Localizing a Robot with Minimum Travel *

Gregory Dudek† Kathleen Romanik‡ Sue Whitesides§

School of Computer Science
McGill University
3480 University Street
Montréal, Québec, Canada H3A 2A7

Abstract

We consider the problem of localizing a robot in a known environment modeled by a simple polygon P. We assume that the robot has a map of P but is placed at an unknown location. The robot must move around and use range sensing and a compass to determine its position (i.e. localize itself). From its initial location, the robot sees a set of points called the visibility polygon V of its location. In general, this will not suffice to uniquely localize the robot, since the set H of points in P with visibility polygon V may have more than one element. To address this difficulty, we combine information from multiple vantage points seeking a strategy that minimizes the distance the robot travels to determine its exact location. An optimal localization strategy would direct the robot to follow a minimum length path to verify its location, but this is impossible to compute without *a priori* knowing which of the hypothetical locations in H is the true initial location of the robot.

In this paper, we define a natural, algorithmic variant of the problem of localizing a robot with minimum travel. We then show this variant is NP-hard. Finally, we give a polynomial time approximation scheme that causes the robot to travel a distance of at most $k = |H|$ times d, where d is the length of a minimum length tour that would allow the robot to verify its true initial location by sensing. This is remarkable in view of the fact that the length d of such a minimum length tour cannot be determined without *a priori* knowledge of which hypothetical location in H is the true one, and yet our strategy determines a path whose length is provably within a factor k of the best possible without using such *a priori* knowledge.

*Research supported by NSERC, FCAR, National Network of Centres of Excellence IRIS programme

†dudek@cim.mcgill.ca

‡romanik@cs.mcgill.ca

§sue@cs.mcgill.ca

1 Introduction

Numerous mobile robot tasks call for a robot that has a map of its environment and knowledge of where it is located in the map. Determining the position of the robot in the environment is known as the *robot localization problem*. To date, mobile robot research that uses a map generally assumes either that the position of the robot is always known, or that it can be estimated using sensor data acquired by displacing the robot only small amounts [KMK93, TA92]. However, self-similarities between separate portions of the environment prevent a robot that has been dropped into or activated at some unknown place from uniquely determining its exact location without moving around. This motivates a search for strategies that direct the robot to travel around its environment and to collect additional sensory data [BD90, DJMW93] to deduce its exact position.

In this paper, we view the general robot localization problem as consisting of two phases. The first phase is to determine the set H of *hypothetical locations* that are consistent with the sensing data obtained by the robot at its initial location. The second phase is to determine, in the case that H contains two or more locations, which location is the true initial position of the robot; i.e. to eliminate the incorrect hypotheses. Ideally, for reasons of speed and accuracy, the robot should travel the minimum distance necessary to determine its exact location.

A solution to the hypothesis generation phase of robot localization has been given by Guibas, Motwani and Raghavan in [GMR92], and we describe their results later. Our paper is concerned with minimizing the distance traveled in the hypothesis elimination phase of robot localization. Together, the two papers give a solution to the general robot localization problem.

We show that the problem of localizing a robot with minimum travel is NP-hard. We then solve the hypothesis elimination phase with what we call a greedy localization strategy. Our strategy causes the robot to travel a distance that is bounded above by $k = |H|$ times the length d of a minimum length tour that allows the robot to verify its true initial position by sensing. Such a minimum length tour cannot be determined without *a priori* knowledge of which hypothetical location in H is the true one, and yet our strategy determines a tour whose length is within a factor k of the minimum without using such *a priori* knowledge.

2 Preliminaries

In this section, we describe our robot model and give some key definitions.

2.1 Assumptions about the robot

- The robot is mobile and moves in a static 2-dimensional obstacle-free environment. We model the movement of the robot in the environment by a point p moving inside and along the boundary of an n-vertex polygon P positioned somewhere in the plane.
- The robot has a map of its environment, i.e., it knows both P and the orientation of P in the plane.
- The robot has a compass and a range sensing device. It is essential that the robot be able to determine its orientation (with the compass); otherwise it cannot determine its exact location in an environment with non-trivial symmetry such as a square.
- The robot's sensor can detect the orientations of, and the distances to, those walls for which an unobstructed straight line can be drawn from its current location. The observations at a particular location determine a polygon V of points that the robot can see from that location. This is analogous to a laser range sensor or a simple model of sonar sensing.

In order to abstract the sensory interpretation process, we use a *visibility skeleton* (defined later) that the robot will compute for its current location, based on its observations. We use this abstraction because there are only a finite number of visibility skeletons for all the points in P.

2.2 Some definitions and an example

Two points in P are *visible* to each other or *see* each other if the straight line segment joining them does not intersect the exterior of P. The *visibility polygon* $V(p)$ for a point $p \in P$ is the polygon consisting of all points in P that are visible from p. We denote by V the visibility polygon of the initial location of the robot. There may be more than one location in P with visibility polygon V, so $V = V(p)$ for one or more points $p \in P$. The number of vertices of V is denoted by m. Since the robot has a compass, we assume the representations of P and V have a common reference direction.

