

© Copyright by Jason Matthew O'Kane, 2005

ALMOST-SENSORLESS LOCALIZATION

BY

JASON MATTHEW O'KANE

B.S., Taylor University, 2001

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2005

Urbana, Illinois

Acknowledgment

I offer sincere thanks to my family for teaching me the value of education and encouraging my progress at every step. Many teachers and professors have also been major influences in my development as computer scientist and as a person. Special, lexicographically-ordered thanks are due to Leon Adkison, Felix Aguilar, Stefan Brandle, Tim Diller, George Lepsch, Richard Malay, Anna Marie Nucci, Frau Diana O'Donnell, Janet Robb, Wally Roth, Bill Toll, and Art White. I am indebted in various ways to each of my present-day colleagues Peng Cheng, Hamid Chitsaz, Steve Lindemann, Benjamin Tovar, Anna Yershova, and George Zaimis. Finally, my advisor Steve LaValle is to be particularly thanked for his insight, guidance, and wisecracking.

Abstract

Localization is a fundamental problem for many kinds of mobile robots. A variety of sensor systems of varying ability have been proposed and successfully used to address the problem. This thesis probes the lower limits of this range by describing an extremely simple robot with which localization is still possible. More precisely, it presents a localization method for a robot equipped with only a compass, a contact sensor, and a map of the environment. In this context, a localization strategy can be described as a sequence of directions in which the robot moves until it comes into contact with the environment boundary. The main contribution is to show that a localizing sequence exists for any simply connected polygonal environment by presenting an algorithm to compute such a sequence. An implementation with simulated examples is presented. We also show that the sensing model is minimal by proving that in any simply-connected polygonal environment, replacing the compass with an angular odometer precludes the possibility of performing localization.

Table of Contents

List of Figures	vi
List of Tables	vii
Chapter 1 Introduction	1
1.1 Organization	2
1.2 Related work	2
1.2.1 Localization	2
1.2.2 Minimalist robotics	4
Chapter 2 Problem Statement	6
2.1 Robot model	6
2.2 Problem formalization	6
2.3 Localization as a search in information space	8
Chapter 3 An Algorithm to Generate Localizing Sequences 10	10
3.1 The information transition function	10
3.2 Generating localizing sequences	12
3.2.1 From all of ∂X to a finite subset	13
3.2.2 From a finite subset to a single point	15
3.3 Computed examples	20
Chapter 4 Minimality of the Model	36
Chapter 5 Discussion	38
References	40

List of Figures

3.1	Computing $F(\eta, u)$ by a line sweep algorithm.	12
3.2	(a) A motion along \overline{ab} will collapse \overline{ab} to a single point. (b) No motion not parallel to \overline{ab} will collapse \overline{ab}	14
3.3	If p_k can see q_k , then a motion in the direction of $\overline{p_k q_k}$ maps p_k and q_k to the same place.	16
3.4	(a) A visibility polygon. Spurious edges are dashed. (b) The shortest path to any point not in the visibility polygon begins with a motion in the direction of a spurious edge.	17
3.5	(a) The spurious edge $\overline{t_k v_k}$ hides p_k from q_k . (b) The point q_{k+1} cannot cross $\overline{t_k v_k}$ because its motion is parallel to $\overline{t_k v_k}$	17
3.6	The special case when $\overline{t_k v_k}$ is a bitangent.	19
3.7	Execution traces of the localization sequence depicted in Table 3.3 for 6 different starting positions. For each starting position, the final position is the lower right corner of the environment.	35

List of Tables

2.1	A localizing sequence for a simple non-convex polygon. Possible states at each step are shaded.	7
3.1	A localizing sequence computed by our algorithm for a highly symmetric environment.	21
3.2	A modified version of the environment from Table 3.1 in which the symmetries have been broken. Our algorithm generates a 28 step localizing sequence for this environment.	22
3.3	An irregular environment for which the localizing sequence computed by our algorithm requires 30 steps.	27

Chapter 1

Introduction

Localization, the task of systematically eliminating uncertainty in the pose of a robot, is a fundamental problem for nearly any autonomous mobile robotic system. It has long been known that robot designs with simplified sensing and actuation models can lead to decreased costs and increased robustness [60]. This thesis applies this idea to the problem of localization in an attempt to describe the simplest possible robot with which localization is still possible. This work has potential applications for extremely low-cost robots in industrial and residential settings. In addition, it is of basic scientific interest to explore the minimum sensing requirements for robotic tasks.

We propose a robot model in which only a map, a compass and a contact sensor are available. Odometry, range sensing and wall-following abilities are omitted. With such a robot, the only reliable courses of action are to select a motion direction and move in that direction as far as possible. During any execution, the robot can never gather any new information from sensors about its position within the environment. It must instead rely on actions that are conformant in the sense that they map multiple possible current states to a single resulting state. A plan in this model is an action sequence rather than a decision tree.

This thesis presents two main results. First, we give an algorithm that accepts as input a description of a simply-connected polygonal environment and outputs a sequence of actions that will localize a robot that starts at some unknown position within the environment. The correctness of this algorithm constitutes a constructive proof that a localizing sequence exists for any such environment. Second, we argue for the minimality of the robot model by addressing the need for each of the map, contact sensor, and compass.

A preliminary version of this work appeared in [47].

1.1 Organization

The balance of this thesis is organized as follows. The remainder of this chapter will review related work. Chapter 2 formalizes our robot model, defines the localization problem and characterizes this problem as a search through an abstract space of information states. An algorithm to solve this problem is in Chapter 3. Chapter 4 addresses the minimality of the robot model. Some concluding remarks appear in Chapter 5.

1.2 Related work

There are two primary lines of antecedent research: First, a diverse collection of works have studied the localization problem itself on theoretical and practical levels. Second, a recurring theme over two decades of robotics research has been the notion of *minimalism*, the idea that simple but carefully designed robotic systems can offer advantages in cost, efficiency and robustness over more complex systems that are richer in sensors and actuators.

1.2.1 Localization

Much attention has been given to the problem of localization for robots with varying degrees of sensing capability. We can generally separate the research on this topic into two flavors: *passive localization*, which concentrates on using any information available to the robot to draw conclusions about its position, and *active localization*, in which the goal is to prescribe motions for the robot in order to fully determine its position.

Passive localization Because it is both extremely practical and algorithmically interesting, the dominant sensor model for localization research is that of range sensing. A *range sensor* provides as input to the robot the distance to the nearest obstacle in each direction. This information can be used to compute the *visibility region* of the robot's position, which contains every point in the environment reachable by a single straight-line motion. The static problem of finding the set of candidate locations for a given visibility region in a polygonal environment was solved in [32]. In [21], an algorithm for candidate generation is given that places stronger emphasis on robustness to missing and spurious range data, as would be required for experimental mobile robotics.

Another large body of work has focused on localization using *landmarks*. In [52], a problem is posed in which the environment contains a collection

of landmark objects in fixed locations. At any time, the robot’s sensors can detect some subset of these landmarks. The robot is aware of the direction (but not distance) to each of these detected landmarks. The problem of finding the set of points in the environment consistent with this sensor data is solved for the case where the landmarks are distinguishable in [11]; the distinguishability requirement is relaxed in [10]. One might also consider the problem of “landmark design” in which landmarks can be placed in locations in the environment carefully selected to facilitate localization. One possible realization of this idea is to strategically place reflectors along the walls of the environment and equip the robot with a sensor that can detect the orientations of each reflector in the robot’s visibility region [52]. An algorithm for computing a placement of reflectors such that no two points in the environment have identical reflector signatures appears in [23]. The method of [41] is also essentially landmark-based, but the landmarks are wireless ethernet base stations whose signal strength informs the robot’s position estimate.

Finally, a large family of methods use probabilistic models to estimate the current state [22, 33, 41, 42, 49, 54, 55, 59]. Generally these methods employ a probabilistic model for the robot’s position given sensor data (i.e. range data, odometry, landmark positions, etc.) to form a probability distribution over states in the environment that represents the robot’s “belief” about its current location.

Active localization We now turn to methods that, rather than only reasoning about uncertainty in the robot’s position, also generate motion plans to reduce or eliminate this uncertainty. Algorithms in this context are often considered in an online sense and are evaluated in terms of their *competitive ratio* [45, 51], which compares the lengths of paths generated by the algorithm to the length of the shortest possible path that could have been selected if the robot started with full information.

In [38], the environment is constrained to an embedding of a bounded-degree acyclic graph into \mathbb{R}^n with sensing limited to the orientations of incident edges. This algorithm has competitive complexity $O(n^{2/3})$, in which n is the number of leaves in the graph. Later improvements [50] shaved this to $O(n^{1/2})$, which is known to be optimal up to a constant factor [25]. Also addressed in [38] is the case where the robot can move among a collection of non-intersecting open axis-aligned rectangles in the plane; this problem is solved with a $O\left(n\sqrt{\frac{\log n}{\log \log n}}\right)$ -competitive algorithm.

More generally, the problem of computing a localization strategy that

minimizes the worst-case distance traveled by a robot equipped with a visibility sensor was proved NP-hard in [25]. In this case, a localization strategy has the form of a decision tree with branches at points where two or more candidate positions are disambiguated. The hardness proof proceeds by reduction from the Abstract Decision Tree problem [34]. The optimal decision tree can, however, be approximated and [25] gives an algorithm based on the visibility-cell decomposition that does this. An important weakness of this algorithm is that it relies on motion commands that direct the robot into visibility cells that may be arbitrarily small. In [48], this difficulty is addressed by introducing randomization.

1.2.2 Minimalist robotics

Robots are composed, essentially by definition, of sensors and actuators that interact with the physical world. Both sensors and actuators are subject in practice to significant errors in precision and accuracy. Effective robots must, in some way, be robust to these errors. One solution paradigm is to design more and more complex robots with richer sensor sets and actuators with greater freedom, and then take advantage of these additional capabilities through careful algorithm design and programming. Starting, perhaps, with Whitney’s critique of mid-1980’s robotics research [60], an alternative approach has arisen in which these difficulties are dealt with by designing extremely simple robots that exploit the compliant properties of the system in question to execute their assigned tasks. In industrial settings, more complex tasks can be solved by sequences of these simple robots [17]. Other work has explored the more general question of the minimal sensing requirements to complete a given task [13, 24]. The whole of this approach has been called *minimalist robotics*.

