UN ROBOT QUE JUEGA A LAS DAMAS

Nourdine Aliane
DARC - Universidad Europea de Madrid,
Villa viciosa de Odón, S/N (28670) Madrid (Spain), aliane@uem.es

Sergio Bemposta,
DARC - Universidad Europea de Madrid, bemposta@uem.es

Diego Gachet
DARC - Universidad Europea de Madrid, gachet@uem.es

Resumen

Este trabajo presenta una aplicación donde un robot juega a las damas contra una persona. Para llevarlo acabo, se ha utilizado un robot manipulador, un sistema de visión, un ordenador personal y un tablero especialmente diseñado para esta aplicación. El control global del sistema muestra como se han integrado diferentes tecnologías como la robótica, la visión artificial y inteligencia artificial.

En este articulo, se presentan los elementos tecnológicos utilizados, la explicación de las técnicas utilizadas y finalmente se dan las directrices para simplificar el proceso de calibración.

Palabras Clave: Robótica, Inteligencia Artificial, Visión artificial, Programación.

1 INTRODUCCIÓN

En este articulo, se va a presentar una aplicación del juego de las damas, donde uno de los jugadores es un robot. Nuestro robot no solamente piensa y calcula las jugadas, sino las culmina realizando el desplazamiento de las fichas como si fuese un jugador humano.

El primer problema al que nos hemos enfrentado es la integración de las diferentes tecnologías [2, 5, 6]. Este problema es cada vez más creciente y la robótica es un ejemplo viviente. En este sentido, nuestra aplicación muestra un grado de integración más que notable como veremos más adelante.

En la literatura no abundan aplicaciones donde un robot participa en un juego. Sin embargo, podemos señalar que el primer robot que juega a las damas fue desarrollado en la universidad de Rochester [7], Oklahoma en 1991. El robot, conocido por el nombre de *LEFTY*, se encuentra hoy en el mueso de "*the Omniplex Science Museum*". El robot está dotado de un sistema de visión y de un sistema de voz. El objetivo de *LEFTY* era mostrar una aplicación con una integración de diferentes tecnologías como la robótica, la visión artificial, la inteligencia artificial, etc.

El resto de la comunicación está estructurado en cuatro apartados: El segundo apartado nos da una visión general de los elementos utilizados en nuestra aplicación. En el tercer apartado, es el corazón de nuestro articulo y describe en detalles los elementos más relevantes del sistema de control: En concreto, veremos los elementos más relevantes en el procesamiento de las imágenes, en la elaboración de las jugadas y la ejecución de los movimientos. En el cuarto y último apartado, se harán unas observaciones sobre la calibración del sistema así como comentarios sobre el funcionamiento general de la aplicación

2 DESCRIPCIÓN DEL SISTEMA

Nuestra aplicación cuenta con un manipulador, un sistema de visión y un tablero con las fichas tal como se muestra el figura 1.

Figura 1: Un robot jugando a las Damas

El robot es un SCORBOT-ER IX de 5 grados de libertad. Todos los ejes son servo-accionados. Tiene

una repetibilidad de ± 0,09mm. Para programar el robot desde otras aplicaciones, se utiliza el modo de directo canalizando los comandos a través del puerto serie. Este modo nos permite invocar la ejecución de un programa previamente descargado en el controlador [4] del robot.

El sistema de visión es una cámara CCD monocromo con una placa digitalizadora de Matrox-Meteor-II conectada al PC mediante un bus PCI. El sistema se complementa con 2 herramientas software para el procesamiento de imágenes: el Inspector y las MILs. El Inspector es un paquete con muchas herramientas de procesamiento de imágenes integradas en un entorno visual fácil de utilizar. Es una herramienta muy productiva para procesamiento fuera de línea. En cuanto a las MILs [8], es una herramienta independiente del hardware y diseñado para facilitar las tareas de programación de aplicaciones de visión. Es una de librería de funciones especialmente diseñada para los entornos de desarrollo como Microsoft Visual C++ o Borland C++.

Finalmente, el tablero y las fichas han sido diseñados para adecuarlos al espacio de trabajo del robot y a las condiciones del sistema de visión. Las fichas son piezas cilíndricas de tamaño suficiente para poder manipularlas con facilidad y pintadas de blanco o negro. Las damas tienen un tamaño mayor. El tamaño ha sido diseñado para permitir al robot manipular las fichas con una destreza suficiente.

