Design of an Omnidirectional Linear Camera
Diseño de una Cámara Omnidireccional Lineal

Ana V. Contreras García, Viridiana Demetrio Aguirre
Jorge Rivera Andrade Gulliver Paredes Aguirres
Bruno Lara Guzmán*

Facultad de Ciencias
Universidad Autónoma del Estado de Morelos
Av. Universidad 1001 Col. Chamilpa C.P. 62210
Cuernavaca, Morelos

Resumen

This article describes the design and implementation of a robot simulator. The robot lives in a 2-D world formed of pixels and has a visual and a tactil sense. The visual sense is provided by an linear omnidirectional camera. The tactil sense is provided by a binary collision bumper. This simulator is integrated as an important part in the research of areas such as Cognitive Robotics and Evolutionary Robotics.

Este artículo describe el diseño e implementación de un simulador de un robot. El robot vive en un mundo bidimensional de pixeles y cuenta con un sensor visual y un sensor táctil. El sentido visual se simula con una cámara omnidireccional y lineal, el sentido táctil es simulado con un sensor binario que indica al robot si existe una colisión o no. Este simulador se integra como una parte importante en la investigación en áreas tales como robótica cognitiva y robótica evolutiva.

keywords: cognición embebida, agentes autónomos artificiales, simuladores de robots, cámara omnidireccional

1. Introducción

El proyecto que se reporta en este articulo esta enmarcado en el campo de *cognición embebida* que representa un nueva forma de pensar en el área de Inteligencia Artificial [(7)]. Cognición

*Autor de contacto: bruno.lara@servm.fc.uaem.mx

J. Crispín Zavala-Díaz, Marco Antonio Cruz-Chávez (eds.). AGECOMP2005, Memorias del 4to Congreso de Cómputo AGECOMP, ISBN(e) 968-878-250-5, pp. 37-45

embebida sostiene como principio básico que los agentes tienen un cuerpo e interactúan dinámicamente con su hábitat.A través de esta interacción los agentes entienden el ambiente en el que se desarrollan.

De acuerdo a las ideas en *Modelaje Sintético*, se intenta que hipótesis y teorías funcionen dentro de un ciclo cerrado junto con implementaciones reales e implementaciones en simulación. Teorías de las ciencias cognitivas se tratan de probar y corroborar en agentes simulados, lo que permite llevar a cabo modificaciones y reflexiones acerca de estas. Al mismo tiempo se lleva a cabo la implementación en agentes reales cerrando un ciclo que se espera finalmente nos brinde un mejor entendimiento de los agentes artificiales y reales.

Como primer paso en esta dirección, el objetivo principal de este proyecto es el desarrollo de un simulador de un robot. Este robot vive en un mundo de pixeles, en donde existen obstáculos representados por circulos de diferentes tamaños. El robot vivira en este mundo de pixeles en donde contara con ciertas capacidades sensoriales que le daran una *idea* de su entorno. Este ambiente simulado representa el nicho ecológico de nuestro agente simulado. Es ahí donde nuestro agente se desarrollará y aprenderá a interactuar, lo que nos permite afirmar que este agente esta situado, [(4)], embebido [(8)] y cimentado [(9)].

A diferencia de otros simuladores, el proyecto desarrollado no está casado con una arquitectura específica, lo que permitirá que por medio de él se prueben hipótesis y teorías independientemente de la arquitectura o robot *real* que se quiera usar. Asi mismo este simulador se planea distribuir como software libre, lo que es poco comun en estas implementaciones.

2. Implementación

El robot simulado que se desarrolló cuenta con sensores de choque y una cámara lineal omnidireccional. Estos sensores proveen al robot con un sentido visual y un sentido táctil.

2.1. Cámara omnidireccional

Uno de los principales objetivos en la implementación de la cámara fue que el robot tuviera la capacidad de detectar todos los obstáculos visibles de acuerdo a su posición, la cual está determinada por un par de coordenadas y un ángulo, tomando esta referencia de posición se calcula cuale es el lugar en que se encuentran los obstáculos respecto al robot. Con obstáculos visibles nos referimos a aquellos que no están situados detrás de otros obstáculos.

La parte principal en el desarrollo de la cámara, consistio en calcular dos ángulos por cada obstáculo. El primer ángulo es entre el centro del robot y el centro del obstáculo. El segundo ángulo es entre el centro del robot y la orilla del mismo obstáculo. Este ángulo se calcula con la finalidad de calcular cuál es la anchura que el robot percibe del obstáculo. Como siguiente paso se almacenan los grados de los obstáculos que son visibles en un arreglo. Por último se

desplegan los datos almacenados.