Universal traversal sequences On a purely abstract level, we may think of the universal traversal sequences that arise in graph theory [7, 8] as a minimalist approach. Let g denote a d -regular graph. For each vertex of g , we may bijectively label the incident edges with the labels $\{1, \dots, d\}$. Then, fixing a start vertex v , a string $s \in \{1, \dots, d\}^*$ can be considered a path in g simply by following edges in the indicated order. If s visits every vertex in g , then we call s a *traversal sequence* for g starting at v . Now consider the family $\mathcal{G}(n, d)$ of all connected n -vertex d -regular graphs. A sequence s is a (n, d) -*universal traversal sequence* if s , for each $g \in \mathcal{G}(n, d)$ and each $v \in g$, s is a traversal sequence for g starting at v .

Universal traversal sequences can be considered as solutions to planning

problems with uncertainty both in environment space (that is, the selection of g from $\mathcal{G}(n, d)$) and in state space (that is, the selection of a start vertex $v \in V(g)$). More concretely, observe that $\mathcal{G}(n, 4)$ contains all grid-like environments in the plane with n unoccupied cells, so that an $(n, 4)$ -universal traversal sequence will visit every square of any n -element planar grid.

Borodin *et al.* [15] give several lower bounds on the lengths of universal traversal sequences. In [9], the problem is addressed for complete graphs. Bounds for other special cases appear in [18, 35, 53].

Manipulation Some of the most effective systems for *manipulation*, the problem of using robots to move unarticulated objects, have used a minimalist approach. Akella and Mason [6] give a complete planner for pushing objects on a planar surface while avoiding obstacles. A broader focus of research has been on part-orienting systems, in which a stream of identical parts with unknown initial orientation are manipulated into some known final orientation. This has been accomplished with limited sensing using tilting trays [27, 30], parallel-jaw grippers [28, 29], vibratory bowl feeders [3, 12, 14] and active [4, 5] or passive [16, 61] fences over constant-speed conveyor belts. Many of these methods are surveyed in [58]. For some of these cases, the problem of planning to orient parts can be reduced to that of finding a sequence that resets a finite state machine from an unknown initial state to a known final state [26].

Exploration and navigation Finally, others have considered certain navigation and exploration tasks for mobile robots with minimal sensing. An analysis of the basic requirements for navigation in an unknown three-dimensional environment appears in [39, 40]. Bug algorithms [36, 37, 43, 44] are used for navigation by robots capable only of moving toward obstacles and following walls. In [56, 57], the robot has an extremely crude range sensor that can only detect discontinuities in depth information. As the robot explores its environment, this information is used to construct a data structure that allows for optimal navigation between previously-visited locations. More explicit maps based on metric measurements can be built with a range sensor by traversing the generalized Voronoi graph of the environment boundaries [1, 2, 20, 46].

Chapter 2

Problem Statement

In this chapter, we formally define a localization problem for a particular model of robot with extremely weak sensing capability. We also present by way of definition the primary machinery (the information space) used in Chapter 3 to solve this localization problem.

2.1 Robot model

A robot, equipped with a compass and a contact sensor, moves in some environment. In the absence of odometry, the only reliable actions for the robot are maximal linear motions. That is, the robot can select a direction and use its compass to move reliably in that direction as far as the environment allows. Importantly, the robot cannot gather any information about its position within the environment as a result of taking an action; the only information available to the robot is a set of possible initial positions and the history of selected actions.

2.2 Problem formalization

This section formalizes the abstract robot model we have described. Allow a point robot to move in a compact simply-connected polygonal state space X . Let ∂X denote the boundary of X . Observe that $\partial X \subset X$ because X is closed. The robot is consequently allowed to come in contact with the walls of the environment. The robot has access to an accurate map of X , including its orientation in the plane.

The robot's action space $U = S^1$ is the unit circle, denoting the set of directions in the plane. We will represent elements of U as unit vectors in \mathbb{R}^2 . Given a state $x \in X$ and an action u , the resulting state is governed by the state transition equation $f : X \times U \rightarrow X$, in which (x, u) maps to the

Table 2.1: A localizing sequence for a simple non-convex polygon. Possible states at each step are shaded.

i	u_i	η_{i+1}
0		
1		
2		
3		

opposite endpoint of the maximal segment in X starting at x and having direction u . We also define an iterated version of f to denote the result of a sequence of actions:

$$f^k(x, u_1, \dots, u_k) = f(\dots f(f(x, u_1), u_2) \dots), u_k). \quad (2.1)$$

Now we can define the notion of a solution.

Definition 1 A **localizing sequence** for X is a sequence of actions u_1, \dots, u_K such that there exists $x_f \in X$ with $f^K(x, u_1, \dots, u_K) = x_f$ for all $x \in \partial X$.

The intuition is that regardless of the robot's initial location within ∂X , after executing a localizing sequence, the robot's final position, x_f , is certain. The localizing sequence eliminates the uncertainty in the robot's state.

Table 2.1 shows a sample environment and a localizing sequence for it. The task, then, is to design an algorithm that accepts as input a description of X and outputs a localizing sequence for X .

Localizing sequences are distinguished from the decision trees that arise in some forms of sensor-based localization in that every $x \in X$ must map to the same x_f , rather than allowing different initial states to reach distinct but known final states. This change is a direct result of the lack of feedback in our localization strategies.

Also, note that although this definition only considers points on the boundary of X as possible initial states, a localizing sequence that works for any initial state in the interior of X can be created by prepending an arbitrary initial action (which necessarily will reach ∂X) to a localizing sequence as according to this definition.

2.3 Localization as a search in information space

The problem of finding a localizing sequence for a given environment can be seen as a planning problem in which the initial state is unknown and the current state is unobservable. To manage this uncertainty, we transform the problem from an unobservable planning problem in state space to an observable problem in a more complex space called the robot’s information space, which we now define.

At each step, the next action selected by the robot must be based solely on its map of the environment and the history of actions it has taken so far. This action sequence can be used to rule out certain elements of ∂X as possible positions for the robot. The set of positions consistent with this action sequence is called the robot’s information state. The next definition makes this idea more precise.

Definition 2 *Suppose the robot has executed some sequence of actions u_1, \dots, u_{k-1} . The **information state** η_k of the robot is*

$$\eta_k = \{x \in \partial X \mid \exists x_I \in \partial X, x = f^{k-1}(x_I, u_1, \dots, u_{k-1})\}. \quad (2.2)$$

Definition 3 *The **information space** \mathcal{I} is the set $2^{\partial X}$ of all information states, in which 2^S denotes the power set of S .*

We can view the problem of computing a localizing sequence for X as a planning problem in \mathcal{I} with initial state ∂X and goal region

$$\mathcal{I}_G = \{\eta \in \mathcal{I} \mid |\eta| = 1\}, \quad (2.3)$$

in which $|\cdot|$ denotes the (possibly infinite) cardinality of a set.

It is possible to define a transition function for information states in a very natural way. Let $F : \mathcal{I} \times U \rightarrow \mathcal{I}$ according to the forward projection

$$F(\eta, u) = \{x' \in \partial X \mid \exists x \in \eta, x' = f(x, u)\}. \quad (2.4)$$

As with f , we define

$$F^k(\eta, u_1, \dots, u_k) = F(\dots F(F(\eta, u_1), u_2) \dots), u_k). \quad (2.5)$$

In this notation, an action sequence u_1, \dots, u_K is a localizing sequence if and only if

$$|F^K(\partial X, u_1, \dots, u_K)| = 1. \quad (2.6)$$

Our algorithm presentation in Chapter 3 will take this view of localizing sequences.

Chapter 3

An Algorithm to Generate Localizing Sequences

This chapter presents a solution to the problem posed in Chapter 2. The presentation will proceed in two parts. First, in Section 3.1 we give an algorithm for computing the information transitions defined by (2.4). This algorithm is used in Section 3.2 as a subroutine for our main algorithm, which computes the localizing sequences.

3.1 The information transition function

This section presents a simple algorithm for computing $F(\eta, u)$ given X , η and u . We restrict our attention to information states that can be reached from the initial state $\eta_1 = \partial X$. Alg. 3.1 summarizes the algorithm, which is justified by the next two lemmas.

Consider an information state η that can be expressed as the union of a finite collection s_1, \dots, s_l of open segments and a finite set of points p_1, \dots, p_m on ∂X . To be precise, each s_i is a linear subset of ∂X not containing its endpoints. Each s_i need not be a complete edge of ∂X and since it is linear, cannot contain any vertex of ∂X . Without loss of generality, assume that the s_i 's are pairwise disjoint.

The next lemma characterizes the set of reachable information states.

Lemma 4 *Every information state η reachable from ∂X by an action sequence u_1, \dots, u_k can be expressed as a finite union of open segments and points on ∂X .*

Proof: Use induction on k . When $k = 0$, $\eta = \partial X$, which is the union of the vertices and edges bounding X . Assume inductively that η_{k-1} can be expressed as a finite union of open segments and points. Because F maps

Algorithm 3.1 INFOTRANS($X, s_1, \dots, s_l, p_1, \dots, p_m, u$)

```

 $\eta' \leftarrow \emptyset$ 
for  $i \leftarrow 1 \dots l$  do
   $\{a, b\} \leftarrow$  endpoints of  $s_i$ 
   $E \leftarrow$  vertices of  $X$ 
   $E \leftarrow E - \{v \in E \mid \text{CCW}(a, a + u, x) = \text{CCW}(b, b + u, x)\}$ .
   $E \leftarrow$  SORT( $E$ ) by perpendicular distance from  $\overleftarrow{a(a+u)}$ 
   $p \leftarrow$  SHOOTRAYFORSWEEP( $X, a, u, b - a$ )
  for  $e \in E$  do
    if SAMEEDGE( $p, e$ ) then
       $\eta' \leftarrow \eta' \cup \overline{pe}$ 
       $p \leftarrow$  SHOOTRAYFORSWEEP( $X, e, u, b - a$ )
    else
       $p' \leftarrow$  SHOOTRAYFORSWEEP( $X, e, u, b - a$ )
       $\eta' \leftarrow \eta' \cup \overline{pp'}$ 
       $p \leftarrow p'$ 
    end if
  end for
end for

for  $j \leftarrow 1 \dots m$  do
   $\eta' \leftarrow \eta' \cup \{\text{SHOOTRAY}(X, p_j, u)\}$ 
end for

return  $\eta'$ 

```

each segment to a finite set of polygonal chains on ∂X and each point to another single point, η_k also has a representation as a finite set of points and segments. \square

Lemma 5 For any action $u \in U$ and any reachable information state

$$\eta = \left[\bigcup_i s_i \right] \cup \left[\bigcup_j \{p_j\} \right],$$

in which the s_i 's are open segments and the p_j 's are points in ∂X ,

$$F(\eta, u) = \left[\bigcup_i F(s_i, u) \right] \cup \left[\bigcup_j \{f(p_j, u)\} \right]. \quad (3.1)$$

Proof: Immediate from the definition of F (Eq. 2.4). \square

The significance of Lemma 5 is that to compute any transition $F(\eta, u)$ from a reachable η , it will be sufficient to give an algorithm for the cases where η is an open segment and a single point. Then the complete $F(\eta, u)$ can be formed by unioning each of these partial results.