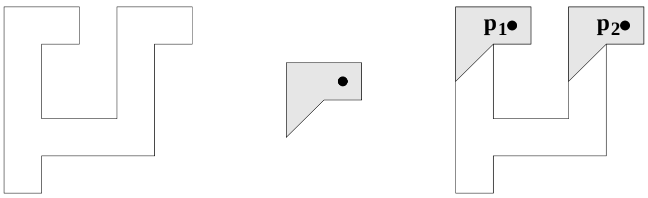
We break the general problem of localizing a robot into two phases as follows.

The Robot Localization Problem

Hypothesis Generation: Given P and V, determine the set H of all points $p_i \in P$ such that the visibility polygon of p_i is exactly V (i.e. $V(p_i) = V$).

Hypothesis Elimination: Devise a strategy by which the robot can correctly eliminate all but one hypothesis from H, thereby determining its exact initial location. Ideally, the robot should travel a distance as small as possible.

Consider the example illustrated in Figure 1. The robot knows the map polygon P of its environment and the visibility polygon V representing what it can "see" in the environment from its present location. It also knows that P and V should be oriented as shown. The black dot represents the robot's position in the visibility polygon. By examining P and V, the robot can determine that it is at either point p_1 or point p_2 in P, i.e. $H = \{p_1, p_2\}$. It cannot distinguish between these two locations because $V(p_1) = V(p_2) = V$. However, by traveling out into the "hallway" and taking another probe, the robot can determine its location precisely.



P **V** **V(p1) = V(p2)**

Figure 1: Given a map polygon P (left) and a visibility polygon V (center), the robot must determine which of the 2 possible initial locations p_1 and p_2 (right) is its actual location in P.

An optimal strategy for the hypothesis elimination phase would direct the robot to follow an optimal verification tour, defined as follows.

Definition. A *verification tour* is a tour along which a robot that knows its position *a priori* can travel to verify this information by probing and then return to its starting position. An *optimal verification tour* is a verification tour of minimum length d.

Since we do not assume *a priori* knowledge of which hypothetical location in H is correct, an optimal verification tour for the hypothesis elimination phase cannot be pre-computed. For this reason, we seek an interactive probing strategy to localize the robot. In each step of such a strategy, the robot sends out range sensors, receives back the visibility polygon of its present position, and from this information decides where to move next to make another probe. To be precise, the type of strategy we seek can be represented by a localizing decision tree, defined as follows.

Definition. A *localizing decision tree* is a tree consisting of two kinds of nodes and two kinds of weighted edges. The nodes are either sensing nodes (S-nodes) or reducing nodes (R-nodes), and the node types alternate along any path from the root to a leaf. Thus tree edges directed down the tree either join an S-node to an R-node (SR-edges), or join an R-node to an S-node (RS-edges).

- Each S-node is associated with a position defined relative to the initial position of the robot. The robot may be instructed to probe the environment from this position.
- Each of the R-nodes is associated with a set $H' \subseteq H$ of hypothetical initial locations that have not yet been ruled out. The root is an R-node associated with H, and each leaf is an R-node associated with a singleton hypothesis set.
- Each SR-edge has weight 0. Such an edge represents the computation that the robot does to rule out hypotheses in light of the information gathered at the S-node end of the edge. An SR-edge does not represent physical travel by the robot.
- Each RS-edge has an associated path defined relative to the initial location of the robot. This is the path along which the robot is directed to travel to reach its next sensing point. The weight of an RS-edge is the length of its associated path.

Since we want to minimize the distance traveled by the robot, we define the weighted height of a localizing decision tree as follows.

Definition. The weight of a root-to-leaf path in a localizing decision tree is the sum of the weights on the edges in the path. The *weighted height* of a localizing decision tree is the weight of a maximum-weight root-to-leaf path. An *optimal localizing decision tree* is a localizing decision tree of minimum weighted height.

