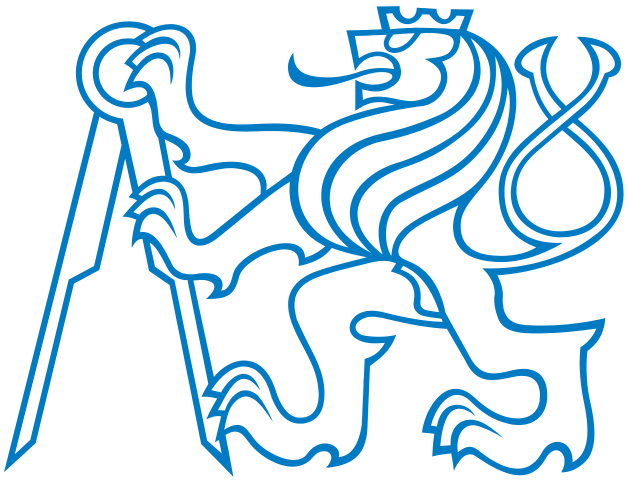
CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF ELECTRICAL ENGINEERING

BACHELOR THESIS



Jan Krys

Exploration Strategies for Mobile Robots

Department of Cybernetics
Supervisor: **Ing. Jan Faigl, Ph.D.**

Prague, 2014

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne .....

.....
Podpis autora práce

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student: Jan K r y s

Study programme: Cybernetics and Robotics

Specialisation: Robotics

Title of Bachelor Project: Exploration Strategies for Mobile Robots

Guidelines:

- Familiarize yourself with the problem of mobile robot exploration of unknown environments.
- Implement and compare performance of the exploration strategies [1, 2, 3, 4] using a given exploration framework and simulation environment Player/Stage.
- Discussed used methods for evaluation of performance of the exploration strategies and influence of components of the whole robotic system, in particular robot motion control.
- Consider an informative planning in the exploration task to collect relevant data from a partially known environment.

Bibliography/Sources:

[1] B. Yamauchi: A frontier-based approach for autonomous exploration. In Proc. of IEEE Int. Symposium on Computational Intelligence in Robotics and Automation. IEEE Comput. Soc. Press, 146–151, 1997.
[2] S. Garrido, L. Moreno, and D. Blanco: Exploration of 2D and 3D Environments using Voronoi Transform and Fast Marching Method. In J. Intell. Robotics Syst. 55:1, 55-80, 2009.
[3] A. Bautin, O. Simonin, and F. Charpillet: Minpos: A novel frontier allocation algorithm for multi-robot exploration. In ICIRA (2), pages 496–508, 2012.
[4] M. Kulich, J. Faigl, and L. Preucil: On distance utility in the exploration task. ICRA, pages 4455-4460, 2011.

Bachelor Project Supervisor: Ing. Jan Faigl, Ph.D.

Valid until: the end of the winter semester of academic year 2014/2015

L.S.

doc. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, December 1, 2014

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Jan K r y s

Studijní program: Kybernetika a robotika (bakalářský)

Obor: Robotika

Název tématu: Strategie průzkumu neznámého prostředí mobilním robotem

Pokyny pro vypracování:

- Seznamte se s úlohou průzkumu neznámého prostředí mobilním robotem.
- Implementujte a porovnejte strategie průzkumu [1, 2, 3, 4] v dodaném rámci pro robotickou exploraci a v prostředí robotického simulátor Player/Stage.
- Implementované metody porovnejte z hlediska potřebného reálného času na průzkum prostředí.
- Diskutujte používané metody měření kvality strategie průzkumu a vliv dílčích komponent robotického systému, zejména řízení pohybu robotu.
- Uvažujte rozšíření úlohy průzkumu o tzv. informatické plánování, ve kterém je cílem získat relevantní informace v částečně známém prostředí.

Seznam odborné literatury:

[1] B. Yamauchi: A frontier-based approach for autonomous exploration. In Proc. of IEEE Int. Symposium on Computational Intelligence in Robotics and Automation. IEEE Comput. Soc. Press, 146–151, 1997.
[2] S. Garrido, L. Moreno, and D. Blanco: Exploration of 2D and 3D Environments using Voronoi Transform and Fast Marching Method. In J. Intell. Robotics Syst. 55:1, 55-80, 2009.
[3] A. Bautin, O. Simonin, and F. Charpillet: Minpos: A novel frontier allocation algorithm for multi-robot exploration. In ICIRA (2), pages 496–508, 2012.
[4] M. Kulich, J. Faigl, and L. Preucil: On distance utility in the exploration task. ICRA, pages 4455-4460, 2011.

Vedoucí bakalářské práce: Ing. Jan Faigl, Ph.D.

Platnost zadání: do konce zimního semestru 2014/2015

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 1. 12. 2014

Abstrakt

Tato práce se zabývá problémem explorace prostředí mobilním autonomním robotem. V práci jsou představeny tři metody pro jednoho robota a jedna pro skupinu robotů. Cílem práce je porovnat tyto metody z hlediska času potřebného na prozkoumání celého prostředí. Hlavním problémem je zde najít nejvhodnější cíl kam se má robot v daný moment vydat. Tři z těchto metod patří do skupiny *frontier-based* přístupů. Zde je definována skupina kandidátů na navigační cíle robotu, které jsou na pomezí prozkoumaného a neprozkoumaného prostředí. V každé metodě je však vhodná cílová pozice vybírána jinak. První přístup vybírá nejbližší, druhý se snaží najít nejkratší cestu přes všechny tyto potencionální cíle pomocí řešení úlohy obchodního cestujícího a třetí metoda jpřiřazuje roboty k těmto cílům. V posledním přístup je navigace řešena pomocí propagace vlnoplochy v nehomogenním prostředí. Porovnání je provedeno simulacemi pro různé parametry a explorační scénáře.

Abstract

The thesis addresses the problem of the mobile robot exploration of unknown environment. There are presented three approaches for a single robot exploration and one for multi-robot exploration mission. The goal of the thesis is to compare these approaches regarding the time needed to explore the whole environment. The main problem is to find the most suitable goal location where the robot should navigate in order to explore the environment efficiently. Three of these methods belongs to the class of the so called *frontier-based* methods. In these methods, a set of candidate locations is defined to be at the the border of the already known and not yet explored parts of the environment; however, in each approach, the most suitable goal location is selected differently. The first method selects the nearest candidate, while the second method finds the shortest path to visit all the candidates as a solution of the traveling salesman problem. The last frontier-based method is designed to the multi-robot exploration and it assigns robots to the goal candidates. In the last approach, the selection of the goal candidate and path planning is solved using a wave front propagation in an inhomogeneous environment. The comparison is performed simulations for various parameters settings and selected exploration scenarios.

Děkuji vedoucímu své práce, Ing. Janu Faiglovi, Ph.D., za množství času, ochotu a množství dobrých rad během tvoření této práce. Dále bych chtěl poděkovat svým rodičům, nejbližším a přátelům za podporu, kterou mi během celého studia poskytovali.

