Z

Combinatorial Filters: Sensor Beams, Obstacles, and Possible Paths

BENJAMIN TOVAR, Northwestern University
FRED COHEN, University of Rochester
LEONARDO BOBADILLA, University of Illinois
JUSTIN CZARNOWSKI, University of Illinois
STEVEN M. LAVALLE, University of Illinois

A problem is introduced in which a moving body (robot, human, animal, vehicle, and so on) travels among obstacles and binary detection beams that connect between obstacles or barriers. Each beam can be viewed as a virtual sensor that may have many possible alternative implementations. The task is to determine the possible body paths based only on sensor observations that each simply report that a beam crossing occurred. This is a basic filtering problem encountered in many settings, under a variety of sensing modalities. Filtering methods are presented that reconstruct the set of possible paths at three levels of resolution: 1) the possible sequences of regions (bounded by beams and obstacles) visited, 2) equivalence classes of homotopic paths, and 3) the possible numbers of times the path winds around obstacles. In the simplest case, all beams are disjoint, distinguishable, and directed. More complex cases are then considered, allowing for any amount of beams overlapping, indistinguishability, and lack of directional information. The method was implemented in simulation. An inexpensive, low-energy, easily deployable architecture was also created, which implements the beam model and validates the methods of the paper with experiments.

Categories and Subject Descriptors: I.2.9 [**Robotics**]: Sensors

General Terms: Algorithms, Design, Experimentation, Theory

Additional Key Words and Phrases: Filtering. Sensor fusion. Foundations of sensor networks. Robotics. Inference. Topology.

ACM Reference Format:
Tovar, B., Cohen, F., Bobadilla, L., Czarnowski, J., LaValle, S. M., XXXX. Combinatorial Filters: Sensor Beams, Obstacles, and Possible Paths. ACM Trans. Sens. Net. X, Y, Article Z (March 1885), 32 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

Imagine installing a bunch of cheap, infrared eye beams throughout a complicated warehouse, office, or shopping center; see Figure 1. Just like the safety beam on a motorized garage door, a single bit of information is provided: Is the beam currently obstructed? Now suppose that there are one or more moving bodies, which could be people, robots, animals, and so on. If the beams are distinguishable and we know the

This work was supported in part by NSF grants 0904501 (IIS Robotics) and 1035345 (Cyberphysical Systems), DARPA SToMP grant HR0011-05-1-0008, and MURI/ONR grant N00014-09-1-1052.
B. Tovar is with the Mechanical Engineering Department, Northwestern University. Evanston, IL 60208 USA. b-tovar@northwestern.edu. F. Cohen is with the Department of Mathematics, University of Rochester. Rochester, NY 14627 USA. fcohen@rochester.edu. L. Bobadilla, J. Czarnowski, and S. M. LaValle are with the Department of Computer Science, University of Illinois. Urbana, IL 61801 USA. {bobadil1,jczarno2,lavalle}@uiuc.edu.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

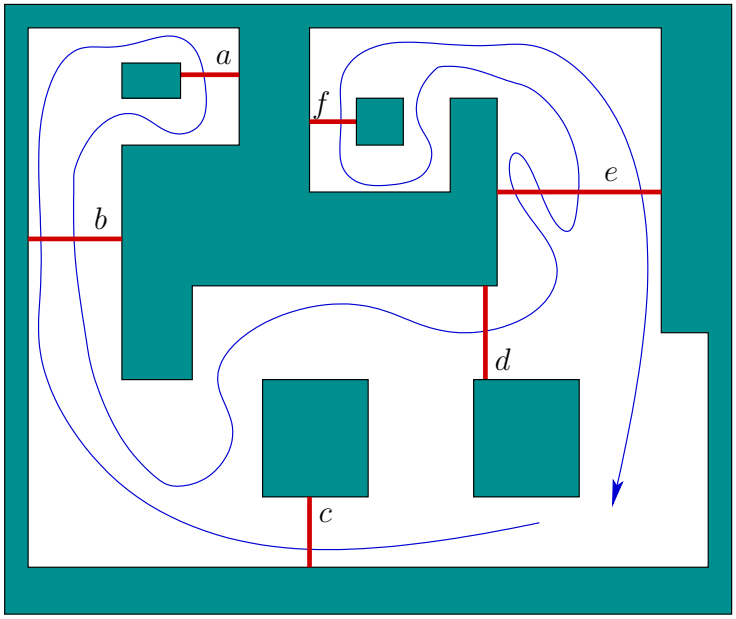


Fig. 1. What can be determined about the path using only the word $cbabdeeefe$, which indicates the sequence of sensor beams crossed?

order in which beams were crossed, what can we infer about the paths taken by the moving bodies? This may be considered as a filtering problem, but with minimal, combinatorial information, in contrast to popular Kalman filters and particle filters. We introduce the notion of *combinatorial filters*, which are minimalist analogs to common Bayesian filters. They instead reduce complexity by employing combinatorial reasoning, which lies at the heart of computational geometry and some parts of computational topology. The idea is to reason in an exact but discrete way about continuous spaces by identifying or representing critical pieces of information.

This paper proposes the study of a family of inference problems that arise from moving bodies crossing sensor beams among obstacles. It turns out that the subject is much more general than the particular scenario just described. In addition to binary detection beams or regions placed around an environment, the mathematical model arises in other contexts. For example, if a robot carries a camera and certain image features critically change, then the event may be equivalent to crossing a "virtual" beam in the environment (see Section 3).

Our questions are inspired by many problems that society currently faces. There is widespread interest in developing *assisted living* systems that use sensors to monitor the movements of people in their homes or hospitals. How much can be accomplished with simple detection beams, which are affordable, robust, and respect privacy? Alternatively, imagine the field of *sensor-based forensics*, in which police investigators or lawyers would like to corroborate or refute a testimony about how people moved at a crime scene (for related work, see [Yu and LaValle 2010] and references therein). A simple verification test based on the sequence of beam crossings might establish whether someone is lying. Other problems include tracking wildlife movement for conservation purposes, landmark-based navigation with outdoor vehicles, sensor-assisted safe child care, and security.

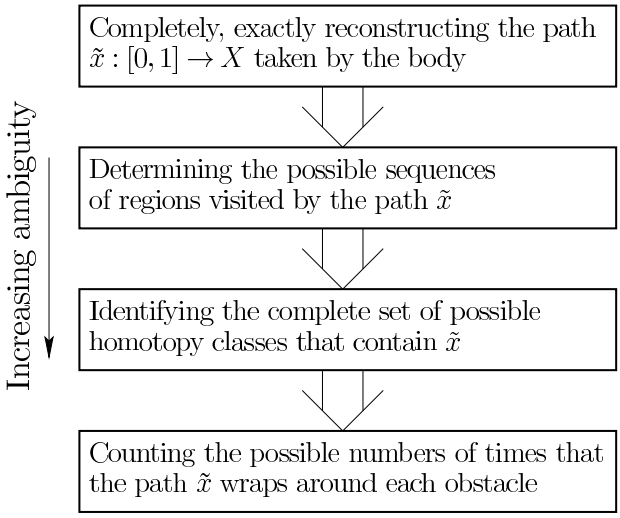


Fig. 2. This paper considers sensors that are clearly too weak to fully reconstruct the path taken by the body, but are nevertheless able to produce useful information regarding the path. Rather than approach the most common problem of complete reconstruction (top of the diagram), we present filtering methods that determine information about the path at three levels of ambiguity. The result of each level in the diagram could be derived from the information directly above it; however, the complete path is never given. Only the sequence of crossed sensor beams is given.

Suppose there is one moving body and we receive information that a particular sequence of sensor beams was crossed. We present a family of methods for reconstructing information about the path taken by the body. See Figure 2, which relates various forms of information that could be obtained about the path, ordered by the amount of ambiguity. Considering the example in Figure 1, note that the beams and obstacles partition the space into regions in which the body can move without being detected. Section 4 develops filtering methods that reconstruct the possible sequences of regions traversed by the body. This is the tightest possible representation of the set of possible body paths that explain the observed data. In Section 5, we then introduce the homotopy equivalence relation on the set of paths. In that case, we reconstruct possible paths up to the resolution of homotopy classes. This is a more coarse, and possibly more compact, summary of the possible paths than the sequences of possible regions. In Section 6, an even more coarse characterization of possible paths is made, in terms of winding numbers: How many times does the path effectively "wrap" around each obstacle?

Our approach is inspired by works in mathematics, algorithms, robotics, and sensor networks. From mathematics, our reconstructions based on homotopy and winding numbers inspired by topology, and are motivated by homotopy and homology, respectively. Furthermore, they are closely related to *word problems* in combinatorial group theory, in which it must be determined whether two words or group presentations are equivalent. In the general group-theoretic setting, such questions go back to 1910 with Dehn's fundamental problems [Dehn 1987]; decidability and complexity results are reviewed in [Epstein et al. 1992].

Regarding algorithms, some of the most closely related works are algorithms that decide whether two paths in a punctured plane are homotopic [Cabello et al. 2002; Efrat et al. 2006]. These algorithms are based on extending vertical lines from each of the punctures (which are interior obstacles in our paper). The vertical lines serve

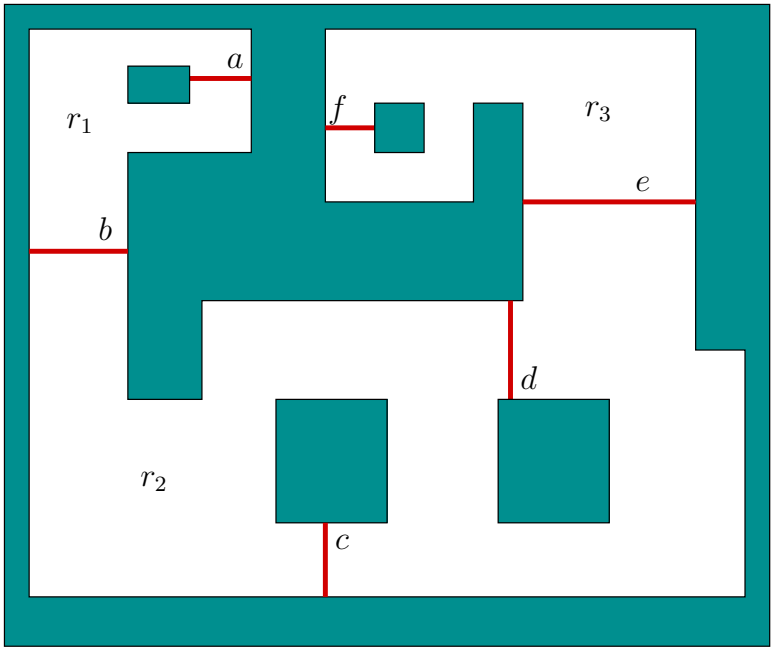
two purposes: First, any given path is represented by the sequence of vertical rays that it intersects. Second, they connect the different fibers of a covering space of the punctured plane. In this context, two paths are homotopic if and only if they have the same endpoints when they are lifted to the universal covering space. Other recent works introduce topological methods to compute shortest paths within a homotopy class [Grigoriev and Slissenko 1998; Kim et al. 2012]. Our work draws inspiration from these; however, we start with *sensor* words and must first convert them into path descriptions. This represents an *inverse problem* that is constrained by the geometry and topology of the sensors and obstacles.

In the context of robotics and sensor networks, we draw inspiration primarily from works that focus on minimalism and also on combinatorial reasoning, which is prevalent in computational geometry. This has led, for example, to the notion of a *relational sensor* [Guibas 2002]. Once combinatorial reasoning is applied, several topological methods have been developed to infer whether there are holes in connectivity [Silva and Ghrist 2006; Lobaton et al. 2010; Marinakis et al. 2002; Sarkar and Gao 2010]. Similarly, works such as [Kim et al. 2005; Shrivastava et al. 2006b; Singh et al. 2007] propose binary proximity sensors to detect and count targets. The binary proximity sensors can be considered as *overlapping beams* that go off when a body is in range. Target tracking is often accomplished with a particle filter, in which each particle is a candidate trajectory of a target. In robotics, the use of particle filters has been very successful in solving tasks such as simultaneous localization and mapping (SLAM) [Castellanos et al. 1999; Choset and Nagatani 2001; Dissanayake et al. 2001; Montemerlo et al. 2002; Parr and Eliazar 2003; Thrun et al. 1998]. Traditionally, the focus of SLAM approaches is the production of an environment representation based on metric information. From a sensing perspective, algorithms such as the ones used in SLAM, are concerned with the problem of sensor fusion, in which several sensors are added to increase the accuracy of a solution. In contrast, others have studied the *minimal sensing requirements* to solve a particular task [Erdmann and Mason 1988; Goldberg 1993; O'Kane and LaValle 2007; Tovar et al. 2007b]. This typically involves a characterization and simplification of the information space associated with the task [LaValle 2006; 2012], which considers the whole histories of commands given to the actuators and sensing observations. From an information space perspective, in [Yu and LaValle 2008] the location of moving bodies is inferred from combinatorial changes in sensing observations. Such combinatorial changes may correspond to *visual events* [Durand 1999], which can be abstracted in our work as sensor beams. An example of this is presented in Section 3, in which the combinatorial changes correspond to the crossing of landmarks within the field of view [Tovar et al. 2007a]. The careful consideration of visual events is the basis for solutions to problems such as localization [Dudek et al. 1998; Guibas et al. 1995] and visibility-based pursuit-evasion [Gerkey et al. 2004; Guibas et al. 1999; Kameda et al. 2006; Lee et al. 1999; Suzuki and Yamashita 1992].