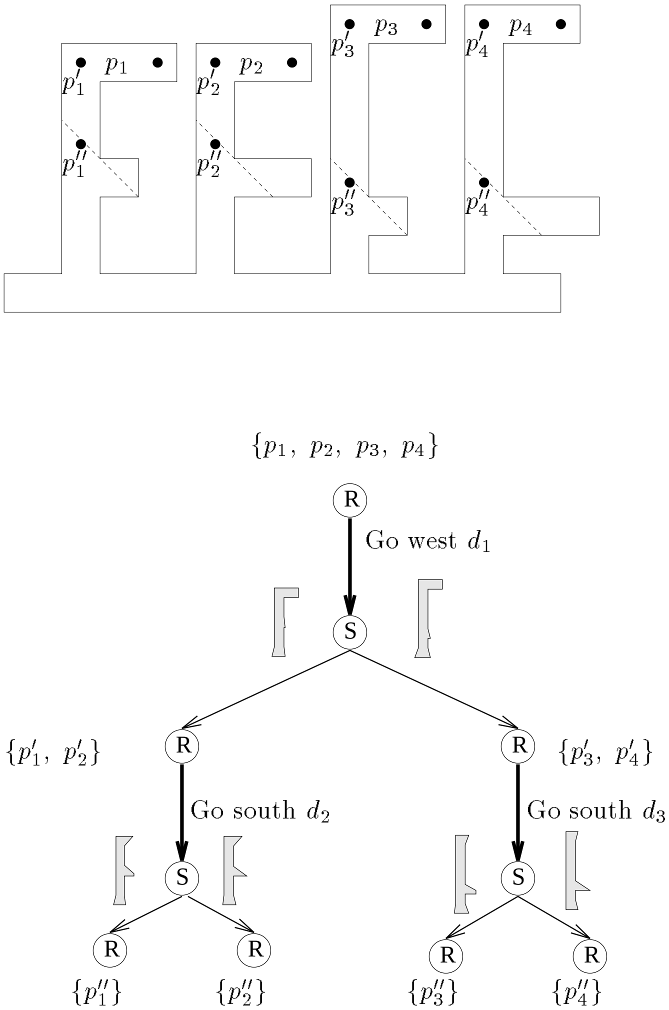


Figure 2: A map polygon and 4 hypothetical locations $\{p_1, p_2, p_3, p_4\}$ with a localizing decision tree for determining the true initial position of the robot.

We call a localization strategy that can be associated with a localizing decision tree a *localizing decision tree strategy*. As an example of such a strategy, consider the map polygon P shown in Figure 2. From the visibility polygon sensed by the robot at its initial location it is determined that the set of hypothetical locations is $H = \{p_1, p_2, p_3, p_4\}$. Hence the root of the localizing decision tree (also shown in Figure 2) is associated with H. In the figure, the SR-edges are labeled with the visibility polygons seen by the robot at the S-node endpoints of these edges, and the RS-edges are labeled with the path the robot should follow. Assuming that north points straight up, the strategy given by the tree directs the robot first to travel west a distance d_1, which is the distance between p_1 and p_1'. This positions the robot at one of p_1', p_2', p_3' or p_4'. The strategy then directs the robot to take another probe at its new location. This is represented by an S-node in the decision tree. Depending on the outcome of the probe, the robot knows it is located either at one of $\{p'_1, p'_2\}$ or at one of $\{p'_3, p'_4\}$. It then travels south either d_2 or d_3, to a position just past the dotted line segment shown in P, and takes another probe, which determines its unique location in P. The farthest that the robot must travel to determine its location is $d_1 + d_3$, so the weighted height of this decision tree is $d_1 + d_3$.

2.3 Previous work

Previous work on robot localization by Guibas, Motwani, and Raghavan in [GMR92] showed how to preprocess a map polygon P so that given the visibility polygon V that a robot sees, the set of points in P whose visible polygon of P is congruent to V, and oriented the same way, can be returned quickly. Their algorithm preprocesses P in $O(n^5 \log n)$ time and $O(n^5)$ space, and it answers queries in $O(m + \log n + A)$ time, where n is the number of vertices of P, m is the number of vertices of V, and A is the size of the output (the number of places in P at which the visibility polygon is V).

Theoretical work has also been done on navigating a robot in an unknown environment (see [BRS91, PY91]).

3 Hardness of Localization

In this section we show that the problem of constructing an optimal localizing decision tree, as defined in the previous section, is NP-hard. To do this, we first formulate the problem as a decision problem.

Robot-Localizing Decision Tree (RLDT)

INSTANCE: A simple polygon P and a star-shaped polygon V, both with a common reference direction, the set H of all locations $p_i \in P$ such that $V(p_i) = V$, and a positive integer h.

QUESTION: Does there exist a localizing decision tree of weighted height less than or equal to h that localizes a robot with initial visibility polygon V in the map polygon P, where H is the set of possible initial locations?

We show that this problem is NP-hard by giving a reduction from the Abstract Decision Tree problem, proven NP-complete by Hyafil and Rivest in [HR76]. The Abstract Decision Tree problem is stated as follows:

Abstract Decision Tree (ADT)

INSTANCE: A set $X = \{x_1, \ldots, x_k\}$ of objects, a set $\mathcal{T} = \{T_1, \ldots, T_n\}$ of subsets of X representing binary tests, where test T_j is positive on object x_i if $x_i \in T_j$ and is negative otherwise, and a positive integer $h' \leq n$.

QUESTION: Does there exist an abstract decision tree of height less than or equal to h', where the height of a tree is the maximum number of edges on a path from the root to a leaf, that can be constructed to identify the objects in X? Such a decision tree has a binary test at all internal nodes and an object at every leaf. To identify an unknown object, the test at the root is performed on the object, and if it is positive the right branch is taken, otherwise the left branch is taken. This procedure is repeated until a leaf is reached, which identifies the unknown object.

