

UNIVERSITATEA "ȘTEFAN CEL MARE" SUCEAVA
FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR
SPECIALIZAREA CALCULATOARE

PROIECT DISCIPLINĂ POO

"Identificare traseu între 2 puncte"

Canevschii Daniel

Tema și motivația alegerii

Tema dată presupune găsirea celui mai scurt traseu între două puncte. Ca scop final al proiectului se cere ca programul dat să poată găsi cel mai scurt traseu posibil între două puncte de pe ecran, poziția cărora este dată de către utilizator.

Deși problema dată reprezintă în mare parte o problemă de algoritmică, implementarea acesteia în limbajul de programare **C++** presupune o realizare a acestuia în stil **OOP**.

Motivația de a alege acest proiect se află în oportunitatea studierii mai extensiv atât al **Programării Orientate pe Obiecte**, cât și a algoritmilor de căutare bazați pe grafuri.

Inspirația alegerii:

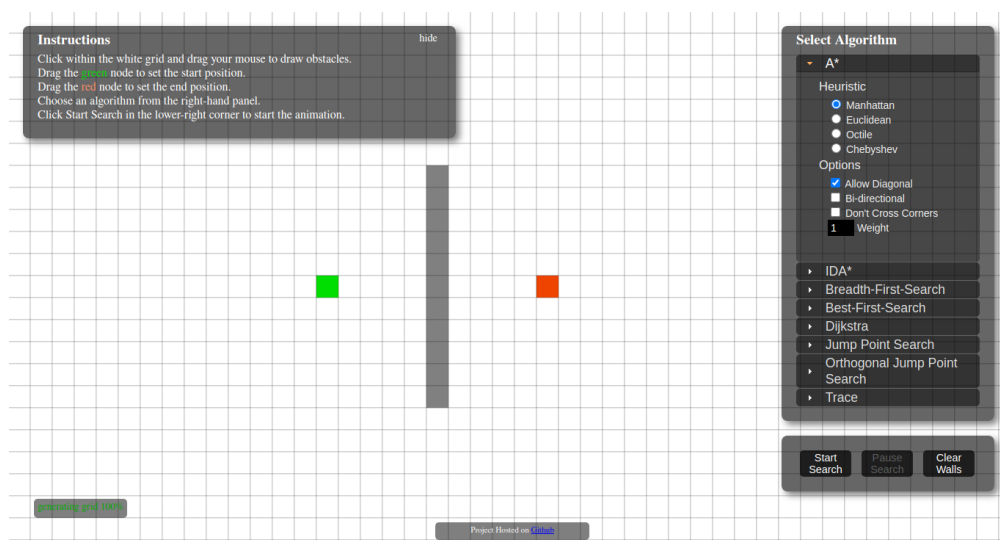


Fig.1 Proiectul de inspirație ¹ (punct de start și final cu barieră la mijloc)

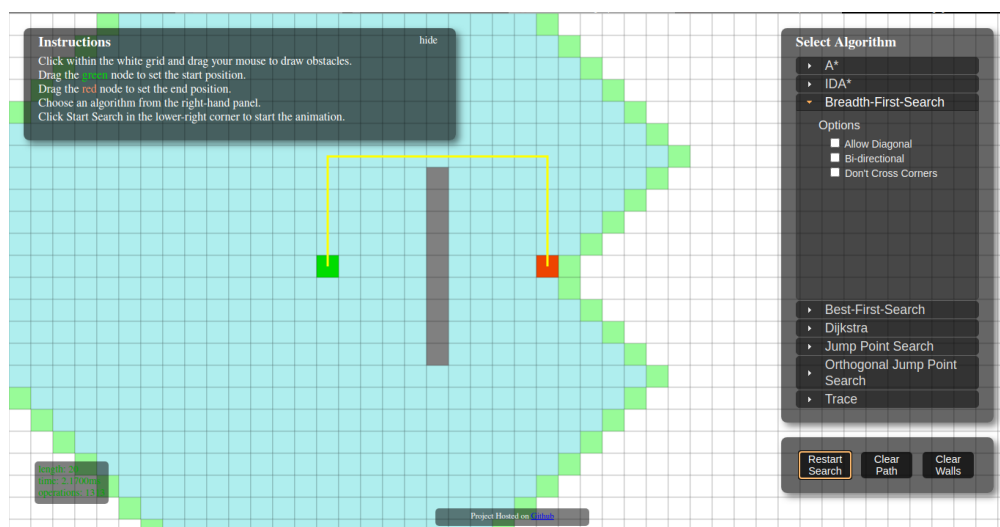


Fig.2 Proiectul de inspirație(faza finală)

¹<https://qiao.github.io/PathFinding.js/visual/>

Cuprins

1	Elemente teoretice	3
1.1	Descrierea Proiectului	3
1.2	Elemente specifice POO	4
1.3	Elemente specifice Interfață	5
1.4	Idei adăgate în curs de dezvoltare	5
2	Implementare	7
2.1	Tehnologii folosite	7
2.2	Diagrama de clase	7
3	Analiza soluției implementate	8
3.1	Harta folosită pentru teste de performanță	8
4	Manual de utilizare	12
5	Concluzii	14
6	Bibliografie	15
6.1	Articole	15
6.2	Surse Diverse	15

Capitolul 1

Elemente teoretice

1.1 Descrierea Proiectului

Proiectul dat prezintă cautarea și afișarea în timp real a găsirii celui mai scurt drum între două puncte de pe grid.

Interfața proiectului presupune două puncte, poziția cărora poate fi modificată de către utilizator, adițional acesta permite utilizatorului să definească obstacole ce pot modifica drumul rezultat.

Interfața, la baza căreia stă librăria open-source FTXUI ¹ pentru dezvoltarea interfețelor de tip terminal utilizând limbajul C++, are o latență de reîmprospătare de 100ms, care trece prin întreaga matrice, în care este definită starea rețelei afișată pe ecran.

În matrice fiecare celulă reprezintă o celulă afișată pe ecran, valoarea fiecărei celule din matrice e reprezentată prin numerele {0, 1, 2, 3, 4, 5} ce reprezintă respectiv {empty, start, end, wall, visited, path} care au la rândul lor asiguate culori specifice.

