Covering the Path Space: A Casebase Analysis for Mobile Robot Path Planning

Maarja Kruusmaa
Tallinn Technical University
Dept. of Mechatronics
Ehitajate tee 5, Tallinn, Estonia

Jan Willemson
Tartu University
Dept. of Computer Science
Liivi 2, Tartu, Estonia

Abstract

This paper presents a theoretical analysis of a casebase used for mobile robot path planning in dynamic environments. Unlike other case-based path planning approaches, we use a grid map to represent the environment that permits the robot to operate in unstructured environments. The objective of the mobile robot is to learn to choose paths that are less risky to follow. Our experiments with real robots have shown the efficiency of our concept. In this paper, we replace a heuristic path planning algorithm of the mobile robot with a seed casebase and prove the upper and lower bounds for the cardinality of the casebase. The proofs indicate that it is realistic to seed the casebase with some solutions to a path-finding problem so that no possible solution differs too much from some path in the casebase. This guarantees that the robot would theoretically find all paths from start to goal. The proof of the upper bound of the casebase cardinality shows that the casebase would in a long run grow too large and all possible solutions cannot be stored. In order to keep only the most efficient solutions the casebase has to be revised at run-time or some other measure of path difference has to be considered.

Keywords. Case-based reasoning, path planning, covering in metric spaces

1 Introduction

The work presented in this paper is a theoretical extension of a research in mobile robotics. We investigate the problem of generating a seed casebase to cover the solution space of a mobile robot. In our earlier work we have implemented a case-based reasoning approach to mobile robot path planning and tested it on real robots [1]. In this work we prove the theoretical upper and lower bound of the casebase to show the feasibility and limitations of our approach.

The rest of this paper is organised as follows. In the next subsection we give an insight to the field of robot navigation and explain the relevance of the problem. Section 2 describes our approach and reviews our previous experimental results. In Section 3 we give the basic definitions and state the main robot path space covering problem in Section 4. Sections 5 and 6 prove (exact) lower and upper bounds, respectively, for the stated covering problem. Section 7 compares the old heuristic approach to the new one presented in the current paper. Section 8 draws some conclusions and discusses the future work.

Motivation

Our work is motivated by the fact that most of mobile robot applications imply 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.

Very few research studies reported so far consider this problem of path selection [2, 3]. 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.

2 System Description

Our approach to mobile robot path selection consists of a general world model and of a memory that stores the path traveling experiences for later use. The memory is a casebase. The casebase stores the paths traversed in the past in a form of cases.

The world model is a map that permits path planning. Since in a dynamic environment the robot is not able to model all the aspects of its surrounding, the map is always more or less imprecise.

Figure 1 captures the bottom line of this approach. The global planner receives tasks from the user. The tasks are requests to move to a specific point from its present location. The global planner has a map of the environment that represents only the very general geometry of the environment and the locations of the target points. The presence and location of dynamic obstacles in the environment are unknown. Given a new task, the global planner can either find a new solution by using a map-based path planner or re-use one of the earlier found paths from the casebase. The path planned by the global planner is presented to the low-level planning and execution unit that is responsible for task decomposition (if necessary), replanning, localisation, sensor data processing and actuator control.

The objective of the global planner is to choose the best travel path according to some criterion (e.g. time, distance, safety). Case-based reasoning permits the robot to remember and learn from its past experiences. The robot will adapt to the changes in the dynamic environment and learn to use paths that are better.

