Learning Combinatorial Map Information from Permutations of Landmarks

Benjamín Tovar, Luigi Freda and Steven M LaValle

The International Journal of Robotics Research published online 4 October 2010
DOI: 10.1177/0278364910381416

The International Journal of
Robotics Research
00(000) 1–13

DOI: 10.1177/0278364910381416
ijr.sagepub.com

Learning Combinatorial Map Information from Permutations of Landmarks

Benjamín Tovar[1], Luigi Freda[2], Steven M. LaValle[3]

Abstract

This paper considers a robot that moves in the plane and that is able to sense only the cyclic order of landmarks with respect to its current position. No metric information is available regarding the robot or landmark positions; moreover, the robot does not have a compass or odometers (i.e., knowledge of coordinates). We carefully study the information space of the robot, and establish its capabilities in terms of mapping the environment and accomplishing tasks, such as navigation and patrolling. When the robot moves exclusively inside the perimeter of the set of landmarks, the information space may be succinctly characterized as an order type *that provides information powerful enough to determine which points lie inside the convex hulls of subsets of landmarks. In addition, if the robot is allowed to move outside the perimeter of the set of landmarks, the information space is described with a* swap cell decomposition*, that is, an aspect graph in which each aspect is a cyclic permutation of landmarks. Finally, we show how to construct such decomposition through its dual, using two kinds of feedback motion commands based on the landmarks sensed.*

Keywords

Mapping, mobile and distributed robotics SLAM, localization, learning and adaptive systems, cognitive robotics

1. Introduction

In this paper we study a robot moving in the plane with very limited sensing: it knows only the cyclic ordering of landmarks as they appear from the robot's current position (no distance information can be measured and there are no other sensors). Given the sensor limitations, we avoid estimation of the position of the robot and of landmarks, and instead concentrate on the landmarks' relative orderings to construct the algorithms. Eventually, the representation of the environment (i.e., the *map*), is a sequence of cyclic permutations of landmarks. After establishing what the robot can learn from its simple sensor, we then illustrate the kinds of tasks that it can solve, including surveillance/patrolling around the perimeter (convex hull) of landmarks.

In robotics, landmarks have classically been used for navigation (Betke and Gurvits 1994; Borenstein et al. 1996; Hu and Gu 2000; Hayet et al. 2002; Steck and Mallot 2000; Shimshoni 2002). For example, in works such as Lazanas and Latombe (1992) and Tashiro et al. (1995), robot paths that minimized localization errors were found using pre-images (Erdmann 1986) for a known arrangement of landmarks. In more recent years, landmarks have been used to construct geometric models of the environment, along with an explicit estimation of the robot position. In the most well-known form of simultaneous localization and mapping (SLAM) (Simmons and Koenig 1995; Thrun et al. 1998; Montemerlo et al. 2002; Yamauchi et al. 2002; Parr and Eliazar 2003), the addition of some Gaussian assumptions allows the estimation of the position of the robot as well as the position of landmarks by probabilistic filters. These approaches have achieved impressive implementation success, but the probabilistic information spaces they generate are hard to characterize given that they are infinite-dimensional.

In works such as those of Bulata and Devy (1996), Dean et al. (1993), Dudek et al. (1993), Kuipers (2000), Kuipers et al. (1993), Remolina and Kuipers (2004), Shatkay and Kaelbling (1997) and Smith and

[1] Department of Mechanical Engineering, Northwestern University, Evanston, IL, USA
[2] U.T.R.I. S.p.A., Trieste, Italy
[3] Department of Computer Science, University of Illinois, Urbana, IL, USA

Corresponding author:
Benjamín Tovar
Department of Mechanical Engineering
Northwestern University
2145 Sheridan Road, Evanston, IL 602081, USA
Email: b-tovar@northwestern.edu

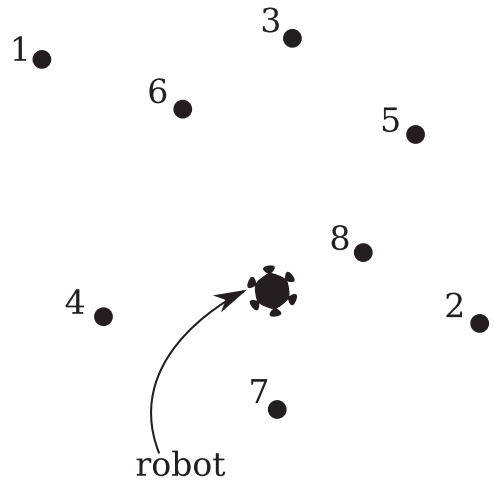

Fig. 1. The landmark order detector gives the cyclic order of the landmarks around the robot. Note that only the cyclic order is preserved, and that the sensed angular position of each landmark may be quite different from the real one. Thus, the robot knows only that the sequence of landmarks detected is [7,2,8,5,3,6,1,4], up to a cyclic permutation.

Cheeseman (1986), the environment is represented with graph-like spatial descriptions. In this context, a vertex in a graph represents a *place*, and edges represent possible paths among places. In this context, places are commonly defined as environment locations from which a certain arrangement of landmarks is visible.

In this paper we follow a minimalist approach, in which our goal is to have the robot and its sensors to be as simple as possible. This philosophy has been successful in several works (e.g., Brost (1991), Erdmann and Mason (1988) and Levitt and Lawton (1990); Goldberg (1993)). Is it really necessary for the robot to build an explicit representation of the environment? Is knowing the exact position of the robot crucial for the completion of the task? While the models and assumptions used in this paper were inspired by basic sensing issues in robotics, there are also closely related to sensor networks, which are becoming increasingly important in security applications: the landmarks can be imagined as "sensors" in a network, and the robot provides the "communication" link between them. We hope that the insights obtained from our work may help in the development of robust, cost-effective robotic systems and sensor networks applied to surveillance, tracking, pursuit-evasion, and other sensor-based problems. In this sense, the work presented in Levitt and Lawton (1990) is similar to the ideas presented here; however, in this paper the cyclic permutation sensor amounts *to detecting when two landmarks cross in the field of the robot*, whereas in Levitt and Lawton (1990), the sensor detects when *the robot becomes collinear with a pair of landmarks*. Therefore, the sensor considered in Levitt and Lawton (1990) is more powerful.

The ideas in this paper are based on the material presented in Tovar et al. (2007) and Freda et al. (2007).

2. Basic Definitions

We model a robot as a moving point in the plane. Its configuration $q \in SE(2) = \mathbb{R}^2 \times S^1$ is described by its position in $\mathbb{R}^2$ and a heading angle in S^1. Let L be a finite set of n points in $\mathbb{R}^2$, and let $s : \mathbb{R}^2 \to \{0, \ldots, n\}$ be a mapping such that every point in L is assigned a different integer in $\{1, \ldots, n\}$, and $s(p) = 0$ if $p \notin L$. The mapping s is referred to as a *landmark identification function* and L is referred to as the set of *landmarks*. For landmark $p \in L$, $s(p)$ is referred to as the *landmark label* of p. In the following, for any landmark $p_i \in L$, the subscript indicates the landmark label (i.e., $s(p_i) = i$).

Let $\mathcal{L}$ be the set of all possible finite subsets of $\mathbb{R}^2$ (i.e., $\mathcal{L}$ is the set of all possible landmarks arrangements). We define the *state* as the pair $\mathbf{x} = (q, L)$, and the *state space* X as the set of all such pairs $(SE(2) \times \mathcal{L})$. A landmark sensor is defined in terms of a landmark identification function s. Such a sensor is called a *landmark cyclic order detector*, and it is denoted with $\mathrm{lcd}_s(\mathbf{x})$, for $\mathbf{x} \in X$. The landmark order detector gives the counterclockwise cyclic permutations of landmark labels as seen from the current state (see Figure 1). Note that the robot does not have any coordinate estimate of its position, nor the position of the landmarks, and that the landmark order detector does respect the cyclic order of landmarks, but does *not* measure the angle between them. No metric information is readily available, and moreover, $\mathrm{lcd}_s(\mathbf{x})$ does not provide by itself any notion of front, back, left or right to the robot. We assume that landmarks do not obstruct the visibility of the robot (i.e., the landmarks are considered *transparent*). When two landmarks appear in the same angular position, their ordering in the sensor reading is arbitrary, but does not change until the crossing is completed. All of our results do hold for *opaque* landmarks, but we ignore this case for clarity of exposition. We also assume that the landmarks are in general position (i.e., no three landmarks are collinear). Furthermore, the landmark order detector has infinite range. (We discuss how some of these assumptions may be removed in Section 8.)