Partea dreaptă a ecranului presupune un mesaj de introducere și o listă de tip radiobutton care permite alegerea algoritmului de *pathfinding* dorit.

Se presupun deasemenea butoane care efectueaza actiuni de tipul:

- *START* - butonul de start a algoritmului de cautare;
- *CLEAR* - curățare screen de tot inafara de punctul de start și end;
- *RESET* - butonul de reset a gridului (poziția inițială a celulelor *start* și *end*;

¹<https://github.com/ArthurSonzogni/FTXUI>

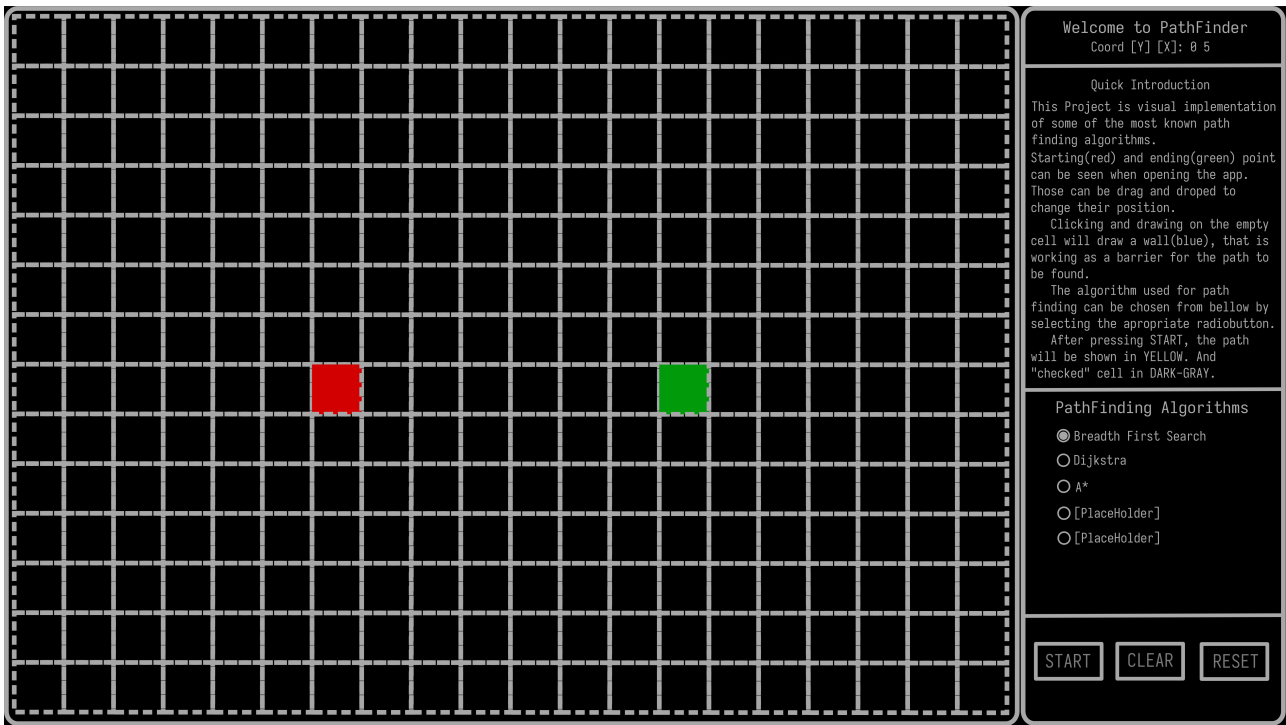


Fig. 3 Interfața propusă spre realizare

1.2 Elemente specifice POO

Fișierul *main* are la bază doar partea de interfață, care crează clasa de tip *Grid* și apelează metodele specifice a acesteia în dependență de *Event-ul* curent.

Clasa *Grid* are ca variabilă privată de tip clasă *Matrix*, și metodele acesteia efectuează modificări asupra variabilei *matrix*.

Un fișier cu funcții de calculare a traseului cel mai eficient în dependență de algoritmul de cautare ales.

Deasemenea se va crea o clasă de pastrare(matrice) a stării gridului care e modificat de către clasa de calculare a traseului și a interfeței. Datele căreia sunt preluate pentru afișare de către *thread-ul* de *refresh* a ecranului.

1.3 Elemente specifice Interfață

Canvas

Pentru partea de interfață se consideră funcție de convertire a unei matrici într-un grid de tip *ftxui::canvas*, verificând valoare fiecărei celule a matricii și atribuindu-i culoarea corespunzătoare valorii.

Aceasta face o reîmprospătare la fiecare 100ms, și la fiecare *Event* primit de la utilizator (în mare parte *Mouse Event*).

La un *Event* de tip mouse, în caz ca pointerul mouse-ului se află în zona de grid, se calculează poziția acestuia în referință cu celula pe care se află pe ambele axe x și y .

În cazul în care celula pe care s-a aflat pointerul mouse-ului e o celula de tip *empty*, aceasta se va schimba pe o celulă de tip *wall* și viceversa. De asemenea se poate apăsa pe o celula goală și face hover pe celule, dacă sunt de tip *empty* se convertesc în celule de tip *wall*.

Modificarea poziției celulelor de tip *start* și *end*, se efectuează utilizând drag-and-drop.

La apăsarea butonul START se începe cautarea drumului într-un nou *thread* și modifică valoarea celulelor vizitate cu o latență de 25 ms.

Iar în momentul când găsește celula de tip *end*, căutarea se oprește și se începe recreierea drumului de la final la început.

1.4 Idei adăugate în curs de dezvoltare

În cursul dezvoltării aplicației date precum și studierii pentru realizarea acesteia, au fost adăugate o serie de funcțiuni.

Pentru o realizare eficientă și utilă al algoritmului Dijkstra este nevoie ca blocurile prin care caută algoritmul să aibă un "cost". Acesta constă în crearea diferitor *real world* tipuri de celule, precum *apă*, *pădure*, *nisip etc*, și atribuie fiecăreia un cost diferit și specific fiecăreia.