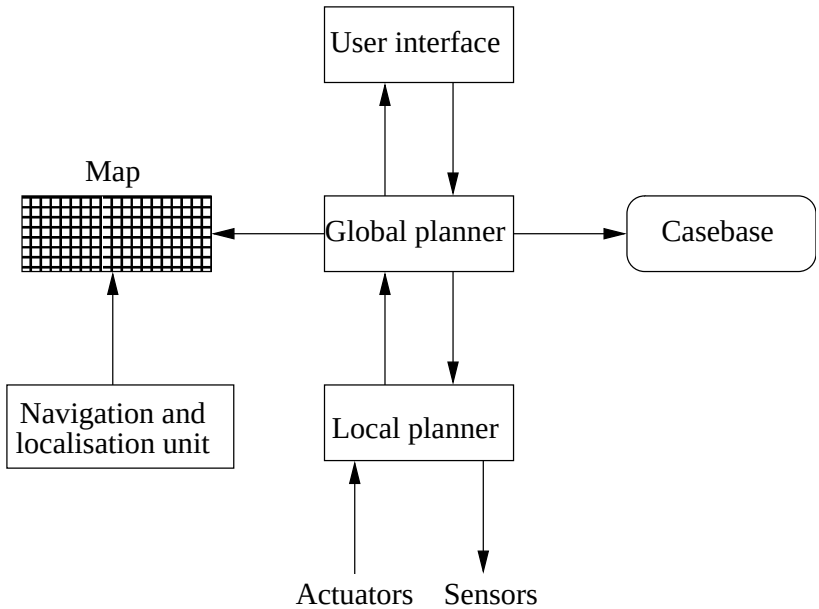

Figure 1: General overview of path planning

2.1 Case-based Reasoning

Case-based reasoning (CBR) solves new problems by adapting previously successful solutions to similar problems. The past experiences are stored in a casebase which is managed by applying database techniques. To facilitate the case retrieval the cases in a casebase are indexed. When a new problem occurs, the indices are extracted from its features and used to find matching cases in a casebase. If more than one matching case is found the candidate cases are evaluated to find the most suitable one.

Unless the retrieved case is a close match, the solution will probably have to be modified before using it for problem solving. If the modified case is found to be successful, it produces a new case which is stored in a casebase. Thus, in case-based reasoning, learning is through accumulation of new cases.

In the approach described in this paper, the problem is to find the best path from a given starting point to a given goal. The solution to the problem is a path to that goal. The outcome of the solution is the cost of the path that reflects how easy it was to follow that path. If the robot traverses a path repeatedly, the cost of the path is updated after every traversal. The cost will then reflect the average characteristics of that path. By choosing the path with the lowest cost, the robot can reduce collision risk and minimise travel distance.

Path planning by means of CBR is described in [4, 5, 6, 7]. These approaches are used for planning in static environments. PRODIGY/ANALOGY uses CBR for dynamic path planning in the city of Pittsburgh [8]. Unlike these studies our study

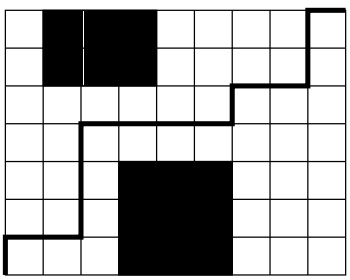

Figure 2: A grid map

does not use a graph-based map for path planning but a grid map that permits more detailed path planning, offers more alternatives to path-find problems and does not presume a rigid structure of the environment.

Since there can be several alternative paths between given target points, every problem in the casebase can have several solutions. To find the most similar problem from the casebase we use the nearest neighbor retrieval. i.e. the robot will look for a case where the start and goal points are as close as possible to the current problem.

However, to analyse our casebase, we assume that our casebase consists of only one problem having many solutions. This will be the problem having the greatest number of possible solutions, namely, the problem of traversing between the diagonally opposite corners. It follows from our problem formalisation that all other path planning problems are subproblems of this most general one. Our objective is to seed the casebase with all different solutions to the current problem according to our similarity measure.

2.2 Path Planning

In the context of mobile robotics, path planning means finding a collision-free path from start to goal. The method of path planning depends on the world model of the robot (see e.g. [9] for an overview of intelligent robotics and path planning techniques).

In this work we use a common world representation in mobile robotics – a grid map. A grid map represents the world as a grid of small cells. Some cells that are known to contain static obstacles can be marked as occupied. In our previous work we have developed a modification of a heuristic algorithm that due to random perturbations on the map generates many alternative solutions to a single find-path problem.

Figure 2 shows a grid map. Black cells on the map represent static obstacles. The paths form start to goal are generated with our heuristic algorithm.

There are two problems associated with our method of path generation. First, it cannot be guaranteed that it would theoretically find all the possible paths from the given start to the given goal.

Second, even on a relatively small grid the amount of different paths between

specified target points is overwhelming. At the same time most of the paths differ form each other only by a very small amount. To overcome this problem, we have defined a similarity metrics (see Section 3). With the help of similarity metrics we can treat paths that differ very little to be the same.

