Inteligencia Artificial

Práctica 2 [NIVEL 3]
Agente híbrido

Alumno: Ángel Barrilao Bensrhir
DNI: 75170767M
Grupo: A3

Búsqueda de ruta

El método que he escogido es el **algoritmo de dijkstra.**
Lo primero que tenemos que hacer es obtener un punto de origen y el punto de destino. Creamos un nodo que represente al origen y lo insertaremos en una cola de prioridad. Esta cola es ordenada de menor a mayor según la distancia del nodo hasta el objetivo. Esta cola será la **cola de abiertos.**
La distancia la calculamos con la geometría del taxista (también conocida como distancia **Manhattan**).
De esta forma tendremos en el tope de la cola el nodo mas prometedor para llegar al objetivo.
A continuación creamos una matriz booleana que nos servirá para ir marcando si una posición ha sido evaluada y así ahorrarnos cálculos innecesarios. Por defecto, inicializaremos sus posiciones a false.
Ahora tendremos que ir iterando sobre los nodos mas prometedores e ir obteniendo sus vecinos (también los mas prometedores) hasta llegar al nodo solución.
Para ello usamos un bucle que funcionará mientras la cola con prioridad de abiertos no este vacía . Hacemos una copia del elemento que estamos evaluando actualmente (el del tope de la cola). Lo sacamos de la cola de abiertos y lo metemos en una **lista de cerrados.**
Para cada nodo expandimos sus cuatro nodos vecinos (norte, sur, este y oeste).
Y analizamos que cada uno de los vecinos cumpla:

-Que este descubierto en el mapa y su terreno sea transitable.
-Que ya lo hayamos evaluado anteriormente y lo tengamos marcado en la matriz booleana(en caso de que sea true lo obviamos).

Si el nodo se ha evaluado satisfactoriamente lo meteremos en la cola con prioridad de abiertos, para evaluar nuevamente sus vecinos.
Finalmente si el nodo que estamos evaluando es la solución, saldremos del bucle.
Después de haber calculado y recorrido los nodos correspondientes, disponemos de la **lista de nodos cerrados** que es la que nos marca el camino hasta el objetivo.

Recorreremos la lista de cerrados desde su ultimo elemento hasta el primero. Para ir iterando usamos el padre del nodo actual y lo meteremos en una **lista de recorrido.**
Por último recorremos esta lista y calculamos, según las coordenadas de cada nodo cual sería la acción para llevar a cabo el desplazamiento.

Descubrimiento de terreno

El descubrimiento de terreno es imprescindible para planificar una buena búsqueda de un objetivo, por lo que necesitamos tener el máximo de mapa descubierto.
Como solución mas factible a este problema he usado un **campo de potencial artificial** para ello he implementado una matriz de enteros que por defecto lo inicializamos a un valor bajo, en mi caso -1. Conforme vallamos pasando por la casillas del mapa vamos marcando la posición con un valor que se ira incrementando conforme el personaje se valla moviendo gracias a un contador de pasos. Cuando encontramos terreno infranqueable, por ejemplo un precipicio lo marcamos con un valor lo suficientemente grande para no pasar por el.

Con esto conseguimos que el agente valla priorizando las zonas inexploradas.

GPS y mapas

Para poder movernos bien por los mundos de Belkan necesitamos ubicarnos correctamente, para tal fin usamos una brújula para saber en que orientación estamos situados y también para coordinar las filas y columnas que corresponden al mapa.

Sobre los mapas tenemos en total cuatro. Dos de ellos para el terreno , y otros dos para el mapa de potencial artificial.

La razón de usar 4 (realmente son 3 , ya que el mapaResultado viene implementado) es que antes de descubrir la verdadera ubicación necesitamos ir guardando la información que vamos percibiendo para podernos mover de una forma mas eficiente y descubrir mapa. La forma de hacerlo es con mapas "en sucio" para que después, cuando encontremos un punto kilométrico, podamos contrastar la información anteriormente guardada y así no perderla.

Control de acciones penalizantes

Esta es una parte importante ya que nos permite movernos sin peligro por cualquier mapa. Para ello usamos los sensores de colisión y de visión tanto de terreno como de superficie.
Controlamos que el agente no se precipite por el precipicio o se meta en el agua , bosque o muro.
Simplemente debemos de comprobar en cada acción, que no provoque una colisión con el terreno o con un aldeano. Para ello corregimos la última acción tomada.
Por ejemplo si la acción tomada es ir hacia delante y hay un muro, producirá colisión. Esto lo podemos corregir fácilmente girando el personaje.
Lo mismo es aplicable cuando estamos ejecutando las acciones de un plan. Si la acción del plan va a chocar , entonces borramos el plan y tomamos una acción correcta. Si durante la ejecución de un plan se cruza un aldeano ponemos en el tope de la lista una acción de espera, hasta que no veamos al aldeano y prosigamos con nuestro plan.
Hay otros casos que se pueden presentar también relacionados con los aldeanos, y es que nuestro personaje se puede quedar atascado en una zona concreta junto con el aldeano. Esto suele ocurrir en espacios pequeños y escalonados.
Para solucionar esta faena ponemos en nuestro mapa de gravedad ,alrededor de nuestro personaje, unos valores elevados obligándolo a salir y volver por donde vino sin mas opción.