Almost-Sensorless Localization

Jason M. O'Kane and Steven M. LaValle
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
{*jokane, lavalle*}*@cs.uiuc.edu*

***Abstract*— We present a localization method for robots equipped with only a compass, a contact sensor and a map of the environment. In this framework, a localization strategy can be described as a sequence of directions in which the robot moves maximally. We show that a localizing sequence exists for any simply connected polygonal environment and present an algorithm for computing such a sequence. We have implemented the algorithm, and we present several computed examples. We also prove that the sensing model is minimal by showing that replacement of the compass by an angular odometer precludes the possibility of performing localization.**

I. Introduction

Localization – the task of systematically eliminating uncertainty in the pose of a robot – is a fundamental problem for any practical autonomous robotic system. It has long been known that robot designs with simplified sensing and actuation models can lead to decreased costs and increased robustness [26]. This paper applies this idea to the problem of localization in an attempt to describe the simplest possible robot with which localization is still possible. In particular, we propose a robot model in which only a compass and a contact sensor are available. Odometry, range sensing and wall-following abilities are notably 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. The primary contribution of this paper is an algorithm which, given a simply-connected polygonal environment, will generate a sequence of motions that will localize the robot under total uncertainty in the initial state. The correctness of this algorithm constitutes a constructive proof that a localizing sequence exists for any simply-connected polygonal environment.

Much attention has been given to the problem of localization for robots with varying degrees of sensing capability. In [16], the environment is constrained to an embedding of an acyclic graph into $\mathbb{R}^n$, and sensing is limited to the orientations of incident edges. The static problem of finding the set of candidate locations for a given visibility region was solved in [13]. The problem of localization with a visibility sensor while minimizing distance traveled was proven NP-hard in [7]. This work also provides a greedy algorithm that approximates the minimum distance localization strategy. In [20], randomization is used to select motions to disambiguate candidate locations in a visibility-based approach. Other approaches have used visual data [8] and wireless ethernet signal strength [17]. Also, a large family of methods use probabilistic models to estimate the current state [14, 18, 21, 22].

A long line of research has investigated robotic systems with limited sensing capability. Manipulation problems have been solved without sensing in a variety of ways [1, 3, 4, 9, 11, 10, 25, 27]. Exploration and navigation tasks are solved with a sensor for discontinuities in depth information in [23, 24]. Bug algorithms are used for navigation by robots capable only of moving toward obstacles and following walls in [15, 19]. Other work has explored the more general question of the minimal sensing requirements to complete a given task [2, 6].

The remainder of this paper is organized as follows. Sec. II formalizes our model, defines the localization problem and characterizes this model as a search through an abstract space of information states. Sec. III gives an algorithm for computing transitions in information space. The main algorithm is described in Sec. IV. Sec. V proves that a variant of this problem in which the robot has no compass cannot be solved. Sec. VI presents concluding remarks.

II. Problem Statement

A. 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.

In this paper we show that this information is sufficient for localization.

B. Problem Formalization

This section formalizes the abstract robot model we have described. We 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 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:

$$f^k(x, u_1, \ldots, u_k) = f(\cdots f(f(x, u_1), u_2) \cdots), u_k). \quad (1)$$

Now we can define the notion of a solution.

Definition 1 *A* **localizing sequence** *for X is a sequence of actions $u_1, u_2, \ldots, u_K$ such that there exists $x_f \in X$ with $f^K(x, u_1, \ldots, u_K) = x_f$ 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. Fig. 1 shows a sample polygon and a localizing sequence for it. 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.

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.