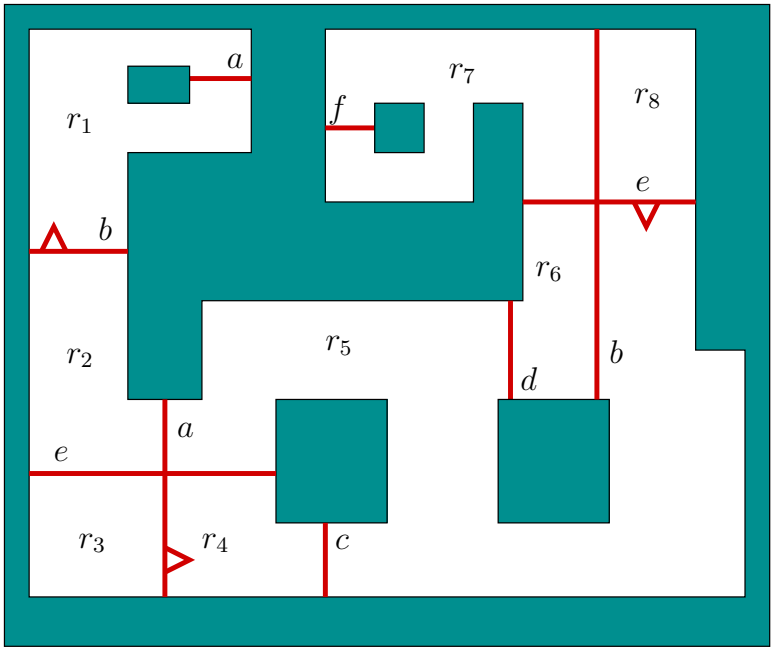
This paper is an expanded and updated form of [Tovar et al. 2009].

2. PROBLEM FORMULATION

Let $W \subseteq \mathbb{R}^2$ be the closure of a simply connected (contractible) open set. A common case is $W = \mathbb{R}^2$. Let $\mathcal{O}$ be a set of n pairwise-disjoint *obstacles*, which are each the closure of a simply connected open set. Let X be the *free space*, which is the open subset of W that has every $o \in \mathcal{O}$ removed. Let $\mathcal{B}$ be a set of m *beams*, each of which is an open linear subset of X. It is possible to generalize $\mathcal{B}$ to allow nonlinear beams without affecting the results in this paper; however, this will be avoided in the presentation to improve clarity.



(a)



(b)

Fig. 3. (a) A simple example, which includes 6 beams and three regions, r_1, r_2, and r_3, in which the body can move without being detected. (b) Beams may be directional, may intersect, and may be indistinguishable. The positive beam direction is indicated by an arrow placed along the beam. There are 8 regions.

The model is defined to allow the cases of both W bounded or unbounded. If W is bounded, then every beam is a line segment with both endpoints on the boundary of X. Figure 3(a) shows an example in which W is bounded and there are four obstacles. Note that beams may connect an obstacle boundary to itself, another obstacle's boundary, or the boundary of W; also, a beam may connect the boundary of W to itself. If W is unbounded, then some beams may be infinite rays that emanate from the boundary of an obstacle, and others may even be infinite lines that are contained in the interior of W.

Regions. The collection of obstacles and beams induces a decomposition of the free space X into connected *cells*. If the beams in $\mathcal{B}$ are pairwise disjoint, then each $B \in \mathcal{B}$ is a 1-cell and the 2-cells are maximal regions bounded by 1-cells and portions of the boundary of X. If beams intersect, then the 1-cells are maximal segments between any beam intersection points or boundary elements of X; the 2-cells follow accordingly. Every 2-cell will be called a *region*. The regions are shown for the examples in Figure 3. It is assumed that beams are arranged in general position so that if a pair of beams intersects, then the intersection must occur at one point.

Body path. Suppose that a *body* moves along a *state trajectory* or *path*, $\tilde{x} : [0, 1] \rightarrow X$, in which $[0, 1]$ is imagined as a time interval, but time scaling is unimportant to our questions. (Alternatively, $[0, t_f]$ could be allowed for any $t_f > 0$.)

Sensor model. If the body crosses a beam, what exactly is observed? Assume that the set of possible $\tilde{x}$ is restricted so that: 1) every beam crossing is transverse (the body cannot "touch" a beam without crossing it and the body cannot move along a beam) and 2) the body never crosses an intersection point between two or more beams (if any such intersections exist).

Let L be a finite set of *labels*. Suppose each beam is assigned a unique label by some bijection $\alpha : \mathcal{B} \rightarrow L$. The sensor model depicted in Figure 1 can be obtained by a sensor mapping $h : X \rightarrow Y$, in which $Y = L \cup \{\#\}$ is the *observation set*. If $\tilde{x}(t) \in B$ for some $B \in \mathcal{B}$, then $h(\tilde{x}(t)) = \alpha(B)$; otherwise, $h(\tilde{x}(t)) = \#$, which is a special symbol to denote "no beam". This is referred to as the *undirected beam* model because it indicates that the beam was crossed, but we do not know the direction.

To obtain a *directed beam* model, let $D = \{-1, 1\}$ be a set of *directions*, in which 1 is called the *positive* and -1 is called the *negative* direction, respectively. In this case, the observation set is $Y = (L \times D) \cup \{\#\}$, and the sensor mapping yields the orientation of each beam crossing. Note that in addition to $\tilde{x}(t)$, the domain of the sensor mapping h must include open subsets of X so that the side of the beam that the body was shortly before time t can be measured.

So far, the beams have been *fully distinguishable* because α is a bijection. It is possible to make $|L| < m$ (the number of beams) and obtain some *indistinguishable* beams, in which case $\alpha : \mathcal{B} \rightarrow L$ is not bijective. It might seem odd that beams cannot be distinguished, but this can occur frequently in practice; see Section 3.

If a collection $\mathcal{B}$ of beams is disjoint, distinguishable, and directed, the case will be referred to as *DDD beams*, which is the most ideal situation. We consider the most general cases, however, in which these conditions are not met. Figure 3(b) shows one such example.

Sensor words. What observations are accumulated after $\tilde{x}$ is executed? Since there is no precise timing information in our problems, all $\#$ observations can be safely ignored, resulting in a sequence $\tilde{y}$, called the *sensor word*, of the remaining observations ($\tilde{y}$ is a kind of *observation history* [LaValle 2006]). For the example in Figure 1, $L = \{a, b, c, d, e, f\}$ and the sensor word is $cbabdeeefe$. For a directed beam with label λ, the symbol λ is used to denote traversal in the positive direction, and λ' is used to denote

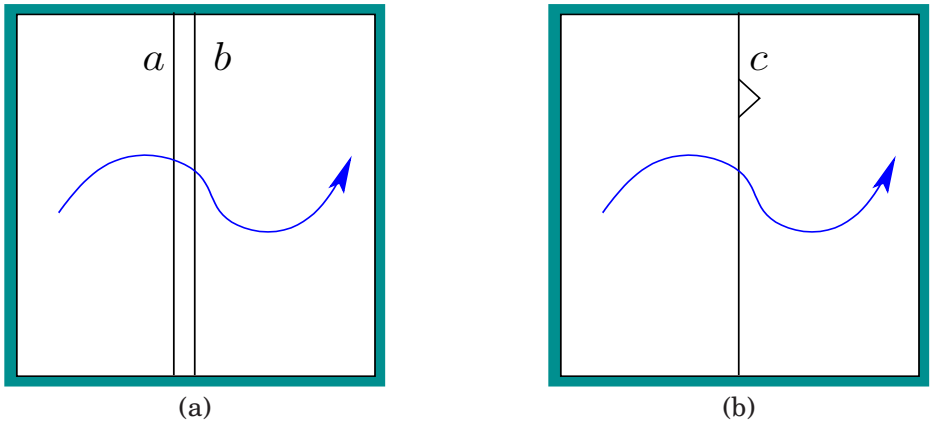


Fig. 4. (a) The left-to-right traversal of the two undirected, nearby, parallel beams yields the sensor word ab. (b) The beams effectively simulate a directed beam, in which c is observed when the original word is ab and c' when the original word is ba. The triangle along the beam indicates the positive c direction.

traversal in the negative direction. A sensor word for directed beams may then appear as $aba'bb'acbc'$, for example.

Note that if two parallel, undirected beams are placed closely together (see Figure 4), then they can simulate one directed beam. This requires an assumption that the pair must always be crossed transversely, rather than crossing one beam and returning.

Inference and filtering. Let $\tilde{Y}$ be the set of all possible sensor words and let $\tilde{X}$ be the set of all possible paths. Let $\phi : \tilde{X} \rightarrow \tilde{Y}$ denote the mapping that produces the sensor word $\tilde{y} = \phi(\tilde{x})$.

Suppose that $\tilde{y}$ has been obtained with no additional information. What can be inferred about $\tilde{x}$? Let $\phi^{-1}(\tilde{y})$ denote the *preimage* of $\tilde{y}$:

$$\phi^{-1}(\tilde{y}) = \{\tilde{x} \in \tilde{X} \mid \tilde{y} = \phi(\tilde{x})\}. \tag{1}$$

The main task in this paper is to characterize the preimage (or collection of plausible paths) from a given sensor word. Section 4 will provide a simple and efficient algorithm that provides an exact characterization of $\phi^{-1}(\tilde{y})$ in terms of sequences of regions that may have been traversed. Furthermore, the method works incrementally as a combinatorial filter by updating its information state [Kuhn 1953; von Neumann and Morgenstern 1944] efficiently as each new observation occurs. This process is analogous to the popular Kalman filter, which computes the next mean and covariance based on the new sensor observation and the previous mean and covariance [Kalman 1960; Kumar and Varaiya 1986]. The Kalman filter falls under the general family of Bayesian filters, to which particle filtering techniques are usually applied [Thrun et al. 2005].

For a sensor word $\tilde{y}_k$ of length k, let $\kappa(\tilde{y}_k)$ denote an information state, which could, for example, be the set of possible current regions. A combinatorial filter efficiently computes $\kappa(\tilde{y}_{k+1})$ using only $\kappa(\tilde{y}_k)$ and y_{k+1}, in which y_{k+1} is the last (most recent) letter in $\tilde{y}_{k+1}$. This implies that $\tilde{y}_k$ does not need to be stored in memory; only $\kappa(\tilde{y}_k)$ is needed. Simple filters of this form will be presented in Section 4.

In addition to simple region-based filters, we provide methods that directly transform the sensor word into topological path descriptions. Section 5 converts the sensor word into one or more elements of the fundamental group. Each element corresponds to a class of homotopically equivalent paths that are possible given the sensor word $\tilde{y}$. This provides a more coarse characterization of the preimage $\phi^{-1}(\tilde{y})$ than the methods

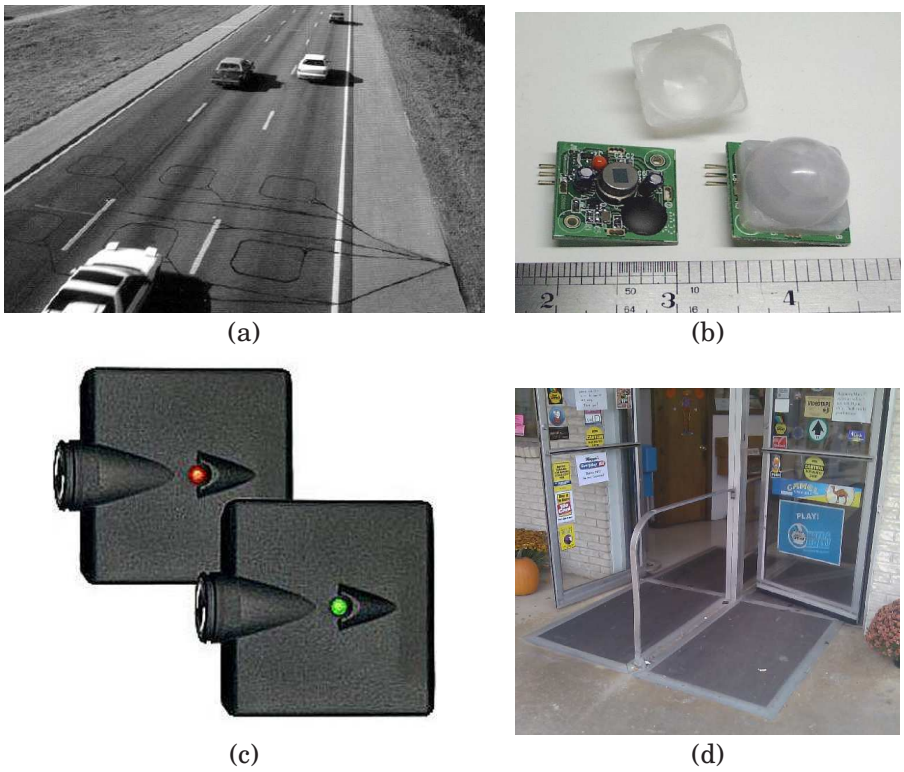


Fig. 5. Some examples simple systems to implement sensor beams: (a) Inductive coils are placed in roads to detect car crossings. (b) Passive infrared sensors detect movement within a specific zone. (c) Infrared garage beams are placed under garage doors for safety, but can be used in many other settings. (d) Pressure mats or cables can be placed along or into the ground.

of Section 4. The information is useful in many contexts, including search-based planning for navigation [Bhattacharya et al. 2011]. Section 6 produces information about $\phi^{-1}(\tilde{y})$ that is even more coarse. Given $\tilde{y}$, a method is presented that calculates the winding number with respect to each obstacle. In other words, it determines the number of times, positive or negative, that the body path wrapped around each obstacle counterclockwise.

3. MODEL MOTIVATION

Before moving on to computing descriptions of $\phi^{-1}(\tilde{y})$, this section motivates the general formulation of Section 2 to illustrate the wide range of settings to which it applies.

In many settings, sensors may be placed in the environment to directly obtain the beam detection behavior. Some examples are shown in Figure 5. In Section 7.1, we describe our own beam detection system, which costs around $5 US per beam. Using simple sensors, virtually any beam model from Section 2 can be implemented.

