Heero, K.; Aabloo, A. & Kruusmaa, M. / Learning Innovative Routes for Mobile Robots in Dynamic Partially Unknown Environments, pp. 209 - 222, International Journal of Advanced Robotic Systems, Volume 2, Number 3 (2005), ISSN 1729-8806

Learning Innovative Routes for Mobile Robots in Dynamic Partially Unknown Environments

Kristo Heero, Alvo Aabloo & Maarja Kruusmaa
Institute of Technology, Tartu University
Vanemuise 21, 51014 Tartu, Estonia
maarja.kruusmaa@ut.ee

Abstract: *This paper examines path planning strategies in partially unknown dynamic environemnts and introduces an approach to learning innovative routes. The approach is verified against shortest path planning with a distance transform algorithm, local and global replanning and suboptimal route following in unknown, partially unknown, static and dynamic environments. We show that the learned routes are more reliable and when traversed repeatedly the robot's behaviour becomes more predictable. The test results also suggest that the robot's behaviour depends on knowledge about the environemnt but not about the path planning strategy used.*
Keywords: *path planning, mobile robots, dynamic environment, robot learning.*

1. Introduction

Mobile robots in human inhabited environments should operate safely and reliably. At the same time they are expected to minimise energy consumption, travel time and distance. Optimisation of these parameters implies that the robot must be able to predict its behaviour in a partially unknown and changing environment.

Robots use path planning algorithms to plan a path from start to goal. In dynamic environments the environment can change during path following. To avoid collisions with unknown obstacles robots use local replanning. While classic AI planners are used to produce the global path to the goal, local replanners usually act reactively.

Mobile robot path planning is considered to be a well-established field incorporating several efficient path planning techniques and their modifications [1,2,3,4,5]. These path planners are proven to give a globally optimal plan in a static completely modelled environment. Several implementations also prove that by combining global and local path planning methods mobile robots are able to operate in dynamic environments [6,7,8,9]. However, there is no study that investigates their efficiency during long periods of time and the predictability of their behaviour.

Some studies investigate the long-term behaviour of localisation methods [10], fault tolerance [11] and reactive behaviour [12] but efficiency of path planners is usually evaluated only by examining few trials. Results prove that a robot using path planning methods is able to negotiate unknown obstacles and find a way to the goal but do not address the problem of the stability of its behaviour and optimality of solutions.

The reason for it might be that most of mobile robot path planners originally stem from the research with robot manipulators. The algorithms worked perfectly in small-size well-defined worlds and were easily adapted for 2D environments. As the field of mobile robotics developed the test environments grew large and more complex. Local replanning techniques and probability maps were introduced to cope with the new situation. At the same time global behaviour of path planning algorithms have remained unattended.

Predictability of robot's behaviour is an important parameter of a mobile robot. A robot that can predict its behaviour can estimate its position after a certain period of time. Increased predictability, in turn, makes it possible to optimize other parameters like travel time, distance, energy consumption, collision risk, etc. For robots working in a team predictability of each other's actions is inevitable. A robot interacting with a human is also expected to be stable and predictable. Knowing robot's location with a great precision makes it easier to find a lost robot in very large and complex environments.

The goal of this study is to investigate robot's behaviour in large and partially unknown environments as well as

to find a way to increase predictability of robot's behaviour. We have set up a model environment for a mini-robot Khepera [13] and completed 600 test runs to investigate its long-term behaviour. We plan paths with wave transform algorithms widely used in mobile robotics [14]. The questions that we aim at answering are the following:

- How much does the efficiency of path planning algorithms depend on the accuracy of the world model?
- How much does explicit global replanning help to improve the performance of a path planner?
- How important is it to find a globally optimal path?
- Is it possible to increase predictability of robot's long-term behaviour if the environment is dynamic and partially unmodelled?

To answer the last question we introduce a learning method for fetch-and-carry tasks that looks for reliable trajectories in a partially modelled dynamic environment and compare its efficiency against shortest path following with a wavetransform algorithm. The results show that in a complex environment there may exist paths that are easier to follow than the shortest path. Finding these innovative traits and following them helps to reduce collision risk as well as to minimize travel time, distance and deviation from the originally planned path.

To examine the performance of path planners we verify:

- Optimal path planning to sub-optimal path planning
- Local path planning to global path planning
- Path planning in static and dynamic environments
- Path planning in known partially known and unknown environments.

The next section we describe finding innovative paths. The goal of this method is to find reliable trajectories to complete fetch-and carry tasks. Next, we describe the experimental setup. In section 4 we represent the experimental results. Section 5 discusses the experimental results.

2. Finding Innovative Routes

To illustrate the problem and to explain its relevance let us verify two test runs with the robot Khepera (Fig. 1 and Fig. 2). The robot has to traverse from the start point in the lower left corner of the map to the goal point in the upper right corner in both cases. The environment is completely unknown. Black rectangles on the grid map represent obstacles that the robot has detected during path following. Every detected obstacle causes the robot to replan globally.

Replanning is done with a wavetransform algorithm. The solid line on the figures is the path that the robot has planned, the dotted line is the actual path detected with an overview camera. The only difference is that while in Fig. 1 the robot always plans the shortest path, in Fig. 2 the robot first takes a sub-optimal path to the goal until it counts the first unknown obstacle. At this point both of the algorithms become identical.

Eventually it appears that the robot in Fig. 1 has taken a much longer route to the goal and deviated considerably from its original course. The sub-optimal path in Fig. 2 appears to be much shorter and easier to follow. Since the robot counts only few obstacles, collision risk is less. Replanning does not cause large deviation. Even when the robot replans it eventually drifts back to its original course.

Many mobile robot applications assume repeated traversal between predefined target points. The most common is a transportation task (fetch and carry), but also surveillance, guiding, rescuing etc. may fall into this category. If the robot was able to find stable trajectories like in Fig. 2 then following them would considerably speed up the mission, reduce risk to the robot or to the environment and reduce energy consumption.

The problem is thus how to find trajectories that increase the predictability of robot's behaviour and reduce risk.

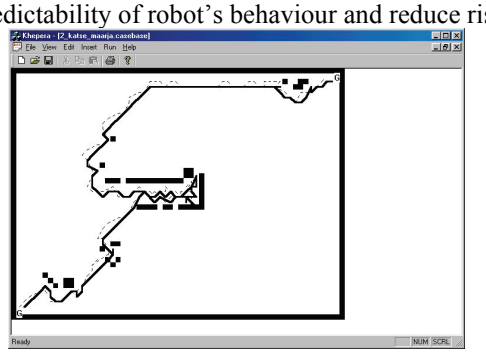


Fig. 1. Path planning in an unknown environment. Shortest path planning with a wavetransform algorithm

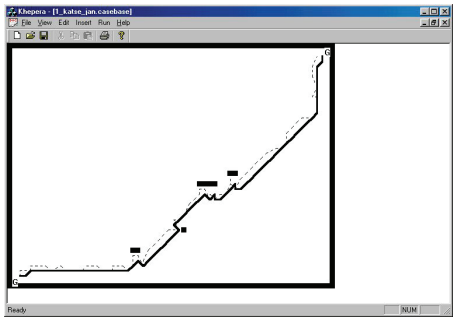


Fig. 2. Path planning in an unknown environemnt. Sub-optimal path following

One possibility would be to include information about reliability on the map. For grid-based maps, there exist reliable methods for incorporating ambiguity caused by sensor readings to the map [15]. Land-rovers also consider roughness and slope of the terrain to find the most reliable path [16,17]. The optimal path is thus not expressed in terms of distance, but in terms of a cost function considering the traversability of the path. Cost functions are also used with topological path planning since exact distances between graph nodes are often unknown [18].

To increase reliability of path planning it would be necessary to include information about dynamic obstacles to the map. In [19] this is done by observing the obstacle distribution with an overview camera. A path planner then avoids areas that are more likely occupied. If it is not possible to observe the environment globally, the map is very difficult to keep up to date.

