Algorithmic Generation of Path Fragment Covers for Mobile Robot Path Planning

Jan Willemson, Maarja Kruusmaa

Abstract— **This paper describes a framework for mobile robot path planning in dynamic environments where the environments are represented by grid maps. The planning heuristics considered in the paper is to generate the minimal set of paths so that all the possible path fragments of two edges are covered. We show that the number of required paths is linear in the dimensions of the grid (thus making application of the approach realistic and scalable). The main contribution of the paper is the description of all the minimal covers by means of an efficient algorithm. We prove its correctness and conclude that there are $2^{(m-1)(n-2)+(m-2)(n-1)}$ minimal path fragment covers in an $m \times n$ grid.**

Keywords— **Grid path algorithms**

I. Introduction

Mobile robot applications often require repeated traversal in a changing environment between predefined start and goal points. For example, a mobile robot could be used to transport details and sub-assemblies between a store and production lines; this task implies repeated traversal between the store and the production cells. A mobile robot can also be used for surveillance; this task implies visiting certain checkpoints on a closed territory in a predefined order.

Real environments where these kind of mobile robots have to operate are dynamic by nature. From the point of view of mobile robot navigation, it means that unexpected obstacles can appear and disappear. The nature and density of the obstacles is usually unknown or too difficult to model.

At the same time, a mobile robot in a dynamic environment has to fulfill its assignment as fast and safely as possible. This means choosing paths between target points that are most likely unblocked and where the robot does not spend too much time maneuvering between obstacles.

A straightforward approach in robotic applications has been to choose the shortest paths to the goal. Various methods have been developed in a great number of studies that address this problem very essential to mobile robotics. Wavefront planners [1], A* planners and D* planners [2], planners on potential fields and harmonic functions [3] are all concerned with finding the shortest collision free path with the minimal effort. Another type of problems often tackled in mobile robotics is covering a territory. This problem statement is relevant in applications like demining or surveillance and is investigated in many studies (e.g. [4]) The problem of repeated traversal between predefined start and goal points has elements from both of the previously mentioned task descriptions. However, if the environment is dynamic and difficult to model, none of the approaches above is perfectly suitable. Modeling a dynamic environment would require fully covering the territory. But the real problem is to find best path to the goal using the model environment. However, in a dynamic and changing environment the best path to the goal is not necessarily the shortest that is planned on the possibly incomplete and incorrect map. We therefore suggest a more flexible approach that balances between exploring the environment and carrying out a task at the same time. We suggest an approach where the robot tries a number of possibly dissimilar paths while traversing between the predefined locations. Thereby it carries out the mission as well as explores the environment in the search for better paths at the same time. In comparison to shortest path following methods this method is more flexible permitting the search of the best paths over the whole environment. At the same time, in comparison with the area covering approaches it requires less time and effort and thus permits faster fulfillment of the robot's actual mission.

This paper represents the algorithmic description of the method. This method is experimentally verified in [5] and demonstrates that the robot is able to reduce collision risk on repeated traversal as well as to decrease travel distance, time and deviation from the pre-planned route.

Our basic approach for building such a strategy is described in [6]. The environment is represented by means of a rectangular grid (see Subsection III-A). The paths are planned on the grid and information about the best paths (e.g. in the sense of traversability) is stored in a case base memory. If the robot needs a new path, it may try to use its knowledge about previously best paths. If this attempt fails, a new path has to be generated. As the grid map is mostly more or less imprecise and the environment is changing, we do not try to compile paths from parts previously known to be good, but rather try to generate and evaluate the path as a whole.

In this paper we concentrate on finding a good path generation mechanism that generates paths, as dissimilar as possible, from a given start to a given goal thus permitting the robot to try as many alternative approaches as possible in its search for better paths. The rest of the paper is organized as follows. Section II presents current status of the research, Section III introduces the idea of path fragment covers, Section IV discusses the cardinality of such covers. Section V presents the main contribution of the paper – an algorithm for generating all minimal path fragment cov-

Tartu University, Institute of Coputer Science, Liivi 2, Tartu, Estonia. Email: jan@ut.ee; Cybernetica Ltd., Aleksandri 8a, Tartu, Estonia

Tartu University, Institute of Technology, Nooruse 1, Tartu, Estonia. Email: maarjakr@ut.ee

ers – and Section VI draws some conclusions together with directions for future work.

