Implementación de la estrategia de juego "Tic-Tac-Toe" para la interacción con un brazo robótico

Dora-Luz Almanza-Ojeda[1], Francisco Luna-Rodríguez[1], David Isaac Hernández Santos[1], Esteban Pérez Flores[1] y Mario-Alberto Ibarra-Manzano[2]

[1] Universidad Politécnica de Guanajuato,
Av, Universidad Norte S/N Comunidad Juan Alonso Cortazar, Guanajuato, México, 38483
luzdora@ieee.org
{luis.luna.rdz,disaachs,steve.epf}@gmail.com
http://www.upgto.edu.mx
[2] Universidad de Guanajuato,
Carretera Salamanca-Valle de Santiago Km.3.5+1.8Km, Salamanca, Guanajuato, 36885, México
ibarram@ugto.mx
http://www.ugto.mx

Resumen En este artículo se presenta un sistema para la interacción hombre-máquina, el cual consiste de un algoritmo que permite a un robot manipulador jugar el clásico juego de "Tic-Tac-Toe" con un humano. El estado actual del tablero de juego es proporcionado al robot mediante una cámara color. El análisis de imágenes adquiridas proveen al robot la información sobre el desarrollo del juego. Se describe la sincronización entre el módulo de control del robot, el sistema de visión y el algoritmo desde una estación de trabajo. El análisis de desempeño del algoritmo y del sistema integrado en funcionamiento es presentado. Al final del documento se presentan las conclusiones y perspectivas.

Palabras clave: Robótica, algoritmo Tic-Tac-Toe, visión por computadora, interacción hombre-máquina, sistemas inteligentes.

1. Introducción

La automatización de tareas en robots manipuladores es un área muy explotada a nivel industrial y de investigación. Dichas tareas requieren diversos conocimientos de ingeniería, como son programación, control, procesamiento de señales, procesamiento de imágenes, entre otras. En el interés de acercar el desarrollo de proyectos multidisciplinarios en la formación de nuevas generaciones de ingenieros, comúnmente, se involucran plataformas robóticas y la programación modular con otros dispositivos. Aunque el desarrollo de inteligencia artificial para

robots ha sido bastante aplicada recientemente con fines educativos, nuevas estrategias de solución e implementación de sistemas inteligentes, permiten que siga siendo un recurso importante para la enseñanza.

Proponer estragías de interacción entre sistemas robóticos y personas, en este caso, para realizar un juego de "Tic-Tac-Toe", resulta en un proyecto motivante que muestra de forma sencilla, pero completa, las diferentes etapas involucradas para alcanzar el objetivo. A pesar de que, este proyecto ha sido ya presentado para diferentes tipos de robots manipuladores, aún continúa el interés por presentar estragías diferentes y mejoradas de estos sistemas, gracias a la mejora continua del desempeño en los equipos de cómputo. Este hecho permite proponer algoritmos de desición basados en el análisis de grandes cantidades de información, para el diseño de sistemas inteligentes más robustos.

En este trabajo se describe el desarrollo de un sistema robótico para jugar "Tic-Tac-Toe" con una persona, utilizando información visual. Este sistema involucra, desde la programación de un algoritmo de solución al juego, para la toma de decisiones en tiempo real, hasta la integración de diversos módulos y su sincronización para el funcionamiento satisfactorio del sistema.

2. Trabajos relacionados

Un aréa importante en el desarrollo de la inteligencia artificial son los algoritmos genéticos. En [2] y [5], los autores proponen una estrategia de juego "Tic-Tac-Toe" basada en algoritmos genéticos, con el fin de evolucionar la población inicial hacia la optimización deseada, en este caso, una estrategia donde el algoritmo no pierda. En este trabajo, los autores aprovechan la simetría del tablero para reducir de manera considerable el número de estados posibles en el juego. Sin embargo, a pesar de dicha reducción alcanzada, el uso de estas estrategias no es posible en sistemas que requieren respuesta en tiempo real.