When does the perceived cyclic permutation change? Each pair of landmarks supports two pairs of half lines, such that each half line has its endpoint in one landmark, but does not contain the other. Specifically, for two different landmarks $p_i, p_j \in L$ (see Figure 2), consider the half line that starts at p_i, and is collinear with but does not contain p_j. We refer to such a half line as the *swap-line* $\vec{p_i p_j}$ The swap-line $\vec{p_j p_i}$ is defined in a similar manner. Note that when the robot is arbitrarily close to $\vec{p_i p_j}$ or $\vec{p_j p_i}$, p_i and p_j appear consecutive in $\mathrm{lcd}_s(\mathbf{x})$,

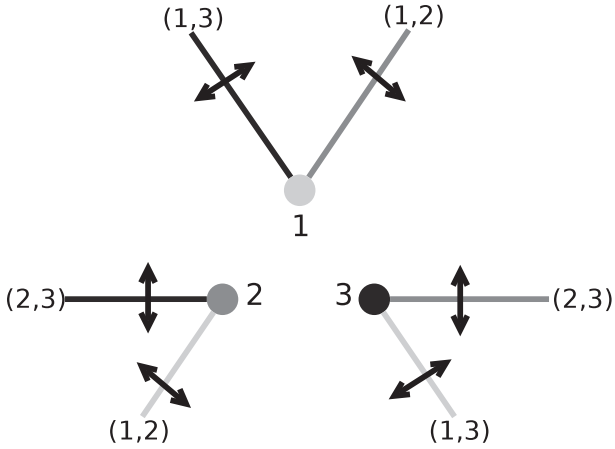

Fig. 2. Swap lines. Crossing a half line swaps the order of the respective landmarks in the reading of the landmark order detector. Such half lines are called *swap lines*.

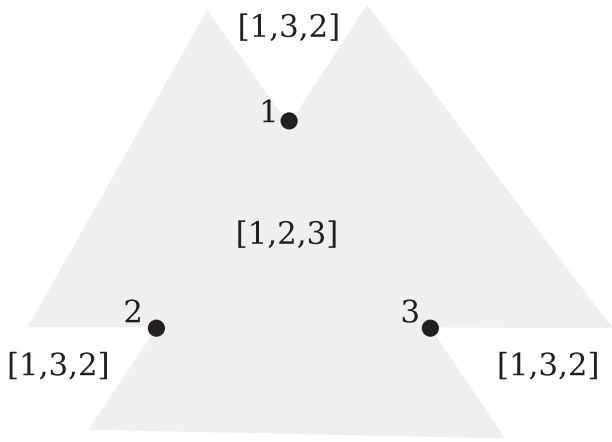

Fig. 3. Cyclic permutations of three landmarks. Purely by sensing, the robot cannot even know if it is inside the convex hull defined by the three landmarks. Nevertheless, the orientation of the triangle (the counterclockwise cyclic order of the landmarks as sensed from inside their convex hull) can be determined with an information state.

and when the robot crosses one of the swap lines, there is a change in the cyclic permutation sensed.

We assume that the robot can choose a particular landmark label $s(p)$ and move towards the landmark position p. This landmark tracking motion primitive is denoted by track($s(p)$). For simplicity, we assume that when the robot arrives at p, $\text{lcd}_s(\mathbf{x})$ no longer detects the landmark just tracked[1]. The motion primitive track($s(p)$) therefore may terminate only when there is a change in the reading of $\text{lcd}_s(\mathbf{x})$. Although we do not discuss here the real implementation of the landmark order detector, it can be constructed, for example, with an omnidirectional camera with standard feature tracking software (i.e., filter-based tracking (Thrun et al. 1998; Montemerlo et al. 2002; Lowe 2003; Thrun et al. 2005)).

3. Order Types and Landmarks

Given the sensing and control models introduced, consider the robot as it moves in the environment. The only information received consists of the changes in the cyclic permutations (e.g., for three landmarks, only two sensing readings are possible). Purely by sensing, the robot cannot even know if it is inside the convex hull defined by the three landmarks (see Figure 3). Nevertheless, consider the robot traveling from the landmark with label 1 to the landmark with label 2. Since the reading from the landmark order detector follows a counterclockwise order, the robot can determine whether the landmark with label 3 is to the *left* or the *right* of the directed segment that connects the landmarks with labels 1 and 2. Thus, the robot can combine sensing with action histories to recover some structure of the configuration of landmarks.

We generalize the previous idea to encode information states with the concept of *order type*. Two ordered sets A and B are said to have the same order type if there is a bijection $f : A \to B$ such that for all $a_1, a_2 \in A$, $a_1 \leq_A a_2 \Leftrightarrow f(a_1) \leq_B f(a_2)$, in which $\leq_A$ and $\leq_B$ are the relations defining the orders of A and B, respectively. Think of this definition in the following manner. Sets A and B have the same order type if they have the same number of smallest elements, the same number of second-to-smallest elements, etc. For a configuration of labeled points, the order relation $\leq$ can be defined through the relative orientation of three points, which is computed as follows (Goodman and Pollack 1983). The ordered triplet of points p_1, p_2, p_3, with $p_i = (x_i, y_i)$, is said to have positive orientation if the determinant

$$\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} \tag{1}$$

is strictly bigger than 0, and this is denoted by $p_1 p_2 p_3^+$. Negative orientation is defined in a similar manner, and denoted by $p_1 p_2 p_3^-$. Given our general position assumption, this determinant cannot be zero. The order type of a labeled configuration of points P is determined by the relative orientation of each triplet of points in P. The order type of the configuration of points can be encoded by a function defined as follows:

$$\Lambda(i,j) = \{k \mid p_i p_j p_k^+ \text{ for } p_i, p_j, p_k \in P\}. \tag{2}$$

The function Λ takes the indices i, j of two points $p_i, p_j \in P$, and returns the indices corresponding to the points in $P \setminus \{p_i, p_j\}$ positively oriented with respect to p_i and p_j (in that order). For example, following Figure 1, $\Lambda(3,7) = \{2,5,8\}$, and $\Lambda(7,3) = \{1,4,6\}$. Alternatively, the order type can be specified with the

function $\lambda(i,j) = |\Lambda(i,j)|$. It is not immediately clear that, once the function λ is known, Λ can be deduced. Surprisingly, this is not only true for the plane, but for any dimension (Goodman and Pollack 1983). The order types generalize the common notion of linear sorting for real numbers into the so-called *geometric sorting*. Here, minimum and maximum become extremal subsets of points in P. For example, if $\lambda(i,j) = 0$, then there are no points to the left of the directed edge $\overline{p_i p_j}$, and both p_i and p_j belong to the boundary of the convex hull. Note that the other direction works too; in this case $\lambda(j,i)$ will be a non-unique maximum of λ.

3.1. Oriented Matroids

Consider a line arrangement in the plane, and the decomposition it induces. Some properties of this decomposition do not depend on the fact that the lines are *straight*, but only on whether any two of them intersect in at most one point. These lines (that are not necessarily straight) are called *pseudo-lines*, and such an arrangement is called an arrangement of pseudo-lines. We refer to properties such as these as *combinatorial properties* (Björner et al. 1993). For a given order type, Λ is independent of homogeneous transformations applied to the set of points. In addition, we have great freedom in moving points around before the value of Λ changes. There is a strong connection between the combinatorial properties of pseudo-line arrangements and sets of points. In fact, *they are the same*, and the structures that make this apparent are called *oriented matroids*.

Oriented matroids are combinatorial abstractions of point configurations over the reals, of real hyperplane arrangements, of convex polytopes, and of directed graphs (Björner et al. 1993). There are several equivalent ways of defining an oriented matroid. In our case, given that we are dealing with points in the plane, the most useful is in terms of *chirotopes* (Björner et al. 1993), defined as the sign of the determinant in Equation 1 for each triplet of points. This allows us to describe the combinatorial structure of the set of landmarks with the following axioms (Knuth 1992).

Axiom 1. *The orientation of a triangle is independent of a cyclic reordering of its vertices:*
$p_1p_2p_3^+ \implies p_3p_1p_2^+$.

Axiom 2. *A triangle cannot have two orientations:*
$p_1p_2p_3^+ \implies \neg p_1p_3p_2^+$.

Axiom 3. *A triangle has at least one orientation:*
$p_1p_2p_3^+ \vee p_1p_3p_2^+$.

Axiom 4. *Inside a triangle relation:*
$p_1p_2p_4 \wedge p_2p_3p_4 \wedge p_3p_1p_4 \implies p_1p_2p_3$.

