To appear in Proc. 1999 IEEE International Conference on Robotics and Automation

Visibility-Based Pursuit-Evasion: The Case of Curved Environments

Steven M. LaValle
Dept. of Computer Science
Iowa State University
Ames, IA 50011 USA
lavalle@iastate.edu

John E. Hinrichsen
Dept. of Mathematical Sciences
Carnegie Mellon University
Pittsburgh, PA 15213 USA
john4@andrew.cmu.edu

Abstract

We consider the problem of visually searching for an unpredictable target that can move arbitrarily fast in a simply-connected, curved, two-dimensional environment. A complete algorithm is presented that is guaranteed to find the elusive target if it is possible for a single pursuer. The key to the algorithm is a cell decomposition based on critical visibility events that occur because of inflections and bitangents of the environment boundary. We have implemented the cell decomposition algorithm, and show several computed examples. The technique is an extension and simplification of a previous technique for searching a polygonal environment. Our solution can also be considered as a step towards a unified approach to pursuit-evasion strategies that have little dependency on the representation of the environment.

1 Introduction

Imagine entering a cave in complete darkness. You are given a lantern and asked to search for any people who might be moving about. Several questions might come to mind. Does a strategy even exist that guarantees I will find everyone? If not, then how many other searchers are needed before this task can be completed? Where should I move next? Can I keep from exploring the same places multiple times? This kind of scenario might apply to firepersons engaged in a rescue effort, law enforcement officials in a hostage situation, or soldiers attempting to secure a potentially-hostile area. Since it is always preferable to place robots at risk instead of humans, we might like to determine whether successful searching strategies can be computed automatically for a mobile robot. Such strategies can also provide valuable advice to people as they plan for high-risk operations.

It is assumed in this paper that there is a single point pursuer in a curved, planar environment that is given the task of searching for any moving evaders, who have unbounded speed. A search strategy is successful if all evaders will eventually fall within the line of sight of the pursuer. See Figure 1. In this paper, we propose a complete algorithm that will compute a path that a pursuer must follow to be guaranteed that all evaders will be seen, regardless of their paths. The approach developed in this paper extends previous work by LaValle et al. [13], from the case of a polygonal environment to an environment with arbitrary curves. In both cases, critical

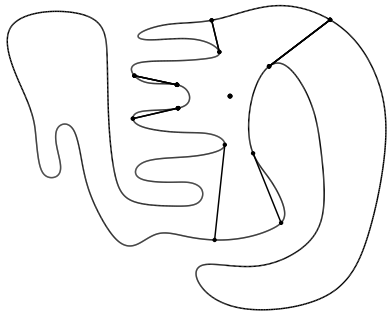

Figure 1: The pursuer is asked to find any moving evaders in the curved environment.

events are found in an information space, and a finite, combinatorial structure is searched that is induced by a special cell decomposition of the environment. The method in [13] would tend toward an infinite number of cells if we considered approximating curved models with arbitrarily-fine polygons. This problem motivated the current work, which identifies a finite set of critical events for a curved environment, and ultimately leads to a simplification of the cell decomposition for polygonal environments. The extension to curved environments is significant because it brings us one step closer to a unified approach to pursuit-evasion problems that is not sensitive to the particular environment representation. Completeness of the pursuit-evasion algorithms can generally be argued in terms of sensor-based representations of the environment, taken from the perspective of a pursuer and its sensors.

Related problems have been considered in dynamic (or differential) game theory, graph theory, and computational geometry. In game theory, pursuit-evasion scenarios, such as the Homicidal Chauffeur problem, express differential motion models for two opponents, and conditions of capture or optimal strategies are sought [1, 9, 10]. In graph theory, the several interesting results have been obtained for pursuit-evasion in a graph, in which the pursuers and evader can move from vertex to vertex until eventually a pursuer and evader lie in the same vertex [14, 15, 16, 18]. One interesting result, which does not hold true for our problem, is that a graph can be searched monotonically (without clearing places

multiple times) [3, 11]. Art gallery problems in computational geometry [4, 17, 21] can be considered as a limiting case for our problem, in which the pursuers are not allowed to move. Several variations of pursuit-evasion in a polygonal environment have also been considered in [5, 22].

