Project title:

RObot path finding

course:

data structures

**Date**: january 4,2021

Prepared for

**dr. amna khan**

prepared by

**ayyan babar**

**nafeesa arif**

TABLE OF CONTENTS

1.abstract 3

project Statement 3

summary 3

2.description 4

Introduction 4

objectives 4

functionalities 4

3.implementation 5

Programming methodology 5

Requirements of solution 5

data structures used 5

Flowcharts of algorithms 6,7

demonstrations 8

Functions 9

4.experimental evaluation 12

Testing & optimization 12

different Algorithms 12

different data structures 13

Readability 13

Manageability 13

Reusability 13

5.conclusion 13

Work Cited 13,14

**abstract**

**Project Statement**

One of the most researched topics in the field of autonomous robotics is the path finding of the robot. How would the robot determine the path provided that there are obstacles in the path and find the shortest distance in order to pronounce it the best suitable path?

**Summary of project**

* **Description:** 
  + **Introduction.** The project is of Robot Path Planning to find the shortest optimal distance between two points, given that there are obstacles and cost should be least.
  + **Objectives.** The objectives include successful implementation of path finding algorithms using C++ language features and Graphics
  + **Functionalities.** The functionalities include GUI, implementation of Algorithms, Manual allocation of marks, switching between algorithms
* **Implementation:**
  + **Programming Methodology.** The programming methodologies of Object-oriented and Logical Programming are used.
  + **Functions.** Several functions are explained
  + **Requirements.**  The requirements include identification of inputs and outputs and alternative solutions and are categorized based upon the data structures and algorithms used. The flowcharts are also shown.
  + **Figures.** Figures of the project screen are shown.
* **Experimental Evaluation.** 
  + **Testing.** The testing is based upon data structures used, algorithms implemented, and other aspects included along with the feasibility and comparison with alternatives.
  + **Optimization.** The optimization is done with reference to the testing. Furthermore, the properties of Reusability, Readability and Manageability were encouraged.
* **Conclusion.** 
  + **Concluding Statement.** The project was successfully implemented when errors were treated, and the project was evaluated.
  + **Works Cited**

**Description**

**Introduction**

The project of Robot Path finding by providing the source and destination aims at finding the shortest distance that is the best suitable and follows optimal criteria. There is a possibility of multiple paths from source and destination. There are walls/obstacles created manually and the path is chosen which is the shortest of all the paths i.e. the cost is least.

**Objectives**

The objectives of this project are as follows:

* To successfully implement the path finding algorithms: A\* algorithm and Dijkstra algorithm
  + To achieve the goal of creation of the Graphical Robot Path Finding by using features of C++language and maximum optimization

**Functionalities**

The functionalities of the project include:

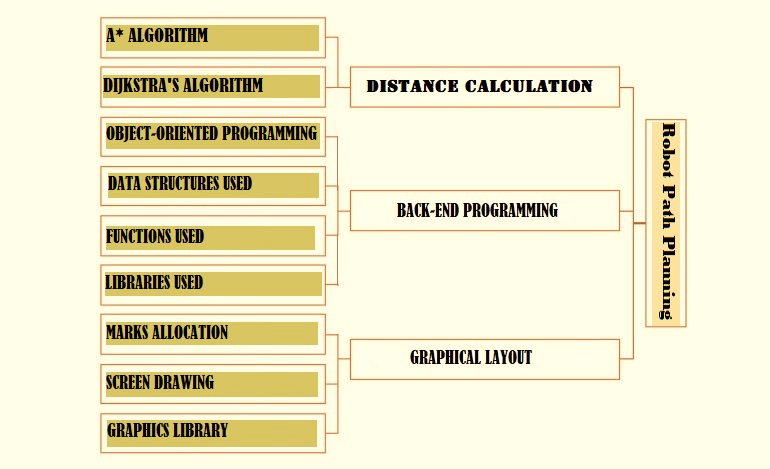
* Graphical User Interface (GUI)
* Option to switch to A\* algorithm from Dijkstra algorithm and vice versa
* Manual selection of source and destination points
* Change of destination and source while the program is running
* Implementation of two algorithms, namely A\* algorithm and Dijkstra algorithm
* The walls are colored dark blue, source is green, and destination is red

*Shown are the figures on Pg.*

**implementation**

**Programming methodology**

The programming methodologies used in the project are Object-oriented programming (based around the objects that are parts of the problem) and Logical Programming (based on logical units performing defined roles).



*Fig.01 Bottom-Up Approach of Logical Programming*

**Requirements for Solution**

The requirements for the solution involve the aspects needed to create a solution of the project statement. It also includes the identification of the problem inputs and outputs and provision of alternative solutions, which is discussed later on. The usage of data structures and flowcharts of algorithms are also presented as requirement for the solution.

