Doble Grado Ingeniería Informática y Matemáticas Juan Valentín Guerreo Cano

Memoría de la Práctica 2

La práctica se basaba en combinar comportamientos deliberativos con reactivos principalmente. El nivel 0 se encontraba ya implementado, y por tanto, procederé a explicar la práctica desde el nivel 1.

Nivel 1:

Para implementar el nivel 1 simplemente he modificado el ya implementado algoritmo de profundidad del nivel 0. Concretamente he cambiado el tipo de lista que eran los "Abiertos" dentro del algoritmo. En el caso del algoritmo de búsqueda en profundidad esta lista era una pila, y para poder implementar el nivel 1 correctamente mediante el algoritmo de búsqueda en anchura simplemente he utilizado una cola en lugar de la pila ya mencionada.

Nivel 2:

Para implementar el nivel 2 he decidido utilizar el algoritmo A*. Para ello me he basado en los algoritmos previamente programados aunque con ciertas modificaciones sustanciales. He utilizado colas con prioridad lo que nos permite saber que nodo es mejor que otro dentro de la lista de abiertos. También para poder aplicar correctamente estas comparaciones he sobrecargado los correspondientes operadores del struct "nodo" e incluso he creado un struct nuevo para implementar los nodos A* que tienen ciertos atributos aparte de los que posee un nodo (principalmente los costes).
Aparte de lo ya mencionado, para poder asignar correctamente los atributos de mis nodos A* he creado funciones que asignen adecuadamente los costes de estos: una heurística (la máxima diferencia entre la fila del objetivo y la fila en la que me encuentro, y la columna del objetivo y la columna en la que me encuentro) y una función calcular coste que permite saber cuanto coste supone realizar una determinada acción en el estado en el que nos encontramos.

Cabe destacar que al principio intenté implementar el algoritmo con iteradores pero trabajar con estos sobre conjuntos no me ha traído buenos resultados en niveles posteriores así que procuré evitar usarlos, de tal manera que he conseguido construir adecuadamente el algoritmo A*, que cumple satisfactoriamente los objetivos.

Nivel 3:

Para la implementación del nivel 3 me he basado en el algoritmo implementado en el nivel 2, salvando ciertas peculiaridades del nivel. Básicamente, en este nivel, he generado objetivos aleatorios en el mapa, procurando que cumplan ciertas condiciones para intentar conocer la mayor cantidad de mapa posible, y utilizando el algoritmo A* he perseguido estos mismos.
Aunque esto parezca fácil he tenido que realizar ciertas funcionalidades para poder cumplir con lo dicho: entre ellas a la función del nivel anterior que me calculaba costos he tenido que añadir costos a casillas desconocidas (basándome en la media de costos que supondría la acción en cualquier casilla) y he tenido que realizar una función que almacene lo ya conocido del mapa entre otras cosas.

También cabe destacar que para mejorar el consumo de la batería al trazar un plan y este pasar por un muro o precipicio, o pasar por un determinado número de casillas de bosque (sin zapatillas) o

un determinado número de casillas de agua (sin bikini) he recalculado el plan al objetivo. De esta manera he ahorrado bastante batería, aunque en cambio, en mapas con un tamaño grande esto me lleva a mayores consumos de tiempo en calcular planes grandes constamente. Para evitar el problema de tiempo al crear los objetivos a perseguir he tenido en cuenta entre otras cosas, que la distancia a el objetivo generado aleatoriamente no sea demasiado grande, y que el objetivo sea desconocido.
También cabe destacar que una vez mi personaje conoce el borde en este nivel, presupone el resto, lo cuál me ayuda a ganar cierto porcentaje.

Nivel 4:

Para la implementación del nivel 4 he tomado como base el nivel 3. La mayoría de funcionalidades de este nivel son las mismas que las del nivel 3 aunque he tenido mayor rigurosidad a la hora de trabajar con la fila y columna en la que se encontraba mi agente para poder pintar correctamente el mapa, y he tenido en cuenta las colisiones con aldeanos y lobos. El objetivo principal de este nivel es evitar problemas (fallar en el plan preestablecido) al chocar contra un lobo o un aldeano, para ello mi agente debería haber conseguido localizar cuando un lobo le empuja y en ese caso llamar a la acción actWHEREIS pero no he sido capaz de que mi agente siga este comportamiento. Lo he intentado por medio de un estado anterior que me permita compararlo con el estado actual y en caso de que varíen cuando no deba se interpretaría como un empujón de lobo, además del sensor de colisión que también he utilizado, aunque sin demasiados frutos.