Noname manuscript No.
(will be inserted by the editor)

Exploring Unstructured Environment with Frontier Trees

R. Korb · A. Schoettl

the date of receipt and acceptance should be inserted later

Abstract **This paper presents the Frontier Tree exploration algorithm, a novel approach to autonomously explore unknown and unstructured areas. Focus of this work is the exploration of domestic environments with arbitrary obstacles, for example furbished appartements. Existing and well studied approaches like greedy algorithms perform worse, when obstacles are included and the range of distance sensors is limited. Base of this research is the Frontier Tree. This data structure offers two main features. It is a memory of past poses during exploration and is utilized to decide between future navigation goals. This approach is compared to a basic but efficient nearest neighbour exploration. The algorithm is tested in simulation with maps similar to appartement ground maps including furniture as obstacles. The paper shows, that frontier trees minimize the travelled path of the mobile robot by tracking open boundaries.**

Keywords Robotics · Exploration · Path Planning · Frontiers · Domestic Environment

Rudolf F. S. Korb
University of Applied Sciences - Munich
Dept. 04 Electrical Engineering and Information Technology, Laboratory for Autonomous Systems in Munich (LASIM), Lothstr. 64, 80335 Munich, Germany
Tel.: +49-89-1265-3415
E-mail: rudolf.korb@hm.edu

Alfred Schoettl
University of Applied Sciences - Munich
Dept. 04 Electrical Engineering and Information Technology, Laboratory for Autonomous Systems in Munich (LASIM), Lothstr. 64, 80335 Munich, Germany
Tel.: +49-89-1265-3415
E-mail: alfred.schoettl@hm.edu

1 Introduction

Mobile robots must be familiar with their surroundings to execute complex tasks, for example pick-up and delivery services. As an initial step, the workspace has to be explored autonomously. This problem can be divided into three parts; mapping, localization and exploration. Focus of this research is planning future exploration steps to uncover the environment efficiently, depending on past steps, a map of the area already explored and the agent's current position.
When robots participate in everyday life, their operation space consists of unstructured, dynamic environments and has to deal with interieur and crowded areas. As a consequence, when service robots operate in private appartements or public buildings like hospitals or retirement homes, they need to adapt itselves to their surroudings. Therefore it is crucial for a mobile robot to explore areas autonomously and not rely on an environment designed specifically for a service device.
Basic algorithms follow a greedy approach to select future robot positions during the exploration problem. These approaches use the robot's current state and plan one step ahead by minimizing a cost function, for example the agent's euclidian distance to a boundary of known and unknown space. For exploration of simple maps, unfurbished rooms and corridors, basic algorithms provide acceptable results, depending on for example sensor constraints. With high range distance sensors and obstacle free areas, robots cover the entire room before leaving and most likely do not create open boundaries. Therefore, rooms do not have to be visited again and the traveled distance is less compared to low range sensing robots. Furthermore, the number of possible decisions to choose the next goal from is rising with the environment's complexity. As a result, the greedy al-

gorithm's performance will decrease. The robot has to revisit parts of the map where boundaries have been forgotten, increasing the traveled distance.
Consequently the two main influences for travelling distance during exploration is the complexity of an agent's environment and constraints of its range sensors. Mobile autonomous robots in unstructered environment are dealing with both, high complexity and sensor limitations. Many algorithms use maps with reduced difficulty, expecting an empty environment or assume a sensor with infinite distance measurements. It follows, that algorithms covering only one impact decrease in efficiency, when a complex environment and sensor limitations are present.
Greedy algorithms dealing with both influences have one major flaw, which leads to a rise of traveled distance and exploration time. Approaching a boundary already discovered several steps ahead, is an indication that the robot's current exploration path is a cycle. As a result, open boundaries between the starting and endpoint of the cycle will be reached with high travel costs later on during the exploration task. The frontier tree recognizes cycles and hence is able to avoid high costs and react early on forgotten boundaries.

