Localizing an object with finger probes

Robert Freimer(1) Samir Khuller(2) Joseph S.B. Mitchell(3)
Christine Piatko(4) Kathleen Romanik(5)1 Diane Souvaine(6)

(1)Caliper Corporation
Newton, MA
(2)Department of Computer Science
University of Maryland, College Park, MD 20742
(3)Department of Applied Mathematics and Statistics
State University of New York, Stony Brook, NY 11794-3600
(4)National Institute of Standards and Technology
Gaithersburg, Maryland 20899
(5)DIMACS Center for Discrete Mathematics and Theoretical Computer Science
Rutgers University, Piscataway, NJ 08855-1179
(6)Department of Computer Science
Rutgers University, New Brunswick, NJ 08903

ABSTRACT

We consider the problem of identifying one of a set of polygonal models in the plane using point probes and finger probes. In particular, we give strategies for using a minimum number of finger probes to determine a finite number of possible locations of an unknown interior point p in one of the models. A finger probe takes as input an interior point p of a polygon P and a direction θ, and it outputs the first point of intersection of a ray emanating from p in direction θ with the boundary of P.

We show that without a priori knowledge of what the models look like, no finite number of finger probes will suffice to localize the point p. When the models are given in advance, we give both batch and dynamic probing strategies for solving the problem. We consider both the case where the models are aligned rectilinear polygons and the case where the models are simple polygons.

Keywords: Model-based computer vision, automatic target recognition, probing, localizing, pinning

1. INTRODUCTION

This paper considers the following problem in model-based computer vision: given a set of planar polygonal models and a computer image that contains exactly one instance of one of the models, discover which model is in the image, as well as its exact location. The model instance may be translated anywhere within the image, but it is complete and unoccluded and is not rotated or scaled.

The sensing device used to identify which model is present in the image is a *point probe.* This type of probe determines whether a given pixel of the image is black or white. If pixels have grey-scale values, then a threshold is used to determine whether a pixel is black or white. Using point probes we can also simulate finger probes. A *finger probe* takes as input a point p in the image and a direction θ, and it returns the first boundary point of the object that is hit by a ray emanating from p in direction θ, or ∞ if the object

[1]Corresponding author, romanik@dimacs.rutgers.edu

is not hit. Note that a finger probe may originate from either inside or outside of an object. It is easy to see that a finger probe can be simulated by a sequence of point probes.

Using point and finger probes, the above model-based computer vision problem can be solved by the following three part strategy:

1. Use a sparse set of point probes to find one "hit" point. That is, find one point that is on the boundary or in the interior of the object present in the image. This part of the problem has been addressed in [3].

2. Use finger probes from the hit point found in Step 1 to narrow down the number of possible (model, location) pairs to a finite number. We address this part of the problem in this paper.

3. Use point probes in a dynamic "decision tree" strategy to exactly identify the (model, location) pair present in the image. This part of the problem has been solved in [1].

To illustrate the above strategy, consider the following example. Suppose the three models shown on the left of Figure 1 are given, and suppose that after a hit point p is returned by Step 1 of the strategy, two finger probes are shot from p, yielding the outcome shown on the right of Figure 1. These two probes yield four possible placements of the hit point p in a model, as illustrated in Figure 2.

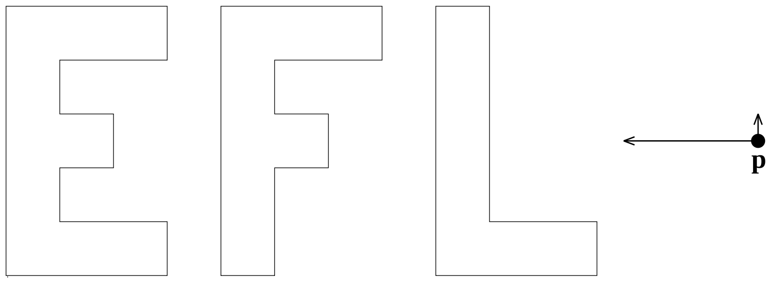

Figure 1: Three models (left) and the outcome of finger probes from point p (right).