C. 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.

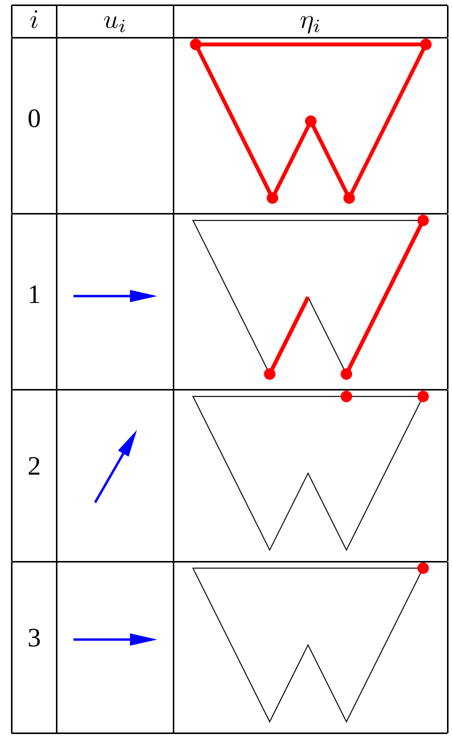

Fig. 1. A localizing sequence for a simple non-convex polygon. Possible states at each step are shaded.

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 $u_1, \ldots, u_{k-1}$. The* **information state** *η_k of the robot is*

$$\eta_k = \{x \in \partial X \mid \exists x_I \in \partial X, x = f^{k-1}(x_I, u_1, \ldots, u_{k-1})\}. \quad (2)$$

The **information space** *$\mathcal{I}$ is defined as 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

$$\mathcal{I}_G = \{\eta \in \mathcal{I} \mid |\eta| = 1\}, \quad (3)$$

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

Algorithm 1 $\textsc{InfoTrans}(X, s_1, \ldots, s_l, p_1, \ldots, p_m, u)$

```
η' ← ∅
for i ← 1 ... l do
   {a, b} ← endpoints of s_i
   E ← vertices of X
   E ← E − {v ∈ E | CCW(a, a + u, x) = CCW(b, b + u, x)}.
   E ← SORT(E) by perpendicular distance from a(a + u)
   p ← SHOOTRAYFORSWEEP(X, a, u, b − a)
   for e ∈ E do
      if SAMEEDGE(p, e) then
         η' ← η' ∪ pe
         p ← SHOOTRAYFORSWEEP(X, e, u, b − a)
      else
         p' ← SHOOTRAYFORSWEEP(X, e, u, b − a)
         η' ← η' ∪ pp'
         p ← p'
      end if
   end for
end for

for j ← 1 ... m do
   η' ← η' ∪ {SHOOTRAY(X, p_j, u)}
end for

return η'
```

to the forward projection

$$F(\eta, u) = \{x' \in \partial X \mid \exists x \in \eta, x' = f(x, u)\}. \quad (4)$$

As with f, we define

$$F^k(\eta, u_1, \ldots, u_k) = F(\cdots F(F(\eta, u_1), u_2) \cdots), u_k). \quad (5)$$

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

$$\left|F^K(\partial X, u_1, \ldots, u_K)\right| = 1. \quad (6)$$

Our algorithm presentation in Section IV will take this view of localizing sequences.

III. Computing the Information Transition Function

This section presents a simple algorithm for computing $F(\eta, u)$ given X, η and u. Computing these information transitions will play a crucial role in the algorithm to generate localizing sequences, which appears in Section IV. We restrict our attention to information states that can be reached from the initial state $\eta_1 = \partial X$. Alg. 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 $s_1, \ldots, s_l$ of open segments and a finite set of points $p_1, \ldots, p_m$ 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.

Lemma 3 *For any action $u \in U$ and any information state $\eta = [\bigcup_i s_i] \cup \left[\bigcup_j \{p_j\}\right]$, in which the s_i's are open segments and the p_j's are points in ∂X,*

$$F(\eta, u) = \left[\bigcup_i F(s_i, u)\right] \cup \left[\bigcup_j F(\{p_j\}, u)\right]. \quad (7)$$

Proof: Immediate from the definition of F (Eq. 4). □

The next lemma characterizes the set of reachable information states.

