IMPLEMENTACIÓN BASADA EN LÓGICA BORROSA DE JUGADORES PARA LA ROBOCUP

José María Álvarez Rey
Universidad Rey Juan Carlos
jmalvare@gsyc.escet.urjc.es

Vicente Matellán Olivera
Universidad Rey Juan Carlos
vmo@gsyc.escet.urjc.es

José María Cañas Plaza
Universidad Rey Juan Carlos
jmplaza@gsyc.escet.urjc.es

Resumen

Este artículo presenta los trabajos realizados en la Universidad Rey Juan Carlos para construir un equipo de jugadores simulados según las normas de la RoboCup. Utilizando las acciones básicas permitidas por el servidor oficial se han construido unos comportamientos primitivos. Para cada jugador se ha creado una memoria local que integra la información subjetiva que tiene sobre su entorno y su estado interno. Sobre esta memoria un controlador borroso determina en cada instante la activación de los comportamientos primitivos que, a su vez, generan el comportamiento observable de cada jugador. Se han implementado siete roles hasta completar los distintos jugadores del equipo real: portero, central, delantero, media punta, lateral, medio e interior.

Palabras Clave: Fuzzy, RoboCup, robótica.

1. INTRODUCCIÓN

Uno de los eventos más conocidos en el campo de la robótica móvil actual es la competición mundial denominada RoboCup[1] [6]. Esta competición nació como una iniciativa para fomentar la ciencia y la tecnología. El objetivo es construir un conjunto de agentes (o robots) que jueguen un partido de fútbol (real o simulado) de manera autónoma e inteligente, proponiendo una plataforma común de experimentación. Rápidamente recogió el interés de investigadores y desde entonces se organizan periódicamente campeonatos y conferencias en distintas categorías y ámbitos geográficos. Los campeonatos permiten la competición entre distintas aproximaciones al problema y suponen un escenario desafiante para la investigación en distintos campos como robótica, agentes, inteligencia artificial, etc . Presenta un entorno dinámico, de tiempo real y distribuido donde se pueden investigar distintas tecnologías, con el aliciente de la competición y la vistosidad.

Dentro de sus diversas categorías la del simulador es la más indicada para probar tecnologías relacionadas con el control y coordinación de agentes porque permite abstraerse de los problemas prácticos asociados al uso de sensores y actuadores reales.

La mayoría de los equipos presentados en las anteriores ediciones de la liga simulada tienen el comportamiento de sus jugadores diluído en el código de los programas que ejecutan y usan estructuras típicas como semáforos o *handlers* para manejar a sus jugadores y tomar decisiones. Aunque estas opciones son totalmente válidas y eficaces (la mayoría de los ganadores las usan) hemos querido añadir al diseño del equipo otra herramienta: la lógica difusa.

El uso de la lógica borrosa en este escenario no es novedoso. Por ejemplo E. Aguirre [1] presenta un equipo de fútbol con un controlador borroso que activa o desactiva habilidades individuales para un simulador en Java (JavaSoccer). Nosotros presentamos una implementación para un entorno diferente, utilizando el simulador oficial de Robocup, equipos completos de 11 jugadores y desarrollamos 7 tipos de jugadores en lugar de 3.

Otros enfoques al mismo problema hacen énfasis en otros aspectos como por ejemplo la toma coordinada de decisiones. Así, en De la rosa et al. [4] se aborda el problema desde la perspectiva de agentes, donde cada jugador propone su acción en una primera fase y la comunica a sus compañeros. En una segunda fase se pueden detectar posibles inconsistencias entre decisio-

[0] El trabajo descrito en este artículo ha sido objeto de ayuda del programa PROFIT, (proyecto FIT-070000-2001-118) con cargo al presupuesto del Ministerio de Ciencia y Tecnología

[1] http://www.robocup.org

nes de compañeros, corrigiendo las acciones propuestas. En una tercera fase, un agente entrenador valida o corrige las decisiones desde el punto de vista de un observador global. Albert Oller et al. [10] abunda en el enfoque coordinado separando la toma de decisiones en cada jugador en una parte privada, que recoge los aspectos relativos al robot concreto, y una parte cooperativa, que recoge los aspectos colectivos.