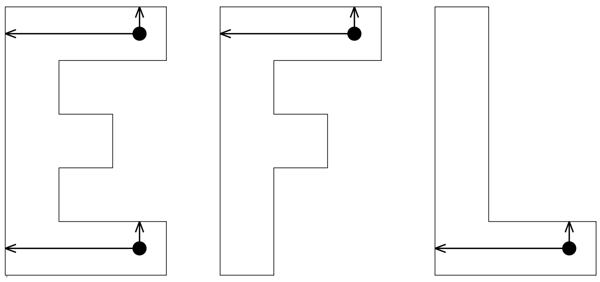

Figure 2: Four possible placements of the hit point p in the models.

By overlaying instances of the models so that the four possible placements of p coincide, we get an overlay arrangement, as illustrated on the left side of Figure 3 (in the figure the model instances are shifted slightly for clarity). Using this overlay arrangement we can develop a dynamic decision tree strategy, as illustrated on the right side of Figure 3, to further probe the image to determine the exact (model, location) pair in the image. First an additional point probe is made at location 1. Note that all probe locations are chosen relative to the hit point p. If it is black (indicated by a **B**), then another probe is made at location

2. If this probe is also black, then the point p must be located in the bottom of the E. Otherwise, if location 2 is white, then p is in the L. Similarly, if probe 1 is white, the right subtree is followed to determine the location of p.

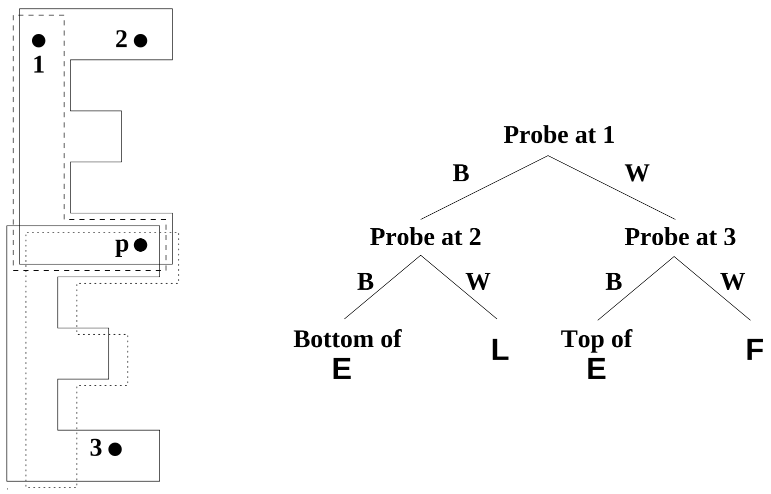

Figure 3: Overlay arrangement (left) and decision tree to determine model present (right).

This paper addresses the second step of the above probing strategy for identifying which model is present in an image. That is, a set of planar polygonal models is given, as well as a computer image with one hit point marked. The hit point is an interior or boundary point of an instance of one of the models. The goal is to use a minimum number of finger probes from this point to narrow down the number of possible locations of this point in the models to a finite number.

It is shown that without a priori knowledge of the models, no finite number of probes will suffice. When the models are given in advance, both batch and dynamic probing strategies for solving the problem are presented. Both the case where the models are aligned rectilinear polygons and the case where the models are simple polygons are considered.

2. IDENTIFYING A MODEL WITH FINGER PROBES

In this section we describe our model for identifying and localizing one of a set of polygonal models using a minimum number of finger probes.

We are given a set of k models, $\{M_1, M_2, \ldots, M_k\}$, each with a known orientation and scale, and we are given an image that contains exactly one of the models. Each model M_i is a simple polygon with n_i vertices, which are given in clockwise order around M_i so that when traversing the boundary of M_i in this order, the interior of M_i always lies to the right. The vertices of each M_i are given with respect to a local coordinate system, where the first vertex lies at the origin. The total number of vertices in all models is $n = \sum_{i=1}^{k} n_i$.

We are also given a hit point p in the image, which is given as a point in $\mathbb{R}^2$, that is in the interior of the model present in the scene. The origin of the coordinate system for the image is assumed to be at the lower lefthand corner of the image. The outcome of our localization strategy will be a set of model, location pairs (M_i, t_j), where t_j is a translation vector that translates the local coordinate system of M_i so that p lies within M_i and all the finger probes emanating from p end at the boundary of M_i (without ever crossing the boundary).

