Learning Combinatorial Information from Alignments of Landmarks

Luigi Freda* Benjamín Tovar† Steven M. LaValle†

* Dipartimento di Informatica e Sistemistica
Universitá di Roma "La Sapienza"
Via Eudossiana 18, 00184 Roma, Italy
freda@dis.uniroma1.it

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

***Abstract*— This paper characterizes the information space of a robot moving in the plane with limited sensing. The robot has a landmark detector, which provides the cyclic order of the landmarks around the robot, and it also has a touch sensor, that indicates when the robot is in contact with the environment boundary. The robot cannot measure any precise distances or angles, and does not have an odometer or a compass. We propose to characterize the information space associated with such robot through the *swap cell decomposition*. We show how to construct such decomposition through its dual, called the *swap graph*, using two kinds of feedback motion commands based on the landmarks sensed.**

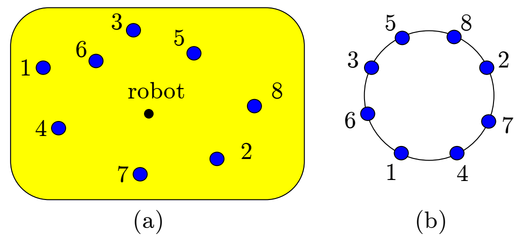

Fig. 1. The landmark order detector gives the cyclic order of the landmarks around the robot. For example, a valid reading for the configuration on (a) is shown in (b). Note that only the cyclic order is preserved, and that the sensed angular position of each landmark may be quite different from the real one. Thus, the robot only knows reliably, up to a cyclic permutation, that the sequence of landmarks detected is [7,2,8,5,3,6,1,4].

I. Introduction

This paper introduces a robot that has minimal sensing. The robot has two sensors, a landmark detector, which reports the cyclic order of landmarks, and a touch sensor, that indicates when the robot is in contact with the environment boundary. What information such robot can learn from its environment? For example, is such information sufficient to perform navigation? To answer these questions, we propose to study the information space associated with this robot model. Information spaces consider the whole histories of actions and observations [9], therefore representing the complete knowledge of the robotic task.

Landmarks generally represent distinctive environment features (e.g., corners) or a priori known objects (e.g., radio beacon) which can be detected by robot's sensors. A considerable amount of literature has focused on the problem of localization using landmarks. Many researchers proposed different triangulation approaches to localize a robot which can measure only landmark bearings [1], [3], [11]. Some researchers addressed the problem of landmark placement, i.e, placing a set of landmarks in a given environment so that a robot can localize itself from any point [3], [14]. In these context, landmarks can be assumed to be distinguishable [13] or identical [12]. Some authors suggest to use both natural and artificial landmarks to localize a robot during a navigation [6]. In SLAM algorithms, the robot automatically determines landmarks while building a map.

Landmarks can be also considered as *distinctive visual events* [10] which can be used to identify distinct places. Using suitable control strategies, distinct places can be linked to build a topological network description of the environment [8]. Indeed, topological maps play a central role in a hierarchical description (cognitive map) of a large-scale environment. In [4] an algorithm is proposed for building a graph-like map of an unknown environment. In [2] a topological approach to SLAM is presented: the topology of the explored free space is used to localize the robot.

In this paper we extend the robot model presented in [16] and we show how to build a topological representation of the environment. In [16] the robot was confined to the convex hull of the set of landmarks, because it could only move bearing towards landmarks. In the present work the robot is also provided with a touch sensor, and in certain situations, can move away from a landmark. This allows the robot to leave the convex hull of the set of landmarks. In comparison, in [16], the information states are compactly represented with the order type [5] of the configuration of landmarks, while in this paper it takes the form of a decomposition of the plane through an aspect graph. The present paper can be considered as an extension to the work proposed in [10]. In that work, Levitt and Lawton subdivide the plane into local *viewframes*, in which a robot can localize itself with a local *landmarks reference frame*, using angular information without any range information. A viewframe is a face of the line arrangement spawned by the lines incident to each pair of landmarks. Levitt and Lawton use some information of the underlying landmark tracking process (in addition to the cyclic order of landmarks) to detected when the robot crosses a line and enters a new viewframe. In the present paper, we abstract the underlying landmark tracking process, and work uniquely with the information provided with the cyclic order of landmarks. This does not allow the construction of the full line arrangement. Instead, we have to work with the arrangements of half-lines that start at each landmark position.