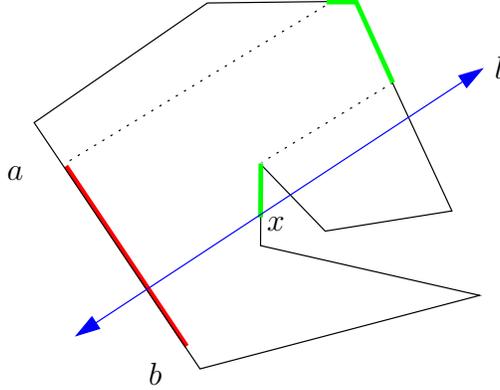


Figure 3.1: Computing $F(\eta, u)$ by a line sweep algorithm.

- If η is a segment \overline{ab} , where the notation \overline{ab} denotes the open segment with endpoints a and b , sweep a line l perpendicular to \overline{ab} starting at a and moving toward b . Maintain as an invariant that the nearest point $x \in \partial X$ to \overline{ab} intersected by l is known. Qualitative changes to x will occur only when l reaches a vertex of ∂X . At each such event, a segment is generated in $F(\eta, u)$ corresponding to the segment swept by l since the last event. An updated value for x can be computed by a modified ray shooting query, in which the ray stops at boundary vertices for which both incident edges are beyond l . Fig. 3.1 illustrates the sweeping algorithm.
- If η is a single point p_1 , then $F(\eta, u) = f(p_1, u)$ can be computed by a ray-shooting query in X from p_1 in direction u . In a simple polygon, data structures are known to answer such queries in $O(\log n)$ time, with $O(n)$ preprocessing time and $O(n)$ space [19].

The preceding exposition has established the correctness of the Alg. 3.1. Its run time is determined by the description length of η and the complexity of planar ray shooting.

Theorem 6 *Alg. 3.1 runs in time $O((m + nl) \log n)$ to compute the transition from an information state described by m points and l segments in an environment with n vertices.*

3.2 Generating localizing sequences

We now present an algorithm to compute a localizing sequence for any simply-connected polygonal environment X . The algorithm proceeds in two

Algorithm 3.2 LOCALIZINGSEQUENCE(X)

```
 $\eta_1 \leftarrow \partial X$   
 $k \leftarrow 1$   
while  $\eta_k$  contains at least one segment do  
   $\overline{ab} \leftarrow$  leftmost segment in  $\eta_k$   
  if  $(a - b).x > 0$  then  
     $u_k \leftarrow (a - b) / \|a - b\|$   
  else  
     $u_k \leftarrow (b - a) / \|b - a\|$   
  end if  
   $\eta_{k+1} \leftarrow \text{INFOTRANS}(X, \eta_k, u_k)$   
   $k \leftarrow k + 1$   
end while  
  
while  $\eta_k$  contains at least two points do  
  Select  $p, q$  from  $\eta_k$ .  
   $p_k \leftarrow p, q_k \leftarrow q$   
  while  $q_k \notin \text{Vis}(p_k, X)$  do  
     $t_k \leftarrow$  first vertex of shortest path from  $p_k$  to  $q_k$   
     $u_k \leftarrow (t_k - p_k) / \|t_k - p_k\|$   
     $\eta_{k+1} \leftarrow \text{INFOTRANS}(X, \eta_k, u_k)$   
     $p_{k+1} \leftarrow \text{SHOOTRAY}(X, p_k, u_k)$   
     $q_{k+1} \leftarrow \text{SHOOTRAY}(X, q_k, u_k)$   
     $k \leftarrow k + 1$   
  end while  
   $u_k \leftarrow (q_k - p_k) / \|q_k - p_k\|$   
   $\eta_{k+1} \leftarrow \text{INFOTRANS}(X, \eta_k, u_k)$   
   $k \leftarrow k + 1$   
end while  
  
return  $(u_1, \dots, u_{k-1})$ 
```

parts. First, actions are selected which reduce the uncertainty in the robot's position to a finite set of possibilities. Second, additional actions are chosen to reduce the uncertainty from this finite set to a single point. The complete localizing sequence u_1, \dots, u_K is divided into two parts u_1, \dots, u_{K_1} and $u_{K_1+1}, \dots, u_{K_2}$ generated by the respective parts of the algorithm. The complete algorithm is shown in Alg. 3.2; the subsequent discourse will explain and justify it.

3.2.1 From all of ∂X to a finite subset

This section presents a sweep line algorithm for computing a sequence of actions to reduce the robot's information state to a finite set of points. The following lemma, whose intent is illustrated in Fig. 3.2, provides the basis for the algorithm.

Lemma 7 *For any segment $s = \overline{ab} \subset X$, $F(s, u)$ is a single point if and*

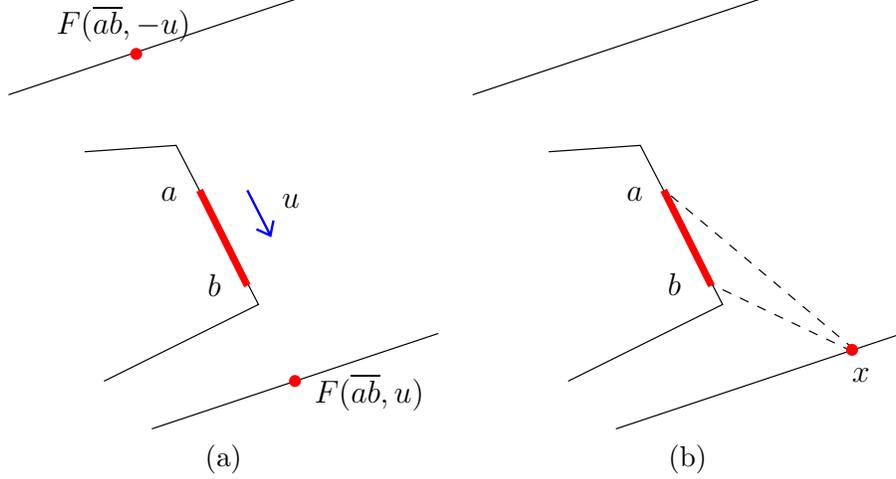


Figure 3.2: (a) A motion along \overline{ab} will collapse \overline{ab} to a single point. (b) No motion not parallel to \overline{ab} will collapse \overline{ab} .

only if $u = (a - b)/\|a - b\|$ or $u = (b - a)/\|b - a\|$.

Proof: For the forward part, note that since \overline{ab} is contained in X and is therefore itself collision-free, the maximal collision-free segment starting from each $x \in \overline{ab}$ will be the same. Hence each $x \in \overline{ab}$ maps to the same point under f . For the backward part, suppose u is not parallel to \overline{ab} and $F(\overline{ab}, u)$ is a single point x . Then a , b , and x form a nondegenerate triangle. This is in itself a contradiction because by definition of f , we must have \overline{ax} parallel to \overline{bx} . \square

Starting with $\eta_1 = \partial X$, the algorithm maintains a “current” information state η_k and a sequence of actions u_1, \dots, u_{k-1} mapping η_1 to η_k . Computation proceeds by sweeping a vertical line l from left to right across X , maintaining the invariant that η_k has no segments on the left side of l . Each time l reaches the endpoint of a segment \overline{ab} in η_k , it selects as u_k whichever of $(a - b)/\|a - b\|$ and $(b - a)/\|b - a\|$ has nonnegative x coordinate. The resulting $\eta_{k+1} = F(\eta_k, u_k)$ maintains the sweep invariant because the x -component of the motion of each segment in η_k is positive; hence, no segment can cross l . When l passes the rightmost vertex of X , it is certain that no segments remain in η_k . It remains to show that this method generates a plan of finite length.

Lemma 8 *The above algorithm generates $K_1 = O(n^3)$ actions for an environment with n edges.*

Proof: Let e_1, \dots, e_n denote the edges of ∂X and let $v(e_i)$ denote a unit vector in direction of e_i oriented so that its x component is positive. For a fixed i and j , $F(e_i, v(e_j))$ is a set of polygonal chains on ∂X with total complexity $O(n)$. Let R_{ij} denote the set of endpoints of segments in $F(e_i, v(e_j))$ and let $R = \bigcup_{i,j} R_{ij}$. Observe that $|R| = O(n^3)$. Clearly every segment s reached by l will correspond to the initial condition η_1 or to some transition from another edge. There are n segments in η_1 and R describes a set of earliest possible points at which an information state segment may begin on any edge. These events are sufficient to maintain the sweep invariant, so $K_1 = O(n) + O(n^3) = O(n^3)$. \square

3.2.2 From a finite subset to a single point

The previous section showed how to select actions u_1, \dots, u_{K_1} that map $\eta_1 = \partial X$ to a finite set $\eta_{K_1} = \{p_1, p_2, \dots, p_m\}$ of points. It remains to generate additional actions $u_{K_1+1}, \dots, u_{K_2}$ mapping $\{p_1, p_2, \dots, p_m\}$ to a single point. We will derive this part of the algorithm by reduction to the special case when $m = 2$. The more general problem for m points can be solved by iterating the algorithm for two points.

Let $\eta = \{p, q\}$. The ordering of the points is arbitrary but must be fixed. Our goal is to design a sequence of actions $u_{K_1+1}, \dots, u_{K_2}$ such that

$$f^{K_2-K_1}(p, u_{K_1+1}, \dots, u_{K_2}) = f^{K_2-K_1}(q, u_{K_1+1}, \dots, u_{K_2}). \quad (3.2)$$

That is, we want to design a sequence of actions mapping p and q to the same destination.