En el presente trabajo proponemos un enfoque reactivo, sin comunicación explícita entre los distintos jugadores, para conseguir un equipo precompetitivo. Hemos desarrollado un conjunto de comportamientos primitivos comunes como *chutar_a_gol, avanzar_bola, mirar_pelota*, etc. Cada jugador ejecuta un controlador borroso que activa en cada momento, y de modo exclusivo, el comportamiento primitivo adecuado para la situación del robot. El controlador utiliza un conjunto de reglas borrosas, escritas en un fichero, para decidir que comportamiento ejecutar. De este modo el comportamiento observable del robot es fácilmente legible y modificable.

En la siguiente sección presentamos las características y la interfaz que permite el servidor de la Robocup. En la sección 3 describimos la arquitectura implementada para gobernar cada robot: comportamientos primitivos y controlador borroso, comentando la información sobre la que se toman decisiones y las reglas concretas utilizadas para el portero y para el delantero. Comentamos los experimentos realizados en la sección 4. Finalizamos el artículo con las conclusiones del trabajo y las líneas que apreciamos pueden mejorar el comportamiento del equipo.

2. SOCCERSERVER

El simulador de la RoboCup es un sistema de simulación de fútbol que permite a jugadores virtuales competir en un escenario dinámico, complejo y multiagente. Ofrece un entorno para la prueba de sistemas basados en inteligencia artifical, que permite a los investigadores centrarse en la creación y diseño del "cerebro" de cada agente.

El sistema de simulación funciona con estructura cliente/servidor: un servidor central, *soccerserver*, que se comunica con 22 clientes. Cada cliente controla un jugador y se comunica con el servidor mediante *sockets* UDP. Los clientes se pueden construir en cualquier lenguaje que permita comunicarse con el servidor vía *sockets*. En nuestro caso hemos elegido C.

Hay dos tipos de reglas que rigen el desarrollo de un partido, las que evalúa el servidor y las que evalúan los humanos. El simulador controla el partido de acuerdo con el reglamento típico del fútbol, incluyendo fueras

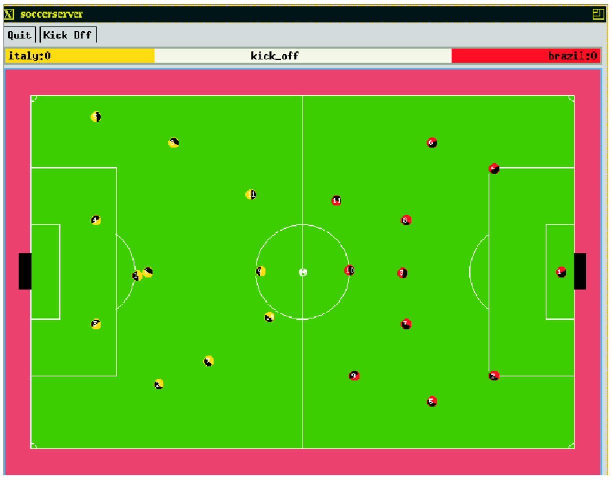

Figura 1: Imagen del terreno de juego que ofrece el servidor

de juego, *corners*, gol, etc. Pero estas reglas son insuficientes para interpretar ciertas situaciones del juego, como son las referentes a la intencionalidad de los jugadores. Para resolver esto el servidor viene provisto de un interfaz que permite a un árbitro humano detener el juego y conceder faltas, botes neutrales, etc. El servidor también envía a los jugadores mensajes para indicar cual es la situación del juego: fuera de juego, falta, gol, saque de esquina, saque de puerta... Por ejemplo cuando detecta fuera de juego lo indica y además reubica a los jugadores del equipo infractor en posiciones aleatorias.

La interacción entre servidor y clientes se organiza como una sucesión cíclica de turnos de tiempo que el servidor dedica periódicamente a comunicarse con cada cliente. En esta comunicación el servidor envía a cada cliente información acerca de donde está, lo que ve, lo que oye,... Con estos datos, cada cliente debe decidir qué comando realizar (moverse, girar, chutar,...) en esa iteración. El servidor materializa en el escenario virtual la acción básica que cada jugador decide en su iteración.