These arrangements of half-lines spawn a cell decomposition in which distinctive places are identified as location in which the landmark sensor returns a constant circular order. An algorithm is proposed to build a topological representation of this decomposition which can be used to fulfil subsequent navigation tasks.

The paper is organized as follows. In Sect. II we describe the robot model and the working assumptions. In Sect. III the order type of the configuration of landmarks is introduced and used to encode the information states of the robot. In Sect. IV-V the swap cell decomposition is presented and its properties are analyzed. Sections VI and VII-B describe how to perform navigation and exploration task with the given robot model.

II. Model

Let P be a set of n points in $\mathbb{R}^2$. Let $s : \mathbb{R}^2 \to \{0, \ldots, n\}$ be a mapping such that every point in P is assigned a different integer in $\{1, \ldots, n\}$, and $s(p) = 0$ implies $p \notin P$. Such mapping s is referred to as a *landmark identification function* and P is referred to as the set of points selected by s. Given a point $p \in P$, the pair $l = (s(p), p)$ is referred to as a *landmark*, and $s(p)$ is called a *landmark label*. Let L be the set of all the landmarks $(s(p), p)$, $p \in P$, and let $\mathrm{hull}(L)$ be the convex hull spawned by the landmarks positions. The robot is modeled 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. For a set $R \subset \mathbb{R}^2$, an *environment* E is defined as a pair (R, s). In this paper we assume that the boundary ∂R of R, is a simple piece-wise analytic closed curve, and that R is a convex set. The space of environments $\mathcal{E}$ is the set of all such pairs. 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 has two sensors, the *landmark order detector*, and a *touch* sensor. The landmark order detector provides the counterclockwise cyclic ordering of the landmark labels around the position of the robot (see Figure 1). We write $\mathrm{lod}(\mathbf{x})$ to refer to the reading of the landmark order detector at state $\mathbf{x} \in X$. No precise metric information is available, and given that the robot does not have a compass, readings from $\mathrm{lod}(\mathbf{x})$ are different up to a cyclic permutation of the landmark labels. The landmark order detector is assumed to have unbounded range. The touch sensor indicates if the robot is in contact with the environment boundary. We write $\mathrm{touch}(\mathbf{x})$ to refer to the reading of the touch sensor on state $\mathbf{x}$.

A. Motion primitives

Given a label $s(p) \in \mathrm{lod}(\mathbf{x})$, a $\mathrm{chase}(s(p))$ motion primitive commands the robot to move towards point p. When the robot arrives at p, the cyclic order of $s(p)$ in $\mathrm{lod}(\mathbf{x})$ is undefined. This serves as model for the termination condition of $\mathrm{chase}(s(p))$: when the landmark is not detected anymore, then the robot is *on top* of the landmark. Note that this is only a simplification for further developments, and similar termination conditions can be stated without the robot being *on top* of the landmark. Also note that during the execution of $\mathrm{chase}(s(p))$, the coordinates of the landmark and robot position remain unknown.