Contents

List of Figures

Introduction

Mobile robotics is a very fast developing field these days, due its wide use. We use mobile robots in many places. Due to their durability, we can use them in environments, where human could not survive. For example we can use them in space, at areas of nuclear disasters, at burning buildings. But we can also use them as a cheaper labor in factories or as a messenger in an office. And nowadays, they are also becoming part of our normal life as vacuum cleaners.

The first robots were radio controlled by an operator; so, they did not need any kind of artificial intelligence or sensors. It is good an idea, because they just do what you want and it is also not so hard to create it. But radio controlled robots can't be used everywhere and you need someone who will control it. For example you can't control robot on Mars due to long delay. In these situations you need robots with a certain level of autonomy. But this is not that easy task to do. You need to give a robot some knowledge about environment (map), where it will be operating, but then robot will need to know where it is on the map.

However at some situations you do not have any a priory knowledge about the robot's operating environment. In these situations you need to install some additional sensor to gather information about environment around the robot. With this information, robot should be able to navigate safely through the environment. But there are more ways how to do it. Firstly your robot could follow a line on the ground, which will lead him towards his goal. Secondly your robot could have reactive algorithm, where it considers only current scan. But these ways may not be enough for some applications, for example if you need to create a map for later use.

Here, the robotic exploration comes. It is a problem, where a mobile robot goes through an unknown environment and creates a map of it. This is a complex task, where the robot needs to track its position, integrate sensor readings into a map and determine a goal location and last but not least, it is also needed to find route to goal and follow it. In this thesis, we assume that the robot is able to localize itself and thus the localization problem of the mobile robot navigation is considered to be solved. Although we consider this assumption, there is still a lot of work on other aspects of the exploration. Integration of sensor readings is a quite simple task, but determination of the next robot goal location and planning a route to it are a more difficult tasks. There are many different approaches how to do it. There are differences in when to determine a new goal, but also where to the goal should be located.

In most cases, we want to explore the given environment in the shortest time possible, but we also need to do it safely. In this thesis, we aim to compare exploration strategies by the time needed to explore whole environment, and by the length of the exploration path. We also aim to compare different settings and environments and find out which have the most influence.

Chapter 1

Problem Definition

If we want to explore an environment with a mobile robot, we need to solve the robot's navigation. The navigation consists of the localization, path planning and motion control, but it is also related to the mechanical parts of the mobile robot such as dynamic and kinematic properties of our robot. In this thesis, we focus on solving the navigational part of problem where we need to find the best goal to accomplish the mission, i.e., to explore the environment as quickly as possible.

In this thesis, we use robot that is able to move on a flat surfaces (floor) and has a differential motion control. We also assume it is equipped with laser range finder, which is omnidirectional and has maximum range $\rho = 10m$.

The algorithmic part contains the following parts. First, we need to choose how to represent the environment. There are various models, which are briefly described in Section 1.3. Once we have environment model, we have to read data from sensors that are integrated into map, which is described in Section 1.2. Then we have to solve the localization problem. There are many ways to do it which will be described in next section. Last but not least, we have to navigate our robot towards computed goal location, which is described in Section 1.4.

1.1 Localization

There are many ways to localize our robot. Firstly we can use odometry, which measures the travelled distance by counting how many times robot's wheels are turned. But this may not be exact, because if we use wheels they can slip and our measurement is ruined. Secondly we can use some positioning system such as GPS, but this also may not be exact, because signals may not be available everywhere and it does not work in indoor.

We can use some external observer or additional reference localization system. But we must ensure that it can see the robot at every place, which is not always possible. We can also use some pivot points and our robot can navigate by them. But we do not always have them. So we can create them while we are creating map. Furthermore we can use two robots to localize each other, but to ensure correct localization, they can be moving only one at the time. And there are many other ways to do it.

In our experiments, we are considering that the robot is localized sufficiently precise.

1.2 Sensors

Sensors are robot's eyes and ears. There are many types of sensors, but most useful for our problem are distance sensors. We use laser a distance sensor, which is fast and accurate, but it only measures distance in one plane; so, when there is something, which is under or above this plane we can't see it with this sensor. So, we can consider an ultrasonic sensor, which generates an ultrasonic wave with a conical shape sound that is able to sense in that cone and we can use it to prevent bumping into obstacles that are not visible to the laser sensor.

All these sensors are suffering by noise. There is natural noise, but when we use multi-robot exploration we also have to count with eventual interference between them. Because of this we use a model of this sensor, which considers probability that the data we measured are not real.

The robot in our experiments has one laser range finder situated at the center of the robot body and have omnidirectional field of view. The sensing range of the sensor is limited and it is one of the parameter studied in the experimental evaluation of the methods.

1.3 Environment model

Basic models of environment are: occupancy grid, geometrical map, topological map and symbolic map. Every model has its pros and also its cons.

A geometrical map represents environment using geometrical entities. Type of representation is chosen with regard to the computational complexity of finding the relative positions of two geometric entities. The environment is approximated by segments or by curves of the second order, which is more accurate, but also a more complex to work with.

A topological map is defined by states and transitions between these states. These maps lack scale, which means distance and direction can change, but the relationship between states is maintained.

A symbolic map can be considered as an extension of the topological map. This map consists of objects and relations between them. The objects are names of places in environment and relations define if something is in something else or how far away the objects are.

The last type of the environment model is occupancy grid which is used in all strategies we use in this thesis. The occupancy grid consists of a set of cells that are arranged into a grid. Each cell contains probability value indicating it is an obstacle. Because of it, it is very easy to create and manipulate with it. But if we have a large space to explore or if we want a very accurate map it can be memory-intensive. With today's level of computer technology it is not a significant issue, and thus occupancy grids are widely used. New scans are included into map using Bayes' theorem.

1.4 Navigation

By navigation we mean planing a safe route towards the particular goal location. We can divide the navigation problem into two separate parts:

1. local navigation
2. global navigation

The global navigation works with the given map and tries to find a safe and fast route to reach the given goal location. For this purpose, there are many solutions that can be used. In this thesis we use four of them. They are described in Chapter 2. Regardless of the particular global navigation method, all of them share the same problem that is to determine a path from the current robot location to the desired goal location. However the environment may not be static or some measurement may not be exact. Thus, the precomputed path at the global level may not be valid when a new information about the environment is available. Therefore, we have to compute the route very often, but this may be computationally demanding because of considering a global (large) map of the environment.

An alternative, and probably more suitable, method is to use a local navigation. The local navigation uses current sensor measurement to prevent collisions of the robot with an obstacle. In a case a new obstacle is detected it can either try to navigate the robot around the obstacle and then back to planned path. Or it can stop robot and call global navigation to recompute the path with the updated map that includes this new obstacle.

Chapter 2

Robotic Exploration and Strategies

In robotic exploration, the robot is requested to create model (map) of unknown environment. We want to do it according to the specified rules. Based on the considered criteria robot needs to select the most suitable goal location where to navigate. In this thesis, the main criterion is the time needed to explore the whole environment.