It is possible, however, to obtain the sensor beam model in a more indirect way. For example, imagine that a robot moves in a large field, in which several landmarks (e.g., radio towers) are visible using an omnidirectional camera. This can be modeled by $W = \mathbb{R}^2$ and $\mathcal{O}$ as a set of point obstacles. Suppose that the landmarks are fully distinguishable and some simple vision software indicates when a pair of landmarks are "on top of each other" in the image. In other words, the robot and two landmarks are

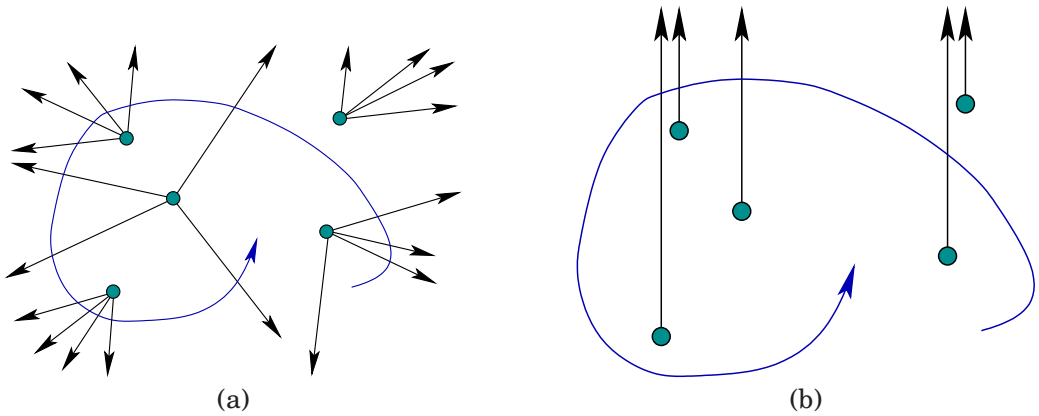


Fig. 6. (a) Imagine how the landmarks (small discs) appear in an omnidirectional camera while traversing the shown trajectory. The arrows show each perceived virtual beam. (b) Here the virtual beams are based on passing directly north of a landmark.

collinear, with one of the two landmarks in the middle. The result is mathematically equivalent to placing $n(n-1)$ beams as shown in Figure 6(a), in which rays extend outward along lines passing through each pair of landmarks.

Several interesting variations are possible based on precisely what is detected in the image. If the only information is that o_i and o_j crossed each other in the image, then all beams are undirected and the two beams associated with o_i and o_j are indistinguishable. If we know whether o_i passes in front of or behind o_j, then the two beams become distinguishable from each other. If we know whether o_i passes to the left or right of o_j in the image, then the beams even become directed. Scenarios such as these motivated the consideration of beam labels and indistinguishability in Section 2.

For another example, consider changing the landmark sensing so that instead of detecting pairwise landmark crossings, the robot simply knows when some landmark is directly south. This could be achieved by using a compass to align the vehicle and noting when a landmark crosses a fixed spot on the image plane or windshield. Figure 6(b) shows virtual beams that are obtained in this way. Directed and undirected beam models are possible, based on whether the sensor indicates the left-right direction that the landmark moves as it crosses the fixed spot. An important property of this model is that the beams do not intersect (assuming the points are not collinear). Section 5 utilizes this property to reconstruct the path up to homotopy equivalence.

4. REGION FILTERS

In this section, we present a simple method to keep track of the possible regions in which the body might be after obtaining the sensor word. The possible sequences of regions traversed can also be computed. Section 4.1 covers the case of a single body, and Section 4.2 provides some extensions to multiple bodies.

4.1. One Body

Suppose that n obstacles $\mathcal{O}$ and m beams $\mathcal{B}$ are given along with L and α (beam labels). Furthermore, assume that W and $\mathcal{O}$ are represented in a way that enables exact, efficient computations. For example, it is sufficient to assume all sets are polygonal. The beams may intersect, may or may not be directed, and some beams may be indistinguishable. Using standard representations of subdivisions (such as the half-edge data structure) and planar decomposition algorithms, the regions and their connectivity in-

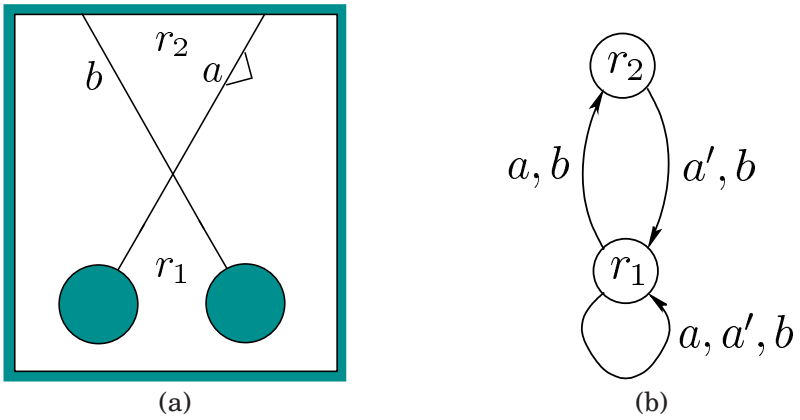


Fig. 7. (a) Two intersecting beams: a is directed and b is undirected; (b) the corresponding multigraph G; multiple edges are compressed into a single edge with a list of the labels that cause the transition.

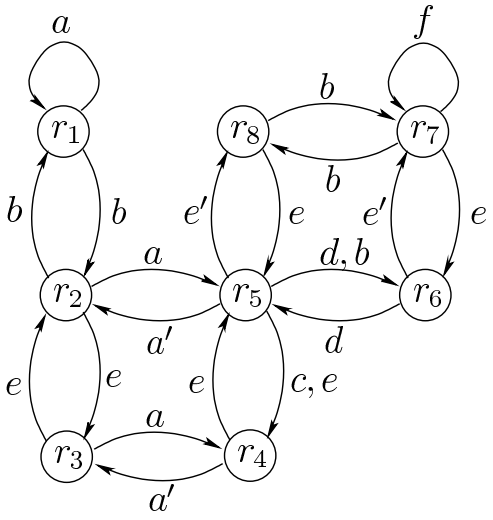


Fig. 8. The multigraph G corresponding to Figure 3(b).

formation can be easily computed (see [de Berg et al. 2000] for an overview of such methods).

Let R_0 denote the set of possible regions that initially contain the body, before any sensor data is observed. Let R_k denote the smallest set of possible regions that contain the body after a sensor word $\tilde{y}_k$, of length k, has been obtained. The task is to design a combinatorial region filter, which is a function ϕ of the form

$$R_{k+1} = \phi(R_k, y_{k+1}) \tag{2}$$

that can be efficiently computed. In the first application of ϕ, R_1 is computed from R_0 and y_1. Each subsequent R_k is similarly computed.

Before implementing the filter ϕ, a special graph is constructed in a preprocessing phase. Let G be a directed multigraph that possibly contains self-loops. Each vertex of G is a region, and a directed edge is made from region r_1 to region r_2 if the corresponding regions are adjacent (an interval along a beam lies on the boundary of r_1 and

r_2). The edge is labeled according to the sensor observation that is received when the body crosses the shared beam from r_1 to r_2. A self-loop in G is made if it is possible to cross a beam and remain in the same region. Figure 7 provides a simple example. Figure 8 shows G for the example in Figure 3(b). Once again, G can be computed from well-known decomposition algorithms [de Berg et al. 2000]; even thousands of beams would present little computational challenge.

The region filter (2) is implemented over G in a way similar to the simulated operation of a nondeterministic finite automaton. Let V denote the set of all vertices of G, which corresponds bijectively to the set of all regions. Let $m_k : V \rightarrow \{0, 1\}$ be a function that is computed as each stage k. Let $m_k(v) = 1$ mean that the body might be in the region that corresponds to v, and let $m_k(v) = 0$ mean that the body is certainly not in that region. Initially, m_0 is computed by assigning $m_0(v) = 1$ to each vertex that corresponds to a region in R_0. For all others, $m_0(v) = 0$.

For the incremental operation of the filter, assume that R_k has been already calculated from $\tilde{y}_k$. This means that $m_k(v) = 1$ for each corresponding region in R_k and no others. Suppose that y_{k+1} is observed, which extends the sensor word by one observation. Initially, assign $m_{k+1}(v) = 0$ for all $v \in V$. For each vertex $v \in V$ for which $m_k = 1$, assign $m_{k+1}(v) = 1$ for every outgoing edge that has y_{k+1} as its label. After this is completed, the set of all regions for which $m_{k+1}(v) = 1$ represents R_{k+1}, which is the desired result (2). Note that R_{k+1} may be larger than R_k because multiple outgoing edges may match y_{k+1} at each vertex. Also, this approach works for the case of partially distinguishable beams because the match is based on the observed beam label y_{k+1}, rather than the particular beam.

Now suppose that after computing R_k, we would like to know the possible *sequences* of regions traversed by the body. Construct a $(k+1)$-partite graph H in which the ith set of vertices in the partition is denoted by V_i (there are no edges between vertices in V_i). Each V_i corresponds bijectively to the regions in R_i. The edges in H are formed directly from the computations of the region filter. For every $r' \in R_{i+1}$, if for $r \in R_i$, there is an edge in G from r to r' and it is labeled with y_i, then an edge from the corresponding vertex is made in H. This edge is formed from the vertex in V_i corresponding to r to the vertex in V_{i+1} corresponding to r'. Once H has been computed, a compact encoding of the set of all possible region sequences given $\tilde{y}_k$ is obtained: It is the set of all paths in H from any vertex in V_0 to any vertex in V_k. Note that from some $v \in V_0$ a path might not even exist to any $v' \in V_k$ because it was learned from later observations that the body must not have initially been in the region corresponding to v. In other words, observations gained at later stages can refine our knowledge about what might have occurred several stages earlier.

Note that the sequences of possible regions provides a tight characterization of the set of possible paths given $\tilde{y}_k$: It is the set of all paths that traverse any one of the computed possible region sequences. Any paths that traverse a region sequence not listed as a possible sequence must not be included because it would have yielded a different sensor word.

The region filter runs in time $O(|V|+|E|)$ for each update from k to $k+1$, in which $|V|$ and $|E|$ are the numbers of vertices and edges in G, respectively. The bipartite graph H is extended with no additional overhead. Note that the number of computations grows with the number of vertices for which $m_k(v) = 1$, which reflects the amount of uncertainty about the current region. In some cases the number of marked vertices cannot increase, as in the case of DDD beams.

To illustrate the region filter, we present simulations for the concrete scenario of beams coming from crossings of landmarks. We computed two cases. In the first one, all beams are distinguishable and directed (see Figure 9). In the second, the only information available at a crossing is the corresponding pair of landmarks. Therefore,

the beams are not directed, and each beam label can be reported by exactly two beams (see Figure 10). In these examples, a beam is identified with the pair of landmarks that produce it.

4.2. Multiple Bodies

The formulation in Section 2 can be naturally extended by allowing more than one body to move in X. In this case, suppose that the sensor beams cannot distinguish between bodies. They simply indicate the beam label whenever crossed. Furthermore, assume that bodies never cross beams simultaneously. The task is to reconstruct as much information as possible about what path they might have taken.

Figure 11(a) shows a simple example of this, in which there is one obstacle, two bodies, and three undirected beams. This yields a set of three regions: $R = \{r_1, r_2, r_3\}$. Using (r, r') to denote the region that contains the first and second bodies, respectively, there are nine combinations of region assignments: (r_1, r_1), (r_1, r_2), (r_1, r_3), (r_2, r_1), (r_2, r_2), (r_2, r_3), (r_3, r_1), (r_3, r_2), and (r_3, r_3). Consider $\mathcal{I} = \mathrm{pow}(R \times R)$, which is the set of all subsets of possible region assignments. A region filter can be made over this set in the form

$$\iota_{k+1} = \phi(\iota_k, y_{k+1}), \tag{3}$$

in which ι_k represents the set of all possible region assignments after $\tilde{y}_k$ has been observed. Initially, some $\iota_0 \in \mathcal{I}$ is given. After each observation is received, $\iota_{k+1} \in \mathcal{I}$ is computed from $\mathcal{I}_k$ and y_{k+1}. The method of Section 4.1 can be easily extended to compute (3). Let G^2 be the multigraph formed by taking the Cartesian product $G \times G$ in the sense that the vertices correspond to all ordered pairs of regions. Each edge in G^2 is formed if a transition from one ordered pair to another is possible after a single observation y_{k+1}. Once G^2 is formed, the method calculating m_k from Section 4 can be easily extended.

If there are n bodies in the environment, then the method can be extended by forming an n-fold Cartesian product of R to obtain $\mathcal{I}$ and a n-fold product of G to obtain G^n. Although conceptually simple, the number of region assignments grows exponentially in the number of bodies, and the power set needed to obtain $\mathcal{I}$ is exponentially larger. Therefore, an important direction of future research on region filters is to identify ways to reduce complexity from the task description. For example, consider the following question for the example in Figure 11(a): Are the two bodies together in a region, or are they separated by a beam? Consider designing the simplest region filter that correctly answers this question.

Figure 11(b) shows a surprisingly simple combinatorial filter that answers the question for any sensor word and uses only four information states. The T information state means they are together in some room. Each D_x information state means they are in different rooms, with beam x separating them. The set of all information states is $\mathcal{I} = \{T, D_a, D_b, D_c\}$, and a filter of the form (3) in nicely obtained. With only two bits of memory, arbitrarily long sensor words can be digested to produce the answer to the question, in constant time during each iteration. It is required, however, to know the initial information state in $\mathcal{I}$.

Under what other conditions can such dramatic reductions be made? The example in Figure 11(b) can be extended to more regions and bodies by asking the question of whether each region contains an odd or even number of bodies. In this case, a filter can be made that records only one bit per region (the parity). It is challenging, however, to find more useful settings in which simple region filters exist for multiple bodies.