El cálculo de los ángulos entre el robot y cada obstáculo se dificulta dado que en la computadora es más complicado encontrar ángulos, tomando en cuenta que ésta no es capaz de medir ángulos en grados como estamos acostumbrados ($0-360^o$) dado que las funciones trigonométricas necesarias para dichos cálculos sólo están definidas en ciertos intervalos.

Para la implementación, se generó una lista de obstáculos, la cual almacena las coordenadas y el radio de cada obstáculo, ya que éstos siempre son círculos; los datos del robot son almacenados en una clase que contiene las coordenadas del centro, el ángulo de la dirección de su frente y las funciones necesarias para desarrollar lo descrito en el párrafo anterior.

Para realizar los cálculos de la cámara omnidireccional hizo lo siguiente:

1. Trabajamos con un eje coordenado común, ubicado en el centro del robot, mediante el cual se obtuvo el ángulo del obstáculo (α) respecto al robot (ver Figura 1).

2. Calculamos la recta perpendicular de la pendiente anterior que pasa por el centro del obstáculo C (ver Figura 1), para lograr determinar el punto de intersección de esta recta con una de las orillas del obstáculo y especificar las coordenadas de tal punto, denotado como O.

3. Después, se calculó el arcotangente de la pendiente que forma O con el centro del robot, para encontrar el ángulo entre el centro del robot y la orilla del obstáculo, llamado (β).

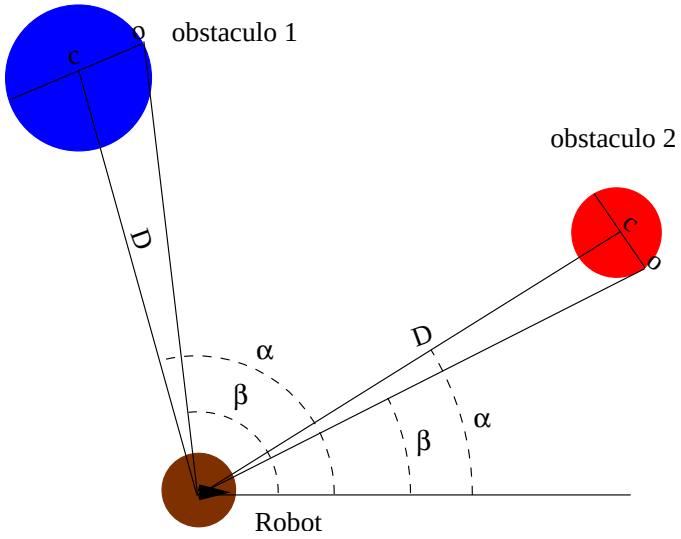

Figura 1: Angulos necesarios

4. Una vez que se obtuvieron α y β se realizó la diferencia absoluta a fin de encontrar en cuantos grados de la vista del robot es percibido el obstáculo, cabe mencionar que esto sólo corresponde a la mitad del obstáculo, esta diferencia es llamada γ.

5. Posteriormente el ángulo α se transforma en absoluto, determinando en que cuadrante del nuevo eje coordenado se encuentra el obstáculo. En particular esta parte del procedimiento es crucial, ya que los ángulos que son encontrados por la computadora van de $-\frac{\pi}{2}$ a $\frac{\pi}{2}$, por lo que no importaba si dos obstáculos estaban en cuadrantes opuestos, a 180^o de distancia, nos daba el mismo ángulo para ambos.

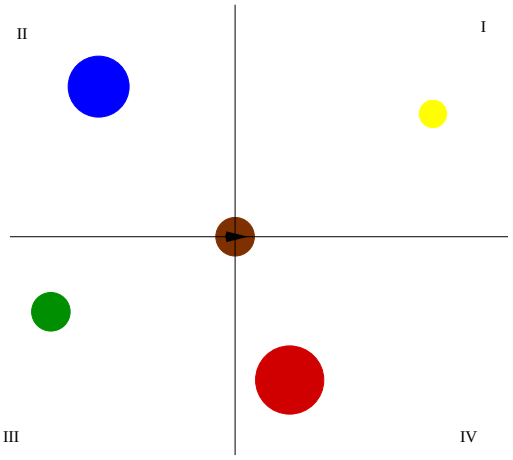

Figura 2: Cuadrantes de la vista del robot

6. Se rotó el eje coordenado ubicado en el centro del robot, de tal manera que el eje positivo de las **x** tuviera la misma dirección que el frente del robot(θ), esto sirvió para determinar el ángulo relativo del obstáculo respecto al robot, llamado $centre$ y que se convertiría en el índice del arreglo.