II. State of the art

In the context of mobile robotics, path planning means finding a collision-free path from start to goal (see e.g. [7] for an overview of intelligent robotics and path planning techniques).

Very few research studies reported so far consider the problem of path selection in dynamic environments [8], [9]. Unlike these approaches, we do not assume that the structure of the environment is known a priori. Therefore our approach is applicable also in cases where very little is known about the environment and where the structure of the environment may change.

In our own previous work, supported by extensive tests in simulated environments as well as on real robots in real environments, we have used a heuristic algorithm for generating new innovative paths [10]. The tests have revealed that the robot is able to adapt quickly to the changes in the environment. As a result, risk of collisions, time delays and traveled distances are minimized.

However, the tests also pointed out the shortcomings of our approach. We observed that many paths that the heuristic algorithm generated were similar to each other. It also appeared that sometimes the robot got trapped to local minima – some paths that would have been easy to follow were never generated.

The first attempt to overcome this problem is made in [6]. In that paper, first a Hausdorff-like [11] distance measure between the paths is defined. Then, a construction is given for a seed path set such that for every possible path from start to goal there exists a path in the seed set that is not more than of distance δ away from the original path. This corresponds to covering the whole path space with path balls of radius δ. This way we guarantee that when traversing all the seed set paths, a close match is traversed for every possible path.

It is proved in [6] that for a $m \times n$ grid the seed path set can not be made smaller than

$$\binom{\lfloor m/(2\delta+1)\rfloor + \lfloor n/(2\delta+1)\rfloor}{\lfloor m/(2\delta+1)\rfloor}.$$

The size of the seed set is reasonable for relatively large values of the similarity threshold δ, but when this threshold is decreased, the seed set cardinality increases very fast.

On the other hand, while being distant with respect to the distance measure, many paths in the constructed seed set overlap significantly. This consideration leads to another kind of interpretation for space covering introduced in the current paper. We will not try to generate a close match for every possible path (as implied by [6], this is hopeless anyway), but rather divide the whole area into small fragments and try to design such path sets that cover all the fragments. We will see that this approach leads to a much more scalable solution, namely the number of required paths is linear in grid dimensions.

A crucial question is how to choose the fragments to consider. The simplest solution would be to take all the grid points and select a path set to cover all of them. This idea does not work well for the following reason. If for example $m \geq n$, the minimal path set covering all the grid nodes contains $n+1$ paths organized as shown in Figure 1 for $m = 4$, $n = 3$. If an obstacle occurs on the left boundary of the grid, many paths will be disregarded as unsuitable and as a result, many grid points never get traversed.

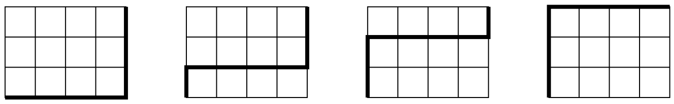

Fig. 1. Node cover for $m = 4$, $n = 3$

III. Covering path fragments

A. Grid and paths

Let $[a, b]$, $a \leq b \in \mathbb{Z}$ denote the set $\{a, a+1, \ldots, b\}$. We will consider a robot moving on a rectangular grid $[0, m] \times [0, n]$ allowing only right and up moves. We will also refer to such a grid as $m \times n$ grid, although it in fact contains $(m+1) \times (n+1)$ points.[1] The robot starts from the point $(0, 0)$ (which we consider to be the bottom left corner of the grid) and it must reach the point (m, n).