Exploration of unknown environment effectively is a complex task, because it is unknown what is the best navigational goal according to the global optimization criterion that can be evaluated only after the mission is completed. It is because we do not have initial knowledge about the environment and thus we cannot plan the robot moves beforehand. So, we have to plan the robot moves in the field and thus the we need to perform on-line , in situ, decision making based only the information about the environment acquired so far. We have to use some methods to plan our path as close as possible to perfect path.

Each method in this thesis is an iterative procedure, where we can identify the following important steps:

1. Division of current map into some section e.g. explored and unexplored.
2. Check if there are any unexplored reachable places.
3. Determination of possible goal candidates (e.g. frontiers described in Section 2.1.1).
4. Selection of the next robot goal from the set of goal candidates.
5. Navigation towards selected goal while the robot constantly collects new data from sensors and integrates them into the current map of environment.
6. When the robot reaches the goal, or there is some interruption the next iteration is started.

2.0.1 Occupancy grid

In all methods, in this thesis, the occupancy grid is used. To create it we have to divide the environment into a grid of squared cells of desired size. Each cell is assigned a value of probability that there is an obstacle. At the beginning, all cells have the same value of the probability being occupied set to 0.5. For every scan we create group of cells where we

did not found an obstacle and group of cells where we found something. Then we update value of these cells accordingly to this formula.

$$P = \frac{PM}{2PM - M - P + 1}, \tag{2.1}$$

Where M is sensor model and P is the probability of the cell is an obstacle. Sensor model is used because there are some measurement uncertainties; so, we cannot add 0%/100% but something about 40%/90% depending on the particular sensor. So, we need more than one scan to be sure what is there as we can see in Figure 2.1, where white denotes free space, black represent detected obstacles and gray denotes unknown, but more white means higher probability of free space.

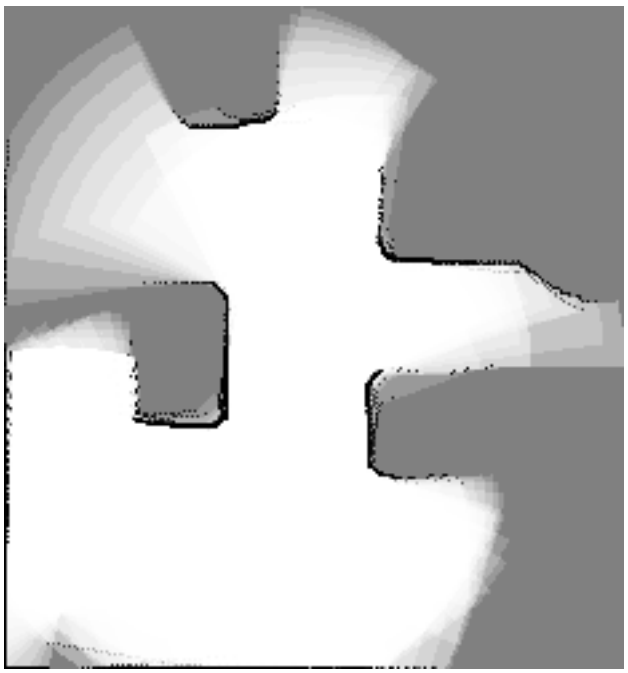


Figure 2.1: Example of occupancy grid.

Since the calculation of paths do not use the physical dimensions of the robot, we need to ensure the robot will not collide with obstacles. This can be done by inflating obstacles. It means that for the following calculations we enlarge all obstacle cells by a radius disc the shaped robot.

2.1 Frontier-based "greedy"

The greedy selection of the next robot goal is the fundamental approach. It was introduced in [10].

2.1.1 Frontiers

When we have filled our occupancy grid with newest data we have to determine goal where to navigate the robot next. To do so we need to simplify the grid. It is not necessary

to know the exact value of probability a cell is an obstacle. We only need to distinguish if the cell is freespace, occupied or unexplored. To do this we need to set particular thresholds of the probability. In this thesis, we set the threshold as follows: if the probability is lower than 30% the cell is considered as freespace, if it is higher than 70% it is considered as obstacle and unexplored area otherwise.

To gain some new information the robot has to navigate towards unexplored area, but it also has to go only through the explored freespace to be sure that it doesn't crash into some not yet explored obstacles. So, the robot has to navigate to the border between the detected freespace cells and unknown cells. If a map is large we may end with lots of these cells; so, we have to cluster them into sets. These sets are represented by one point, which Yamauchi named as $frontier$ in his work [10].

2.1.2 Navigation

When we have $frontiers$ we need to select the right one where we want to go. There are many different approaches to do so. We can select the one, which is composed of the highest number of cells; so, there may be most new information. We can also select the nearest one as is done in this approach. Or we can create the shortest route through all of these which is explained in Section 2.2.

The selection if the nearest $frontier$ is the easiest and the most obvious thing to do. But still we need to decide, which frontier is the nearest one. The robot can past particular places at different speed than through others, e.g., passing large freespace area vs navigation through a narrow passage This is considered in the Fast Marching approach described in Section 2.3. For this greedy Yamauchi's approach this aspect is not taken into account.

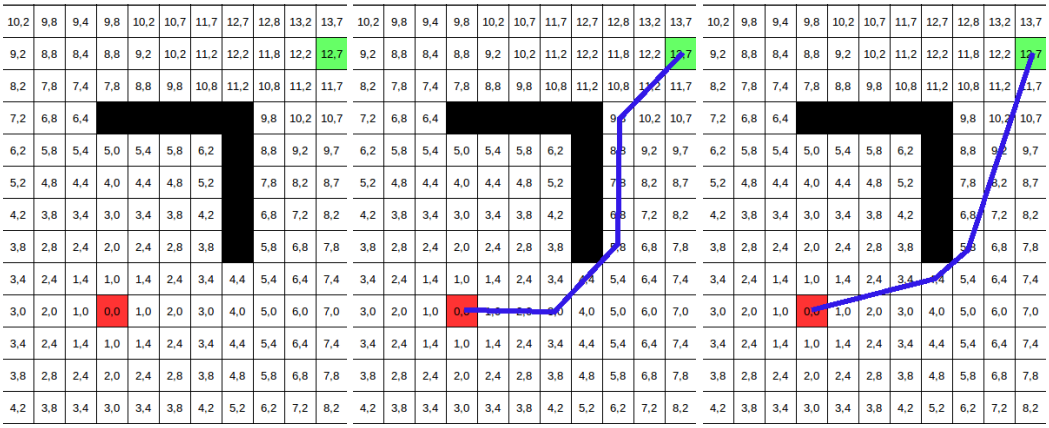
However, we must reckon with the fact that there can be obstacles and thus using a pure Euclidean distance is not sufficient and we need to find paths among obstacles to provide a collision free navigation. One of the approaches can be based on a conversion of the map into a graph where each cell is a vertex that is connected with the adjacent cells (vertices). Then, we can use a general path planning algorithm that operates on graphs , e.g., A*, Breadth-first and Depth-first search.

