Learning combinatorial information from permutations of landmarks

Benjamín Tovar*, Luigi Freda†, and Steven M. LaValle‡

Abstract

This paper considers a robot that moves in the plane and is only able to sense 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 (e.g., 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 convex hull of the set of landmarks, the information space can be nicely characterized as an *order type*, which provides information powerful enough to determine which points lie inside the convex hulls of subsets of landmarks. Additionally, if the robot is allowed to move outside the convex hull of the set of landmarks, the information space is described with a *swap cell decomposition*, which is an aspect graph in which each aspect is a cyclic permutation of landmarks. We show how to construct such decomposition through its dual, using two kinds of feedback motion commands based on the landmarks sensed.

[0]Manuscript submitted as a regular paper.
The corresponding author is B. Tovar.
This work was founded by NSF grant 0904501 (IIS robotics), DARPA SToMP grant HR0011-05-1-0008, and MURI/ONR grant N00014-09-1-1052.

*B. Tovar is with the Dept. of Mechanical Engineering, Northwestern University, 2145 Sheridan Road, Evanston, IL 602081, USA. email: b-tovar@northwestern.edu

†L. Freda is with U.T.R.I. S.p.A., Via del Follatoio 12 -34147 Trieste, Italia.

‡S. M. LaValle is with the Dept. of Computer Science, University of Illinois, 201 N. Goodwin 3340, Urbana, IL, 61801. USA. email: lavalle@uiuc.edu

1 Introduction

Consider a walk in a prairie in a cloudy day. We do no have a compass available, so we do not know where north is. As we walk around, some landmarks (i.e., the occasional tree) cross our way. Without shadows from the sun, we cannot calculate any distances either. In fact, the only thing we know for certain is when a pair of landmarks align with our moving position, and then swap places in a circular ordering around us. What can we tell about our walk in the prairie? What can we tell if we move on the plane with very limited sensing?

We study the above scenario from a robotics perspective. What can a robot learn about its environment from this limited sensor? In this paper we consider 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 *map*, or representation of the environment, is a sequence of landmark cyclic permutations. After establishing what the robot can learn from its simple sensor, we then illustrate the kinds of tasks that it can solve, including a surveillance/patrolling behavior around the perimeter (convex hull) of landmarks.

In robotics, landmarks have classically have been used for navigation [1, 4, 16, 17, 35, 30]. For example, in works such as [22, 36], robot paths that minimized the localization error were found using preimages [10] for a known arrangement of landmarks . In more recent years, landmarks have been used to construct geometric models of the environment, together with an explicit estimation of the robot position. In the most well-know form of simultaneous localization and mapping (SLAM) [27, 26, 32, 38, 40], the addition of some Gaussian assumptions allows the estimation of the position of the robot and the landmarks through the use of probabilistic filters. These approaches have an impressive implementation success, but the probabilistic information spaces they generate are hard to characterize given that they are infinite-dimensional.

In works such as [5, 7, 8, 20, 21, 28, 29, 34], the aim is to represent the environment through a graph-like spatial description. There, a vertex in a graph represents a *place*, and edges represent possible paths between places. In this context, places are commonly defined as environment

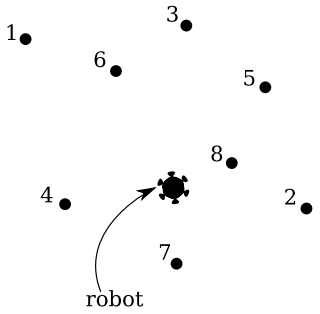

Figure 1: The landmark order detector gives the cyclic order of the landmarks around the robot. Note that only the cyclic order is preserved, and that the sensed angular position of each landmark may be quite different from the real one. Thus, the robot only knows reliably, up to a cyclic permutation,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

locations from which a certain arrangement of landmarks is visible.