Trabajos presentados en [1] y [7], muestran también la interacción del robot con una persona para realizar una partida de juegos de mesa. Estos proyectos son descritos como motivantes y muy apropiados para el desarrollo académico de estudiantes en ingeniería. Con el mismo fin académico, en [3] y [6] los autores se enfocan en el manejo del robot a distancia a través de una plataforma web. Se sabe que tales plataformas robóticas educativas, no importando que tan sencilla y adaptable sean en su uso, son principalmente cerradas a la programación abierta hecha por el usuario, limitando el control del robot de forma externa. Recientemente, en [4], los autores proponen una librería de tipo código abierto, mejor conocida como "Open Source", con la cual es posible desarrollar el control de cada articulación del robot y con ello implementar trayectorias especificadas por el usuario.

Por otra parte, aunque varios sistemas robóticos de interacción proponen el análisis de imágenes para conocer la información sobre el tablero de juego, la descripción detallada de las técnicas de procesamiento de la imagen es sólo muy ligeramente introducida. Parte de este trabajo consiste en presentar una

propuesta sobre el análisis de las imágenes y la entrada de datos al sistema robótico que eviten el problema de cambios de iluminación y calibración [1].

Generalmente, la interacción de manipuladores con personas ha sido presentada para diferentes plataformas. Sin embargo, en nuestra experiencia las soluciones propuestas anteriormente, pueden ser todavía mejoradas en el aspecto de desempeño y el modo de procesamiento de la información de entrada, es por ello que decidimos exponer nuestras propuestas al respecto.

Los principales aspectos discutidos en este artículo son:

- Análisis de imágenes
- Algoritmo de juego
- Comunicación entre módulos
- Desempeño

3. Diagrama global del sistema

Nuestro sistema de juego de "Tic-Tac-Toe" en interacción con un brazo robótico, propone el uso de un tablero divido en 9 celdas formadas por 3 renglones y 3 columnas. En lugar del clásico simbolo del jugador "X" ó "O", se utilizan fichas de 2 colores diferentes, cada color distingue al jugador. Además, el desarrollo de una partida de juego, requiere una computadora, una cámara, un tablero y las fichas de juego y un robot y su controlador. La relación de estos queda descrita en el diagrama esquemático de la figura 1.

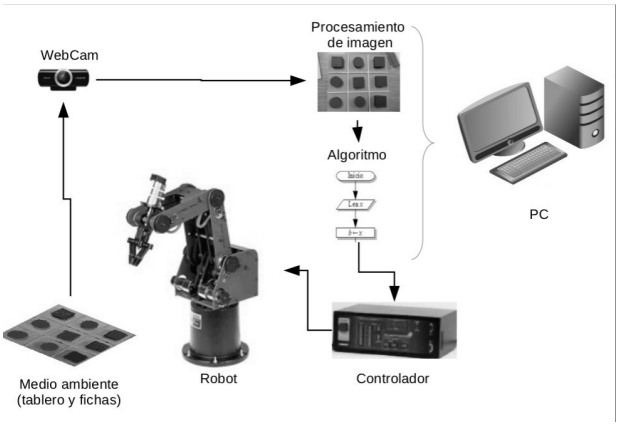


Figura 1. Diagrama global del sistema

La imagen capturada desde la cámara web proporciona el estado del tablero. El procesamiento de la imagen se lleva a cabo en la PC mediante algoritmos de filtrado de imágenes para la detección del color, lo cual va constituir la entrada de datos al algoritmo. Una vez que la estrategía de la siguiente jugada a realizar

con el SCORBOT es definida por el algoritmo, entonces se envían las órdenes correspondientes al controlador del robot. Cabe mencionar que el tablero fue diseñado de acuerdo con el espacio de trabajo del robot, se realizaron fichas de diferentes colores con los cuales se realizaron las pruebas del sistema completo. Finalmente, se utilizó MatLab para desarrollar la programación del algoritmo y el procesamiento de la imagen. Los movimientos del robot para colocar las fichas sobre el tablero, fueron codificadas usando el lenguaje propio del SCORBOT. La sección siguiente presenta una descripción general de cada módulo del sistema.