3 SISTEMA DE CONTROL

El sistema de control está organizado en varios módulos independientes organizados alrededor del modulo principal tal como lo muestra la figura 2.

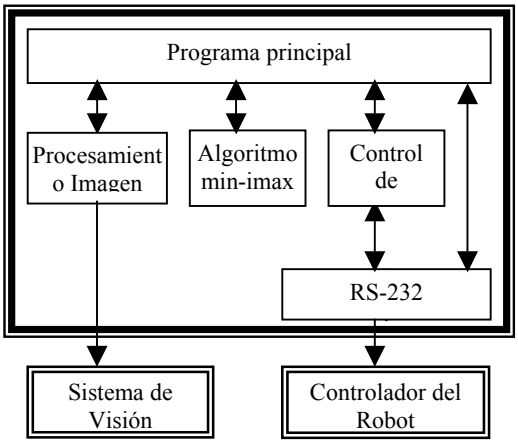

Figura 2: Arquitectura sistema de control

El modulo principal y el modulo de la comunicación con el puerto serie, están escritos en C++. El modulo de procesamiento de las imágenes se basa en las MILs, que en realidad, son llamadas a funciones escritas en C. El modulo que se encarga del control del robot se basa en una serie de programas escritos en ACL (*Advanced Control Lenguaje*) [1] que es el lenguaje de los ESCORBOT. Finalmente, el modulo que se encarga de calcular las mejores jugadas, es un software de libre distribución escrito en JAVA que hemos reutilizado y modificado para adecuarlo a nuestro entorno.

3.1 PROCESAMIENTO DE IMAGES

Para facilitar el procesamiento de las imágenes y tener que realizar un mínimo de transformaciones, la colocación del tablero y de la cámara de visión con respecto del robot es importante. Hemos optado por alinear los ejes x. Esta operación, aparentemente trivial, es de suma importancia ya que simplifica el proceso de calibración. La relación entre los sistemas de referencia queda ilustrada en la figura 3.

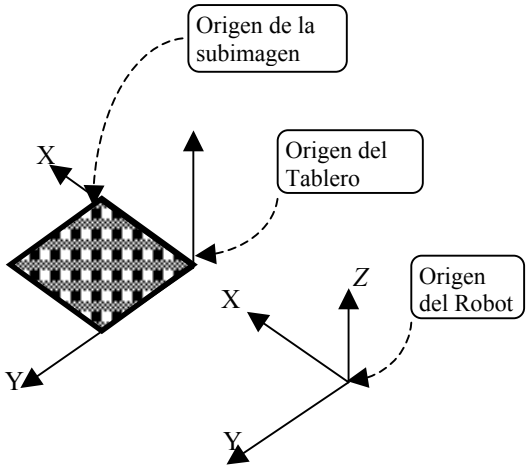

Figura 3: Relación entre los sistemas de referencia

Cuando se recoge una imagen del tablero, antes de procesar para obtener la información relativa a las fichas sobre el tablero, se realizan una serie de operaciones previas. En primer lugar, se crea una sub-imagen que corresponde sólo a la parte que contiene el tablero. Esta operación se realiza fuera de línea utilizando el Inspector, y consiste en la obtención de una esquina de la sub-imagen, su ancho y su alto. Para hacer coincidir la posición (0, 0) de la sub-imagen con el origen del tablero, algunas rotaciones, en memorias, de la imagen son necesarias. Esto depende del convenio utilizado para indexar las posiciones del tablero. En nuestro caso una de rotación de 90º es suficiente para realinear la imagen en memoria.

Para identificar la posición y el tipo de las fichas (blancas o negras, peón o dama), se realizan varias operaciones.

En primer lugar, para separar las blancas de las negras, se realizan dos operaciones de umbralización seguidas de las correspondientes operaciones de binarización. Los valores umbrales, se obtienen mediante un análisis fuera de línea con el Inspector. Cabe señalar que estos valores dependen mucho de las condiciones de luz.

En segundo lugar, para distinguir las damas de los peones, se realiza un análisis de contornos (o análisis de blob) aislando los objetos que cumplen una serie de condiciones como el área, perímetro, etc. En nuestro caso, el criterio del área resulta suficiente. El análisis permite obtener la posición de las fichas en píxel que luego se traducen en posiciones dentro del tablero.

