ÚSTAV AUTOMATIZACE A INFORMATIKY

Algoritmy umelej inteligencie (VAI)

Meno:		ID študenta:
Bc. Daniel Dlugoš		217099
Dátum zadania:	Dátum odovzdania:	Študijná skupina:
07.02.2024	12.06.2024	4pAIŘ/1

Názov úlohy:

Vizualizácia hľadania optimálnej cesty pomocou algoritmu A* v 2D priestore.

1 Algoritmus A*

Je úplný a optimálny algoritmus usporiadaného prehľadávania na základe hodnotiacej funkcie f. Ak existuje riešenie, je zaručené ho nájde. Hlavným problémom algoritmu je pamäťová náročnosť, pretože v pamäti udržiava všetky generované stavy. Pri rozsiahlych problémoch môže algoritmus predčasne skončiť vyčerpaním pamäte.

1.1 Informované metódy prehľadávania stavového priestoru

Informované metódy prehľadávania sú založené na využití hodnotiacej funkcie *f*.

- Určuje ohodnotenie pre každý uzol stromu stavového priestoru.
- Ohodnotenie sa používa pre výber ďalšieho uzlu k expanzii.
- Vždy sa expanduje ten uzol, ktorý mal doteraz najlepšie ohodnotenie a vyhodnocujú sa iba jeho následníci.
- Hľadanie je zastavené, keď je dosiahnutý stav, ktorý má lepšiu hodnotu funkcie f ako jeho následníci.

1.2 Hodnotiaca funkcia A*

$$f(i) = g(i) + h(i)$$

$g(i)$ — cena optimálnej cesty z počiatočného stavu do stavu i.
$h(i)$ — cena optimálnej cesty zo stavu i do niektorého z cieľových stavov.

1.2.1 Funkcia *h (i)*

Kvantitatívne vyjadruje „odhad" ceny cesty z aktuálneho stavu do niektorého z cieľových stavov. „Odhad" predstavuje heuristickú znalosť o tom, aké sú šance nájsť riešenie, ak by sme pokračovali expanziou daného stavu. Funkcia *h (i)* je preto nositeľom heuristickej informácie a nazýva sa aj heuristickou funkciou.

2 Zdrojový kód

V tejto kapitole bude predstavený zdrojový kód a funkcie, ktoré je naprogramovaný vykonávať.

2.1 Predrekvizity

Importuje moduly pygame, math a PriorityQueue. Inicializuje rozmery zobrazovacieho okna s názvom "A* Snake Hunting It's Prey".

Pygame je sada modulov programovacieho jazyka Python určených pre vitváranie videohier. Pygame má veľa vstavaných funkcií a nakoľko nie je potrebné využiť mnoho z nich, intuitívnejší prístup bude vytvoriť vlastnú hernú logiku a triedy.

2.2 Globálne nastavenia

Medzi prvé kroky zaradíme deklaráciu konštantných hodnôt. Tie zahŕňajú nastavenia:

- Absolútnej veľkosti šírky hracieho poľa - hracie pole bude nastavené pre štvorcovú plochu.
- RGB kód pre farby uzlov tzv. spots. Uzly budú vyfarbené na základe funkcie, ktorú v programe vykonávajú.

2.3 Trieda Spot

Bude reprezentovať každú bunku (uzol) v mriežke a všetky metódy týkajúce sa vlastností uzlov, aktualizácie ohodnotenia uzlov a vyfarbenia uzlov. Každý uzol má vlastnosti ako riadok, stĺpec, šírka, farba a susedia. Význam jednotlivých funkcií je popísaný v následujúcich kapitolách:

2.3.1 __init__()

Metóda, ktorá slúži na inicializáciu vlastností uzlov a všetkých nastavení objektu.

2.3.2 get_pos()

Získava polohu daného uzla. Polohu reprezentuje index stĺpca a riadku mriežky v zobrazovacom okne.

2.3.3 is_closed(), ..., is_end()

Metódy kontrolujú stav uzlu:

- Jeho výskyt v zozname OPEN alebo CLOSED.
- Či sa jedná o uzol reprezentujúci stenu.
- Či sa nejedná o počiatočný alebo koncový uzol.

2.3.4 reset() ... make_path()

Metódy menia farbu uzlu na základe jeho stavu.