We develop strategies that use finger probes from a point p at an unknown interior location in an unknown model to narrow down the number of possible placements of p in a model to a finite number. To do this we must shoot a sufficient number of finger probes to ensure that the probes are "pinned" in the polygon and cannot "slide". If the finger probes that are shot from point p all hit the same edge or parallel edges of the model, then there may be an infinite number of possible placements of p in the model (see Figure 4) since the arrangement of probes will be able to slide along these edges.

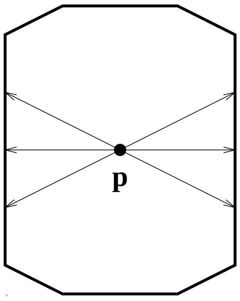

Figure 4: Many probes on the same or parallel edges can slide.

If the models are constrained only to be simple polygons, and the number of vertices is unknown, then no finite number of probes will suffice to solve the problem, as we now show.

Theorem 1 *No finite number of finger probes emanating from an interior point p of an unknown simple polygon P suffice to narrow down the number of possible placements of p in P to a finite number.*

Proof: We take an adversary approach to constructing P. Consider a circle C centered at p. For each finger probe shot from p, the adversary returns the point on C hit by the probe. After any finite number of probes are shot, a diameter d of C can be drawn that does not pass through any of these probes, and a polygon P can be constructed such that each edge hit by a probe is parallel to d, and therefore the probes can slide along these edges (see Figure 5). $\square$

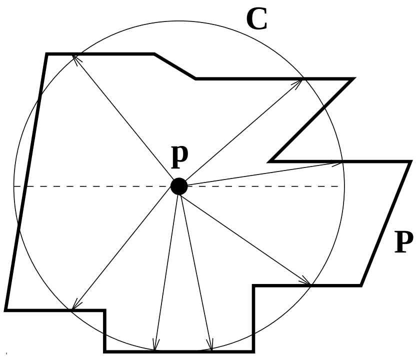

Figure 5: Probes are not sufficient to localize p.

When the models are given in advance, the placement of the initial hit point can always be narrowed down to a finite number of possibilities.

Research with a similar flavor was done by Czyzowicz, Montejano, Stojmenovic and Urrutia [4, 7] on immobilizing a shape in the plane. They considered the problem of finding a minimal set of points I on the boundary of a planar shape P such that any rigid motion of P in the plane forces at least one point in I to penetrate the interior of P.

3. BATCH STRATEGY VERSUS DYNAMIC STRATEGY

For some settings it is desirable to produce a set of batch probes, which can be executed simultaneously from any interior hit point in any model to determine a finite number of possible placements of the hit point in the models. In other settings finger probes must be executed sequentially, so a dynamic strategy can be used, which determines the direction of a new probe based on the outcome of all previous probes.

If all the models are aligned rectilinear polygons, then it is easy to see that two probe directions – one horizontal and one vertical – suffice to narrow down the number of possible placements of the initial hit point to a finite number. Therefore, the strategy for producing either batch or dynamic probes will be the same.

On the other hand, if the models are simple polygons, then a dynamic strategy can yield a smaller number of probes than a batch strategy. In the following sections we consider the two cases of determining probing strategies for rectilinear polygons and for general simple polygons, and we discuss both batch and dynamic strategies for simple polygons.

4. RECTILINEAR MODELS

For aligned rectilinear models the strategy for choosing probe directions is trivial since two probe directions suffice to narrow down the number of possible placements of the initial hit point to a finite number. Therefore, we consider the next part of the problem, which is how to preprocess the models so that given the outcome of the finger probes, the set of possible locations of the initial hit point can be found quickly. To do this, we partition the set of all possible outcomes of the finger probes into a small number of equivalence classes.

We can represent the outcome of horizontal and vertical finger probes by the horizontal and vertical distances from p to the boundary of the model. If we use only two probes, then there is an infinite number of possibilities for these distances. Instead we use four probes, two horizontal and two vertical, yielding a "cross" that fits inside the model (See Figure 6). The distances a, b, c, d determine the cross completely. Although there is an infinite number of possibilities for the distances a, b, c, d corresponding to a cross, if we use the width $w = a + b$ and the height $h = c + d$ of a cross to represent it, then there can be at most $O(n_i^2)$ distinct crosses for a rectilinear model with n_i vertices (see Figure 7). Therefore, by using the width and height of a cross, we get a finite number of equivalence classes for the possible outcomes of the probes.