In our case, we will not try to reconstruct a complete model of the environment, but rather characterize it with the *order type* of a configuration of points in the plane [13]. This implies that two locations in the plane become indistinguishable to the sensor if they yield the same cyclic permutation of landmarks. The models and assumptions used in this paper were inspired from basic sensing issues in robotics. There are also close connections 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. 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 [23] is similar to the ideas presented here. In our case, the cyclic permutation sensor amounts to detecting when two landmarks cross in the field of the the robot, whereas in [23], the sensor detects when the robot becomes collinear with a pair of landmarks. Therefore, the sensor considered in [23] is more powerful.

The ideas in this paper are based on the material presented in [39, 11].

2 Basic Definitions

We model a robot as a moving point in $\mathbb{R}^2$. Its configuration $q \in SE(2)$ is described by its position in $\mathbb{R}^2$ and heading 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 label (i.e., $s(p_i) = i$).

Let $\mathcal{L}$ be the set of all possible finite subsets of $\mathbb{R}^2$. That is, $\mathcal{L}$ 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 sensor is called a *landmark cyclic order detector*, and it is denoted with $\text{lcd}_s(\mathbf{x})$, 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, and 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, $\text{lcd}_s(\mathbf{x})$ 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, that is 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 (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 cyclic permutation sensed 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 half-line as the *swap-line* $p_i\vec{p}_j$ (think of this notation as a ray in the direction of p_i to p_j, starting at p_j). The swap-line $p_j\vec{p}_i$ is defined in a similar manner. Note that when the robot is arbitrarily close to $p_i\vec{p}_j$ or $p_j\vec{p}_i$, p_i and p_j appear consecutive in $\text{lcd}_s(\mathbf{x})$, 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 is denoted by $\text{track}(s(p))$. For simplicity, we assume that $\text{track}(s(p))$ ends when the robot arrives at p, which means that $\text{lcd}_s(\mathbf{x})$ no longer

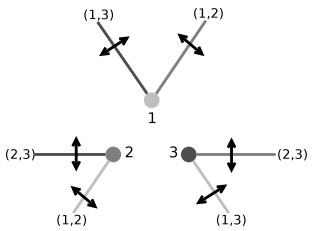

Figure 2: Swap lines. Crossing a half-line swaps the order of the respective landmarks in the reading of the landmark order detector. Such half-lines are called *swap lines*.

detects the landmark just tracked [1]. 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 [24, 26, 37, 38]).

3 Order Types and Landmarks

Given the sensing, and control models introduced, consider the robot as it moves in the environment. The only information the robot receives is the changes in the cyclic permutations. For example, 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 with 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 *right* of the directed segment that connects 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 the 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

[1] We might as well define track($s(p)$) to stop *just before* p is reached, but the essence of further developments does not change, and it clutters some descriptions. Moreover, it already models some robotic systems, as a robotic agent flying over a terrain.

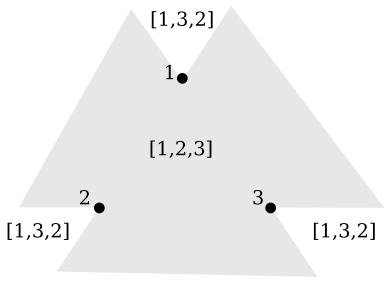

Figure 3: Cyclic permutations of three landmarks. Purely by sensing, the robot cannot even know if it is inside the convex hull defined by the three landmarks. Nevertheless, the orientation of the triangle (the counterclockwise cyclic order of the landmarks as sensed from inside their convex hull) can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

follows [13]. 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 [13]. 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 arrangements 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 in that any two of them intersect in at most one point. These lines, that are no necessarily straight, are called pseudo-lines, and such an arrangement is called an arrangement of pseudo-lines. We refer to properties such as these ones as *combinatorial properties* [2]. For the order type, the value of Λ is independent to homogeneous transformation we apply to the set of points. Additionally, 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 a combinatorial abstractions of point configurations over the reals, of real hyperplane arrangements, of convex polytopes, and of directed graphs [2]. There are several equivalent ways of defining an oriented matroid. In our case, given that we are dealing with points in the plane, the one most useful is in terms of *chirotopes*. A chirotope (of rank 3) is a mapping $\chi : L \times L \times L \leftarrow \{-1, 0, +1\}$ which satisfies:

1. χ is not identically zero.
2. χ is alternating, meaning that for all p_1, p_2, and $p_3 \in L$, $\chi(p_1, p_2, p_3) = \chi(p_3, p_1, p_2) = \chi(p_2, p_3, p_1)$, and $\chi(p_1, p_3, p_2) = \chi(p_2, p_1, p_3) = \chi(p_3, p_2, p_1)$.
3. For all p_1, p_2, p_3, p'_1, p'_2, and $p'_3 \in L$:
   (a) $\chi(p'_1, p_2, p_3) \cdot \chi(p_1, p'_2, p'_3) \geq 0 \implies \chi(p_1, p_2, p_3) \cdot \chi(p'_1, p'_2, p'_3) \geq 0$
   (b) $\chi(p'_2, p_2, p_3) \cdot \chi(p'_1, p_2, p'_3) \geq 0 \implies \chi(p_1, p_2, p_3) \cdot \chi(p'_1, p'_2, p'_3) \geq 0$
   (c) $\chi(p'_3, p_2, p_3) \cdot \chi(p'_1, p'_2, p_3) \geq 0 \implies \chi(p_1, p_2, p_3) \cdot \chi(p'_1, p'_2, p'_3) \geq 0$

It is straightforward to prove that a valid choice for χ is the sign of the determinant in equation 1. Therefore, χ and Λ encode the same information. One nice consequence is that now we can describe the combinatorial structure of the set of landmarks with the following axioms [18]:

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 χ (and therefore Λ) satisfies the previous restrictions, there are ways in which to assign signs to the triplets such that they do not correspond to points in the Euclidean plane. In particular, they may fail to satisfy the Pappus's hexagon theorem [2, 18][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 [13] 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, in the segment between two landmarks, or aligned with two landmarks but not on the segment joining them (i.e., when one landmark occludes another). An information state is defined as the pair (Q', Λ'), in which Q' refers

[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.

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 definition is extended naturally to our landmark definitions, 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 6. *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 $\overline{p_i p_j}$, 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 $\overline{p_i p_j}$. Consider any point in the interior of $\overline{p_i p_j}$, 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 $\overline{p_i p_j}$. 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 $\text{lcd}_s(\mathbf{x})$(the robot is at p_i). Next, the robot is commanded to

track p_j, and at the moment i is detected again, the robot is guaranteed to be on $\overline{p_i p_j}$, pointing towards p_j. Applying Lemma 6 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 in 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 6, 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.

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 in this strategy an orientation is found more than once, it is taken as the correct orientation of the triangle. This strategy allows for a control error in the direction of the robot up to $2\pi/3$.

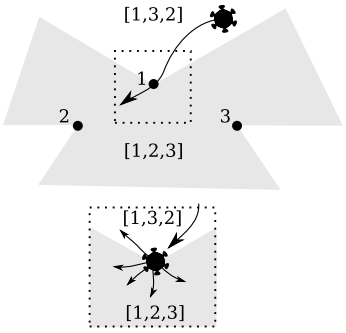

Figure 4: Triangle orientation error. A small control error may find the wrong orientation for the triangle. On the bottom, if the robot follows the top-left arrow, the orientation is not computed correctly.

