Automatic Sensor Configuration for Task-Directed Planning*

Amy J. Briggs Bruce R. Donald

Department of Computer Science
Cornell University
Ithaca, New York 14853

Abstract

We consider the problem of planning the configuration of a sensor within the context of a robotic task. In this paper, we focus on geometrically specified tasks in the plane, and give algorithms for computing the regions from which an idealized point-and-shoot sensor can detect a polygonal robot. Our main algorithm allows sensor configurations from which the robot may be partially obstructed, and computes the regions from which the robot can be detected as it translates through the goal at a known orientation. This algorithm runs in time $O(kmn^3(n+m))$ for an environment of complexity n, a robot of complexity m, and a goal of complexity k. The regions constructed can be used by an active sensing system to configure sensors that are guaranteed to observe the robot as it enters the goal.

1 Introduction

This paper addresses the problem of *sensor configuration for task-directed planning*, forging connections between robotics, computational geometry, and active sensing. In task-directed planning, one seeks to generate plans that are tailored to the task at hand. In this paper, we explore a problem in *task-directed sensing*, where sensing is driven by the information needs of the system. We introduce a computational framework in which to study sensor configuration, and give combinatorially precise algorithms for specific cases.

For an example of where such strategies would be useful, consider the Error Detection and Recovery framework [7]. An *Error Detection and Recovery* (EDR) strategy is one that guarantees the goal will be reached if the goal is recognizably reachable, and that failure will be signalled otherwise. Our algorithms can be used in conjunction with an EDR planner to compute where a sensor should be placed in order to recognize success or failure of a motion plan.

A further motivation is in the domain of cooperating robots. Suppose robot A is performing a task, and robot B is equipped with sensors and should "watch" robot A. Then our algorithms can be used, for example, to compute the regions that B should not enter, if A is to remain visible. Less generally, A and B could be part of the same robot, for example, a physically distributed but globally controlled system.

Many other applications of automatic sensor configuration arise in the area of robot surveillance. In particular, our algorithms apply to the problems of intruder detection, execution monitoring, and robot reconnaissance. A natural extension of our work can be made to the problem of beacon placement for robot navigation.

1.1 Goal recognizability

Let's start with the example of two cooperating mobile robots. Suppose robot A is executing a motion plan and should reach a goal G. Where should robot B, the sensor, move to if it wants to detect when robot A has reached the goal? We simplify the planning problem by projecting the robots and their environment onto the plane, and assume that we have a planar map of the environment. The environment, or scene, consists of polygonal obstacles, which may include a polygon representing the boundary of the environment. Then we can ask whether or not the two dimensional "footprint," or projection of robot A onto the plane, is visible from a given sensor location.

Our robot configuration space is denoted C_r, and

*This paper describes research done in the Robotics and Vision Laboratory at Cornell University. Support for our robotics research is provided in part by the National Science Foundation under grants No. IRI-8802390, IRI-9000532, IRI-9201699, and by a Presidential Young Investigator award to Bruce Donald, and in part by the Air Force Office of Sponsored Research, the Mathematical Sciences Institute, Intel Corporation, and AT&T Bell laboratories. The first author's work was additionally supported by an AT&T Bell Laboratories Graduate Fellowship sponsored by the AT&T Foundation.

we consider two types of planar robot motion. We have $C_r = \mathbb{R}^2$ when the robot has two translational degrees of freedom and a fixed orientation. When the robot is allowed to both rotate and translate in the plane, the robot configuration space is $\mathbb{R}^2 \times S^1$.

The sensor we employ is an idealized but physically realizable model of a point-and-shoot sensor, such as a laser range finder. When at a particular sensor location, the sensor returns a distance reading to the nearest obstacle, which is accurate to within some known bound. Such a ranging device has been developed in our robotics laboratory at Cornell, and has been used for both map-making and robot localization. Details of how the sensor operates can be found in the work of Brown, Chew, and Donald [3].