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

Proof Sketch: Use induction on k. When $k = 0$, $\eta = \partial X$, which is the union of the vertices and edges bounding X. For the inductive step, note that the image under F of a segment is a finite set of polygonal chains. □

As a consequence of these two lemmas, we can write any reachable information state η by listing the points $p_1, \ldots, p_m$ and segments $s_1, \ldots, s_l$ that compose it, and to compute any transition $F(\eta, u)$, it will be sufficient to give an algorithm for the cases where η is an open segment and a single point.

- If η is a singe 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 [5].
- 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. 2 illustrates the sweeping algorithm.

Alg. 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 a polygon X with n vertices.

IV. 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 parts. First, actions are selected which reduce the uncertainty in the robot's position to a finite set of possibilities. Second, actions are selected to reduce the uncertainty from this finite set to a single point.

Algorithm 2 LOCALIZINGSEQUENCE(X)

$\eta_1 \leftarrow \partial X$
$k \leftarrow 1$
while η_k contains at least one segment **do**
  $\overline{ab} \leftarrow$ leftmost segment in η_k
  if $(a-b).x > 0$ **then**
    $u_k \leftarrow (a-b)/||a-b||$
  else
    $u_k \leftarrow (b-a)/||b-a||$
  end if
  $\eta_{k+1} \leftarrow$ INFOTRANS(X, η_k, u_k)
  $k \leftarrow k+1$
end while

while η_k contains at least two points **do**
  Select p, q from η_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
    $u_k \leftarrow (t_k - p_k)/||t_k - p_k||$
    $\eta_{k+1} \leftarrow$ INFOTRANS(X, η_k, u_k)
    $p_{k+1} \leftarrow$ SHOOTRAY(X, p_k, u_k)
    $q_{k+1} \leftarrow$ SHOOTRAY(X, q_k, u_k)
    $k \leftarrow k+1$
  end while
  $u_k \leftarrow (q_k - p_k)/||q_k - p_k||$
  $\eta_{k+1} \leftarrow$ INFOTRANS(X, η_k, u_k)
  $k \leftarrow k+1$
end while

return $(u_1, \ldots, u_{k-1})$

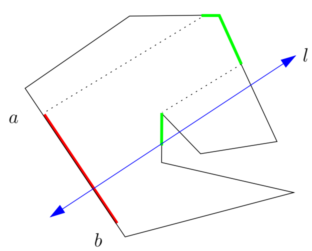

Fig. 2. Computing $F(\eta, u)$ by a line sweep algorithm.

The complete localizing sequence $u_1, \ldots, u_K$ is divided into two parts $u_1, \ldots, u_{K_1}$ and $u_{K_1+1}, \ldots, u_{K_2}$ generated by the respective parts of the algorithm. The complete algorithm is shown in Alg. 2; the subsequent exposition will explain and justify it.

A. 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 provides the basis for the algorithm.

Lemma 5 *For any segment $s = \overline{ab} \subset X$, $F(s,u)$ is a single point if and only if $u = (a-b)/||a-b||$ or $u = (b-a)/||b-a||$.*

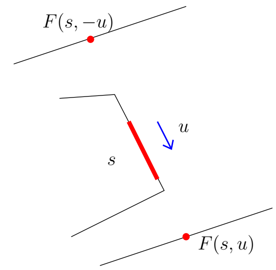

Fig. 3. A motion along s will collapse s to a single point.

Proof Sketch: Follows from the definitions of F and f and the fact that $\overline{ab}$ is collision-free. □

Informally, this means any segment can be collapsed to a point by a single motion along its length. This situation is illustrated in Fig. 3.

Starting with $\eta_1 = \partial X$, the algorithm maintains a "current" information state η_k and a sequence of actions $u_1, \ldots, u_{k-1}$ 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.

Lemma 6 *The above algorithm generates $K_1 = O(n^3)$ actions for an environment with n edges.*

Proof Sketch: Let $e_1, \ldots, e_n$ 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 $F(e_i, v(e_j))$. 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 or to some transition from another edge. There are n segments in the initial condition and R describes a the 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 $K_1 = n + O(n^3) = O(n^3)$. □

B. From a finite subset to a single point

The previous section showed how to select actions $u_1, \ldots, u_{K_1}$ that map ∂X to a finite set $\{p_1, p_2, \ldots, p_m\}$ of points. It remains to generate additional actions $u_{K_1+1}, \ldots, u_{K_2}$ mapping $\{p_1, p_2, \ldots, p_m\}$ to a single point.

