Path Selection for Mobile Robots in Dynamic Environments*

Maarja Kruusmaa
Maarja.Kruusmaa@ide.hh.se
Halmstad University,
School of Computer Science,
Electrical and Electronic Engineering
BOX 823, Halmstad, Sweden

Jan Willemson
jan@cs.ut.ee
Tartu University,
Dpt. of Computer Science
Liivi 2, Tartu, Estonia

Kristo Heero
kristo@math.ut.ee
Tartu University,
Dpt. of Computer Science
Liivi 2, Tartu, Estonia

Abstract. This paper evaluates a path selection algorithm for mobile robots in large dynamic environments. The aim of the work is to reduce the risk of collisions and time of path following in cases when the robot repeatedly traverses between predefined target points (e.g. fot transportation or inspection tasks). The algorithm is usable even if very little is known about the environment or if it gets completely restructured during the mission. This article concentrates on evaluating the performance of the algorithm. The tests are completed on the mini-robot Khepera that operates in an unknown dynamically changing environment. The test results show that the path selection algorithm is able to reduce collision risk, travel time and travel distances as well as increase the predictability of robot's behaviour and its degree of autonomy.

1 Introduction

The approach presented in this paper is motivated by the fact that many mobile robot applications imply repeated traversal between predefined target points in a dynamically changing environment. Examples of this kind of implementations are fetch-and-carry task of industrial and agricultural applications or visiting certain checkpoints in security and surveillance applications.

An efficiently operating robot is expected to fulfill its assignment as fast and as safely as possible. It means that it is worthwhile to avoid situations where the robot is forced to replan its route, take a detour, can drive into a deadlock or collide with unexpected obstacles.

Real world environments are dynamic by nature. Therefore, all the possible situations that can delay the robot or imply a hazard cannot be foreseen. However, by modeling the environment or learning its properties, the time delays can be minimized and the risk can be reduced.

1.1 Problem Statement

It is further assumed that:

1. The environment is dynamic and large. It is not possible or feasible to model it precisely or keep the model constantly updated.
2. The environment contains obstacles with unknown size and location. Traversing this environment implies risk of colliding with these obstacles, being delayed when maneuvering around them or ending up in a deadlock.
3. Sensorial capabilities of the robot are insufficient to distinguish between static, dynamic and semi-dynamic obstacles (e.g. between pillars and people, steady and replaced furniture).
4. The robot is working under time constraints and it has limited computational resources. It is expected to fulfill its mission as fast and safely as possible.
5. The localization errors are small and do not accumulate (like it is with GPS) and it is therefore possible to follow a preplanned path rather precisely.

* This research was supported by Estonian Science Foundation grant ETF5613.

The problem we aim at solving is the following: find paths between previously determined target points so that following them minimizes risk of collisions and speeds up the mission.

Our approach to the problem is based on the following observation. In a dynamic environment with an unknown obstacle distribution, the best path to the goal is not necessarily the shortest one. Depending on the nature of the environment, there may exist routes that are longer but easier to follow. By introducing a path generation algorithm, the robot can test several alternatives to reach the goal. By remembering its path following experiences, it can learn to follow paths that save time and reduce risk. As the environment changes, the robot will reevaluate its past experience and adapt to use new easily traversable paths.

1.2 Related Work

For a mobile robot operating in an uncertain environment, a natural requirement is obstacle avoidance (e.g. in case of several implementations of potential fields [3]). However, obstacle avoidance, no matter how good it is, always implies a collision risk due to the uncertainty in sensor readings, motion planning, obstacle position or computational imprecision.

To minimize collision risk and time delays it is therefore important to select paths where as few obstacles as possible are encountered. Very few research studies reported so far consider the problem of path selection in dynamic environments [1, 2]. Unlike these approaches, we do not assume that the structure of the environment is known a priori.

The other difference from the above-cited approaches is that instead of the topological map we use a grid-based map to model the environment. The advantage of the grid-based map in this context is that the model does not have to be reorganized when the world changes and it therefore permits the robot to learn and adapt even in situations when very little is known about the environment or when the whole structure of the environment changes.

In our own previous work, we have used an heuristic algorithm to generate new innovative paths [4]. The tests have revealed that the robot is quickly able to adapt to the changes in the environment. As a result, the risk of collisions, time delays and traveled distances are minimized.

