Memoria Práctica 2

Francisco Lopez-Bleda y Manuel Hernández

RESUMEN

Si algo demuestra que un IA existe es que interactúa con su entorno, y una de las formas más primitivas es que es capaz de moverse por su entorno sin cometer errores, pero esto a veces entra en conflicto con los juegos en tiempo real, sea el Age of Empires, ya que calcular la ruta puede ser muy costoso, y puede perderse tiempo, lo que se hace es calcular la ruta limitando los tiempos de búsqueda.

ENUNCIADO

Las condiciones del taller eran las siguientes

CONTENIDOS DEL TABLERO

- El tablero tiene que ser como mínimo de 10x10.
- Hay 3 tipos de casillas que tienen características especiales:
    - Casilla normal, el coste de acceder a la casilla es 1.
    - Casilla embarrada, el coste de acceder a la casilla es 2.
    - Casilla bloqueada, esta casilla es inaccesible.
- Unidades
    - Mínimo de 3 unidades (rojo, verde y azul)
    - No pueden compartir casilla con otra unidad.
    - Solo puede haber una seleccionada.
- Flechas
    - Puede compartir casilla y no bloquean.
    - Son del mismo color que la unidad que se dirige hacia ella.

COMPORTAMIENTO CASILLAS

Todas la interactuaciones son con el clic izquierdo del ratón

Efecto	**Casilla normal**	**Casilla embarrada**	**Casilla bloqueada**	**Casilla con unidad X**
Sin unidad seleccionada	Pasa a ser casilla embarrada	Pasa a ser bloqueada	Pasa a ser normal	Se selecciona la unidad X
Con unidad Y seleccionada	Se añade flecha Y a la casilla		Se deselecciona unidad Y	No pasa nada

RESTRICCIONES

- Solo se puede un único resolutor, que recibe una posición y tiene que calcular una ruta
    - Al calcular la ruta tiene en cuenta la situación actual del tablero, ignorando otras unidades.
    - Usa A* o una variante.
- Que no se interrumpa al jugador o que no lo note.

RESULTADO

En este apartado analizamos y comparamos lo obtenido con el enunciado propuesto, por el orden de las secciones.

CONTENIDOS DEL TABLERO

En este apartado hemos cumplido los requisitos mínimos, no solo eso, sino que los hemos mejorado. El tamaño de tablero es de libre elección, si se cambia en Unity antes de ejecutarlo, pero el estándar es 20x20. Respecto el tipo de casillas hacen lo mismo, pero solo tienen nombres diferentes. Hemos separado la unidad de la casilla, para que la casilla solo sepa si está ocupada o no.

Puede haber un numero grande de unidades, las añades durante la ejecución, en un modo que nos hemos inventado llamado, modo edición, que explicare más adelante. La unidad seleccionada se muestra en el GUI en vez de iluminarse.

En nuestro caso en vez de usar flechas usamos cruces del color de la unidad para indicar su destino si la ruta es posible.

COMPORTAMIENTO DE LAS CASILLAS

Hemos creado un modo edición del mapa para poder editar el mapa, y usando la tecla E vamos alternando entre ese modo y el modo mover.

<table>
<tr><th colspan="2">Efecto</th><th>Botón ratón</th><th>Casilla normal</th><th>Casilla embarrada</th><th>Casilla bloqueada</th><th>Con una unidad X</th></tr>
<tr><td colspan="2" rowspan="2">Modo edición</td><td>Izquierdo</td><td>A embarrada</td><td>A bloqueada</td><td>A normal</td><td>Nada</td></tr>
<tr><td>Derecho</td><td colspan="2">Se crea una unidad</td><td>Nada</td><td>Borra a X</td></tr>
<tr><td rowspan="4">Modo mover</td><td rowspan="2">Con Unidad</td><td>Izquierdo</td><td colspan="3">Se deselecciona</td><td>Selecciona X</td></tr>
<tr><td>Derecho</td><td colspan="2">Se calcula ruta*</td><td colspan="2">Nada</td></tr>
<tr><td rowspan="2">Sin unidad</td><td>Izquierdo</td><td colspan="3" rowspan="2">Nada</td><td>Selecciona X</td></tr>
<tr><td>Derecho</td><td>Nada</td></tr>
</table>

* cuando se calcula ruta y el camino es posible se pone una cruz en el destino de la unidad

RESTRICCIONES

A la hora de calcular el camino, no cumplimos el apartado de ignorar las otras unidades, porque había una contradicción en el propio enunciado, la cual indica que las unidades se bloquean entre si ya que no puede haber una en una misma casilla, pero luego te dice que las ignores a la hora de calcular el camino, nosotros hemos optado por tener cuenta las uanidades a la hora de calcular el camino y también cuando lo está realizando porque puede que una unidad se halla puesto en medio de su camino original, lo que causa que vuelva a recalcular la ruta, desde la casilla actual. Hay un fallo cuando dos unidades van en direcciones opuestas y se encuentran, que lo que ocurre es que se recalcula la ruta, la ejecutan y se bloquean indefinidamente el camino la una a la otra este error lo hemos llamado "*-pase usted. -no, pase usted.*", que es lo que dicen dos personas que les pasa esto en la calle.