If $|L| > 2$, for two labels $s(p), s(r) \in \mathrm{lod}(\mathbf{x})$, the motion command $\mathrm{repel}(s(p), s(r))$ first moves the robot to p with a chase motion primitive, and then moves the robot away from p and r, following the line defined by p and r. The

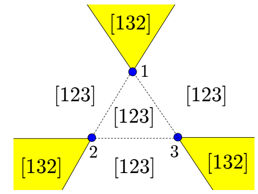

Fig. 2. Cyclic permutations of three landmarks. Purely by sensing, the robot cannot determine whether it is inside the convex hull defined by the three landmarks. Nevertheless, the orientation of the triangle (the counterclockwise cyclic order of the landmarks as sensed from inside their convex hull) can be determined with an information state.

repel motion primitive terminates when $\mathrm{touch}(\mathbf{x})$ reports that the robot is at the environment boundary. The cyclic ordering of $\mathrm{lod}(\mathbf{x})$ provides enough feedback information to execute $\mathrm{repel}(s(p), s(r))$ without any exact measurements. With this feedback, the robot may not be able to follow exactly this line, but as we present later, this is enough for our purposes. Denote with pr the oriented half-line that would be ideally followed by the robot with $\mathrm{repel}(s(p), s(r))$. With a general position assumption in which no three landmarks are collinear, the following observations can be made:

1) When the robot is close enough to pr, the landmark label $s(p)$ and $s(r)$ are consecutive in $\mathrm{lod}(\mathbf{x})$

2) The labels $s(p)$ and $s(r)$ swap their orderings in $\mathrm{lod}(\mathbf{x})$ every time the robot crosses pr

3) If the robot is close enough to the right (resp. left) of pr then $\mathrm{lod}(\mathbf{x}) = [...s(p), s(r), ...]$ (resp. $\mathrm{lod}(\mathbf{x}) = [...s(r), s(p), ...]$). Therefore, in a close vicinity of pr, the robot can distinguish on which side of pr is moving.

Assuming the existence of control errors, the robot can approximately follow pr through a continuous zig-zag: starting close to pr and heading away from p, the robot turns left or right when $s(p)$ appears before or after $s(r)$ in $\mathrm{lod}(\mathbf{x})$, respectively.

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 [15]). Certainly, more information besides the landmarks cyclic ordering can be obtained from a particular implementation [10], but such information is discarded. This simplifies the model such that an information space can be completely characterized. The extra information may be used, for example, to increase robustness.

III. Encoding the Information States

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 sensor 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 2). Nevertheless, consider the robot travelling 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

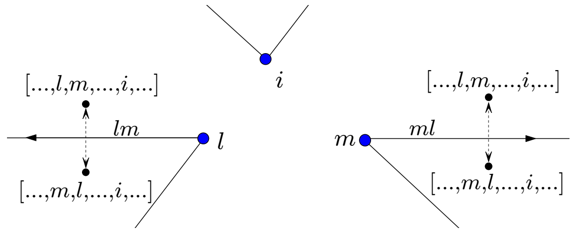

Fig. 3. The swap lines lm and ml associated to the couple of landmarks l and m. If x is arbitrarily close to lm or ml, then the two landmark labels l and m become adjacent in $\text{lod}(\mathbf{x})$. When the robot crosses lm or ml, l and m swap.

2. Thus, the robot can combine sensing with action histories to recover some structure of the configuration of landmarks.

A. Order Types and Landmarks

In [16] we generalized the previous idea, encoding the information states with the *order type* of the configuration of landmarks. The order type was sufficient to encode all the knowledge the landmark order detector can provide for a robot that only has the chase motion primitive (no repel), and that moves inside $\text{hull}(L)$. For a configuration of labeled points, an order relation $\leq$ can be defined through the relative orientation of three points, which is computed as follows [5]. 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

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

is strictly bigger than 0, and this is denoted by $p_1p_2p_3^+$. 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:

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

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). This 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, as it is shown in [16]. In this way, a partial knowledge of Λ encodes an information state.

If the robot would move outside $\text{hull}(L)$, there would be some information that cannot be described using uniquely order types, as it is shown later in this paper. To deal with this cases, the robot model is extended with an additional motion command (repel), and with an additional sensor (touch). With these additions, the information space can be represented as a decomposition of the plane, as it shown in the next section.