Definition 1: By a *path* on the grid $[0, m] \times [0, n]$ we mean a sequence of grid points $((x_0, y_0), (x_1, y_1), \ldots, (x_{m+n}, y_{m+n}))$ such that
1. $x_0 = y_0 = 0$, $x_{m+n} = m$, $y_{m+n} = n$;
2. for each $i \in [0, m+n-1]$, the condition

$$(x_i = x_{i+1} \,\&\, y_i + 1 = y_{i+1}) \vee (x_i + 1 = x_{i+1} \,\&\, y_i = y_{i+1})$$

holds.
The set of all paths on the grid is denoted by $\mathcal{P}_{m,n}$

B. Path fragment cover

In order to ensure higher probability for every grid point to be potentially traversed we need a more complicated condition than just the requirement to visit all the grid points. In this paper we use the requirement to cover all the *path fragments of length* 2 instead.

Definition 2: Path fragment of length 2 (or *path fragment* for short) is a sequence of grid points $((x_1, y_1), (x_2, y_2), (x_3, y_3))$ where one of the following holds:
- $x_1 = x_2 = x_3 \,\&\, y_1 + 1 = y_2 = y_3 - 1$;
- $x_1 + 1 = x_2 = x_3 - 1 \,\&\, y_1 = y_2 = y_3$;
- $x_1 + 1 = x_2 = x_3 \,\&\, y_1 = y_2 = y_3 - 1$;
- $x_1 = x_2 = x_3 - 1 \,\&\, y_1 + 1 = y_2 = y_3$.

The four possible path fragments are depicted in Figure 2 and later on they will be called path fragments of type one, two, three and four, respectively.

Definition 3: A subset $X \subseteq \mathcal{P}_{m,n}$ is called a *path fragment cover* if for every possible path fragment (c_1, c_2, c_3) there exists a path $P \in X$ such that $c_1, c_2, c_3 \in P$.

[1] This is mainly due to visual reasons as the graphical representations of the grids used throughout this paper consist of $m \times n$ squares.

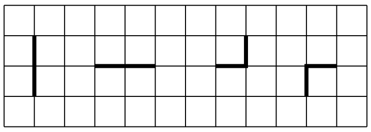

Fig. 2. Path fragments of length 2

Locally speaking, there are (at most) four different possibilities to traverse an inner grid point. Choosing path fragment cover as our seed path set guarantees that all these four possibilities are (at least theoretically) tried.

Now we can state the main problem of the current paper: Find the minimal size of a path fragment cover for a grid $[0,m] \times [0,n]$. Describe all the path fragment covers of minimal size and design the means for generating them.

IV. Minimal path fragment cover

Definition 4: Denote by $\tau(m,n)$ the minimal cardinality of path fragment cover of a $[0,m] \times [0,n]$ grid.

Theorem 1:

$$\tau(m,n) = \begin{cases} 1, & \text{if } m = 0 \vee n = 0; \\ m+n, & \text{if } (m = 1 \,\&\, n \geq 1) \vee \\ & (n = 1 \,\&\, m \geq 1); \\ 2m+2n-2, & \text{if } m,n \geq 2. \end{cases}$$

Proof: The first case when one of the dimensions of the grid is 0 is trivial. If one dimension (say, n) is 1 then we can see that the grid contains m path fragments of type four and we need m paths to cover these. Besides that, one fragment of type three still remains uncovered and one more path is required, giving $m+1 = m+n$ paths altogether. It is easy to see that all the other path fragments are also covered by these paths (see Figure 3).

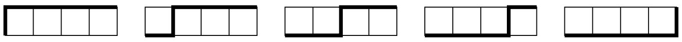

Fig. 3. Minimal cover for $m = 4$, $n = 1$

In case $m,n \geq 2$ we first consider the left boundary of the grid (without the point $(0,0)$). We say that a path fragment of length 2 *leaves the left boundary* if it has the form $((0,a),(1,a),(b,c))$, where $a \neq 0$. Now we compute the number of such fragments. It is clear that only the fragments of types two and three can leave the left boundary. For each one of the points $(0,1),(0,2),\ldots,(0,n-1)$ there exist fragments of both types that begin in these points (note that here we use the assumption $m \geq 2$). From the point $(0,n)$ only the fragment of type two can begin; altogether we get $2(n-1)+1 = 2n-1$ path fragments leaving the left boundary.