Although the general graph-based search techniques can be utilized, we rather use a different techniques called Distance Transform (DT) [8]. The DT algorithm is a variant of front wave propagation planning approach (e.g., similar to an artificial potential field techniques) and it fills all freespace cells of map with their distance from the selected location (i.e., robot position in the map) as shown in Figure 2.2(a). In our case, the location is the robot position that is shown as the red cell. Once the distance map is computed, it is easy to find the nearest $frontier$ which is shown as the green cell.

After selecting the next robot goal, we need to plan a path for the robot to reach the goal location. To achieve this, we start from the $frontier$ and we are looking for a neighboring cell with the lowest value of its distance to the robot location. Although this approach provides a feasible path for a robot, such a path may consists of unnecessary turns because it is computed using only in eight possible directions from each cell as it is shown in Figure 2.2(b).

So, we can perform a simple smoothing algorithm, which provides a shorter path with less turns as it is shown in Figure 2.2(c). This algorithm goes from one end of path to other. When it reaches the cell, from which the starting cell is not directly visible, it comes back

to previous cell and deletes all cell between starting cell and this one. Then, it starts from that cell and continues to the last one in the plan.



(a) Freespace cells filled with their distance from the selected location

(b) Route computed only by following minimum from eight neighbors

(c) Route smoothed by smoothing algorithm

Figure 2.2: Distance Transform

Now, we have selected next goal location and computed path to reach the goal and we can finally ask our robot to follow the path. The final part of the decision-making strategy is to decide when we should recompute new goal. This is discussed in Chapter 3.

2.2 Traveling salesman problem

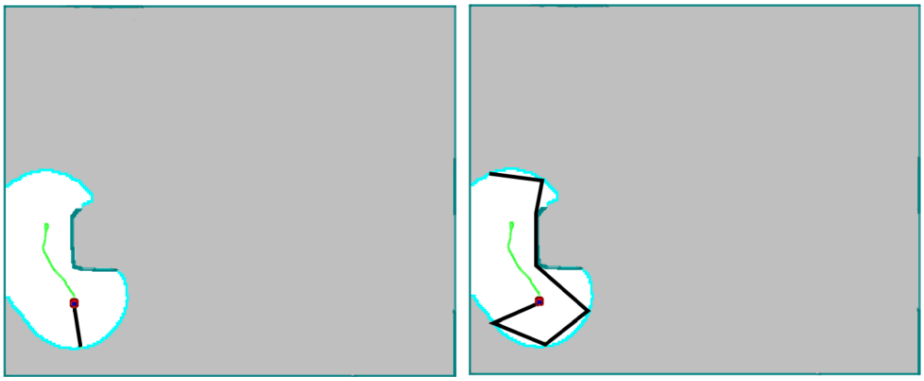
The previous approach computes the cost to frontier cell as the distance to it and selects the closest frontier cell to the robot. But this can lead to a situation when the robot is navigated to new unexplored parts of the environments while there are some frontiers that have to be visited later, which can be seen in Figure 2.3(a). This can lead to significant (and probably unnecessary) prolongation of the exploration time.

To avoid this we can to use a more informed approach. One such approach has been introduced in [7]. This approach does not compute the cost only by distance to frontier cell, but as a length of the path to visit all frontiers cells. This path can look like the path showed in Figure 2.3(b). This problem can be interpreted as the Traveling Salesman Problem, which have been first studied by mathematicians starting in the 1930s [9].

The TSP stands to find a shortest Hamilton cycle in the weighted graph. It is NP-hard problem, and therefore, approximation algorithms can be used. The TSP is formulated to find the best closed tour, while for the problem of selecting the next goal to navigate the robot in the exploration mission we rather need an open path. This can be addressed by transforming the problem into an extended graph adding a fictive vertex s_∞ to a set of vertices V, where $d(s_\infty, s_0) = 0$ and $\forall i \in \{1, n\} : d(s_\infty, s_i) = \omega$ and ω is a sufficiently large number [7]. This will ensure that s_∞ and s_0 are neighbors in the found tour visiting all vertices V of the graph.

In general, the TSP approaches work on graphs, and therefore, we need to convert the grid based representation of the environment into a graph. This can be done as follows. We select goal candidates using Representatives of Free Edges (RFE) introduced in [3]

Then, we take these cells as vertices of our graph and each vertex is connected with all other vertices. Weights of the edges are computed as the lengths of the paths between the particular vertices (cells) using the Distance Transform [8].



(a) Greedy approach (b) Traveling Salesman Problem approach

Figure 2.3: Difference between greedy and TSP approach. Taken from [7]

2.3 Voronoi Transform and Fast Marching Method

This approach, introduced in [5], is trying to make the navigational paths smooth and also as far from obstacles as possible. This should ensure the fastest motion of the robot along the path, because it is not necessary to slow down the robot due to proximity of obstacles. Moreover, such paths do not contain sharp turns, where the robot needs to slow down. This is achieved by applying the Extended Voronoi Transform (EVT) onto a map of the environment, which computes speedmap for Fast Marching method (FM).

2.3.1 Extended Voronoi Transform

EVT converts a binary image of visible environment into a gray scale image. This image is darker near the obstacles as it is shown in Figure 2.4. Each point in this image corresponds to one cell in the occupancy grid. But values in cells do not represent probability of cell being obstacle, but its distance to the nearest obstacle. The algorithm imitates repulsive electric potential, which is exerted by the obstacles and the robot. It means the robot tries to stay as far away from obstacles as possible and selects a safe trajectory.

Considering this principle provides a smooth trajectory; however, simple following the gradient of the distance may lead the robot to places of local minima. Therefore, it is necessary to address this issue and to avoid getting stuck at these locations.

2.3.2 Fast Marching Planning Method

Fast Marching Method is based on propagation of front wave in a heterogeneous environment. This is similar to the Distance Transform, but it allows us to set different speed in a different parts of environment.

The Fast Marching method alone does not guarantee smooth trajectory, because it determines only a shortest geometrical path. This path is not only choppy but it hugs corners

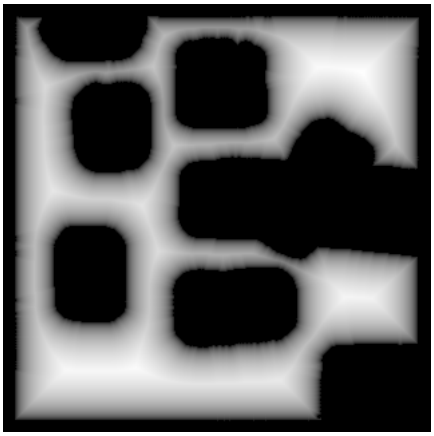


Figure 2.4: Example of Extended Voronoi Transfer.

very closely; thus, the robot can collide with that obstacles. This can be avoided by enlarging obstacles, but the trajectory would still not be smooth. To ensure smoothness and safety of the trajectory we need to combine the Fast Marching with the Extended Voronoi Transform.

