POLITECNICO DI MILANO
DEPARTMENT OF AEROSPACE ENGINEERING

Operations Research Project

AA 2023-2024

Professor: Maurizio Bruglieri

Person Code	Surname	Name
10713569	Grandinetti	Roberta
10730683	Nuccio	Gabriele

November 19, 2024

Contents

1 Introduction

Among the broad set of optimization problems, the shortest path problem finds a wide range of real applications, especially in the transportation and network field, gaining a lot of interest in recent years. The resolution of the minimum cost path problems consists in the search for a path from a given origin to a given destination that optimizes the sum of the costs associated to the arcs. A big set of algorithms are specialized in the resolution of this famous and common problem i.e. Dijkstra, Dynamic Programming, Path planning and others. [10]

This report is intended to analyze the shortest path problem in the framework of a multi-objective optimization, which involves multiple objectives that often conflict with each other. Typically, achieving the optimal value for all individual objectives simultaneously is impossible. Consequently, the goal of multi-objective optimization is to identify a range of trade-off solutions, enabling decision-makers to choose the most suitable option from these alternatives. [13]

The specific problem under study is called multimodal multi-objective path planning (MMOPP) optimization which is a particular extension of the multi-objective shortest path problem. The resolutive approach proposed is based on the integration of Pareto optimality into every step of the multi-objective A* search, which is therefore renamed as A*-PO, to exactly solve the MMOPP, finding all the Pareto optimal solution at the end.

To validate the real-world applicability of A*-PO and further showcasing the algorithm's viability in practical scenarios, a case study involving a planetary exploration rover is conducted. To this purpose the open-source software 'Multi-objective A* algorithm for the multimodal multi-objective path planning optimization', by Jin Bo (winner of 2021 IEEE Congress on Evolutionary Computation(CEC))[12] has been selected and adapted to our particular case study.

2 MMOPP problem

The multimodal multi-objective path planning aims to find the Pareto optimal paths from a start area to an end area on a grid map, while passing through several designated must-visit areas. It is a typical optimization problem raised in different fields such as urban transportation and motion planning, especially in robotics applications, including space rovers for planetary exploration, on which the simulation will focus. 5

Generally a single objective function is not sufficient to characterize path planning problems. Additional objectives to distance length, such as travel time, energy and other costs may all need to be optimized at the same time. Therefore MMOPP in its general form of multi-objective optimization problem can be stated as:

$$\min(\mathbf{f(x)}) = (f_1(\mathbf{x}), ..., f_N(\mathbf{x})) \tag{1}$$

where N is the number of the objective functions to be minimized at the same time. $\mathbf{x} \in$ X are the decision variables of the problem. A feasible solution $\mathbf{x}$ is said to *dominate* another feasible solution $\mathbf{x'}$ if $\mathbf{f(x)}$ dominates $\mathbf{f(x')}$ and $\mathbf{x}$ is said to be *Pareto optimal* if it is not dominated by any other feasible solution in X. The corresponding points in the objective space of the Pareto set form the Pareto front.

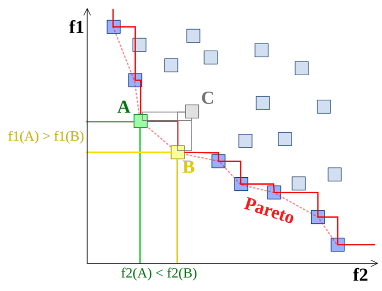


Fig. 1: Two dimensional Pareto space, where A and B lie on the Pareto front[4]

The goal of a Pareto evolutionary algorithm is indeed to find a set of solutions along the Pareto front. [2] In recent decades, evolutionary algorithms have become the primary approach for solving multi-objective optimization problems, largely due to their use of the population concept, which enables simultaneous exploration of multiple solutions in a single simulation. Examples include NSGA-II and SPEA2. [3] However, most existing multi-objective evolutionary algorithms focus solely on approximating the Pareto front but often disregarding the distribution of solutions in the decision space. Instead having multiple Pareto optimal solutions enhances the reliability of the decision-making process. Past studies have applied also genetic algorithms to the MMOPP, but these have the disadvantages of computational complexity and suboptimal solutions.[11] A different approach is here proposed, according to which the global path plan is calculated with the A* search method, which guarantees to deliver a complete and optimal solution that minimizes the path cost.