Axiom 5. *Transitivity relation:*
$p_4p_5p_1 \wedge p_4p_5p_2 \wedge p_4p_5p_3 \wedge p_4p_1p_2 \wedge p_4p_2p_3 \implies p_4p_1p_3$.

We use Λ to record partial information about the landmarks' arrangement as the robot moves. Given the previous axioms, it is clear that Λ cannot have arbitrary values. What is perhaps surprising is that even if the arguments for Λ satisfy the previous axioms, they may not correspond to points in the Euclidean plane. In particular, they may fail to satisfy Pappus's hexagon theorem (Knuth 1992; Björner et al. 1993)[2]. Those oriented matroids that do correspond to points in the plane are called *realizable*. In fact, given our definition of order type based on determinants, order type is just a synonym for realizable oriented matroid. In Goodman and Pollack (1983), the number of realizable oriented matroids for n points in the plane was found to be $2^{\theta(n \log n)}$.

3.2. The Information Space

Consider the state $\mathbf{x} = (q, L)$, which is unknown to the robot. Although $\mathbf{x}$ is unknown, information about q and L is available to the robot. In particular, partial knowledge of the order type of L can always be computed. Also, using tracking commands together with readings from $\mathrm{lcd}_s(\mathbf{x})$, the position of the robot can be determined to be either on a landmark, or in the segment between two landmarks. An information state is defined as the pair (Q', Λ'), in which Q' refers to the possible positions of the robot with respect to the landmarks, and Λ' is the partial knowledge of Λ. The information space $\mathcal{I}$ is the space of all such pairs.

Let $\mathcal{I}(L)$ be the information states for which Λ' does not contradict the configuration of landmarks in the environment. Note that, up to a relabeling of the landmarks, $|\mathcal{I}(L)|$ is finite. This is because for n landmarks there are $2^{\theta(n \log n)}$ possibilities for Λ'. Also, the number of distinct sets Q' of possible positions is bounded by the number of combinatorial elements of the line arrangement drawn from the lines passing through each pair of landmarks.

3.3. Retrieving the Order Type

The order type concepts extend naturally to our landmark setting, using the landmark labels as the indices for Λ. Of course, the robot cannot compute the determinants, because it lacks any coordinates. Nevertheless, it is possible to compute Λ for any pair of landmark labels. For this computation we establish the following lemma:.

Lemma 1. *Let the output of the sensor be of the form* $lcd_s(\mathbf{x}) = [X, i, Y, j, Z]$, *in which* X, Y, Z *are subsequences of* $lcd_s(\mathbf{x})$ *separated by the labels corresponding to landmarks* p_i *and* p_j. *If the robot is on the line*

segment $\overline{p_ip_j}$, and its heading is pointing towards p_j, then $\Lambda(i,j) = X \cup Z$ and $\Lambda(j,i) = Y$.

Proof. To determine $\Lambda(i,j)$, we are looking for the landmarks to the *left* of the directed segment $\overline{p_ip_j}$. Consider any point in the interior of $\overline{p_ip_j}$ as a pivot of a counterclockwise radial sweep starting at p_j and ending at p_i. It is clear that all the landmarks swept lie to the left of $\overline{p_ip_j}$. If the robot is placed according to the conditions of the lemma, this sweep can be obtained from the cyclic sequence given by $\text{lcd}_s(\mathbf{x})$, starting at j, until i is found. By symmetry, $\Lambda(j,i)$ is also found. □

Strategy 1. *Given landmarks p_i and p_j, determine which landmarks lie to the left of the directed line from p_i to p_j $\Lambda(i,j)$.*

Description. The value of $\Lambda(i,j)$ is determined as follows. The robot is commanded to track landmark p_i until i disappears from $\text{lcd}_s(\mathbf{x})$. This means that now the robot is at p_i. Next, the robot is commanded to track p_j and, at the moment when i is detected again, the robot is guaranteed to be on $\overline{p_ip_j}$, pointing towards p_j. Applying Lemma 1 to the sensor reading, $\Lambda(i,j)$ and $\Lambda(j,i)$ are found.

---

4. Solving Some Simple Robotic Tasks

In this section we present some simple tasks that can be solved using the concepts presented previously. In the following examples, L is the set of landmarks detected, and $n = |L|$.

4.1. Landmarks Inside a Triangle

The task is to compute the subset of landmarks of L that are inside of the triangle defined by the landmarks labeled with i,j and k. In other words, if $k \in \Lambda(i,j)$, the robot should determine $\Lambda(i,j) \cap \Lambda(j,k) \cap \Lambda(k,i)$, or if $k \notin \Lambda(i,j)$, then $\Lambda(j,i) \cap \Lambda(i,k) \cap \Lambda(k,j)$ should be computed. These two cases correspond to the two possible orientations of a triangle (as defined before with the determinant). Since both the orientation of the triangle and the needed values of Λ can be computed with Lemma 1, we use this simple example to introduce a motion strategy that deals with control uncertainty. Refer to Figure 4. The problem here is that the internal angle of the triangle at landmark i is obtuse. This gives little margin of error for the control, and the triangle orientation may not be computed correctly. As it can be seen for landmarks j and k with acute angles, the error in the control should be almost π before the orientation is computed incorrectly.

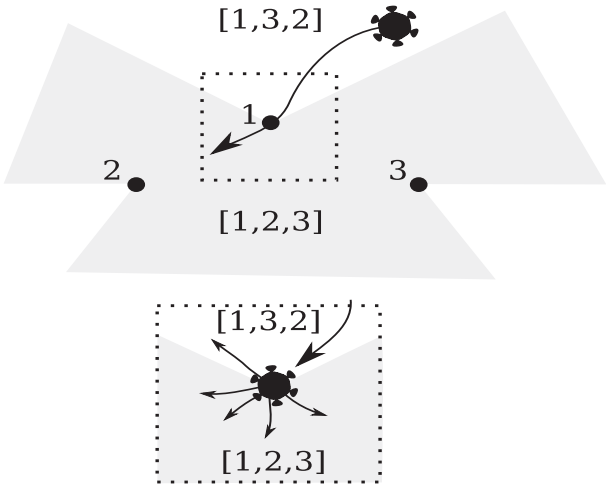

Fig. 4. Triangle orientation error. A small control error may find the wrong orientation for the triangle. On the bottom, if the robot follows the top-left arrow, the orientation is not computed correctly.

Strategy 2. *Robust triangle orientation measurement.*

Description. Given that a triangle has at most one obtuse angle, the robot repeats Strategy 1 three times, one for each edge of the triangle. If this strategy yields an orientation more than once, it is taken as the correct orientation of the triangle.

The incorrect reading is made if the robot crosses one of the swap lines supported by pairs of landmarks of the triangle. The angle between two swap lines corresponds to an opposite angle of an internal angle of the triangle. In the worst case, the three internal angles of the triangle are as close to π as possible, which means the triangle is equilateral with each of the internal angles being $\pi/3$. This means that the control error allowed corresponds to the supplementary angles of the internal angles, which allows an error of $2\pi/3$.

With Strategy 2, assume that we find that $j \in \Lambda(k,i)$ but, as a result of errors in control, the observed value of $\Lambda(k,i)$ is incorrect. Note that the triangle $p_ip_jp_k$ has the same orientation of a triangle $p_ip_lp_k$ if and only if p_l is inside the triangle $p_ip_jp_k$. With this, the robot can compute the orientation of the triangles $p_ip_lp_k$, with $p_l \in \Lambda(i,j) \cap \Lambda(j,k)$, and select those p_l which yield the same orientation as $p_ip_jp_k$.

4.2. Boundary of the Convex Hull

Let $\text{hull}(L)$ be the convex hull of the set of landmarks. In this task the robot should determine which landmarks are on the boundary $\partial\text{hull}(L)$ of $\text{hull}(L)$. This task can be solved easily (and efficiently) by finding which landmarks do not belong to the interior of any

triangle defined by three landmarks. However, a significantly more efficient algorithm can be constructed based on the well-known *Graham scan* for the computation of the convex hull (Graham 1972). In its regular setting, the Graham scan starts by finding one landmark in the convex hull (e.g., the leftmost), and sorting the remaining landmarks radially around it. Next the landmarks are considered three by three according to this radial order. Particular landmarks are included or removed from the boundary of the convex hull depending on whether they lie to the left or the right of the landmarks in the triplet.

In our setting, we need to find first a pair of landmarks that appear consecutively $\partial\text{hull}(L)$.

Strategy 3 (!t)**.** *Find a pair of landmarks in* $\partial hull(L)$.

Description. The strategy is based on an iteration that tracks some landmarks sequentially. For clarity of exposition, we make the label of the landmark tracked at step i to be i.