Plenty of applications exist that could benefit from visibility-based pursuit-evasion strategies. They can be embedded in surveillance systems that use mobile robotics with various types of sensors (motion, thermal, cameras, etc.). Small mobile robots with pursuit-evasion strategies can be used by special forces in high-risk military operations to systematically search a building in enemy in territory before it is declared safe for entry. In scenarios that involve multiple robots that have little or no communication, a pursuit-evasion strategy could be used to help one robot locate others. One robot could even try to locate another that is malfunctioning. For remote presence applications, it would be valuable if a robot can locate automatically other robots and people using sensors. Beyond robotics, software tools can be developed that assist people in many applications that involve systematically searching or covering complicated environments. Relevant pursuit-evasion scenarios can be imagined in law enforcement, search-and-rescue, toxic cleanup, and in the architectural design of secure buildings.

2 Problem Formulation

The pursuer and evader are each points that move in an open region, R, in the plane. It is assumed that R is bounded by a simple, closed, smooth curve (the curve must also have bounded variation). This curve could be expressed either parametrically or implicitly. Let $e(t) \in R$ denote the position of the *evader* at time $t \geq 0$. It is assumed that $e : [0, \infty) \rightarrow R$ is a continuous function, and the evader is capable of moving arbitrarily fast (i.e., it moves at a finite, unbounded speed). Let $\gamma(t)$ denote the position of the pursuer at time $t \geq 0$. The function $\gamma : [0, \infty) \rightarrow R$ is also continuous, and is referred to as a *strategy*. For any $x \in R$, let $V(x) \subseteq R$ denote the set of all $y \in R$ such that the line segment that joins x and y does not intersect the boundary of R. Let $V(x)$ be called the *visibility region*.

The task can now be formulated. It is assumed that the pursuer does not know the starting position, $e(0)$, or the path, e, of the evader. Initially, an evader could be anywhere in R that is not visible from $\gamma(0)$. A strategy, γ, is called a *solution strategy* if for every continuous $e : [0, \infty) \rightarrow R$, there exists a time $t \in [0, \infty)$ such that $e(t) \in V(\gamma(t))$. In other words, the evader will eventually be seen, regardless of its path. Two observations should be made. Because the evader has unbounded speed, the existence of a solution strategy does not even depend on the maximum speed of the pursuer. The primary concern is the route taken by the pursuer. Also, arbitrarily many evaders could be considered without changing the problem. In other words, guaranteeing that one evader will be found is the same as guaranteeing that for n evaders, they will all be found.

Note that the set of points not visible, $R \setminus V(x)$, is a finite collection of disjoint subsets of R. In the spirit of [18], any such subset of R that might contain the evader is referred to as a *contaminated* region. If it is guaranteed not to contain the evader, then it is referred to as *cleared*. If a region is contaminated, becomes cleared, and then becomes contaminated again, it will be referred to as *recontaminated*.

3 A Combinatorial Representation

The Information Space Let $x \in R$ represent the current pursuer position. Let $S \subseteq R$ represent the set of all contaminated points in R. Let $\eta = (x, S)$ represent an *information state*. The set of all possible information states will be referred to as the *information space*, $\mathcal{I}$. The information space is a standard representational tool for problems that have imperfect state information, and variations of it have appeared in many related planning contexts [2, 6, 7, 8, 12].

Suppose that a strategy is parameterized with a time interval $t \in [0, t_f]$ for some fixed $t_f > 0$. For a fixed strategy, γ, and an initial set of contaminated points, $S(0)$, a path in the information space is obtained. At a given $0 < t \leq t_f$ the set of contaminated points, $S(t)$, can be determined from the history $\{\gamma(t') | t' \in [0, t]\}$. Let $\Psi(\eta, \gamma, t_0, t_1)$ represent the information state that will be obtained by starting from information state η, and applying the strategy γ from t_0 to t_1. The function Ψ can be thought of as a "black box" that produces the resulting information state when a portion of a given strategy is executed. Using Ψ, observe that a path in R induces a path in the information space, $\mathcal{I}$.

