Window Queries On Planar Subdivisions Arising From Overlapping Polygons

Stuart A. MacGillivray* Bradford G. Nickerson†

Abstract

We present efficient algorithms for partitioning 2-dimensional space into faces arising from the boundaries of overlapping polygons. In particular, we examine the case where the partitioning arises from overlaying m k-sided simple polygons. A dynamic data structure is presented for storing the partitioning that arises from m polygons with at most k sides each in space $O(km^2)$ if the polygons are assumed to be convex, and space $O(k^2m^2)$ for simple polygons. Our structure returns T distinct regions from an axis-aligned orthogonal range search in worst case $O(mk + (m + k)(\log mk)T)$ time, and can be updated in $O(k^2m^2)$ time.

1 Introduction

Geometric range searching has been widely studied [2] [6] [7]. Range search on a variety of geometric primitives such as points, line segments, half spaces and simplices is of interest in many fields, with point range search being widely studied (see e.g. [1]) due to the direct connection with efficient database search. Here we consider the less studied problem of orthogonal range search on a planar subdivision induced by a set of intersecting simple polygons. Our motivation arises from a computational geometry viewpoint where we wish to maintain a collection of polygonal regions acting as an index to massive point datasets collected in overlapping regions at different times [5]. The dynamic data structure presented here supports efficient orthogonal range search on a set of faces created by an arrangement of intersecting simple polygons.

2 Definitions

We concern ourselves with planar subdivisions that arise from the intersection of m overlapping k-sided simple polygons.

We define a set P of simple polygons; if convexity is assumed, then these polygons are convex. $|P| = m$ by definition, and each polygon $p \in P$ is defined by at most k line segments. These line segments are referred to as the “sides” of the polygon they define. We further assume that these defining line segments are in general position such that at most two line segments touch any given point.

The sides defining the m polygons in P will likely (but not necessarily) intersect. Wherever two sides intersect, we cut both into smaller line segments, using the point of intersection as the endpoints for each resulting segment. Once all intersecting sides have been treated in such a fashion, we have a set E of line segments such that any endpoint is shared by either 2 or 4 line segments, and no two line segments cross. Each of these segments is a part or whole of one side of a polygon in P.

Finally, we define a set R of faces bounded by the segments in E. Each face is a polygon, but not necessarily a simple one, as faces can contain holes. An example of a face with a hole is shown in Fig. 5 (e). Each face $r \in R$ is thus, by definition, a single component which can include holes. For clarity, when we refer to an element of R, we will use the term 'face'; the term 'polygon' is restricted to the original elements of P, even though many faces are themselves simple polygons.

We explore the use of window queries on these planar subdivisions, which report all faces that partially or entirely overlap an axis-aligned rectangle W.

3 Geometry

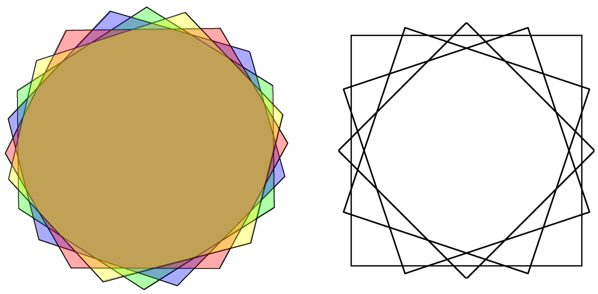
Theorem 1 *A planar subdivision composed of m k-sided convex polygons can be decomposed into $O(km^2)$ segments and $O(km^2)$ faces.*

Proof. Each of the mk sides that define the m convex polygons in P can, by the definition of convexity, intersect each polygon at most twice. This means that when the mth polygon is added, each side defining it passes through at most $O(m)$ existing faces and splits them. Therefore, adding the mth polygon creates at most an additional $O(km)$ faces; summing over m polygons gives $O(km^2)$ faces. As shown in Figure 1a, each side will intersect $O(m)$ other sides, and decompose into $O(m)$ segments; summation again gives $O(km^2)$ total segments. □

Theorem 2 *A planar subdivision composed of m k-sided simple polygons can decompose into $O(k^2m^2)$ faces.*

*Faculty of Computer Science, University of New Brunswick, t172q@unb.ca

†Faculty of Computer Science, University of New Brunswick, bgn@unb.ca



(a) (b)

Figure 1: Convex cases. (a) Example of a set of $m = 4$, $k = 6$-sided convex polygons, constructed to maximize the number of segment intersections; $|E| = 168$. (b) Example of a set of $m = 4$, $k = 4$-sided polygons, constructed to apply the worst case from Figure 5 (d) to the outside face for all mk vertices.