2 Related Work

Many exploration algorithms use the frontier approach introduced by Yamauchi [4]. A frontier is defined as the boudary between known and unknown space in the currently explored map. The algorithm proposed in this paper also utilizes frontiers as atomic elements in the tree data structure and are discussed in detail in section 4.3. Yamauchi's exploration algorithm is a greedy approch and chooses the shortest obstacle free path to minimize the distance between a robot's position and the next frontier.
Koenig et al. [9] show, that greedy algorithms explore ares with acceptable results. For improvements, map segmentation algorithms were introduced [5, 6] to reduce the travel distance of greedy approaches. Wurm et al. [6] use Voronoi Graphs to divide the explored environment in segments. While large emtpy spaces are segmented well, narrow areas, for example corridors, yield a vast amount of partitions. As a result, the disadvantage of greedy algorithms remains. Holz et al. [5] extend the segmentation by adding a step for merging several partitions.
To keep track of open boundaries, Nasir and Elnagar [7] use a data structure called Gap Navigation Tree, introduced by Tovar et al. [3].
Another approach is using information based heuristics [1,2,8]. Visser and Slamet [2] minimize information entropy by selecting frontiers with the largest area. Additionally, they consider travel distance to each frontier in their cost function, to increase the algoritm's efficiency. In [1], Mobarhani et al. also discuss the tradeoff between an information based heuristic and travel distance. They count the number of points of all frontiers at an angle interval from the robot's current position and fill an angle histogram. The next best frontier is selected with a weighted cost funtion by maximising the number of frontier cells at a specific direction and minimizing the travel time to the frontier.
Information based approaches are especially good for rescue scenarios.Focus of these algorithms is uncovering large parts of the map quickly. It is possible that these heuristics result in alternating poses between opposite parts of the map. This behaviour leads to an increased travel distance.

3 Outline

Although the difference to between the worst case of greedy algorithms and the worst case optimal travel distance is reasonable small, the performance is reduced with rising map complexity and sensor limitations. A major weakness of an efficient greedy algorithm is illustrated in figure 1. When entering a room with obstacles, it is possible, that the robot runs in a cycle and returns to a previous area before finishing the room. This behaviour results in higher travel cost, because the robt has to revisit the room. Segmentation approaches [5, 6] to further reduce travel cost, highly rely on the geometrical appearence of the map. With an environment containing obstacles, the probability of revisiting a room is increasing. Segmentation in combination with merging minimizes the number of segments, but does not eliminate the problem, where the robot's path is in a cycle.
With Gap Navigation Trees Tovar et al. [3] solve the problem of map complexity with arbitrary obstacles, but assume infinite distance measurement und 360° opening angle and therefore do not have sensor constraints. A reduced map size grants this property for real life experiments.
The exploration algorithm proposed in this paper performs well with complex structure and sensor limitations. It offers the ability to react, when the agent runs in a cycle to prevent further travel cost.

4 The Frontier Tree Algorithm

Foundation of this exploration algorithm is a tree data structure with frontiers as nodes. The definition of a

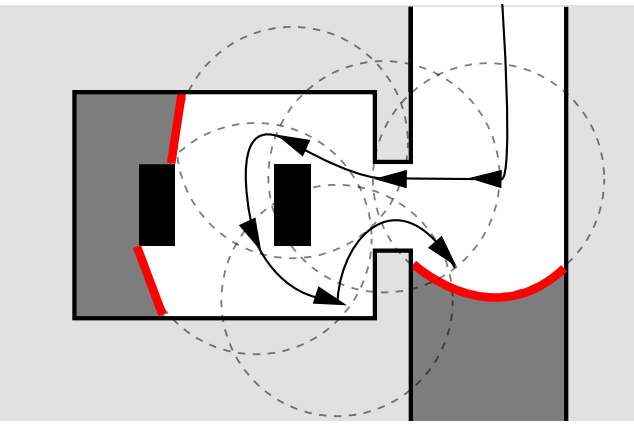


Fig. 1: This illustration shows six robot states during exploration, following a greedy algorithm. The robot enters a room with obstacles and leaves before uncovering the whole area. As a result, the agent has to return at a later exploration step.

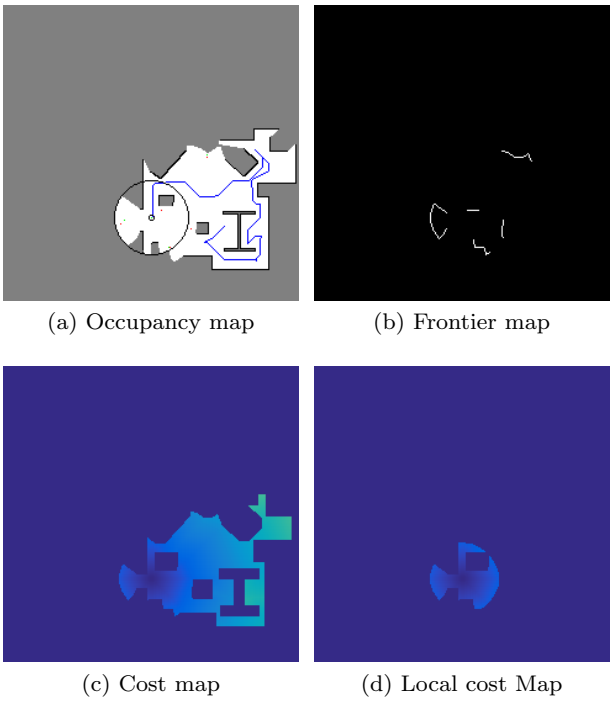
frontier is related to Yamauchi's approach [4]. Similar to Gap Navigation Trees [3], the data structure is, inter alia, used for tracking open boundaries during exploration. Additionally, the resulting topological tree map is used to select the next exploration goal. The final path planning to the new pose is executed on a 2D grid map.