7. Guardado de los datos en el arreglo

   - Esta etapa fue importante, ya que $centre$ determinaría la posición en la que se guardarían los datos en el arreglo, $centre - \gamma$ y $centre + \gamma$ juntos representan el ancho total del obstáculo que es visto por el robot.
   - Dado que $centre$ es el ángulo del centro del obstáculo siempre está dentro del rango de 0^o a 360^o, pero con $centre - \gamma$ y $centre + \gamma$ no siempre ocurre lo mismo, ya que en los casos en que el centro es cercano a 0^o ó 360^o, al restar o sumar γ, la cantidad puede ser menor o mayor a 0^o ó a 360^o grados respectivamente, a este excedente le denominaremos δ.
   - Lo anterior ocasionó que algunos de los datos del obstáculo salieran de los límites del arreglo. Dado que la cámara es omnidireccional había que simular esta característica en el arreglo, por lo que se hizo necesario igualar las posiciónes de inicio y fin del arreglo, con el fin de que δ fuera almacenado al final o inicio del arreglo según correspondiera.

Es importante mencionar que la vista o imagen proporcionada por la cámara del robot consiste de un solo pixel, esto es, la cámara proporciona una imagen de 360^o de ancho por un pixel de alto.

2.2. Sensor binario

Este sensor se puede considerar como un parachoques en el robot, su única función es indicar si existe una colisión o no, por lo que tiene dos valores: 0 cuando no hay contacto con algún obstáculo y 1 si es que lo hay.

La implementación consiste en calcular la distancia D en la Fig 1, si esta distancia es menor que la suma de los radios (obstáculo respectivo más el del robot), el valor del parachoques cambia de 0 a 1.

3. Simulador

El simulador consiste de una ventana que contiene el mundo del robot, en donde se pueden poner obstáculos de diferentes colores y tamaños (ver Figura 3).

Cada vez que el robot se mueve ya sea a la izquierda, derecha o adelante, se actualiza la ventana de vistas, los movimientos se hacen de acuerdo a la cantidad de grados y pixeles que se especifica en el simulador.

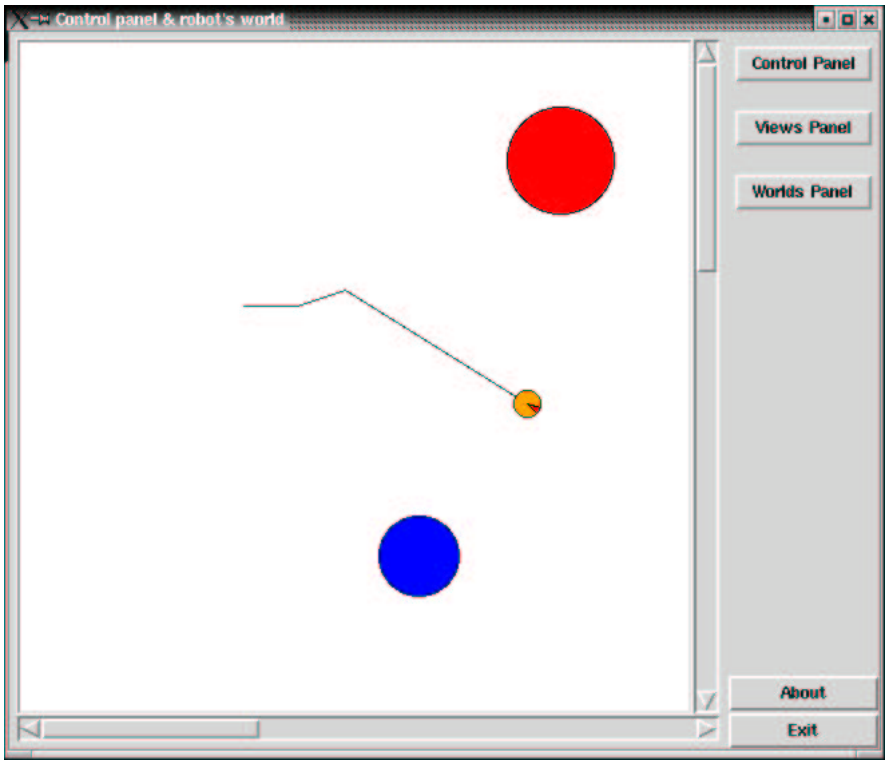

Figura 3: Mundo de pixeles

4. Resultados

En las primeras pruebas se observó que el robot veía los obstáculos como en un espejo, respecto a las coordenadas que estos tenían, por lo que se tuvo que invertir el arreglo de vistas para que los obstáculos desplegados tuvieran la posición correcta.