Theorem 1 *RLDT is NP-hard.*

Proof: Given an instance of ADT, we create an instance of RLDT as follows. We construct P to be a staircase polygon, with a stairstep for each object $x_i \in X$ (see Figure 3). For each stairstep we construct $n = |\mathcal{T}|$ protrusions, one for each test in $\mathcal{T}$ (see Figure 4). If test T_j is a positive test for object x_i, then protrusion T_j on stairstep x_i has an extra hook on its end (such as T_3, T_4, and T_n in Figure 4). The length of a protrusion is denoted by l and the distance between protrusions T_1 and T_n is denoted by d, where d and l are chosen so that $dh' < l$. The vertical piece between adjacent stairsteps is longer than $(2l + d)h'$, and the width w of each stairstep is much smaller than the other measurements. The polygon P has $O(nk)$ vertices, where $n = |\mathcal{T}|$ and $k = |X|$.

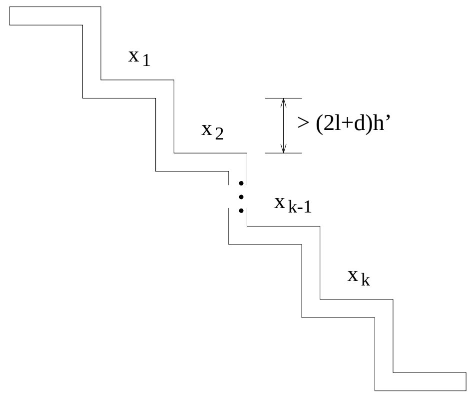


Figure 3: Construction showing localization is NP-hard

Consider a robot that is initially located at the shaded circle shown in Figure 4 on one of the k stairsteps.

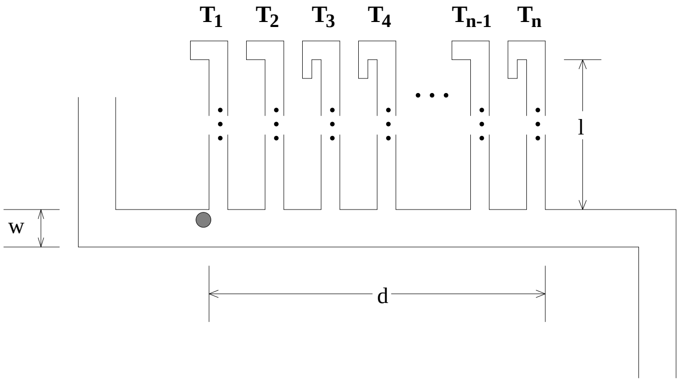


Figure 4: Close-up of a stairstep x_i in NP-hard construction. Not to scale: $l >> d >> w$.

The visibility polygon V at this point has $O(n)$ vertices and is the same at an analogous point on any internal stairstep x_i. We output the polygons P and V, which can be constructed in polynomial time, the k locations $p_i \in P$ such that $V(p_i) = V$, and weighted height $h = (2l + d)h'$ as an instance of RLDT.

Any localizing decision tree that examines protrusions on the stairstep corresponds to an equivalent abstract decision tree to identify the objects of X using tests in $\mathcal{T}$. To "perform" test T_j the robot must travel distance $2l$ to the end of protrusion T_j and back to see if it has a hook. It travels at most d in between tests. Therefore, if the robot can always localize itself by examining no more than h' protrusions, then it has a tree of weighted height no more than $h = (2l + d)h'$, which corresponds to an abstract decision tree of height h' for the ADT problem. Similarly, any abstract decision tree to identify the objects of X using tests in $\mathcal{T}$ corresponds to an equivalent localizing decision tree. □

4 Using Visibility Cells

In this section we discuss geometric issues involved in building a data structure for our localization strategy.

4.1 Cells and the overlay

When we consider positions where the robot can move to localize itself, we reduce the infinite number of locations in P to a finite number by first creating a visibility cell decomposition of P [BLM92, GMR92].

A *visibility cell* (or *visibility region*) C of P is a maximally connected subset of P with the property that any two points in C see the same subset of vertices of P ([BLM92]). A *visibility cell decomposition* of P is simply a subdivision of P into visibility cells. This decomposition can be computed in $O(n^3 \log n)$ using techniques in [BLM92]. The number of cells in this decomposition, as well as their total complexity, is $O(n^2 r)$ (see [GMR92]), where r is the number of reflex vertices[1] of P.

Although two points p and q in the same visibility cell C see the same subset of vertices of P, they may not have the same visibility polygon (i.e. it may be that $V(p) \neq V(q)$). This is because some edges of $V(p)$ may not actually lie on the boundary of P (these edges are collinear with p and are produced by visibility lines), so these edges may be different in $V(q)$. Therefore, we need a different structure to represent the portion of P visible to a point p in a visibility cell C. The structure that we use is the *visibility skeleton* of p.

