INTELIGENCIA ARTIFICIAL: PRÁCTICA 2

Los extraños mundos de BelKan

Montserrat Rodríguez Zamorano

16 de mayo de 2016

Índice

1. Introducción

El objetivo de este documento es explicar el funcionamiento básico del agente reactivo que he desarrollado en la práctica, así como las funciones más importantes a la hora de recorrer el mapa.

2. Algoritmo de búsqueda

El algoritmo de búsqueda empleado es una variación del ya conocido algoritmo A*. Las funciones más importantes en el algoritmo de búsqueda serán las siguientes:

- `generarCuadroObjetivo`: en función de lo que el agente haya visto, y siguiendo una lista de prioridades, se establece como cuadro al que llegar uno determinado en la vista del agente. Es importante activar una variable `bool` que indique que ya tiene objetivo, porque si no generaría un nuevo cuadro objetivo a cada iteración de la función `AlgoritmoEstrella`. La lista de objetivos prioritarios será la siguiente:

  1. Si la matriz todavía no está orientada, un punto de referencia.
  2. Objetos, si la mochila no tiene más de 3 objetos. Dentro de los objetos, se priorizará coger las zapatillas y en otro caso, el bikini. Sólo se considerarán objetos aquellos que puedan dar puntos a los personajes del mundo.
  3. Si tengo algo en la mochila, y hay una persona a la vista, entregárselo. Como hay pocos personajes en el mundo y todos los objetos que se han cogido son útiles, darle todos los objetos al personaje nos dará puntos.

  Que la mochila tenga como máximo 3 objetos es una técnica para permitir guardar un objeto en la mochila sin quitarnos las zapatillas, por ejemplo (ver función `guardarEnMochila`). Si el agente no puede alcanzar ninguno de esos objetivos prioritarios, generará un cuadro objetivo aleatorio.

- `setCuadrosAdyacentes:` guarda en una lista los cuadros a los que puede pasar. Aquí se descartan los obstáculos. Esta lista se limpiará al final de cada iteración.

- `funcionObjetivo`: valorará cuál de las posiciones de la lista abierta es la más prometedora para llegar al cuadro objetivo, utilizando la distancia de Manhattan.

- `hallarRuta`: el movimiento dependerá de la orientación del personaje. Este método calcula la orientación del personaje y devuelve un vector de acciones que permitirán llegar al cuadro adyacente seleccionado

Todas estas funciones hacen uso de una serie de métodos auxiliares que no se comentarán en este documento.

3. Orientar la matriz

Orientar la matriz es una función importante en el desarrollo de la práctica, por encontrar los puntos de referencia será el objetivo prioritario de nuestro agente. Para orientar la matriz, una vez se tengan los dos `pk`, tendremos la siguiente información:

- Coordenadas “falsas” del primer punto de referencia.
- Coordenadas “verdaderas” del primer punto de referencia.
- Coordenadas “falsas” del segundo punto de referencia.
- Coordenadas “verdaderas” del primer punto de referencia.

Si consideramos la matriz bien orientada como un eje de coordenadas, encontramos que nuestro agente "tiene los ejes de coordenadas cambiados". Para ver la orientación de cada uno de ellos, se calcularán los dos vectores dirección, uno calculado a partir de las coordenadas falsas del punto de referencia, y otro calculado a partir de las coordenadas verdaderas. Una vez hecho esto se nos pueden dar tres casos:

- Los vectores dirección son iguales, es decir, el agente está bien orientado.
- El vector dirección de uno es el otro multiplicado por -1, es decir, el agente está girado 180^{o}.
- Los vectores dirección son ortogonales. Es importante distinguir que puede haber dos casos de ortogonalidad (el agente está girado a la izquierda, o está girado a la derecha).

Es necesario hacer un cambio de variable, para llevar el origen del eje de coordenadas cambiado al eje de coordenadas verdadero. Este cambio de variable se ha hecho a mano y no se adjunta en el documento, pero aparece implementado en el código.

4. Otras funciones

Otras funciones que se consideran fundamentales en el código son:

- `guardarEnMochila`: si no tengo ningún objeto en uso, lo guarda directamente. En otro caso, como queda un hueco libre en la mochila, hace un bucle con `push` y `pop` hasta que el agente quede con el mismo objeto que tenía en uso.
- `devuelveCoordenadas`: calcula, a partir de la orientación del personaje, las coordenadas de un cuadro en una determinada posición en el vector `VISTA_` ó `SURFACE_`.
- `esTransitable`: devuelve `true` si el agente puede pasar por una casilla y `false` si el terreno es intransitable.

5. Comentarios finales

A base de ensayo y error, se han realizado los siguientes cambios y consideraciones sobre el código:

- Se tendrá un contador de pasos, de forma que si el agente ha establecido un objetivo inalcanzable, aleatorio o no, tendrá que llegar a él en menos de ese número de pasos. Si se supera, se generará otro cuadro objetivo aleatoriamente. Este contador se ha establecido en 75 pasos.
- El porcentaje de acierto y la puntuación para el mismo comportamiento del agente genera distintas puntuaciones, por lo que para establecer cuál es el mejor valor para el contador de paso se deben hacer varias pruebas.