Memorias del XVI Congreso Latinoamericano de Control Automático, CLCA 2014
Octubre 14-17, 2014. Cancún, Quintana Roo, México

RFCSIM Simulador Interactivo de Robótica Móvil para Control de Formación con Evitación de Obstáculos

E. Fabregas * **G. Farias** ** **S. Dormido-Canto** * **S. Dormido** *

* *Departamento de Informática y Automática. Universidad Nacional de Educación a Distancia (UNED). Calle Juan del Rosal, 16, 28040, Madrid, España (e-mail: efabregas@bec.uned.es).*
** *Pontificia Universidad Católica de Valparaíso. Av. Brasil 2147, Valparaíso, Chile (e-mail: gonzalo.farias@ucv.cl).*

Resumen: RFCSIM es un simulador de robótica móvil con altas prestaciones gráficas e interactivas desarrollado con *Easy Java Simulations*. El simulador permite llevar a cabo experimentos de control de posición de un robot móvil y el control de formaciones de un sistema multi-robots con evitación de obstáculos. Las formaciones se basan en una arquitectura de líder-seguidores en un entorno cooperativo y no-cooperativo. El simulador permite demostrar varios criterios de robótica móvil, así como diferentes leyes de control incluyendo obstáculos estáticos y dinámicos. Este tipo de simulaciones resultan de gran interés tanto para el campo de la investigación como en el ámbito de la docencia, ya que permiten demostrar conceptos fundamentales asociados a la robótica móvil, control de formación y evitación de obstáculos. El simulador proporciona además la posibilidad de guardar las configuraciones y los datos de los experimentos realizados para su posterior análisis.

Keywords: Simulación, Robots Móviles, Control de Formaciones.

1. INTRODUCCIÓN

En los últimos años el campo de la robótica ha experimentado un intenso desarrollo y crecimiento. Durante este tiempo los robots han sido creados para realizar tareas peligrosas, difíciles o repetitivas, como por ejemplo: limpieza de residuos tóxicos, exploración espacial, búsqueda y rescate de personas, entre otras.

Desde el punto de vista de la investigación los robots son sistemas muy interesantes ya que incluyen varios campos de la tecnología como son: la electrónica, la mecánica, el control automático, la programación y la inteligencia artificial, por solo mencionar algunos, ver Bermejo (2004). Entre los campos de investigación de la robótica actualmente se dedica mucho esfuerzo al control de formaciones de un conjunto de robots móviles, debido a las ventajas de implementar el control a un grupo de robots y no solo a uno. Entre estas ventajas se encuentran la reducción del coste a la hora de realizar una tarea o misión concreta, la robustez, la eficiencia energética y la mejora en rendimiento, entre otros, ver Li et al. (2005).

El control del movimiento de un grupo de robots, puede ser útil en tareas tales como: manipulación y transporte de objetos pesados y de grandes dimensiones, exploración, construcción de mapas en territorios desconocidos, vigilancia y operaciones de seguridad, búsqueda y misiones de rescate, ver Chang et al. (2013). Existen diversas formas de controlar sistemas multi-robots, entre las que se pueden mencionar: a) usando una estructura virtual (el control se basa en una estructura virtual rígida), ver Do and Pan (2007); b) control basado en comportamientos (varios comportamientos deseados son asignados a cada robot con un objetivo común: enjambres), ver Lawton et al. (2003) y Jevtic et al. (2012) y c) control basado en una arquitectura líder-seguidores (uno de los robots es designado como el líder, el resto como seguidores. Los robots seguidores tienen que posicionarse con relación al líder manteniendo una posición relativa deseada). Ésta última es una de las formas más usadas, ver Das et al. (2002) y Kurabayashi et al. (2009).

En este trabajo se presenta el simulador RFCSIM (*Robots Formation Control SIMulator*) desarrollado con *Easy Java Simulations* (*EJS*). *EJS* es una herramienta habitualmente utilizada con fines de educativos y de aprendizaje, que ha sido diseñada para crear simulaciones interactivas en *Java* sin la necesidad de poseer profundos conocimientos de programación, ver Esquembre (2014). El resultado es una aplicación totalmente interactiva que permite llevar a cabo experimentos de control de robots móviles en un entorno de simulación que resulta muy atractivo y versátil. El simulador facilita la realización de experimentos con un solo robot (control de posición) o con varios (control de formaciones). Además, permite seleccionar varios tipos de formaciones, diferentes leyes de control de posición y algoritmos de evitación de obstáculos, usando para ello una arquitectura de líder-seguidores en un entorno cooperativo o no-cooperativo. Al incluir obstáculos en los experimentos es posible crear configuraciones de situaciones típicas con obstáculos estáticos o móviles que facilitan la creación de entornos dinámicos muy vistosos desde el punto de vista de la experimentación. Este tipo de situaciones pueden