The paths that the robot has followed are stored in the casebase. We here take an advantage of our similarity metrics and store only cases (e.g. paths) that are different from each other. This remarkably reduces the size of the casebase.

In our experiments we have started with an empty casebase. If a necessary solution is not in the casebase or its quality is not good enough, a new solution can be generated using map-based path planning. We have tested our approach intensively both in simulated environments and with real robots. The tests show that the robot is very quickly able to adapt to the changes in the environment and learn to use less risky paths. All the tests indicate that the size of the casebase is very small even if the environment is large.

However, the tests also point to the shortcomings of our approach. Our path planning algorithm modifies the map to generate new random solutions. During the tests it was observed that many paths that the algorithm generates are very similar to each other. The robot spent much time waiting for a different solution to be found. It also appeared that sometimes the robot got trapped to the local minima – some paths that would have been easy to follow where never generated.

2.3 Seeding the Casebase

To overcome these disadvantages of our current implementation we investigate the possibility to seed the robot's casebase with all possible solutions to the current problem. Theoretically, the amount of possible paths from a given start to a given goal is enormous. Many of them differ from each other only by a very small amount and most of them are infeasible from the point of view of path following (i.e. unnecessarily crooked or long). We therefore have constrained the set of paths that are used in the casebase.

To justify our constraints we have to explain some details of robot path following. In a real life, a robot is never able to follow precisely the path that it has planned. It will always drift away form its course because of localisation errors, sensor noise and imprecision of mechanical linkages. In dynamic environments, it also has to drive around unexpected obstacles. The latter can significantly deviate the robot from its course. Therefore our similarity metrics is based on the deviation of paths. We consider paths that deviate much from each other to be different. On the contrary, paths that lead through the approximately same regions of the map are similar. Our similarity metrics is defined in the next section.

Mobile robots use obstacle avoidance routines and run-time re-planning to negotiate dynamic obstacles. In our experiments with real robots we have observed that the planned path and the eventually followed path often differ significantly because replanning around obstacles and localisation errors sometimes lead the robot in a completely other direction than it was heading first. We therefore found that the most practical measure of the traversability of a path is the similarity between the planned and the actually followed path. The better the robot was able to follow the path that it

planned the better the path is. The secondary measure that we have used to evaluate paths is the time of path following. This measure is also very practical because mobile robots are normally expected to fulfill their assignment as fast as possible.

It therefore seems to be feasible to seed the casebase with paths that are rather short and straight. These paths are most likely to satisfy our criteria of good traversability. Short paths are most likely to lead the robot faster to the goal. Too curved paths are technically too hard to follow (and therefore it also takes longer time to reach the goal). We have therefore constrained our set of paths to those allowing only right and up moves. It will exclude all the unnecessarily long and complex paths (actually all paths having back turns).

The drawback of this condition is that the robot would not operate efficiently in a maze-like environment. However, most of real environments are not mazes. Another shortcoming, though not so severe, is the occurrence of the zigzagged paths since we allow only up and right moves. This is typical to all grid-based path planning methods and mobile robots usually use path relaxation techniques to smoothen the path at run-time.

In the rest of the paper we address two problems of casebase management. The first question is whether it is possible to seed the casebase with a relatively low number of different cases so that the whole solution space will be covered. The answer to this question is affirmative.

The second question is whether it is possible to run the system without managing the casebase. For instance, the robot may generate the potential solutions randomly with the only condition that the paths in the casebase should not be too similar. The answer to this question is negative. The maximal number of different solutions may become too high, the casebase grows too large and the robot looses its ability to manage the whole base. Therefore, less useful solutions have to be forgotten. In the next sections we will formalise the problem, define a similarity metrics and prove lower and upper bounds for the cardinality of the casebase.

3 Basic Definitions

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. The robot starts from the point $(0, 0)$ (which we consider to be the lower 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 $\mathcal{P}_{m,n}$

Next we define a notion of similarity between the paths. Intuitively speaking, we say that two paths taken by a robot are similar if they do not diverge from each other too much. In order to express this idea, we need to define an appropriate distance measure first. There are several possibilities for such a definition, in this paper we use the following one.

