tendrá disponible una estimación de la postura en todo instante k. El efecto del retardo en $z_{k,p}$ puede ser compensado según la plataforma siempre que su valor sea medido por el robot, que es el caso considerado en la presente tesis. Los detalles de este ajuste se detallan en el capítulo 8.

6.2.2 Definición del evento

Establecida la corrección basada en eventos para el caso de localización, se procede a realizar la definición del evento R_A mediante el cual se administra el uso del sensor global para adquirir $z_{k,p}$ tal y como se describe a continuación.

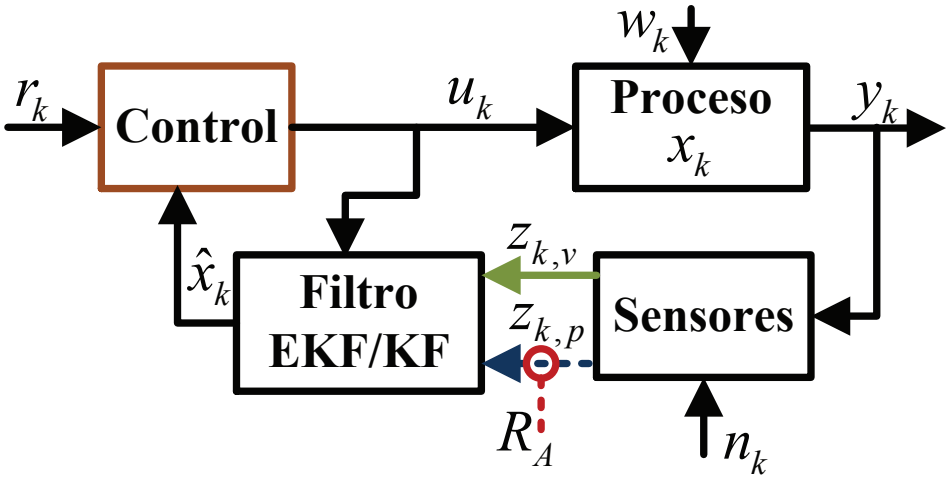


Figura 6.1: Lazo de control realimentado con el filtro de fusión basado en el evento R_A.

6.2.2.1 Problemática asociada a las definiciones existentes

Como se mencionó anteriormente, se suele definir R_A con base en el error de estimación en el caso de filtros de fusión o bien cuando se da un cambio en el valor de la variable medida. Para el caso específico de localización, el error de estimación es desconocido para el robot al menos que se consulte constantemente un sensor global. Debido a esto, no es conveniente definir un evento que dependa del error de estimación para limitar el uso del ancho de banda, ya que no sería posible conocer el error sin antes utilizar el ancho de banda para consular el sensor global. Además, al ser la postura del robot una variable en constante evolución según el robot se mueve en el entorno, definir un evento con base al cambio en el valor de la postura del sensor global producirá una actualización en cada instante en donde el robot se mueva, con lo que el uso del ancho de banda puede ser excesivo en plataformas de recursos limitados. Por estas razones, conviene definir el evento en el caso de localización utilizando la información local del robot y no la información accesible mediante el uso de comunicaciones (sensores globales remotos, robots vecinos, etc.).

Algunos métodos existentes [68, 148] utilizan el error en la referencia del control de postura (error de navegación) para definir el evento para el algoritmo de estimación por eventos (predicción y corrección) aplicado en problemas de consenso multiagente. Sin embargo, esta definición del evento tiene un gran inconveniente debido a que el error de navegación (que indica si el robot ha llegado al punto de destino o a un punto intermedio de la trayectoria a seguir) puede ser independiente del error de localización (que indica con que precisión se determinó la postura del robot o si el robot se ha perdido). Por esta razón, se pueden dar casos en donde el error de localización sea grande (el robot ha estimado una postura distinta a la real) pero con un error de navegación pequeño (la postura estimada erróneamente se encuentra cerca de un punto objetivo), por lo que el evento definido con el error de navegación no superará el nivel predeterminado para realizar la estimación de la postura y de esta forma disminuir su error.

Esto es una desventaja considerable principalmente en el caso de robots con recursos limitados que dependen en gran medida de las estimaciones de la postura utilizando las mediciones locales $z_{k,v}$, las cuales acumulan el error de localización mediante el proceso de integración recursiva (del instante anterior al actual), por lo que al cabo de poco tiempo el error de estimación puede ser considerable y no disminuirá al menos que se utilice la información global (por ejemplo una cámara cenital) para realizar la corrección. En este ejemplo, en caso de definir el evento con el error de navegación, no se utilizará la información global en caso de estar cerca de un punto objetivo, con lo que no se estimará adecuadamente la postura. Además si el robot se encuentra muy alejado del punto objetivo (error de navegación alto) el evento superará $R_{A,lim}$ en cada instante k hasta que el robot se acerque al punto de destino, por lo que se hará un uso intensivo del ancho de banda en los primeros instantes de la navegación, lo que no resulta conveniente en este tipo de plataformas.

6.2.2.2 Evento para localización: aproximación inicial

Ante esta problemática queda en evidencia la necesidad de realizar una nueva definición del evento, específica al caso de localización, de forma que se obtenga un desempeño adecuado al utilizar el esquema de fusión con corrección basada en eventos (figura 6.1) en plataformas de recursos limitados. En la presente tesis se propone realizar la definición del evento en función de la *covarianza* del error de estimación local de la postura $P_{k,p}$, la cual es determinada por la etapa local del filtro de fusión EKF/KF en cada instante de muestreo k.

Como se mencionó anteriormente, el error de estimación en el filtro local crece indefinidamente debido al proceso de integración recursiva y únicamente disminuye cuando se realiza la actualización con la información global. De esta forma, aunque este error no está disponible al menos que se utilice el sensor global, se tiene la ventaja de que la covarianza del error de estimación de la postura del robot $P_{k,p}$ obtenida del filtro de fusión tiene el mismo comportamiento creciente si es determinada utilizando únicamente la información local, y disminuirá su valor al utilizar la información global (siempre y cuando la precisión de la medición sea adecuada). Por esta razón un evento R_A definido con base en la $P_{k,p}$ local puede indicar claramente cuando el error de estimación de la postura es considerable, indicando la necesidad de realizar la corrección utilizando la $z_{k,p}$ global, además de reducir su valor una vez que se realice la corrección del filtro global.

Una primera definición del evento se establece en la ecuación (6.1) como la traza de la matriz $P_{k,p}$ obtenida del filtro local, la cual sirve como una primera aproximación ya que al ser la suma de las covarianzas de la postura puede no ser clara la definición de un límite $R_{A,lim}$ adecuado para robots móviles.

$$R_A = \mathrm{Tr}\, P_{k,p} = \sigma_x^2 + \sigma_y^2 + \sigma_\theta^2 \tag{6.1}$$

Se puede realizar una mejor definición de R_A intentando establecer de forma intuitiva el límite $R_{A,lim}$. Para esto se recurre a la interpretación geométrica

de la matriz de covarianza gaussiana la cual puede representarse mediante el uso de los elipsoides de incertidumbre según se define a continuación.

6.2.2.3 Elipsoides de incertidumbre

Según la definición de la *pdf* [151] de una variable aleatoria gaussiana X de orden n con autocovarianza C_X (ecuación (B.7), apéndice B), su exponente es el único término que depende del valor actual de X siendo el término que multiplica al exponente una constante de normalización. De esta forma, para definir un contorno en donde la *pdf* sea constante se estudia únicamente su exponente descrito en la ecuación (6.2).

$$\left(X - \bar{X}\right)^T C_X^{-1} \left(X - \bar{X}\right) = 1 \tag{6.2}$$

Como se observa en la ecuación (6.2), al igualar el exponente a 1 y al ser C_X una matriz simétrica y definida positiva (en el caso de localización), se obtiene la definición matemática para un elipsoide general de dimensión n, centrado en la media $\bar{X}$. La dirección de los ejes del elipsoide viene dada por los vectores propios de C_X, y la inversa de la magnitud de los semiejes al cuadrado es igual a los valores propios de C_X. El elipsoide definido corresponde al lugar geométrico de la ecuación (6.2), es decir, a todos los puntos X que satisfacen (6.2) y que describen un contorno de densidad de probabilidad *pdf* constante [155, 154], debido a que la *pdf* es constante si lo es su exponente, lo cual se cumple en la ecuación (6.2) al haberse igualado a 1.

Para el caso general, se hace uso de la definición de la distancia *Mahalanobis*, la cual es el operador de distancia cuadrática normalizada definido en la ecuación (6.3). Con esta definición se observa claramente como el elipsoide de la ecuación (6.2) corresponde con el lugar geométrico de los puntos X que se encuentran a una distancia Mahalanobis unitaria de la media $\bar{X}$

$$\mathrm{MD}(X, \bar{X}; C_X) = \sqrt{\left(X - \bar{X}\right)^T C_X^{-1} \left(X - \bar{X}\right)} \tag{6.3}$$

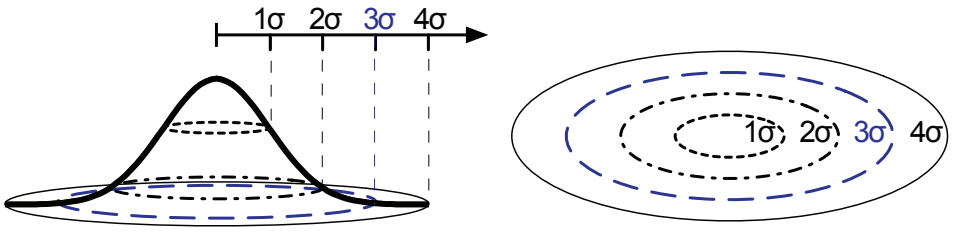


Figura 6.2: Descripción mediante los $n\sigma$ elipsoides de la matriz de covarianza para una variable aleatoria gaussiana bidimensional (2D) [155, 154].

En el caso específico de localización resulta conveniente la definición de elipsoides con una distancia Mahalanobis mayor a la unidad. Estos se conocen como elipsoides n-sigma ($n\sigma$) los cuales se definen en función de n según se muestra en la figura 6.1 y en la la ecuación (6.4), la cual describe el lugar geométrico de todos los puntos X a una distancia Mahalanobis n de la media $\bar{X}$.

$$\left(X - \bar{X}\right)^T C_X^{-1} \left(X - \bar{X}\right) = n^2 \tag{6.4}$$

Con esta definición se denota el elipsoide definido en la ecuación (6.2) del lugar geométrico de los puntos a una distancia Mahalanobis unitaria como el elipsoide 1σ (~ 1 desviación estándar). Se pueden construir elipsoides mayores utilizando la ecuación (6.4), de los cuales son de utilidad en localización y SLAM los elipsoides 2σ y 3σ [155, 154].

Al integrar la *pdf* contenida dentro de los elipsoides se pueden establecer regiones de confianza. De esta forma puede obtenerse las probabilidades porcentuales de que el valor actual (real) la variable aleatoria X se encuentre dentro de su elipsoide $n\sigma$ [155, 154], obteniéndose para el caso 2D (2 dimensiones) una confianza de un $39,4\,\%$ para el elipsoide 1σ, un $86,5\,\%$ para el 2σ, un $98,9\,\%$ para el 3σ y un $99,97\,\%$ para el elipsoide 4σ. De estas regiones de confianza se observa que se requiere utilizar al menos los elipsoides 3σ para representar adecuadamente la covarianza de la variable aleatoria X de forma gráfica.

A manera de ejemplo se muestra en la figura 6.3 la evolución de la covarianza de la estimación de la postura de un robot que utiliza únicamente la odometría a partir de los encoders para seguir una trayectoria circular. Se observa cómo según avanza el robot la incertidumbre representada por los elipsoides 3σ se incrementa debido a la acumulación del error. De esta forma la covarianza crecerá indefinidamente por lo que también lo harán los respectivos elipsoides 3σ hasta que se utilice una medición con información global.

Con esto se comprueba la conveniencia de utilizar la covarianza de la postura para definir el evento R_A, con lo que si se utiliza su representación geométrica dada por los elipsoides 3σ se obtienen las mismas ventajas además de poder seleccionar el límite $R_{A,lim}$ de forma intuitiva. Por ejemplo se puede describir el evento y su límite mediante características geométricas obtenidas a partir de los elipsoides 3σ. Para el caso de navegación 3D se podría utilizar el volumen de los elipsoides, entretanto el uso del área es más conveniente para el caso de navegación 2D el cual es el estudiado en la presente tesis. Con esto se utilizará el área para definir R_A según se describe a continuación.

6.2.2.4 Evento normalizado

Utilizando la descripción previa de la representación gráfica de la matriz de covarianza mediante los elipsoides $n\sigma$, se propone definir el evento en función del área del elipsoide 3σ obtenido de la matriz de covarianza del error de postura $P_{k,p}$ estimada por el filtro local. Para esto primeramente se reescribe la ecuación (6.4) utilizando la nomenclatura del filtro y del error de estimación tal y como se muestra en la ecuación (6.5).

$$(x_{k,p} - \hat{x}_{k,p})^T P_{k,p}^{-1} (x_{k,p} - \hat{x}_{k,p}) = n^2 \tag{6.5}$$

Para el caso de localización, al ser la covarianza de θ propagada en la posición (x, y) mediante la actualización local de los filtros en cascada o mediante la ecuación (5.42) en el caso de modelos en cascada, se puede utilizar únicamente el área definida con las covarianzas de (x, y) ya que estas incluyen el efecto de la covarianza θ. De la misma forma, para el caso en 3D (no considerado en la presente tesis) se utilizaría el volumen del elipsoide considerando úni-

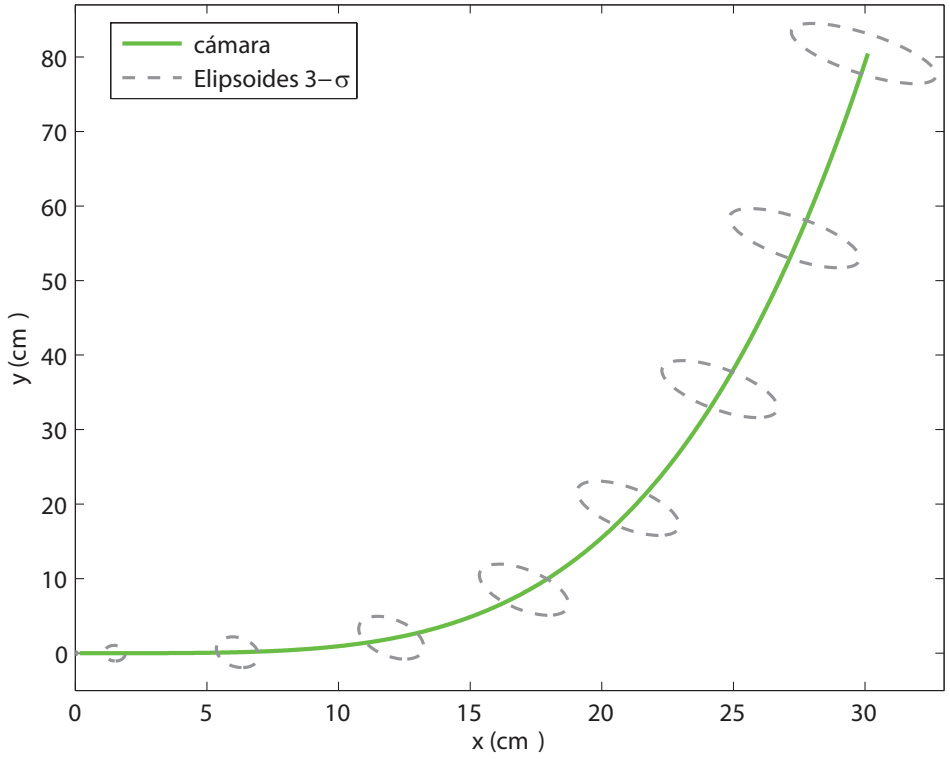


Figura 6.3: Evolución creciente de la covarianza del error de estimación indicada por medio de los elipsoides 3σ en un robot diferencial que estima su postura mediante odometría (encoders) y sigue una trayectoria circular; trayectoria medida utilizando una cámara cenital.

camente la covarianza en la posición (x, y, z). De esta forma, al considerar la $P_{k,xy}$ del filtro local, se obtienen las magnitudes de los semiejes del elipsoide de la ecuación (6.5) como a_σ y b_σ al obtener los autovalores de $P_{k,xy}$, con $n = 3$ para el elipsoide 3σ, según se establece en la ecuación (6.6).

$$\begin{aligned} \mathbf{P}_{k,xy} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^2 \end{bmatrix}, T_\sigma = \sqrt{\sigma_x^4 + \sigma_y^4 - 2\sigma_x^2\sigma_y^2 + 4\sigma_{xy}^4} \\ a_\sigma = \sqrt{\frac{2l^2|P_{xy}|}{\sigma_x^2+\sigma_y^2+T_\sigma}}, b_\sigma = \sqrt{\frac{2l^2|P_{xy}|}{\sigma_x^2+\sigma_y^2-T_\sigma}} \end{aligned} \tag{6.6}$$

Si se calcula el área de los elipsoides 3σ y se dividen por el área del robot se tendría un indicador normalizado del momento en el cual el error en la estimación de la postura del robot es igual que el tamaño del robot (su área). De esta forma, al definir el evento $\boldsymbol{R}_A$ como la razón de estas áreas según la ecuación (6.7), se podría definir de forma intuitiva un límite $\boldsymbol{R}_{A,lim}$ adecuado para el evento. Esto debido a que se tiene la interpretación geométrica que cuantas áreas equivalentes del robot se dejaría crecer el error antes de realizar la actualización global utilizando $z_{k,p}$. Por ejemplo, se consultaría el sensor global si R_A supera un nivel determinado, por ejemplo 1,6, indicando que el área del error es 1,6 veces el área de robot, con lo que una vez superado el valor del evento en 1,6 se obtendría y utilizaría $z_{k,p}$ para actualizar la postura del robot. Esta forma de actualización resultaría mucho más eficiente en términos de recursos computacionales y de comunicación ya que no utilizaría el ancho de banda en todo instante k, a diferencia de la actualización por tiempo expuesta en la sección anterior.

$$\begin{aligned} A_{ellip} &= \pi a_\sigma b_\sigma \\ \boldsymbol{R}_A &= A_{ellip}/A_{robot} \end{aligned} \tag{6.7}$$

Además, con esta definición de la ecuación (6.7), se tiene la ventaja de asociar el número de consultas al sensor global de acuerdo a la precisión de la estimación local, ya que si se tienen sensores muy precisos, se podría tardar un tiempo considerable en llegar al valor $R_{A,lim}$, con lo que se utilizaría con menor frecuencia el medio de comunicación según se defina el valor de $R_{A,lim}$. De esta forma, $R_{A,lim}$ se escoge como un compromiso entre el número de actualizaciones globales que se realizan (consumo de ancho de banda,

procesador y energía) y la precisión deseada en la estimación de la postura del robot.

La escogencia del valor de $R_{A,lim}$ se puede realizar además como una indicación al filtro basado en eventos de que tan buena es la precisión del filtro local. Un $R_{A,lim}$ alto indica, por ejemplo, que se utiliza un filtro local con sensores precisos, por lo que la estimación local tendrá una precisión adecuada por un intervalo de tiempo considerable (mientras $R_A < R_{A,lim}$), en el que no se requiere utilizar $z_{k,p}$ para actualizar la postura. Una vez se supera $R_{A,lim}$ se procede a realizar la actualización global con lo que mejora la precisión de la postura de forma considerable, disminuyendo el error acumulado en la etapa global y por lo tanto disminuyendo el valor de R_A con lo que se cumple de nuevo con $R_A < R_{A,lim}$ y se repite el ciclo de actualización global una vez que se vuelva a cumplir con $R_A > R_{A,lim}$. Por el contrario, un valor $R_{A,lim}$ bajo indicaría por ejemplo que se utiliza una fusión local poco precisa (por ejemplo utilizando únicamente un sensor o varios o sensores independientes pero con poca precisión) por lo que se requiere una mayor frecuencia de actualización global para corregir la postura con $z_{k,p}$, requiriendo mayor acceso al medio de comunicación y al sensor global.

La definición del evento se puede ajustar si fuera necesario para incorporar casos adicionales como la navegación en exteriores, la disponibilidad de sensores, la localización cooperativa, etc. lo cual se realiza según la plataforma a utilizar. Por ejemplo, en el caso de navegación en exteriores, se puede agregar una condición adicional para que el algoritmo realice la corrección global únicamente si el número de satélites utilizados por el sensor GPS $\boldsymbol{N}_{sat}$ es mayor a un valor mínimo $\boldsymbol{N}_{sat,min}$ preestablecido (por ejemplo 6 satélites). Con esto se busca garantizar una precisión mínima en la medición global a utilizar en la fusión sensorial. En este caso además se debe realizar el ajuste dinámico de la matriz $R_{k,p}$ (magnitud), utilizando una función inversamente proporcional, es decir, si el número de satélites es alto entonces los valores de $R_{k,p}$ (correspondientes a $z_{k,p} = L_{GPS}$) deben ser bajos, de forma que el filtro de mayor importancia a estos datos en la fusión y viceversa.

Finalmente, en los capítulos de pruebas experimentales se presenta un estudio del desempeño de los filtros de fusión ante las variaciones en el valor de

$R_{A,lim}$ que ilustran de una mejor forma el compromiso entre la precisión y el acceso al sensor global utilizando el medio de comunicación del robot al utilizar los algoritmos basados en eventos. Realizada la definición de R_A se procede a modificar los algoritmos de la sección anterior para incorporar la actualización basada en eventos según se describe a continuación.

6.3 Filtros de fusión sensorial con corrección basada en eventos

Se modifican a continuación los algoritmos con actualización temporal del capítulo 5 para incorporar la actualización global por eventos, utilizando el evento definido en la ecuación (6.7). La modificación a realizar es la misma en todos los algoritmos: se comprueba el valor del evento R_A en cada instante k, en caso de que $R_A > R_{A,lim}$ se procede a realizar la corrección global utilizando $z_{k,p}$, lo que reduce el valor de $P_{k,p}$ y por lo tanto el de R_A, con lo que $R_A < R_{A,lim}$, por lo que al siguiente ciclo se vuelve a determinar la postura únicamente utilizando $z_{k,v}$. De esta forma se obtienen los algoritmos basados en eventos del 9 al 12 tal y como se exponen a continuación.

Algoritmo 9: Algoritmo EKF recursivo en cascada, asignación de entradas, predicción local y corrección global basada en eventos

Entrada: $u_{k-1} = [\ u_1 \quad u_2 \quad u_3\ ]^T$,$\hat{x}_{k-1}$,$P_{k-1}$

Medición: $z_k = [\ z_{k,p} \quad z_{k,v}\ ]^T$, $z_{k,v} = [\ v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ ]_k^T$

$z_{k,p} = L_{GM/GPS} = [\ x_k\ y_k\ \theta_k\ ]_{GM/GPS}^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,R_k,A_{robot},$R_{A,lim}$

Salida: $\hat{x}_k$,P_k

Inicialización: $\hat{x}_0$,P_0

Para *el instante actual* k **hacer**

Predicción:

$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$

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

$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$

$H_k = \frac{\partial h}{\partial x}\Big|_{\hat{x}_k,0} = [H_p\ \ H_{pv}; H_{vp}\ \ H_v]_k = [H_p\ \ 0; 0\ \ H_v]_k$, $N_k = \frac{\partial h}{\partial n}\Big|_{\hat{x}_k,0}$

Corrección Local: utilizar H_p nula para actualizar con $z_{k,v}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k \left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

Elipsoide 3σ: utilizar $P_{k,xy}$ para calcular el evento R_A:

Obtener Semiejes del Elipsoide 3σ: a_σ,b_σ, ecuación (6.6)

$A_{ellip} = \pi a_\sigma b_\sigma$ ecuación (6.7)

$\boldsymbol{R}_A = A_{ellip}/A_{robot}$

Si $R_A > R_{A,lim}$ **entonces**

Corrección Global: utilizar H_v nula para actualizar con $z_{k,p}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k \left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

fin

fin

Algoritmo 10: Algoritmo EKF recursivo en cascada–reducido, asignación de entradas, predicción local y corrección global basada en eventos

Entrada: $u_{k-1} = [\ u_1 \quad u_2 \quad u_3\ ]^T$,$\hat{x}_{k-1}$,$P_{k-1}$

Medición: $z_k = [\ z_{k,p} \quad z_{k,v}\ ]^T$, $z_{k,v} = [\ v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ ]_k^T$

$z_{k,p} = L_{GM/GPS} = [\ x_k\ y_k\ \theta_k\ ]_{GM/GPS}^T$

Datos: $f(\cdot)$ *y* $h(\cdot)$ *(modelo no lineal* (3.24) *o* (3.51)*)*,Q_k,

$R_k = [R_p\ \ R_{pv}; R_{vp}\ \ R_v]_k$,$A_{robot}$,$R_{A,lim}$

Salida: $\hat{x}_k = [\ x_{k,p} \quad x_{k,v}\ ]^T$,$P_k = [P_p\ \ P_{pv}; P_{vp}\ \ P_v]_k$

Inicialización: $\hat{x}_0$,P_0

Para *el instante actual* k **hacer**

Predicción:

$\hat{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$

$A_k = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1},u_{k-1},0}$, $W_k = \left.\frac{\partial f}{\partial w}\right|_{\hat{x}_{k-1},u_{k-1},0}$

$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$

$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_k,0} = [H_p\ \ H_{pv}; H_{vp}\ \ H_v]_k = [H_p\ \ 0; 0\ \ H_v]_k$, $N_k = \left.\frac{\partial h}{\partial n}\right|_{\hat{x}_k,0}$

Corrección Local: utilizar H_p nula para actualizar con $z_{k,v}$

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k\left[z_k - h(\hat{x}_k, 0)\right]$

$P_k = (I - K_k H_k) P_k$

Elipsoide 3σ: utilizar $P_{k,xy}$ para calcular el evento R_A:

Obtener Semiejes del Elipsoide 3σ: a_σ,b_σ, ecuación (6.6)

$A_{ellip} = \pi a_\sigma b_\sigma$ ecuación (6.7)

$\boldsymbol{R}_A = A_{ellip}/A_{robot}$

Si $R_A > R_{A,lim}$ **entonces**

Corrección Global: actualizar únicamente $(\hat{x}_{k,p}, P_{k,p})$ con $z_{k,p}$

$K_{k,p} = P_{k,p} H_{k,p}^T \left(H_{k,p} P_{k,p} H_{k,p}^T + R_{k,p}\right)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right)$

$P_{k,p} = P_{k,p} - K_{k,p} H_{k,p} P_{k,p}$

fin

fin

Algoritmo 11: Algoritmo EKF recursivo en cascada con modelos local/global en cascada, asignación de entradas, predicción local de velocidades, modelo global de postura y corrección global basada en eventos

Entrada: $u_{k-1} = u_{Aks} = [u_1\ u_2\ u_3]^T,\ \hat{x}_{k-1,v}, P_{k-1,v},\ \hat{x}_{k-1,p}, P_{k-1,p}$

Medición: $z_{k,v} = [v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}]_k^T, z_{k,p} = [x_k\ y_k\ \theta_k]_{GM/GPS}^T$

Datos: $f_v(\cdot)$ *y* $h_v(\cdot)$ *del modelo no lineal local* (3.50), $Q_{k,v}, R_{k,v}, A_{robot}, R_{A,lim}$
$f_p(\cdot)$ *y* $h_p(\cdot)$ *del modelo no lineal global* (3.36), $Q_{k,p}, R_{k,p}$

Salida: $\hat{x}_{k,v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}_k^T, \hat{x}_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T, P_{k,p},\ P_{k,v}$

Inicialización: $\hat{x}_{0,v}, P_{0,v}, \hat{x}_{0,p}, P_{0,p}$

Para *el instante actual* k **hacer**

Predicción Local:

$\hat{x}_{k,v} = f_v(\hat{x}_{k-1,v}, u_{k-1}, 0), A_{k-1,v} = \frac{\partial f_v}{\partial x_v}\Big|_{\substack{x_{k-1} \\ u_{k-1}}},\ W_{k-1,v} = \frac{\partial f_v}{\partial w_v}\Big|_{\substack{x_{k-1} \\ u_{k-1}}}$

$P_{k,v} = A_{k-1,v} P_{k-1,v} A_{k-1,v}^T + W_{k-1,v} Q_{k-1,v} W_{k-1,v}^T$

$H_{k,v} = \frac{\partial h_v}{\partial x_v}\Big|_{\hat{x}_k,0},\ N_{k,v} = \frac{\partial h_v}{\partial n_v}\Big|_{\hat{x}_k,0}$

Corrección Local:

$K_{k,v} = P_{k,v} H_{k,v}^T \Big(H_{k,v} P_{k,v} H_{k,v}^T + N_{k,v} R_{k,v} N_{k,v}^T \Big)^{-1}$

$\hat{x}_{k,v} = \hat{x}_{k,v} + K_{k,v} [z_{k,v} - h_{k,v}(\hat{x}_{k,v}, 0)]$

$P_{k,v} = (I - K_{k,v} H_{k,v}) P_{k,v}$

Modelo Global, Covarianza Global:

$\hat{x}_{k,p} = f_p(\hat{x}_{k-1,p}, \hat{x}_{k,v}),\ A_{k,p} = \frac{\partial f_p}{\partial x_p}\Big|_{\substack{x_{k-1} \\ u_{k-1}}}, A_{k,pv} = \frac{\partial f_p}{\partial x_v}\Big|_{\substack{x_{k-1} \\ u_{k-1}}}$

$P_{k,p} = A_{k,p} P_{k-1,p} A_{k,p}^T + A_{k,pv} P_{k,v} A_{k,pv}^T + Q_{k,p}$

Elipsoide 3σ: utilizar $P_{k,xy}$ para calcular el evento R_A:

Obtener Semiejes del Elipsoide 3σ: a_σ, b_σ, ecuación (6.6)

$A_{ellip} = \pi a_\sigma b_\sigma,\ \Rightarrow \boldsymbol{R}_A = A_{ellip}/A_{robot}$, ecuación (6.7)

Si $R_A > R_{A,lim}$ **entonces**

Corrección Global: $H_{k,p} = \frac{\partial h_p}{\partial x_p}\Big|_{\hat{x}_k,0} = I_{3\times3}$, ecuación (5.36)

$K_{k,p} = P_{k,p} H_{k,p}^T \Big(H_{k,p} P_{k,p} H_{k,p}^T + R_{k,p} \Big)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p} (z_{k,p} - H_{k,p} \hat{x}_{k,p})$

$P_{k,p} = P_{k,p} - K_{k,p} H_{k,p} P_{k,p}$

fin

fin

Algoritmo 12: Algoritmo KF recursivo en cascada con modelos local/global en cascada, asignación de entradas, predicción local de velocidades, modelo global de postura y corrección global basada en eventos

Entrada: u_{k-1}=u_{Akns}=$[u_1 \ u_3]^T$=u_{dif}=$[a \ \alpha]^T$,$\hat{x}_{k-1,v}$,$P_{k-1,v}$,$\hat{x}_{k-1,p}$,$P_{k-1,p}$

Medición: $z_{k,v} = [\ v_{x,enc} \ v_{y,enc} \ \omega_{enc} \ \omega_{gyr} \ \omega_{comp}\ ]_k^T$,
$z_{k,p} = [\ x_k \ y_k \ \theta_k\ ]_{GM/GPS}^T$

Datos: A_k *y* B_k *modelo lineal local* (3.23) *(o* (3.52)*)*,$Q_{k,v}$,$R_{k,v}$,A_{robot},
$f_p(\cdot)$,$h_p(\cdot)$ *modelo no lineal global* (3.11) *(o* (3.12)*)*,$Q_{k,p}$,$R_{k,p}$,$R_{A,lim}$

Salida: $\hat{x}_{k,v} = \left[\ v_x \quad v_y \quad \omega\ \right]_k^T$,$\hat{x}_{k,p} = \left[\ x \quad y \quad \theta\ \right]_k^T$,$P_{k,p}$, $P_{k,v}$

Inicialización: $\hat{x}_{0,v}$,$P_{0,v}$,$\hat{x}_{0,p}$,$P_{0,p}$

Para *el instante actual* k **hacer**

Predicción Local:

$\hat{x}_{k,v} = A_{k-1,v}\hat{x}_{k-1,v} + B_{k-1,v}u_{k-1}$

$P_{k,v} = A_{k-1,v}P_{k-1,v}A_{k-1,v}^T + W_{k-1,v}Q_{k-1,v}W_{k-1,v}^T$

Corrección Local:

$K_{k,v} = P_{k,v}H_{k,v}^T\Big(H_{k,v}P_{k,v}H_{k,v}^T + N_{k,v}R_{k,v}N_{k,v}^T\Big)^{-1}$

$\hat{x}_{k,v} = \hat{x}_{k,v} + K_{k,v}\left(z_{k,v} - H_{k,v}\hat{x}_{k,v}\right)$

$P_{k,v} = \left(I - K_{k,v}H_{k,v}\right)P_{k,v}$

Modelo Global, Covarianza Global:

$\hat{x}_{k,p} = f_p\left(\hat{x}_{k-1,p}, \hat{x}_{k,v}\right)$,$A_{k,p} = \frac{\partial f_p}{\partial x_p}\Big|_{\substack{x_{k-1}\\u_{k-1}}}$,$A_{k,pv} = \frac{\partial f_p}{\partial x_v}\Big|_{\substack{x_{k-1}\\u_{k-1}}}$

$P_{k,p} = A_{k,p}P_{k-1,p}A_{k,p}^T + A_{k,pv}P_{k,v}A_{k,pv}^T + Q_{k,p}$
$\approx P_{k-1,p} + A_{k,pv}P_{k,v}A_{k,pv}^T + Q_{k,p}$

Elipsoide 3σ: utilizar $P_{k,xy}$ para calcular el evento R_A:

Obtener Semiejes del Elipsoide 3σ: a_σ,b_σ, ecuación (6.6)

$A_{ellip} = \pi a_\sigma b_\sigma$, $\Rightarrow \boldsymbol{R}_A = A_{ellip}/A_{robot}$, ecuación (6.7)

Si $R_A > R_{A,lim}$ **entonces**

Corrección Global: $H_{k,p} = \frac{\partial h_p}{\partial x_p}\Big|_{\hat{x}_k,0} = I_{3\times3}$, ecuación (5.37)

$K_{k,p} = P_{k,p}H_{k,p}^T\Big(H_{k,p}P_{k,p}H_{k,p}^T + R_{k,p}\Big)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k,p} + K_{k,p}\left(z_{k,p} - H_{k,p}\hat{x}_{k,p}\right)$

$P_{k,p} = P_{k,p} - K_{k,p}H_{k,p}P_{k,p}$

fin

fin

6.4 Conclusiones del Capítulo

A partir de los filtros basados en el tiempo del capítulo 5 se han desarrollado los algoritmos basados en *eventos* expuestos en el presente capítulo, que realizan la etapa de corrección del filtro cuando, durante la ejecución del algoritmo, un evento definido según la información local del robot supera un nivel predeterminado:

- Son métodos adecuados cuando existen problemas de disponibilidad en la medición, como por ejemplo un retardo en la medición debido a la utilización del medio de comunicación de la plataforma, o bien frecuencias distintas de medición entre sensores o variables en el tiempo debido al procesamiento requerido por sensores complejos, además de los casos de no disponibilidad del sensor o de un ancho de banda limitado dentro de la plataforma. Tienen ventaja sobre los métodos existentes que únicamente presentan soluciones parciales a estos problemas, siendo en muchos de los casos métodos complejos y no adecuados para dar una solución general, o bien no implementables en plataformas de recursos limitados.

- En el caso de fusión para la localización, se observó que los problemas de disponibilidad de la medición solamente están presentes en la etapa de corrección global y no en la etapa de fusión con la información local. Por esta razón se propuso la incorporación de la corrección global basada en eventos en los filtros en cascada propuestos, de forma que se ejecuta esta etapa si el evento $\boldsymbol{R_A}$ supera su valor límite predeterminado $\boldsymbol{R_{A,lim}}$. Esto a diferencia de los métodos existentes que ejecutan ambas etapas (predicción, corrección) con base al evento.

- Se realizó una definición del evento adecuada al caso de localización de robots móviles para lo cual se estudiaron primeramente las definiciones existentes del evento, de las cuales se determinó que no son adecuadas para el caso de localización, principalmente los eventos basados en el error de navegación (el que indica si el robot ha llegado al punto objetivo), ya no refleja la evolución del error de localización. Con esto se procedió a definir un evento basado en la covarianza del error de estima-

ción de la postura $P_{k,p}$, que resulta muy adecuado para indicar cuando el robot estima adecuadamente su postura y cuando debe realizar una corrección con la información global.

- Se escogió una representación geométrica de $P_{k,p}$ con el fin de poder realizar la definición del nivel límite del evento $R_{A,lim}$ de forma intuitiva, para lo cual se decidió utilizar la definición de los elipsoides $n\sigma$. De esta forma se definió el evento como el área de los elipsoides 3σ normalizada con el área del robot, con lo que se realiza la actualización por eventos cuando el área del elipsoide 3σ (confianza del $98,9\,\%$) de $P_{k,p}$ supera un número determinado de áreas del robot, siendo una forma sencilla e intuitiva de definir el nivel del evento $R_{A,lim}$ con el que se pueden solucionar los problemas de disponibilidad de la medición global.

- Al incorporar la corrección global basada en eventos y la definición de R_A basada en el área normalizada de los elipsoides 3σ en los esquemas en cascada basados en el tiempo, se obtuvieron los algoritmos de fusión basados en eventos (algoritmos 9 al 12), adecuados para la mejora de la localización en plataformas de recursos limitados al realizar la actualización con la información global únicamente cuando es necesario (si la incertidumbre en la postura supera un límite definido mediante $R_{A,lim}$).

- Finalmente se estableció un evento adicional para navegación en exteriores, en el cual se realiza la corrección global únicamente si $R_A > R_{A,lim}$ y el número de satélites utilizados por el sensor GPS $\boldsymbol{N}_{sat}$ es mayor a un valor mínimo $\boldsymbol{N}_{sat,min}$ preestablecido, con lo que se obtiene una mayor precisión en la medición global y por lo tanto en la fusión sensorial.

7 | Fusión Sensorial por Eventos en Localización Cooperativa

Los métodos de localización propuestos en el capítulo anterior son adecuados cuando se utiliza un único robot, que navega en el entorno ejecutando diversas tareas para cumplir con su misión. Sin embargo, existen casos en los que solo un robot no es suficiente y se requiere utilizar un grupo de robots para poder cumplir con la misión, o bien para realizarla de una mejor forma o en un menor tiempo. Cualquiera que sea la misión a realizar, la localización es un problema fundamental en equipos de robots ya que en la mayoría de aplicaciones se requiere una precisión alta en su estimación.

La localización del grupo de robots se podría realizar utilizando los algoritmos del capítulo anterior de forma individual, con lo que cada integrante del equipo estimaría su postura utilizando sus propios recursos y sensores, y la compartirá con otros miembros utilizando el medio de comunicación en caso de ser necesario. Sin embargo, una mejor opción es equipar a cada miembro del grupo con un sensor *relativo* (un sensor de barrido láser, infrarrojos, ultrasonido, cámara, etc.), que identifica a cada integrante del grupo cercano al robot y obtiene su distancia y orientación relativa.

Utilizando la postura y covarianza de cada robot vecino detectado (recibidas por el medio de comunicación) junto con la medición relativa, cada robot puede estimar su postura mediante métodos geométricos, al considerar a cada vecino como una baliza móvil de la cual se conoce su posición, covarianza, distancia relativa y orientación relativa (entre el robot y la baliza). Con este procedimiento, denominado como *Localización Cooperativa*, se logra agregar un sensor adicional de postura a cada robot (comunicaciones + sensor relativo). Éste puede ser incorporado dentro del filtro de fusión sensorial para mejorar la precisión de la estimación de la postura, la cual será mayor a la obtenida cuando cada miembro estima su postura de forma independiente [146].

Además del incremento en la precisión, el esquema cooperativo tiene otras ventajas. Permite, por ejemplo, utilizar una solución distribuida para realizar la localización del grupo, por lo que no se requiere un centro de fusión que calcule todas las posturas y las transmita a cada integrante, lo que disminuye el costo de implementación al requerir menos componentes, además de proveer mayor flexibilidad de movimiento al grupo. Por otra parte, en el caso de grupos heterogéneos en donde no todos los integrantes tienen las mismas capacidades, sino que se tienen miembros avanzados con gran cantidad de sensores disponibles, así como miembros con capacidades limitadas disponiendo de pocos sensores locales y en algunos casos sin capacidad de localización global, el esquema cooperativo permitirá distribuir la información global de los robots avanzados a los robots de recursos limitados mediante la medición relativa y la comunicación entre los integrantes. De esta forma, estos robots incrementarán su precisión en la localización a pesar de no disponer de un sensor global.

Estas características del esquema cooperativo son esenciales en muchas de las aplicaciones actuales de grupos multirobot, en donde se requiere una gran precisión en la localización. Por ejemplo, en coordinación de equipos submarinos [16, 52], en grupos de robots aéreos y/o terrestres ejecutando misiones cooperativas [117, 41, 39, 166, 5] o en exploración de entornos desconocidos [142, 62, 143, 144].

Se puede apreciar que la utilización del ancho de banda del medio de comunicación (Bluetooth, Wi-Fi, ZigBee, etc.) es fundamental en el esquema cooperativo, por lo que si el robot requiere su utilización extensiva para otras tareas distintas a la localización cooperativa como es el caso, por ejemplo, de la coordinación y control de navegación y formación del grupo, de los enlaces de video en tiempo real, además de la supervisión, monitoreo y diagnóstico de la plataforma; se deberá regular su utilización para permitir la transmisión de la información de las distintas tareas. De la misma forma, si el robot es de recursos limitados, no podrá realizar la transmisión de todos los datos requeridos en el esquema cooperativo (posturas, matrices de covarianza, mediciones relativas, etc.) en cada instante de muestreo, con lo cual se debe limitar el uso del ancho de banda para poder ejecutar el método de localización.

Según se expuso en los antecedentes, la mayoría de los métodos de localización multirobot existentes requieren la transmisión de gran cantidad de información en cada instante de muestreo, además de utilizar de forma intensiva el ancho de banda de la plataforma. Por esta razón son métodos inadecuados cuando el acceso al medio de comunicación es limitado. Algunos métodos existentes consideran algún tipo de limitación en el uso del ancho de banda [145, 125, 158, 103, 137] al reducir en alguna medida la cantidad de información transmitida o la frecuencia de intercambio de mensajes. Sin embargo, estos métodos requieren la comunicación constante entre miembros del grupo por lo que no permiten la incorporación o supresión dinámica de algún integrante del equipo con movimiento libre en el entorno (cuando un robot entra o sale del rango de detección del sensor relativo), o bien, cuando se requiere utilizar una frecuencia de comunicación variante en el tiempo (debido al uso del ancho de banda por otras tareas). Además, estos métodos requieren gran cantidad de memoria y recursos computacionales, lo que impide su implementación en plataformas de recursos limitados.

Por estas razones, los métodos basados en eventos desarrollados en el capítulo anterior pueden aplicarse al caso multirobot para obtener un algoritmo de fusión cooperativo y distribuido, que permita mejorar la localización de cada miembro del grupo, utilizando de forma eficiente los recursos computacionales y de comunicación de la plataforma a la vez que se obtiene un desempeño adecuado. Para esto se utilizará la actualización basada en eventos para incorporar la información relativa en la fusión sensorial. De esta forma, se limitará el uso del ancho de banda empleado por cada integrante del grupo, al realizar la transmisión de la información para el método cooperativo cuando un evento, definido con base en la información local del robot, supere un nivel predeterminado.

Tomando en consideración estos factores, se expone a continuación el desarrollo del método cooperativo multirobot basado en eventos para grupos de robots heterogéneo. Se describe primeramente la definición del grupo de robots con localización cooperativa y su red de comunicación, así como el modelo que permite incorporar las mediciones relativas en el esquema de fusión. Posteriormente se exponen el algoritmo cooperativo con actualización

basada en el tiempo, el cual es utilizado como punto de partida para obtener el método cooperativo con actualización relativa basada en eventos.

7.1 Grupo de robots con localización cooperativa

Se expone a continuación la definición del esquema multirobot cooperativo al delimitar las condiciones requeridas para conformar el grupo multirobot y la red de comunicación, utilizada para transmitir la información relativa del filtro de localización. Se define además el modelo de medición relativa que relaciona las distancias, ángulos y posturas de los robots vecinos con la postura del robot local.

7.1.1 Grupo multirobot cooperativo

Las siguientes condiciones definen el grupo multirobot considerado en la presente tesis:

1. Un grupo de robots G_R consiste en un número N de miembros (integrantes) denotados como R_i, con $i = 1, \ldots, N$ y $N \geq 2$ desconocido a priori por cada integrante. Esto implica la necesidad de un ajuste dinámico en las ecuaciones de fusión "tradicionales" dependiendo del número de robots vecinos detectados por cada R_i en cada instante k.

2. En cada $R_i \in G_R$ su postura es definida con su posición (x, y) y orientación θ respecto a los ejes de referencia global (X_G,Y_G), los cuales son comunes a todos los miembros en G_R. Cada R_i incorpora un eje de referencia local (X_L,Y_L) que enmarca la medición de las velocidades locales (v,ω). No se consideran los casos de localización relativa (localización respecto a un eje local de algún R_i) o de múltiples ejes globales en G_R.

3. Cada $R_i \in G_R$ puede tener distintos sensores, locomoción, rango de detección D_r (distancia máxima a la que se puede detectar un vecino), rango de comunicación C_r (distancia máxima en la que se puede mantener la comunicación con un robot vecino) y capacidad de cómputo y comunicación (recursos computacionales y ancho de banda). Por esta razón, G_R se considera un grupo heterogéneo que puede dividirse

en dos subconjuntos: G_{RA} de robots avanzados y G_{RL} de robots con recursos limitados.

4. Las mediciones locales son obtenidas por cada $R_i \in G_R$ y pueden incluir las velocidades lineal v y angular ω desde los encoders $(v, \omega)_{enc}$, ω_{gyr} de un giróscopo, ω_{comp} de una brújula, además de las aceleraciones lineal a y angular α de los acelerómetros disponibles en la plataforma (u_1, u_2, u_3 en el caso con deslizamiento), colocados de acuerdo al tipo de robot.

5. Las mediciones globales $L_{GM/GPS} = (x, y, \theta)_{GM/GPS}$ se obtienen del sensor global (cámara cenital, GPS, etc.) según el entorno de navegación del grupo. Estos sensores están disponibles principalmente en G_{RA}.

6. Al menos un miembro R_i tiene acceso a $L_{GM/GPS}$ de forma que se proporcione convergencia global al grupo [145], esta cantidad puede ser superior según el tamaño del grupo (N).

7. Las mediciones relativas M_r se realizan en cada $R_i \in G_R$ por un sensor que obtiene la distancia (rango) ρ_{ij}, el ángulo relativo (orientación relativa) φ_{ij} y la identidad M_j de cada robot vecino R_j dentro del rango de detección D_r (constante) del robot R_i. Se considera en la presente tesis que la medición relativa se puede realizar en toda la circunferencia del robot, con lo que el rango de detección del sensor relativo forma un círculo de radio D_r con centro en la posición del sensor sobre la plataforma móvil (por ejemplo en el centro geométrico o de masa del robot).

8. El "*Matching*" es el proceso mediante el cual se asigna la identidad del vecino M_j con la correspondiente medición del sensor relativo local, $(\rho, \varphi)_{ij}$. Ésta se realiza en general por el sensor relativo (cámara, lector de etiquetas RFID, etc.), o bien se puede realizar al comparar la información de la postura del robot R_j (identidad M_j) recibida por comunicación con la postura determinada por el sensor relativo de R_i.

9. No se consideran errores de asignación (error de *Matching*) ya que $\forall R_i$ puede diferenciar entre un miembro del equipo G_R y un obstáculo estático o dinámico en el entorno.

10. Como D_r puede ser distinto entre los miembros del grupo, se pueden dar distintos casos de detección. Por ejemplo, un robot R_i con D_r alto detectará y se comunicará con un robot vecino R_j de D_r bajo aunque R_j no pueda observar a R_i. Estos casos se definen de acuerdo al modelo de medición relativa expuesto en la sección 7.1.3.

11. Cada R_i estima su postura L_k utilizando todas las mediciones disponibles (locales, globales y relativas) y empleando comunicaciones para realizar el intercambio de información cooperativo con los robots dentro de D_r. El algoritmo de fusión selecciona cuales mediciones debe realizar en cada instante k según el esquema basado en eventos.

12. Cada R_i se mueve a voluntad según su misión específica (de forma individual, simultánea, continua, etc.), pero no requerirá detener su movimiento para comunicarse, realizar las mediciones relativas o bien ejecutar el algoritmo de localización cooperativa (a diferencia de algunos métodos existentes como el expuesto en [29]).

13. Los miembros de G_R no están limitados a mantener una formación rígida, aunque pueden realizar formaciones temporales de subconjuntos o del grupo completo en caso de requerirse.

14. Se considera un entorno dinámico (robots y obstáculos en movimiento), por lo que las frecuencias con las que se detectan los robots vecinos son variantes en el tiempo y desconocidas a priori.

15. Cada R_i cuenta con un módulo de evasión de obstáculos (por ejemplo un algoritmo Braitenberg [150, 102]) y un algoritmo de planificación de trayectorias que determina la ruta que debe seguir el robot (evitando obstáculos y/o colisiones [107, 73, 108, 35, 111, 100, 98]).

16. Información global asociada a un mapa del entorno puede estar disponible o ser obtenida únicamente por los robots en G_{RA} si fuera necesario (de acuerdo a las capacidades de los robots en este subconjunto), sin

embargo, este caso SLAM cooperativo [58, 33, 126] no se contempla en la presente tesis.

17. Para cada robot en G_R se determina un modelo de acuerdo a su locomoción con el fin de utilizarlo en el algoritmo de localización y navegación, siguiendo por ejemplo los procedimientos propuestos en el capítulo 3. En el caso específico de las pruebas experimentales simuladas del capítulo 11, se utilizará el modelo diferencial de la ecuación (3.24) junto con las aceleraciones del modelo (3.34) (versión 3 partículas) el cual utiliza como entrada el modelo dinámico de las aceleraciones de las ruedas del robot de la ecuación (3.57).

7.1.2 Red de comunicación, arquitectura basada en agentes

Las comunicaciones entre los miembros del grupo multirobot se gestionan utilizando un enfoque multiagente [148]. De esta forma, el mecanismo utilizado para coordinar la comunicación entre los agentes $R_i \in G_R$, que permite realizar la localización cooperativa, se establece mediante el uso de la plataforma de desarrollo de agentes en el lenguaje JAVA, denotada como JADE [20] por sus siglas en inglés (Java Agent Development Framework – JADE).

Esta plataforma provee una implementación estándar del lenguaje de comunicación entre agentes (FIPA-ACL) que contiene la especificación para realizar el envío y recepción de mensajes entre miembros del grupo. La utilización de JADE permite dotar cada R_i de un sistema de gestión de mensajes, que realizará el intercambio de información según los principios que se detallan a continuación:

1. En el ámbito del problema de localización en un grupo multirobot cooperativo, el término "*comunicaciones*" se refiere a la transmisión de la información requerida para la localización cooperativa entre los miembros $R_i \in G_R$.

2. La información transmitida i_T (denotado con el subíndice T) por cada R_i incluye la postura estimada L_T, la covarianza del error de estimación de la postura $P_{T,p}$, las mediciones relativas $M_{T,r}$ (si el vecino se encuentra dentro de D_r), las velocidades en los ejes globales $(\dot{x}_T, \dot{y}_T, \dot{\theta}_T)$

y el tiempo requerido para realizar el envío de la información (si existiera, se determina mediante un temporizador activado al recibir una solicitud de información de un robot vecino). La información recibida i_R contempla las mismas variables (denotadas con el subíndice R) e incluirá $M_{R,r}$ según el D_r del vecino.

3. Los retardos en la trasmisión de la información causados por el medio de comunicación son conocidos y medibles, al conocer la distancia entre miembros y la velocidad de transmisión, o bien al utilizar un temporizador activado al realizar la llamada a un vecino y detenido al recibir su respuesta. Como alternativa para determinar el retardo puede utilizarse un módulo con reloj de tiempo real, si está disponible $\forall R_i \in G_R$ y se sincroniza de forma adecuada.

4. $C_r \geq D_r$ pero cada R_i iniciará la comunicación únicamente con los vecinos contenidos en D_r. El número total de R_i contenidos en D_r se denotará como N_D. Por esta razón, si un robot puede detectar a un vecino, también podrá establecer comunicaciones con él.

5. Cuando un R_i se comunica con un vecino, transmitirá su información i_T y esperará la recepción de i_R. Esto es gestionado mediante múltiples hilos de ejecución en la unidad de control de la plataforma, que previenen cualquier espera bloqueante o interbloqueo mientras se recibe el mensaje de respuesta de la comunicación (se consideran comunicaciones no bloqueantes en cada R_i).

6. El algoritmo de localización cooperativa (en adelante denotado como *CLA*) determina cuándo y con qué frecuencia el robot R_i se comunica con los vecinos contenidos en D_r. De esta forma, las comunicaciones en G_R no se llevan a cabo en cada instante de muestreo k (definido por el tiempo de muestreo T_s constante) o con una frecuencia de comunicación constante predefinida, siendo ésta variante en el tiempo.

7. El *CLA* es distribuido, ejecutado de forma local $\forall R_i \in G_R$, e iniciará las comunicaciones cuando un evento R_A, definido según $P_{k,p}$ (elipsoides 3σ, ecuación (6.7)) supera un nivel predeterminado $R_{A,lim}$. Hasta que el evento no supere $R_{A,lim}$, R_i no iniciará las comunicaciones con sus

vecinos, aunque estos se encuentren contenidos en D_r. Sin embargo, si un vecino solicita información (debido a que su R_A local superó su $R_{A,lim}$), el robot envía i_T aunque su $R_{A,lim}$ no haya sido superado.

8. Debido a que D_r puede ser distinto entre miembros, un R_i puede enviar i_T para "*ayudar*" a un vecino R_j si fuera necesario, aunque el sensor relativo de R_j no haya detectado a R_i (con lo que tampoco ha solicitado la información i_T). Esto es necesario cuando existen miembros del grupo con un D_r muy limitado, lo que causa que el robot no detecte rápidamente a un vecino para ejecutar la localización cooperativa, aunque se haya alcanzado el $R_{A,lim}$, con lo que la covarianza del error de estimación puede crecer considerablemente. Por el contrario, si este robot ha superado su $R_{A,lim}$ y es detectado por un vecino con D_r alto y que envía la i_T aunque no sea solicitada, se podrá utilizar esta información para mejorar la estimación de la postura, utilizando el modelo de medición relativo de la sección 7.1.3, evitando así el crecimiento excesivo del error en la localización.