* + Data Structures Used:

The data structures used are as follows:

1. Queues
2. Priority Queues
3. Linked List
4. Graph
   * Algorithms and their flowcharts:

There are two algorithms used in the project:

1. Diagram

   Description automatically generated***A\* algorithm***  
   The A\* search algorithm is a refinement to the shortest path algorithm that directs the search towards the desired goal rather than exploring nodes based simply on distance from the initial vertex. The key idea is to define a heuristic function f(n)=h(n)+g(n), that estimates how far a given vertex v is from the goal vertex,

where **h(n) is an estimate of the minimum cost from any vertex n to the goal** and **g(n) is the cost of moving from the initial cell to the current cell**. To simplify, it is the sum of all the cells that have been visited since leaving the first cell.

*Fig. 02 Flowchart of A\* Algorithm*

1. ***Dijkstra’s algorithm***

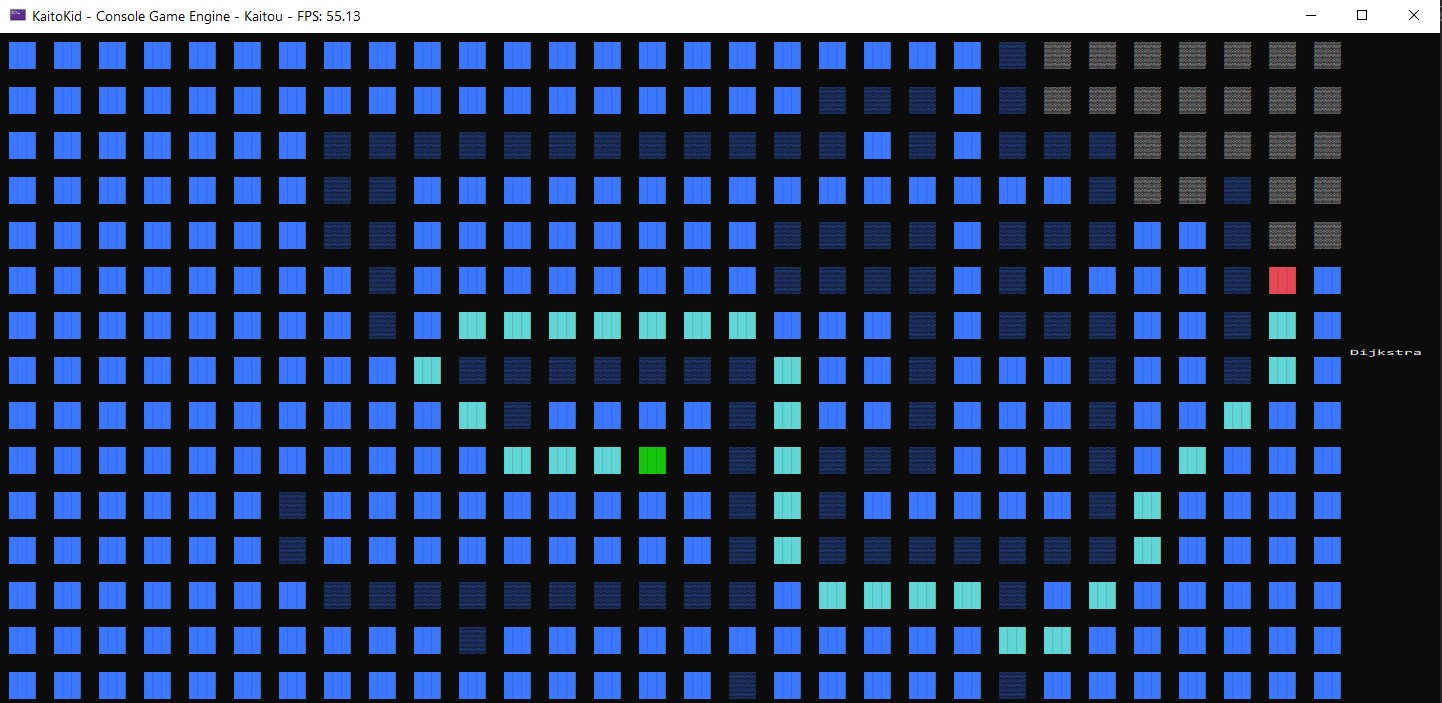
In comparison to A\* algorithm, Dijkstra's algorithm greedily visits nodes in order of increasing distance from the initial vertex. The destination node’s place is not considered and so, algorithm will spend equal time in exploring nodes that are along the optimal path and even those that are in opposite direction from the desired goal. It expands almost radially.