Definition 2 *We say that the* grid distance *between a point $c_1 \in [0,m] \times [0,n]$ and a path $P_2 \in \mathcal{P}_{m,n}$ is the quantity*

$$d_g(c_1, P_2) = \min_{c_2 \in P_2} \{d(c_1, c_2)\}$$

where $d(c_1, c_2)$ denotes the $\mathbb{R}^2_\infty$-distance between the points c_1 and c_2, i.e. $d(c_1, c_2) = \max\{|x_1 - x_2|, |y_1 - y_2|\}$, where $c_1 = (x_1, y_1)$ and $c_2 = (x_2, y_2)$.

By grid distance *of the paths $P_1, P_2 \in \mathcal{P}_{m,n}$ we mean the quantity*

$$d_g(P_1, P_2) = \max_{c_1 \in P_1} \{d_g(c_1, P_2)\}.$$

The maxmin construction of d_g is generally known as *directed Hausdorff distance* (see e.g. [10]). It is not the case that for any inner metrics d the directed Hausdorff distance is a real distance since it mostly fails to be symmetric; such a problem occurs, for instance, if d is the Euclidean distance. Several approaches can be taken in order to fix the problem, most commonly one replaces the directed distance by $\max\{d_g(P_1, P_2), d_g(P_2, P_1)\}$ [10] or even $d_g(P_1, P_2) + d_g(P_2, P_1)$ [11].

In this paper, however, we show that for our very special choice of base set and inner metrics, the directed Hausdorff distance gives rise to a real distance.[1]

Lemma 1 *The pair $(\mathcal{P}_{m,n}, d_g)$ is a metric space.*

Proof. First we note that $d_g(P_1, P_2)$ is always a non-negative integer. The identity and triangle inequality axioms of metric space are easy to check (in fact, they hold for every directed Hausdorff distance).

In order to prove symmetry, we first prove that the following assertion holds true for every two paths $P_1, P_2 \in \mathcal{P}_{m,n}$:

$$\forall c_1 \in P_1 \, \exists c_2 \in P_2 \; [d_g(c_1, P_2) = d_g(c_2, P_1)]. \tag{1}$$

Take a point $c_1 \in P_1$. Note first that if $c_1 \in P_2$, then we have $d_g(c_1, P_2) = 0$ and we can take $c_2 = c_1$ to satisfy (1).

If $c_1 \notin P_2$, then assume w.l.o.g. that the path P_2 runs above the the point c_1. Take the closed ball B with center c_1 and radius $d_g(c_1, P_2)$ in metric space $([0,m] \times [0,n], d)$ where d is the $\mathbb{R}^2_\infty$-distance as in Definition 2. (Note that this

[1] One may argue that the Euclidean $\mathbb{R}^2$ metric is better suited for practical purposes than the $\mathbb{R}^2_\infty$ metric. On the other hand, we feel that working on the (in fact discrete) grid, the $\mathbb{R}^2_\infty$ metric is actually more natural. Besides that, the norms corresponding to the two distance measures are equivalent and do not differ more than $\sqrt{2}$ times.

ball actually looks like a square with edge length $2d_g(c_1, P_2)$ and center c_1.) We see from Definition 2 that no interior point of B belongs to the path P_2, but some of the boundary points do. In particular, the upper left corner of B must always belong to P_2. Now taking this point as c_2 we see that no point of P_1 can be closer to c_2 than c_1 is, hence $d_g(c_1, P_2) = d_g(c_2, P_1)$ as was required to prove (1).

Now let $\bar{c}_1 \in P_1$ be such a point that $d_g(\bar{c}_1, P_2) = \max\limits_{c_1 \in P_1}\{d_g(c_1, P_2)\}$. Then by (1) we can choose a point $\bar{c}_2 \in P_2$ such that $d_g(\bar{c}_1, P_2) = d_g(\bar{c}_2, P_1)$. Thus

$$d_g(P_1, P_2) = d_g(\bar{c}_1, P_2) = d_g(\bar{c}_2, P_1) \leq \max_{c_2 \in P_2}\{d_g(c_2, P_1)\} = d_g(P_2, P_1).$$

Similarly we prove that $d_g(P_2, P_1) \leq d_g(P_1, P_2)$ and hence $d_g(P_1, P_2) = d_g(P_2, P_1)$. □

In the light of Lemma 1, the next definition is just a utilisation of a standard definition of a (closed) ball in metric space.