Definition. The *visibility skeleton* $V^*(p)$ of a location $p \in P$ is the skeleton of the visibility polygon $V(p)$. That is, it is the polygon induced by the non-spurious vertices of $V(p)$, where a *spurious* vertex of $V(p)$ is one that lies on an edge of $V(p)$ that is collinear with p, and the other endpoint of this edge is closer to p. The edges of the skeleton are labeled to indicate which ones correspond to real edges from P and which ones are *artificial* edges induced by the spurious vertices. If p is outside P, then $V^*(p)$ is equal to the special symbol $\emptyset$.

For a complete discussion of visibility skeletons and a proof that $V^*(p) = V^*(q)$ for any two points p and q in the same visibility cell, see [BLM92, GMR92]. Because points in the same visibility cell have the same visibility skeleton, we use it as an abstraction of the sensory information received by the robot at a particular location.

As stated in Section 2, the hypothesis generation phase of the robot localization problem generates a set $H = \{p_1, p_2, \ldots p_k\} \subset P$ of *hypothetical locations* at which the robot might be located initially. The number k of such locations is bounded above by r (see [GMR92]). From this set H, we can select the first location p_1 to serve as an origin for a local coordinate system. For each location p_j, $1 \leq j \leq k$, we define the *translation vector* $t_j = p_j - p_1$ that translates location p_j to location p_1, and we define P_j to be the *translate* of P by vector t_j. That is, P_j is the translate of P in which p_j and p_1 are coincident (note that $P_1 = P$). We thus have a set $\{P_1, P_2, \ldots P_k\}$ of translates of P corresponding to the set H of hypothetical locations. Note that in each P_j, the point corresponding to the hypothetical location p_j is located at the origin.

[1] A *reflex vertex* of P is a vertex that subtends an angle greater than 180°.

In order to determine the hypothetical location corresponding to the true initial location of the robot, we construct an *overlay arrangement* A that combines the k translates P_j.

Definition. The *overlay arrangement* A for the map polygon P corresponding to the set of hypothetical locations H is obtained by taking the union of the edges of each translate P_j as well as the visibility edges in the visibility cell decomposition of P_j.

See Figure 5 for an example of an overlay arrangement. Since each visibility cell decomposition is created from $O(nr)$ lines introduced in the interior of P_j, a bound on the total number of cells in the overlay arrangement as well as their total complexity is $O(k^2n^2r^2)$, which may be $O(n^6)$. In fact, there are map polygons whose corresponding overlay arrangements for certain visibility polygons have $\Omega(n^5)$ cells.

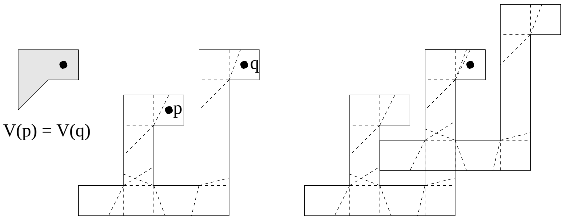


Figure 5: A visibility polygon, a map polygon and the corresponding overlay arrangement

4.2 The reference point set Q

Each cell in the overlay arrangement A represents a potential probe position, which can be used to distinguish between different hypothetical locations of the robot. For each cell C of A and for each translate P_j that contains C, there is an associated visibility skeleton $V_j^*(C)$. If two translates P_i and P_j have different skeletons for cell C, or if C is outside of exactly one of P_i and P_j, then C distinguishes hypothetical location p_i from p_j.

For our localization strategy we choose a set Q of *reference points* in A that can be used to distinguish between different hypothetical locations. For each cell C in A that lies in at least one translate of P, and for each translate P_j that contains C, let $q_{C,j}$ denote the point on the boundary of C that is closest to the origin (recall that translating P to P_j moves p_j to the origin). Here, the distance $d_j(q_{C,j})$ from the origin to the closest point in C is measured inside P_j. We choose $Q = \{q_{C,j}\}$. In the remainder of this paper we drop the subscripts from $q_{C,j}$ when they are not necessary.

Computing the reference points in Q involves computing Euclidean shortest paths in P_j from the origin to each cell C. To compute these paths we can use existing algorithms in the literature for shortest paths in simple polygons (see [GHL+87]), and we omit the details here. For each cell C we will have up to k reference points $\{q_{C,1}, \ldots, q_{C,k}\}$ and their corresponding distances $\{d_1(q_{C,1}), \ldots, d_k(q_{C,k})\}$. We define $d_j(q) = \infty$ for all points q not within P_j.

Partition of H

For each cell C we compute a *partition of* H that represents which hypothetical locations can be distinguished from one another by probing from inside C. If two translates P_i and P_j have the same visibility skeleton for cell C, then p_i and p_j are in the same subset of the partition of H corresponding to cell C. Also, if C is outside of both translates P_i and P_j, then p_i and p_j are in the same subset of the partition.

Although there may be $O(n^6)$ cells in the overlay arrangement A, yielding up to $O(kn^6)$ reference points, we show in Section 5.3 that only $O(k^2)$ reference points are needed for our localization strategy, so we do not need to compute a partition of H for all $O(n^6)$ cells.