IV. Swap Cell Decomposition

Given two different landmarks $l, m \in L$, consider the half-line lm collinear with l and m, that starts at l, and is not incident to m. The half-line lm is referred to as a *swap line*. The swap line ml is defined analogously. Each pair of landmarks defines a pair of swap lines (see Figure 3). When the robot is arbitrarily close to lm or ml, l and m appear consecutive in $\text{lod}(\mathbf{x})$. If the robot crosses any of these swap lines, then l and m *swap* their order in the landmark order detector.

If the robot is on either lm or ml, l and m are collinear with the robot, and a priori, the robot cannot tell which of them is closer. However, the robot can easily determine which landmark is closer during the execution of a repel motion command as follows. Assume that l' and m' are the landmarks that become collinear with the position of the robot while $\text{repel}(l, m)$ is executed, and that l' is the landmark closer to the robot. Using $\Lambda(m, l)$, the robot can determine if both l' and m' lie to the left or to the right of the lm half-line [16]. If after the alignment $\text{lod}(\mathbf{x}) = [l, m, \ldots, l', m' \ldots]$ or $\text{lod}(\mathbf{x}) = [m, l, \ldots, l', m' \ldots]$, then l' and m' are to the right of lm. This is determined by the counterclockwise order given by $\text{lod}(x)$. Note that in both cases, l' appears closer to the pair l, m. The case for the left side is analogous, but with m' closer to the pair l and m. For simplicity, we will further assume that $\text{lod}(\mathbf{x})$ also reports which landmark is closer. Note that this assumption is not really necessary given that such information can be retrieved combining sensing and movements.

Consider the decomposition induced on R by all the swap lines. Each open 2-dimensional cell of these decomposition is called a *swap cell*. This decomposition called the *swap cell decomposition* can be considered as the base of an aspect graph [7], in which a cyclic permutation is an *aspect* of the configuration of landmarks. Given that the swap lines are half-lines, swap cells are in general not convex. The reading of $\text{lod}(\mathbf{x})$ inside a swap cell is constant (up to a cyclic permutation). Moreover, adjacent swap cells differ only on the ordering of two landmark labels. These correspond to the landmarks generating the swap line separating the two swap cells.

Let K be the set of all of the swap cells. Abusing notation, for cells $K' \subset K$, we let $\text{lod}(K')$ be the set of all possible readings from cells in K'. Also, for a cell $C \in K$, we let $\text{lod}(C)$ be the unique reading obtained when the robot is inside C. A reading in $\text{lod}(K')$ has exactly $|L|$ landmark labels, since the robot is not on top of any landmark. In the general case $|K| \geq |\text{lod}(K)|$. That is, one reading may correspond to more than one cell. The swap cell decomposition has other interesting properties. First, we will discuss some simple examples, and then we will discuss in more detail these properties.

A. Examples

In the cases when $|L| < 3$, there is a single, unbounded swap cell. With $|L| = 3$ (see Figure 2), there are four swap cells ($|K| = 4$), but only two sensor readings are possible ($|\text{lod}(K) = 2|$).

In Figures 4 and 5, two examples with four landmarks are shown. Note that the case shown on the left of Fig. 4 is not "stable", given that two swap lines that are not generated by the same landmarks pair have to be parallel. Therefore the case shown in Fig. 5 will appear exclusively under a general

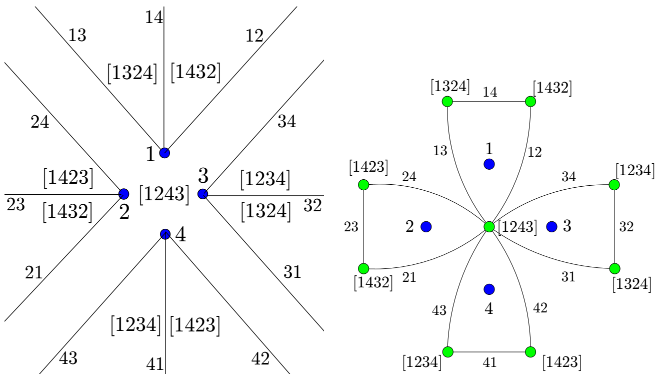

Fig. 4. *Left*: an example with four landmarks. The sensor readings associated to the cells are shown in squared brackets. Note that adjacent swap cells differ just on a swap between adjacent labels. This arrangement of landmarks is not stable: two swap lines which are not generated by the same landmarks pair have to be parallel. *Right*: the swap graph corresponding to the landmark deployment on the right.