For $K_1 < k \leq K_2$, let

$$p_k = f^{K-K_1}(p, u_{K_1+1}, \dots, u_k)$$

and likewise

$$q_k = f^{K-K_1}(q, u_{K_1+1}, \dots, u_k).$$

Our algorithm will select u_k using only p_k and q_k . We begin with the simple base case:

Lemma 9 *If $\overline{p_k q_k} \subset X$, then the action $u = (q_k - p_k)/\|q_k - p_k\|$ is a localizing sequence for $\{p_k, q_k\}$.*

Proof: Follows directly from Lemma 7. \square

The intuition is that if p_k can “see” q_k in the sense that there is an unobstructed path between them, then a motion in the direction of this

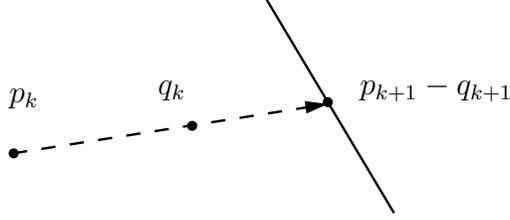


Figure 3.3: If p_k can see q_k , then a motion in the direction of $\overline{p_k q_k}$ maps p_k and q_k to the same place.

path will map both p_k and q_k to the same place. Fig. 3.3 illustrates this situation.

Now suppose $\overline{p_k q_k} \not\subset X$. The following definition will be useful in this case.

Definition 10 For any $x \in X$, let $\text{Vis}(x, X)$ denote the *visibility polygon* of x in X , defined as

$$\text{Vis}(x, X) = \{x' \in X \mid \overline{xx'} \subset X\}. \quad (3.3)$$

We follow [32] in characterizing the boundaries visibility polygons in terms of non-spurious edges which are parts of ∂X and spurious edges which are not. Observe that since X is simply connected, the spurious edges subdivide X in such a way that every point $x' \notin \text{Vis}(x, X)$ can be associated with exactly one spurious edge such that the shortest path from x to x' crosses this spurious edge. Further, the first segment of the shortest path from x to x' will be parallel to this spurious edge. See Fig. 3.4. Let $\overline{t_k v_k}$ denote the spurious edge crossed by the shortest path from p_k to q_k . It is shown in [31] that such initial shortest path segments can be computed using a data structure with $O(\log n)$ query time, $O(n)$ preprocessing time and $O(n)$ storage.

Assume for the moment that $\overline{t_k v_k}$ is not a bitangent of X . Since this case creates some complications in the analysis, we will deal with it separately. Choose $u_k = (t_k - p_k)/\|t_k - p_k\|$. That is, select a motion in the direction of the spurious edge that hides q_k from p_k . Fig. 3.5 illustrates this selection (and the intuition behind the proof of Lemma 11). This completes the definition of our action sequence $u_{K_1+1}, \dots, u_{K_2}$:

$$u_i = \begin{cases} (q_i - p_i)/\|q_i - p_i\| & \text{if } q_i \in \text{Vis}(p_i, X) \\ (t_i - p_i)/\|t_i - p_i\| & \text{otherwise} \end{cases}, \quad (3.4)$$

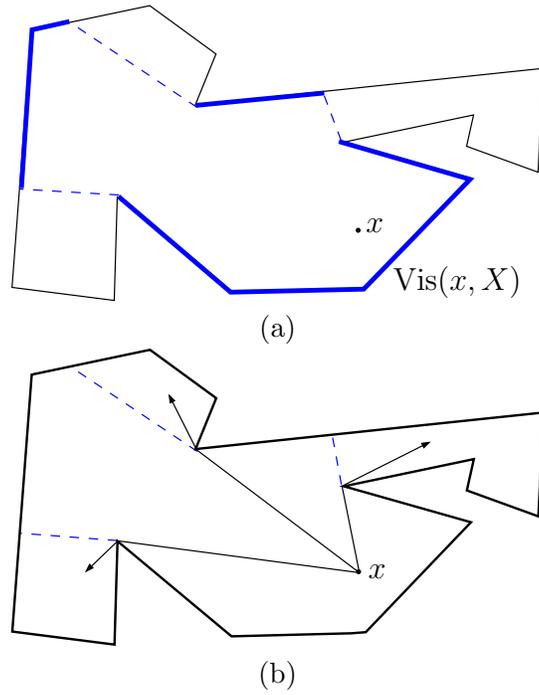


Figure 3.4: (a) A visibility polygon. Spurious edges are dashed. (b) The shortest path to any point not in the visibility polygon begins with a motion in the direction of a spurious edge.

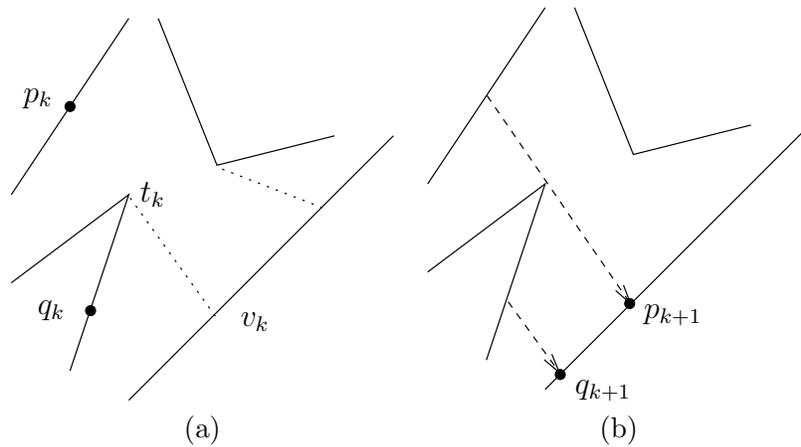


Figure 3.5: (a) The spurious edge $\overline{t_k v_k}$ hides p_k from q_k . (b) The point q_{k+1} cannot cross $\overline{t_k v_k}$ because its motion is parallel to $\overline{t_k v_k}$.

in which K_2 is the minimal i for which the first case applies. We will show in Theorem 12 that K_2 is well-defined, but we need the following lemma to do so:

Lemma 11 *Let $Q_k = X - \bigcup_{i=K_1, \dots, k} \text{Vis}(p_i, X)$. Then for $K_1 \leq k < K_2$ if K_2 is well-defined or $K_1 < k$ otherwise, $q_k \in Q_k$.*

Proof: Use induction on k . The statement is trivially true by construction when $k = K_1$. For the inductive step, note that q_k moves parallel to $\overline{t_k v_k}$, so that q_{k+1} is still behind this spurious edge. If $q_k \notin Q_k$, then q_k must be contained in a region that is not visible to some p_i , or in some region not seen by any p_i but separated from q_k by $\overline{t_k v_k}$. In either case, we can form a non-trivial loop in X , contradicting the simply-connected property of X . \square

One informal way to understand Lemma 11 is to imagine that p is “chasing” q . With each motion, p takes a step in pursuit of q and eliminates a portion of the environment Q_k in which q could be hiding. If K_2 is well-defined (and we will show momentarily that this is the case) then p will eventually “catch” q .

Finally, we must consider the special case when $\overline{t_k v_k}$ is a bitangent. This case is problematic because choosing $u_k = (t_k - p_k)/\|t_k - p_k\|$ is no longer sufficient to ensure that $Q_{k+1} \subset Q_k$. The algorithm as stated would alternate between the actions $t_k - v_k$ and $v_k - t_k$. This problem can be avoided by rotating u_k by a sufficiently small ϵ that $\overline{q_k q_{k+1}}$ will not intersect $\overline{t_k v_k}$. Then select $u_{k+1} = (v_k - p_{k+1})/\|v_k - p_{k+1}\|$. Fig 3.6 illustrates this situation. This modification adds an additional action each time p_k falls at the endpoint of a bitangent complement, but does not substantially change the analysis.

Now we can prove the algorithm’s correctness.

Theorem 12 *The sequence $u_{K_1+1}, \dots, u_{K_2}$ is a localizing sequence for $\{p, q\}$.*

Proof: If K_2 is well-defined, it follows from Lemma 9 that $u_{K_1+1}, \dots, u_{K_2}$ is a localizing sequence for $\{p, q\}$. To show that K_2 is well-defined, note that each p_k is in a different cell of the visibility cell decomposition [32] of X . There are only $O(n^2)$ such cells on the boundary, so $K_2 = O(n^2)$. \square

Now we can finally return to the general case with m points. If $m > n$ (recall n is the complexity of the environment boundary ∂X), then by the pigeonhole principle, at least two points must lie on the same edge of ∂X . This pair of points can see each other, and one motion will collapse them

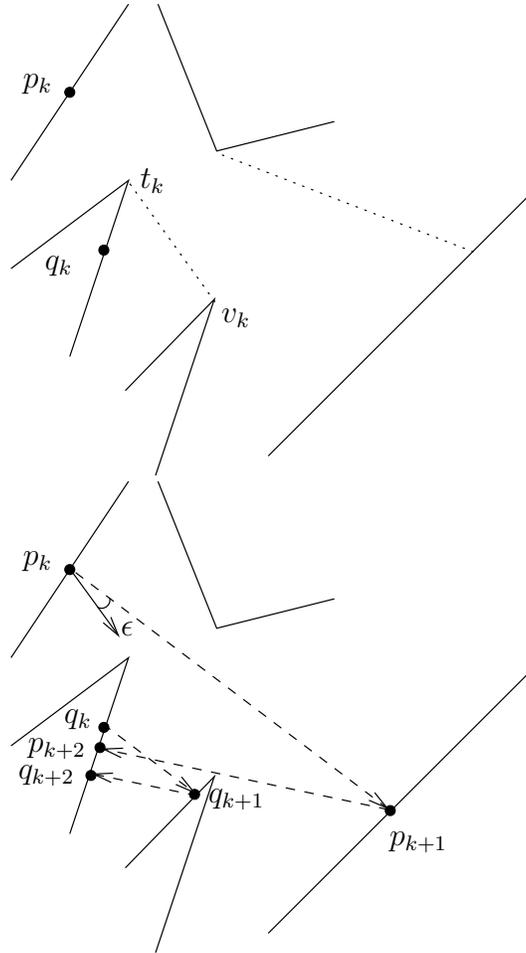


Figure 3.6: The special case when $\overline{t_k v_k}$ is a bitangent.

to a single point. In this way, we can reduce the information state to a set of at most n points using only $m - n$ actions. Then select an arbitrary pair of points p and q from the current information state η . We have just shown how to merge p and q in $O(n^2)$ steps. Repeating this process at most n times results in a plan of length $O(n^3)$ to map $\{p_1, \dots, p_m\}$ to a single point. Combining this with the $O(n^3)$ steps from the first part of the algorithm (Sec. 3.2.1) yields a total plan length of $K = K_1 + K_2 = O(n^3)$.