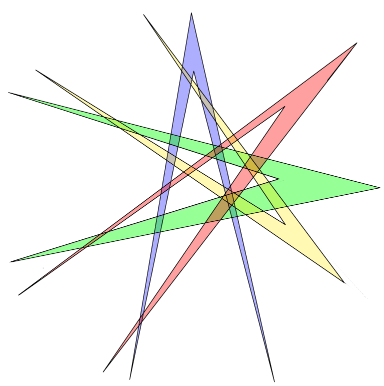


Figure 2: Example of a set of $m = 4$, $k = 4$-sided polygons, constructed to maximize the number of segment intersections; $|E| = 208$.

Proof. Every side of the mth polygon can pass through as many as $k(m-1)$ other sides. Adding the mth polygon can create at most an additional $O(k^2m)$ faces; summing over m polygons gives $O(k^2m^2)$ faces. As shown in Figure 2, each side can cross every side from every other polygon, and decompose into $O(km)$ segments for a total of $O(k^2m^2)$ segments. □

How much complexity can be added to a given face from the addition of one more convex polygon? Examples can be constructed where a single line adds $O(m)$ sides to the face, as shown in Figure 3, but these are special cases. It is possible to add $O(k)$ sides by the addition of a k-sided polygon, as shown in Figure 4, and this scenario may be repeated as many times as desired.

Theorem 3 *Given a planar subdivision composed of m k-sided overlapping simple polygons, a single face can have $O(km)$ line segments.*

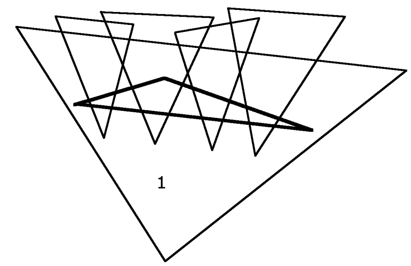


Figure 3: Example of a set of $m = 5$, $k = 3$-sided polygons, constructed such that the addition of the bolded polygon results in face 1 going from 12 sides to 21, in a fashion that is an increase of $O(m)$.

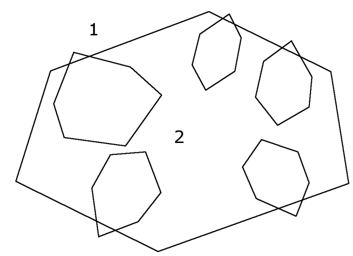


Figure 4: Example of a set of $m = 6$, $k = 6$-sided polygons, constructed such that each additional polygon adds $O(k)$ sides to face 2.

Proof. An arrangement of m k-sided simple polygons consists of km line segments, any pair of which can cross at most once. Furthermore, by the assumption of general position, any given line segment s can touch at most $2m + 2$ other line segments, i.e. the two segments adjacent to s on the polygon where s resides, and two interactions from every other polygon.

A vertex of a polygon can add at most a constant number of sides to a single face. Figure 5 shows the general scenario for the number of sides added to a face by the presence of a single vertex. The worst case for any given vertex is the scenario shown in part (d) of Figure 5, where the presence of one vertex results in three additional sides being present in the face. Figure 1b shows that this scenario can be repeated for every vertex of every polygon, even with the assumption of convexity.

As illustrated in Figures 5 and 1b, each vertex contributes at most 3 segments to any face. Conversely, every segment in a face can correspond to one of the mk vertices; the $O(m)$ segments added by a single additional side in Figure 3, for instance, are examples of vertices transforming from Figure 5c cases to 5d ones.