resultar de gran interés tanto para el campo de la investigación como en el ámbito de la docencia, ya que permiten demostrar y probar varios conceptos fundamentales asociados a la robótica móvil. El simulador permite además guardar configuraciones de experimentos para usarlos con posterioridad, así como almacenar los datos de dichos experimentos para su posterior análisis.

El trabajo está dividido como se describe a continuación. En la sección 2 se presenta el modelo cinemático de un robot diferencial con ruedas usado en el simulador; los detalles relacionados al control de su posición; los algoritmos de evitación de obstáculos; los detalles referentes al control de posición de un robot móvil; el control de formación de un sistema multi-robots y los elementos asociados a la arquitectura líder-seguidores. En la sección 3 se describen las partes que forman el simulador (el modelo y la vista), así como su funcionamiento. Finalmente, en la sección 4 se presentan las conclusiones y las futuras líneas de investigación relacionadas con este trabajo.

2. MODELO CINEMÁTICO Y CONTROL DE POSICIÓN DE UN ROBOT MÓVIL

Un robot diferencial con ruedas es un robot móvil cuyo movimiento está basado en dos ruedas colocadas a cada lado del cuerpo del robot y manipuladas independientemente. La ecuación (1) muestra la velocidad lineal del robot, que viene dada por las velocidades instantáneas de la rueda izquierda (ν_L) y de la rueda derecha (ν_R). Mientras que la ecuación (2) muestra la relación de dichas velocidades con la velocidad angular del robot (ω) y el radio de curvatura (R) que es la distancia entre el centro de las ruedas y el centro de curvatura instantáneo (ICC). Mientras que L representa la distancia entre las ruedas. Ver Angel et al. (2013).

$$\nu = \frac{\nu_R + \nu_L}{2} \tag{1}$$

$$\nu_{R,L} = \omega \left(R \pm \frac{L}{2} \right) \tag{2}$$

Las velocidades lineales izquierda y derecha son dos vectores paralelos y perpendiculares al eje de las ruedas y además, se considera que las ruedas ruedan sin deslizarse. Estas condiciones imponen las conocidas restricciones no holonómicas, que implican que el robot tiene que desplazarse para poder girar. Para cambiar su dirección el robot varía la rotación de las ruedas sin necesidad de ningún movimiento adicional para girar.

El modelo cinemático del robot en coordenadas cartesianas se representa por las ecuaciones (3), donde θ es el ángulo de orientación del robot. Las coordenadas del centro de masa del robot se representan por el punto (x_c,y_c). Ver Chwa et al. (2006) y Jinyan et al. (2005).

$$\begin{cases} \dot{x}_c = \nu cos\,(\theta) \\ \dot{y}_c = \nu sin\,(\theta) \\ \dot{\theta} = \omega \end{cases} \tag{3}$$

El objetivo del control de posición es que el robot alcance el punto de destino $P(x_p, y_p)$ desde su posición actual. La Fig. 1 representa las variables involucradas en este caso. Para alcanzar dicho punto, es necesario conocer la

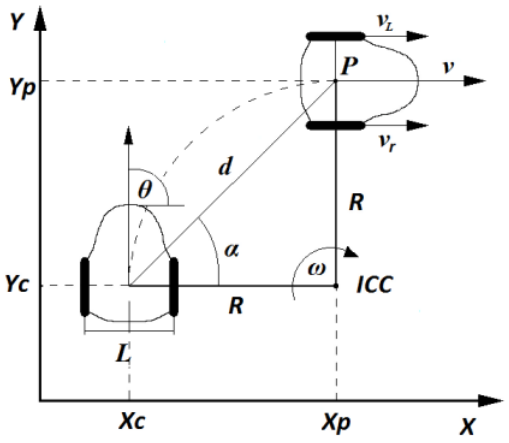

Figura 1. Control de posición de un robot móvil

distancia d y el ángulo α al que éste se encuentra respecto a la posición actual del robot. Las ecuaciones (4 y 5) muestran cómo se obtienen estos valores.