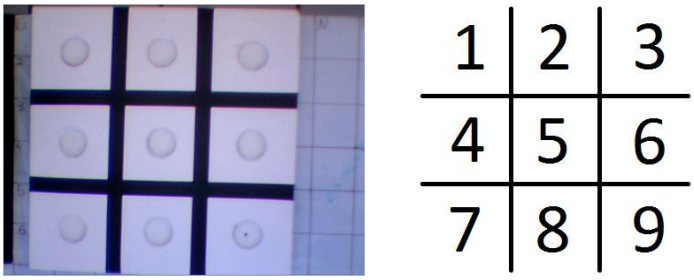
4. Descripción modular del sistema

La integración del sistema requiere de los siguientes componentes: el módulo de visión, el algoritmo de juego, el módulo del controlador y el robot. El algoritmo de juego puede definirse como el más importante de todos, ya que permite que nuestro robot se comporte como un sistema capaz de tomar una decisión de manera autónoma, dependiendo del escenario de juego. La estrategia de juego se basa en realizar jugadas de ataque sobre el centro y las esquinas, debido a que éstas permiten una más alta probabilidad de triunfo sobre el resto de las casillas en el tablero. Así, el centro permite realizar 4 de las 8 posibles jugadas de gane, siendo ésta la más alta probabilidad dada por una casilla. Después, las casillas de las esquinas permiten realizar jugadas de triunfo en sentidos ortogonales o sobre la diagonal. Finalmente, el resto de las casillas permite solo triunfos en sentidos ortogonales. A continuación, con el fin de describir los diferentes módulos del sistema y la dinámica de juego, nos referiremos a los jugadores como el usuario (jugador "O") y el SCORBOT (jugador "X").

4.1. Detección de fichas en el tablero mediante análisis de imágenes

El propósito de un sistema de visión consiste en cuantificar, de manera automática y sin intervención humana, las características físicas de un objeto y extraer información del entorno para controlar un proceso. Éste queda conformado por una o más fuentes de iluminación, una cámara y su óptica y un equipo de cómputo para realizar el procesamiento de la imagen. En este proyecto aprovechamos las ventajas de usar un sistema de visión para adquirir información del escenario de juego. La figura 2(a) muestra una imagen del tablero vacío adquirida desde la cámara. La detección inicial de los bordes permite identificar los limites de las nueve regiones de interés. Tales regiones son enumeradas como se muestra en la figura 2(b) y son utilizadas en el algoritmo de decisión.

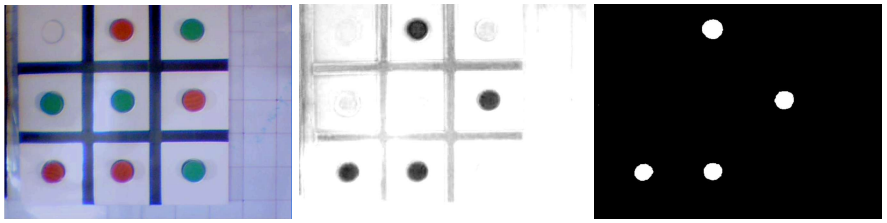
La figura 3 muestra un ejemplo del tablero en algún turno de la partida. De la imagen de entrada inicial, figura 3(a), obtuvimos los valores RGB de las fichas en el tablero. Utilizando la distancia euclidiana de cada pixel con respecto a un modelo establecido del color, se obtiene una imagen de probabilidad, figura 3(b). En esta figura los tonos obscuros representan la probabilidad más alta en similitud con el modelo de color establecido. Los tonos claros en la imagen de probabilidad, indican la presencia del color opuesto al modelo de referencia en



(a) (b)

Figura 2. a) Imagen del tablero de prueba vacío, b) Numeración de casillas del tablero.

el espacio RGB. Finalmente, aplicando una umbralización a la imagen de probabilidad, se obtiene una imagen binaria, en donde, los pixeles blancos indican la posición de las fichas con el color buscado, tal como se muestra en la figura 3(c).