Definition 3 *By a* ball *with center P and radius δ in the space $(\mathcal{P}_{m,n}, d_g)$ we mean the set*

$$B(P, \delta) = \{P' \in \mathcal{P}_{m,n} :\ d_g(P, P') \leq \delta\}.$$

Denoting $\pi\,(m, n) := |\mathcal{P}_{m,n}|$, we have the following standard result with the standard one-line proof.

Lemma 2 $\pi\,(m, n) = \binom{m+n}{m}$.

Proof. Each path contains $m + n$ steps, out of which m are made rightwards. □

4 Problem Statement

We will be looking at the following covering problem in the metric space $(\mathcal{P}_{m,n}, d_g)$.

Problem. For a given integer δ and grid dimensions m and n, find a lower and upper estimate to the cardinality of a subset $S \subseteq \mathcal{P}_{m,n}$ such that the following conditions hold.

$$\bigcup_{P \in S} B(P, \delta) = \mathcal{P}_{m,n} \tag{2}$$

$$\forall P' \in S \left[P' \notin \bigcup_{P \in S \setminus \{P'\}} B(P, \delta) \right] \tag{3}$$

The lower estimate corresponds to the question about efficient covering. In this setting we ask what is the minimal number of pre-planned paths required in the casebase in order embrace all the possible paths with deviation not exceeding the threshold δ.

The upper estimate, on the contrary, deals with the worst case. In this case we consider a process of random path generation and ask what is the largest path set that can occur if we every time only include new paths that deviate more than by δ from all the previously recorded paths.

5 Lower Bound

Theorem 1 *For every $\delta \in \mathbb{N}$ and every subset $S \subseteq \mathcal{P}_{m,n}$ satisfying the properties (2) and (3), the inequality*

$$|S| \geq \pi\left(\left\lfloor \frac{m}{2\delta+1} \right\rfloor, \left\lfloor \frac{n}{2\delta+1} \right\rfloor\right) \tag{4}$$

holds. Evenmore, there exists such a set S that the properties (2) and (3) are satisfied and equality holds in inequality (4).

Proof. First we consider a special case when $m, n \vdots 2\delta+1$ and a subset $T \subseteq \mathcal{P}_{m,n}$ defined by the following condition:

$$T = \{((x_0, y_0), \ldots, (x_{m+n}, y_{m+n})) \in \mathcal{P}_{m,n} : \forall i \in [0, m+n]\ x_i \vdots 2\delta+1 \vee y_i \vdots 2\delta+1\}.$$

We note that for $P_1 \neq P_2 \in T$ the inequality $d_g(P_1, P_2) \geq 2\delta+1$ holds. Hence, no two different elements of the set T can be contained in the same ball of radius δ. Consequently, when covering the space $\mathcal{P}_{m,n}$ with balls of radius δ and with centers in the elements of S (as required by condition (2)), there must be at least the same number of balls as there are elements in the set T. But the paths of T are essentially grid paths on a grid with dimensions $\frac{m}{2\delta+1} \times \frac{n}{2\delta+1}$ (and grid squares of dimensions $(2\delta+1) \times (2\delta+1)$), hence

$$|T| = \pi\left(\frac{m}{2\delta+1}, \frac{n}{2\delta+1}\right)$$

and the inequality stated in the theorem is proven for this special case.

In order to prove the inequality in the case $m, n \not\vdots 2\delta+1$, simply note that it is always possible to consider a subgrid of dimensions

$$(2\delta+1)\left\lfloor \frac{m}{2\delta+1} \right\rfloor \times (2\delta+1)\left\lfloor \frac{n}{2\delta+1} \right\rfloor$$

and a set of (partial) paths defined in a similar manner as in the case of the set T. The argument presented above can then be easily adopted to this case as well.

The existence of a set S providing equality will also be proven for the case $m, n \vdots 2\delta+1$ first. We will prove that we can take $S = T$ where T is the set defined above. It has the right cardinality and the condition (3) is obvious. It remains to prove that the condition (2) also holds. In order to do so, we have to show that every possible path from the set $\mathcal{P}_{m,n}$ belongs to some ball $B(P', \delta)$, where $P' \in T$.