3 Case study

The applications of navigation planning are numerous, but to better demonstrate the application of the algorithm, it has been opted for a specific example of mobile robot path planning. An interesting case, also related to our field of study, is the analysis of the planned path for a Mars exploration rover.

The race to Mars exploration is an ongoing challenge for the space industry, which already achieved outstanding goals (just think of *Curiosity*). On the diverse and still unknown Mars landscape and with a limited amount of available resources on-site, navigation precision and efficiency are crucial for the completion of missions themselves. Therefore it's essential to detect and follow the shortest and the most comfort path from a start position to a goal position while at the same time avoiding the obstacles along the course.

The proposed model develops accordingly to this application, consequently the objective functions to be minimized are selected in a coherent way:

- The length of the path between the start and goal points;
- Number of traversed points with excessive elevation or depression.

Minimizing the length of the path is the better choice to reach the objective area while saving time. The second objective function instead focuses on excessive elevation or depression points on the Mars surface that have to be avoided, if possible, because they would require an higher effort for the rover to overcome them and rover's functionalities may be jeopardized. Beside this, not traversable region have to be highlighted and avoided, especially for rover using wheels e.g *Curiosity*, which wouldn't be able to roll over high obstacles in the Mars landscape [6]. This is a crucial aspect to be taken into account in the problem formulation as areas where the robot cannot pass through.

The considered scenario is based on different metrics and constraints (obstacles) along the path, whose positions are fixed and known *a priori* during all the simulation. Also dynamic obstacles that vary their position with time may be considered but this requires an higher complexity and in the case of space exploration a static approach is the more reliable, making the algorithm lean towards a global path planning behaviour.

4 Model and Algorithm

The onward sections will focus on the resolution of the aforementioned problem for our specific case study, by means of a suitable open-source software [12]. The workflow of the 'Multi-objective A* algorithm for the multimodal multi-objective path planning optimization' by Jin Bo will be briefly outlined with an emphasis on some key aspects in the modelling and optimization procedures.

4.1 Occupancy grid

During the travel, the rover needs to go from a start area to a goal area. The configuration spaces, namely the robot representation of the world, for global path planning algorithms (as for the one under study) can be represented by an occupancy grid map built in this way:

- *(x,y)* are the square area associated with the x-th column and the y-th row;
- **binary** map where 0 means that *(x,y)* is passable and 1 the opposite;
- For each area *(x,y)* occupied by the robot, there are four possible path steps (two vertical and two horizontal);
- each area *(x,y)* is associated with a cost vector that accumulates the cost of all the areas passed through. The cost vector is formed by the objective functions considered in the problem.

Before solving the problem, a preliminary feasibility check is performed: if the goal position cannot be reached from the starting one, namely it is *not connected*, there is no feasible solution and the algorithm stops.

The grid can be then subjected to a reduction by considering both the unreachable areas and the articulation points, which is a node whose removal together with its associated edges disconnects the graph. [8] A time saving operation, in fact, is to remove the connected areas separated by an articulation point that cannot be contained in the path. The process that allows to obtain the reduced map from the starting one is based on the DFS (depth-first search) spanning tree algorithm. It starts exploring the graph from one node and explores its depth before the backtracking. In other words when there are no more nodes along the current path, to find another node to explore, one follows backwards the exactly same path.

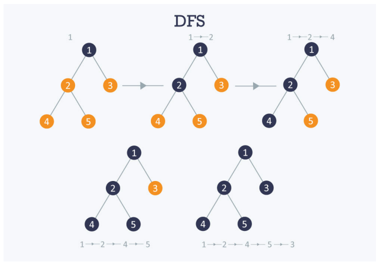
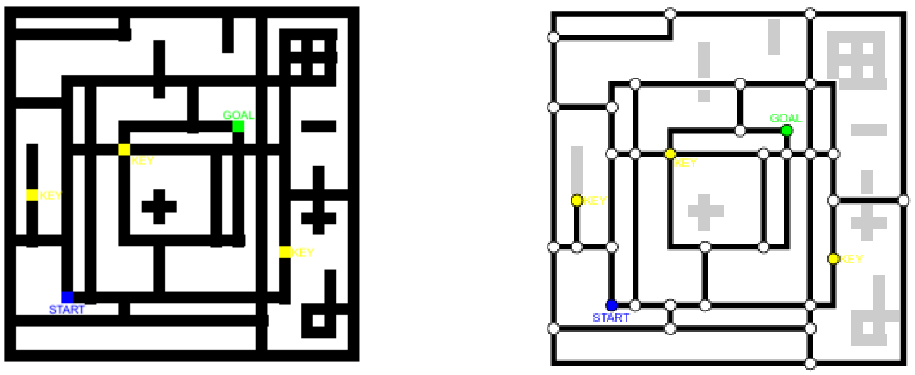


Fig. 2: DFS spanning tree algorithm[1]

The aforementioned algorithm performs 2 steps [7]:

- Selection of a node of the graph to travel across;
- Pass through the nodes adjacent to that vertex.