3.3 Computed examples

We have implemented Algs. 3.1 and 3.2 in simulation. Table 3.1 shows the 5 step localizing sequence generated by our implementation for an environment with many regularities. In contrast, our algorithm needs 28 steps for the similar but irregular environment in Table 3.2. In this sense, the localizing sequence for Table 3.1 appears to “exploit” these symmetries in the sense that uncertainty is simultaneously reduced in each of the identical branches. This is in sharp contrast to visibility-based localization, in which such symmetries are precisely what make localization problems difficult.

Fig. 3.7 shows a very irregular environment for which our algorithm generates a 30 step localizing sequence. This sequence is executed from six different initial positions. Note that because some actions in the sequence will lead to an immediate collision with the wall, these execution traces need not in general contain 30 segments. Table 3.3 depicts the information states at each of these steps.

Table 3.1: A localizing sequence computed by our algorithm for a highly symmetric environment.

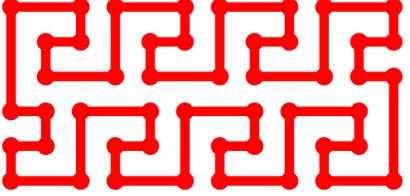
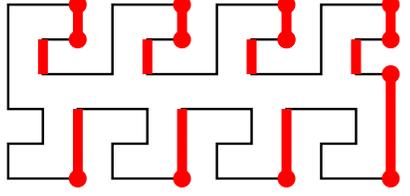
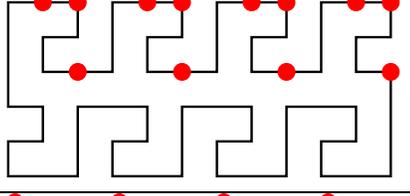
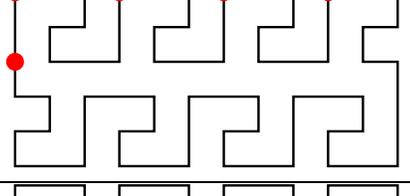
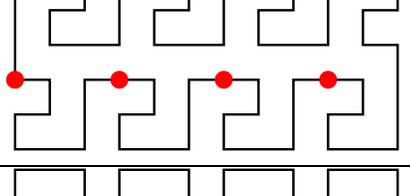
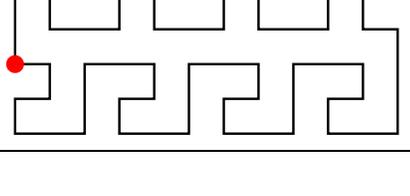
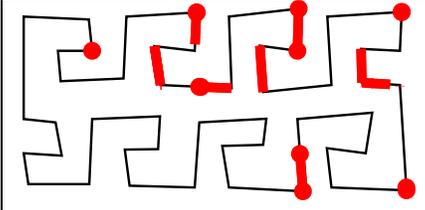
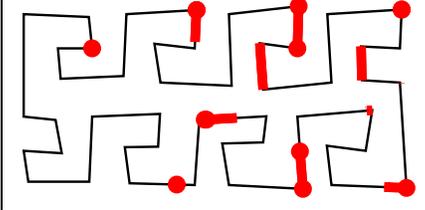
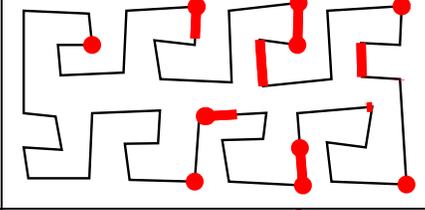
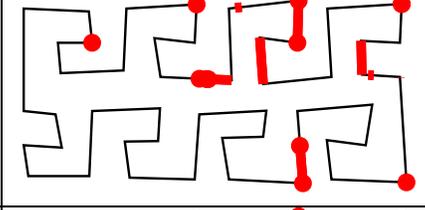
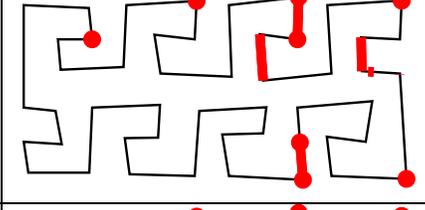
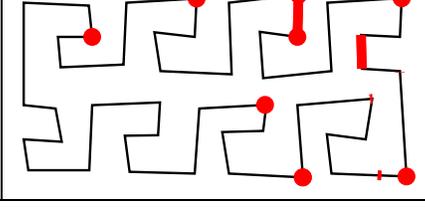
i	u_i	η_{i+1}
0		
1		
2		
3		
4		
5		

Table 3.2: A modified version of the environment from Table 3.1 in which the symmetries have been broken. Our algorithm generates a 28 step localizing sequence for this environment.

i	u_i	η_{i+1}
0		
1	↓	
2	↓	
3	→	
4	→	
5	↑	

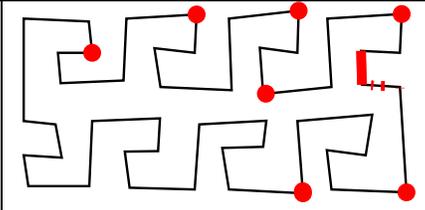
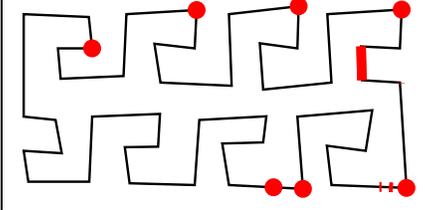
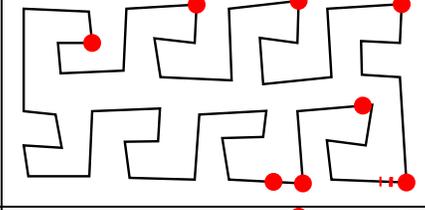
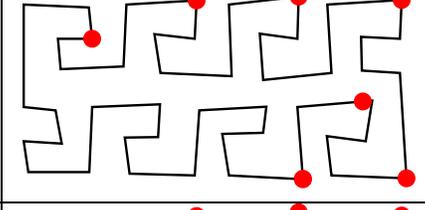
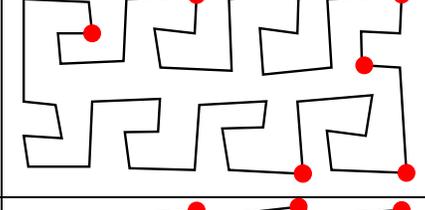
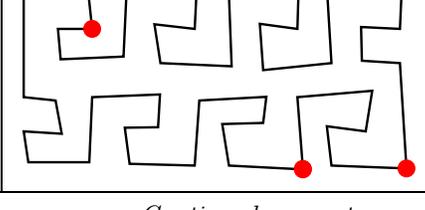
Continued on next page.

Table 3.2, cont.

i	u_i	η_{i+1}
6		
7		
8		
9		
10		
11		

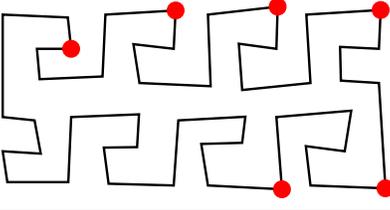
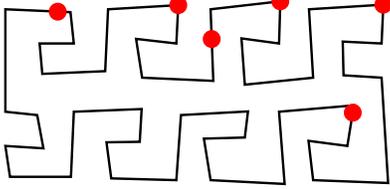
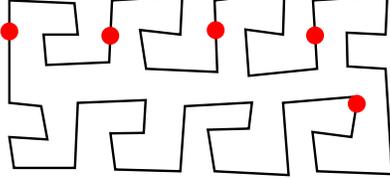
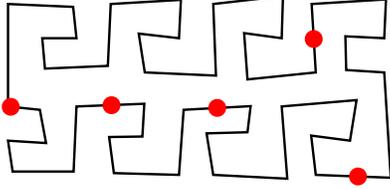
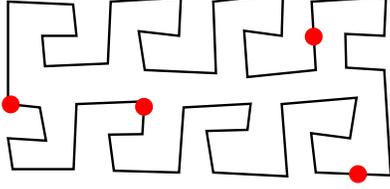
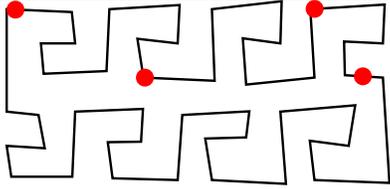
Continued on next page.

Table 3.2, cont.

i	u_i	η_{i+1}
12		
13		
14		
15		
16		
17		

Continued on next page.

Table 3.2, cont.

i	u_i	η_{i+1}
18		
19		
20		
21		
22		
23		

Continued on next page.

Table 3.2, cont.

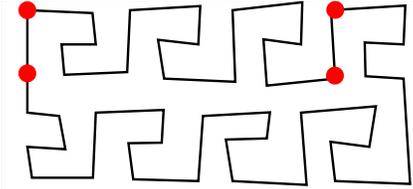
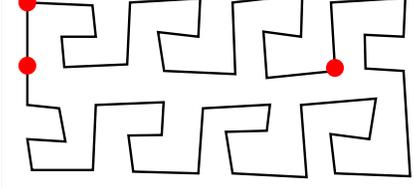
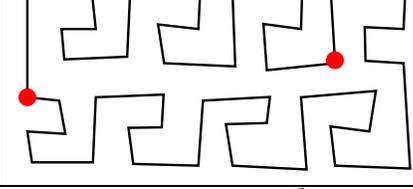
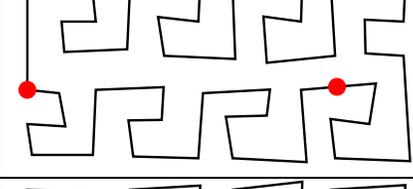
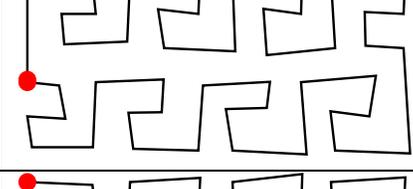
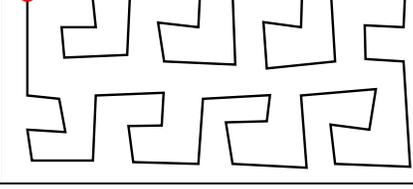
i	u_i	η_{i+1}
24		
25		
26		
27		
28		
29		

Table 3.3: An irregular environment for which the localizing sequence computed by our algorithm requires 30 steps.

i	u_i	η_{i+1}
0		
1		
2		
3		

Continued on next page.