Pentru realizarea a fost adăugat un "widow" care apare la "click dreapta" al mouse-ului și care afișează o lista de buttoane cu tipurile de celule disponibile.



Fig.4 Fereastra de alegere a tipului de celulă pentru desenat

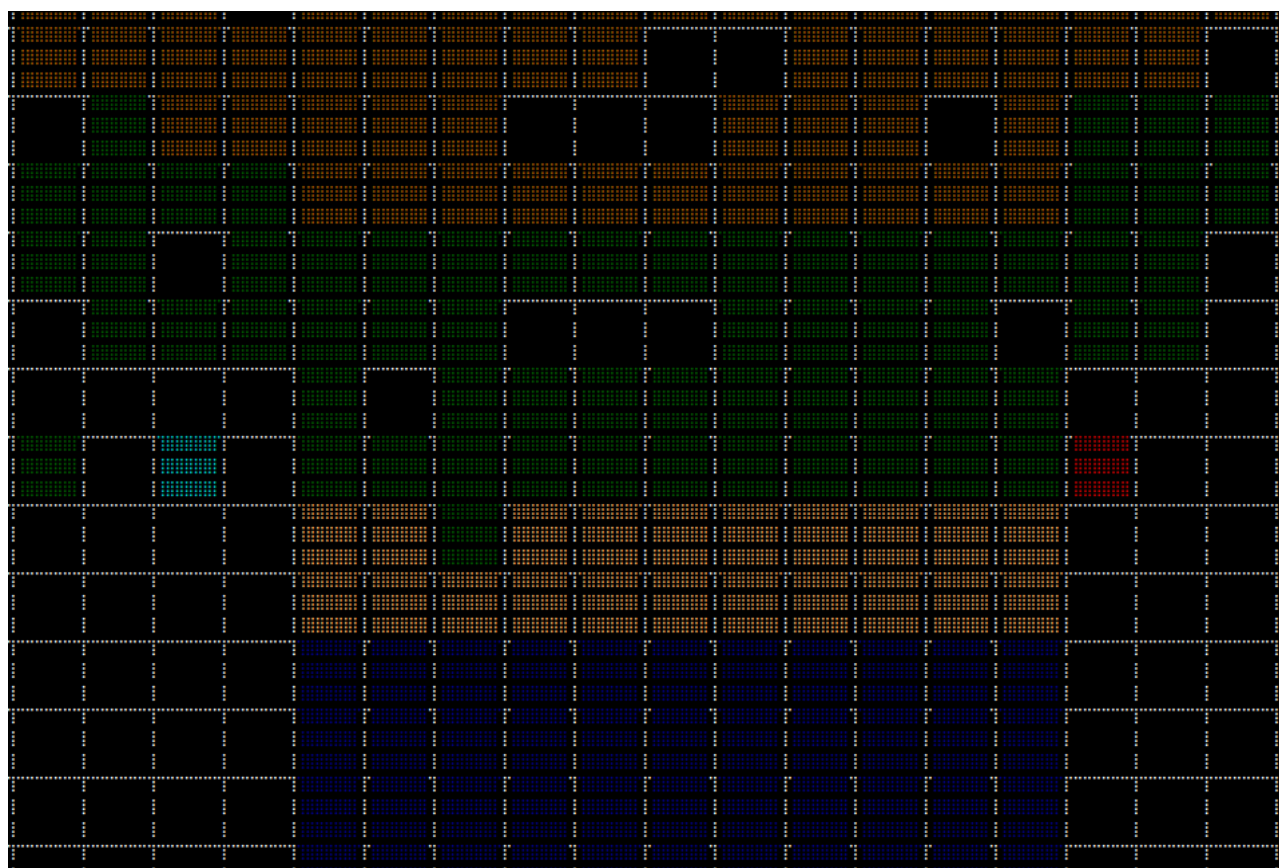


Fig.5 Tipurile de celule adăgate pe grid

Capitolul 2

Implementare

2.1 Tehnologii folosite

Limbajul de bază a proiectului e **C++**, compilat folosit compilatorul **g++**, utilizând utilitarul de compilare și administrare a build-ului automat **CMake** și **Make**.

Drept mediu de dezvoltare se va utiliza editorul de fișiere **VIM**, iar pentru managmentul modificărilor făcute în proiect se utilizează, sistemul de control a versiunii **GIT**.

Proiectul are o copie up-to-date încărcată pe **GITHUB**.

Interfața grafică a acestuia se va crea utilizând o librerie open source **FX-TUI**.

Pentru o vizualizare real-time a cautării traseului s-a utilizat tehnologia de *threading*, și încetinire a căutării acestuia.