We preprocess the k models into equivalence classes of crosses by extending the lines incident to each reflex angle of a model M_i into the interior of the model (See Figure 8). This yields an arrangement formed by at most n_i lines within the model. Each cell (or face) in the arrangement is a rectangle that represents an equivalence class of crosses with a *fixed* value for w and h, where the center of each cross lies in the rectangle. The crosses that have their centers in the same cell are incident on the same set of edges and are essentially the same, except for the position of the center of the cross.

Four probes yield a cross of a particular size, and we can use the w, h values of the cross to locate the possible cells in which its center could lie. If we sort all the cells for all the models lexicographically according to their (w, h) values, then we can do a binary search in $O(\log n)$ time to find the set of cells with a particular (w, h) value. Since each model can have at most $O(n_i)$ cells with the same (w, h) value (See Figure 9), there will be at most $O(n)$ cells for a given (w, h) value.

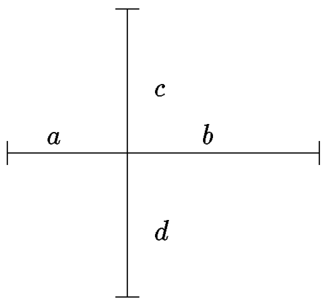

Figure 6: Four probes determine a cross.

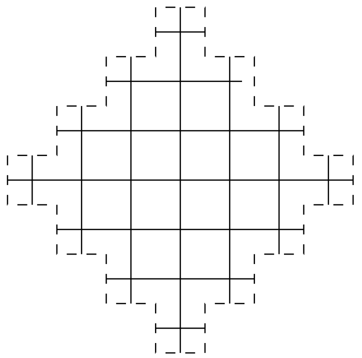

Figure 7: A model with $\Omega(n_i^2)$ distinct crosses.

Among the set of cells with the same (w, h) value, we have to further search by examining the a, c values of the cross until we arrive at the set of consistent placements among all the models. We use the a, c values to decide which cells are invalid. Each cell has a range for its a value and a range for its c value, forming a rectangle in 2-dimensions, where the axes indicate the length of these two pieces of the cross.

By laying down all $O(n)$ of these rectangles we get a planar subdivision with no more than $O(n^2)$ regions. Given the a, c values of a particular cross, we can do a planar point location on this subdivision in $O(\log n)$ time [6] to discover the feasible set of cells in which the cross could lie. Although a particular region in this subdivision may be associated with $\Omega(n)$ cells, we can use a total storage space of $O(n^2)$ by storing with each edge of the subdivision the change in cells between the adjacent regions and only storing a list of cells in the regions of the subdivision where this list is a local minima.

5. SIMPLE POLYGONAL MODELS

In this section we devise both batch and dynamic strategies for probing simple polygonal models.

5.1 Batch strategy

If no model has parallel edges, then shooting two probes in opposite directions (say positive and negative y directions) suffices for a batch strategy. These two probes will hit two different edges of the model, and since a vertical sweep line can intersect a model M_i with a line segment of length l no more than $O(n_i)$ times, these two probes suffice. In fact, shooting two probes in opposite directions works even with parallel edges if there is an orientation θ such that a sweep line with orientation θ never simultaneously hits two

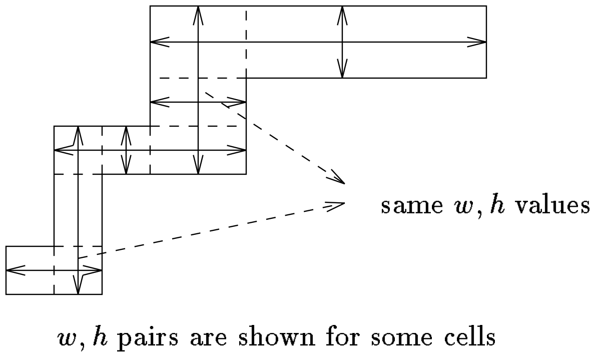

Figure 8: Forming the cells in the polygon.

