APLICACIONES DE LA LÓGICA BORROSA EN EL EQUIPO DE FÚTBOL ROBÓTICO TeamChaos

Francisco Martín[1] **Carlos E. Agüero**[1] **José M. Cañas**[1] **Pablo Barrera**[1] **Vicente Matellán**[2]

[1] Depto. de Sistemas Telemáticos y Computación. Universidad Rey Juan Carlos, {fmartin, caguero, barrera, jmplaza}@gsyc.es
[2] Depto. de Ingenierías Mecánica, Informática y Aeroespacial. Universidad de León, vicente.matellan@unileon.es

Resumen

El equipo TeamChaos ha participado en las últimas ediciones de la RoboCup en la categoría de robots con 4 patas. Varios de los subsistemas del software que controla los cuatro robots del equipo utilizan la lógica borrosa. En concreto en este artículo describimos el sistema de auto-localización de los robots que se basa en una mezcla de un sistema basado en una rejilla uniforme borrosa y una población de tamaño variable de filtros extendidos de Kalman.

Palabras Clave: Localización, robot, robocup, lógica borrosa.

1. INTRODUCCIÓN

Una de las habilidades más básicas que ha de ser capaz de exhibir un robot móvil es la de auto-localización [4]. Puede definirse como la capacidad de un robot de determinar su posición usando sus propios sensores. Esa posición puede calcularse sobre un mapa ya compilado del entorno, o determinarse mientras se construye el mapa (SLAM).

Las técnicas clásicas que se ha propuesto para resolver este problema varían enormemente dependiendo de los sensores disponibles en cada tipo de robot, del entorno y de la aplicación. Las más exitosas, consideradas ahora mismo estándar en la literatura, son las aproximaciones probabilísticas. Así por ejemplo, en [9, 8, 6, 5, 2] se han propuesto soluciones a la localización de robots móviles con ruedas equipados con sensores de ultrasonidos y/o láser en entornos de interiores no muy dinámicos.

La propuesta presentada en este artículo también utiliza esta misma aproximación. La diferencia principal de estos trabajos con nuestra propuesta es el tipo de robot al que está dirigida y el entorno. En nuestro caso robots se trata de robots con patas, donde la información odométrica es muy poco fiable y que no disponen de información instantánea de todo el entorno del robot (de los 360 grados alrededor), mientras que las aproximaciones anteriores están realizadas para robots con odometría muy precisa y una abundante información del entorno de 360º.

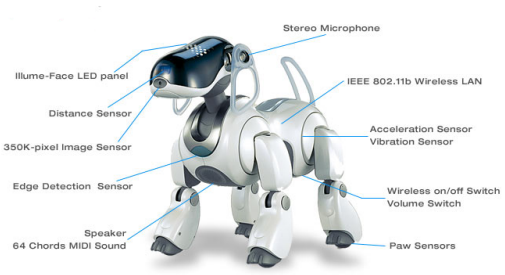

Figura 1: Sony aiBo ERS7

En concreto, en este trabajo utilizaremos el robot AIBO (figura 1). Este robot tiene como sensor principal una cámara situada en su cabeza. Las imágenes que se obtienen de la cámara han de ser procesadas para obtener información de ella. Otra característica de este robot es la de tener como principales actuadores de locomoción cuatro patas, lo que hace difícil obtener una información odométrica precisa.

Este modelo de robot se usa en la competición de la RoboCup[1] en la categoría de cuatro patas[2], que es como indicábamos la otra característica diferenciadora, el dinamismo y los requisitos de vivacidad de la competición. El campo de juego de esta categoría es el que refleja la figura 2, donde se puede apreciar que hay elementos fácilmente distinguibles por color, como las porterías, las balizas y las líneas del campo.

[1] http://www.robocup.org
[2] http://www.tzi.de/4legged/bin/view/Website/WebHome

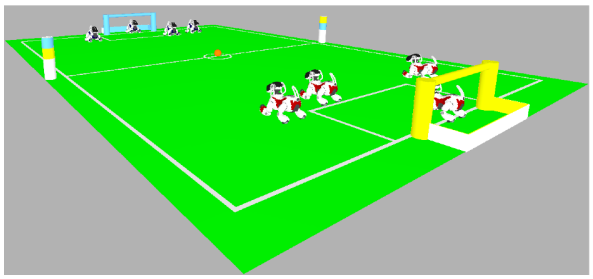

Figura 2: Campo de la RoboCup de la liga de 4 patas.