2.3.5 draw()

Metóda pre zakreslenie uzlu do zobrazovacieho okna.

2.3.6 update_neighbors

Funkcia pre aktualizáciu susedov. Pozerá sa na susedov aktuálneho uzlu v každom smere. do zoznamu OPEN zapíše uzol, ktorý je najviac optimálny pre priblíženie sa ku konečnému uzlu. Uzly sa preskúmavajú v smere od ľavého horného uzlu (počiatok súradnicového systému) po pravý dolný uzol. Funkcia si vyberá susedov, ktorých expanduje následovne:

- Smerom dole, ak sa nejedná o posledný riadok v mriežke a zároveň sa nejedná o uzol, ktorý reprezentuje stenu.
- Smerom hore, ak sa nejedná o prvý riadok v mriežke a zároveň sa nejedná o uzol, ktorý reprezentuje stenu.
- Doľava, ak sa nejedná o prvý stĺpec v mriežke a zároveň sa nejedná o uzol, ktorý reprezentuje stenu.
- Doprava, ak sa nejedná o posledný stĺpec v mriežke a zároveň sa nejedná o uzol, ktorý reprezentuje stenu.

2.4 Heuristika *h()*

Definuje heuristickú funkciu *h*. Heuristika používa Manhattanskú vzdialenosť na odhad vzdialenosti medzi dvoma bodmi. Obmedzenie pohybu iba na horizontálny a vertikálny smer. Vracia súčet absolútnych vzdialeností v oboch smeroch.

2.5 reconstruct_path()

Funkcia, ktorá spätne vykresľuje respektíve rekonštruuje cestu z koncového bodu k počiatočnému.

2.6 Raw A*

Funkcia, ktorá programovo interpretuje algoritmus A* začína:

- open_set – Inicializáciou zoznamu OPEN pomocou modulu Queue.
- count – premenná udržiavacia poradie uzlov, podľa toho ako vstupovali do zoznamu. Počítadlo nadobúda dôležitosť v prípade že algoritmus objaví viacero uzlov s rovnakým ohodnotením. V tomto prípade si podľa premennej count algoritmus zvolí uzol, ktorý sa v zozname objavil ako prvý.
- came_from – je zoznam, do ktorého sa zapisujú uzly cesty. Spätne sa využíva pre zvýraznenie optimálnej cesty vo funkcii recostruct_path().
- g_val – deklarácia vzdialenosti od počiatočného uzlu do aktuálneho uzlu [*g(i)*].
- f _val – deklarácia ohodnocovaciej funkcie (súčet g_val a heuristickej funkcie h()).
- open_set_hash – Zoznam, ktorý kontroluje obsah zoznamu queue a odstraňuje duplikáty.

Po deklarácií počiatočných premenných algoritmus prebieha v slučke a je ukončený v prípade, že sa prehľadal každý uzol alebo neexistuje riešenie (cieľový uzol je nedostupný). Slučka obsahuje podmienku, v ktorej kontroluje či aktuálny uzol nie je koncový. V opačnom prípade prehľadáva susedné uzly.

Pri výbere susedného uzlu s najlepším ohodnotením sa pracuje s premennou temp_g_val, ktorá predpokladá doposiaľ najlepšiu cestu od počiatočného uzlu k aktuálnemu. Ak existuje lepšia cesta, algoritmus si aktualizuje jej ohodnocovaciu funkciu.

Algoritmus pokračuje vyfarbovaním uzolov, pričom kontroluje podmienku či je aktuálny uzol zároveň počiatočný. V tomto prípade prefarbí uzol na červeno.

2.7 Visualization Tool

Súbor funkcií, ktoré vykonávajú všetky dôležité aspekty pre vizualizáciu princípu fungovania algoritmu A*.

2.7.1 make_grid()

Funkcia, ktorá vytvára mriežku.

2.7.2 draw_gridlines()

Funkcia, ktorá vytvára zvislé a vodorovné čiary mriežky.

2.7.3 draw()

Funkcia, ktorá vykresľuje zobrazovacie okno, mriežku a čiary mriežky.