Moreover, from Fig. 1and Fig. 2 it appears that some obstacles deviate the robot more away from its original path than others and to decrease deviation form the original path this information should also be taken into account.

To increase predictability of robot's behaviour, the grid map had to reflect the ambiguity caused by sensor readings, localization errors, dynamic obstacles and their effect to replanning procedures. To permit efficient path planning this map must be constantly updated. At the moment there is no such a method available that combines all these sources of uncertainty in a reliable manner and is easy to manage in real time.

To increase the predictability of path following for fetch and carry tasks we look at the problem from an opposite viewpoint. Instead of modelling the environment as accurately as possible we look for paths where inaccuracy of the world model does not significantly influence the result of path planning. Practically we choose a set of sub-optimal paths and evaluate their traversability by trail-and error until we find some that satisfies our criteria of safety and reliability.

This approach is applicable for fetch and carries tasks since it implies repeated traversal between predefined targets. It also presumes that localisation errors are small and do not accumulate since the robot is expected to determine its position rather accurately. It is therefore easiest to apply the method if GPS or pseudolite navigation is used, for example like in [20].

2.1 Problem Statement

It is further assumed that:

1. The environment is dynamic and large. It is not possible or feasible to model it precisely and/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 manoeuvring 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 environment is unstructured or the structure of the environment is unknown.
5. Mapping, path planning and localisation are not the main objectives of the robot. These are presumptions to make the successful completion of a mission possible. Therefore they cannot take all of time and the computational recourses as some resources are also needed for the main task.
6. The robot is expected to fulfil its mission as fast and safely as possible.
7. Localisation errors are small and do not accumulate and therefore it is possible to follow a pre-planned path rather precisely.

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

Our approach to the problem solving 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. Depending on the nature of the environment, there may exist routes that are longer but easier to follow in terms of time or safety. By generating innovative tracks with 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 re-evaluate its experience and will adapt to use new easily traversable paths.

2.2 Suboptimal path generation

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 unfeasibly long, crooked and difficult to follow. So the aim of the path selection algorithm is to:

- generate innovative routes that are easy to follow if free from obstacles;
- generate paths that are as much as possible different from each other to let the robot find out as many innovative solutions as possible;
- provide a mechanism that in practice is able to discover virtually all possible alternatives;
- cover the whole space of innovative solutions with as few alternatives as possible in order to maintain the robot's ability to generalise 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. The full description of the method and its formal analysis is presented in [21].

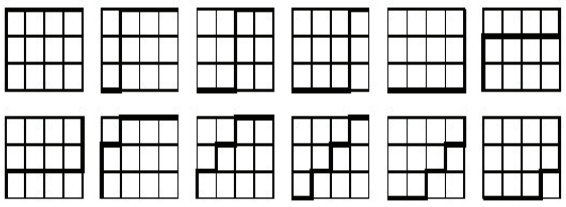


Fig. 3. The cover of a 3 × 4 grid

Fig. 3 illustrates one possible cover of a 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 [21] that for a grid of the size m × 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, which makes it well scalable for very large domains.

2.3 Learning Innovative Routes

In the previous section we presented a method of covering the m × n grid map with a set of sub-optimal path. We assume that although these paths are longer than the shortest one, in a dynamic and partially modelled environment some of these routes may be easier to follow. Since the map may be inadequate and does not contain information that permits to evaluate reliability of paths, we can evaluate their traversability only after following them. Therefore the application of the algorithm is also limited to missions that assume repeated traversal between predefined targets. The general form of the algorithm represented in Fig. 4.

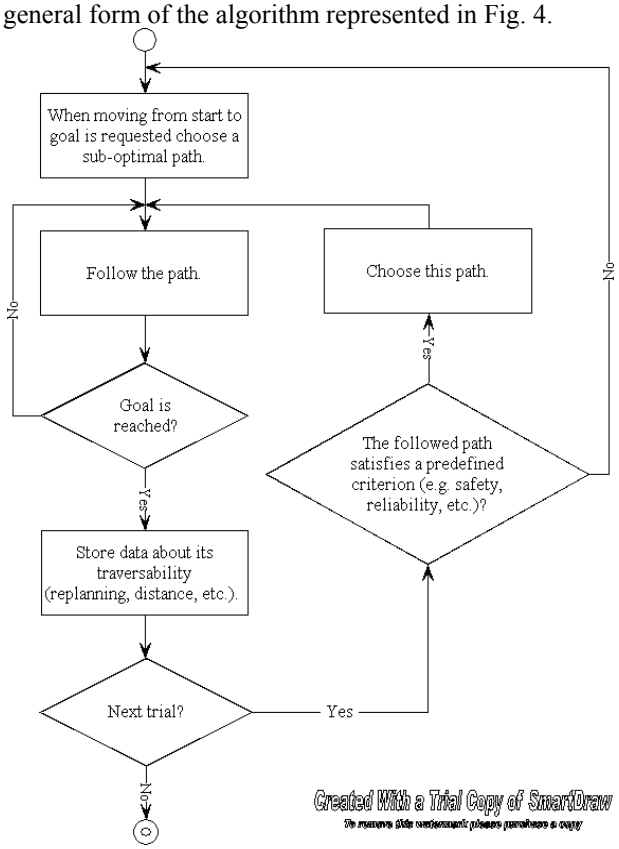


Fig. 4. The path selection algorithm

In the next section we describe the experimental design and after that represent the results of the experiments. The experiments were set up to verify the path selection algorithm against shortest path following, specifically to show how dynamic or how well modelled the environment can be for the path selection algorithm to outperform shortest path following. For the sake of completeness and in order to draw some more general conclusions we have also included some test runs published earlier in [22] and [23].

3. Experimental Design

3.1 Robot

The experiments are conducted using a mini-robot Khepera (see Fig. 5). It is a differential drive miniature circular robot (with radius 26 mm) equipped with IR sensors for collision avoidance and it can be connected to a PC over a serial link

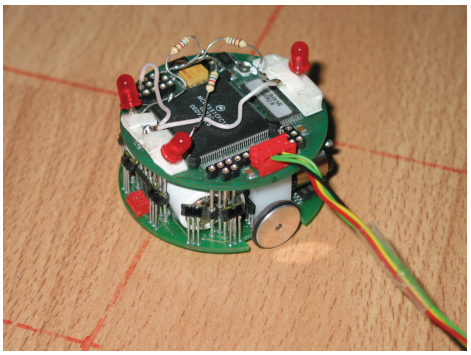


Fig. 5. The Khepera robot. LEDs are mounted on its top to detect it with an overview camera

3.2 Localization

The localization system is presented in Fig. 6. A video camera is mounted to the ceiling to recognize the position and orientation of the robot. The PC processes the camera image to find robot's position and a computer algorithm controls the robot over a serial link. In this way the localisation errors are rather small (usually comparable to the size of the robot).

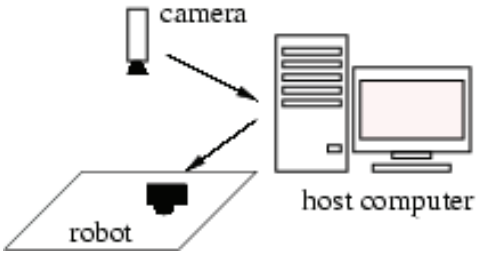


Fig. 6. Localization system with an overview camera.

3.3 Test environments

To test the path planning algorithms we have set up a test environment represented in Fig. 7. The size of the test environment is 1860 mm × 1390 mm and it contains replaceable obstacles. Fig. 8 represents the same environment observed though the overview camera. Khepera is detected with the help of 3 LEDs mounted on its top. The robot is shown in the middle of the Fig. 8, the serial cable connecting it to the computer is also visible. The isosceles triangle formed by the LEDs determines the position as well as orientation of the robot.