4.1 Robot Model

In this paper the robot is implemented as mobile base with a circular footprint and three degrees of freedom. The position and orientation is defined by $\mathbf{r} = (x, y, \theta)^T$ with x, y as position and θ as orientation. Additionally, the agent provides a distance sensor, represented by $\mathbf{s} = (d, \alpha)^T$. Component d is the sensor's range and α its opening angle. As an assumption, the robot is able to localize itself on a 2D grid map using a simultaneous localization and mapping (SLAM) approach.

4.2 Map Representations

To fulfill the exploration task, the algorithm uses five maps of three different types. A basic map type is the occupancy grid map with values between 0 and 1, where white pixels (1) mark free and black pixels (0) unpassable space. Values in the interval $(0, 1)$ can be interpreted as unknown space. An oocupancy map is the output of a SLAM algorithm. Unpassable pixels are inflated by the radius of the robot's footprint, resulting in a global occupancy map occ_g where the agent can be



(a) Occupancy map (b) Frontier map

(c) Cost map (d) Local cost Map

Fig. 2: (a)The Occupancy map displays the knowledge of the robot about its surroundings with white pixels as free, black as occupied and grey as unknown space. (b)Exploration goals for the next steps are displayed in the Frontier Map. (c)The Cost Map combines information about travel cost and reachable points. Pure black is not reachable and travel cost rise with the brighness. (d)The local cost map shows travel cost and reachable points from $\mathbf{r}$ to a distance $d + \sigma$ around the robot.

treated as a point. occ_g is used to create a local (with dependency of $\mathbf{r}$) occupancy map occ_l.

$$\text{occ}_l(\mathbf{x}) = \begin{cases} 0.5, & \text{if } \text{dist}(\mathbf{x}, \mathbf{r}) \geq l \\ \text{occ}_g(\mathbf{x}), & \text{otherwise} \end{cases} \tag{1}$$

where $\text{dist}(\mathbf{x}, \mathbf{r})$ is the euclidian distance between the considered pixel $\mathbf{x}$ in occ_{local} to the current robot position $\mathbf{r}$.
Both, occ_g and occ_l, as well as the current pose $\mathbf{r}$ are utilized for the calculation of global and local cost maps cost_g and cost_l. These are calculated with Dijstra's Algorithm. Without a return criterion, the result is a map with the $\text{cost}(\mathbf{x})$ to travel from $\mathbf{r}$ to $\mathbf{x}$.
Lastly, occ_g is used to create a frontier map F, containing the possible future goals for exploration. Since occ_g is an inflated representation, every point in F is a possible navigation goal. Section 4.3 gives a formal description of frontiers.

4.3 Frontiers

Frontiers are extracted from the global occupancy map. A Frontier f^i, with index i as unique id, is the boundary between unknown and free space in occ_g [4]. An elementary part of frontier f^i are frontier cells f^i_j, where superscript j denotes the index of the single cell inside the frontier. A cell is defined by the following two properties:

$$\mathrm{occ}_g(\mathbf{x}) = 1 \tag{2}$$

$$\exists\, \mathrm{occ}_g(\mathbf{y}) \in N_8(\mathbf{x}) : \mathrm{occ}_g(\mathbf{y}) = (0, 1) \tag{3}$$

with N_8 as the Moore Neighbourhood of index $\mathbf{x}$ in occupancy map occ_g. Adjacent frontier cells are merged to a single frontier f^i. $\mathbb{F}$ is the set of all frontiers f^i. Every frontier must specify a cell f^i_g that declares a navigation goal to plan a path from $\mathbf{r}$ to f^i. This position is given as

$$f^i_g = \min(\|f^i_j - \bar{f}^i\|_2), \tag{4}$$

The frontier cell with the minimum euclidian distance from f^i_j to the mean value of all frontier cells $\bar{f}^i$ is selected.

4.4 Tree Structure

Root of the tree structure is the initial position $\mathbf{r}_{init}$ of the robot on the map. Each node n has a connection to its parent and contains a list of all children. These links offer the ability to traverse the tree in both directions. The number of a node's descendants is dependent on $\mathbf{r}$ and cost_l. Additionally, every n_i saves f^i_g as data element, representing the navigation goal of the node. For basic operations on the tree, a breadth first search was implemented.