However, the tests also pointed to the shortcomings of the approach. During the tests it was observed that many paths that the heuristic algorithm generated were 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 local minima – some paths that would have been easy to follow where never generated.

To overcome this problem we created a new algorithm for paths generation [6]. The aim was to generate a minimal cover of the solution space. We proved that we can seed the memory with a relatively small number of alternative solutions and can keep it constrained without loosing the robots ability to learn and generalize.

This paper tests another, improved path selection mechanism. In the next section, we describe the algorithm and motivate our approach. Next, we describe some experiments with the mini-robot Khepera to compare our approach with the traditional shortest-path following and demonstrate the advantages of our approach. Finally, we end this paper with conclusions and directions for the future work.

2 Path Selection in Dynamic Environments

Figure 1 describes building blocks of our approach. The global planner receives tasks from the user. The tasks are requests to move to a specific point from its present location. Given a new task, the global planner can trust on its ealier experiences stored in the memory and use paths that have proven to be easy to traverse. If the properties of the paths in the memory are not good enough (e.g. too risky, long or crooked), the global planner can use the path generation unit to suggest new solutions for traversing to the given goal.

The task of the path generation unit is to generate innovative solutions to the path planning problem using the map of the environment. If a new solution turns out to be good, it can be stored in the memory and used later when the same problem will be encountered.

The chosen path will be 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.

In this paper we concentrate on finding a good path generation mechanism that generates possibly dissimilar paths from a given start to a given goal thus permitting the robot to try as many alternative approaches in its search for better paths as possible.

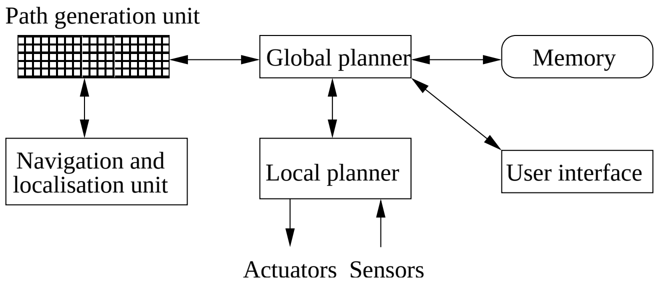

Fig. 1. General overview of path planning

Theoretically, the number of different paths on a grid-based map is overwhelming. There are too many alternatives to travel between two points and the robot could never try them all. In addition, most of those paths are infeasibly long, crooked and two difficult to follow. So the aim of the path selection algorithm is to:

1. generate paths that are easy to follow if they are free from obstacles;
2. generate paths that are as much different from each other as possible to let the robot find out as many innovative solutions as possible;
3. provide a mechanism that is able to discover virtually all possible alternatives;
4. cover the whole space of innovative solutions with as few alternatives as possible in order to maintain the robot's ability to generalize and keep the memory constrained.

We propose a method that works by dividing the grid into paths segments and then generating paths that cover all these segments. Full description of the method and its formal analysis is presented in [5].

Figure 2 illustrates one possible cover of the 3×4 grid.

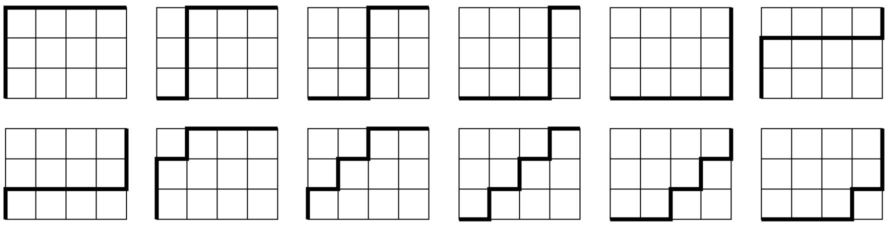

Fig. 2. Cover of the 3×4 grid

The paths selected by the robot are limited to those not having back turns and covering all the grid segments of length 2. In practice, paths relaxation is used to smoothen the paths and the zig-zags will be straightened.

It is proven in [5] that for a grid of the size $m \times n$, the cardinality of the minimal cover is $2m + 2n - 2$ paths. It means that the number of different paths is very small and grows linearly with a small constant, that makes it well scalable for very large domains.

While [5] gives a thorough insight to the theoretical aspects of the algorithm, this paper concentrates on demonstrating its advantages in practice. The following sections describe the test environment, experiments and compares the results of the path selection algorithm to the shortest paths approach.