One important point to note about the problem in Figure 11 is that it did not distinguish between the bodies. If there are n bodies and the task does not require distinguishing between then, the number of region assignments is reduced from m^n to

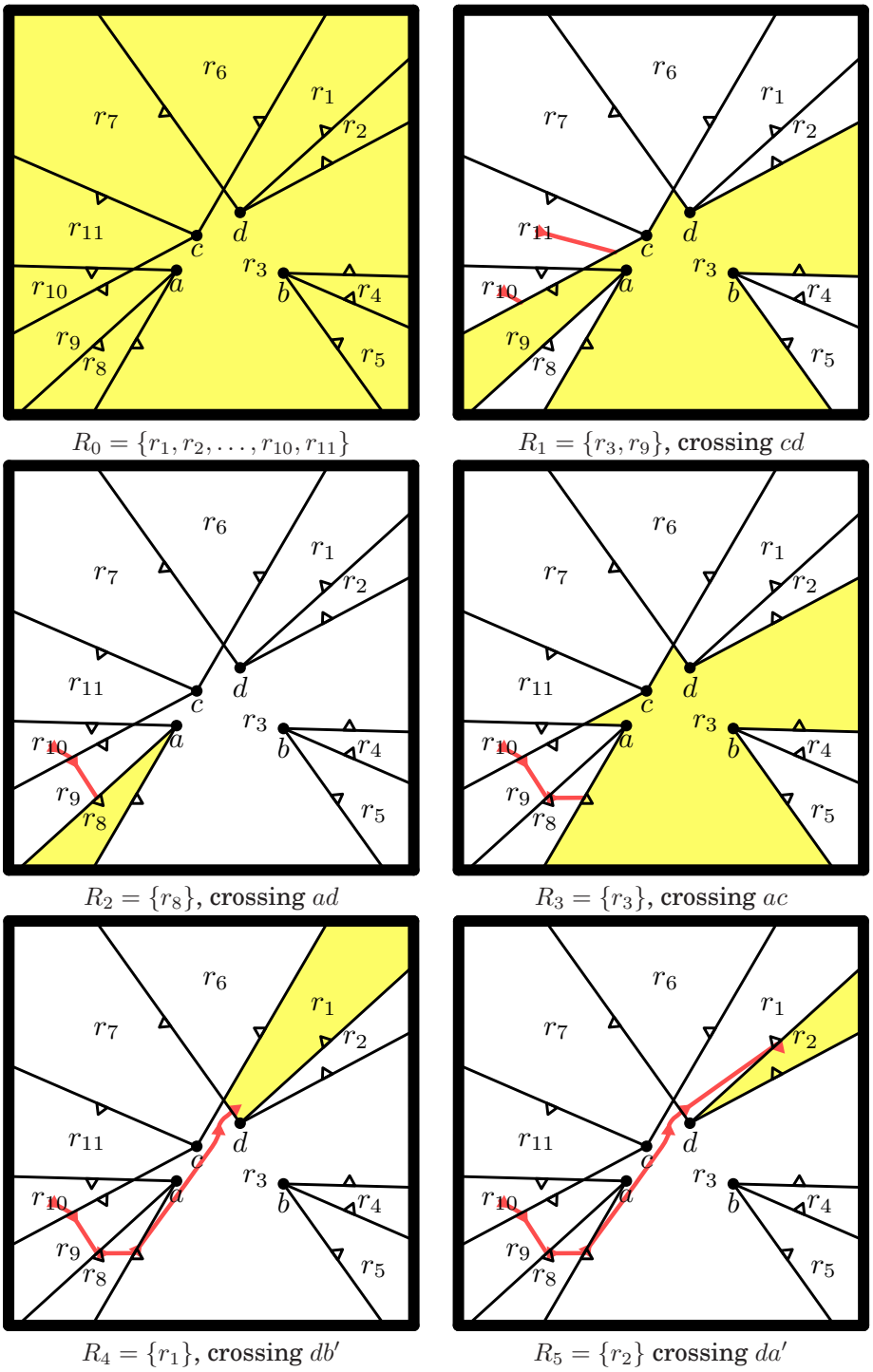


Fig. 9. In this region filter simulation the beams are formed by the crossings of landmarks in the field of view, resulting in directed and distinguishable beams. The obstacles are small points. A reconstructed sample path is shown in red.

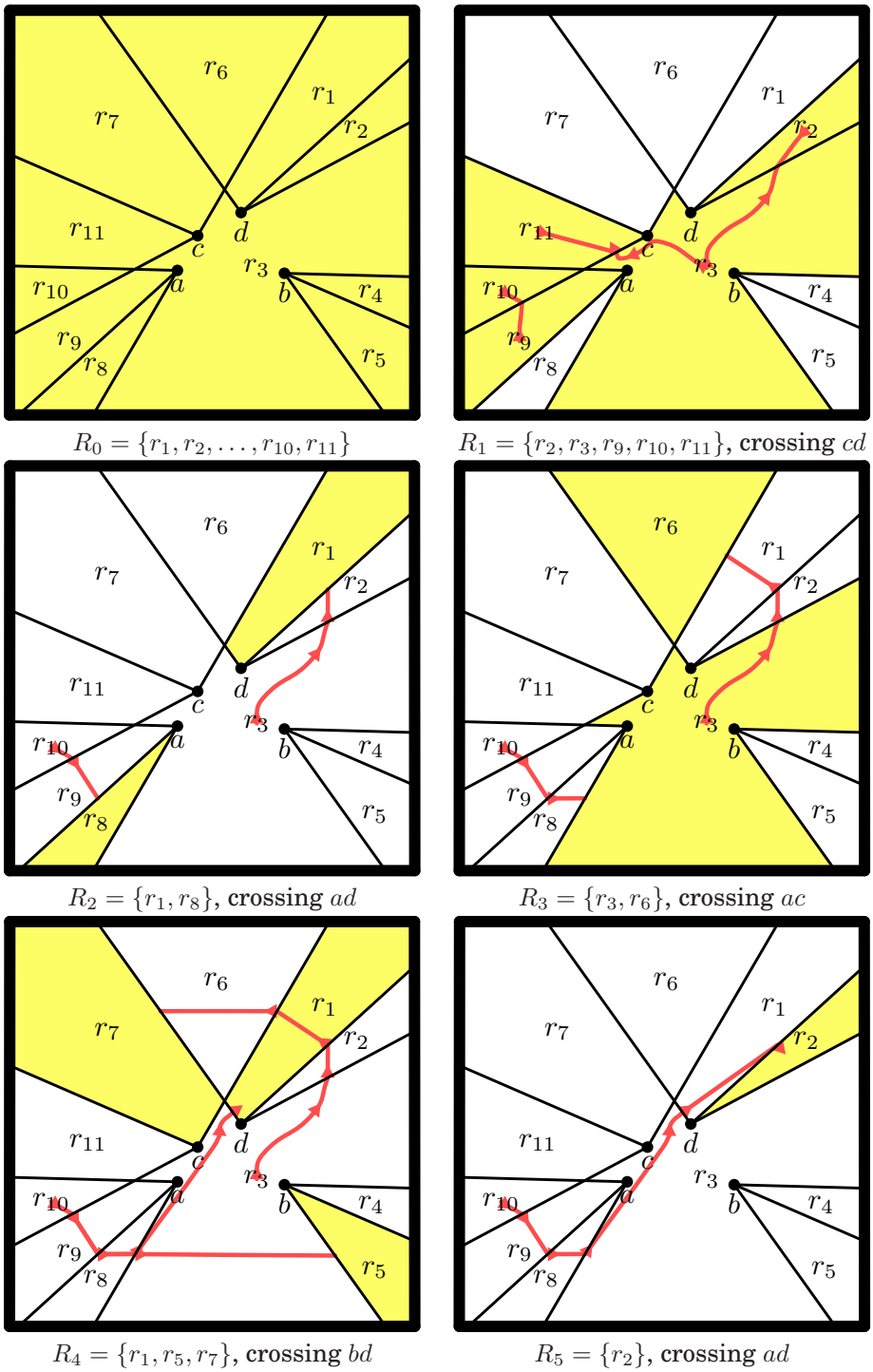


Fig. 10. This region filter simulation corresponds to the case of crossings of landmarks that results in undirected, partially distinguishable beams. Each beam label can be reported by exactly two beams.

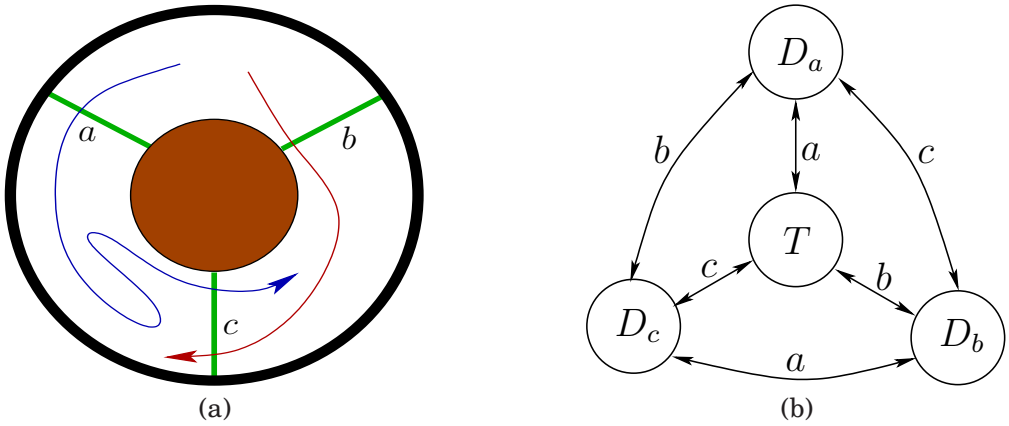


Fig. 11. (a) A three-region problem with two bodies; (b) a tiny combinatorial filter that determines whether the bodies are together in a room.

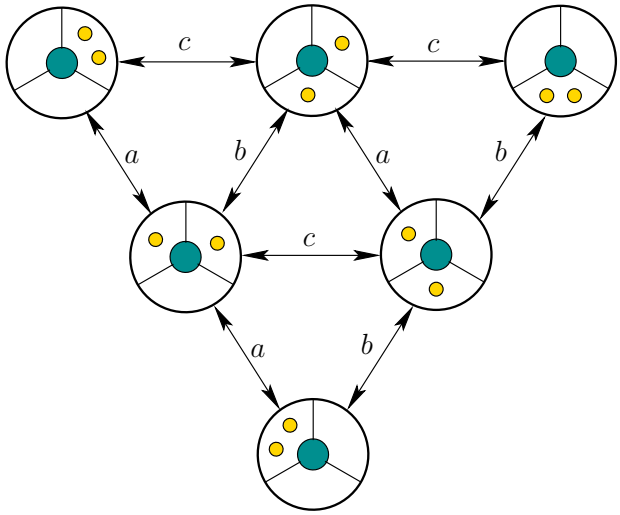


Fig. 12. Keeping track of only the number of bodies per region, rather than the particular region assignments, dramatically reduces the complexity. In this simple example, there are only two bodies and three regions. Rather than obtain $2^3 = 9$ possible region assignments, only $\binom{3}{2} = 6$ are needed if we only care about the number per region.

$\binom{m+n-1}{n}$, in which there are m regions. Figure 12 shows a simple example; only the number of bodies per region matters. It is the classical *balls and urns* problem from combinatorics. Only the number of bodies per region needs to be maintained in the filter.

5. RECONSTRUCTION UP TO HOMOTOPY

In this section, the task is to use the sensor word $\tilde{y}$ to reconstruct a description of possible paths $\tilde{x} \in \tilde{X}$ up to an equivalence class of homotopic paths. The required topological definitions are given in the Appendix.

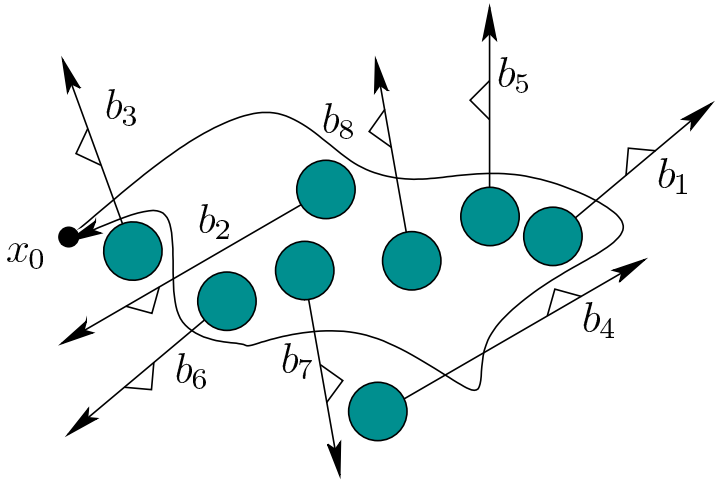


Fig. 13. A perfect collection of beams and a loop path.

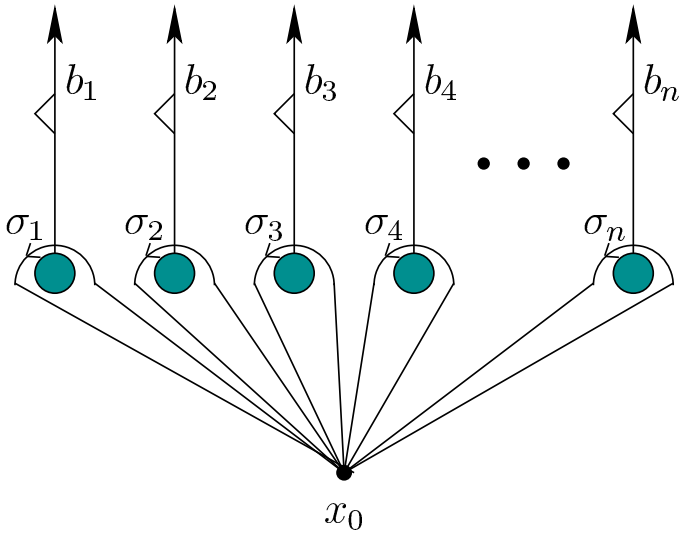


Fig. 14. With this collection of obstacles and beams, the sensor word $\tilde{y}$ directly transforms into an element of the fundamental group F_n by renaming symbols.

5.1. Perfect Beams

We now handle the case that is conceptually as simple as Figure 23. More complicated cases are built upon it. Let a beam be called *outer* if it is either an infinite ray (possible only if X is unbounded) or it is a finite segment that connects an obstacle to the boundary of W. For a set of n obstacles, let a *perfect* collection of beams mean that there are exactly $m = n$ DDD beams (recall that this means *disjoint, distinguishable, and directed beams*), with exactly one outer beam attached to each obstacle. For convenience, further assume that all beams in a perfect collection are oriented so that a counterclockwise traversal corresponds to the positive direction, as shown in Figure 13. If they are not, then we can easily transform our solution to reverse the corresponding directions.