Es importante notar que para representar los 360^o en la vista se pueden emplear cantidades menores o mayores de pixeles para la representación lineal de los obstáculos sin gran pérdida de información, pues si un obstáculo visible es muy pequeño y lejano al robot, siempre aparece representado por lo menos con un pixel y se guarda en un elemento del arreglo.

El tamaño de la vista en el simulador es independiente del valor que se utiliza para guardar las vistas como arreglos. Como se sabe, en implementaciones reales, es recomendable usar la mayor resolución de los sensores disponibles para que al llevar acabo el procesamiento de los datos la pérdida de información sea mínima.

En la parte izquierda del panel mostrado en la Figura 4 se observa lo que el robot ve. Esto es, se observa la posición de los obstáculos que se encuentran en el mundo del robot con respecto a éste. El eje horizontal representa los 360^o de la vista del robot. La parte frontal del robot se localiza en la parte central de la vista. El eje vertical representa el tiempo, esto es, como va avanzando el robot. La vista en (t_0) se encuentra localizada en la parte superior del panel. A medida que el robot se va desplazando por el mundo, la vista va cambiando, las vistas sucesivas se van mostrando hacia abajo en la Figura.

En la parte derecha del panel se observa la succesión de comandos motrices que el robot ejecuta. De igual manera, la parte superior representa t_0, si el cuadro de la parte izquierda está lleno esto significa que el comando motriz ejecutado fue un giro hacia la izquierda. Cuando el cuadro central se encuentra lleno entonces el robot se movió hacia adelante sin girar. El cuadro de la derecha representa un giro en esta dirección. En la Figura 3 podemos ver que el robot ejecutó una serie de comandos a-a-a-i-i-a-a-a...etc, donde a representa avance sin giro hacia adelante e i representa giro hacia la izquierda.

5. Aplicaciones

Como se mencionó en la introducción este simulador puede ser usado para demostrar, implementar y corroborar teorías de las áreas cognitivas para despueś ser transferidas a agentes reales. Dentro de las áreas de la robótica en las que este simulador planea utilizar encontramos:

- Robótica cognitiva: En esta área se intenta entender el funcionamiento de los procesos cognitivos a través de la investigación e implementación de estos en robots. Específicamente existe interés en estudiar los modelos directos [(2), (1)]. Principalmente usados en el campo de control motriz, un modelo directo (*Forward Model*) es un modelo interno que incorpora conocimiento acerca de cambios sensoriales producidos por acciones re-

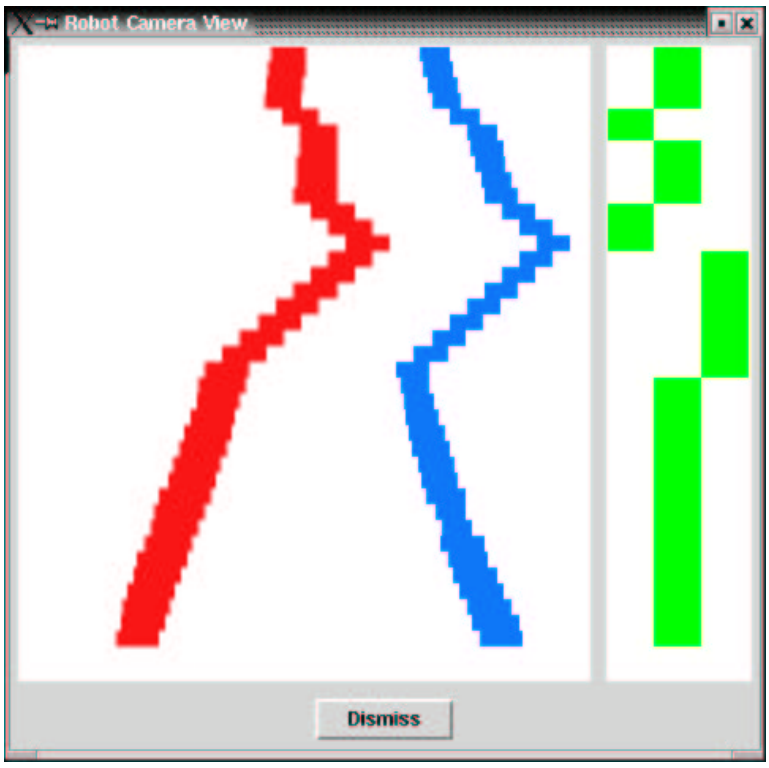

Figura 4: Vista de la cámara omnidireccional del mundo de la Fig. 3

alizadas por el agente mismo. Dada una situación sensorial S_t y un comando motriz M_t (intención o acción real) el modelo directo predice una posible situación sensorial S_{t+1} (ver Fig. 5).

