****

**Asignatura:** Sistemas expertos

**Profesor:** D. Sc. Gerardo García Gil

**Alumno:** Juan Pablo Cruz Martinez.

**Registro:** 19310167. **Ciclo:** 2022-A

**Ingeniería en Desarrollo de Software**

**Centro de Enseñanza Técnica Industrial (CETI)**

***Centro de Enseñanza Técnica Industrial (CETI)***

***Algoritmo A\* pathfinding***

**Introducción.**

El pathfinding o búsqueda de camino por su nombre en español, es un área de la inteligencia artificial el cual consiste en encontrar el mejor camino posible entre un punto a y otro punto b de un mapa. Este algoritmo fue desarrollado por Peter Hart, Bertram Raphael y Nils J. Nilsson en el año de 1968. Existen diferentes y muchas más maneras de realizar el algoritmo de pathfinding, en esta ocasión se ha desarrollado este algoritmo en el lenguaje JavaScript, ya que me resultó más sencillo realizarlo con este y representarlo más gráficamente.

Se utiliza en muchas partes este tipo de algoritmo como por ejemplo en Google Maps, Uber, Didi, etc. Para poder indicarnos no solo una ruta, sino varias rutas para poder llegar a nuestro destino si la ruta más corta se encuentra con trafico o el conductor toma otra calle, este algoritmo lo adaptaron estas plataformas para siempre encontrar la ruta más corta, no importa las desviaciones.

**Antecedentes.**

El algoritmo de Pathfinding provee una base importante para gran parte de los actuales sistemas de navegación, ya que estos se utilizan en áreas relacionadas con la robótica, pero también se usa especialmente en videojuegos.

El principal objetivo es encontrar el mejor camino posible de un punto inicial a un punto final en representaciones del entorno llamadas mapas malla. Los algoritmos que definen lo que es el movimiento de personajes de videojuegos o también robots son esenciales para que éste parezca lo más cercano a la realidad posible, haciéndolo parecer inteligente. Estos algoritmos tienen que definir por donde deben de pasar, esto es basándose en la información que se tiene del lugar o ambiente donde la cual va a estar representada en planillas para facilitar su procesamiento.

La búsqueda de un camino en lo que a videojuegos se refiere, es la forma en que una entidad en movimiento encuentra un camino alrededor de un obstáculo. El género donde más frecuente se encuentra es en los videojuegos de estrategia en tiempo real, ya que tu seleccionas a un personaje y le indicas el punto al que debe de llegar, el cual una vez iniciada la petición este debe de encontrar el camino más corto y rápido para llegar a ese determinado punto.

La importancia de la búsqueda de rutas ha crecido tanto en importancia como los entornos de los juegos se han vuelto cada vez más complejos, y eso da como resultado que varios paquetes de software de IA ya han desarrollado los algoritmos para resolver el problema.

**Desarrollo.**

El objetivo de esta práctica es desarrollar un programa que permita saber cual es el camino más corto entre dos puntos dados, para ello se utiliza el algoritmo A\*, el cual es un algoritmo de búsqueda que puede ser empleado para el cálculo de caminos mínimos en una red. Se va a tratar de un algoritmo heurístico, ya que una de sus principales características es que hará uso de una función de evaluación heurística, mediante la cual etiquetará los diferentes nodos de la red y que servirá para determinar la probabilidad de dichos nodos de pertenecer al camino óptimo. La heurística estima el costo mínimo desde un nodo al objetivo. Dependiendo de cual se utilice, puede ayudar o perjudicar bastante la búsqueda del camino apropiado.

Muchas veces lo que pasa con este programa es que tienes que colocar de manera manual los muros y todo lo necesario para buscar el camino, en mi caso decidí que los muros se generen aleatoriamente para que sea más rápido de visualizar el camino más corto y automáticamente se inicia la búsqueda para encontrarlo, algo que no supe como implementar de manera efectiva fue el colocar el inicio y el final, ya que solo lo puedes cambiar entrando al código, al igual que el tamaño de la cuadricula, solo se modifica como ya lo mencioné a través del código.

**Código**

**Texto

Descripción generada automáticamenteImagen en blanco y negro

Descripción generada automáticamente con confianza media**

**Texto

Descripción generada automáticamenteFondo negro con letras blancas

Descripción generada automáticamente con confianza mediaTexto

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza mediaTexto

Descripción generada automáticamenteImagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente**

**Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente**

**Texto

Descripción generada automáticamente**

**Resultados**

Ya terminado el programa, los resultados son buenos, ya que cumple con su objetivo que es encontrar el camino más corto, pero como ya mencioné anteriormente se generan de manera aleatoria los muros para ser rápidos al momento de querer encontrar el camino, y a veces es tardado el tener que estar colocando muro por muro en todo el mapa.

**Código QR

Descripción generada automáticamenteCódigo QR

Descripción generada automáticamente**

**Código QR

Descripción generada automáticamente**

Como se puede observar, los muros que son de color negro cada vez los genera aleatoriamente, y el camino más corto es lo marcado en color amarillo, algo curioso que a veces suele ocurrir es que hay casos en los que los muros impiden llegar al punto final, por lo que no se puede resolver y no marca ninguna línea amarilla.

**Conclusión**

Este tipo de algoritmo es muy útil y se puede aplicar en muchas partes desde videojuegos donde la inteligencia artificial la utiliza para que un personaje camine del punto A al punto B en la ruta más corta o también se usa en la aplicación de Google Maps para indicarnos como llegar a nuestro destino más rápido e incluso lo han mejorado aquí ya que nos puede marcar más de una ruta para llegar y nos puede decir donde hay tráfico.

Tuve unas complicaciones al realizarlo, por lo que tuve que estar buscando información durante varias horas, lo único que no pude resolver para que se viera de manera más grafica fueron los puntos de inicio y fin, ya que como menciono anteriormente, tengo que entrar al código y modificar directamente estos puntos para que cambien de posición.

**Referencias**

1. Algoritmo A\* | IDELab. (s. f.). IDELab. <http://idelab.uva.es/algoritmo>
2. Danilo Alejandro Cuevas Guzmán. (2013, diciembre). Análisis de Algoritmos Pathfinding. <http://opac.pucv.cl/pucv_txt/txt-5000/UCE5372_01.pdf>