This is done by decreasing speed of front wave near obstacles according to the repulsive electric potential. Then the trajectory tends to be close to Voronoi diagram but thanks to the propagation of the front wave it will not be stucked in places of local minima.

2.3.3 Finding the Next Navigational Goal

The important part of the exploration strategy is a selection of the next navigational goal in order to explored the whole environment as quickly as possible. We can use frontier based approach similarly to the greedy or the TSP-based approach, but authors of [5] proposed a different method based on the direct utilization of the Fast Marching method. The approach is based on trying to navigate the robot into the most unexplored area, which is achieved by creating an attractive potential emitted by the unexplored areas.

So, we create two new maps. One similar to the one, which we use in other strategies, but unknown cells are considered as freespace and expected the border of map is considered as the obstacles. In the second map are unknown cells considered as freespace and all other cells are considered as obstacles. Then, EVT is applied and matrices W and VT are obtained. The final matrix (map) WV is computed as

$$WV = VT \cdot 0.5 + W. \tag{2.2}$$

Then, the next goal is picked as the cell with the maximal value in the matrix WV. This cell should be the most unexplored place; so, the knowledge gain should be maximal. Such a new goal is computed when robot reaches it (path to it is short) or when this goal becomes unreachable.

2.4 MinPos

The methods described in the previous sections use only a single robot for the exploration. To decrease the required time needed for exploration of the whole environment more robots can be utilized. This will not only decrease the time, but it may also provide a more robust solution. When one robot stops working, the other robots can continue the mission. But if we use n robots, we cannot except that the time will be decreased n-times . This is caused by sharing the common working space and also the efficiency of the coordination of several robots.

This lead to some problems that have to be addressed. First, we need to avoid detecting other robots as obstacles. We also need to distribute targets across all robots to avoid more robots going to the same target. One of possible approaches to deal with multi-robot exploration has been introduced in [1].

In [1], authors proposed to assign the frontiers to particular robots in a decentralized way. This means that each robot decides where to go autonomously. Each robot knows where other robots are located and robots also share location of all current frontiers. Then, each robot assigns a rank for each frontier cell. The rank represents the number of robots that are closer to the frontier than the robot assigning the rank. Then, the robot selects the frontier with the lowest rank. If two frontier cells have same rank the closest is selected. This assignment is illustrated in Figure 2.5.

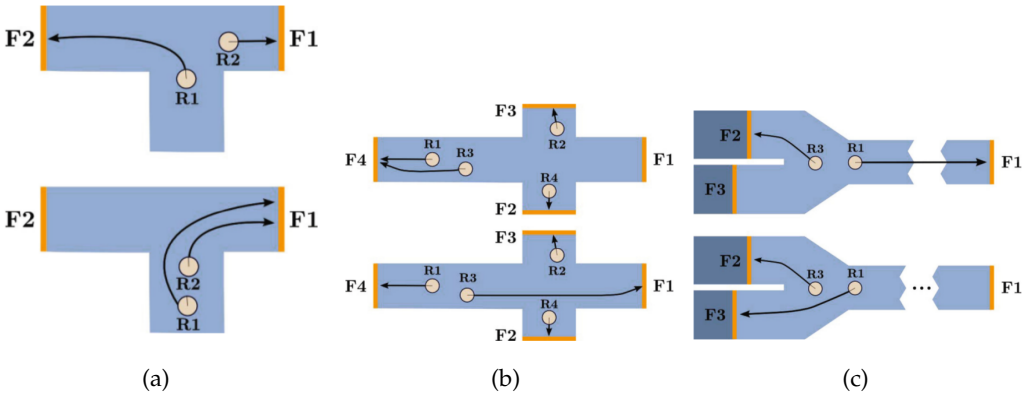


Figure 2.5: (a) displays frontier assignment in two steps with MinPos, (b)(c) compares MinPos (top) and greedy (bottom) approaches, taken from [1]

Chapter 3

Experiments

In this thesis we have compared exploration strategies by the travelled distance needed to explore the whole environment. Because all strategies considered in this thesis are composed of the same components, we can combine these parts to create hybrid strategies, which could have better results than the basic ones. We have combined the following aspects:

- Determination of the goal candidates
- Selection of the next robot goal from the set of goal candidates
- Planning a path to the selected goal location that is followed by the robot during the autonomous navigation

3.1 Considered Strategies

Of these combinations, the following combinations were created:

Greedy approach (GR) – An original Yamauchi's approach [10], where the goal candidates are all frontier cells and the robot selects the closest one according to the length of the path from the robot current location to the frontier cell using the Distance Transform.

Greedy clustered approach (GRS) – A modified greedy assignment, where the goal candidates are not all frontier cells, but they are clustered by the K-means algorithm [7]. Then, the closest representative cell of the frontier cell cluster is selected as the next robot goal.

Greedy with FM approach (FM) – A modified Yamauchi's approach [10]. The goal candidates are all frontier cells, but the robot selects the closest one using EVT and FM metric described in Section 2.3.2.

Greedy with FM clustered approach (FMS) – This approach is the same as the previous one, but the frontier cells are clustered with the K-means algorithm.

Travelling salesman approach (TSP) – This approach is based on solving the traveling salesman problem and is described in Section 2.2. In this approach, we have to cluster frontier cells, otherwise it would be computationally very demanding [7].

All these strategies were also tested with different path planning. We used DT path planning, which produces shorter plans, but paths are close to walls. Then, we used FM

planning method, that produces path that are longer, but they are also farther away from walls. This could be helpful, because for example if a robot, with a short laser range, is in a corridor and hugs the wall, it cannot see the other side of corridor, but if it goes through the middle of corridor it may see both sides.

So all tested strategies are following:

- GR-dtpath − greedy approach with DT based path planning
- GR-fmpath − greedy approach with FM based path planning
- GRS-dtpath − greedy approach with DT based path planning and clustered frontier cells
- GRS-fmpath − greedy approach with FM based path planning and clustered frontier cells
- FM-dtpath − greedy approach using FM metric with DT based path planning
- FM-fmpath − greedy approach using FM metric with FM based path planning
- FMS-dtpath − greedy approach using FM metric with DT based path planning and clustered frontier cells
- FMS-fmpath − greedy approach using FM metric with FM based path planning and clustered frontier cells
- TSP-dtpath − TSP based approach with DT based path planning
- TSP-fmpath − TSP based approach with FM based path planning