Symmetrically, we define a path fragment to *leave the bottom boundary* if it has the form $((a,0),(a,1),(b,c))$ (where $a \neq 0$). Similar to above, we find that there are $2m-1$ path fragments leaving the bottom boundary.

It is clear that every path can leave the left or bottom boundary of the grid only once and hence there must be at least $(2n-1)+(2m-1) = 2m+2n-2$ paths to cover all the path fragments. On the other hand, the construction showing that $2m+2n-2$ paths are sufficient is quite simple. All the fragments of type one can be covered with $m+1$ paths. For covering all the still uncovered fragments of type two $n-1$ paths are enough. Finally, the remaining fragments of types three and four can be covered using $m+n-2$ paths in a "diagonal" way. Altogether we have $(m+1)+(n-1)+(m+n-2) = 2m+2n-2$ paths.

For the case $m = 4$, $n = 3$ this construction is depicted in Figure 4. ■

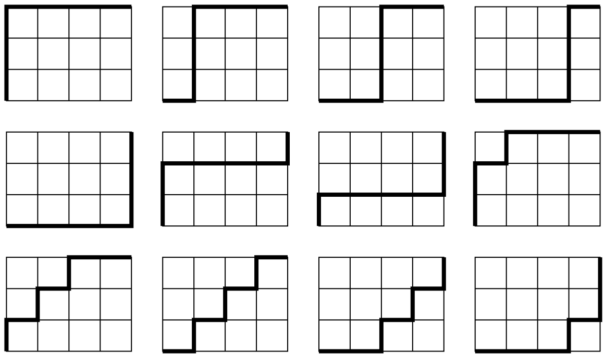

Fig. 4. Minimal cover for $m = 4$, $n = 3$

V. Generating the minimal covers

The last construction given in the proof of Theorem 1 and shown in Figure 4 will later be referred to as *trivial cover*. It is easy to see that generally (i.e. when $m \geq 2$, $n \geq 2$ and $m+n \geq 5$) this cover is not unique. Since this is exactly the interesting non-trivial case, we will from now on assume that $m \geq 2$ and $n \geq 2$.

For example, if we select two paths from the cover that have a common edge, we can replace the selected paths by "crossed" paths that take the beginning from one and ending (onwards from the common edge) from another original path. Such an operation obviously generates a new minimal cover and it is easy to see that if the common edge does not have an endpoint on the edge of the grid, the resulting cover differs from the original one.

Intuitively we can also say that the trivial cover is not a very good one. Although we have all the fragments of length 2 covered, we still have not achieved our original goal of trying as different paths as possible; for example two neighboring "diagonal" paths are rather close. In order to compare different minimal path fragment covers suitable measures must be introduced (e.g. based on the Hausdorff-like distance defined in [6]) and analyzed.

The analysis itself remains the subject for future research, but the rest of the current paper discusses the problem of generating all the minimal path fragment covers needed for the analysis.

We will show that a minimal path fragment cover can be fully described by its behavior on the following subset of grid edges:

$$\mathcal{S}_{m,n} = \{((i,j),(i,j+1)) : 0 < i < m,\ 0 < j < n-1\} \cup \\ \cup \{((i,j),(i+1,j)) : 0 < i < m-1,\ 0 < j < n\}.$$

Stating it otherwise, $\mathcal{S}_{m,n}$ consists of the edges of the grid that have no endpoint on the grid's boundary. Clearly, $|\mathcal{S}_{m,n}| = (m-1)(n-2) + (m-2)(n-1)$.