4.2 Determine which landmark is closer to the robot

Consider two different landmarks p_i, $p_j \in L$, and suppose $|L| > 2$, such that the robot is one of the swap lines $p_i\vec{p}_j$ or $p_j\vec{p}_i$. From a single reading of $\text{lcd}_s(\mathbf{x})$, we cannot determine whether p_i or p_j is closer to the robot. However, combining sensing actuation this is easily determined as follows:

Strategy 3. *Determine which landmark of a swap line is closer to the robot.*

Description: By assumption, there is at least one landmark $p_k \in L$, such that $k \in \Lambda(i, j)$ or $k \in \Lambda(j, i)$. Here we assume that $k \in \Lambda(i, j)$, as the other case follows symmetrically. The robot tracks p_k until p_i and p_j are not collinear anymore with the robot. There are two possibilities for the reading of $\text{lcd}_s(\mathbf{x})$. If p_i is closer to the robot, then $\text{lcd}_s(\mathbf{x}) = [i, j, X, k, Y]$, in which X and Y are (perhaps empty) subsequences of landmark labels. Otherwise, if p_j is closer, then the reading looks like $\text{lcd}_s(\mathbf{x}) = [j, i, X, k, Y]$.

4.3 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 the landmarks that are on the boundary $\partial\text{hull}(L)$ of $\text{hull}(L)$. This task can be easily solved, if not 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 *three coin algorithm* for the computation of the convex hull [15, 33]. In its regular setting, the three coin algorithm starts by finding one landmark in the convex hull (the leftmost for example), 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 if they lie to the left or 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 4. *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 will make the label of the landmark the same as the step it was selected to be tracked.

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

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 7. *Strategy 4 finds a pair of landmarks in hull(L) with $O(n)$ tracking motion primitives.*

Proof. Consider the swap line $p_{i+1}\vec{p}_{i+2}$, 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 4, $\partial\text{hull}(L)$ is easily computed:

Strategy 5. *Find ∂hull(L) of the set of landmarks L*

Description. Perform Strategy 4, 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 4. 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 $W \subset M$, ∂hull$(M) \cap W = \emptyset$ and the size of M is minimal.*

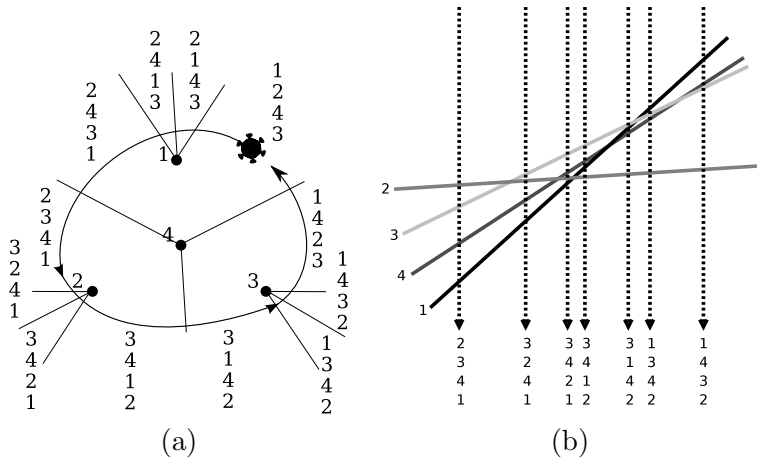

Figure 5: Retrieving the permutations that encode the configuration of landmarks. In (a) the robot travels outside the convex hull of a set of landmarks. This is naturally expressed in the dual line arrangement on (b).

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-know properties of such dual arrangements[6, 9], 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 [9]. 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 it is shown in Figure 5, when the robot travels 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 $\binom{}{n2+1}$ permutations are needed to describe the line arrangement, when actually $\binom{2}{n2}$ could be read by the robot traveling outside the convex hull. These permutations have other nice symmetric properties, and the reader is referred to [9].

