Efficient Robot Self-Localization in Simple Polygons

Sven Schuierer*

1 Introduction

One of the basic tasks faced by an autonomous mobile robot is the problem of *self-localization*, that is determining its position in its environment; this is also sometimes called the "where am I?" problem. There are many variations of this problem depending on the environment and the sensor data and a-priori information available to the robot. Here, we consider a robot that is given a map of its environment but has no knowledge of its location on the map. Often, it is assumed that this problem can be solved by only using sensor data and allowing the robot to make a small local pertubation of its current position. But if there are self-similar parts in the environment, then this approach may not suffice to distinguish between the possible locations of the robot.

This issue has been addressed in numerous contexts, ranging from aerial photography to autonomous vehicles for the exploration of landscapes. We consider an idealized version of the problem where the robot is located in an obstacle-free indoor environment which can be modeled as a simple polygon P; the sensor data available to the robot are all the points that are visible from its position, i.e. its local visibility polygon V, via a range sensing device. The robot is also assumed to know its orientation, e.g. by a compass, and to be able to perform error-free movements.

The first task in the process of self-localization faced by the robot is to determine its possible placements within P; that is, all points p in P such that the visibility polygon of p equals V. An efficient enumeration of all such positions is possible with a data-structure by Guibas, Motwani, and Raghavan in time $O(nm)$ if P consists of n and V consists of m vertices [2].

If more than one placement exists, then the robot has to follow a path that allows it to distinguish between the different placements. Dudek, Romanik, and Whitesides consider a setting in which the environment can be modeled as a simple polygon P as described above [1]. They propose a simple strategy where the robot repeatedly travels to the closest point that eliminates at least one of the possible placements. Unfortunately, the algorithm presented by Dudek *et al.* has a time and space complexity of $O(k^2n^4)$ which is prohibitively large, even for a small number of placements and very simple environments.

In this talk we present an implementation of the localization strategy by Dudek *et al.* which reduces the time complexity of the algorithm to $O(kn \log n)$ and the space complexity to $O(kn)$. As we use a number of advanced techniques in our algorithm which are difficult to implement, we also present an alternative implementation of our algorithm with an increased running time of $O(kn^2)$ that uses only fairly simple subroutines.

*Institut für Informatik, Am Flughafen 17, Geb. 051, D-79110 Freiburg.
email: `schuiere@informatik.uni-freiburg.de`

2 Definitions

We deal with environments that can be modeled by a *simple polygon.* This is motivated by the fact that it is often possible to construct a polygonal map of indoor environments which are one of the most important application domains for autonomous mobile robots.

We also need some definitions that deal with visibility.

Definition 2.1 *Let P be a polygon and p a point in P. The set of points that are visible to p is called the* visibility polygon of *p and denoted by $vis(p)$.*

A maximal line segment of the boundary of $vis(p)$ that does not belong to the boundary of P is called a *window* of $vis(p)$.

In the following let π be the set of points p in P with $vis(p) = V$. The points in π are called the *placements* of the robot, i.e., a placement is one possible "wake-up" position of the robot in the beginning. In the following we assume that $|\pi| = k$.

3 Minimum Distance Localization

A mentioned before a simple strategy for robot localization is to repeatedly go to the closest point from the "wake-up" position of the robot at which at least one of the initial placements can be eliminated. The idea for this strategy was first proposed by Dudek *et al.* [1].

The following is one key observation.

Lemma 3.1 *If q^* is the closest point to the origin at which at least one placement can be eliminated , then there is a placement p in π and a vertex v of P such that $p + q^*$ is located on the window of $vis(v)$ that is the closest to p. In fact, $p + q^*$ is the closest point of w to p.*

In order to compute the closest point at which the robot can eliminate at least one placement we need the following structure.

3.1 The Overlay Tree

A *geometric tree* $T = (V, E)$ is a tree embedded into $\mathbb{E}^2$ such that each vertex $v \in V$ is a point and each edge $e \in E$ is a line segment whose end points lie in V. The line segments of E intersect only at points in V, and they do not induce any cycles.