$$d = \sqrt{(y_p - y_c)^2 + (x_p - x_c)^2} \tag{4}$$

$$\alpha = \tan^{-1} \left(\frac{y_p - y_c}{x_p - x_c} \right) \tag{5}$$

La Fig. 2 muestra el diagrama de bloques del algoritmo de control. Los valores de d y α se calculan con las ecuaciones (4 y 5) representadas por el bloque *COMPUTE*. Con estos valores el algoritmo trata de minimizar el error del ángulo de orientación ($\theta_e = \alpha - \theta$) al mismo tiempo que intenta hacer cero la distancia al punto deseado ($d = 0$). Este experimento es conocido como "*point stabilization*", ver Baillieul (2005). El bloque *CONTROL* representa la acción de control a partir de la cual se obtienen ν y ω partiendo de θ_e y d. Con estos dos valores se obtienen las velocidades ν_L y ν_R implementando las ecuaciones (1 y 2) en el bloque *DIFERENCIAL*.

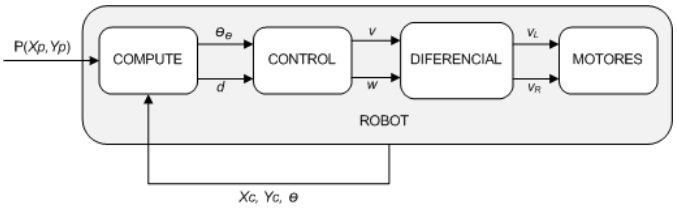

Figura 2. Diagrama de bloques del algoritmo de control

2.1 Algoritmo de evitación de obstáculos

El algoritmo implementado para evitar obstáculos es el *VFH* (*Vector Field Histogram*) en todas sus variantes. Ver Borenstein and Koren (1990), Borenstein and Koren (1991) y Ulrich and Borenstein (1998). En este algoritmo, a cada paso de ejecución, se construye un histograma de la celda que constituye el margen de visión al alrededor. Dentro de este margen se encuentran los obstáculos que el robot puede "ver" y por lo tanto, evitar. Las celdas ocupadas por los obstáculos se representan en el histograma como sectores circulares ocupados alrededor del robot y cuyo valor en el histograma se calcula en función de la proximidad del obstáculo. Estos sectores ocupados

serán los que el robot no puede utilizar para realizar su trayectoria.

Este método permite evitar los obstáculos de una forma rápida y continua, garantizando un buen funcionamiento en ambientes dinámicos. Como principal ventaja de este método a diferencia de otros, es que el robot no tiene que "conocer" su entorno completamente para poder evitar los obstáculos, solo necesita "ver" su entorno más cercano; por lo que el ambiente puede ser siempre cambiante.

Las velocidades angular y lineal del robot se calculan en función del histograma de su entorno. De forma tal que en ausencia de obstáculos, el robot trata de mantener la velocidad lineal máxima durante su desplazamiento. Pero cuando aparecen obstáculos en su entorno esta velocidad se ve reducida en aras de aumentar la velocidad angular y poder evitar los obstáculos por medio del giro. Las ecuaciones (6 y 7) muestran cómo se calculan estas velocidades en función del histograma en cada instante de simulación.

$$\nu = \nu' \left(1 - \frac{\omega}{\omega_{max}}\right) \tag{6}$$

$$\omega = \omega_{max} sin\left(\theta_e\right) \tag{7}$$

Donde ω_{max} representa la máxima velocidad angular permitida, la cuál se alcanza cuando el error angular de orientación es máximo ($\theta_e = \pm 90°$) y ν' se define en las ecuaciones (8 y 9).

$$\nu' = \nu_{max} \left(1 - \frac{h_c''}{h_m}\right) \tag{8}$$

$$h_c'' = min\left(h_c', h_m\right) \tag{9}$$

Donde ν_{max} es la velocidad lineal máxima, h_m es una constante que se determina empíricamente para proporcionar una reducción en la velocidad en presencia de obstáculos, mientras que h_c' es el valor del sector circular en el histograma. Si $h_c' > 0$ significa la presencia de un obstáculo en frente del robot. Un valor elevado de h_c' indica que el obstáculo en cuestión está muy cerca del robot, por lo que la velocidad lineal debe ser reducida drásticamente al tiempo que la velocidad angular debe aumentar para efectuar una maniobra de giro intentando de este modo evitar dicho obstáculo, ver Ulrich and Borenstein (2000).