5 A Greedy Strategy

In this section we present a localizing decision tree strategy, called *Strategy Q* , for completing the solution of the hypothesis elimination phase of the robot localization problem. Our strategy, which has a greedy flavor, is called Strategy Q because in choosing locations for probes, it uses the set Q of reference points described previously. Strategy Q will enable the robot to localize itself by traveling distance at most kd, where $k = |H|$ and d is the length of an optimal verification tour.

In devising a localizing decision tree strategy, there are two main criteria to consider when deciding where the robot should make the next probe: (1) the distance to the new probe position, and (2) the information to be gained at the new probe position. However, even a strategy that considers both criteria can do poorly. For example, if the robot employs an incremental strategy that at each step tells it to travel to the closest probe location that yields some information, then a map polygon can be constructed such that in the worst case the robot will travel distance 2^kd.

Using Strategy Q for hypothesis elimination, a strategy for the complete robot localization problem can be obtained as follows. Preprocess the map polygon P using a method similar to that in [GMR92]. This preprocessing yields a data structure that stores for each equivalence class of visibility polygons either the location in P yielding that visibility polygon, if there is only one location, or a localizing decision tree that tells the robot how to travel to determine its true initial location.

5.1 Strategy Q

In this subsection we present the details of Strategy Q . Using the results of Section 4, it is possible to pre-compute Strategy Q 's entire decision tree. However, we will describe the strategy by explaining how it directs the robot to behave. This amounts to describing the nature of a root-to-leaf path in the tree. This is easier to understand than the description of an algorithm to compute the entire decision tree. Also, in practice, it may sometimes be preferable *not* to pre-compute the entire tree, but rather to compute the robot's next move on an interactive basis, as the robot carries out the strategy.

Information used by Strategy Q

- The map polygon P.
- The set H generated in the hypothesis generation phase.
- The set Q of reference points defined in Section 4.2.
- For each point $q \in Q$ the distance $d_j(q)$ of q from the origin, measured within P_j.
- For each point $q \in Q$, a path $path_j(q)$ within P_j of length $d_j(q)$, which is defined by a series of relative motions that take the robot from the origin to q in P_j. Using the data structure in [GHL+87] these paths can be easily computed.
- For each point $q_{C,j} \in Q$, the partition of H associated with cell C, as defined in Section 4.2.

Next we describe how Strategy Q directs the robot to behave. Initially, the set of hypothetical locations is the given set H. As the robot carries out the strategy, hypothetical locations are eliminated from H. Thus in our description of Strategy Q , we abuse notation and use H to denote the shrinking set of *active hypothetical locations*; i.e. those that have not yet been ruled out. Similarly, we use Q to denote the shrinking set of *active reference points*; i.e. those that non-trivially partition the set of active hypothetical locations. We call a path $path_j(q)$ *active* if $p_j \in H$ and $q \in Q$ are both active.

Notation. Let $d_*(q*)$ denote the minimum of

$$\{\ d_j(q) \mid q \in Q \text{ and } p_j \in H \text{ are active } \}.$$

Let $path_*(q*)$ denote an active path of length $d_*(q*)$.

Strategy Q

Strategy Q directs the robot to travel along paths from the origin to points in the overlay arrangement A. Suppose that p_j is the true initial location of the robot. We will prove in the next subsection that Strategy Q only directs the robot to follow paths that are contained in translate P_j. Note that a path from the origin that is contained in P_j is analogous to a path in P from location p_j.

From the initial H and Q, an initial $path_*(q*)$ can be selected. The strategy directs the robot to travel along this path and to make a probe at its endpoint. The robot then uses the information gained at the probe position to update H and Q and to determine a new $q*$ and a new $path_*(q*)$ from the origin. The strategy then directs the robot to retrace its previous path back to the origin, and then to follow the new path to its endpoint, which is the next probe location. This process stops when the size of H shrinks to 1. At this point the initial location of the robot is determined, and the robot can, if desired, be directed to return to its initial location by retracing its last path.

5.2 A performance guarantee

The following theorems show that Strategy Q directs the robot along a path whose length compares favorably with the minimum verification length d. First we show that Strategy Q never directs the robot to pass through a wall. Then we show that Strategy Q eliminates all hypothetical locations except the valid one, and we establish an upper bound on the length of the path produced by Strategy Q . A corollary of Theorem 3 is that the localizing decision tree associated with Strategy Q has a weighted height that is at most $2k$ times the weighted height of an optimal localizing decision tree.

Theorem 2 *Strategy Q never directs the robot to pass through a wall.*

Proof: The proof is by contradiction. Suppose that p_j is the true initial location of the robot and x_j is the point on the boundary of P_j where the robot would first pass through a wall. Furthermore, suppose that when the robot attempts to pass through the wall at x_j, the path it has been directed to follow is $path_i(q)$.

Let C denote the cell of arrangement A (see Section 4.1) that contains the portion of $path_i(q)$ just before x_j.

Since cell C is contained in P_j, it contributes a reference point $q_{C,j}$ to the set Q of reference points.