   Este método "*altruista*" puede administrarse dentro de cada robot R_i con rango D_r amplio, al comprobar si al enviar i_T se recibe o no del vecino R_j un i_R que incluye M_r (lo que indica que el vecino R_j puede detectar al robot R_i con su sensor relativo). En caso de no recibir M_r, se agregaría el identificador del vecino R_j a una lista (en R_i), según la cual el robot R_i le enviaría i_T cada vez que detecte a R_j y aunque en R_i no se haya alcanzado el $R_{A,lim}$. Esto, a pesar de incrementar la cantidad de mensajes enviados en G_R, puede mejorar la precisión de la postura estimada por los robots con D_r muy limitado. Cabe destacar que en este caso, el robot R_j (al que se le ayuda) puede decidir si utilizar o no la información i_R según la definición del evento dentro de este robot (si se ha alcanzado o no su $R_{A,lim}$).

9. Pueden establecerse definiciones adicionales del evento R_A además del basado en el área de los elipsoides 3σ (ecuación (6.7)) según las necesidades específicas de la misión del grupo. Por ejemplo, puede definirse el evento de forma que se ejecute la fusión con la información relativa siempre que se detecte un nuevo robot vecino dentro de D_r (en cuyo

caso se contactaría una sola vez hasta que no salga de D_r y vuelva a ser detectado), o bien siempre que N_D sea mayor a un valor mínimo $N_{D,min}$ (por ejemplo, ejecutar la fusión relativa si hay 3 o más vecinos dentro de D_r, $N_{D,min} = 3$, $N_D \geq N_{D,min}$). Se puede; además, utilizar la información de la evitación de obstáculos para ejecutar la fusión relativa siempre que se evite un obstáculo y mejorar la precisión cuando se retoma la trayectoria deseada en el robot, o bien ejecutar el método cooperativo si el robot ha avanzado una distancia mínima (definida según la precisión de la estimación local del robot, obtenida de P_k). La selección del evento y de su valor límite se definen de acuerdo a las necesidades propias de cada R_i a la hora de implementar el método (se exponen algunos casos de interés en la sección 7.2).

10. Cuando se detectan múltiples vecinos dentro de D_r (por ejemplo $N_D > 3$), el algoritmo cooperativo puede determinar cuáles medidas utilizar en la fusión sensorial basándose en la covarianza del error de estimación recibida $P_{R,p}$ y la medición relativa M_r (recibida $M_{R,r}$ o propia según disponibilidad). Esto le permite al algoritmo *CLA* descartar la información que no es lo suficientemente precisa, basándose en los valores de la covarianza (mayor covarianza $\Rightarrow$ mayor incertidumbre $\Rightarrow$ menor precisión en la estimación de la postura), lo que puede mejorar la precisión del método de localización además de disminuir el uso de recursos computacionales (se calcularía la etapa de corrección relativa para un número menor de vecinos).

11. Se puede realizar la etapa de corrección relativa de forma secuencial calculando la etapa de corrección del filtro con dimensión unitaria (para un único vecino) y ejecutándola secuencialmente para cada vecino dentro de D_r. Esto puede disminuir el coste computacional al utilizar ecuaciones de menor dimensión además de facilitar la implementación del filtro cooperativo, esto se detalla en la descripción de *CLA* de la sección 7.2.

12. Es esencial proveer a cada R_i de la capacidad de almacenar localmente cada mensaje enviado y recibido, junto con el tiempo (local o del módulo de tiempo real) en el que se recibe cada i_R o se envía i_T, así

como la identidad del robot al que se envía o del que se recibe la información. Esto permite llevar un registro del desempeño del esquema cooperativo que puede utilizarse para comparar entre los esquemas con corrección temporal y basada en eventos. Este mismo tipo de registro debe establecerse en cualquier prueba experimental simulada con el fin de permitir la comparación entre esquemas de fusión.

Definido el grupo multirobot y las reglas de intercambio de mensajes, se describe a continuación el modelo de medición relativa requerido por el algoritmo *CLA*.

7.1.3 Modelo de medición relativa

Las mediciones relativas $M_r = \{\rho_{ij}, \varphi_{ij}\}$ se utilizan junto con la información i_R obtenida de los vecinos en D_r para obtener la medición de la postura, utilizando las relaciones geométricas entre M_r y cada L_R recibido. Existen diversos modelos de medición relativa [23, 112, 116, 138, 49, 59, 137] definidos de acuerdo al tipo de información que proporciona el sensor relativo (rango, ángulo, rango y ángulo, etc.). Siguiendo el enfoque propuesto en estos trabajos, se define el modelo relativo utilizando el caso de dos robots vecinos con postura $\{L_1(x_1, y_1, \theta_1), L_2(x_2, y_2, \theta_2)\}$ tal y como se muestra en la figura 7.1.

El robot R_1 ubicado en L_1 mide el rango ρ_{12} y ángulo relativo φ_{12}, de la misma forma el robot R_2 ubicado en L_2 obtiene ρ_{21} y φ_{21}. El ángulo relativo $[-\pi < \varphi \leq \pi]$ se mide en los ejes locales, por lo que desde los ejes globales se mide desde el ángulo actual de avance θ; si $\varphi = 0$ implica que el vecino se encuentra justo al frente del robot. La relación entre las posturas L_1 y L_2 con las distancias y ángulos relativos se muestra en la ecuación (7.1).

$$\begin{aligned} \rho_{12} &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} = \sqrt{(\Delta x)^2 + (\Delta y)^2} \\ \psi &= \varphi_{12} + \theta_1 = \operatorname{atan2}(\Delta y / \Delta x) \end{aligned} \tag{7.1}$$

El modelo de la ecuación (7.1) puede utilizarse directamente en los esquemas de fusión sensorial al utilizar su linealización, la cual aproxima la solución,

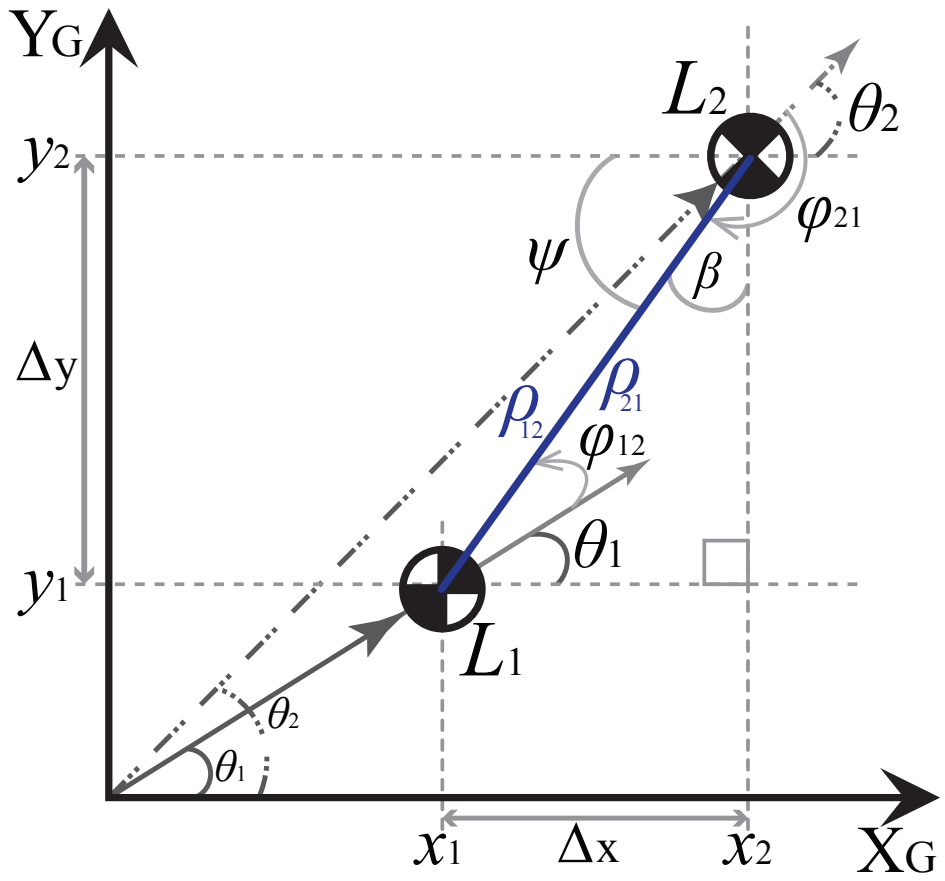


Figura 7.1: Descripción geométrica de la medición relativa M_r

o bien puede resolverse de forma exacta según el número N_D de vecinos R_j, D_r y la información i_R.

Cuando se detectan más de un R_j se pueden emplear técnicas similares a la triangulación o trilateración de balizas con posición conocida [23, 56, 49, 69, 134] para obtener la postura del robot desde la información relativa. Sin embargo, con el fin de obtener un método más general, éstos métodos *no* se consideran en la presente tesis, sino que se estudia el peor caso, el cual se produce cuando se detecta un único vecino R_j; ya que, como D_r puede ser distinto en cada robot, podría disponerse de menos información para determinar la postura de R_i.

Bajo esta perspectiva, si se detectara más de un vecino en D_r ($N_D > 1$), se puede utilizar la ecuación (7.1) múltiples veces dentro del esquema de fusión según N_D, ajustando correspondientemente el vector de medición z_k y la matriz covarianza de la medición R_k para incorporar la información relativa de todos los vecinos detectados (solución tradicional, esquema basado en el tiempo), o bien, puede incorporarse la ecuación (7.1) una única vez y realizar la actualización con la información relativa de forma secuencial (lo cual se utiliza en el método por eventos propuesto en la presente tesis).

De esta forma, cuando $N_D = 1$, se establecen tres casos básicos utilizando el ejemplo mostrado en la figura 7.1 para obtener la postura de R_1 a partir de la información recibida de R_2 y la medición relativa según los valores de D_r:

- **Caso 1** R_1 y R_2 se detectan mutuamente: En este caso, ambos robots se encuentra en rango cuando $D_1 \cong D_2$ y la posición de cada R_i permite la detección (en cuyo caso $\rho_{12} \leq \{D_1 \cong D_2\}$. En esta condición R_1 recibe $\{L_2, \rho_{21}, \varphi_{21}\}$ y se utilizan para obtener la postura completa L_1 según se muestra en la ecuación (7.2).

$$\begin{array}{ll} \psi = \pi - \varphi_{21} + \theta_2, & \theta_1 = \psi - \varphi_{12} \Rightarrow \psi = \varphi_{12} + \theta_1 \\ x_1 = x_2 - \rho_{12}\cos\psi & y_1 = y_2 - \rho_{12}\sin\psi \end{array} \tag{7.2}$$

- **Caso 2** Solo R_1 detecta a R_2: En este caso $D_1 > D_2$ y $\rho_{12} > D_2$. En esta condición R_1 recibe únicamente $\{L_2\}$ y se utiliza para obtener los

términos de la posición (x_1, y_1) en la postura L_1 pero no θ_1 según se muestra en la ecuación (7.3).

$$\begin{aligned} \psi &= \varphi_{12} + \theta_1, \\ x_1 &= x_2 - \rho_{12}\cos\psi \quad y_1 = y_2 - \rho_{12}\sin\psi \end{aligned} \tag{7.3}$$

- **Caso 3** Solo R_2 detecta a R_1: En este caso $D_1 < D_2$ y $D_1 < \rho_{21}$. En esta condición si R_2 se comunica con R_1 para *ayudarle*, entonces R_1 recibe únicamente $\{L_2, \rho_{21}, \varphi_{21}\}$ pero no se dispondrá de $\{\rho_{12}, \varphi_{12}\}$. Igual que en el caso anterior, con esta información únicamente se pueden determinar (x_1, y_1) en la postura L_1 pero no θ_1 según se muestra en la ecuación (7.4).

$$\begin{aligned} \psi &= \pi - \varphi_{21} + \theta_2, \\ x_1 &= x_2 - \rho_{21}\cos\psi \quad y_1 = y_2 - \rho_{21}\sin\psi \end{aligned} \tag{7.4}$$

Las ecuaciones (7.2) a (7.4) permiten incorporar la información relativa dentro del algoritmo de localización cooperativa para grupos heterogéneos según se describe a continuación.

7.2 Algoritmos de Localización Cooperativa

En esta sección se expone los algoritmos de localización cooperativa desarrollados en la presente tesis. Primeramente se expone el algoritmo de localización cooperativa basado en el tiempo (**TCLA**) el cual utiliza la información relativa en cada instante k junto con todas las mediciones locales y globales disponibles en la plataforma. Utilizando como punto de partida el *TCLA*, se desarrolla el algoritmo de localización cooperativa basado en eventos (**ECLA**) el cual utiliza la corrección basada en eventos desarrollada en el capítulo anterior para incorporar la información relativa, con el fin de reducir la utilización del ancho de banda en la plataforma.

Los algoritmos que se exponen a continuación están definidos para una plataforma de configuración diferencial cuyo modelo se establece en la ecuación (3.24) junto con las aceleraciones del modelo (3.34) (versión 3 partículas), el

cual utiliza como entrada las aceleraciones de las ruedas (a_R, a_L) que pueden ser obtenidas de los acelerómetros en la plataforma o bien del modelo de dinámico de las aceleraciones de las ruedas del robot (ecuación (3.57)). Cabe destacar que los algoritmos son generales y pueden ser adaptados a otras plataformas al utilizar los modelos correspondientes y al ajustar las dimensiones de los vectores de estado x_k, entrada u_k y medición z_k según se requiera.

Por otra parte, y para simplificar la descripción y notación del método cooperativo, en el presente capítulo se exponen los algoritmos considerando que las mediciones de la velocidad $z_{k,v}$ y de la postura global $z_{k,p}$ (si el robot tiene acceso a este sensor), se incorporan mediante un EKF (predicción y corrección) ejecutado en cada instante k (basado en el tiempo, tradicional), siendo este EKF intercambiable por los métodos expuestos en los capítulos 5 y 6 según las capacidades de cada miembro del grupo.

Por ejemplo, si el robot pertenece a G_{RA} es posible que realice la fusión mediante los algoritmos tradicionales (Algoritmos 1 y 2) o con asignación de entradas (Algoritmos 3 y 4) suponiendo que no hay problemas de disponibilidad en la medición, o bien al utilizar un algoritmo basado en eventos en cascada (algoritmo 9) de manera que se limite el acceso al sensor global y se tomen en cuenta los retardos temporales asociados, pero utilizando un nivel del evento $R_{A,lim}$ bajo, para acceder a la información global más frecuentemente que un robot de recursos limitados.

Por el contrario, para los robots pertenecientes a G_{RL}, se puede recurrir a los algoritmos en cascada reducido (Algoritmo 6) en caso de no tener problemas de acceso al sensor global, o al algoritmo en cascada con el modelo en cascada (algoritmo 8), el cual puede utilizarse además sin la corrección global en caso de no disponer del sensor global. De igual forma, pueden utilizarse los algoritmos en cascada por eventos (algoritmos 11 y 12) si se debe limitar el acceso al sensor global, en cuyo caso el $R_{A,lim}$ será mayor que para el caso de los integrantes de G_{RA}, con el fin de disminuir el acceso al sensor global y utilizarlo solo en caso necesario.

Por estas razones, en los algoritmos del presente capítulo se describe principalmente la etapa de corrección de la estimación mediante la información cooperativa (sensor relativo y comunicaciones). Esta etapa se puede agregar a conveniencia en los algoritmos propuestos en los capítulos 5 y 6 según las capacidades de cada miembro del grupo heterogéneo G_R.

En la ecuación (7.5) se exponen las definiciones de x_k, u_k y z_k a utilizar a lo largo del capítulo, de la que se observan los mismos supuestos del capítulo 6 (por eventos) en cuanto a las mediciones de velocidad local $z_{k,v}$ y postura global $z_{k,p}$, pero agregando la medición del sensor relativo $z_{k,r}$ obtenida de la medición relativa M_r y la información recibida mediante comunicación i_R según las ecuaciones del modelo relativo (ecuaciones de la (7.1) a la (7.4)).

$$\begin{aligned}
\boldsymbol{x}_{k,p} &= \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T, \quad \boldsymbol{x}_{k,v} = \begin{bmatrix} v_x & \omega \end{bmatrix}_k^T \\
\boldsymbol{x}_k &= \begin{bmatrix} x_{k,p} & x_{k,v} \end{bmatrix}^T \\
\boldsymbol{u}_k &= \begin{bmatrix} a & \alpha \end{bmatrix}_k^T \\
\boldsymbol{z}_{k,p} &= H_{k,p} x_{k,p} = \boldsymbol{L}_{GM} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_{GM}^T \\
\boldsymbol{z}_{k,v} &= H_{k,v} x_{k,v} = \begin{bmatrix} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T \\
\boldsymbol{z}_{k,r} &= H_{k,r} x_{k,p} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_r^T
\end{aligned} \tag{7.5}$$

7.2.1 Localización Cooperativa Basada en el Tiempo (*TCLA*)

El algoritmo de localización cooperativa basado en el tiempo (**TCLA**) utiliza la información relativa en cada instante k junto con todas las mediciones locales y globales disponibles en la plataforma $z_k = [z_{k,p} \quad z_{k,v}]$. Utilizando la definición de x_k, u_k y z_k de la ecuación (7.5) junto con el modelo del robot diferencial (ecuaciones (3.34),(3.24) y (3.57)) se constituye el EKF cooperativo basado en el tiempo mostrado en el algoritmo 13, el cual utiliza las covarianzas del proceso Q_k y medición R_k para realizar la estimación del estado x_k y la covarianza del error de estimación P_k.

El algoritmo 13 representa la aproximación "tradicional" de los métodos cooperativos en la utilización del modelo relativo (7.1), ya que se asigna una ecuación (7.1) $\forall R_j$ dentro de D_r detectado, por lo que si $N_D \geq 1$ se deberán ajustar dinámicamente las dimensiones de z_k, R_k y H_k en cada instante k, de acuerdo a N_D. Una vez ajustadas las dimensiones, se incorpora cada M_r realizado y cada i_R recibido en la etapa de corrección de la postura con la información relativa. Conviene recordar que el "matching" se ha realizado previamente (por el sensor relativo, por ejemplo) con lo que cada ecuación (7.1) agregada se utiliza correctamente con el correspondiente M_r e i_R de cada vecino.

El esquema *TCLA* del algoritmo 13 tiene toda la información disponible en el instante k, pero no puede implementarse en los robots pertenecientes a G_{LR} ya que requiere una inversión de matrices de dimensión alta (según z_k y N_D) para obtener la ganancia del filtro lo que requiere un uso extenso del procesador y memoria de la plataforma. Además, se requiere un uso considerable del ancho de banda ya que el robot debe comunicarse con todos los vecinos en D_r para obtener i_R cada instante k.

7.2.2 Localización Cooperativa Basada en Eventos (*ECLA*)

El algoritmo de localización cooperativa basado en eventos (**ECLA**) obtendrá y utilizará la información relativa para actualizar la estimación de la postura únicamente cuando un evento R_A supera su valor límite predefinido $R_{A,lim}$, con el fin de reducir los requerimientos en procesamiento, memoria y ancho de banda de la estrategia basada en el tiempo *TCLA*. Este evento indicará cuando el error local es considerable, definiendo $R_{A,lim}$ según la covarianza de la estimación de la postura, calculada en cada instante k por el filtro EKF/KF.

Como primer paso para incorporar la estrategia basada en eventos, se evitará la inversión y manipulación de matrices de orden alto (según N_D) del algoritmo 13, *TCLA*. Para lo cual se puede actualizar la estimación de la postura con la información relativa de forma secuencial, al considerar la fusión para un único vecino pero realizando la etapa de corrección N_D veces. Esto disminuye los requerimientos en procesamiento y memoria (inversa de

Algoritmo 13: Algoritmo EKF cooperativo, distribuido y recursivo, corrección basada en el tiempo: *TCLA*

Entrada: u_{k-1}=$[a\ \ \alpha]^T$,x_{k-1},P_{k-1}
Medición: $z_{k,p}$,$z_{k,v}$,M_r
Salida: $\hat{x}_k$,P_k
Datos: $f(\cdot)$ *y* $h(\cdot)$ *del modelo* (3.24),Q_k,R_k,i_R
Inicialización: x_0,P_0
Para *el instante actual* k **hacer**
  Predicción:
    $\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$
    $A_k = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1},u_{k-1},0}$, $W_k = \left.\frac{\partial f}{\partial w}\right|_{\hat{x}_{k-1},u_{k-1},0}$
    $P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$
  Actualización Cooperativa:
  Si $N_D \geq 1$ **entonces**
    $M_r = [\rho_{ij} \quad \varphi_{ij}]^T$, $j = 1 \cdots N_D$
    $z_k = \left[\begin{array}{ccc} z_{k,p} & z_{k,v} & M_r \end{array}\right]^T$
    Ajustar dimensión de R_k (según M_r y N_D)
    Agregar la ecuación (7.1) a $\boldsymbol{h()}$ (N_D veces)
  si no
    $z_k = \left[\begin{array}{cc} z_{k,p} & z_{k,v} \end{array}\right]^T$
    Utilizar R_k con $N_D = 0$
    Utilizar $\boldsymbol{h()}$ sin la ecuación (7.1)
  fin
  H_k=$\left.\frac{\partial h}{\partial x}\right|_{\hat{x}_k,0}$,$N_k = \left.\frac{\partial h}{\partial n}\right|_{\hat{x}_k,0}$
  Corrección:
    $K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$
    $\hat{x}_k = \hat{x}_k + K_k \left[z_{k,r} - h\left(\hat{x}_k, 0\right)\right]$
    $P_k = \left(I - K_k H_k\right) P_k$
fin

menor dimensión) aunque aun así, podría requerirse un tiempo de muestreo considerable para N_D altos según las características de la plataforma.

Como segundo paso, se considera la medición relativa como un sensor de postura adicional $z_{k,r} = \left[\begin{array}{ccc} x_k & y_k & \theta_k \end{array} \right]_r^T$. Siguiendo las propuestas del capítulo 5, se realiza la actualización de la estimación de la postura incorporando $z_{k,r}$ mediante una etapa de corrección KF en cascada, similar a la del algoritmo 6 (cascada reducido), la cual se ejecutará recursivamente si $N_D > 1$. El cálculo de $z_{k,r}$ se realiza de acuerdo al valor de D_r según los casos expuestos en las ecuaciones (7.2) a (7.4) utilizando los correspondientes M_r e i_R.

Como tercer paso, al utilizar $z_{k,r}$ en la fusión, se debe realizar la propagación de la covarianza asociada a M_r e i_r según el caso de D_r a través del modelo relativo con el fin de obtener la $R_{k,r}$ a utilizar en el filtro de fusión, para lo cual se procede con una estrategia similar a la utilizada en los filtros en cascada reducido (ecuación (5.42)).

De esta forma se realiza una aproximación lineal de $R_{k,r}$ mediante la ecuación (7.6), en donde $\nabla \boldsymbol{z}_u$ es el operador gradiente aplicado a las ecuaciones (7.2) a (7.4)) respecto a las *entradas* del modelo relativo según D_r (las cuales pueden incluir $x_2, y_2, \theta_2, \rho_{12}, \varphi_{12}, \varphi_{21}$), y R_{M_r,i_R} es la covarianza asociada a M_r e i_r según D_r (las entradas del modelo relativo). Conviene destacar que el operador gradiente realiza la linealización del modelo relativo (según D_r) respecto a las entradas del mismo, correspondiente a las variables de M_r e i_r, lo cual permite propagar la covarianza (de la entrada del modelo relativo u^* a la salida del mismo y^*) y obtener $R_{k,r}$, de forma similar a la ecuación (5.42), con la diferencia de que para $R_{k,r}$ la propagación en el tiempo no se realiza mediante la aproximación lineal (ecuación (7.6)), sino que se considera que ésta es realizada en las covarianzas asociadas a M_r (propias del sensor relativo) e i_r (recibidas por comunicación).

$$R_{k,r} = R_{x_1 y_1 \theta_1} = \nabla z_u R_{M_r,i_R} \nabla z_u^T \tag{7.6}$$

Al aplicar la ecuación (7.6) a las ecuaciones de la (7.2) a la (7.4) se obtiene $R_{k,r}$ para los distintos casos de detección tal y como se muestra en las

ecuaciones de la (7.7) a la (7.9) en donde se considera una matriz R_{M_r,i_R} diagonal. Como se observa de estas ecuaciones, $R_{k,r}$ es variante en el tiempo al depender de los valores de que se reciban en R_{i_R} en cada instante k, con lo que la ganancia del filtro que incorpora la medición relativa debe calcularse en cada instante k y N_D veces según la cantidad de vecinos detectados.

Caso 1:

$$
\begin{aligned}
u^* &= \begin{bmatrix} x_2 & y_2 & \theta_2 & \rho_{12} & \varphi_{12} & \varphi_{21} \end{bmatrix}^T, y^* = \begin{bmatrix} x_1 & y_1 & \theta_1 \end{bmatrix}^T \\
\nabla z_u &= \begin{bmatrix} 1 & 0 & 0 & -\cos(\varphi_{12}+\theta_1) & \rho_{12}\sin(\varphi_{12}+\theta_1) & 0 \\ 0 & 1 & 0 & -\sin(\varphi_{12}+\theta_1) & -\rho_{12}\cos(\varphi_{12}+\theta_1) & 0 \\ 0 & 0 & 1 & 0 & -1 & -1 \end{bmatrix} \\
R_{M_r,i_R} &= \begin{bmatrix} \sigma^2_{x_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y_2} & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{\theta_2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\rho_{12}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{\varphi_{12}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{\varphi_{21}} \end{bmatrix} = \begin{bmatrix} r_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & r_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & r_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & r_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & r_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & r_{66} \end{bmatrix} \\
R_{k,r} &= R_{x_1y_1\theta_1} = \nabla z_u R_{M_r,i_R} \nabla z_u^T = \begin{bmatrix} r_{1a} & r_{1b} & r_{1c} \\ r_{1d} & r_{1e} & r_{1f} \\ r_{1g} & r_{1h} & r_{1i} \end{bmatrix} \\
r_{1a} &= r_{55}\rho_{12}^2 \sin^2(\varphi_{12}+\theta_1) + r_{44}\cos^2(\varphi_{12}+\theta_1) + r_{11} \\
r_{1b} &= r_{1d} = -r_{55}\rho_{12}^2 \cos(\varphi_{12}+\theta_1)\sin(\varphi_{12}+\theta_1) \\
&\quad + r_{44}\cos(\varphi_{12}+\theta_1)\sin(\varphi_{12}+\theta_1) \\
r_{1c} &= r_{1g} = -r_{55}\rho_{12}\sin(\varphi_{12}+\theta_1) \\
r_{1e} &= r_{55}\rho_{12}^2 \cos^2(\varphi_{12}+\theta_1) + r_{44}\sin^2(\varphi_{12}+\theta_1) + r_{22} \\
r_{1f} &= r_{1h} = r_{55}\rho_{12}\cos(\varphi_{12}+\theta_1) \\
r_{1i} &= r_{33} + r_{55} + r_{66}
\end{aligned}
\tag{7.7}
$$

Caso 2:

$$
\begin{aligned}
u^* &= \begin{bmatrix} x_2 & y_2 & \rho_{12} & \varphi_{12} \end{bmatrix}^T, y^* = \begin{bmatrix} x_1 & y_1 & \theta_1 \end{bmatrix}^T \\
\nabla z_u &= \begin{bmatrix} 1 & 0 & -\cos(\varphi_{12}+\theta_1) & \rho_{12}\sin(\varphi_{12}+\theta_1) \\ 0 & 1 & -\sin(\varphi_{12}+\theta_1) & -\rho_{12}\cos(\varphi_{12}+\theta_1) \end{bmatrix} \\
R_{M_r,i_R} &= \begin{bmatrix} \sigma^2_{x_2} & 0 & 0 & 0 \\ 0 & \sigma^2_{y_2} & 0 & 0 \\ 0 & 0 & \sigma^2_{\rho_{12}} & 0 \\ 0 & 0 & 0 & \sigma^2_{\varphi_{12}} \end{bmatrix} = \begin{bmatrix} r_{11} & 0 & 0 & 0 \\ 0 & r_{22} & 0 & 0 \\ 0 & 0 & r_{33} & 0 \\ 0 & 0 & 0 & r_{44} \end{bmatrix} \\
R_{k,r} &= R_{x_1y_1\theta_1} = \nabla z_u R_{M_r,i_R} \nabla z_u^T = \begin{bmatrix} r_{1a} & r_{1b} \\ r_{1c} & r_{1d} \end{bmatrix} \\
r_{1a} &= r_{44}\rho_{12}^2 \sin^2(\varphi_{12}+\theta_1) + r_{33}\cos^2(\varphi_{12}+\theta_1) + r_{11} \\
r_{1b} &= r_{1c} = -r_{44}\rho_{12}^2 \cos(\varphi_{12}+\theta_1)\sin(\varphi_{12}+\theta_1) \\
&\quad + r_{33}\cos(\varphi_{12}+\theta_1)\sin(\varphi_{12}+\theta_1) \\
r_{1d} &= r_{44}\rho_{12}^2 \cos^2(\varphi_{12}+\theta_1) + r_{33}\sin^2(\varphi_{12}+\theta_1) + r_{22}
\end{aligned}
\tag{7.8}
$$

Con estos pasos, se incorpora la etapa de corrección basada en eventos, para lo cual se utiliza el evento definido en el capítulo 6, ecuación (6.7), con lo que se obtiene y utiliza la información relativa para mejorar la estimación de la postura del robot si el evento R_A, definido según el área de los elipsoides 3σ (calculada a partir de $P_{k,p}$) normalizada con el área A_{Ri} del robot R_i, supera un nivel predeterminado $R_{A,lim}$. Este límite se escoge como un compromiso entre el número de consultas realizadas entre robots (del que depende en uso del ancho de banda, consumo de energía y tiempo de cómputo) y la precisión deseada de la postura estimada (la cual puede incrementarse al realizar un mayor número de actualizaciones).

Caso 3:

$$
\begin{aligned}
u^* &= \begin{bmatrix} x_2 & y_2 & \theta_2 & \rho_{21} & \varphi_{21} \end{bmatrix}^T, y^* = \begin{bmatrix} x_1 & y_1 & \theta_1 \end{bmatrix}^T \\
\psi &= \pi - \varphi_{21} + \theta_2 \\
\nabla z_u &= \begin{bmatrix} 1 & 0 & \rho_{21}\sin(\psi) & -\cos(\psi) & -\rho_{21}\sin(\psi) \\ 0 & 1 & -\rho_{21}\cos(\psi) & -\sin(\psi) & \rho_{21}\cos(\psi) \end{bmatrix} \\
R_{M_r,i_R} &= \begin{bmatrix} \sigma^2_{x_2} & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y_2} & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{\theta_2} & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\rho_{21}} & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{\varphi_{21}} \end{bmatrix} = \begin{bmatrix} r_{11} & 0 & 0 & 0 & 0 \\ 0 & r_{22} & 0 & 0 & 0 \\ 0 & 0 & r_{33} & 0 & 0 \\ 0 & 0 & 0 & r_{44} & 0 \\ 0 & 0 & 0 & 0 & r_{55} \end{bmatrix} \\
R_{k,r} &= R_{x_1 y_1 \theta_1} = \nabla z_u R_{M_r,i_R} \nabla z_u^T = \begin{bmatrix} r_{1a} & r_{1b} \\ r_{1c} & r_{1d} \end{bmatrix} \\
r_{1a} &= r_{44}\cos^2(\varphi_{21} - \theta_2) + r_{33}\rho^2_{21}\sin^2(\varphi_{21} - \theta_2) \\
&\quad + r_{55}\rho^2_{21}\sin^2(\varphi_{21} - \theta_2) + r_{11} \\
r_{1b} &= r_{1c} = r_{33}\rho^2_{21}\cos(\varphi_{21} - \theta_2)\sin(\varphi_{21} - \theta_2) \\
&\quad - r_{44}\cos(\varphi_{21} - \theta_2)\sin(\varphi_{21} - \theta_2) \\
&\quad + r_{55}\rho^2_{21}\cos(\varphi_{21} - \theta_2)\sin(\varphi_{21} - \theta_2) \\
r_{1d} &= r_{44}\sin^2(\varphi_{21} - \theta_2) + r_{33}\rho^2_{21}\cos^2(\varphi_{21} - \theta_2) \\
&\quad + r_{55}\rho^2_{21}\cos^2(\varphi_{21} - \theta_2) + r_{22}
\end{aligned}
\tag{7.9}
$$

Utilizando esta definición de R_A, se establece el algoritmo de localización cooperativa basado en eventos *ECLA* tal y como se muestra en el algoritmo 14. De la misma forma que los algoritmos basados en eventos para robots individuales (capítulo 6), cuando se cumple la condición del evento ($R_A > R_{A,lim}$), el *ECLA* corregirá la estimación de la postura $x_{k,p}$ (obtenida con la información local y global) utilizando la medición relativa $z_{k,r}$ y la covarianza de medición $R_{k,r}$. La covarianza del error resultante $P_{k,p}$ decrecerá (de acuerdo a $R_{k,r}$) cuando se actualiza la postura, con lo que se reestablecerá el evento ($R_A < R_{A,lim}$) lo que inhabilita la corrección con la información relativa hasta que se vuelva a dar la condición del evento ($R_A > R_{A,lim}$).

Bajo el esquema de fusión con corrección basada en eventos, se obtendrá una matriz $P_{k,p}$ menor si los valores correspondientes a $R_{k,r}$ son bajos, por ejemplo, si el robot determina M_r mediante un sensor de precisión adecuada y si i_R se recibe de un vecino $R_j \in G_{RA}$ con acceso a la información global. Otro caso en el que $R_{k,r}$ tiene un valor reducido se produce cuando i_R es obtenida de un vecino que ha actualizado recientemente su postura, a partir de otro $R_j \in G_{RA}$ con acceso a la información global. De forma similar, si i_R proviene de un robot sin acceso a la información global (o perteneciente a G_{RL}) puede darse un decremento en $P_{k,p}$ (según $R_{k,r}$) pero de menor magnitud (comparado con el caso anterior), lo cual disminuirá R_A levemente. En este caso se alcanzará $R_{A,lim}$ con mayor frecuencia hasta que se realice la actualización utilizando una i_R con $R_{k,r}$ baja. De estos ejemplos se observa que el esquema de localización cooperativa permite el *flujo* de la información de localización entre miembros, los cuales tomarán ventaja de la información con bajo $R_{k,r}$ para mejorar su estimación de la postura, incrementando su precisión respecto al caso de localización individual. Además, este esquema localizará con una precisión adecuada a los miembros de G_{RL} que no tengan acceso a la información global de forma directa.

Como se definió en la descripción del grupo heterogéneo y su red de comunicación (sección 7.1.2) se pueden utilizar distintos tipos de eventos según las características del grupo cuya elección debe considerarse durante la implementación del grupo heterogéneo, en donde además se deben considerar los casos en los que se deban ajustar dos $R_{A,lim}$, uno para la etapa global y otro para la localización cooperativa. En este caso en particular, si el robot pertenece a G_{RA} y utiliza el esquema basado en eventos para limitar el ancho de banda, se puede priorizar uno de los sensores, por ejemplo al utilizar un $R_{A,lim,p}$ bajo para el sensor global (GPS por ejemplo) de forma que se usa frecuentemente y un $R_{A,lim,r}$ mayor en el caso del método cooperativo, con lo que se utilizará mayoritariamente el sensor global para corregir la postura y si por alguna razón se perdiera el acceso a éste, la covarianza crecerá lo suficiente como superar $R_{A,lim,r}$ con lo que se emplea el método cooperativo con lo que se reduce el error de estimación y su respectiva covarianza. También en el caso de los integrantes de G_{RA}, se puede considerar el caso de $R_{A,lim,p} = R_{A,lim,r}$ y asignar ambos con un nivel de evento bajo si no se

requiere restringir en gran medida la utilización del ancho de banda o alto en el caso contrario.

Por otra parte si el robot pertenece a G_{RL} es posible que únicamente tenga acceso a la localización cooperativa, en cuyo caso solo se requiere la asignación de $R_{A,lim,r}$ según la regulación requerida en el ancho de banda del robot, con lo que pueden asignarse valores altos para limitar en mayor medida el uso de medio de comunicación. En caso de que el robot disponga de acceso a ambos sensores puede darse prioridad a alguno de ellos al igual que en el caso G_{RA} o bien asignar $R_{A,lim,p} = R_{A,lim,r}$ a un nivel alto con el fin de restringir el uso del ancho de banda de la plataforma cuando se utilizan ambos sensores. En este caso en particular también pueden combinarse distintas definiciones del evento, al usar el evento basado en el área 3σ para controlar el acceso al sensor global y un evento basado en la cantidad de vecinos detectados para limitar el uso del método cooperativo.

Como se observa de estos ejemplos de posibles definiciones del $R_{A,lim}$, tanto el método basado en eventos como las respectivas definiciones del evento, propuestos tanto para robots individuales como para grupos de robots (sección 7.1.2), ofrecen una gran flexibilidad a la hora de implementar la localización en los distintos miembros del grupo tanto para integrantes de G_{RA} como de G_{RL}. De esta forma y a diferencia de los métodos existentes, se puede realizar la localización de grupos heterogéneos al utilizar el mismo algoritmo 14 distribuido, pero adaptando la definición del evento y de $R_{A,lim}$ según las capacidades de cada plataforma.

Finalmente conviene destacar algunos aspectos del funcionamiento del método cooperativo propuesto. Como se mencionó anteriormente, la actualización con la información relativa se puede realizar de forma secuencial cuando $N_D > 1$, sin embargo, como el robot recibe dentro de i_R la covarianza asociada a la postura de cada vecino R_j, el algoritmo *ECLA* podría escoger de entre las mediciones relativas, las mejores o la mejor, (por ejemplo, al determinar si se da el caso 1 de detección mutua, o al utilizar la de menor valor de covarianza de la postura de i_R, o al emplear la de rango ρ_{12} menor con los robots más cercanos, etc.) para realizar la corrección de la postura y de esta forma mejorar la precisión de la estimación del método cooperativo. Este

procedimiento es opcional y se puede implementar según lo requiera la plataforma, ya que con esto se podría conseguir un tiempo de cómputo menor del algoritmo (se utilizan menos recursos: procesador y memoria) permitiendo un mejor desempeño en los integrantes de G_{LR}, a la vez que se disminuye la incertidumbre (si se recibe un i_R con $R_{k,r}$ baja).

Otro aspecto a considerar en el esquema propuesto, es la falta de corrección para la orientación del robot en los casos de detección 2 y 3 (ecuaciones (7.3) y (7.4)), con lo que si algún integrante de G_{LR} posee un rango de detección D_r limitado, realizará en un menor número de ocasiones el caso de detección 1 (múltiple, ecuación (7.2)), con lo que la covarianza del error en la orientación θ puede incrementarse considerablemente. Por esta razón es recomendable incorporar un sensor de orientación absoluta (una brújula por ejemplo) en el caso de robots con D_r limitado, de forma que la incertidumbre en θ crezca de forma gradual (lentamente comparado a cuando no se tiene acceso a una brújula en la plataforma), la cual será corregida adecuadamente cuando se produzca el caso de detección 1. Como opción adicional, se podría considerar un caso especial en estas situaciones al incorporar algún método de triangulación ejecutado en los casos donde $N_D > 3$, con lo que se puede corregir θ aunque todos los N_D sean de los casos 2 y 3.

Tomando en consideración estos aspectos, el método de localización cooperativo basado en eventos que se ha propuesto en el presente capítulo tendrá la ventaja de ser flexible, en cuanto puede acoplarse la etapa cooperativa de corrección basada en eventos, para incorporar la información relativa, a los distintos métodos propuestos en los capítulos 5 y 6, según las capacidades propias de cada integrante del grupo heterogéneo. Además, es posible seleccionar distintos tipos de evento para realizar la corrección cooperativa según los requerimientos de cada $R_i \in G_R$, permitiendo contemplar distintos casos de $R_{A,lim}$ y D_r.

Expuesto el método cooperativo se procede con la definición de plataformas y aspectos de implementación con el fin de proceder con las pruebas experimentales de los métodos propuestos en los capítulos 5,6 y 7.

Algoritmo 14: Algoritmo EKF cooperativo, distribuido y recursivo, corrección basada en eventos: *ECLA*

Entrada: u_{k-1}=$[a\ \ \alpha]^T$,x_{k-1},P_{k-1}

Medición: $z_k = \begin{bmatrix} z_{k,p} & z_{k,v} \end{bmatrix}^T$,$M_r$,$i_R$

Salida: $\hat{x}_k$,P_k

Datos: *f y h del modelo* (3.24),Q_k,R_k,A_{Ri},$R_{A,lim}$,$H_{k,r} = I_{3\times3}$

Inicialización: x_0,P_0

Para *el instante actual k* **hacer**

Predicción:

$\hat{x}_k = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right)$

$A_k = \frac{\partial f}{\partial x}\Big|_{\hat{x}_{k-1},u_{k-1},0}$, $W_k = \frac{\partial f}{\partial w}\Big|_{\hat{x}_{k-1},u_{k-1},0}$, H_k=$\frac{\partial h}{\partial x}\Big|_{\hat{x}_k,0}$, $N_k = \frac{\partial h}{\partial n}\Big|_{\hat{x}_k,0}$

$P_k = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$

Corrección local y global:

$K_k = P_k H_k^T \left(H_k P_k H_k^T + N_k R_k N_k^T\right)^{-1}$

$\hat{x}_k = \hat{x}_k + K_k\left[z_k - h\left(\hat{x}_k, 0\right)\right]$

$P_k = \left(I - K_k H_k\right) P_k$

Actualización Cooperativa KF:

Área elipsoide 3σ: A_{ellip} (ecuación (6.7))

Calcular Evento: $\boldsymbol{R}_A = A_{ellip}/A_{Ri}$

Si $\boldsymbol{R}_A > R_{A,lim}$ **entonces**

Obtener M_r e i_R $\forall R_j$ en D_r

fin

Si *Se han obtenido* N_D *mediciones* M_r **entonces**

Realizar Matching de cada M_r con cada i_R

Opcional: Descartar las medidas de baja precisión o escoger la mejor

Obtener $z_{k,r}$ $\forall R_j \in D_r$ según el caso de D_r: ecuaciones (7.2) a (7.4)

Obtener $R_{k,r}$ $\forall R_j \in D_r$ según la ecuación (7.6) ((7.7) a (7.9))

Ejecutar la corrección relativa (secuencialmente si $N_D > 1$):

$K_{k,p} = P_{k,p} H_{k,r}^T \left(H_{k,r} P_{k,p} H_{k,r}^T + R_{k,r}\right)^{-1}$

$\hat{x}_{k,p} = \hat{x}_{k-1,p} + K_{k,KF}\left(z_{k,r} - H_{k,r}\hat{x}_{k-1,p}\right)$

$P_{k,p} = \left(I - K_{k,p} H_{k,r}\right) P_{k,p}$

fin

fin

7.3 Conclusiones del Capítulo

En el presente capítulo se propuso un algoritmo de localización cooperativa basado en eventos (*ECLA*, algoritmo 14) entre robots R_i pertenecientes a un grupo G_R heterogéneo, que navega en un entorno dinámico (con la presencia de obstáculos en movimiento):

- Se consideró que cada R_i dispone de un sensor *relativo* (un sensor de barrido láser, infrarrojos, ultrasonido, cámara, etc.), que identifica a cada integrante del grupo cercano al robot y obtiene su distancia y orientación relativa. Empleando esta información junto con la postura y covarianza de cada robot vecino cercano (recibidas por el medio de comunicación), cada robot estima su postura mediante un modelo relativo (ecuaciones (7.1) a (7.4)), al considerar a cada vecino como una baliza móvil de la cual se conoce su posición, covarianza, distancia relativa y orientación relativa (entre el robot y la baliza). De esta forma se logra agregar un sensor adicional de postura al cada robot (comunicaciones + sensor relativo) que es incorporado en el filtro de fusión, con lo que se consigue mejorar la precisión de la estimación de la postura, que resulta mayor a la obtenida cuando cada miembro estima su postura de forma independiente [146].

- El método propuesto es distribuido por lo que no se requiere un centro de fusión para realizar la estimación de las posturas de todo el grupo, lo cual conlleva un menor costo de implementación. Además, el esquema cooperativo para un grupo heterogéneo permite la distribución y "flujo" de la información global de los robots avanzados a los robots de recursos limitados mediante el sensor relativo y las comunicaciones, de forma que se mejora la precisión de la estimación de la postura en estos robots limitados a pesar de no disponer de un sensor global.

- Al utilizar las técnicas de fusión basadas en eventos propuestas, el algoritmo cooperativo distribuido (*ECLA*, algoritmo 14) realiza la fusión sensorial con la información relativa únicamente cuando $R_A > R_{A,lim}$, con lo que se limita el uso del ancho de banda de la plataforma (que puede ser requerido en otras tareas relevantes) al no realizarse el intercam-

bio de información relativa en cada instante k con todos los miembros del grupo. Este es el principal aporte del algoritmo propuesto respecto a los métodos existentes, ya que éstos hacen un uso intensivo del ancho de banda (en cantidad de datos transmitidos, frecuencia de comunicación y cantidad de vecinos a utilizar en el método) o de los recursos computacionales (memoria y procesador requeridos para cumplir con el tiempo de muestreo), siendo no aplicables en los integrantes de recursos limitados de un grupo heterogéneo con movimiento dinámico en el entorno (no restringido a una formación rígida).

- Se pueden incorporar los algoritmos propuestos en los capítulos 5 y 6 según las capacidades de cada miembro del grupo heterogéneo G_R permitiendo, además de la reducción del ancho de banda, limitar el uso de los recursos computacionales requeridos para la fusión sensorial en localización. Esto debido a que el *ECLA* (algoritmo 14) considera las mediciones relativas como un sensor de postura adicional (realizando la correspondiente propagación de covarianzas, ecuaciones (7.6) a (7.9)), el cual se incorpora como una etapa en cascada dentro del filtro que es ejecutada cuando se supera $R_{A,lim}$. Esto permite la integración recursiva de la medición relativa de cada vecino dentro de la fusión, evitando el ajuste dinámico de las matrices del filtro de Kalman y la inversión de matrices de gran dimensión requeridas en el método tradicional cooperativo con actualización temporal (*TCLA*, algoritmo 13)

- La definición del grupo multirobot y de su red de comunicación con arquitectura basada en agentes, establece un amplio rango de posibles aplicaciones del método cooperativo de grupos heterogéneos, considerando variaciones en las capacidades sensoriales de los integrantes del grupo (cantidad de sensores, rango de detección D_R, etc.). No se requiere además la detección de todos los vecinos en cada instante k o el mantener una formación rígida, permitiendo frecuencias de detección entre vecinos variantes en el tiempo (causadas por la presencia de obstáculos en el entorno o porque los vecinos salen del rango de detección).

- Se pueden realizar definiciones adicionales del evento R_A que pueden utilizarse en combinación con el evento basado en los elipsoides 3σ como por ejemplo, la fusión con un número mínimo de vecinos en D_r que permitiría incorporar métodos adicionales como los de triangulación dentro del esquema de fusión. Se pueden seleccionar además distintos tipos y niveles de eventos para incorporar la corrección global, utilizando distintos eventos o en caso de asignar el mismo tipo, definiendo los respectivos $R_{A,lim}$ para dar prioridad a alguno de los sensores (global o relativo) o bien dar la misma importancia a ambos. El valor de $R_{A,lim}$ será alto si se desea restringir en mayor medida el uso del ancho de banda, o bajo si se desea realizar la corrección de forma más frecuente, esto según las capacidades de la plataforma.

- El *ECLA* permite eliminar las mediciones relativas de baja precisión o escoger la mejor (robots más cercanos, mayor precisión de la postura recibida, etc.) para realizar la fusión sensorial, lo que puede disminuir en mayor medida $P_{k,p}$ a la vez que se realiza la fusión para un menor número de vecinos (requiriendo menos recursos computacionales). Como aspecto de implementación, si algún integrante es de recursos limitados y con rango de detección D_r limitado, es recomendable proveer acceso a un sensor global de orientación (una brújula por ejemplo) de forma que se limite el crecimiento en la incertidumbre de la orientación del robot, además de utilizar el método "altruista" (sección 7.1.2) para recibir la información relativa aunque no se detecten vecinos en D_r, de forma que se mejore la estimación evitando un crecimiento elevado en $P_{k,p}$.

8 | Plataformas y Aspectos de Implementación

En la actualidad existen gran variedad de plataformas experimentales [99] que permiten la implementación de los diversos métodos de localización propuestos en la presente tesis. De entre todas las opciones se ha escogido el LEGO NXT ®[85] para el desarrollo de los robots móviles utilizados en las pruebas de los robots individuales, al ser esta plataforma de bajo coste, con gran cantidad de sensores disponibles, de implementación sencilla y de recursos limitados, por lo que se pueden probar correctamente los métodos basados en eventos propuestos.

Empleando el LEGO NXT se proponen en el presente capítulo tres plataformas experimentales: dos para navegación en interiores (Diferencial y Ackerman) y una para navegación en exteriores (Ackerman), las cuales se describen a continuación, junto con el desarrollo realizado para permitir el acceso de los robots a los sensores globales (cámara y GPS respectivamente) utilizando el medio de comunicación disponible en el robot (Bluetooth y USB). Además, se expone el esquema de control general de navegación utilizado en las plataformas, junto con los parámetros del filtro de Kalman utilizados en las distintas pruebas (matrices Q y R). Finalmente se concluye con la presentación del simulador utilizado para las pruebas del algoritmo cooperativo, el cual utiliza los modelos identificados para los robots construidos con el LEGO NXT.

8.1 Navegación en interiores

En cuanto a navegación en interiores se describe a continuación las características de la plataforma LEGO con la que se construyen los robots Diferencial y Ackerman. Para ambos robots se expone la calibración de los sensores locales, el algoritmo de navegación y se describe el esquema de sensorización

global utilizado, el cual emplea una cámara cenital y la comunicación mediante Bluetooth para transmitir la postura global al robot. Finalmente se realiza la obtención de los parámetros del filtro de Kalman requeridos por los distintos algoritmos de fusión propuestos.

8.1.1 Robot Móvil LEGO NTX

LEGO Mindstorms ®NXT [85] es una plataforma móvil de bajo coste. Incorpora la unidad de control NXT, basada en un microcontrolador ARM7 de 32-bits, con 256 Kb FLASH y 64 Kb 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 versión básica del LEGO proporciona 4 tipos de sensores: contacto, luz, sonido y distancia. Pero, además de éstos, actualmente 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. [85]. Otros componentes LEGO muy importantes son los actuadores, los cuales consisten en motores de corriente continua con encoders integrados de 360º de resolución por revolución del eje del motor.

Existen múltiples opciones de Firmware y lenguajes para la programación del robot, de entre los cuales se ha optado en la presente tesis por el Firmware *LeJOS* [84] el cual permite la programación del NXT en el lenguaje *JAVA*. Esto debido a que LeJOS permite la implementación de múltiples hilos de ejecución (programación multithreading) en la unidad de control, además de ofrecer diversas librerías que implementan desde las funciones matemáticas básicas hasta el manejo de los sensores y actuadores del robot así como la administración del medio de comunicación Bluetooth, lo cual permite establecer el intercambio de mensajes entre el robot y un ordenador para supervisar la operación y funcionamiento del mismo, y para la comunicación entre robots.

Como se mencionó anteriormente, el LEGO NXT es una plataforma de recursos limitados, principalmente en cuanto a memoria disponible siendo capaz de incorporar únicamente 255 variables locales, 1024 constantes y una lon-

gitud máxima del código de programación de $64kb$ al ser programado con LeJOS. Además, al solo disponer de Bluetooth (clase II) para las comunicaciones inalámbricas, presentará retardos no despreciables en las comunicaciones entre robots o entre el robot y un ordenador (sensor global externo, supervisor, etc.) que dependerán de la cantidad de información a transmitir. Por estas razones, los robots desarrollados a partir de esta plataforma pueden beneficiarse del esquema basado en eventos propuesto en la presente tesis, tanto para permitir la implementación de los filtros de fusión sensorial en la plataforma (ya que no tiene suficiente memoria para implementar filtros de fusión con muchos estados debido al cálculo de la inversa de dimensión alta), como para limitar el uso de su ancho de banda al realizar las comunicaciones.

8.1.2 Robot móvil Diferencial y Ackerman

El robot diferencial construido utilizando el LEGO NXT se muestra en la figura 8.1. Los sensores a utilizar en el esquema de fusión se indican en esta figura y corresponden a dos acelerómetros ubicados en la plataforma justo sobre las ruedas del robot y alineados con los ejes de las mismas (de forma que se pueda utilizar el modelo dinámico por partículas, ecuaciones (3.22) y (3.34), figuras 3.3 y 3.4), además de un giróscopo, una brújula y los encoders de los motores del robot.

De forma similar, el robot Ackerman construido con el LEGO NXT se muestra en la figura 8.2. Nuevamente, los sensores a utilizar en el esquema de fusión se indican en esta figura y corresponden a dos acelerómetros ubicados en la plataforma justo sobre el centro de los ejes de las ruedas del robot (de forma que se pueda utilizar el modelo dinámico por partículas, ecuación (3.49) y figura 3.6), además de un giróscopo, una brújula y los encoders de los motores trasero y delantero del robot. Cabe destacar que se han agregado dos engranajes a la salida del motor delantero, para incrementar el rango de movimiento y permitir un mejor control de la dirección del vehículo. Además, se emplea un diferencial en la transmisión del motor trasero el cual permite transmitir el movimiento del motor a las ruedas traseras a la vez que se permite la rotación de ambas a distintas velocidades (siendo v_x el promedio de ambas), lo que permite el giro adecuado del robot al negociar una curva.