4.5 Tree Operations

The Frontier Tree offers two basic operations to directly manipulate the data structure. An Insert function extends the existing children of a node by attaching an element. Every inserted node is also a leaf and therefor can not be merged between a parent element and its existing descendants.
Instead of a delete function, the Frontier Tree offers an operation for marking nodes. Marked nodes are not considered for further exploration steps and cannot be extended with the insert function, consequently all marked nodes are also leafs.

4.5.1 Frontier Sets

Inserting and marking are straightforward functions. The essential and difficult part is the identification, which frontiers have to be inserted or marked in the tree structure. Frontier trees depend on two sets $\mathbb{F}^t$ and $\mathbb{F}^{t-1}$ to perform the decision procedure. Both sets $\mathbb{F}$ represent the frontiers of two following exploration steps. Initially, $\mathbb{F}^t$ is divided into two disjoint subsets, with

$$\mathbb{F}^t_{in} = \{f^i \in \mathbb{F} \mid \mathrm{cost}_l(f^i_g) > 0\} \tag{5}$$

$$\mathbb{F}^t_{out} = \overline{\mathbb{F}_{in}} \tag{6}$$

The set of the prior step is defined by all the unexplored leaf nodes in the Frontier Tree. Since the tree is in the state before node insertion, the robot's current position f^t_c is also a leaf but already explored. Consequently the f^t_c has to be excluded, resulting in $\mathbb{F}^{t-1}\backslash\{f^t_c\}$.

4.5.2 Binary Identification Matrix

The frontier set $\mathbb{F}^t_{in}$ can be inserted at the current position. It contains all reachable frontiers from f^t_c. In the following step, the algorithm marks all leafs, which are not considered anymore in the exploration progress. There exist two cases where nodes have to be marked: First, when space is uncovered and as a result, frontiers disappears in the exploration step.Secondly, Frontiers previously discovered are in the set $\mathbb{F}^t_{in}$.
Initially, the distance matrix $\mathbf{D}$ is determinded by a conjunction of $\mathbb{F}^{t-1}\backslash\{f^t_c\}$ and F^t_{out}:

$$\mathbb{F}^{t-1}\backslash\{f^t_c\} \times \mathbb{F}^t_{out} \longrightarrow \mathbb{R} \tag{7}$$

$$\mathbf{D} : (f^{i,t-1}_g, f^{j,t}_g) \longmapsto \|f^{i,t-1}_g, f^{j,t}_g\|_2 \tag{8}$$

The resulting matrix has the dimensions $|\mathbb{F}^{t-1}\backslash\{f^t_c\}| \times |\mathbb{F}^t_{out}|$

4.6 Cycles

5 Results

6 Conclusion

References

1. Amir Mobarhani, Shaghayegh Nazari, Amir H. Tamjidi, Hamid D. Taghirad: Histogram Based Frontier Exploration (2011)
2. Arnoud Visser, B.A. Slamet: Balancing the information gain against the movement cost for multi-robot frontier exploration. In: European Robotics Symposium 2008 by H. Bruyninckx & L. Peuil & M. Kulich. Prague (2008)
3. Benjamin Tovar, Steven M. LaValle, Rafael Murrieta: Optimal Navigation and Object Finding without Geometric Maps or Localization. In: Robotics and Automation, 2003. Proceedings. ICRA '03 (2003)

4. Brian Yamauchi: A frontier-based approach for autonomous exploration. In: CIRA'97., Proceedings (2007). DOI 10.1109/CIRA.1997.613851
5. Dirk Holz, Nicola Basilico, Francesco Amigoni, Sven Behnke: Evaluating the Efficiency of Frontier-based Exploration Strategies (2010)
6. Kai M. Wurm, Cyrill Stachniss, Wolfram Burgard: Coordinated Multi-Robot Exploration using a Segmentation of the Environment (2008)
7. Reem Nasir, Ashraf Elnagar: Gap Navigation Trees for Discovering Unknown Environments. Intelligent Control and Automation **2015**(6), 229–240 (2015)
8. Robert Grabowski, Pradeep Khosla, Howie Choset: Autonomous Exploration via Regions of Interest. In: Proceedings of the 2003 IEEE Intl. Conference on lntelligent Robots and Systems. Las Vegas (2003)
9. Sven Koenig, Craig Tovey, William Halliburton: Greedy mapping of terrain. In: Robotics and Automation, 2001. Proceedings 2001 ICRA (2001). DOI 10.1109/ROBOT.2001.933175