En este entorno es importante que los robots estén localizados en todo momento para que su comportamiento sea coherente: los robots no deben de salirse de los límites del campo; en determinadas situaciones deben situarse en una posición inicial conocida, como por ejemplo, al iniciar el partido o después de cada gol; también deben saber hacia dónde han de dirigir la pelota; en que parte del campo se encuentran para poder generar una estrategia común entre los miembros de un equipo, etc.

Los métodos de localización usados por los demás equipos son muy variados. La mayor parte de los equipos usan métodos basados en Filtros de Partículas. Nuestro equipo[3] ha usado tradicionalmente un método basado en lógica difusa que proporcionaba una localización global satisfactoria en el campo orignl[1], pero que resultaba demasiado costoso en el campo ampliado que se usa en las últimas competiciones. En este artículo describiremos los trabajos para mejorar esta localización.

El resto de este artículo se organiza de la siguiente forma. En la segunda sección presentamos los fundamentos de la localización probabilística borrosa que se ha venido utilizando en el equipo TeamChaos. En la tercera presentamos una posible mejora basada en la utilización de filtros de Kalman para mejorar el funcionamiento local. En la cuarta presentamos algunos resultados y en la quinta resumimos nuestra opinión.

2. Localización probabilística borrosa

El método que se ha venido utilizando hasta la fecha se ha denominado Fuzzy-Markov, o abreviadamente FMK. Una descripción detallada del mismo se puede encontrar en [3]. Básicamente, consiste en dividir el campo de juego en una cuadrícula G_t tal que $G_t(x, y)$ representa la probabilidad ($[0, 1]$) de que el robot se encuentre en la posición (x, y). Cada una de las celdas, que definiremos en adelante como *fcell*, contiene información sobre la probabilidad de que el robot esté en esa celda, e información sobre cual es el rango de orientaciones más probables para el robot, es decir, se trata realmente de una cuadrícula de $2\frac{1}{2}$D

Cada *fcell* se representa por medio de un trapezoide difuso. En la figura 3 podemos ver este trapezoide, definido por la tupla:

$$< \theta, \Delta, \alpha, h, b >$$

Intuitivamente, si h es bajo, la probabilidad de estar en esta celda es baja. Si h es alto, es muy probable que el robot esté en esta posición. Si el trapezoide es ancho, existe gran incertidumbre sobre la orientación del robot. Si el trapezoide es estrecho, o tiene incluso forma de triángulo (porque Δ es prácticamente nulo), la incertidumbre de orientación es tan baja que podemos afirmar que es θ.

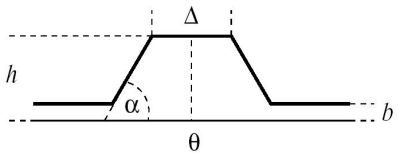

Figura 3: *fcell* representando el ángulo θ

El proceso de localización usado en este método es iterativo, teniendo cada ciclo un paso de predicción y otro de actualización. La fase de predicción se realiza cada vez que se realiza un movimiento, lo que hace que la probabilidad se distribuya (difumine) en la dirección del movimiento. En la fase de actualización se incorpora la información visual. Cada observación de una marca visual se compone de una distancia y un ángulo a cada una de las marcas visibles. Para codificar la información de una marca visual conocida en el instante t, construimos la distribución de probabilidad $S_t(\cdot|r)$, tal que $S_t(x, y|r)$ es la posibilidad de que el robot se encuentre en la posición (x, y), siendo la distancia a una marca visual determinada r. Para cada observación, actualizamos G_t como se muestra en la ecuación 1.

$$G_t(x, y) = G_t(x, y) \times S_t(x, y|r) \qquad (1)$$

donde $\times$ es un operador de intersección difuso.