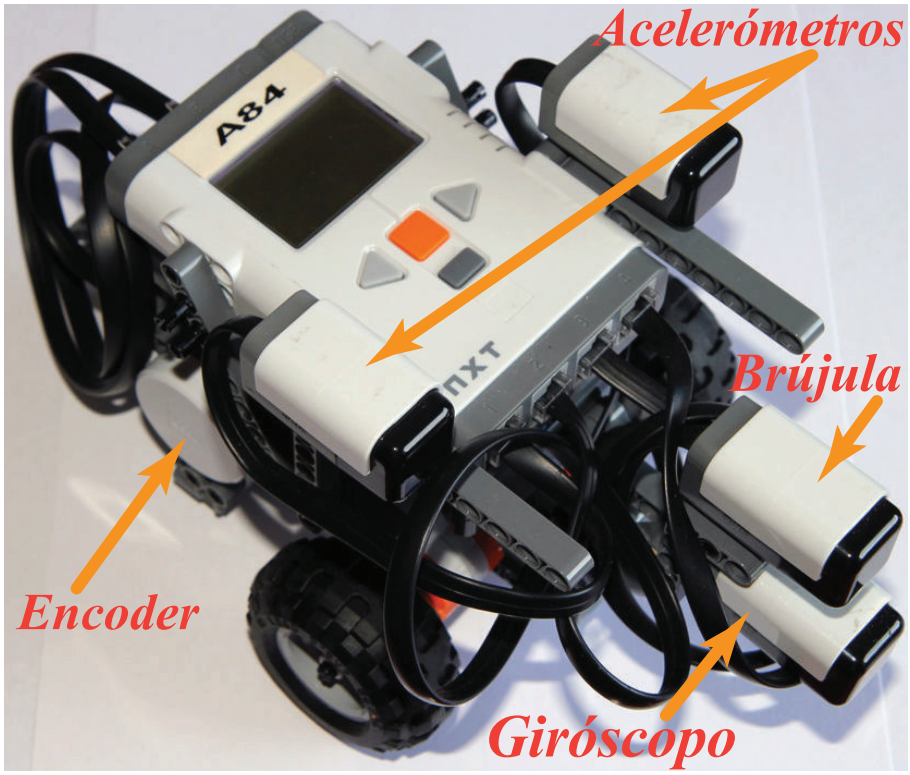


Figura 8.1: Robot móvil diferencial basado en el LEGO NXT

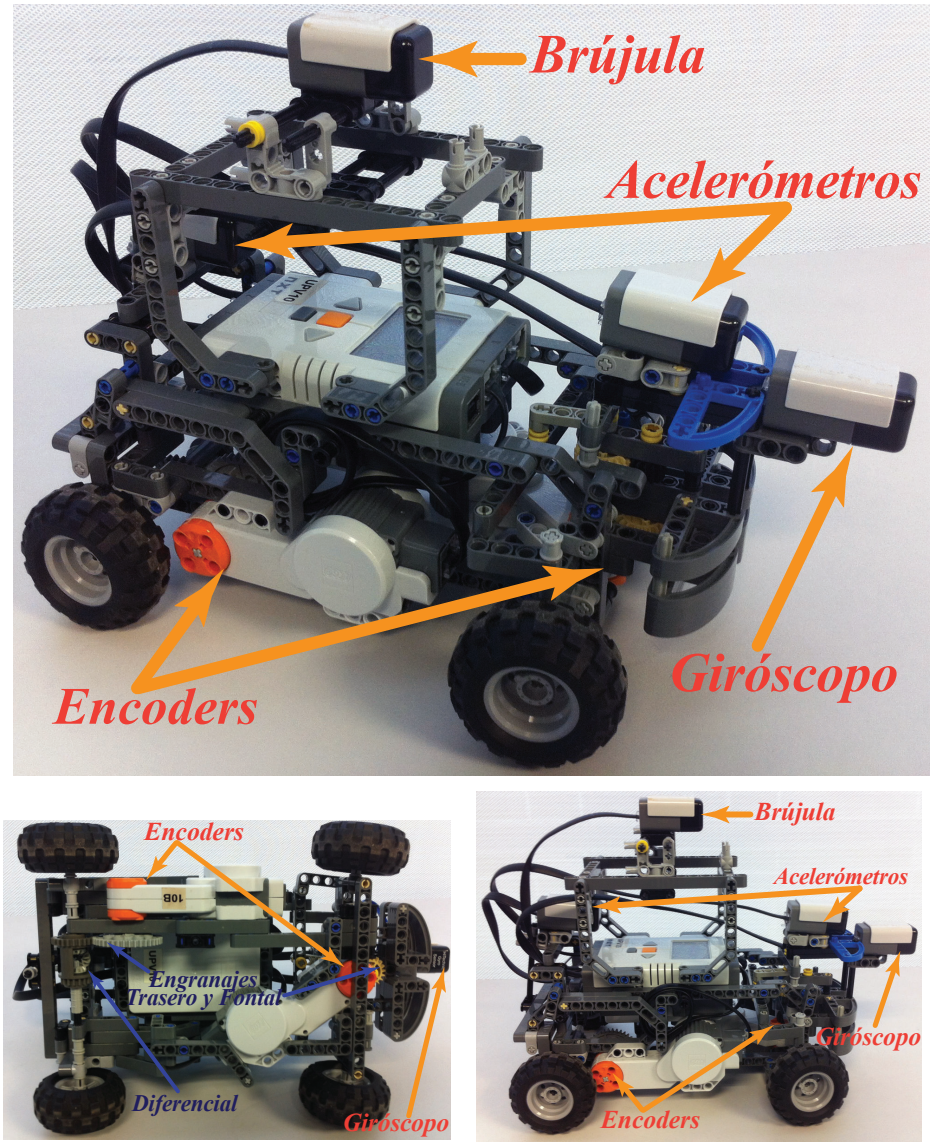


Figura 8.2: Robot móvil en configuración Ackerman basado en el LEGO NXT

8.1.3 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 los filtros de fusión, deben ser transformadas a este sistema. Además se debe remover cualquier «Bias» (si la salida del sensor no es cero cuando debería serlo) para lo cual se realiza una prueba inicial con los sensores de la plataforma midiendo un mismo valor durante un tiempo determinado (sin movimiento en el robot). Con esta prueba se puede determinar cualquier Bias presente en cada sensor, para poder restarlo y obtener el valor real de la variable que se desea medir. Este procedimiento debe realizarse siempre que se remplace un sensor, porque los valores del Bias cambian entre sensores aunque sean de un mismo fabricante. Con las señales calibradas se procede al preprocesamiento de los sensores que consiste en transformar la señal de salida de cada sensor a la variable que se utilizará en u_k o z_k en el esquema de fusión. En el caso de las plataformas desarrolladas se procede de la siguiente forma:

- Acelerómetros: No se requiere de preprocesamiento ya que se utiliza la salida (aceleración lineal) en los modelos de partículas propuestos, solo se realiza el cambio de unidades (de cantidad de aceleraciones de la gravedad g a m/s^2 o mm/s^2).

- Giróscopo: En este caso el sensor provee una medición de la velocidad angular ω_{gyr}, por lo que puede utilizarse directamente en los esquemas de fusión propuestos. Además podría integrarse y acumularse obtener la medición de la orientación robot θ_{gyr}.

- Brújula: La mayoría de estos sensores (también conocidos como magnetómetros) dan una medida absoluta de la dirección del norte magnético, ésta en general tiene un rango de medición entre los 0º y los 360º, por lo que no se puede incorporar de forma directa como una medición de ω_{comp} o θ_{comp}. Para esto se transforma la medición a una señal de evolución continua (sin limitarla a 360º), al comparar su valor en el instante anterior con el valor actual y al acumular su diferencia, teniendo en cuenta tanto el sentido de giro (horario y anti horario) como el cruce

de 0º a 360º o viceversa. Con esto se obtiene la medida de θ_{comp} la cual se deriva (dividiendo la diferencia entre el valor actual y el anterior por el tiempo de muestreo T_s) para obtener ω_{comp}.

- Encoders: Estos sensores tienen como salida un contador que indica la cantidad de incrementos realizados por la rueda (360 pulsos por vuelta para el LEGO NXT) y su preprocesamiento depende de la plataforma:

  - En el caso del robot diferencial, se realiza la resta del valor anterior al valor actual del contador con lo que se obtiene el desplazamiento angular de las ruedas para un periodo de muestreo, este se transforma a desplazamiento lineal utilizando un factor que depende del radio de las ruedas del robot y la cantidad de incrementos por vuelta. Al dividir este resultado por el tiempo de muestreo ($T_s = 50ms$ generalmente en el LEGO NXT) se obtiene la velocidad lineal de cada rueda (v_L,v_R) las cuales son utilizadas en la ecuación (3.13) para obtener v_{enc} y ω_{enc}. Cabe destacar que este método es sensible a variaciones o errores en la medición del radio de ruedas y de la separación entre éstas (b), por lo que ambos parámetros deben determinarse de forma precisa para mejorar la exactitud en las mediciones de v_{enc} y ω_{enc}. Por esta razón, se utiliza el procedimiento de calibración para el robot diferencial según se define en [121] para determinar estas distancias de forma precisa. Existen otros métodos disponibles [7, 46] pero se utiliza [121] debido a su sencillez y precisión.

  - En el caso del robot Ackerman se procede de la misma forma que para el robot diferencial para obtener v_x a partir del encoder del motor de la parte trasera del robot. Del encoder delantero integra la velocidad con $T_s = 50ms$ para obtener el ángulo ϕ, el cual se utiliza en la ecuación (3.39) para obtener ω_{enc}, con la que se aproxima $v_{y,enc} \cong v_x \tan\phi$. El procedimiento de calibración de [121] puede aplicarse únicamente para determinar el radio de las ruedas traseras de forma que se mejore la estimación de v_x.

Cabe destacar que tanto la calibración como el preprocesamiento dependen de los sensores y robots utilizados, por lo que se deben repetir ambos proce-

dimientos siempre que se cambie el robot utilizado o cuando se remplace un motor o sensor en los mismos. Además, la detección del Bias debe realizarse previo a cada prueba con las plataformas, lo cual puede ser realizado por una tarea dentro del robot, ejecutada antes de iniciar el movimiento del mismo. Con esto se tienen las entradas y mediciones requeridas en la fusión sensorial, con lo que se procede a exponer el algoritmo de navegación utilizado en las pruebas realizadas.

8.1.4 Algoritmo de navegación

Las pruebas experimentales se realizan utilizando el algoritmo de navegación mostrado en la figura 8.3 en donde $\Omega_k = \begin{bmatrix} \omega_L & \omega_R \end{bmatrix}^T_{ref}$ y $U_k = \begin{bmatrix} u_{Lw} & u_{Rw} \end{bmatrix}^T$ corresponden al vector de velocidades de referencia y al vector de la acción de control de los motores izquierdo y derecho respectivamente, definidos para el robot diferencial, y $\Omega_k = \begin{bmatrix} \omega_r & \omega_f \end{bmatrix}^T_{ref}$ junto con $U_k = \begin{bmatrix} u_r & u_f \end{bmatrix}^T$ (motor trasero y delantero) se definen para el robot Ackerman.

En el algoritmo de navegación, se define una trayectoria a seguir (un cuadrado, un círculo, etc.) que es almacenada en la memoria del robot como un conjunto de puntos globales (objetivo) que el robot debe seguir. A partir de la postura inicial del robot (la cual es conocida), el algoritmo obtiene la distancia del robot a los puntos objetivo junto con la distancia mínima $D_{min,ref}$, que indica cuál punto es el más cercano al robot, $(x, y)_{ob}$, al cual se le agrega una distancia constante, denominada como distancia *look ahead*. El punto $(x, y)_{ob}$ es utilizado por el algoritmo de navegación (persecución pura [42] o punto descentralizado [150, 35, 98, 121]) para determinar Ω_k. Esta referencia es seguida por el control PID de los motores del robot, el cual genera la acción de control $\boldsymbol{U}_k$ requerida para seguir la trayectoria e intentar alcanzar $(x, y)_{ob}$. En el instante de muestreo siguiente, los sensores obtienen las mediciones requeridas para el filtro de fusión que estima la postura actual del robot $x_{k,p}$, la cual es utilizada nuevamente en el algoritmo de navegación para obtener un nuevo punto objetivo que el robot debe alcanzar. De esta forma se logra mantener el robot en la trayectoria deseada.

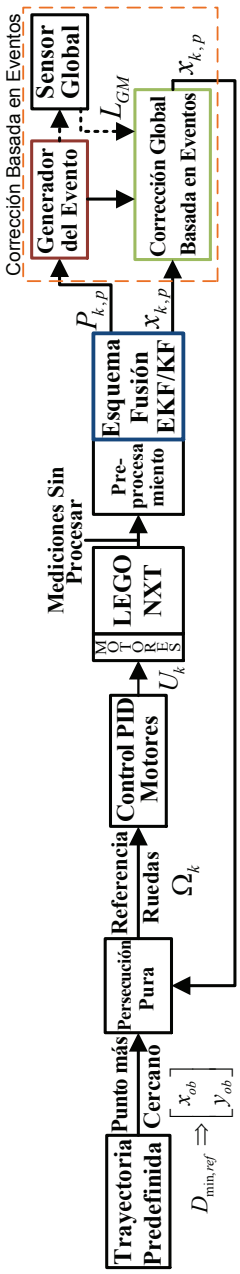


Figura 8.3: Algoritmo de navegación y localización basada en eventos

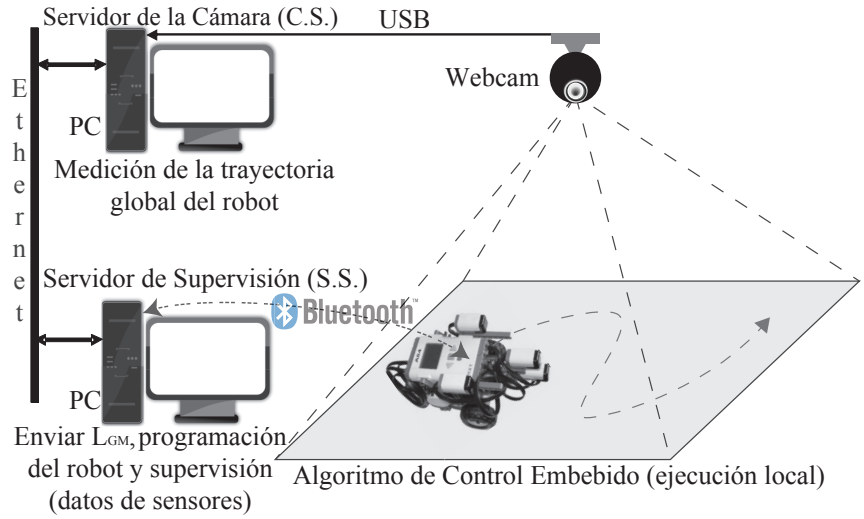


Figura 8.4: Configuración del sistema de acceso al sensor global, medición de la postura mediante una cámara cenital

8.1.5 Esquema de sensorización global

El sistema de acceso al sensor global se muestra en la figura 8.4, el cual está compuesto por una cámara (640x480, 30fps) en posición cenital, conectada a un ordenador que realiza el procesamiento de la imagen para obtener la postura del robot, denominado como Servidor de la Cámara (C.S.), el cual transmite la postura a un Servidor de Supervisión (S.S.), encargado de comunicarla al robot. El C.S. ejecuta un programa basado en JAVA que obtiene constantemente la imagen de la cámara y determina (mediante el preprocesamiento de ésta) la postura global del robot L_{GM}, junto con el tiempo que utilizó el servidor para obtenerla T_{cam} (50ms en promedio, pero se envía el valor real medido). Ambas variables forman el mensaje que es transmitido al S.S. para su distribución al robot o robots móviles. Se almacena además en el C.S. un video con la captura del movimiento del robot además del registro de posturas y tiempos para su análisis posterior.

En los algoritmos propuestos, el retardo de comunicación ΔT se asume conocido o medible, por lo que puede obtenerse como un valor promedio de varios experimentos e introducir este valor constante dentro de los algoritmos como una aproximación numérica al valor real de ΔT. Una mejor solución es utilizar la medición del valor real de ΔT cada vez que se utiliza la información global en el esquema de fusión, lo cual puede mejorar la precisión de la estimación. En el caso del esquema de la figura 8.4, cada vez que un evento se genera dentro del robot ($R_A > R_{A,lim}$, por ejemplo), éste solicita al S.S. la medición global utilizando las comunicaciones por Bluetooth y utiliza la estimación local (utilizando los sensores locales) del filtro de fusión hasta que el mensaje desde S.S. se recibe. Además, se almacenan en la memoria del robot las velocidades en los ejes globales del robot Vx_e, Vy_e en el instante en donde se supera el límite del evento. Debido a que el S.S. ejecuta un programa en Java que escucha continuamente a la espera de la llamada del robot que solicita la postura global, al recibir esta solicitud la procesa de forma inmediata, al realizar el envío al robot del último mensaje procedente del C.S. que contiene L_{GM} y T_{cam}.

Al recibir este mensaje, el robot ha realizado la medición del tiempo T_{ss} que ha tardado su petición (desde la solicitud hasta la llegada del mensaje) mediante un temporizador interno. De esta forma, el robot puede utilizar los tiempos T_{ss} y T_{cam} para obtener el retardo de comunicación ΔT, el cual se utiliza (junto con Vx_e, Vy_e) para desplazar L_{GM} desde el instante en donde se obtuvo hasta el instante actual, para su utilización por el esquema de fusión basado en eventos según se expone en la ecuación (8.1), en la cual se considera que L_{GM} se obtiene justo al recibirse la solicitud en el S.S. y que el retardo de envío y recepción entre el robot y el S.S. son iguales.

Por consiguiente, se utiliza en la ecuación (8.1) el ΔT para calcular el desplazamiento $\Delta(x, y)$ causado por el retardo de comunicaciones en el L_{GM} recibido, el cual se corrige al agregar el $\Delta(x, y)$, con el fin de tomar en cuenta el movimiento del robot desde que se realizó la medición de L_{GM} hasta que se recibe en el robot. Al realizar el desplazamiento del L_{GM} recibido

para obtener el L_{GM} actualizado, se puede utilizar esta medición global en el esquema de fusión basado en eventos considerándola como $z_{k,p}$.

$$\begin{array}{c} \Delta T = T_{cam} + 0{,}5\,(T_{ss} - T_{cam}) \\ \Delta X = (V x_e \Delta T) \\ \Delta Y = (V y_e \Delta T) \end{array} \quad , \quad \begin{array}{c} L_{GM,x} = L_{GM,x,recibido} + \Delta X \\ L_{GM,y} = L_{GM,y,recibido} + \Delta Y \\ L_{GM,\theta} = L_{GM,\theta,recibido} \end{array} \tag{8.1}$$

Cabe destacar que la ecuación (8.1) considera un desplazamiento lineal de la plataforma, por lo que θ no varía durante T_{ss}. Por esta razón debe establecerse una condición adicional en el algoritmo de navegación, para evitar la ejecución de la actualización global cuando el robot se mueva de forma brusca (variación repentina en ω), como por ejemplo cuando se aplica una acción de control alta a los robots para negociar una curva o para evitar un obstáculo. La detección de este tipo de movimientos se puede implementar de distintas formas, según las características físicas de la plataforma y los sensores disponibles. Para las plataformas propuestas, se puede detectar un movimiento brusco al observar el error entre la velocidad de los motores y sus respectivas referencias ($e_{L,R}$ en el robot diferencial y $e_{F,R}$ para el robot Ackerman), debido a que este error se incrementa cuando la dirección de avance del robot cambia y decrece rápidamente debido a la actuación de los controladores PID de los motores.

Por ejemplo, en la figura 8.5 se observa la evolución de $e_{L,R}$ ante el movimiento del robot diferencial al inicio del seguimiento de una trayectoria cuadrada. De esta figura, se puede establecer un límite dual que define si el robot está girando rápidamente o no. De esta forma, si $|e_{L,R}| > 2$ ($|e_f| > 2$ para el robot Ackerman), entonces el robot se considera en movimiento brusco, con lo que no se debe realizar la actualización global. Por último, para permitir la estabilización del robot luego del movimiento realizado y evitar la actualización en caso de que se tengan movimientos repentinos consecutivos, el algoritmo puede esperar un tiempo predefinido (al regresar a $|e_{L,R}| < 2$) y una vez cumplido este tiempo, realizar la nuevamente la solicitud de la información al S.S. para intentar nuevamente la corrección global por eventos. Este tiempo de espera depende del tiempo de asentamiento (restablecimiento) del bucle de control PID de la velocidad de los motores, el cual en el caso

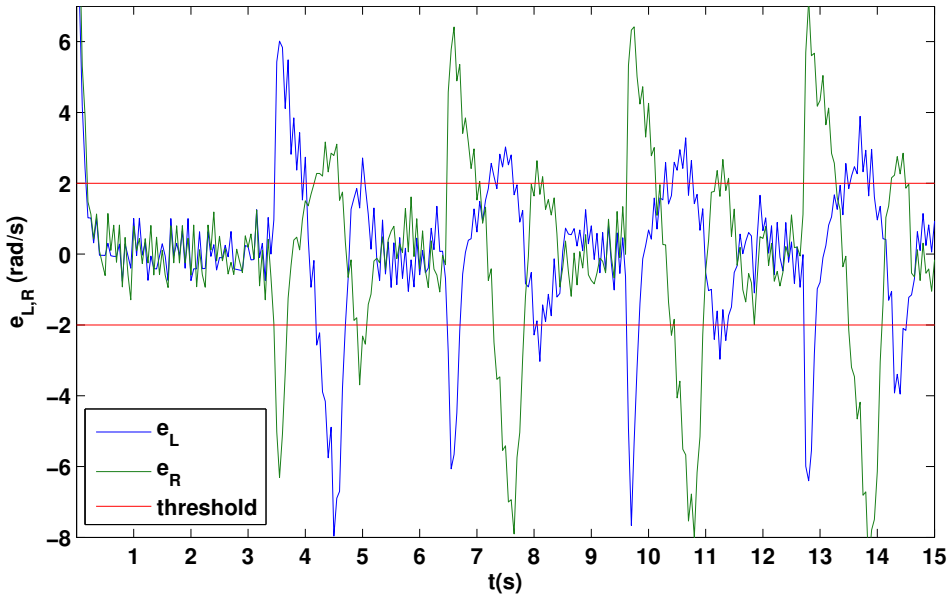


Figura 8.5: Evolución del error en la referencia del control PID de las ruedas izquierda y derecha ($e_{L,R}$) y definición de límites, robot diferencial siguiendo una trayectoria cuadrada.

de la plataforma LEGO NXT, se debe considerar como mínimo 1s debido al ruido asociado a la evolución de los errores $e_{L,R}$. En caso de otras plataformas, se debe realizar un estudio similar de los errores $e_{L,R}$ con el fin de determinar los límites $|e_{L,R}|$ y el tiempo de espera correspondiente.

Finalmente, bajo el esquema de sensorización global de la figura 8.4 puede realizarse un tipo de corrección global simplificado, el cual es útil para plataformas de recursos muy limitados, de forma que no se realiza la corrección global de la postura utilizando la corrección de un filtro de Kalman, por lo que se evita el cálculo de la ganancia $K_{k,p}$ (por ejemplo en el algoritmo 12 con modelo en cascada por eventos), requiriendo menos recursos computacionales para implementar el método por eventos, pero con la gran desventaja de no actualizar la covarianza del error $P_{k,p}$. Aun así, puede ser de utilidad y presentar resultados adecuados ya que permitiría la implementación de una fusión sensorial por eventos simplificada, mejorando la estimación de la

postura respecto a estimación por odometría. De esta forma, el algoritmo almacena la posición del robot (x_{ev}, y_{ev}) en el instante en el que $R_A > R_{A,lim}$, además de las velocidades globales (Vx_e, Vy_e), con el fin de que al recibir la postura de la cámara L_{GM} se pueda obtener los desplazamientos $(\Delta X, \Delta Y)$ en los ejes globales, que constituyen el error en la estimación de la posición local. Al agregar los desplazamientos a la estimación actual de la posición (x_{KF}, y_{KF}), se corregirá el error de la estimación local. Para esto se debe calcular el retardo ΔT utilizando T_{ss} y T_{cam}, pero en este caso se debe corregir (desplazar) la estimación en el instante del evento (x_{ev}, y_{ev}) de forma que se traslade al instante en donde se obtiene la medición global L_{GM}, con lo que se puede obtener correctamente $(\Delta X, \Delta Y)$, esto a diferencia de la corrección del retardo propuesta en la ecuación (8.1) en la que se desplaza L_{GM}. De esta forma el retardo se calcula como $\Delta T = (T_{ss} - T_{cam})/2$, al considerar que se realiza la medición de L_{GM} en el instante en el que se recibe la solicitud en el S.S.

Con este procedimiento se actualiza la postura global mediante la ecuación (8.2), para la cual nuevamente se considera que θ no varía durante T_{ss} por lo que se actualiza directamente al sustituir la medición global. Esto requiere cumplir con la actualización en instantes en donde no se realiza un movimiento brusco del robot, con lo que utiliza la comprobación en $e_{L,R}$ según se describió con anterioridad. En el caso de utilizar los filtros con corrección por eventos y al no tener la actualización de $P_{k,p}$ al utilizar la ecuación (8.2), se debe reestablecer $P_{k,p}$ a su valor inicial (lo que asume que la medición global es muy precisa) o bien asumir que su valor es cercano al de los términos de covarianza R del sensor global, esto con el fin de reestablecer el evento y permitir el funcionamiento correcto del filtro de fusión.

$$\begin{array}{ll} \Delta T = (T_{ss} - T_{cam})/2 & x = x_{KF} + \Delta X \\ \Delta X = (\hat{x}_{ev} + Vx_e \Delta T) - x_{GM}\,, & y = y_{KF} + \Delta Y \\ \Delta Y = (\hat{y}_{ev} + Vy_e \Delta T) - y_{GM} & \theta = \theta_{GM} \end{array} \tag{8.2}$$

8.1.6 Parámetros del filtro de fusión

Los parámetros requeridos para la fusión basada en eventos se obtienen para el LEGO NXT. Para el modelo dinámico del robot diferencial, los parámetros del sistema de partículas corresponden con un sólido rectangular con $b = 148mm$ y $c = 220mm$. Utilizando las ecuaciones de la tabla 3.1 para la versión de dos partículas se obtienen los parámetros $\gamma = 0{,}5$, $R_N = 76{,}542mm$ y $\lambda = 0{,}0131$. Para el modelo dinámico del robot Ackerman, los parámetros del sistema de partículas corresponden con un sólido rectangular con $b = 154mm$, $c = 167mm$ y $\lambda_a = 0{,}5$ en la ecuación (3.42).

Los parámetros de los filtros de fusión, la matriz de covarianza en la medida R_k y el proceso Q_k, se consideran invariantes en el tiempo en las pruebas realizadas, aunque los algoritmos propuestos permiten su variación temporal por lo que pueden modificarse en tiempo real según lo requiera el algoritmo de navegación. Esto permite considerar situaciones en las que se requiere modificar el aporte de los modelos (variación en Q_k) o sensores en la fusión (variación en R_k), administrar el fallo en alguno de los sensores (incremento en el término correspondiente de R_k), o bien considerar la pérdida transitoria en la precisión de algún sensor avanzado o retardos en la medición (considerando una función que realice la variación requerida en R_k). Sin embargo, aun en estas condiciones se requiere de un punto de partida para los valores de los términos de R_k y Q_k, para lo que se suelen obtener las versiones invariantes en el tiempo de ambas matrices de forma experimental.

En el caso de Q, se siguen los lineamientos propuestos en el capítulo 5 con el fin de ajustar el desempeño de los algoritmos de fusión, de forma que al incrementar los valores de Q se realiza la fusión dando mayor importancia a las mediciones que al modelo o viceversa en caso necesario. En cuanto a los algoritmos con entradas asignadas, la variación en Q regulará además el aporte del modelo de partículas y de los acelerómetros en la fusión sensorial. En cuanto a R se puede tomar como punto de partida las especificaciones del fabricante en cuanto a la desviación estándar en la medición para los sensores utilizados en la fusión sensorial, o bien determinarlos experimentalmente si no se cuenta con estos datos.

Tomando en consideración estos factores y al hecho de que el desempeño en la estimación del filtro puede ser mejorado mediante el ajuste de las matrices R y Q, sus términos se obtienen en la presente tesis de forma experimental utilizando la información de los sensores locales de la plataforma y del esquema global (figura 8.4), cuando el robot sigue una trayectoria cuadrada y circular (las pruebas de estas trayectorias se exponen en el capítulo 9 para el robot diferencial y en el capítulo 10 para el robot Ackerman).

De esta forma, los parámetros Q y R a utilizar en las distintas pruebas son los que hacen que la estimación local del filtro sea muy similar a la medida por la cámara cenital. El ajuste de los componentes en Q y R se realiza tomando en cuenta que un valor reducido en un término de estas matrices (covarianza reducida $\Rightarrow$ sensor o modelo preciso) hace que el filtro de fusión KF/EKF considere este valor como más relevante para la estimación que los otros términos. Por ejemplo, un valor reducido en ω_{gyr} produce que el filtro realice la estimación basándose principalmente en ésta medición, con lo que las demás mediciones de ω y la predicción del modelo (en el estado ω) serán menos relevantes (sus términos correspondientes de la ganancia del KF tendrán un valor de menor magnitud). Para el caso de la actualización global por eventos, los términos de R asociados al sensor global se ajustan partiendo inicialmente de la covarianza del sensor global y reduciendo su valor en caso necesario, de forma que al realizar la corrección por eventos se le dé mayor importancia al sensor global que a la estimación local, para lograr la reducción deseada del error de estimación en la postura.

Realizado este procedimiento, se obtienen las matrices Q y R diagonales con sus respectivos valores de los componentes del estado y la medición, tal y como se muestran en la tabla 8.1 para el robot diferencial y en la tabla 8.2 para el robot Ackerman. En ambas se indican los algoritmos en los que se pueden utilizar los parámetros de Q y R para realizar las pruebas experimentales y comparaciones de los capítulos 9 y 10.

Tabla 8.1: Parámetros del filtro de fusión, robot diferencial LEGO NXT

	Estado	Algoritmo 3	Algoritmo 10	Algoritmo 12
Q	x y θ	0,0002		0,0002 ($Q_{k,p}$)
	v	0,3		0,5
	ω	0,1		0,2
	Medición	Algoritmo 3	Algoritmo 10	Algoritmo 12
R	x_{GM} y_{GM}	0,015	0,015 (basado en eventos)	
	θ_{GM}	0,00001	0,00001 (basado en eventos)	
	v_{enc}	0,4		
	ω_{enc}	0,015		
	ω_{gyr}	0,023		
	ω_{comp}	0,088		

Tabla 8.2: Parámetros del filtro de fusión, robot Ackerman LEGO NXT

	Estado	Algoritmo 3	Algoritmo 10	Algoritmo 11	Algoritmo 12
Q	x	0,0001		0,0001 ($Q_{k,p}$)	
	y				
	θ				
	v_x		0,01		0,01
	v_y				-
	ω		0,03		0,9
	Medición	Algoritmo 3	Algoritmo 10	Algoritmo 11	Algoritmo 12
R	x_{GM}	0,015	0,015 (basado en eventos)		
	y_{GM}				
	θ_{GM}	0,00001	0,00001 (basado en eventos)		
	$v_{x,enc}$			0,01	
	$v_{y,enc}$		0,098		-
	ω_{enc}			0,038	
	ω_{gyr}			0,013	
	ω_{comp}		-		0,97

8.2 Navegación en Exteriores

Para la navegación en exteriores, se utiliza un robot móvil Ackerman basado en la plataforma LEGO NXT la cual se construye utilizando como punto de partida la plataforma de navegación en interiores (figura 8.2) a la cual se agrega el sensor global GPS (IG-500N) conectado a través de una tarjeta que permite su acceso y registra las variables de interés durante los experimentos (IGEPv2).

La plataforma para exteriores se muestra en la figura 8.6, en la que nuevamente se observan los sensores empleados en la fusión que incluyen los dos acelerómetros colocados en el centro de los ejes de las ruedas del robot (para el modelo por partículas (3.49), figura 3.6), un giróscopo, una brújula y los encoders de los motores trasero y delantero del robot. De la misma forma, se utiliza el diferencial y los engranajes para mejorar el funcionamiento de la dirección del vehículo y el desempeño del robot durante los giros.

El sensor global utilizado consiste en un GPS IG-500N de SBG Systems [80], el cual es un sensor GPS de alta precisión que incorpora una unidad inercial (IMU) cuyas medidas pueden ser utilizadas en un filtro EKF que incorpora el propio sensor para mejorar las mediciones del GPS. En el caso de la plataforma propuesta se desactiva esta opción (IMU y EKF interno), para utilizar en la fusión del LEGO únicamente la medición del GPS. De esta forma se realiza la corrección global de la estimación del filtro local que utiliza los sensores del LEGO NXT. Además, este sensor indica el número de satélites disponibles, por lo que se puede utilizar un evento adicional para realizar la corrección global únicamente si la medición del GPS se realiza con un número mínimo de satélites ($\boldsymbol{N}_{sat} \geq \boldsymbol{N}_{sat,min}$ según se establece en el capítulo 6).

El GPS incorpora una interfaz USB con la que se transmite la medición global al robot, para lo que se requiere utilizar un dispositivo que gestione esta transmisión (administrando el acceso al sensor GPS) y almacene las mediciones realizadas en trayectorias de larga duración. Para esto se utiliza una tarjeta IGEPv2 [83], la cual es un ordenador embebido (procesador ARM CORTEX A8 de 720MHz, DSP TMS320C64x+ y 512MB de memoria

RAM y Flash) con sistema operativo Linux, que administra el controlador del GPS IG-500N para obtener y almacenar la medición global y comunicarla mediante el USB al LEGO NXT. La interconexión entre el LEGO NXT y la IGEPv2 se realiza mediante un Hub USB que es alimentado por una batería LiPo al utilizar un convertidor DC-DC que reduce su voltaje a 5V.

Además de obtener, comunicar y almacenar L_{GPS}, se utiliza la IGEPv2 para recibir la comunicación de las mediciones y la estimación local provenientes del LEGO NXT para almacenarlas y permitir el análisis posterior del desempeño del filtro de localización. Cabe destacar que la IGEP *no* se utiliza para realizar los cálculos asociados al filtro de fusión propuesto, ya que esta función la realiza el LEGO NXT con el fin de demostrar la utilidad del esquema basado en eventos en plataformas de recursos limitados.

El acceso a L_{GPS} por parte del LEGO NXT se realiza siguiendo un método similar al definido en el esquema de sensorización global del caso de navegación en interiores, al considerar la IGEPv2 como el equivalente al C.S. y al S.S. y realizando la comunicación mediante USB. En este caso, aunque se tiene un retardo de comunicación ΔT menor al caso Bluetooth, se realiza la medición de éste para ser utilizado en el ajuste del L_{GPS} requerido en la corrección global por eventos, de forma que se corrija el error de la estimación local del filtro de fusión ejecutado en el NXT, por lo que se sigue el mismo procedimiento que en el caso de navegación en interiores (ecuación (8.1)).

Para la plataforma en exteriores se siguen además los mismos procedimientos de la plataforma en interiores en cuanto a la calibración, preprocesamiento, uso del algoritmo de navegación y parámetros del filtro de fusión (tabla 8.2, Algoritmo 12).

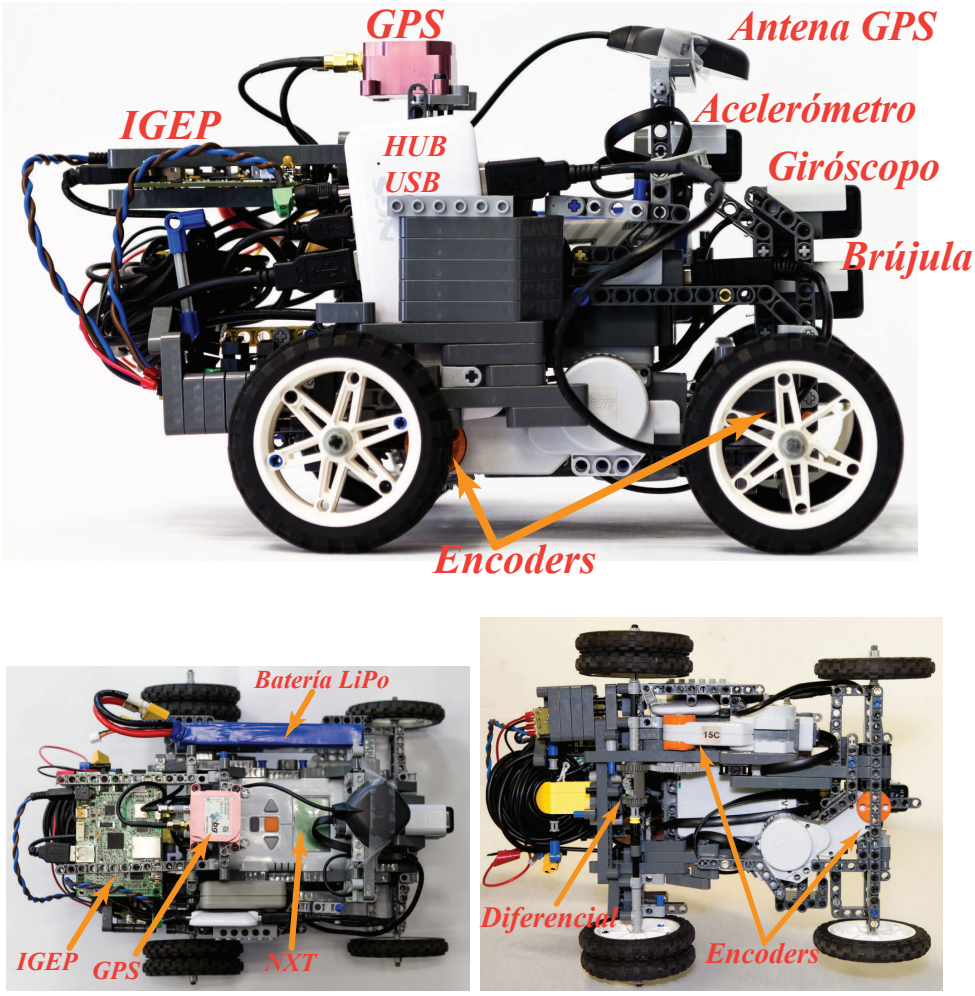


Figura 8.6: LEGO NXT en configuración Ackerman utilizado en las pruebas experimentales para exteriores

8.3 Plataforma de Simulación Multirobot

En cuanto al sistema multirobot, las pruebas experimentales se realizan mediante una plataforma de simulación basada en Java y JADE que implementa las condiciones definidas previamente (capítulo 7) para el grupo de robots G_R y su red de comunicación mediante agentes. El simulador permite la definición de robots de recursos limitados $G_{R,L}$, los cuales se consideran sin acceso a la medición global de forma directa, y robots avanzados $G_{R,A}$ con acceso a la medición global, los cuales se consideran con una precisión muy alta en la obtención de la postura del robot, con covarianza del error y error de localización nulos.

Para simular el movimiento de los robots se utiliza el modelo dinámico de las aceleraciones de las ruedas del robot de la ecuación (3.57) que se utiliza como entrada al modelo (3.34) (versión 3 partículas) el cual define las entradas del modelo de la postura del robot diferencial de la ecuación (3.24). Además, el modelo relativo de la ecuación (7.1) define las distancias y ángulos relativos (ρ, φ) entre todos los miembros $R_i \in G_R$ (al conocer el simulador la postura de cada R_i) que son obtenidas por cada integrante y transmitidas según las reglas establecidas en el capítulo 7 y según el tipo de actualización del algoritmo (tiempo, TCLA o eventos, ECLA).

El simulador realiza el registro de la información de localización, medición relativa y comunicaciones entre todos los robots simulados con el fin de realizar el análisis posterior y la comparación del funcionamiento de los algoritmos TCLA y ECLA.

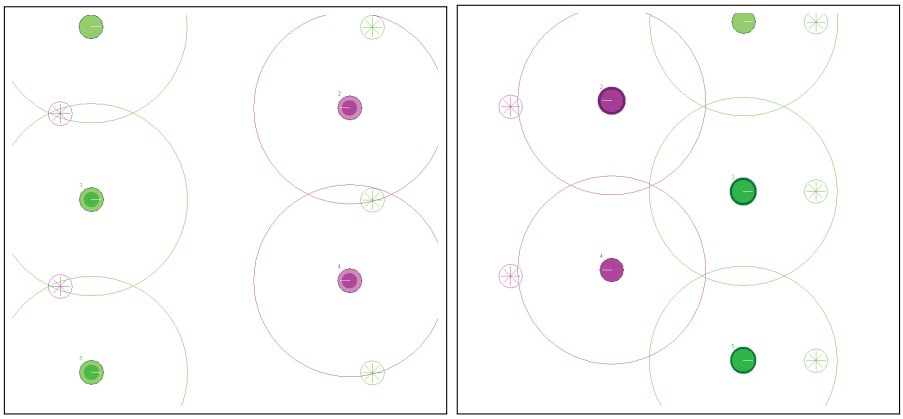
La interfaz de la plataforma de simulación se observa en la figura 8.7 en donde se define un entorno de navegación delimitado por paredes que forman un cuadrado, de forma que el movimiento de los robots esté contenido. Cada R_i se muestra como un círculo pequeño con un color de relleno claro, en el cual se indica el correspondiente valor del índice i del robot y la dirección (orientación) de avance del mismo mediante una línea interna blanca (del centro hacia el borde). El rango de detección D_r se representa mediante una circunferencia (sin relleno) centrada en cada R_i. Se representa además el área A_{ellip} de los elipsoides 3σ, obtenidos a partir de la covarianza del error

de estimación de la postura $P_{k,p}$, mediante un círculo con un color de relleno oscuro cuya área es igual a A_{ellip}. Al ejecutar la simulación en la plataforma, se observa en tiempo real tanto el movimiento del robot como la evolución de A_{ellip} la cual crece conforme avanza el tiempo (debido a la acumulación del error de la estimación local) a excepción de los robots en $G_{R,A}$ en los que se asume una precisión alta en la estimación de la postura por lo que el A_{ellip} se considera nula.

En el ejemplo de la figura 8.7 se observa la ejecución de la simulación. Al iniciar el movimiento (figura 8.7(a)), A_{ellip} es pequeña debido a que no se ha acumulado un error considerable en la postura, por lo que ésta se representa mediante círculos pequeños (relleno oscuro). Conforme el robot se mueve, se va incrementando A_{ellip} la cual puede llegar a ser igual al área del robot R_i ($A_{ellip} \cong A_{Ri}$, figura 8.7(b)) según la definición del evento. Al cabo de cierto tiempo, el error en la estimación de la postura del robot en movimiento será considerable, superando el área del robot ($A_{ellip} > A_{Ri}$, figura 8.7(c)) con lo que $R_{A,lim}$ estará cerca de superar la condición del evento $R_{A,lim}$. Cuando esto sucede, se producirá la corrección de la postura utilizando la información relativa (si existen R_i dentro de D_r), lo que reducirá A_{ellip} a un valor cercano a su valor inicial (figura 8.7(a)) según la precisión de las mediciones relativas y de la información recibida.

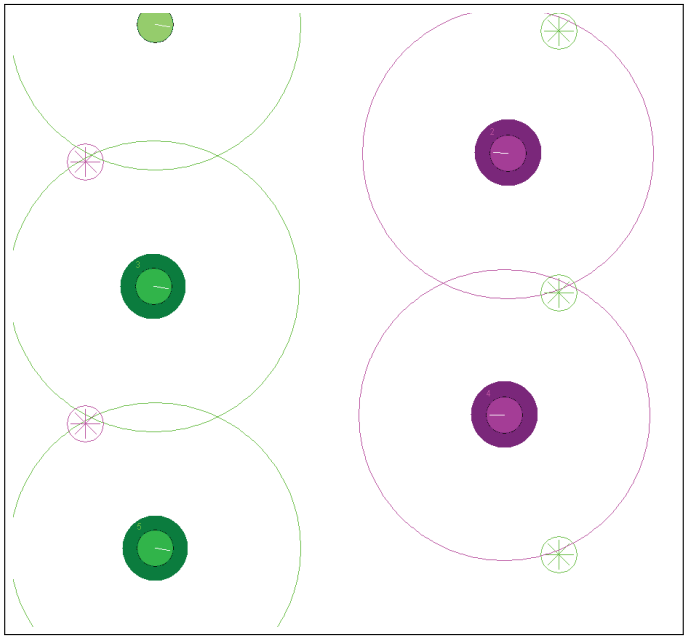
Finalmente conviene mencionar que la plataforma de simulación permite la utilización de un algoritmo de seguimiento de trayectorias de forma que los robots sigan una trayectoria deseada (por ejemplo, lineal y cíclica en la figura 8.7), lo cual es conveniente para comprobar el funcionamiento de la localización cooperativa cuando los robots se detectan con una frecuencia regular. Además, se puede establecer el movimiento de los robots utilizando una velocidad constante de avance junto con la ejecución de un algoritmo Braitenberg (figura 8.8), que realice la evitación de obstáculos, lo que permite observar el desempeño del algoritmo cuando la frecuencia de detección es variante en el tiempo.

Establecidas las plataformas y los aspectos de implementación se procede a presentar los resultados experimentales, que ejemplifican la utilidad y buen desempeño de los algoritmos propuestos en la presente tesis.



(a) Covarianza $P_{k,p}$ baja

(b) Covarianza $P_{k,p}$ media



(c) Covarianza $P_{k,p}$ alta, cercana al límite del evento

Figura 8.7: Plataforma de simulación para la localización cooperativa multirobot, G_R con 5 integrantes, trayectoria lineal cíclica, destino temporal indicado con $(*)$.

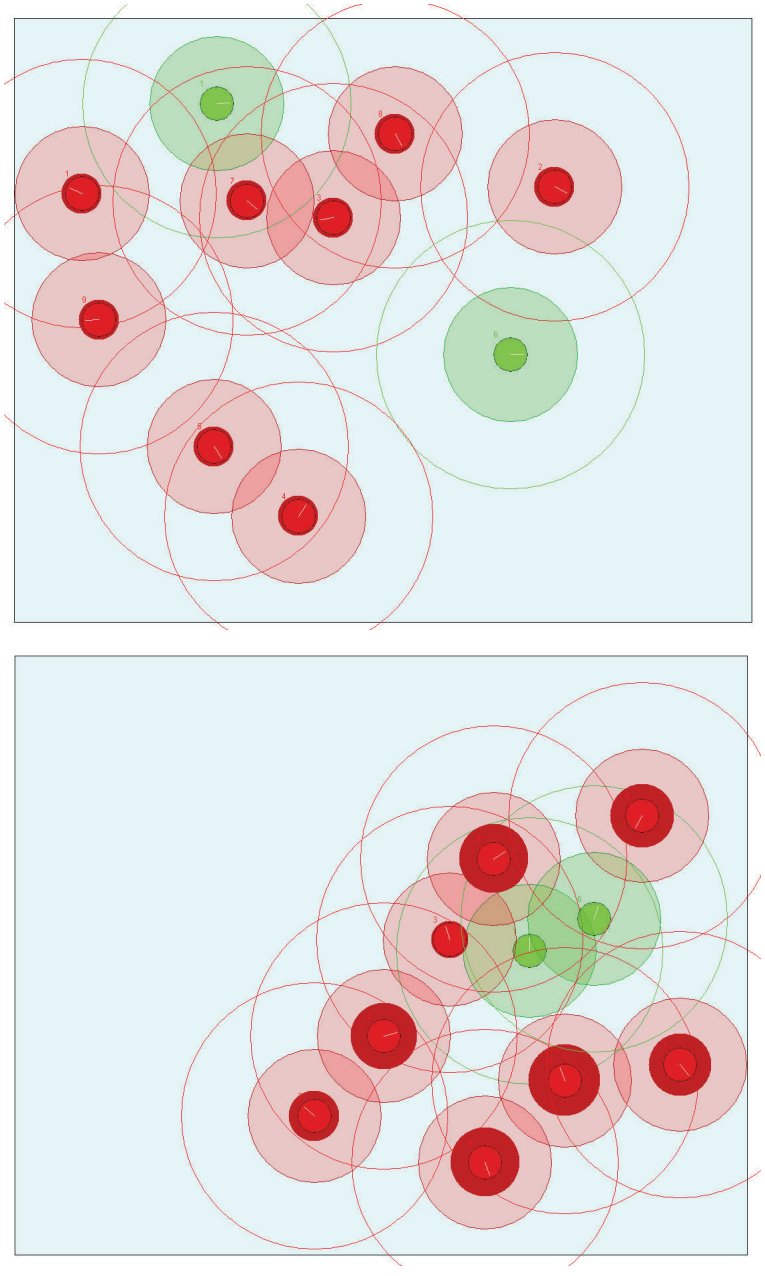


Figura 8.8: Plataforma de simulación para la localización cooperativa multirobot, G_R con 10 integrantes, algoritmo de navegación Braitenberg.

8.4 Conclusiones del Capítulo

En el presente capítulo se establecieron las distintas plataformas experimentales para la implementación de los algoritmos de fusión basada en eventos propuestos:

- Se utiliza el LEGO NXT para desarrollar dos plataformas para navegación en interiores (Diferencial en la figura 8.1 y Ackerman de la figura 8.2, medición global mediante una cámara cenital) y una para navegación en exteriores (Ackerman 8.6, medición global mediante un GPS gestionado mediante una tarjeta IGEPv2). Se eligió el LEGO NXT debido a que es una plataforma de bajo coste, con gran cantidad de sensores disponibles (acelerómetros, giróscopo, brújula, encoders, etc.), de implementación sencilla y de recursos limitados (memoria, procesamiento y comunicación), lo que permite implementar los métodos basados en eventos propuestos y comprobar su utilidad en cuanto a la mejora de la localización junto con su uso reducido de recursos y ancho de banda. Cabe destacar que los métodos desarrollados pueden ser implementados en otras plataformas móviles según los recursos computacionales y el ancho de banda disponibles en las mismas, seleccionando el algoritmo más conveniente según estas características y ajustando sus respectivas dimensiones según la cantidad de sensores disponibles.

- El preprocesamiento y la calibración de los sensores debe utilizarse para adecuar las unidades de medición a las empleadas por el esquema de fusión (S.I.) y para obtener las variables de entrada y medición para el algoritmo. Se debe realizar la calibración cada vez que se va a iniciar una prueba para eliminar el "bias" en cada sensor. Se estableció un algoritmo de navegación (figura 8.3) para las diversas pruebas, que permite el seguimiento de una trayectoria deseada durante la cual se ejecutan los algoritmos de localización para estimar la postura del robot.

- Se definió un esquema para la medición global de la postura en las pruebas en interiores (figura 8.4) que utiliza una cámara cenital para determinar la postura global del robot y transmitirla cuando es solicitada. Se definieron además dos métodos para la incorporación del retardo de comunicación en la fusión sensorial, al considerar la corrección del retardo en la postura global (ecuación (8.1)) y en la postura del robot (ecuación (8.2)), las cuales se utilizan en la etapa de corrección global del filtro de fusión para actualizar la estimación local y disminuir así el error de estimación. La ecuación para la corrección del retardo en la postura del robot (ecuación (8.2)) se puede utilizar en una corrección global simplificada, útil para plataformas de recursos muy limitados ya que evita la utilización del filtro de Kalman global para corregir la estimación local, lo que evita el cálculo de la ganancia de la corrección global requiriendo menos recursos computacionales para implementar el método por eventos.

- Se obtuvieron los parámetros de los filtros de Kalman requeridos para las pruebas experimentales, los cuales corresponden a las matrices de covarianza diagonales Q y R con sus respectivos valores de los componentes del estado y la medición, descritos en la tabla 8.1 para el robot diferencial y en la tabla 8.2 para el robot Ackerman.

- Se describió el simulador utilizado para las pruebas del algoritmo cooperativo (ECLA, Algoritmo 14), el cual consiste en una plataforma de simulación basada en Java y JADE que implementa las condiciones definidas para el grupo de robots G_R y su red de comunicación mediante agentes. La plataforma simula el movimiento de los robots mediante el modelo dinámico de las aceleraciones de las ruedas del robot (ecuación (3.57)) que se utiliza como entrada al modelo de aceleraciones (ecuación (3.34), versión 3 partículas) que a su vez se utiliza como entrada al modelo de la postura del robot diferencial (ecuación (3.24)). Para proveer las mediciones relativas (distancias y ángulos relativos (ρ, φ)) entre todos los miembros $R_i \in G_R$, se utiliza el modelo relativo de la ecuación (7.1).

- La plataforma de simulación multirobot representa gráficamente a cada robot miembro del grupo y su respectiva área 3σ indicando la covarianza del error de estimación. Además, permite la simulación de distintos comportamientos para los miembros del grupo, entre los que destacan el seguimiento de trayectorias (que permite una frecuencia de detección constante y regular entre los vecinos) y el movimiento con velocidad constante y evitación de obstáculos Braitenberg (que produce una frecuencia de detección variante en el tiempo) siendo ambos utilizados en las pruebas experimentales a realizar en el capítulo 11.

9 | Aplicaciones: Localización del Robot Diferencial

Se presentan a continuación diversas pruebas experimentales utilizando el robot diferencial de navegación en interiores (figura 8.1) y el esquema de localización global con la cámara cenital (figura 8.4), con el fin de comprobar el desempeño correcto de los algoritmos de fusión basados en eventos propuestos, así como el consumo de recursos en la plataforma medido tanto en el tiempo de ejecución del algoritmo como en el uso del ancho de banda (según el número de solicitudes al sensor global).

9.1 Pruebas de Desempeño

9.1.1 Comprobación de la fusión local

Para observar el desempeño de la fusión local, la primera prueba se realiza con un tiempo de muestreo $T_s = 50ms$ y *sin* utilizar la etapa de corrección global basada en eventos (denotada como ***EBGC***) de los filtros de fusión propuestos, por lo que L_{GM} no es utilizada en la estimación de la postura del robot. De esta forma se establece que el robot siga una trayectoria de referencia cuadrada y circular, utilizando la estimación de la postura mediante odometría (a partir de los encoders únicamente, sin fusión sensorial) para el algoritmo de navegación, y almacenando las mediciones locales junto con la información del sensor global (cámara cenital).

Utilizando las mediciones obtenidas, se realiza una simulación utilizando Matlab® para implementar, simular y comparar el desempeño de los algoritmos de fusión KF (propuesto, estimación local de las velocidades y modelo global para obtener la postura, Algoritmo 8 con 2 estados y 4 mediciones), el EKF (estado completo, entradas asignadas, cascada reducido, algoritmo 6, 5 estados y 4 mediciones) y el UKF [70] (estado completo, entradas asig-

Tabla 9.1: Resumen prueba de desempeño: robot diferencial LEGO NXT, $T_s = 50ms$

Robot	Diferencial, Fig.8.1	**Modelos**	(3.22),(3.23) y (3.24)
Corrección ΔT	——	**Parámetros**	Tabla 8.1
Estados	$x_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T, \quad x_{k,v} = \begin{bmatrix} v & \omega \end{bmatrix}_k^T$	**Entradas**	$u_k = \begin{bmatrix} a & \alpha \end{bmatrix}_k^T$
Mediciones	$z_{k,p} = \emptyset \quad z_{k,v} = \begin{bmatrix} v_{enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T$		

Abreviatura	**Algoritmo**	**Estado** x_k	**Medición** z_k	**Postura** L_k
KF (2s4m)	Alg. 8	$x_{k,v}$	$z_{k,v}$	modelo (3.11)
EKF (5s4m)	Alg. 6	$[x_{k,p} \quad x_{k,v}]^T$	$z_{k,v}$	$x_{k,p}$
UKF (5s4m)	Alg. 4	$[x_{k,p} \quad x_{k,v}]^T$	$z_{k,v}$	$x_{k,p}$
Encoder KF	Fig 4.3/modelo (3.13)	$x_{k,v}$	$[v_{L,enc} \quad v_{R,enc}]^T$	modelo (3.11)

nadas, algoritmo 4, 5 estados y 4 mediciones). Además, se implementa en la simulación una versión simplificada del KF que utiliza únicamente la información de los encoders (encoder KF) y el modelo cinemático tradicional (ecuación (3.13)), el cual utiliza las velocidades de las ruedas $(v_{L,enc}, v_{R,enc})$ para estimar las velocidades del robot (lineal v y angular ω), y a partir de estas calcular la postura (mediante la ecuación (3.11)) con el fin de observar el desempeño del filtro KF tradicional, sin utilizar la fusión sensorial (ya que no incorpora más sensores además de los encoders) ni el modelo dinámico propuesto.

El resultado de la simulación de los algoritmos junto con la medición de la cámara, los elipsoides 3σ y la estimación odométrica a partir de los encoders (*sin* calibrar según [121]) se muestran en las figuras 9.1 y 9.2 respectivamente. Se presenta además un resumen de los algoritmos, modelos y parámetros utilizados en esta prueba experimental (de la cual se obtienen ambas figuras) en la tabla 9.1 con el fin de facilitar la interpretación de los resultados.