All the nodes will be visited on the current path until every unvisited node have been traversed. After that the next path will be selected. Thus,the feasibility of the MMOPP problem can be proved in the obtained reduced map in order to reduce the scale and the complexity of the problem. Once the reduced map is detected, it is then modelled as an undirected graph $G = \{N, M\}$ with N nodes and M edges. Where each node is a mandatory area or an area of degree 3 or 4, while the edge is a sequence of continuous areas of degree 2 between two nodes. The following figures show the initial map and the undirected graph built directly from the reduced map, where the circles represents a node and the edges are the lines linking them.



(a) Initial map (b) Graph

Fig. 3: Complete Map and correspondant Graph

4.2 A*-PO Algorithm

In order to better describe in detail the workflow of the algorithm, first it is needed to define what is a "status". The status is defined by a bit array of dimension K where the k-the element is a bit 1 or 0. 1 means that the k-th area currently inspected has been visited, 0 otherwise. The set of the possible status goes from 0 (start) to $2^k - 1$ (goal).

A node-status pair *(n,s)* is assigned to every node, through which a Tentative set of cost vector at node n and status s is defined for every couple in order to store the non-dominated solutions of all the identified paths from initial pair to the node-status pair at every moment of the algorithm.

When the A*-PO terminates the set of the goal node-status pair obtained represents in an exact way the searched Pareto front. To meet the goal of multimodal optimization it is needed to record all the Pareto optimal paths in a vector of predecessors defined by a triplet *(n,s,c)* that contains the node-status pair with the cumulative cost c. To meet this objective a predecessors list that led to the couple *(n,s)* for any of the cost c in the tentative set is defined. Any of this list is indicated in the form: *(n',s',c')* that is the preceding triplet of the one currently examinated.

The algorithm at each step selects the best triplet to expand next, in particular the triplet that has a non-dominated cost estimated in this way $f = c + h(n,s)$ where $h(n,s)$ is introduced as an heuristic function that tries to estimate the ideal cost from the current start to the goal node-status pair. By doing this, the chosen triplet has always a non-dominated cost. If, instead, the estimated cost f is dominated by the goal set, this implies that it is impossible to find any paretian solution.

The algorithm uses also a priority queue Q to maintain all the triplets that needs to be updated, when Q becomes empty the algorithm terminates. The search algorithm gains a lot of efficiency by the use of this queue.

Algorithm 1 Multi-objective A* algorithm

```
global $\tau$: tentative sets;
  global $P$: predecessors lists;
    Save $cost[x_{start}][y_{start}]$ in $c_{start}$;
    Insert $c_{start}$ into $\tau_{n_{start},s_{start}}$;
    Create an empty priority queue $Q$;
    Push $(n_{start}, s_{start}, c_{start})$ into $Q$;
    while $Q \neq 0$ do
      Pick the triplet $(n, s, c)$ with the best value of $f = c + h(n, s)$ from $Q$;
      if n = $n_{goal}$ and s = $s_{goal}$ then
        continue;
      end if
      if $f$ is dominated by $\tau_{n_{goal},s_{goal}}$ then
        Remove $c$ from $\tau_{n,s}$;
        Delete $P_{n,s,c}$;
        continue;
      end if
      for edge $e$ incident in $n$ do
        if $c'$ is dominated by $\tau_{n',s'}$; then
          continue;
        end if
        if $c' \in \tau_{n',s'}$ then
          insert $(n, s, c, e)$ to $P_{n',s',c'}$;
          continue;
        end if
        for $c'' \in \tau_{n',s'}$ that is dominated by $c'$ do
          Remove $c''$ from $\tau_{n',s'}$;
          Delete $P_{n',s',c''}$;
          if $(n', s', c'') \in Q$ then
            Remove $(n', s', c'')$ from $Q$
          end if
        end for
        Insert $c' + h(n', s')$ into $f'$;
        if $f'$ is dominated by $\tau_{n_{goal},s_{goal}}$ then
          continue;
        end if
        Insert $c'$ into $\tau_{n',s'}$, add $(n, s, c, e)$ to $P_{n',s',c'}$ and push $(n', s', c')$ to $Q$
      end for
    end while
```