0	Sin Ficha
1	Ficha Negra
2	Dama Negra
3	Ficha Blanca
4	Dama Blanca

3.2 LA ESTRATEGIA MINI-MAX

Como hemos comentado antes, después del procesamiento de la imagen, la información del tablero se traduce en una matriz (8x8) guardada en un archivo.

El modulo mini-max se encarga de calcular la jugada óptima a partir de una distribución determinada de fichas sobre el tablero.

Las técnicas y algoritmos que se usan en los juegos-ciencia entre 2 personas, como las damas, el ajedrez, o el tres en raya son básicamente los mismos [10], cambiando solo alguna particularidad del juego tratado.

Para nuestra aplicación, hemos elegido la estrategia mini-max, reutilizando recursos disponibles [9] en la red. El algoritmo mini-max consiste en generar un árbol de todas las jugadas posibles a las cuales se asocia una puntuación. El objetivo consiste en seleccionar la jugada con mejor puntuación. La variante del algoritmo con poda alfa-beta sirve para reducir el espacio de búsqueda eliminando aquellas ramas del árbol que no van a producir mejores resultados. El algoritmo tiene un parámetro que establece la profundidad del árbol y es el factor que nos permite establecer un compromiso entre tiempo de búsqueda y calidad de las jugadas.

Una vez ejecutado, el algoritmo mini-max genera unos resultado que hay que adecuar a nuestra aplicación. Utilizando, una sincronización mediante ficheros, el resultado se guarda en un fichero con el siguiente formato:

Mov *a b c d*	Mueve la ficha desde la posición (*a, b*) hasta la (*c, d*)
Ret *a b*	Retirar la ficha en (*a, b*)
Dab *a b*	Poner una dama en (*a, b*)

Donde *a, b, c* y *d* sirven para indexar las posiciones del tablero.

El resultado de una jugada puede resultar en varias jugadas elementales como, por ejemplo, comer 2 peones, culminar una dama, retirar las fichas ganadas, etc.

3.3 CONTROL DEL ROBOT

Este módulo tiene como objetivo realizar las jugadas contenidas en el archivo generado por el algoritmo mini-max. La idea consiste en traducir el contenido del archivo y ver que tipo de comando hay que realizar.

Un ejemplo de jugada generada por el algoritmo tiene el siguiente aspecto:

Mov 1 2 3 4
Ret 2 3

El primer comando indica mover la ficha de la posición (*1, 2*) a la posición (*3, 4*) y el segundo comando nos indica retirar la ficha que se encuentra en la posición (*2, 3*).

Para implementar esta tarea, hemos optado por el criterio de transmitir un mínimo de información por el puerto serie. Por un lado, se han definido en la memoria del controlador 4 variables globales *a, b, c y d*, para poder indexar 2 posiciones del tablero; o sea las posiciones (*a, b*) y (*c, d*). Por otro lado, hemos escrito un programa ACL nombrado *CGDJ* que recoge una ficha en la posición (*a, b*) y la deja en la posición (*c, d*). Este programa se encuentra previamente descargado en el controlador del robot.

Para realizar efectivamente la jugada, hemos de canalizar los comandos correspondientes por el puerto serie. Así para el comando anterior, (*Mov 1 2 3 4*) los comandos son:

```
Send_Com ("set a = 1");
Send_Com ("set b = 2");
Send_Com ("set c = 3");
Send_Com("set d = 4");
Send_Com("run CGDJ");
```

Después de realizar un análisis de los diferentes movimientos en el juego de las damas, hemos identificado 10 movimientos elementales. Para cada

movimiento, se ha escrito su programa ACL correspondiente. Estos movimientos son:

1. *INIT* : Inicializar el robot y moverlo a la posición de reposo.
2. *CGDJ*: Coger una ficha en la posición (*a, b*) y dejarla en la posición (*d, c*).
3. *CJMV*: Coger una ficha en (*a, b*) e ir a (c, d) sin soltarla la ficha.
4. *MVMV*: Ir de la posición actual a la posición (*c, d*), sin soltar la pieza para seguir moviéndola.
5. *MVDJ*: Ir a la posición (*c, d*) y deja la ficha en dicha posición.
6. *MVRE*: Retirar una ficha propia en la posición actual después de culminar una dama y dejarla en la posición Palet0.
7. *FICHA*: Retira una ficha en la posición (*a, b*) y se coloca en el Palet0.
8. *DAMA*: Coge una dama del Palet1 y la coloca en el tablero.
9. *REPOS*: Lleva el robot a su posición de reposo para permitir el juego al contrario.
10. *FINAL*: Lleva el robot a su posición del Homing.