Conservative Regions The next definition describes an information invariance property, which when satisfied allows the information space to be partitioned into equivalence classes. A connected set $C \subseteq F$ is *conservative* if for any η such that $q \in F$, and for any $\gamma : [t_0, t_1] \rightarrow C$ such that γ is continuous and $\gamma(t_0) = \gamma(t_1) = q$, then the same information state, $\eta = \Psi(\eta, \gamma, t_0, t_1)$, is obtained. This implies that the information state cannot be altered by moving along closed paths in C. Just as in the case of motions in a conservative field, the following holds [13]:

Lemma 1 *If C is conservative then for any two continuous paths, γ_1, γ_2, mapping into C such that $\gamma_1(t_0) = \gamma_2(t_0)$ and $\gamma_1(t_1) = \gamma_2(t_1)$ then $\Psi(\eta, \gamma_1, t_0, t_1) = \Psi(\eta, \gamma_2, t_0, t_1)$, for any η.*

A Sensor-Based Viewpoint To help us identify conservative regions in R, consider the representation shown in Figure 2, which indicates how the environment might appear from the perspective of the pursuer's visibility sensor in Figure 1. Suppose that the pursuer has a sensor that performs an angular sweep from 0 to 2π and measures line-of-sight distance to the nearest wall. There will be a finite set of orientations at which there is a

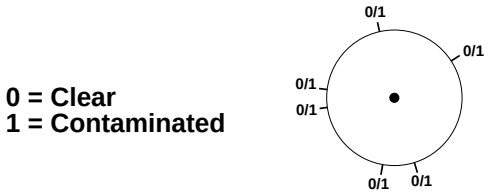

Figure 2: Discontinuities in depth measurements partition the set of viewing directions. Each discontinuity could hide an evader.

discontinuity in the depth data. Figure 2 shows a representation of S^1 with the data discontinuities indicated. Imagine how these discontinuities move in S^1 as the pursuer moves. Let $d : R \times S^1 \rightarrow \Re$ denote the real-valued function that corresponds to the ideal distance measurements. The value $d(x, \theta)$ gives the distance from x to the boundary of R along the ray emanating from x at an angle θ. We call each data discontinuity a *gap* in $d(x)$ ($d(x)$ is considered as a function of θ).

The following lemma establishes the conservative region concept in terms of gaps in S^1. To establish this important lemma, we define what we will refer to as a piecewise homotopy. Let X denote any topological space. Let $f : X \rightarrow \Re$ and $g : X \rightarrow \Re$ denote two piecewise continuous functions that each have exactly k points of discontinuity. Let $h : [0, 1] \times X \rightarrow \Re$ denote a function for which the following hold: 1) $h(0, x) = f(x)$ for each $x \in X$; 2) $h(1, x) = g(x)$ for each $x \in X$; 3) $h(t, x) : X \rightarrow \Re$ has exactly k points of discontinuity for any fixed $t \in [0, 1]$; 4) h is continuous, except at any of the k discontinuities for each $h(t, x)$. Using h, attempt to construct k functions, $\phi_1, \phi_2, \ldots, \phi_k$, that "track" the discontinuties in h as follows. Let $\phi_i : [0, 1] \rightarrow X$ be defined such that if $\phi(t) = x$, then $h(t, x)$ is a point of discontinuity. If the functions $\phi_1, \phi_2, \ldots, \phi_k$ can be defined such that each is continuous, and $\phi_i(t) \neq \phi_j(t)$ for each $t \in [0, 1]$ and each $i \neq j$, then we say that h defines a *piecewise homotopy*. Intuitively, piecewise homotopy can be considered as a generalization of classical homotopy to piecewise continuous functions. The points of discontinuity must move continuously in the piecewise homotopy.