Fig. 7. The test environemnt

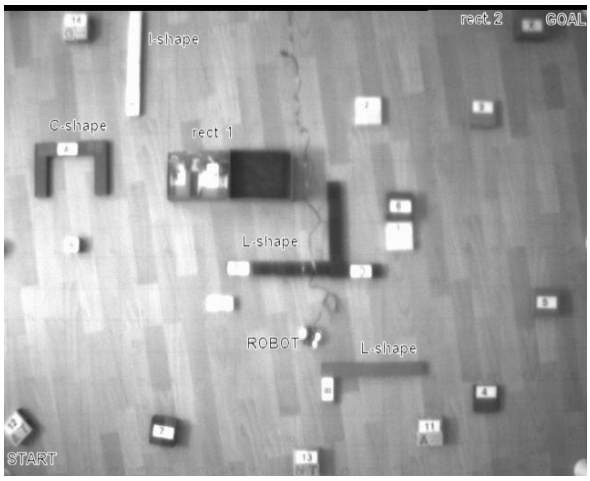


Fig. 8. The test environment seen through the overview camera

A computer program controls the robot, detects its position with the overview camera and records its behaviour. The user interface of the program is shown in Fig. 9.
The robot repeatedly traverses between the lower-left and upper-right corner of the test environment and back again. Black rectangles in Fig. 9 are occupied cells; either modeled *a priori* or detected with the IR sensors of Khepera. Every obstacle that the robot detects forces it to replan its route according to a previously determined strategy. The solid line is the planned route; the dotted line is the actual route of the robot recorded with help of the localization system.

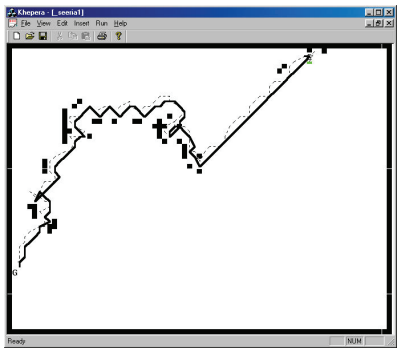


Fig. 9. The user interface of the control program

3.4 Obstacles
The specification of the obstacles is given in. Their location is generated with a random number generator. The location of the larger obstacles specified as I-shape, L-shape, C-shape, *rect.1* and *rect2* is kept constant during all the test runs (Fig. 8 shows their locations). In dynamic environments we change the location of obstacles *rect. 3* and *rect.4*. Keeping the location of the static obstacles constant during all the test runs makes it possible to verify the experimental results.
The performance of the robot is verified in 6 environments with a different degree of dynamicity and different amount of *a priori* known information. In next subsections, we describe in detail the path planning algorithms and test environments.

3.4.1 Environment 1 – static, known
This environment is set up to test the performance of the robot in perfect conditions.
The environment 1 is the complete and correct model of the environment. Locations of all obstacles is precisely known and do not change. The model is represented in Fig. 10. The start and goal points are indicated with a letter G.

Fig. 10. The complete model of the environment. The letter G indicates start and goal points

3.4.2 Environment 2 – static, large obstacles known
The physical environments 1, 2 and 3 are identical, only the models of the environments differ. All the obstacles in this environment are kept static during the tests.
The environments 2 and 3 were set up to verify the performance of the path selection algorithm in a partially modelled world.
In environment 2 the large obstacles are known a priori while the presence and location of small obstacles is unknown (see Table 1 for details). The model of the environment 2 is shown in Fig. 11.

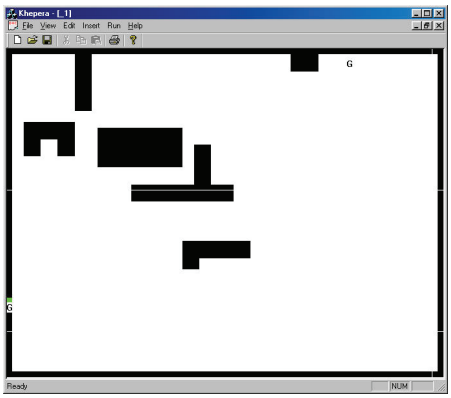


Fig. 11. The partially modeled environment. Large obstacles are modeled

3.4.3 Environment 3 – static, small obstacles known
The model of the environment 3 is the reverse of the model of the environment 2. All obstacles that were known in environment 2 are unknown while all obstacles, which were unknown in environment 2, are known here (see Table 2). Like it was stated above, the physical environment that the robot traverses is the same for environments 1,2 and 3. The model of the environment 3 is represented in Fig. 12.

	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 × 240 × 160	140 × 385	100 × 105	80 × 80	40 × 40

Table 1. Specification of the obstacles

	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 × 240 × 160	140 × 385	100 × 105	80 × 80	40 × 40
changing probability	0	0	0	0	0	0	0
type	modeled	modeled	modeled	modeled	modeled	unknown	unknown

Table 1a. Obstacles in environment 2

	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 × 240 × 160	140 × 385	100 × 105	80 × 80	40 × 40
changing probability	0	0	0	0	0	0	0
type	unknown	unknown	unknown	unknown	unknown	modeled	modeled

Table 2. Obstacles in environment 3

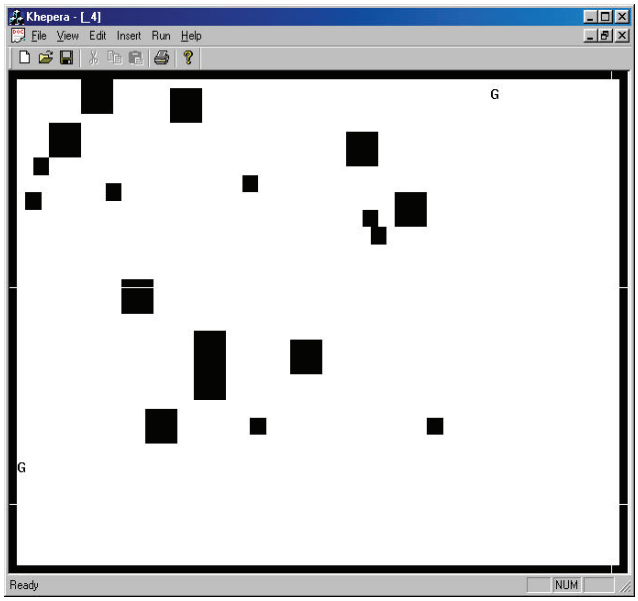


Fig. 12. The model of the test environment with small obstacles modeled

3.4.5 Environment 4 – static unknown

The test environment 4 was set up to test the performance of the path selection algorithm in an extreme case where nothing is known about the environment except its size and goal locations.

The model of the environment is represented in Fig. 13. All obstacles are static (see Table 3).

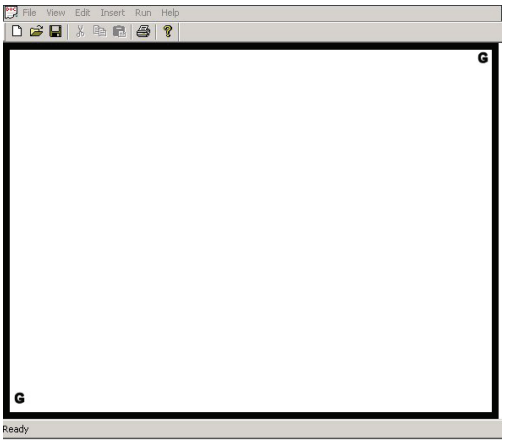


Fig. 13. Model of the unknown environment

3.4.6 Environment 5 – slightly dynamic, unknown

The environments 5 and 6 are dynamic and unknown. Tests in these environments evaluate how well does the path selection algorithm adapt to the environmental changes.