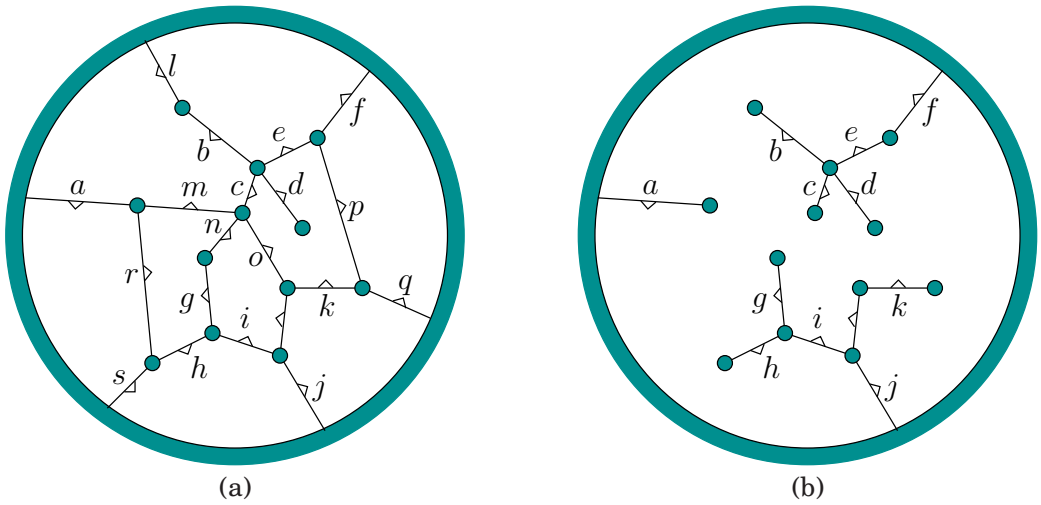


Fig. 15. (a) A sufficient collection of DDD beams; (b) a minimally sufficient collection of DDD beams forms trees that each contain one outer beam. This example is obtained by removing beams from the the figure on the left.

Let F_n denote the free group over an n-letter alphabet Σ. Suppose that a perfect collection of beams is given with label set L. Let $\beta : L \rightarrow \Sigma$ denote any bijection, in which Σ denotes an alphabet as used in the definition of F_n. We obtain the following proposition.

Proposition 5.1 *For any sensor word $\tilde{y}$ for a perfect collection of beams, if the transformation β is applied to every element, the resulting transformed word is the corresponding element of F_n, under the basis of one counterclockwise loop per obstacle (as depicted in Figure 14).*

Proof: While preserving homotopy equivalences, the obstacles and basepoint can be moved into a canonical form as shown in Figure 14. This corresponds to choosing a particular basis of F_n in which each generator σ_i is exactly a counterclockwise loop around one obstacle. It does not matter what loop in particular is chosen, provided that exactly one obstacle is encircled for each generator. Each b_i corresponds to a beam and letter in L. A word $w \in F_n$ is formed as follows. Let y_k denote the kth observation in $\tilde{y}$. The kth element of w is defined as $\beta(y_k)$. Each beam crossing then corresponds directly to a generator of F_n under the chosen basis. This converts every $\tilde{y}$ into an element of F_n that represents the loop path that was traversed using basepoint x_0. $\square$

Recall that arbitrary words in F_n can be simplified to reduced words by applying the group axioms. Note that this simplification can even be performed directly on the sensor word, before the transformation into F_n, without changing the result. For example, $b_1 b_2 b_2' b_3$ can be simplified to $b_1 b_3$ because $\sigma_2 \sigma_2^{-1} = \varepsilon$ would cause a cancellation anyway after the transformation β is applied.

5.2. Sufficient DDD Beams

What if the collection of beams is not perfect? For some arrangements of obstacles, it might not even be possible to design a perfect collection (unless beams are allowed to be nonlinear). Suppose that a collection $\mathcal{B}$ of $m \geq n$ DDD beams is given. A collection of beams called *sufficient* (regardless of whether they are DDD) if all of the resulting

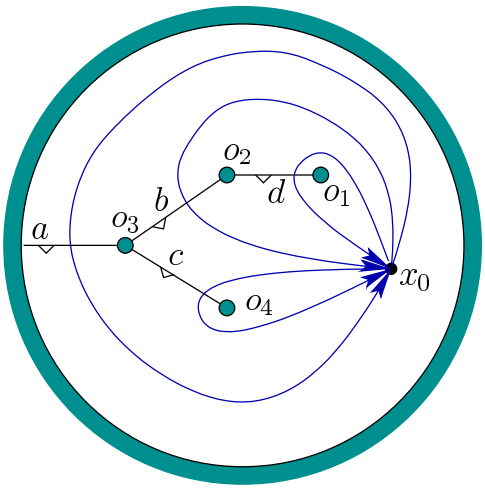


Fig. 16. Forming a basis using a sufficient tree of beams.

regions are simply connected; see Figure 15(a). Note that any sufficient collection must contain at least one outer beam. Also, any perfect collection is also sufficient.

Recall the previous question regarding whether two paths must be homotopic if the region sequence they traverse is the same. We obtain the following proposition:

Proposition 5.2 *If the collection of beams is sufficient and two paths $\tilde{x}$ and $\tilde{x}'$ traverse the same sequence of regions, then they must be homotopic.*

Proof: For a sufficient collection, they must be homotopic because if they are not, then it implies that there exists an obstacle $o \in \mathcal{O}$ that blocks the deformation of one path into another. This would be possible only if o lies in the interior of a region; however, o must border two or more regions because in a sufficient collection, a beam is attached to o. Therefore, the homotopy map would cause a change in the region sequence, which violates that assumption that it is the same for both paths. □

The implication of Proposition 5.2 is that path reconstruction up to homotopy is more coarse than reconstruction up to region sequences. The region filters provide the tightest possible description of the path, but homotopy equivalence allows some paths that traverse different region sequences to be declared as being "the same".

Suppose that a sensor word $\tilde{y}$ is obtained for a sufficient collection $\mathcal{B}$ of beams. The first step in describing the path as an element of F_n is to disregard redundant beams. To achieve this, let $\mathcal{B}' \subseteq \mathcal{B}$ be a minimal subset of $\mathcal{B}$ that is still sufficient. Such a collection is called *minimally sufficient* and can be computed using linear-time spanning tree algorithms such as depth-first or breadth-first search. In this case, the outer boundary and each obstacle is considered a vertex, so that connection to the outer boundary is assured. An example spanning tree is shown in Figure 15(b).

Recall the bijective transformation $\beta : L \to \Sigma$. The function can be applied to obtain elements of F_n as was done for Proposition 5.1; however, a different representation is obtained in comparison to the nice one in Figure 23. This amounts to a careful choice of basis for F_n. The relationship between this basis and the nice basis of Figure 23 is given by an element of $\mathrm{Aut}(F_n)$, which is described in the Appendix.

Proposition 5.3 *For a minimally sufficient collection $\mathcal{B}$ of DDD beams, the sensor word $\tilde{y}$ maps directly to the corresponding element of F_n by simply applying β to each letter.*

Proof: A basis for the fundamental group is defined as follows. For each tree of beams in $\mathcal{B}$, a collection of loop paths can be formed as shown in Figure 16. Each loop must cross transversely the interior of exactly one beam and enclose a unique nonempty set of obstacles. Such loops always exist and can be constructed inductively by first enclosing the leaves of the tree and then progressing through parents until a loop is obtained that traverses the outer beam. Since there is only one region, it is possible to inductively construct such a collection of loops for every tree of beams in $\mathcal{B}$. The total collection of loops forms a basis of F_n, which can be related to the basis in Proposition 5.1 via Tietze or Nielsen transformations [Magnus et al. 1976]. The mapping β from $\tilde{y}$ to $\beta(\tilde{y}) \in F_n$ is once again obtained by mapping each letter in $\tilde{y}$ to its corresponding unique loop that traverses the beam. □

Using Proposition 5.3, a simple algorithm is obtained. Suppose that any sufficient collection $\mathcal{B}$ of DDD beams is given and a sensor word $\tilde{y}$ is obtained. A spanning tree $\mathcal{B}' \subseteq \mathcal{B}$ of beams is computed, which is minimally sufficient. Let $L' \subseteq L$ denote the corresponding set of beam labels. To compute the element of F_n, the first step is to delete from $\tilde{y}$ any letters in $L \setminus L'$. This yields a reduced word $\tilde{y}'$ for which each letter can be mapped directly to a loop using the proof of Proposition 5.3 to obtain a representation of the corresponding path in F_n. Once again, reductions based on the identity and inverses in F_n can be performed before or after the mapping is applied.

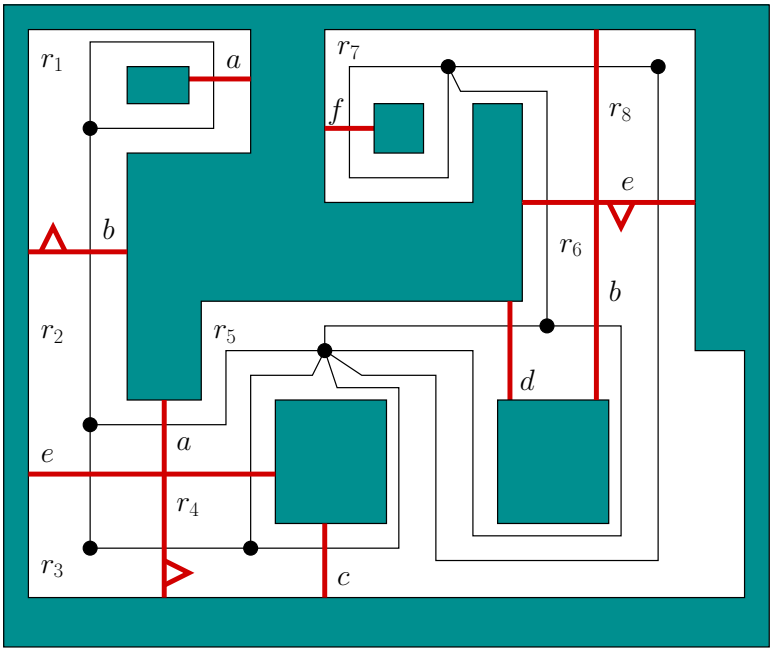
5.3. The General Case

We finally return to most general collection of beams, which includes examples such as Figure 3(b). Consider a collection $\mathcal{B}$ of beams in which some may intersect, some may be undirected, and some may even be indistinguishable. The collection is nevertheless assumed to be *sufficient*, which means that all of the corresponding regions are simply connected. Rather than worry about making a minimal subset of $\mathcal{B}$, the method for the general case works by inventing a collection of fictitious beams that happens to be minimally sufficient. Since the ambiguity may be high enough to yield a set of possible paths, the region filter from Section 4 is used.

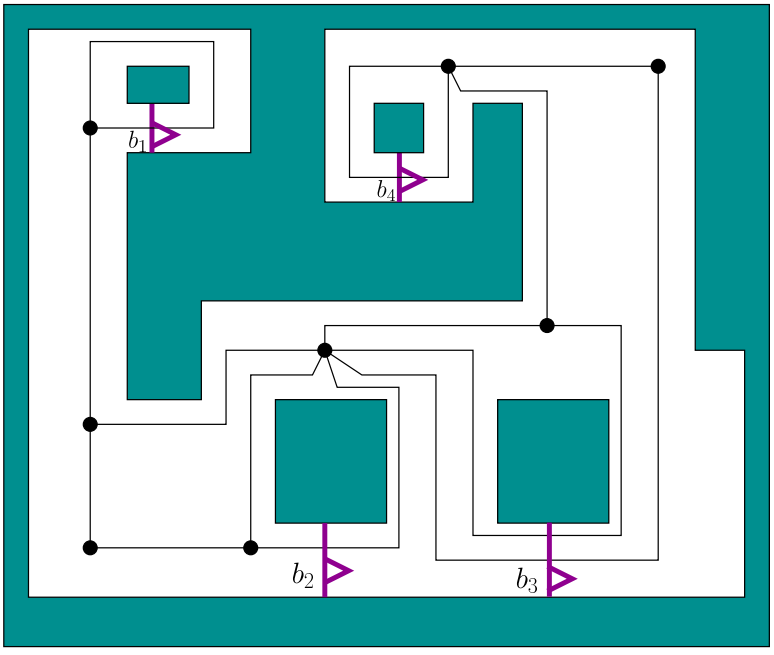
There are two phases to the computation. In the first phase, a set of "imaginary" DDD beams is constructed from the original collection of obstacles and beams. In the second phase, the sensor word $\tilde{y}$ is processed and a representation of possible path classes is given in terms of imaginary beams.

For the first phase, suppose W, $\mathcal{O}$, $\mathcal{B}$, L, and α (beam labels) are given in a way that once again facilitates exact computation (as Section 4.1). The algorithm proceeds as follows:

(1) Compute the arrangement of regions and multigraph G from Section 4.
(2) For each vertex in G choose a *sample point* in its corresponding region.
(3) For each directed edge e in G, compute a piecewise-linear *sample path* that: i) starts at the sample point of the source vertex of e, ii) ends at the sample point of the destination vertex of e, and iii) crosses the beam associated with e in a manner consistent with its label. The sample path must be chosen so that it does not intersect additional beams. The sample paths can be computed using standard motion planning techniques, such as trapezoidal decomposition or triangulations (see [LaValle 2006]). The result is an embedding of G into the free space X, as depicted in Figure 17(a).



(a)



(b)

Fig. 17. (a) This shows the embedding of the multigraph G into the free space X for the example of Figure 3(b). (b) A minimally sufficient collection of DDD beams forms trees that each contain one outer beam. This example is obtained by removing beams from the figure on the left.

(4) Construct any minimally sufficient collection $\mathcal{B}_I$ of *imaginary* DDD beams. The choice does not depend at all on previous steps. A convenient choice is to make all imaginary beams vertical, one from each obstacle. See Figure 17(b).
(5) For all computed sample paths from Step 3, compute their intersections with the imaginary beams of Step 4 and record the order in which they occur.