Next we "color" all the elements of $\mathcal{S}_{m,n}$ with one of two bits, i.e. 0 or 1. All the possible "colorings" can be considered as the elements of the function set

$$F_{m,n} = \{f : \mathcal{S}_{m,n} \to \{0,1\}\}.$$

Now we take a "coloring" function $f \in F_{m,n}$ and construct a minimal path fragment cover based on it. Note that in every such cover, there are at least four paths that contain no edges from the set $\mathcal{S}_{m,n}$ (e.g. in Figure 4 they are the paths number 1, 5, 8 and 12, respectively). If $m = 2$ or $n = 2$, there can be even more such paths (see the example given in Figure 5). It follows from the proof of Theorem 1 that all the $2m + 2n - 2$ paths are identified by the first path fragment of length 2 that leaves the (left or bottom) boundary of the grid. Order these fragments in some order $r_1, r_2, \ldots, r_{2m+2n-2}$.

In order to produce a suitable cover, we apply Algorithm 1.

Algorithm 1 Producing a minimal path fragment cover

Require: Function $f \in F_{m,n}$, initial path fragments $r_1, r_2, \ldots, r_{2m+2n-2}$

Ensure: Minimal path fragment cover

- **for** $i = 1$ to $2m + 2n - 2$ **do**
  - Take the fragment r_i and include all the edges from the grid's bottom left corner to the fragment r_i (the edges of fragment r_i included) into the new path.
  - **while** the last edge of the current path part belongs to $\mathcal{S}_{m,n}$ **do**
    - Let $e = ((x_1, y_1), (x_2, y_2))$ be the last edge of the current path
    - **if** ($f(e) = 0$ and the edge e has not been traversed before) **or** ($f(e) = 1$ and the edge e has been traversed before) **then**
      - Let the next edge on the path be $((x_2, y_2), (x_2, y_2 + 1))$
    - **else**
      - Let the next edge on the path be $((x_2, y_2), (x_2 + 1, y_2))$
    - **end if**
  - **end while**{Completing this loop we have reached the boundary of the grid}
  - Complete the path to the top right corner of the grid
  - Add the completed path to the cover
- **end for**

We illustrate the work of Algorithm 1 for the case $m = 4$, $n = 2$. Out of $2 \cdot 4 + 2 \cdot 2 - 2 = 10$ paths in every minimal cover there are 7 that belong to every one of these covers. These are exactly the paths that have no edge in $\mathcal{S}_{4,2}$: 4 trivial paths and 3 arising from fragments r_i of type one (see Figure 5).

Other $2^{(4-1)(2-2)+(4-2)(2-1)} = 4$ variations of the remaining three paths are shown in Figure 6.

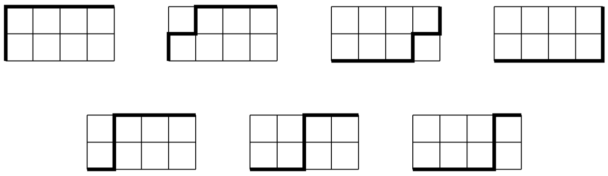

Fig. 5. Common paths in minimal covers for $m = 4$, $n = 2$

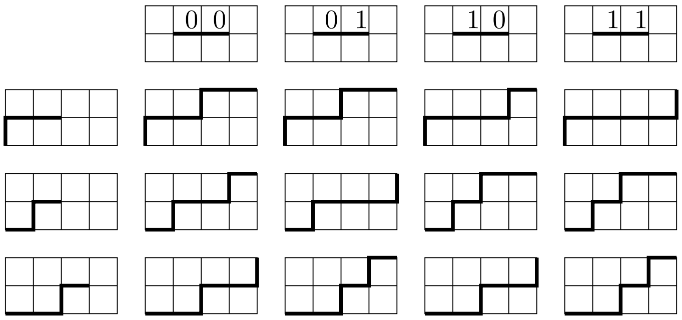

Fig. 6. Varying paths in minimal covers for $m = 4$, $n = 2$

We now prove correctness of Algorithm 1.

Theorem 2: Algorithm 1 produces a minimal path fragment cover.