We denote the space of sensor placements $C_{s_p} = \mathbb{R}^2$ and the space of sensor aims $C_{s_c} = S^1$. Our sensor configuration space is $C_s = C_{s_p} \times C_{s_c} = \mathbb{R}^2 \times S^1$. For a given sensor configuration $(\mathbf{p}, \psi)$, the sensor returns distance readings for a subset of $\mathbb{R}^2$. We call this subset the *sensitive volume* of the sensor, and denote it by $SV(\mathbf{p}, \psi)$. In what follows, we restrict our attention to questions of visibility within the sensitive volume.

Given a subset X of the robot configuration space C_r, the *recognizability region* for X is the set of all sensor configurations from which the sensor can detect the robot when its reference point lies in X. In sections 2 and 3, we give algorithms for computing the recognizability region of the goal under two different notions of visibility. We justify our use of visibility as a model for detectability by noting that the recognizability region of a robot given most sensors is a subset of the region from which the robot is visible. For an idealized ranging device, the visibility region of the robot is equivalent to the recognizability region. We show in Section 4 how the error characteristics of an imperfect sensor can be used to construct the recognizability region from the visibility region.

1.2 Polygon visibility

The main algorithmic paradigm we use in the computation of recognizability regions is polygon visibility. Polygon visibility questions in the plane typically involve the computation of regions inside a polygon that are visible from some point or edge inside the polygon. We are interested in *external* polygon visibility, which we define as follows. We say that polygon A is completely visible or *unobstructed* from a point p external to A if every point y in the closure of A is visible from p (where A itself is not considered an obstacle). That is, A is completely visible from p in an environment of line segment obstacles if for no point y in A does the segment $\overline{py}$ cross the interior of an obstacle segment. Polygon A is *weakly* visible from p external to A if at least one point in the closure of A is visible from p.

Detecting whether or not a polygon is weakly externally visible can be answered in $O(n)$ time [1]. We give algorithms for explicitly constructing the locus of points from which a polygon is visible, and we analyze how this visibility region changes as the distinguished polygon moves.

1.3 Related work

Researchers in robotics and machine vision have explored many aspects of the sensor configuration problem. Hager and Mintz [8] define the sensing problem to be that of constructing a consistent geometric description (shape, size, and position) of objects. They present a mathematical framework for describing geometric sensing problems and employ the Bayesian decision-making theory to compute optimal sensing strategies. Hutchinson and Castaño [9, 4] introduce the concept of *visual constraint surfaces* to control motion, adding visual feedback to compensate for uncertainty. Cowan and Kovesi [5] study the problem of automatic camera placement for vision tasks. They consider the constraints on camera location imposed by resolution and focus requirements, visibility and view angle, and forbidden regions depending on the task. In the area of grasp planning, Laugier, Ijel and Troccaz [10] employ partial and complete visibility computations in selecting sensor locations to acquire information about the environment and to guide grasping movements.

The questions of detecting polygon visibility and constructing visibility regions under a variety of assumptions is a rich area of past and ongoing research in computational geometry. We mention here a few of the papers most closely related to our problem. Suri and O'Rourke [14, 12] give an $\Theta(n^4)$ algorithm for the problem of computing the locus of points weakly visible from a distinguished edge in an environment of line segments. Their lower bound of $\Omega(n^4)$ for explicitly constructing the boundary of the weak visibility region holds as well for our computation of recognizability regions under a weak visibility assumption. Bhattacharya, Kirkpatrick and Toussaint [1] introduce the concept of *sector visibility* of a polygon, and give $\Theta(n)$ and $\Omega(n \log n)$ bounds, depending on the size of the visibility wedge, for determining if a polygon is weakly externally visible. In a recent paper, de Berg *et al.* [6] study planar motion planning for a robot with bounded directional uncertainty. They give algorithms for constructing the regions from which goals may be reached, and show that the complexity of the regions depends on the magnitude of the uncertainty angle.

Teller [16] solves the weak polygon visibility problem for a special case in $3D$. Namely, he computes the antipenumbra[1] of a convex area light source shining through a sequence of convex areal holes in three dimensions. Tarabanis and Tsai [15] examine the question of complete visibility for general polyhedral environments in $3D$. For a feature polygon of size m and a polyhedral environment of size n, they present an $O(m^3n^3)$ algorithm for computing the locus of all viewpoints from which the fixed feature polygon can be entirely seen.