2.2 Diagrama de clase

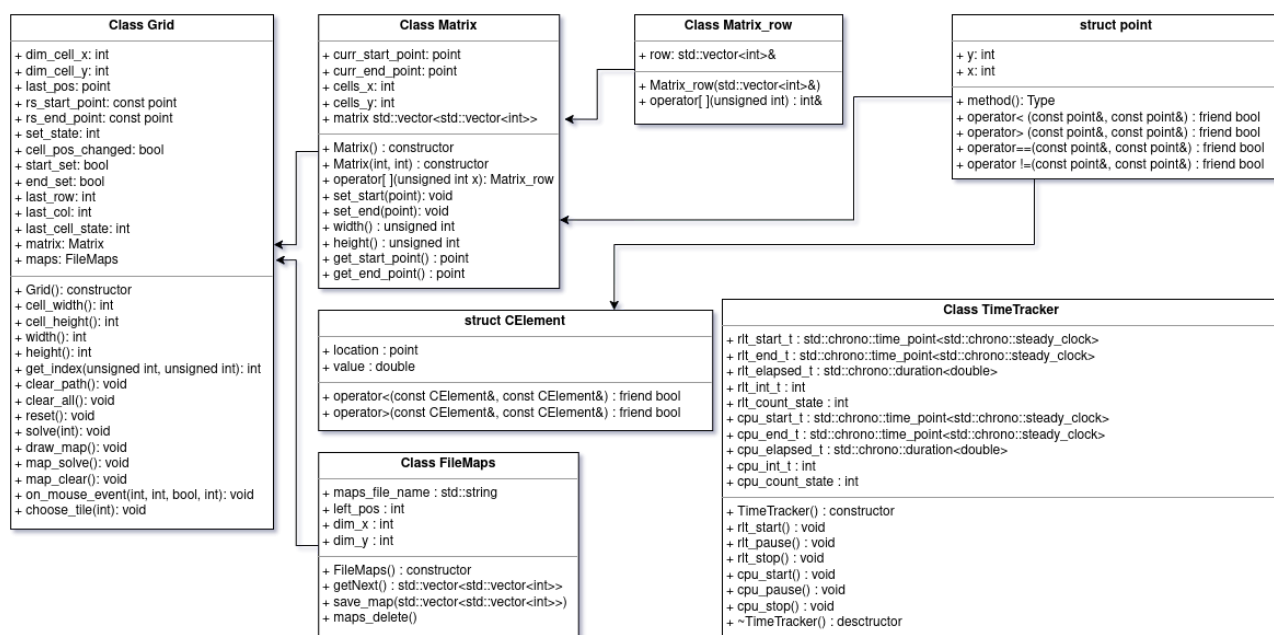


Fig.x Diagramă UML a proiectului

Capitolul 3

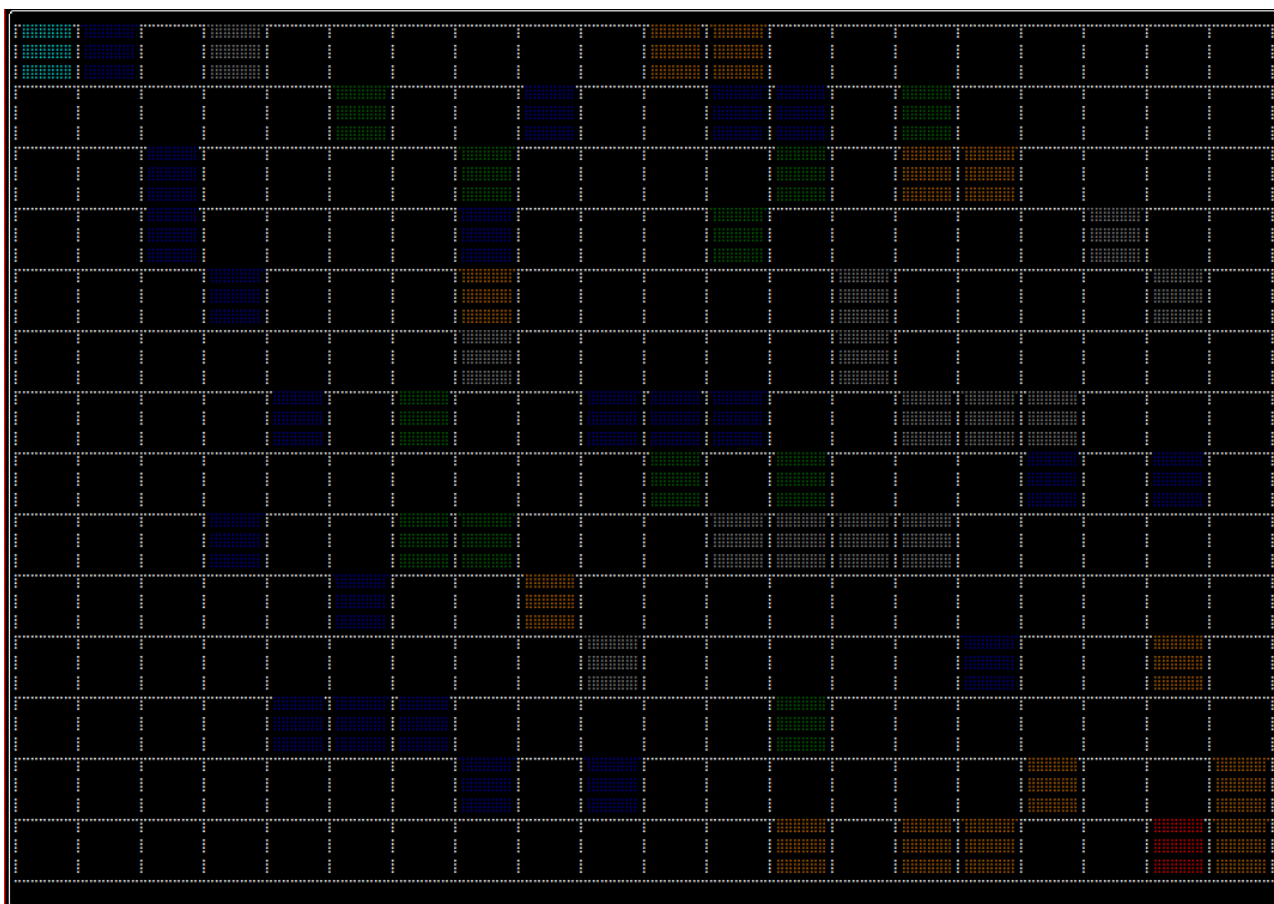
Analiza soluției implementate

Pentru analiza algoritmilor implementate se va utiliza o clasă creată pentru măsurarea timpului **CPU** și **Real Time**. Și de asemenea de o clasă realizată pentru păstrarea și redesenarea repetată a unor hărți desenate special pentru testarea eficienței.