Se observa en ambas figuras (9.1 y 9.2) que los filtros de fusión KF, EKF y UKF que utilizan el modelo dinámico y la fusión sensorial local (utilizando $z_{k,v}$) tienen un desempeño adecuado y similar entre sí, en cuanto su estimación de la postura es cercana a la determinada por la cámara cenital. De esto se observa que no hay una diferencia relevante en cuanto a la precisión de la estimación de la postura entre las distintas versiones del filtro de Kalman para esta prueba, con lo que el filtro KF en cascada con modelos en cascada

obtiene un desempeño similar a las versiones más complejas del filtro (EKF, UKF) pero utilizando menos recursos computacionales (según se expuso en el capítulo 5).

Además se observa de estas pruebas para la plataforma LEGO NXT, que ni la estimación de la postura utilizando la odometría a partir de los encoders sin calibrar, ni el filtro KF que estima de velocidades del robot utilizando únicamente la velocidad de las ruedas obtenidas de los encoders, son adecuados para obtener una estimación precisa de la postura del robot.

Comprobado el buen desempeño del algoritmo KF con modelos en cascada (Algoritmo 8, configuración en la tabla 9.1) para el caso de la simulación a partir de la información local, se procedió a su implementación en la unidad de control NXT de la plataforma, con lo que se realiza la ejecución de las pruebas mostradas en las figuras 9.3 y 9.4 en las que el robot efectúa el seguimiento de distintas trayectorias. En esta prueba se observa el correcto funcionamiento tanto del esquema de localización sin información global como el de la navegación para seguir la trayectoria deseada, además del logro relevante de poder ejecutar la fusión dentro del robot que calcula en cada instante de muestreo la ganancia requerida por el filtro para incorporar $z_{k,v}$ a la predicción del modelo y de esta forma obtener la postura del robot.

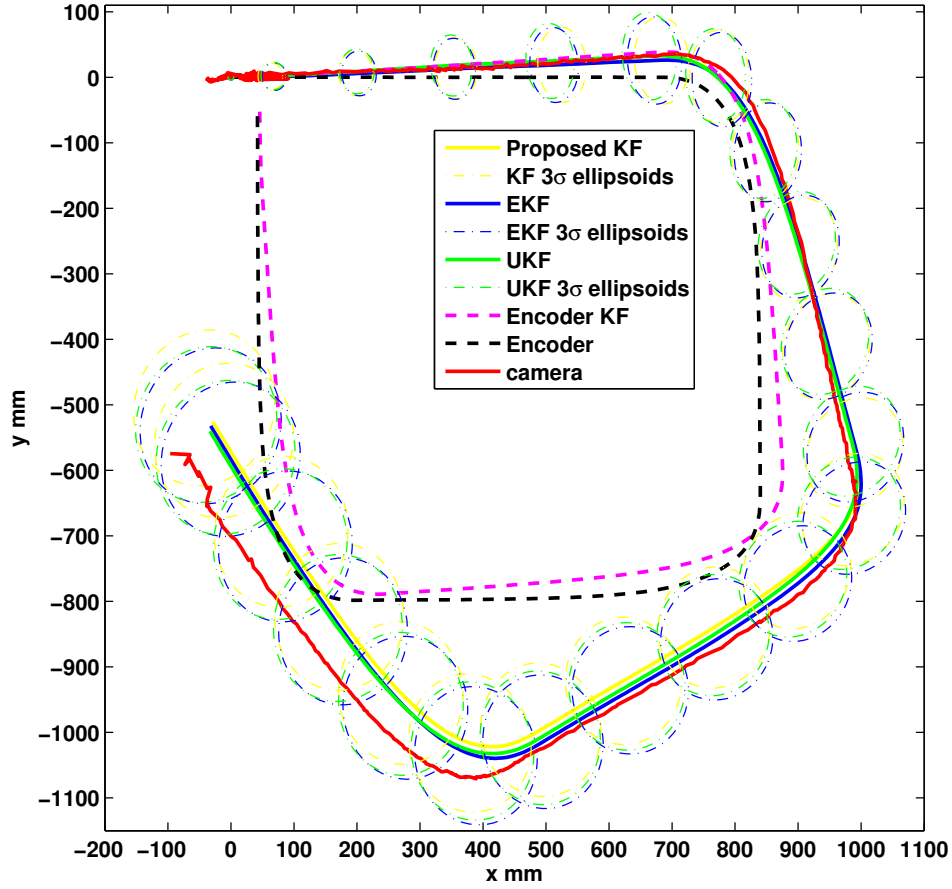


Figura 9.1: Desempeño de los algoritmos de localización, robot diferencial, prueba simulada, trayectoria de referencia cuadrada

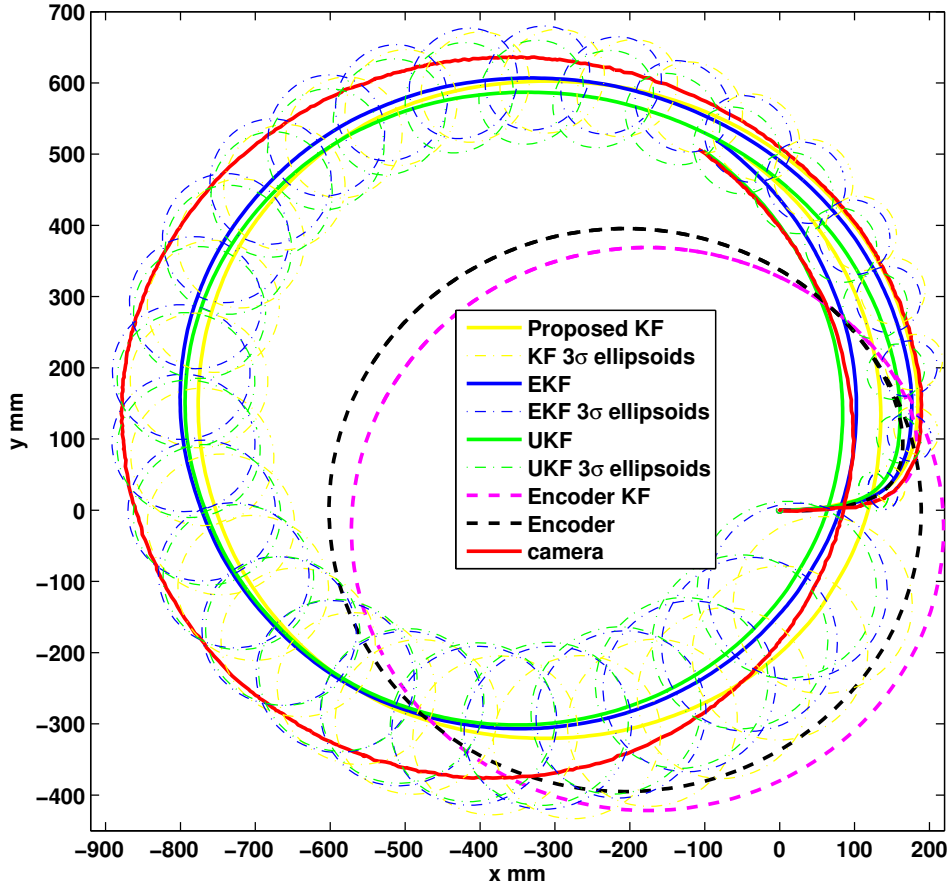
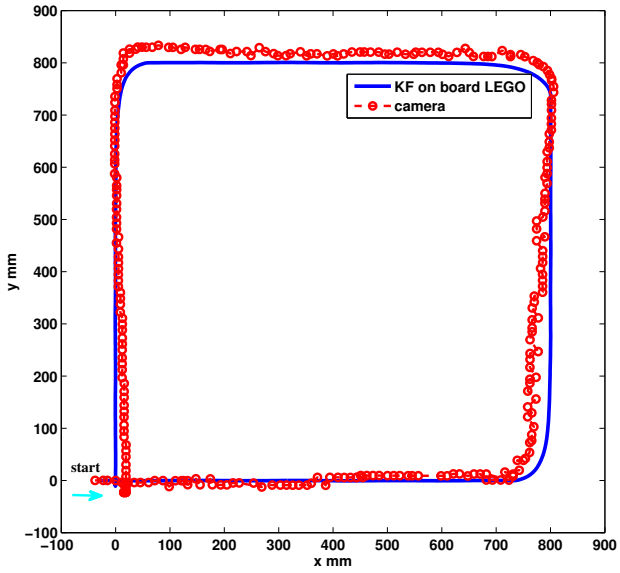
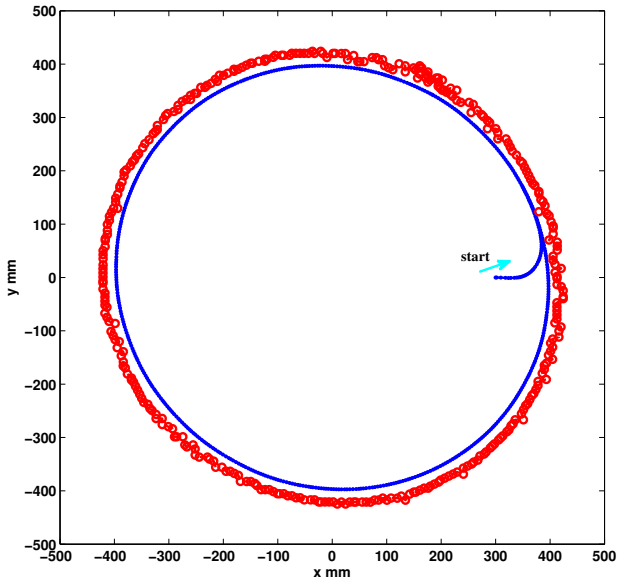


Figura 9.2: Desempeño de los algoritmos de localización, robot diferencial, prueba simulada, trayectoria de referencia circular

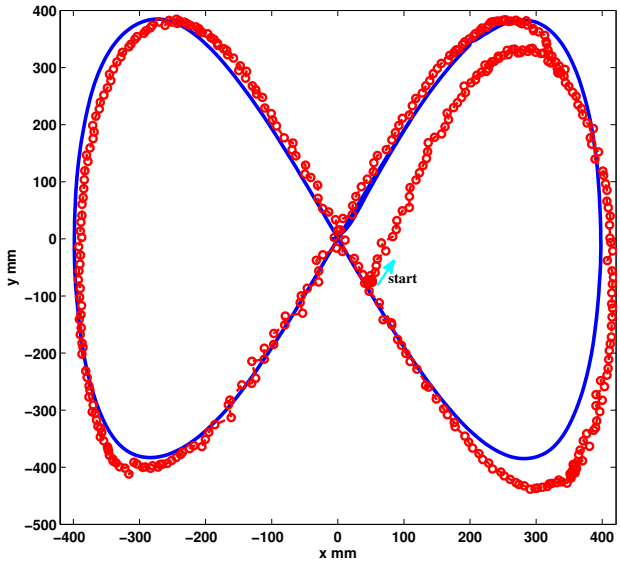


(a) Referencia cuadrada

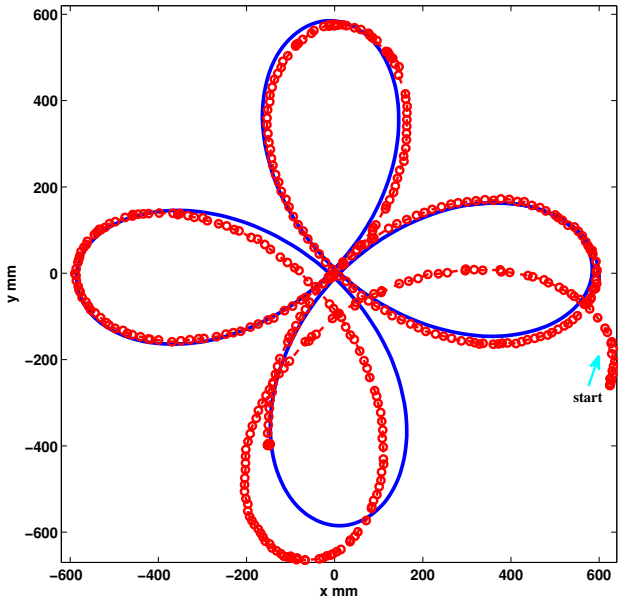


(b) Referencia circular

Figura 9.3: Desempeño de los algoritmos de localización y navegación, KF implementado en el robot diferencial, trayectorias de referencia cuadrada y circular



(a) Lemniscata



(b) Rosa Polar

Figura 9.4: Desempeño de los algoritmos de localización y navegación, KF implementado en el robot diferencial, trayectorias de referencia lemniscata y rosa polar

9.1.2 Prueba de larga duración, algoritmo basado en eventos

Comprobado el desempeño local se procede a implementar el esquema de corrección global basado en eventos, para lo cual se realiza la incorporación de la medición global (figura 8.4) de forma directa (sin filtro global), al utilizar el método simplificado de la ecuación (8.2) para corregir la estimación local del filtro de Kalman con modelos en cascada por eventos (algoritmo 12, que es el equivalente *por eventos* del algoritmo 8 de la prueba anterior, figuras 9.1 a 9.4).

La prueba es de larga duración y consiste en el seguimiento de una trayectoria de referencia cuadrada durante 30min utilizando la odometría de los encoders (calibrados mediante [121]), el KF con modelos en cascada sin información global (KF) y el KF con modelos en cascada con la corrección global basada en eventos (KF EBGC), tal y como se muestra en las figuras 9.5, 9.6 y 9.7 respectivamente. En estas figuras se indica tanto la estimación calculada por el robot para la primera vuelta (1min aproximadamente, color azul) como la medición de la cámara cenital (para la totalidad de la prueba 30min, color verde).

El funcionamiento de la detección del movimiento brusco y la evolución del evento R_A (basado en el área normalizada de los elipsoides 3σ) para el caso del filtro KF por eventos se muestra en la figura 9.8. Además, el resumen de la configuración de la prueba de larga duración se muestra en la tabla 9.2.

Tabla 9.2: Resumen prueba de desempeño de larga duración: robot diferencial LEGO NXT, $T_s = 50ms$

Robot	Diferencial, Fig.8.1	**Modelos**	(3.22),(3.23) y (3.24)
Corrección ΔT	ecuación (8.2)	**Parámetros**	Tabla 8.1
Estados	$x_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T$, $x_{k,v} = \begin{bmatrix} v & \omega \end{bmatrix}_k^T$	**Entradas**	$u_k = \begin{bmatrix} a & \alpha \end{bmatrix}_k^T$
Mediciones	$z_{k,p} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_{GM}^T = L_{GM}$	$z_{k,v} = \begin{bmatrix} v_{enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T$	

Abreviatura	**Algoritmo**	**Estado** x_k	**Medición** z_k	**Postura** L_k
Odometry	modelo (3.13)	–	$[v_{L,enc} \quad v_{R,enc}]^T$	modelo (3.12)
velocity KF	Alg. 12	$x_{k,v}$	$z_{k,v}$	modelo (3.11)
KF EBGC	Alg. 12	$x_{k,v}$	$[z_{k,p} \quad z_{k,v}]^T$	modelo (3.11)

Se observa en el resultado de esta prueba que cuando el robot utiliza únicamente la odometría (figura 9.5), su estimación de la postura coincide con la medición global de la cámara cenital únicamente durante la primera vez que se completa la trayectoria, divergiendo rápidamente a medida que el error de estimación se acumula. Este error de estimación no es tomado en cuenta por el robot, que considera que la estimación es adecuada y sigue correctamente la trayectoria, por lo que en la unidad de control NXT del robot el error de navegación es mínimo, aunque en realidad la postura es errónea según se observa de la medición de la cámara, por lo que el robot no sigue la trayectoria de forma adecuada e incluso llega a salir del espacio medible por el sensor global.

Se obtiene una mejora apreciable al utilizar el esquema de fusión con las mediciones locales (algoritmo 12 *sin* corrección global) tal y como se observa en la figura 9.6. El filtro de fusión funciona mejor que la odometría de los encoders, de forma que la estimación de la postura coincide con la medición global para las primeras vueltas en las que se completa la trayectoria cuadrada, pero diverge lentamente según se acumula el error de estimación que no es corregido ya que no se dispone de información global en la estimación.

En el caso del filtro con corrección global basada en eventos (KF EBGC, algoritmo 12, modelos en cascada) se observa en la prueba de la figura 9.7 una estimación adecuada de la postura del robot, al corregir el error de estimación utilizando la información del sensor global (ecuación (8.2)) cuando el evento R_A basado en los elipsoides 3σ supera el límite $R_{A,lim} = 0{,}5$. Bajo este esquema, la precisión en la estimación de la postura es adecuada en cuanto el error de estimación no crece indefinidamente, sino que es acotado (su valor máximo dependerá de la definición de $R_{A,lim}$). De esta forma la trayectoria del robot se asemeja a la referencia cuadrada y no diverge como en los casos anteriores.

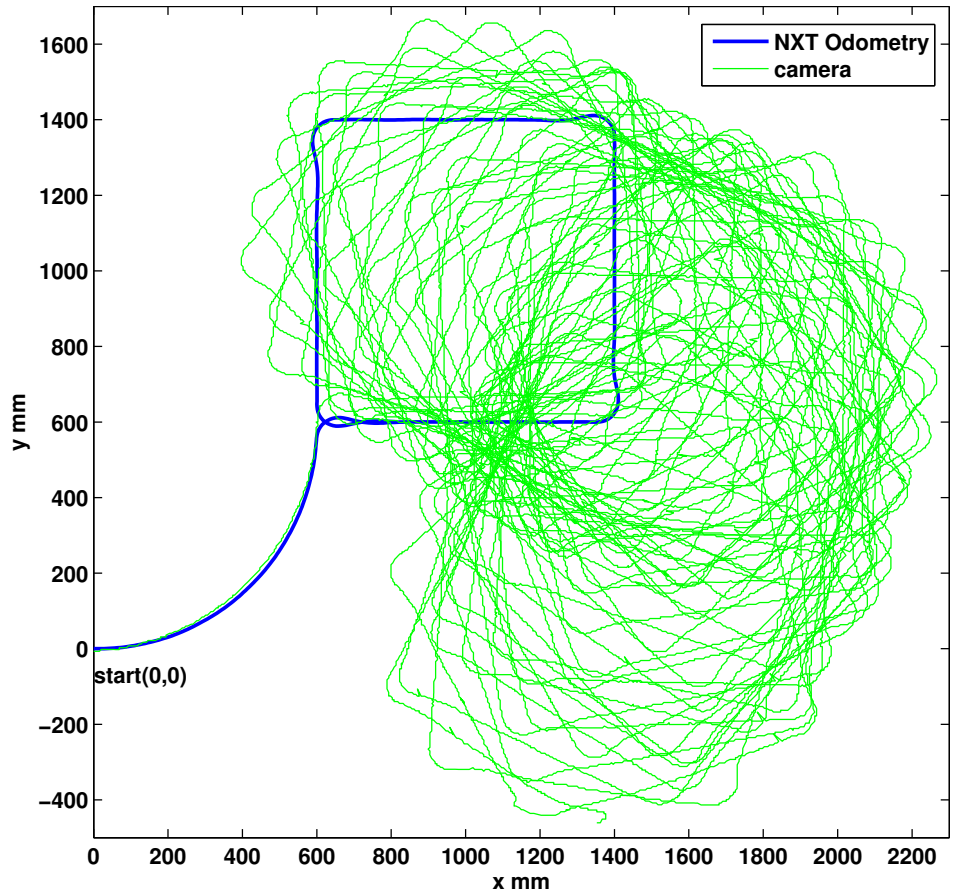


Figura 9.5: Prueba de larga duración, seguimiento de la trayectoria durante 30min, robot diferencial con odometría a partir de los encoders

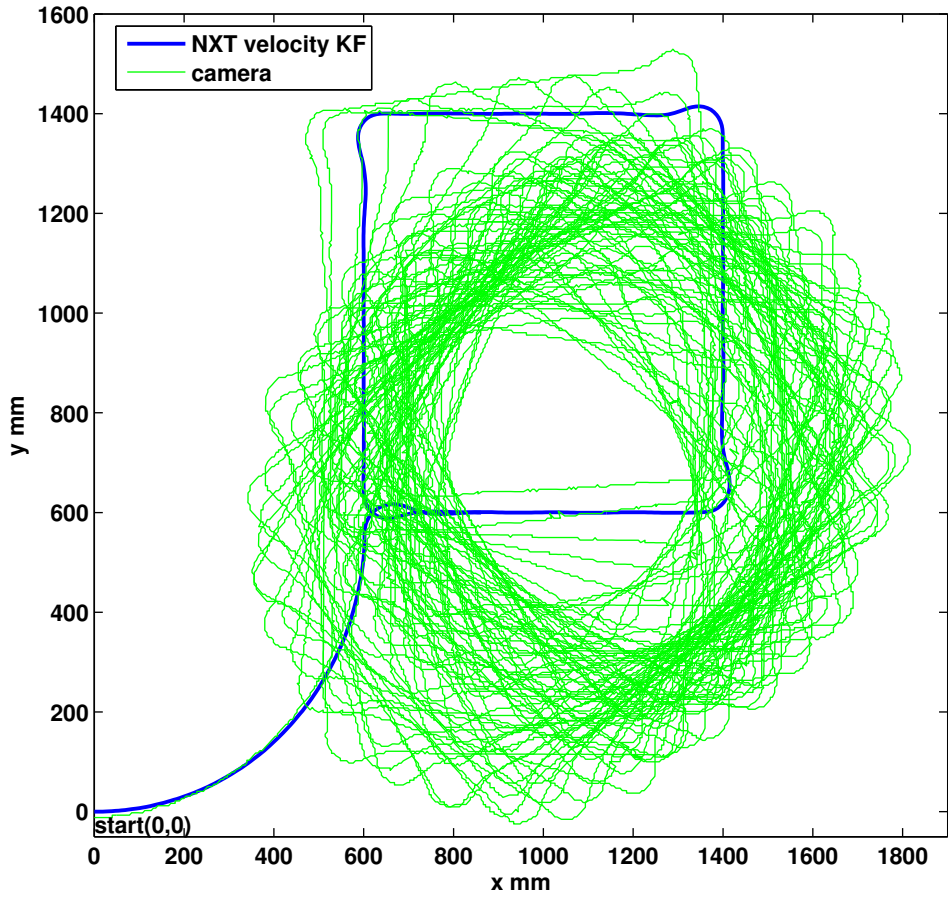


Figura 9.6: Prueba de larga duración, seguimiento de la trayectoria durante 30min, robot diferencial con el filtro local, Alg. 12 sin L_{GM}

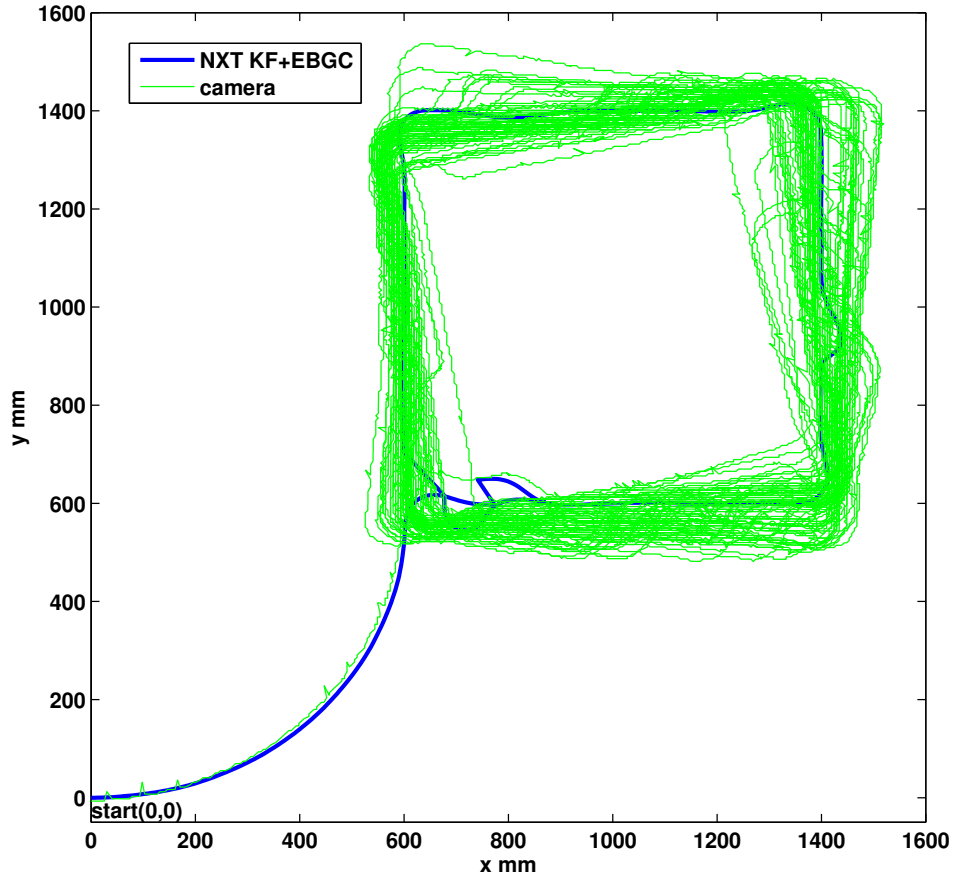


Figura 9.7: Prueba de larga duración, seguimiento de la trayectoria durante 30min, robot diferencial con el filtro local, Alg. 12 con L_{GM} y EBGC, $R_{A,lim} = 0{,}5$

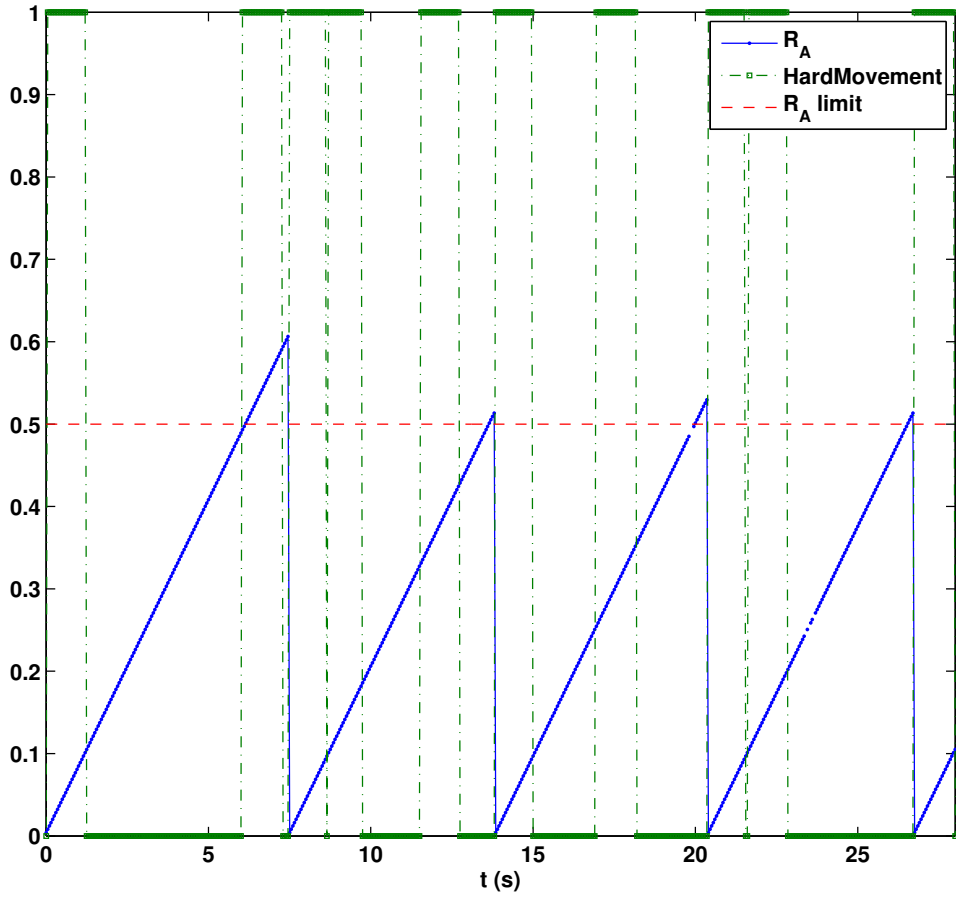


Figura 9.8: Prueba de larga duración, evolución inicial del evento R_A, robot diferencial con el filtro local, Alg. 12 con L_{GM} y EBGC, $R_{A,lim} = 0{,}5$

En la figura 9.8 se muestra la evolución de R_A en los primeros instantes de la prueba realizada para el algoritmo con corrección basada en eventos (KF EBGC, figura 9.7). En ésta se observa como la covarianza del error de estimación, representada por medio del área normalizada de los elipsoides 3σ, crece desde que se inicia la navegación hasta que se realiza la corrección con la información global (ecuación (8.2)), que reinicia el valor de $P_{k,p}$ a su valor inicial $P_{0,p}$, lo que a su vez disminuye R_A a su valor inicial.

La información global es solicitada cuando $R_A > R_{A,lim} = 0{,}5$, por ejemplo en el tiempo 6,25s de la figura 9.8 (primera vez que se supera $R_{A,lim}$). En este instante, el robot comprueba la condición de movimiento brusco ($|e_{L,R}| > 2$), la cual resulta como "verdadera" (valor 1 en la figura 9.8), por lo que no se realiza la actualización global (según los lineamientos de la sección 8.1.5). Con esto el algoritmo espera 1s y vuelve a comprobar la condición $|e_{L,R}| > 2$, que resulta ser "falsa" (valor 0 en la figura 9.8), por lo que el algoritmo procede a obtener la información global y a realizar la actualización (aproximadamente en el tiempo 7,45s) con lo que R_A vuelve a su valor inicial. De esta forma se logra acotar el error de estimación sin saturar el uso del ancho de banda de comunicación de la plataforma.

Conviene destacar nuevamente que la utilización de la corrección basada en eventos permite la implementación del filtro de fusión (algoritmo 12) en la unidad de control NXT de la plataforma, de forma que utiliza el ancho de banda para obtener la medición global únicamente cuando $R_A > R_{A,lim}$. Esto evita el uso del medio de comunicación en cada instante de muestreo (a diferencia de los métodos tradicionales) evitando su saturación y permitiendo asignar los recursos de la plataforma a otras tareas relevantes como la supervisión del sistema, el control de motores, etc.

Adicionalmente, se obtuvo el índice **IAE**, definido como la integral del valor absoluto del error de estimación, para la posición en x e y de las pruebas mostradas en las figuras 9.5 a 9.7, el cual es útil para apreciar el aporte del algoritmo basado en eventos propuesto en la mejora del desempeño al compararlo con la odometría tradicional y con el método de fusión local (sin actualización global). Los resultados se observan en la tabla 9.3, en donde se calcula el promedio de los índices para x e y y su equivalente

Tabla 9.3: Índice de desempeño IAE y errores porcentuales para la prueba de larga duración (30min), robot diferencial

Pruebas en las figuras 9.5 a 9.7	IAE			Error (%)	Mejora % respecto a la odometría
	x	y	Promedio xy	Promedio xy	Promedio xy
Odometría	$9{,}541e+05$	$6{,}750e+05$	$8{,}146e+05$	46,057	–
KF	$2{,}949e+05$	$4{,}376e+05$	$3{,}663e+05$	20,734	238,9
KF EBGC	$6{,}221e+04$	$5{,}342e+04$	$5{,}781e+04$	3,270	1398,8

en error porcentual (obtenido a partir del IAE de una ampliación de un 1 % en las trayectorias x e y), además de la mejora porcentual respecto a la estimación odométrica. Se aprecia nuevamente el buen desempeño de la fusión sensorial basada en eventos implementada en la unidad de control NXT de la plataforma, al obtener la postura con un bajo error de estimación (IAE cercando al 3 %) y proveer una mejora considerable en la precisión respecto a la odometría tradicional y a la fusión local.

Como última prueba del desempeño del robot diferencial, se realiza el seguimiento de dos trayectorias adicionales según se muestra en las figuras 9.9 y 9.10, de las cuales se observa nuevamente el buen desempeño del filtro de fusión con la corrección por eventos (KF EBGC, algoritmo 12), al estimar y seguir de forma adecuada ambas trayectorias con un error acotado (sin divergencia).

Finalmente, un video con el resumen de las pruebas presentadas en las figuras de la 9.5 a la 9.10 puede encontrarse en `/http:///wks./gii./upv./es/ /cobami//webfm_/send//5`.

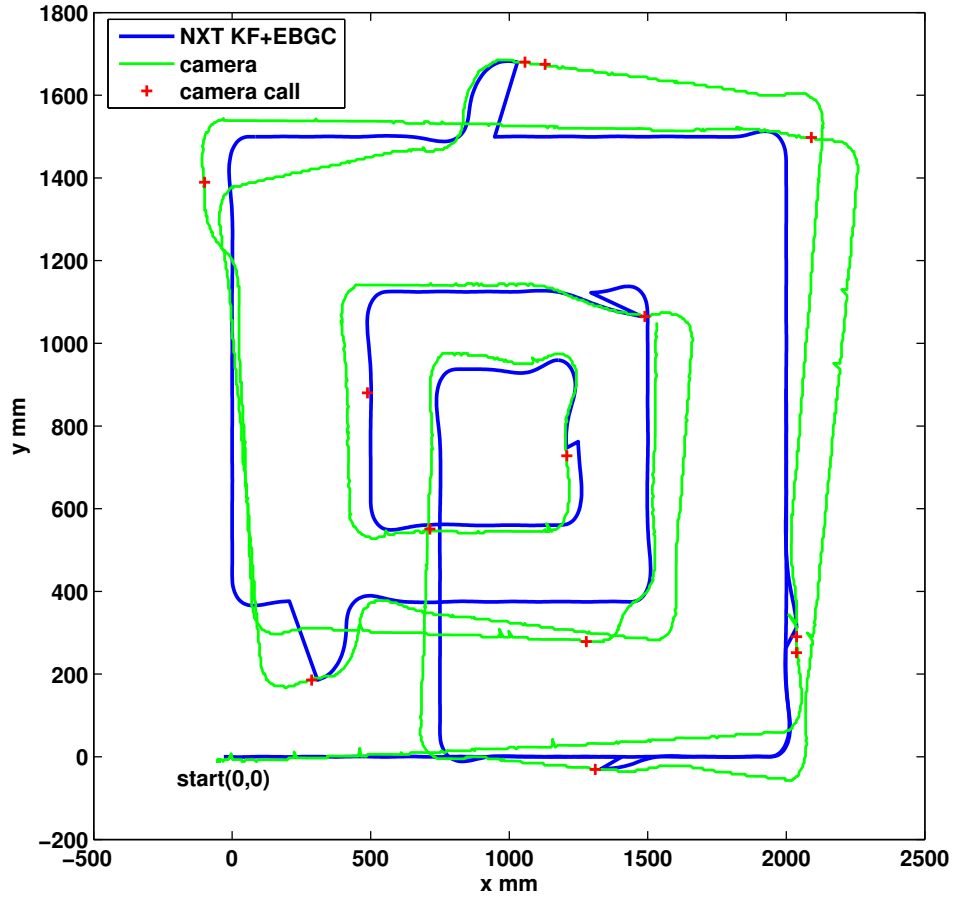


Figura 9.9: Prueba de desempeño, seguimiento de una trayectoria espiral rectangular durante 5min, robot diferencial con el filtro local, Alg. 12 con L_{GM} y EBGC, $R_{A,lim} = 1{,}0$

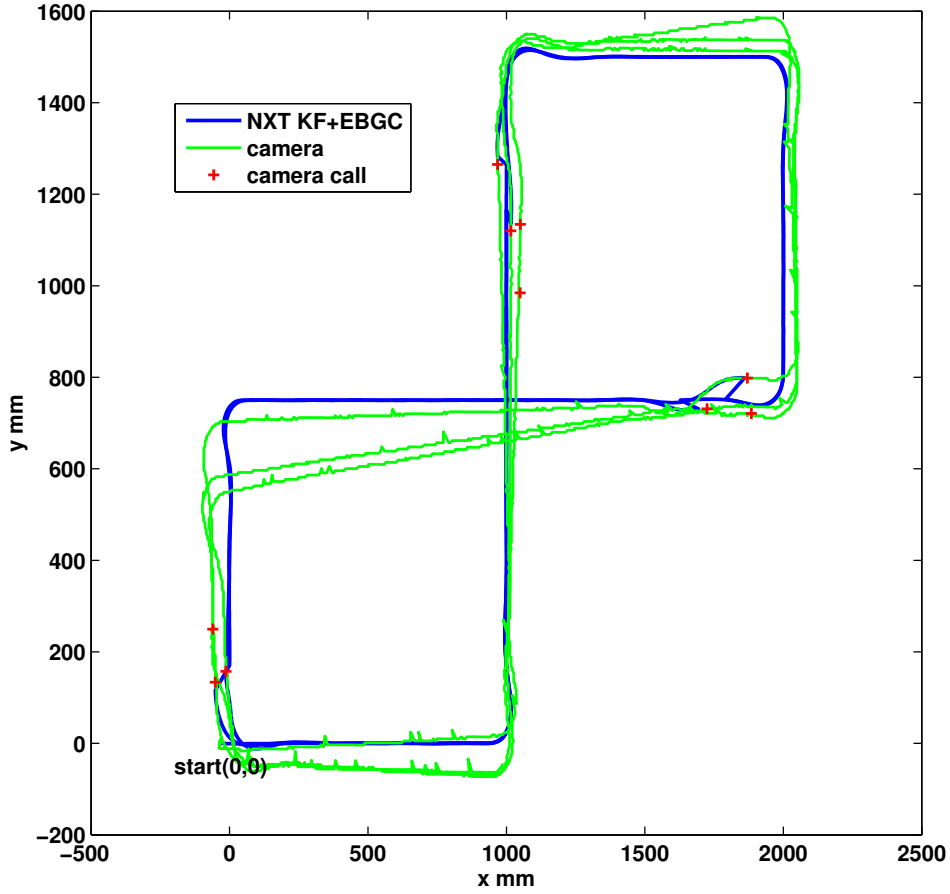


Figura 9.10: Prueba de desempeño, seguimiento de una trayectoria cuadrada doble en diagonal durante 5min, robot diferencial con el filtro local, Alg. 12 con L_{GM} y EBGC, $R_{A,lim} = 1{,}0$

9.1.3 Influencia del nivel del evento en el desempeño

Comprobado el buen desempeño tanto de la fusión local como de la actualización basada en eventos al haber implementado ambos métodos en la unidad de control, se procede a estudiar la relación entre el desempeño del esquema basado en eventos (medido mediante el índice IAE) y el uso de comunicaciones en la plataforma (ancho de banda representado por la cantidad de solicitudes realizadas al sensor global), la cual es regulada mediante el nivel límite del evento $R_{A,lim}$. Para esto se realizó una serie de pruebas asignando una referencia cuadrada que el robot diferencial sigue durante 3min para distintos valores de $R_{A,lim}$, almacenando durante cada prueba la estimación de la postura realizada por el filtro (KF EBGC, algoritmo 12), la medición de la postura global de la cámara (L_{GM}) y el número de consultas realizadas por el robot. Utilizando esta información se obtuvo el IAE para cada prueba en la posición x e y, así como el error promedio porcentual en ambas coordenadas (utilizando el mismo procedimiento seguido en la tabla 9.3) y se graficó su variación respecto al $R_{A,lim}$, con lo que se obtuvo la figura 9.11.

Se observa claramente de esta prueba, que el índice IAE (en la posición x, y) se crece según el valor de $R_{A,lim}$, lo cual es de esperarse debido a que un límite con valor alto producirá menos actualizaciones globales (se tarda más en alcanzar $R_{A,lim}$), con lo que se incrementa el error de estimación de la postura y por lo tanto el IAE. Por otra parte, un valor bajo en $R_{A,lim}$ producirá un mayor número de consultas a la cámara (más costo computacional) y un incremento en la precisión del método, produciendo una disminución en el valor IAE hasta un valor límite que depende de la plataforma, definido según los recursos y ancho de banda disponibles y de acuerdo a los retardos ΔT de comunicación presentes, siendo estos factores los que limitan el número máximo de llamadas que se pueden realizar. En caso de los métodos tradicionales, se realizarían llamadas al sensor global en cada instante de muestreo, siendo posible obtener un menor IAE pero con la desventaja de no ser implementables en una plataforma con ancho de banda o recursos limitados.

De la figura 9.11 se observa claramente el compromiso entre el uso del ancho de banda (consultas a la cámara) y la precisión del método (índice IAE),

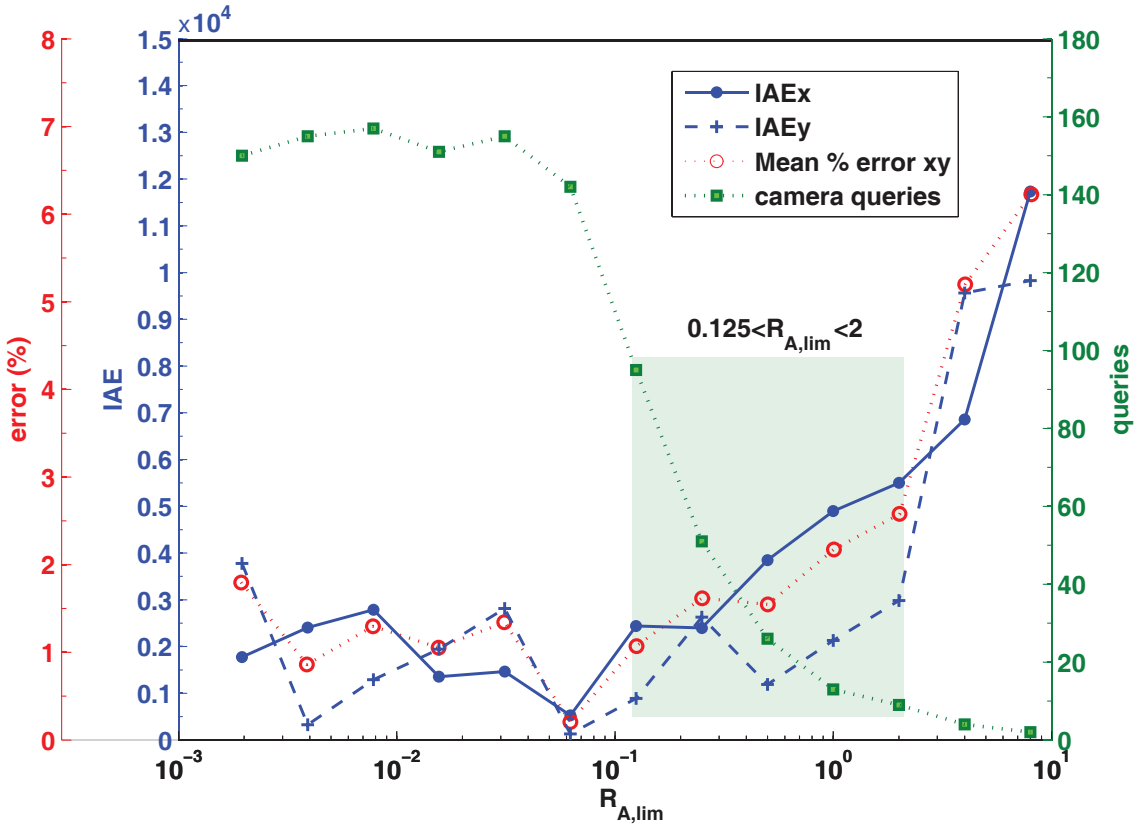


Figura 9.11: Relación entre $R_{A,lim}$, el índice IAE, el error promedio porcentual y el número de consultas al sensor global para el conjunto de pruebas de seguimiento de trayectoria en 3min, robot diferencial.

el cual es controlado mediante la asignación del límite del evento $R_{A,lim}$. Se aprecia cómo se utilizan menos recursos y realizan un menor número de llamadas si $R_{A,lim} > 2$, pero esto produce a su vez una menor precisión en la estimación con un IAE alto. Por el contrario, valores de $R_{A,lim} < 0{,}125$ producen un IAE reducida pero requiriendo un mayor número de consultas al sensor global. De esta forma, un rango adecuado para el límite del evento puede establecerse en $0{,}125 < R_{A,lim} < 2$ para la plataforma LEGO NXT, de forma que se obtiene una solución de compromiso entre un valor IAE adecuado junto con un número de llamadas reducido.

Por último, en la figura 9.11 se muestra un eje adicional indicando el error promedio porcentual en las posiciones x e y, obtenidas en cada prueba utilizando como referencia el error IAE de una ampliación de un 1 % en las trayectorias x e y, para escalar los valores IAE del error de estimación. Se observa que para el rango $R_{A,lim}$ establecido como adecuado para el robot diferencial, se produce un incremento del error de un 1,1 % a un 2,6 % con una reducción de 95 a 9 consultas al sensor global en 3min. Si $R_{A,lim} > 2$, entonces el error puede incrementarse hasta un 6,2 % utilizando únicamente 2 consultas en 3min, y si $R_{A,lim} < 0{,}125$, entonces se obtiene un error mínimo para la plataforma diferencial de un 0,2 % pero utilizando una gran cantidad de recursos computacionales, realizando 142 consultas o más en los 3min de la prueba.

De este análisis se observa que el algoritmo con corrección basada en eventos (KF EBGC, algoritmo 12), con un valor adecuado del $R_{A,lim}$, puede realizar una estimación de la postura con un valor aceptable del IAE (y en el error porcentual) a la vez que se ahorran recursos comunicaciones y ancho de banda, ya que únicamente se realizan las consultas necesarias al sensor global para corregir la estimación local. Este análisis puede realizarse al implementar el método en otras plataformas, con el fin de obtener el rango óptimo de $R_{A,lim}$ a utilizar según los recursos disponibles en el robot.

Concluidas las pruebas del desempeño para la plataforma diferencial se procede con las pruebas del tiempo de ejecución del algoritmo tal y como se expone a continuación.

9.2 Pruebas de tiempo de ejecución

Se presentan a continuación dos pruebas experimentales realizadas para comprobar la eficiencia de los algoritmos propuestos en cuanto al tiempo de ejecución que requieren para realizar la estimación de la postura, tal y como se describe a continuación.

La primera prueba consiste en una simulación de varios de los algoritmos de fusión basados en el tiempo en donde se obtiene el tiempo de ejecución promedio, tal y como se muestra en la figura 9.12. Los algoritmos son implementados en un ordenador (2.4GHz y 4GB RAM) mediante Matlab®, el cual ejecuta los distintos filtros utilizando la información experimental obtenida de la prueba de desempeño simulada (trayectoria cuadrada, figura 9.1, tabla 9.1), con lo que el promedio del tiempo de ejecución de cada algoritmo incluye aproximadamente 600 iteraciones (cerca de 30s con el $T_s = 50ms$).

Tabla 9.4: Resumen prueba del tiempo de ejecución en un ordenador externo: robot diferencial LEGO NXT, $T_s = 50ms$

Robot	Diferencial, Fig.8.1	**Modelos**	(3.22),(3.23) y (3.24)
Corrección ΔT	——	**Parámetros**	Tabla 8.1
Estados	$x_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T$, $x_{k,v} = \begin{bmatrix} v & \omega \end{bmatrix}_k^T$	**Entradas**	$u_k = \begin{bmatrix} a & \alpha \end{bmatrix}_k^T$
Mediciones	$z_{k,p} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_{GM}^T$ $z_{k,v} = \begin{bmatrix} v_{enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T$		

Abreviatura	**Algoritmo**	**Estado** x_k	**Medición** z_k	**Postura** L_k
KF2s4m	Alg. 8	$x_{k,v}$	$z_{k,v}$	modelo (3.11)
EKF5s4m	Alg. 6	$[x_{k,p} \quad x_{k,v}]^T$	$z_{k,v}$	$x_{k,p}$
EKF5s7m	Alg. 3	$[x_{k,p} \quad x_{k,v}]^T$	$[z_{k,p} \quad z_{k,v}]^T$	$x_{k,p}$
EKF7s6m	Alg. 1	$[x_{k,p} \ x_{k,v} \ u_k]^T$	$[z_{k,v} \quad u_k]^T$	$x_{k,p}$
EKF7s9m	Alg. 1	$[x_{k,p} \ x_{k,v} \ u_k]^T$	$[z_{k,p} \ z_{k,v} \ u_k]^T$	$x_{k,p}$
UKF5s4m	Alg. 4	$[x_{k,p} \quad x_{k,v}]^T$	$z_{k,v}$	$x_{k,p}$

La configuración de los algoritmos utilizados en la prueba de la figura 9.12 se resume en la tabla 9.4, estos consisten en el algoritmo KF con modelos en cascada (KF2s4m, 2 estados y 4 mediciones), el EKF en cascada reducido (EKF5s4m), el EKF con asignación de entradas (EKF5s7m), el EKF con estimación local sin entradas asignadas (aceleraciones de u_k como estados, EKF7s6m), el EKF tradicional (todas las mediciones, agregando las acele-

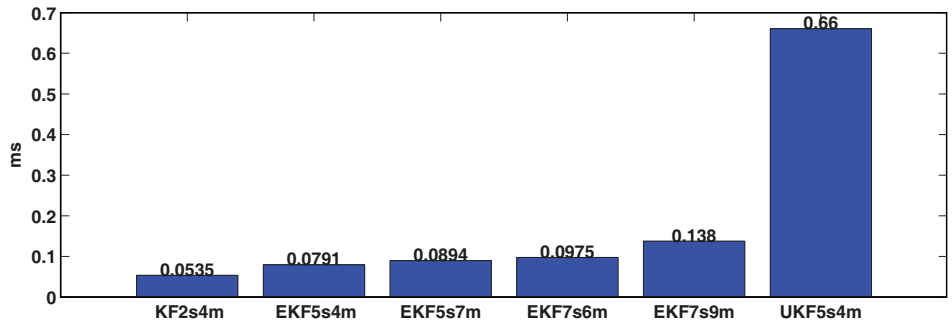


Figura 9.12: Tiempo promedio de ejecución, simulación mediante Matlab en un ordenador, gráfica comparativa para el robot diferencial

raciones como estados, EKF7s9m) y finalmente el filtro UKF con entradas asignadas y estimación local (UKF5s4m).

En esta prueba muestra claramente el costo computacional de los distintos algoritmos propuestos y tradicionales en cuanto al tiempo que se requiere para calcular las fusión sensorial y obtener la estimación de la postura, de la cual se puede apreciar que los algoritmos en cascada propuestos (KF2s4m, EKF5s4m) utilizan menos recursos computacionales (al requerir un menor tiempo del procesador, medido en milisegundos ms) que los algoritmos tradicionales o con asignación de entradas (EKF5s7m, EKF7s6m, EKF7s9m, UKF5s4m). A pesar de requerir un menor tiempo de cómputo, los algoritmos en cascada presentan un desempeño y precisión en la estimación similar al esquema más complejo utilizando el UKF, tal y como se expuso en la prueba de desempeño (figuras 9.1 y 9.2).

La segunda prueba consiste en obtener el tiempo de ejecución de las tareas principales que operan *en* la unidad de control del robot NXT, al utilizar los algoritmos de fusión en cascada propuestos (KF2s4m, EKF5s4m), tanto en sus versiones temporales (algoritmos 8 y 6 *sin* $z_{k,p}$) como las que realizan la corrección global basada en eventos (EBGC, algoritmos 12 y 10, *con* $z_{k,p}$) y compararlas con el algoritmo EKF con asignación de entradas (EKF5s7m, algoritmo basado en el tiempo 3). Para el caso particular del EKF5s7m, su ejecución se realizó por secciones para obtener el tiempo promedio y al

considerar una medición global sin retardo (considerando una $z_{k,p}$ constante predefinida en el robot), esto debido a que las limitaciones en memoria y ancho de banda en la unidad de control NXT impiden la implementación y ejecución completa de este algoritmo en cada instante k. Se resume la configuración de los algoritmos utilizados en esta prueba en la tabla 9.5.

Tabla 9.5: Resumen prueba del tiempo de ejecución en la unidad de control NXT: robot diferencial, $T_s = 50ms$

Robot	Diferencial, Fig.8.1	**Modelos**	(3.22),(3.23) y (3.24)
Corrección ΔT	ecuación (8.2)	**Parámetros**	Tabla 8.1
Estados	$x_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T$, $x_{k,v} = \begin{bmatrix} v & \omega \end{bmatrix}_k^T$	**Entradas**	$u_k = \begin{bmatrix} a & \alpha \end{bmatrix}_k^T$
Mediciones	$z_{k,p} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_{GM}^T$ $z_{k,v} = \begin{bmatrix} v_{enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T$		

Abreviatura	**Algoritmo**	**Estado** x_k	**Medición** z_k	**Postura** L_k
KF(2s4m)	Alg. 8	$x_{k,v}$	$z_{k,v}$	modelo (3.11)
KF(2s4m) EBGC	Alg. 12	$x_{k,v}$	$[z_{k,p} \quad z_{k,v}]^T$	modelo (3.11)
EKF(5s4m)	Alg. 6	$[x_{k,p} \quad x_{k,v}]^T$	$z_{k,v}$	$x_{k,p}$
EKF(5s4m) EBGC	Alg. 10	$[x_{k,p} \quad x_{k,v}]^T$	$[z_{k,p} \quad z_{k,v}]^T$	$x_{k,p}$
EKF5s7m	Alg. 3	$[x_{k,p} \quad x_{k,v}]^T$	$[z_{k,p} \quad z_{k,v}]^T$	$x_{k,p}$

Se obtienen en esta prueba el tiempo de ejecución de las tareas de lectura de los sensores (incluyendo calibrado y preprocesamiento, *TRead*), el filtro de fusión implementado (*TKalman*), el algoritmo de control (navegación y acción de control en los motores, *TControl*) y la tarea de escritura que almacena las variables requeridas para la supervisión en un archivo de texto (*TWrite*), tal y como se muestra en la figura 9.13.

Los tiempos de ejecución se miden en cada ciclo del algoritmo (cada $T_s = 50ms$) durante la duración de la prueba de 1min, al no ser constantes, se representan en la figura 9.13 tres casos: el *caso* **a** representa el tiempo promedio de ejecución de la tarea por ciclo, el *caso* **b** corresponde al tiempo máximo de ejecución medido (peor ejecución durante 1min), y el *caso* **c** muestra los tiempos máximos de cada tarea (el máximo durante la prueba de 1min) aunque estos *no* se presentan simultáneamente en ningún ciclo de ejecución (inclusive, no se presentan simultáneamente durante la prueba de 30min de las figuras 9.6 y 9.7), pero es un indicador del peor caso posible.

Los resultados de la figura 9.13 muestran que únicamente los métodos propuestos (KF2s4m y EKF5s4m) cumplen el tiempo de muestreo de $50ms$ para los casos a y b, siendo el filtro KF (KF2s4m, en cascada con modelos en cascada) el que consume menos recursos. El caso del filtro EKF (en cascada reducido, EKF5s4m) cumple con T_s en todos casos excepto en el caso c, pero este es un caso teórico el cual nunca se presentó en las pruebas realizadas. Se observa además que el esquema basado en eventos (EBGC) produce un leve aumento en el tiempo de ejecución respecto a cuándo solo se realiza la estimación con la información local. A pesar de esto, no se excede el T_s durante la ejecución, permitiendo el correcto funcionamiento del robot al navegar en el entorno (según se expuso en las pruebas de desempeño).