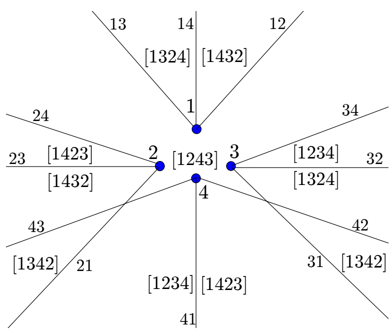

Fig. 5. In this second example with four landmarks, we have a different swap cell decomposition outside the convex hull. A robot provided only with $\mathrm{lod}(\mathbf{x})$ and a chasing motion primitive can not distinguish this case from the previous one.

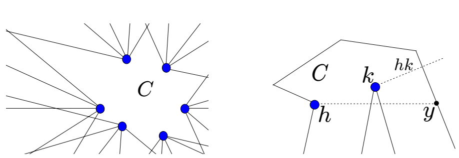

Fig. 6. *Left*: an example of a swap cell with $2n$ edges: all the $n = 6$ landmarks are the vertices of the swap cell C. *Right*: a swap cell C with a landmark h on its boundary. Given a point $y \in cl(C)$, the line segment hy can not intersect the two edges of another reflex vertex (landmark) k on ∂C, otherwise the swap line hk would subdivide C in two distinct swap cells.

position assumption. This is, incidentally, what motivate us to add the repel motion command to the work on [16]. Note that inside the convex hull of the swap cells, we did not have to worry about the differences between Figures 4 and 5.

V. Swap Decomposition Properties

Theorem 1: In the swap cell decomposition for set of landmarks L, with $n = |L|$:

1) A swap cell can have at most $2n$ swap lines in its boundary.

2) Any swap cell $C \in K$ is a star-shaped polygonal region[1].

Proof: Both statements follow directly from the fact that the swap cell decomposition is a complex defined over $n(n-1)$ half-lines.

1. The swap lines starting at a landmark $l \in L$ partition the plane in $n-1$ regions. Thus, for each particular cell, a landmark may provide at most two swap lines. Hence, the cell has at most $2n$ edges when all the landmarks are considered. This bound is tight, as shown on the left of Figure 6).

2. The edges of any swap cell are the result of half-line intersections, and thus the boundary ∂C of C is polygonal. If there is not a landmark in ∂C, then all the vertices of ∂C are the result of swap line intersections. Therefore $cl(C)$ is a polygonal convex and star-shaped. If ∂C has only one landmark as a vertex, then ∂C is a polygonal boundary with only one reflex vertex[2], and thus C is star-shaped. For the remaining cases, let h be a landmark defining a vertex in ∂C, and assume there is a point $y \in C$ such that the line segment $hy \nsubseteq C$ (see Figure 6 right). The line segment hy has to cross two edges of ∂C incident to another landmark in ∂C, say k. This is not possible, because the swap line hk divides C, contradicting that $y \in C$. ■

The following lemma describes how landmark labels can swap when the robot travels in a line segment:

Lemma 1: If p and r are two points belonging to different swap cells, then any pair of landmark labels may swap ordering at most once when the robot moves along the line segment joining p and r.

Proof: Only one swap line of each pair of landmarks may intersect the line segment pr. This is because p and r belong to different swap cells, therefore pr cannot be collinear with any swap line. Thus, each label swapping involves a different pair of landmarks. ■

The proof of the following lemma was presented in [16].

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

The two following theorems give more insights about the information encoded in the swap cell decomposition (for the proofs refer to [16]).

Theorem 2: Two swap cells bounded by the same swap line generate different sensor readings in the landmark order detector.