3 Experimental Setup

The experiments are conducted using mini-robot Khepera. Khepera is a differencial drive miniature circular robot (with radius 26 mm) equipped with IR sensors for collision avoidance and it can be connected to a PC over serial link.

The size of our test environment is 1860×1390 mm. Since Khepera lacks sensors for accurate localization we solved this problem with a global vision system. The localization system is presented in figure 3. A video camera is mounted to the ceiling to recognize the position of the robot. The PC processes the camera image to find robot's position and an algorithm running on it controls the robot over a serial link. In this way the localization errors are rather small (usually comparable to the size of the robot).

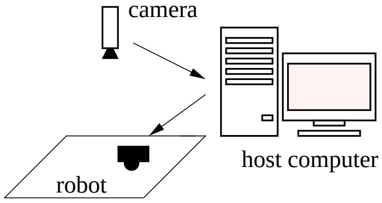

Fig. 3. Localization system

The environment is represented in figure 4, left. Figure 4, right, represents the same environment as shown from the overview camera.

Fig. 4. Environment in reality and as seen from the overview camera

The presence and location of the obstacles on the scene is randomly changed and unknown for the robot. The details of the obstacle distribution are presented in table 1. The frequency of obstacle replacement depends on its size, the 8 smallest obstacles are replaced after every traversal, and so the robot faces a changed environment for every traversal. An independent program using a random number generator determines the locations of the obstacles before every traversal.

	I-shapes	L-shapes	C-shapes	rect. 1	rect. 2	rect. 3	rect. 4
amount	2	2	1	1	1	10	8
size (mm)	320×40	120×320 & 160×280	$160 \times 240 \times 160$	140×385	100×105	80×80	40×40
changing probability	0	0	0	0	0	0	1

Table 1. Obstacles

Figure 5 represents the model of the environment. The robot does not know anything about its environment except its size and the location of the target points. It has no idea about the presence and location of the obstacles, neither does it know about the program that determines the obstacle distribution. Therefore its model of the environment describes only the geometry of the environment and the location of its target points (marked with the letter G). Encountered obstacles that cause the robot to replan its route are marked with black rectangles. Since the robot cannot distinguish between frequently replaces and static obstacles, the obstacle map is cleaned after every traversal. The solid line represents the planned path from start to goal, the dotted line represents the actually followed path.

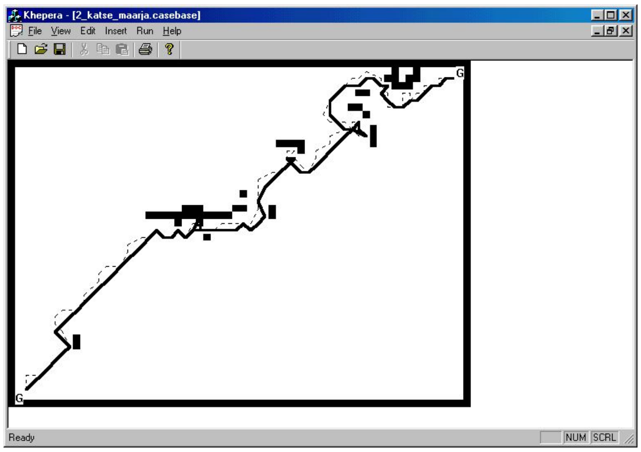

Fig. 5. Model of the environment

The mission of the robot consists of 50 tasks of traversing between the target points. Every task implies selecting a path, following the path, replanning on-line when obstacles occur and storing the parameters of the path when reaching the goal.

To evaluate the efficiency of our path selection algorithm, two missions consisting of 50 tasks are compared. The first mission is the conventional path following procedure. The robot always follows the shortest path to the goal. A wave transform algorithm is used to calculate the shortest path [7].

The second mission tests our path selection algorithm. When a new task is given, the robot either uses the path selection algorithm to generate an innovative path or selects an old path from its memory. The better the path in its memory is, the more likely it will be selected. The criterion of the path quality is the number of obstacles encountered on its way:

$$\text{path choosing probability} = \begin{cases} 1, & \text{if replannings} < 5; \\ 0, & \text{replannings} > 30; \\ 1 - \text{replannings}/30, & \text{otherwise.} \end{cases}$$

If the path selection algorithm is efficient, it helps the robot to find innovative paths that are easier traversable than the shortest path. As a result, the risk of collisions and time delays should be reduced.

