Mapping and Navigation from Permutations of Landmarks

Benjamín Tovar[†] **Luigi Freda**[‡] [†]**Steven M. LaValle**

[†]Department of Computer Science
University of Illinois
201 N. Goodwin
Urbana, IL 61801 USA.
{btovar,lavalle}@uiuc.edu

[‡]Dipartimento di Informatica e Sistemistica
Universit di Roma "La Sapienza"
via Eudossiana 18
00184 Roma, Italy.
freda@dis.uniroma1.it

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 (e.g., coordinates) is available regarding the robot or landmark positions; moreover, the robot does not have a compass or odometers. We carefully study the information space (belief space) of the robot, and establish its capabilities in terms of mapping the environment and accomplishing tasks, such as navigation and patrolling. The information space can be nicely characterized using the notion of *order type*, which provides information powerful enough to determine which points lie inside the convex hulls of subsets of landmarks.

1 Introduction

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). This paper considers the question:
What can a robot learn about its environment from an extremely limited sensor?

The answer to this question for various robot systems comes from carefully studying the resulting information spaces (or belief spaces). Information spaces represent the complete knowledge of the robotic task, considering the histories of actions and observations [11; 12]. Such spaces have been usually applied to construct a complete environment model, and an explicit estimation of the robot position. In the most well-know form of simultaneous localization and mapping (SLAM) [14; 18], the addition of some Gaussian assumptions allows the estimation of the position of the robot and the landmarks through the use of probabilistic filters. Although these approaches have a very impressive implementation success, the probabilistic information spaces they generate are hard to characterize, given that they are infinite-dimensional.

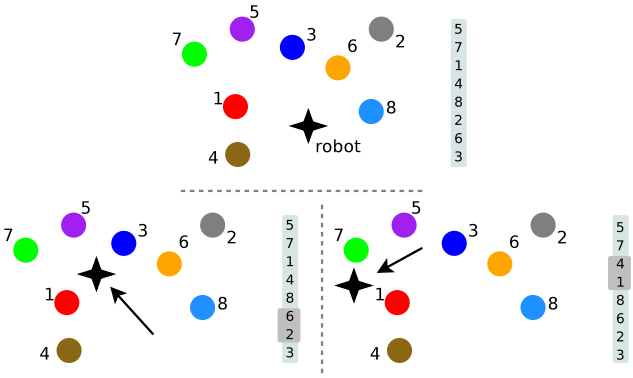


Figure 1: The robot can only detect the cyclic order of landmarks around its position. Estimates of the coordinates of the position of the landmarks, or the robot itself, are not available.

In this paper, we take a minimalist philosophy, which implies that we want the robot, its sensors, and its models to be as simple as possible. Although our problem has not been considered before, this philosophy has been successful in a number of works (e.g., [1; 4; 13; 6]). 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? After establishing what the robot can learn from its simple sensor (see Figure 1), we then illustrate the kinds of tasks that it can solve, including a surveillance/patrolling behavior around the perimeter (convex hull) of landmarks.

Such tasks can be achieved by forgoing complete

SLAM, and instead manipulating the robot's information space. The information space is characterized using the concept of the *order type* of a configuration of points in the plane [7]. 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.

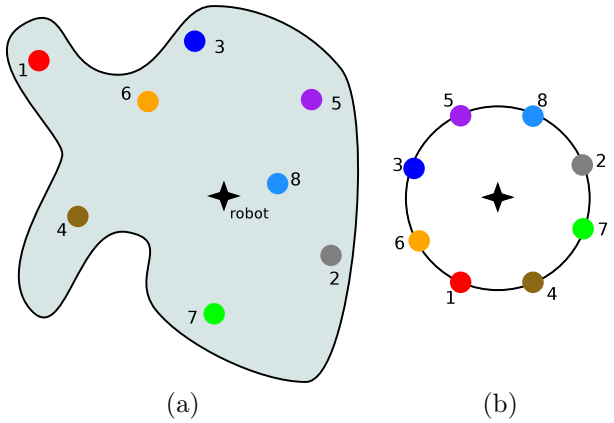
The particular models and assumptions used in this paper are appropriate for resolving basic sensing issues in robotics. There are also close connections to sensor networks, which are becoming increasingly important in security applications. The landmarks can be imagined as "sensors" in a network, and 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.