Environment 5 is unmodelled with the 8 obstacles replaced after every trial. The model of the environment is represented in Fig. 13.

Obstacle distribution is presented in Table 4. Locations of all 8 obstacles specified as *rect. 4* are replaced before every trial with probability 1. Their new locations are generated randomly. If the new location of an obstacle coincides with an existing obstacle, a new location is generated.

	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 × 240 × 160	140 × 385	100 × 105	80 × 80	40 × 40
changing probability	0	0	0	0	0	0	0
type	unknown	unknown	unknown	unknown	unknown	unknown	unknown

Table 3. Obstacles in environment 4

	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 × 240 × 160	140 × 385	100 × 105	80 × 80	40 × 40
changing probability	0	0	0	0	0	0	1
type	unknown	unknown	unknown	unknown	unknown	unknown	unknown

Table 4. Obstacles in environment 5

	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 × 240 × 160	140 × 385	100 × 105	80 × 80	40 × 40
changing probability	0	0	0	0	0	0.2	1
type	unknown	unknown	Unknown	unknown	unknown	unknown	unknown

Table 5. Obstacles in environment 6

A realistic environment can change during the path following. However, this is a feature that we were not able to model, since changing the environment during the path following would have disturbed the localisation system that uses an overview camera. We therefore changed the environment before every trial. In this way the robot faces a new changed world every time it travels to its goal location, but it does not observe the environment changing during the path following.

3.4.7 Environment 6 – moderately dynamic, unknown

The model of the environment is the same as in Fig. 11. Environment 6 is more dynamic as 18 obstacles are replaced after every trial. The obstacle distributions are represented in Table 5. Like in the environment 5, the obstacles specified as *rect. 4* are replaced with the probability 1.0. In addition 10 more, twice as large obstacles *rect. 3*, are replaced after every trial, each with a probability 0.2.

Dynamic obstacles actually correspond to those that are unknown in environment 2 while all static obstacles correspond to those that are known in environment 2.

3.5 Algorithms

This subsection describes path planning strategies used to evaluate the path selection algorithm. The performance of the path selection algorithm is verified against shortest path following with wavetransform algorithms. The wavetransform algorithm is selected as one of the commonly used path planning methods on a grid map. Also our own path selection algorithm uses the same wavetransform methods for replanning. Therefore we can assure that any differences in performance are caused only by the learning algorithm but not by the replanning method.

The robot uses 3 different path planning strategies:

1. Shortest path following always plans the shortest path to the goal from its current position.
2. Path selection with global replanning uses the path selection algorithm to look for suboptimal routes to the goal. It follows the suboptimal route until the first unexpected obstacles is detected. Then the shortest path following is used to reach the goal.
3. Path selection with local replanning uses the path selection algorithm to find suboptimal routes to the goal. When unexpected obstacles are detected, local replanning is used to find its way back to the pre-planned route.

Next, we describe every path planning strategy in detail. The program code corresponding to every algorithm is available at http://math.ut.ee/~kristo/khepera/.

3.5.1 Shortest path following

Fig. 14 describes the shortest path following strategy. A robot using this strategy always tries to find the shortest possible path to the goal considering all global information available. To find the shortest path, the distance transform algorithm is used [14]. This algorithm always finds the globally optimal path to the goal and at the same time keeps the robot at the safe distance from obstacles. This strategy uses no memory to store the information about previously followed tracks.

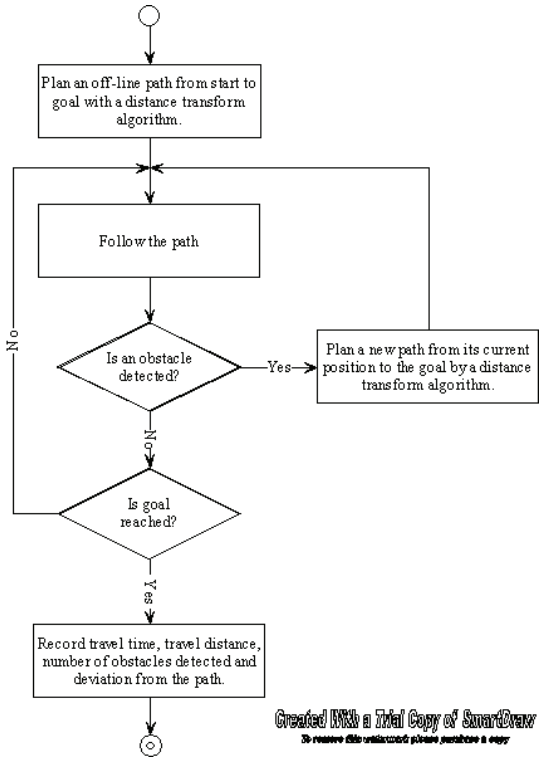


Fig. 14. The algorithm of shortest path following

3.5.2 Path selection with global replanning

Fig. 15 represents the flow diagram of the path selection algorithm with global replanning. The bottomline of this path following strategy is to first travel a suboptimal route and if this route appears to be blocked with an obstacle, abandon the pre-planned route and turn to the shortest path following strategy.

The followed path is usually different from the pre-planned one because unknown obstacles deviate the robot from its initial course. It is also crooked since the robot repeatedly corrects its localization error and avoids collisions with obstacles. The zigzags of the followed path are therefore straightened and gaps and cycles are removed. This procedure is documented in [24]. When the goal is reached the followed path is remembered and stored together with statistical data (nr. of obstacles detected, travel time, distance, deviation from the original path).

This strategy minimizes the risk of path following; therefore it chooses routes that are most likely to be free from obstacles. The robot first looks for a stored path with an acceptably low number of replannings while repeatedly traversing between the target points. If such a path exists, it is repeatedly followed as long as it satisfies the robot's criterion of safety. Otherwise a new suboptimal route is chosen, followed and remembered. The criterion of path selection among the stored paths is the probability inversely proportional to the number of unexpected events:

Equation 1

$$probability = \begin{cases} 1, replannings < 5 \\ 0, replannings > 30 \\ 1 - replannings / 30, otherwise \end{cases}$$

Obviously, the efficiency of this algorithm depends on how fast the robot finds a route with acceptable parameters. When this route or several good routes are found, it usually keeps following them, robot's behaviour stabilizes and the overall performance increases. The final result thus largely depends on how many trials are performed and how lucky the robot is to find a good route. Longer test runs would thus increase the efficiency of the path selection algorithm compared to shortest path following strategy which does not learn and remember and therefore does not improve its performance. To avoid a biased interpretation of the test results we have therefore analyzed separately the parameters of innovative suboptimal routes and learned (repeatedly traversed) routes.

