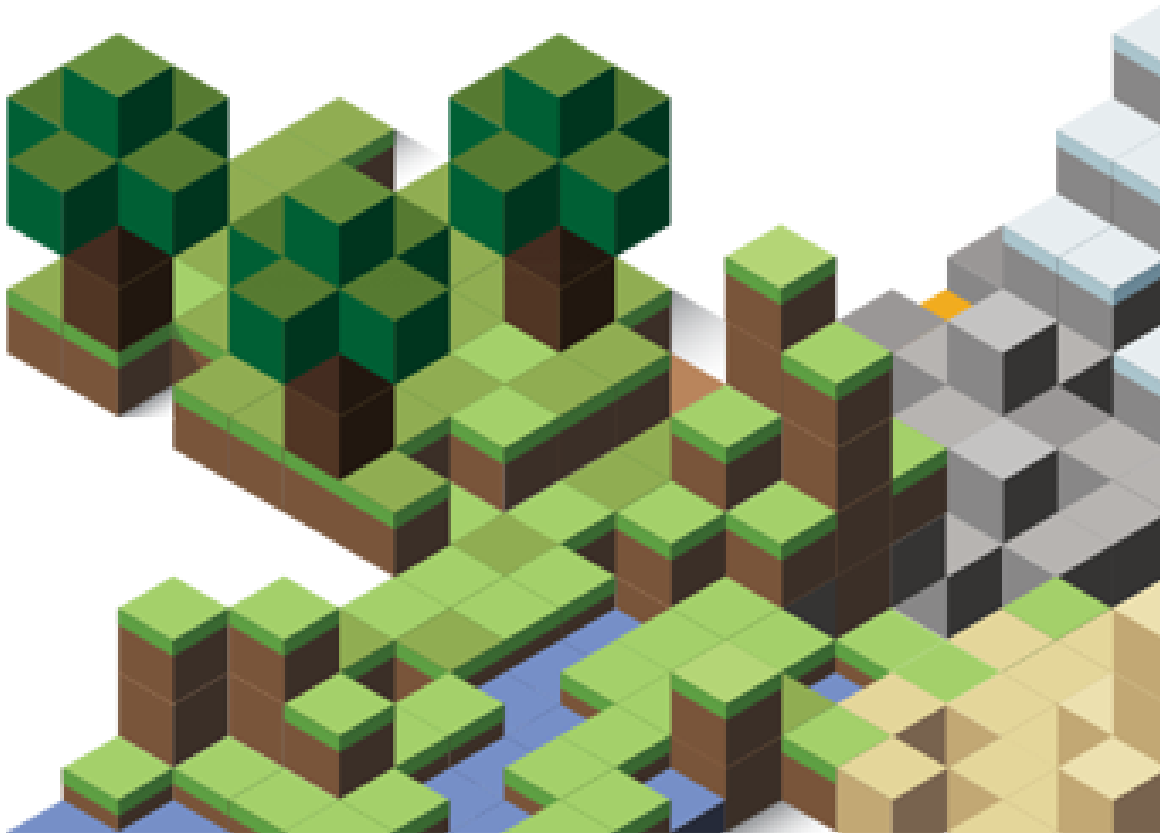
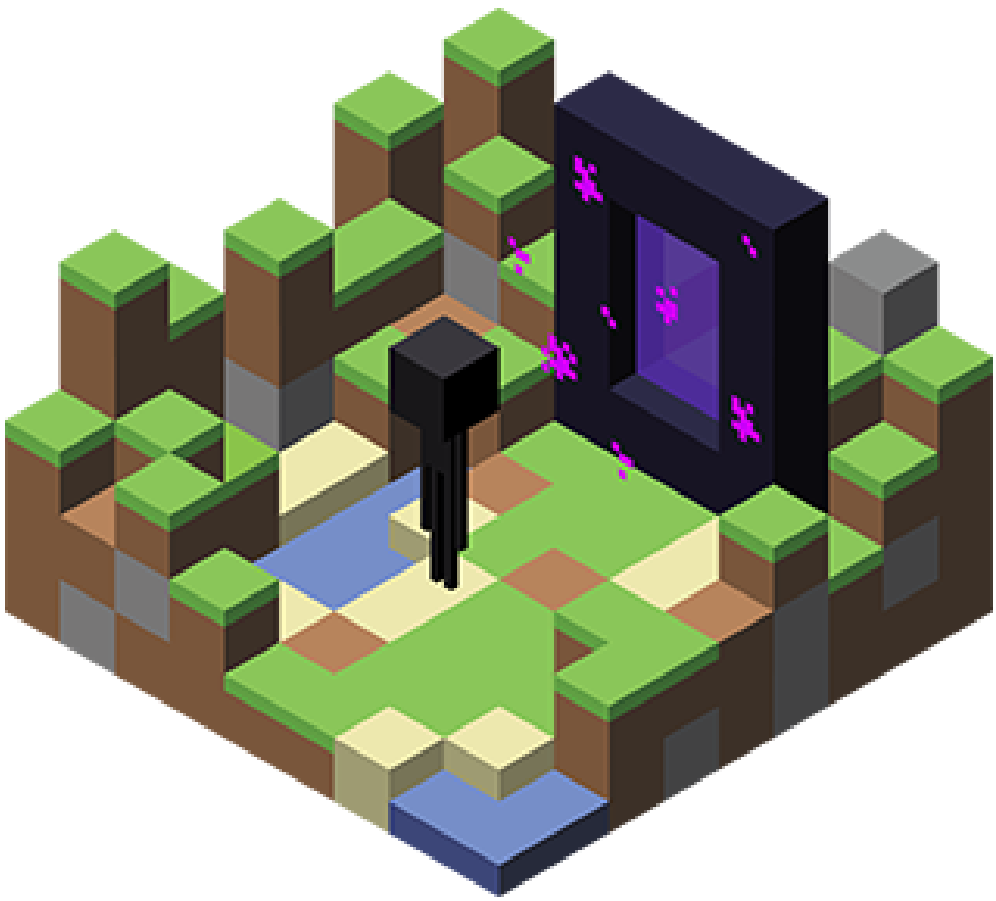
Práctica 1 Inteligencia Artificial