En cuanto al filtro EKF5s7m (con entradas asignadas), se excede el T_s en todos los casos para la unidad de control NXT, por lo que aunque se pudiera implementar éste filtro, no sería adecuado para el T_s utilizado. De esta forma y tomando en consideración la prueba de la figura 9.12, los filtros más complejos (EKF7s6m, EKF7s9m y UKF5s4m) excederían también el T_s si pudieran implementarse en la unidad de control, lo cual no es factible debido a las limitaciones en la memoria de la unidad NXT al realizar el cálculo de K_k en cada ciclo.

Comprobado el buen desempeño de los algoritmos propuestos en el robot diferencial, así como el ahorro que producen en el tiempo de ejecución al compararse con los métodos tradicionales, se realizan las pruebas en la plataforma Ackerman según se expone en el capítulo siguiente.

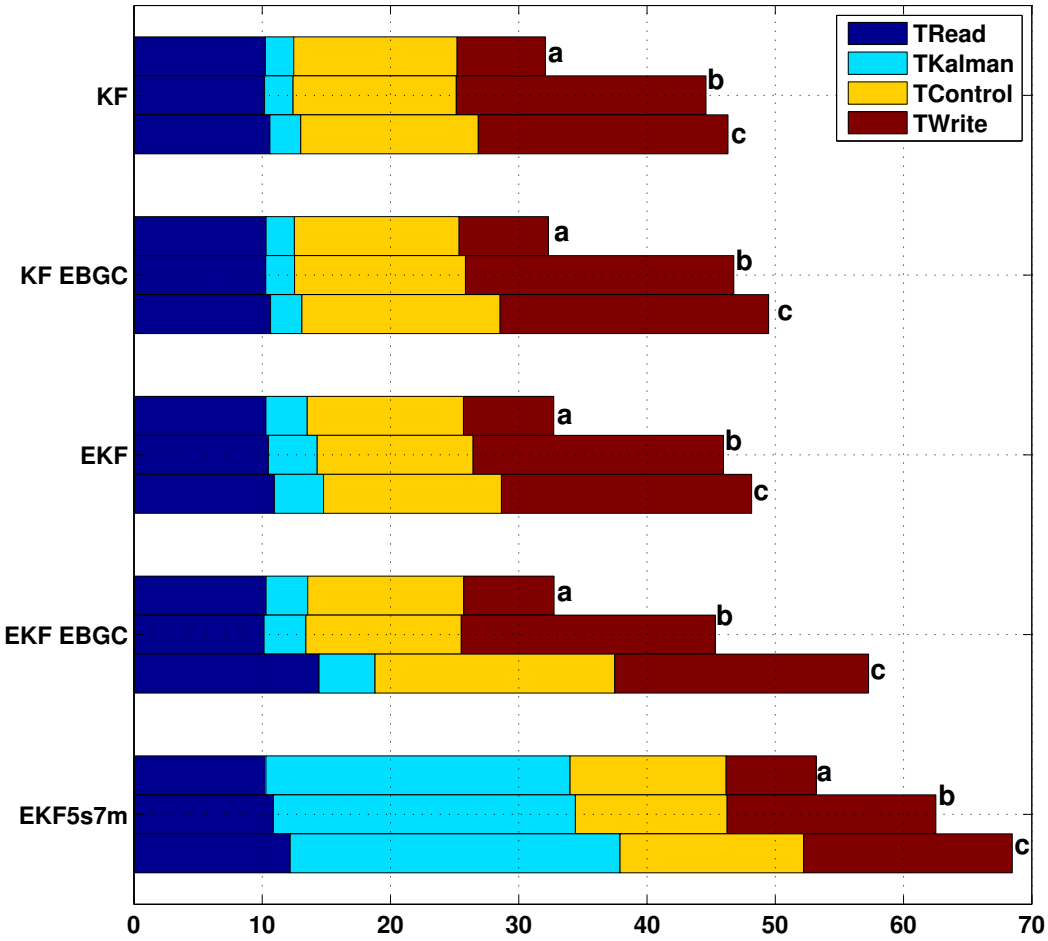


Figura 9.13: Tiempos de ejecución de las distintas tareas realizadas en la unidad de control NXT durante 1min, robot diferencial, caso *a*: tiempo promedio, caso *b*: tiempo de la peor ejecución y caso *c*: tiempos máximos de cada tarea

9.3 Conclusiones del Capítulo

En el presente capítulo se realizaron diversas pruebas experimentales utilizando el robot diferencial de navegación en interiores (figura 8.1) y el esquema de localización global con la cámara cenital (figura 8.4), incluyendo diversas pruebas de desempeño de los algoritmos propuestos así como las pruebas del consumo de recursos en la plataforma, tanto en tiempo de ejecución del algoritmo como en el uso del ancho de banda:

- A partir de la prueba de simulación en Matlab que utiliza las mediciones de los sensores del robot y la cámara cenital de las figuras 9.1 y 9.2, se concluye que el filtro de Kalman con modelos en cascada propuesto (algoritmo 8, tabla 9.1) tiene un desempeño adecuado en cuanto su estimación es similar a la postura real (obtenida de la cámara cenital) y a los esquemas más complejos de fusión (EKF, UKF), pero utilizando menos recursos computacionales. Por esta razón, se concluye que no hay una diferencia relevante en cuanto a la precisión de la estimación de la postura entre las distintas versiones del filtro de Kalman en el caso del robot diferencial propuesto, con la gran ventaja de que el KF con modelos en cascada sí puede ser implementado en la unidad de control del robot NXT. Queda en evidencia además, la baja precisión de la estimación a partir de los encoders en esta plataforma, no siendo conveniente utilizar estos sensores como única fuente de información.

- Se implementó el filtro KF (Algoritmo 8) en la unidad de control NXT, con lo que se comprobó su funcionamiento correcto, realizando una estimación adecuada de la postura con la información local, según se observa en las figuras 9.3 y 9.4.

- A partir de la prueba de larga duración (figuras 9.5 a 9.8), se concluye que el método basado en eventos (KF EBGC, algoritmo 12, tabla 9.2) provee una estimación adecuada de la postura del robot, corrigiendo el error de estimación con la información del sensor global evitando que éste crezca indefinidamente (error acotado, sin divergencia), a diferencia del caso de la odometría y del filtro KF sin corrección global. Se observó además un funcionamiento adecuado del método por eventos

en la evolución de R_A (evento del área normalizada de los elipsoides 3σ), acotado según la figura 9.8). Adicionalmente se observó una corrección adecuada del retardo de comunicación con el sensor global mediante el método simplificado de la ecuación (8.2).

- Según los valores obtenidos del índice IAE para la prueba de larga duración, se observó una reducción considerable en este índice al utilizar el método basado en eventos según se expuso en la tabla 9.3. Además, se apreció un seguimiento de trayectorias y desempeño adecuado (con error acotado) en las pruebas adicionales de las figuras 9.9 y 9.10, así como en el video resumen con los resultados de la prueba de larga duración: `/http:///wks./gii./upv./es//cobami//webfm_/send//5`.

- A partir del estudio realizado de la variación de la precisión del esquema basado en eventos (índice IAE), y el ancho de banda utilizado para obtener la información global (número de consultas a la cámara) ante la modificación límite del evento $R_{A,lim}$ (figura 9.11), se concluye que existe un compromiso entre la precisión y el uso del ancho de banda que puede regularse mediante el valor de $R_{A,lim}$. El IAE crece según crece $R_{A,lim}$ ya que se realizan menos llamadas al sensor global (uso reducido del ancho de banda y recursos), por otra parte, conforme $R_{A,lim}$ decrece, se incrementan el número de consultas al sensor global lo que a su vez produce una reducción en el IAE, incrementando la precisión del método hasta un límite que depende de la precisión del sensor y de los recursos de la plataforma (siendo el límite teórico el obtenido al utilizar una estrategia basada en el tiempo, realizando la corrección global en cada instante k). Para el caso de la unidad de control NXT se estableció un rango adecuado para el límite del evento en $0{,}125 < R_{A,lim} < 2$.

- Finalmente, a partir de las pruebas del tiempo de ejecución (simulación de la figura 9.12 con la tabla 9.4, y la implementación en la unidad de control NXT de la figura 9.13 con tabla 9.5) se concluye que los algoritmos propuestos (EKF en cascada reducido: algoritmos 6 y 10, y el KF en cascada con modelos en cascada: algoritmos 8 y 12) utilizan menos recursos computacionales al requerir un menor tiempo del procesador para realizar la fusión (medido en milisegundos ms) que los algorit-

mos tradicionales o con asignación de entradas (EKF5s7m, EKF7s6m, EKF7s9m y UKF5s4m), pero obtienen un desempeño y precisión similar a estos esquemas según expuso en la prueba de desempeño (figuras 9.1 y 9.2).

- A diferencia de los métodos existentes, los algoritmos basados en eventos propuestos son implementables en robots de recursos limitados (NXT en el caso del robot diferencial), debido a que actualizan la postura con la información global haciendo un uso eficiente del medio de comunicación, de forma que se puede utilizar para otras tareas relevantes dentro del robot. Además no se incumple con el tiempo de muestreo en la ejecución de múltiples tareas según se comprobó en la figura 9.13. Esto se consiguió mediante la selección de un valor adecuado del $R_{A,lim}$, con el que se estima la postura con un valor aceptable del IAE (y en el error porcentual) a la vez que se ahorran recursos computacionales y ancho de banda, realizando únicamente las consultas necesarias al sensor global para corregir la estimación local.

10 | Aplicaciones: Localización del Robot Ackerman

Se presentan a continuación diversas pruebas experimentales utilizando el robot Ackerman de navegación en interiores (figura 8.2) y el esquema de localización global con la cámara cenital (figura 8.4) así como el robot Ackerman para navegación en exteriores (figura 8.6), con el fin de comprobar el desempeño correcto de los algoritmos de fusión basados en eventos propuestos, así como el consumo de recursos en esta plataforma midiendo el respectivo tiempo de ejecución de los algoritmos.

10.1 Pruebas de Desempeño

10.1.1 Comprobación de la fusión local

Siguiendo el procedimiento realizado para el robot diferencial, se comprueba primeramente el desempeño de la fusión local en el robot Ackerman sin utilizar la corrección global basada en eventos ($EBGC$), al realizar una prueba con el robot siguiendo una trayectoria cuadrada mientras se almacenan las mediciones locales y la información del sensor global (cámara cenital). Utilizando estas mediciones se realiza una simulación en Matlab®, utilizando los algoritmos configurados según la tabla 10.1 con la que se obtiene la figura 10.1 en la cual se indica además la medición global de la cámara y la estimación odométrica a partir de los encoders.

Se observa de la figura 10.1 que todos los filtros simulados presentan un desempeño adecuado en cuanto a la estimación de la postura es cercana a la del sensor global. En el caso del EKF con entradas asignadas (EKF6e7m), al realizar en este caso la fusión utilizando la información global $z_{k,p}$ obtiene la estimación con menor error. Los demás algoritmos, el EKF en cascada reducido (EKF6e4m), el EKF en cascada con modelos en cascada (EKF3e4m) y

Tabla 10.1: Resumen prueba de desempeño: robot Ackerman LEGO NXT, simulación en Matlab, $T_s = 50ms$

Robot	Ackerman, Fig.8.2	**Modelos**	(3.49),(3.50)/(3.51) y *ns*: (3.52)/(3.53)
Corrección ΔT	——	**Parámetros**	Tabla 8.2
Entradas	$u_k = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}_k^T, \quad u_{k,ns} = \begin{bmatrix} u_1 & u_3 \end{bmatrix}_k^T$		
Estados	$x_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T, \quad x_{k,v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}_k^T, \quad x_{k,v,ns} = \begin{bmatrix} v_x & \omega \end{bmatrix}_k^T$		
Mediciones	$z_{k,p} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_{GM}^T, \quad z_{k,v} = \begin{bmatrix} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} \end{bmatrix}_k^T$		
	$z_{k,v,ns} = \begin{bmatrix} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T$		

Abreviatura	**Algoritmo**	**Entrada** u_k	**Estado** x_k	**Medición** z_k	**Postura** L_k
EKF6e7m	Alg. 3	u_k	$[x_{k,p} \quad x_{k,v}]^T$	$[z_{k,p} \quad z_{k,v}]^T$	$x_{k,p}$
EKF6e4m	Alg. 6	u_k	$[x_{k,p} \quad x_{k,v}]^T$	$z_{k,v}$	$x_{k,p}$
EKF3e4m	Alg. 7	u_k	$x_{k,v}$	$z_{k,v}$	modelo (3.36)
KF2e4m	Alg. 8	$u_{k,ns}$	$x_{k,v,ns}$	$z_{k,v,ns}$	modelo (3.11)

el KF en cascada con modelos en cascada (KF2e4m), muestran una medición cercana a la cámara, aunque es apreciable la necesidad de la información global con el fin de obtener una mejor precisión en la estimación de la postura, lo cual se obtiene mediante el uso de la corrección basada en eventos.

En el caso de la odometría a partir de los encoders, nuevamente se observa que no son adecuados para la estimación de la postura en esta plataforma. Se aprecia además la similitud de la estimación del filtro KF2e4m con la de los filtros más complejos, requiriendo menos recursos computacionales al utilizar el modelo sin deslizamiento (*ns*, ecuaciones (3.52) y (3.53)) el cual *no* considera la velocidad transversal del robot (movimiento baja velocidad, ver sección 3.2.2).

Comprobado el buen desempeño de la estimación utilizando la información local, se procede a implementar los algoritmos en cascada (EKF6e4m, EKF3e4m y KF2e4m) en la unidad de control NXT de la plataforma utilizando la corrección global basada en eventos (algoritmos 10, 11 y 12 respectivamente) según los parámetros descritos en la tabla 10.2 (agregando $z_{k,p}$ al vector de medición z_k).

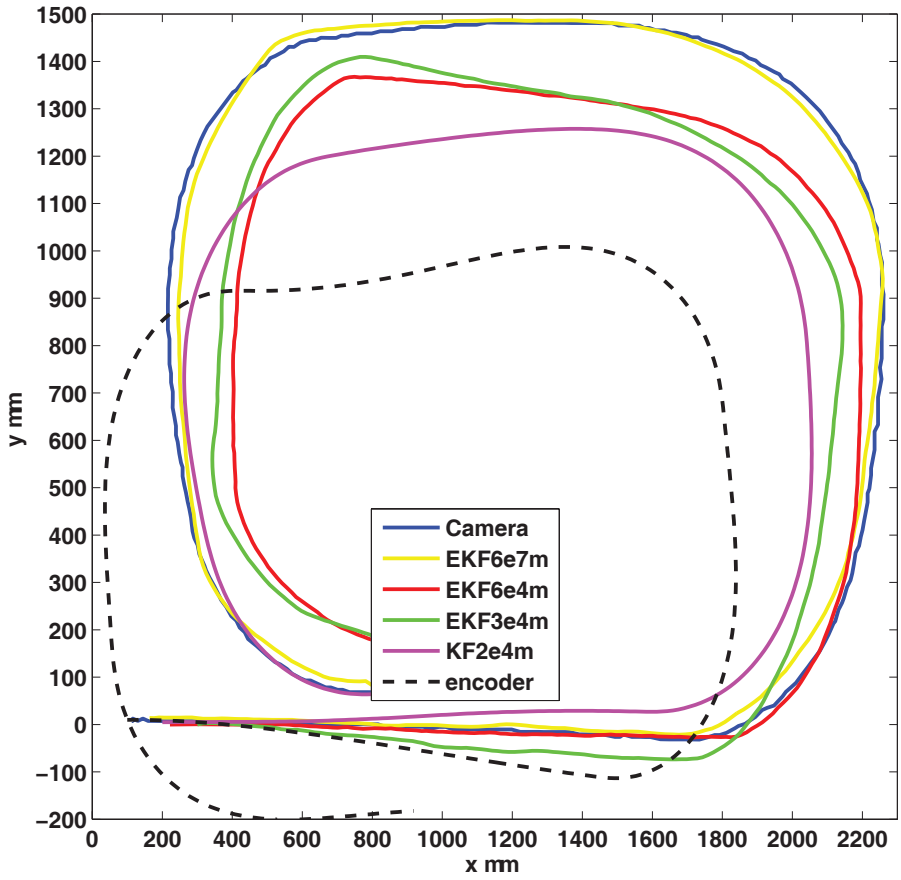


Figura 10.1: Desempeño de los algoritmos de localización, robot Ackerman, prueba simulada, trayectoria de referencia cuadrada.

Tabla 10.2: Resumen prueba de desempeño: robot Ackerman LEGO NXT, algoritmos basados en eventos implementados en la unidad de control NXT, $T_s = 50ms$

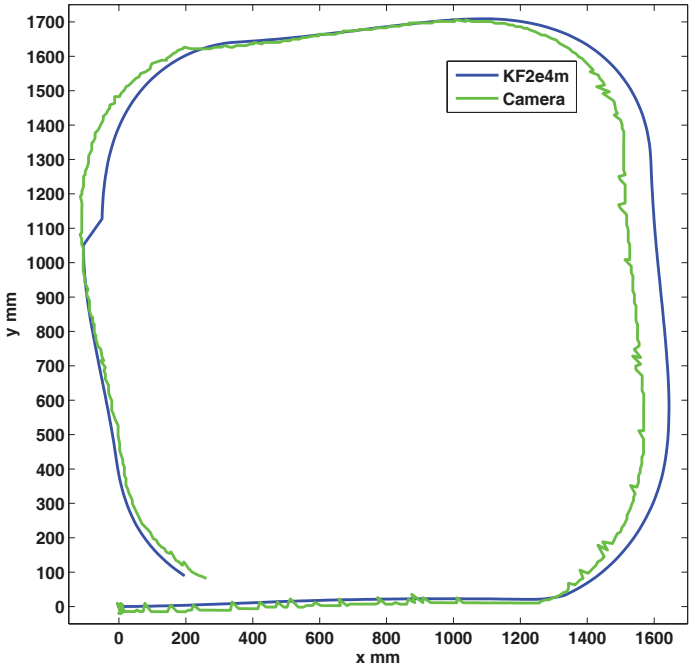
Robot	Ackerman, Fig.8.2	**Modelos**	(3.49),(3.50)/(3.51) y *ns*: (3.52)/(3.53)
Corrección ΔT	ecuación (8.1)	**Parámetros**	Tabla 8.2
Entradas	$u_k = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}_k^T$, $u_{k,ns} = \begin{bmatrix} u_1 & u_3 \end{bmatrix}_k^T$		
Estados	$x_{k,p} = \begin{bmatrix} x & y & \theta \end{bmatrix}_k^T$, $x_{k,v} = \begin{bmatrix} v_x & v_y & \omega \end{bmatrix}_k^T$, $x_{k,v,ns} = \begin{bmatrix} v_x & \omega \end{bmatrix}_k^T$		
Mediciones	$z_{k,p} = \begin{bmatrix} x_k & y_k & \theta_k \end{bmatrix}_{GM}^T$, $z_{k,v} = \begin{bmatrix} v_{x,enc} & v_{y,enc} & \omega_{enc} & \omega_{gyr} \end{bmatrix}_k^T$		
	$z_{k,v,ns} = \begin{bmatrix} v_{x,enc} & \omega_{enc} & \omega_{gyr} & \omega_{comp} \end{bmatrix}_k^T$		

Abreviatura	**Algoritmo**	**Entrada** u_k	**Estado** x_k	**Medición** z_k	**Postura** L_k
EKF6e4m EBGC	Alg. 10	u_k	$[x_{k,p} \quad x_{k,v}]^T$	$[z_{k,p} \quad z_{k,v}]^T$	$x_{k,p}$
EKF3e4m EBGC	Alg. 11	u_k	$x_{k,v}$	$[z_{k,p} \quad z_{k,v}]^T$	modelo (3.36)
KF2e4m EBGC	Alg. 12	$u_{k,ns}$	$x_{k,v,ns}$	$[z_{k,p} \quad z_{k,v,ns}]^T$	modelo (3.11)

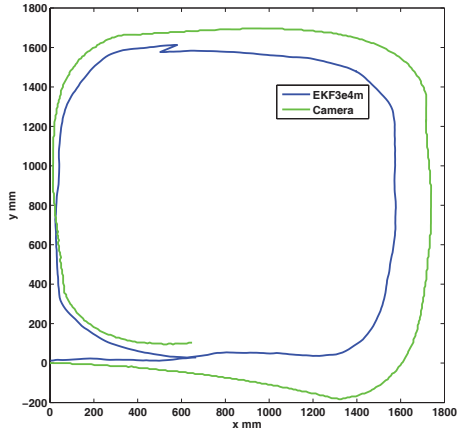
En esta plataforma se utiliza el filtro KF/EKF para incorporar la información global, de forma que se actualicen correctamente tanto la postura como la covarianza del error de estimación de la misma, $P_{k,p}$. Por esta razón se utiliza la ecuación (8.1) para corregir el retardo en la medición global, que es utilizada por el filtro global para la corrección de la postura. Esto a diferencia de las pruebas realizadas para el robot diferencial, en las que *no* se utiliza un filtro KF/EKF para incorporar la información global (junto con la ecuación (8.1)), sino que se incorpora directamente utilizando la ecuación (8.2) tal y como se describió en los capítulos 8 y 9.

De esta forma, se realiza la prueba con el robot siguiendo la trayectoria cuadrada y los algoritmos basados en eventos configurados según la tabla 10.2, con lo que se obtienen los resultados mostrados en la figura 10.2. En ésta se aprecia el buen desempeño de todos los algoritmos en cascada por eventos propuestos, realizando una estimación adecuada de la postura la cual es cercana a la determinada por el sensor global, mejorando la precisión comparado al caso en la que no se utiliza la corrección basada en eventos (figura 10.1).

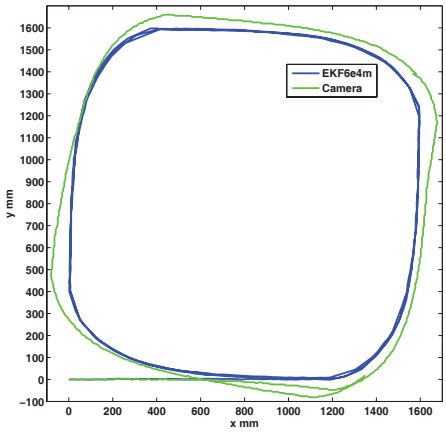
Además, se tiene la ventaja relevante de que los algoritmos basados en eventos son ejecutados en la unidad de control NXT del robot, gracias al uso



(a) EKF2e4m EBGC



(b) EKF3e4m EBGC



(c) EKF6e4m EBGC

Figura 10.2: Desempeño de los algoritmos en cascada basados en eventos (EBGC) implementados en la unidad de control NXT del robot Ackerman.

reducido de los recursos computacionales (memoria, procesador) y ancho de banda, ya que se utiliza la información global solo en caso necesario (si $R_A > R_{A,lim}$). Esto a diferencia del algoritmo con asignación de entradas EKF6e7m (o los tradicionales) que no pueden ser implementados en ésta plataforma Ackerman.

Realizada la comprobación de los algoritmos tanto por simulación como al ser implementados en la unidad de control del robot, se procede con las pruebas de larga duración según se expone a continuación.

10.1.2 Prueba de larga duración, algoritmo basado en eventos

De forma similar al robot diferencial, se realiza la prueba de larga duración en el robot Ackerman al seguir una trayectoria cuadrada doble durante 30min, en la cual se estima la postura del robot utilizando el algoritmo menos demandante de recursos: el algoritmo KF en cascada con modelos en cascada basado en eventos (KF2e4m EBGC), el cual no considera el deslizamiento en la plataforma. En esta prueba y a diferencia de las pruebas del robot diferencial, se utiliza la ecuación (8.1) para corregir el retardo en la medición global, tal y como se muestra en la tabla resumen 10.3 con la configuración de los algoritmos de la prueba.

Tabla 10.3: Resumen prueba de larga duración: robot Ackerman LEGO NXT, algoritmos basados en eventos implementados en la unidad de control NXT, $T_s = 50ms$

Robot	Ackerman, Fig.8.2	**Modelos**	(3.49),(3.50)/(3.51) y *ns*: (3.52)/(3.53)
Corrección ΔT	ecuación (8.1)	**Parámetros**	Tabla 8.2
Entradas	$u_k = \left[\ u_1\ u_2\ u_3 \right]_k^T,\quad u_{k,ns} = \left[\ u_1\ u_3 \right]_k^T$		
Estados	$x_{k,p} = \left[x\ y\ \theta \right]_k^T,\quad x_{k,v} = \left[v_x\ \ v_y\ \omega\ \right]_k^T,\quad x_{k,v,ns} = \left[v_x\ \omega\ \right]_k^T$		
Mediciones	$z_{k,p} = \left[\ x_k\ y_k\ \theta_k\ \right]_{GM}^T,\quad z_{k,v} = \left[\ v_{x,enc}\ v_{y,enc}\ \omega_{enc}\ \omega_{gyr}\ \right]_k^T$		
	$z_{k,v,ns} = \left[\ v_{x,enc}\ \omega_{enc}\ \omega_{gyr}\ \omega_{comp}\ \right]_k^T$		

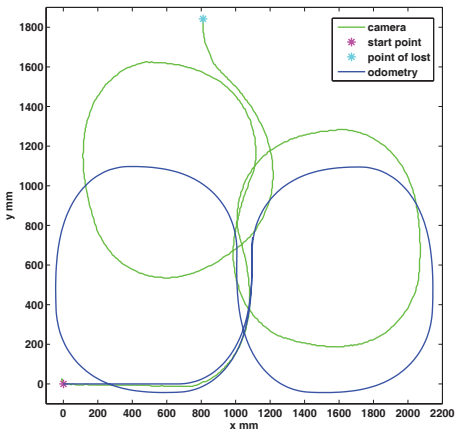
Abreviatura	**Algoritmo**	**Entrada** u_k	**Estado** x_k	**Medición** z_k	**Postura** L_k
Odometry	modelo (3.39)	—	—	$[v_{R,enc}\quad \phi_{F,enc}]^T$	modelo (3.36)
KF2e4m	Alg. 12	$u_{k,ns}$	$x_{k,v,ns}$	$z_{k,v,ns}$	modelo (3.11)
KF2e4m EBGC	Alg. 12	$u_{k,ns}$	$x_{k,v,ns}$	$[z_{k,p}\quad z_{k,v,ns}]^T$	modelo (3.11)

Con esto se obtienen los resultados mostrados en la figura 10.3 en la que se realiza la comparación de la estimación utilizando la odometría (figura 10.3(a)) con la estimación sin información global (KF2e4m figura 10.3(b)) y con el método basado en eventos (KF2e4m EBGC 10.3(c)). En estas figuras se indican la estimación calculada por el robot para la primera vuelta (1min aproximadamente, color azul) y la medición de la cámara cenital (para los 30min, color verde).

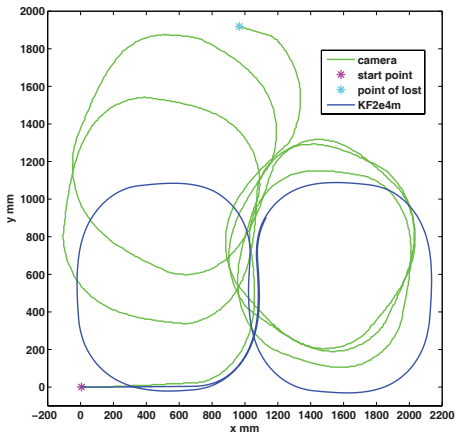
Se observa en el resultado de esta prueba el correcto funcionamiento del esquema basado en eventos (figura 10.3(c)) en cuanto se corrige el error de estimación utilizando la información del sensor global (ecuación (8.1)) cuando el evento R_A basado en los elipsoides 3σ supera el límite $R_{A,lim} = 1{,}2$. Bajo este esquema, la precisión en la estimación de la postura es adecuada en cuanto el error de estimación no crece indefinidamente, sino que es acotado durante la duración de la prueba, por lo que la trayectoria del robot se asemeja a la referencia cuadrada y no diverge como en los casos de la odometría (figura 10.3(a)) y el filtro sin información global (figura 10.3(b)).

El buen desempeño del algoritmo de estimación basado en eventos (KF2e4m EBGC) se observa además en la figura 10.4 en la que se presenta el resultado de la prueba de larga duración cuando se establece una trayectoria de referencia rectangular. Cabe destacar que el error de estimación puede disminuirse si fuera necesario (según los requerimientos de la aplicación) al disminuir el valor de $R_{A,lim}$ de forma que se incremente la precisión de la estimación de la postura. Esto se realiza como un compromiso entre la precisión deseada y el ancho de banda (y recursos computacionales) disponibles en la plataforma, tal y como se describió en la sección 9.1.3 (figura 9.11).

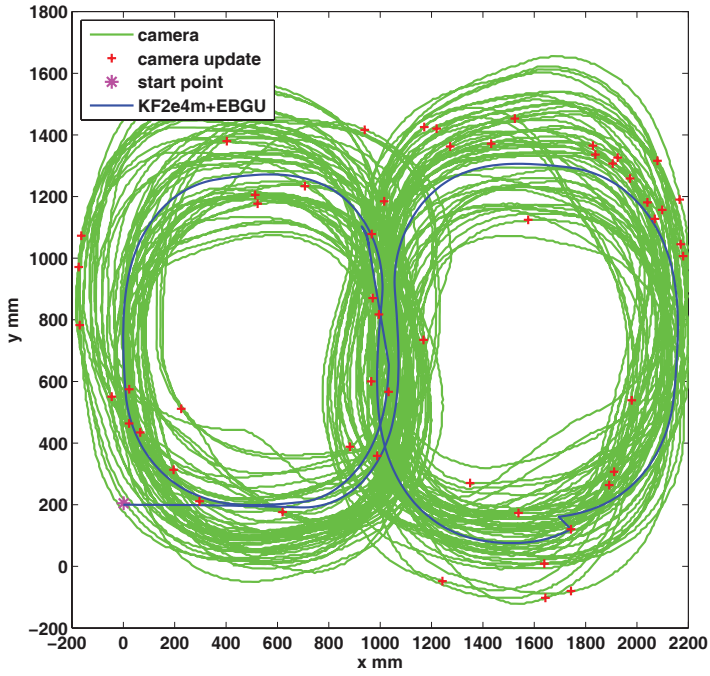
Con el buen resultado obtenido en estas pruebas de larga duración que muestran la precisión adecuada del filtro de fusión basado en eventos KF2e4m, así como su estabilidad y convergencia, los algoritmos más complejos en cascada (KF6e4m y KF3e4m) tendrán también un desempeño adecuado en la estimación de la postura. Finalmente, un video con el resumen de las pruebas de larga duración (figuras 10.3 y 10.4) puede encontrarse en `/http://wks./gii./upv./es//cobami//webfm_/send//6`. Con esto, se procede a continuación con las pruebas en exteriores.



(a) Odometría



(b) KF2e4m



(c) KF2e4m+EBGC, $R_A = 1{,}2$

Figura 10.3: Prueba de larga duración, seguimiento de la trayectoria cuadrada doble durante 30min, robot Ackerman navegando en interiores.

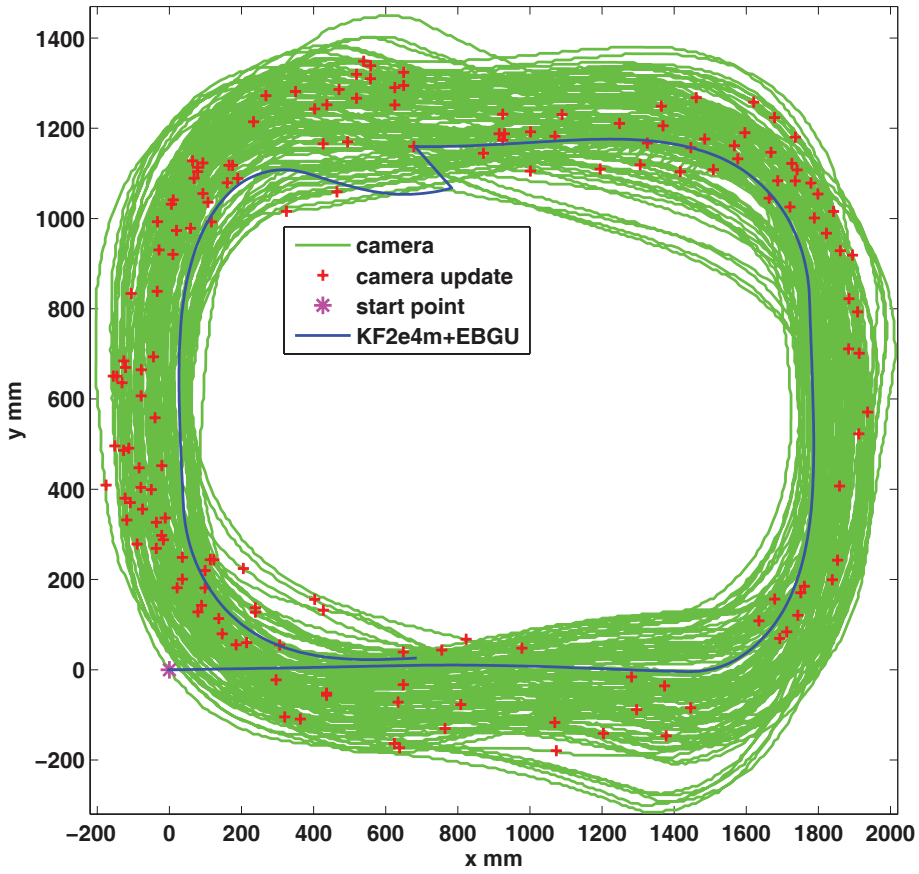


Figura 10.4: Prueba de larga duración, seguimiento de la trayectoria rectangular durante 30min, robot Ackerman navegando en interiores.

10.1.3 Prueba en exteriores

Previo a la prueba en exteriores, se comprueba el funcionamiento del robot Ackerman (figura 8.6) en una prueba de larga duración (30min) en interiores, utilizando el algoritmo KF2e4m con la corrección global basada en eventos (EBGC, tabla 10.3) y la información de la cámara cenital (ecuación (8.1), figura 8.4) en lugar del GPS. El resultado de esta prueba se muestra en la figura 10.5 en donde se muestra la primera vuelta estimada por el KF2e4m (color azul) y las 30 vueltas medidas por la cámara (color verde). Se observa que el algoritmo propuesto tiene un comportamiento adecuado, estimando la postura de forma similar a la medida por la cámara, lo que produce un seguimiento suave de la trayectoria cuadrada durante toda la prueba.

Para la prueba en exteriores se realiza un recorrido en sentido horario sobre una pista deportiva de superficie plana, utilizando el GPS IG-500N sobre el LEGO como fuente de información global y agregando al evento basado en el área 3σ la condición $N_{sat} > N_{sat,min}$ (con $N_{sat,min} = 6$) y el ajuste dinámico de la matriz R (a partir del valor nominal de la covarianza del sensor, disminuyendo su valor si N_{sat} es alto y aumentándolo en caso contrario) según se describió en la definición del evento (sección 6.2.2). Además, se obtiene la trayectoria seguida por el LEGO con una precisión alta, al ser medida con un GPS diferencial con el fin de validar los resultados obtenidos. Cabe destacar que la información de éste sensor no es utilizada dentro de la fusión. De esta forma, para la prueba en exteriores se modifica la configuración de la plataforma para incorporar temporalmente el receptor del GPS diferencial, tal y como se muestra en la figura 10.6.

Los resultados de esta prueba con el robot modificado con el receptor del GPS diferencial (figura 10.6) se muestran en la figura 10.7, en la que se observa un desempeño adecuado del filtro de fusión, con lo que la postura estimada es similar a la medida del GPS diferencial.Según aumenta el error en la estimación (covarianza del error de la postura del robot) se genera el evento ($R_A > R_{A,lim} \& N_{sat} > N_{sat,min}$) que produce la actualización utilizando información global y por lo tanto se elimina el error acumulado en la estimación local.

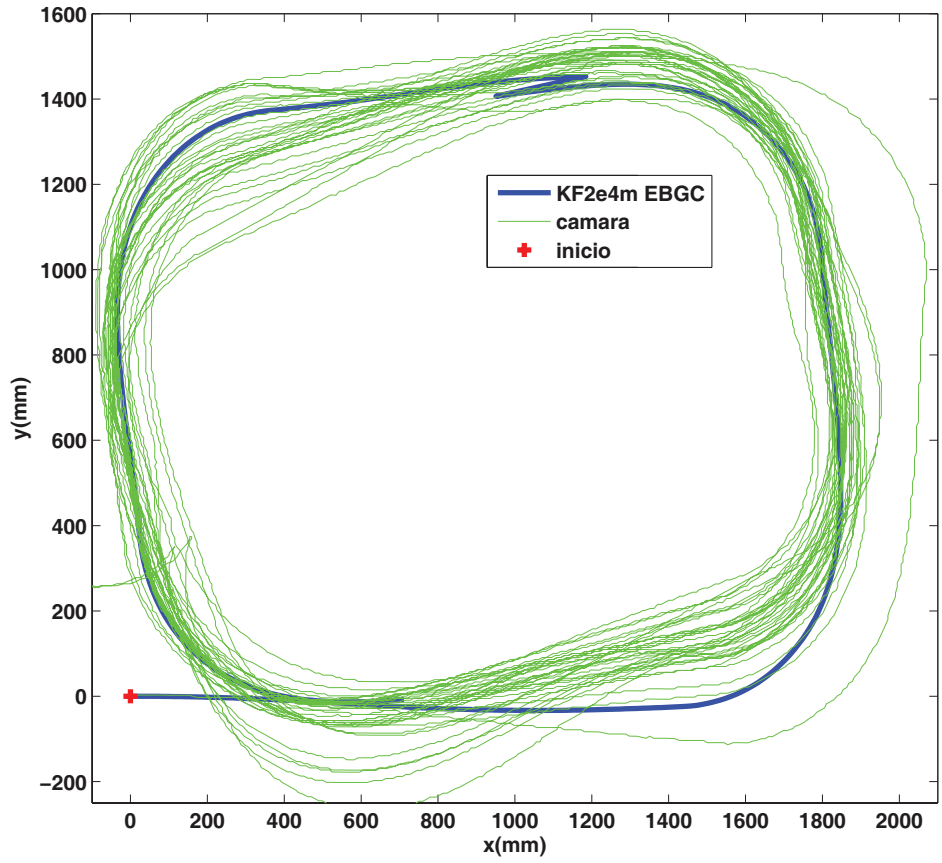


Figura 10.5: Prueba en interiores de larga duración del robot Ackerman para exteriores, utilizando la cámara cenital como sensor global.

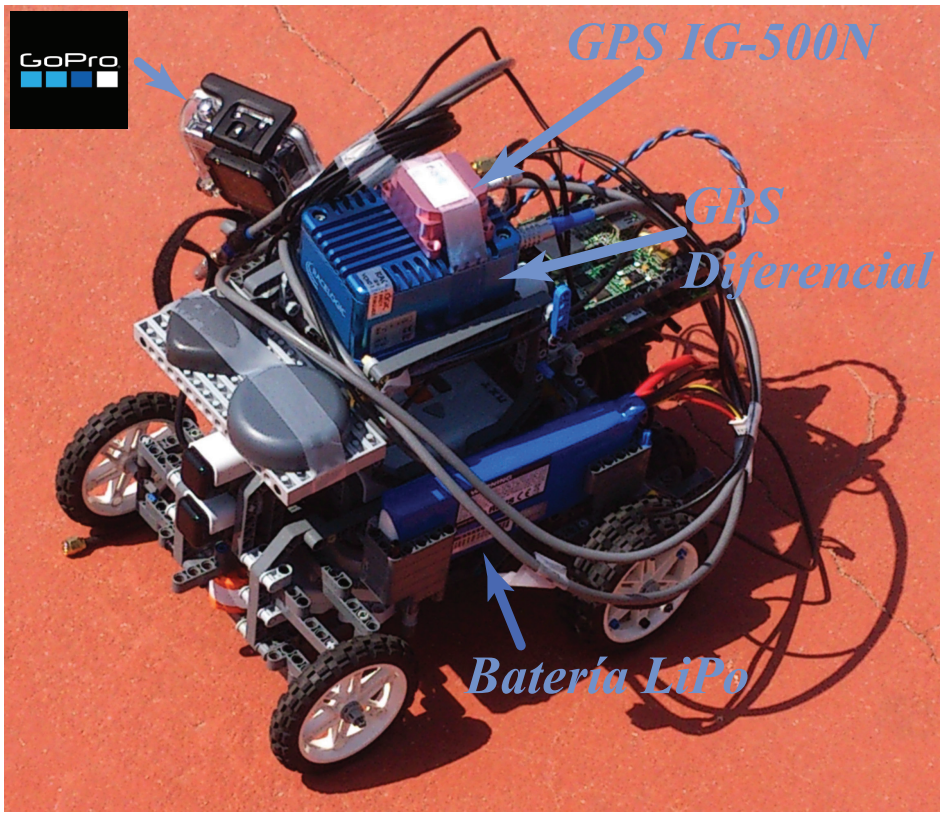


Figura 10.6: Plataforma Ackerman en exteriores agregando el GPS diferencial para comprobación de la trayectoria navegada.

Además, se obtiene una precisión aceptable según las gráficas del error y error absoluto (respecto al GPS, figuras 10.8 y 10.9) siendo ambos acotados, y con la ventaja de evitar la actualización de la estimación utilizando la información del GPS en cada instante de muestreo, lo que requeriría más recursos de los disponibles en la plataforma.

A pesar del buen comportamiento se observa que el GPS utilizado no es tan preciso como el GPS diferencial por lo que, aunque se utilice la información global, existe un error inherente al sensor utilizado que puede ser considerable y que se trasladaría a la estimación del algoritmo de fusión. Por esta razón, en el caso de navegación en exteriores, deberán utilizarse sensores de precisión alta, con el fin de mejorar la estimación de la postura con el filtro local y además se debe realizar el ajuste dinámico de los términos de la matriz R correspondientes al GPS, utilizando información adicional al número de satélites N_{sat} (por ejemplo: la covarianza dinámica interna del sensor si estuviese disponible, la información meteorológica del entorno o de la presencia de edificios aledaños [55, 153]) de forma que se limite la influencia del error en el sensor global sobre la estimación de la postura del robot. Cabe destacar además, que la actualización se puede hacer de forma más frecuente (utilizando un $R_{A,lim}$ menor) para eliminar el bias de forma más regular, siempre y cuando la medición del GPS se obtenga con una precisión adecuada, esto se realiza según los requerimientos de la misión del robot.

Concluidas las pruebas de desempeño se procede con las pruebas del tiempo de ejecución del algoritmo tal y como se expone a continuación.

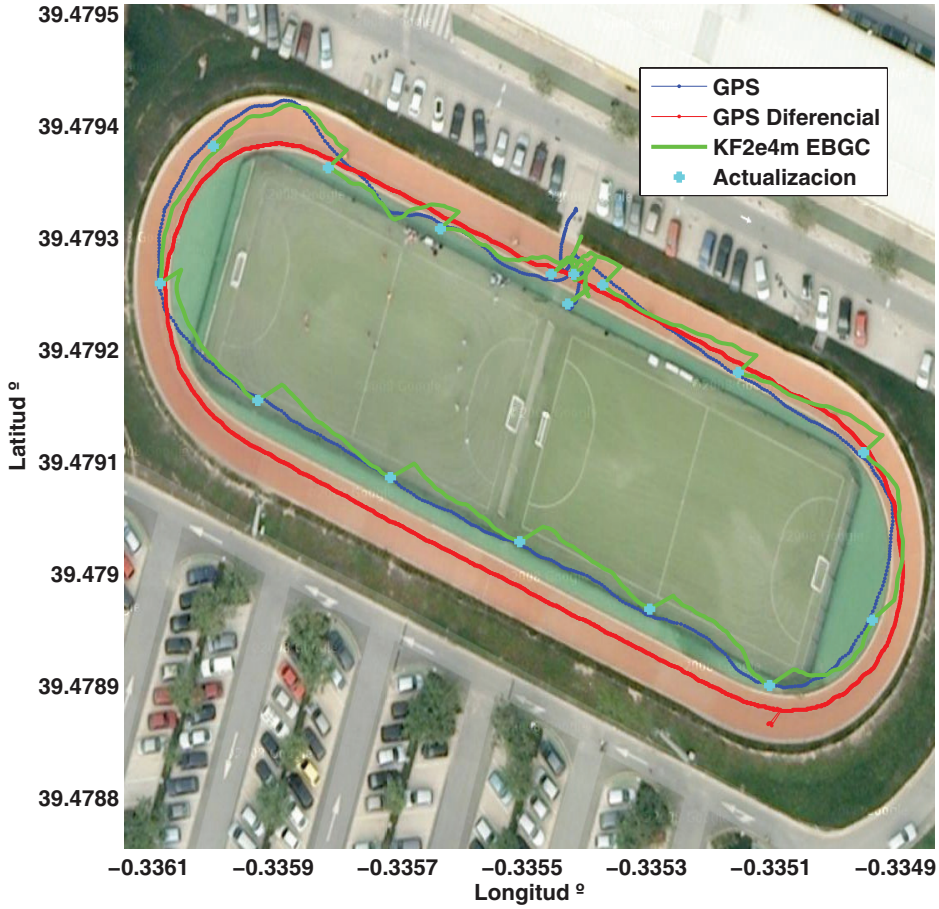


Figura 10.7: Prueba en exteriores con el GPS diferencial, trayectorias georeferenciadas sobre la imagen satelital de la pista deportiva obtenida de Google Earth®.

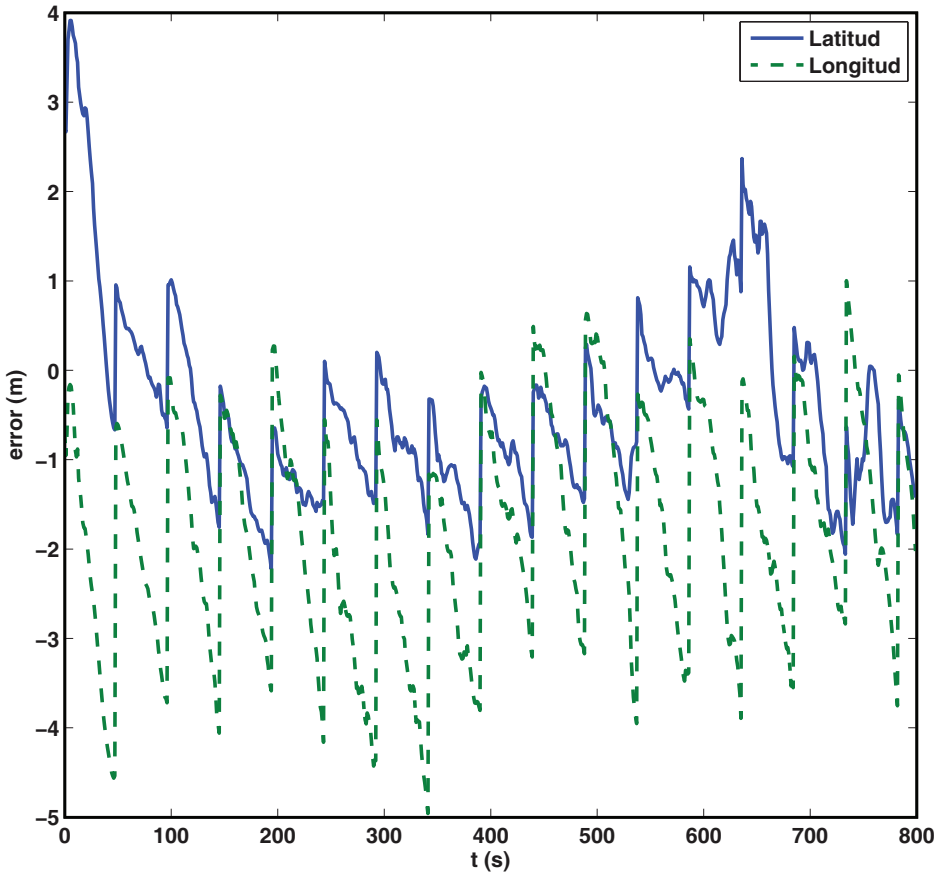


Figura 10.8: Prueba en exteriores, evolución del error de estimación.

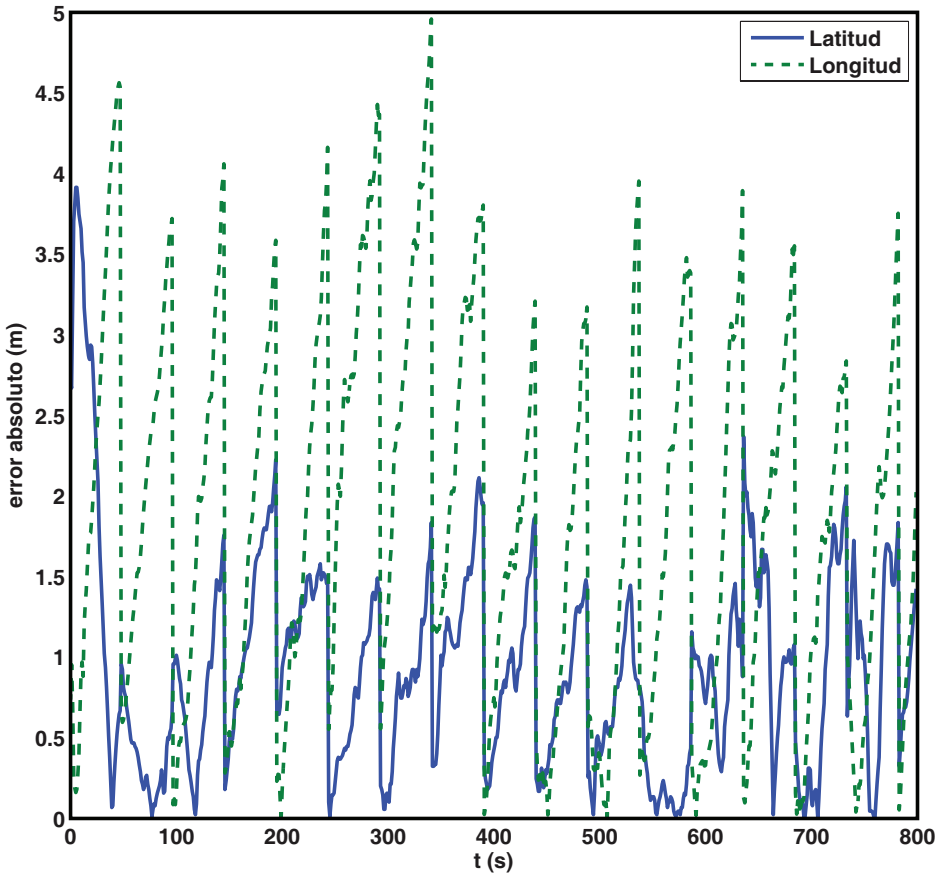


Figura 10.9: Prueba en exteriores, evolución del error absoluto de estimación.

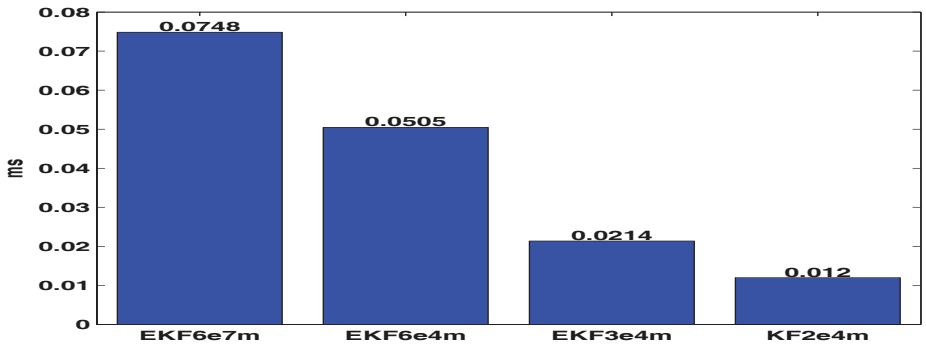


Figura 10.10: Tiempo promedio de ejecución, simulación mediante Matlab en un ordenador, gráfica comparativa para el robot Ackerman.

10.2 Pruebas de tiempo de ejecución

De forma similar al robot diferencial, se realizan dos pruebas experimentales para comprobar la eficiencia de los algoritmos propuestos en cuanto al tiempo de ejecución que requieren para realizar la estimación de la postura, según se describe a continuación.

La primera prueba es la simulación de los algoritmos de fusión basados en el tiempo (EKF6e7m, EKF6e4m, EKF3e4m y KF2e4m, tabla 10.1, figura 10.1) en donde se obtiene el tiempo de ejecución promedio, tal y como se muestra en la figura 10.10. Los algoritmos son implementados en un ordenador (2.4GHz y 4GB RAM) mediante Matlab®, el cual ejecuta los distintos filtros utilizando la información experimental obtenida de la prueba de desempeño simulada (trayectoria cuadrada, figura 10.1, tabla 10.1).

Nuevamente, en esta prueba se observa que los algoritmos en cascada con modelos en cascada (EKF3e4m y KF2e4m) utilizan menos recursos computacionales para realizar la estimación de la postura (al requerir un menor tiempo del procesador, medido en milisegundos ms) que los algoritmos más complejos (en cascada reducido EKF6s4m y con asignación de entradas EKF6s7m). A pesar de requerir un menor tiempo de cómputo, los algoritmos en cascada con modelos en cascada presentan un desempeño y precisión en la estima-

ción similar los esquemas complejos, tal y como se expuso en la prueba de desempeño (figuras 10.1 y 10.2).

La segunda prueba corresponde a la medición del tiempo de ejecución obtenido para las tareas principales que operan *en* la unidad de control del robot NXT, al utilizar los algoritmos de fusión en cascada propuestos (EKF6s4m, EKF3e4m y KF2e4m), tanto en sus versiones temporales (*sin* $z_{k,p}$, tabla 10.1) como las que realizan la corrección global basada en eventos (EBGC *con* $z_{k,p}$, tabla 10.2), así como el modelo más tradicional con asignación de entradas (EKF6s7m), cuya ejecución se realiza por secciones debido a que las limitaciones en memoria y ancho de banda en la unidad de control NXT impiden la implementación y ejecución completa de este algoritmo en cada instante k. Los tiempos se obtienen durante el primer minuto de seguimiento de una trayectoria cuadrada similar a la de la figura 10.1.

De esta forma se determinan los tiempos de ejecución de las tareas del robot (lectura de sensores *TRead*, filtro de fusión *TKalman*, algoritmo de control *TControl* y la tarea de escritura para la supervisión *TWrite*) tal y como se muestra en la figura 10.11. Nuevamente se identifican tres casos en la figura, el caso *caso* **a** representa el tiempo promedio de ejecución por ciclo, el *caso* **b** corresponde al tiempo máximo de ejecución (peor ejecución), y el *caso* **c** con los tiempos máximos de cada tarea, los cuales *no* se presentan simultáneamente (ni en esta prueba ni durante la de larga duración, figuras 10.3 y 10.4), siendo un indicador del peor caso posible.

Se observa de la figura 10.11 que todos los métodos en cascada propuestos cumplen con el tiempo de muestreo de $50ms$ para todos los casos (a, b y c), utilizando o no el EBGC, siendo el algoritmo KF2e4m EBGC (cascada con modelos en cascada sin deslizamiento basado en eventos) el que consume menos recursos (menor tiempo de ejecución en todos los casos a, b y c). El EKF6s7m excede el T_s en todos los casos para la unidad de control NXT, no siendo adecuado para ser implementado en esta plataforma debido a las limitaciones en la memoria y cómputo de la unidad de control, principalmente cuando se debe realizar el cálculo de K_k en cada ciclo o comunicarse con el sensor global en cada instante k.