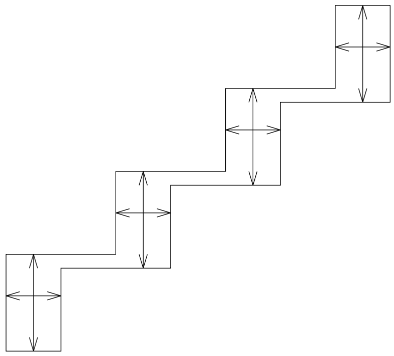

Figure 9: Placements of the same cross in a model.

parallel, visible edges of a model.

If the edges of the models have a limited number of orientations (such as in the case of rectilinear polygons), then a small number of batch probes also suffices. If each model edge has one of m different orientations, then m probes, one parallel to each of the m orientations, suffices.

In the worst case, however, a batch strategy requires $\Omega(n)$ probes. Consider the piece of a polygon shown in Figure 10. If the hit point p is in the middle of this long narrow piece, then only a probe that is almost parallel to the sides of this piece will not hit one of the sides. Therefore, in order to ensure that two non-parallel edges are hit, a batch probing strategy must chose at least one probing direction almost parallel to the sides of this piece. If all of the models consist of long narrow pieces, each at a different angle, then $\sum_{i=1}^{k} \frac{n_i - 2}{2} = \frac{n}{2} - k$ batch probes are necessary to ensure that two non-parallel edges are hit.

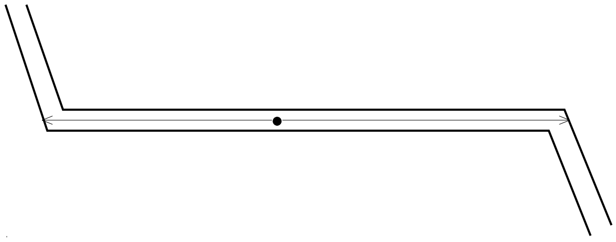

Figure 10: Only probes almost parallel to edge will hit a non-parallel edge.

As a general batch strategy, we shoot two probes in opposite directions along a line with orientation θ, and we shoot an additional probe along each line with orientation σ where a sweep line with orientation θ simultaneously hits two parallel, visible edges with orientation σ. For example, for the two models shown in Figure 11 we shoot two probes along a vertical line and three additional probes, all of which are indicated by rays in the figure. Each additional probe is parallel to a pair of parallel edges that are hit by a vertical sweep line.

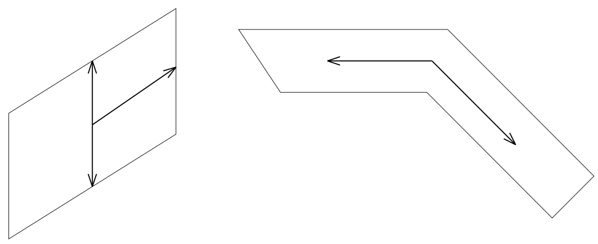

Figure 11: Rays indicate five probe directions chosen for two models.

This batch strategy guarantees that two non-parallel edges of a model will be hit. The first two probes along the line with orientation θ will intersect the model in a line segment of length l. Consider a sweep line with orientation θ that sweeps through each of the k models. Either the sweep line has only a finite number of places where it intersects a model in a line segment of length l, in which case the first two probes are sufficient, or the sweep line hits two parallel edges that are a distance l apart (where distance is measured along the sweep line). In this case, however, an additional probe was chosen parallel to these edges, and this probe together with the first two suffice to narrow down the number of possible placements of p.

In order to use a minimum number of probes, we must find an orientation θ for the initial two probes that minimizes the number of additional probes necessary. For each pair of mutually visible parallel edges of a model there is a cone of directions such that a pair of opposite probes with direction in this cone might hit both edges (see Figure 12). We can map these cones onto a circle and find the region with the least overlap to use for the orientation θ of the initial two probes.