Let $P \in \mathcal{P}_{m,n}$ be any path in the grid. We divide the grid to $(2\delta+1) \times (2\delta+1)$ squares and construct a new path P' so that

1. it goes along the edges of the big squares (then $P' \in T$); and

2. for every point c_1 on the path P there exists a point c_2 on the path P' such that $d(c_1, c_2) \leq \delta$ (then $d_g(P, P') \leq \delta$).

We will follow the path P through big squares and show for each case which edges or vertices of the big squares must be taken into the path P'. We distinguish the cases by location of start- and endpoints of the path P in a big square. There are four possible regions for start- and endpoints, each containing two segments of length δ and they are situated in four corners of a big square. There are 8 possible cases and the corresponding parts of the path P' are for all the cases shown in Figure 3.

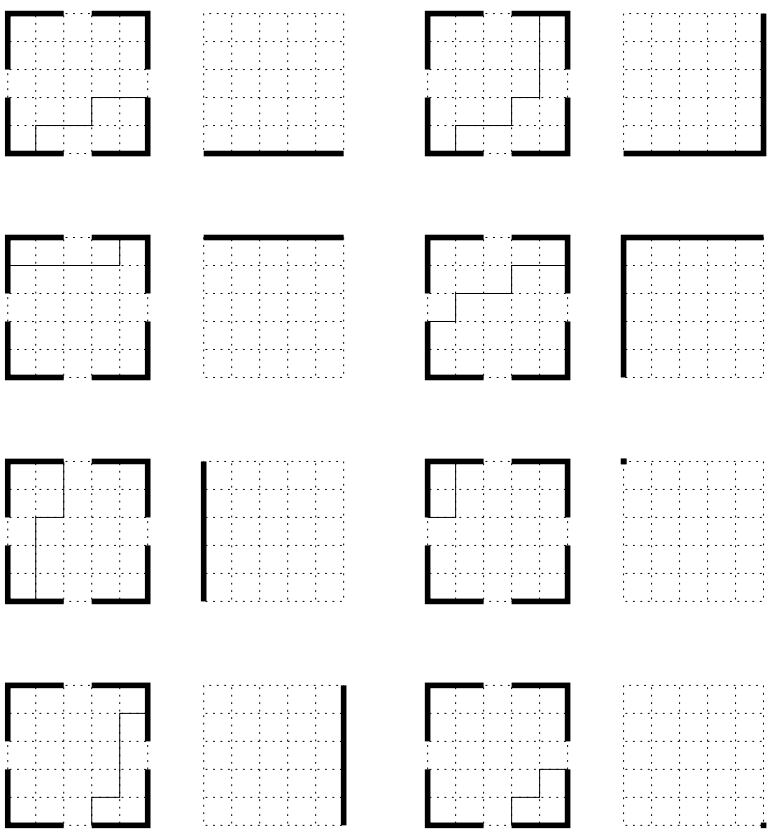

Figure 3: Construction of path P' from path P

Generalising this existence proof to the case $m, n \not\equiv 2\delta + 1$ is once again easy. As done above, we choose a subgrid with dimensions

$$(2\delta + 1) \left\lfloor \frac{m}{2\delta + 1} \right\rfloor \times (2\delta + 1) \left\lfloor \frac{n}{2\delta + 1} \right\rfloor,$$

but we have to be a bit more careful. Namely, the subgrid must be located so that no distance between an edge of the subgrid and the corresponding edge of the whole grid is not bigger than δ in order to embrace all the possible paths in the grid. Since the largest remainder of m and n when divided by $2\delta + 1$ is 2δ this locating can be done. It only remains to pad the paths given by the set T in the subgrid with any subpath joining the lower left corner of the whole grid with the lower left corner of the subgrid; and similarly for the upper right corners. □

6 Upper Bound

Theorem 2 *For every $\delta \in \mathbb{N}$ and every subset $S \subseteq \mathcal{P}_{m,n}$ satisfying the properties (2) and (3), the inequality*

$$|S| \leq \begin{cases} \pi\left(\left\lfloor \frac{m}{\delta} \right\rfloor, \left\lfloor \frac{n}{\delta} \right\rfloor\right), & \text{if } \delta \text{ is odd} \\ \pi\left(\left\lfloor \frac{m}{\delta+1} \right\rfloor, \left\lfloor \frac{n}{\delta+1} \right\rfloor\right), & \text{if } \delta \text{ is even} \end{cases}$$

holds. Evenmore, there exists such a set S that the properties (2), (3) and

$$|S| = \pi\left(\left\lfloor \frac{m}{\delta+1} \right\rfloor, \left\lfloor \frac{n}{\delta+1} \right\rfloor\right)$$

are satisfied.

