ELEE 4034

**Robotics and Mechatronic Systems Egineering Design II**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**A\* searching algorithm

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Instructor: Dr. Mariam Faied

Author 1: Liang Mi

Author 2: Jingya Wang

Author 3: Fujun Yang

Author 4: Chunxu Zhao

Content

**1.introduction**

**2.Methodology**

**3.Results**

**4.Discussions and Conclusion**

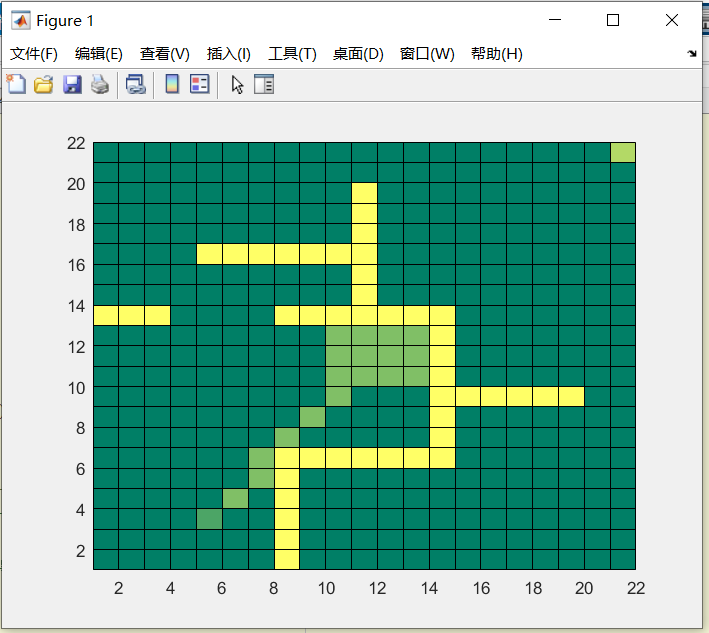
1.Introduction

A\* (pronounced "A-star") is a graph traversal and path search algorithm, which is often used in computer science due to its completeness, optimality, and optimal efficiency. One major practical drawback is its O(bd)space complexity, as it stores all generated nodes in memory. Thus, in practical travel-routing systems, it is generally outperformed by algorithms which can pre-process the graph to attain better performance, as well as memory-bounded approaches; however, A\* is still the best solution in many cases.

Peter Hart, Nils Nilsson and Bertram Raphael of Stanford Research Institute (now SRI International) first published the algorithm in 1968.[4] It can be seen as an extension of Edsger Dijkstra's 1959 algorithm. A\* achieves better performance by using heuristics to guide its search.

1. Methodology

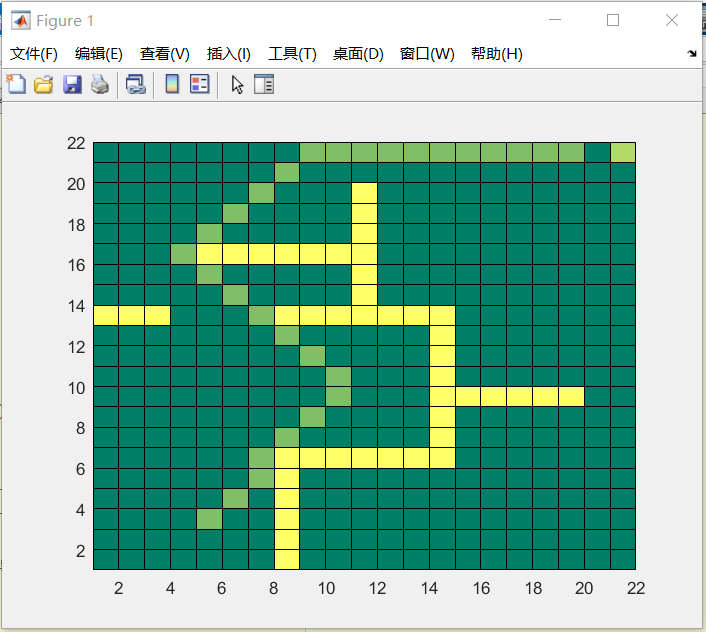
Firstly, we use map function to build a map. The basic unit of the map is cell. Every cell has a coordinate. Before the other functions work, we only know two cells’ coordinates: the starting cell’s coordinate and the destination coordinate. We set the starting cell’s coordinate is (xs,ys) and the destination cell’s coordinate is (xf,yf).

 The next step is using surrounding function to calculate the 8 cells around the starting cell. The starting cell be marked “2”, which means starting cell has been put into the close-list. The cells around the starting cell will be marked “1”, which means they’re been put into open-list. Besides these, the cells being marked “0” means these cells didn’t been put into open-list. Every cell will be put into close-list if it leaves the open-list.