Theorem 3: If C and C' are two different swap cells that intersect $\mathrm{hull}(L)$, then $lod(C) \neq lod(C')$.

The following corollary to Theorem 3, extends the uniqueness of sensor readings inside the $\mathrm{hull}(L)$.

Corollary 1: If C is a cell that intersects $\mathrm{hull}(L)$, then there is no swap cell C' such that $lod(C') = lod(C)$.

Proof: If C' does not intersect $\mathrm{hull}(L)$, then by Lemma 2 $\mathrm{lod}(C) \neq \mathrm{lod}(C')$. Otherwise, $\mathrm{lod}(C) \neq \mathrm{lod}(C')$ by Theorem 3. ■

Unfortunately, the previous result cannot be generalized to all the cells in the decomposition:

[1]A subset $S \subset \mathbb{R}^2$ is star-shaped if there exists a point $p \in S$ such for any point $r \in S$, the line segment connecting p and r is completely contained in S.

[2]A vertex is called *reflex* if its incident edges form an angle strictly greater than π.

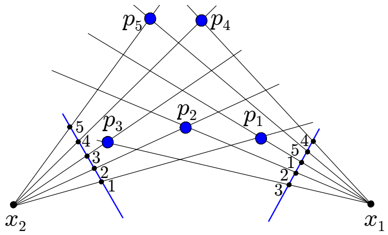

Fig. 7. A construction for obtaining a deployment of landmarks such that two points outside the convex hull of the landmarks are associated to the same cyclic reading $[1, 2, 3, 4, 5]$. First, two points x_1 and x_2 are chosen. Second, the cyclic permutation $[1, 2, 3, 4, 5]$ is represented on two segments in the form of two equivalent permutations. Third, each landmark position 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.

Theorem 4: If C and C' are two different swap cells that do not intersect hull(L), then lod(C) and lod(C') may not necessarily be different.

Proof: Refer to the construction on Figure 7. ■

For landmarks l and m, assume that the robot is moving away from l during the execution of repel(l, m). If the robot is not at the intersection of lm with another swap line, then there is exactly one swap cell to right C_r, and one swap cell to left C_l of the robot, considering the frame of reference that lm provides. Then the readings of C_r and C_l differ only in the ordering of l and m. Particularly, $lod(C_l) = [\dots, m, l, \dots]$ and $lod(C_r) = [\dots, l, m, \dots]$. This information will be used to build a topological representation of the unknown environment.

VI. Goal-Based Navigation

Given a desired sensor reading g, the robot can be commanded to reach a swap cell C, for which $lod(C) = g$. No a priori knowledge of the landmarks deployment, or of the swap cell decomposition is assumed. In [16] it was shown how to perform this for sensor readings inside hull(L). Here we extend these ideas for sensor readings outside the convex hull, also using the repel motion command to have more efficient movements. First, we prove a relation between cell C and its sensor reading g.

Theorem 5: Given a cyclic permutation of landmark labels g, if a swap cell C exists such that $lod(C) = g$, then it is bounded by a swap line lm, for which l and m appear consecutively in g.

Proof: Consider the change in the sensor reading when the robot enters cell C, crossing a swap line lm. This means that l and m appear consecutively just before entering C, and after swapping, remaining consecutive inside of C. ■

Theorem 5 provides the base for a simple navigation strategy. Assume that $g = [l_1, l_2, ..., l_n]$. Then the robot executes sequentially repel(l_i, l_{i+1}) and repel(l_{i+1}, l_i), for all the consecutive pairs of landmarks in g. As the robot executes the repel motions, information can be gathered to determine if g can be achieved, before all the repel commands are executed [16]. As the robot moves on a swap line, the readings of the cells to the left or to the right of the robot are determined as explained before.

