
Proyecto 1

Carnet 1 – Pablo Alejandro Franco Lemus

Resumen

Se diseñó una aplicación que permite calcular la mejor ruta para el robot de exploración r2e2. Para ello se utilizó el algoritmo A*, presentado por primera vez en 1968.

El problema consistía en encontrar la mejor ruta para el robot dada una coordenada inicial y final, tomando en cuenta el costo de combustible para llegar al objetivo.

El proyecto abarca la implementación del algoritmo A*, lectura de archivos XML, estructuras de datos y representación de estructuras mediante Graphviz.

Desarrollar este proyecto me permitió profundizar bastante en el contenido del curso de introducción a la programación 2 y también tiene bastantes aplicaciones en la vida real.

Palabras clave

Graphviz
Estructuras de datos
XML
*Algoritmo A**
Camino de menor costo

Abstract

An application was designed to allow the calculation of the best route for the exploration robot r2e2. The A algorithm was implemented to achieve it. This algorithm was first presented in 1968.*

The problema consisted in finding the best route for the robot given an initial and a final coordinate taking into consideration the fuel cost to get to the target.

The Project comprises the implementation of the A algorithm, XML file handling, data structures and data structure representation using Graphviz.*

Developing this Project allowed me to get a more solid understanding of the “introduction to programming and computing 2” course and it also has several applications in real life.

Keywords

Graphviz

Data structures

XML

Algoritmo *

Least costly path

Introducción

El proyecto consistía principalmente en aplicar los conceptos de estructuras de datos vistos en la clase de IPC 2 e implementar un algoritmo propio o ya inventado para poder encontrar el camino de menor costo entre dos nodos.

La pieza central de este programa es el algoritmo A* que fue inventado por Nils John Nilson, Bertram Raphael y Peter E. Hart, presentado por primera vez en 1968.

Para poder dar solución a este problema en cuanto a la parte técnica hubo que realizar una investigación a profundidad del lenguaje Python, XML e implementación de listas enlazadas.

Desarrollo del tema

Se escogió el formato XML para guardar y cargar los datos de los terrenos a procesar debido a su sencillez para trabajar en Python. En cuanto al algoritmo A* tuvo algunos interesantes retos como poder comprenderlo e implementarlo.

El algoritmo A* es perfecto para situaciones para situaciones de la vida real en donde se quiere aproximar el camino mas corto, como por ejemplo en los videojuegos, o en mapas en donde puede haber muchos obstáculos. El algoritmo introduce una heurística a un algoritmo convencional de búsqueda en grafos esencialmente planeando por adelantado cada paso para tomar una decisión óptima. Lo que hace al algoritmo A* diferente y mucho mejor para muchas búsquedas es que por cada nodo, A* utiliza una función $f(n)$ que da un estimado del costo total de un camino usando ese nodo.

```
Python
1  make an openlist containing only the starting node
2  make an empty closed list
3  while (the destination node has not been reached):
4      consider the node with the lowest f score in the open list
5      if (this node is our destination node) :
6          we are finished
7      if not:
8          put the current node in the closed list and look at all of its neighbors
9          for (each neighbor of the current node):
10             if (neighbor has lower g value than current and is in the closed list) :
11                 replace the neighbor with the new, lower, g value
12                 current node is now the neighbor's parent
13             else if (current g value is lower and this neighbor is in the open list) :
14                 replace the neighbor with the new, lower, g value
15                 change the neighbor's parent to our current node
16
17             else if this neighbor is not in both lists:
18                 add it to the open list and set its g
```

The time complexity of A* depends on the heuristic. In the worst case, the number of nodes expanded is exponential in the length of the solution (the shortest path), but it is polynomial when the search space is a tree.

Pseudocódigo A*, fuente: brilliant.org

Ejemplo

Examples

EXAMPLE

Use A* to find the shortest path from the green square to the yellow square in the grid below.

Let us start by choosing an admissible heuristic. For this case, we can use the Manhattan heuristic. We then proceed to the starting cell. We call it our 'current cell'.

Fuente: Brilliant.org

		F = 6.6 G = 5.6 H = 1	F = 5.2 G = 5.2 H = 0
	F = 7.2 G = 4.2 H = 3	F = 5.8 G = 3.8 H = 2	F = 5.2 G = 4.2 H = 1
F = 7.8 G = 2.8 H = 5	F = 6.4 G = 2.4 H = 4	F = 5.8 G = 2.8 H = 3	F = 5.8 G = 3.8 H = 2
F = 7 G = 1 H = 6	F = 6.4 G = 1.4 H = 5		F = 7.2 G = 4.2 H = 3
	F = 7 G = 1 H = 6		

Se comienza por escoger una heurística aceptable. En este caso se puede usar la heurística Manhattan. Se procede entonces desde la celda inicial (cuadro verde). Lo llamamos celda actual y procedemos a mirar a sus vecinos y a procesar. F, g y h para cada uno de ellos. Entonces seleccionamos el vecino con el menor costo f. Esta será nuestra nueva celda actual y después repetimos el proceso (Llenar vecinos y procesar f, g y h y escoger el más bajo). Hacemos esto hasta llegar a la celda destino.

Donde:

G: Es el costo que ha sido acumulado para llegar a la celda.

H: Es la distancia Manhattan hacia la celda amarilla

F: Es la suma de h y g

Referencias bibliográficas

A* Search. (s.f.). Recuperado el 29 de agosto de 2021, de <https://brilliant.org/wiki/a-star-search/>

A* Search Algorithm. Recuperado el 29 de Agosto de 2021. De <https://www.geeksforgeeks.org/a-search-algorithm/>

Conclusiones