We need a pairs of landmarks for which the value of Λ is zero, and we want to find this pair without computing the entire Λ. Initially, a pair of landmarks is arbitrarily selected, p_1 and p_2, and the robot tracks p_1 until 1 disappears from $\text{lcd}_s(\mathbf{x})$. Set $i = 1$, and thereafter we have the following:

1. The motion track$(i+1)$ is executed. During its execution, $\Lambda(i, i+1)$ and $\Lambda(i+1, i)$ are computed. If one of them is zero, then terminate, since the desired pair has been found.
2. Let $i+2$ be the label following $i+1$ in $\text{lcd}_s(\mathbf{x})$, immediately before $i+1$ disappears from $\text{lcd}_s(\mathbf{x})$.
3. Increment i, and go to step 1.

Theorem 2. *Strategy 3 finds a pair of landmarks in* $hull(L)$ *with* $O(n)$ *tracking motion primitives.*

Proof. Consider the swap line $p_i\vec{p}_{i+1}$, and "sweep it" radially counterclockwise around p_{i+1}. Given that $i+2$ is the label following $i+1$ in $\text{lcd}_s(\mathbf{x})$ before $i+1$ disappears, it follows that the first landmark found in the sweep is p_{i+2}. There are three cases: □

1. $p_{i+1} \in \partial\text{hull}(L)$. This implies that $p_{i+2} \in \partial\text{hull}(L)$, and the desired pair has been found.
2. $p_{i+2} \in \partial\text{hull}(L)$. Similar to the previous case, but the pair is guaranteed to be found in the next iteration.
3. Otherwise, observe that if $p \in \partial\text{hull}(L)$ and $s(p) \in \Lambda(i, i+1)$, then $s(p) \in \Lambda(i+1, i+2)$. This implies that the iteration cannot loop forever. To see this, suppose $L' \subset L$ is a minimal set of landmarks that cause a loop. Consider the Λ values for consecutive landmarks in $\partial\text{hull}(L')$, and a landmark $p \in \partial\text{hull}(L)$. Since $p \notin \partial\text{hull}(L')$, then, for some i, $s(p) \in \Lambda(i, i+1)$, but $s(p) \notin \Lambda(i+1, i+2)$, which is a contradiction.

Based on Strategy 3, $\partial\text{hull}(L)$ is easily computed.

Strategy 4. *Find* $\partial hull(L)$ *of the set of landmarks* L.

Description. Perform Strategy 3, and assume that the pair of landmarks found on $\partial\text{hull}(L)$ is (p_i, p_j), with $\Lambda(j, i) = \emptyset$. Using Strategy 1, the robot can be positioned somewhere along the line segment joining p_i and p_j. At this point, if we assume $|L| > 2$, the sensor reading has the form $\text{lcd}_s(\mathbf{x}) = [i, j, k, X]$, in which X is a (perhaps empty) sequence of landmarks. It follows that $\Lambda(k, j) = \emptyset$, since p_k is the first landmark found after a radial sweep around p_j. This process is repeated, but now between p_j and p_k, until p_i is found again.

In Strategy 4, the more expensive action in terms of motion primitives is the execution of Strategy 3. Therefore, Strategy 4 finds the convex hull of L using $O(n)$ tracking motion primitives.

5. Patrolling

In this section we model robotic tasks in which a robot carefully monitors some area of the environment. As a concrete example, imagine an unmanned flying vehicle above a terrain. The flying vehicle is given a set of way points, which are visited sequentially. In this example, we solve a version of the patrolling problem in which the robot performs loops around a given subset of the landmarks. Formally, let $W \subset L$, with $W \cap \partial\text{hull}(L) = \emptyset$. The *patrolling* task for set W is defined as follows: *Find* $M \subset L$, *such that* $W \subset M$, $\partial hull(M) \cap W = \emptyset$ *and the size of* M *is minimal.*

To solve this task, the *dual* of the configuration of landmarks is introduced. The dual of a landmark p_i, with $p_i = (p_i^x, p_i^y)$, is defined as the line $p_i^* = (p_i^x x + p_i^y y)$. There are well-known properties of such dual arrangements (Edelsbrunner 1987; de Berg et al. 1997), such that the intersection of two lines p_i^* and p_j^*, which defines a vertex in the dual, corresponds to the line passing through p_i and p_j in the primal space. Also, ordering relations are respected. Namely, if a point p is above a line m in the primal space, then the point m^* is above the line p^* in the dual. Figure 5 shows the dual arrangement for a configuration of four landmarks.

A line arrangement can be encoded with a sequence of permutations (Edelsbrunner 1987). This is done by sweeping a vertical line from left to right in the line arrangement, recording the vertical order of the intersections of the vertical line with the lines of the arrangement. Such permutations can be obtained from the primal space. As shown in Figure 5, when the robot travels

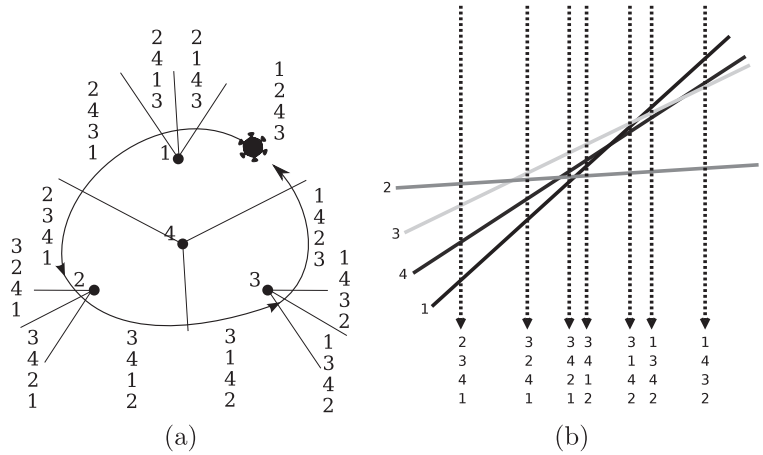

Fig. 5. Retrieving the permutations that encode the configuration of landmarks. In (a) the robot travels outside the convex hull of a set of landmarks. This is naturally expressed in the dual line arrangement in (b).

outside a convex hull of a set of landmarks, a vertex of the line arrangement is read whenever two labels swap places from one permutation to the other. Since a vertex in the dual corresponds to a line in the primal, only $\binom{n}{2}+1$ permutations are needed to describe the line arrangement, when actually $2\binom{n}{2}$ could be read by the robot traveling outside the convex hull. These permutations have other nice symmetric properties, and the reader is referred to Edelsbrunner (1987).

There some minor complications for obtaining such permutations with the robot model described. First, the robot cannot, in general, travel outside a convex hull, since it only knows how to track landmarks. To solve this, we need the following lemma.

Lemma 3. *Let L be a set of landmarks and let Z be a subsequence of $\mathrm{lcd}_s(\mathbf{x})$ containing only the labels corresponding to the landmarks in $\partial hull(L)$ (elements of Z may not necessarily appear consecutively in $\mathrm{lcd}_s(\mathbf{x})$). Then Z is the same circular subsequence for any position of the robot inside $hull(L)$.*

Proof. For labels i and j to switch places in Z, at some point they should map to the same position in the landmark order detector. This means that p_i, p_j and the robot are collinear, and that either p_i is contained in the line segment from the robot position to p_j, or p_j is contained in the line segment from the robot position to p_i. Since no three landmarks are collinear, the robot must be outside hull(L). □

From Lemma 3, the robot can obtain the counterclockwise order of the landmarks on the boundary of the convex hull. Instead of traveling properly outside the convex hull, the robot tracks each of the landmarks in the boundary sequentially, following the order found. When the robot arrives at a landmark $p_i \in \partial\mathrm{hull}(L)$, the corresponding permutations as if the robot was travelling outside hull(L) but arbitrarily close to ∂hull(L) are generated as follows. First note that the only swap lines crossed when the robot moves arbitrarily close to p_i are the swap lines that start at p_i. Therefore, only swaps involving i occur in $\mathrm{lcd}_s(\mathbf{x})$. Furthermore, the order in which these swap lines are crossed is determined by the cyclic order given by $\mathrm{lcd}_s(\mathbf{x})$ when the robot arrives at p_i. Finally, the landmark order detector gives cyclic permutations, but the arrangement description needs the extremal point in a particular direction to come first. This is easily solved by ordering the cyclic permutation such that the label of the landmark being tracked appears first. The following lemma is a well-known result for dual line arrangements (as expressed in our framework).