Un ejemplo de la aplicación secuencial de estas dos operaciones para la localización de un robot se puede observar en la figura 4. El robot parte de un estado de total incertidumbre, y mediante la información odométrica y la información de la posición relativa de la portería y de la baliza superior derecha, termina localizándose correctamente.

[3] http://www.teamchaos.es

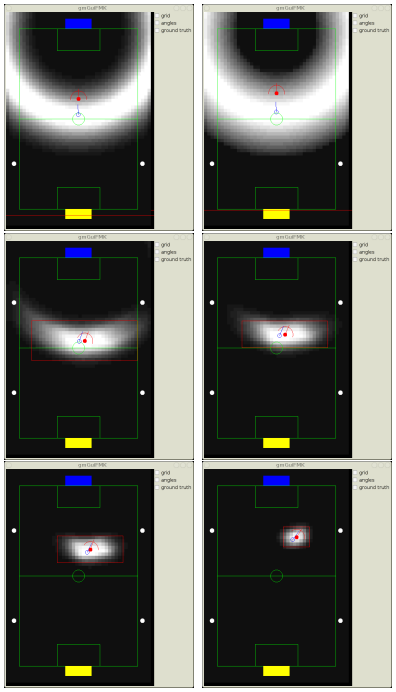

Figura 4: Proceso de localización del robot mediante grid borroso..

El proceso anterior calcula la probabilidad de estar en una posición de la cuadrícula, pero no aporta información angular. Podría ser natural considerar un cubo 3D $G_t(x, y, \theta)$, pero el tiempo de computación de todas las posibles posiciones del robot haría que el algoritmo no fuera abordable computacionalmente. En lugar de eso se mantiene una cuadrícula 2D de *fcell*, que es una forma compacta de representar información angular. Con el procedimiento descrito anteriormente se obtiene la componente h de esta *fcell*, pero aún es necesario describir como obtener el resto de las componentes cuando se produce una observación. Esta operación es muy ligera y calcula las nuevas probabilidades de orientación a partir de una estimación anterior y una nueva observación.

Este algoritmo es un método global, es decir, se almacena la probabilidad de que el robot esté en cada cuadrícula y orientación. El tamaño de cada celda es de 100 mm de lado, lo que significa que para el campo de juego original, cuyas medida eran 2700×4200 mm, se necesitaban 1134 celdas. Para una división típica en rejilla que almacenara 16 posibles orientaciones por cada celda, que es lo mínimo para tener una precisión en cuanto a la orientación aceptable, supondría tener 18144 estados.

El campo ha aumentado sus dimensiones. Las nuevas dimensiones, 2700×4200 aumentan el número de celdas a más del doble, 2400. El aumento de coste compu-

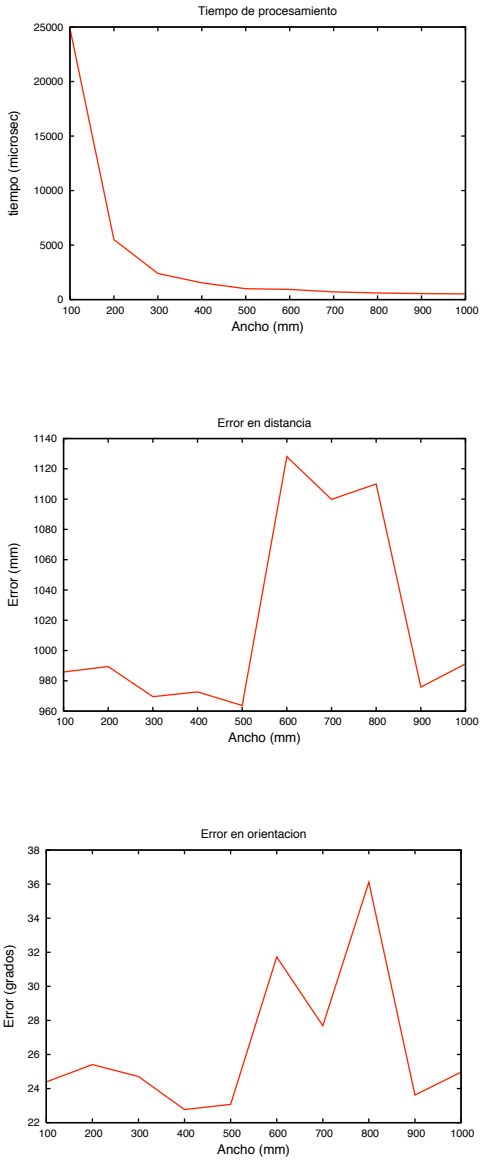

Figura 5: Variación del tiempo de computación (izquierda), error en la estimación de x, y (centro) y error en estimación θ dependiendo del tamaño de la celda.