La simulación que produce el *soccerserver* es un campo virtual en el que todos los objetos son 2D y en el que se representan tanto pelota como jugadores con círculos, según muestra la figura 1. El servidor envía información a cada cliente sobre su situación en el campo. Esta información es de 3 tipos: visual, auditiva e interna. La primera se refiere a la distancia y orientación relativa de los objetos y jugadores. El servidor envía la información con error y cada jugador sólo es consciente de lo que ocurre delante suyo, concretamente, dentro de su cono de visión. La información auditiva se refiere a mensajes que puede "oír" el jugador tanto del árbitro como de otros clientes. Por último, la información interna contiene datos sobre el cansancio del jugador (*stamina*), su velocidad, el número de acciones que ha realizado, etc. Un ejemplo de información visual enviada por el servidor sería:

```
recv 2035: (see 0 ((goal r) 73 -7) ((flag r t) 84.8 -32)
           ((flag r b) 76.7 18) ((flag p r t) 63.4 -28)
           ((flag p r c) 56.8 -10) ((flag p r b) 56.8 10)
           ((ball) 22.2 -26 0 0) ((line r) 72.2 -90)
```

Respecto de la actuación cada jugador puede realizar una serie de comandos básicos como son girar (*turn*), avanzar (*dash*), chutar (*kick*), girar el cuello (*turn_neck*), mandar un mensaje “sonoro” (*say*) y cambiar las características del modo de visión (*change_view*). Para que el juego sea lo más parecido a lo que ocurre en el mundo real, el simulador introduce ruido en los movimientos de los objetos.

3. EL EQUIPO

A la hora de diseñar el equipo de fútbol hemos usado un enfoque totalmente distribuido, construyendo un conjunto de jugadores reactivos que toman sus decisiones de modo individual, sin comunicarse explícitamente con otros, ni con ningún controlador centralizado. En cada uno de los jugadores se ha descompuesto el problema en dos niveles. En el primer nivel se ha desarrollado un conjunto comportamientos primitivos utilizando las acciones básicas que permite el servidor (*dash*, *turn*, etc). En el segundo nivel, cada jugador utiliza un controlador borroso para decidir cuál de estos comportamientos primitivos debe tomar el control en cada momento.

Cada jugador posee una memoria local individual en la que integra toda la información relevante para su actuación en forma de variables. Se incluyen las posiciones de otros jugadores (tanto compañeros como rivales) y la pelota si es que caen dentro de su cono de visión. También se almacena la posición propia (que se calcula en función de las marcas del campo que cada jugador ve), su orientación en el campo y variables internas como la *stamina* que determina la energía disponible por el jugador para sus movimientos. Cuanto menor sea esa energía sus desplazamientos serán más lentos y sus chutes más flojos. Esta subjetividad en la percepción es también una gran diferencia con respecto al trabajo de Aguirre et al.[1].

Los comportamientos primitivos se han implementado como funciones que determinan la acción final para un ciclo de control, justo aquel en el que son invocadas. Utilizan la información que hay en la memoria local y las acciones básicas permitidas para conseguir algún objetivo concreto sencillo. Constituyen los bloques con los que construiremos el comportamiento observable de los jugadores. Con distintos nombres estos bloques primitivos aparecen con profusión en la literatura; son las habilidades de Aguirre et. al [1], las *capabilities* de [4] y los *behaviors* de [5]. En nuestra implementación son: mirar_bola, buscar_meta, ir_bola, chutar_gol, ir_posición, pasar_pelota, salir, despejar, buscar_pase, avanzar_bola y deambular.

Un ejemplo de comportamiento primitivo es *Chutar_gol*. Cuando este comportamiento es activado, el jugador intentará chutar la pelota. Para ello comprobará si está en su radio de tiro (comprueba una variable de su memoria interna que le dice la distancia a la pelota). Si esto no ocurre, se acercará a la pelota (ira hacia ella). Si está en su radio de acción, procederá a chutar (enviando un comando *kick*).