The planar subdivision is induced by the m polygons of P, each defined by k vertices, so a single face has at most $3mk = O(mk)$ segments defining its boundary. □

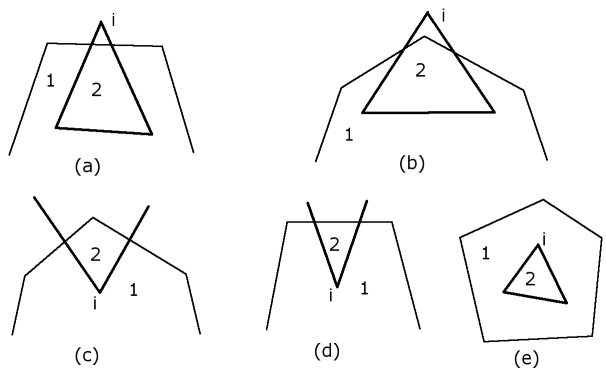


Figure 5: The possible cases for additional line segments added to face 1 by given vertex i. (a) Vertex outside face; both edges cut the same edge of the face adding 2 edges. (b) Vertex outside face; each edge cuts a different edge of the face adding 1 edge. (c) Vertex inside face; each edge cuts a different edge of the face adding 2 edges. (d) Vertex inside face; both edges cut the same edge of the face adding 3 edges. (e) Vertex inside face; neither edge cuts an edge of the face. Each such vertex adds 1 edge to the face.

Theorem 4 *Given a planar subdivision composed of m k-sided overlapping polygons, the average number of segments per face is $O(k)$.*

Proof. We start with a lower bound for the worst case. Let the m polygons of P, each of which has k sides, be disjoint. This gives us $|E| = mk$, $|R| = m + 1$, and as m approaches infinity $\frac{|E|}{|R|} = k$.

Asymptotically, the worst case cannot be worse than k unless $k \leq 4$. This proof is by induction.

Our base case is a subdivision where $|P| = 1$. As the single polygon does not intersect any others, we have $|E| = k$ and $|R| = 2$. The two faces are the interior and exterior of the single polygon. For every polygon after the first, we have the following scenarios.

We add a new k-sided polygon p to a set of faces R. For every face r that contains some portion of the sides of p, we create at least 1 new face as r is separated into the interior and exterior of p. As at least one face must contain any given vertex of p, we add at least 1 face to R and k line segments to E.

The sides of p can intersect other line segments in E. At every point where a newly added segment crosses an existing one, we cut both segments into four non-crossing segments sharing a common vertex, and increase $|E|$ by 2. There are limitations on how this can occur. Every face r is bounded by line segments which we call "contours". We define a contour as a closed loop that describes either the boundaries of the outside of r or some island within it. By definition, these contours are disjoint. If the sides of p cross a contour of r at all, they must cross that contour an even number of times since the sides of p themselves form a closed loop.

Since the number of intersections between p and any given contour of r is even, we look at those intersections in pairs. Every contour of r can have one pair of intersections with p without creating additional faces beyond the previously defined "interior of p within r". For every pair of intersections between p and any given contour of r beyond the first, an additional face arises. Further intersections with a given contour will add a set of line segments to cut some subset of r into an additional face. If face r has c distinct contours, the maximum number of new line segments that creates the minimum number of faces is $4c$; i.e. two intersections creating two new segments each for every one of the c contours of r, while separating r into two faces.

Each contour is disjoint, and different faces must be associated with each one, so the edges of p pass into at least c other faces in R in this scenario, creating at least c new faces. The sum total of the contributions caused by the addition of p is thus $c + 1$ new faces, and $k + 4c$ new line segments. As c increases, the ratio of new segments to new faces approaches 4.

If the sides of p intersect a contour of r more than twice, for instance $2h$ times, we obtain $4h$ new line segments, $h - 1$ new faces cut from r, and at least h new faces from the opposite side of the contour, possibly more. As the number of intersections h increases, the ratio of new edges to new faces approaches 2.