2.7.4 get_clicked_pos()

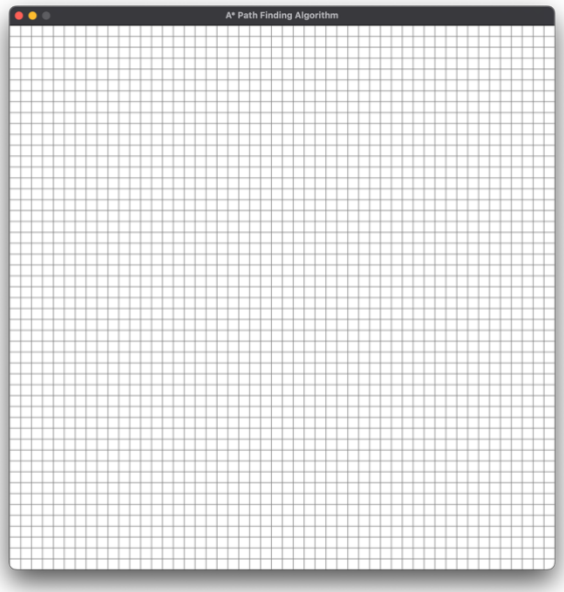
Funkcia, ktorá získava polohu mriežke kde došlo ku kliku myšou.

2.8 Funkcia main()

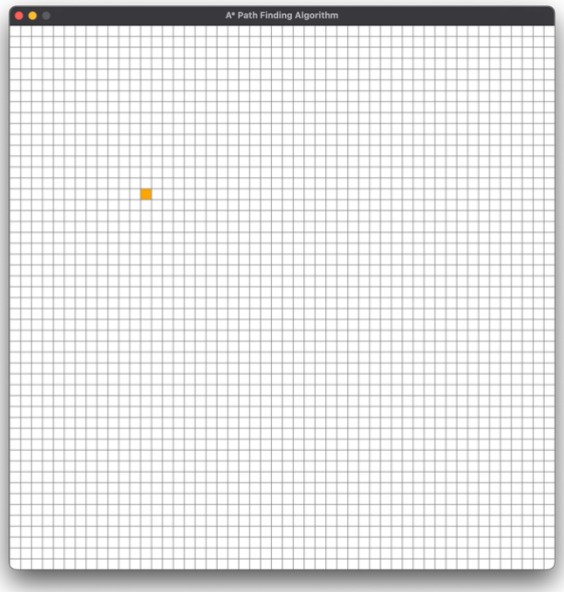
Hlavná funkcia, ktorá inicializuje mriežku a spracováva všetky interakcie užívateľa. Po vykreslení mriežky je užívateľ vyzývaný aby si ľubovoľne zvolil počiatočný bod, koncový bod a zakreslil steny.

3 Pokyny k používaniu

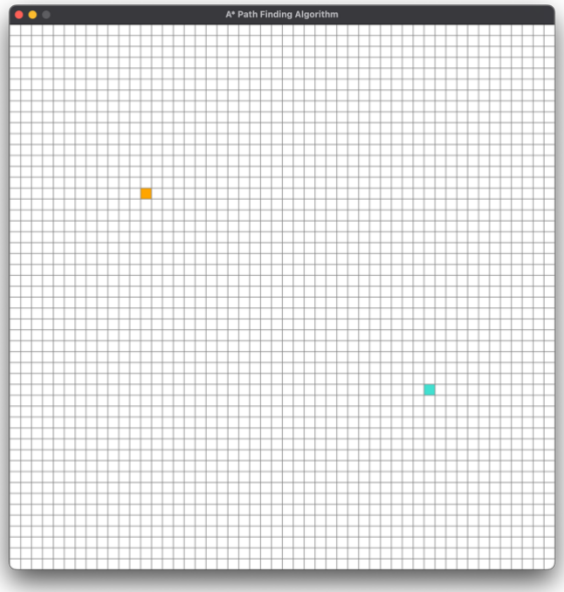
- Program začína s vykreslením prázdnej mriežky vo fixnej veľkosti.
- Nastavenie počiatočného bodu, cieľového bodu a steny sa vykonáva kliknutím ľavého tlačidla myši na bunky mriežky (pri kreslení steny možno tlačidlo aj držať).
- Uzly sa resetujú (späť do bielej farby) prvým klikom myši.
- Stlačenie medzerníka spustí algoritmus A*.
- Mriežku možno po skončení algoritmu vrátiť do pôvodného stavu stlačením klávesy „C“.