2 Model

The robot is modeled as a moving point in $\mathbb{R}^2$. Let P be a set of n points in $\mathbb{R}^2$. Let $m : \mathbb{R}^2 \to \{0, \ldots, n\}$ be a mapping such that every point in P is assigned a different integer in $\{1, \ldots, n\}$, and $m(p) = 0$ for any $p \notin P$. The mapping m is referred to as a *feature identification function*, and P is referred to as the set of points selected by m. For a point $p \in P$, a *feature* is defined as the pair $(m(p), p)$. For a set $R \subset \mathbb{R}^2$, an environment E is defined as a pair (R, m). The space of environments $\mathcal{E}$ is the set of all such pairs. Let $q \in SE(2)$ be the configuration of the robot (position in the plane and heading). The *state* is defined as the pair $\mathbf{x} = (q, E)$, and the *state space* X is the set of all such pairs $(SE(2) \times \mathcal{E})$.

The robot is able to detect and recognize features. This is modeled as a mapping $s : \mathbb{R}^2 \to \mathbb{N} \cup \{0\}$. Such mapping is referred to as a *landmark identification function*. For a point $p \in \mathbb{R}^2$ such that $s(p) \neq 0$, a landmark is defined as the pair $(s(p), p)$, and $s(p)$ is called a *landmark label*. Let $P' = \{p \in \mathbb{R}^2 \mid s(p) \neq 0\}$. The mapping s is called *sufficient* with respect to m if $P \subset P'$ and $s(p) = s(r) \Leftrightarrow p = r$, for any $p, r \in P$. If furthermore, $P = P'$, then s is called *complete* with respect to m. If P is not a subset of P', or $s(p) = s(r)$ does not imply that $p = r$, then s is said to make *identification errors* with respect to m.

A landmark sensor is defined in terms of a landmark identification function s. Such sensor is called a *landmark order detector*, and it is denoted with $\text{lod}_s(\mathbf{x})$, for $\mathbf{x} \in X$. The landmark order detector gives the counterclockwise cyclic permutations of



(a) (b)

Figure 2: The landmark order detector gives the cyclic order of the landmarks around the robot. For example, a possible reading for the configuration on (a) is shown in (b). 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 [7,4,1,6,3,5,8,2].

landmark labels as seen from the current state (see Figure 2). Note that no metric information is available to the robot. The robot does not have any coordinate estimate of its position, and the position of the landmarks. Moreover, we assume that the landmark order detector does respect the cyclic order of landmarks, but does not measure the angle between them. In other words, $\text{lod}_s(\mathbf{x})$ does not provide by itself any notion of front, back, left or right with respect to the robot. It is assumed, though, 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 $\text{lod}_s(\mathbf{x})$ no longer 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 [17]).

We assume that landmarks obstruct the visibility of the robot. In this case, only the landmark closest to the robot is detected. In this paper we assume that the environment is of the form $E = (\mathbb{R}^2, m)$,

[1] We might as well define $\text{track}(s(p))$ 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.

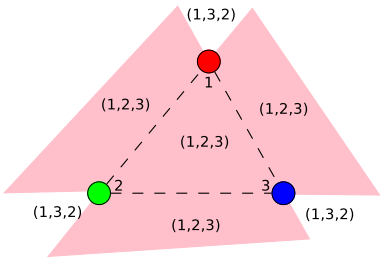


Figure 3: Cyclic permutations of three landmarks. Purely by sensing, the robot cannot even know if it is inside the convex hull defined by the three landmarks. Nevertheless, the orientation of the triangle (the counterclockwise cyclic order of the landmarks as sensed from inside their convex hull) can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

and that the landmarks are in general position (no three landmarks are collinear). Furthermore, we assume that the landmark identification functions are complete in their respective environments, and that the landmark order detector has infinite range. We propose these assumptions for the sake of presentation, and we remove them in an upcoming paper (see Section 5).