Let $\gamma : [0, 1] \rightarrow R$ be a continuous, closed-loop path for the pursuer. Note that γ can be composed with the first argument of d to yield a function $h : [0, 1] \times S^1 \rightarrow \Re$.

Lemma 2 *If, for some continuous, closed-loop path γ, h defines a piecewise homotopy, then the image of γ in R is contained in a conservative region.*

Proof: If h defines a piecewise homotopy, then $d(\gamma(t)) : S^1 \rightarrow \Re$ and its discontinuities are required to change continuously as $\gamma(t)$ varies. This implies that topology of the image of $d(\gamma(t))$ does not change. In terms of the representation in Figure 2, it implies that although gaps can move in S^1, none of the following can occur: i) no new gap can appear; ii) no existing gap can disappear; iii) no two gaps can merge into one; iv) no gap can split into two or more gaps. These are the only conditions that could cause a topological change that would violate the piecewise homotopy.

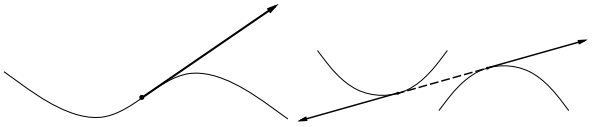

Figure 3: Inflections and bitangents represent critical events in visibility.

Each gap corresponds to a connected component of $R \setminus V(x)$ that is not visible to the pursuer. Each component must be either clear or contaminated (it cannot be a mixture). This implies that the information state can be expressed by placing a binary label on each gap: "1" if the corresponding component of $R \setminus V(x)$ is contaminated, and "0" otherwise. If none of the previous four conditions can occur, then the connected components of $R \setminus V(x)$ will remain preserved (although they will gradually change). This implies that the binary labels cannot change, and hence the information state is the same. □

Critical Events: Inflections and Bitangents Consider the extending rays as shown in Figure 3. On the left, a ray is extended outward from an inflection point, and is terminated when it reaches the boundary of R. On the right, two rays are extended outward from a pair of bitangent points, and are terminated with they reach the boundary of R. We will show that if the pursuer stays inside of a region bounded by these rays, the region will be conservative.

Lemma 3 *If a continuous, closed-loop path γ does not cross the ray of an inflection or bitangent, then h defines a piecewise homotopy.*

Proof: Recall the proof of Lemma 2, which enumerated four ways that a topological change occurs in visibility. The first two correspond to a gap appearing or disappearing on S^1. This corresponds precisely to the case in which an inflection ray is crossed. In one direction a gap appears, and in the other, a gap disappears. The last two cases correspond to a two or more gaps splitting or merging. This corresponds precisely to the case of crossing a bitangent. In one direction a pair of gaps merge, and in the other direction, the same pair splits. These are the only topological changes possible, and each has been identified with crossing a visibility ray of an inflection or bitangent. Thus, if γ does not cross one of this rays, the corresponding h is a piecewise homotopy. □

Theorem 1 *If no visibility rays of inflections or bitangents intersect a region $C \subseteq R$, then C is conservative.*

Proof: If no visibility rays of inflections or bitangents intersect a region $C \subseteq R$, then any continuous path γ whose image is C must not cross the visibility ray of an inflection or bitangent. By Lemma 3, the corresponding function h must be a piecewise homotopy. It follows from Lemma 2 that C is conservative. □

4 A Complete Algorithm

Theorem 1 provides the basis on which to build a complete algorithm. The problem of finding all of the inflections and bitangents reduces to computing roots of polynomial equations. If the ray extensions are performed for all of these critical events, and the intersections between rays are computed, the environment, R, can be partitioned into a finite collection of cells. Each cell is conservative according to Theorem 1, and the boundary of each cell consists of segments of different extension rays, and possibly parts of the boundary of R. Two cells are adjacent if they share a one-dimensional boundary. From this adjacency, a finite graph, G, can be derived (i.e., the dual) in which the vertices represent cells and the edges represent adjacencies between cells. The resulting cell decomposition is similar to a visibility complex in computational geometry [20] and an aspect graph in computer vision [19].