Lemma 4. *Let L^* be the set of lines dual to the set of landmarks L. Let m_v be a vertical line, and let $[p_1^*, p_2^*, \ldots, p_n^*]$ for $p_i^* \in L^*$ be sorted according to the y-coordinate of the intersection between m_v and p_i^*. Then the landmarks p_1 and p_n belong to $\partial hull(L)$.*

Proof. Let m_v intersect the x-axis at x. Consider all the lines intersecting the convex hull of L with slope x. Since the duality transformation is order-preserving, then p_1 is below and p_n is above all such lines. □

Corollary 5. *Let L^* be the set of lines dual to the set of landmarks L. Let m_v be a vertical line, and let $[p_1^*, p_2^*, \ldots, p_{n-1}^*, p_n^*]$ for $p_i^* \in L^*$ be sorted according to the y-coordinate of the intersection between m_v and p_i^*. Let p_1, p_2, p_{n-1}, and p_n be the duals of p_1^*, p_2^*, p_{n-1}^*, and p_n^* respectively. Then p_2 is in $\partial hull(L \setminus \{p_1\})$, and p_{n-1} is in $\partial hull(L \setminus \{p_n\})$.*

Proof. Consider $L^* \setminus \{p_1^*\}$ and $L^* \setminus \{p_n^*\}$, and apply Lemma 4. □

Strategy 5. *Given a set $W \subset L$ of landmarks to patrol, find $M \in L$ such that $W \subset M$, $|M|$ is minimal, and $\partial hull(W) \cap \partial hull(M) = \emptyset$.*

Description. Corollary 5 immediately provides a patrolling strategy as follows. Assume that the robot has computed the permutations encoding the dual arrangement of L. The strategy is based on the following iteration. Set $L_0 = L$. For $u \geq 0$, find $p \in \partial\text{hull}(L_u)$ such that $\partial\text{hull}(L_u \setminus \{p\})$ does not contain any landmark in W. If no such landmark exists, set $M = L_u$. Else, set $L_{u+1} = L_u \setminus \{p\}$ and repeat.

By Corollary 5, landmarks can be removed from L_u, and the boundary of the convex hull can be read directly from the permutations encoding the dual arrangement. Moreover, those permutations with the landmark removed encode the dual arrangement of L_{u+1}. A landmark may not be removed if this will make a landmark in W become the first, or last, in the permutations encoding the dual arrangement. The robot can then *patrol* the landmarks in W by following the landmarks on the boundary of M in counterclockwise order.

6. Navigation

The final task described is navigation. Consider the partition of the plane in which locations inside the same cell generate the same reading in the landmark order detector. This can be considered as an aspect graph (Koenderink and van Doorn 1976), in which a cyclic permutation is an *aspect* of the configuration of landmarks. This partition is uniquely determined by the swap lines. In this framework, a navigation goal is a sequence g of landmark labels. Formally, the *navigation* task is defined as follows: *Move the robot such that a state* $\mathbf{x}$ *with* $lcd_s(\mathbf{x}) = g$ *is reached. Report if* g *cannot be realized given the configuration of landmarks in the plane*.

6.1. Swap Cell Decomposition

We refer to the decomposition induced in $\mathbb{R}^2$ by all the swap lines as the *swap cell decomposition*. The 0-cells are landmarks, the 1-cells are swap lines, and the 2-cells, called the *swap cells*, are connected open regions of $\mathbb{R}^2$ from which $\text{lcd}_s(\mathbf{x})$ reports the same cyclic permutation. For a set of landmarks L, let K_L be the set of all of the swap cells. Abusing notation, for swap cell $C \in K_L$, let $\text{lcd}(C)$ be the reading of the landmark cyclic order detector from a point in C.

The swap cell decomposition can be naturally encoded in a graph, which is an aspect graph (Koenderink and van Doorn 1976) in which a cyclic permutation is an *aspect* of the configuration of landmarks. We will explore this idea in Section 7. For now, in this section we are interested in moving the robot between swap cells without the complete knowledge of the swap cell decomposition. As in Section 5, the robot can easily learn the order type of a set of landmarks L, by traveling once around the convex hull boundary. Therefore, in this section, we assume that a complete knowledge of Λ is available.

Given that the robot cannot travel outside $\text{hull}(L)$, the navigation task is only defined for cells whose intersection with $\text{hull}(L)$ is not empty. The navigation task is meaningful if different cells generate different cyclic permutations for the landmark order detector. To prove this, the following lemma is proposed.

Lemma 6. *Let* K_L *be the set of swap cells of the swap cell decomposition induced by the landmark set* L*, and let* $C_u, C_v \in K_L$*. If* $C_u \neq C_v$ *and they are bounded by the same swap line* $\vec{p_i p_j}$*, then they generate different readings in the landmark order detector.*

Proof. Consider a motion of the robot from C_u to C_v in a straight line arbitrarily close to $\vec{p_i p_j}$. This makes labels i and j appear consecutive in $\text{lcd}_s(\mathbf{x})$ for the duration of the motion. Since C_u and C_v are different, there are two cases: (1) either C_u and C_v are neighboring cells whose closure intersects at $\vec{p_i p_j}$; or (2) at least another swap line intersects $\vec{p_i p_j}$ between cells C_u and C_v. In the first case, going from C_u to C_v crosses $\vec{p_i p_j}$, and $\text{lcd}(C_u)$ is the same as $\text{lcd}(C_v)$, but with the pair of labels i and j transposed. For the second case, assume that the intersecting swap line is $\vec{p_k p_l}$. Crossing this line swaps the order of k and l. This transposition could be reverted only if $\vec{p_k p_l}$ is crossed, or if one of k or l transposes with all of the other landmarks labels. The first situation is not possible, since both swap lines lie on the same line, and $\vec{p_i p_j}$ can only intersect one of them. Furthermore, the other case would imply that i and j are at some instant not consecutive in $\text{lcd}_s(\mathbf{x})$. This is not possible by traveling arbitrarily close to $\vec{p_k p_l}$. Thus, the readings of $\text{lcd}_s(\mathbf{x})$ from C_u and C_v will differ in at least a pair of landmarks. □

The next theorem states that the landmark order detector generates different readings for cells intersecting the convex hull of the configuration of landmarks.

Theorem 7. *Let* $\hat{K}_L$ *be the set of swap cells of the swap cell decomposition induced by the landmark set* L *whose intersection with hull*(L) *is not empty. For any two different cells* C_u*,* $C_v \in \hat{K}_L$*, the cyclic permutations generated by* $lcd_s(\mathbf{x})$ *when the robot is inside* C_u *or* C_v *are different.*

Proof. The proof is by induction on the number of landmarks $n = |L|$. When $n = 3$, there is a single cell intersecting $\text{hull}(L)$. For $n > 3$, assume that the statement is true for n landmarks. Then, for $n+1$, adding the new landmark generates $2n$ swap lines, some of which stab cells in C. Cells stabbed by the same swap line will have different cyclic permutations, by Lemma 6. Since the new landmark does not change the relative ordering

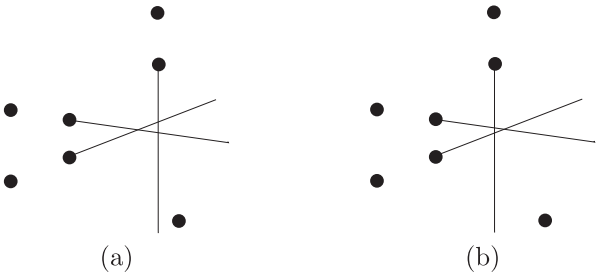

Fig. 6. Different swap cell decompositions with the same order type.

of any of the other landmarks, by the induction assumption, cells that do not share one of the new swap lines will also have different permutations. □

Note that in Theorem 7 the conditions refer only to $\hat{K}_L$, the set of cells that intersect hull(L). This fact is used in the base of the induction. For three landmarks, there is only one cell that intersects hull(L), but there are three swap cells outside hull(L), all associated with the same cyclic permutation.

6.2. Is a Cyclic Permutation Realizable?

Given a goal cyclic permutation g, the first issue we need to settle is whether it is realizable. That is, is there a swap cell $C \in \hat{K}_L$ for which lcd(C) = g?. We would like a result that, given a cyclic permutation g and the order type Λ, determines whether g is realizable. For example, with Lemma 3, we can establish that g cannot be realizable if the subsequence of landmarks in the convex hull boundary of L appears in the *wrong order* in g. However, in general it is not possible to determine whether g is realizable solely from the order type.