Now suppose that a sensor word $\tilde{y}$ is given. The following steps construct the possible paths up to homotopy equivalence:

(1) Apply the region filter of Section 4 is used to determine the set of possible region sequences.
(2) Each region sequence corresponds to a cyclic walk through G. Using the embedding of G into W from the first phase of computation (recall Figure 17(b)), a loop path $\tilde{x}'$ is obtained by concatenating the corresponding sample points and sample paths in G.
(3) Using the construction in the proof of Proposition 5.3, $\tilde{y}'$ is mapped directly to an element of F_n.
(4) The obtained word in F_n is simplified to obtain the reduced word. As before, simplifications can be applied to $\tilde{y}'$ in advance or to its image in F_n and the same result is obtained.
(5) In the final step, duplicate reduced words are removed from the collection obtained for each sequence produced by the region filter.
As usual, reductions can be applied to $\tilde{y}'$ or its image in F_n. Once elements of F_n are computed and reduced for each possible region sequence, duplicates are removed to obtain the complete set of possible homotopically distinct paths based on the sensor word $\tilde{y}$.

Note that $\mathcal{B}_I$ essentially gives the user the freedom to define whatever basis of F_n is desired to express the result. By extending the method to nonlinear beams, the solution can even be expressed in terms of perfect beams, instead of the unusual loop paths produced by the method of sufficient DDD beams.

Proposition 5.4 *The given algorithm reconstructs from $\tilde{y}$ the complete set, up to homotopy equivalence, of paths that could have possibly produced $\tilde{y}$ in terms of a common basis for F_n.*

Proof: The sensor word $\tilde{y}$ is given. By applying the region filter from Section 4, all possible sequences of regions traversed by the path are obtained. Constructing the walk through G and converting it into a sample path provides one representative from the correct homotopy class. This is true because the sample path was constructed by traversing the same region sequence and all paths that traverse the same region sequence are homotopic, as established by Proposition 5.2. It is now simple a matter of picking a basis for F_n to describe the homotopy class. The collection of imaginary beams is a sufficient DDD collection. Proposition 5.3 therefore implies that the corresponding element of F_n is obtained by the direct mapping β. This establishes that the computed representation is indeed the correct homotopy class that corresponds to the region sequence. Since the class is correctly characterized for every possible region sequence, the complete set of paths is characterized up to homotopy equivalence, as required. Furthermore, all classes are described with respect to the same basis of F_n because the imaginary beams are fixed in advance and all homotopy classes are expressed in terms loops that encompass them. □

It remains an open problem to characterize the complexity of the set of possible homotopy classes in terms of the sensor word.

We now remove the assumption that only loop paths are executed using a basepoint x_0. If all paths start at some x_0 and terminate at some x_1, then a fixed path segment that connects x_1 back to x_0 can be chosen. This path may intersect some beams, which is false information; however, the possible body paths are nevertheless characterized correctly up to homotopy equivalence. If instead we allow either of the path endpoints to vary, then all paths become trivial (homotopic to a constant path) by continuously deforming each path to be a constant function into one basepoint. One possibility is to assume that there are several possible fixed points based on the starting and final regions produced in each sequence from the region filter. In this way, possible paths can at least be compared if their starting and terminating regions match.

6. PATH WINDING NUMBERS

The section provides a reconstruction of the path at a level that is more coarse than homotopic equivalence. Suppose that the body travels along a loop path. There is an integer *winding number* $v_i \in \mathbb{Z}$ for each $o_i \in \mathcal{O}$, in which m_i is defined as the number of times the path wraps counterclockwise around o_i after deleting all other obstacles and pulling the path tight around o_i using homotopy. If there are n obstacles, then a vector of n winding numbers is obtained. Two paths are called *homologous* if and only if their vectors of winding numbers are identical. This notion of equivalence is crucial to algebraic topology, and in particular homology theory [Hatcher 2002], in which a *homology group* is computed. The *first homology group* $H_1(X)$ can be considered as the "abelianized" version of the fundamental group $\pi_1(X)$. For our problem, $H_1(X)$ is obtained by applying the equivalence relation $\sigma_1\sigma_2 = \sigma_2\sigma_1$ over all words in F_n, for all $\sigma_1, \sigma_2 \in \Sigma$.

6.1. Perfect Beams

In the case of perfect beams, the winding numbers are obtained by directly "abelianizing" the sensor word $\tilde{y}$. Using Proposition 5.1, the sensor word maps directly to a word F_n by applying β to every letter. For example, consider the sensor word:

$$\tilde{y} = b_1 b_2 b_1' b_2 b_2 b_1 b_2' b_2' b_1 b_2' b_2' b_2'. \tag{4}$$

After applying β to each symbol, the word

$$w = \sigma_1\sigma_2\sigma_1^{-1}\sigma_2\sigma_2\sigma_1\sigma_2^{-1}\sigma_2^{-1}\sigma_1\sigma_2^{-1}\sigma_2^{-1}\sigma_2^{-1} \in F_n \tag{5}$$

is obtained. By applying the commutativity relation, we simply sort the symbols in the word, perform cancellations, and count the resulting number of each. For the example, the result is

$$\begin{aligned} w =& \sigma_1\sigma_2\sigma_1^{-1}\sigma_2\sigma_2\sigma_1\sigma_2^{-1}\sigma_2^{-1}\sigma_1\sigma_2^{-1}\sigma_2^{-1}\sigma_2^{-1} \\ =& \sigma_1\sigma_1\sigma_1\sigma_1^{-1}\sigma_2\sigma_2\sigma_2^{-1}\sigma_2^{-1}\sigma_2^{-1}\sigma_2^{-1}\sigma_2^{-1} \\ =& \sigma_1^2\sigma_2^{-3}. \end{aligned} \tag{6}$$

The winding numbers are simply the exponents; for (6) we obtain $v = (2, -3)$. The application of β was unnecessary in this case to obtain the result. The operations can be performed directly on $\tilde{y}$ due to the simplicity of β. Note that the winding numbers can be computed in time $O(|\tilde{y}|)$ without actually sorting by simply maintaining n counters, one for each letter in $\lambda \in L$. Scan across $\tilde{y}$ and increment or decrement each counter, based on whether λ or λ', is encountered, respectively. Note that this makes a constant-time combinatorial filter of the form $v_{k+1} = \phi(v_k, y_{k+1})$, by computing the winding numbers v_{k+1} at step $k+1$ from the winding number vector v_k and the last observation y_{k+1}, which is the last letter of $\tilde{y}_{k+1}$.

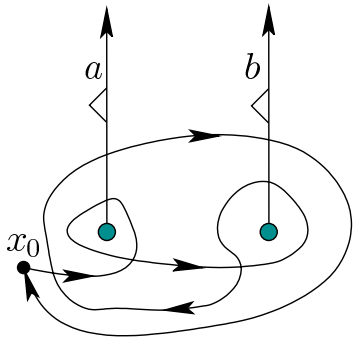


Fig. 18. A simple commutator example that yields sensor word $aba'b'$, group element $\sigma_1\sigma_2\sigma_1^{-1}\sigma_2^{-1} \in F_2$, and winding numbers $v = (0, 0)$, but corresponds to path that is not homotopic to a constant path.

6.2. Sufficient DDD Beams

Once a minimally sufficient collection is determined (recall Figure 15), the winding numbers can be calculated in the same way as in the perfect beams case. However, the result needs to be transformed to obtain the correct result because the path may wind around multiple obstacles simultaneously. Recall the basis from Figure 16 and suppose that for a path, the sensor word is

$$\tilde{y} = aabcbd'cbdbd'ab'b'a'cb'acaab'b'b'd'. \tag{7}$$

Applying the simple method from the perfect beams case suggests winding numbers $(5, -3, 4, 1)$. For example, there are 5 more as than a's in $\tilde{y}$. This means that the path wraps 5 times around o_3, but it also wraps 5 times around o_1, o_2, and o_4. Likewise, it wraps -3 times around o_1 and o_2. A counter is made for each obstacle and each computed exponent raises or lowers some counters. After being performed for each component, the correct result is obtained. For the sensor word in (7), the corrected winding numbers are $(3, 2, 5, 9)$. Note that if the positive direction of a beam is in the clockwise direction, then the computed winding number needs to be multiplied by -1.

6.3. The General Case

Now suppose that a sufficient collection of general beams has been given, which is the model used in Section 5.3. A straightforward approach is to first run the algorithm of Section 5.3. After the sufficient collection of imaginary beams has been placed and the elements of F_n have been computed, each can be abelianized to obtain winding numbers using the method just described. This yields a set of vectors of winding numbers because there might not be enough sensor data to infer the exact numbers.

In some cases, this approach computes more information than is needed to obtain the winding numbers. For example, suppose that a sufficient collection of beams is given that is not necessarily disjoint, but all beams are directed and distinguishable. Since the winding number essentially ignores all other beams, an approach can be developed by picking a minimally sufficient collection of beams that is not necessarily disjoint. The beam intersections do not interfere with the calculation of winding numbers. For a given sensor word, any letters that do not appear in the minimally sufficient collection can simply be deleted. The method then proceeds as in the case of DDD beams.

6.4. Higher Order Winding Numbers

There is actually a way to provide path descriptions that are more coarse than homotopy equivalence but are finer than homology equivalence. The winding numbers give a measure of how many times a body circles around a given obstacle, but ignores

how the body weaves in between obstacles. They are insensitive to paths such as a commutator around obstacles, as shown in Figure 18.

There are "higher order winding numbers" which keep track of how a body does in fact interweave through different obstacles. They count the cumulative number of times that a commutator was achieved. It is also possible, however, to count commutators that are build from other commutators. The notion of higher order arises from the levels of nesting of commutators. These structures can be captured by a Lie algebra in which the Lie bracket is the group commutator: $[\sigma_1, \sigma_2] = \sigma_1\sigma_2\sigma_1^{-1}\sigma_2^{-1}$. These higher order winding numbers in the case above arise in two classical ways reflecting the interplay between geometry and a free group. These are the Lie algebras which arise from either (i) the descending central series of a free group, or (ii) principal congruence subgroups of level p^r in the group of $SL(2, \mathbb{Z})$. These Lie algebras provide measures of complexity in addition to "higher order winding numbers" and it remains an open problem to develop computation methods that characterize them for a given sensor word. It remains an open problem to efficiently compute higher order winding numbers for problems presented in this paper.

7. EXPERIMENTAL IMPLEMENTATION

In this section, we present an inexpensive hardware architecture that implements some simple, reliable, low-cost beams, both directed and undirected. We conducted several experiments that involved reconstructing the path of moving bodies in a laboratory setting by applying the region filters from Section 4. Since the method was validated for these filters, they would also clearly work for the methods of Sections 5 and 6 because they are derived from the same sensor observations. Many more details regarding our hardware choices, their costs, and our experiments appear in [Czarnowski 2011].

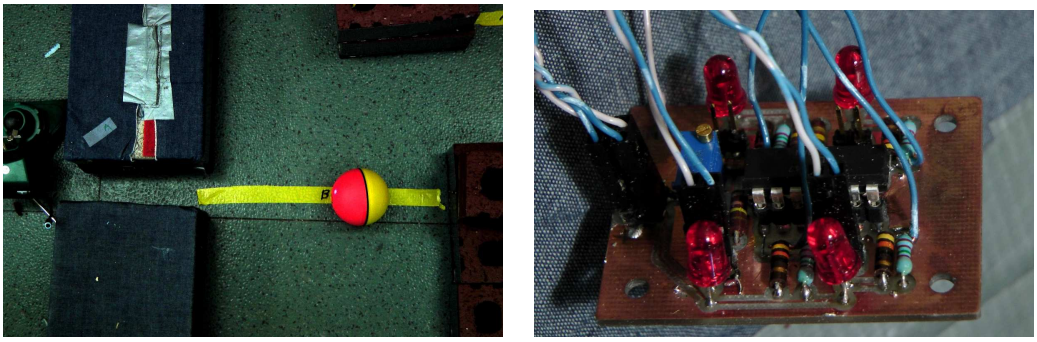
7.1. Hardware Architecture

We implemented the beam sensor using optical emitter-detectors pairs. One pair was used to make an undirected beam. To make a directed beam, two pairs were placed close together using the idea in Figure 4, and some simple logic circuitry was used to determine the crossing direction. We chose presentation laser pointers due to their low cost (about $3 US each) and because they can be aimed easily. The pointers were modified to use external battery packs using three AA batteries. Inexpensive and easily obtainable photodiodes (about $2 US each) were placed on the opposite side to detect body crossings. A change in voltage across the photodiode is observed when a body crosses the beam, thereby blocking the laser light from reaching the photodetector. This change in voltage is detected by a basic ADC circuit using the LM339 comparator (about $0.20 US each). The threshold voltage for the beams can be set using a potentiometer. For the purposes of this experiment, simple circuit boards were fabricated to accommodate the ADC circuit (Figure 19 (b)). Each board can handle four inputs from the outputs of the photodetectors.

The outputs of the ADC board are connected to the digital I/O pins of a Complex Programmable Logic Device (CPLD). The CPLD is a programmable logic device that takes care of debouncing and denoising the inputs. It then outputs an ASCII character corresponding to the beam label over a serial port. The CPLD was designed using the Verilog hardware description language.

There are several reasons why we chose a CPLD for use in our system:

(1) Price: The least expensive Altera MAX IIZ CPLD costs less than $7 US.
(2) Energy Consumption: The Altera MAX IIZ CPLD can run on as little as 25μA.



(a) (b)

Fig. 19. (a) An emitter-detector pair with an body crossing the beam; (b) a simple Analog to Digital Conversion board using the ubiquitous LM339 comparator.

(3) Reconfigurability: The circuit implemented on a CPLD can easily be reconfigured in-circuit by changing the Verilog code and reprogramming the device with a PC.

Additionally, the CPLD design software reports the hardware resources (including logic elements) that are used for a given design. Thus, it is possible to quickly estimate the hardware cost of a mass-produced (ASIC-based) product. For example, the CPLD usage for a six-beam implementation uses 78 logic elements and 21 total pins. Thus, this design easily fits in the sub-$7 device mentioned above, which has 240 logic elements and 54 IO pins. Detailed analysis of the resource usage suggests that a design incorporating at least 20 beams can fit in even this small CPLD. If a larger system is desired, the Verilog design can be seamlessly migrated to a larger CPLD.