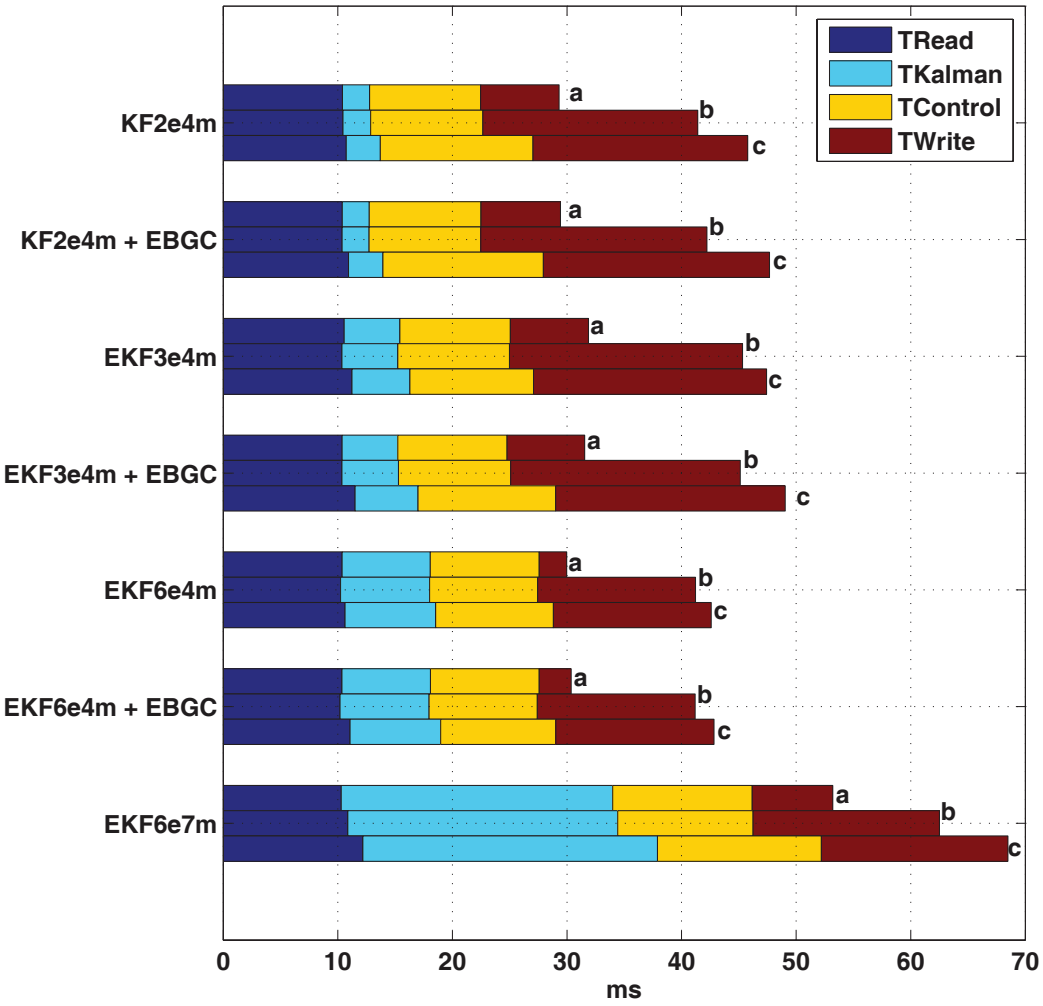


Figura 10.11: Tiempos de ejecución de las distintas tareas realizadas en la unidad de control NXT durante 1min, robot Ackerman, caso *a*: tiempo promedio, caso *b*: tiempo de la peor ejecución y caso *c*: tiempos máximos de cada tarea.

Comprobado el buen desempeño de los algoritmos propuestos en el robot Ackerman para navegación en interiores y exteriores, así como el ahorro que producen en el tiempo de ejecución al compararse con los métodos tradicionales, se procede a realizar las pruebas para el esquema de localización cooperativa utilizando la plataforma de simulación en JAVA y JADE, según se expone en el capítulo siguiente.

10.3 Conclusiones del Capítulo

En el presente capítulo se realizaron diversas pruebas experimentales utilizando el robot Ackerman de navegación en interiores (figura 8.2) y el esquema de localización global con la cámara cenital (figura 8.4), así como el robot Ackerman para navegación en exteriores (figura 8.6), incluyendo diversas pruebas de desempeño de los algoritmos propuestos así como las pruebas del consumo de recursos en la plataforma, midiendo el respectivo tiempo de ejecución de los algoritmos:

- A partir de la simulación en Matlab con las mediciones de los sensores del robot y la cámara cenital de la figura 10.1 se observó el buen desempeño de los filtros simulados, con una estimación de la postura cercana a la del sensor global. Se apreció la similitud en la estimación del algoritmo menos complejo computacionalmente (KF2e4m, sin deslizamiento) y los algoritmos más avanzados. De esta figura se concluye que la corrección basada en eventos propuesta es necesaria con el fin de incorporar la información global a la estimación local, para obtener una mejor precisión en la estimación de la postura. Además, se comprobó nuevamente que la odometría a partir de los encoders no provee una precisión adecuada en la estimación.

- En la prueba de la figura 10.2 se comprobó el buen desempeño de los algoritmos en cascada con corrección global basada en eventos (EBGC, algoritmos 10, 11 y 12, configuración en la tabla 10.2) implementados en la unidad de control NXT de la plataforma, en cuando realizan una estimación adecuada de la postura del robot, con precisión mejorada cuando se comprara con la figura 10.1 sin utilización del EBGC. Se

observó el funcionamiento adecuado de la corrección del retardo de comunicación mediante la ecuación (8.1), que permite el ajuste de la postura y de $P_{k,p}$ mediante el filtro KF/EKF global.

- A partir de la prueba de larga duración (30min, figuras 10.3 y 10.4) se comprobó el buen desempeño del método basado en eventos (KF2e4m EBGC 10.3(c)), el cual realiza la estimación de la postura con error acotado a diferencia de la fusión local y de la odometría. Se concluye a partir del buen resultado obtenido en cuanto a la precisión del filtro KF2e4m, así como de su estabilidad y convergencia, que los algoritmos más complejos en cascada (KF6e4m y KF3e4m) tendrán también un desempeño adecuado en la estimación de la postura. Un resumen de las pruebas con el robot Ackerman en interiores puede encontrarse en `/http:///wks./gii./upv./es//cobami//webfm_/send//6`.

- En el caso de las pruebas en exteriores del robot Ackerman (figura 8.6), se comprobó previamente el funcionamiento en interiores mediante la prueba de larga duración de la figura 10.5 obteniendo una estimación adecuada. Con esto se procedió con la prueba en exteriores al agregar un GPS diferencial a la plataforma (figura 10.6) para validar los resultados sin utilizar su medición en la fusión sensorial. A partir de la prueba en exteriores de la figura 10.7 se comprueba el desempeño adecuado del algoritmo al estimar la postura con un error acotado tal y como se observa en las gráficas del error (figura 10.8) y error absoluto (figura 10.9).

- A partir de la prueba en exteriores se concluye que existe un error inherente al sensor global utilizado, que puede ser considerable y que se traslada a la estimación del algoritmo de fusión, tal y como se aprecia en la diferencia de la estimación con el GPS IG-500N y la medición que realiza el GPS diferencial (figura 10.7). De esta forma se debe mitigar este error, ya sea utilizando sensores locales de mayor precisión o bien realizando un mejor ajuste dinámico de los términos correspondientes a la postura del GPS en la matriz R_k del filtro de fusión. Para esto se puede incorporar información adicional al número de satélites N_{sat}, como por ejemplo la covarianza dinámica interna del sensor GPS (si el

sensor suministra esta información), la información meteorológica del entorno de navegación o bien la presencia de edificios aledaños, siendo estos problemas comunes en la precisión del GPS [55, 153]). Incorporando esta información en el valor de R_k, se puede limitar la influencia del error del sensor global en la estimación de la postura del robot. Adicionalmente, se puede incrementar la frecuencia de actualización al disminuir $R_{A,lim}$, lo que disminuiría el error de estimación siempre y cuando la covarianza del sensor global sea baja y se disponga de los recursos computacionales en la plataforma que permitan un número considerable de consultas al sensor global.

- Finalmente, a partir de las pruebas del tiempo de ejecución (simulación de la figura 10.10 con la tabla 10.1, y la implementación en la unidad de control NXT de la figura 10.11 con tabla 10.1 y 10.2) se concluye que los algoritmos en cascada con modelos en cascada (EKF3e4m, algoritmo 11 y KF2e4m, algoritmo 12) utilizan menos recursos computacionales para realizar la estimación de la postura al requerir un menor tiempo del procesador (medido en milisegundos ms), que los algoritmos más complejos como el cascada reducido (EKF6s4m, algoritmo 10) y con asignación de entradas (EKF6s7m, algoritmo 3), pero realizando una estimación con una precisión similar a estos algoritmos (según la prueba de desempeño en las figuras 10.1 y 10.2). Nuevamente se tiene la ventaja de poder implementar los algoritmos en cascada propuestos en la unidad de control NXT, en donde no incumplen con el tiempo de muestreo en ningún caso ($T_s = 50ms$, figura 10.11), ni saturan el uso del ancho de banda, por lo que realizan la fusión sensorial para obtener la postura del robot de forma adecuada, además de permitir la corrección del retardo de comunicación en la medición global mediante el uso de la ecuación (8.1).

11 | Aplicaciones: Localización Cooperativa

En el presente capítulo se utiliza la plataforma de simulación basada en Java y JADE, expuesta en la sección 8.3, para realizar las pruebas del sistema multirobot (definido en el capítulo 7) con los algoritmos de localización cooperativa con corrección basada en el tiempo *TCLA* y basada en eventos *ECLA*.

Se realizan dos pruebas principales para analizar tanto la evolución de la covarianza del error de estimación (del grupo de robots G_R y de cada miembro R_i de éste), como la cantidad de mensajes transmitidos dentro de G_R (enviados y recibidos entre sus integrantes) para la localización cooperativa. Además, se asigna en los algoritmos *TCLA* y *ECLA* un rango de detección D_r igual en cada R_i, así como la fusión cooperativa utilizando únicamente la mejor información relativa recibida (la de menor covarianza $R_{k,r}$ recibida). Esto permite realizar una mejor comparación entre la actualización temporal y por eventos, ya que la principal diferencia entre los algoritmos queda reducida al tipo de actualización y por lo tanto a la forma en la que se realiza el intercambio de mensajes (en cada instante k, o cuando el evento, basado en el área normalizada de los elipsoides 3σ, supera un nivel predeterminado: $R_A > R_{A,lim}$, ecuación (6.7)).

La primera prueba consiste en la simulación de un grupo de cinco integrantes R_i realizando un seguimiento de una trayectoria lineal cíclica, en la que los robots avanzan hacia el punto de destino y una vez alcanzado regresan al punto de partida, lo que permite el estudio del esquema cooperativo cuando las frecuencias de detección de cada vecino son regulares (aproximadamente constantes).

La segunda prueba se realiza al considerar un grupo de diez integrantes que se mueven en el entorno con velocidad constante y ejecutan un algoritmo

de evasión de obstáculos Braitenberg, evitando colisiones con los límites del entorno (caja de forma cuadrada) y con los robots vecinos. Esta configuración en particular permite observar el comportamiento del esquema cooperativo cuando las frecuencias de detección son variantes en el tiempo.

Se presentan a continuación los resultados obtenidos, así como el estudio del compromiso entre la precisión y el uso del ancho de banda del método cooperativo basado en eventos, con el cual se define el rango adecuado de valores para el nivel del evento $R_{A,lim}$, de forma similar a las pruebas realizadas para el robot diferencial (figura 9.11)

11.1 Grupo de cinco robots con seguimiento de una trayectoria lineal cíclica

Esta prueba consiste en la simulación de un grupo G_R de cinco agentes durante 20min, los cuales realizan un seguimiento de una trayectoria lineal cíclica. Uno de los integrantes de G_R se establece con acceso a la medición global de la postura: R_1. Este robot se considera dentro del simulador con un error de estimación de la postura nulo (y por lo tanto su covarianza del error de estimación será también nula). Esto con el fin de observar con más claridad el efecto de la corrección con la información relativa y la diferencia entre utilizar la información de un vecino con baja covarianza (con acceso al sensor global o recientemente actualizado con la información de éste vecino), y el utilizar la información de un vecino de covarianza alta. En la realidad, aunque el error de estimación y la covarianza son distintos de cero, suelen ser valores reducidos si el sensor global tiene una precisión adecuada. Además, en el esquema cooperativo al menos un robot debe poseer acceso a la información global, de forma que se proporciona convergencia global al grupo [145] (se evita el crecimiento no acotado del error de estimación de la postura).

El seguimiento de la trayectoria lineal cíclica se muestra para el primer recorrido (ida al punto de destino y vuelta al punto de partida indicado con "*") en la figura 11.1 para todos los cinco integrantes de R_i (tres inician desde la izquierda y dos desde la derecha). Al mantener el seguimiento de esta trayectoria, los vecinos se detectarán entre sí aproximadamente a la

mitad del recorrido. Además, como cada robot avanza con una velocidad lineal constante que son similares entre cada R_i, los vecinos se detectarán de forma regular, con frecuencias de detección similares aunque con leves variaciones. Esta característica permite, por una parte, observar la evolución de la covarianza del error cuando el método cooperativo dispone de la información relativa de forma regular y, por otra parte, posibilita la variación en el orden de detección de los vecinos, con lo que se realizará el método cooperativo utilizando tanto información de alta precisión (vecino R_1 o un vecino actualizado recientemente a partir de éste) como de baja precisión (vecinos restantes sin actualización reciente). Esto resulta relevante en el esquema cooperativo *ECLA* cuando, por ejemplo, se ha alcanzado el límite del evento $R_{A,lim}$ en un R_i, pero éste no dispone de vecinos dentro de D_r ($N_D = 0$) para realizar la actualización con información relativa. En esta condición, el primer vecino que entra en D_r es detectado por R_i, el cual obtiene la medición relativa M_r y la utiliza junto con la información i_R (recibida del vecino) para corregir la postura. De esta forma, si el primer vecino detectado es R_1 la covarianza del error de estimación de la postura $P_{k,p}$ resultante será baja, lo que sucede también en caso de detectar otro R_i con covarianza del error baja en su postura. Por otra parte si el vecino detectado tiene error alto en la estimación de su postura, se producirá una disminución en la $P_{k,p}$ de R_i, pero esta reducción será de menor magnitud, por lo que $R_{A,lim}$ será alcanzado antes que en el caso de detección con R_1. Este tipo de variabilidad en la detección es conveniente en la simulación ya que permite observar distintos casos de detección, al no existir una secuencia predefinida.

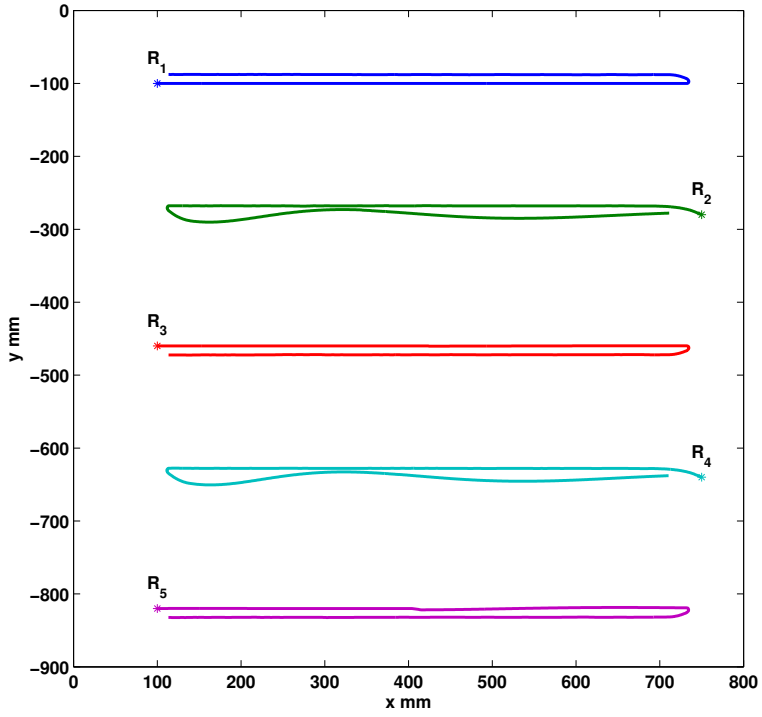


Figura 11.1: Trayectorias lineales cíclicas del grupo G_R con cinco integrantes, prueba de 20min

11.1.1 Evolución de la covarianza, comparativa entre *TCLA* y *ECLA*

El comportamiento del esquema de localización cooperativa se observa con claridad en la evolución de la covarianza del error de estimación $P_{k,p}$, obtenida para cada robot R_i, o bien como un promedio $\forall R_i \in G_R$, en lo que se puede observar el efecto de la corrección de la postura con la información relativa, además de la precisión general del algoritmo de fusión cooperativo. Como mínimo, $P_{k,p}$ debe ser acotada, ya que si ésta crece indefinidamente indicaría que el robot se ha perdido (no conoce su postura con precisión $\rightarrow$ con $P_{k,p}$ acotada). En el caso de la localización cooperativa, $P_{k,p}$ será acotada cuando R_i tiene acceso a información sensorial con baja covarianza (sensor global, o M_R con alta precisión en conjunto con i_R de un robot recientemente actualizado), y $P_{k,p}$ tenderá a valores cada vez menores conforme esta información se utilice de forma cada vez más frecuente (valores bajos del evento o utilizando la información en cada instante k).

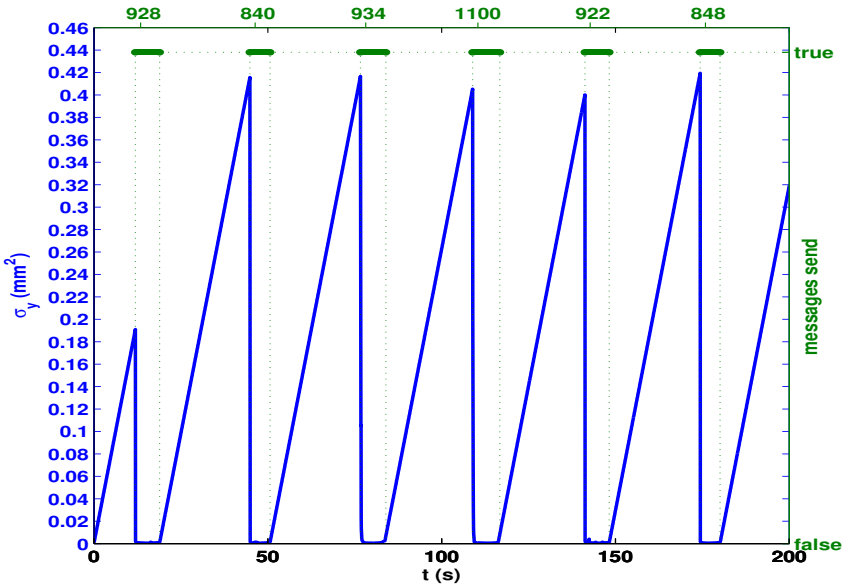
Se obtiene la evolución de $P_{k,p}$ para la prueba de 20min utilizando los algoritmos de localización cooperativa *TCLA* y *ECLA* con los niveles de evento $R_{A,lim} = \{3{,}73, 5{,}6, 11{,}2\}$, tal y como se muestra en las figuras 11.2 y 11.3, en donde se muestra la evolución de $P_{k,y}$ para el robot R_2 (figuras 11.1 y 8.7). En cada gráfica de $P_{k,y}$ se indica además la cantidad de mensajes intercambiados por el robot para realizar la localización cooperativa, en un eje adicional en la parte superior de las figuras 11.2 y 11.3. Se puede apreciar en el caso del *ECLA*, la corrección con información precisa (proveniente de R_1) cuando $P_{k,y}$ disminuye hasta un valor cercano a cero, y la corrección con información menos precisa (proveniente de R_3) cuando $P_{k,y}$ disminuye levemente. En este caso en particular se volverá a sobrepasar $R_{A,lim}$ rápidamente, con lo que se realiza una nueva corrección relativa, pero en este caso R_2 puede tener en D_r tanto a R_1 como a R_3, por lo que escoge la información con la menor covarianza en la postura (en este caso R_1) con lo que $P_{k,y}$ disminuye a un valor cercano a cero.

Se observa de ambas figuras 11.2 y 11.3, que el error de estimación es acotado para ambos métodos de localización cooperativa (*TCLA* y *ECLA*), ya que la covarianza del error de estimación no crece indefinidamente debido al

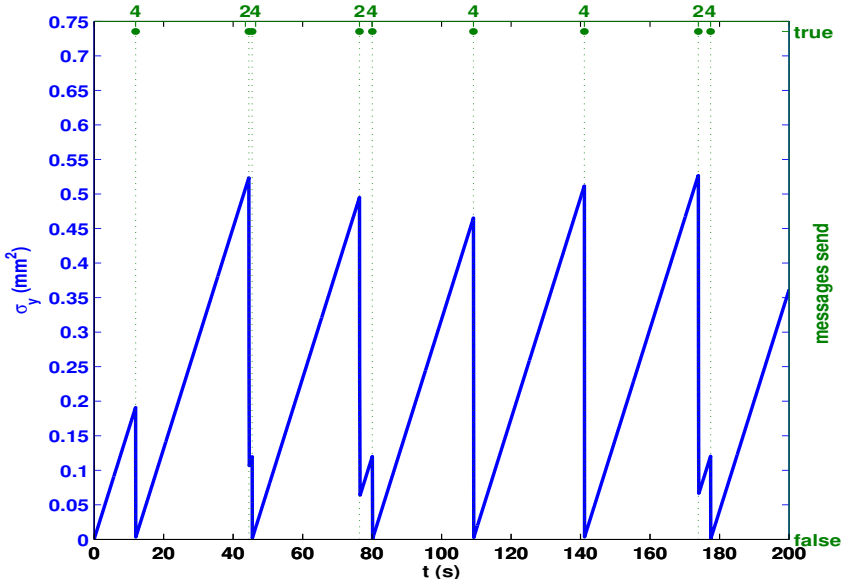
método cooperativo y a la utilización de sensores con precisión adecuada (incluyendo las mediciones relativas y la información recibida de los robots vecinos). Además, la evolución de $P_{k,y}$ es similar para ambos métodos al ser las condiciones de simulación de éstos similares entre sí. Sin embargo, la principal diferencia entre ambas estrategias radica en la cantidad de mensajes intercambiados, requiriendo el método basado en eventos entre 2 y 4 mensajes por actualización (corrección de la postura mediante la información relativa), contrario a los 840 mensajes que utiliza como mínimo el *TCLA* (figura 11.2).

Esta diferencia muestra uno de los aportes fundamentales del método cooperativo basado en eventos propuesto, en cuanto el *ECLA* presenta una evolución similar en la estimación de la postura y límites de la covarianza del error de estimación que el método *TCLA*, pero utilizando una fracción del ancho de banda para realizar la actualización cooperativa (por lo que requiere menos recursos computacionales). La selección del límite $R_{A,lim}$ ajusta el ancho de banda requerido (permitiendo menos actualizaciones $R_{A,lim} = 11{,}2$ o más $R_{A,lim} = 3{,}73$) por lo que también modificará el promedio y valor máximo de $P_{k,p}$ ($P_{k,y}$ en las figuras 11.2 y 11.3). Por ejemplo, un pequeño incremento de alrededor de $0{,}1mm^2$ en el valor máximo de $P_{k,y}$ se obtiene al utilizar *ECLA* con $R_{A,lim} = 3{,}73$ comparando con el resultado de *TCLA*, pero consiguiendo una reducción elevada en el ancho de banda utilizado

Conviene destacar que al ser la detección entre los miembros del grupo dinámica, la mejora del esquema cooperativo basado en eventos *ECLA* al reducir $R_{A,lim}$ es limitada. De esta forma, el error de estimación crecerá hasta que se alcance el límite del evento $R_A > R_{A,lim}$ y se detecte un vecino dentro de D_r ($N_D \geq 1$). Por esta razón, la reducción de $R_{A,lim}$ disminuirá $P_{k,p}$ siempre y cuando se detecten vecinos en D_r, y estos posean una baja covarianza del error de estimación (transmitida en i_T para el método cooperativo).

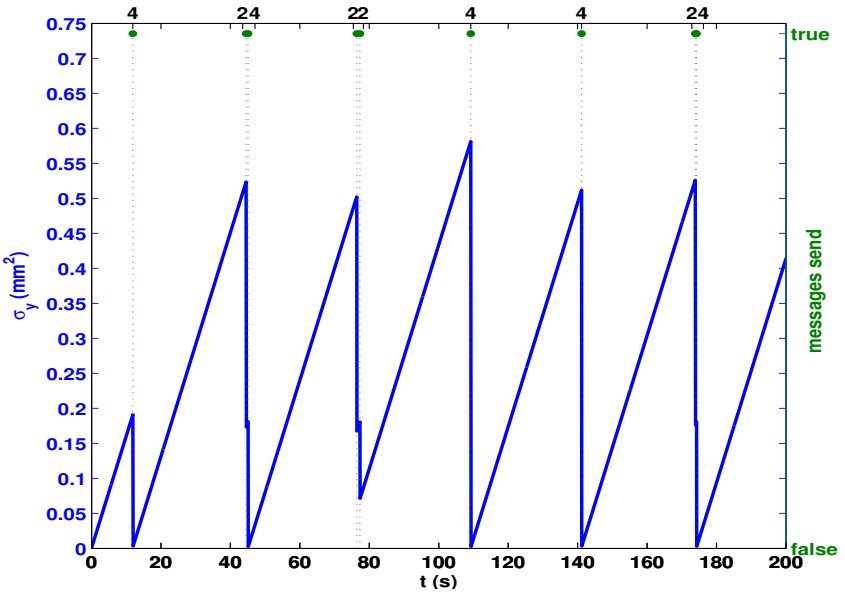


(a) TCLA

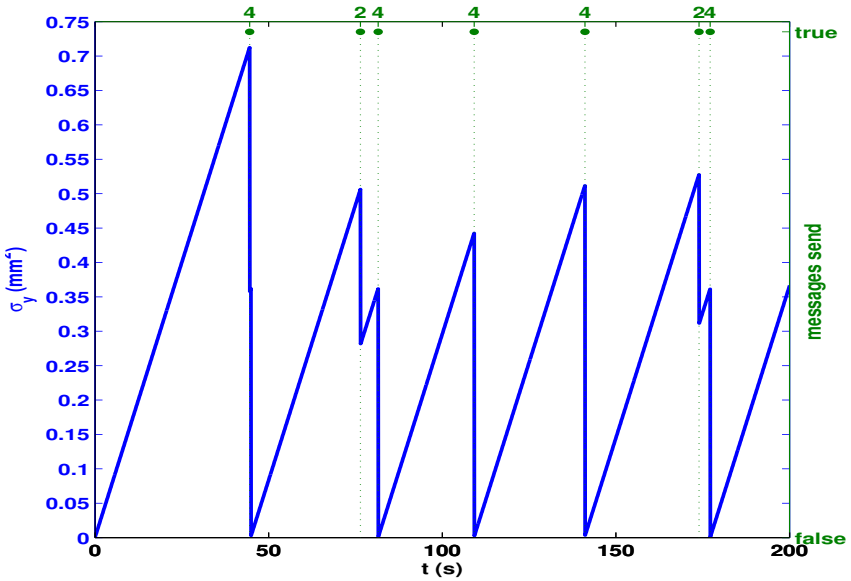


(b) ECLA, $R_{A,lim} = 3{,}73$

Figura 11.2: Evolución de la covarianza de R_2 y mensajes intercambiados, G_R con cinco robots, *TCLA* y ECLA con $R_{A,lim} = 3{,}73$, $A_{Ri} = 0{,}9136$, primeros 3min de la prueba de 20min



(a) ECLA, $R_{A,lim} = 5{,}6$



(b) ECLA, $R_{A,lim} = 11{,}2$

Figura 11.3: Evolución de la covarianza de R_2 y mensajes intercambiados, G_R con cinco robots, ECLA con $R_{A,lim} = \{5{,}6, 11{,}2\}$, primeros 3min de la prueba de 20min

El efecto de $R_{A,lim}$ en los límites de la covarianza del error de estimación $P_{k,y}$ (y en general de $P_{k,p}$) del método cooperativo basado en eventos, se puede estudiar de una mejor forma al obtener la media de la covarianza del grupo G_R: $\bar{P}_{k,y}$, la cual se calcula al obtener el promedio, en cada instante k de muestreo, de las cinco $P_{k,y}$ de G_R (5 integrantes, una $P_{k,p}$ por cada R_i), tal y como se muestra en la figura 11.4. Además, es conveniente observar la evolución temporal de la media $\bar{P}_{k,y}$, para lo cual se calcula la media acumulativa en el tiempo: $c\bar{P}_{k,y}$ (covarianza promedio acumulativa), al obtener el promedio utilizando todos los valores disponibles de $\bar{P}_{k,y}$ desde el instante inicial de la prueba hasta el instante actual k, tal y como se muestra en la figura 11.5. Por ejemplo, el valor de $c\bar{P}_{k,y}$ en $k = 1s$ es la media de todos los valores de $\bar{P}_{k,y}$ en el intervalo $k = [0, 1s]$.

A partir de las figuras 11.4 y 11.5 se puede observar con claridad la influencia de $R_{A,lim}$ en la precisión del método cooperativo basado en eventos. Al disminuir el valor de $R_{A,lim}$ en *ECLA*, los límites de $P_{k,y}$ también decrecen, tendiendo al valor de los límites de la covarianza obtenidos con el método *TCLA* (se incrementa la precisión del método conforme decrecen los valores máximos de $P_{k,y}$). Sin embargo, este incremento en la precisión tiene el costo asociado de incrementar el número de mensajes utilizados para realizar la localización cooperativa (se requerirán por lo tanto más recursos computacionales). Con este resultado, se puede realizar el ajuste del valor de $R_{A,lim}$ de acuerdo a la cantidad de recursos disponibles en la plataforma y según la precisión deseada en la estimación de la postura. Por otra parte, la evolución temporal de $\bar{P}_{k,y}$ representada mediante $c\bar{P}_{k,y}$ (figura 11.5) muestra el comportamiento estable de la estimación, en cuanto el error de estimación es acotado y su covarianza promedio (de G_R) tiende a un valor constante (no crece indefinidamente). En el caso de la localización cooperativa, se alcanza este valor constante debido a la presencia de una fuente de información global con precisión adecuada (en el caso de esta prueba es el robot R_1), y al intercambio de información de los algoritmos *TCLA* y *ECLA* que distribuyen la información global a todos los robots del grupo de forma directa (si se encuentran dentro del D_r) o de forma indirecta (mediante los robots recientemente actualizados por el algoritmo cooperativo).

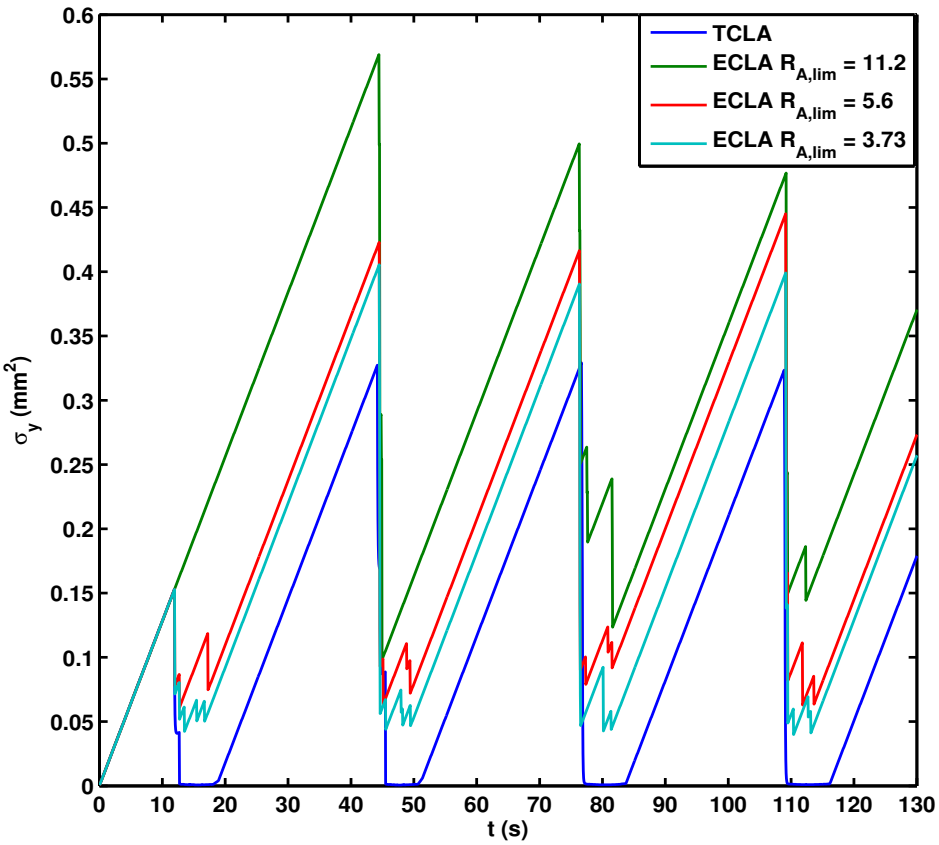


Figura 11.4: Covarianza Promedio $\bar{P}_{k,y}$, evolución para el G_R con cinco robots, prueba de 20min

Finalmente, se observa de la figura 11.5 el comportamiento similar de las estrategias *TCLA* y *ECLA*, siendo ambas acotadas en cuanto al error y covarianza del error de estimación, pero en el caso de *ECLA* se puede realizar el ajuste de la precisión del método y del ancho de banda utilizado para obtener la estimación de la postura.

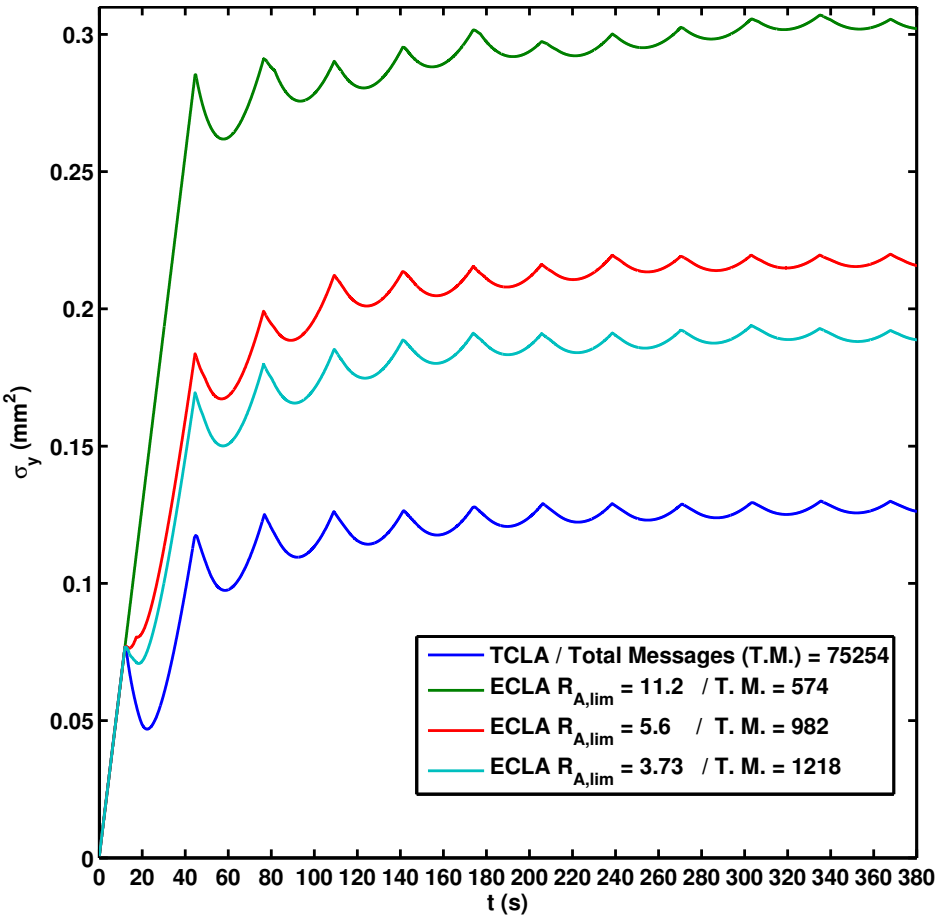


Figura 11.5: Covarianza promedio acumulativa $c\bar{P}_{k,y}$, para el G_R con cinco robots, prueba de 20min

Se presenta un resumen de la prueba de cinco robots (20min) utilizando el *TCLA* y el *ECLA* con el límite del evento definido en $R_{A,lim} = \{3{,}73, 5{,}6, 11{,}2\}$ en la tabla 11.1. En ésta se observan la cantidad de mensajes intercambiados (envío y recepción) y los valores promedio $\bar{P}_{k,x}$ y $\bar{A}_{ellip}$ para la duración total de la prueba de cada R_i dentro del grupo (promedio temporal). Se muestra además el promedio del grupo G_R (sin considerar los valores de R_1) y el número total de mensajes intercambiados en G_R. Se puede apreciar la evolución del promedio $\bar{P}_{k,x}$ y $\bar{A}_{ellip}$ los cuales tienden a decrecer conforme se disminuye el valor de $R_{A,lim}$ por las razones anteriormente descritas (para el caso de $\bar{P}_{k,y}$ en las figuras 11.4 y 11.5). Se puede observar nuevamente el compromiso entre la precisión del método y el ancho de banda (costo) requerido para obtenerla, ajustable en *ECLA* según la selección de $R_{A,lim}$.

En robots de recursos limitados, el $R_{A,lim}$ puede ser modificado en concordancia con el ancho de banda disponible en la plataforma, para realizar un menor número de interacciones con los vecinos dentro de D_r (y por lo tanto un menor número de ejecuciones de la corrección del filtro de fusión, reduciendo el uso de recursos computacionales), pero esto tiene el costo asociado del incremento en la covarianza del error de estimación (y del error en la postura).

Por ejemplo en la tabla 11.1, los valores de $\bar{P}_{k,x}$ y $\bar{A}_{ellip}$ se incrementan cerca de un 50 % utilizando *ECLA* con $R_{A,lim} = 3{,}73$ respecto a los valores obtenidos con *TCLA*, pero utilizando únicamente un 1,62 % de los mensajes intercambiados en la actualización basada en el tiempo. El incremento en $\bar{P}_{k,x}$ y $\bar{A}_{ellip}$ es cercano al 70 % con $R_{A,lim} = 5{,}6$ utilizando el 1,3 % de los mensajes intercambiados y finalmente, el duplica el incremento hasta el 147 % al utilizar $R_{A,lim} = 11{,}2$, lo que resulta en una utilización de mensajes de únicamente el 0,76 % de los utilizados en la actualización temporal. Con estos resultados se puede establecer el límite del evento en $R_{A,lim} = 3{,}73$ como la mejor solución de compromiso en términos de costo-beneficio para el grupo de cinco robots con movimiento lineal cíclico. Un análisis similar se puede llevar a cabo en otros escenarios con el fin de obtener el mejor $R_{A,lim}$ para el grupo de robots.

Tabla 11.1: Resumen de resultados, prueba con el G_R de cinco integrantes, TCLA y ECLA con $R_{A,lim} = \{3{,}73, 5{,}6, 11{,}2\}$

R_i	Algoritmo *TCLA*			Algoritmo *ECLA*, $R_{A,lim} = 3{,}73$		
	$\bar{P}_{k,x}$	$\bar{A}_{ellip}$	Mensajes Intercambiados	$\bar{P}_{k,x}$	$\bar{A}_{ellip}$	Mensajes Intercambiados
1	0	0	18848	0	0	76
2	0,1622	4,5818	9546	0,2398	6,7536	202
3	0,1589	4,4883	18768	0,2419	6,8102	370
4	0,1589	4,4911	18762	0,2483	6,9951	346
5	0,1705	4,8139	9330	0,2457	6,9190	224
G_R	0,1627	4,5938	75254	0,2439	6,8695	1218

R_i	Algoritmo *ECLA*, $R_{A,lim} = 5{,}6$			Algoritmo *ECLA*, $R_{A,lim} = 11{,}2$		
	$\bar{P}_{k,x}$	$\bar{A}_{ellip}$	Mensajes Intercambiados	$\bar{P}_{k,x}$	$\bar{A}_{ellip}$	Mensajes Intercambiados
1	0	0	68	0	0	50
2	0,2534	7,1396	166	0,3106	8,7603	110
3	0,2703	7,6138	292	0,4346	12,2663	118
4	0,2911	8,2092	286	0,4001	11,2925	202
5	0,2920	8,2343	170	0,4634	13,0857	94
G_R	0,2767	7,7992	982	0,4022	11,3512	574

11.2 Grupo de diez robots con evitación de obstáculos Braitenberg

Esta prueba consiste en la simulación de un grupo G_R de diez agentes durante 46min, los cuales se mueven con una velocidad constante de avance en el entorno de navegación, ejecutando un algoritmo Braitenberg que realice la evitación de colisiones entre los robots y los límites del entorno (las paredes que limitan la superficie cuadrada en la que los robots se mueven, figura 8.8). Se consideran dos robots con acceso a mediciones globales ($\{R_1, R_6\}$ con $P_{k,p} = 0$) para proveer convergencia global al grupo [145] (evitando el crecimiento no acotado del error en la postura). A medida que los robots avanzan e interaccionan entre sí, enviarán colisiones y ejecutarán la localización cooperativa *ECLA* cuando sea necesario (si $R_A > R_{A,lim}$)

En este caso, como los agentes se mueven dinámicamente por todo el entorno los vecinos se detectarán entre sí sin un orden predeterminado, por lo que las frecuencias de detección son variantes en el tiempo. Esto permite observar la evolución de la covarianza del error al recibir en distintos tiempos la información relativa. Esta configuración contempla casos en los que un robot ha alcanzado $R_{A,lim}$ pero al no detectar a un vecino en D_r de forma regular, puede darse un crecimiento considerable en $P_{k,p}$ hasta que se logre detectar un vecino. De la misma forma pueden darse casos en que se realice la corrección con la información relativa rápidamente, ya que al alcanzar $R_{A,lim}$ se pueden disponer de uno o varios vecinos en D_r ($N_D \geq 1$). Además, la variabilidad en la detección entre los vecinos permite realizar la corrección con información relativa de alta precisión (vecinos $\{R_1, R_6\}$ o recientemente actualizados) así como con información menos precisa (vecinos restantes sin actualización reciente).

11.2.1 Evolución de la covarianza del error de estimación

El movimiento de los robots para el primer minuto de la simulación con la evitación de obstáculos Braitenberg, así como la evolución de la covarianza $P_{k,p}$ para los primeros 18min se muestra en la figura 11.6. Nuevamente, se observa un desempeño adecuado del algoritmo cooperativo por eventos pro-

puesto ($ECLA$) en cuanto el error de estimación es acotado para todos los robots de G_R, ya que la covarianza del error de estimación $P_{k,p}$ no crece indefinidamente.

Nuevamente, de acuerdo a N_D y a la i_R, la actualización de la postura producirá un mayor decremento en $P_{k,p}$ si dentro de D_r se incluye un miembro de G_{RA} (R_1 o R_6 en esta prueba), o un R_i recientemente actualizado con la información de los integrantes de G_{RA}, tal y como se observa en detalle para la covarianza $P_{k,y}$ del robot R_2 en la figura 11.6. En esta prueba se muestra una evolución similar para los restantes R_i de recursos limitados (en G_{RL}, sin acceso a la medición global). Se observan algunos valores altos en $P_{k,p}$ los cuales ocurren cuando se alcanza $R_{A,lim}$ pero $N_D = 0$, debido a que $P_{k,p}$ crecerá hasta que se realice la localización cooperativa (cuando se detecte un vecino, con lo que $N_D > 0$ y se puedan obtener M_r y recibir i_R). Los valores máximos en $P_{k,p}$ dependerán de la configuración del grupo G_R (misión, movimientos a realizar, etc.), del número de $R_i \in G_{RA}$ con acceso a la información global, así como de la frecuencia con la que éstos se encuentran dentro del rango de detección D_r de los $R_i \in G_{RL}$, siendo aspectos que deben ser tomados en consideración durante la implementación y diseño del grupo multirobot cooperativo.

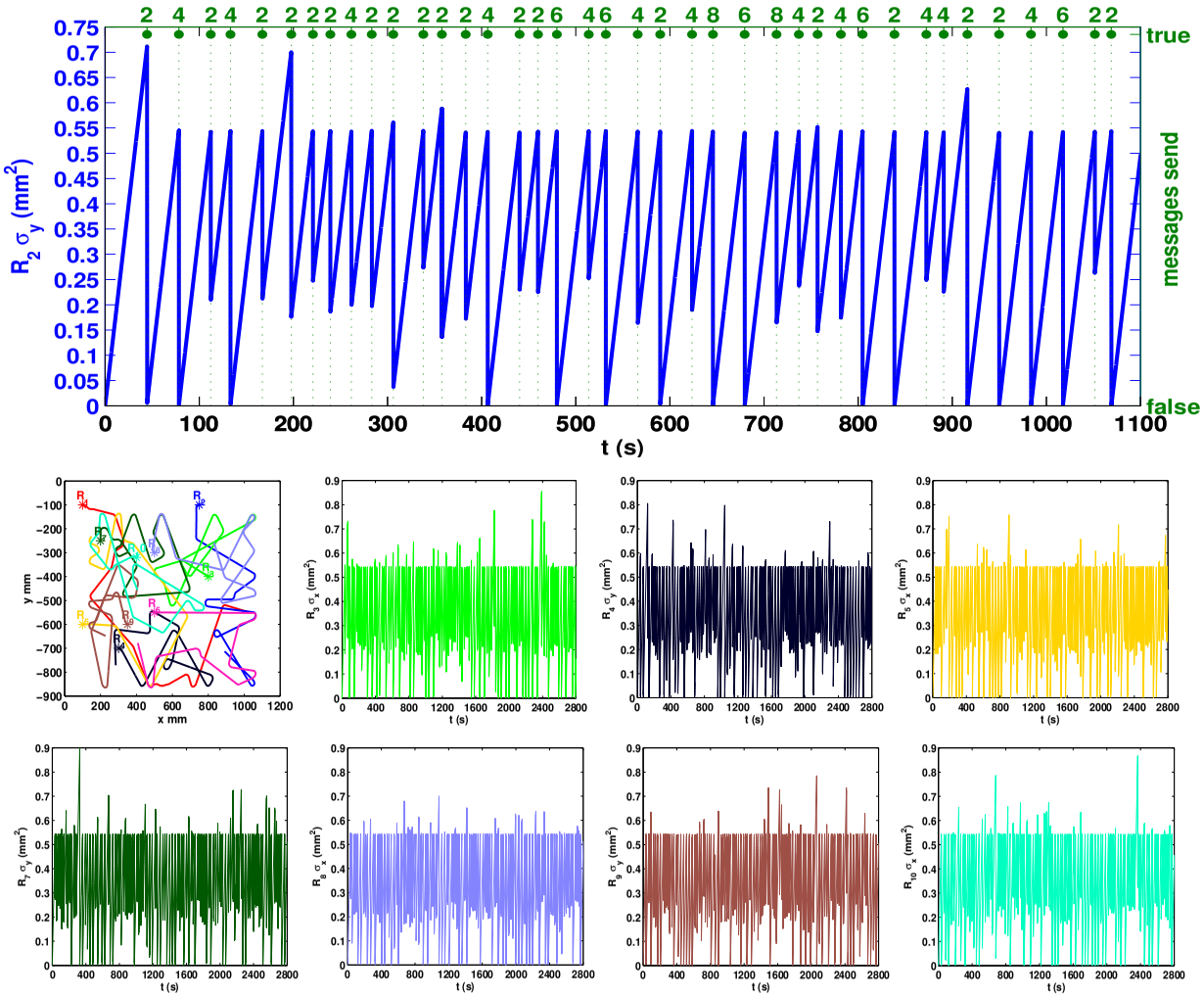


Figura 11.6: *ECLA* con $R_{A,lim} = 16{,}8$, G_R con $N = 10$, simulación de 46min: movimiento de los robots con la evitación de obstáculos Braitenberg (primer minuto), evolución de la covarianza $P_{k,p}$ y mensajes intercambiados en R_2 (primeros 18min)

11.2.2 Compromiso entre la precisión y uso del ancho de banda

Para observar el efecto de $R_{A,lim}$ en el algoritmo cooperativo *ECLA*, se repite la prueba de simulación con la evitación de obstáculos Braitenberg para distintos niveles de $R_{A,lim}$. A partir de cada prueba se obtienen la covarianza promedio del grupo $\bar{P}_{k,y}$ (promedio para los integrantes de G_R, removiendo R_1 y R_6), y la covarianza promedio acumulativa $c\bar{P}_{k,y}$ (evolución temporal del promedio $\bar{P}_{k,y}$) tal y como se muestra en las figuras 11.7 y 11.8 respectivamente. Como es de esperarse, el esquema de localización cooperativo *ECLA* muestra un comportamiento estable para todos los valores de $R_{A,lim}$ utilizados, en cuanto el error de estimación es acotado y la covarianza del error de estimación no crece indefinidamente (su promedio alcanza un valor estable según la figura 11.8). Se observa además que se obtiene una mejor precisión del método (mayor uso del ancho de banda) cuando se utilizan valores de $R_{A,lim}$ bajos, en el caso contrario (incremento en $R_{A,lim}$) se disminuye la precisión pero se realizaría un menor intercambio de mensajes entre integrantes del grupo. Con esto se obtiene un resultado similar a las pruebas para el grupo de cinco integrantes, figuras 11.4 y 11.5.

El compromiso entre la precisión del método (promedio de $P_{k,y}$ de todos los robots para la duración total de la prueba de 46min: $T\bar{P}_{k,y}$) y el uso del ancho de banda de comunicación (total de mensajes intercambiados por los robots en G_R en la duración total: *T.M.*) ante la variación del límite del evento $R_{A,lim}$, realizado para la prueba con la evitación de obstáculos Braitenberg se muestra en la figura 11.9 (similar a la del robot diferencial, figura 9.11).

Se puede apreciar de esta figura la influencia de $R_{A,lim}$ en la precisión y el número de mensajes intercambiados para realizar la corrección cooperativa con la información relativa. Al disminuir $R_{A,lim}$ también disminuirán los márgenes de error (covarianza $T\bar{P}_{k,y}$) del método *ECLA* hasta un límite, definido por el máximo número de consultas que la plataforma puede realizar (según los recursos disponibles). Por el contrario, cuando se incrementa $R_{A,lim}$, se disminuye la precisión del método (se incrementa $T\bar{P}_{k,y}$) pero con la ventaja de reducir la cantidad de mensajes intercambiados (*T.M.*) y por lo tanto los recursos computacionales requeridos para la localización del grupo de robots.

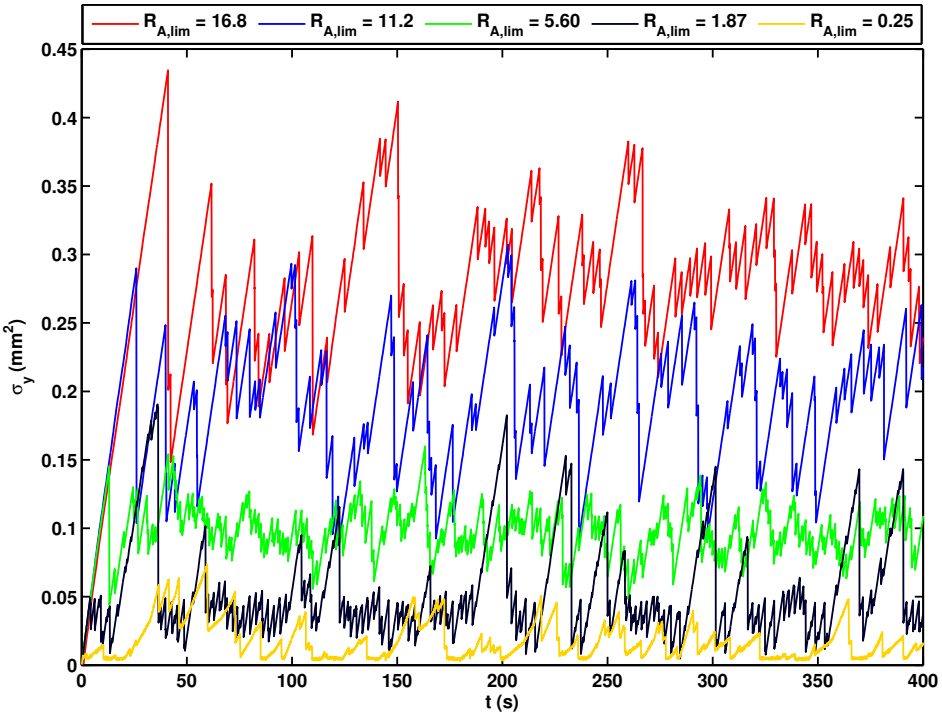


Figura 11.7: Covarianza Promedio $\bar{P}_{k,y}$, evolución para el G_R con diez robots, prueba de 46min

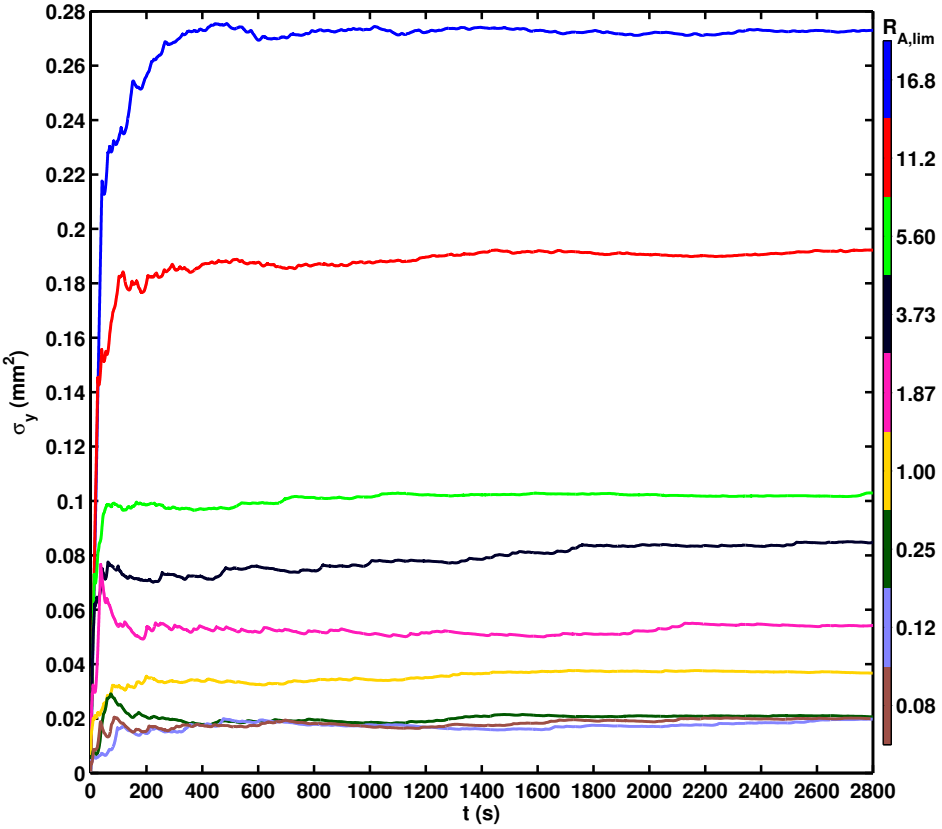


Figura 11.8: Covarianza promedio acumulativa $c\bar{P}_{k,y}$, para el G_R con diez robots, prueba de 46min

A partir de la dinámica de la figura 11.9, se puede escoger un rango adecuado para $R_{A,lim}$, de forma que se obtenga una solución de compromiso entre la precisión requerida en la localización cooperativa del grupo de robots y la cantidad de recursos computacionales y ancho de banda disponibles en las plataformas (en cada R_i del grupo).