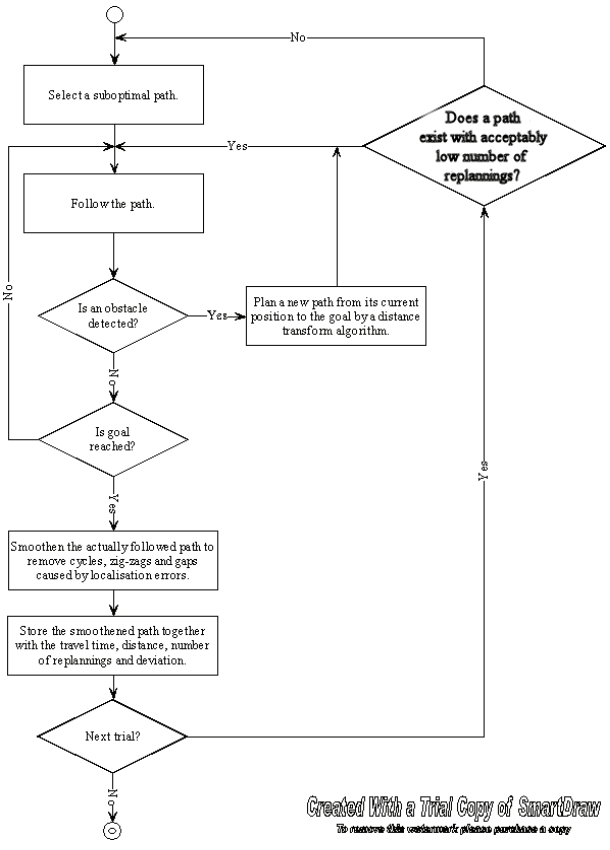


Fig. 15. Path selection algorithm with global replanning

3.5.3 Path selection with local replanning

This path planning strategy described in Fig. 16 follows the initially chosen sub-optimal route to the end. When the unexpected obstacle is detected, it plans a path around it and tries to return to the pre-planned path. If new obstacles are detected during replanning, it always tries to turn back to the pre-planned path a little further away from the initial returning point. The local replanning algorithm is therefore somewhat smarter than the plain global task decomposition algorithm. It does not try to reach to a certain sub-goal that could be hard to achieve but chooses a new sub-goal. Put differently, it avoids sub-goal obsession, as R. Murphy calls it [25]. To replan locally the robot uses the same distance transform algorithm that is used for shortest path planning.

Like in case of the previous strategy, the test results depend on how lucky the robot is to find a good route and how long is the test run. Therefore in experimental results, the innovative suboptimal path and learned path are analyzed separately.

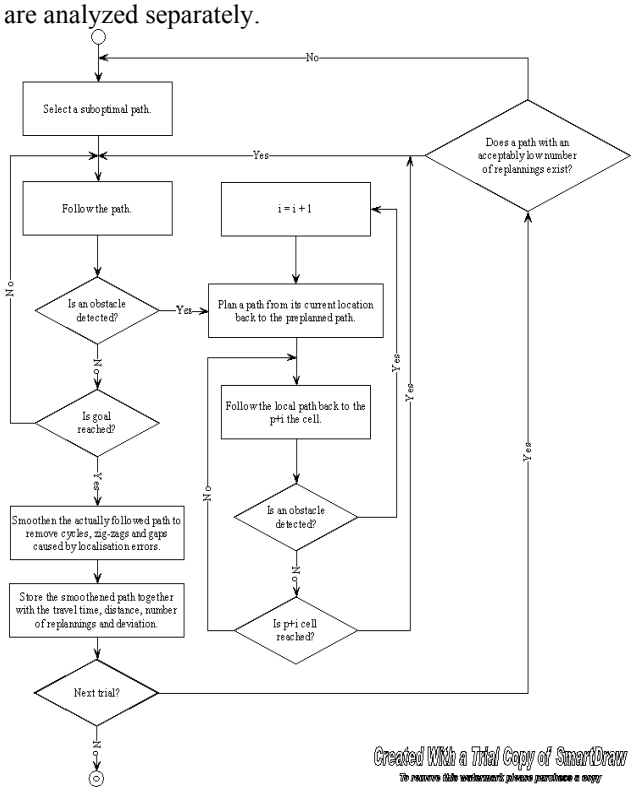


Fig. 16. Path selection algorithm with local replanning

4. Experimental Results

4.1 Shortest path following in a fully modelled static environment

We here represent the detailed results of 10 test runs in a fully modelled static environment. This test evaluates the efficiency and stability of shortest path planning with a wavetransform algorithm and gives a reference for evaluating the performance in partially modelled and unknown environments.

One trial means planning a path from the start to the goal, following the path and replanning if necessary. In the end of every trail four parameters are recorded: number of replannings due to unexpected obstacles, travel time, deviation from the originally planned path and travel distance.

Table 6 represents detailed experimental results. Figure 17 represents the 5th trail recorded by the computer program and Fig. 18 represents the 6th trial where the robot traverses in the opposite direction.

This data indicates that the program is able to control the robot rather precisely and the behaviour of the robot is predictable. The deviation from the original path is not more than by the diameter of the robot and only in one case a detected obstacle or sensor noise forces the robot to replan its course (trial nr. 3).

We can conclude that localisation errors, imprecision of mechanical linkages or the control program do no influence the test results significantly. Keeping all other things equal and changing only the model of the environment or the path planning algorithm we can thus claim that any changes in performance are caused by one of these reasons.

The tests represented in this paper consist of 600 trials. In the following subsections we represent only the average values of all the test runs. The detailed data about every trail as well as snapshots of every followed path similar to Figure 17 and Fig. 18 are available online at http://math.ut.ee/~kristo/khepera/.

No. of trial	No.of replannings	Travel time	Deviation	Distance
1	0	105	29,2	2535,3
2	0	103	58,4	2551,2
3	3	108	29,2	2561,3
4	0	104	58,4	2576,9
5	0	102	29,2	2535,7
6	0	104	58,4	2592,7
7	0	104	29,2	2540,8
8	0	102	58,4	2540,8
9	0	105	29,2	2543
10	0	103	58,4	2573
Average	0,3	104	43,8	2555,07

Table 6. Shortest path following in a fully modelled static environment

Figure 17. Path followed in a fully modelled static environment. Robot moves from the lower-left corner to the upper-right corner

Environment		1	2	3	4	5	6
Shortest path following		10	50	50	50	50	50
Path selection with global replanning	Sub-optimal path	-	1	5	16	14	31
	Learned path	-	9	25	34	36	19
Path selection with local replanning	Sub-optimal path	-	-	-	8	28	17
	Learned path	-	-	-	42	22	33

Table 7. Number of trials

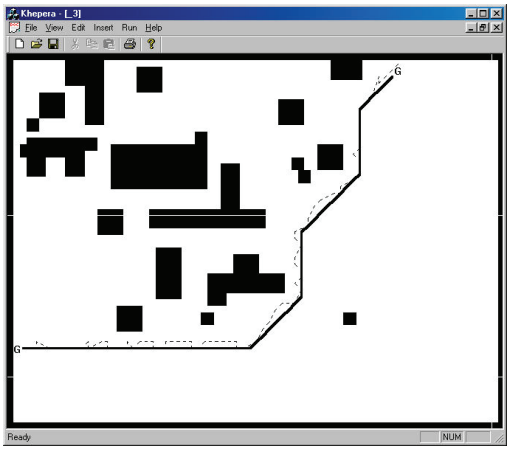


Fig. 18. Path followed in a fully modelled static environment. Robot moves from the upper-right corner to the lower-left corner

The number of trials in every test environment is represented in Tabele 7. Except the trials in a static fully modelled environment where the process was very stable, all other tests to evaluate shortest path following consist of 50 trials.
Path planning with path selection algorithm in unknown environments also consists of 50 trials. As it was described above, the path selection algorithm tries several innovative sub-optimal path and learns to follow these sub-optimal path which are more stable. Learned suboptimal path and innovative sub-optimal paths are therefore analyzed separately and depending on the environment the number of trials differ. However, the total number of trials is still 50 in unknown environments. Since in partially known environments the learning process stabilizes very quickly, the total number of trials is less (10 in environment 4 and 30 in environment 6).

4.2 Number of Replannings
Fig. 19 represents the average number of replannings caused by unexpected obstacles. This is the parameter that the path selection algorithm explicitly minimizes and therefore learning considerably minimizes the risk of collisions. The number of replannings also decreases when the environment becomes better known, which is not a surprising result since the modelled obstacles can now be avoided in advance. The unexpected result is that following suboptimal path gives a better result than following the shortest one. Since the shortest path following algorithm that globally replans always maximizes the expected utility, one would expect that the total outcome of this strategy would be better.