Algorithm 2 Pareto set algorithm

```
global $X$: set of points of coordinates (x,y);
  global $C$: set of couples (n,s);
    for $c_{goal} \in$ Pareto front do
      Save $x_{goal}, y_{goal}, n_{goal}, s_{goal}$ in $X, C$;
      if n = $n_{start}$ and s = $s_{start}$ then
        Insert $X$ in Pareto set;
      else
        for triplet $(n', s', c') \in$ Predecessors list do
          if $(n', s')$ is not in $C$ then
            Save $(x_{n'}, y_{n'}, n', s')$ with $X, C$ in $X', C'$;
            Repeat recursively the procedure with $(n', s', c', X', C')$;
          end if
        end for
      end if
    end for
```

4.3 Heuristic Function

The role of the heuristic function proposed in the algorithm is to estimate with a lower bound the ideal cost from any node-status pair to the goal node-status pair. It is used to compute the minimum cost between any node by solving a single-objective shortest path problem and is equivalent to find the Hamiltonian path with the least total distance from node n that ends in the goal node. A Hamiltonian path is a particular case of the well known travel salesman problem (TSP) which is represented by a path between two nodes that visit all the vertex exactly once. Since this latter is a NP-complete problem[9], it requires exponential time to find the exact solution so it's really impractical. A heuristic function that approximate the solution is needed. [5]
The idea behind the heuristic function is: the ideal cost h_{ideal} is first initialized by accumulating the cost of the goal node and the costs of all the unvisited must-visit nodes. Then the ideal cost is enhanced by adding the minimum total distance from the current node to the goal node. Moreover, to guarantee the optimality of the algorithm a consistency condition is required. A heuristic is said to be consistent if its estimate is always less than or equal to the estimated distance from any neighbouring vertex to the goal, plus the cost of reaching that neighbour. [11] For every node N and each successor S of N, the estimated cost of reaching the goal from N is:

$$h(N) \leq c(N, S) + h(S) \tag{2}$$

that is basically the triangle inequality.
An inconsistent heuristic functions might overestimate the cost of some path and instead explore other more expensive ones, missing the optimal solution.

At the end of the algorithm all the Pareto optimal solution paths are obtained.

5 Simulation

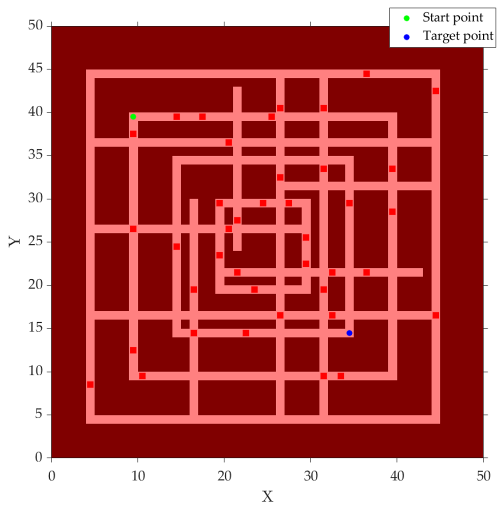
The path planning algorithms are normally tested in simulated mobile robot environments. The adopted code [12] is an open-source software written in C++ 17, while the input data were selected among the test problems available in the software repository. The results are then graphically represented in MATLAB R2024b by the means of a set of build in functions, belonging to the `Navigation Toolbox`.

5.1 Data of the simulation

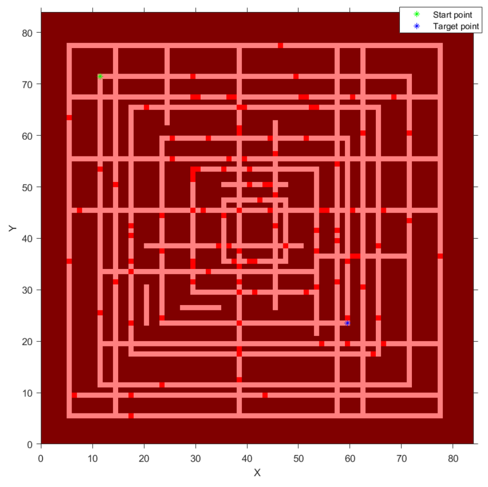
A set of test problems among the ones available in the `Data` section of the software repository, has been analysed in order to find the most suitable one for our case study presented in section 3

`Problem 3` and `Problem 5` are the ones eventually identified and solved. They share similar characteristics, but they are developed on a different level of accuracy, starting from the resolution of the occupancy grid: respectively 50x50 for `Problem 3` and 84x84 for `Problem 5`.