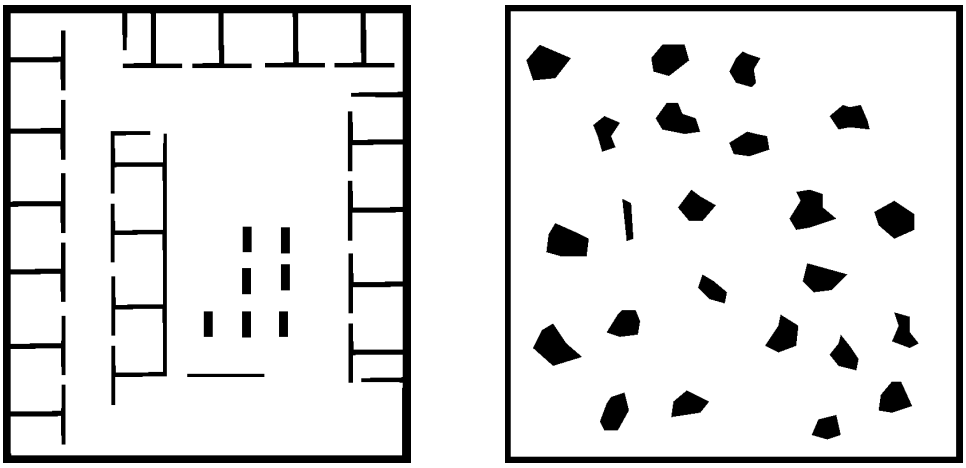
3.2 Considered parameters

The comparison of the exploration strategies was done for different scenarios. Each of this scenario consist of the sensor range $\rho \in 3m, 5m, 7m$, replanning condition and environment with starting position. Replanning conditions are: **TR** − goal replanning, where the new goal is selected when robot reaches the current one, **7SR** − where the new goal is selected after 7 discrete step, or when the robot reaches current one. Environments was jh, which is office like environment, and potholes, which is unstructured environment, with dimensions $21m \times 24m$ and $40m \times 40m$. We can see them in Figure 3.1. Because the used framework is deterministic, we used small random perturbation in starting position, so we created 20 variants. Thus for each strategy we have $2 \times 2 \times 3 \times 20 = 240$ scenarios.

The occupancy and navigational grids used in this experiments had resolution of $0.05m$. Robot had diameter of $0.3m$ and is equipped with omnidirectional laser ranger.

3.3 Discrete time results

All these experiments were done in the framework [4], which works in discrete time. Therefore the computational demands are not considered in this thesis. We studied influence of several parameters and it cannot be compared in one table so we have to divide



(a) jh (office like) environment (b) potholes environment

Figure 3.1: Environments used in experiments.

this comparison to more subsections. As we made many experiments for several scenarios, only selected results are presented here. [1]

3.3.1 Influence of the laser range ρ

The influence of the laser range can be summarized in one sentence: the farther we see, the more we see. But we cannot see around the corner; thus, we can miss some unexplored places especially in a ragged environment. So there is no direct proportion in the laser range and the time needed for exploration of whole environment. There's more logarithmic proportion as we can see in Figure 3.2.

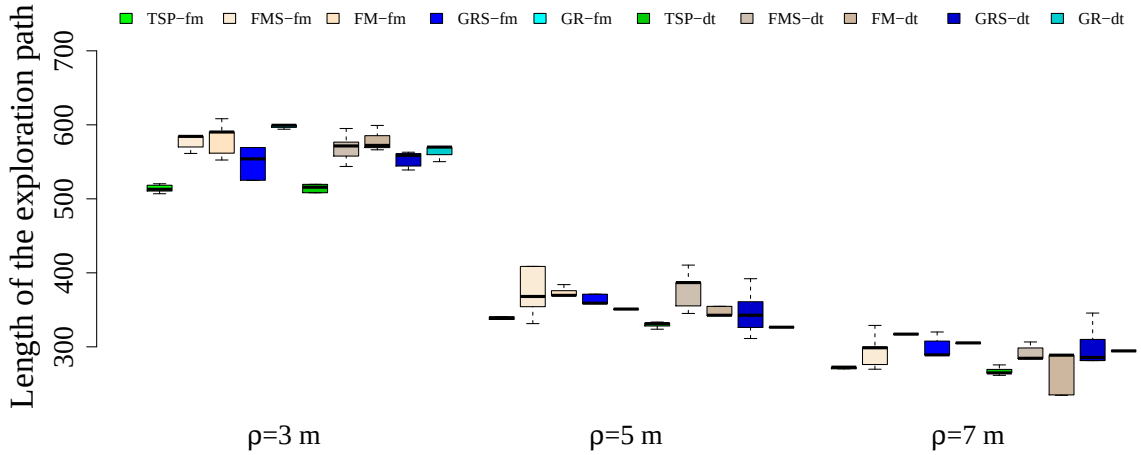
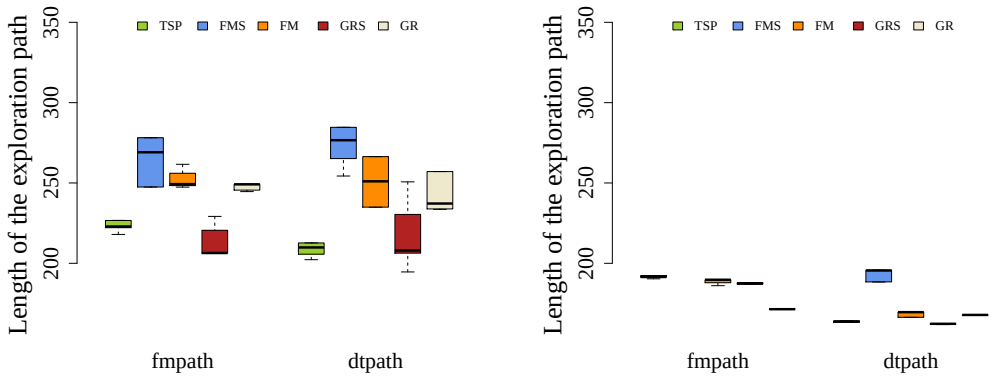


Figure 3.2: Distance travelled at *potholes* environment and with TR replanning

[1] All the results in a form of tables presenting the average required lengths of the exploration paths are depicted in Appendix B.

3.3.2 Influence of the path planning

The results indicates, that the FM paths are almost always longer than the paths determined by the DT. But we these differences are more significant for longer sensor ranges, which support our assumption, that traveling further from walls could be helpful, because for example if a robot, with a short laser range, is in a corridor and hugs the wall, it cannot see the other side of corridor, but if it goes through the middle of corridor it may see both sides. We can see this in Figure 3.3.



(a) $\rho = 3$ (b) $\rho = 7$

Figure 3.3: Comparsion of the required traveled distance to explore the whole environment by a signle robot with different path planning methods in the *jh* environment

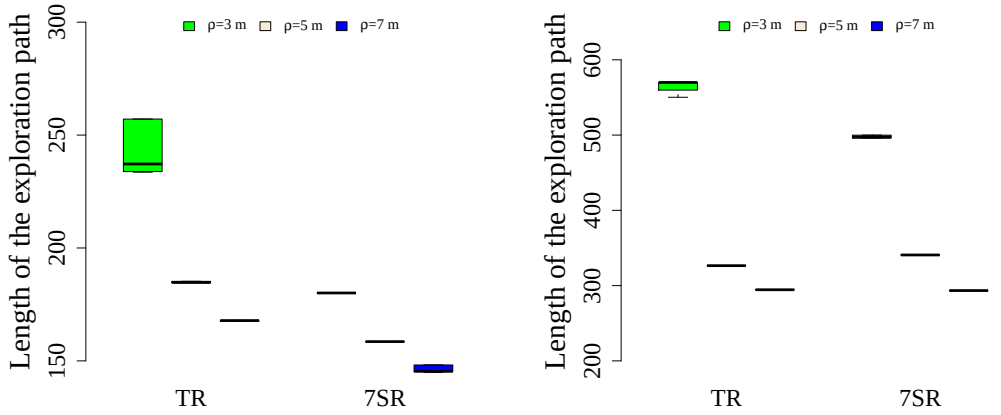
But we thought that the FM paths are faster in real-time; so, we also tested this aspect and present the obtained results in Section 3.4.