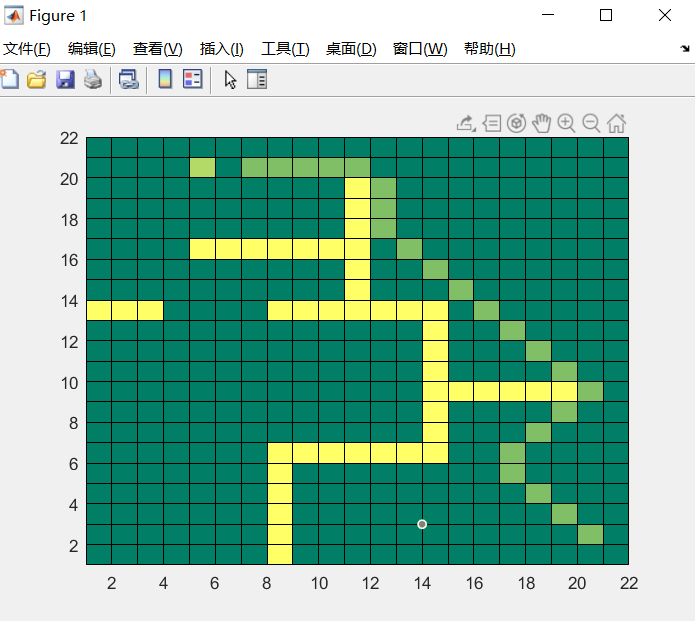
And program apply different reaction to cells with different characteristics. We use judge\_character function to judge every cells which characteristic it belongs to. If the cell being judged as a barrier or in the close-list, there is no calculation with the cell. If the cell is listed in the open-list, we use old\_G\_funtion to calculate a new G value for it. If the new G value is smaller than the old G value, update the G value. If all the surrounding cells’ G values have been compared by Comparing\_G function, update open-list and close-list. The path move forward.

As for the new cells, the same steps will be repeat till reaching the destination.If there are two or more cells have the same F value, add the cell has the smallest F value at the bottom of open-list into the close list. When reached the destination, the close-list finished, we use the Mark function to find the path through father cells.

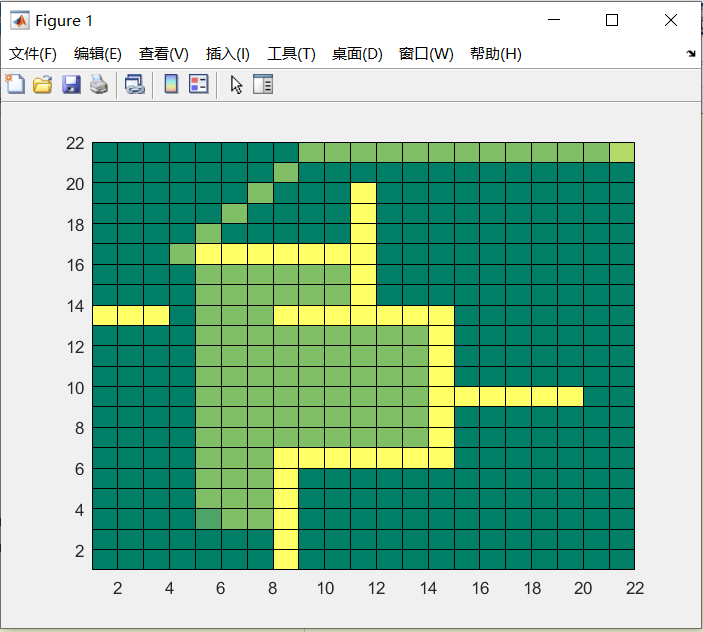
1. Result

 When the destination cell added in the close-list, we use the Mark function to find the path through father cells. So we get this, which is nearly the shortest way. Our implementation of A\* algorithm is successful. The result is shown in the below.

Another map result：



1. Discussions and Conclusion

 When A\* terminates its search, it has found a path from start to goal whose actual cost is lower than the estimated cost of any path from start to goal through any open node. When the heuristic is admissible, those estimates are optimistic, so A\* can safely ignore those nodes because they cannot possibly lead to a cheaper solution than the one it already has. In other words, A\* will never overlook the possibility of a lower-cost path from start to goal and so it will continue to search until no such possibilities exist. Sometimes, it will cost a lot of time. The example is shown as below.

While the admissibility criterion guarantees an optimal solution path, it also means that A\* must examine all equally meritorious paths to find the optimal path. To compute approximate shortest paths, it is possible to speed up the search at the expense of optimality by relaxing the admissibility criterion.