3 Order Types and Landmarks

Given the model described in the last section, 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 labeled with 1 to the landmark labeled with 2. Since the reading from the landmark order detector follows a counterclockwise order, the robot can determine whether the landmark labeled with 3 is to the *left* or *right* of the directed segment that connects landmark 1 to landmark 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.

An intuition behind this definition is that 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 [7]. 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 $p_1p_2p_3^+$. Negative orientation is defined in a similar manner, and denoted by $p_1p_2p_3^-$. 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:

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

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 2, $\Lambda(3,7) = \{2,5,8\}$, 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 [7]. 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 $\overline{p_ip_j}$, 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 Order type as an information state

The order type definition is extended naturally to our landmark framework, 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 $lod_s(\mathbf{x}) = [X, i, Y, j, Z]$, in which X, Y, Z are subsequences of $lod_s(\mathbf{x})$ separated by the labels corresponding to landmarks (i, p_i) and (j, 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 $\overrightarrow{p_i p_j}$. Consider any point in the interior of $\overrightarrow{p_i p_j}$, 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 $\overrightarrow{p_i p_j}$. If the robot is placed according to the conditions of the lemma, this sweep can be obtained from the cyclic sequence given by $\mathrm{lod}_s(\mathbf{x})$, starting at j until i is found. By symmetry, $\Lambda(j,i)$ is found also. □

The value of $\Lambda(i,j)$ is determined as follows. The robot is commanded to track landmark (i,p_i) until (i,p_i) disappears (the robot is at p_i). Next, the robot is commanded to track (j,p_j), and at the moment (i,p_i) is detected again, the robot is guaranteed to be on $\overrightarrow{p_i p_j}$, pointing towards p_j. Applying Lemma 1 to the sensor reading, $\Lambda(i,j)$ and $\Lambda(j,i)$ are found.

Now we use Λ to construct the information space generated by the robot model. Consider the state $\mathbf{x} = (q,E)$, which is unknown to the robot. Although is not known in the general case, information about q and E is available to the robot. In particular, partial knowledge of the order type of E can always be computed. Also, using tracking commands together with readings from $\mathrm{lod}_s(\mathbf{x})$, 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 to the position 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 $\mathcal{I}(E)$ be the information states for which Λ' does not contradict the configuration of landmarks in E. Note that up to a relabeling of the landmarks, $|\mathcal{I}(E)|$ is finite. This is because for n landmarks, there are $b = n(n-1)$ index pairs, and thus 2^b possibilities for Λ'. Also, the number of possible values for q' is bounded by the number of combinatorial elements of the line arrangement drawn from the lines passing through each pair of landmarks.

4 Solving Robotic Tasks

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

4.1 Landmarks inside a triangle

The task in this section 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. Given that a triangle has at most one obtuse angle, the robot repeats the orientation procedure 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 $2\pi/3$.

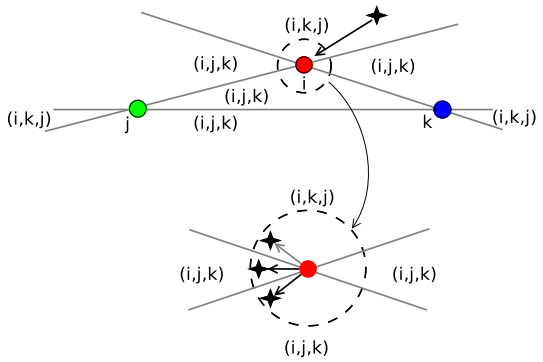


Figure 4: Orientation error. Even a very small control error may find the wrong orientation for the triangle. On the bottom, if the robot follows the top arrow, the orientation is not computed correctly.