Respecto al algoritmo hemos usado un A*, basado en el tiempo que tarda, si tarda más de 16 ms, se interrumpe y luego se continua por donde iba, mientras tanto la ficha no se mueve hasta que demuestre que es posible o no llegar al destino, en el caso de que no sea posible no ocurrirá nada, se quedará parada. La base del algoritmo la encontramos en una sección de la página de la Universidad de Stanford[1] que hablaba sobre programación de videojuegos, y ahí se podía encontrar varios enlaces sobre búsqueda de caminos etc. Usamos una librería de Priority Queues[2] que encontramos en github para poder realizar la búsqueda más rápido.

Respecto a los tiempos y que no lo note el jugador, se cumple sin ningún problema, un recorrido desde la casilla [0,0] hasta la casilla [100,100], teniendo que explorar 10.000 casillas, ha tardado en calcularlo menos de 3 ms

ADICIONES PERSONALES A LA PRÁCTICA

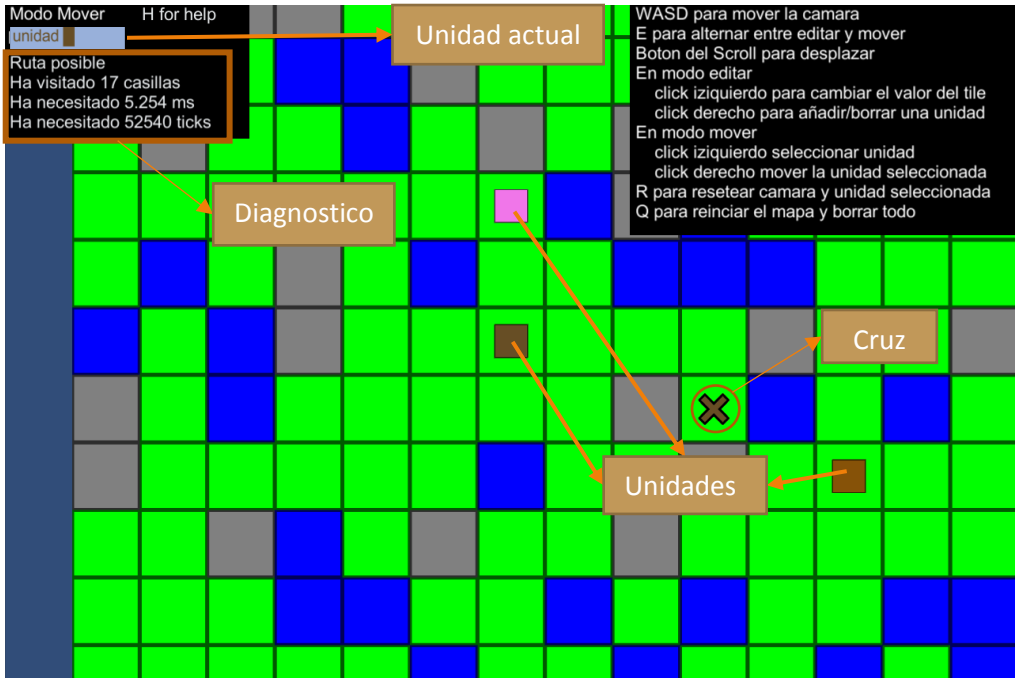
Nos tomamos la libertad de añadir el modo edición para que sea más cómodo el editar el mapa y poder añadir unidades, el cual hemos explicado su funcionamiento en el apartado del funcionamiento de las casillas.

Usamos un script llamado RTS Camera[3] para la cámara que encontramos en el asset store de Unity para poder mover la cámara de forma cómoda, con el WASD, el ratón y arrastrando pulsando el botón del scroll del ratón.

En nuestro GUI se indica toda la información sobre el movimiento y el coste de estos, y las instrucciones de cómo usar el juego, el cual explicamos en el siguiente apartado.

LAYOUT

El layout de la práctica es una implementación simple para el uso sencillo de la práctica. Se basa en el uso del elemento Grid para controlar que las casillas, las cruces y las unidades están en una posición de formato entero, para evitar desvíos por la coma flotante. Con múltiples elementos en el GUI para indicar información sobre múltiples datos.



CONCLUSIONES

Las conclusiones que sacamos de esta práctica son las siguientes.

LA IMPORTANCIA DE LOS TIEMPOS

A la hora del path finding en los juegos en tiempo real lo más importante es que el tiempo no se pare, aunque a veces haya que sacrificar la obtención del camino más corto, o el uso de hilos y corrutinas.

FALTA DE CONOCIMIENTO UNITY Y C#

No conocemos bien en ocasiones como usar correctamente Unity y C#, sobre todo a la hora de crear una interfaz o las formas de interactuar con el usuario, es una pena porque la mayor parte uno se dedica a crear una estructura de interactuar en el juego y "pierde" tiempo programandolo, esto sin contar que se cometan erratas durante el proceso de programación de dicha estructura, que luego causarán problemas haciendo más costoso cada vez el simple hecho de programar lo importante que es la IA. Y respecto a C# el problema puede ser que estamos acostumbrados a C++, pero es un problema que nosotros debemos resolver

REFERENCIAS

1. Universidad de Stanford Heurísticas Implementación
2. Autor: BlueRaja el repositorio
3. RTS camera asset de Unity