Here below follows the graphical representation of their respective occupancy maps: where the start and target points are chosen (visible as green and blue dots, respectively) and the darker areas are not traversable, which in the scenario of a Mars exploration rover can either represent still unknown - and therefore dangerous - areas or they can highlight the presence of obstacles.



(a) Problem 3



(b) Problem 5

Fig. 4: Occupancy grids of the problems under study

The light red zones show instead the overall track of the eligible paths. It's worth-notice that the reported map is still in its non-reduced form.

5.2 Objectives

Given the inherent nature of an MMOPP problem, as the ones under study, the primary objective function to be minimized is the length of the path. The other additional objectives can can be selected among the available choices provided by the software [12].

The number of crossed '`Red Areas`' is the second objective function that is wanted to be minimized. The location of these area is identified by the red squared markers scattered around the grid 4. In the framework of a planetary exploration to this points is assigned the meaning of areas of excessive elevation or depression of the waypoints (hills or craters in other words), that may damage or limit rover's functionalities.

5.3 Results and comments

The computational results for the MMOPP problem 3 and 5 are here presented:

Problem	**M**	**Initial Map** (#areas)	**Reduced Map** (#areas)	**Graph** *(N,E)*	**A*-PO iterations**
Problem 3	2	623	612	(57,103)	61
Problem 5	2	1727	1689	(118,218)	192

Table 1: Computational results of MMOPP

Where M is the dimension of the problem, namely the number of objective function to be minimized. Initial Map and Reduced Map columns represent the number of passable areas in the respective maps. In the Graph column are reported the number of nodes and the number of edges of the undirected graphs obtained from the maps. The last column contains instead the number of iterations required to the algorithm to solve the problem and is equivalent to the number of triplets that has been placed in the priority queue Q. The latter is an result directly obtained as output by the software. It's immediate to notice that the complexity of the algorithm increases accordingly with the dimensions of the problem.

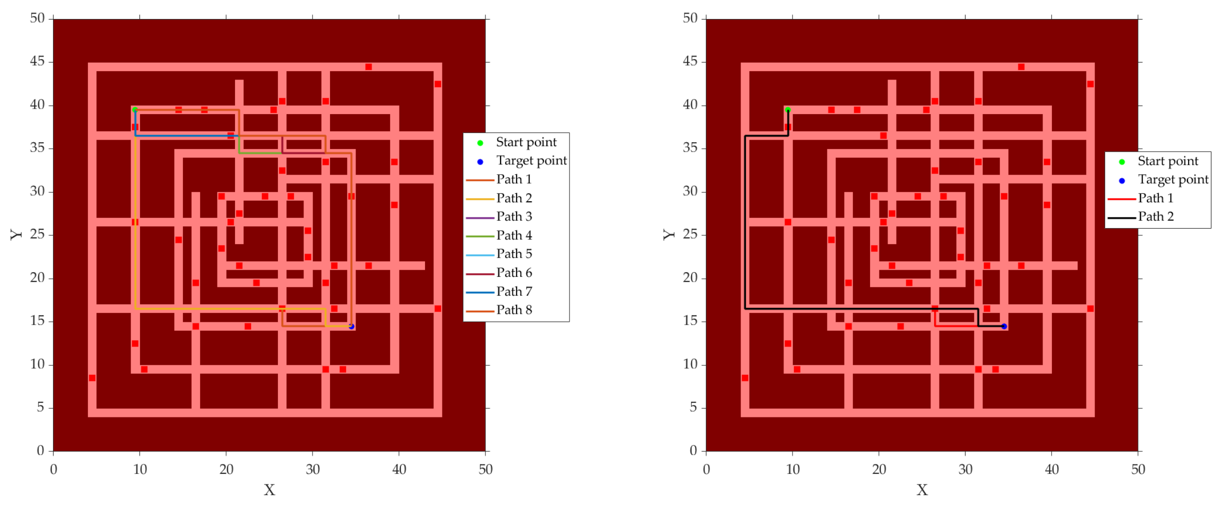
The algorithm eventually provides the values of the objective functions for all the found Pareto optimal solutions can be found, with different values for the first and the second objective functions. The choice of the best solution will be then dictated from the importance given to one of the functions with respect to the other. The results obtained for the 2 problems of interest are summed up in the table below.