The cost of a six-beam deployment is under $30 US. Once properly set up, the beams do not miss any body crossings. Furthermore, our system has low energy consumption. When using three AA alkaline batteries, the current draw was measured to be 21.7 mA. Using a standard AA alkaline with 2700 mAh capacity, a beam can be powered for over 120 hours. The CPLD board can be powered with a simple rechargeable battery pack or through an USB port. Using Altera's energy consumption estimator, the Altera MAX IIZ would use 0.072 mA when using a six-beam implementation. Thus, the CPLD could theoretically run for around 37,000 hours using only a set of three alkaline batteries.

Wireless communication can be easily added to our architecture if needed. We have experimented with 2.4GHz XBee modules that implement the 802.15.4 protocol. For under $25 US, these devices allow very reliable and simple wireless communication. Each of this Zig-bee modules is capable of handling up to six beams directly using the on-board ADC circuitry.

To the best of our knowledge, our hardware implementation compares favorably in terms of cost with previous implementations for tracking using binary sensors [Kim et al. 2005; Shrivastava et al. 2006a]. First of all, we used cheaper sensors than Passive Infrared Sensors (PIR) or acoustic sensors. Furthermore, instead of using a full sensor mote that may excessive for simple computations, we determined the precise computational requirements for solving our task and implemented them in hardware. This leads to an overall decrease in price and energy consumption.

7.2. Reconstructing the Motion of a Single Body

We will illustrate how the architecture works with a simple example. As illustrated in Figure 20, we deployed six sensor beams in a 1.67m by 1.3m environment. The outer

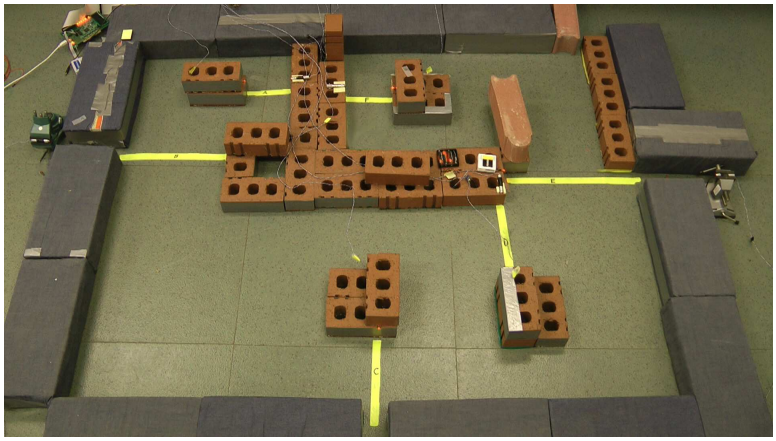


Fig. 20. A physical implementation of the six-beam example from Figure 1.

boundary of the free space is formed from cinder blocks and the obstacles made of bricks. This physical setup, shown in Figure 20, was designed to match the example in Figure 1.

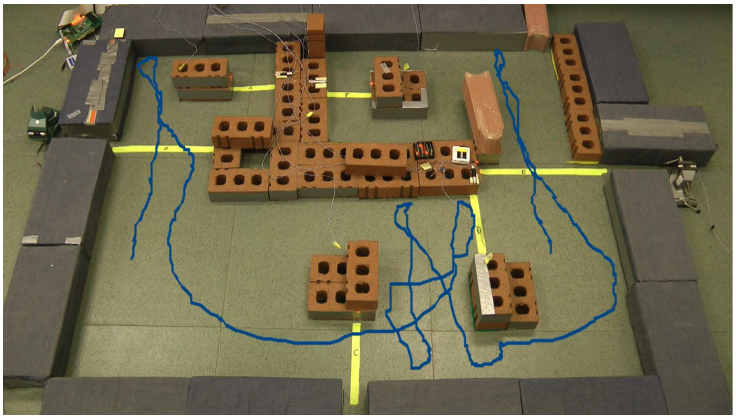
After creating the environment, we let an unpredictable rolling ball, called a Weaselball, to wander for several minutes as the moving body. (See [Bobadilla et al. 2011a] for more information on Weaselball motions.). We used an overhead camera and algorithm implementations in OpenCV to extract the ground truth path of the body (Figure 21(a)). For the sake of clarity, we show only the first several seconds of the ground truth tracking. In Figure 21 (b), we show the reconstruction of the path that was obtained from the computed region sequence. Note that the actual experiment ran for three minutes, with 62 detected beam crossings and successful reconstruction of the path up to the sequence of regions traversed by the body.

In this experiment and others, the beam did not fail to detect any of the body crossings. This was observed in dozens of experiments that were performed, with hundreds of beam crossings. The only type of mistake made was that the system reported a crossing when the body entered the beam but did not cross through it. This violates the assumption in Section 2 that the body must cross transversely. If desired, this problem can be alleviated by the use of two adjacent beams to ensure the complete crossing of the body before reporting an observation.

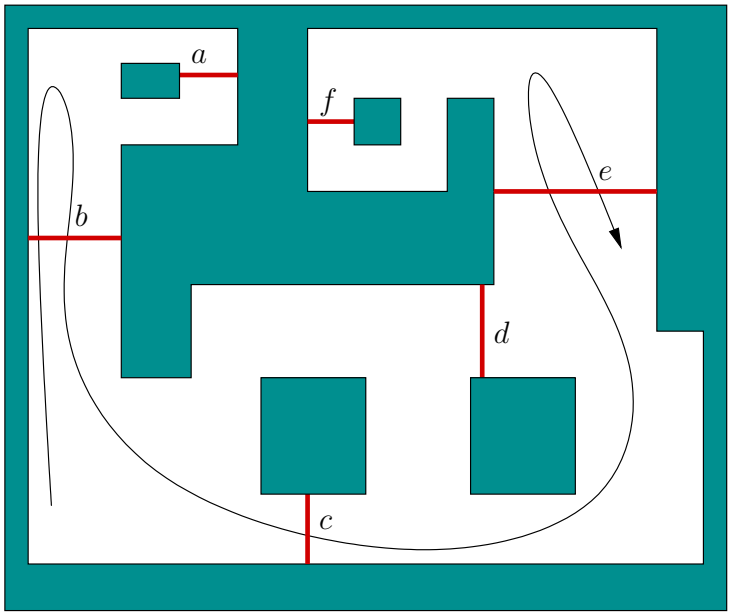
7.3. The Two-Body, Three-Region Filter

We implemented the two body problem illustrated in Figure 11(a), and the corresponding experimental version appears in Figure 22. A group of small bricks serves as the barrier (the center of the ring), whereas larger paving bricks form the outer boundary of the free space. The ring is then partitioned into three sections by our laser-detector pairs.

Recall the simple filter shown in Figure 11(b), which keeps track of whether the two bodies are in separate regions. We implemented the simple automaton on the Altera MAX II CPLD. The CPLD resource usage for the three room design is as follows: 52 logic elements, 18 total pins. Thus, the design is easily implemented on the inexpensive MAX IIZ CPLD mentioned above. The two moving balls in the environment and the filter successfully kept track of whether they were together for long periods of time. Once again, the errors occurred only when a body entered a laser beam but did not complete the crossing. The use of two emitter-detector pairs mounted close to each other should



(a)



(b)

Fig. 21. A physical implementation of the example in Figure 1: (a) The ground truth path (causing $\tilde{y} = bbcee$) of the body; (b) the reconstruction of the path over about 15 seconds of motion. A sample path is shown based on the reconstructed region sequence.

alleviate this issue. Other multibody tracking experiments appear in [Bobadilla et al. 2011b] and [Czarnowski 2011].

8. CONCLUSIONS AND OPEN QUESTIONS

In this paper we identified a basic inference problem based on bodies moving among obstacles and detection beams. Recall from Section 3 that the beams may directly model physical sensors or they may arise virtually from a variety of other sensing models. Therefore, the region filter, homotopic reconstruction, and winding-number computations provide basic information that arises in numerous settings such as robotics,

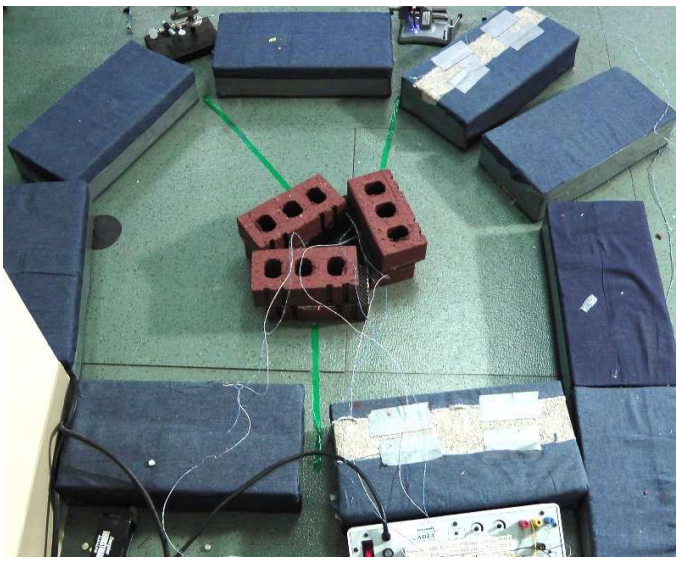


Fig. 22. A physical implementation of the environment and filter described in Figure 11. The beam locations are shown with green tape. Wires connecting to the photodiodes are visible.

security, forensics, environmental monitoring, and assisted living. Recall that the region filter provides the tightest characterization of the set of possible paths given the sensor word $\tilde{y}$. Characterization up to homotopy equivalence is more coarse than providing region sequences. Following this, the winding number characterization is even more coarse. It is possible to provide higher order winding numbers based on group commutators, yielding a level of coarseness that lies between homotopy equivalence and winding numbers.

We implemented the filters in simulation. We also developed an inexpensive, low energy consumption hardware architecture to show the practicality of the proposed approach. This hardware architecture scales well and allows the use of wireless communication.

The results presented here represent a first step in understanding this broad class of problems. Many open issues remain for future research, several of which are suggested here: 1) It is assumed that the geometric arrangement of obstacles and beams is known. What happens when this is uncertain? For example, we might not even know which beams intersect. The sensor words can be used to make simultaneous inferences about the body path and the beam arrangement. 2) What happens in the case of faulty beams? There could be false positives and false negatives in the detections. In this case, probability distributions over regions, homotopy classes, and winding numbers could be studied. It is difficult, however, to develop realistic prior distributions and error models in many settings. 3) Without the assumption of transverse beam crossings and crossings are intersection points, significantly more ambiguity arises. How do these affect the computations? 4) What are the limits of path reconstruction when there are two or more bodies? How efficient can filters be made for such problems when there are many obstacles and beams? 5) What other specific path statistics can be computed efficiently from beam data? Can Lie algebra constructions be applied to efficiently compute higher-order winding numbers (based on commutators) for the paths? Can the sensor data be used to compare paths as elements of the braid group? 6) Since the methods so far provide only inference, how can their output be used to

design motion plans? In other words, how can the output be used as a filter that provides feedback for controlling how the bodies move to achieve some task? 7) Finally, what other classes of combinatorial filters can be defined and efficiently computed? This depends on developing appropriate abstract sensor models and identifying tasks that depend on critical pieces of information provided by such sensor models. Some steps in this direction are suggested in [LaValle 2012].

Acknowledgments. The authors thank Andrew Lycas for implementing the visual tracking code, based partly on OpenCV, for ground truth comparison. They also thank Justin Kopinsky for suggesting the parity-based extension of the example in Figure 11(b).

REFERENCES

BHATTACHARYA, S., LIKHACHEV, M., AND KUMAR, V. 2011. Identification and representation of homotopy classes of trajectories for search-based path planning in 3D. In *Proceedings of Robotics: Science and Systems*.

BOBADILLA, L., SANCHEZ, O., CZARNOWSKI, J., GOSSMAN, K., AND LAVALLE, S. M. 2011a. Controlling wild bodies using linear temporal logic. In *Proceedings Robotics: Science and Systems*.

BOBADILLA, L., SANCHEZ, O., CZARNOWSKI, J., AND LAVALLE, S. M. 2011b. Minimalist multiple target tracking using directional sensor beams. In *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

CABELLO, S., LIU, Y., MANTLER, A., AND SNOEYINK, J. 2002. Testing homotopy for paths in the plane. In *Proceedings of the eighteenth annual symposium on Computational geometry*. ACM, New York, NY, USA, 160–169.

CASTELLANOS, J., MONTIEL, J., NEIRA, J., AND TARDÓS, J. 1999. The SPmap: A probabilistic framework for simultaneous localization and mapping. *IEEE Transactions on Robotics & Automation 15,* 5, 948–953.

CHOSET, H. AND NAGATANI, K. 2001. Topological simultaneous localization and mapping (T-SLAM). *IEEE Transactions on Robotics & Automation 17,* 2, 125–137.

CZARNOWSKI, J. T. 2011. Minimalist hardware architectures for agent tracking and guidance. M.S. thesis, University of Illinois.

DE BERG, M., VAN KREVELD, M., OVERMARS, M., AND SCHWARZKOPF, O. 2000. *Computational Geometry: Algorithms and Applications, 2nd Ed.* Springer-Verlag, Berlin.

DEHN, M. 1987. *Papers on Group Theory and Topology*. Springer-Verlag, Berlin.

DISSANAYAKE, G., NEWMAN, P., CLARK, S., DURRANT-WHYTE, H. F., AND CSORBA, M. 2001. A solution to the simultaneous localisation and map building (SLAM) problem. *IEEE Transactions on Robotics & Automation 17,* 3, 229–241.

DUDEK, G., ROMANIK, K., AND WHITESIDES, S. 1998. Global localization: Localizing a robot with minimal travel. *SIAM Journal on Computing 27,* 2, 583–604.

DURAND, F. 1999. 3d visibility: Analytical study and applications. Ph.D. thesis, Université Grenoble I – Joseph Fourier Sciences et Géographe.