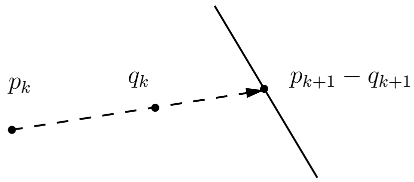

Fig. 4. If p_k can see q_k, then a motion in the direction of $\overline{p_k q_k}$ maps p_k and q_k to the same place.

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 $u_{K_1+1}, \ldots, u_{K_2}$ such that

$$f^{K_2-K_1}(p, u_{K_1+1}, \ldots, u_{K_2}) = f^{K_2-K_1}(q, u_{K_1+1}, \ldots, u_{K_2}). \tag{8}$$

For $K_1 < k \leq K_2$, let $p_k = f(p, u_{K_1+1}, \ldots, u_k)$ and likewise $q_k = f(q, u_{K_1+1}, \ldots, u_k)$. Our algorithm will select u_k using only p_k and q_k. We begin with the simple base case:

Lemma 7 *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 5. □

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 path will map both p_k and q_k to the same place. Fig. 4 illustrates this situation.

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

Definition 8 *For any $x \in X$, let $\mathrm{Vis}(x, X)$ denote the **visibility polygon** of x in X, defined as*

$$\mathrm{Vis}(x, X) = \{x' \in X \mid \overline{xx'} \subset X\}. \tag{9}$$

We follow [13] in characterizing 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 $x' \notin \mathrm{Vis}(x, X)$ 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. 5. Let $\overline{t_k v_k}$ denote the spurious edge crossed by the shortest path from p_k to q_k. 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 [12].

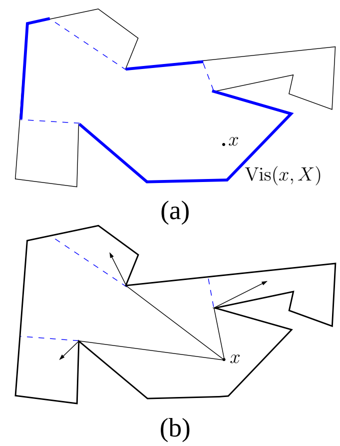

Fig. 5. (a) A visibility polygon. Spurious edges are dashed. (b) The shortest path to any point not in the visibility polygon begins with a motion in the direction of a spurious edge.

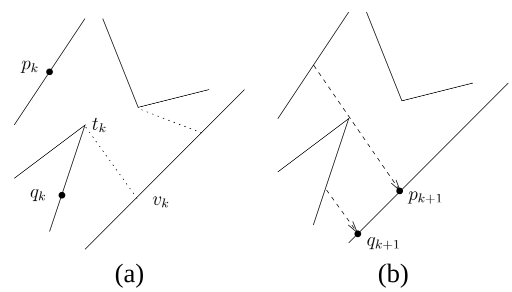

Fig. 6. (a) The spurious edge $\overline{t_k v_k}$ hides p_k from q_k. (b) The point q_{k+1} cannot cross $\overline{t_k v_k}$ because its motion is parallel to $\overline{t_k v_k}$.