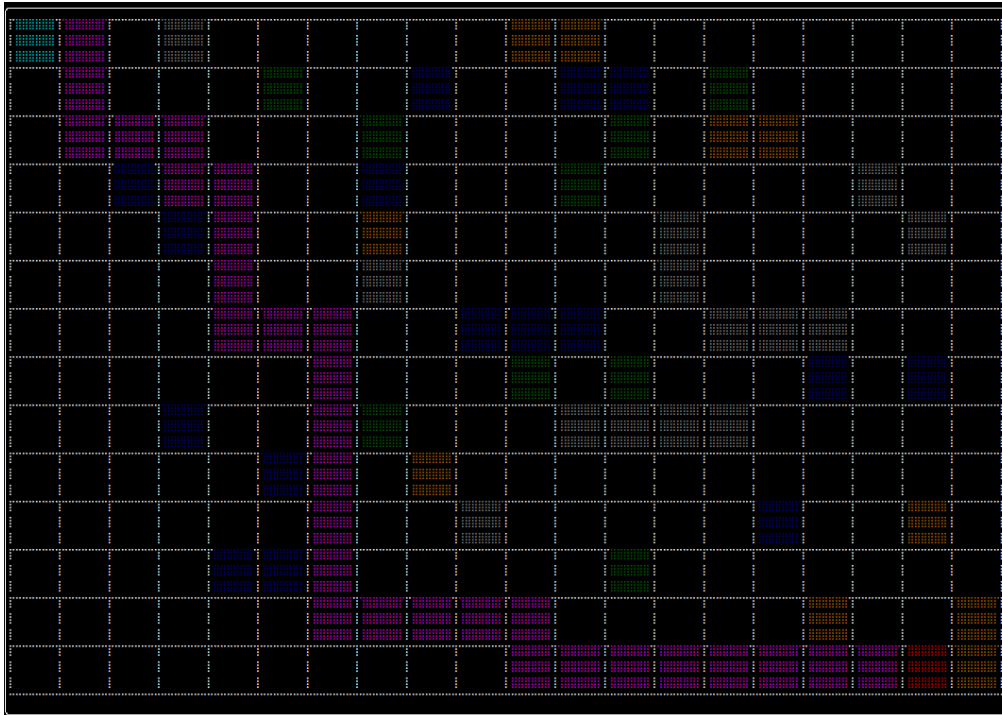
În dreapta deasupra butoanelor principale se afișează datele obținute în urma rulării programului(algoritmului).

Pe lângă timpul de procesare și cel real se afișează de asemenea și lungimea drumului găsit, și eficiența din punct de vedere al costului.

3.1 Harta folosită pentru teste de performanță

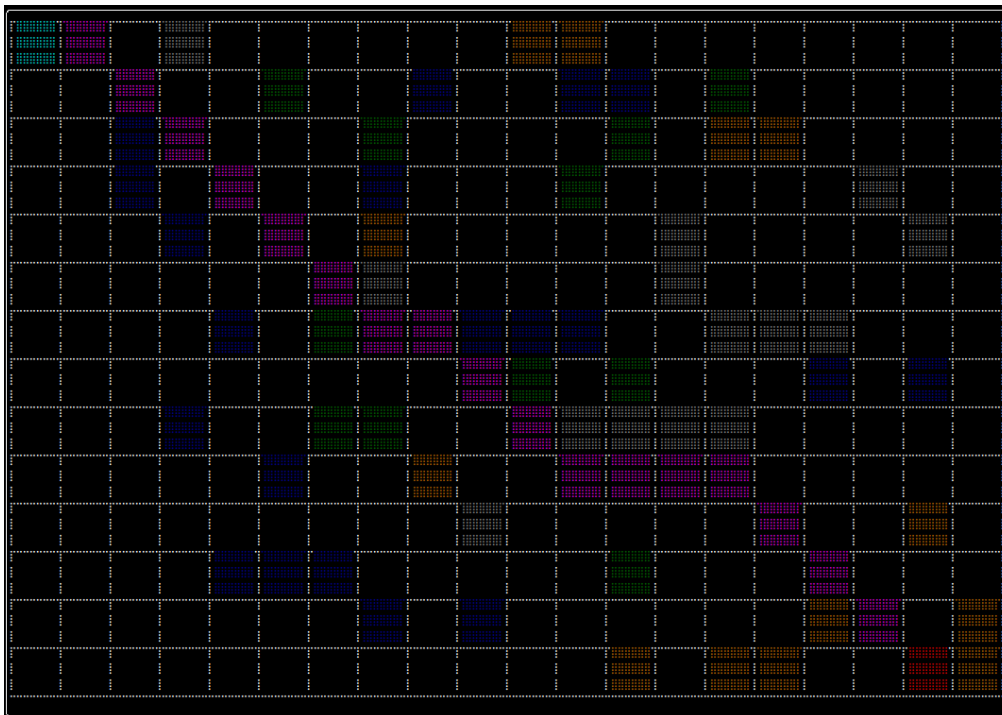


Breadth First Search



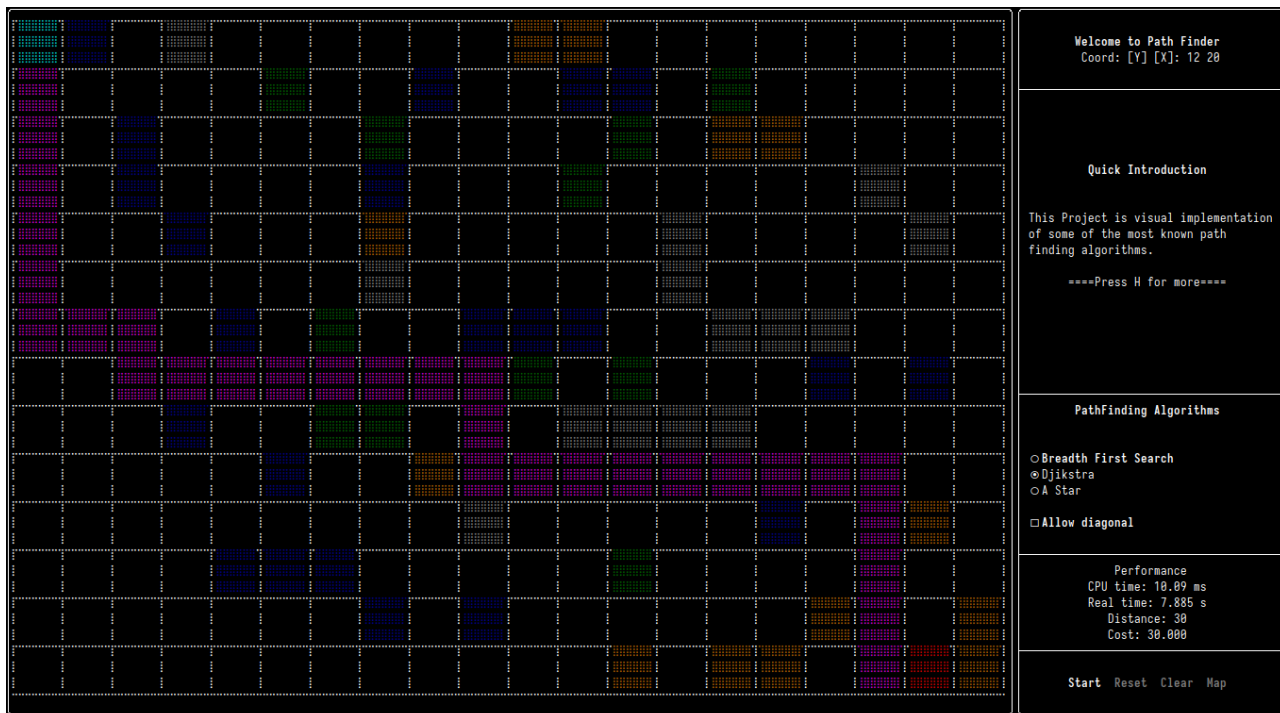
Welcome to Path Finder Coord: [Y] [X]: 11 20
Quick Introduction This Project is visual implementation of some of the most known path finding algorithms. ====Press H for more====
Pathfinding Algorithms <input checked="" type="radio"/> Breadth First Search <input type="radio"/> Dijkstra <input type="radio"/> A Star <input type="checkbox"/> Allow diagonal
Performance CPU time: 11.01 ms Real time: 8.169 s Distance: 30 Cost: 98.000
Start Reset Clear Map