Proof. The proof of this theorem is very similar to the proof of Theorem 1. First we consider the case when δ is even and $m, n \vdots \delta$. We define the set T as follows:

$$T = \{((x_0, y_0), \ldots, (x_{m+n}, y_{m+n})) \in \mathcal{P}_{m,n} : \forall i \in [0, m+n]\ x_i \vdots \delta+1 \vee y_i \vdots \delta+1\}.$$

Assume that we also have a set S corresponding to the conditions (2) and (3).

Similarly to the proof of Theorem 1, one can show that for every path $P \in S$ there exists a path $P' \in T$ such that $d_g(P, P') \leq \frac{\delta}{2}$. If for two different paths $P_1, P_2 \in S$ the corresponding path $P' \in T$ is the same, we have

$$d_g(P_1, P_2) \leq d_g(P_1, P') + d_g(P', P_2) \leq \frac{\delta}{2} + \frac{\delta}{2} = \delta,$$

a contradiction with the condition (3). Hence, the set S cannot have more elements than the set T does. As in Theorem 1, we have

$$|T| = \pi\left(\frac{m}{\delta+1}, \frac{n}{\delta+1}\right)$$

and the inequality stated in the theorem is proven in this case.

In order to generalise the result for the case $m, n \not\vdots \delta + 1$ as well, we once again consider a subgrid of dimensions

$$(\delta + 1) \left\lfloor \frac{m}{\delta + 1} \right\rfloor \times (\delta + 1) \left\lfloor \frac{n}{\delta + 1} \right\rfloor$$

situated in the center of the $m \times n$ grid. We also note that choosing $S = T$ gives the required set to achieve equality in the theorem's inequality.

The proof for the case when δ is odd is completely analogous. The only difference arises from the fact that in this case we can not require any path P to have a path in the (appropriately chosen) set T at most of distance $\frac{\delta}{2}$ away, but we have to use the value $\frac{\delta - 1}{2}$ instead. □

7 Comparison of Approaches

To demonstrate the advantage of the current approach compared to our previous heuristic one we use parameters from one of our previous real-world experiments. For a grid of 51×67 cells and with the similarity measure $\delta = 5$ the size of the seed casebase would be

$$\pi\left(\left\lfloor \frac{51}{2 \cdot 5 + 1} \right\rfloor, \left\lfloor \frac{67}{2 \cdot 5 + 1} \right\rfloor\right) = \binom{4 + 6}{4} = 210$$

cases.

At the same time, if the heuristic path generation algorithm was used, the new generated paths never covered the solution space although the experiments consisted of more than 500 runs. Another set of our previous experiments, 20 000 runs in a simulated environment, gave a similar result. Moreover, the heuristic algorithm often generated random paths that were similar to old paths already stored in the casebase. Roughly quarter of the new generated paths were innovative ones, i.e. dissimilar to all paths in the casebase.

The advantage of the new approach is thus twofold:

1. It generates the seed casebase that is guaranteed to cover the paths space of the robot while the previous heuristic approach did not give such a guarantee.

2. It speeds up learning because all paths are stored in the seed casebase. Unlike the heuristic approach, the robot does not spend time on looking for new innovative solutions.

However, if δ is reduced, the size of the seed casebase increases rapidly. For $\delta = 2$ the seed casebase would have to contain over a million cases already.

In practice, a robot would never try all the possible solutions. When it has found a solution that is good enough, it reduces exploration. When the cost of the current path increases, it starts exploring new possibilities again. The seed casebase gives a theoretical guarantee that none of the possible solutions remains undiscovered.

The idea of proving the upper bound was to check if we could show the maximum number of possible cases in the casebase. It would give a confidence to the casebase designer that the casebase would not grow bigger than a certain feasible amount. Unfortunately, the upper bound of the casebase is too high. For an example given above ($m = 51$, $n = 67$, $\delta = 5$), the casebase would contain

$$\pi\left(\left\lfloor\frac{51}{5}\right\rfloor, \left\lfloor\frac{67}{5}\right\rfloor\right) = \binom{10+13}{10} = 1144066$$

cases.