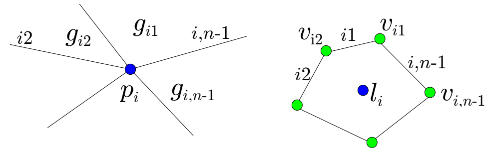

Fig. 8. *Left*: the landmark $l_i = (i, p_i)$ and the $n - 1$ swap lines starting from p_i. The sensor readings $g_{i1}, g_{i2}, ..., g_{i,n-1}$ are respectively associated to the $n-1$ swap cells $C_{i1}, C_{i2}, ..., C_{i,n-1}$ incident to p_i. *Right*: the swap graph representation of the group of cells $C_{i1}, C_{i2}, ..., C_{i,n-1}$. The node v_{ij} represents the swap cell C_{ij}.

VII. Swap Graph and Topological Map Building

The *swap graph* $G(V, E)$ is a data-structure representing the dual of the swap cell decomposition. A vertex $v \in V$ represents a swap cell with its associated sensor reading. An edge $e \in E$, incident to vertices $v, u \in V$, is labeled with lm if the associated swap cells are neighbours through the swap line lm (that is, the readings encoded in v and u differ only in the ordering of l and m).

The swap graph offers an alternative to order types for representing the information space of our robot model. Its construction is more complex than that of the order type, but it can deal with information outside hull(L). Also, it handles some degeneracies, as the one presented in Figure 4.

A. Landmark Representation in the Swap Graph

For landmark l_i, let $C_{i2}, C_{i3}, ..., C_{i,n-1}$ be the $n-1$ swap cells incident to l_i. Let $lod(C_{ij})$ be the corresponding sensor reading (see left of Figure 8). Without loss of generality, assume that $g_i = [1, 2, \dots, i-1, i+1, \dots, n]$ is the $(n-1)$-length cyclic permutation returned by lod$(\mathbf{x})$ when the robot is at l_i.

The n-length sensor readings associated to the $n-1$ swap cells surrounding l_i can be easily obtained displacing the landmark label i between each couple of adjacent landmark labels in g_i. For example, $g_{i1} = [i, 1, 2, ..., n]$, $g_{i2} = [1, i, 2, ..., n]$, $g_{i,n-1} = [1, 2, ..., i, n]$. In this case the swap line ij bounds the swap cells C_{ij} and $C_{i,j+1}$. Using the swap cells properties, it is also easy to deduce which swap line separates a pair of these cells in the general case.

The swap graph representation of the group of cells $C_{i1}, C_{i2}, ..., C_{i,n-1}$ is depicted on the right of Figure 8. A cycle of edges on the swap graph represents the star of swap lines expanding from l_i. For sake of clarity, the corresponding enclosed region can be marked with a blue (dark tone) circle labeled l_i.

The swap graph corresponding to the landmark deployment on the left of Figure 4 is shown on the right of Figure 4.

B. Building the Swap Graph

The swap graph can be built by driving the robot along all the $n(n-1)$ swap lines associated to the n landmarks. Moving along each swap line, the robot reconstructs the sensor readings of the coasted swap cells and incrementally extend the graph.

The swap graph construction algorithm can be described as follows:

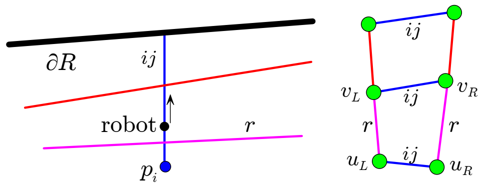

Fig. 9. *Left:* the robot has just crossed the swap line r while performing repel(i, j). The robot can reconstruct the sensor readings associated to the new swap cells on its left and right. *Right*: correspondingly, two vertices v_R and v_L are added to the swap graph; u_R and u_L are the vertices corresponding to left and right swap cells before r is crossed.

1) For each landmark $l_i = (i, p_i) \in L$, chase l_i, and build the "ring" of vertices around l_i, as described in Sect VII-A. Some of this new vertices may actually represent a swap cell already represented in G, but there is not enough information to detect this yet.

2) For each swap line ij starting at l_i (with corresponding pair $l_j = (j, p_j)$), repel(i, j) until the robot reaches the boundary of the environment.