Estos comportamientos primitivos no están todos activos a la vez. Para cada jugador un controlador borroso determina cuál de ellos debe tomar el control del robot en cada momento a partir de la información que recibe del servidor. En cada iteración de control se invoca al controlador borroso para que decida qué comportamiento primitivo se activa. De este modo el comportamiento observable final se compone como una secuencia de comportamientos primitivos. El controlador borroso utiliza las reglas borrosas incluidas en el fichero de configuración. Este fichero está escrito en texto plano y tiene un formato sencillo. De este modo se puede modificar con facilidad, lo que resulta muy útil para sintonizar las reglas y hacer pequeños ajustes del comportamiento final.

Cara al controlador borroso, la memoria local se plasma en tres variables: la distancia a la pelota [4] *DistBola*, a la posición de referencia en el campo de cada jugador [8]*DistRef*, y la distancia al jugador del equipo contrario más cercano *DistContrario*. Estas variables borrosas son las de entrada al controlador y dan cuerpo a los antecedentes de las reglas.

La figura 2 presenta la descripción borrosa de la variable *DistBola*. En esta implementación el diseño de las etiquetas borrosas de se ha realizado de forma heurística, incluyendo la etiqueta *NoConocido* para cubrir el vacio de control cuando la distancia a la bola no es conocida. En la literatura se pueden encontrar mecanismos automáticos de diseño, por ejemplo mediante el uso de algoritmos genéticos [7].

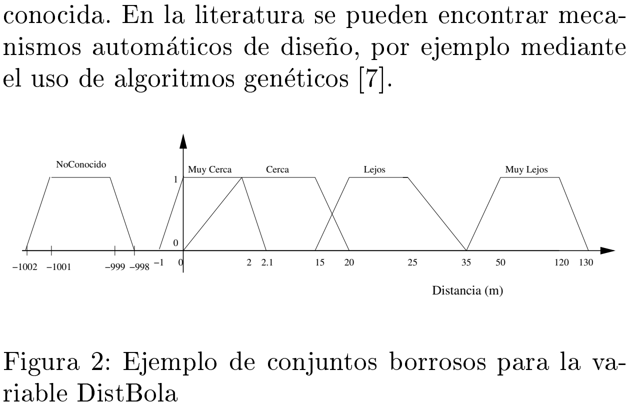

Figura 2: Ejemplo de conjuntos borrosos para la variable DistBola

Las variables de salida del controlador marcan la motivación para cada uno de los comportamientos primitivos, como por ejemplo chutar a gol, despejar, buscar un pase, etc. La motivación es gradual, dependiendo del grado con el que se activen los antecedentes de las reglas, sin embargo la salida global del controlador se utiliza de modo discreto. Es decir, sólo un comportamiento primitivo se activa después de cada llamada al controlador, aquel cuya motivación sea la mayor. De esta manera hay correspondencia directa entre las decisiones del controlador borroso y la acción que realiza el jugador.

El bucle de control que ejecuta cada jugador consiste en (1) leer la información suministrada por el servidor y actualizar la memoria, (2) calcular las variables de entrada, (3) evaluar las reglas obteniendo unos valores para las variables de salida que se transforman en acciones, (4) invocar al comportamiento primitivo seleccionado y (5) enviar las acciones decididas al servidor.

Dentro de un jugador el controlador borroso se puede ver como el árbitro central que determina qué comportamiento básico debe activarse en cada momento. Este problema se conoce como selección de acción y ha recibido mucha atención en la literatura. Por ejemplo [5] lo aborda para su equipo de Robocup con una red de comportamientos primitivos cuya activación depende de una red distribuida de motivación. La ventaja de materializarlo como un controlador borroso es que se expresa a través de reglas lingüísticas, próximas al lenguaje humano, y que se puede modificar de modo sencillo.

3.1. LOS JUGADORES

Las reglas son diferentes para cada jugador puesto que dependen de la posición que tenga en el campo. Cada posición está asociada a un comportamiento por lo que no tiene sentido que tengan todos las mismas reglas.