Thus, a casebase maintenance strategy that keeps the size of the casebase under control is still needed. In our previous work, we have used forgetting strategies based on the quality of a solution. For example, when the learning process is success-driven the robot remembers only best solutions. When learning is failure-driven, the robot remembers the worst solutions. The casebase is then only used to verify whether a new solution generated from scratch is good enough. Our experiments have not shown the superiority of one of the forgetting strategies. It rather seems that the casebase management technique strongly depends on the characteristics of the environment and the problem at hand.

8 Conclusions and Future Work

Our work was motivated by the fact that our previous heuristic technique was not able to generate all possible different solutions to the current findpath problem. We therefore investigated the possibility to create a seed casebase that covers the whole solution space of the robot.

In this paper, we have proven the lower and upper bound of the solution space. The proof of the lower bound shows that it is realistic to seed the casebase with a solution set that contains a close match to every possible solution path. This would give a guarantee that none of the paths theoretically remains undiscovered.

At the same time, the proof of the upper bound indicates that it is unrealistic to save all possible different solutions to the casebase. In order to keep the casebase constrained, it has to be revised at runtime.

One of the possible solutions to the casebase explosion can be based on the following observation. Although our casebase contains paths distant from each other in the sense of grid distance, many paths still overlap significantly. It would make the casebase much smaller, if we only would try to cover all the fragments of the grid with the paths instead of traversing all the paths themselves. The question how well this can be done is a subject for future research.

It is also important to emphasize that the examples above represent the number of solutions to only one problem. This problem is the most general one – traversing between the diagonally opposite corners. Other problems will be subproblems of this one and will require a smaller number of solutions to cover their solutions spaces. In practical mobile robot applications there are usually rather few target points (unless it is a mapping or exploration problem). In our future research we also intend to analyse the dependency between the number of solutions and the size of the casebase.

We also intend to investigate some different similarity measures to reduce the size of the solutions space even more. One possibility is to define similarity as the size of the area surrounded by the paths.

References

[1] M.Kruusmaa. Repeated Path Planning for Mobile Robots in Dynamic Environments. Ph.D.Thesis. Chalmes Univeristy of Technology, Gothenburg, Sweden, 2002.

[2] H.Hu, M.Brady. Dynamic Global Path Planning with Uncertainty for Mobile Robots in Manufacturing. *IEEE Transactions on Robotics and Automation.* Vol.13, No.5, October 1997, pp.760-767. 1997.

[3] K.Z.Haigh. M.M.Veloso. Planning, Execution and Learning in a Robotic Agent. *AIPS-98* pp.120–127, June 1998.

[4] C.Vasudevan, K.Ganesan. Case-based Path Planning for Autonomous Underwater Vehicles. *Proc. of 1994 IEEE Int. Symposium on Intelligent Control,* pp.160–165, August 16-18, 1994.

[5] A.K.Goel, K.S.Ali, M.W.Donnellan, A.Gomex de Silva Garza, T.J.Callantine. Multistrategy Adaptive Path Planning. *IEEE Expert*, Vol.9, No.6, Dec.1994, pp.57–65. 1994.

[6] S.Fox, D.B.Leake. Combining Case-based Planning and Introspective Reasoning. *Proc. of the Sixth Midwest Artificial Intelligence and Cognitive Science Society Conference,* Carbondale, IL, April 1995.

[7] L.K.Branting, D.W.Aha. Stratified Case-based Reasoning: Reusing Hierarchical Problem Solving Episodes. *Proc. of the Fourteenth International Joint Conference on Artificial Intelligence,* Montreal, Canada, August 20–25, 1995.

[8] K.Z.Haigh, M.Veloso, Route Planning by Analogy. *Case-Based Reasoning Research and Development, Proc. of ICCBR-95,* pp.169–180. Springer-Verlag, 1995.

[9] R.R.Murphy. Introduction to AI Robotics. The MIT Press, 2000.

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

[11] E. Belogay, C. Cabrelli, U. Molter, and R. Shonkwiler. Calculating the Hausdorff distance between curves. *Information Processing Letters*, 64(1):17–22, 14 October 1997.