(a) (b) (c)

Figura 3. Etapas del procesamiento de la imagen. a) Imagen RGB inicial, b) Imagen de probabilidad de detección del color (rojo en este caso), c)Máscara binaria final

De la imagen binaria son detectadas las fichas en cada una de las nueve regiones de interés. En base a un conteo del número de "1's" (color blanco) en cada una de las regiones, es posible identificar la presencia de la ficha del modelo de color de referencia. Este proceso es realizado para cada color con el objetivo de detectar, el color de la ficha y en cuál región se encuentra, de esta manera es obtenido el estado actual del tablero.

Con el fin de optimizar el tiempo de procesamiento deben tomarse en cuenta los siguientes puntos:

1. El campo de visión de la cámara debe cubrir solamente el área correspondiente al tablero para optimizar la información en la imagen.
2. El análisis de detección de las fichas debe hacerse solamente en las regiones de interés para reducir el tiempo de procesamiento.
3. El conteo de "1's" para la detección de fichas debe de hacerse de forma horizontal, con el fin de reducir el número de puntos a analizar.

4.2. Algoritmo de "Tic-Tac-Toe" o tres en raya

El algoritmo desarrollado para este sistema es una modificación del propuesto en [8]. Éste plantea una estrategia sencilla pero eficiente, al momento de verificar posibles triunfos tanto del robot como del usuario, evitando una situación de derrota y buscando el posible triunfo del sistema. Sin embargo, realizando un análisis más a detalle de ese algoritmo, se encontró su principal debilidad cuando el usuario comienza jugando en las casillas pares como la 2, 4, 6 u 8. Como ya se mencionó anteriormente, dichas casillas no son de gran peso cuando se juega al "Tic-Tac-Toe", ya que las esquinas y el centro proporcionan mayores posibilidades de triunfo. Esto constituía un lado muy vulnerable del método propuesto en [8], por lo cual debimos agregar ciertas modificaciones, resultando el algoritmo presentado en la tabla 1. Ciertos valores son asignados dependiendo del estado de la casilla, esto es: el número 2 indica vacío, el 3 indica que la casilla contiene una ficha de color rojo y el 5 indica que contiene una ficha de color verde. Por lo tanto, la metodología definida en cada turno dependerá del estado del tablero, para lo cual, el color rojo siempre es asignado al robot.

Tabla 1. Estrategía de juego "Tic-Tac-Toe". Aquí se considera que "X" es el SCORBOT (estado 3 en el tablero) y "O" es el usuario (estado 5 en el tablero).

Turno	Estrategía
1	*mover(n)*, donde n es cualquiera de las esquinas
2	Si casilla[5] está vacía, *mover(5)*, si no *mover(n)* donde n es cualquiera de las esquinas que este vacía
3	*mover(n)*, donde n es cualquiera de las esquinas vacías
4	Verificar si los siguientes pares de casillas están ocupadas simultáneamente: (2,6), (6,8), (8,4), (4,2), entonces mover a casilla 3, 9, 7 ó 1 de forma respectiva. Si en este turno se tiene como estados [3,5,5] en 3 casillas consecutivas o de las diagonales, esto es, donde la multiplicación sea 75, entonces el robot mueve la pieza a cualquiera de las esquinas que se encuentre desocupada. Si *posible_triunfo(X)* no regresa un valor de 0, entonces *mover(posible_triunfo(X))*.
5	Si *posible_triunfo(X)* no regresa un valor de 0, entonces *mover(posible_triunfo(X))* (victoria de "X"), sí no, si *posible_triunfo*(O) no regresa un valor de 0, entonces *mover(posible_triunfo(O))* (se evita victoria de "O"), de otra manera *mover(n)* donde n es cualquier casilla vacía
6	Se repite turno 5
7	Se repite turno 5
8	Se repite turno 5
9	Se repite turno 5