Cada jugador posee un cometido dentro del equipo. Dependiendo de este cometido tendrá una serie de reglas u otras. Así, se ha definido siete tipos de jugadores que son:

Portero cuyo cometido es evitar que la pelota entre en la portería.

Lateral que se encarga de proteger la zona cercana a la línea y evitar que la pelota se aproxime demasiado a la portería.

Central que debe evitar que el balón llegue al área, extremando las precauciones ya que son los últimos jugadores (es por ello que no participan en el juego de ataque).

Medio cuya función dentro del equipo es la de llevar la pelota lo más cerca posible del área rival para que los delanteros puedan rematar a gol.

Interior que debe apoyar al medio en su objetivo de llevar el balón al área rival, pero su radio de acción es mucho más pequeño y se reduce al centro del campo.

Delantero que es el encargado de meter goles.

Media punta cuyo objetivo es meter goles, al igual que el delantero, pero su posición no está tan adelantada.

En el enfoque presentado no hay comunicación explícita entre los distintos jugadores. En este sentido contrasta con otras aproximaciones donde los jugadores se pasan mensajes entre sí [10] o incluso se coordinan con un entrenador [4] que de algún modo vela por el interés colectivo.

Sin embargo el diseño de los jugadores se ha hecho pensando en que tienen compañeros cuya función es conocida. De este modo se logra un comportamiento implícito de equipo. Como en [8], a esta coordinación implícita ayuda que cada jugador tiene asociada una posición de referencia, que es distinta para cada uno de ellos. Por ejemplo la referencia de un defensa central está cerca de su propia portería, mientras que la de los delanteros es el área del rival. Esta posición influye en las reglas borrosas a través de una variable de entrada que es precisamente la distancia hasta esa posición.

3.2. EL PORTERO

El portero es un punto crítico del equipo, ya que es el último jugador entre la pelota y la portería. Por lo tanto, hay que diseñar sus reglas de comportamiento con un cuidado especial, llegando hasta el punto de diseñarle comportamientos primitivos especiales para él, como son *Salir* y *Despejar*.

Uno de los principales problemas del portero es cómo conseguir que se mantenga dentro del área y, a la vez, pueda salir a por la pelota en el momento justo. Probando experimentalmente con distintas reglas llegamos al siguiente conjunto, con el cual el portero sale a por la pelota si está cerca, pero si su situación en el campo es demasiado lejana de su portería, la ignora y vuelve a su área.

> SI DistRef es Lejos → IrPosicion es Alto & Salir es Bajo
> SI DistBola es Cerca & DistContrario es Cerca → Despejar es Alto & IrPosicion es Bajo
> SI DistBola es Cerca & DistContrario es Lejos → Salir es Alto & IrPosicion es Bajo

SI DistBola es Cerca & DistContrario es MuyLejos → Salir es Alto

El ajuste de estas reglas ha sido crítico para un buen funcionamiento. Se ha conseguido que el portero despeje la pelota hasta un punto en el que vuelve al área. Este punto coincide con la posición en el campo de los centrales, que se encargan de despejar la pelota definitivamente hasta una zona no peligrosa. Un fallo típico que aparecía en el diseño de reglas consiste en que cuando la pelota está cerca de la línea que delimita el área pero fuera de ella el portero salía, pero se volvía hacia atrás sin despejarla. Esto se debía a que llegaba un punto en que la regla de mover al portero hacia su posición de referencia era más fuerte que la de despejar. Si por el contrario se favorecía la regla del despeje, si resultaba corto el portero intentaba despejar nuevamente, siguiendo a la pelota por todo el área, porque la regla de mantener al portero en su posición de referencia tenía poca prioridad.

3.3. EL DELANTERO

El delantero es el encargado de meter goles. Su posición en el campo es la más adelantada y no pasa de la línea de 30 metros del campo rival. Sus reglas son totalmente diferentes del resto ya que lo principal es que esté siempre adelantado y repleto de *stamina* para llegar a la pelota y chutar con fuerza.