Lemma 8. *Two different sets of landmarks may have the same order type, but induce different swap cell decompositions.*

Proof. See Figure 6. □

Given Lemma 8, we would like to find necessary conditions for cyclic permutations to be realizable. The main motivation here is to determine whether the permutation is not realizable by moving the robot as little as possible. To achieve this, we need to somehow relate the ordering of a cyclic permutation with the order type. This is done by finding *polar pairs*. A pair (i,j) is called a polar pair of a cyclic permutation g if i and j appear consecutively in g, and g can be partitioned into subsequences $g = [i,j,X,Y]$, such that either $\Lambda(i,j) = X$ or $\Lambda(j,i) = X$. The line supported by the landmarks with labels i and j is called a *polar line*.

Take two landmarks p_i and p_j. Suppose that (i,j) is a polar pair of some cyclic permutation g. We can think of the polar line supported by p_i and p_j as composed of three line segments: the swap line $\vec{p_ip_j}$, the line segment with endpoints at p_i and p_j, and the swap line $\vec{p_jp_i}$. We can easily find a relation between g and swap lines supported by polar lines.

Lemma 9. *If a cyclic permutation g is realizable in the landmark set L at cell $C \in K_L$, then the swap lines in the boundary of C are supported by polar lines.*

Proof. With lcd(C) = g, assume $\vec{p_ip_j}$ is a swap line in the boundary of C. Clearly, the labels i and j appear consecutively in g. Now, separate the rest of the labels of g into two sets, according to on which side of $\vec{p_ip_j}$ they appear. These two sets correspond to $\Lambda(i,j)$ and $\Lambda(j,i)$. □

The necessary condition on polar lines of Lemma 9 becomes stronger with the following lemma.

Lemma 10. *If a cyclic permutation g is realizable in the landmark set L at cell $C \in K_L$, with $|L| > 2$, then g has at least two polar pairs, and every polar line intersects at least another polar line with this intersection occurring in the swap line sections of both polar lines.*

Proof. Observe that, for $|L| > 2$, every swap cell is bounded by at least two swap lines, and that every landmark is an endpoint of $|L| - 1 \geq 2$ swap lines. Therefore, a swap line cannot appear isolated in the boundary of a swap cell, as it has to intersect another swap line. Lemma 9 tells us that the swap lines in the boundary of a swap cell are supported by polar lines, and the result follows. □

Note that Lemma 10 considers $|L| > 2$. For $L \leq 2$, determining whether g is realizable is trivial, since there is only one swap cell. One issue remains for Lemma 10. We need to determine whether two polar lines intersect and, if they do, whether the intersection is at the swap line sections. The next two lemmas are useful.

Lemma 11. *For four different landmarks $p_i, p_j, p_k, p_l \in L$, the line segment $\overline{p_ip_j}$ intersects one of the swap lines $\vec{p_kp_l}$ or $\vec{p_lp_k}$ if (1) $|\Lambda(k,l) \cap \{i,j\}| = 1$, and (2) $k \in \Lambda(i,j) \iff l \in \Lambda(i,j)$.*

Proof. Condition (1) states that p_i and p_j appear on different sides of the line supporting the swap lines $\vec{p_kp_l}$ and $\vec{p_lp_k}$. Condition (2) states that both p_k and p_l appear on the same side of the line supported by p_i and p_k. With these two conditions, the result easily follows. □

Lemma 12. *For four different landmarks p_i, p_j, p_k, $p_l \in L$, the line segments $\overline{p_ip_j}$ and $\overline{p_kp_l}$ intersect if $|\Lambda(i,j) \cap \{k,l\}| = 1$ and $|\Lambda(k,l) \cap \{i,j\}| = 1$.*

Proof. When line segment $\overline{p_kp_l}$ intersects $\overline{p_ip_j}$, endpoints p_k and p_l appear on different sides of the line

Fig. 7. By order type alone, it cannot be determined whether two swap lines intersect.

supporting line segment $\overline{p_i p_j}$. This implies that exactly one of k or l is in $\Lambda(i,j) \cap \{k,l\}$, and similarly for i or j and $\Lambda(k,l) \cap \{i,j\}$. □

Lemmas 11 and 12 fall short of the desired result. They give conditions for when two polar lines *do not intersect* at their swap line sections. In particular, they cannot determine whether two polar lines are parallel. As we illustrate in Figure 7, polar lines may be parallel or not, independently of the particular order type. In Section 8 we propose a general position assumption that makes Lemmas 11 and 12 sufficient to determine whether two polar lines intersect at their swap line sections.

Now we are ready to present a navigation strategy.

Strategy 6. *Navigation to a cyclic permutation of landmarks.*

Description. Assume that the robot has gathered the order type information for the set of landmarks L, and now it is commanded to navigate to a swap cell in which a cyclic permutation of landmark labels g appears on the sensor. The first step is to compute the polar pairs of g, and determine which polar lines may intersect at their swap line sections (Lemmas 11 and 12). Now the robot needs to visit each of the intersections, until g appears on its sensor. If there are no intersections on which g appears on the sensor, then, according to Lemma 10, g is not realizable.

We need to determine how the robot may move to an intersection point of two polar lines. Assume (i,j) and (k,l) are two different polar pairs which may intersect, and assume, for the time being, that neither p_k or p_l belong to $\partial\text{hull}(L)$. The main insight here is to note that every swap line that belongs to a boundary of a swap cell that intersects $\text{hull}(L)$ also intersects $\partial\text{hull}(L)$. From Strategy 5, we know which landmarks belong to $\partial\text{hull}(L)$, together with their counterclockwise cyclic order. Using Lemma 12, we can find the line segments of $\partial\text{hull}(L)$ that intersect $\vec{p_i p_j}$ and $\vec{p_j p_i}$. Such segments are easily transversed by tracking the respective landmarks in $\partial\text{hull}(L)$ until i and j swap places in the sensor. At this point, the robot tracks p_i (or p_j, it does not matter), until k and l swap or the robot reaches p_i. This guarantees that one of the swap lines of the polar pair (i,j) is transversed. If the robot reaches p_i, the other swap line of (i,j) should be transversed in a similar manner, until k and l swap, or the tracked landmark is reached.

If one of p_k or p_l belong to $\partial\text{hull}(L)$, then the unique swap line that intersects $\text{hull}(L)$ may be transversed following the previous procedure. If both p_k and p_l belong to $\partial\text{hull}(L)$, then a valid intersection for Lemma 10 may occur only at p_k or p_l, which the robot may easily track.

---

7. The Swap Graph

In the previous section we introduced the swap cell decomposition, for a set of landmarks L. In this section we introduce the *swap graph* $G_L = (V_L, E_L)$, which is simply the dual graph of the swap cell decomposition. A vertex in V_L represents a swap cell, and an edge $(C_u, C_v) \in E_L$ indicates that swap cells C_u and C_v are neighbors, separated by exactly one swap line (see Figure 8).

As we saw in the previous section, the order type sometimes provides incomplete information about K_L. In this section, we study the construction of the swap graph for a set of landmarks, focusing on the cells of K_L that intersect $\text{hull}(L)$. This construction is more complex than that for the order type, but it provides all the combinatorial information about K_L. Furthermore, we do not need to develop new tools to construct the swap graph, as we can reuse the techniques developed for Lemmas 7 and 11, and Strategies 5 and 6.

Strategy 7. *Swap Graph construction inside hull*(L).

Description: From Strategy 5 we learned $\partial\text{hull}(L)$, and from Strategy 6, we know how to transverse the portion of all the swap lines that intersect $\text{hull}(L)$. To construct the swap graph G_L for a set of landmarks L, the robot simply tracks each of the swap lines. Lemma 7 ensures that there is a bijection between sensor readings and swap cells, so that vertices and edges of the graph are created in the natural manner: every unique sensor reading corresponds to a vertex, and there is an edge between two vertices if and only if there is a swap line separating the two readings.

We can view the swap cell decomposition as a line arrangement with some of the "middle segments" removed. Therefore, all of the complexity results of for arrangements of m half lines hold for the swap cell decomposition, such as number of swap cells, $O(m^2)$, the number of edges, $O(m^2)$, or the complexity of the boundary of a swap cell, $O(m)$ (Boissonnat et al. 1998).