Este estudio puede realizarse de acuerdo con las características del grupo y de las plataformas integrantes (de cada R_i), de forma que se obtenga un rango adecuado de compromiso para $R_{A,lim}$ (a usar en la totalidad de G_R o uno para cada R_i). A partir de la figura 11.9 se puede definir por ejemplo $0{,}2 \leq R_{A,lim} \leq 7$. El valor utilizado para $R_{A,lim}$ puede ser definido a voluntad durante la etapa de diseño del sistema para predefinir el máximo intercambio de mensajes deseado, o bien podría modificarse dinámicamente durante la ejecución del programa en cada robot. Por ejemplo, se puede ajustar el valor de $R_{A,lim}$ según los recursos (en tiempo de cómputo y ancho de banda) disponibles en tiempo real en la plataforma, disminuyendo su valor en caso de que todo el ancho de banda de la plataforma esté disponible, o bien incrementar $R_{A,lim}$ si se requieren recursos computacionales o ancho de banda para otras tareas distintas a la localización cooperativa.

Siguiendo con el ejemplo de la figura 11.9, el rango $0{,}2 \leq R_{A,lim} \leq 7$ produce una variación de $0{,}0204 \leq T\bar{P}_{k,y} \leq 0{,}1262$ en la precisión y $158265 \geq T.M. \geq 4904$ en el ancho de banda. De esta forma, el rango de $R_{A,lim}$ representa una reducción en la $T\bar{P}_{k,y}$ con $R_{A,lim} = 0{,}2$ cercana al $16{,}2\,\%$ del valor de $T\bar{P}_{k,y}$ con $R_{A,lim} = 7$, entretanto el esquema cooperativo con $R_{A,lim} = 7$ utiliza únicamente un $3{,}1\,\%$ de los mensajes requeridos cuando $R_{A,lim} = 0{,}2$. Un video con el resumen de las pruebas presentadas para el esquema de localización cooperativa puede encontrarse en `/http:///idecona./ai2./upv./es//multirobot`.

Concluidas todas las pruebas experimentales de los diversos algoritmos propuestos y realizado el análisis de los resultados obtenidos, se procede a exponer las principales conclusiones de la presente tesis según se describe en el siguiente capítulo.

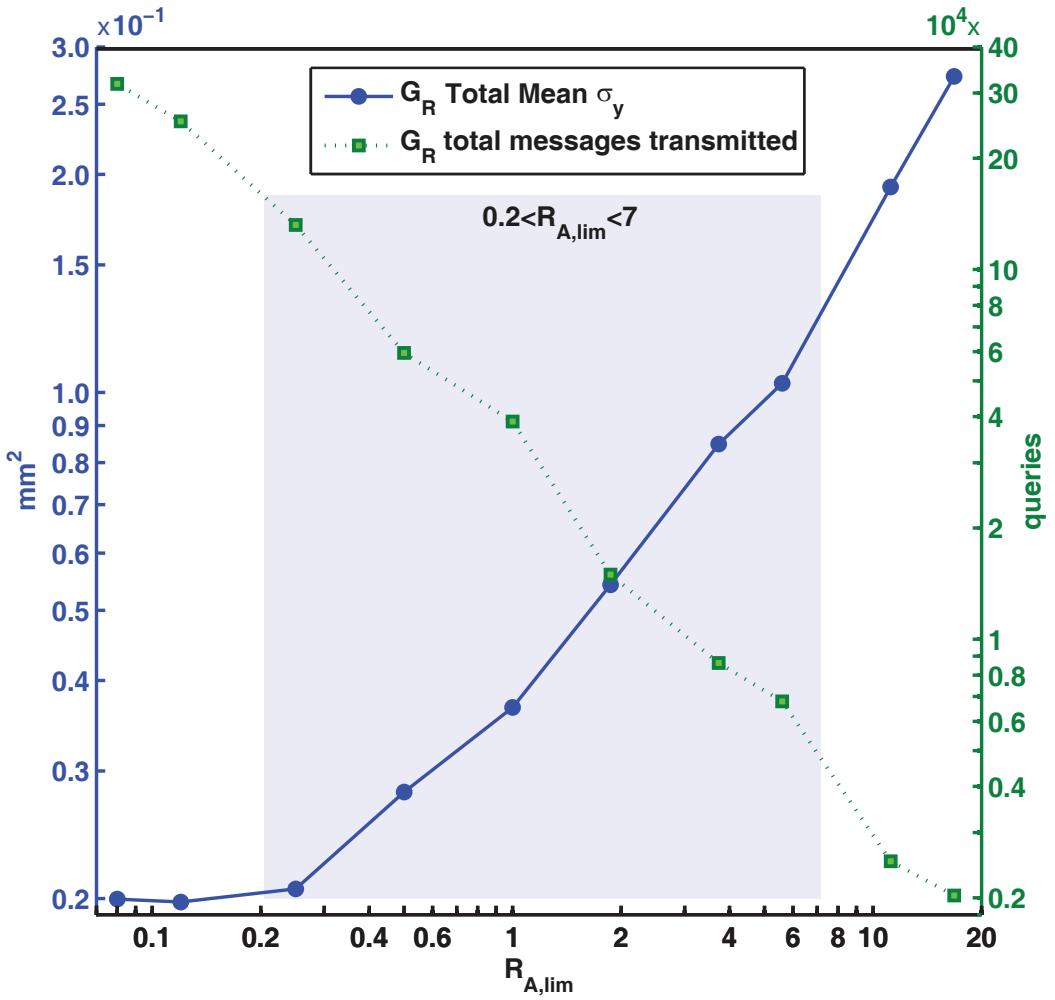


Figura 11.9: Compromiso entre el desempeño (covarianza total $T\bar{P}_{k,y}$) y el ancho de banda (total de mensajes T.M.) definido por el límite del evento $R_{A,lim}$ para el G_R con diez robots, prueba de 46min

11.3 Conclusiones del Capítulo

En el presente capítulo se utilizó la plataforma de simulación basada en Java y JADE (sección 8.3) para realizar las pruebas del sistema multirobot (expuesto en el capítulo 7) con los algoritmos de localización cooperativa con corrección basada en el tiempo *TCLA* (algoritmo 13) y basada en eventos *ECLA* (algoritmo 14). Se configuró la simulación para asignar el mismo rango de detección D_R a todos los integrantes R_i de los grupos, además de utilizar únicamente la mejor medición relativa (mínima $R_{k,r}$ recibida) para la localización cooperativa y el evento basado en el área normalizada de los elipsoides 3σ ($R_A > R_{A,lim}$, ecuación (6.7)).

En la primera prueba se realizó la simulación de un grupo G_R de cinco agentes R_i realizando un seguimiento de una trayectoria lineal cíclica (frecuencia de detección regular entre vecinos) durante 20min, en donde el robot R_1 se establece con acceso a la medición global de la postura con el fin de proporcionar convergencia global al grupo [145]:

- Se comprobó la precisión del método en cuanto a la evolución de la covarianza del error de estimación $P_{k,p}$ para ambos algoritmos *TCLA* y *ECLA* ($R_{A,lim} = \{3{,}73, 5{,}6, 11{,}2\}$) según se expuso en las figuras 11.2 y 11.3 y en la tabla resumen 11.1, en las cuales se indicaron además la cantidad de mensajes que se intercambiaron para realizar la localización cooperativa. A partir de estos resultados se concluye que el error de estimación es acotado para ambos métodos ya que $P_{k,y}$ no crece indefinidamente debido al método cooperativo y a la utilización de sensores con precisión adecuada (incluyendo las mediciones relativas y la información recibida de los robots vecinos). Además, el *ECLA* presenta una evolución de $P_{k,y}$ similar al *TCLA*, pero requiriendo un número muy reducido de mensajes respecto a esta estrategia basada en el tiempo. En esta prueba se observa uno de los principales aportes del método cooperativo basado en eventos propuesto, en cuanto este *ECLA* presenta una evolución similar en la estimación de la postura y límites de $P_{k,p}$ que el método *TCLA*, pero utilizando una fracción del ancho de banda para realizar la actualización cooperativa con lo que se reducen los recursos computacionales utilizados de la plataforma (me-

nor número de ejecuciones de la corrección del filtro con información relativa).

- A partir de la covarianza del grupo $\bar{P}_{k,y}$ (promedio en cada instante k de $P_{k,y}$ de los $R_i \in G_R$) y la covarianza promedio acumulativa $c\bar{P}_{k,y}$ (evolución de $\bar{P}_{k,y}$ en el tiempo) mostradas en las figuras 11.4 y 11.5, se observa con claridad la influencia de $R_{A,lim}$ en la precisión del método cooperativo basado en eventos. A partir de estas figuras se puede concluir que el valor asignado a $R_{A,lim}$ permite incrementar la precisión del método (valores bajos en $R_{A,lim}$) al incrementar la cantidad de mensajes utilizados para el método cooperativo, o bien disminuir el costo computacional y número de mensajes al reducir la precisión de la estimación ($R_{A,lim}$ alto). Este ajuste se realiza según la cantidad de recursos disponibles en cada R_i, o bien según la precisión deseada en la estimación de la postura.

- Se concluye además que la mejora en precisión al disminuir $R_{A,lim}$ queda limitada a la detección de un vecino dentro de D_r ($N_D \geq 1$) con el que se pueda realizar el método cooperativo para reducir $P_{k,p}$, siempre y cuando la medición relativa y la postura recibida tengan una precisión adecuada.

- Finalmente, de la evolución de $c\bar{P}_{k,y}$ (figura 11.5) se comprueba nuevamente el comportamiento estable de la estimación del algoritmo *ECLA*, con un error de estimación acotado al evolucionar la covarianza promedio de G_R a un valor constante (no crece indefinidamente). Esto debido a la presencia de un R_i con acceso a información global con precisión adecuada y al intercambio de información cooperativo que distribuye esta información a todos los integrantes del grupo, de forma directa (si se encuentran dentro del D_r) o de forma indirecta (mediante los robots en D_r recientemente actualizados por el algoritmo cooperativo).

En la segunda prueba se consideró un G_R de diez agentes R_i que se mueven en el entorno (figura 8.8) con velocidad constante y ejecutan un algoritmo de evasión de obstáculos Braitenberg durante 46min, evitando colisiones con los límites del entorno (caja de forma cuadrada) y con los robots vecinos (frecuencia de detección entre vecinos variante en el tiempo). Se establecen los

robots R_1 y R_6 con acceso a la información global (para proveer convergencia global al grupo [145]):

- Se obtuvo la evolución de $P_{k,p}$ para todos los integrantes del grupo en la figura 11.6, de la cual se concluye que el método basado en eventos *ECLA* realiza una estimación con error de estimación acotado en donde $P_{k,p}$ no crece indefinidamente, esto a pesar de que la frecuencia de detección de los vecinos es variante en el tiempo. Nuevamente, la disminución en $P_{k,p}$ depende de la precisión de la postura del vecino detectado dentro de D_r, y el valor máximo de $P_{k,p}$ dependerá de la definición de $R_{A,lim}$, de la configuración del grupo (misión, movimientos a realizar, etc.) y de frecuencia con la que se detecte un vecino en D_r, siendo éstos aspectos a considerar durante la implementación y diseño del grupo multirobot cooperativo.

- La evolución de $\bar{P}_{k,y}$ (promedio para los integrantes de G_R, removiendo R_1 y R_6) y la covarianza promedio acumulativa $c\bar{P}_{k,y}$ (evolución temporal del promedio $\bar{P}_{k,y}$) se presentaron en las figuras 11.7 y 11.8 para distintos valores de $R_{A,lim}$ en el algoritmo *ECLA*. Se concluye a partir de estas figuras que el método basado en eventos tiene un comportamiento estable en todos los casos, con un error de estimación acotado y con una $P_{k,y}$ que no crece indefinidamente, ya que su promedio alcanza un valor estable.

- El compromiso entre la precisión del método (promedio total en 46min $T\bar{P}_{k,y}$) y el uso del ancho de banda de comunicación (total de mensajes) se obtuvo en la figura 11.9 ante la variación del límite del evento $R_{A,lim}$. Se concluye a partir de esta figura que un valor adecuado en $R_{A,lim}$ permite ajustar la precisión deseada del método cooperativo según los recursos disponibles de la plataforma (capacidad de cómputo y ancho de banda). Valores bajos de $R_{A,lim}$ producen una mayor precisión ($T\bar{P}_{k,y}$ bajo) lo que requiere un mayor intercambio de mensajes (se incrementa el ancho de banda y recursos computacionales requeridos), y por el contrario, valores altos de $R_{A,lim}$ permite reducir el costo (computacional y en comunicaciones) del método a cambio de una reducción en su precisión. Por lo tanto, es posible obtener una solución de

compromiso entre la precisión requerida en la localización cooperativa del grupo de robots y la cantidad de recursos computacionales y ancho de banda utilizados en las plataformas (en cada R_i del grupo), con una selección adecuada del límite del evento $R_{A,lim}$ ($0{,}2 < R_{A,lim} < 7$ en el caso multirobot).

- Por último, el estudio realizado para obtener la figura 11.9 y el rango adecuado de $R_{A,lim}$ se puede repetir durante el diseño del grupo cooperativo, de acuerdo a sus características y condiciones del intercambio de mensajes así como a los recursos disponibles en las plataformas que lo integran. Con esto se puede obtener un rango adecuado de compromiso para $R_{A,lim}$, a utilizar en la totalidad de G_R o uno distinto para cada R_i. A partir de la figura 11.9 definió por ejemplo $0{,}2 < R_{A,lim} < 7$. El valor utilizado para $R_{A,lim}$ puede ser definido a voluntad durante la etapa de diseño del sistema para predefinir el máximo intercambio de mensajes deseado, o bien podría modificarse dinámicamente durante la ejecución del programa en cada robot. Finalmente se puede observar un video con el resumen de las pruebas presentadas para el esquema de localización cooperativa en `/http:///idecona./ai2./upv./es//multirobot`.

12 | Conclusiones y Trabajo Futuro

Se exponen a continuación las conclusiones obtenidas en el desarrollo de la presente tesis, así como las posibles líneas de extensión de los métodos propuestos como trabajos a realizar en un futuro. Se indican finalmente los artículos presentados en congresos y revistas que se derivan de los resultados obtenidos.

12.1 Conclusiones

Objetivos alcanzados

A partir del estudio de los antecedentes del capítulo 2, se determinó la necesidad de desarrollar algoritmos de fusión para la localización precisa de robots y grupos de robots móviles de recursos limitados (en capacidad de cómputo, memoria y ancho de banda para comunicación) de forma que no se requiera de un ordenador externo para realizar la fusión y transmitirla al robot, evitando además la utilización constante de la información global o relativa, y buscando un uso eficiente del medio de comunicación de forma que se puedan ejecutar tareas adicionales a la localización.

Con los algoritmos y modelos desarrollados (capítulos del 3 al 7) y los buenos resultados obtenidos (capítulos del 8 al 11) se cumple el objetivo principal de la presente tesis, en cuanto se propusieron los nuevos algoritmos utilizando los principios de control y muestreo basado en eventos, con los que se obtiene una precisión adecuada en la estimación de la postura del robot o del grupo de robots, con un con error acotado y realizando la fusión de distintos sensores de forma eficiente computacionalmente, y con un uso reducido del ancho de banda para el acceso a la información global o relativa.

Con los resultados obtenidos se cumple además con los objetivos específicos:

- Se realizó una revisión del estado del arte en localización a partir de las técnicas de fusión de datos basadas principalmente en el filtro de Kalman (KF, EKF, UKF, etc.) y en localización cooperativa de grupos de robots, mediante el cual se realizó la justificación y el planteamiento de los objetivos alcanzados en la presente tesis.

- Se obtuvieron modelos dinámicos locales y cinemáticos globales de un robot móvil en configuración diferencial y Ackerman, utilizando métodos de descomposición por partículas inerciales. Con estos modelos se obtiene la dinámica de la plataforma al colocar acelerómetros en posiciones adecuadas, reduciendo la cantidad de parámetros que se deben identificar, así como la complejidad computacional de los mismos. Se identificaron además los modelos dinámicos de velocidad y aceleración para los motores de corriente continua de la plataforma LEGO NXT empleada en las pruebas experimentales.

- Se realizó un estudio sobre los métodos actuales en estimación de estados y fusión sensorial, a partir del cual se seleccionaron las técnicas de fusión basadas en el filtro de Kalman, en cuanto consideran el efecto del ruido en el proceso y la medición y son adecuadas para el desarrollo de algoritmos de fusión para la mejora de la localización en plataformas de recursos limitados.

- Al separar las mediciones locales (velocidades y aceleraciones) y globales (postura de un GPS, cámara, etc.) y al utilizar los modelos dinámicos obtenidos, se desarrollaron los algoritmos de fusión EKF/KF en cascada, los cuales permiten la localización de un robot móvil de recursos limitados al no considerar problemas en el acceso a la información global.

- Al incorporar los principios de control y muestreo basado en eventos en los algoritmos en cascada desarrollados, se obtuvieron los algoritmos de fusión EKF/KF en cascada con corrección basada en eventos, los cuales permiten la incorporación de la información global en la fusión para la localización del robot móvil de recursos limitados, considerando

casos de retardos y problemas de acceso a la medición global. Se logra reducir la utilización del ancho de banda entre el robot y el sensor global, utilizando este sensor únicamente si un evento, definido con base en la covarianza del error de estimación de la postura del robot, excede un nivel determinado. Se realizaron distintas definiciones del evento y de su límite que son adecuadas para navegación en interiores y exteriores.

- Para el caso de un grupo heterogéneo de robots móviles, se realizó la extensión de la definición de los algoritmos en cascada basados en eventos para incorporar la información relativa (distancias, ángulos y posturas de los robots vecinos) como un sensor de postura adicional, de forma que se obtuvo el algoritmo de localización cooperativo distribuido basado en eventos, el cual reduce la utilización del ancho de banda de las comunicaciones entre robots. En este caso, se realiza el intercambio de información cooperativo según la definición de un evento basado no en el error de navegación, sino en la covarianza del error de estimación de la postura.

- Se desarrollaron las plataformas experimentales de recursos limitados basadas en el LEGO NXT para las configuraciones diferencial y Ackerman en interiores y exteriores. Se realizó la calibración de los distintos sensores y se expuso el algoritmo de navegación utilizado, el esquema de sensorización global y la corrección del tiempo de retardo debido a la comunicación con el sensor global. Se obtuvieron además los parámetros necesarios para los filtros de Kalman propuestos y se describió la plataforma de simulación multirobot utilizada en las pruebas de localización cooperativa.

- A partir de las pruebas experimentales desarrolladas para las distintas plataformas, se realizó la comparación de los algoritmos propuestos con los métodos existentes mediante simulaciones, utilizando la información sensorial de los robots al seguir distintas trayectorias. Con esto se pudo comprobar el buen desempeño de los algoritmos propuestos en cuanto a la precisión y error acotado de la estimación, pero con la ventaja de un uso reducido de recursos computacionales.

- Se obtuvo con los algoritmos propuestos un buen desempeño, precisión y uso reducido del ancho de banda según las pruebas con los algoritmos implementados en los robots individuales, comprobados mediante las pruebas de larga duración. Se observó además el compromiso entre el ancho de banda y la precisión de los métodos, la cual es ajustada mediante el límite del evento en los algoritmos. Se comprobó el ahorro de recursos computacionales al determinar el tiempo de ejecución de los algoritmos, el cual, al ser muy reducido respecto a los métodos tradicionales, permite la implementación de los esquemas de fusión en cascada propuestos en la unidad de control de los robots de recursos limitados.

- Se observó un desempeño adecuado, error acotado y consumo reducido del ancho de banda para el algoritmo de localización cooperativo basado en eventos, al realizar diversas simulaciones utilizando la plataforma de simulación multirobot, tanto para el seguimiento de trayectorias cíclicas predefinidas (lineal) como para un grupo de robots con evitación de obstáculos Braitenberg navegando en un entorno cerrado. Se observó el ahorro de recursos considerable respecto a los métodos existentes y se estudió el compromiso entre la precisión y el uso del ancho de banda de comunicación entre robots, regulado por el nivel del evento del algoritmo cooperativo propuesto.

HITOS

- Se ha propuesto en la presente tesis múltiples esquemas de localización basados en la fusión sensorial con los filtro de Kalman en cascada y la corrección global y relativa basada en eventos, los cuales, según los resultados obtenidos en las diversas plataformas utilizadas, funcionan de manera adecuada y eficiente, proporcionando una solución fiable al problema de localización de robots o grupos de robots al obtener la postura con una precisión adecuada, con error de estimación acotado y uso reducido del ancho de banda y demás recursos de la plataforma (tiempo de cómputo, memoria, batería, etc.). Esto permite su implementación en variedad de aplicaciones y plataformas, al poder adaptar la precisión

de los algoritmos basados en eventos según los recursos computacionales y ancho de banda disponibles en la plataforma a utilizar, mediante la modificación del límite del evento definido según el área normalizada de los elipsoides 3σ de la covarianza del error de estimación $P_{k,p}$, lo que permitirá dedicar tiempo de ejecución y ancho de banda a las demás tareas de la plataforma (navegación, control, supervisión, coordinación, etc.), además del método de localización.

- Los modelos por partículas dinámicas presentaron un funcionamiento adecuado en las pruebas realizadas, al proveer una predicción adecuada de la postura que es corregida mediante la fusión sensorial. Presentan además una complejidad computacional reducida y facilitan el desarrollo de los filtros en cascada.

- La estimación de la postura a partir de la información local (integración de velocidades y aceleraciones) sin información global (cámara, GPS, SLAM, etc.) presenta el problema de acumulación del error, el cual crece indefinidamente conforme el robot navega en el entorno y transcurre el tiempo. Solo se reduce este error al incorporar la información global de precisión adecuada de forma directa (mediante el sensor global) o de forma indirecta (mediante mediciones relativas). Por esta razón, la fusión sensorial consume recursos considerables al incorporar la información de múltiples fuentes de información, lo que requiere en el caso de los filtros de Kalman la inversión de matrices de dimensión alta (y la raíz cuadrada en el caso del UKF), por lo que en general no son implementables en plataformas de recursos limitados.

- Los algoritmos en cascada propuestos tienen un funcionamiento (precisión) similar a algoritmos más complejos pero consumiendo menos recursos (memoria y tiempo de cómputo), de forma que permiten incorporar la información global en la estimación de la postura sin requerir el cálculo de inversas de matrices de dimensión alta, por lo que pueden implementarse en plataformas de recursos limitados.

- La corrección basada en eventos utiliza la información global (o relativa) únicamente cuando es necesario, de acuerdo a la definición del evento

basada en el área de los elipsoides 3σ, con lo que se incorpora ésta información si $P_{k,p}$ ha crecido considerablemente, lo que reduce el uso del ancho de banda y recursos de la plataforma además de producir un ahorro en el consumo de baterías. Además, el esquema basado en eventos permite incorporar la corrección necesaria al retardo debido a las comunicaciones, el cual es medible por la plataforma o con la utilización de un dispositivo externo (reloj de tiempo real).

- El esquema basado en eventos permite incorporar diversas definiciones del evento y su límite $R_{A,lim}$, de forma que se puede adaptar el esquema a las condiciones de funcionamiento del robot o del grupo de robots, permitiendo por ejemplo incorporar la información del número de satélites utilizados por el sensor GPS en el caso de navegación en exteriores o el número de vecinos detectados en el caso de localización cooperativa.

- Existe un compromiso entre la precisión (índice IAE, o covarianza promedio total en un grupo de robots) y el consumo de recursos (costo computacional definido como la cantidad de veces que se realiza la fusión global o relativa y el costo en el ancho de banda según el número de consultas realizadas al sensor global o de la cantidad total de mensajes transmitidos por la localización cooperativa) en el caso de los algoritmos basados en eventos propuestos. Este compromiso se regula mediante la asignación del límite del evento $R_{A,lim}$, de forma que al ajustar su valor se puede mejorar la precisión de la estimación consumiendo más recursos (si estuviesen disponibles) y viceversa. El estudio de esta relación es necesario, con el fin de obtener un rango adecuado de $R_{A,lim}$ según las características de la plataforma o grupo multirobot.

12.2 Trabajo Futuro

A partir del trabajo realizado en la presente tesis se pueden plantear distintas líneas de extensión a desarrollar en investigaciones futuras:

- Desarrollo de modelos en partículas para el caso de movimiento en 3D así como para otras configuraciones. Por ejemplo, modelar los robots con ruedas omnidireccionales o mecanum o robots del tipo carretilla [63], al agregar las partículas correspondientes para representar el movimiento en las ruedas de la plataforma. Además se puede extender el método a plataformas aéreas o submarinas.

- Extensión de los filtros de fusión en cascada y por eventos para la localización mediante otros esquemas de fusión sensorial. Para esto se pueden considerar otras versiones del filtro de Kalman, incorporando por ejemplo el ruido coloreado, la estimación con modelo múltiple así como las versiones con estabilidad numérica mejorada (versión raíz cuadradada y representaciones adicionales de las ecuaciones del KF), o bien utilizando filtros adaptativos con ajuste en tiempo real de sus parámetros (Matrices Q_k,R_k, modelo, etc.), así como el monitoreo de residuos para comprobar el desempeño del filtro. Por otra parte se pueden considerar los filtros de fusión que *no* estiman la postura del robot, sino el error en la estimación odométrica, sin embargo estos requieren un acceso constante a la medición global del sensor para calcular el error y poder operar adecuadamente.

- En el caso de plataformas con recursos computacionales no limitados, se pueden utilizar las estrategias basadas en eventos para disminuir la utilización del ancho de banda al comunicarse con sensores o referencias globales y con robots vecinos. En este caso se pueden incorporar versiones más complejas del filtro de Kalman, como por ejemplo el UKF y el CKF, los cuales mejorarían la estimación de la covarianza en el caso de requerir modelos más complejos de la plataforma.

- Se puede incorporar dentro de los esquemas de fusión propuestos otras representaciones de la orientación del robot, como por ejemplo utilizando los ángulos de Euler, cuaterniones o mediante matrices de rotación,

que pueden proporcionar ventajas en la fusión entre distintos sensores de orientación, a pesar de requerir en algunos casos de transformaciones no lineales para la conversión entre representaciones. Además, se puede comparar la precisión de la estimación al incorporar la brújula como un sensor global en el esquema de fusión (realizando la transformación de coordenadas correspondiente), a diferencia del enfoque utilizado en las pruebas realizadas al considerar la brújula como un sensor de velocidad angular.

- Extender el estudio de los algoritmos basados en eventos para considerar la incorporación de múltiples eventos dentro del esquema, permitiendo definir y utilizar más de un tipo de evento de forma simultánea para, por ejemplo, en el caso multirobot realizar la fusión si el área 3σ supera un nivel predeterminado y que además se realice siempre que se detecten 3 robots vecinos. En este caso se debe realizar el estudio de la influencia de la combinación de los distintos eventos en la precisión de la estimación y el consumo de recursos en la plataforma, de forma que se puedan establecer los niveles de los eventos adecuados según los recursos del robot en la localización o localización cooperativa. Además, se puede estudiar el caso de ajustar dinámicamente el límite del evento en tiempo real, según la cantidad de recursos (tiempo de cómputo y ancho de banda) disponibles en la plataforma.

- Considerar la utilización del esquema basado en eventos para modificar otros aspectos de la fusión sensorial. Por ejemplo, se puede realizar una definición del evento que permitan el funcionamiento de la localización en el caso de una falla en alguno de los sensores (local o global), al realizar la modificación de la matriz R_k del filtro de fusión incrementando o reduciendo los valores de los sensores que han fallado. En este caso el evento se definirá al supervisar los valores de los sensores y detectar anomalías en la señal del sensor. En este caso se pueden incorporar filtros KF con estimación multimodelo o bien establecer distintos filtros a utilizar según los sensores que presenten un fallo (variando el modelo, dimensiones y parámetros del filtro), con el fin de realizar la mejor estimación posible con la información que esté disponible en el robot en cada instante k.

- Extensión del estudio del filtro de fusión para localización en exteriores al considerar fuentes adicionales del error en la medición GPS y ajustar los métodos basados en eventos a estas condiciones, considerando además del número de satélites detectados, los casos de problemas de disponibilidad y medición multiruta así como los efectos atmosféricos, etc. [55, 153]. Esto con el fin de mejorar la estimación de la postura al fusionar la información del GPS únicamente si es lo suficientemente precisa. En este caso se puede considerar la inversión de la definición del evento, al fusionar la información global siempre que esté disponible, y utilizar la fusión con la información local en los casos en donde no se reciba adecuadamente la señal satelital. En este caso se desactivaría la corrección global si el evento definido con la covarianza del error de estimación supera un nivel predeterminado, o bien al definir un evento que dependa de la covarianza de la medición del GPS (medición de la matriz R_k del GPS en tiempo real).

- Estudio de la influencia del esquema de localización basado en eventos en el consumo de energía de la plataforma, con el fin de comprobar la influencia del límite del evento utilizado en la duración de las baterías del robot. Además, se pueden definir distintos modos de fusión (utilizando más o menos sensores, usando o no el ancho de banda, etc) en caso de requerir la limitación del consumo de energía en la plataforma.

- Extensión de las aplicaciones propuestas: se pueden extender los métodos propuestos a grupos de robots híbridos compuestos por equipos terrestres y aéreos, en cuyo caso se deben considerar las condiciones de coordinación entre los equipos así como la localización cooperativa para distribuir la información global (GPS en los robots aéreos) a los robots sin acceso directo a este sensor, utilizando sensores relativos (cámaras por ejemplo).

- Incorporar sensores avanzados al esquema de fusión adaptando las dimensiones correspondientes del filtro (matrices Q_k, R_k). Por ejemplo se pueden utilizar sensores de flujo óptico, el cual utiliza una cámara para determinar la velocidad y aceleración de la plataforma mediante el procesamiento de imágenes consecutivas capturadas con una frecuencia

elevada [47, 94, 19, 74]. Además, se puede considerar el caso de mapeo y navegación simultánea (SLAM), incorporando un sensor de barrido láser dentro del esquema de fusión sensorial basado en eventos.

- Finalmente, se puede realizar la extensión del método multirobot a escenarios no limitados (sin bordes, o espacios de gran dimensión). En este caso se tiene el problema de que algunos integrantes del grupo no se detectarán frecuentemente, por lo que se deberán formar subgrupos de robots para explorar el entorno, dentro de los cuales se realizará la localización cooperativa. Estos subgrupos se definirán según la misión a cumplir y la cobertura requerida en el área, pero deben incorporar al menos un robot con capacidad de localización global (sensor GPS, cámara, SLAM, etc.) para proveer convergencia global al subgrupo.

12.3 Artículos Publicados

La validez de la presente tesis queda demostrada con la variedad de publicaciones realizadas a partir de los resultados obtenidos en la misma, los cuales cumplen con todos los objetivos propuestos. Estos artículos presentados en congresos y revistas se detallan a continuación:

- **L. Marín**, M. Vallés, A. Valera, P. Albertos. "Implementation of a bug algorithm in the e-puck from a hybrid control viewpoint". En: *15th International Conference on Methods and Models in Automation and Robotics (MMAR)*, 23-26 Aug. 2010, Miedzyzdroje (Poland), páginas: 174 - 179, ISBN: 978-1-4244-7828-6, DOI: 10.1109/MMAR.2010.5587242. Indexado en: The Conference Proceedings Citation Index (ISI Web of Science database), SciVerse Scopus database y en IEEE Xplore Digital Library.

- **L. Marín**, M. Vallés, A. Valera, P. Albertos. "Control de Trayectorias en el Robot Móvil E-Puck". En: *XXXI Jornadas de Automática*, Jaén (España), 8 - 10 de septiembre, 2010, ISBN: 978-84-693-0715-1.

- G. Sanahuja a, A. Valera b, A.J. Sánchez b, C. Ricolfe-Viala b, M. Vallés b, **L. Marín**. "Control Embebido de Robots Móviles con Recursos

Limitados Basado en Flujo Óptico". En: *Revista Iberoamericana de Automática e Informática Industrial*, Volumen 8, Número 3, 2011, páginas 250-257. ISSN: 1697-7912, DOI: 10.1016/j.riai.2011.06.012. Factor de Impacto 0.375, cuartil Q4 (Automation & Control Systems; Robotics).

- A. Valera, M. Valles, **L. Marín**, A. Soriano, A. Cervera, A. Giret. "Application and evaluation of Lego NXT tool for Mobile Robot Control". En: *18th World Congress of the International Federation of Automatic Control (IFAC)*, Milano (Italy), Agosto 28 - Setiembre 2, 2011, ISBN: 978-3-902661-93-7, DOI: 10.3182/20110828-6-IT-1002.01846. Indexado en: SciVerse Scopus database y en IFAC-PapersOnLine Digital Library.

- A. Valera, M. Vallés, **L. Marín**, P. Albertos. "Design and Implementation of Kalman Filters applied to Lego NXT based Robots". En: *18th World Congress of the International Federation of Automatic Control (IFAC)*, Milano (Italy), Agosto 28 - Setiembre 2, 2011, ISBN: 978-3-902661-93-7, DOI: 10.3182/20110828-6-IT-1002.02799. Indexado en: SciVerse Scopus database y en IFAC-PapersOnLine Digital Library.

- **L. Marín**, M. Vallés, A. Valera, P. Albertos. "Mejora en la Navegación de Robots con Recursos Limitados Mediante Fusión de Datos de Distintos Sensores". En: *XXXII Jornadas de Automática*, Sevilla (España), 7 - 9 septiembre, 2011, ISBN: 978-84-694-6454-0.

- A. Soriano, **L. Marín**, J. Gómez-Moreno, A. Valera, M. Vallés, A. Giret. "Organizaciones Holónicas Multiagente Para Resolver Misiones Mediante Robots Móviles.". En: *XXXIII Jornadas de Automática*, Vigo (España), 5 - 7 septiembre, 2012, ISBN: 978-84-8158-583-4.

- **L. Marín**, M. Vallés, A. Soriano, A. Valera, P. Albertos. "Event-Based Localization in Ackermann Steering Limited Resource Mobile Robots". En: *IEEE-ASME Transactions on Mechatronics*, 2013, Volumen 19 Número 4, páginas 1171-1182, ISSN: 1083-4435, DOI: 10.1109/TMECH.2013.2277271. Factor de Impacto 3.315, cuartil Q1 (Automation & Control Systems; Electrical & Electronic Engineering; Manufacturing - Mechanical Eng.).

- **L. Marín**, A. Soriano, V. Mayans, M. Vallés, A. Valera, P. Albertos. "Localización asistida por GPS para robots móviles en configuración Ackermann de recursos limitados". En: *XXXIV Jornadas de Automática*, Terrassa, Barcelona (España), 4 - 6 septiembre, 2013, ISBN: 978-84-616-5063-7.

- A. Soriano, **L. Marín**, R. Juan, J. Cazalilla, A. Valera, M. Vallés, P. Albertos. "Plataforma robótica de bajo coste y recursos limitados basada en Arduino y dispositivos móviles". En: *XXXIV Jornadas de Automática*, Terrassa, Barcelona (España), 4 - 6 septiembre, 2013, ISBN: 978-84-616-5063-7.

- **L. Marín**, M. Vallés, A. Soriano, A. Valera, P. Albertos. "Multi Sensor Fusion Framework for Indoor-Outdoor Localization of Limited Resource Mobile Robots". En: *Sensors, Special Issue "State-of-the-Art Sensors Technology in Spain 2013"*, 2013, Volumen 13 Número 10, páginas 14133-14160, ISSN 1424-8220, DOI: 10.3390/s131014133. Factor de Impacto 1.953, cuartil Q1 (Instruments & Instrumentation).

- A. Soriano, **L. Marín**, M. Vallés, A. Valera, P. Albertos. "Low Cost Platform for Automatic Control Education Based on Open-Source Hardware". En: *19th World Congress of the International Federation of Automatic Control (IFAC)*, Cape Town (South Africa), 24-29 de Agosto, 2014, Aceptado como artículo regular a ser expuesto el 29 de agosto. Indexado en: SciVerse Scopus database y en IFAC-PapersOnLine Digital Library.

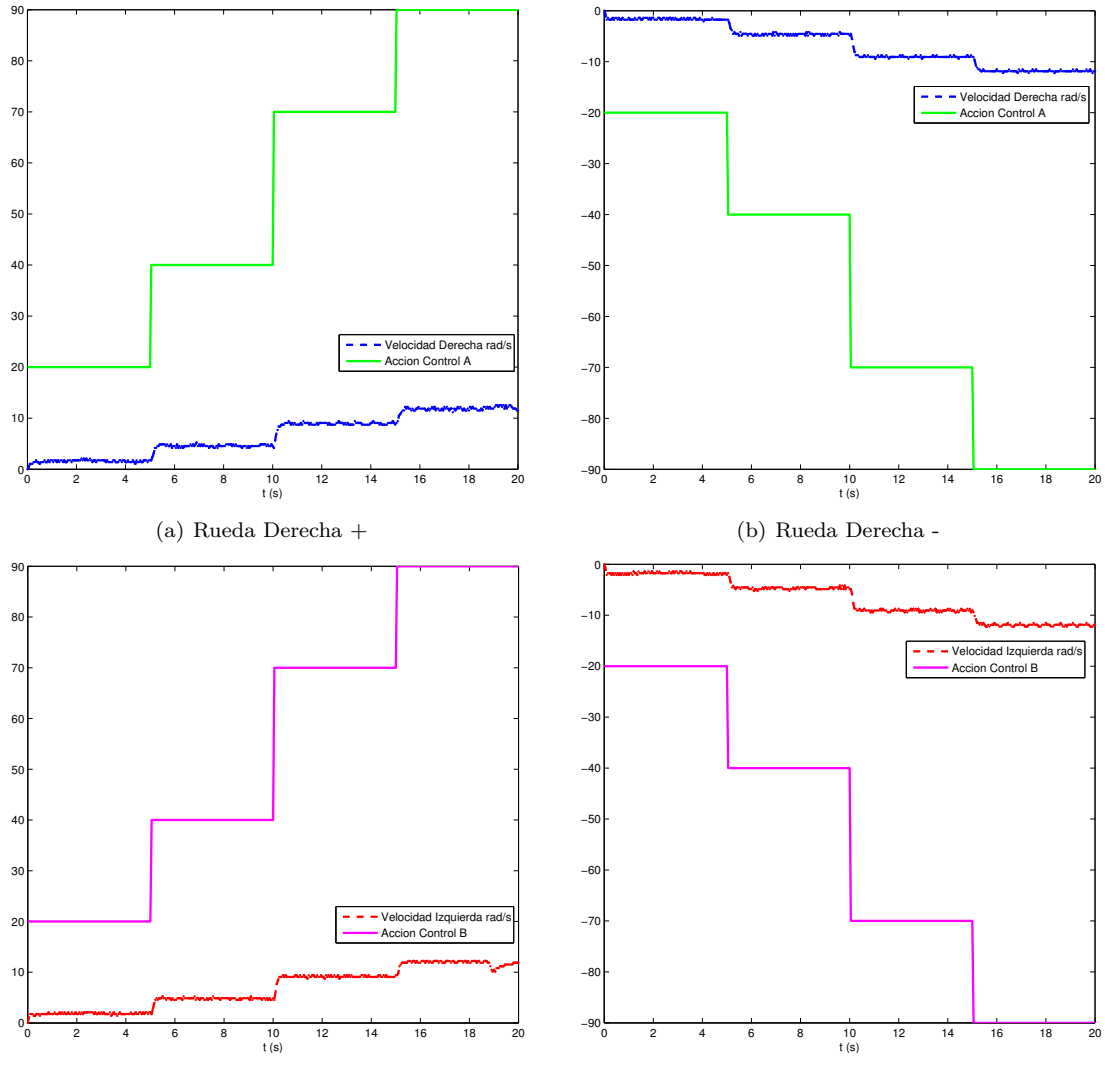
A | Modelo dinámico de los motores del LEGO NXT

Se identifica a continuación un modelo para las velocidades de las ruedas del motor y calcular su derivada para obtener una aproximación a la aceleración lineal o angular de cada rueda. Este tipo de modelos pueden ser utilizados como entrada a los modelos de aceleraciones propuestos en el capítulo 3 en caso de no disponer de acelerómetros en la plataforma.

En lugar de utilizar un único escalón para realizar la identificación, se utiliza una seria de escalones positivos y negativos en la acción de control con el fin de obtener la función de transferencia de bucle abierto del motor del robot. El procedimiento expuesto a continuación sigue el ejemplo de identificación en un robot diferencial con motores DC basado en la plataforma LEGO MINDSTORMS NXT, pudiendo ser aplicado en otras plataformas comerciales o de construcción propia.

La identificación del modelo de velocidad angular de los motores del robot se realiza al introducir un escalón en la acción de control, para obtener la respuesta dinámica del motor como una función de transferencia de bucle abierto de primer orden para la velocidad. En el caso del LEGO NXT, la acción de control es el voltaje PWM especificado en %, así un valor de +100 indica que se aplica el voltaje máximo posible al motor en sentido horario. Se realizan dos pruebas con escalones positivos y negativos para cada motor (cuatro pruebas en total), las cuales se observan en la figura A.1.

En total se deben identificar 8 funciones de transferencia para cada motor (16 en total). Para facilitar la tarea se utiliza el «System Identification Toolbox» de Matlab ®(pueden emplearse herramientas similares de identificación de modelos como el Scilab, Octave, entre otros). 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 éstas



(a) Rueda Derecha +

(b) Rueda Derecha -

(c) Rueda Izquierda +

(d) Rueda Izquierda -

Figura A.1: Pruebas realizadas a los motores del LEGO NXT

(modelos locales). Los parámetros del modelo se muestran en la tabla A.1 en donde se calculan los modelos con los parámetros promedio de cada rueda.

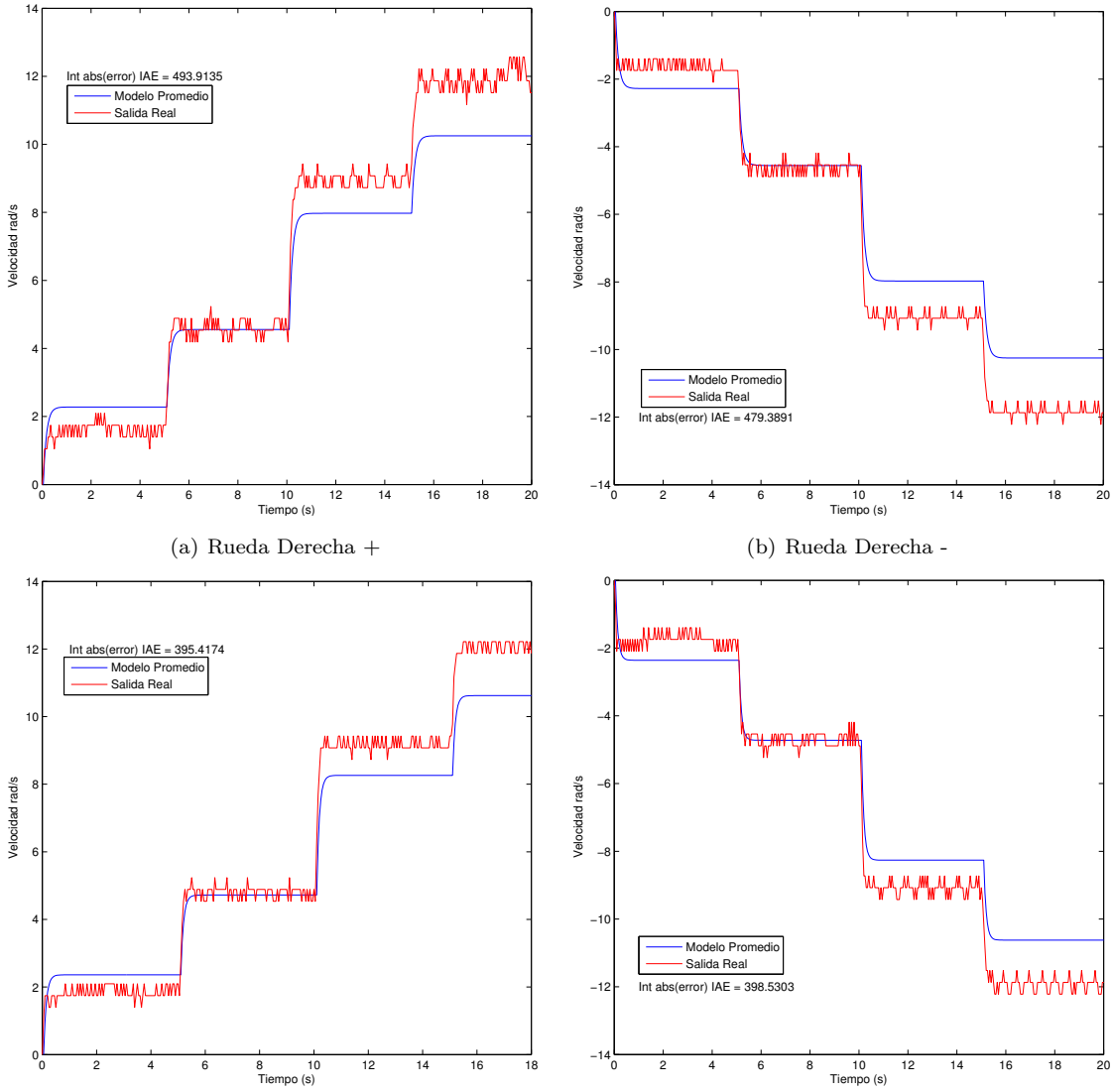
Tabla A.1: Modelos locales para los motores del LEGO NXT

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/individual) 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 A.2.

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, utilizando por ejemplo la función «pem» en Matlab). Los resultados de la identificación se resumen en la tabla A.2 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



(a) Rueda Derecha +

(b) Rueda Derecha -

(c) Rueda Izquierda +

(d) Rueda Izquierda -

Figura A.2: Respuesta del modelo local promedio de las ruedas del LEGO NXT

Tabla A.2: 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)	$G_{gMA} = \frac{0.1265}{0.141\ s+1}$		$G_{gMB} = \frac{0.1288}{0.1061\ s+1}$	

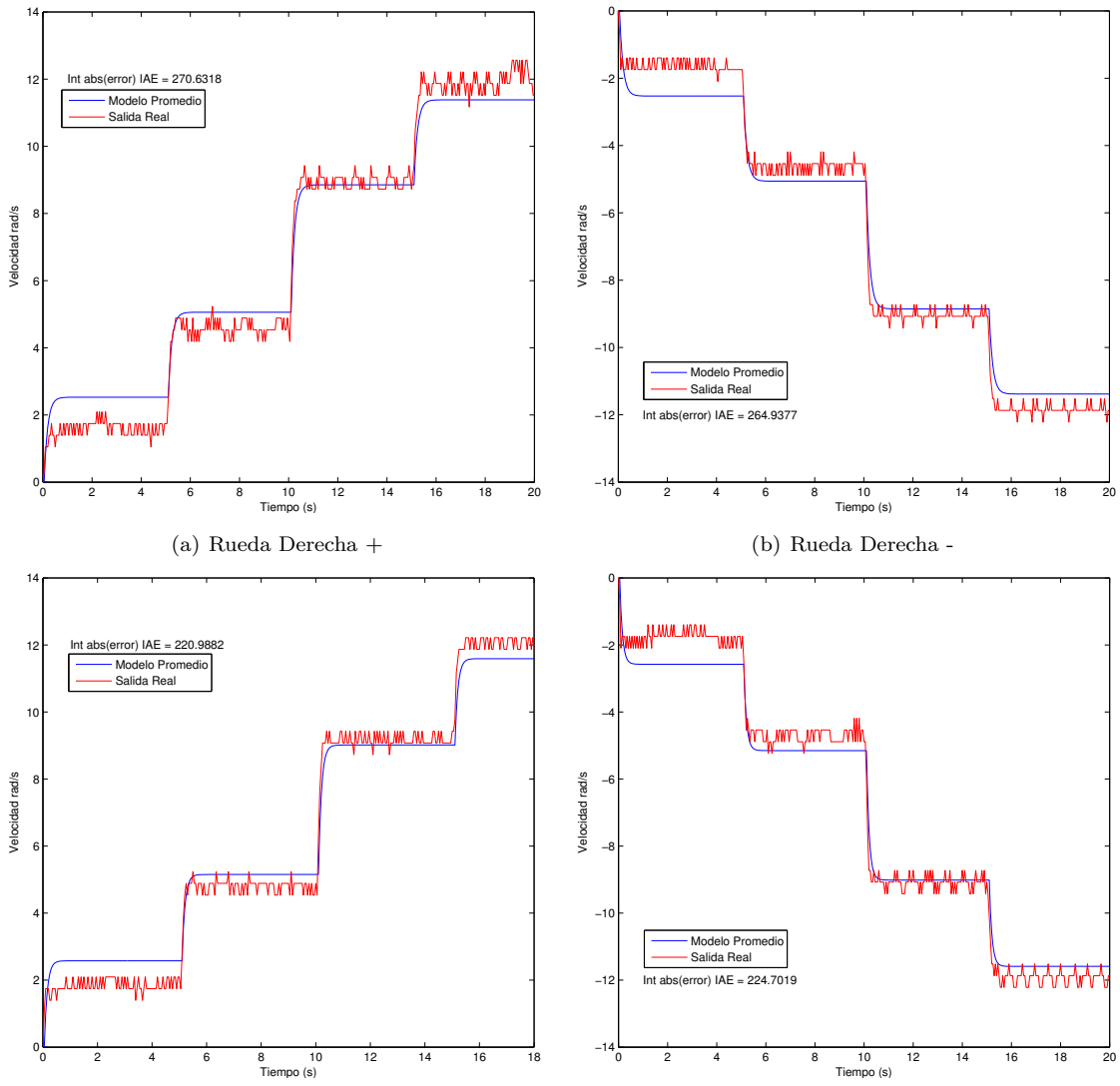
(global de cada rueda) se grafica su respuesta ante la entrada de prueba (serie de escalones) y se observa su desempeño al compararlo con la salida del motor real. Nuevamente se calcula el valor de IAE para poder comparar con los modelos promedio locales. Las pruebas para el modelo se muestran en la figura A.3.

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

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 (A.1). Su desempeño en las pruebas con los datos de los motores se muestra en la figura A.4.

$$G_{gMA} = \frac{0{,}1276}{0{,}1235\ s + 1} \tag{A.1}$$

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. Además, cabe señalar que al utilizar un tren de escalones positivo y negativo junto con la identificación global para obtener el modelo, se consigue un menor índice IAE que en el caso local (al identificar cada modelo por separado) y en el caso de emplear el modelo de un único escalón para todo el rango de actuación.



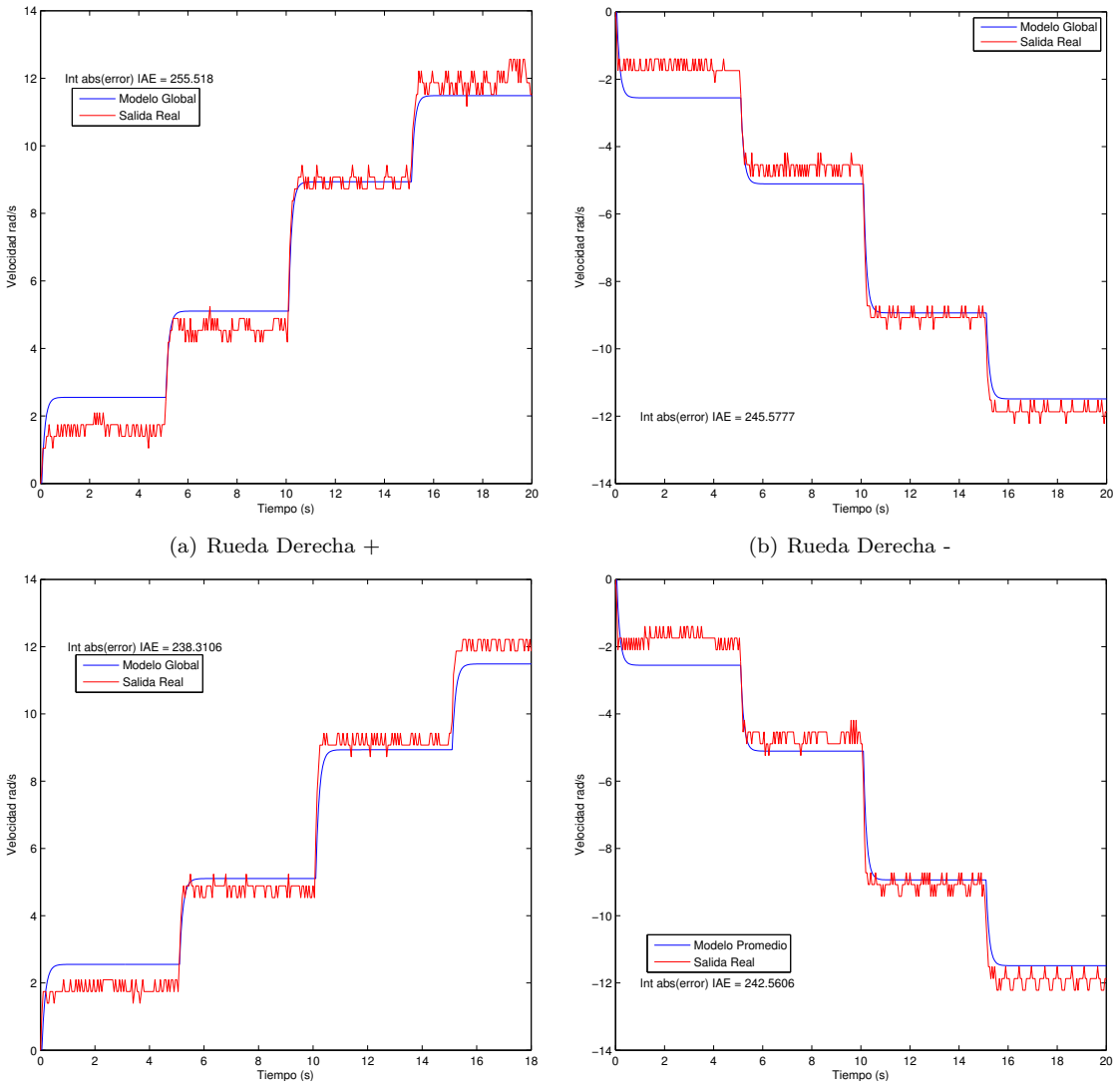
(a) Rueda Derecha +

(b) Rueda Derecha -

(c) Rueda Izquierda +

(d) Rueda Izquierda -

Figura A.3: Respuesta del modelo global de las ruedas del LEGO NXT



(a) Rueda Derecha +

(b) Rueda Derecha -

(c) Rueda Izquierda +

(d) Rueda Izquierda -

Figura A.4: Respuesta del modelo global Promedio de las ruedas del LEGO NXT

Para determinar la aceleración de las ruedas se deriva la ecuación (A.1) para obtener (A.2).

$$G_{gAcc} = \frac{a_{ccRueda}}{V_{control}} = \frac{0{,}1276\ s}{0{,}1235\ s + 1} \tag{A.2}$$

Este modelo relaciona la acción de control $V_{control}$ con la aceleración de la rueda $a_{ccRueda}$. Se discretiza (A.2) con un periodo de muestreo $T_s = 50ms$ (adecuado para el LEGO NXT, puede modificarse según las características de la plataforma utilizada) con lo que se obtiene (A.3). Finalmente se escribe este modelo en forma de ecuación en diferencias en (A.4) lo que facilita su implementación dentro del robot.

$$G_{gAcc}(z) = \frac{1{,}033z - 1{,}033}{z - 0{,}6671} = \frac{1{,}033 - 1{,}033z^{-1}}{1 - 0{,}6671z^{-1}} \tag{A.3}$$

$$a_{R,k} = 1{,}033V_{c,k} - 1{,}033V_{c,k-1} + 0{,}6671a_{R,k-1} \tag{A.4}$$

B | Propiedades estadísticas de una variable aleatoria

Para el desarrollo de los filtros de estimación se requiere conocer las definiciones matemáticas de las propiedades estadísticas de una variable aleatoria X, las cuales se resumen a continuación (un desarrollo extenso puede consultarse en [151]).