4.2 Boundary of the Convex Hull

In this task the robot should determine the landmarks that are on the boundary $\partial\mathrm{hull}(L)$ of the convex hull of the locations of the landmarks. This task can be easily solved, if not efficiently, repeating the previous example for each possible triplet of landmarks, until the landmarks that do not belong to interior of any triangle are found. However, a significantly more efficient algorithm can be constructed based on the well-known *three coin algorithm* for the computation of the convex hull [9; 16]. In its normal 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. Although space restrictions forbid us to give an exact detail of how this algorithm works, it is clear that the information needed can be computed from the corresponding values of Λ. It remains to be explained how to compute the first landmark in the convex hull, and how the radial order is obtained.

The information needed is actually any landmark pair for which $\Lambda(i,j) = 0$. These can be found as follows. First, select randomly a pair of landmark labels, i, j, and compute both $\Lambda(i,j)$ and $\Lambda(j,i)$. If one of them is zero, the required pair has been found. Otherwise, repeat this process on $\Lambda(i,j) \cup \{i,j\}$ or $\Lambda(j,i) \cup \{i,j\}$, whichever has the smallest cardinality. Since at each iteration at least half of the remaining landmarks are discarded, only $O(\log(n))$ sensing operations are needed. Once this pair has been found, the radial order is computed counterclockwise, from any point on the segment joining p_i and p_j, with the robot pointing to either (i, p_i), or (j, p_j).

4.3 Patrolling

In this example 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 minimum.*

To solve this task, the *dual* of the configuration of landmarks is introduced. The dual of a landmark $l = (s(p), p)$, with $p = (p_x, p_y)$, is defined as the labeled line $p^* = (s(p), p_x x + p_y)$. There are well-know properties of such dual arrangements[2; 3], 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 [3]. 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 $\binom{n}{2}+1$ 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 [3].

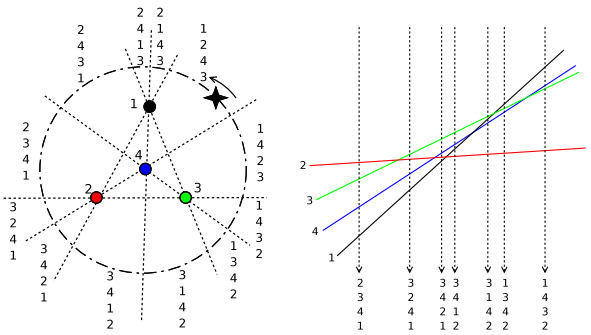


Figure 5: Retrieving the permutations that encode the configuration of landmarks. In (a) the robot travels outside the convex hull of a set of landmarks. This is naturally expressed in the dual line arrangement on (b).

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 2. *Let L be a set of landmarks, let Z be a subsequence of $lod_s(\mathbf{x})$ and containing only the labels corresponding to the landmarks in $\partial hull(L)$ (elements of Z may not appear consecutively in $lod_s(\mathbf{x})$). Then Z is the same sequence for any position of the robot inside hull(L).*

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

From Lemma 2, 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 3. *Let L^* be the set of lines dual to the set of landmarks L. Let m_v be a vertical line, and let $[l_1^*, l_2^*, \ldots, l_n^*]$ for $l_i^* \in L^*$ be sorted according to the y-coordinate of the intersection between m_v and l_i^*. Then the landmarks l_1 and l_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 l_1 is below and l_n is above all such lines. □

Corollary 1. *Let L^* be the set of lines dual to the set of landmarks L. Let m_v be a vertical line, and let $[l_1^*, l_2^*, \ldots, l_{n-1}^*, l_n^*]$ for $l_i^* \in L^*$ be sorted according to the y-coordinate of the intersection between m_v and l_i^*. Let l_1, l_2, l_{n-1}, and l_n be the duals of l_1^*, l_2^*, l_{n-1}^*, and l_n^* respectively. Then l_2 is in $\partial hull(L \setminus \{l_1\})$, and l_{n-1} is in $\partial hull(L \setminus \{l_n\})$.*

Proof. Consider $L^* \setminus \{l_1^*\}$ and $L^* \setminus \{l_n^*\}$, and apply Lemma 3. □

The patrolling problem can be solved as follows. Assume the robot has computed the permutations encoding the dual arrangement of L. The algorithm is based in the following iteration. Set $L_0 = L$. For $i > 0$, find $l \in \partial\text{hull}(L_i)$ such that $\partial\text{hull}(L_i \setminus \{l\})$ does not contain any landmark in W. If no such landmark exists, set $M = L_i$. Else, set $L_{i+1} = L_i \setminus \{l\}$ and repeat.