2.2 Control de formación de robots móviles

El control de formación de robots puede llevarse a cabo de diferentes formas. En este caso se realiza siguiendo una arquitectura de líder-seguidores. Donde uno de los robots es el líder del grupo y aunque todos toman decisiones, las decisiones de los seguidores siempre dependen de las decisiones del robot líder. En el caso del control de posición, cada robot necesita saber su posición (x_c, y_c) y el punto a donde quiere dirigirse $P(x_p, y_p)$, ver Fig. 1. En este caso los robots seguidores realizan la formación tomando como referencia la posición del robot líder $(M_p(S_{nRef}))$, por lo que para todos los robots seguidores, las coordenadas del punto P son las coordenadas de la posición del robot líder en cada paso de simulación. El punto de referencia del robot líder (M_R) se le envía directamente cuando el usuario hace click sobre el espacio de trabajo (*Arena*). La Fig. 3 muestra el diagrama de comunicaciones entre los robots (*M-Líder*; S_1-*Seguidor*$_1$; S_2-*Seguidor*$_2$; S_3-*Seguidor*$_3$).

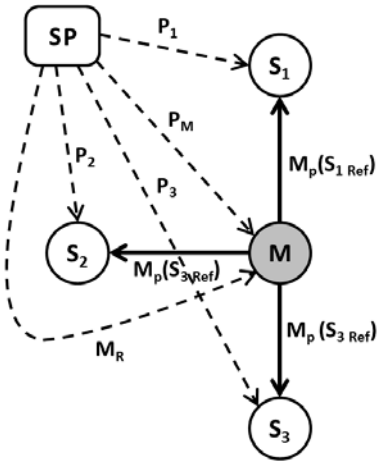

Figura 3. Diagrama de comunicaciones

El robot líder implementa las ecuaciones 4 y 5 para obtener su ley de control. Mientras que los robots seguidores implementan las ecuaciones (10 y 11) donde las coordenadas (x_d, y_d) representan la distancia entre cada robot seguidor y el robot líder dentro de la formación.

$$d = \sqrt{\left(y_p - y_c - y_d\right)^2 + \left(x_p - x_c - x_d\right)^2} \tag{10}$$

$$\alpha = \tan^{-1}\left(\frac{y_p - y_c - y_d}{x_p - x_c - x_d}\right) \tag{11}$$

El control de formaciones de tipo líder-seguidores se puede llevar a cabo de dos modos diferentes: a) cooperativo, donde el robot líder tiene en cuenta la posición de los robots seguidores en el cálculo de su velocidad; b) no cooperativo, donde el robot líder no tiene en cuenta la posición en la formación de los robots seguidores, ver Lawton et al. (2003). La ecuación (12) representa el error total (E_T) que se tiene en cuenta en el control de formaciones. Dicho error está formado por el error de la formación desde su posición actual hasta el punto de destino (E_f), sumado al error de posición de cada uno de los robots que la componen (E_p).

$$E_T\left(t\right) = E_f(t) + E_p(t) \tag{12}$$

En el caso del control en modo cooperativo, el robot líder calcula su velocidad (ν_m) como se muestra en la ecuación (13).

$$\nu_m\left(t\right) = K_p E_{pm}\left(t\right) - K_f E_f\left(t\right) \tag{13}$$

Donde su error de posición (E_{pm}) se multiplica por una constante proporcional (K_p) y se le resta el error de posición de los $N-1$ robots en la formación (E_f), que a su vez se obtiene como la sumatoria de los errores de posición de cada uno de los robots dentro de la formación (ecuación (14)), multiplicado por la constante (K_f).

$$E_f(t) = \sum_{i=1}^{N} E_{pi}(t) \tag{14}$$

En el caso de control en modo no cooperativo el robot líder calcula su velocidad usando la ecuación (13) haciendo cero la constante (K_f), por lo que el error de los N robots seguidores dentro de la formación no se tiene en cuenta.

3. EL SIMULADOR RFCSIM

Para desarrollar el simulador se ha usado el *Easy Java Simulations (EJS)*. *EJS* es una herramienta diseñada para crear simulaciones interactivas en *Java* habitualmente con fines de enseñanza o aprendizaje, ver Esquembre (2014). Esta aplicación consta de tres partes fundamentales: *Descripción*, *Modelo* y *Vista*. En la sección *Descripción* permite mostrar al usuario una descripción de la simulación y las instrucciones de uso. La sección *Modelo* incluye las variables, las ecuaciones que constituyen el modelo de la simulación, algunas relaciones fijas entre las variables y otras ecuaciones personalizadas. En esta sección se describen las partes que conforman el simulador en correspondencia con las secciones de *EJS*.