Palet0 es una posición absoluta que define el punto inicial donde se ordenan las fichas ganadas, colocándose en forma de palet bidimensional 3xn.

Palet1 es una posición absoluta donde se colocan inicialmente las damas blancas del robot y están ordenadas en forma de un palet unidimensional.

Los 10 programas anteriores se han codificados en ACL, se encuentran descargados en la memoria del controlador, listos para ejecución.

La aplicación se realiza dentro de un proceso cíclico, donde se realizan los siguiente pasos:

1. Toma de la imagen y su procesamiento
2. Iniciar el algoritmo mini-max
3. Realizar las jugadas
4. Esperar la señal

Para darle el turno al robot, hemos de pasar la mano sobre un sensor óptico conectado a una entradas del controlador del robot. Un programa ACL vigila constantemente la entrada y cuando detecta la señal avisa al programa principal para iniciar un nuevo ciclo (o una jugada.) El proceso termina cuando quedan solo fichas del mismo color.

4 CONCLUSIÓN

El tiempo total de procesamiento de la imagen tomada del tablero y del calculo de la jugada es del orden de 3 segundos. Este tiempo se obtiene para un nivel de profundidad de 7, que establece un buen compromiso entre rapidez y calidad de las jugadas.

En el algoritmo mini-max, la activación de la poda alfa-beta es efectiva y reduce el tiempo de búsqueda casi en un 50%.

El punto débil de nuestra aplicación es la necesidad de calibración del sistema de forma frecuente. Los factores que más intervienen en esta operación son las condiciones de luz y la no-fijación de la cámara y del tablero. El primer problema se solventa midiendo los niveles de grises para la discriminación de las fichas, siempre cuando se sospecha de la mala discriminación. Y por otro lado, para el reajuste fácil del tablero con respecto del robot, hemos diseñado un programa ACL que permite al robot de posicionarse en las esquinas del tablero.

Finalmente, después de someter nuestro sistema a prueba, en más de 15 partidas, se ha mostrado robusto y el robot ha ganado en todos los casos.

APÉNDICE: Reglas más importantes del juego de las damas [3]

Se juega sobre las casillas blancas del tablero, quedando la diagonal principal a la derecha del jugador. La esquina inferior derecha es de color blanco. La primera jugada se realiza siempre por las fichas blancas. Los movimientos de las fichas son en diagonal, una sola casilla y en sentido de avance. Si una ficha llega a la línea base del contrario, entonces se convierte en dama. La dama se mueve también en diagonal, pero hacia adelante y hacia atrás, recorriendo una sola casilla cada vez. La dama no puede saltar 2 fichas juntas o una ficha de su color. El capturar es obligatorio. Ley de cantidad: Es obligatorio capturar al mayor número de fichas. Ley de calidad: A igual número de piezas, es obligatorio capturar a las de mayor calidad, dama antes que fichas. Un juego se considera perdido cuando se pierden todas sus piezas o no puede realizar ningún movimiento (ahogado).

REFERENCIAS

[1] ACL, Advanced Control Language, V.2.28 For controller–B; Reference Guide, 1995

[2] BARRIENTOS, A., Fundamentos de robótica, Mac-Graw-Hill, 1997

[3] Checker Rules: http://www.triplejump.net/

[4] Controller-B, V.2.28; User Manual, 1999

[5] CRAIG J., Introduction to Robotics Mechanics and control, Addison-Wesley, 1989

[6] GROOVER, M.P, Industrial Robotics, Technology, Programming & Application, Mac-Graw-Hill, 1986

[7] LEFTY: The Checker Playing Robot at the Omniplex Science Museum 1984: http://www.omniplex.org/

[8] MIL Matrox Imaging Library, Version 6.0 User guide, 1999

[9] Mini-Max Strategy: http://www.cs.caltech.edu/~vhuang/cs20/c/applet/Checkers.java

[10] RUSSELL, S., Artificial Intelligence: A Modern Approach, Prentice Hall, New Jersey, 1995