Moreover, when h(n) and g(n) become equal, the A\* algorithm becomes Dijkstra’s algorithm.

Diagram

Description automatically generated

*Fig. 03 Flowchart of Dijkstra’s Algorithm*

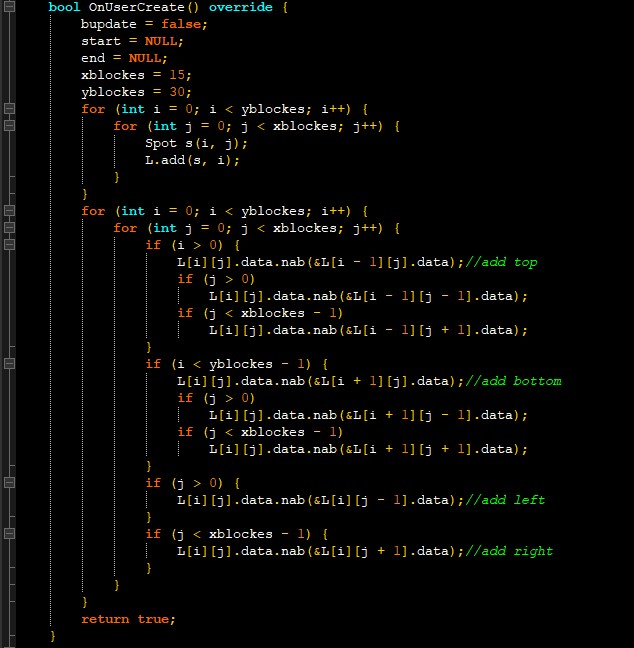
**A picture containing chart

Description automatically generateddemonstration**

*By Dijkstra’s Algorithm*

*By A\* Algorithm*

**Functions**

The project relies extremely on the functions made according to the needs of the project. There are various functions whose codes would be given as well. (from test.cpp)

1. OnUserCreate:

This function creates the board of size x rows and y columns and creates a link among the matrix of blocks and returns true to ensure.

1. Text

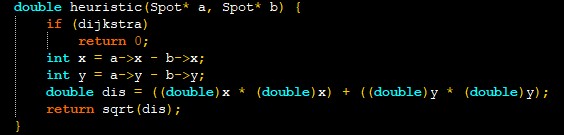
   Description automatically generatedrefresh:

This function refreshes the board components to make it ready for the next

drawing and resetting the values, open list and close list, resetting

heuristic and cost.

1. heuristic:

This calculates the heuristic using the Euclidean distance. //\*2

1. visualHuristic:

This is the function to be used if g(n)=h(n), to check or manipulate the values.

Graphical user interface

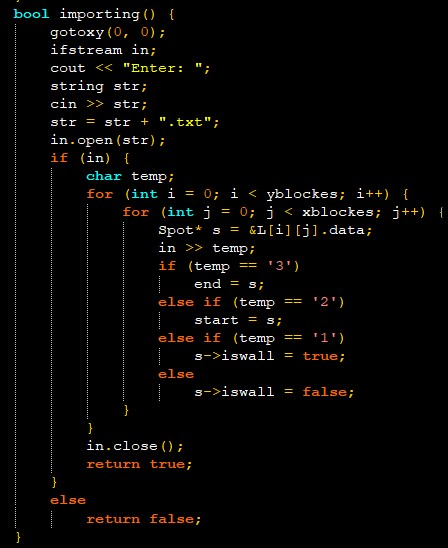
Description automatically generated

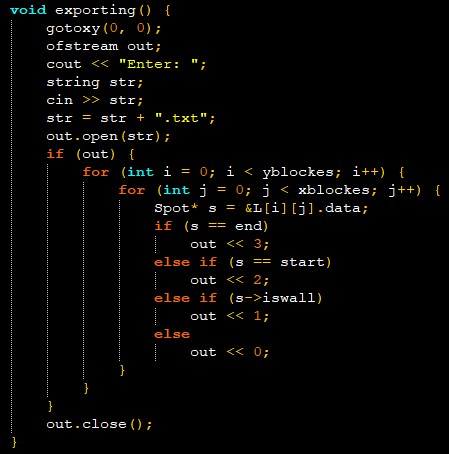
1. A\_star:Text

   Description automatically generated If the board is to be updated, only then A\_star would work. The open list (priority queue containing the element with least f(n) at top) is used to check the likely visiting nodes, the ones that need to be checked. The close list (queue) is populated by the elements recovered from open list. The loop iterates up to number of neighbors, checks open and close queues ignoring the walls and also checks the diagonals. If the element is already in open and new element g(n) value is less than previous g(n), it chooses the path, else if the g(n) is greater, the neighbors are searched from current node and works for all neighbors.