Problem	**Objective values** (ObjFun1, ObjFun2)	**Number of pareto optimal paths**
Problem 3	51,3	8
	61,2	2
Problem 5	97,4	6
	101,3	1
	111,2	3
	161,1	8

Table 2: Results of the Problems

What clearly stands out is that more than one Pareto optimal path can present the same values of the objective functions. Therefore they have the same 'approval rating' in the final decision maker process, except if additional constraints of mandatory areas are taken into account.

The results presented in Table 2 are then finally plotted in MATLAB by means of a set of functions belonging to the `Navigation Toolbox`.

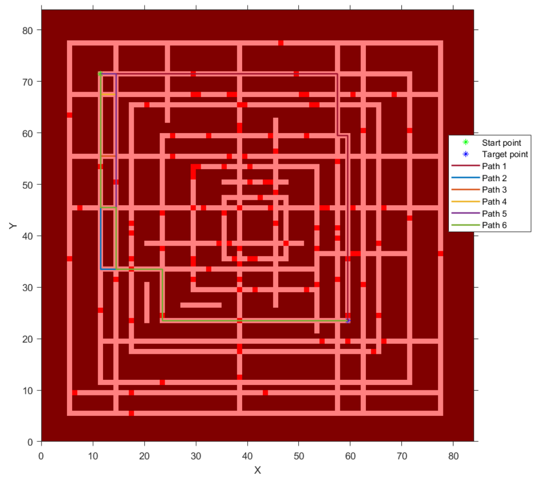


(a) Problem 3 - Objective 1 (b) Problem 3 - Objective 2

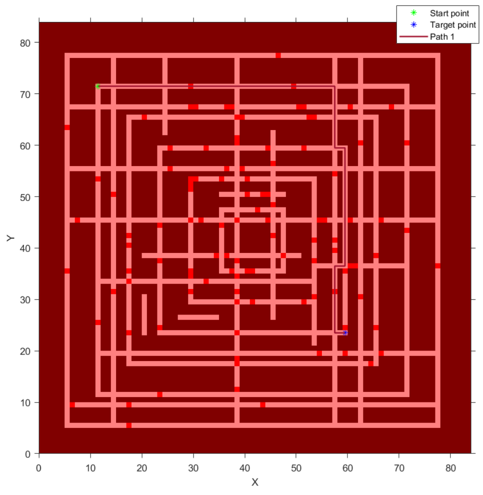
Fig. 5: Pareto optimal paths of problem 3

Problem 3's optimal paths start at the point of coordinate $(10, 40)$ and finish in $(35, 15)$. In Figure 5a all the 8 paths that better minimize the length objective are shown together; while Figure 5b identifies the only 2 other paths able to improve the value of the second objective function. The final choice among the two different results strongly depends on the mission itself. For instance,in the scenario of an exploration mission with a short operational lifetime and targeted goals, the optimization of the path length plays a crucial role.

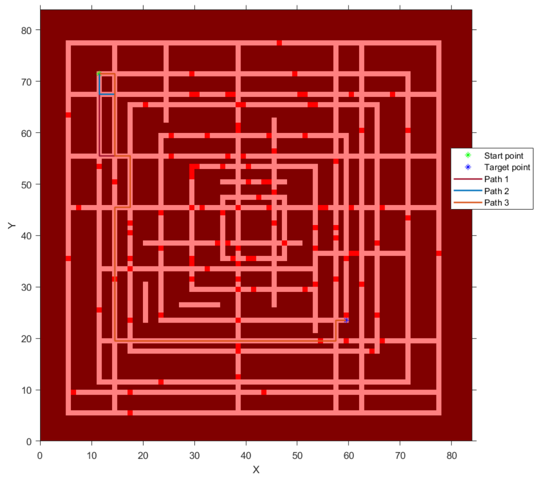
On the other hand for long and diversified missions, the need of preserving the integrity and functionalities of the rover as long as possible will be surely more demanding than the rapidity of execution of the planned tasks. The second objective function will weight indeed significantly in the conclusive decision-making process.



