DOCUMENTACIÓN PRÁCTICA 2

Antonio José Blánquez Pérez 2ºD2
INTELIGENCIA ARTIFICIAL

NIVEL 1
En el método en anchura simplemente se ha copiado el método en profundidad y se ha cambiado la pila(stack) por una cola(queue).

En el método con coste uniforme se comenzó cambiando la cola(queue) por una cola con prioridad(priority_queue) introduciendo en esta un comparador. Primeramente se intentó crear un struct donde se realizaban todas las operaciones necesarias, lo cual da muchos problemas(sobre todo en el acceso a algunas variables). Por tanto se optó por añadir un campo al struct estado en el cual apoyarse al calcular el coste de cada nodo y crear una función externa que variara el coste del nodo según su padre y el tipo de terreno. Para comprobar la lista de abiertos se ha usado una estructura set auxiliar.

NIVEL 2
Para el nivel 2 he comenzado usando el algoritmo de coste uniforme centrándome primero en rellenar el mapa, para lo cual he creado un método escribirMapa que crea una correspondencia entre el vector del sensor y la matriz del mapa. Después comencé a cambiar el método think adaptándolo al nivel 2, para lo cual decidí separar con una estructura if lo que ya tenía y los cambios de éste nivel con la intención de no afectar a la corrección del nivel anterior.

He diferenciado el comportamiento del robot entre antes y después de encontrar un punto de referencia, teniendo como primera aproximación una búsqueda reactiva básica(seguir adelante o girar derecha) antes de encontrarlo y, para cuando lo encuentra, se llama a pathFinding para buscar un plan, teniendo en cuenta que hay que recalcular la ruta si encontramos zonas que no podemos cruzar y que si encuentra un aldeano, como primera medida, espere. Con éste punto de partida, se comienza a mejorar algunos puntos clave.

Se ha comenzado añadiendo un valor distinto de 1 para las casillas en blanco, para lidiar con las decisiones de ir por un camino nuevo o uno que ya está explorado. Experimentalmente se ha llegado a un valor aceptable de 4.

A continuación se ha implementado una función que busca en el sensor de terreno una casilla amarilla y, si la hay, traza un plan hasta ella y lo sigue(a no ser que encuentre un muro, en cuyo caso se aborta el plan). También se copia todo lo que ha visto el agente antes de saber donde está en el mapa, que ha sido guardado en una matriz auxiliar, ayudándose de la estructura Action actual, donde han quedado guardadas la coordenadas provisionales, de manera que podemos establecer una relación entre éstas y las coordenadas reales.

Posteriormente se ha ampliado la búsqueda reactiva para que el agente sea capaz de llegar a todos los puntos del mapa utilizando el método bautizado en clase cómo Pulgarcito, añadiéndole la capacidad de quedarse a 3 bloques de los obstáculos para mejorar su eficiencia de búsqueda generalmente.

Por último, se ha añadido una comprobación extra para evitar agua que no vemos aún, de manera que si ve 3 bloques de agua delante suya recalcula el camino, a no ser que decida que sí que tiene que ir por el agua.