EFRAT, A., KOBOUROV, S., AND LUBIW, A. 2006. Computing homotopic shortest paths efficiently. *Computational Geometry 35,* 3, 162–172.

EPSTEIN, D. B. A., PATERSON, M. S., CAMON, G. W., HOLT, D. F., LEVY, S. V., AND THURSTON, W. P. 1992. *Word Processing in Groups*. A. K. Peters, Natick, MA.

ERDMANN, M. A. AND MASON, M. T. 1988. An exploration of sensorless manipulation. *IEEE Transactions on Robotics & Automation 4,* 4, 369–379.

GERKEY, B., THRUN, S., AND GORDON, G. 2004. Clear the building: Pursuit-evasion with teams of robots. In *Proceedings AAAI National Conference on Artificial Intelligence*.

GOLDBERG, K. Y. 1993. Orienting polygonal parts without sensors. *Algorithmica 10*, 201–225.

GRIGORIEV, D. AND SLISSENKO, A. 1998. Polytime algorithm for the shortest path in a homotopy class amidst semi-algebraic obstacles in the plane. In *Proc. ACM Symposium on Symbolic and Algebraic Computations*. 17–24.

GUIBAS, L. 2002. Sensing, tracking, and reasoning with relations. *IEEE Signal Processing Magazine 19,* 2, 73–85.

GUIBAS, L. J., LATOMBE, J.-C., LAVALLE, S. M., LIN, D., AND MOTWANI, R. 1999. Visibility-based pursuit-evasion in a polygonal environment. *International Journal of Computational Geometry and Applications 9,* 5, 471–494.

GUIBAS, L. J., MOTWANI, R., AND RAGHAVAN, P. 1995. The robot localization problem. In *Algorithmic Foundations of Robotics*, K. Goldberg, D. Halperin, J.-C. Latombe, and R. Wilson, Eds. A.K. Peters, Wellesley, MA, 269–282.

HATCHER, A. 2002. *Algebraic Topology*. Cambridge University Press, Cambridge, U.K. Available at http://www.math.cornell.edu/ hatcher/AT/ATpage.html.

HOCKING, J. G. AND YOUNG, G. S. 1988. *Topology*. Dover, New York.

KALMAN, R. 1960. A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering 82*, 35–45.

KAMEDA, T., YAMASHITA, M., AND SUZUKI, I. 2006. Online polygon search by a seven-state boundary 1-searcher. *IEEE Transactions on Robotics 22,* 3, 446–460.

KIM, S., SREENATH, K., BHATTACHARYA, S., AND KUMAR, V. 2012. Trajectory planning for systems with homotopy class constraints. in 13th international symposium on advances in robot kinematics. In *Proc. International Symposium on Advances in Robot Kinematics*.

KIM, W., MECHITOV, K., CHOI, J., AND HAM, S. 2005. On target tracking with binary proximity sensors. In *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, Piscataway, NJ, USA, 40.

KUHN, H. W. 1953. Extensive games and the problem of information. In *Contributions to the Theory of Games*, H. W. Kuhn and A. W. Tucker, Eds. Princeton University Press, Princeton, NJ, 196–216.

KUMAR, P. R. AND VARAIYA, P. 1986. *Stochastic Systems*. Prentice-Hall, Englewood Cliffs, NJ.

LAVALLE, S. M. 2006. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K. Also available at http://planning.cs.uiuc.edu/.

LAVALLE, S. M. 2012. Sensing and filtering: A tutorial based on preimages and information spaces. *Foundations and Trends in Robotics*. To appear.

LEE, J.-H., SHIN, S. Y., AND CHWA, K.-Y. 1999. Visibility-based pursuit-evasions in a polygonal room with a door. In *Proceedings ACM Symposium on Computational Geometry*.

LOBATON, E., VASUDEVAN, R., BAJCSY, R., AND SASTRY, S. 2010. A distributed topological camera network representation for tracking applications. *IEEE Transactions on Image Processing 19,* 10, 2516–2529.

MAGNUS, W., KARRASS, A., AND SOLITAR, D. 1976. *Combinatorial Group Theory*. Dover, New York.

MARINAKIS, D., GIGUERE, P., AND DUDEK, G. 2002. Learning network topology from simple sensor data. In *Prof. Canadian Conference on Artificial Intelligence*.

MONTEMERLO, M., THRUN, S., KOLLER, D., AND WEGBREIT, B. 2002. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *AAAI National Conference On Artificial Intelligence*.

O'KANE, J. M. AND LAVALLE, S. M. 2007. Sloppy motors, flaky sensors, and virtual dirt: Comparing imperfect, ill-informed robots. In *Proceedings IEEE International Conference on Robotics and Automation*.

PARR, R. AND ELIAZAR, A. 2003. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In *Proceedings International Joint Conference on Artificial Intelligence*.

SARKAR, R. AND GAO, J. 2010. Differential forms for target tracking and aggregate queries in distributed networks. In *Proc. of the 16th Annual International Conference on Mobile Computing and Networking (MobiCom'10)*. 377–388.

SHRIVASTAVA, N., MADHOW, R. M., AND SURI, S. 2006a. Target tracking with binary proximity sensors: fundamental limits, minimal descriptions, and algorithms. In *Proceedings of the 4th international conference on Embedded networked sensor systems*. 251–264.

SHRIVASTAVA, N., MADHOW, R. M. U., AND SURI, S. 2006b. Target tracking with binary proximity sensors: fundamental limits, minimal descriptions, and algorithms. In *Proceedings of the 4th international conference on Embedded networked sensor systems*. ACM, New York, NY, USA, 251–264.

SILVA, V. D. AND GHRIST, R. 2006. Coordinate-free coverage in sensor networks with controlled boundaries via homology. *International Journal of Robotics Research 25,* 12, 1205–1222.

SINGH, J., MADHOW, U., KUMAR, R., SURI, S., AND CAGLEY, R. 2007. Tracking multiple targets using binary proximity sensors. In *Proceedings of the 6th international conference on Information processing in sensor networks*. ACM, New York, NY, USA, 529–538.

SUZUKI, I. AND YAMASHITA, M. 1992. Searching for a mobile intruder in a polygonal region. *SIAM Journal on Computing 21,* 5, 863–888.

THRUN, S., BURGARD, W., AND FOX, D. 2005. *Probabilistic Robotics*. MIT Press, Cambridge, MA.

THRUN, S., FOX, D., AND BURGARD, W. 1998. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning 31*, 29–53.

TOVAR, B., COHEN, F., AND LAVALLE, S. M. 2009. Sensor beams, obstacles, and possible paths. In *Algorithmic Foundations of Robotics, VIII*, G. Chirikjian, H. Choset, M. Morales, and T. Murphey, Eds. Springer-Verlag, Berlin.

TOVAR, B., FREDA, L., AND LAVALLE, S. M. 2007a. Mapping and navigation from permutations of landmarks. In *AMS Contemporary Mathematics Proc*. Vol. 438. 33–45.

TOVAR, B., MURRIETA-CID, R., AND LAVALLE, S. M. 2007b. Distance-optimal navigation in an unknown environment without sensing distances. *IEEE Transactions on Robotics 23,* 3, 506–518.

VON NEUMANN, J. AND MORGENSTERN, O. 1944. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ.

YU, J. AND LAVALLE, S. M. 2008. Tracking hidden agents through shadow information spaces. In *Proceedings IEEE International Conference on Robotics and Automation*.

YU, J. AND LAVALLE, S. M. 2010. Cyber detectives: Determining when robots or people misbehave. In *Proceedings Workshop on Algorithmic Foundations of Robotics (WAFR)*.

A. HOMOTOPY AND THE FUNDAMENTAL GROUP

Assume that all body paths start and stop at some fixed *basepoint* $x_0 \in X$ which lies in the interior of some region; this assumption is lifted at the end of the section. Two paths, $\tilde{x}$ and $\tilde{x}'$ are called *homotopic* if there exists a continuous function $h : [0,1] \times [0,1] \rightarrow X$ for which the following four conditions are met:

(1) **(Start with first path)** $h(s,0) = \tilde{x}(s)$ for all $s \in [0,1]$.
(2) **(End with second path)** $h(s,1) = \tilde{x}'(s)$ for all $s \in [0,1]$.
(3) **(Hold starting point fixed)** $h(0,t) = h(0,0)$ for all $t \in [0,1]$.
(4) **(Hold ending point fixed)** $h(1,t) = h(1,0)$ for all $t \in [0,1]$.

The parameter t can be interpreted as a "knob" that is turned to gradually deform the path from $\tilde{x}$ into $\tilde{x}'$ without jumping over obstacles. This definition induces an equivalence relation on the set $\tilde{X}$ of all paths. Note that paths that follow different sequences of regions could possibly be equivalent homotopically. It is natural to ask: Under what conditions can be it guaranteed that if two paths traverse the same region sequence, then they are homotopic? We would like to argue that a description of the path up to homotopy equivalence is coarser than a description up to the region sequence. The solution to this problem is provided in Section 5.

Describing the equivalences classes of paths up to homotopy equivalence is part of basic algebraic topology. An algebraic group can be formed by defining the following binary operation on $\tilde{X}$. Let $\tilde{x}_1$ and $\tilde{x}_2$ be two loop paths with the same basepoint x_0. Their product $\tilde{x} = \tilde{x}_1 \circ \tilde{x}_2$ is defined as

$$\tilde{x}(t) = \begin{cases} \tilde{x}_1(2t) & \text{if } t \in [0, 1/2) \\ \tilde{x}_2(2t-1) & \text{if } t \in [1/2, 1]. \end{cases} \tag{8}$$

This results in a continuous loop path because $\tilde{x}_1$ terminates at x_0, and $\tilde{x}_2$ begins at x_0. In a sense, the two paths are concatenated end-to-end. The operation $\circ$ forms a group on the space of all paths. Using the homotopy equivalence relation, we do not want to distinguish between paths that equivalent. Therefore, a quotient group is formed by starting with the operation $\circ$ and applying to the homotopy equivalent classes. The result is called the *fundamental group*, denoted by $\pi_1(X)$, on the space of all equivalence classes of paths [Hocking and Young 1988].

The structure of this group reveals much about the topological structure of the space X. In our case, the topology of X is somewhat simple in that it depends only on the number n of holes, which from Section 2 is the number of obstacles in $\mathcal{O}$. For this case, the fundamental group $\pi_1(X)$ is isomorphic to F_n, in which F_n is called the *free group on n letters*. It is called "free" because it can be presented using generators and

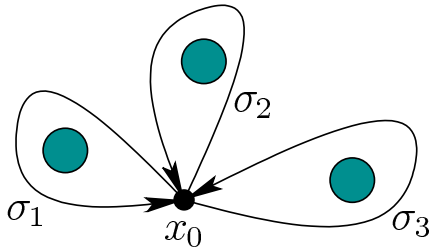


Fig. 23. There are three obstacles and the fundamental group is F_3. Three loop paths are chosen as representatives that correspond to σ_1, σ_2, and σ_3 in Σ. In this way, each word σ_i^k nicely represents a class of paths that wraps k times counterclockwise around the ith obstacle.

no relations (other than the ones appearing in the group axioms). The group F_n can be described as follows. Start with an alphabet Σ of n letters. For every letter $\sigma \in \Sigma$, make an inverse letter denoted by σ^{-1}. This simply doubles the size of the alphabet to make $2n$ symbols. The elements of F_n are simply the set of all finite-length words (or strings) that can be formed from the $2n$ symbols.

For example, suppose $\mathcal{O}$ consists of three obstacles. The fundamental group is F_3, the free group on 3 letters. Let the alphabet be $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$. The group F_3 consists of all words that can be formed from σ_1, σ_2, σ_3, σ_1^{-1}, σ_2^{-1}, and σ_3^{-1}. An example is $\sigma_3\sigma_2^{-1}\sigma_1\sigma_1\sigma_2 \in F_n$.

This is not the complete story, however, because the group axioms contain relations for identities and inverses, even in a free group. Let ε denote the identity element in F_n. It must be true for any element $\sigma \in \Sigma$ that $\varepsilon\sigma = \sigma\varepsilon = \sigma$ and also that $\sigma\sigma^{-1} = \sigma^{-1}\sigma = \varepsilon$. This induces an equivalence relation on the set of all words formed from $2n$ symbols. For example, $\sigma_3\sigma_2^{-1}\sigma_2\sigma_1\sigma_2 = \sigma_3\sigma_1\sigma_2$. An arbitrary word can often be simplified into an equivalent, shorter word by apply these relations. The shortest word in the equivalence class is called a *reduced word*.

A familiar problem in group theory is determining whether two group presentations are equivalent, in other words, their corresponding groups are isomorphic. Recall from linear algebra that there are many ways to define and transform bases for a vector space. A similar but more complicated situation exists for F_n. If we say that F_n is the fundamental group of X, what do the symbols $\sigma \in \Sigma$ actually represent? Each σ should correspond to an equivalence class of homotopic paths. Within an equivalence class, there exists the common issue of picking a representative path for the whole class. However, the situation is more complicated than this because the class that σ represents is somewhat arbitrary.

Figure 23 shows a simple way to represent the symbols for F_3. This way is common for explaining the fundamental group in textbooks. All elements of F_3 can be constructed using $\circ$ and it seems the topic is finished. However, since we are reconstructing paths from sensor data, we may be forced to work with other representative elements of F_n and transform them back into easy-to-interpret representations. Even without the complication of paths, F_n itself can be represented in many ways. Suppose it is presented using the alphabet $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$. Consider a function $g : \Sigma \to F_n$, which replaces every $\sigma \in \Sigma$ with a word in F_n. In some cases, g yields an automorphism (isomorphism to itself) of F_n. The group $\mathrm{Aut}(F_n)$ of all automorphisms of F_n is remarkably complicated. One way to describe them is to perform Nielsen transformations on the alphabet and use them to generate $\mathrm{Aut}(F_n)$ [Magnus et al. 1976]. Our goal in the paper, however, is to carefully sidestep complicated issues regarding automorphisms of F_n while nevertheless reconstructing simple representations of equivalence classes of paths.