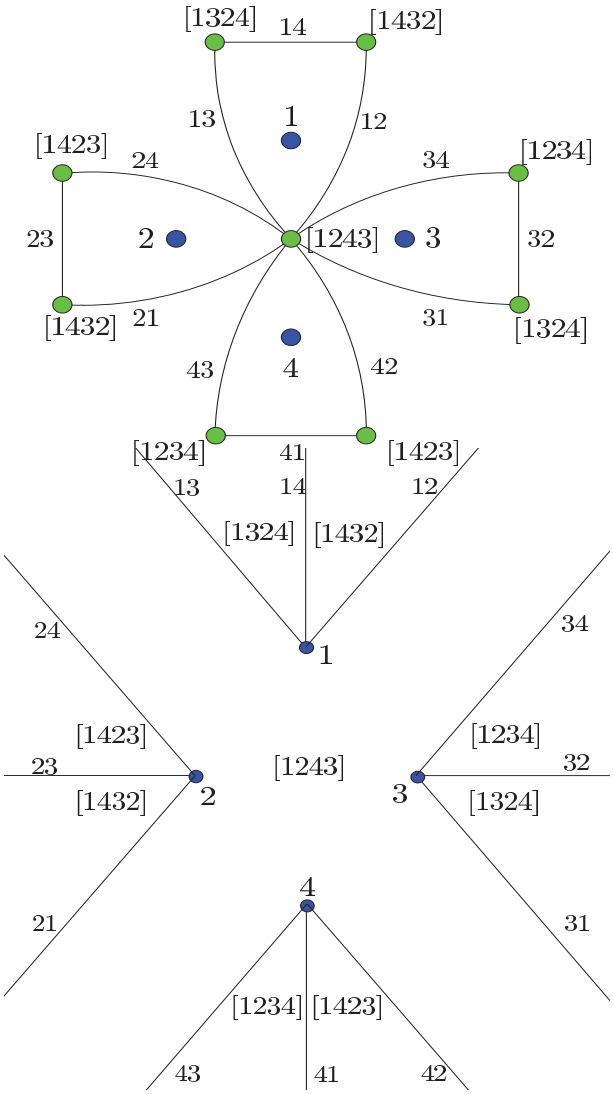

Fig. 8. A set of landmarks and its swap graph.

8. Extensions

8.1. Guaranteed Intersection of Swap Lines

In Section 6, we presented Lemmas 11 and 12 as a mean to determine whether two polar lines did intersect at their swap lines sections. As we explained, these lemmas fall short of this goal, since we could not determine whether two polar lines were parallel. One alternative is to ban parallel lines from existence. This is certainly not very intellectually satisfying, but can be easily done with a standard general position assumption, in which no two lines supported by four different landmarks are parallel. This general position assumption is justified by the fact that, given a finite set of lines in the plane, choosing one point at random in the plane to be collinear with one of the lines has probability zero.

8.2. Outside the Convex Hull

Until now, we assumed that the robot could only move tracking landmarks. This restricts the robot to movement inside the convex hull of the set of landmarks. We can extend the robot model with a second motion primitive, called *repel*. Unlike track, a repel command is only applicable when the robot is arbitrarily close to a swap line. The motion primitive repel(i,j) moves away from the landmarks $p_i, p_j \in L$, along the swap line $\vec{p_i p_j}$.

The first complication we encounter with repel is the termination of the motion primitive. If the set of landmarks satisfies the general position assumption of Section 8.1, then from Λ we can determine which of the swap lines intersect with a particular swap line $\vec{p_i p_j}$. Thus, repel(i,j) terminates once a particular swap line intersection is found. A second alternative is to modify the landmarks model more aggressively. We could assume that all the landmarks are contained inside a convex, compact, path-connected subset of the plane, and provide the robot with a *contact sensor* that indicates whether the robot is in contact with the boundary of the environment.

In Theorem 7 we proved that two swap cells inside the convex hull of L generate different cyclic permutations. Unfortunately, this is not true in general for all swap cells in the decomposition.

Theorem 13. *If C_u and C_v are two different swap cells that do not intersect hull(L), then lcd(C_u) and lcd(C_v) are not guaranteed to be different cyclic permutations, independently of the size of L.*

Proof. Refer to the construction on Figure 9. □

However, the following theorem extends the uniqueness of sensor readings inside the hull(L).

Theorem 14. *If C_u is a cell that intersects hull(L), then lcd(C_u) = lcd(C_v) implies $C_u = C_v$.*

Proof. Assume that C_u intersects hull(L), and that lcd(C_u) = lcd(C_v), but $C_u \neq C_v$. If the cell C_v does not intersect hull(L), we have a contradiction by Lemma 3 (the cyclic order of the landmarks in ∂hull(L) is different, and lcd(C_u) $\neq$ lcd(C_v)). Otherwise, if C_v intersects hull(L), by Theorem 7 we also have a contradiction, since cells intersecting hull(L) have unique readings. □

Strategy 8. *Swap graph construction.*

Description: Using Strategy 7, the robot constructs the swap graph for hull(L). To learn the swap graph outside hull(L), the robot performs a repel motion primitive on the portion outside the convex hull of every

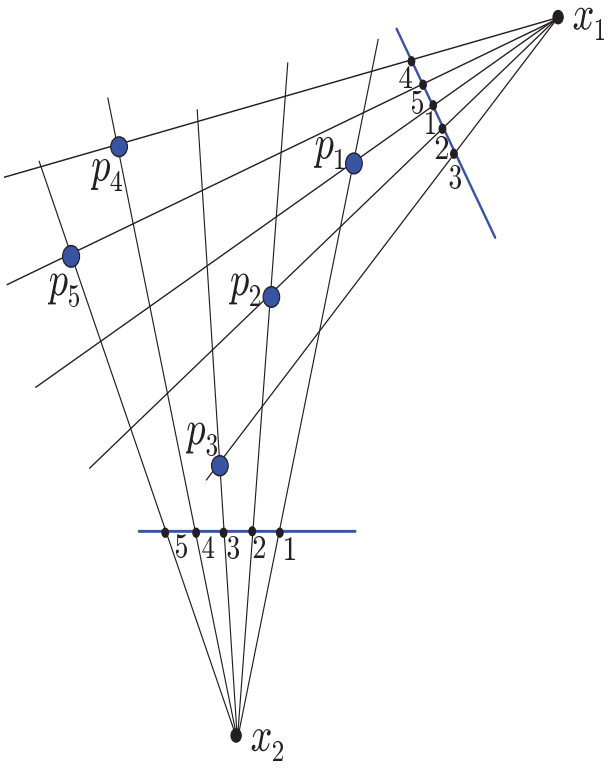

Fig. 9. Construction of a set of landmarks, such that two swap cells are associated with the same cyclic permutation. First, two points x_1 and x_2 outside hull(L) are chosen. Second, the cyclic permutation $[1, 2, 3, 4, 5, \ldots, n]$ is represented on two segments in the form of two equivalent permutations. Third, each landmark p_i is found as the intersection of two correspondent rays emanating from x_1 and x_2 and passing through the points labeled i. Suitable label arrangements on the two segments allow the retrieval a deployment for which x_1 and x_2 belong to different swap cells.

swap line. Every time the cyclic permutation changes, a vertex C is added to the swap graph, together with the corresponding edges found through the swap lines' crossings. In addition, the robot records the swap lines known to bound the associated swap cell for every vertex. Following Lemma 6, two vertices in the graph are merged into one vertex if they are associated with the same cyclic permutation, and share at least one swap line.

In the previous section we described a goal-based navigation algorithm without assuming *a priori* knowledge of the environment. Now, given a swap graph representation of the environment, we can easily drive the robot from one cell to another. A swap cell is identified by a vertex of G and its incident edges (two distinct vertices can share the same cyclic permutation but cannot have the same incident edges). Given a vertex $C \in G$, the corresponding swap cell can be reached with a repel motion along one of the swap lines labeling an edge incident in C.

9. Conclusions and Open Questions

In this paper we established the capabilities of a robot which is only able to detect the cyclic angular order of landmarks (distinguishable points in the plane) around it. The combinatorial properties of the set of landmarks were studied and established in terms of its order type. We computed the convex hull of the set of landmarks, and solved the tasks of patrolling and navigation uniquely in terms of cyclic permutations of landmarks. We did not use any coordinates to express these tasks, which made it unnecessary to model errors in the positions of the robot and the landmarks, and which made any precise measurements of angles and distances traveled unnecessary.

Given the information provided by the permutations, one may wonder if it is possible to recover the coordinates of an equivalent set of landmarks. That is, is it possible to construct the coordinates of a set of landmarks such that this construction has the same order type as the original set? This turns out to be a very hard question. Simply deciding if a sequence of permutations can be realized in the plane is NP-hard (Shor 1991). Moreover, representing such coordinates may require an exponential number of bits (Goodman et al. 1989). Our problem may be simpler, since the robot *proves* that the permutations are realizable by sensing them. If not for the general case, realizations can be easily found for small subsets of landmarks.