Proof: Since Algorithm 1 clearly produces us $2m + 2n - 2$ paths, it remains to show that all the path fragments of length 2 will be covered. This is immediate for the fragments $r_1, r_2, \ldots, r_{2m+2n-2}$ leaving the left or bottom boundary, but also for all the fragments that have their first fragment on either of these boundaries.

Next we argue that every edge not belonging to the boundary is traversed exactly twice during the run of Algorithm 1. This is clear for the set of edges

$$\begin{aligned}\mathcal{R}_{m,n} = \{((i,0),(i,1)) : 0 < i < m\} \cup \\ \cup \{((0,j),(1,j)) : 0 < j < n\},\end{aligned}$$

i.e. for the edges that have exactly one end on the bottom or left boundary (but not in the corner). Let us also denote

$$\begin{aligned}\mathcal{T}_{m,n} = \{((i,n-1),(i,n)) : 0 < i < m\} \cup \\ \cup \{((m-1,j),(m,j)) : 0 < j < n\},\end{aligned}$$

i.e. the set of edges that have exactly one end on the upper or right boundary (but not in the corner).

For the edges $e \in \mathcal{S}_{m,n} \cup \mathcal{T}_{m,n}$ we will use induction on $x_1 + y_1$ (where $e = ((x_1, y_1), (x_2, y_2))$).

The induction step can be proven by considering different possible cases. First, let the edges potentially preceding e on some paths be from the set $\mathcal{R}_{m,n}$, i.e.

$$\begin{aligned}e_1 = ((x_1 - 1, y_1), (x_1, y_1)) \in \mathcal{R}_{m,n} \text{ and} \\ e_2 = ((x_1, y_1 - 1), (x_1, y_1)) \in \mathcal{R}_{m,n}.\end{aligned}$$

This can happen only if $x_1 = y_1 = 1$, i.e. for $e = ((1,1),(1,2))$ or $e = ((1,1),(2,1))$. Consider for example $e = ((1,1),(1,2))$ (see Figure 7). The edge e is traversed

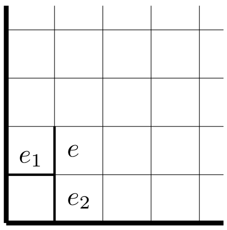

Fig. 7. The first case of the proof of Theorem 2

by the fragment of type three originating from $(0,1)$ and by the fragment of type one originating from $(1,0)$. The case $e = ((1,1),(2,1))$ is similar.

Second, let the edges potentially preceding e on some paths be from the sets $\mathcal{R}_{m,n}$ and $\mathcal{S}_{m,n}$, e.g.

$$e_1 = ((x_1-1,y_1),(x_1,y_1)) \in \mathcal{R}_{m,n} \text{ and}$$
$$e_2 = ((x_1,y_1-1),(x_1,y_1)) \in \mathcal{S}_{m,n}.$$

By the observation above and induction hypothesis, they both get traversed (exactly) two times. Consider further the case $(x_2,y_2) = (x_1+1,y_1)$ and $f(e_2) = 0$ (see Figure 8). It is now clear that the edge e is traversed twice – when

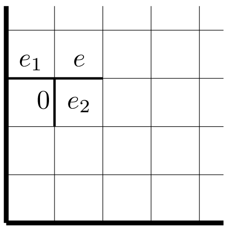

Fig. 8. The second case of the proof of Theorem 2

traversing the edge e_1 with the path fragment of type two, and after the edge e_2 is traversed for the second time. There are other cases of this type, but they all are handled in a similar way.

Next, consider the case when

$$e_1 = ((x_1-1,y_1),(x_1,y_1)) \in \mathcal{S}_{m,n} \text{ and}$$
$$e_2 = ((x_1,y_1-1),(x_1,y_1)) \in \mathcal{S}_{m,n},$$

$(x_2,y_2) = (x_1+1,y_1)$, $f(e_1) = 0$ and $f(e_2) = 1$ (see Figure 9). By induction hypothesis, the edges e_1 and e_2 are

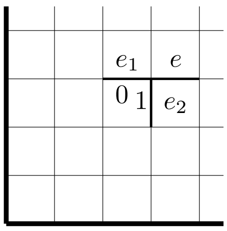

Fig. 9. The third case of the proof of Theorem 2

traversed exactly twice. We see that the edge e is traversed after the second traversal of edge e_1 and after the first traversal of edge e_2. Again, there are other cases of this type, but they all are handled in a similar way. This completes the induction step.