The graph G corresponds to a finite collection of conservative cells in R, but what is needed is a finite collection of cells in the information space $\mathcal{I}$. A directed *information graph*, G_I, can be derived from G. Let $B(C)$ denote a binary sequence that corresponds to labelings that are assigned to gaps using the representation shown in Figure 2. For each possible binary sequence, $B(C)$, a different information state is obtained, but if the pursuer stays within C, the binary labels cannot change. This results in a collection of 2^m cells in $\mathcal{I}$ that each corresponds to a possible labeling of m gaps when the pursuer is in C. Let the graph G_I contain one vertex for each combination of cell C and its possible labelings $B(C)$. The vertices of G_I correspond to a partition of $\mathcal{I}$ into a finite set of equivalence classes.

To complete the construction of G_I, the set of edges must be defined. Each vertex of G_I will have a corresponding vertex in G that corresponds to a cell in R. When an adjacent cell is entered, the pursuer must cross an inflection ray or a bitangent ray (assuming general position). In terms of information states, it must be determined which information equivalence class is reached when going from a vertex in G_I that corresponds to C, to another vertex that corresponds to an adjacent cell, C' in R. This reduces to finding the appropriate binary labeling $B(C')$. If an inflection ray is crossed, then either a gap appears or a gap disappears. If the gap disappears, a bit simply disappears when going from $B(C)$ to $B(C')$. If a gap appears, then it always receives a "0" label. If a bitangent is crossed, then gaps either merge or split. If several gaps merge into one, then the corresponding bit in $B(C')$ will be the logical OR of the corresponding bits in $B(C)$. This is correct because one contaminated region could spread to other regions. If one gap splits into several, the corresponding bits in $B(C')$ will receive the label of the corresponding bit in $B(C)$. Note that G_I is directed because inflection and bitangent rays have different effects on the information state when crossed in opposite directions.

The task of finding a solution now reduces to searching G_I for a path between any information state with $B(C) = [1\,1\,\cdots\,1]$ to any information state with $B(C') = [0\,0\,\cdots\,0]$. The path in G_I induces a path in G. The path in G corresponds to a sequence of cells that must be visited by the pursuer. A path for the pursuer can be constructed by choosing a point in each cell and constructing a path between adjacent cells in the sequence. The cells are generally not convex; however, it is straightforward to determine a path that connects points between two adjacent cells without entering other cells or crossing the boundary of R.

The following theorem establishes the completeness of the algorithm.

Theorem 2 *An algorithm that finds any path to a goal vertex from an initial vertex in G_I is complete for the visibility-based pursuit-evasion problem defined on R.*

Proof: The algorithm is complete if the existence of any solution strategy implies that a path exists in G_I between initial and goal vertices. Let $\{D_1, \ldots, D_n\}$ denote the sequence of conservative cells (as we defined them with inflection and bitangent rays) that are traversed by any given solution strategy, γ. Each D_i corresponds to a vertex in G, and $\{D_1, \ldots, D_n\}$ corresponds to a path in G. This in turn corresponds to a path in G_I. By Lemma 1 and Theorem 1, the information state does not depend on the path chosen within each region, D_i. From this and the fact that γ is a solution strategy, the vertex obtained at the end of the corresponding path in G_I is a goal vertex. □

Number of cells If the total number of inflection and bitangent rays is m, then by Euler's formula there will be $m + i - 1$ cells in R (or vertices in G), in which i is the number of intersections between rays. There can be at most a quadratic number of intersections; therefore, there are at most $O(m^2)$ cells. The graph G_I is considerably larger because of the binary labels; however, it appears that this graph does not need to be completely represented or explored. It remains an open problem to determine if G_I can be searched in polynomial time, even for a polygonal environment.