Our work bridges the gap between research on the theoretical problems of polygon visibility and research on sensor configuration which assumes simple geometric environments. In cluttered environments, requiring complete visibility may restrict sensor placement too severely, while allowing weak visibility can greatly increase the set of sensor positions from which the robot is detectable. Under both models of detectability, our work is the first to precisely characterize the complexity of computing goal recognizability, and to compute visibility maps for a robot at any position within a goal polygon.

2 Complete visibility

In this section we briefly consider a simplified version of the robot detection problem, in which the only accepted sensor placements are those that allow an unobstructed view of the robot at any position or orientation within the goal region. Our result is the following:

Theorem 1 *The recognizability region of the robot translating and rotating through a goal of size k can be computed in time $O(n\alpha(n) + nk)$ in the complete visibility model.*

The idea is that we construct in time $O(nk)$ a set of $O(n)$ constraint rays that bound the recognizability region. The arrangement of the constraint rays and the environment now partitions the sensor placement space C_{s_p} into regions from which the robot in the goal is either partially occluded or entirely visible. But instead of computing the entire arrangement, we note that it suffices to compute a single cell in the arrangement, namely the cell containing the goal. Computing this cell is equivalent to computing the lower envelope of a set of line segments in the plane, which for a set of size n takes time $O(n\alpha(n))$, where $\alpha(n)$ is the inverse Ackerman function. For details, see [2].

[1] The *antipenumbra* is the volume from which some, but not all, of a light source can be seen.

3 Weak visibility

We turn now to the question of computing recognizability regions under a weak visibility assumption. First we consider the problem of detecting a stationary robot within a polygonal obstacle environment. We then apply these tools to the problem of detecting the robot as it enters the goal.

3.1 Detecting a stationary robot

For robot A at configuration $\mathbf{q} \in C_r$, we construct the weak visibility region using an angular sweep technique to compute visibility triangles. The approach is similar to that given by Suri and O'Rourke for computing the region weakly visible from an edge [14].

The union of the visibility triangles forms the region from which A is weakly visible. The complement of the union of triangles and the scene is a collection of holes in the visibility region, which we call *shadows*. Note that in the partial visibility model, shadows are not necessarily bounded by tangents between an obstacle and the goal. See Figure 1 for an illustration. The overall time for explicitly computing the boundary of the weak visibility region for robot A at any fixed configuration $\mathbf{q}$ is $O(n^2(n+m)^2)$. The region can be output as a union of triangles, without computing the boundary, in time $O(n(n+m))$.

3.2 Moving within the goal

We now address the problem of maintaining line-of-sight contact with the robot as it moves within the confines of a particular polygon, for example, as the robot moves within the goal. How do the visibility triangles and shadows change as the robot moves? To answer this question, we need to introduce some additional terminology. Let e be a visibility edge, which, when extended maximally, intersects the robot at point x. The endpoint of e lying closer to x (possibly x itself) is defined as the *anchor* vertex of e, while the further endpoint is called the *attachment* vertex of e. If a shadow vertex lies in free space, i.e., if it lies inside the bounding polygon and is not on the boundary of an obstacle, then we call it a *free* vertex of the shadow.

We now show that as the robot translates, free

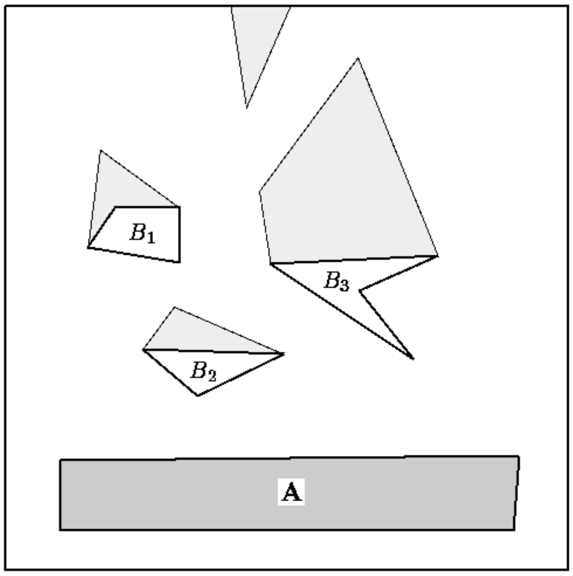

Figure 1: **The shadows cast by obstacles B_1, B_2, and B_3 are shown lightly shaded; the free space represents the weak visibility region of robot A, which is shown darkly shaded.**