De la tabla 1, identificamos dos subrutinas principales: *mover(n)* y *posible_triunfo*. La subrutina *mover(n)*, donde n es una de las 9 casillas del tablero, indica llevar la pieza a una posición. Por otra parte, *posible_triunfo* es una sub-

rutina que formula un movimiento hacia cierta casilla en la cual el robot pueda ganar. Esencialmente, esta subrutina realiza su evaluación mediante una multiplicación de los estados en el tablero. Al tener una combinación [3, 3, 2], donde su multiplicación es "18", entonces sabemos que el SCORBOT puede ganar, por lo tanto, la salida de la función será el número de casilla en la cual se puede generar el triunfo, si no es así arrojará como resultado un cero. En el caso de obtener un cero en la evaluación del triunfo del robot, esta subrutina es utilizada nuevamente, ahora para evaluar si el usuario "O" puede ganar. El procedimiento es similar al anterior pero en este caso la combinación buscada es [5, 5, 2] equivalente a "50" en la multiplicación. Esta subrutina también arroja como resultado el número de la casilla en la cual se puede generar la victoria del usuario. Puede notarse que el turno 4 requiere más evaluaciones que el resto de los turnos, debido a que en él se ha agregado la verificación de las casillas pares para evitar una derrota del robot, en el caso de que el usuario juegue con esas casillas menos ponderadas en el juego.

5. Robot manipulador: SCORBOT

El manipulador SCORBOT-ER 4U es un robot vertical con 5 grados de libertad y una pinza como elemento terminal para la manipulación de objetos. El robot incluye un controlador USB el cual cumple con varias funciones, destacando la función de actuar como la interfaz entre el manipulador y la computadora. Algunas características relevantes del manipulador son:

- Radio de operación máxima del robot es de 24,4"
- Apertura máxima de la pinza es de 2,6"
- Encoders ópticos en cada eje para posicionamiento.
- Carga máxima 1 kg
- Peso 10,8 kg
- Velocidad máxima de 600 mm/seg

En este proyecto se utiliza el controlador del robot para que ejecute las tareas de llevar la ficha de una posición "a" hacia una posición "b". Donde "a" indica la posición donde el robot toma sus fichas de juego y "b" representa cada una de las posiciones del tablero. Además se tiene indicada una posición, que llamaremos posición de reposo, en la cual el robot espera a que el usuario realice su movimiento y llega a ella después de colocar una ficha en el tablero. Todas estas posiciones son guardadas, y después recuperadas desde un programa hecho en el lenguaje propio del robot para su envió al controlador a través del usb. Cabe mencionar que la decisión sobre cuál posición "a" hacia "b" debe realizarse, es directamente indicada por el algoritmo de juego.

5.1. Interfaz de comunicación entre módulos

Llamamos interfaz de comunicación al módulo de software que indica la secuencia de intervención de los distintos módulos del sistema. La interfaz utilizada

se muestra en la figura 4. El primero de los botones que presenta la interfaz es "Inicia", el cual selecciona aleatoriamente si será el robot o el usuario quién inicie la partida. El segundo botón es "terminar turno" que indica al robot que el usuario ha realizado su movimiento y que puede salir de su posición de reposo, para realizar su siguiente jugada. Cada vez que el usuario indica el fin de su movimiento desde este segundo botón, la rutina del sistema de visión se encarga de capturar la imagen, procesarla y enviar el estado del tablero hacia el algoritmo de juego. El siguiente movimiento del robot es entonces calculado y enviado al controlador para que el SCORBOT realice el movimiento y termine su participación llegando a la posición de reposo. Finalmente el botón "Salir" de la interfaz da por terminado el juego, preparando todo el sistema para una nueva partida.

Figura 4. Interfaz de comunicación

6. Resultados

Una vez integradas las diferentes partes de nuestro sistema (sistema de visión, computadora, controlador y robot), se realizaron diversas jugadas a modo de prueba, observando detenidamente su comportamiento para identificar la existencia de irregularidades durante el desarrollo de la partida. A ejemplo, se extiende la jugada mostrada en la figura 5. Cada imagen representa uno de los turnos desarrollados en esta partida. En este caso, el robot tiene el turno de inicio, sin embargo, como se mencionó anteriormente cualquiera de los dos jugadores puede iniciar.