tacional, al ser FMK un sistema $O(n^2)$, ha hecho que el tiempo de cálculo del sistema de localización impida que el sistema completo de control del robot pueda ejecutarse en un solo ciclo de control, que corresponde al tiempo entre que se procesa una imagen y la siguiente. La solución inicial consistió en aumentar el tamaño de la celda, pero esto supone que en el mejor de los casos el sistema tiene una precisión menor, lo cual resultaba inaceptable.

El tiempo de proceso de los algoritmos de localización es crítica. Hay que tener en cuenta que el robot está ejecutando múltiples módulos (coordinación, asignación de roles, visión, cálculo de movimientos de las patas, etc.). La suma de los tiempos de cada uno no debe exceder del tiempo de ciclo. Este tiempo de ciclo está marcado por la frecuencia en la que se recibe una imagen desde la cámara, que desencadena todo el procesamiento subsiguiente. El robot percibe una nueva imagen, en media, cada $33810\mu s$.

Uno de los principales requisitos como ya hemos mencionado es la reducción del tiempo de computación del módulo de localización. El tiempo de este módulo depende sobre todo del número de celdas que conforman la rejilla que divide al campo.

En la figura 5 mostramos un análisis del tiempo de cómputo y la precisión obtenida en distancia y en orientación que persigue averiguar el tamaño de celda óptimo. Los tamaños de celda que se han analizado varían desde 100 mm hasta 1000 mm para el ancho de cada celda. En la primera figura se observa como la diferencia entre 100 mm, que es el ancho de celda que se usa en el algoritmo *FMK* y tamaños de 300 mm a 400 mm es considerable, no obteniendo mejoras sensibles a partir de 500 mm. La gráfica central muestra el error que el algoritmo *FMK* presenta. El mínimo se sitúa en 500 mm y tamaños mayores no son capaces de localizar al robot. El mismo resultado se observa en el caso de la gráfica situada a la derecha, que corresponde al error en la orientación. Este comportamiento del algoritmo se debe a que el tamaño de celda es tan elevado que es verosímil obtener medidas sensoriales válidas en ellas.

3. Combinación de FMK con EKF

La solución que hemos adoptado ha consistido en aumentar el tamaño de la celda para mantener un sistema de localización global, y conseguir la precisión local requerida mediante un Filtro Extendido de Kalman (no indicaremos los detalles del filtro, únicamente indicar que estima la posición y la orientación en el plano).

La alternativa de utilizar únicamente un método de localización local como es el EKF no es válida. Básicamente, los problemas están relacionadas con la incapacidad del filtro para localizar al robot partiendo de una posición desconocida o cuando el robot sea desplazado manualmente de un lugar a otro del entorno. También es complicado evaluar si la estimación de la posición del robot se ha degradado debido a errores sensoriales.

En esta sección se presentará la forma en la que la combinación del método global *FMK* y uno o varios EKFs pueden solucionar estos problemas. En concreto, plantearemos dos estrategias de combinación de los métodos: combinar un filtro extendido de Kalman con *FMK* (*KFMK*) o varios EKFs con *FMK* (*NKFMK*) . En las siguientes secciones se mostrará en detalle como se combinan ambos métodos.

3.1. *KFMK*

La primera estrategia de combinación de varios métodos combina un único filtro extendido de Kalman con *FMK*. Como se puede observar en la figura 6, ambos métodos se ejecutan de manera independiente, teniendo las mismas entradas, que son la información sensorial en forma de odometría y la posición relativa a los objetos.