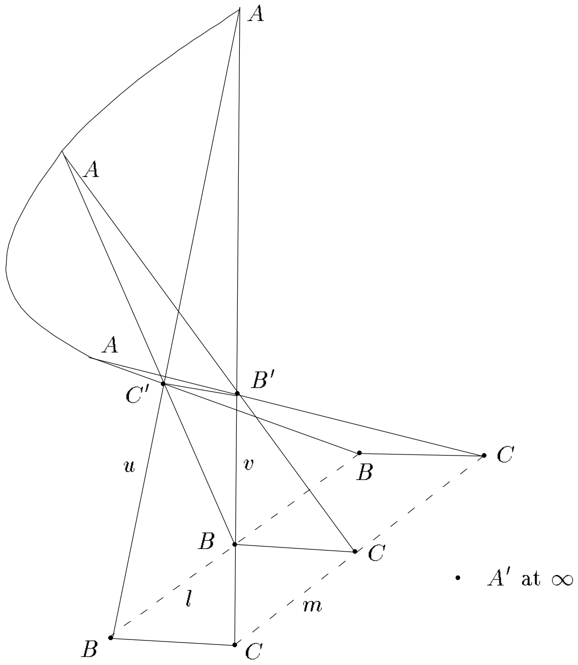

Figure 2: **Correspondence between Maclaurin's construction and shadows. Segment BC represents an edge of the robot; segment $B'C'$ represents an obstacle edge. A describes a point conic, with its supporting edges bounding the shadow. When robot motion is translational, lines l and m are parallel, and point A' lies at infinity.**

shadow vertices trace out point conics[2] if their generating edges are anchored on the robot.

Theorem 2 *Let u and v be two extended visibility rays, anchored at robot vertices and tangent to obstacles, such that they intersect to form a free vertex on the boundary of a shadow. The path traced by the intersection point of rays u and v describes a point conic as the robot translates.*

Proof: Consider the robot at a fixed orientation, with its reference point translating along an edge e of the goal. Let visibility rays u and v be anchored at robot vertices B and C, with obstacle attachment points C' and B', respectively. The attachment points B' and C' form fixed points through which the visibility rays u and v must pass. As the robot translates along edge e, line segment BC translates rigidly, with B and C constrained to lie on lines parallel to edge e. The line through B and C can be thought of as passing through a fixed point A' at infinity. What we have constructed corresponds precisely with Maclaurin's construction of a conic [13]:

Lemma 1 (Maclaurin 1721) *ABC is a variable triangle which is such that the vertices B and C respectively move on two fixed lines l and m, and each side BC, CA and AB of the triangle passes through a fixed point. Then the vertex A describes a conic section.*

See Figure 2 for an illustration of the correspondence between Maclaurin's construction and our case.

Since parallel lines intersect at infinity in projective geometry, we can put fixed point A' at infinity on the line through B and C, so that the robot can translate. If the three fixed points are colinear, the conic is singular, describing a line through the intersection point of lines l and m. In the case of translational robot motion, three colinear fixed points arise when

[2] A *point conic* is simply a *conic* (i.e., an ellipse, a parabola, or a hyperbola) that is formed by a locus of intersection points. We use the term "point conic" to distinguish from the dual case of a "line conic", which is formed by the envelope of a set of lines. Formally, a point conic is the locus of the points that are common to pairs of corresponding lines of two pencils between which there is a projectivity.

the line through attachment points C' and B' is parallel to the line through anchor points B and C. The free shadow vertex formed by the intersection of u and v would then describe a line parallel to edge e as the robot translates along e. □

3.3 Swept shadows

We have shown how to compute shadows for any fixed robot position, and have seen how these shadows change as the robot translates. In order to detect the robot as it enters the goal, we must compute the shadows swept for all positions of the robot in the goal. We define a *swept shadow* of the goal to be a maximal connected region of C_{s_p}, such that for each point $\mathbf{p}$ in the region, there exists a configuration of the robot in the goal from which the robot is totally occluded.