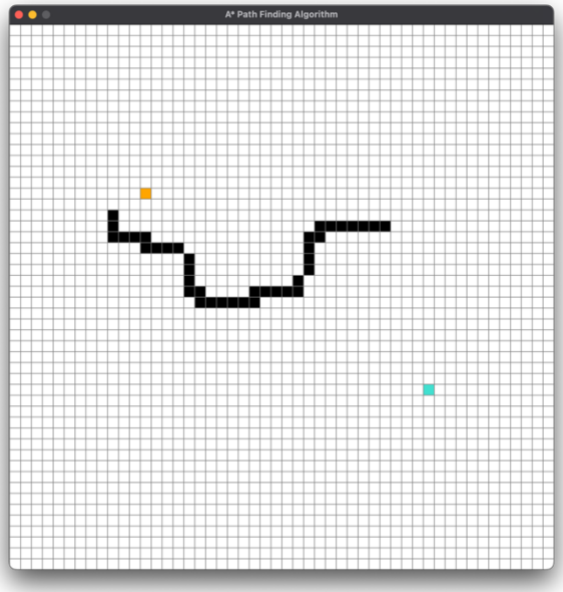
Obr. 1 – Prázdna mriežka



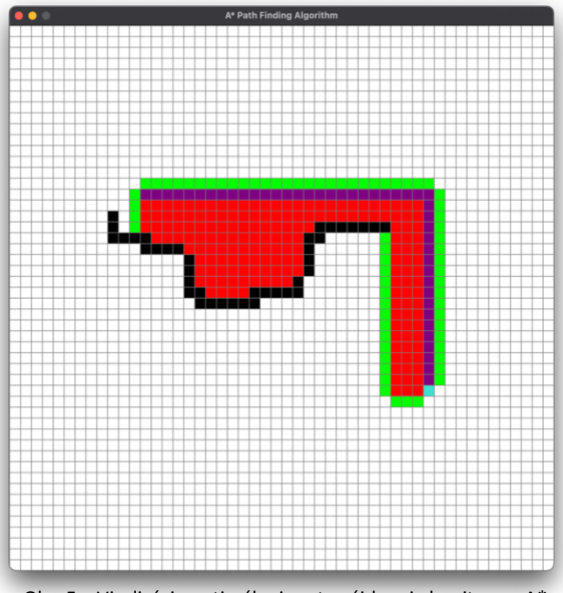
Obr. 2 – V mriežke je zakreslený počiatočný bod



Obr. 3 – V mriežke je zakreslený počiatočný a cieľový bod



Obr. 4 – Počiatočný bod, cieľový bod a prekážka (stena)



Obr. 5 – Vizalizácia optimálnej cesty nájdenej algoritmom A*

4 Záver

V tejto práci bol implementovaný a vizualizovaný algoritmus informovaného prehľadávania stavového priestoru A*. Grafický modul Pygame umožňuje simulovať správanie a schopnosti algoritmu, ktorý je schopný nájsť cestu najoptimálnejšej cesty, pokiaľ existuje. Vylepšenie programu by priniesla úprava heuristickej funkcie z Manhattanskej na Euklidovksú.

5 Zdroje

1. *A* Algorithm Concepts and Implementation: How to Implement the A* Algorithm in Python?* Online. 2024. Dostupné z: https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/a-star-algorithm#how_to_implement_the_a_algorithm_in_python. [cit. 2024-06-12].

2. *A* Search Algorithm.* Online. 2024. Dostupné z: https://www.geeksforgeeks.org/a-search-algorithm/?ref=ml_lbp. [cit. 2024-06-12].

3. *Implementation of A*.* Online. 2020. Dostupné z: https://www.redblobgames.com/pathfinding/a-star/implementation.html. [cit. 2024-06-12].

4. *A* Pathfinding (E01: algorithm explanation).* Online. 2014. Dostupné z: https://www.youtube.com/watch?v=-L-WgKMFuhE. [cit. 2024-06-12].

5. *Pygame: About - Wiki.* Online. Dostupné z: https://www.pygame.org/wiki/about. [cit. 2024-06-12].

6. *Python PyGame Tutorial: The Complete Guide.* Online. Dostupné z: https://coderslegacy.com/python/python-pygame-tutorial/?utm_content=cmp-true. [cit. 2024-06-12].