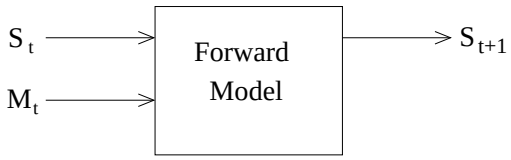

Figura 5: Modelo directo general.

Se espera que los modelos directos provean al agente la capacidad de interactuar con su ambiente através del entendimiento de sus acciones, las consecuencias de éstas asi como la diferenciación entre ellas y las producidas por agentes externos [(3)]. Estamos seguros de que el simulador que se ha implementado tiene las suficientes capacidades para llevar a cabo este tipo de investigación.

- Robótica evolutiva: En esta área se desarrollan controladores *cerebros* para agentes a través de procesos evolutivos [(6), (5)]. Por medio del diseño de una función de aptitud se evalúa el comportamiento de un número de individuos a través de muchas generaciones. De una generación se seleccionan un cierto número de individuos de acuerdo a diferentes criterios (e.g. selección de los más aptos, de ruleta, por torneo, etc.), para así formar

la siguiente generación. Este proceso continúa hasta que se obtiene el comportamiento *deseado* en los individuos.

6. Conclusiones

El simulador desarrollado en este proyecto permite que se agregen extensiones que a su vez facilitaran la investigación en las diferentes areas de robótica arriba mencionadas. Es importante notar que dado que el robot simulado no esta casado con arquitectura alguna permite la implementación y prueba de diferentes hipótesis que mas adelante se podran implementar en robots reales.

Las características principales de este robot es la simulación de un sensor binario y una cámara lineal omnidireccional. La cámara es omnidireccional ya que le permite al robot tener una imagen de 360^o del mundo y lineal por que sólo *ve* un pixel. El sensor binario provee al robot con un sentido tactil, esto es al moverse le da información con respecto a colisiones.

Dentro del marco de cognición embebida, modelaje sintético e inteligencia artificial, este simulador representa un importante y sólido primer paso para llevar a cabo la implementación de diferentes ideas y teorías enfocadas al diseño de Agentes Artificiales Autónomos Inteligentes.

Referencias

[1] BLAKEMORE, S. J., GOODBODY, S. J., AND WOLPERT, D. M. Predicting the consequences of our own actions: The role of sensorimotor context estimation. *The Journal of Neuroscience 18*, 18 (1998), 7511–7518.

[2] BLAKEMORE, S. J., WOLPERT, D., AND FRITH, C. Why can't you tickle yourself? *Neuroreport 11* (2000), 11–16.

[3] BOSBACH, S., COLE, J., PRINZ, W., AND KNOBLICH, G. Understanding another's expectation from action: The role of peripheral sensation. *Nature Neuroscience* (2005).

[4] BROOKS, R. A. Elephants Don't Play Chess. *Robotics and Autonomous Systems 6* (1990), 3–15.

[5] HÜLSE, M., ZAHEDI, K., AND PASEMANN, F. Representing robot-environment interactions by dynamical features of neuro-controllers. In *Anticipatory Behavior in Adaptive Learning Systems, LNAI 2684*, M. Butz, Ed. Springer, Cambridge, MA, 2003, pp. 222–242.

[6] LARA, B., HÜLSE, M., AND PASEMANN, F. Evolving different neuro-modules and their interfaces to control autonomous robots. In *World Multiconference on Systemics, Cybernetics and Informatics 2001* (2001), vol. IX, pp. 259–264.

[7] PFEIFER, R., AND SCHEIER, C. *Understanding Intelligence*. MIT Press, Cambridge, MA, 1999.

[8] WILSON, M. Six views of embodied cognition. *Psychonomic Bulletin & Review 9(4)* (2002), 625–636.

[9] ZIEMKE, T. Rethinking grounding. In *Understanding Representation in the Cognitive Sciences*, A. Riegler, M. Peschl, and A. von Stein, Eds. Kluwer Academic Publishers, New York, 1999, pp. 177–190.