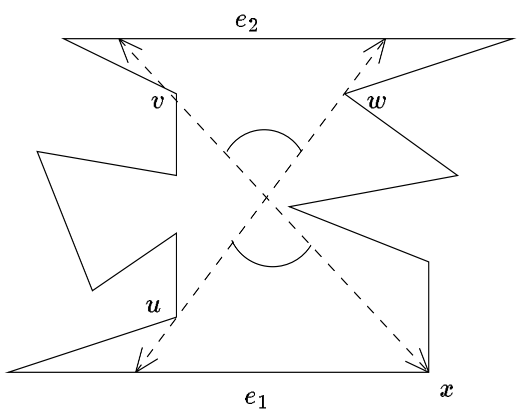

Figure 12: Parallel edges are visible within a cone of directions.

To map the cones onto a circle efficiently, we first compute for each model M_i its visibility graph in time $O(e_i)$, where e_i is the number of edges in the visibility graph [5]. Next we sort all of the visibility graph edges by slope in time $O(e \log n)$, where $e = \sum_{i=1}^{k} e_i$.

The *visibility graph* of a model M is a Euclidean graph whose vertices are the vertices of M and whose edges include the edges of M as well as edges between any two vertices of M that are visible to one another. For each pair of mutually visible parallel edges of M there are visibility graph edges whose slopes determine the ends of the cone within which the edges are visible. For example, in Figure 12 visibility edge $\overline{uw}$ determines one end of the cone in which e_1 and e_2 are visible, and $\overline{vx}$ determines the other end.

After sorting the visibility graph edges by slope we compute a trapezoidal diagram with orientation α for each model, where α is the slope of some visibility graph edge (see Figure 13(a)). The trapezoidal diagrams are computed in $O(n)$ time by extending rays from each vertex of a model M_i in the positive and negative α direction until they hit the boundary of M_i [2]. While computing the initial trapezoidal diagrams we keep track of the number and orientation of pairs of parallel edges that are visible along a line of orientation α.

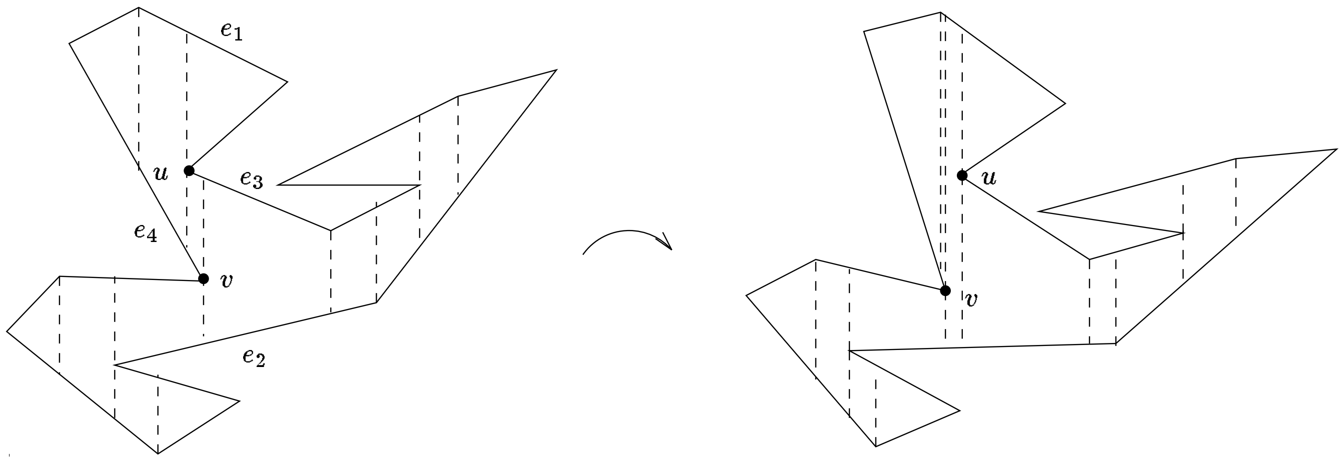

Figure 13: (a) A trapezoidal diagram (b) Rotating a trapezoidal diagram.

For every other slope β, in sorted order, we compute a trapezoidal diagram with orientation β for each model. Each new trapezoidal diagram can be computed from the previous one in constant time. For example, when we rotate the trapezoidal diagram in Figure 13(a) clockwise just beyond the direction through edge $\overline{uv}$, the trapezoidization changes, giving the diagram shown in Figure 13(b). In Figure 13(a) e_3 and e_4 are vertically visible (and e_1 and e_2 are not), but after the rotation e_1 and e_2 are vertically visible, while e_3 and e_4 are not. All other vertical visibilities in the diagram remain unchanged.