It suffices to show that $q_{C,j}$ is active at the time Strategy Q chooses $path_i(q)$ for the robot to follow. This is because $d_j(q_{C,j}) \leq d_j(x_j)$ by definition of $q_{C,j}$; furthermore, $d_j(x_j) \leq d_i(x_j)$ since the portion of $path_i(q)$ from the origin to x_j is contained within P_j, and $d_j(x_j)$ is equal to the length of a shortest path in P_j from the origin to x_j; finally, $d_i(x_j) < d_i(q)$ because x_j is an intermediate point on $path_i(q)$. Chaining these inequalities together gives $d_j(q_{C,j}) < d_i(q)$. Hence Strategy Q would choose $path_j(q_{C,j})$ rather than $path_i(q)$ provided that $q_{C,j}$ is active at the time $path_i(q)$ is selected.

Now we show that $q_{C,j}$ is active when $path_i(q)$ is selected. Point $q_{C,j}$ is active if and only if the following two conditions hold: (1) p_j has not been eliminated from H and (2) the visibility skeleton associated with C distinguishes between at least two active hypothetical locations. Clearly condition (1) holds, since the correct hypothetical location is never eliminated from H. Condition (2) holds because the skeleton $V_j^*(C)$ associated with C relative to P_j has a real edge through the point x_j, whereas the skeleton $V_i^*(C)$ associated with C relative to P_i does not have a real edge through x_j. Therefore, the skeleton associated with points in C distinguishes between p_i and p_j, which are both active at the time $path_i(q)$ is chosen, so point $q_{C,j}$ is active. □

Theorem 3 *Strategy Q directs the robot along a path whose length is at most kd, where $k = |H|$ and d is the length of an optimal verification tour for the robot's initial position.*

Proof: Let p_t denote the true initial location of the robot. First we show that Strategy Q eliminates all hypothetical initial locations in H except p_t. Suppose the contrary is true. This means that the set Q of active reference points becomes empty before the size of H shrinks to one. Let p_i be an active hypothetical initial location different from p_t at the time Q becomes empty. Translates P_i and P_t are not identical, so there is some point x_t on the boundary of P_t that does not belong to the boundary of P_i. Let C be the cell of arrangement A contained in P_t and containing x_t. C distinguishes between p_i and p_t because the skeletons associated with C relative to P_i and P_t are not the same. Therefore $q_{C,i}$ and $q_{C,t}$ are still in the active set Q, a contradiction.

Next we establish the upper bound on the length of the path determined by Strategy Q . Because the strategy never directs the robot to a probing site that does not eliminate one or more elements from H, it requires the robot to make a trip from its initial location to some sensing point and back at most $k-1$ times.

We claim that each round trip has length at most d. To see this, we first consider how a robot traveling along an optimal verification tour L would rule out an arbitrary incorrect hypothetical location p_b. Then we consider how Strategy Q would rule out p_b.

Consider a robot traveling along tour L that eliminates each invalid hypothetical location at the first point x on L where the visibility skeleton of x relative to the invalid hypothetical location differs from the skeleton of x relative to P_t. Let w be the point on L where the robot rules out p_b. The point w must lie on the boundary of some cell C in the arrangement A that distinguishes p_b from p_t. Cell C generates a reference point $q_{C,t} \in Q$, which is the closest point of C to the origin, where distance is measured inside P_t, so $d_t(q_{C,t}) \leq d_t(w)$. Since p_t is the true initial location of the robot, the distance $d_t(w)$ is equal to or less than the distance along L of w from the origin, as well as the distance along L from w back to the origin. Putting these inequalities together, we deduce that the distance $d_t(q_{C,t})$ is equal to or less than half the length of L.

Since p_t is the true initial location of the robot, it is active at the moment Strategy Q directs the robot to move from the origin to the probing site where it eliminates p_b. Since p_b is about to be ruled out, it is also still active. That means that the reference point $q_{C,t}$ considered in the previous paragraph is still active, since it distinguishes p_b from p_t.

At this time Strategy Q directs the robot to travel along $path_*(q*) = path_j(q)$. By design, the length $d_*(q*) = d_j(q)$ of this path, which is the distance the robot will travel from the origin to the next probing position, is the minimum over all $d_i(q)$ for active $p_i \in H$ and $q \in Q$. In particular, since point $q_{C,t}$ is still active, $d_*(q*)$ is equal to or less than $d_t(q_{C,t})$. But as we have already seen, this latter distance is equal to or less than half the length of L. Therefore, Strategy Q directs the robot to travel from the origin to some probing position where the robot eliminates p_b and back, and the length of this loop is at most d. □

Note that if a verifying path is not required to return to its starting point, the bound for Theorem 3 becomes $2kd$. In this paper, we do not comment further on computation time as there are many ways to implement Strategy Q .

Corollary 4 *The weighted height of the localizing decision tree constructed by Strategy Q is at most $2k$ times the weighted height of an optimal localizing decision tree* *for the same problem.*

The bound given in Corollary 4 for the weighted height of the localizing decision tree built by Strategy Q is also a lower bound. That is, a map polygon P and a visibility polygon V can be given such that the weighted height of the localizing decision tree built by Strategy Q for P and V is $\Omega(k)$ times the weighted height of an optimal localizing decision tree.