4 Experimental Results

Table 2 represents our experimental results.

	replannings	time (s)	deviation (mm)	length (mm)	interruptions
shortest path algorithm	23,8	273,1	381,5	3878,5	20
path selection algorithm	9,9	191,2	281.0	3020,7	8
performance improvements	58%	30%	26%	22%	60%

Table 2. Experimental results

Five parameters were recorded and compared. The number of replannings is the parameter we explicitly wanted to minimize. Every time the robot detects an obstacle in its vicinity, it will replan its path. The number of replannings thus reveals the number of situations where a possible collision with an obstacle would have occurred. The test results show that the path selection algorithm reduced the number of possibly hazardous situation more than by half.

Although the paths generated by the path selection algorithm are longer than paths generated by the shortest path algorithm, they lead the robot faster to the goal. The time of path following was reduced mainly because the robot did not spend so much time on replanning its route and maneuvering between obstacles.

The deviation from the originally planned path reveals the predictability of robot's behaviour. When the deviation is small, the robot followed more closely the route that it intended to. The path selection algorithm helped to reduce the deviation by 26%.

Although we did not aim at minimizing the traveled distance, on the contrary, longer paths were preferred if they were safer, the paths selection algorithm reduced the length of the paths by 22%. The reason is mainly that maneuvering around the obstacles increases the traveled distances.

During the tests, there were occasions when the robot got stuck in the corners of obstacles or did not manage to localize itself and therefore needed help from the experimenter. These cases were removed from statistics since the mission usually got interrupted and data was incomplete. However, it is important to notice that when the paths selection algorithm was used, the autonomy of the robot significantly increased. The occasions when the experimenter had to intervene were reduced by 60%.

Generally, it can be said that compared to the shortest path algorithm, the path selection algorithm improved all the measured parameters of the performance of the robot.

5 Conclusions and Future Work

This paper analyzed a path selection algorithm for repeated traversal in dynamic environments. The algorithm generates innovative paths between predefined target points and helps the robot to reduce time and risk of collisions. Compared to the other approaches to this problem, this approach is usable even when very little is known about the environment or when the environment is completely restructured during the mission. The disadvantage of the approach is that if very little is known about the surrounding, the robot needs a global localization system to keep track on its position.

The advantage of the current approach compared to our own earlier work is that we can prove theoretically the number of possible solutions and show that a relatively low number of solutions is needed to cover all path segments. An even more important theoretical outcome is that the algorithm scales up well to large domains. The number of solutions increases linearly with a small constant when the size of the environment increases. The practical advantage of this algorithm was supported by experiments. The algorithm helped to reduce the number of replannings, travel time, and distances as well as to increase the predictability of robot's performance and its degree of autonomy.

A possible direction of the future work could be to modify this approach to solve combined problems where the robot simultaneously works on two missions. The possible examples are carrying a load and cleaning the environment, or visiting checkpoints and inspecting the area at the same time. In these kind of missions it is favorable to use as different paths between target points as possible. The path selection algorithm will very efficiently cover the whole space even if the environment is large.

References

1. Karen Zita Haigh and Maria Manuela Veloso. Planning, execution and learning in a robotic agent. In *AIPS-98*, pages 120 – 127, June 1998.
2. H. Hu and M. Brady. Dynamic Global Path Planning with Uncertainty for Mobile Robots in Manufacturing. *IEEE Transactions on Robotics and Automation*, 13(5):760–767, October 1997.
3. J.S.Zelek. Dynamic issues for mobile robot real-time discovery and path planning. In *Proc. of Computational Intelligence in Robotics and Automation (CIRA'99)*, pages 232–237, 1999.
4. Maarja Kruusmaa. Global navigation in dynamic environments using case-based reasoning. *Autonomous Robots*, 14:71–91, 2003.
5. Maarja Kruusmaa and Jan Willemson. Algorithmic Generation of Path Fragment Covers for Mobile Robot Path Planning. Submitted for 11th Annual European Symposium on Algorithms, 2003.
6. Maarja Kruusmaa and Jan Willemson. Covering the Path Space: A Casebase Analysis for Mobile Robot Path Planning. *Knowledge-Based Systems*, 2003. Elsevier Science, to appear.
7. A. Zelinsky. Using path transforms to guide the search for findpath in 2D. *The Int. Journal of Robotics Research*, 13(4):315–325, August 1994.

This article was processed using
the TEX macro package with ECMR2003 style