Following the arguments above we see that for each edge $e \in \mathcal{R}_{m,n} \cup \mathcal{S}_{m,n} \cup \mathcal{T}_{m,n}$, both the fragments having e as the second edge get traversed. It is also clear that the fragments having their first edges in $\mathcal{T}_{m,n}$ get traversed, since we proved above that all the edges in this set are traversed (even twice).

The only path fragments we have not yet covered are the ones that belong to the boundary of the grid, but they are covered by the two paths that never leave this boundary. These paths are generated by Algorithm 1 due to fragments $((0,n),(1,n),(2,n))$ and $((m,0),(m,1),(m,2))$. This concludes the proof. ■

In order to show that we actually achieve every possible cover exactly once, we give a construction of a suitable function f based on a minimal path fragment cover, which can be easily seen to induce the inverse mapping to the one of Algorithm 1. In order to construct such a function, we will need the following fact.

Proposition 1: Every edge $e \in \mathcal{S}_{m,n}$ is traversed by exactly two paths in a minimal path fragment cover.

Proof: The definition of a path fragment cover readily implies that every $e \in \mathcal{S}_{m,n}$ is covered by at least two paths from the cover (since each such e belongs to two fragments as the first edge). Suppose for some $e \in \mathcal{S}_{m,n}$, there are three paths traversing it and let $e = ((x_1,y_1),(x_2,y_2))$ be such that x_1+y_1 is the smallest possible. Then one of the edges $e_1 = ((x_1-1,y_1),(x_1,y_1))$ and $e_2 = ((x_1,y_1-1),(x_1,y_1))$ must also be traversed three times, which due to minimality of x_1+y_1 implies that this edge (say, e_1) does not belong to $\mathcal{S}_{m,n}$. This is only possible if $e_1 \in \mathcal{R}_{m,n}$. But in this case it is easy to see that traversing some of the edges in $\mathcal{R}_{m,n}$ results in a number of paths larger than $2m+2n-2$ contradicting minimality of the cover. ■

Let us have a minimal path fragment cover. Its paths can be ordered by the first boundary-leaving fragments $r_1, r_2, \dots, r_{2m+2n-2}$ as done above. Let $P_1, P_2, \dots, P_{2m+2n-2}$ be the corresponding ordering of the paths. We define the function $f' \in F_{m,n}$ as follows.

$$f'(e) = \begin{cases} 0, & \text{if for the path } P_i \text{ (with smaller possible} \\ & \text{index } i\text{) that traverses the edge } e, \\ & \text{the next edge after } e \text{ is upward,} \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

Due to Proposition 1, this definition is sound. It remains to understand why f' restores the original function f.

Proposition 2: Let $f \in F_{m,n}$ and let us have a minimal path fragment cover generated by Algorithm 1 using the function f. Then for the function f' defined by (1) we have $f = f'$.

Proof: The claim is actually straightforward. Having $f(e) = 0$ in Algorithm 1 means "If you are on this edge for the first time, then turn upward, if for the second time, then turn right" and vice versa for $f(e) = 1$. Since we consider the paths in the same order as produced by Algorithm 1, we restore the function f fully by following (1). ∎

Now we are ready to count all the possible minimal path fragment covers.

Proposition 3: There are $2^{(m-1)(n-2)+(m-2)(n-1)}$ minimal path fragment covers in an $m \times n$ grid.