En este primer turno, el SCORBOT lleva la ficha a la casilla 1. En el turno 2 el usuario coloca su ficha en la casilla 9. El algoritmo analiza el estado del tablero antes de arrojar su posición al SCORBOT, quien tiene el tercer turno 3. Se verifica que el mejor resultado lo obtiene colocando en alguna esquina, por lo que se coloca en la casilla 3. En el turno 4, la persona coloca la ficha en la casilla 2, bloqueando el posible triunfo del robot. Durante el turno 5, el SCORBOT

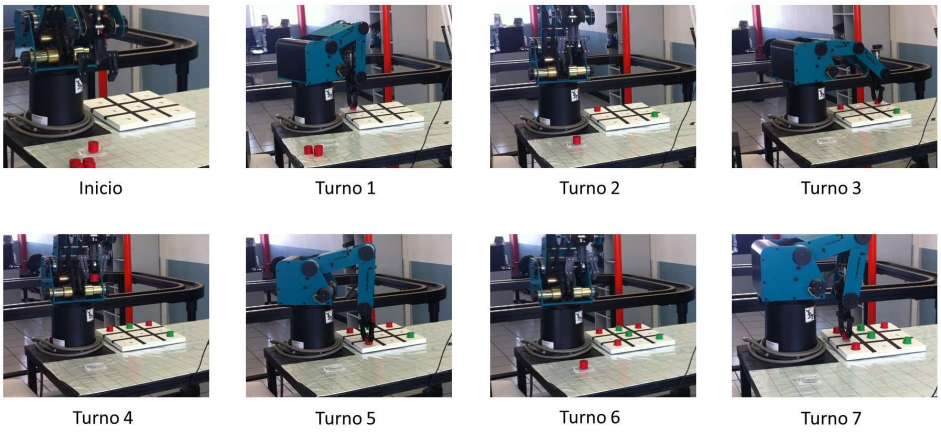


Figura 5. Desarrollo de una partida contra un usuario

lleva la ficha a la casilla 7, asegurando el triunfo con una jugada doble. La persona coloca la ficha a la casilla 5 durante el turno 6, bloqueando uno de los dos posibles triunfos del SCORBOT, sin embargo, el triunfo de éste último ya es inevitable en el turno 7, al colocar la ficha en la casilla 4. El video de esta partida puede ser visto en el enlace: `http://www.youtube.com/watch?v=hxk_h2t8yso`. Muchas otras partidas de "Tic-Tac-Toe" fueron realizadas, iniciando la partida tanto el SCORBOT como el usuario, obteniéndo resultados satisfactorios en el desarrollo de las mismas, logrando al menos el empate del robot.

6.1. Análisis de desempeño del algoritmo

El equipo de cómputo utilizado para la implementación del algoritmo y sincronización con el resto del sistema fue una PC de escritorio con procesador Intel Core i7 2,2 GHz, Memoria cache L3 6 Mb, 8 Gb of RAM y sistema operativo windows XP para mejor compatibilidad con el controlador del robot. Desde la interfaz de Matlab, se realizaron unas mediciones del tiempo de ejecución ocupado por las partes esenciales del programa, (a nivel software) de lo cual encontramos:

- Duración de procesamiento de imagen: $41{,}2ms \pm 4{,}6ms$
- Duración de toma de decisión: $13{,}24\mu s \pm 7{,}9284\mu s$

Con esto garantizamos que toda la parte de código en Matlab, permita la ejecución de la tarea en tiempo real. Aún cuando este software no es de las mejores opciones para programación de sistemas tiempo real, comprobamos que para nuestra aplicación es suficiente. Con este resultado, sólo nos queda mencionar que las principales restricciones de desempeño se tienen a nivel hardware, esto es por el tiempo de adquisición de la imagen y el tiempo de respuesta mecánico

del SCORBOT, en promedio de 100 milisegundos, desde que el usuario indica su fin de turno hasta que el robot inicia su movimiento.