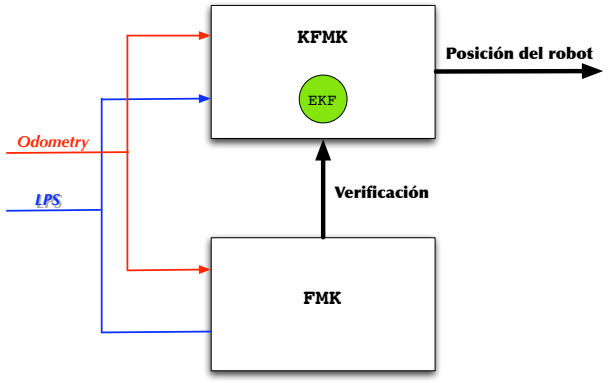

Figura 6: Esquema de combinación de un filtro extendido de Kalman y *FMK*.

3.2. Combinación de varios EKF con FMK

El método descrito en la sección anterior combina un *EKF* y *FMK*. Como la experimentación mostró que era poco costoso desde el punto de vista computacional mantener varios EKFs, nos planteamos usar mas de uno. Esto podría aportar una mejor respuesta a situaciones de secuestro y manejar varias hipótesis.

En resumen, esta segunda estrategia de combinación usa varios filtros extendidos de Kalman que se ejecutan en paralelo con *FMK*. El esquema mostrado en la figura 7 es parecido a la estrategia de combinación des-

crita anteriormente, pero con varios filtros extendidos de Kalman.

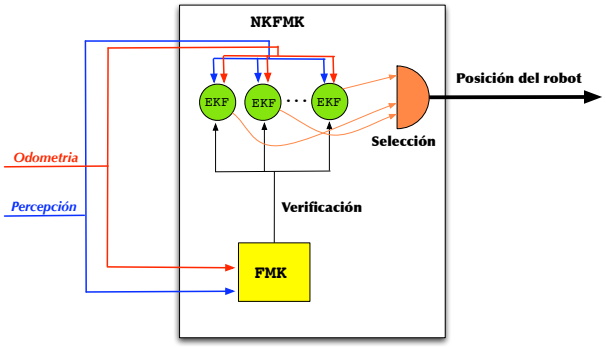

Figura 7: Esquema de combinación de varios filtros extendidos de Kalman y *FMK*.

El número de *EKF* varía durante la operación del robot según las necesidades del sistema. Al principio se crea un único *EKF*. Cada vez que *FMK* diverge de la estimación del EKF se inicia un nuevo filtro en la posición que estima FMK hasta alcanzar el número máximo de filtros permitidos. Si ya se ha alcanzado el número máximo de filtros permitidos, se comprueba la calidad del filtro que depende de la incertidumbre.

Si dos o más filtros convergen a una posición muy cercana, se elimina el filtro que tiene peor calidad. Esto hace que se tenga el menor número de filtros posible que cubran todas las hipótesis posibles. La posición del mejor filtro será la que se transmita hacia los otros módulos.

4. Validación experimental

El objetivo de la experimentación es comprobar el grado de precisión de las diferentes opciones, en concreto vamos a analizar el uso de un EKF, del sistema FMK original frente a las dos opciones propuestas en la sección anterior, FMK combinado con un EKF y FMK combinado con n EKFs.

Se han realizado diferentes trayectorias, en concreto analizaremos el recorrido descrito en la figura 8 que incluye una trayectoria por el campo y un "secuestro", es decir, un traslado de posición del robot por un agente extraño al robot (no por los propios movimientos del robot). En color rojo se muestra el recorrido que realiza el robot sobre el terreno de juego. Después del punto de control número 2, se produce un secuestro del robot desde el punto "so" (secuestro origen) al punto "sd" (secuestro destino). Este experimento también se ha repetido 28 veces, siendo la duración del mismo de unos 35 a unos 40 segundos.