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 8. *Let L be a set of landmarks, let Z be a subsequence of $lcd_s(\mathbf{x})$ and containing only the labels corresponding to the landmarks in $\partial hull(L)$ (elements of Z may not necessarily appear consecutively in $lcd_s(\mathbf{x})$). 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 8, 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 sequentially each of the landmarks in the boundary, following the order found. When the robot arrives at a landmark, the corresponding permutations are generated in the natural manner from the reading of the landmark order detector at such location. 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 (expressed in our framework):

Lemma 9. *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 10. *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 9. □

Strategy 6. *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. The patrolling problem can be solved as follows. Assume the robot has computed the permutations encoding the dual arrangement of L. The strategy is based in 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 10, 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, the 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 to 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 [19], 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 the landmarks in the plane.*

6.1 Swap cell decomposition

We refer to the decomposition induced on $\mathbb{R}^2$ 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 an point in C.

The swap cell decomposition can be naturally encoded in a graph, which is an aspect graph [19] 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 around once 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 11. *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 $p_i\vec{p}_j$, 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 $p_i\vec{p}_j$. This makes labels i and j to 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: Either C_u and C_v are neighboring cells whose closure intersects at $p_i\vec{p}_j$, or at least another swap line intersects $p_i\vec{p}_j$ between cells C_u and C_v. In the first case, going from C_u to C_v crosses $p_i\vec{p}_j$, 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 $p_k\vec{p}_l$. Crossing this line swaps the order of k and l. This transposition could be reverted only if $p_k\vec{p}_l$ is crossed, or if one of k or l transposes with all the other landmarks labels. The first situation is not possible, since both swap lines lie on the same line, and $p_i\vec{p}_j$ 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 $p_k\vec{p}_l$. 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 12. *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. By induction on the the number of landmarks $n = |L|$. When $n = 3$, there is a single cell intersecting hull(L). For $n > 3$, assume 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 11. Since the new landmark does not change the relative ordering of any other three 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 12, 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 $\text{lcd}(C) = g$?. 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 8, 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, determining weather g is realizable studying solely the order type in general is not possible:

Lemma 13. *Two different sets of landmarks may have the same order type, but induce different swap cell decompositions.*

Proof. See Figure 6. □

Given Lemma 13, we would like to find necessary conditions for a cyclic permutations to be realizable. The idea here is to determine whether the permutation is not realizable by moving the

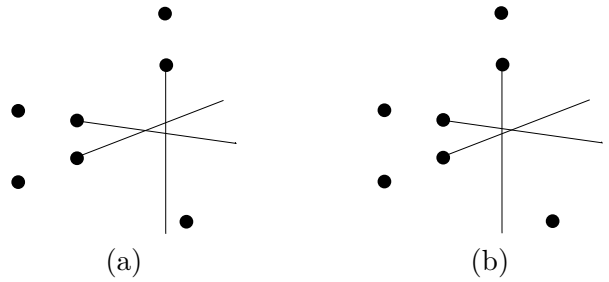

Figure 6: Different swap cell decompositions with the same order type.

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 consecutive in g, and g can be partition 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, and suppose (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 $p_i\vec{p}_j$, the line segment with endpoints at p_i and p_j, and the swap line $p_j\vec{p}_i$. We can easily find a relation between g and swap lines supported by polar lines:

Lemma 14. *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 $\text{lcd}(C) = g$, assume $p_i\vec{p}_j$ is a swap line in the boundary of C. Clearly, the labels i and j appear consecutive in g. Now, separate the rest of the labels of g into two sets, according to which side of $p_i\vec{p}_j$ they appear. These two sets correspond to $\Lambda(i, j)$ and $\Lambda(j, i)$. □

The necessary condition on polar lines of Lemma 14 becomes stronger with the following lemma:

Lemma 15. *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 14 tells

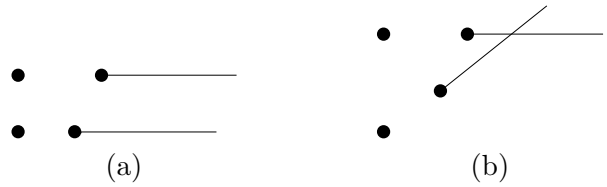

Figure 7: By order type alone, it cannot be determined whether two swap lines intersect.

us that the swap lines in the boundary of a swap cell are supported by polar lines, and the result follows. □