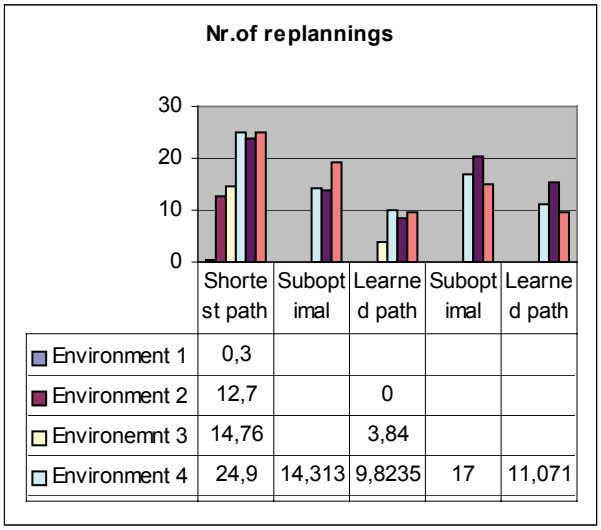
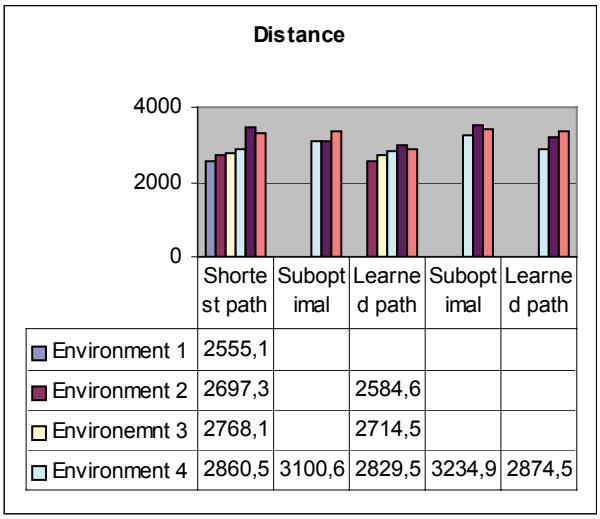


Fig. 19. Average number of replannings caused by unexpected obstacles

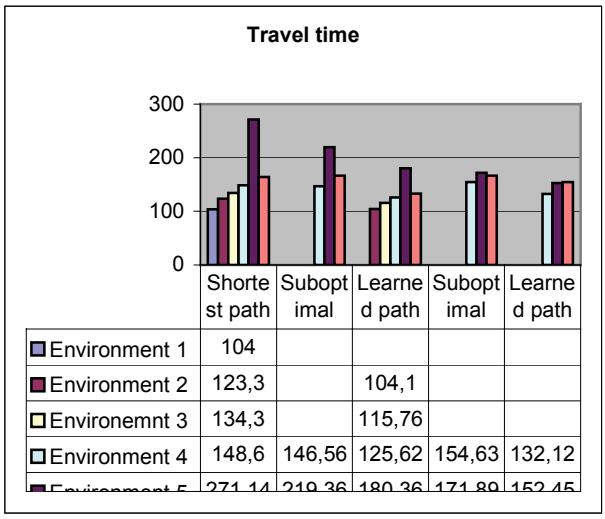
4.3 Travel distance
Fig. 20 shows the average distance travelled at every trial. Although the path selection algorithm prefers longer path to shorter and risky ones, it can be seen that learning slightly decreases the average travelled distance. The most logical explanation is that manoeuvring around obstacles eventually increases the path length more than covering distances between obstacles. Again, like in case of replannings, it can be seen that suboptimal path are not worse than optimal ones.

4.4 Travel time
Fig. **21** shows the average travel time of path following. The general conclusion about this data is the same as about the previous parameters. Learning improves the performance and modelling the environment decreases travel time. Again, following suboptimal path is about as time consuming as following the globally optimal one.
The very long travel time in the environment 1 with the shortest path algorithm seems somewhat anomalous. It is caused by few trials where the robot got trapped in a box canyon (see Fig. 22). When trying to escape the canyon, the robot replanned always it counted an obstacle. Since it replanned globally, every replanning increased travel time. At the same time travel distance and deviation almost did not increase and therefore this anomaly is not visible on other charts.



	Shortest path	Suboptimal	Learned path	Suboptimal	Learned path
Environment 1	2555,1				
Environment 2	2697,3		2584,6		
Environemnt 3	2768,1		2714,5		
Environment 4	2860,5	3100,6	2829,5	3234,9	2874,5

Fig. 20. Average distance



	Shortest path	Suboptimal	Learned path	Suboptimal	Learned path
Environment 1	104				
Environment 2	123,3		104,1		
Environemnt 3	134,3		115,76		
Environment 4	148,6	146,56	125,62	154,63	132,12
Environment 5	271,14	219,36	180,36	171,89	152,45

Fig. 21. Average travel time

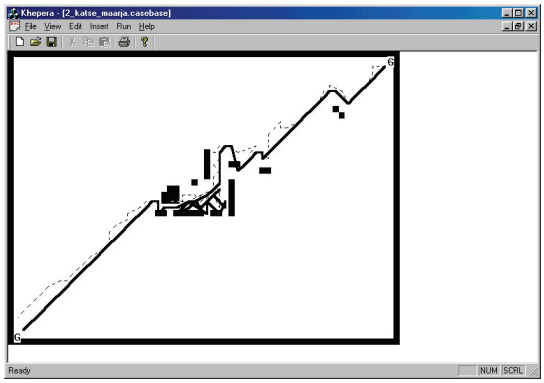
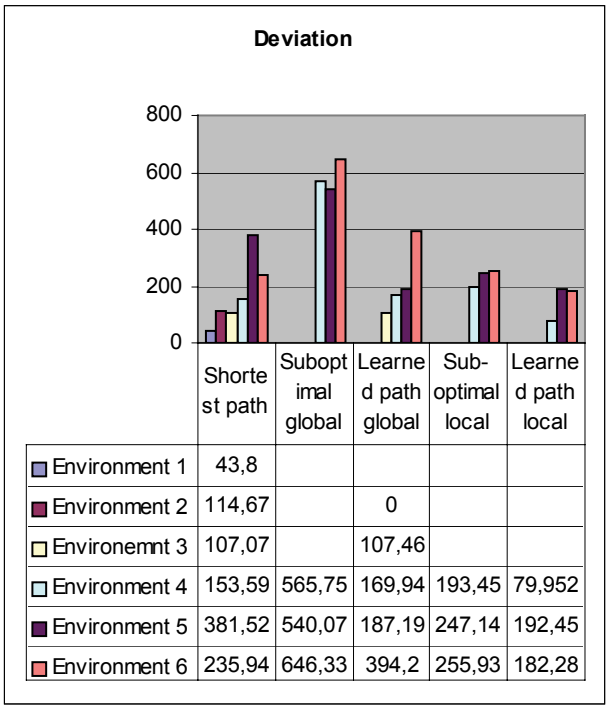


Fig. 22. The robot escaping the box canyon

4.5 Deviation

Fig. 23 shows the average deviation from the original path. Learning usually decreases deviation but it is not true in all cases. The deviation is large when the robot follows a suboptimal path and after counting an obstacle replans globally. This is not surprising since it might have initially driven in another direction and then switches to shortest path following.



	Shortest path	Suboptimal global	Learned path global	Sub-optimal local	Learned path local
Environment 1	43,8				
Environment 2	114,67		0		
Environemnt 3	107,07		107,46		
Environment 4	153,59	565,75	169,94	193,45	79,952
Environment 5	381,52	540,07	187,19	247,14	192,45
Environment 6	235,94	646,33	394,2	255,93	182,28

Fig. 23. Average deviation from the originally planned path

4.6 Learning

The experimental results represented above show that the path selection algorithm outperforms shortest path following and the learned routes are better than the shortest or suboptimal ones in terms of collision risk, travel time, distance and deviation. However, they do not prove that the performance increases because of learning and remembering.