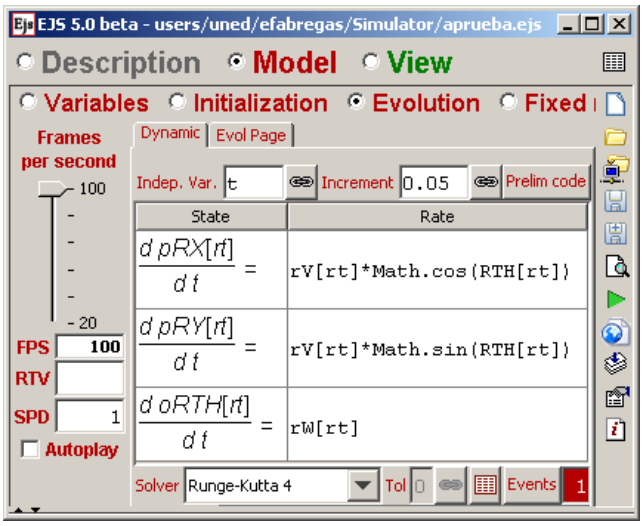

Figura 4. Modelo *EJS* del simulador

3.1 Modelo de RFCSIM

La Fig. 4 muestra el modelo *EJS* del simulador. Las expresiones del modelo implementan las ecuaciones (3). Las variables involucradas se representan como vectores de elementos de dimensión rt que es la cantidad de robots seguidores que conforman el experimento (el modelo del robot líder es el mismo pero se implementa independiente del resto). Las variables son las siguientes: $pRX[rt]$ y $pRY[rt]$ que son las coordenadas $(x;y)$ de la posición de cada robot respectivamente; $rV[rt]$ es la velocidad lineal; $RTH[rt]$ es la orientación y $rW[rt]$ es la velocidad angular.

3.2 Vista del simulador RFCSIM

La Fig. 5 muestra la vista del simulador en tiempo de diseño. Como se puede apreciar, los elementos visuales que componen la simualción se disponen en una estructura de árbol a partir de los objetos predefinidos de *EJS*. La raíz del árbol es la ventana principal del simulador (*frame*) que contiene todos los elementos que se describen a continuación. Dicha ventana principal de la aplicación en tiempo de ejecución está dividida en cuatro paneles fundamentales (*Menú*, *Arena*, *Panel de gráficos* y *Panel de controles*), como se muestra en la Fig. 6.

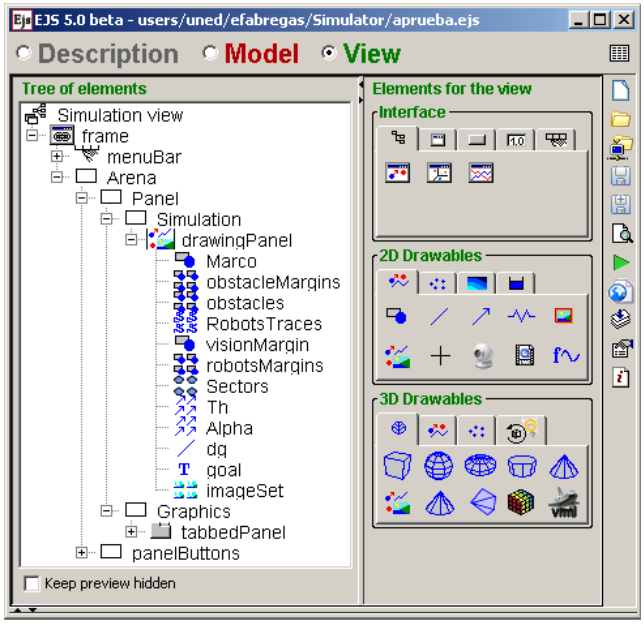

Figura 5. Vista *EJS* del simulador en tiempo de diseño

Panel de Menú

El Panel de Menú contiene las opciones que permiten controlar por medio de botones la ejecución de la simulación (*Ejecutar*, *Detener*, *Pausar* y *Reiniciar*). También permite seleccionar la formación deseada para los robots (*Circular*, *Línea* y *Libre*). Esta última permite realizar una formación personalizada. A través de este panel también se puede seleccionar la cantidad de robots y obstáculos de cada experimento. Otra opción de mucha utilidad es la de guardar y cargar experimentos realizados, de forma tal que el usuario puede salvar una configuración de obstáculos y robots para usarlos nuevamente en otro momento y continuar con el experimento. Al guardar un experimento lo que se hace es guardar el estado de todas las variables de la simulación en un fichero *.ejs* que puede ser posteriormente abierto desde *EJS*.