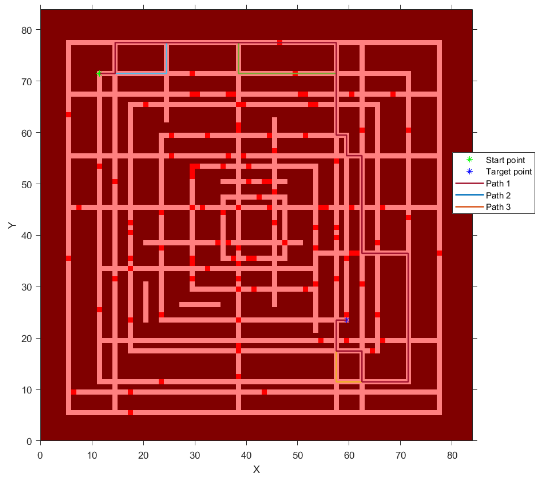
(a) Problem 5 - Objective 1



(b) Problem 5 - Objective 2



(a) Problem 5 - Objective 3



(b) Problem 5 - Objective 4

Fig. 7: Pareto optimal paths of problem 5

Problem 5's paths start instead at the point of coordinate $(12, 72)$ and finish in the point of coordinate $(60, 24)$. The occupancy map is here denser and more complex, resulting in a wider range of optimal objective values, that once again can be assigned to a multiple number of paths. Therefore the final decision is not trivial, but it would require a more accurate weighted evaluation of the importance of the objective functions for the specific mission. It is interesting to notice that in both the problems taken into account there does not exist a path that completely avoids the passage through the red areas. Further considerations might be required on the subject to better mitigate risk in the accomplishment of the rover tasks.

6 Conclusions

As demonstrated with the simulations, the proposed approach can provide exact Pareto solutions for the MMOPP problem in an efficient and simple way. Its applicability is obviously not limited to the investigated kind of problems where costs are only assigned to passable areas. There can also be corresponding costs for the movements between adjacent passable areas or additional features can be added such as the possibility of moving diagonally. In addition, the same algorithm can include other objective functions if needed or substitute the original ones, while still maintaining its efficacy, in order to make the problem formulation more complete and suitable for a real space mission: for example by minimizing the incident sunlight on the rover or by formulating the elevation objective function in a more precise way, modelling the terrain altitude distribution.

Leaving rovers aside, the algorithm lends itself perfectly for any other navigation purposes in robotics applications but also for path planning in urban transportation. It is just a matter of providing to the software reasonable and coherent input data and choosing the objective functions accordingly.

References

[1] https://www.hackerearth.com/practice/algorithms/graphs/depth-first-search/tutorial/.

[2] A Niched Pareto Genetic Algorithm for Multi-Objective Optimization .
https://www.researchgate.net/publication/3575291$_{AN}iched_{P}areto_{G}enetic_{A}lgorithm_{f}or_{M}ulti - Objective_{O}ptimization.$

[3] Comparison of NSGA-II and SPEA2 on the Multiobjective Environmental/Economic Dispatch Problem .
https://www.ajol.info/index.php/umrj/article/view/131124/120715.

[4] Pareto front .
https://en.wikipedia.org/wiki/Pareto$_{f}ront.$

[5] A Multi-objective Time-dependent Route Planner: A Real World Application to Milano City.
https://www.sciencedirect.com/science/article/pii/S2352146514001902.

[6] Curiosity Mars Rover: Traversing the Obstacles.
https://www.leonarddavid.com/curiosity-mars-rover-traversing-the-obstacles/.

[7] DFS Algorithm.
https://www.educba.com/dfs-algorithm/.

[8] Finding Articulation Points of a Graph.
https://www.baeldung.com/cs/graph-articulation-points.

[9] Hamiltonian Path.
https://mathworld.wolfram.com/HamiltonianPath.html.

[10] Operation Research course notes.

[11] Alexander Lavin . A Pareto Front-Based Multiobjective Path Planning Algorithm .
https://www.researchgate.net/publication/277145078$_{AP}areto_{F}ront - Based_{M}ultiobjective_{P}ath_{P}lanning_{A}lgorithm.$

[12] Jin Bo. Multi-objective A* algorithm for the multimodal multi-objective path planning optimization.
https://github.com/jinboszu/mmopp.

[13] Kalyanmoy Deb Lihui Wang, Amos H. C. Ng.
Multi-objective Evolutionary Optimisation for Product Design and Manufacturing. 2011.