Los extraños mundos de Belkan

David Gil Bautista



Índice



Descripción del problema

Nos situamos en un mundo bidimensional desconocido en el que nuestro agente debe ser capaz de moverse así como evitar obstáculos e interactuar con diversos objetos.

Para la implementación de un algoritmo que permita recorrer este mapa y cumplir sus objetivos se nos da una serie de sensores los cuales muestran información inmediata que percibe el agente en una posición concreta.

Estructura método Think

Para comenzar describiré los sensores más importantes:

- Terreno[] : Es la vista que tiene nuestro agente del mapa en un instante preciso. Este sensor nos muestra si en las casillas frontales hay tierra, agua, bosque… etc.
- Superficie[] : Este vector es una vista que nos muestra si hay presente algún objeto en alguna casilla frente a nuestro agente.
- Reset : Es una variable que se pone como verdadera si el agente ha colisionado o muerto.

Para el comportamiento básico del agente se usarán estos sensores descritos anteriormente.

Lo principal es conseguir que nuestro agente se mueva por el mapa sin colisionar, para ello usamos la función *recorreMapa()* a la cual le damos los valores de la fila y la columna, una matriz de enteros que usaremos para marcar las casillas que recorremos, los sensores y una variable booleana que tiene como fin indicar si el agente puede evitar obstáculos (se verá más adelante) o no.

Acto seguido, comprobamos mediante los sensores si la casilla próxima es agua o bosque, y si tenemos los objetos que nos permitan atravesar estos complicados terrenos.
Si podemos cruzar por dicha casilla llamaremos a la función recorreMapa() otra vez, solo que esta vez en la variable booleana indicaremos que si puede atravesar ese terreno y si no podemos pero tenemos el objeto disponible iremos buscando en la mochila hasta equiparlo y poder seguir andando por esas casillas.

Ahora debemos comprobar si estamos orientados, y si no lo estamos y nos encontramos sobre un punto PK debemos coger el valor de la fila y columna para trasladar todos los movimientos que habíamos hecho sobre una matriz inicial a una con el resultado del mapa.

Después de esto comprobaremos si nos hemos orientado y si es así guardaremos en el resultado la casilla actual, en caso contrario la guardaremos en la matriz inicial que luego copiaremos al orientarnos en la matriz resultado.

Como última condición para tomar una acción está la de coger objetos, en la cual usaremos el sensor de la superficie para comprobar si la casilla de enfrente está ocupada y en ese caso si aún no poseemos ese objeto lo cogeremos.

Definición funciones implementadas

Action recorreMapa(Sensores s, bool puedo){

En esta función comprobamos las casillas adyacentes a nuestro agente para lo cual hacemos uso de la fila, columna y la orientación.

Una vez tengamos los valores de las casillas escogeremos aquella por la cual el agente haya pasado menos veces.

Y en caso de que no se pueda avanzar a una casilla se le dará un valor alto para que el agente evite esa casilla a la hora de elegir a donde moverse.

Para casos en los que el agente tenga un objeto y pueda moverse a una casilla que en principio no podía se hace uso de la misma variable *puedo* para indicar que ahora si puede avanzar a esa casilla.

}

void actualizar(Action last, Sensores s){

Esta función se usa para actualizar las variables de estado en función de la última acción que haya realizado nuestro agente.

}

bool avanza(Sensores sensores){

Esta función mira los sensores del agente y devuelve VERDADERO en caso de que el agente pueda avanzar y FALSO si no puede.

}

Action objetoMochila(){

A esta función la llamamos una vez tenemos disponible un objeto pero no lo tenemos equipado para equiparlo y poder avanzar por el agua o por el bosque.

Es recursiva y trabaja en función de la última acción del agente para ir rotando entre los elementos de la mochila (PUSH, POP) hasta que se equipa el objeto que se necesita y se puede avanzar por la casilla.

}

void pintaBordes(){

Pinta en la matriz resultado los precipicios en las filas y columnas correspondientes.

}

void pintaBosques(){

Pinta en la matriz resultado las casillas que no haya podido capturar y las transforma en bosques.

}

void capturaPK(Sensores s){

Actualiza las variables ***fila*** y ***columna*** y cambia el valor de ***orientado*** a TRUE.

}

void copiaMapa(int f_, int c_){

Se usa al orientar el mapa y gracias a la nueva fila y columna junto con las antiguas desplazamos el mapa provisional que teníamos al resultado.

}

void rellena(Sensores s){

Se usan los sensores para pintar nuestra matriz resultado con el mapa del mundo.

}

Conclusiones

Para concluir he de decir que la práctica no funciona completamente y que puede mejorar hasta límites extremos pero conlleva un trabajo cuyo esfuerzo no merece la pena para la función que va a tener el agente. Si lo consideramos un agente reactivo podemos afirmar que cumple su función, recibe estímulos y actúa, aunque a veces no de la forma más inteligente.

Se esperan depurar los errores para la próxima práctica así como desarrollar un poco más la inteligencia del agente para que sea más deliberativo.