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. Fig. 6 illustrates this selection (and the intuition behind the proof of Lemma 9). This completes the definition of our action sequence $u_{K_1+1}, \ldots, u_{K_2}$:

$$u_i = \begin{cases} (q_i - p_i)/||q_i - p_i|| & \text{if } q_i \in \mathrm{Vis}(p_i, X) \\ (t_i - p_i)/||t_i - p_i|| & \text{otherwise} \end{cases}, \tag{10}$$

in which K_2 is the minimal i for which the first case applies. We will show in Theorem 10 that K_2 is well-defined, but we need to following lemma to do so:

Lemma 9 *Let $Q_k = X - \bigcup_{i=K_1 \ldots k} \mathrm{Vis}(p_i, X)$. 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 Sketch: 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

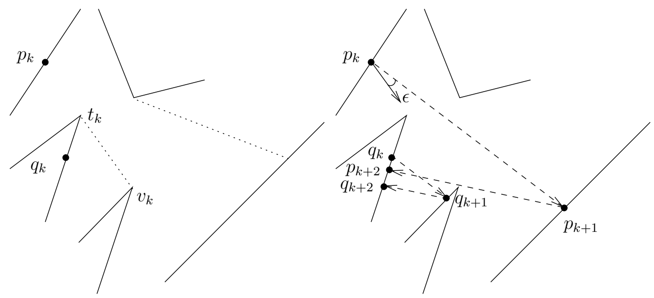

Fig. 7. The special case when $\overline{t_k v_k}$ is a bitangent.

this spurious edge. Use the fact that X is simply connected and the inductive hypothesis to complete the proof. □

One informal way to understand Lemma 9 is to imagine that p is "chasing" q. At each step p takes a step closer to q and eliminates a portion of the environment Q_k in which q could be hiding.

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 q will not cross $\overline{t_k v_k}$. Then select $u_{k+1} = (v_k - p_{k+1})/||v_k - p_{k+1}||$. Fig 7 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 10 *The sequence $u_{K_1+1}, \ldots, u_{K_2}$ is a localizing sequence for $\{p, q\}$.*

Proof Sketch: If K_2 is well-defined, it follows from Lemma 7 that $u_{K_1+1}, \ldots, u_{K_2}$ 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 [13] of X. There are only $O(n^2)$ such cells on the boundary, so $K_2 = O(n^2)$. □

Now we can finally return to the general case with m points. If $m > n$, 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 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 $\{p_1, \ldots, p_m\}$ to a single point. Combining this with the $O(n^3)$ steps from the first part of the algorithm (Sec. IV-A) yeilds a total plan length of $K = K_1 + K_2 = O(n^3)$.

C. Computed Examples

We have implemented this algorithm in simulation. Fig. 8 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 Fig. 9. In this sense, the localizing sequence for Fig. 8 appears to localization sequence 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. 10 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.

V. The Need for a Compass

Throughout this paper, we have worked under the assumption that the robot has a compass. 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 the compass-free variant of our problem.

The problem of localization without a compass is identical to the formulation in Sec. II, 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 11 *An information state – action pair (η, u) is a **collapsing transition** if u is parallel to some segment in η.*

Lemma 12 *Every localizing sequence contains at least one collapsing transition.*

Proof Sketch: Suppose there exists some localizing sequence $u_1, \ldots, u_K$ with no collapsing transitions. Arbitrarily pick a segment $s_1 \in \partial X$. At every step $1 \leq k \leq K$, $F(s_k, u_k)$ contains at least one segment s_{k+1} (because of Lemma 5). We have constructed a segment $s_K \subseteq \eta_K$. Therefore $|\eta_K|$ is infinite, a contradiction. □

Theorem 13 *For a robot with only angular odometry and a contact sensor in any polygonal environment X, no localizing sequence exists.*