Table 3.3, cont.

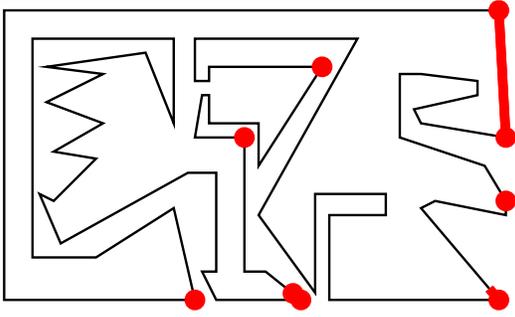
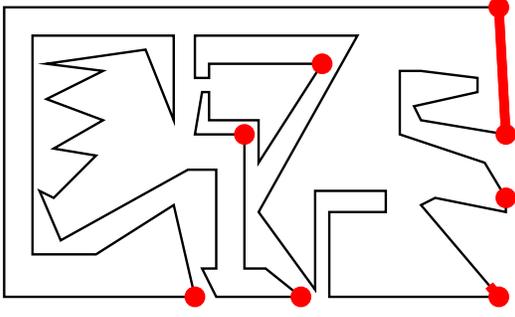
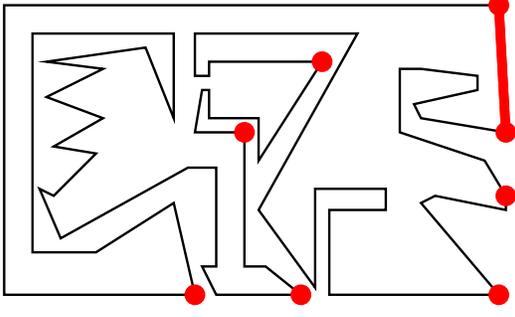
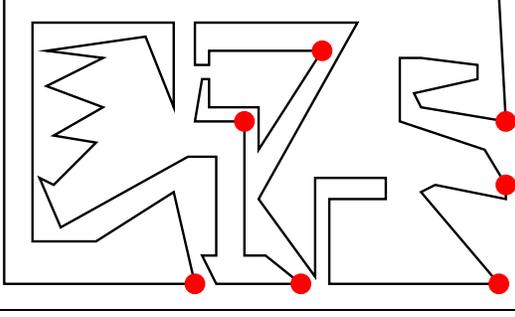
i	u_i	η_{i+1}
8		
9		
10		
11		
<i>Continued on next page.</i>		

Table 3.3, cont.

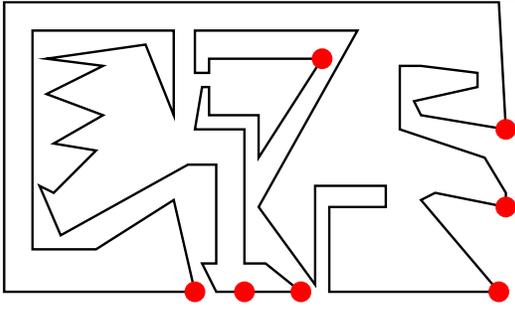
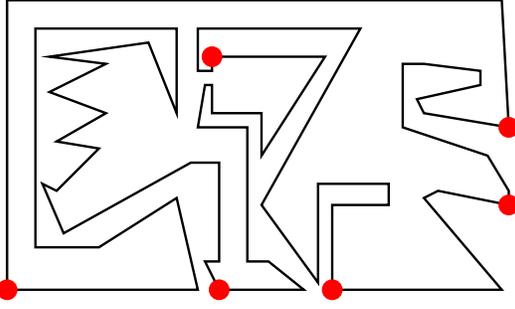
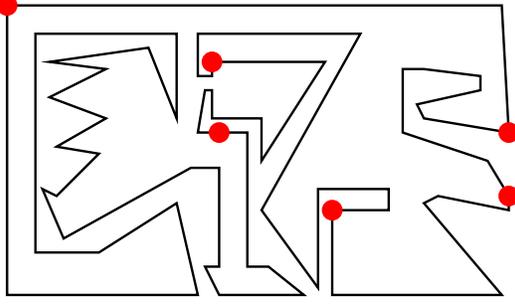
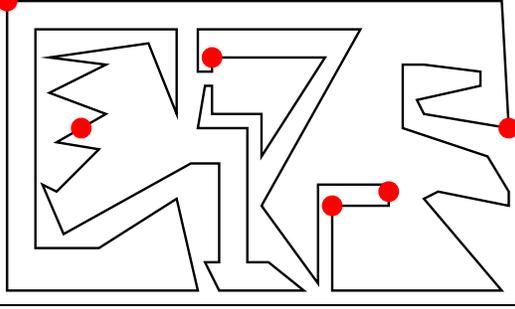
i	u_i	η_{i+1}
12		
13		
14		
15		
<i>Continued on next page.</i>		

Table 3.3, cont.

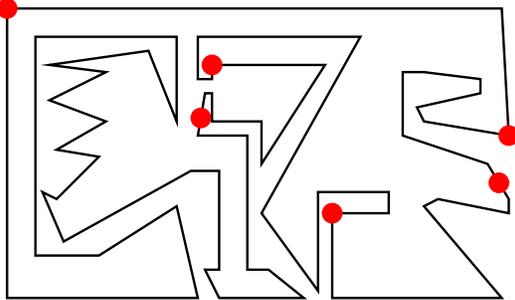
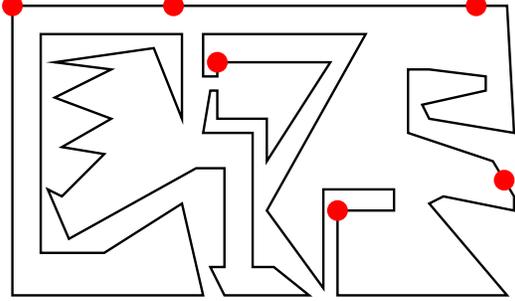
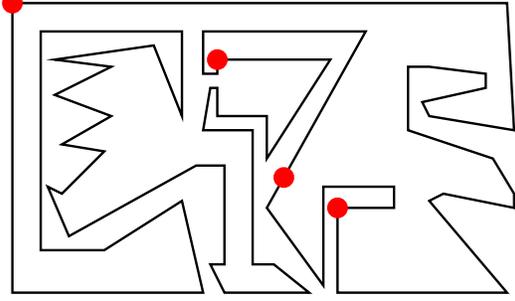
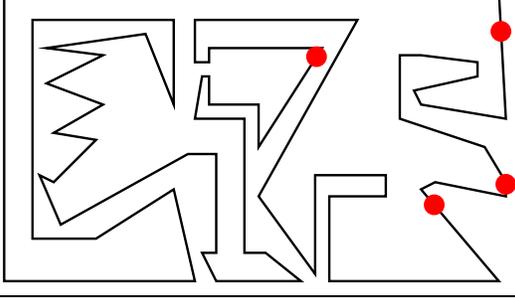
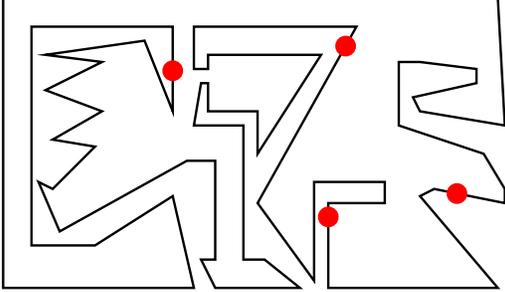
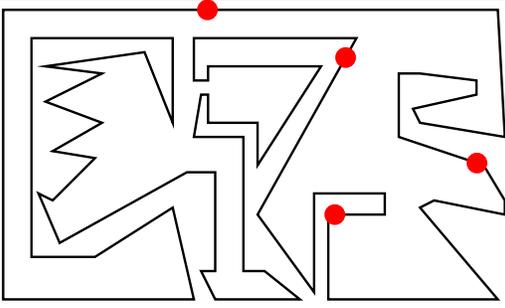
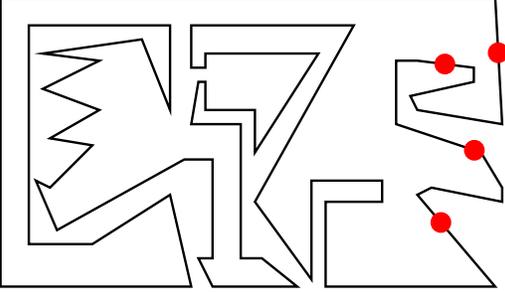
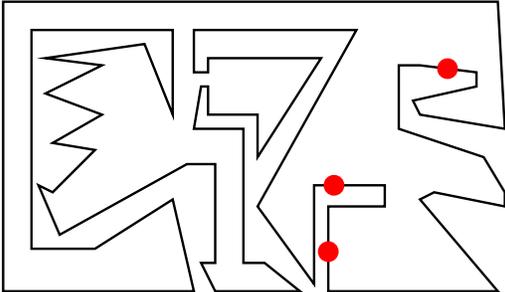
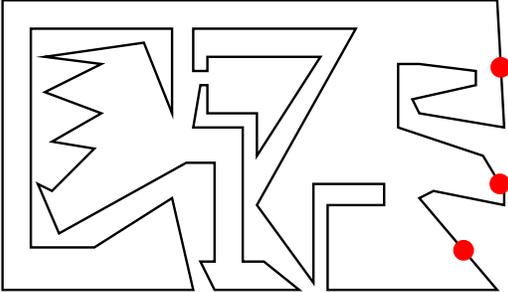
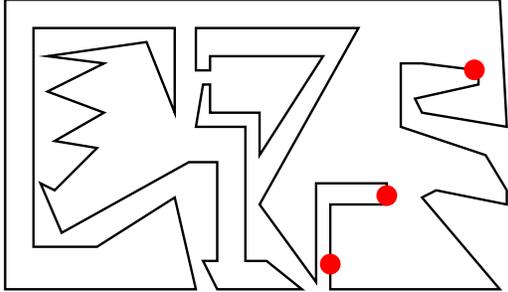
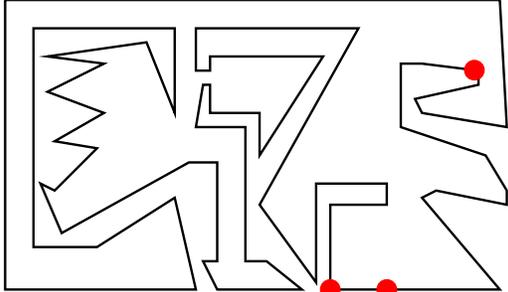
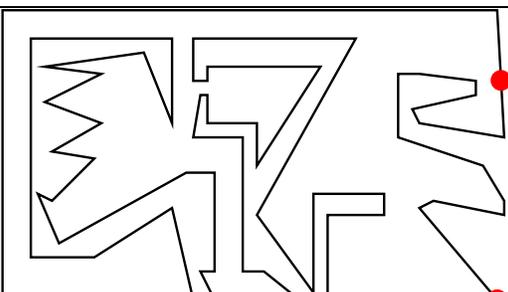
i	u_i	η_{i+1}
16		
17		
18		
19		
<i>Continued on next page.</i>		