Simple bounds can be constructed on the number of inflection and bitangent rays, which relates directly to the number of cells. Suppose that the boundary of R is represented by the set of solutions to an implicit polynomial equation of the form $f(x, y) = 0$ (here we use (x, y) to denote a point in R, instead of $x \in R$). The number of inflections and bitangents can be related to the degree of f. An inflection corresponds to a change in sign of the curvature. The curvature is

$$\frac{f_{xx} f_y^2 - 2 f_{xy} f_x f_y + f_{yy} f_x^2}{(f_x^2 + f_y^2)^{3/2}}, \tag{1}$$

and the inflections are the solutions of $f(x, y) = 0$ and $f_{xx} f_y^2 - 2 f_{xy} f_x f_y + f_{yy} f_x^2 = 0$. If the total degree of f is d, then the second equation has degree $3d - 3$. There at most $3d(d-1)$ inflections by Bezout's Theorem. The bitangents occur as solutions to the equations: $f(x1, y1) = 0$, $f(x2, y2) = 0$, $(x1 - x2) f_y(x1, y1) -$

$(y1-y2)f_x(x1,y1)=0$, and $(x1-x2)f_y(x2,y2)-(y1-y2)f_x(x2,y2)=0$. Once again, by Bezout's Theorem, there are no more than d^4 solutions. These results imply that in the worst case, there are no more than $O(d^8)$ cells, in which d is the total degree of $f(x,y)=0$. In practice, we expect the number of cells to be much smaller because it is unlikely that most of the rays will intersect.

5 An Implementation with Examples

The cell decomposition algorithm was implemented using Linux, GNU C++ and the Library of Efficient Data Types and Algorithms (LEDA). Instead of representing the boundary of R implicitly, we decided that it would be easier to work directly with a parametric representation. In either case, the bitangent and inflection rays can be found as the solutions to polynomial equations. We chose to find the critical events numerically, without giving major consideration to stability issues. The purpose of our implementation is to gain some further insights to the problem through computed examples. We currently generate the conservative cell decomposition, and the corresponding graph, G. We are in the process of extending G to G_I to perform a search on the information space.

The program represents the environment as an array of uniform cubic B-splines. Examples can be easily created using a UNIX drawing tool, such as Idraw. The postscript output is then parsed and edited, creating an array of splines. The spline points held in the array are then used to obtain the equations that represent the boundary of R. Each spline curve is described by a cubic parametric equation of the form $x(t)=a_xt^3+b_xt^2+c_xt+d_x$ and $y(t)=a_yt^3+b_yt^2+c_yt+d_y$.

Figure 4 shows the computed cell decomposition for the example shown in Figure 1. Figure 4.d shows the cell decomposition, which is an overlay of boundaries shown in Figures 4.b and 4.c. There are 66 cells in this example. Figure 5 shows four more computed examples. In Figures 5.a and 5.b there are 13 and 26 cells, respectively. In 5.c, the spiral tunnel produces exactly what we would intuitively expect: the pursuer needs to search from end to end. There are only three cells. Figure 5.d shows an example that produced 244 cells. This example is more complicated because many pairs of inflection and bitangent rays intersect.

6 Discussion

We presented a complete algorithm for the problem of computing a strategy for a pursuer in a curved environment that must find an unpredictable evader using line-of-sight visibility. The approach can be considered as an extension to the case of a polygonal environment [13], that brings some additional insight into pursuit-evasion problems. In particular, we used a sensor-based representation that enabled us to develop a complete algorithm based on inflections and bitangents. This technique can also be used to reduce the number of cells considerably in the polygonal case, and ultimately, we