Por otra parte, una de las desventajas a considerar en nuestro sistema ocurre cuando el tablero o la cámara son movidos de su posición inicial seleccionada para los límites de las casillas. Dicho procedimiento, es una rutina que se realiza fuera de línea antes de iniciar la partida, por lo que es recomendable que una vez hecho esto, tanto el tablero como la cámara ya no se muevan de su posición. Sin embargo, de ser necesario, tomaría menos de un minuto realizar nuevamente esta tarea.

7. Conclusiones y perspectivas

En este documento se ha presentado un sistema integral de interacción hombre-máquina que desarrolla el clásico juego de "Tic-Tac-Toe" o "Tres en raya". El sistema involucra diferentes áreas de aplicación como son la visión por computadora, programación de robots y de algoritmos de inteligencia artificial, permitiendo la realización de experimentos reales. La decisión de implementar el algoritmo presentado aquí y no otros, por ejemplo los que mencionamos basados en algoritmos genéticos, surgió de que este algoritmo nos permitiría una ejecución en tiempo real deseada para el sistema, obteniendo un buen compromiso entre tiempo de ejecución y eficiencia a la hora de formular un movimiento. Se pretende mejorar en el procesamiento de la imagen y hacer más robusto el sistema a cambios de iluminación, realizando un cambio de espacio de color, de RGB a CIELab. Por otra parte, se buscará la completa autonomía del sistema, ya sea aumentando la velocidad de adquisición de la cámara o algún otro tipo de sensor por el cual el robot identifique que el usuario ha terminado su turno y evitar dar clic sobre la interfaz. Sin embargo, nuestra principal perspectiva consiste en la puesta en marcha de este sistema, dado su carácter didáctico, en alguno de los centros de ciencias para niños de la región (los cuales cuentan con robots manipuladores en exposición), con el fin de que los niños y el público en general experimenten la interacción y el interés por los sistemas robóticos.

Agradecimientos Este trabajo fue parcialmente soportado por la Universidad de Guanajuato mediante el proyecto titulado "Sistema de Identificación de objetos mediante los atributos del color y la textura utilizando arquitectura reconfigurable con FPGA" con numero 000175/11.

Referencias

1. Aliane, N., Bemposta, S.: Checkers playing robot: A didactic project. Latin America Transactions, IEEE (Revista IEEE America Latina) 9(5), 821 –826 (Sep 2011)
2. Bhatt, A., Varshney, P., Deb, K.: In search of no-loss strategies for the game of tic-tac-toe using a customized genetic algorithm. In: Proceedings of the 10th annual conference on Genetic and evolutionary computation. pp. 889–896. GECCO '08, ACM, New York, NY, USA (2008), http://doi.acm.org/10.1145/1389095.1389269

3. Bicchi, A., Caiti, A., Pallottino, L., Tonietti, G.: Online robotic experiments for tele-education at the university of pisa. Journal of Robotic Systems 22(4), 217–230 (Apr 2005), http://onlinelibrary.wiley.com/doi/10.1002/rob.20061/abstract
4. Esposito, J.M., Wick, C., Knowles, K.: Matlab toolbox for the intelitek scorbot: An open source robotics education library. In: American Society of Engineering Education Annual Conference (2011)
5. Hochmuth, G., Koza, J.: On the genetic evolution of a perfect Tic-Tac-Toe strategy. In: Genetic Algorithms and Genetic Programming at Stanford 2003, pp. 75–82. Stanford University (2003)
6. Jara, C.A., Candelas, F.A., Torres, F.: Virtual and remote laboratory for robotics e-learning. In: 18th European Symposium on Computer Aided Process Engineering. vol. 25, pp. 1193–1198. Elsevier (2008), http://rua.ua.es/dspace/handle/10045/10159
7. Kay, J.S.: From mad libs to tic tac toe: Using robots and game programming as a theme in an introduction to programming course for Non-Majors. In: Twenty-Second International FLAIRS Conference (Mar 2009), http://aaai.org/ocs/index.php/FLAIRS/2009/paper/viewPaper/126
8. Rich, E., Kevin, K.: Inteligencia Artificial. McGrawHill (1994)