If the end node is reached, it is in open list and check is applied and backtrack() function is called, which changes the color.

1. Exporting

This function is used to write/export to the file.



1. Importing

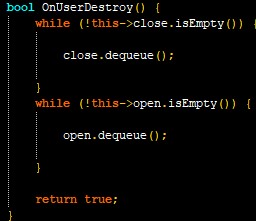
It imports from the file the input of the path. If file is loaded, the board is refreshed and returns true, else returns false.

1. Text

   Description automatically generatedOnUserUpdate:   
   This is called on every frame, it is used to draw the graphics. This is the main function which handles the graphics, input and output. If ESC key is pressed, the program is closed. If ‘E’ is pressed, it exports the location of source, destination and walls in a file. If ‘I’ is pressed, the import function starts working. If a click is made, the wall is toggled. If a click is made on the wall, the wall is removed. The end point is set to NULL.

If a click is made outside the border, the algorithms are switched. The FILL statements print the rectangle with size and border, with respect to the next one.   
The value of g(n) is Infinite if it is unchanged. If it is not a wall and not in open list, the value of g(n) is infinite.  
The source is colored green and destination red.

1. OnUserDestroy:It behaves as a destructor, to free the memory and is called when OnUserUpdate returns false. Until the close list is not emptied, it empties the close list.



Text

Description automatically generated**experimental evaluation**

(cont.) OnUserUpdate

**testing and optimization**

The testing was done to experimentally evaluate the project and analyze the results. The testing of the Program was based upon two factors:

1. Algorithms: The purpose or case was to find the shortest path and the following algorithms were tried and out of these the best ones were chosen.

|  |  |
| --- | --- |
| Algorithm | Feasibility |
| Depth-first Search | The algorithm was not suitable as it did not return the optimal path, but returns the path from which the goal is reached first. |
| Breadth-first  Search | This algorithm also failed as it did not return the desired output. It does not tell how far the path has been crossed and how far the goal is. |
| A\* | It calculates heuristic function's value at each node on the work area and involves the checking of too many adjacent nodes for finding the optimal solution with zero probability of collision. |
| Dijkstra | This algorithm found the shortest optimal path by calculating the distance to each unvisited node. |

*Table 1: Evaluation on basis of Algorithms*

1. Data Structures Used

|  |  |  |
| --- | --- | --- |
| data structures Used | alternative | Comparison |
| Queues | Stacks | Since priority was to be considered, priority queue was more favorable than stacks as stacks only allow the topmost element to be extracted. |
| Graphs | Array | Graphs can be expanded and are dynamic as used using linked list, while arrays cannot be expanded as are static |

*Table 2: Evaluation on basis of Data Structures Used*

1. Other Aspects

|  |  |  |
| --- | --- | --- |
| Name | alternative | Comparison |
| Euclidean Distance | Manhattan Distance | Euclidean distance is the length of the segment between two points whereas Manhattan distance is the sum of distances of all dimensions |
| Game Console Engine | SFML | Game Console Engine visualizes better than SFML and operates faster. |

*Table 3: Evaluation basis of other Aspects*

The code was also optimized on the following basis:

* **Readability:** The code was easy to be read.
* **Manageability**: The code is easy to be maintained i.e. if a small change is to be occurred, it can be done easily.
* **Reusability**: Several segments of code for a specific purpose (functions) were used to shorten the longevity of the code.

**conclusion**

To summarize, the project of Robot Path Planning was a success, as it fully fit the criteria of shortest distance and optimization. Moreover, further improvements can be made. However, the design and implementation, in a nutshell, was satisfactory at this level.

**works cited**

Zidane and Ibrahim (September 2018), Wavefront and A-Star Algorithms for Mobile Robot Path Planning [Web Publication] (flowcharts) Retrieved from: <https://www.researchgate.net/publication/319404402_Wavefront_and_A-Star_Algorithms_for_Mobile_Robot_Path_Planning>

Information on Algorithms, Retrieved From: <http://www.cs.cornell.edu/courses/cs312/2007sp/recitations/rec26.html#:~:text=An%20obvious%20choice%20for%20a,current%20node%20and%20the%20destination>.

<http://theory.stanford.edu/~amitp/GameProgramming/Heuristics.html>

<https://journals.sagepub.com/doi/full/10.1177/1729881416663663>

https://en.wikipedia.org/wiki/A\*\_search\_algorithm#:~:text=A\*%20is%20an%20informed%20search,shortest%20time%2C%20etc.).

Information on Programming Methodologies, Retrieved from:

<https://www.tutorialspoint.com/programming_methodologies/programming_methodologies_introduction.htm>