3.3.3 Influence of the replanning condition

Generally, we can assume, that more frequent determination of the next navigational goal, a more recent information about the environment is used for the planning and thus a can obtain better But if we are replanning too often, then we may run into some oscillations. In Figure 3.4, there are required length of the exploration paths for two considered replanning conditions, three sensor ranges, and two environments. We can see that for the *jh* environment the influence of the frequency of replanning is quite important and faster replanning significantly decrease the exploration time. On the other hand, for the used open space environment *potholes* the frequency does not have a significant impact to the exploration performance.

3.3.4 Influence of the goal candidate generation

During the experiments we found out that clustering of the frontier cells using the K-means method can produce shorter travel distances. But this has been observed only for exploration strategies that uses DT for selection of the next robot goal. It is because for the clustered frontier cells, a central cell of the cluster is selected as the representative of the cluster and the robot travels farther from a wall and thus explores a larger portion of the environment. But for the FM based approaches the central cell is selected by the algorithm itself, because the middle is the furthest cell from the obstacle. When we cluster cells and

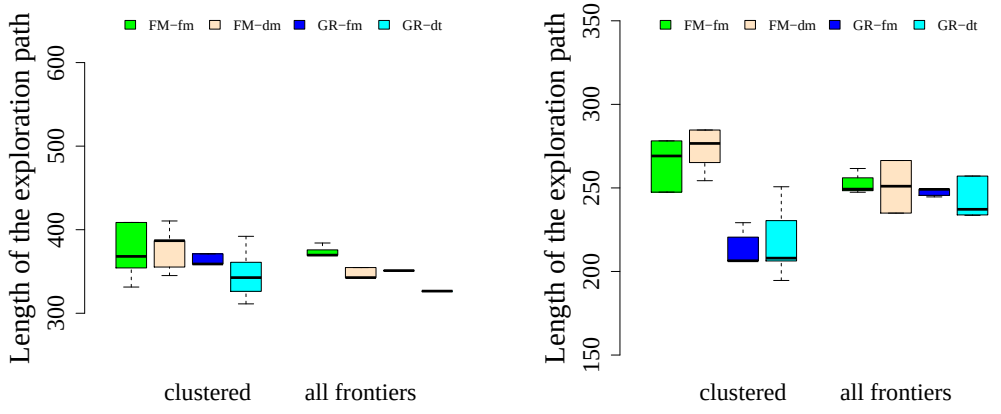


(a) GR-dtpath in *jh* environment (b) GR-dtpath in *potholes* environment

Figure 3.4: Comparsion of distance travelled by a robot with different replanning conditions and sensor ranges in different environments

the frontier edge is long, the middle may not be the representative cell and the robot can be forced to travel to a different one. We can see this in results presented in Figure 3.5(b).

Also in ragged environments there is not a huge difference as we can see in Figure 3.5.



(a) GR-dtpath in *jh* environment (b) GR-dtpath in *potholes* environment

Figure 3.5: Comparsion of distance travelled by robots with different replanning condition and with different laser ranges in *jh* and *potholes* environments

3.4 Real time speeds of path planning methods

Because our robot cannot go through all the paths with the same speed, we also tested how fast it can go with the different path planning. For this purpose we used the Player/Stage framework [6]. In this setup, we utilize the local motion control to follow the planned path based on the Smooth Nearness-Diagram (SND) algorithm, introduced in [2]. This controller slows down the robot near obstacles, but also drives the robot further from them to prevent the robot from crashing into an obstacle. This causes that the path the robot actually passes may differ from the generated by our strategy.

The influence of the path generation to the ability of the robot to follow the planned path has been evaluated in was tested on fully explored jh environment and with the robot with the same dimensions used in the previous experiments. A path to travel has been prepared to visit 7 selected locations. The locations and their required visit is depicted in Figure 3.6.

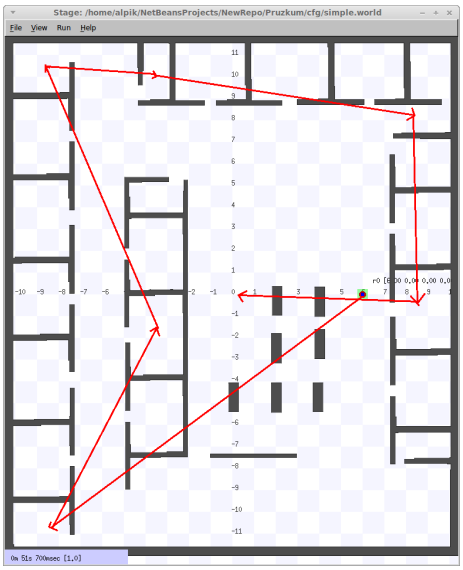


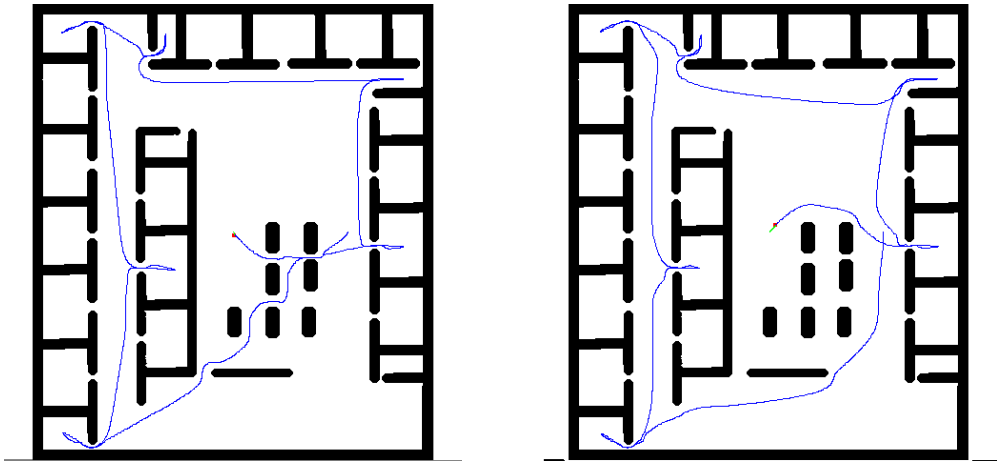
Figure 3.6: The map with the path the robot must follow

We run this experiment 20 times for each strategy and we found out that the paths produced by the FM based path planner were longer, but still the robot passed them in a shorter time than the paths determined by the DT algorithm, see Table 3.1.