Breadth First Search cu diagonală

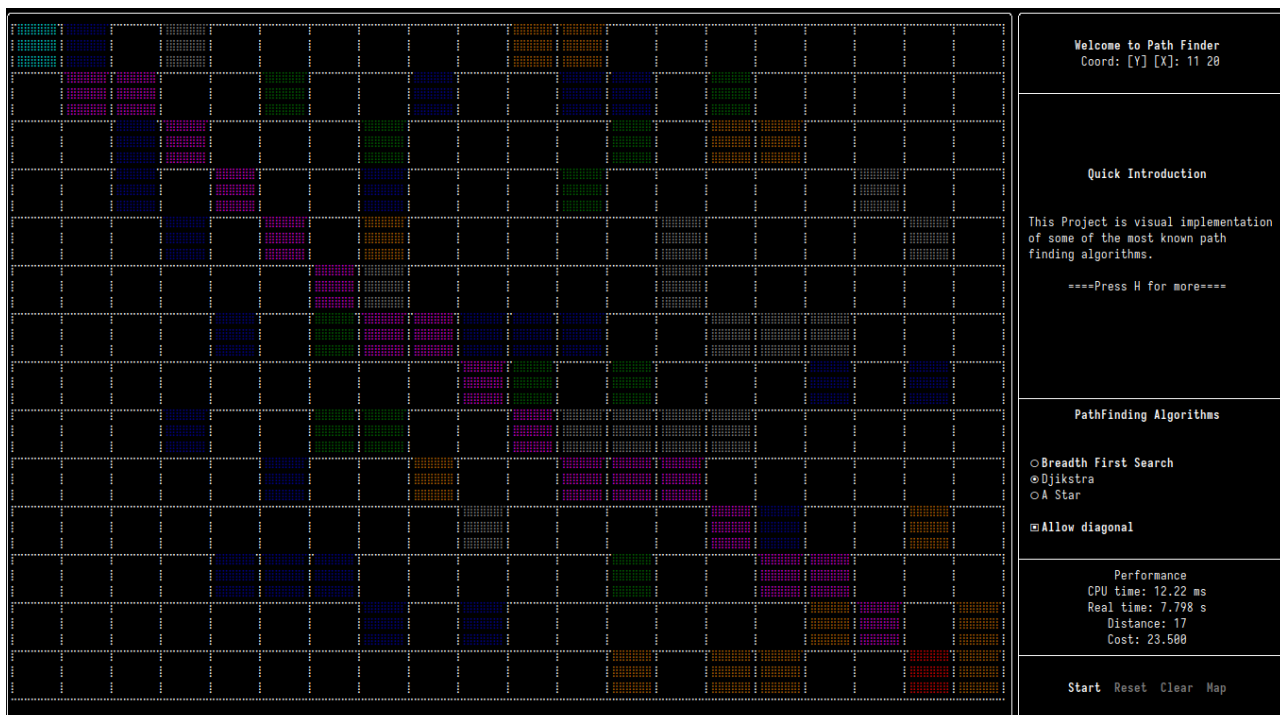


Welcome to Path Finder Coord: [Y] [X]: 18 20
Quick Introduction This Project is visual implementation of some of the most known path finding algorithms. ====Press H for more====
Pathfinding Algorithms <input checked="" type="radio"/> Breadth First Search <input type="radio"/> Dijkstra <input type="radio"/> A Star <input checked="" type="checkbox"/> Allow diagonal
Performance CPU time: 15.40 ms Real time: 8.151 s Distance: 17 Cost: 34.200
Start Reset Clear Map