Todos los resultados presentados en esta sección han

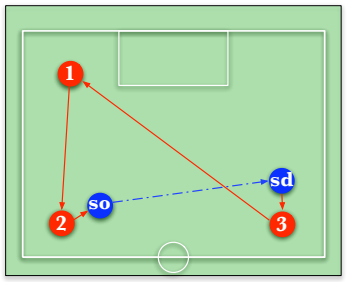

Figura 8: Recorrido del experimento con secuestro.

sido generados a partir de datos procedentes de experimentación con un robot real. Durante los experimentos se recogió en ficheros la información del sistema de "verdad absoluta" y la información perceptiva que el robot usa como entrada al módulo de auto-localización. Posteriormente se han analizado las ejecuciones en un ordenador. Durante estas ejecuciones es posible evaluar distintos métodos de auto-localización con distintas configuraciones, y obtener los resultados experimentales presentados.

Las gráficas de la figura 9 presentan el error medio después de 28 ejecuciones de la trayectoria descrita y el secuestro para cada uno de los métodos (EKF, FMK, EKF+FMK y FMK+nEKF respectivamente de arriba hacia abajo en la figura). La gráfica muestra el error medio y la desviación típica de dicho error en distancia (también se ha estudiado el error en el ángulo pero no se incluye por motivos de espacio) en el recorrido para cada uno de los métodos.

La comparación de las medias de las gráficas anteriores se muestran en la figura 10, donde se observa que la mayor parte del tiempo, salvo en el instante inmediatamente posterior al secuestro, el método *FMK+nEKF* tiene el error más bajo de todos los métodos comparados.

Sin embargo, no es el error medio la métrica más interesante para la aplicación del fútbol robótico. El dato del porcentaje de tiempo que el robot mantiene el error en la estimación por debajo de un umbral es muy descriptivo. Estos datos se encuentran recogidos en las tablas 1. Casi la mitad del tiempo, el error del método *FMK+nEKF* se encuentra por debajo de los treinta centímetros, lo que significa que la estimación de la posición del robot es muy buena. Por debajo del umbral de un metro, que puede entenderse como que el robot no está "perdido", los métodos combinados se encuentran el noventa por ciento del tiempo, mientras que el método *FMK* un sesenta por ciento. El método *FMK+EKF* queda en evidencia con estos datos, demostrando que el escaso error en media proviene del veinticuatro por ciento del tiempo que corresponde a

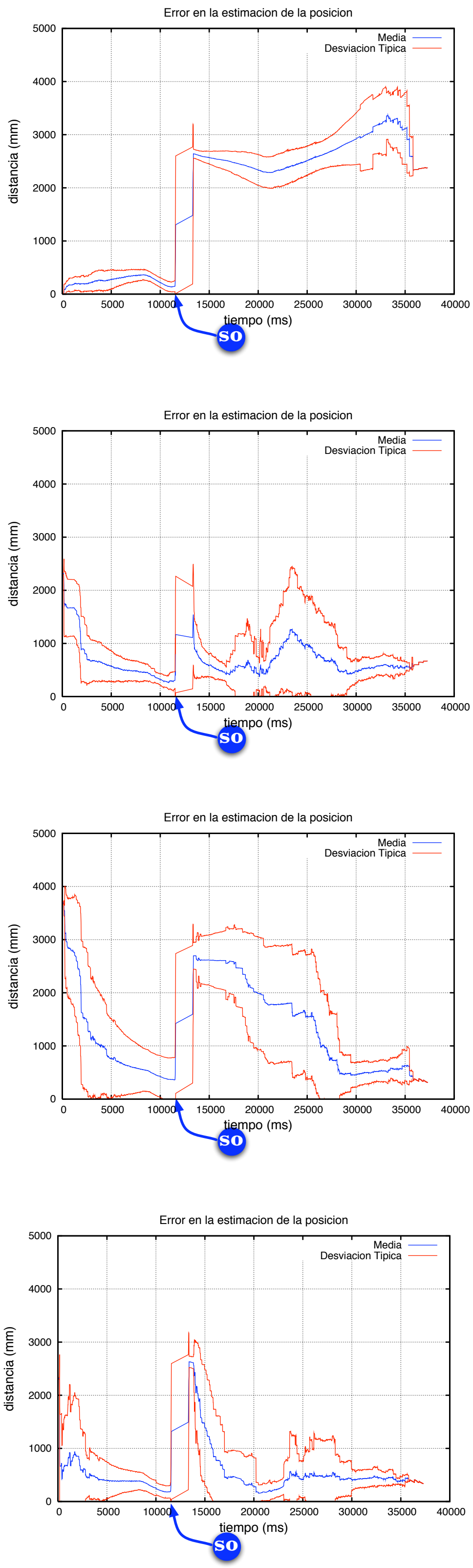

Figura 9: Media de error de en distancia entre la estimación en la posición y la posición real del robot.