A este jugador se le han modificado los parámetros de la función *DistRef* para que no se aleje mucho de su posición en el campo y así evitar que gaste *stamina* en zonas cuyas opciones de tiro son escasas. Teniendo en cuenta esto, tras varias pruebas, se decidió que las reglas del delantero fueran:

SI DistBola es Muy Cerca → ChutarGol es Alto
SI DistBola es Cerca → ChutarGol es Alto
SI DistRef es Lejos → IrPosicion es Alto

4. EXPERIMENTACIÓN

Se han realizado una serie de partidos con diferentes equipos cuyo código está disponible en la página oficial de la *Robocup*. No se ha seguido ningún criterio especial para elegir los equipos rivales, aunque se ha procurado usar aquellos que mejores resultados han tenido en su participación en el torneo, y cuyo código está disponible (para futuras mejoras, comparaciones, etc).

Los resultados de las simulaciones se reflejan en la tabla 1 y tienen dos lecturas, la deportiva y la estrictamente funcional. En lo deportivo se aprecia un empeoramiento de los resultados a medida que los equipos son más recientes. En lo funcional hay que destacar varios aspectos:

Cuadro 1: Resultados de competición

RIVAL	AFILIACIÓN	AÑO	RES
Humboldt	Universidad de Berlín, Alemania	1997	2-0
Andhill97	Inst. de Tecnología de Tokio, Japón	1997	3-1
CMUnited 97	Universidad de Carnegie Mellon, USA	1997	3-3
Andhill98	Inst. de Tecnología de Tokio, Japón	1998	0-1
Sparrows		1998	0-0
Gongeroos	Universidad de Wollogong, Australia	1999	0-7
Ku-Sakura2	Universidad de Kinki, Japón	1999	1-0
Polytech		1999	5-0
FCPortugal	Universidad de Lisboa, Portugal	2000	1-7
Essex-Wizards	Universidad de Essex, Inglaterra	2000	0-17
Dirty-Dozen	Instituto de Osnabrück, Alemania	2001	1-9

- En general, los jugadores se comportan de acuerdo con las reglas que se han diseñado para ellos, otra cosa es que estas reglas sean las adecuadas. Con esto se quiere destacar que las reglas diseñadas no son todo lo eficientes que cabría esperar para algunos jugadores, ya que, mientras que para el portero, los centrales y el medio centro son bastante acertadas, los laterales y media puntas tienen problemas.

- Se aprecia una descoordinación entre las zonas que, teóricamente, deben cubrir los jugadores y las que son útiles que cubran, ya que hay ocasiones del juego en las que el balón se queda en una especia de *tierra de nadie* de la que ningún jugador se encarga de cubrir.

- Los jugadores del equipo tienen tendencia a jugar únicamente por el centro siendo el juego de bandas prácticamente inexistente. Esto supone que sea realmente complicado jugar contra equipos que acumulan muchos jugadores en dicha zona.

- Al no tener un método específico de predicción de movimiento, las salidas del portero no siempre son tan efectivas como debieran, sobre todo cuando el balón viene en diagonal.

- El comportamiento de los centrales es satisfactorio ya que van a por la pelota en el momento adecuado y siempre tienen *stamina* suficiente para despejarla. Esto supone que el equipo reciba, generalmente, pocos goles.

- La implementación de algunos comportamientos primitivos como *Chutar_Gol()* hace que en ciertos casos los jugadores se pierdan buscando el pase cuando chutar no tiene las condiciones para realizarse eficientemente. Por ejemplo esto provoca que los media punta tengan una excesiva tendencia a centrar al área en busca del delantero llegando a desaprovechar ocasiones claras de gol sólo porque la portería está mínimamente tapada por algún jugador del equipo contrario.

En resumen, los jugadores cumplen con las funciones que se les han encomendado aunque, en algunos casos, el diseño de dichas funciones no es todo lo satisfactorio que cabría esperar en el aspecto deportivo.