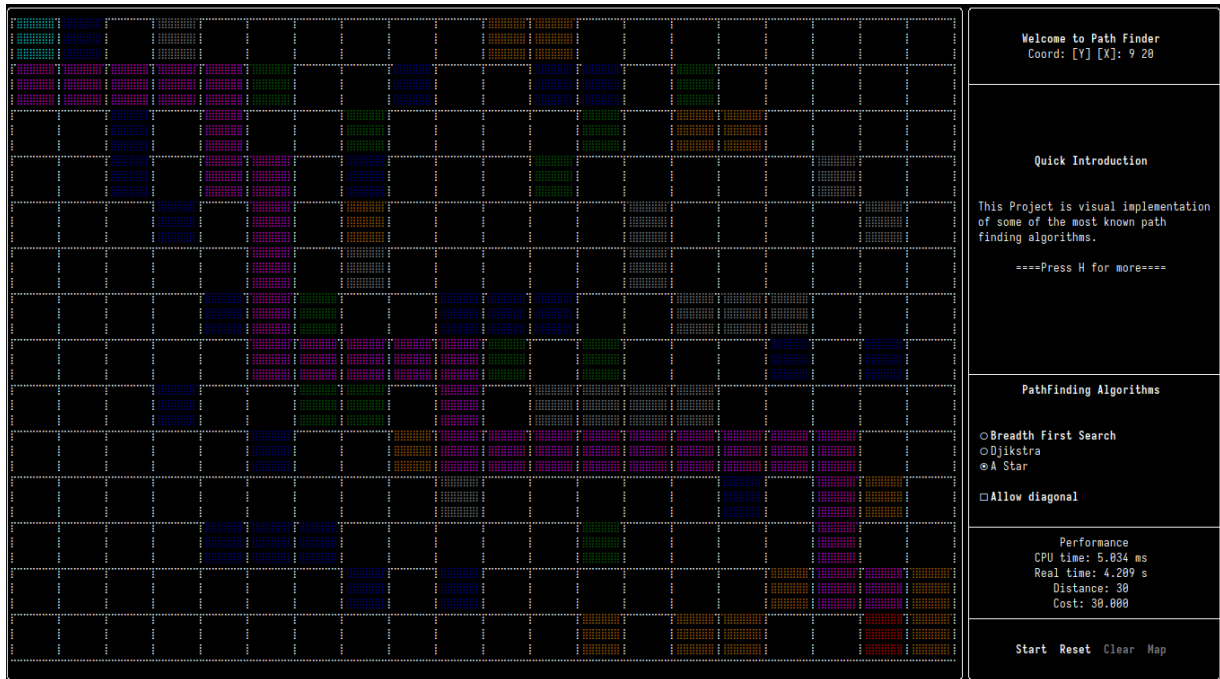
Dijkstra



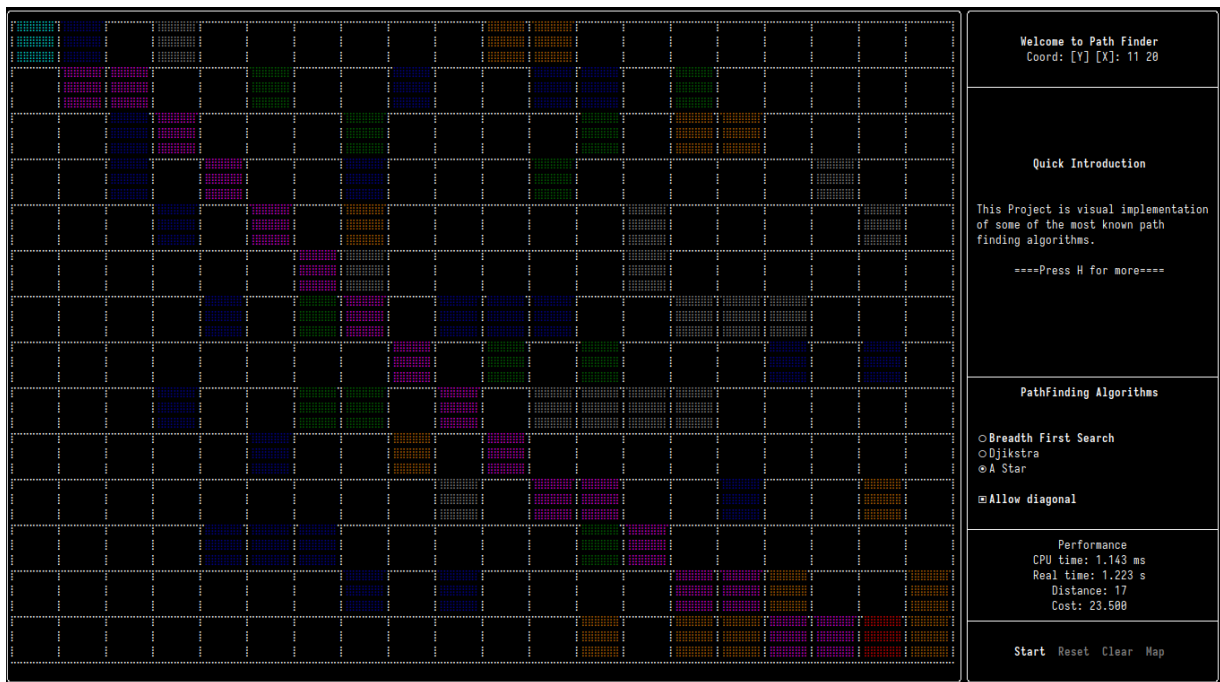
Dijkstra cu diagonală



A-Star



A-Star cu diagonală



	CPU time	Real time	Distance	Cost
BFS	11.01 ms	8.169 s	30	98.00
BFS diag	15.40 ms	8.151 s	17	34.20
Dijkstra	10.09 ms	7.885 s	30	30
Dijkstra diag	12.22 ms	7.798 s	17	23.50
AStar	5.034 ms	4.209 s	30	30
AStar diag	1.143 ms	1.223 s	17	23.50

Capitolul 4

Manual de utilizare

Prim scop al aplicației este găsirea drumului între două puncte pe o hartă definită cu diferite obstacole.