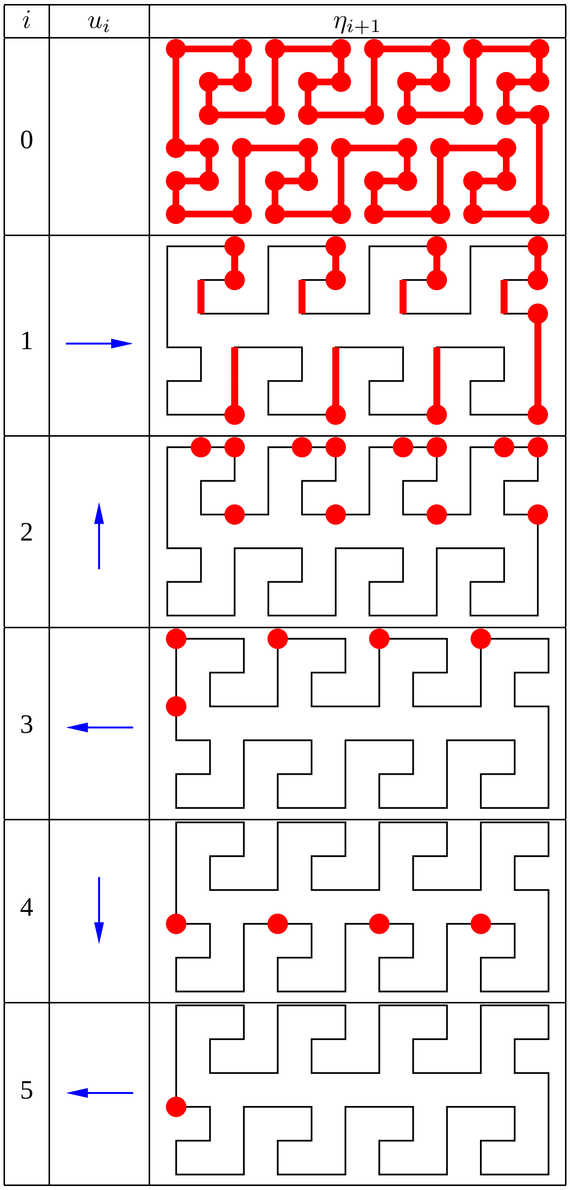

Fig. 8. A localizing sequence computed by our algorithm for a highly symmetric environment.

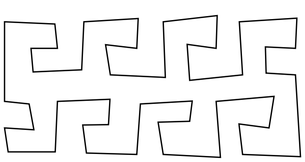

Fig. 9. A modified version of the environment from Fig. 8 in which the symmetries have been broken. Our algorithm generates a 26 step localizing sequence for this environment.

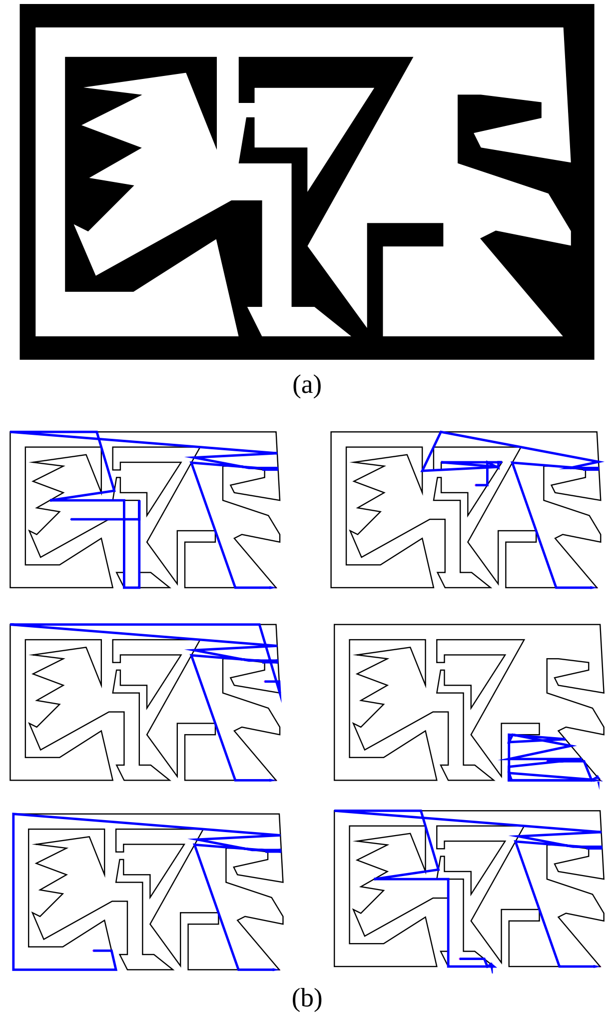

Fig. 10. (a) An irregular environment for which the localizing sequence computed by our algorithm requires 30 steps. (b) Execution traces of this localization sequence for 6 different starting positions. For each starting position, the final position is the lower right corner of the environment.