Fig. 24 shows how the performance of the system increases and the robot's behaviour stabilises during the test run. These tests are conducted in the Environment 4, a static unknown environment. Since the environment is static, the learning algorithm but not the environment causes any change in the robot's behaviour. Since the environment is unknown, it proves that the algorithm is able to learn even in the extreme case when very little information is available.

The path selection algorithm with both local and global replanning are verified against shortest path following. Both path selection algorithms stabilize rather quickly as they find and learn to use path that are better than the shortest one. The globally replanning path selection algorithm adapts to use a route with 0 replannings (Fig. 25), while the path selection algorithm with local replanning adapts to use a more risky one (Fig. 26). These results do not prove that globally replanning algorithm is better than the locally replanning algorithm because both of the algorithms look for routes that satisfy their criterion of safety defined by Equation 1. In this example, a route with couple of detected obstacles was considered to be good enough and the robot accepted the result. The globally replanning robot was fortunate to find a route with 0 obstacles and adapted to use this one.

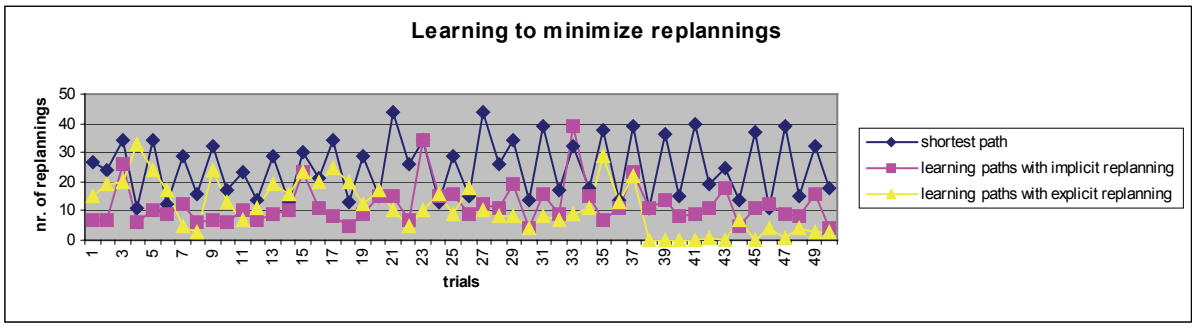


Fig. 24. The learning curve of the path selection algorithm in the environment 4

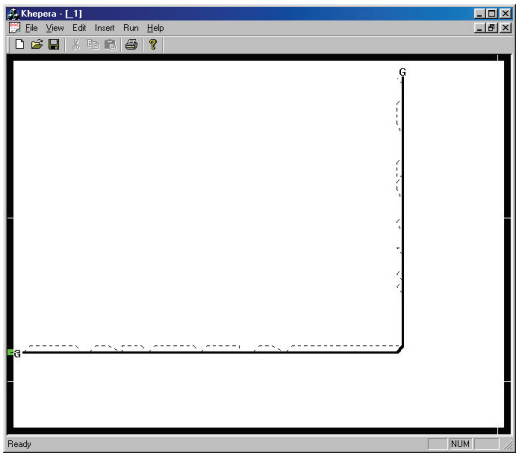


Fig. 25. A learned route with the globally replanning path selection algorithm

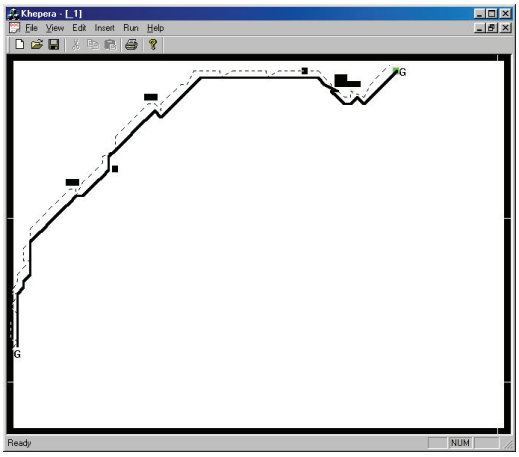


Fig. 26. . A learned route with the locally replanning path selection algorithm

4.7 Correlations

We here represent some more statistical data that helps to explain the experimental results. Tabele 8 represents the correlation coefficients between the number of replannings and other measured parameters. The data is given about the shortest path following in environments from 2 to 6. All these test runs contained 50 trials, which is sufficient for statistical analysis.

It appears that in all cases travel time and the number of replannings is highly correlated. This result is not surprising because the robot replans globally using a wavetransform algorithm. For a reactively replanning robot the correlation could be weaker.

The correlation between the number of replannings and distance is weaker. This can be explained with that the robot does not have all global information available when it replans and therefore it does not always minimize the travel distance.

The correlation between the number of replannings and the deviation from the pre-planned path varies considerably from one test run to another. For example, while in environment 3 the correlation is highly negative; in environment 4 it is highly positive. It means that in some cases larger number of unknown obstacles causes a small deviation while a small number of unknown obstacles can cause a large deviation.

5. Discussion

The most general conclusion about the experimental results is that in a partially modelled large-scale dynamic environment it is possible to optimize robot's behaviour by learning reliable trajectories.

Our initial hypothesis was that the traditional shortest path following strategy would soon outperform the innovative paths learning algorithm when the environment becomes known better. The chronological order of performing the test was in environments 5, 6, 4, 2, 1 and 3. The tests in a partially modelled environment were actually conducted later than in a unknown environment to show the limit where the path selection algorithm breaks down (it is also therefore the test in environments 2 and 3 do not include local replanning strategy because after conducting experiments in environments 4,5 and 6 it was not a research issue any more).

	Number of replannings				
Environment	2	3	4	5	6
Time	0,78	0,98	0,71	0,97	0,96
Distance	0,24	-0,06	0,45	0,63	0,82
Deviation	0,22	-0,67	0,65	-0,06	0,22

Table 8. Correlation coefficients for the shortest path following algorithm.

The test results did not confirm our initial hypothesis. On the contrary, it appeared that even if small obstacles are unmodelled the path selection algorithm improves the performance. The environment 2 depicted in Fig. 11 is a good approximation of an unstructured human inhabited environment with unknown dynamic obstacles (people, vehicles, furniture or even sensor noise). From the experimental data it can be seen that the efficiency of the shortest path planning considerably decreases compared to the completely modelled environment 1.

Another unexpected conclusion is that global replanning does not improve the performance compared to the local replanning. We expected the global replanning strategy to perform better than the local one but the test results did not reveal any significant difference in performance.

Local replanning is performed mostly because sensorial capabilities and computational recourses are not sufficient for real-time global decision-making. Salich and Moreno describe in their paper the dilemma of authority vs. freedom [26]. The dilemma rises from that the global planner produces rigid orders while the local planner acts reactively, therefore the results are not globally optimal.

The test results reported here suggest that the dilemma of authority vs. freedom is irrelevant in partially unmodelled environments. The efficiency of path planning rather depends on the world model than of the path planning strategy. A global planner that does not have all global information available anyway fails to make a globally optimal plan and therefore the locally replanning agent performs equally well. From the test data it is obvious that global replanning does not give any particular advantage in terms of time, distance or collision risk. Moreover, local replanning should be preferred if the aim is to keep the robot close to its initially planned trajectory.

The next conclusion about the test data is that suboptimal path planning gives at least as good results as the optimal one. Intuitively one would expect an opposite conclusion. Speaking in decision theoretic terms, the globally replanning shortest path planner behaves as a utility maximizing rational agent. At every occasion it makes a decision that is best considering all global knowledge available. Therefore one would expect the cumulative utility to be higher. The suboptimal path planner (either globally or locally replanning) can be described as an explorative agent. It sometimes makes suboptimal decisions to explore the solution space and escape the local optimum. A reasonable explanation is that manoeuvring around obstacles takes significantly more recourses than traveling the distance between them.