Note that Lemma 15 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 15. We need to determine whether two polar lines intersect, and if they do, whether the intersections is at the swap line sections. The next two lemmas are useful:

Lemma 16. *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 $p_k\vec{p}_l$ or $p_l\vec{p}_k$ 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 p_i and p_j appear on different sides of the line supporting the swap lines $p_k\vec{p}_l$ and $p_k\vec{p}_l$. 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 17. *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 supporting line segment $\overline{p_ip_j}$. 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 17 and 16 fall short to the result we wanted. 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 17 and 16 sufficient to determine whether two polar lines intersect at their swap lines sections.

Now we are ready to present a navigation strategy:

Strategy 7. *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 17 and 16). 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 15, 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 $\text{hull}(L)$, intersects $\partial\text{hull}(L)$. From Strategy 6, we know which landmarks belong to $\partial\text{hull}(L)$, together with their 1counterclockwise cyclic order. Using Lemma 17, we can find the line segments of $\partial\text{hull}(L)$ that intersect $p_i\vec{p}_j$ and $p_j\vec{p}_i$. 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 landmark tracked is reached.

If one of p_i or p_j 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_i and p_j belong to $\partial\text{hull}(L)$, then a valid intersection for Lemma 15 may occur only at p_i or p_j, which the robot may easily track.

---

7 The Swap Graph

In the previous section we introduce 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 of the swap cell

decomposition. A vertex in V_L is 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.

As we saw on 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 hull(L). This construction is more complex that 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 12 and 16, and Strategies 6 and 7.

Strategy 8. *Swap Graph construction inside hull(L)*

Description: From Strategy 6 we learned ∂hull(L), and from Strategy 7, we know how to transverse the portion of all the swap lines that intersect 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 12 ensures that there is a bijection between sensor readings and swap cells, therefore vertices and edges of the graph are created in the natural manner: Every unique sensor reading correspond to a vertex, and there is an edge between two vertices if and only if there is a swap line separating the two readings.

Some comments on complexity. We can view the swap cell decomposition as a line arrangement with some of the "middle segments" removed. Therefore, all 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, $O(m)$ [3].

8 Extensions

8.1 Guaranteed intersection of swap lines

In Section 6, we presented Lemmas 17 and 16 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 be done with a

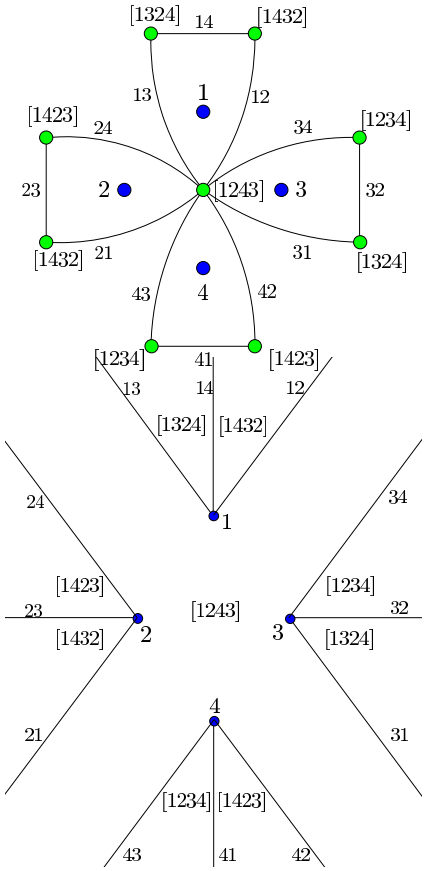

Figure 8: A set of landmarks and its swap graph.

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 allows the robot to move only 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 $p_i\vec{p}_j$.