For each new rotation caused by a visibility graph edge, we update the number of different orientations for pairs of parallel visible edges. If this number is less than the minimum found so far, we store all these orientations. After we have rotated the trapezoidal diagram 360 degrees, we know the best orientation for the initial two probes, as well as how many additional probes are necessary and their directions.

5.2 Dynamic strategy

If no model has parallel edges, then the batch strategy of using two probes in opposite directions works as a dynamic strategy too.

When models have pairs of parallel mutually visible edges, the goal is to shoot enough probes to hit two non-parallel edges. We would like to do the probes *dynamically*, using information returned from each probe to determine the next one, so we use fewer probes than the batch strategy described earlier. The extra information that we use from a dynamic probe is the *distance* from the initial hit point p to the boundary of the polygon.

Similar to the batch strategy, we use a cone of directions for each pair of parallel mutually visible edges, such that a pair of opposite probes with direction in this cone might hit both edges. As in the batch strategy, we can find the region of least overlap of the cones efficiently.

We now preprocess this region to find a particular direction that will yield probes whose distances will determine the set of mutually visible parallel edges as uniquely as possible.

Consider the distance along a probe line between two parallel line segments ab and cd. Let the perpendicular distance between the two segments be D. The distance between the segments varies continuously between $D/\cos\theta_1$ and $D/\cos\theta_2$ (where θ_1 and θ_2 are the angles of the diagonals between the segments with respect to the perpendicular). The minimum is achieved when the probe line is perpendicular to the segments, and is D (see Figure 14).

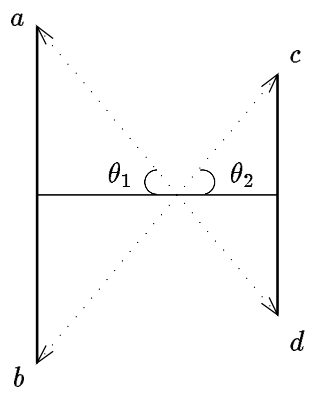

Figure 14: Distance between parallel lines varies continuously inside cone.

When plotted as a function of distance and θ, the distance curves of parallel line segments have nice properties. Curves for two different pairs of parallel line segments will coincide only when the pair segments are parallel in the same direction and are the same distance apart. Otherwise the curves can intersect at most twice.

To see this pictorially, consider a pair Q of parallel line segments $\overline{ab}$ and $\overline{cd}$ and another pair Q' of parallel line segments $\overline{a'b'}$ and $\overline{c'd'}$ (see Figure 15). We would like to determine when two opposite probes from p, made at an angle of θ from the horizontal, will strike the line segment pairs at the same distance. In this case, the pairs Q and Q' are indistinguishable at angle θ. By overlaying the pairs Q and Q' (see Figure 15) we observe that there are *exactly* two angles for which the pairs are indistinguishable, except for the case when the pair segments are parallel in the same direction *and* the same distance apart. Such a pair of parallel line segment pairs is called a *bad pair*.

When choosing a direction θ for our initial two probes, we need to avoid bad probe directions that yield equal distances for different pairs of parallel edges. Clearly there is nothing we can do for parallel edge pairs that are bad pairs, but since the number of choices for θ is infinite, we can always find a probe

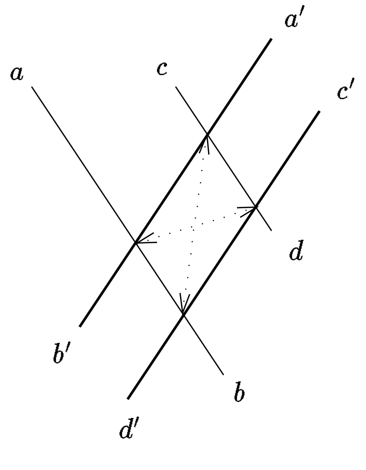

Figure 15: Why only two directions do not distinguish pairs that are not bad.

direction that will distinguish all pairs that are not bad pairs.