The closer analysis of the test data reveals that among all the measured parameters, the robot's trajectory is the one that is most difficult to predict and control. It is difficult to foresee the extent of the deviation from the original track caused by an unknown object. Travel time and distance have a higher correlation to the number of replannings. Any changes in time, distance and deviation are caused by unexpected obstacles. We therefore can conclude that in order to optimize any other parameter (like time, distance, energy consumption, deviation) we have to minimize the number of unexpected events.

The general conclusion about the test data is that robot's behaviour is much more dependent on the environmental model than the path planning strategy. In order to optimize the behaviour of the robot, one should gain better knowledge about the environment. This can be achieved in two ways. The first option is to model the environment as closely as possible. The alternative way, reported in this paper, is to look for routes where uncertainty does not significantly influence the result.

6. Limits and further questions

Obviously these results can not be interpreted as applying to all possible environments. They are limited to one randomly generated environment. If this environment is a good approximation of an unstructured, large, uncertain and dynamic working environment of mobile robots, it is likely that the results are also true in other cases. Repeating these tests in such an environment would give a confidence that the conclusions hold true in a large variety of scenarios. The problem rises from that a large, dynamic and uncertain environment is practically not controllable. The model environment represented here is on the contrary, fully controllable, and therefore we can be certain about the causes of the results.

These conclusions certainly can not be generalised to topological maps and graph-based path planning since the models and methods considerably differ from those used for grid-maps.

At the same time we suggest that the results can be generalised to other path planning methods than distance transform on a grid-map. The performance of the shortest path planner and suboptimal path planner are not different. Therefore it is likely that the performance of the wavetransform planning or any other planning algorithm do not influence the performance either. Actually, any path planned on a gridmap is suboptimal because the grid is digitized. It is possible that methods that use a varying resolution like [7] behave differently.

The approach of learning innovative tracks has two severe limitations. First, it assumes that localisation errors are small and do not accumulate. It is hard to predict how the localisation errors influence the result of path planning. Second, it can be applied only to missions where the robot repeatedly traverses between predefined target points. This assumption makes it possible to try several suboptimal routes and learn to use the most reliable ones.

7. Conclusions

This paper examined path planning strategies in large partially unknown environments. The robot learned innovative routes to find reliable trajectories and optimize robot's behaviour. The approach was verified to shortest path following in 6 different environments.

Experimental results lead to the following conclusions:

- The approach of learning and remembering reliable routes increases the performance of the robot.

- The behaviour of the robot is influenced by the knowledge it has about the environment but does not depend on the path planning strategy.
- To optimize travel time, distance, energy consumption, collision risk or deviation from the original path, the probability of unexpected events should be decreased as changes in the former parameters depend on the last one.
- Robot's trajectory in an uncertain environment is very difficult to predict and control because the deviation from the planned path is weakly correlated to the certainty of the model.

8. References

[1]O.Khatib, Real-time obstacles avoidance for manipulators and mobile robots. IEEE Int. Conf. On Robotics and Automation, March 1985, 500-505.

[2]B. Barraquand, J. Langlois, J.-C. Latombe. Numerical potential field technique for robot path planning. Tech. Rep. Dept. of Computer Science, Report No. STAN-Cs-89-1285, Stanford University, 1989.

[3]R.Jarvis and K.Kang, A new approach to robot collision-free path planning. *Robots in Australia's Future Conference,* pp.71-79, 1986.

[4]V.J.Lumelsky, "A Comparative study of the path length performance of the maze-searching and robot motion planning algorithms.", IEEE Transactions on Robotics and Automation, Vol7, No. 1, February, 1991, pp. 57 – 66

[5]A.Stentz, The Focussed D* algorithm for real-time replanning, Proc. of the Int. joint. Conf. Of Artificial Intelligence (IJCAI-95), August 1995.

[6]F.Xu, H van Brussel, M.Nuttin, R.Moreas "Concepts of Dynamic Obstacle Avodance and their extended application in underground navigation", Robotics and Autonomous Systems 42, 2003 pp.1-15

[7]A.Yahja, S.Singh, A.Stentz, "An Efficient on-line Path Planner for Outdoor Mobile Robots". *Robotics and Autonomous Systems*, 32, pp. 129-143, Elsevier Science, 2000.

[8]H.Seraji, "New Traversability Indices and Traversability Grid for Integrated Sensor/Map-Based Navigation", *Journal of Robotic Systems*, Vol. 20, No.3, pp. 121-134, Wiley periodicals, 2003.

[9]Robin R. Murphy, Ken Hughes, Alisa Marzilli and Eva Noll, "Integrating explicit path planning with reactive control of mobile robots using Trulla", *Robotics and Autonomous Systems,* Vol. 27, Issue 4, pp. 225-245, Elsevier Science, 1997.

[10]S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic algorithms and the interactive museum tour-guide robot minerva. *, GInternational Journal of Robotics Research (IJRR)*, 19 (11), 2000.

[11]Tomatis, N., Terrien, G., Piguet, R., Burnier, D., Bouabdallah, S. and Siegwart, R. Design and System Integration for the Expo.02 Robot. Workshop on Robots in Exhibitions, IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2002 , Lausanne, Switzerland, September 30 - October 4.

[12]Ulrich Nehmzow Quantitative analysis of robot–environment interaction—towards "scientific mobile robotics", *Robotics and Autonomous Systems, Volume 44, Issue 1, 31 July 2003, 55-68*

[13]http://www.k-eam.com/robots/khepera/index.html

[14]A. Zelinsky. "Using Path Transforms to Guide the Search for Findpath in 2D. *The Int. Journal of Robotics Research,* Vol. 3 No. 4, pp. 315-325, August, 1994.

[15]A.Elfes. Using occupancy grids for mobile robot perception and navigation, IEEE Computer, 22(6), pp.46.57, 1989.

[16]Howard, H.Seraji, "Vision-Based Terrain Characterization and Traversability Assessment", *Journal of Robotic Systems*, Vol. 18, No.10, pp. 577-587, Wiley periodicals, 2001.

[17]D.B.Gennery, "Traversability Analysis and Path Planning for Planetary Rovers", *Autonomous Robots*, Vol. 6. pp. 131-146, Kluwer, Academic Publishers, 1999.

[18]U.Nehmzow, C.Owen, "Robot Navigation in the Real World: Experiments with Manchester's Forty Two in Unmodified Large Environments", *Robotics and Autonomous Systems,* 33, pp.223-242, Elsevier Science, 2000.

[19]E.Kruse, F.M.Wahl. Camera-based Observation of Obstacle Motions to Derive Statistical Data for Mobile Robot Motion Planning. *Proc. Of IEEE Conference of Robotics and Automation,* Vol.1, pp.662-667, 1998.

[20]E.Lemaster, S.Rock, A Local-Area GPS Pseudolite-Based Navigation System for Mars Rovers, Autonomous Robots, 14, pp. 209-224, 2003.

[21]M.Kruusmaa, J.Willemson. "Algorithmic Generation of path Fragment Covers for Mobile Robot Path Planning", *Technical Report*, 2003.

[22]M. Kruusmaa, J.Willemson, K.Heero. "Path Selection for Mobile Robots in Dynamic Environments", *Proc. of the 1st European Conference on Mobile Robots,* Poland 2003, pp. 113 - 118.

[23]K.Heero, J.Willemson, A.Aabloo, M.Kruusmaa. Robots Find a Better Way: A Learning Method for Mobile Robot Navigation in Partially Unknown Environments, submitted to IAS-8

[24]M. Kruusmaa. "Repeated Path Planning for Mobile Robots in Uncertain Environments", Proc. of the IASTED Int. Conf. Robotics and Automation, 2001, pp. 226- 231.

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

[26]M.A. Salichs and L. Moreno. "Navigation of Mobile Robots: Open Questions", *Robotica,* Vol. 18, pp. 227-234, Cambridge University Press, 2000. 6