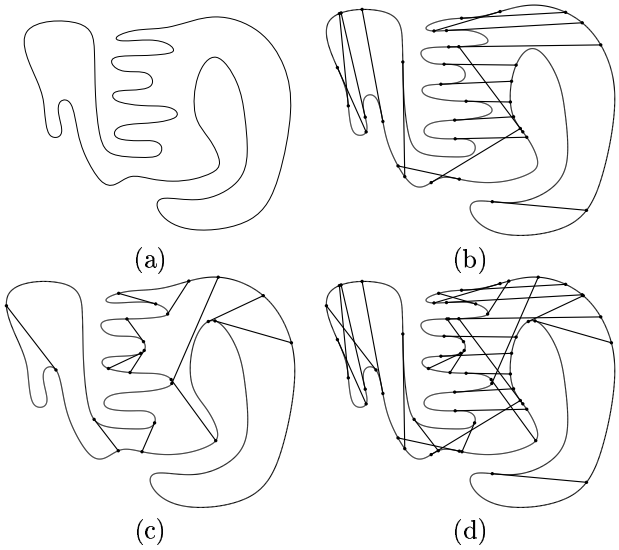

Figure 4: (a) The input; (b) the set of inflection rays; (c) the set of bitangent rays; (d) the conservative cell decomposition.

believe the representation will enable us to approach the problem *without* explicitly building a representation. Consider Figure 6. Figure 6.a shows how sensor data would appear (ideally) to a pursuer in a polygonal environment. In Figure 6.d, we see that the same sensor-based representation implies. Therefore, a cell decomposition of the polygon can be constructed for two kinds of critical events, which are direct analogs of the inflection rays and the bitangent rays.

Many variations of the problem considered in this paper are possible. It might be useful to consider an input representation that contains both curves and line segments. We could also consider problems that require multiple pursuers, including pursuit-evasion in multiply-connected curved environments. This could be handled by extending the sensor-based representation to resolve the information gathered by two or more robots.

Acknowledgments

This work was completed while John Hinrichsen visited Iowa State University as an undergraduate in the summer of 1998. We thank Leo Guibas, Jean-Claude Latombe, David Lin, and Rajeev Motwani for their insights and contributions to the pursuit-evasion problem in a polygonal environment. We thank Giora Slutzki and Boris Simov for their helpful insights about the problem. Finally, we thank Jean Ponce suggesting the use of Bezout's Theorem for bounds on the critical events for implicit polynomials.

References

[1] T. Başar and G. J. Olsder. *Dynamic Noncooperative Game Theory.* Academic Press, London, 1982.

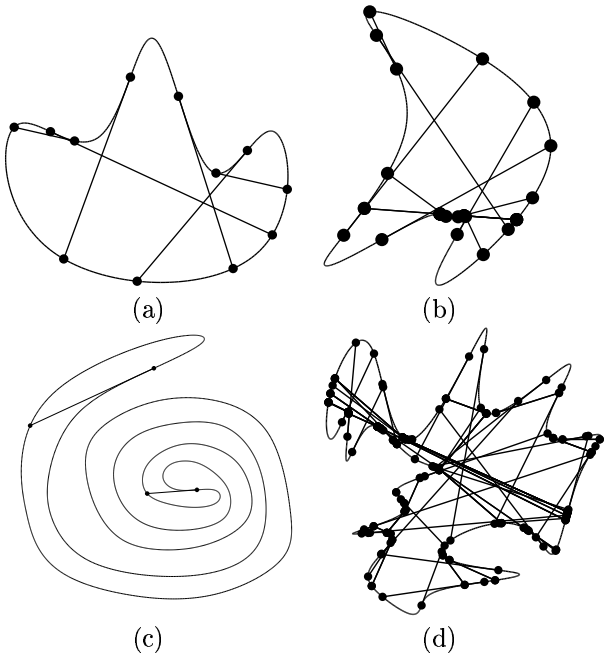

Figure 5: The computed cell decompositions for four different problems.