The first complication we encounter with repel is the termination of the motion primitive. If the set of landmarks satisfy the general position assumption of Section 8.1, then from Λ we can determine all the swap lines that intersect with a particular swap line $p_i\vec{p}_j$. Thus, repel(i, j) terminates once a particular swap line intersection is found. A second alternative is to modify more aggressively the landmarks model. 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 12 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 18. *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 19. *If C_u is a cell that intersects hull(L), then lcd(C_u) = lcd(C_v) implies $C_u = C_v$.*

Proof. If C_v does not intersect hull(L), then by Lemma 8 $\text{lcd}(C_u) \neq \text{lcd}(C_v)$. Otherwise, $\text{lcd}(C_u) \neq \text{lcd}(C_v)$ by Theorem 12. □

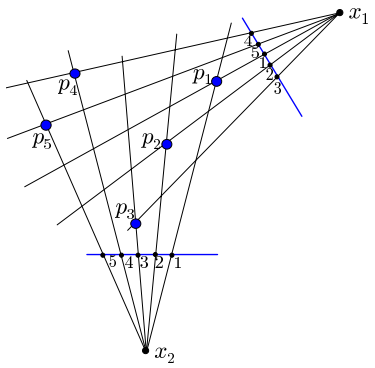

Figure 9: Construction of a set of landmarks, such that two swap cells are associated with the same cyclic permutation. First, two points x_1 and x_2 outside hull(L) are chosen. Second, the cyclic permutation $[1, 2, 3, 4, 5, \ldots, n]$ is represented on two segments in the form of two equivalent permutations.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

.

Strategy 9. *Swap Graph construction*

Description: Using Strategy 8, the robot first construct 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 swap line. Every time the cyclic permutation changes, a vertex C is added to the swap graph, together with the corresponding edges found trough the swap lines crossings. Additionally for every vertex the robot records the swap lines known to bound the associated swap cell. Following Lemma 11, two vertices in the graph are merged into one 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 an a priory 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 unnecessary to model errors in the positions of the robot and the landmarks, and which made unnecessary any precise measurements of angles and distances traveled.

Given the information the permutations provide, 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. Just deciding if a sequence of permutations can be realized in the plane is NP-hard[31]. Moreover, representing such coordinates may require exponential number of bits[14]. Nevertheless, 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 found for small subsets of landmarks.

Work remains to be done to remove some of the assumptions made. One of them is the infinite range assumption for the landmark order detector, since the concepts presented still hold for local neighborhoods of landmarks. One solution, although not very efficient in the number of sensing operations, is to apply directly the algorithms presented in [12], in the context of sensor networks. Determining the relations between neighborhoods of landmarks, also allows the introduction of environment obstacles.

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 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 suggest that the strands are more naturally described as crossing in a cylinder. This description is suitable to answer questions in the absence of control history (i.e., when the path followed by the robot is unknown). For example, was a

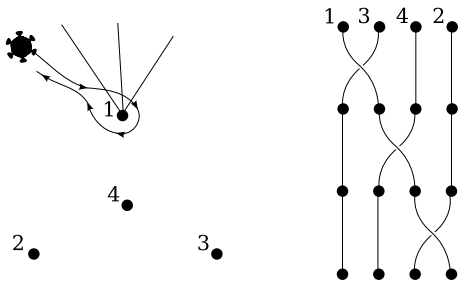

Figure 10: Encoding the sensor history with braids. There is a natural description of the information space of n landmarks with the braid group on n strands, B_n. Each strand represents a landmark, and each crossing represents a change in the circular order. In the figure, the robot follows a path that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

group of landmarks surrounded by the robot?, did the robot perform a non-trivial loop?, etc. Our immediate interest is focus on the first Dehn's fundamental problem [25], that is, to identify the words that the identify the identity. Such words naturally represent all the robot's paths that are loops. We are hopeful that this description will raise other interesting questions.

Finally, 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 indeed a landmark is 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 the appropriate tool for such situations.

References

[1] M. Betke and K. Gurvits. Mobile robot localization using landmarks. In *1994 IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 135–142, 1994.