Table 3.3, cont.

i	u_i	η_{i+1}
20		
21		
22		
23		

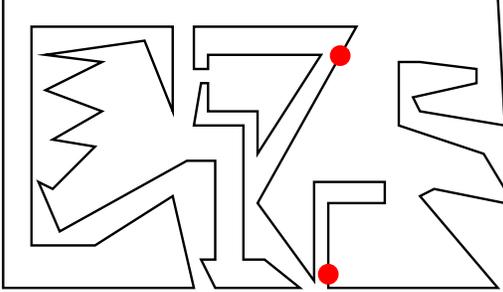
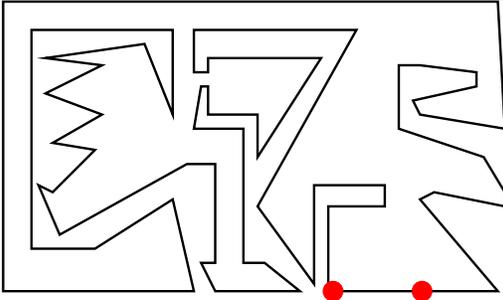
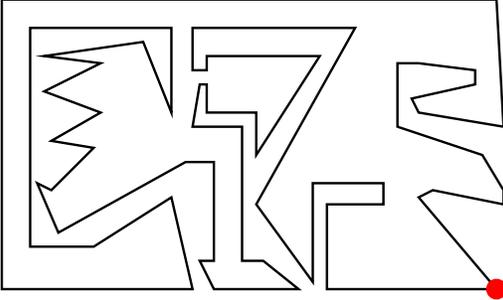
Continued on next page.

Table 3.3, cont.

i	u_i	η_{i+1}
24	→	
25	←	
26	↓	
27	→	

Continued on next page.

Table 3.3, cont.

i	u_i	η_{i+1}
28	←	
29	↓	
30	→	

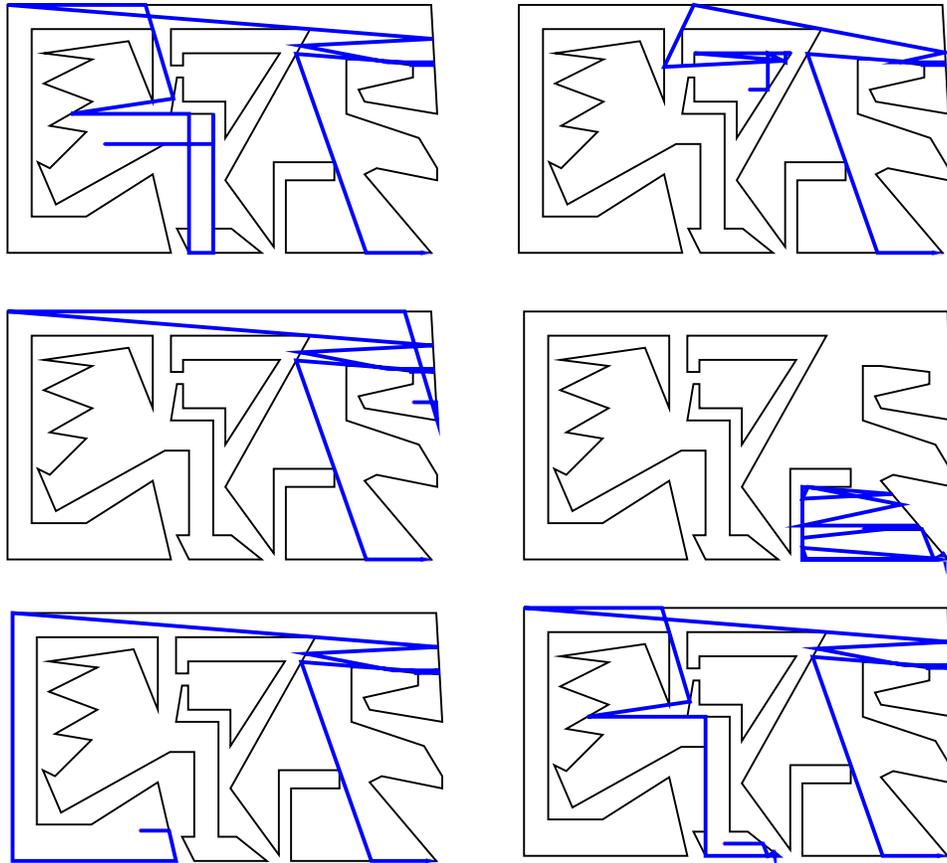


Figure 3.7: Execution traces of the localization sequence depicted in Table 3.3 for 6 different starting positions. For each starting position, the final position is the lower right corner of the environment.

Chapter 4

Minimality of the Model

In Chapter 3, we showed that a robot with only a map, compass, and contact sensor is capable of localizing itself within its environment. This chapter briefly addresses the minimality of this model. We argue that this robot model is locally minimal in the sense that omitting any of the map, compass, or contact sensor will make localization impossible.

Environment map The environment map is essential because the robot has no other way of obtaining information about its surroundings. Indeed, for a robot with no means of sensing its environment, the problem of localization is only well-defined in the case where the robot has some *a priori* description of the environment to give meaning to the idea of reaching a “known” location. We may, of course, imagine that the robot has some incomplete map information. For example, the robot may have a small collection of maps of potential environments. In this case it is a trivial extension of the algorithm we have given to devise an action sequence that localizes the robot, up to the selection of the environment.

Contact sensor Without its contact sensor, the robot cannot draw any firm conclusions about the results of its actions. The robot has no sense of time nor any way of measuring distances, so it cannot reliably execute any motions that are not maximal. We may replace the contact sensor with any sort of crude odometry for which only a lower bound in distance traveled is available. Then, assuming the robot’s wheels will slip after it reaches the opposite wall, maximal linear motions can be accomplished by moving in the commanded direction until a distance greater than the diameter of the environment has been covered. Of course, this sort of robot is strictly more capable than the one described in Chapter 2, since it can also stop before this point and draw some conclusions about its position.

Compass The question of whether localization is still possible without a compass demands more careful attention. To that end, we consider now a weaker robot which has angular odometry rather than a compass. That is, we now consider actions specified relative to an unknown initial orientation, rather than a global reference direction. In this section, we show that localizing sequences do not exist for this compass-free variant of our problem.

The problem of localization without a compass is identical to the formulation in Chapter 2, except that the environment is rotated through an unknown angle θ representing the difference between the global reference direction and the robot's initial orientation. A localizing sequence must map every $x \in X$ to the same x_f , regardless of θ .

Definition 13 *An information state-action pair (η, u) is a **collapsing transition** if u is parallel to some segment in η .*

Lemma 14 *Every localizing sequence contains at least one collapsing transition.*

Proof: Suppose there exists some localizing sequence u_1, \dots, u_K with no collapsing transitions. Arbitrarily pick a segment $s_1 \in \partial X$. Because of Lemma 7, at every step $1 \leq k \leq K$, $F(s_k, u_k)$ contains at least one segment s_{k+1} . We have constructed a segment $s_K \subseteq \eta_K$. Therefore $|\eta_K|$ is infinite, a contradiction. \square

Theorem 15 *For a robot with only angular odometry and a contact sensor in any polygonal environment X , no localizing sequence exists.*

Proof: Suppose such a sequence u_1, \dots, u_K exists. Let e_1, \dots, e_n denote the set of edges of ∂X , and let $\text{Rot}(v, \phi)$ denote the rotation of $v \in \mathbb{R}^2$ by angle ϕ . If there exists no action-edge pair (u_i, e_j) with u_i and $\text{Rot}(e_j, \theta)$ parallel, then u_1, \dots, u_K contains no collapsing transitions. The sequence is required to work for all $\theta \in S^1$ but the subset of S^1 in which some u_i coincides with some $\text{Rot}(e_j, \theta)$ has measure 0. Therefore u_1, \dots, u_K fails for almost every θ . \square

The intuition here is that reaching a finite-cardinality information state requires at least one motion parallel to some environment wall. No finite-length localizing sequence can achieve this for all possible starting orientations.

Chapter 5

Discussion

This thesis presented a localization technique for robots equipped with only a compass, a contact sensor, and a map of the environment. We showed the completeness of this technique for any compact simply-connected polygonal environment and proved that localization is impossible if the compass is replaced by an angular odometer. However, we have left open a number of interesting questions.

Multiply-connected environments Most obviously, the problem of generating a localizing sequence is still well-defined for multiply-connected environments, i.e. environments with “holes.” Our method depends on X being simply connected primarily for Lemma 11. It is not immediately clear whether a similar method can be devised for environments that are not simply connected.

ϵ -bounded error We have assumed that the robot can perfectly execute any commanded motion. We may more generally consider robots with bounded uncertainty in the angle of motion. This uncertainty might arise from errors in actuation or noise in compass readings. Under this model, points in an information state would undergo a “dilation” during each transition with the amount of dilation being an increasing function of the distance traveled. Our two-stage approach clearly fails under this generalization.

Optimality In this thesis we have only considered the existence question for localizing sequences in simple polygons. The $O(n^3)$ bound on the number of steps can quite likely be improved. Also, it remains an open problem to generate localizing sequences that are optimal in any sense. Two reasonable optimality criteria are the number of steps in the sequence and the maximum distance traveled for any initial state in X . Finding decision trees for minimum distance localization of a robot with a range sensor is NP-hard

[25]. In that model, a localization strategy is a decision tree, and the difficulty comes in finding the shallowest decision tree that can discriminate every set of points with equivalent visibility polygons. Since our robot model does not admit branching in the localizing sequence, neither those hardness results nor the general methods used to prove them are applicable.