5. CONCLUSIONES Y LÍNEAS FUTURAS

Se ha conseguido el objetivo de este trabajo que era desarrollar un equipo capaz de jugar razonablemente bien en el simulador de la RoboCup, para participar en la liga simulador.

El uso de la lógica difusa se ha mostrado muy conveniente porque permite mejorar, adaptar y modificar el equipo con relativa facilidad. Basta cambiar el fichero de reglas de comportamiento y volver a ejecutar el equipo. Ni siquiera hay que recompilar el código, pues las reglas se cargan en tiempo de ejecución. Con otros diseños el cambio de comportamiento o ajuste resulta mucho más laborioso, llegando incluso a requerir recodificar gran parte del sistema, como les ha ocurrido a algunos equipos.

Con la experiencia descrita en este artículo hemos detectado algunos puntos oscuros en los que estamos trabajando en la actualidad para mejorar el funcionamiento del equipo:

- Uso de técnicas de predicción para alcanzar la pelota con mayor rapidez y menor gasto de *stamina*.
- Diseñar un protocolo de comunicaciones entre los jugadores que permita un comportamiento conjunto explícito de todos ellos. Permitir el intercambio de roles entre los jugadores según su nivel de cansancio.
- Mejorar algunos comportamientos primitivos como el pase o el tiro a gol, que pueden tener en cuenta al portero y a los jugadores del equipo.
- Diseñar estrategias específicas para situaciones especiales del juego. Por ejemplo para los *corners* los tiros libres y los saques de banda. También se pueden contemplar distintas estrategias que modulen el comportamiento global dependiendo del marcador.

Referencias

[1] Eugenio Aguirre, Juan C. Gámez y Antonio González. Un sistema multi-agente basado en lógica difusa aplicado al ámbito de la RoboCup. *Actas del II Workshop de Agentes Físicos (WAF'2001)*, WAF2001, pp. 131-145, 2001.

[2] Valentino Braitenberg. *Vehicles. Experiments in Synthetic Psychology.* MIT Press, Cambridge, MA (USA), 1984.

[3] Rodney A. Brooks. A robust layered control system for a mobile robot. En *IEEE Journal of Robotics and Automation*, RA-2(1), pp. 14-23 1986.

[4] J.Ll. de la Rosa, A. Oller, J. Vehí y J. Puyol. Soccer team based on Agent-Oriented Programming. En *International Journal on Robotics and Autonomous Systems*, 21(2), pp. 167-176, 1997.

[5] Klaus Dorer. Behavior networks for continuous domains using situation dependent motivations. En *Proceedings of the XVI International Joint Conference on Artificial Intelligence*, IJCAI-99. pp. 1233-1238, 1999.

[6] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. Robocup: The robot world cup initiative. En *Proceedings of the IJCAI-95 Workshop on Entertainment and AI/Life*, pp. 19-24, 1995.

[7] Vicente Matellán, Camino Fernández y José Manuel Molina Genetic Learning of Fuzzy Reactive Controllers. En *Robotics and Autonomous Systems.* Volumen 25, Número 1-2, pp. 33-41, 1998.

[8] F. Fernández, G. Gutiérrez, J. M. Molina. Coordinación Global Basada en Controladores Locales Reactivos en la RoboCup. En *Actas del I Workshop de Agentes Físicos (WAF'2000).* pp. 73-85. Tarragona, España. Septiembre 2000.

[9] Itsuki Noda. Soccer server: A simulator of robocup. En *Proceedings of AI Symposium'95*, pp. 29-34. Japanese Society for Artificial Intelligence, 1995.

[10] A. Oller, J.L. de la Rosa, R. García, J.A. Ramón, J.A. y A. Figueras. Micro-robots playing soccer games: a real implementation based on a multiagent decision-making structure. En *International Journal of Intelligent Automation and Soft Computing* Special Issue on Soccer Robotics: Microrobot WorldCup Soccer Tournament'97, 6(1), pp. 65-74, 2000.

[11] Peter Stone and Manuela Veloso. Towards collaborative and adversarial learning: A case study in robotic soccer. Technical Report, School of Computer Science. CMU-CS-95-207. Carnegie Mellon University, 1995.