By Corollary 1, landmarks can be removed from L_i, 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_{i+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.

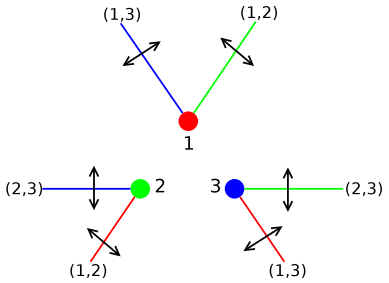


Figure 6: Swap lines. Crossing a half-line swaps the order of the respective landmarks in the reading of the landmark order detector. Such half-lines are called *swap lines*.

4.4 Navigation

The final task described in this paper is navigation. In our 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* $lod_s(\mathbf{x}) = g$ *is reached. Report if* g *cannot be attained given the configuration of landmarks in the plane.*

Before describing the navigation algorithm, we need to describe what is achieved by moving the robot to a place where a particular permutation is sensed. For this purpose, 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 [10], in which a cyclic permutation is an *aspect* of the configuration of landmarks. The decomposition is determined by half-lines, that if crossed, generate a change in the permutation order of a pair of landmarks in $\text{lod}_s(\mathbf{x})$. Each pair of landmarks generate a pair of half-lines, which are referred to as *swap lines* (see Figure 6).

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

Lemma 4. *Let C be the set of cells of the decomposition generated by the swap lines that intersect hull(L), and let $C_i, C_j \in C$. If C_i and C_j are not the same cell, and they are bounded by the same swap line m, then they generate different readings in the landmark order detector.*

Proof. Let $(s(a), a)$ and $(s(a'), a')$ be the landmarks that generated m. Consider a motion of the robot from C_i to C_j in a straight line arbitrarily close to

m. This makes labels $s(a)$ and $s(a')$ to appear consecutive in $\text{lod}_s(\mathbf{x})$ for the duration of the motion. Since C_i and C_j are different, then at least one swap line intersects m between cells C_i and C_j. Let such swap line be generated by landmarks $(s(b), b)$ and $(s(b'), b')$. Crossing this line swaps the order of $s(b)$ and $s(b')$. This swapping could be reverted if the other swap line generated by $(s(b), b)$ and $(s(b'), b')$ is crossed, or if one of $(s(b), b)$ or $(s(b'), b')$ swaps with all the other landmarks. The first situation is not possible, since both swap lines lie in the same line, and m can only intersect one of them. The other case implies that $s(a)$ and $s(a')$ are at some instant not consecutive in $\text{lod}_s(\mathbf{x})$. This is not possible by traveling arbitrarily close to m. Thus, the readings of $\text{lod}_s(\mathbf{x})$ from C_i and C_j 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 1. *Let C be the set of cells of the decomposition generated by the swap lines that intersect hull(L). Then for any two different cells C_i and C_j, the cyclic permutations generated by $lod_s(\mathbf{x})$ when the robot is inside C_i or C_j 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 4. 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. □

By Theorem 1 it is known that different "places" will have different cyclic permutations associated. Given that the robot does not have the landmarks coordinates, the exact geometrical decomposition cannot be constructed. Nevertheless, the robot can navigate such that a particular cyclic permutation g appears in the landmark order detector. First of all, the robot has to decide if g is attainable in the configuration of landmarks. This can be decided using Λ as follows:

Lemma 5. *Let $g = [s(p_1), s(p_2), \ldots, s(p_n)]$, and let $g(s(p_i), s(p_j))$ be the set whose elements are the elements of the subsequence of g starting at $s(p_{i+1})$, ending at $s(p_{j-1})$. If g is attainable in the configuration of landmarks of L, then for each landmark label $s(p_i)$ there is a landmark label $s(p_j)$ such that $\Lambda(s(p_i), s(p_j)) = g(s(p_i), s(p_j))$. The directed line passing from p_i to p_j is called the polar line of (i, p_i), and (j, p_j) is called the pole of (i, p_i).*

Proof. Suppose that g is attained when the robot is at point p. Consider the infinite ray r starting at landmark p_i and passing through p. Now rotate r clockwise, with p_i as a pivot, until r hits a landmark, say $(s(p_j), p_j)$. Then we have that $\Lambda(s(p_i), s(p_j)) = g(s(p_i), s(p_j))$, otherwise the order required for g is not attained ((i, p_i) or (j, p_j) would appear in the wrong place according to g). □

While the region in which g is attained is bounded by polar lines, not all polar lines intersect such region. However, a landmark and its pole appear consecutive in $\text{lod}_s(\mathbf{x})$ if the corresponding polar line bounds the goal region. Thus, the search for the search for the goal permutation g is reduced to such polar lines. Suppose that a polar line is determined by landmarks $(s(a), a)$ and $(s(b), b)$. If both of them belong to the boundary of the convex hull, then the robot traverses the line segment $\overline{ab}$ until g is found. This is done, for example, by tracking $(s(a), a)$ and then tracking $(s(b), b)$ (see Figure 7). If the landmarks do not belong to the boundary of the convex hull, the intersection of the polar line with the convex hull is found by the robot traveling in the boundary of the convex hull, until landmark labels $s(a)$ and $s(b)$ swap places. At this point, the robot tracks any of the landmarks, which traverses at the same time the polar line. Note that the robot may not need to traverse the convex hull boundary to find this intersections, since this information may be already available from Λ. Note also that the tracking takes place once $s(a)$ and $s(b)$ swap places in $\text{lod}_s(\mathbf{x})$, thus the robot travels arbitrarily close, but not exactly on the polar line.

5 Future Work

Given that the Λ and λ functions are equivalent, 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.

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

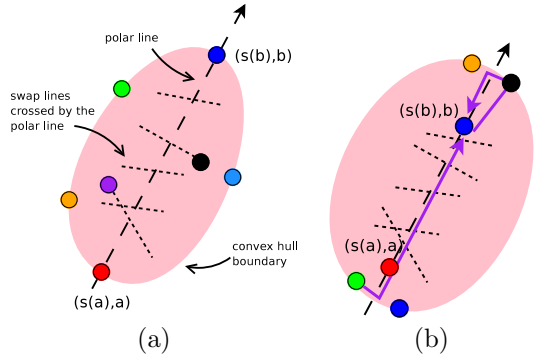


Figure 7: The two general cases for navigation in the polar lines. In (a) the polar line is determined by two landmarks in the boundary of the convex hull, and no further computation is required. In (b), the intersection of the polar line with the boundary of the convex hull is found by a change in the order of the landmarks which determine the polar line. With this, the robot is able to traverse the polar line.

a very hard question. Just deciding if a sequence of permutations can be realized in the plane is NP-hard[15]. Moreover, representing such coordinates may require exponential number of bits[8]. 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.

We are currently working on removing some of the assumptions made on this paper. 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 [5], in the context of sensor networks. Determining the relations between neighborhoods of landmarks, also allows the introduction of environment obstacles.

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. Deciding if a sequence of permutations can be realized in the plane is NP-hard [15]. Moreover, representing such coordinates may require exponential number of bits [8]. 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.

Acknowledgments. The authors would like to thank Sariel Har-Peled and Jeff Erickson for very helpful suggestions and discussions.

References

[1] R. A. Brooks. Intelligence without representation. *Artificial Intelligence Journal*, 47:139–159, 1991.

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

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

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

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

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

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

[8] 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.

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

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

[11] M. Littman L. Kaelbling and and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1998.

[12] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006. Available at http://msl.cs.uiuc.edu/planning/.

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

[14] 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.

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

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

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

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