Espacio de trabajo (Arena)

La *Arena* es el espacio de trabajo donde tienen lugar los experimentos de una forma interactiva. Al hacer click en el espacio de trabajo se modifica la referencia del robot líder, tomando como punto de destino la posición que se ha marcado con el puntero del ratón, tanto para experimentos con un solo robot como con varios. Se pueden cambiar las posiciones de los obstáculos por medio del ratón con acciones del tipo arrastrar y soltar, para crear configuraciones sencillas o de mayor complejidad. Al iniciar un experimento con un solo robot la opción de selección del tipo de formación se encuentra deshabilitada. Al añadir más robots al experimento dicha opción se habilita permitiendo seleccionar la formación que se desee realizar. En el caso de la formación personalizada para modificar dicha formación se debe realizar con la simulación en pausa, puesto que si se realiza con la simulación en ejecución, se tomará como una perturbación al sacar un robot seguidor de su posición en la formación. Los resultados de los experimentos así como el *Histograma* del robot líder, pueden observarse en el *Panel de gráficos*.

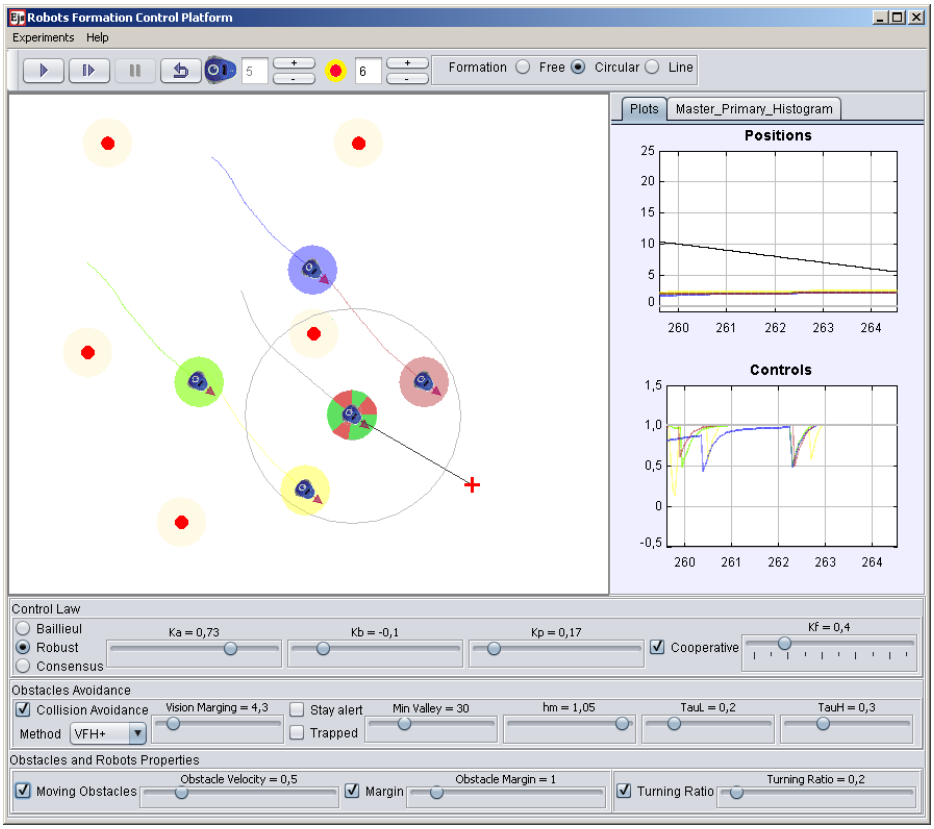

Figura 6. Ventana principal del simulador RFCSIM

Panel de gráficos e Histograma

En el lado derecho del panel del espacio de trabajo se encuentra un panel que contiene dos pestañas: *Gráficos* e *Histograma*. A través de los gráficos se puede observar el resultado de los experimentos (valores de las leyes de control y las posiciones de todos los robots). El color de cada traza indica el robot al que pertenecen los datos. Por su parte la pestaña llamada *Primary Histogram* muestra el histograma del robot líder, a través del cual se pueden observar los valores del algoritmo de evitación de obstáculos a cada paso de simulación. Puesto que los robots se "ven" entre ellos como obstáculos, los valores del histograma del robot líder muestran los sectores ocupados del campo de visión del robot líder, teniendo en cuenta tanto a los obstáculos como a los robots seguidores.