Definition 3.1 *Let $\mathcal{T}$ be a set of geometric trees and $\mathcal{V} = \{v_T \in T \mid T \in \mathcal{T}\}$ a set of vertices, one of each tree T in $\mathcal{T}$. The* overlay tree *$ov(\mathcal{T}, \mathcal{V})$ of the trees in $\mathcal{T}$ w.r.t. $\mathcal{V}$ is defined as follows.*

Let $q \in T$ and $q' \in T'$ with $T, T' \in \mathcal{T}$. We say (q, v_T) and $(q', v_{T'})$ are equivalent *if the path in T from v_T to q is a translation of the path in T' from $v_{T'}$ to q'. We write $(q, v_T) \sim (q', v_{T'})$ in this case. $\sim$ clearly induces an equivalence relation. We define $ov(\mathcal{T}, \mathcal{V})$ to be the set of equivalence classes under $\sim$. For illustration refer to Figure 1.*

Let $\mathcal{T}$ be a set of geometric trees that are contained in P. Let $\overline{q}$ be an equivalence class of $ov(\mathcal{T}, \mathcal{V})$. We say the equivalence class $\overline{q}$ is *complete* if, for all $T, T' \in \mathcal{T}$, there is a (q_T, v_T) and a $(q_{T'}, v_{T'})$ in $\overline{q}$ and $vis(q_T)$ contains a translation of the vertices and edges of $vis(q_{T'})$. An equivalence class of $ov(\mathcal{T}, \mathcal{V})$ is *incomplete* if it is not complete.

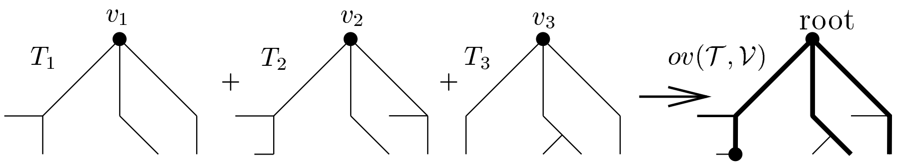

Figure 1: The overlaytree of $\mathcal{T} = \{T_1, T_2, T_3\}$ w.r.t. $\mathcal{V} = \{v_1, v_2, v_3\}$.

3.2 The Algorithm

Our localization algorithm can now be described as follows.

Algorithm Minimum Distance Localization

1 **for all** placements p in π **do**
2   **for all** vertices v of P **do**
        compute the closest point v_p^* of $vis(v)$ to p;
3 compute the shortest path tree T_p^* from p to the points v_p^*, for $v \in P$;
4 let $\mathcal{T}_\pi^* := \left\{ T_p^* \mid p \in \pi \right\}$; compute the overlay tree $ov(\mathcal{T}_\pi^*, \pi)$ of $\mathcal{T}_\pi^*$ w.r.t. π;
5 **while** more than one placement remains **do**
6   let $\overline{\ell}^*$ be the closest incomplete leaf of $ov(\mathcal{T}_\pi^*, \pi)$ to the root;
7   let q^* be a point and p^* be a placement such that $(q^*, p^*) \in \overline{\ell}^*$;
8   follow the shortest path from the origin to $q^* - p^*$ and return to the origin;
9   remove the placements p from π and the corresponding trees T_p^* from $ov(\mathcal{T}_\pi^*, \pi)$ which are eliminated at $q^* - p^*$;
end while;

The following lemmas show that our algorithm is correct.

Lemma 3.2 *Let $\overline{\ell} \in ov(\mathcal{T}_\pi^*, \pi)$ be a complete leaf. If there is no incomplete leaf $\overline{\ell}'$ of $ov(\mathcal{T}_\pi^*, \pi)$ with $d(\overline{\ell}', \overline{r}) \leq d(\overline{\ell}, \overline{r})$, then all the placements are indistinguishable at $\overline{\ell}$.*

Lemma 3.3 *If $\overline{\ell}$ is the leaf chosen in Step 6 and p is a placement such that $(q, p) \in \overline{\ell}$, then $q - p$ is the closest point to the origin at which the robot can distinguish two placements.*

Of course, the above formulation does not cover all of the implementation details, in fact, most are left out but with a careful choice of data-structures the above algorithm can be implemented to run in time $O(kn \log n)$ and space $O(kn)$.

References

[1] G. Dudek, K. Romanik, and S. Whitesides. Localizing a robot with minimum travel. In *Proc. 6th ACM-SIAM Symp. on Discrete Algorithms*, pages 437–446, 1995.

[2] L. Guibas, R. Motwani, and P. Raghavan. The robot localization problem in two dimensions. In *Proc. 3rd ACM-SIAM Symp. on Discrete Algorithms*, 1992.