[2] A. Björner, M. Las Vergnas, B. Sturmfels, N. White, and G. M. Ziegler. *Oriented matroids.* Cambridge University Press, 1993.

[3] J.D. Boissonnat, M. Yvinec, and H. Brönniman. *Algorithmic geometry.* Cambridge University Press, 1998.

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

[5] H. Bulata and Michel Devy. Incremental construction of a landmark-based and topological model of indoor environments by a mobile robot. In *IEEE Int. Conf. Robot. & Autom.*, pages 1054–1060, 1996.

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

[7] T. Dean, K. Basye, and L. Kaelbling. Uncertainty in graph-based map learning. In J. Connell and S. Mahadevan, editors, *Robot Learning*, pages 171–192. Kluwer Academic, Dordrecht, 1993.

[8] G. Dudek, P. Freedman, and S. Hadjres. Using local information in a non-local way for mapping graph-like worlds. In *Proc. Int. Joint Conf. on Artif. Intell.*, pages 1639–1645, 1993.

[9] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer-Verlag, Berlin, 1987.

[10] M. Erdmann. Using backprojections for fine motion planning with uncertainty. *Int. J. Robot. Res.*, 5(1):19–45, 1986.

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

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

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

[14] J.E. Goodman, R. Pollack, and B. Sturmfels. Coordinate representation of order types requires exponential storage. In *ACM Sympos. Theory Comput*, pages 405–410, 1989.

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

[16] J. B. Hayet, C. Esteves, M. Devy, and F. Lerasle. Qualitative modeling of indoor environments from visual landmarks and range data. In *IEEE Int. Conf. Robot. & Autom.*, pages 631–636, 2002.

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

[18] Donald Ervin Knuth. *Axioms and hulls*. Springer-Verlag, 1992.

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

[20] B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.

[21] B. Kuipers, R. Froom, W.Y. Lee, and D. Pierce. The semantic hierarchy in robot learning. In J. Connell and S. Mahadevan, editors, *Robot Learning*, pages 141–170. Kluwer Academic, Dordrecht, 1993.

[22] A. Lazanas and J. C. Latombe. Landmark-based robot navigation. In *AAAI National Conference on Artificial Intelligence*, pages 816–822, 1992.

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

[24] D. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 20, pages 91–110, 2003.

[25] W. Magnus, A. Karrass, and D. Solitar. *Combinatorial Group Theory*. Dover, second, revised edition, 2004.

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

[27] R. Parr and A. Eliazar. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In *Proc. Int. Joint Conf. on Artif. Intell.*, 2003.

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

[29] H. Shatkay and L.P. Kaelbling. Learning topological maps with weak local odometric information. In *International Joint Conference on Artificial Intelligence*, pages 920 – 929, 1997.

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

[31] P. Shor. Stretchability of pseudolines is NP-hard. *Applied Geometry and Discrete Mathematics*, pages 531–554, 1991.

[32] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *Proc. Int. Joint Conf. on Artif. Intell.*, 95.

[33] J. Sklansky. Measuring concavity on a rectangular mosaic. *IEEE Transactions on Computers*, 21:1355–1364, 1972.

[34] R. Smith and P. Cheeseman. On the representation of and estimation of spatial uncertainty. *Int. J. Robot. Res.*, 5:56–68, 1986.

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

[36] K. Tashiro, J. Ota, and T. Arai. Design of the optimal arrangement of artificial landmarks. In *1995 IEEE Int. Conf. on Robotics and Automation*, volume 1, pages 407–413, 1995.

[37] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.

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

[39] B. Tovar, L. Freda, and S. M. LaValle. Mapping and navigation from permutations of landmarks. In *AMS Contemporary Mathematics Proc*, volume 438, pages 33–45, 2007.

[40] B. Yamauchi, A. Schultz, and W. Adams. Mobile robot exploration and map-building with continuous localization. In *IEEE Int. Conf. Robot. & Autom.*, pages 3715–3720, 2002.