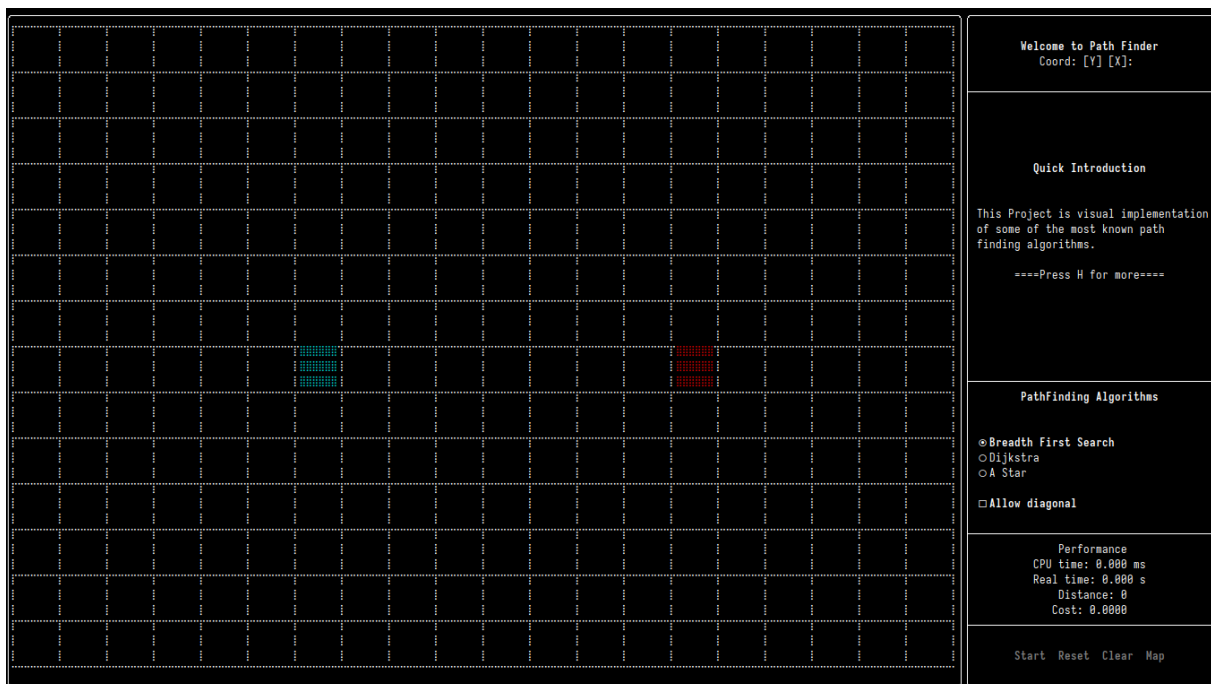
La deschiderea aplicației ca ecran principal avem o fereastră cu informație despre utilizarea aplicației.



După cum se poate citi și în instrucțiune, aceasta se poate ascunde utilizând butonul *ESC*.

Fereastra de bază a aplicației este compusă din două subferestre:

- Fereastra *grid-ului*, pe care se desenează harta ce urmează a fi procesată, și se afișează procesul de căutare, împreună cu drumul găsit.
- Fereastra cu un mic ajutor informațional, meniu cu opțiuni pentru algoritmul ce urmează să fie procesat.



Pentru modificarea poziției punctelor de start și sfârșit, acestea se pot muta prin drag-and-drop utilizând mouse-ul.

Pentru desenarea diferitor tipuri de celule, acestea se pot alege dintr-o fereastră ce apare la click dreapta al mouse-ului.

Utilizatorul poate alege din diferite tipuri de celule definite precum: perete(wall), nisip(sand), padure(woods), apă(water), munți(mountains); fiecare având propriul cost de trecere corespunzător.

Tipul algoritmului de procesare se poate alege dintre algoritmii deja definiți prin alegerea unui radiobutton.

Algoritmul e capabil să proceseze trasee și pe diagonală, efect ce poate fi activat prin checkboxul de desubtul alegerii algoritmilor.

Pentru a începe calcularea traseului se apasă butonul **START** și se așteaptă vizualizându-se procesul de calculare.

...

După procesul de calculare, drumul se afișează cu o culoare distinctă împreună cu informațiile despre procesare corespunzătoare.

Grid-ul poate fi resetat la starea de la începutul programului cu butonul **RESET**.

Însă dacă se dorește editarea hărții, se poate șterge doar drumul apăsându-se pe butonul **CLEAR** apoi pe butonul **PATH**. Iar pentru ștergerea tuturor celulelor cu nemodificarea poziției punctelor de *start* și *end*, se poate apăsa butonul **CLEAR** apoi butonul **ALL**.

Butonul map servește drept extragere de hărți salvate în fișier cu ajutorul combinațiilor de taste descrise în meniul de **Shortcuts**.

Capitolul 5

Concluzii

Proiectul dat poate fi extins cu alți algoritmi precum și diferite tipuri de obstacole.

Acesta necesită o rescriere responsive pentru diferite mărimi de ecran.

În plus e posibil de îmbunătățit UI-ul întregii aplicații, cu alte culori, forme și redesign.

În principiu, am obținut diverse experiențe pe parcursul scrierii acestui proiect. Un lucru mai eficient cu Git, împreună cu noi concepte aflate.

O înțelegere și utilizare minimală a conceptului de *threading* în aplicație.

Lucru mai efektiv și înțelegere a documentației, codului "străin".

Concepte de lucru cu diferite structuri de date, precum și librăria standard a limbajului C++.

Câteva concepte despre lucrul cu terminalele precum modificarea *state-ului* pentru obținerea datelor de la mouse, în format de cod, resetarea stării terminalului la starea normală la închiderea aplicației.

Capitolul 6

Bibliografie

6.1 Articole

<https://en.cppreference.com/w/cpp/thread/thread/detach>

https://en.cppreference.com/w/cpp/thread/sleep_for

<https://www.codespeedy.com/dictionary-in-cpp/>

<https://en.cppreference.com/w/cpp/container/map>

<https://www.redblobgames.com/pathfinding/a-star/introduction.html>

<https://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html>

<https://www.redblobgames.com/pathfinding/a-star/implementation.html>

6.2 Surse Diverse

<https://arthursonzogni.github.io/FTXUI>

https://github.com/ebarmas/minesweeper_ftxui/blob/main/src/minesweeper.cpp

<https://unix.stackexchange.com/questions/497859/text-from-standard-input-not-visible-after-498042#498042>

<https://stackoverflow.com/questions/618511/a-proper-way-to-create-a-matrix-in-c>

<https://stackoverflow.com/questions/2076624/c-matrix-class>

<https://stackoverflow.com/questions/62735210/adding-a-struct-into-a-map>

<https://stackoverflow.com/questions/8963208/gdb-display-output-of-target-application-in-a-s>