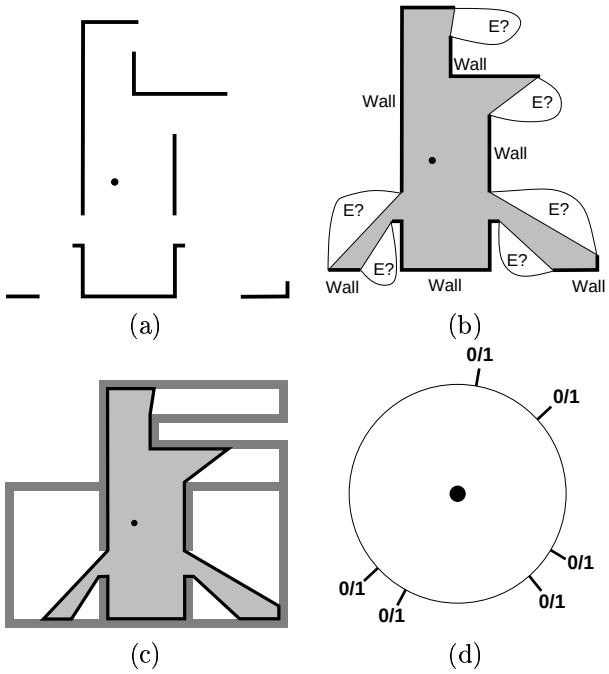

Figure 6: (a) Distance measurements in a polygonal environment; (b) interpreting the data; (c) a possible environment that is consistent with the data; (d) the sensor-based representation can be applied to this case.

[2] J. Barraquand and P. Ferbach. Motion planning with uncertainty: The information space approach. In *IEEE Int. Conf. Robot. & Autom.*, pages 1341–1348, 1995.

[3] D. Bienstock and P. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12:239–245, 1991.

[4] W.-P. Chin and S. Ntafos. Optimum watchman routes. *Information Processing Letters*, 28:39–44, 1988.

[5] D. Crass, I. Suzuki, and M. Yamashita. Searching for a mobile intruder in a corridor – the open edge variant of the polygon search problem. *Int. J. Comput. Geom. & Appl.*, 5(4):397–412, 1995.

[6] B. R. Donald. On information invariants in robotics. *Artif. Intell.*, 72:217–304, 1995.

[7] M. Erdmann. Randomization for robot tasks: Using dynamic programming in the space of knowledge states. *Algorithmica*, 10:248–291, 1993.

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

[9] O. Hájek. *Pursuit Games.* Academic Press, New York, 1975.

[10] R. Isaacs. *Differential Games.* Wiley, New York, NY, 1965.

[11] A. S. Lapaugh. Recontamination does not help to search a graph. *Journal of the ACM*, 40(2):224–245, April 1993.

[12] S. M. LaValle. *A Game-Theoretic Framework for Robot Motion Planning.* PhD thesis, University of Illinois, Urbana, IL, July 1995.

[13] S. M. LaValle, D. Lin, L. J. Guibas, J.-C. Latombe, and R. Motwani. Finding an unpredictable target in a workspace with obstacles. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 737–742, 1997.

[14] F. Makedon and I. H. Sudborough. Minimizing width in linear layouts. In *Proc. 10th ICALP, Lecture Notes in Computer Science 154*, pages 478–490. Springer-Verlag, 1983.

[15] N. Megiddo, S. L. Hakimi, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou. The complexity of searching a graph. *Journal of the ACM*, 35(1):18–44, January 1988.

[16] B. Monien and I. H. Sudborough. Min cut is NP-complete for edge weighted graphs. *Theoretical Computer Science*, 58:209–229, 1988.

[17] J. O'Rourke. *Art Gallery Theorems and Algorithms.* Oxford University Press, New York, NY, 1987.

[18] T. D. Parsons. Pursuit-evasion in a graph. In Y. Alani and D. R. Lick, editors, *Theory and Applcation of Graphs*, pages 426–441. Springer-Verlag, Berlin, 1976.

[19] S. Petitjean, D. Kriegman, and J. Ponce. Computing exact aspect graphs of curved objects: algebraic surfaces. *Int. J. Comput. Vis.*, 9:231–255, Dec 1992.

[20] M. Pocchiola and G. Vegter. The visibility complex. *Int. J. Comput. Geom. & Appl.*, 6(3):279–308, 1996.

[21] T. Shermer. Recent results in art galleries. *Proc. IEEE*, 80(9):1384–1399, September 1992.

[22] I. Suzuki and M. Yamashita. Searching for a mobile intruder in a polygonal region. *SIAM J. Computing*, 21(5):863–888, October 1992.