

Fútbol de Robots en Squeak

*Centro de Altos Estudios en Tecnología Informática
Facultad de Tecnología Informática
Universidad Abierta Interamericana*
*Lic. Zabala, Gonzalo – gonzalo.zabala@vaneduc.edu.ar
Morán, Ricardo – richi.moran@gmail.com
Blanco, Sebastián – sebastiangabrielblanco@gmail.com
Puricelli, Matías – mpuricelli@fibertel.com.ar*

Palabras clave:

Robótica Situada – Squeak – Inteligencia Artificial – Simulador RobotSoccer – Environment – Proxy – Estrategia – Visión Global - Doraemon.

Objetivos:

El objetivo de la investigación consiste en solucionar problemas inherentes a la robótica situada de forma eficiente mediante un ambiente de objetos derivado de Smalltalk llamado “Squeak”. Se utilizará como ejemplo el fútbol de robots (físico y simulado) debido a que vincula conceptos de navegación, visión, comportamiento colaborativo, inteligencia artificial, captación del ambiente y respuestas en tiempo real fusionados con el interés que genera poder realizar pruebas y analizar resultados de manera automática en una actividad lúdica.

Resumen:

Robótica Simulada:

En líneas generales, el ejemplo de robótica simulada tratado puede dividirse en 3 elementos esenciales: Un simulador, una estrategia y un proxy.

Una vez que se crea la estrategia en Squeak, cabe destacar que ésta posee un objeto llamado “environment” que se encarga de la comunicación con el proxy, es decir, recibe los datos del partido y en base a dichos datos, manda la nueva velocidad de las ruedas de los robots de ambos equipos.

Con el environment conectado y preparado para el envío y recepción de información se abre el simulador RobotSoccer y se carga la dll para que actúe de proxy. Dicho proxy toma la información del simulador y la envía en un string por un socket para ser capturada desde cualquier plataforma. El proxy fue desarrollado por la Universidad del Comahue.

Para la creación de una estrategia se debe tener en cuenta las 4 clases fundamentales:

- Environment: Se comunica con el proxy.
- Robot: Representa un robot de nuestro equipo.
- OpponentRobot: Representa un robot del equipo contrario.
- Ball: Representa la pelota.

¿Cómo funciona el environment?

- 1) Cuando el objeto se crea, inicializa también todos los datos relacionados con el partido (los robots, la pelota, etc.), luego espera hasta recibir el mensaje `#start`.
- 2) El método `#start` crea un nuevo proceso que ejecuta el método `#dataArrival` y lo guarda en una variable de instancia.
- 3) El método `#dataArrival` recibe los datos que envía la dll en un string y ejecuta el método `#parser`: pasando como parámetro un String con todos los datos.
- 4) El parser actualiza los datos de los robots, la pelota y las demás variables del partido y llama al método `#estrategia`.
- 5) El método `#estrategia` es el método más importante porque aquí se decide la nueva velocidad de las ruedas de los robots y se modifican para definir su comportamiento. Luego de eso, este método llama a `#enviarMovimientos`, que básicamente crea el string con las velocidades de cada robot y lo manda a la dll.
- 6) Finalmente se ejecuta el método `#start` volviendo a empezar el ciclo.

Robótica Física:

En este caso, el simulador es reemplazado por un sistema de visión que captura el ambiente y envía también por socket un string con el estado del ambiente. En el caso de la velocidad de las ruedas, en vez de enviar la información por otro socket, el paso de mensajes se realiza directamente desde Squeak hacia los robots vía radio o Bluetooth. Es decir, tenemos un único framework donde sólo se modifica la capa de comunicaciones.

La visión es global y el escenario es captado por una cámara asociada a un sistema de visión. Al ser robótica física se necesita establecer una correspondencia entre centímetros reales de la cancha a píxeles. Para ello es necesario armar una alfombra calibradora con una cantidad determinada de cuadrados de igual tamaño y separados en intervalos iguales. El software elegido para la captura del ambiente fue el Doraemon y se ejecuta bajo Linux. Una vez hecha la correspondencia, se pasa a calibrar los colores de equipo y luego de cada robot. Es indispensable tomar en cuenta factores como la posición y el ángulo de la cámara y sobre todo elegir una iluminación adecuada y que no varíe para no tener problemas ulteriores. Razón por la cual es conveniente antes de calibrar, elegir primero los colores de los parches que se le colocarán a los robots y el color de la pelota.

Es muy importante también tener en cuenta las variables físicas (iluminación, rozamiento, choques, pérdida de batería, etc.) a la hora de armar y probar la estrategia ya que al ser implementada puede no comportarse de manera deseada.