Una variable aleatoria X se puede definir como una función que realiza un mapeo desde un conjunto de valores asociados a un experimento aleatorio (mediciones o resultados, no necesariamente numéricos) hacia un conjunto de números reales (rango).

La función de distribución de probabilidad *PDF* asociada a X se denota como F_X y describe la variación de los resultados de un experimento aleatorio al asignarles una probabilidad P de ocurrencia. De esta forma $F_X(x) = P(X \leqslant x)$ siendo x una variable no aleatoria o una constante. Cabe destacar que los posibles valores de esta función se limitan a $F_X(x) \in [0, 1]$.

La derivada de la *PDF*, $f_X(x) = \frac{d}{dx} F_X(x)$, se define como la función de densidad de probabilidad *pdf*, la cual solo puede ser positiva o cero: $f_X(x) \geqslant 0$. Además, su integral cumple con $\int_{-\infty}^{\infty} f_X(x)dx = 1$.

El valor esperado (expectación, promedio, media o valor central) de X es el valor promedio obtenido de una gran cantidad de experimentos realizados. Éste se define en la ecuación (B.1a) para el caso general (número de experimentos ∞). En el caso de una función $g(X)$ aplicada a X y cuyo resultado

por definición será también una variable aleatoria, se define su valor esperado mediante la ecuación (B.1b).

$$E[X] = \bar{x} = \int_{-\infty}^{\infty} x f_X(x) dx \tag{B.1a}$$

$$E[g(X)] = \int_{-\infty}^{\infty} g(x) f_X(x) dx \tag{B.1b}$$

La varianza de X es una medición de cuanta variación respecto a la media $\bar{x}$ se espera en la variable aleatoria, es decir, cuanta variabilidad tiene X. Si la varianza es cero implica que X siempre tiene el mismo valor; por el contrario si X puede tomar cualquier valor en $\Re$ con igual probabilidad, entonces su varianza es ∞. La varianza se define mediante la ecuación (B.2). La raíz cuadrada de la varianza se conoce como la desviación estándar y se denotada por σ.

$$\sigma_X^2 = E\left[(X - \bar{x})^2\right] = \int_{-\infty}^{\infty} (x - \bar{x})^2 f_X(x) dx \tag{B.2}$$

Una variable aleatoria X se define como Gaussiana o normal si su correspondiente *pdf* se puede definir mediante la ecuación (B.3) al utilizar las definiciones previas de $\bar{x}$ y σ. Para una X con $\bar{x} = 0$, la variación de su *pdf* Gaussiana ante distintos $\sigma = \{3\ 2\ 1{,}5\ 1\ 0{,}7\}$ se muestra en la figura B.1. De esta figura se observa como la *pdf* se estrecha respecto a $\bar{x}$ (menor error respecto a la media) según disminuye su variabilidad σ.

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left[-(x-\bar{x})^2/2\sigma\right]} \tag{B.3}$$

En el caso de tener múltiples variables aleatorias, se puede estudiar si tienen alguna relación o influencia entre ellas. Si se consideran las variables aleatorias X e Y, se puede definir el concepto de independencia, según el cual, la ocurrencia de un evento en X no tiene efecto en la probabilidad de ocurrencia de un evento en Y. De esta forma, X e Y son independientes

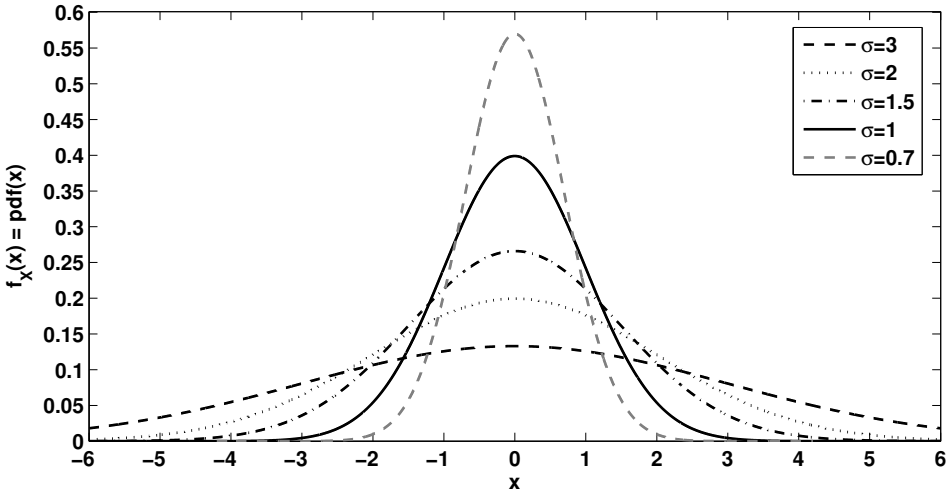


Figura B.1: Variación de la función de densidad de probabilidad para una variable aleatoria Gaussiana de media cero según su desviación estándar [151].

si se puede definir la densidad *pdf* conjunta como la multiplicación de las densidades individuales: $f_{XY}(x,y) = f_X(x)\, f_Y(y)$.

Además, para el caso multivariable, se puede realizar la extensión de la definición relativa a la varianza del caso monovariable. La covarianza entre dos variables vectoriales aleatorias X e Y se define según la ecuación (B.4), la cual se puede utilizar para calcular la autocovariancia de un vector aleatorio X mediante la ecuación (B.5). Ambas ecuaciones se pueden modificar (considerando $\{\bar{X}, \bar{Y}\} = 0$) para calcular la correlación entre (X, Y) y la autocorrelación de X tal y como se muestra en las ecuaciones (B.6a) y (B.6b) respectivamente.

$$C_{XY} = E\left[\left(X - \bar{X}\right)\left(Y - \bar{Y}\right)^T\right] = E\left[XY^T\right] - \bar{X}\bar{Y}^T \qquad \text{(B.4)}$$

$$C_X = E\left[\left(X-\bar{X}\right)\left(X-\bar{X}\right)^T\right] \tag{B.5}$$

$$= \begin{bmatrix} E\left[\left(X_1-\bar{X}_1\right)^2\right] & \cdots & E\left[\left(X_1-\bar{X}_1\right)\left(X_n-\bar{X}_n\right)\right] \\ \vdots & \ddots & \vdots \\ E\left[\left(X_n-\bar{X}_n\right)\left(X_1-\bar{X}_1\right)\right] & \cdots & E\left[\left(X_n-\bar{X}_n\right)^2\right] \end{bmatrix}$$

$$\therefore C_X = \begin{bmatrix} \sigma_1^2 & \cdots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{n1} & \cdots & \sigma_n^2 \end{bmatrix}$$

$$R_{XY} = E\left[XY^T\right] = \begin{bmatrix} E[X_1Y_1] & \cdots & E[X_1Y_m] \\ \vdots & \ddots & \vdots \\ E[X_nY_1] & \cdots & E[X_nY_m] \end{bmatrix} \tag{B.6a}$$

$$R_X = E\left[XX^T\right] = \begin{bmatrix} E[X_1^2] & \cdots & E[X_1X_m] \\ \vdots & \ddots & \vdots \\ E[X_nY_1] & \cdots & E[X_n^2] \end{bmatrix} \tag{B.6b}$$

Cabe destacar que tanto R_X como C_X son siempre simétricas y semidefinidas positivas [151]. Además, por definición, las variables (X, Y) no tienen correlación si $R_{XY} = E(X)E(Y^T)$.

Para el caso de una variable aleatoria multivariable gaussiana de orden n, se define su *pdf* utilizando el determinante de la matriz de autocovariancia C_X junto con $\bar{X}$ y la definición de la ecuación (B.3), tal y como se muestra en la ecuación (B.7).

$$f_X(x) = \frac{1}{(2\pi)^{n/2}|C_X|^{1/2}} \exp\left[-\frac{1}{2}\left(X-\bar{X}\right)^T C_X^{-1}\left(X-\bar{X}\right)\right] \tag{B.7}$$

Finalmente, conviene establecer la evolución en el tiempo de la media y la covarianza del estado x_k de un sistema lineal. La media se obtiene al obtener el valor esperado (ecuación (B.1a)) del modelo del sistema lineal

($x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1}$, ecuación (4.1)) tal y como se muestra en la ecuación (B.8), en donde se considera que w_k tiene media cero.

$$\begin{aligned} x_k &= A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \\ \Rightarrow E(x_k) = \bar{x}_k &= A_{k-1}\bar{x}_{k-1} + B_{k-1}u_{k-1} \end{aligned} \tag{B.8}$$

Con este resultado se puede obtener la evolución de la covarianza de x_k, para lo que primeramente se utiliza (B.8) para obtener (B.9).

$$\begin{aligned} (x_k - \bar{x}_k) &= A_{k-1}(x_{k-1} - \bar{x}_{k-1}) + w_{k-1} \\ \Rightarrow (x_k - \bar{x}_k)(x_k - \bar{x}_k)^T &= A_{k-1}(x_{k-1} - \bar{x}_{k-1})(x_{k-1} - \bar{x}_{k-1})^T A_{k-1}^T \\ &\quad + w_{k-1}w_{k-1}^T + A_{k-1}(x_{k-1} - \bar{x}_{k-1})w_{k-1}^T \\ &\quad + w_{k-1}(x_{k,1} - \bar{x}_{k-1})^T A_{k-1}^T \end{aligned} \tag{B.9}$$

Posteriormente, siguiendo la definición (B.5), se aplica el valor esperado en ambos términos de (B.9) al considerar que el ruido w_k es gaussiano y blanco, por lo que $(x_{k-1} - \bar{x}_{k-1})$ y w_{k-1} no tienen correlación. Esto implica que $E\left[(x_{k-1} - \bar{x}_{k-1})w_{k-1}^T\right] = E\left[(x_{k-1} - \bar{x}_{k-1})\right]E\left[w_{k-1}^T\right] = 0$, al ser w_k de media cero. Con esto se obtiene la covarianza P_k de x_k en (B.10) al considerar la covarianza de w_k como $Q_k = E\left(w_k w_k^T\right)$. Las relaciones básicas definidas son de utilidad en la obtención de los filtros de estimación, principalmente en el caso de los filtros de Kalman.

$$\begin{aligned} P_k &= E\left[(x_k - \bar{x}_k)(x_k - \bar{x}_k)^T\right] \\ &= A_{k-1}P_{k-1}A_{k-1}^T + Q_{k-1} \end{aligned} \tag{B.10}$$

Bibliografía

[1] O. Abumansoor y A. Boukerche. «A Secure Cooperative Approach for Nonline-of-Sight Location Verification in VANET». En: *Vehicular Technology, IEEE Transactions on* 61.1 (2012), págs. 275-285. ISSN: 0018-9545 (vid. pág. 19).

[2] F. Aghili y A. Salerno. «Driftless 3-D Attitude Determination and Positioning of Mobile Robots By Integration of IMU With Two RTK GPSs». En: *Mechatronics, IEEE/ASME Transactions on* 18.1 (feb. de 2013), págs. 21 -31. ISSN: 1083-4435 (vid. pág. 17).

[3] Sanad Al-Areqi y col. «Event-based control and scheduling codesign of networked embedded control systems». En: *American Control Conference (ACC), 2013*. 2013, págs. 5299-5304 (vid. pág. 24).

[4] Abdulgani Albagul, Wahyudi Martono y Riza Muhida. «Dynamic Modelling and Adaptive Traction Control for Mobile Robots». En: *International Journal of Advanced Robotic Systems* 1.3 (2005), págs. 149 -154 (vid. págs. 33, 52).

[5] Yaniv Altshuler y col. «Multi-agent Cooperative Cleaning of Expanding Domains». En: *The International Journal of Robotics Research* 30.8 (ago. de 2011), págs. 1037-1071. DOI: 10.1177/0278364910377245 (vid. pág. 164).

[6] H. Andreasson, A. Treptow y T. Duckett. «Localization for Mobile Robots using Panoramic Vision, Local Features and Particle Filter». En: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. 2005, págs. 3348-3353. DOI: 10.1109/ROBOT.2005.1570627 (vid. pág. 88).

[7] Gianluca Antonelli, Stefano Chiaverini y Giuseppe Fusco. «A calibration method for odometry of mobile robots based on the least-squares technique: theory and experimental validation». En: *IEEE Transactions on Robotics* 21.5 (2005), págs. 994 -1004 (vid. págs. 28, 199).

[8] I. Arasaratnam y Simon Haykin. «Cubature Kalman Filters». En: *Automatic Control, IEEE Transactions on* 54.6 (jun. de 2009), págs. 1254 -1269. ISSN: 0018-9286. DOI: 10.1109/TAC.2009.2019800 (vid. págs. 73, 86).

[9] I. Arasaratnam, Simon Haykin y T.R. Hurd. «Cubature Kalman Filtering for Continuous-Discrete Systems: Theory and Simulations». En: *Signal Processing, IEEE Transactions on* 58.10 (oct. de 2010), págs. 4977 -4993. ISSN: 1053-587X. DOI: 10.1109/TSP.2010.2056923 (vid. págs. 73, 86).

[10] Leopoldo Armesto. «Técnicas de control y fusión sensorial multifrecuenciales y su aplicación a la robótica móvil». Tesis doct. Universidad Politécnica de Valencia, Departamento de Ingeniería de Sistemas y Automática, 2005 (vid. págs. 2, 3, 143).

[11] Leopoldo Armesto y col. «Probabilistic Self-Localization and Mapping - An Asynchronous Multirate Approach». En: *Robotics Automation Magazine, IEEE* 15.2 (2008), págs. 77-88. ISSN: 1070-9932 (vid. págs. 3, 88, 143).

[12] E. Asadi y M. Bozorg. «A Decentralized Architecture for Simultaneous Localization and Mapping». En: *Mechatronics, IEEE/ASME Transactions on* 14.1 (2009), págs. 64-71. ISSN: 1083-4435 (vid. pág. 3).

[13] Karl Aström y Bo Bernhardsson. «Comparison of periodic and event based sampling for firstorder stochastic systems». En: *Proceedings of the 14th IFAC World Congress*. 1999 (vid. pág. 24).

[14] Hazem Ali Attia. «Dynamic model of multi-rigid-body systems based on particle dynamics with recursive approach». En: *Journal of Applied Mathematics* 2005 (2005), págs. 365-382 (vid. págs. 37, 44, 52).

[15] F. Azizi y N Houshangi. «Sensor integration for mobile robot position determination». En: *Systems, Man and Cybernetics, 2003. IEEE International Conference on* 2 (2003), págs. 1136 -1140 (vid. págs. 7, 14).

[16] Alexander Bahr, John J. Leonard y Maurice F. Fallon. «Cooperative Localization for Autonomous Underwater Vehicles». En: *The International Journal of Robotics Research* 28.6 (jun. de 2009), págs. 714-728. DOI: 10.1177/0278364908100561 (vid. pág. 164).

[17] Tim Bailey y H. Durrant-Whyte. «Simultaneous localization and mapping (SLAM): part II». En: *Robotics Automation Magazine, IEEE* 13.3 (2006), págs. 108-117. ISSN: 1070-9932 (vid. pág. 3).

[18] Yaakov Bar-Shalom. «Update with Out-of-Sequence Measurements in Tracking: Exact Solution». En: *IEEE Transactions on Aerospace and Electronic Systems* 38.3 (jul. de 2002), págs. 769-778 (vid. pág. 143).

[19] Steven Bell. *High-Precision Robot Odometry Using an Array of Optical Mice*. Inf. téc. Oklahoma Christian University, 2011 (vid. pág. 306).

[20] F. Bellifemine y col. «JADE: A White Paper». En: *EXP in search of innovation* 3.3 (2003), págs. 6-19 (vid. pág. 169).

[21] Gildas Besançon. «An Overview on Observer Tools for Nonlinear Systems». En: *Nonlinear Observers and Applications*. Vol. 363. Lecture Notes in Control and Information Sciences. Springer, 2007, págs. 1-33. ISBN: 978-3-540-73502-1. DOI: 10.1007/978-3-540-73503-8_1 (vid. págs. 65, 66).

[22] Gildas Besançon, ed. *Nonlinear Observers and Applications*. Lecture Notes in Control and Information Sciences 363. ISBN 978-3-540-73503-8. Springer, 2007 (vid. pág. 64).

[23] M. Betke y L. Gurvits. «Mobile robot localization using landmarks». En: *Robotics and Automation, IEEE Transactions on* 13.2 (1997), págs. 251-263. ISSN: 1042-296X (vid. págs. 3, 173, 175).

[24] D.M. Bevly, J. Ryu y J.C. Gerdes. «Integrating INS Sensors With GPS Measurements for Continuous Estimation of Vehicle Sideslip, Roll, and Tire Cornering Stiffness». En: *Intelligent Transportation Systems, IEEE Transactions on* 7.4 (dic. de 2006), págs. 483 -493. ISSN: 1524-9050 (vid. pág. 17).

[25] David M. Bevly y Bradford Parkinson. «Cascaded Kalman Filters for Accurate Estimation of Multiple Biases, Dead-Reckoning Navigation, and Full State Feedback Control of Ground Vehicles». En: *Control Systems Technology, IEEE Transactions on* 15.2 (mar. de 2007), págs. 199 -208. ISSN: 1063-6536 (vid. pág. 17).

[26] Giacomo Binazzi y col. «Localization of a Swarm of Mobile Agents via Unscented Kalman Filtering». En: *IEEE International Conference on Communications*. 2009, págs. 1-5 (vid. pág. 20).

[27] Mauro Boccadoro, Francesco Martinelli y Stefano Pagnottelli. «Constrained and quantized Kalman filtering for an RFID robot localization problem». En: *Autonomous Robots* 29 (3 2010), págs. 235-251. ISSN: 0929-5593 (vid. pág. 15).

[28] Robert Grover Brown y Patrick Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises and Solutions*. 3rd Edition. John Wiley & Sons, 1997 (vid. págs. 109, 110).

[29] G.C. Calafiore, L. Carlone y Mingzhu Wei. «A Distributed Technique for Localization of Agent Formations From Relative Range Measu-

rements». En: *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 42.5 (2012), págs. 1065-1076. ISSN: 1083-4427 (vid. págs. 22, 168).

[30] G. Campion, G. Bastin y B. Dandrea-Novel. «Structural properties and classification of kinematic and dynamic models of wheeled mobile robots». En: *Robotics and Automation, IEEE Transactions on* 12.1 (feb. de 1996), págs. 47-62 (vid. pág. 27).

[31] Ricardo Carona, A. Pedro Aguiar y José 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 (vid. pág. 33).

[32] Juan Andrade Cetto y Alberto Sanfeliu. *Environment Learning for Indoor Mobile Robots, A Stochastic State Estimation Approach to Simultaneous Localization and Map Building*. Springer, 2006 (vid. pág. 3).

[33] Haoyao Chen y col. «Localization for Multirobot Formations in Indoor Environment». En: *Mechatronics, IEEE/ASME Transactions on* 15.4 (2010), págs. 561-574. ISSN: 1083-4435 (vid. págs. 3, 20, 169).

[34] Rongbao Chen, He Zhao y Benxian Xiao. «Self-localization of mobile robot based on monocular and extended kalman filter». En: *Electronic Measurement Instruments, 2009. ICEMI '09. 9th International Conference on*. 2009, págs. 2-450, 2-454 (vid. págs. 3, 7).

[35] Howie Choset y col. *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, Boston, 2005 (vid. págs. 168, 200).

[36] Charles K. Chui y Guanrong Chen. *Kalman Filtering with Real-Time Applications*. 4th Edition. Springer, 2009 (vid. págs. 20, 23).

[37] Hakyoung Chung, L. Ojeda y J. Borenstein. «Accurate mobile robot dead-reckoning with a precision-calibrated fiber-optic gyroscope». En: *Robotics and Automation, IEEE Transactions on* 17.1 (feb. de 2001), págs. 80 -84 (vid. págs. 7, 14).

[38] M. Cognetti y col. «3-D mutual localization with anonymous bearing measurements». En: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. 2012, págs. 791-798 (vid. pág. 21).

[39] David T. Cole y col. «System Development and Demonstration of a Cooperative UAV Team for Mapping and Tracking». En: *The International Journal of Robotics Research* 29.11 (nov. de 2010), págs. 1371-1399. DOI: 10.1177/0278364910364685 (vid. pág. 164).

[40] Tran Huu Cong, Young Joong Kim y Myo-Taeg 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*. Oct. de 2008, págs. 738 -743 (vid. págs. 3, 7).

[41] Peter Corke, Ron Peterson y Daniela Rus. «Localization and Navigation Assisted by Networked Cooperating Sensors and Robots». En: *The International Journal of Robotics Research* 24.9 (sep. de 2005), págs. 771-786. DOI: 10.1177/0278364905057118 (vid. pág. 164).

[42] R. Craig Coulter. *Implementation of the Pure Pursuit Path Tracking Algorithm*. Inf. téc. Robotics Institute, ene. de 1992 (vid. pág. 200).

[43] Celso De La Cruz y Ricardo Carelli. «Dynamic model based formation control and obstacle avoidance of multi-robot systems». En: *Robotica, Cambridge University Press* 26.3 (2008), págs. 345-356 (vid. págs. 33, 52).

[44] D. Cruz y col. «Decentralized cooperative control - A multivehicle platform for research in networked embedded systems». En: *Con-*

trol Systems, IEEE 27.3 (2007), págs. 58-78. ISSN: 1066-033X (vid. pág. 19).

[45] Ángel Cuenca. «Modelado, Análisis y Diseño de Sistemas de Control con Muestreo no Convencional». Tesis doct. Universidad Politécnica de Valencia, Departamento de Ingeniería de Sistemas y Automática, 2004 (vid. pág. 143).

[46] Colin Das. *Calibration of Kinematic Odometric Parameters for Differential Drive Mobile Robots: An Overview*. Inf. téc. RMSLab, Robotics y neuro-Mechanical Systems Laboratory, University of Arizona, 2010 (vid. págs. 28, 199).

[47] Guilherme N. DeSouza y Avinash C. Kak. «Vision for Mobile Robot Navigation: A Survey». En: *IEEE Transactions on Pattern Analysis and Machine Intelligence, NO. 2,* 24 NO. 2 (2002), págs. 237-267 (vid. págs. 2, 306).

[48] O. Demir y J. Lunze. «Cooperative control of multi-agent systems with event-based communication». En: *American Control Conference (ACC), 2012*. 2012, págs. 4504-4509 (vid. págs. 19, 24, 25, 144).

[49] Y. Dieudonne, O. Labbani-Igbida y F. Petit. «Deterministic Robot-Network Localization is Hard». En: *Robotics, IEEE Transactions on* 26.2 (2010), págs. 331-339. ISSN: 1552-3098 (vid. págs. 3, 22, 173, 175).

[50] M. W M G Dissanayake y col. «A solution to the simultaneous localization and map building (SLAM) problem». En: *Robotics and Automation, IEEE Transactions on* 17.3 (2001), págs. 229-241. ISSN: 1042-296X (vid. págs. 3, 88).

[51] Amaud Doucet, Nando de Freitas y Neil Gordon, eds. *Sequential Monte Carlo Methods in Practice*. ISBN 978-1-4419-2887-0. Springer, 2001. DOI: 10.1007/978-1-4757-3437-9 (vid. pág. 87).

[52] Matthew Dunbabin y col. «Experiments with Cooperative Control of Underwater Robots». En: *The International Journal of Robotics Research* 28.6 (jun. de 2009), págs. 815-833. DOI: 10.1177/0278364908098456 (vid. pág. 164).

[53] H. Durrant-Whyte y Tim Bailey. «Simultaneous localization and mapping: part I». En: *Robotics Automation Magazine, IEEE* 13.2 (2006), págs. 99-110. ISSN: 1070-9932 (vid. págs. 3, 88).

[54] H.F. Durrant-Whyte, B. Y S Rao y H. Hu. «Toward a fully decentralized architecture for multi-sensor data fusion». En: *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on.* 1990, 1331-1336 vol.2 (vid. pág. 18).

[55] P. Enge y P. Misra. «Special Issue on Global Positioning System». En: *Proceedings of the IEEE* 87.1 (1999), págs. 3-15. ISSN: 0018-9219 (vid. págs. 3, 261, 270, 305).

[56] J.S. Esteves, A. Carvalho y C. Couto. «Generalized geometric triangulation algorithm for mobile robot absolute self-localization». En: *Industrial Electronics, 2003. ISIE '03. 2003 IEEE International Symposium on.* Vol. 1. 2003, 346-351 vol. 1. DOI: 10.1109/ISIE.2003.1267272 (vid. pág. 175).

[57] Yuan Fan y col. «Distributed event-triggered control of multi-agent systems with combinational measurements». En: *Automatica* 49.2 (2013), págs. 671 -675. ISSN: 0005-1098 (vid. págs. 24, 145).

[58] Rafael Fierro y col. «A Framework and Architecture for Multi-Robot Coordination». En: *The International Journal of Robotics Research* 21.10-11 (2002), págs. 977-995. DOI: 10.1177/027836490202101098 1 (vid. pág. 169).

[59] A. Franchi, G. Oriolo y P. Stegagno. «Mutual localization in a multi-robot system with anonymous relative position measures». En: *Intelli-*

gent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on. 2009, págs. 3974-3980 (vid. págs. 21, 173).

[60] Antonio Franchi, Giuseppe Oriolo y Paolo Stegagno. «Mutual localization in multi-robot systems using anonymous relative measurements». En: *The International Journal of Robotics Research* 32.11 (sep. de 2013), págs. 1302-1322. DOI: 10.1177/0278364913495425 (vid. pág. 21).

[61] Christoph Fuchs y col. «Indoor tracking for mission critical scenarios: A survey». En: *Pervasive and Mobile Computing* 7.1 (2011), págs. 1 -15 (vid. pág. 3).

[62] Shuzhi Sam Ge y Frank L. Lewis, eds. *Autonomous Mobile Robots: Sensing, Control, Decision Making and Applications*. Taylor & Francis Group, 2006 (vid. pág. 164).

[63] Luis Gracia. «Modelado Cinemático y Control de Robots Móviles con Ruedas». Tesis doct. Universidad Politécnica de Valencia, Departamento de Ingeniería de Sistemas y Automática, 2006 (vid. págs. 33, 303).

[64] Mohinder S. Grewal y Angus P. Andrews. *Kalman Filtering, Theory and Practice Using Matlab*. John Wiley & Sons, 2001 (vid. págs. 4, 67, 109, 110).

[65] H.F. Grip y col. «Nonlinear observer for GNSS-aided inertial navigation with quaternion-based attitude estimation». En: *American Control Conference (ACC), 2013*. 2013, págs. 272-279 (vid. pág. 65).

[66] B. Grocholsky y col. «Cooperative air and ground surveillance». En: *Robotics Automation Magazine, IEEE* 13.3 (2006), págs. 16-25. ISSN: 1070-9932 (vid. pág. 19).

[67] Ramiro Velázquez Guerrero. «Dinámica y Control de Sillas de Ruedas Robóticas, Avances en Ingeniería Electrónica». En: ed. por M. Magos & R. Campos M. Alcaraz. UAM-UdG (México, DF), 2010. Cap. 11, págs. 151-164 (vid. pág. 33).

[68] María Guinaldo y col. «A Mobile Robots Experimental Environment with Event-Based Wireless Communication». En: *Sensors* 13 (2013), págs. 9396-9413 (vid. págs. 24, 25, 148).

[69] S-.B. Han, J-.H. Kim y H. Myung. «Landmark-Based Particle Localization Algorithm for Mobile Robots With a Fish-Eye Vision System». En: *Mechatronics, IEEE/ASME Transactions on* PP.99 (2012), págs. 1-12. ISSN: 1083-4435 (vid. págs. 3, 6, 175).

[70] Jouni Hartikainen y Simo Särkkä. *Optimal filtering with Kalman filters and smoothers, a Manual for Matlab toolbox EKF/UKF*. 1.3. Espoo, Finland. Department of Biomedical Engineering y Computational Science, Helsinki University of Technology. Available: `http://www.lce.hut.fi/research/mm/ekfukf/`, feb. de 2011 (vid. pág. 221).

[71] W. P. M. H. Heemels, J. H. Sandee y P. P. J. Van Den Bosch. «Analysis of event-driven controllers for linear systems». En: *International Journal of Control* 81.4 (2008), págs. 571-590 (vid. pág. 24).

[72] Anne-Kathrin Hess y Anders Rantzer. «Distributed Kalman Filter algorithms for self-localization of mobile devices». En: *Proceedings of the 13th ACM international conference on Hybrid systems: computation and control*. HSCC '10. Stockholm, Sweden: ACM, 2010, págs. 191 -200. ISBN: 978-1-60558-955-8 (vid. pág. 21).

[73] Robert W. Hogg y col. «Algorithms and Sensors for Small Robot Path Following». En: *IEEE International Conference on Robotics and Automation, Washington D.C.* 2002 (vid. págs. 7, 17, 168).

[74] Dominik Honegger y col. «An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications». En: *ICRA*. 2013, págs. 1736-1741 (vid. pág. 306).

[75] Nasser Houshangi y Farouk Azizi. «Accurate mobile robot position determination using unscented Kalman filter». En: *Canadian Conference on Electrical and Computer Engineering*. 2005, págs. 846 -851 (vid. págs. 4, 14).

[76] Nasser Houshangi y Farouk Azizi. «Mobile Robot Position Determination Using Data Integration of Odometry and Gyroscope». En: *World Automation Congress (WAC), Budapest, Hungary*. 2006 (vid. págs. 7, 14).

[77] Nasser Houshangi y Farouk Azizi. «Mobile robot position determination using data from gyro and odometry». En: *Electrical and Computer Engineering, 2004. Canadian Conference on* 2 (2004), págs. 719-722 (vid. pág. 14).

[78] Songlin Hu y Dong Yue. «Event-based H_∞ filtering for networked system with communication delay». En: *Signal Processing* 92.9 (2012), págs. 2029 -2039. ISSN: 0165-1684 (vid. págs. 24, 145).

[79] Dongjun Hyun y col. «Dead-reckoning sensor system and tracking algorithm for 3-D pipeline mapping». En: *Mechatronics* 20.2 (2010), págs. 213 -223 (vid. págs. 7, 15).

[80] *IG-500N GPS, Manual de Usuario, Revisión 19 Noviembre, 2009, SBG Systems,* `http://www.sbg-systems.com/products/ig500-oem-inertial-systems` (vid. pág. 211).

[81] Lars Imsland y col. «Nonlinear Observer for Vehicle Velocity with Friction and Road Bank Angle Adaptation - Validation and Comparison with an Extended Kalman Filter». En: *SAE Technical Paper 2007-01-0808*. 2007. DOI: 10.4271/2007-01-0808 (vid. págs. 65, 90).

[82] Lars Imsland y col. «Vehicle velocity estimation using nonlinear observers». En: *Automatica* 42.12 (2006), págs. 2091 -2103. ISSN: 0005-1098. DOI: http://dx.doi.org/10.1016/j.automatica.2006.06.025 (vid. págs. 64, 65).

[83] *Información en línea de la IGEP (ISEE) Disponible en:* *https://www.isee.biz/products/igep-processor-boards/igepv2-dm3730* (vid. pág. 211).

[84] *Información en línea del Firmware LeJOS (2014), Disponible en:* *http://www.lejos.org/*. (Vid. pág. 194).

[85] *Información en línea del LEGO NXT (2014), Disponible en:* *http://www.mindsensors.com/http://www.hitechnic.com/*, *Descripción motores:* *http://www.philohome.com/motors/motorcomp.htm*. (Vid. págs. 193, 194).

[86] Leopoldo Jetto, Sauro Longhi y Giuseppe Venturini. «Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots». En: *IEEE Transactions on Robotics and Automation* 15.2 (1999), págs. 219 -229 (vid. pág. 3).

[87] S. Julier, J. Uhlmann y H.F Durrant-Whyte. «A new method for the nonlinear transformation of means and covariances in filters and estimators». En: *Automatic Control, IEEE Transactions on* 45 (mar. de 2000), págs. 477 -482 (vid. págs. 4, 73, 79, 80).

[88] S.J. Julier y J.K. Uhlmann. *A General Method for Approximating Nonlinear Transformations of Probability Distributions*. Inf. téc. Department of Engineering Science, University of Oxford, 1996 (vid. pág. 79).

[89] S.J. Julier y J.K. Uhlmann. «A new extension of the Kalman filter to nonlinear systems». En: *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls, Orlando, FL*. 1997 (vid. pág. 79).

[90] S.J. Julier y J.K. Uhlmann. «Unscented filtering and nonlinear estimation». En: *Proceedings of the IEEE* 92 (2004), págs. 401 -422 (vid. págs. 73, 79, 82, 83).

[91] Rudolph Emil Kalman. «A New Approach to Linear Filtering and Prediction Problems». En: *Transactions of the ASME-Journal of Basic Engineering* 82 (1960), págs. 35-45 (vid. pág. 67).

[92] Jungmin Kim, Yountae Kim y Sungshin 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*. Jun. de 2008, págs. 2928 -2933 (vid. págs. 7, 15).

[93] S. Kim y K. Byung Kook. «Dynamic Ultrasonic Hybrid Localization System for Indoor Mobile Robots». En: *Industrial Electronics, IEEE Transactions on* in Press (PP).99 (2012), pág. 1. ISSN: 0278-0046 (vid. págs. 2, 15).

[94] Sungbok Kim y Sanghyup Lee. «Optical Mouse Array Position Calibration for Mobile Robot Velocity Estimation». En: *Robotics, Automation and Mechatronics, 2008 IEEE Conference on*. 2008, págs. 1167 -1172 (vid. pág. 306).

[95] Evgeni Kiriy y Martin Buehler. *Three-state Extended Kalman Filter for Mobile Robot Localization*. Inf. téc. TR-CIM 05.06. Montreal, Canada: McGill University, abr. de 2002 (vid. pág. 3).

[96] Sang il Ko, Jong suk Choi y Byoung hoon Kim. «Performance Enhancement of Indoor Mobile Localization System using Unscented Kalman Filter». En: *SICE-ICASE, 2006. International Joint Conference, Bexco, Busan, Korea*. 2006, págs. 1355 -1360 (vid. pág. 15).

[97] Jayesh H. Kotecha y Petar M. Djuric. «Gaussian particle filtering». En: *Signal Processing, IEEE Transactions on* 51.10 (oct. de 2003), págs. 2592 -2601 (vid. págs. 6, 20, 73, 87, 91, 92).

[98] Krzysztof Kozlowski, ed. *Robot Motion and Control, Recent Developments (Lecture Notes in Control and Information Sciences)*. Springer-Verlag London, 2006 (vid. págs. 168, 200).

[99] James Kramer y Matthias Scheutz. «Development environments for autonomous mobile robots: A survey». En: *Autonomous Robots* 22 (2 2007). 10.1007/s10514-006-9013-8, págs. 101-132. ISSN: 0929-5593 (vid. pág. 193).

[100] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006 (vid. pág. 168).

[101] Jung-Min Lee y col. «Ultrasonic satellite system for the positioning of mobile robots». En: *Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE* 1 (2004), págs. 448 -453 (vid. pág. 15).

[102] Olivier Lefebvre y col. «Obstacles Avoidance for Car-Like Robots Integration And Experimentation on Two Robots». En: *IEEE Int. Conf. on Robotics and Automation*. 2004, págs. 4277-4282 (vid. pág. 168).

[103] K.Y.K. Leung, T.D. Barfoot y H.H.T. Liu. «Decentralized Localization of Sparsely-Communicating Robot Networks: A Centralized-Equivalent Approach». En: *Robotics, IEEE Transactions on* 26.1 (2010), págs. 62-77. ISSN: 1552-3098 (vid. págs. 23, 165).

[104] Martin E. Liggins, David L. Hall y James Llinas, eds. *Handbook of Multisensor Data Fusion: Theory and Practice*. Second Edition. CRC Press, Taylor & Francis Group, 2009 (vid. págs. 20, 23, 73).

[105] S. Liu y D. Sun. «Minimizing Energy Consumption of Wheeled Mobile Robots via Optimal Motion Planning». En: *Mechatronics, IEEE/ASME Transactions on* PP (2013), págs. 1-11. ISSN: 1083-4435 (vid. pág. 7).

[106] C. Losada y col. «Multi-Camera Sensor System for 3D Segmentation and Localization of Multiple Mobile Robots». En: *Sensors* 10 (2010), págs. 3261-3279 (vid. pág. 3).

[107] V.J. Lumelsky y A. A. Stepanov. «Path Planning Strategies for a Point Mobile Automaton Moving Amidst Unknown Obstacles of Arbitrary Shape». En: *Algorithmica* 2 (1987), págs. 403-430 (vid. pág. 168).

[108] Martin Lundgren. «Path Tracking and Obstacle Avoidance for a Miniature Robot». Tesis de lic. Department of Computer Science, Umeå University, Sweden, 2003 (vid. pág. 168).

[109] Jan Lunze y Daniel Lehmann. «A state-feedback approach to event-based control». En: *Automatica* 46.1 (2010), págs. 211 -215. ISSN: 0005-1098 (vid. pág. 24).

[110] Xue Luo y S.S.-T. Yau. «Complete Real Time Solution of the General Nonlinear Filtering Problem Without Memory». En: *Automatic Control, IEEE Transactions on* 58.10 (oct. de 2013), págs. 2563-2578. ISSN: 0018-9286. DOI: 10.1109/TAC.2013.2264552 (vid. pág. 73).

[111] Kristijan Macek, Ivan Petrovic y Roland Siegwart. «A control method for stable and smooth path following of mobile robots». En: *2nd European Conference on Mobile Robots*. 2005 (vid. pág. 168).

[112] R. Madhavan y H.F. Durrant-Whyte. «Natural landmark-based autonomous vehicle navigation». En: *Robotics and Autonomous Systems* 46.2 (2004), págs. 79 -95. ISSN: 0921-8890 (vid. pág. 173).

[113] R. Madhavan, K. Fregene y L.E. Parker. «Distributed heterogeneous outdoor multi-robot localization». En: *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*. Vol. 1. 2002, 374-381 vol.1 (vid. pág. 20).

[114] Raj Madhavan, Kingsley Fregene y LynneE. Parker. «Distributed Cooperative Outdoor Multirobot Localization and Mapping». English. En: *Autonomous Robots* 17 (1 2004), págs. 23-39. ISSN: 0929-5593 (vid. págs. 16, 18, 20).

[115] Andras Majdik y col. «New approach in solving the kidnapped robot problem». En: *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*. 2010, págs. 1-6 (vid. pág. 88).

[116] Guoqiang Mao, Baris Fidan y Brian D.O. Anderson. «Wireless sensor network localization techniques». En: *Computer Networks* 51.10 (2007), págs. 2529 -2553. ISSN: 1389-1286 (vid. págs. 18, 173).

[117] A. Martinelli, F. Pont y R. Siegwart. «Multi-Robot Localization Using Relative Observations». En: *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. 2005, págs. 2797-2802 (vid. págs. 20, 164).

[118] Agostino Martinelli y Roland Siegwart. «Estimating the odometry error of a mobile robot during navigation». En: *European Conference on Mobile Robots (ECMR 2003)*. 2003 (vid. págs. 7, 14).

[119] Xiangyu Meng y Tongwen Chen. «Event based agreement protocols for multi-agent networks». En: *Automatica* 49.7 (2013), págs. 2125 -2132. ISSN: 0005-1098 (vid. págs. 19, 24, 145).

[120] Xiangyu Meng y Tongwen Chen. «Event-driven communication for sampled-data control systems». En: *American Control Conference (ACC), 2013*. 2013, págs. 3002-3007 (vid. pág. 24).

[121] Olivier Michel y col. *Cyberbotics' Robot Curriculum*. Dispobible: `http://en.wikibooks.org/wiki/Cyberbotics%27_Robot_Curriculum`: Cyberbotics Ltd. y Wikibooks contributors, 2012 (vid. págs. 1, 2, 32, 199, 200, 222, 228).

[122] C. Mitsantisuk, S. Katsura y K. Ohishi. «Kalman-Filter-Based Sensor Integration of Variable Power Assist Control Based on Human Stiffness Estimation». En: *Industrial Electronics, IEEE Transactions on* 56.10 (oct. de 2009), págs. 3897 -3905. ISSN: 0278-0046 (vid. pág. 4).

[123] F. Morbidi y G. L. Mariottini. «Active Target Tracking and Cooperative Localization for Teams of Aerial Vehicles». En: *Control Systems Technology, IEEE Transactions on* PP.99 (2012), págs. 1-1. ISSN: 1063-6536 (vid. pág. 19).

[124] Victor M. Moreno y Alberto Pigazo, eds. *Kalman Filter: Recent Advances and Applications*. In-Teh, 2009 (vid. págs. 20, 73).

[125] A.I. Mourikis y S.I. Roumeliotis. «Optimal sensor scheduling for resource-constrained localization of mobile robot formations». En: *Robotics, IEEE Transactions on* 22.5 (2006), págs. 917-931. ISSN: 1552-3098 (vid. págs. 20, 165).

[126] Anastasios I. Mourikis y Stergios I. Roumeliotis. «Predicting the Performance of Cooperative Simultaneous Localization and Mapping (C-SLAM)». En: *The International Journal of Robotics Research* 25.12 (dic. de 2006), págs. 1273-1286. DOI: 10.1177/0278364906072515 (vid. pág. 169).

[127] Hyun Myung y col. «Constrained Kialman Filter for Mobile Robot Localization with Gyroscope». En: *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems Beijing, China*. 2006, págs. 442 -447 (vid. pág. 14).

[128] Keiji Nagatani, Daisuke Endo y Kazuya Yoshida. «Improvement of the Odometry Accuracy of a Crawler Vehicle with Consideration of Slippage». En: *IEEE International Conference on Robotics and Automation*. 2007, págs. 2752-2757 (vid. pág. 14).

[129] Danilo Navarro. «Contribución a la autolocalización de robots móviles basada en la fusión de información multisensorial». Tesis doct. Universitat Politècnica de València. Departamento de Informática de Sistemas y Computadores, 2009 (vid. págs. 2, 3).

[130] S. Noga. «Kinematics and dynamics of some selected two-wheeled mobile robots». En: *Archives of Civil and Mechanical Engineering* 6 (2006), págs. 55-70 (vid. págs. 33, 52).

[131] Kumar Pakki y col. «Cubature Kalman Filter based Localization and Mapping». En: *Proceedings of the 18th IFAC World Congress*. Vol. 18. 1. Università Cattolica del Sacro Cuore, Milano, Italy, 2011. DOI: 10.3182/20110828-6-IT-1002.02909 (vid. pág. 86).

[132] Kumar Pakki y col. «Fusion of an Extended $H_i nfty$ Filter and Cubature Kalman Filter». En: *Proceedings of the 18th IFAC World Congress*. Vol. 18. 1. Università Cattolica del Sacro Cuore, Milano, Italy, 2011, págs. 9091-9096. DOI: 10.3182/20110828-6-IT-1002.02909 (vid. págs. 75, 86).

[133] K. B. Petersen y M. S. Pedersen. *The Matrix Cookbook*. Technical University of Denmark. Nov. de 2012. URL: http://www2.imm.dtu.dk/pubdb/p.php?3274 (vid. pág. 114).

[134] V. Pierlot y M. Van Droogenbroeck. «A New Three Object Triangulation Algorithm for Mobile Robot Positioning». En: *Robotics, IEEE Transactions on* PP.99 (2014), págs. 1-12. ISSN: 1552-3098 (vid. pág. 175).

[135] *Pioneer Online information*. 2012. URL: http://www.mobilerobots.com/ResearchRobots/PioneerP3DX.aspx (vid. pág. 14).

[136] R. Piza y col. «Kalman filtering applied to Profibus-DP systems. Multirate control systems with delayed signals». En: *Industrial Electronics, 2008. IECON 2008. 34th Annual Conference of IEEE*. 2008, págs. 2905-2910 (vid. pág. 143).

[137] A. Prorok, A. Bahr y A. Martinoli. «Low-cost collaborative localization for large-scale multi-robot systems». En: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. 2012, págs. 4236 -4241 (vid. págs. 8, 24, 165, 173).

[138] J. Pugh y col. «A Fast Onboard Relative Positioning Module for Multirobot Systems». En: *Mechatronics, IEEE/ASME Transactions on* 14.2 (2009), págs. 151-162. ISSN: 1083-4435 (vid. págs. 24, 173).

[139] R. Rajamani. *Vehicle Dynamics And Control*. Mechanical Engineering Series. Springer US, 2012. ISBN: 9780387263960 (vid. págs. 27, 49, 57).

[140] B.S. Rao y H.F. Durrant-Whyte. «Fully decentralised algorithm for multisensor Kalman filtering». En: *Control Theory and Applications, IEE Proceedings D* 138.5 (1991), págs. 413-420. ISSN: 0143-7054 (vid. pág. 18).

[141] G. Reina y col. «Adaptive Kalman Filtering for GPS-based Mobile Robot Localization». En: *Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on*. 2007, págs. 1 -6 (vid. págs. 7, 18).

[142] Ioannis M. Rekleitis, Gregory Dudek y Evangelos Milios. «Multi-Robot Collaboration for Robust Exploration». En: *Annals of Mathematics and Artificial Intelligence* 31.1-4 (2001), págs. 7-40 (vid. pág. 164).

[143] Ioannis Rekleitis y col. «Efficient Boustrophedon Multi-Robot Coverage: an algorithmic approach». En: *Annals of Mathematics and Artificial Intelligence* 52.2-4 (abr. de 2008), págs. 109-142 (vid. pág. 164).

[144] Jonathan A. Rogge y Dirk Aeyels. «Multi-Robot Coverage to locate fixed and moving targets». En: *IEEE International Conference on Control Applications (CCA)*. 2009, págs. 902 -907 (vid. pág. 164).

[145] S.I. Roumeliotis y George A. Bekey. «Distributed multirobot localization». En: *Robotics and Automation, IEEE Transactions on* 18.5 (2002), págs. 781-795. ISSN: 1042-296X (vid. págs. 7, 20, 165, 167, 272, 284, 292, 294).

[146] Stergios I. Roumeliotis y Ioannis M. Rekleitis. «Propagation of Uncertainty in Cooperative Multirobot Localization: Analysis and Experimental Results». En: *Autonomous Robots* 17.1 (jun. de 2004), págs. 41-54 (vid. págs. 163, 189).

[147] Nilanjan Sarkar, Xiaoping Yun y R. Vijay Kumar. *Control of Mechanical Systems with Rolling Constraints: Application To Dynamic Control of Mobile Robots*. Inf. téc. University of Pennsylvania, 1992 (vid. pág. 33).

[148] Georg S. Seyboth, Dimos V. Dimarogonas y Karl H. Johansson. «Event-based broadcasting for multi-agent average consensus». En: *Automatica* 49.1 (2013), págs. 245 -252. ISSN: 0005-1098 (vid. págs. 24, 25, 148, 169).

[149] Zhi Shen y col. «Low cost two dimension navigation using an augmented Kalman filter Fast Orthogonal Search module for the integration of reduced inertial sensor system and Global Positioning System». En: *Transportation Research Part C: Emerging Technologies* 19.6 (2011), págs. 1111 -1132. ISSN: 0968-090X (vid. pág. 17).

[150] R. Siegwart e I. Nourbakhsh. *Introduction to autonomous mobile robots*. Cambridge, Massachusetts: The MIT Press, 2004 (vid. págs. 1, 2, 4, 32, 88, 168, 200).

[151] Dan Simon. *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*. John Wiley & Sons, 2006 (vid. págs. 4, 20, 23, 61, 63, 66-74, 76, 77, 79-83, 87, 109, 110, 143, 150, 317, 319, 320).

[152] B. Sinopoli y col. «Kalman filtering with intermittent observations». En: *Automatic Control, IEEE Transactions on* 49.9 (2004), págs. 1453 -1464. ISSN: 0018-9286 (vid. pág. 145).

[153] I. Skog y P. Handel. «In-Car Positioning and Navigation Technologies - A Survey». En: *Intelligent Transportation Systems, IEEE Transactions on* 10.1 (mar. de 2009), págs. 4 -21. ISSN: 1524-9050 (vid. págs. 3, 4, 7, 16, 261, 270, 305).

[154] Joan Solà. *Simulataneous localization and mapping with the extended Kalman filter*. Inf. téc. The Laboratory of Analysis y Architecture of Systems,Toulouse, France., ene. de 2013 (vid. págs. 150, 151).

[155] Joan Solà. «Towards Visual Localization, Mapping and Moving Objects Tracking by a Mobile Robot: a Geometric and Probabilistic Approach.» Tesis doct. Institut National Polytechnique de Toulouse, 2007 (vid. págs. 150, 151).

[156] B. Sridhar y col. «Multirate and event-driven Kalman filters for helicopter flight». En: *Control Systems, IEEE* 13.4 (1993), págs. 26-33. ISSN: 1066-033X (vid. págs. 143, 145).

[157] S. Thrun. «Particle Filters in Robotics». En: *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI), Aug 1-4 2002, Alberta,Canada*. Ago. de 2002 (vid. págs. 87-89).

[158] N. Trawny, S.I. Roumeliotis y G.B. Giannakis. «Cooperative multi-robot localization under communication constraints». En: *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*. 2009, págs. 4394-4400 (vid. págs. 23, 165).

[159] Ángel Valera y col. «Bluetooth-Networked Trajectory Control of Autonomous Vehicles». En: *Eight IFAC Symposium on Cost Oriented Automation*. Vol. 8. 2007 (vid. págs. 4, 8, 16, 23).

[160] Marina Vallés. «Implementación de Sistemas de Control Híbrido con Recursos Limitados de Computación». Tesis doct. Universidad Politécnica de Valencia, Departamento de Ingeniería de Sistemas y Automática, 2004 (vid. pág. 7).

[161] Martin Velasco Villa, Eduardo Aranda Bricaire y Rodolfo Orosco Guerrero. «Discrete-Time Modeling and Path-Tracking for a Wheeled Mobile Robot». En: *Computación y Sistemas* 13.2 (2009), págs. 142-160 (vid. pág. 28).

[162] E.A. Wan 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 (vid. págs. 79, 81).

[163] Zongyao Wang y Dongbing Gu. «Cooperative Target Tracking Control of Multiple Robots». En: *Industrial Electronics, IEEE Transactions on* 59.8 (2012), págs. 3232-3240. ISSN: 0278-0046 (vid. pág. 19).

[164] Chris C. Ward y Karl Iagnemma. «A Dynamic-Model-Based Wheel Slip Detector for Mobile Robots on Outdoor Terrain». En: *IEEE Transactions on Robotics* 24.4 (2008), págs. 821 -831 (vid. págs. 33, 52).

[165] Greg Welch y Gary Bishop. *An Introduction to the Kalman Filter*. University of North Carolina at Chapel Hill. `http://www.cs.unc.edu/~welch/kalman`, mar. de 2007 (vid. págs. 4, 66, 67, 76).

[166] Glenn D. White, Rajankumar M. Bhatt y Venkat N. Krovi. «Dynamic redundancy resolution in a nonholonomic wheeled mobile manipulator». En: *Robotica, Cambridge University Press* 25.2 (2007), págs. 147-156 (vid. págs. 33, 164).

[167] S.-h.P. Won, F. Golnaraghi y W.W. Melek. «A Fastening Tool Tracking System Using an IMU and a Position Sensor With Kalman Filters and a Fuzzy Expert System». En: *Industrial Electronics, IEEE Tran-*

sactions on 56.5 (mayo de 2009), págs. 1782 -1792. ISSN: 0278-0046 (vid. pág. 4).

[168] J.Y. Wong. *Theory of Ground Vehicles*. John Wiley & Sons, 2008. ISBN: 9780470170380 (vid. págs. 17, 49-52, 57).

[169] Yunchun Yang y J.A. Farrell. «Magnetometer and differential carrier phase GPS-aided INS for advanced vehicle control». En: *Robotics and Automation, IEEE Transactions on* 19.2 (abr. de 2003), págs. 269 -282 (vid. pág. 17).

[170] Jingang Yi y col. «Kinematic Modeling and Analysis of Skid-Steered Mobile Robots With Applications to Low-Cost Inertial-Measurement-Unit-Based Motion Estimation». En: *Robotics, IEEE Transactions on* 25.5 (oct. de 2009), págs. 1087 -1097 (vid. págs. 7, 15).

[171] P. Zarchan y H. Musoff. *Fundamentals of Kalman Filtering: A Practical Approach*. EngineeringPro collection v. 208. American Institute of Aeronautics y Astronautics, 2005. ISBN: 9781600864582 (vid. págs. 109, 110).

[172] Tao Zhang y Xiaosu Xu. «A new method of seamless land navigation for GPS/INS integrated system». En: *Measurement* 45.4 (2012), págs. 691 -701. ISSN: 0263-2241 (vid. pág. 17).

[173] Bo Zhou, Yan Peng y Jianda Han. «UKF Based Estimation and Tracking Control of Nonholonomic Mobile Robots with Slipping». En: *Proceedings of the 2007 IEEE International Conference on Robotics and Biomimetics*. 2007, págs. 2058 -2063 (vid. pág. 14).

[174] Ilan Zohar, Amit Ailon y Raul Rabinovici. «Mobile robot characterized by dynamic and kinematic equations and actuator dynamics: Trajectory tracking and related application». En: *Robotics and Autonomous Systems* 59.6 (2011), págs. 343 -353 (vid. págs. 33, 52).