Panel de Controles

En la parte inferior de la ventana principal del simulador se encuentra el *Panel de Controles* a través del cual el usuario puede interactuar con la simulación modificando algunos de sus parámetros dependiendo del experimento que desee llevar a cabo. Este panel está dividido en tres subpaneles, el primer subpanel está relacionado con la ley de control que implementan los robots. Es posible seleccionar de entre tres leyes de control diferentes implementadas: *Baillieul*, ver Baillieul (2005); *Robust*, ver Siegwart et al. (2011) y *Consensus*, ver Olfati-Saber et al. (2007). Dependiendo de la ley de control seleccionada se pueden modificar sus parámetros y observar los resultados. También se puede seleccionar si el control se realiza en modo no cooperativo o cooperativo, pudiéndose en este caso modificar el coeficiente de "cooperación", representado por K_f en la ecuación (13). Dependiendo del valor de este coeficiente, el robot líder tendrá más o menos en cuenta a los robots seguidores durante la traslación de la formación.

El segundo subpanel está relacionado con el algoritmo de evitación de obstáculos. Entre las propiedades que se pueden modificar en este subpanel, se encuentran la selección del algoritmo y sus propiedades (*hm*, *TauL* y *TauH*). Estas opciones permiten comparar las tres variantes de dicho método entre ellas (*VFH*, VFH^+ o *VFH**) y a su vez, con el comportamiento del sistema cuando los obstáculos no se tienen en cuenta durante el movimiento de los robots. También se puede seleccionar el tamaño del *margen de visión* de los robots, para observar su influencia directa en el algoritmo de evitación de obstáculos. Cuanto mayor sea el *margen de visión* de los robots, los obstáculos podrán ser detectados con mayor antelación porque los robots podrán "verlos" antes, disminuir la velocidad y evitarlos con mayor antelación. Esto resulta beneficioso pero a la vez limita las capacidades de movimiento de los robots, por lo que se debe establecer un compromiso teniendo en cuenta estos dos elementos.

El tercer subpanel permite interactuar con las propiedades físicas de los robots y los obstáculos. En cuanto a las propiedades de los robots se puede modificar su radio de giro. Un valor elevado del radio de giro implica limitaciones de movimiento que pueden influir tanto en la evitación de los obstáculos, como en alcanzar sus respectivos puntos de destino. En cuanto a las propiedades físicas de los obstáculos, se puede hacer que éstos sean móviles y además variar su velocidad. También se puede modificar el valor

del margen de seguridad de los obstáculos para lograr que sean evitados con mayor antelación y por lo tanto con mayor seguridad. Con las modificaciones de estas propiedades se logra un entorno mucho más dinámico pero a la vez más complicado desde el punto de vista del control.

El simulador RFCSIM se encuentra disponible online en el siguiente enlace: *http://lab.dia.uned.es/rfcsim/*

4. CONCLUSIONES

El presente trabajo describe el simulador RFCSIM que es una herramienta interactiva para el desarrollo de experimentos de control de posición de robots móviles diferenciales de dos ruedas. A lo largo del trabajo se presentan los conceptos fundamentales relacionados con el control de este tipo de robots y los algoritmos de evitación de obstáculos en un ambiente dinámico (obstáculos estáticos y móviles). Los resultados obtenidos con el simulador se pueden catalogar de muy buenos, lo que demuestra que la implementación del modelo cinemático de los robots y las ecuaciones relacionadas con el control de posición se han realizado de forma efectiva. No obstante, el simulador tiene mucho margen de mejora a través de la incorporación de nuevas funcionalidades que contribuyan a enriquecer sus prestaciones y su interactividad.