We compute swept shadows for the robot entering the goal at a fixed orientation by translating the robot polygon along the edges of the goal polygon. The boundary of a swept shadow is composed of obstacle segments and the curves (lines and conics) traced by free vertices. Discontinuities in the boundary of a swept shadow occur at *critical events*. We characterize the critical events as follows:

1. A moving extended visibility edge becomes aligned with a fixed edge of the visibility graph.

2. A free vertex of a shadow intersects an obstacle edge or the bounding polygon.

3. Two moving extended visibility edges bounding a shadow become parallel.

The extended visibility edges bounding the shadows are all either external local tangents between an obstacle and the robot, or internal local tangents between obstacles. If the obstacles are fixed, then the visibility edges between them remain fixed. As the robot moves, the only visibility edges that move are those that are anchored on a vertex of the robot.

With n vertices in the scene and m robot vertices, there are $O(mn)$ moving visibility edges. As the robot translates along an edge of the goal, a visibility edge anchored at robot vertex a_i and attached at obstacle vertex b_j could become aligned with each of the $O(n)$ fixed visibility edges at obstacle vertex b_j. This gives $O(mn^2)$ critical events of the first type as the robot translates along an edge of the goal. There are $O(m^2n^2)$ free vertices tracing out curves, which may intersect each of the $O(n)$ obstacle segments. This gives $O(m^2n^3)$ critical events of the second type. When the third type of critical event occurs, a free vertex disappears. There are $O(m^2n^2)$ of these events.

At a critical event of the first type, an extended visibility edge appears or disappears, causing a visibility triangle to appear or disappear. The total cost of handling all updates of this type is $O(mn^4 + m^2n^3)$. Only local change is caused by events of the second type and third type.

Between critical events, we simply grow the shadows, either along lines or conics. Note that the shadows never shrink: A point $\mathbf{p} \in C_{s_p}$ is in shadow with respect to a polygonal goal if there exists some robot configuration such that the robot is not at all visible from $\mathbf{p}$. The computation of swept shadows is done by translating the robot polygon along the edges of the goal, updating the boundary at critical events. Total running time of the algorithm for a goal of complexity k is $O(kmn^3(n+m))$.

4 Sensor uncertainty

Given a real sensor, we must account for the fact that the sensor cannot be configured exactly. Rather, it will be subject to both errors in placement and errors in aim. Therefore we would like to compute the set of sensor configurations that account for this uncertainty in sensor configuration. In this section, we sketch how the computation of visibility regions can be extended to handle sensor error.

Let ϵ_{pos} denote the worst-case positional uncertainty of the sensor. If the sensor is commanded to move to position $\mathbf{p}$, its actual position could be any position in the disc of radius ϵ_{pos} centered at $\mathbf{p}$. Let ϵ denote the maximum angular error of the sensor aim. That is, if the sensor is commanded to point in direction ψ, its actual heading could be any direction in the cone $\psi - \epsilon, \psi + \epsilon$.

We handle positional uncertainty by growing the shadows by the uncertainty ball of radius ϵ_{pos}. The complement of the union of these grown shadows with the scene will be the visibility region that accounts for uncertainty in sensor position.

The effect of the directional uncertainty of the sensor is to disallow narrow wedges of visibility. After we compute the extended visibility edges as above, we visit each vertex in the scene, and combine adjacent visibility triangles that end on the same polygon. The maximal union of adjacent visibility triangles anchored on a single vertex will be a simple visibility polygon. We then compute the "core triangle" of this visibility polygon, that is, the maximal inscribed triangle whose apex is at the anchor vertex.

If the angle at the apex of such a maximal visibility triangle is less than our angular uncertainty bound ϵ, we discard the polygon. Otherwise, we classify the maximal visibility triangle as a *ϵ-fat triangle*. After this processing, we now have $O(n^2+mn)$ fat visibility triangles. We can now use a result of Matoušek *et al.* [11] on the union of fat triangles, which bounds the number of *holes* in a union of fat triangles. In our case, the "holes" are shadows in a union of visibility triangles. Their theorem states that for any fixed $\delta > 0$, and family $\mathcal{F}$ of n δ-fat triangles, the union has $O(n)$ holes, with the constant of proportionality depending on δ. When we restrict our visibility triangles to be at least ϵ-fat, we have at most $O((n^2+mn)/\epsilon^{O(1)})$ shadows.