3) While repel(i, j), every time lod$(\mathbf{x})$ changes, G may need to be updated. Call u_R and u_L the vertices corresponding to cells to left and right of ij when the robot is at l_i, and call r the swap line crossed when lod$(\mathbf{x})$ changed (see Fig. 9).

- Vertex v_R, corresponding to the cell to the right of ij is added to G if there is not another vertex $v \in G$ associated with the same sensor reading. If such v exists, v_R is still added if the swap cells corresponding to v and v_R are not bounded by the same swap line. Otherwise let $v_R = v$. An edge labeled with r is added between v_R and u_R. A similar process is repeated for vertex v_L, corresponding to the cell to the left of ij. An edge labeled with ij is also added between v_R and v_L.
- If there is a vertex $v \in G$ such that v is associated with the sensor reading of u_R, and the swap cell corresponding to v is bounded by r, then v and u_R represent the same swap cell, and are merged into a single vertex. The same is repeated for u_L.
- Update u_R to v_R, and u_L to v_L, and continue repel(i, j).

Theorem 6: The swap graph construction algorithm correctly constructs the swap graph G inside R.

Proof: Each swap line in the decomposition is tracked. This guarantees that for each swap cell, at least one vertex is created that represents it. Vertices in G are removed (merged) if and only if the cells they represented are found to be bounded by the same swap line (by Theorem 2). By construction, edges are added to G only when the corresponding vertices represent neighboring swap cells. ■

In the previous section we described a goal-based navigation algorithm without assuming an a priori knowledge of the environment. Now, given a swap graph representation of the environment, we can easily drive the robot from one cell to another. Each swap cell, is unequivocally identified by a vertex of G and its incident edges. Note that two distinct vertices can store the same landmark cyclic permutation but cannot have the same labels on their incident edges (by Theorem 2). Given a vertex $v \in G$, the corresponding swap cell can be reached with a repel motion along one of the swap lines labeling an edge incident in v.

VIII. Conclusions

In this paper we analyzed the information space of a robot which has a landmark order detector and a touch sensor. We presented an algorithm for robot navigation using these capabilities, and introduced the swap graph data structure, to collect all the information available to the robot. The swap graph can be used as a topological description of the environment which allows the execution of goal-based navigation. We proved formally that the robot can construct the swap graph using uniquely the sensing capabilities provided. Our results propose some minimal sensing requirements for navigation and model representation. In other words, if the cyclic order of landmarks and the touch sensor are provided as described in this paper, then a robot can recover some of the environment structure, build a model of it, and navigate. Certainly, the robot movements are not particulary efficient but the implementation on a real robot generally provides more information than the cyclic order of landmarks. This of course can be used to both increase the robustness of the strategies and the efficiency of the movements. As future work, we would like to relax some of the sensing requirements (reliable landmark tracking and unbounded range), by adding some other sensing or movement capabilities.

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] H. Choset and K. Nagatani. Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization. *IEEE Trans. on Robotics and Automation*, 17(2):125–137, 2001.

[3] E. D. Demaine, A. López-Ortiz, and J. I. Munro. Robot localization without depth perception. In *SWAT '02: Proceedings of the 8th Scandinavian Workshop on Algorithm Theory*, pages 249–259, London, UK, 2002. Springer-Verlag.

[4] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. Robotic exploration as graph construction. *IEEE Trans. on Robotics and Automation*, 7(6):859–865, 1991.

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

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

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

[8] B. Kuipers and Y. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. pages 47–63, 1993.

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

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

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

[12] K. Sugihara. Some location problems for robot navigation using a single camera. *Comput. Vision Graph. Image Process.*, 42(1):112–129, 1988.

[13] K. T. Sutherland and W. B. Thompson. Inexact navigation. In *1993 Int. Conf. on Robotics and Automation*, pages 1–7, 1993.

[14] 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.

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

[16] B. Tovar, L. Freda, and S. M. LaValle. Mapping and navigation from permutations of landmarks. Submitted to *20th Int. Joint Conf. on Artificial Intelligence*, 2007.