References

- [1] E. U. Acar and H. Choset. Complete sensor-based coverage with extended-range detectors: A hierarchical decomposition in terms of critical points and voronoi diagrams. In *Proc. of IEEE IROS, Int'l Conference on Intelligent Robots and Systems*, 2001.
- [2] E. U. Acar and H. Choset. Robust sensor-based coverage of unstructured environments. In *Proc. of IEEE IROS, Int'l Conference on Intelligent Robots and Systems*, 2001.
- [3] P.K. Agarwal, A.D. Collins, and J.L. Harer. Minimal trap design. In *Proc. IEEE Int. Conf. Robot. and Autom.*, volume 3, pages 2243–2248, 2001.
- [4] S. Akella, W. Huang, K. M. Lynch, and M. T. Mason. Parts feeding on a conveyor with a one joint robot. *Algorithmica*, 26(3):313–344, March-April 2000.
- [5] S. Akella, W. H. Huang, K. M. Lynch, and M. T. Mason. Sensorless parts feeding with a one joint robot. In J.-P. Laumond and M. Overmars, editors, *Algorithms for Robotic Motion and Manipulation*, pages 229–237. A K Peters, Wellesley, MA, 1997.
- [6] S. Akella and M. Mason. Posing polygonal objects in the plane by pushing. *International Journal of Robotics Research*, 17(1):70–88, January 1998.
- [7] R. Aleliunas. A simple graph traversing problem. Master's thesis, University of Toronto, April 1978.
- [8] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovász, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proc. IEEE. Symp. Found. Comp. Sci.*, pages 218–223, 1979.

- [9] N. Alon, Y. Azar, and Y. Ravid. Universal sequences for complete graphs. *Discrete Appl. Math.*, 27(1-2):25–28, 1990.
- [10] D. Avis and H. Imai. Locating a robot with angle measurements. *J. Symb. Comput.*, 10(3-4):311–326, 1990.
- [11] K. Basye and T. Dean. Map learning with indistinguishable locations. In *Proc. Conf. Uncert. Artif. Intell.*, pages 331–342. North-Holland, 1990.
- [12] R-P. Berretty, K. Goldberg, M. Overmars, and F. Van der Stappen. Trap design for vibratory part feeders. *Int. J. Robot. Res.*, 20(11), November 2001.
- [13] M. Blum and D. Kozen. On the power of the compass (or, why mazes are easier to search than graphs). In *Proc. IEEE. Symp. Found. Comp. Sci.*, pages 132–142, 1978.
- [14] G. Boothroyd, C. Poli, and L. E. Murch. *Automatic Assembly*. Marcel Dekker, Inc., 1982.
- [15] A. Borodin, W. L. Ruzzo, and M. Tompa. Lower bounds on the length of universal traversal sequences. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 562–573. ACM Press, 1989.
- [16] M. Brokowski, M. A. Peshkin, and K. Goldberg. Curved fences for part alignment on a belt. *ASME Journal of Mechanical Design*, 117(1), March 1995.
- [17] J. F. Canny and K. Y. Goldberg. “RISC” industrial robots: Recent results and current trends. In *Proc. IEEE Int. Conf. Robot. and Autom.*, pages 1951–1958, 1994.
- [18] A. K. Chandra, P. Raghavan, W. L. Ruzzo, and R. Smolensky. The electrical resistance of a graph captures its commute and cover times. In *Proc. ACM Symposium on Theory of Comp.*, pages 574–586. ACM Press, 1989.
- [19] B. Chazelle and L. G. Guibas. Visibility and intersection problems in plane geometry. *Disc. and Comp. Geom.*, 4:551–589, 1989.
- [20] H. Choset and J. Burdick. Sensor based planning, part I: The generalized Voronoi graph. In *Proc. IEEE Int. Conf. Robot. and Autom.*, pages 1649–1655, 1995.

- [21] I. J. Cox. Blanche – an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Trans. Robot. and Autom.*, 7:2:193–204, 1991.
- [22] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. IEEE Int. Conf. Robot. and Autom.*, 1999.
- [23] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Robot localization without depth perception. In *Scandinavian Workshop on Algorithm Theory*, 2002.
- [24] B. R. Donald. On information invariants in robotics. *Artif. Intell.*, 72:217–304, 1995.
- [25] G. Dudek, K. Romanik, and S. Whitesides. Localizing a robot with minimum travel. In *SODA: ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1995.
- [26] D. Eppstein. Reset sequences for monotonic automata. *SIAM J. Comput.*, 19(3):500–510, 1990.
- [27] M. A. Erdmann and M. T. Mason. An exploration of sensorless manipulation. *IEEE Trans. Robot. and Autom.*, 4(4):369–379, August 1988.
- [28] K. Y. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10:201–225, 1993.
- [29] K. Y. Goldberg and M. T. Mason. Bayesian grasping. In *Proc. IEEE Int. Conf. Robot. and Autom.*, 1990.
- [30] D.D. Grossman and M.W. Blasgen. Orienting parts by computer controlled manipulation. *IEEE Trans. Systems, Man and Cybernetics*, 5(5):561–565, 1975.
- [31] L. J. Guibas and J. Hershberger. Optimal shortest path queries in a simple polygon. *J. Comput. Syst. Sci.*, 39(2):126–152, 1989.
- [32] L. J. Guibas, R. Motwani, and P. Raghavan. The robot localization problem. In K. Goldberg, D. Halperin, J.-C. Latombe, and R. Wilson, editors, *Proc. Workshop on Alg. Found. of Robot.*, pages 269–282. A.K. Peters, Wellesley, MA, 1995.
- [33] R. Hinkel and T. Knieriemen. Environment perception with a laser radar in a fast moving robot. In *Proceedings of Symposium on Robot Control*, 1988.

- [34] L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is np-complete. *Inf. Process. Lett.*, 5:15–17, 1976.
- [35] J. D. Kahn, N. Linial, N. Nisan, and M.E. Saks. On the cover time of random walks in graphs. *Journal of Theoretical Probability*, 2(1):121–128, January 1989.
- [36] I. Kamon and E. Rivlin. Sensory-based motion planning with global proofs. *IEEE Trans. Robot. and Autom.*, 13(6):814–822, December 1997.
- [37] I. Kamon, E. Rivlin, and E. Rimon. Range-sensor based navigation in three dimensions. In *Proc. IEEE Int. Conf. Robot. and Autom.*, 1999.
- [38] J. M. Kleinberg. The localization problem for mobile robots. In *IEEE Symposium on Foundations of Computer Science*, pages 521–531, 1994.
- [39] K. N. Kutulakos, C. R. Dyer, and V. J. Lumelsky. Provable strategies for vision-guided exploration in three dimensions. In *Proc. IEEE Int. Conf. Robot. and Autom.*, pages 1365–1371, 1994.
- [40] K. N. Kutulakos, V. J. Lumelsky, and C. R. Dyer. Vision-guided exploration: a step toward general motion planning in three dimensions. In *Proc. IEEE Int. Conf. Robot. and Autom.*, pages 289–296, 1993.
- [41] A. M. Ladd, K. E. Bekris, A. P. Rudys, D. S. Wallach, and L. E. Kavraki. On the feasibility of using wireless Ethernet for indoor localization. *IEEE Transactions on Robotics and Automation*, 20(3):555–559, June 2004.
- [42] J. Leonard, H. Durrant-Whyte, and I. Cox. Dynamic map building for an autonomous mobile robot. *Int. J. Robot. Res.*, 11(4):89–96, 1992.
- [43] V. Lumelsky and S. Tiwari. An algorithm for maze searching with azimuth input. In *Proc. IEEE Int. Conf. Robot. and Autom.*, pages 111–116, 1994.
- [44] V. J. Lumelsky and A. A. Stepanov. Path planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.
- [45] M. S. Manasse, L. A. McGeoch, and D. D. Sleator. Competitive algorithms for on-line problems. In *Proc. ACM Symp. Theory Comput.*, pages 322–333, 1988.

- [46] C. Ó. Dúnlaing and C. K. Yap. A retraction method for planning the motion of a disc. *Journal of Algorithms*, 6:104–111, 1982.
- [47] J. M. O’Kane and S. M. LaValle. Almost-sensorless localization. In *Proc. IEEE Int. Conf. Robot. and Autom.*, 2005. To appear.
- [48] M. Rao, G. Dudek, and S. Whitesides. Randomized algorithms for minimum distance localization. In *Proc. Workshop on Algorithmic Foundations of Robotics*, pages 265–280, 2004.
- [49] W. Renken. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 1993.
- [50] K. Romanik and S. Schuierer. Optimal robot localization in trees. In *Proc. Symp. Comp. Geom.*, pages 264–273, 1996.
- [51] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Comm. of the ACM*, 28:202–208, 1985.
- [52] K. Sugihara. Some location problems for robot navigation using a simple camera. *Comp. Vis., Graphics, & Image Proc.*, 42(1):112–129, 1988.
- [53] M. Thompa. Lower bounds on universal traversal sequences for cycles and other low degree graphs. *SIAM J. Comput.*, 21(6), December 1992.
- [54] S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21(4):93–109, 2000.
- [55] S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, pages 1–25, April 1998.
- [56] B. Tovar, L. Guilamo, and S. M. LaValle. Gap Navigation Trees: Minimal representation for visibility-based tasks. In *Proc. Workshop on Alg. Found. of Robot.*, 2004.
- [57] B. Tovar, S. M. LaValle, and R. Murrieta. Locally-optimal navigation in multiply-connected environments without geometric maps. In *IEEE/RSJ Int’l Conf. on Intelligent Robots and Systems*, 2003.
- [58] A. F. van der Stappen, R.-P. Berretty, K. Goldberg, and M. H. Overmars. Geometry and part feeding. In *Sensor Based Intelligent Robots*, pages 259–281, 2000.

- [59] G. Weiss, C. Wetzler, and E. von Puttkamer. Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 1994.
- [60] D.E. Whitney. Real robots don't need jigs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1986.
- [61] J. Wiegley, K. Goldberg, M. Peshkin, and M. Brokowski. A complete algorithm for designing passive fences to orient parts. *Assembly Automation*, 17(2), August 1997.