5.3 A reduced set of reference points

The set Q of reference points may have size $O(kn^6)$. In this subsection, we show that when Strategy Q is run with only a subset $Q' \subseteq Q$ of size at most $k(k-1)$, the kd performance guarantee of Section 5.2 still holds.

Set Q' is defined as the union of subsets $Q_i \subseteq Q$, where there is one Q_i for each $p_i \in H$ and $|Q_i| \leq k-1$. Ignoring implementation issues, we define Q_i as follows. Initially Q_i is empty, and the subset of Q consisting of reference points $q_{C,i}$ generated for translate P_i is processed in order of increasing $d_i(q_{C,i})$. For each successive reference point $q_{C,i}$, the partition of H induced by $Q_i \cup \{q_{C,i}\}$ is compared to that induced by Q_i alone. If the subset of H containing location p_i is further subdivided by the additional reference point $q_{C,i}$, then $q_{C,i}$ is added to Q_i. Conceptually, the reference point $q_{C,i}$ distinguishes another hypothetical initial location from p_i. This process continues until p_i is contained in a singleton in the partition of H induced by Q_i. Since there are only $k-1$ initial locations to be distinguished from p_i, Q_i will contain at most $k-1$ points.

We denote by Strategy Q' the strategy obtained by replacing set Q with Q' in Strategy Q. The proof of the following theorem is similar to those of Theorems 2 and 3.

Theorem 5 *Strategy Q', which uses a set of at most $k(k-1)$ reference points, directs the robot along a path whose length is at most kd, where $k = |H|$ and d is the length of an optimal verification tour for the robot's initial position.*

6 Conclusion

We have shown that the problem of localizing a robot in a known environment by traveling a minimum distance is NP-hard, and we have given an approximation strategy that achieves a bound of k times an optimal solution, where k is the number of possible initial locations of the robot.

The work in this paper is one part of a strategy for localizing a robot. The complete strategy will preprocess the map polygon and store the decision trees for ambiguous initial positions so that the robot only needs to follow a predetermined path to localize itself.

There are many variations to this problem which can be considered. If the robot must localize itself in an environment with obstacles, then the map of the environment can be represented as a simple polygon with holes. In this paper we assigned a cost of zero for the robot to take a probe and analyze it. In a more general setting we would look for a minimum weighted height decision tree, where the edges of a decision tree associated with the outcome of a probe would be weighted with the cost to analyze that probe. A pragmatic variation of the problem would weight reference locations so that those that produce more reliable percepts would be selected first.

References

[BD90] K. Basye and T. Dean. Map Learning with Indistinguishable Locations. In M. Henrion L. N. Kanal J. F. Lemmer, ed., *Uncertainty in Artificial Intelligence 5*, pp. 331–340, 1990.

[BLM92] P. Bose, A. Lubiw, and J.I. Munro. Efficient Visibility Queries in Simple Polygons. In Cao An Wang, ed., *Proc. of the 4^{th} Canadian Conf. on Computational Geometry*, pp. 23–28, Aug. 1992. (Also P. Bose, Master's thesis, University of Waterloo, Dec. 1991.)

[BRS91] A. Blum, P. Raghavan, and B. Schieber. Navigating in Unfamiliar Geometric Terrain. In *Proc. of the 23^{rd} Annual ACM Symp. on Theory of Computing*, pp. 494–504, May 1991.

[DJMW93] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. Map Validation and Self-location in a Graph-like World. In *Proc. of the 13^{th} Intl. Conf. on Artificial Intelligence*, pp. 1648–1653, Aug. 1993.

[GHL+87] L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. Tarjan. Linear-Time Algorithms for Visibility and Shortest Path Problems Inside Triangulated Simple Polygons. *Algorithmica*, 2:209–233, 1987.

[GMR92] L. Guibas, R. Motwani, and P. Raghavan. The Robot Localization Problem in Two Dimensions. In *Proc. of the 3^{rd} Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 259–268, Jan. 1992.

[HR76] L. Hyafil and R. Rivest. Constructing Optimal Binary Decision Trees is NP-Complete. *Information Processing Letters*, 5(1):15–17, May 1976.

[KMK93] A. Kosaka, M. Meng, and A.C. Kak. Vision-guided Mobile Robot Navigation Using Retroactive Updating of Position Uncertainty. In *Proc. of the Intl. Conf. of Robotics and Automation, Volume 2*, pp. 1–7, May 1993.

[PY91] C. Papadimitriou and M. Yannakakis. Shortest Paths without a Map. *Theoretical Computer Science*, 84:127–150, 1991.

[TA92] R. Talluri and J.K. Aggarwal. Position Estimation for an Autonomous Mobile Robot in an Outdoor Environment. *IEEE Trans. on Robotics and Automation*, 8(5):573–584, Oct. 1992.