Proof Sketch: Suppose such a sequence $u_1, \ldots, u_K$ exists. Let $e_1, \ldots e_n$ denote the set of edges of ∂X, and let $\mathrm{Rot}(v, \phi)$ 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 $\mathrm{Rot}(e_j, \theta)$ parallel, then $u_1, \ldots, u_K$ 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 $\mathrm{Rot}(e_j, \theta)$ has measure 0. Therefore $u_1, \ldots, u_K$ fails for nearly every initial condition. $\square$

VI. Conclusion

This paper presented a technique for localization 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.

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 9. It is not immediately clear whether a similar method can be devised for environments that are not simply connected.

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.

In this paper 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. 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 [7]. 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.

ACKNOWLEDGMENT

This work was partially supported by ONR Grant N00014-02-1-0488 and by NSF-CONACyT Award 0296126. The authors are indebted to Steve Lindemann and Benjamin Tovar for helpful discussions.

REFERENCES

[1] 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.

[2] M. Blum and D. Kozen. On the power of the compass (or, why mazes are easier to search than graphs). In *Proc. Annual Symposium on Foundations of Computer Science*, pages 132–142, 1978.

[3] G. Boothroyd, C. Poli, and L. E. Murch. *Automatic Assembly*. Marcel Dekker, Inc., 1982.

[4] M. Brokowski, M. A. Peshkin, and K. Goldberg. Optimal curves fences for part alignment on a belt. *ASME Journal of Mechanical Design*, 117(1), March 1995.

[5] B. Chazelle and L. G. Guibas. Visibility and intersection problems in plane geometry. *Disc. and Comp. Geom.*, 4:551–589, 1989.

[6] B. R. Donald. On information invariants in robotics. *Artif. Intell.*, 72:217–304, 1995.

[7] 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.

[8] G. Dudek and C. Zhang. Vision-based robot localization without explicit object models. In *IEEE Int. Conf. Robot. & Autom.*, pages 76–82, 1996.

[9] M. A. Erdmann and M. T. Mason. An exploration of sensorless manipulation. *IEEE Trans. Robot. & Autom.*, 4(4):369–379, August 1988.

[10] K. Y. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10:201–225, 1993.

[11] K. Y. Goldberg and M. T. Mason. Bayesian grasping. In *IEEE Int. Conf. Robot. & Autom.*, 1990.

[12] L. J. Guibas and J. Hershberger. Optimal shortest path queries in a simple polygon. *J. Comput. Syst. Sci.*, 39(2):126–152, 1989.

[13] 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. 1st Workshop on Algorithmic Foundations of Robotics*, pages 269–282. A.K. Peters, Wellesley, MA, 1995.

[14] R. Hinkel and T. Knieriemen. Environment perception with a laser radar in a fast moving robot. In *Proceedings of Symposium on Robot Control*, 1988.

[15] I. Kamon and E. Rivlin. Sensory-based motion planning with global proofs. *IEEE Trans. Robot. & Autom.*, 13(6):814–822, December 1997.

[16] J. M. Kleinberg. The localization problem for mobile robots. In *IEEE Symposium on Foundations of Computer Science*, pages 521–531, 1994.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] W. Renken. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 1993.

[22] S. Thrun. Probabilisitic algorithms in robotics. *AI Magazine*, 21(4):93–109.

[23] B. Tovar, L. Guilamo, and S. M. LaValle. Gap Navigation Trees: Minimal representation for visibility-based tasks. In *Proc. Workshop on Algorithmic Foundations of Robotics*, 2004.

[24] 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.

[25] 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.

[26] D. E. Whitney. Real robots don't need jigs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1986.

[27] 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.