5 Conclusion

In studying automatic sensor configuration planning, part of our motivation is to shed light on the complexity of computing goal recognizability. We have given algorithms for computing goal recognizability with an idealized ranging sensor under two notions of detectability. Complete visibility of the robot in the goal can be computed efficiently, and provides a good model of detectability in an uncluttered environment. Given a complicated environment, the weak visibility model affords more sensor configurations from which the robot may be partially occluded. We have also described how the regions we compute can be restricted to handle errors in sensor configuration. We are now working on extending the uncertainty model to handle sensor reading errors in addition to sensor configuration errors. A related extension underway is the computation of efficiently-computable conservative approximations to recognizability regions. Most of our algorithms have been implemented, and are being used in goal recognizability experiments with the mobile robots in our lab.

Acknowledgments

This work grew out of discussions at the MSI Workshop on Computational Robotics held at Cornell in June 1991. The authors are grateful to Mike Erdmann, Tomás Lozano-Pérez, and Matt Mason for their inspiration and the ideas they shared at this meeting. We also thank Klara Kedem, Paul Chew, Daniel Scharstein, Russell Brown, Dan Halperin, Leo Guibas, and Susan Landau for their insightful comments and suggestions on this work.

References

[1] B. Bhattacharya, D. G. Kirkpatrick, and G. T. Toussaint. Determining sector visibility of a polygon. In *Proc. ACM Symp. on Comp. Geom.*, pages 247–253, June 1989.

[2] A. J. Briggs. *Task-Level Strategies for Robot Sensing and Control.* PhD thesis, Department of Computer Science, Cornell University, May 1994.

[3] R. G. Brown, L. P. Chew, and B. R. Donald. Localization and map-making algorithms for mobile robots. In *Proc. IASTED Int. Conf. on Robotics and Manufacturing*, pages 185–190, Sept. 1993.

[4] A. Castaño and S. Hutchinson. Hybrid vision/position servo control of a robotic manipulator. In *Proc. IEEE ICRA*, pages 1264–1269, May 1992.

[5] C. K. Cowan and P. D. Kovesi. Automatic sensor placement from vision task requirements. *IEEE PAMI*, 10(3):407–416, May 1988.

[6] M. de Berg, L. Guibas, D. Halperin, M. Overmars, O. Schwarzkopf, M. Sharir, and M. Teillaud. Reaching a goal with directional uncertainty. In *Proc. Int. Symp. on Algorithms and Computation*, Dec. 1993.

[7] B. R. Donald. *Error Detection and Recovery in Robotics*, volume 336 of *Lecture Notes in Computer Science*. Springer-Verlag, 1988.

[8] G. Hager and M. Mintz. Computational methods for task-directed sensor data fusion and sensor planning. *IJRR*, 10(4):285–313, Aug. 1991.

[9] S. Hutchinson. Exploiting visual constraints in robot motion planning. In *Proc. IEEE ICRA*, pages 1722–1727, Apr. 1991.

[10] C. Laugier, A. Ijel, and J. Troccaz. Combining vision based information and partial geometric models in automatic grasping. In *Proc. IEEE ICRA*, pages 676–682, 1990.

[11] J. Matoušek, N. Miller, J. Pach, M. Sharir, S. Sifrony, and E. Welzl. Fat triangles determine linearly many holes. In *Proc. 32nd Annual IEEE Symp. on Foundations of Computer Science*, pages 49–58, 1991.

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

[13] D. Pedoe. *An Introduction to Projective Geometry.* The Macmillan Company, 1963.

[14] S. Suri and J. O'Rourke. Worst-case optimal algorithms for constructing visibility polygons with holes. In *Proc. ACM Symp. on Comp. Geom.*, pages 14–23, 1986.

[15] K. Tarabanis and R. Y. Tsai. Computing occlusion-free viewpoints. In *CVPR*, pages 802–807, 1992.

[16] S. J. Teller. Computing the antipenumbra of an area light source. *Computer Graphics (Proceedings SIGGRAPH '92)*, 26(2):139–148, July 1992.