Proof: We showed above that minimal path fragment covers are in one-to-one correspondence with the elements of the set $F_{m,n}$. Hence their number is equal to

$$|F_{m,n}| = 2^{|\mathcal{S}_{m,n}|} = 2^{(m-1)(n-2)+(m-2)(n-1)}.$$

∎

Time complexity of Algorithm 1 can be computed as follows. We need to generate $2m + 2n - 2$ paths, each path generation takes $m + n$ steps (including both steps on the grid boundary and the while-loop on lines 3–10). Hence, the overall time complexity is $O((m+n)^2)$. Note that practically the same time is needed just for the result printout and thus there can be no substantially faster algorithm for this particular task.

Space complexity of Algorithm 1 is low as well. We need $(m-1)(n-2) + (m-2)(n-1)$ bits to store the function f and additional $(m-1)(n-2) + (m-2)(n-1)$ bits to store information about the traversedness of edges. Hence, the overall space complexity is $O(mn)$.

VI. Conclusions and further work

In this paper we have presented a framework for mobile robot path generation in unknown and dynamically changing environment. Our approach was to represent the area as a grid map and design a special set of paths (called path fragment cover). Following the paths from this set, the robot traverses all grid points in all possible (up to four) ways.

We showed that the minimal number of paths required to cover $m \times n$ grid is in general case equal to $2m + 2n - 2$ and hence, such a solution is highly scalable. Still, minimal path fragment cover is not unique and the number of suitable covers grows exponentially. We described all of them by means of a very efficient algorithm and proved that the algorithm is correct.

Intuitively it seems that not all the possible minimal covers are not equally good. In order to determine the best one from them, it is necessary to define some quantitative measure on covers and try to optimize that. This measure should take into account the relative positions of all pairs of paths. For example, one could define the measure of minimal cover $\mathcal{C}_{m,n}$ as $\min_{P_1,P_2 \in \mathcal{C}_{m,n}} d_g(P_1, P_2)$, where d_g is an appropriate distance measure between the paths (see e.g. [6]), and maximize this quantity. At the moment, it is not clear at all how such a quantity behaves and where does it achieve its maximum, so this remains the subject for future research.

VII. Acknowledgments

This research was supported by Estonian Science Foundation grant no. 6096.

References

[1] A. Zelinsky, "Using path transforms to guide the search for findpath in 2d," *International Journal of Robotics Research*, vol. 13, no. 4, pp. 315–325, 1994.

[2] D. Ferguson and A. Stentz, "The delayed D* algorithm for efficient path replanning," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA 2005)*, April 2005.

[3] C.I. Connolly and R.A. Grupen, "The application of harmonic functions to robotics," *Journal of Robotic Systems*, vol. 10, no. 7, pp. 931–946, 1992.

[4] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition," *Autonomous Robots*, vol. 9, pp. 247–253, 2000.

[5] Kristo Heero, Alvo Aabloo, and Maarja Kruusmaa, "Mobile robots in dynamic partially unknown environments," *International Journal of Advanced Robotic Systems*, vol. 2, no. 3, pp. 209–222, 2005.

[6] Maarja Kruusmaa and Jan Willemson, "Covering the Path Space: A Casebase Analysis for Mobile Robot Path Planing," *Knowledge Based Systems*, vol. 16, no. 5–6, pp. 235–242, July 2003.

[7] Robin R. Murphy, *An Introduction to AI Robotics (Intelligent Robotics and Autonomous Agents)*, MIT Press, 1st edition, 13 Nov. 2000.

[8] H. Hu and M. Brady, "Dynamic Global Path Planning with Uncertainty for Mobile Robots in Manufacturing," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 5, pp. 760–767, Oct. 1997.

[9] Karen Zita Haigh and Maria Manuela Veloso, "Planning, execution and learning in a robotic agent," in *AIPS-98*, June 1998, pp. 120 – 127.

[10] Maarja Kruusmaa, *Repeated Path Planning for Mobile Robots in Dynamic Environments*, Ph.D. thesis, Chalmes Univeristy of Technology, Gothenburg, Sweden, 2002.

[11] D. Huttenlocher, D. Klanderman, and A. Rucklige, "Comparing images using the Hausdorff distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 850–863, September 1993.