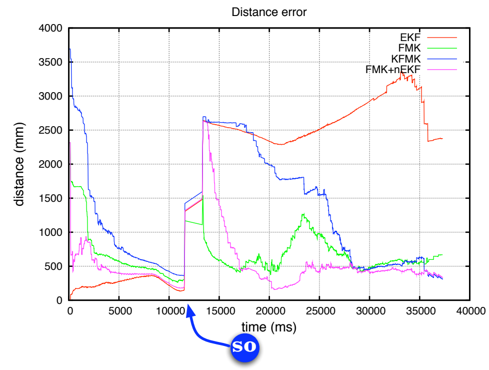

Figura 10: Media de error en distancia entre la estimación en la posición y la posición real del robot para EKF, FMK, EKF+FMK y FMK+nEKF

Rango	*EKF*	*FMK*	*FM* +EKF	*FMK* +nEKF
<= 300	24,60	24,09	21,25	45,32
<= 500	33,93	50,53	37,34	75,39
<= 1000	35,93	83,76	53,47	88,76
<= 1500	36,34	90,42	59,238	90,56

Cuadro 1: Análisis del porcentaje de tiempo que el error en distancia de la estimación de la posición del robot es inferior a un rango determinado (en mm).

su etapa inicial antes del secuestro en que su error era cercano a 0.

5. Conclusiones

En este artículo hemos presentado dos mecanismos para mejorar la precisión del mecanismo de autolocalización borrosa para robots con patas en el entorno de la RoboCup que hemos desarrollado. El experimento descrito demuestra que el algoritmo más interesante desde el punto de vista de la fiabilidad y precisión en distancia es el de fusión del FMK con n EKFs. Experimentos adicionales han demostrado que es igualmente el más fiable en ángulo y que su coste computacional es asumible.

Agradecimientos

Los autores quieren agradecer al resto de miembros del equipo TeamChaos, en particular a los Dres. Humberto Martínez y David Hernández su inmeso trabajo en el equipo. Este trabajo ha sido parcialmente financiado con fondos del Ministerio de Educación y Ciencia (DPI2007-66556-C03-01) y de la Comunidad Autónoma de Madrid (RoboCity 2030: S-0505/DPI/0176)

Referencias

[1] P. Buschka, A. Saffiotti, and Z.Wasik. Fuzzy landmark-based localization for a legged robot. In *Proceedings of the International Conference on Intelligent Robots and Systems 2000*, Takamatsu, Japan, October 2000.

[2] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427, 1999.

[3] D. Herrero-Pérez, H. Martínez-Barberá, and A. Saffiotti. Fuzzy self-localization using natural features in the four-legged league. *Lecture Notes in Computer Science. Robocup 2004*, 3276, 2005.

[4] J.Borenstein, B.Everett, and L.Feng. *Navigating mobile robots: Systems and techniques.* Ltd. Wesley, MA, 1996.

[5] Jana Koseck¡E1¿and Fayin li. Vision based topological markov localization. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, Barcelona (Spain), April 2004.

[6] María E. López, Luis Miguel Bergasa, and M.S.Escudero. Visually augmented POMDP for indoor robot navigation. *Applied Informatics*, pages 183–187, 2003.

[7] Humberto Martínez, Vicente Matellán, and Miguel Cazorla. Teamchaos technical report. Technical report, TeamChaos, 2006.

[8] Dandapani Radhakrishnan and Illah Nourbakhsh. Topological localization by training a vision-based transition detector. In *Proceedings of IROS 1999*, volume 1, pages 468 – 473, October 1999.

[9] Reid Simmons and Sven Koening. Probabilistic navigation in partially observable environments. In *Proceedings of the 1995 International Joint Conference on Artificial Intelligence*, pages 1080–1087, Montreal (Canada), July 1995.