Table 3.1: Results from Player/Stage

Path Planning	**Distance** [m]	**Time** [s]	**Average Speed**[m/s]	**Speed ratio according to DT**
DT	95.21	337.60 ± 15.86	0.282 ± 0.013	1
FM	102.86	287.69 ± 2.36	0.358 ± 0.003	1.27

Times and speeds in this table are averages from 20 runs with the standard deviations. For paths of robots see Figure 3.7.



(a) Path produced by robot following DT path planning method using SND driver

(b) Path produced by robot following FM path planning method usinf SND driver

Figure 3.7: Comparsion of paths produced by different path planning mathod

Chapter 4

Conclusion

In this thesis, four exploration strategies are presented. We also present two path planning methods and tested combinations of the strategies and path planning methods. We evaluated performances of these combinations in a discrete time framework, which does not depend on computational resources; thus, we could compare only the quality of the proposed strategies.

The presented results indicate the performance of exploration strategy depends on several parameters. We discovered that longer sensor range leads to better results, but there is not a proportion relation. We also discovered that a more frequent determination of the next navigational goal significantly decreases the exploration time. However, this does not have a significant impact in open space environments. Then, we discovered that clustering of frontier cells by K-means algorithm decreases the exploration time for strategies that uses DT for selection of the next robot goal. On the other hand, it increases the exploration time for the strategies that uses FM for selection of the next robot goal.

The results for path planning methods can be interpreted in two ways. Results from discrete time framework indicate the length of DT based path planning method is shorter. On the other hand results from Player/Stage framework indicates that using FM based path planning method in combination with SND driver produces paths that are about 30% faster. When we combine these two results we find out that strategies with the DT based path planning are shorter, but the overall time needed for exploration is greater than for strategies with FM based path planning.

We compared strategies for exploring the whole unknown environment, but in most cases we do not need to explore the whole environment. We are only looking for some information. Therefore the next logical step is generalization into informative planing, which tries to get the highest quality information using least possible sources (time or fuel, ...). These strategies are implemented for partially known environment which each iteration in robotic exploration is. So we should use informative planing algorithms to evaluate goal candidates with predicted information quality. Then we will be able to select the goal candidate which will give us the most information instead the nearest one.

Bibliography

[1] Bautin, A.; Simonin, O.; Charpillet, F.: Minpos : A novel frontier allocation algorithm for multi-robot exploration. *ICIRA (2)*, 2012: str. 496–508.

[2] Durham, J. W.; Bullo, F.: Smooth Nearness-Diagram Navigation. *Intelligent Robots and Systems*, 2008.

[3] Faigl, J.; Kulich, M.: On Determination of Goal Candidates in Frontier-Based Multi-Robot Exploration. *European Conference on Mobile Robots (ECMR)*, 2013.

[4] Faigl, J.; Simonin, O.; Charpillet, F.: Comparison of Task-Allocation Algorithms in Frontier-Based Multi-Robot Exploration. V *12th European Conference on Multi-Agent Systems (EUMAS'14)*, 2014.

[5] Garrido, S.; Moreno, L.; Blanco, D.: Exploration of 2D and 3D Environments using Voronoi Transform and Fast Marching Method. *J. Intell. Robotics Syst. 55:1*, 2009: s. 55–80.

[6] Gerkey, B. P.; Vaughan, R. T.; Howard, A.: The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems (2003). *Proceedings of the International Conference on Advanced Robotics (ICAR 2003)*, 2003: s. 317–323,.

[7] Kulich, M.; Faigl, J.; Preucil, L.: On distance utility in the exploration task. *ICRA*, 2011: s. 4455–4460.

[8] Strand, R.; Normand, N.: Distance Transform Computation for Digital Distance Functions. *Theoretical Computer Science*, ročník 448, 2012: s. 80–93, doi:10.1016/j.tcs.2012.05.010.

[9] Traveling Salesman Problem. [cit. 2014-05-15].
URL `http://www.math.uwaterloo.ca/tsp/index.html`

[10] Yamauchi, B.: A frontier-based approach for autonomous exploration. *in Proc. of IEEE Int. Symposium on Computational Intelligence in Robotics and Automation. IEEE Comput. Soc. Press*, 1997: str. 146–151.

Appendix A

CD Content

Attached CD contains source codes of framework, Bachelors Thesis in PDF format and source codes of this text in LaTeX.

CD structure is in next table.

Table A.1: Structure of CD

Folder	Description
`src\mre`	discrete time framework source codes
`src\plstg`	source codes for player stage framework
`doc`	bachelors thesis source codes
`thesis.pdf`	bachelors thesis text

Appendix B

Detailed Results

Table B.1: Map jh TR replanning condition

	fm path			dt path		
	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]
TSP	223	194	192	209	176	164
FM	252	197	189	251	191	169
FMS	266	230	225	274	210	193
GR	248	200	171	242	185	166
GRS	212	185	187	217	165	162

Table B.2: Map jh 7SR replanning condition

	fm path			dt path		
	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]
TSP	177	151	144	164	129	124
FM	180	176	154	177	163	154
FMS	211	158	160	195	154	151
GR	206	164	160	181	159	146
GRS	189		164	196	136	146

Table B.3: Map potholes TR replanning condition

	fm path			dt path		
	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]
TSP	512	337	271	511	331	267
FM	580	373	314	575	350	265
FMS	580	380	293	568	376	290
GR	598	351	305	564	327	295
GRS	551	370	300	558	346	299

Table B.4: Map potholes 7SR replanning condition

	fm path			dt path		
	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]	$\rho = 3$ [m]	$\rho = 5$ [m]	$\rho = 7$ [m]
TSP				480	322	234
FM		375	282	531	345	233
FMS		388	320	543	390	253
GR			298	496	341	293
GRS				486	346	259

Table B.5: Map jh TR replanning condition

	fm path			dt path		
	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]
TSP	619	539	533	746	629	586
FM	700	547	525	896	682	604
FMS	739	639	625	979	750	689
GR	689	556	475	864	661	593
GRS	589	514	519	775	589	579

Table B.6: Map jh 7SR replanning condition

	fm path			dt path		
	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]
TSP	492	419	400	586	461	443
FM	500	489	428	632	582	550
FMS	486	439	444	696	550	539
GR	572	456	444	646	568	521
GRS	525		456	700	486	521

Table B.7: Map potholes TR replanning condition

	fm path			dt path		
	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]
TSP	1422	936	753	1825	1182	954
FM	1611	1036	872	2054	1250	946
FMS	1611	1056	814	2029	1343	1036
GR	1661	975	947	2014	1168	1054
GRS	1531	1028	833	1993	1236	1068

Table B.8: Map potholes 7SR replanning condition

	fm path			dt path		
	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]	$\rho = 3$ [s]	$\rho = 5$ [s]	$\rho = 7$ [s]
TSP				1714	1150	836
FM		1042	783	1896	1232	832
FMS		1078	889	1939	1393	904
GR			828	1771	1218	1046
GRS				1736	1236	925