This work has made two strong sensing assumptions: infinite range in the landmark order detector, and a perfect identification of landmarks. To remove the first assumption we note that the concepts presented still hold for local neighborhoods of landmarks. One of our preliminary ideas is to apply directly the algorithms presented in Ghrist et al. (2006), in the context of sensor networks. Determining the relations between neighborhoods of landmarks also allows the introduction of environment obstacles. For the second assumption, given that the functions Λ and λ provide equivalent information, it is plausible to allow some recognition error of landmarks. This idea is as follows. If the landmark order detector is not able to identify a landmark, but it is able to detect that a landmark is indeed present, this recognition error may be corrected using the λ function. For example, the robot may be able to detect landmarks much farther than the maximum distance for a perfect identification. The λ function seems to be the appropriate tool for such situations.

Finally, we mention that there is a natural description for the information space of n landmarks with the braid group B_n on n strands. Each strand represents a unique landmark, and a crossing between two strands

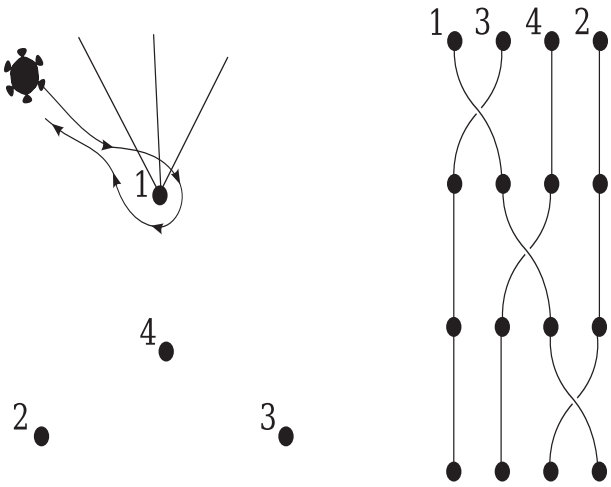

Fig. 10. Encoding the sensor history with braids. There is a natural description of the information space of n landmarks with the braid group on n strands, B_n. Each strand represents a landmark, and each crossing represents a change in the circular order. In the figure, the robot follows a path that surrounds landmark 1. The changes in the circular order are encoded with the braids on the right.

represents a swap in the cyclic order of the landmarks. The strand corresponding to the landmark closer to the robot is defined to *cross over* the other strand. See Figure 10 for an example. Given that we are dealing with circular permutations, this suggests the robot paths that are loops. We are hopeful that this description will raise other interesting questions.

Funding

This work was funded by NSF grant 0904501 (IIS robotics), DARPA SToMP grant HR0011-05-1-0008, and MURI/ONR grant N00014-09-1-1052.

Notes

1. This is done for the sake of clarity, since track($s(p)$) may be designed to stop *just before* p is reached. However, this would not change the essence of further developments, and may clutter some descriptions.
2. Pappus's hexagon theorem specifies the structure of nine lines and nine points, with each line incident to three points, and each point incident to three lines.

References

Betke, M. and Gurvits, K. (1994). Mobile robot localization using landmarks. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 135–142.

Björner, A., Vergnas, M. L., Sturmfels, B., White, N., and Ziegler, G. M. (1993). *Oriented Matroids*. Cambridge, Cambridge University Press.

Boissonnat, J., Yvinec, M., and Brönniman, H. (1998). *Algorithmic Geometry*. Cambridge, Cambridge University Press.

Borenstein, J., Everett, B., and Feng, L. (1996). *Navigating Mobile Robots: Systems and Techniques*. Wellesley, MA, A.K. Peters.

Brost, R. (1991). Analysis and planning of planar manipulation tasks. *Ph.D. Thesis*, Carnegie Mellon University, Pittsburgh, PA.

Bulata, H. and Devy, M. (1996). Incremental construction of a landmark-based and topological model of indoor environments by a mobile robot. *Proceedings of the IEEE International Conference on Robotics & Automation*, pp. 1054–1060.

de Berg, M., van Kreveld, M., Overmars, M., and Schwarzkopf, O. (1997). *Computational Geometry: Algorithms and Applications*. Berlin, Springer.

Dean, T., Basye, K., and Kaelbling, L. (1993). Uncertainty in graph-based map learning. *Robot Learning*, Connell, J. and Mahadevan, S. (eds). Dordrecht, Kluwer Academic, pp. 171–192.

Dudek, G., Freedman, P., and Hadjres, S. (1993). Using local information in a non-local way for mapping graph-like worlds. *Proceedings of the International Joint Conference on Artifical Intelligence*, pp. 1639–1645.

Edelsbrunner, H. (1987). *Algorithms in Combinatorial Geometry*. Berlin, Springer.

Erdmann, M. (1986). Using backprojections for fine motion planning with uncertainty. *International Journal of Robotics Research*, 5(1): 19–45.

Erdmann, M. and Mason, M. (1988). An exploration of sensorless manipulation. *IEEE Transactions on Robotics & Automation*, 4(4): 369–379.

Freda, L., Tovar, B., and LaValle, S. M. (2007). Learning combinatorial information from alignments of landmarks. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Ghrist, R., Lipsky, D., Poduri, S., and Sukhatme, G. (2006). Surrounding nodes in coordinate-free networks. *Workshop in Algorithmic Foundations of Robotics*.

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

Goodman, J. and Pollack, R. (1983). Multidimensional sorting. *SIAM Journal on Computing*, 12(3): 484–507.

Goodman, J., Pollack, R., and Sturmfels, B. (1989). Coordinate representation of order types requires exponential storage. *ACM Symposium on Theory Computing*, pp. 405–410.

Graham, R. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, 7: 175–180.

Hayet, J. B., Esteves, C., Devy, M., and Lerasle, F. (2002). Qualitative modeling of indoor environments from visual landmarks and range data. *Proceedings of the IEEE International Conference on Robotics & Automation*, pp. 631–636.

Hu, H. and Gu, D. (2000). Landmark-based navigation of industrial mobile robots. *Industrial Robot*, 27: 458–467.

Knuth, D. E. (1992). *Axioms and Hulls*. Berlin, Springer.

Koenderink, J. and van Doorn, A. (1976). The singularities of the visual mapping. *Biological Cybernetics*, 24: 51–59.

Kuipers, B. (2000). The spatial semantic hierarchy. *Artificial Intelligence*, 119: 191–233.

Kuipers, B., Froom, R., Lee, W., and Pierce, D. (1993). The semantic hierarchy in robot learning. *Robot Learning*, Connell, J. and Mahadevan, S. (eds). Dordrecht, Kluwer Academic, pp. 141–170.

Lazanas, A. and Latombe, J. C. (1992). Landmark-based robot navigation. In *AAAI National Conference on Artificial Intelligence*, pp. 816–822.

Levitt, T. and Lawton, D. (1990). Qualitative navigation for mobile robots. *Artificial Intelligence*, 44(3): 305–360.

Lowe, D. (2003). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 20: 91–110.

Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. (2002). FastSLAM: A factored solution to the simultaneous localization and mapping problem. *AAAI National Conference On Artificial Intelligence*.

Parr, R. and Eliazar, A. (2003). DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. *Proceedings of the International Joint Conference on Artificial Intelligence*.

Remolina, E. and Kuipers, B. (2004). Towards a general theory of topological maps. *Artificial Intelligence*, 152: 47–104.

Shatkay, H. and Kaelbling, L. (1997). Learning topological maps with weak local odometric information. *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 920–929.

Shimshoni, I. (2002). On mobile robot localization from landmark bearings. *IEEE Transactions on Robotics and Automation*, 18(6): 971–976.

Shor, P. (1991). Stretchability of pseudolines is NP-hard. *Applied Geometry and Discrete Mathematics*, 531–554.

Simmons, R. and Koenig, S. (1995). Probabilistic robot navigation in partially observable environments. *Proceedings of the International Joint Conference on Artificial Intelligence*

Smith, R. and Cheeseman, P. (1986). On the representation of and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5: 56–68.

Steck, S. and Mallot, H. (2000). The role of global and local landmarks in virtual environment navigation. *Presence: Teleoperators and Virtual Environments*, 9(1): 69–83.

Tashiro, K., Ota, J., and Arai, T. (1995). Design of the optimal arrangement of artificial landmarks. *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 407–413.

Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics*. Cambridge, MA, The MIT Press.

Thrun, S., Fox, D., and Burgard, W. (1998). A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31: 29–53.

Tovar, B., Freda, L., and LaValle, S. M. (2007). *Mapping and Navigation from Permutations of Landmarks* (*AMS Contemporary Mathematics Proceedings*, vol. 438). Providence, RI, Americal Mathematical Society, pp. 33–45.

Yamauchi, B., Schultz, A., and Adams, W. (2002). Mobile robot exploration and map-building with continuous localization. *Proceedings of the IEEE International Conference on Robotics & Automation*, pp. 3715–3720.