Entre las tareas que se pretenden llevar a cabo para mejorar la herramienta se encuentran: 1) implementar un espacio de edición de texto para que el estudiante pueda introducir sus propias leyes de control por medio de líneas de código; 2) incorporar la posibilidad de implementar diferentes leyes de control en el robot líder y en los robots seguidores (actualmente ambos usan la misma ley de control seleccionada); 3) incorporar la posibilidad de realizar otro tipo de arquitecturas de formación (estructura virtual o basada en comportamientos: enjambres); 4) dibujar los histogramas de los robots seguidores (actualmente solo se dibuja el del robot líder por limitaciones computacionales de *EJS*); 5) diseñar e implementar un algoritmo para que los robots puedan evitar un obstáculo móvil estando el robot en formación y detenido; 6) dar solución a algunos problemas que tienen los algoritmos de evitación de obstáculos en cuanto las "situaciones trampa"; 7) proveer al usuario de la posibilidad de medir tiempo de desplazamiento, por ejemplo entre dos puntos de la arena; 8) implementar el seguimiento de trayectorias predefinidas.

AGRADECIMIENTOS

Este trabajo ha sido subvencionado en parte por el Ministerio de Economía y Competitividad del Gobierno de España bajo el Proyecto de Investigación DPI2012-31303 titulado: Control Basado en Eventos de Sistemas Distribuidos y Colaborativos.

REFERENCIAS

Angel, L., Hernández, C., and Díaz-Quintero, C. (2013). Modeling, simulation and control of a differential steering type mobile robot. *32nd Chinese Control Conference*, 8757–8762.

Baillieul, J. (2005). *The geometry of sensor information utilization in nonlinear feedback control of vehicle formations*. Springer, College of Engineering, Boston University, Boston, MA 02215.

Bermejo, S. (2004). *Desarrollo de robots basados en el comportamiento*. Editions UPC.

Borenstein, J. and Koren, Y. (1990). Real-time obstacle avoidance for fast mobile robots in cluttered environments. *IEEE Transactions on Systems, Man, and Cybernetics*, 19, 1179–1187.

Borenstein, J. and Koren, Y. (1991). The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics & Automation*, 7(3), 278–288.

Chang, Y.H., Chan, W.S., Yang, C.Y., Tao, C.W., and Su, S.F. (2013). Adaptive dynamic surface control for fault-tolerant multi-robot systems. In *2013 International Conference on System Science and Engineering (ICSSE)*.

Chwa, D., Hong, S., and Song, B. (2006). Robuts posture stabilization of wheeled mobile robots in polar coordinates. *17th International Symposium on Mathematical Theory of Networks and Systems*.

Das, A., Fierro, R., Kumar, V., Ostrowsky, J., Spletzer, J., and Taylor, C. (2002). A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, 18(5), 813–825.

Do, K. and Pan, J. (2007). Nonlinear formation control of unicycle-type mobile robots. *Robotics and Autonomous Systems*, 55(3).

Esquembre, F. (2014). Easy java simulations (EJS). URL `http://fem.um.es/Ejs/`.

Jevtic, A., Gutierrez, A., Andina, D., and Jamshidi, M. (2012). Distributed bees algorithm for task allocation in swarm of robots. *IEEE Systems Journal*, 6(2), 296–304.

Jinyan, S., Guangming, X., Junzhi, Y., and Long, W. (2005). Leader-following formation control of multiple mobile robots. *IEEE Mediterrean Conference on Control and Automation*, 62(1), 808–813.

Kurabayashi, D., Choh, T., Cheng, J., and Funato, T. (2009). Adaptive formation transition among a mobile robot group based on phase gradient. In *IEEE International Conference o Robotics and Biomimetics 2008. ROBIO 2008*, 2001–2006.

Lawton, J., Beard, R., and Young, B. (2003). A decentralized approach to formation maneuvers. *IEEE Transactions on Robotics and Automation*, 19(6), 933–941.

Li, X., Xiao, J., and Cai, Z. (2005). *Backstepping based multiple mobile robots formation control*. IEEE/RSJ International Conference on Intelligent Robots and Systems.

Olfati-Saber, R., Fax, J., and Murray, R. (2007). Consensus and cooperation in networked multi-agent systems. In *Proceedings of the IEEE*, volume 95, 215–233.

Siegwart, R., Nourbakhsh, I.R., and Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots*. MIT Press.

Ulrich, I. and Borenstein, J. (1998). VFH^+: reliable obstacle avoidance for fast mobile robots. *IEEE International Conference on Robotics and Automation*, 1572–1577.

Ulrich, I. and Borenstein, J. (2000). VFH*: local obstacle avoidance with look-ahead verification. *IEEE International Conference on Robotics & Automation*, 2505–2511.