For our dynamic strategy we do two opposite probes initially in direction θ, then obtain the distance between the two boundary points hit by the probes, and finally do one extra probe in the direction parallel to the parallel edges that we might possibly strike. Note that when there are many pairs of edges the same distance apart, they are also parallel, and the final probe we shoot is parallel to them, so we ensure hitting at least two non-parallel edges. Therefore only three probes are necessary to pin down the number of possible placements of the initial hit point p to a finite number.

In $O(n^4)$ steps we can compute all the angles at which the distance between one pair of parallel edges is the same as the distance between another pair, since there are at most $O(n^2)$ parallel edge pairs, and each pair of parallel edge pairs yields two bad angles. This information will enable us to pick an appropriate angle θ that will distinguish all the pairs that do not form bad pairs.

6. CONCLUSIONS AND FUTURE WORK

We have examined the problem of identifying one of a set of polygonal models in the plane using point probes and finger probes. In particular, we have studied the problem of using a minimum number of finger probes emanating from an unknown interior point p of one of the models to determine a finite number of possibilities for the exact location of this interior point in one of the models.

For rectilinear models we have shown how to partition all possible outcomes of the finger probes into a small number of equivalence classes so that given the outcome of the probes from p, we can quickly find all the possible placements of p in a model. For simple polygonal models we have given both batch and dynamic strategies for determining a set of probes to localize p and have shown that a batch strategy may require $\Omega(n)$ probes, while a dynamic strategy never uses more than three probes.

In future research we will extend our strategies for simple polygonal models to quickly determine the set of all possible placements of p in a model. Other extensions include the case where there is more than one model in the scene, or where a model may be rotated or scaled. Additional future work includes implementing and testing the algorithms given here.

7. ACKNOWLEDGEMENTS

These results grew out of discussions that began in a series of workshops on Geometric Probing in Computer Vision, sponsored by the Center for Night Vision and Electro-Optics, Fort Belvoir, Virginia, and monitored by the U.S. Army Research Office. The authors thank Teresa Kipp and Vince Mirelli of the Center for Night Vision and Electro-Optics and all of the participants of the series of workshops. The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

Samir Khuller acknowledges support from NSF Research Initiation Award CCR-9307462. Joseph S.B. Mitchell acknowledges support from grants from Hughes Research Laboratories, Boeing Computer Services, Air Force Office of Scientific Research contract AFOSR-91-0328, and National Science Foundation Grants ECSE-8857642 and CCR-9204585. Kathleen Romanik acknowledges support from the IRIS National Network of Centres of Excellence, NSERC, and DIMACS. DIMACS is an NSF Science and Technology Center, funded under contract STC-88-09648, and also receives support from the New Jersey Commission on Science and Technology. Diane Souvaine acknowledges support from National Science Foundation Grant CCR-91-04732.

References

[1] E. M. Arkin, H. Meijer, J. S. Mitchell, D. Rappaport, and S. S. Skiena, "Decision Trees for Geometric Models", *Proc. of the* 9^{th} *Annual Symp. on Computational Geometry*, San Diego, ACM Press, pp. 369–378, May, 1993.

[2] B. Chazelle, "Triangulating a Simple Polygon in Linear Time", *Proc. of the* 31^{st} *Annual Foundations of Computer Science*, IEEE Computer Society Press, pp. 220–229, 1990.

[3] L. P. Chew and K. Kedem, "Some Results on Probing in the Plane", Working notes for the Workshop on Geometric Probing in Computer Vision, December 5 1991.

[4] J. Czyzowicz, I. Stojmenovic, and J. Urrutia, "Immobilizing a Shape in the Plane" in T. Shermer, ed., *Proc. of the* 3^{rd} *Canadian Conf. on Computational Geometry*, Vancouver, pp. 41–45, August, 1991.

[5] J. Hershberger, "An Optimal Visibility Graph Algorithm of Triangulated Simple Polygons", *Algorithmica*, 4:141–155, 1989.

[6] D. G. Kirkpatrick, "Optimal Search in Planar Subdivisions", *SIAM J. of Computing*, 12:28–35, 1983.

[7] L. Montejano and J. Urrutia, "Immobilizing Figures on the Plane" in T. Shermer, ed., *Proc. of the* 3^{rd} *Canadian Conf. on Computational Geometry*, Vancouver, pp. 46–49, August, 1991.