The ratio of segments to faces, $\frac{|E|}{|R|}$, starts at $\frac{k}{2}$ in the base case with $|P| = 1$. The ratio of new segments to new faces at each step is at most $\frac{k}{1}$, with additional intersections with contours of faces adding at most 2 or 4 segments for every additional face. As m increases, $\frac{|E|}{|R|}$ cannot be greater than $\frac{mk}{m+1} = O(k)$. □

Lemma 5 *Two faces i, j from a planar subdivision composed of m k-sided overlapping simple polygons can only share line segments if there is exactly one polygon p for which i is a subset of p and j is disjoint from p, or vice versa.*

Proof. Given a face $i \in R$, for each of the m polygons $p \in P$, i is either a subset of or disjoint from p. Every line segment of i is derived from one of the sides of one of the polygons in P, and is either contained or excluded by all other polygons. Suppose there exists a line segment ℓ that is part of the boundaries of faces i and $j \in R$. There exists some polygon p for which ℓ is a portion of one of the sides of p. Thus, if i and j are different faces, one must be a subset of p and the other must be exterior to p. For every other polygon $q \neq p \in P$, if ℓ is contained by q then i and j must be subsets of q. Conversely, if ℓ is not contained by q, then i and j are disjoint from q. □

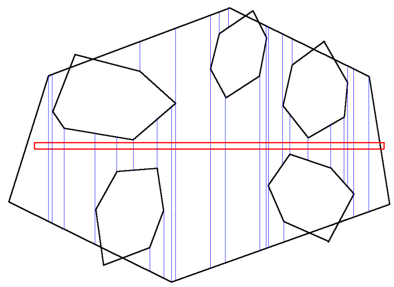


Figure 6: Example of a set of $m = 6$, $k = 6$-sided polygons, the $O(mk)$ trapezoids from the most complex face, and a rectangular query that hits a majority of them but only one line segment.

Theorem 6 *Two faces from a planar subdivision composed of m k-sided overlapping simple polygons have at most $m + k - 1$ common line segments.*

Proof. From Lemma 5, two faces $i, j \in P$ can only share line segments if there is exactly one polygon p for which one face is a subset and the other is disjoint. Any shared line segments will be drawn from the sides of this single polygon p. Furthermore, the k sides of p can be cut by other polygons at most m times to create the border between two faces. Figure 1b shows that while a polygon can cut multiple edges and add up to $O(k)$ additional line segments, doing so will create additional faces that use some of those segments as their boundaries; conversely, Figure 4 shows how the boundary between faces 1 and 2 can be cut m times. As shown in Theorem 4, any polygon that cuts a contour of a face i more than twice separates a portion of i into a new face. No polygon can have its sides belonging to more than one contour of i. Thus, any polygon cutting the edges of p more than twice will separate a portion of the interior or exterior of p to become a new face, which will have the newly cut segment as a boundary. As such, the sides of p shared by i and j can only be broken m times by the other $m-1$ polygons of P without creating a portion separate from i and j, meaning i and j can share at most $m + k - 1$ line segments.

□

4 Structure

Our initial design was an adaptation of the trapezoidal map [8], which can be used for point location in $O(\log n)$ time and $O(n \log n)$ space for n line segments defining the set of polygons. We know from Theorem 2 that the intersection of m k-sided polygons can have at most $O(k^2m^2)$ line interactions, and an equal number of line segments. More efficient algorithms exist for point location [9] [4] [3], but our problem requires a window query.

Our query is only concerned with reporting which faces intersect the query window. Any given face can consist of a large number of trapezoids. The worst case has a face consisting of $O(km)$ vertices, which decomposes into an equal number of trapezoids. Figure 6 shows an example of such a scenario. This would give an overall query time of $O(\log(k^2m^2) + kmT)$, where T is the number of faces reported.

One simple alternative is to keep a list of all faces, and check intersection of each one against a query rectangle. As Theorem 4 shows, there are $O(k)$ segments per face on average; as such, this technique can be completed in $O(k|R|)$ time. While less than $|E|$, this still requires $O(k^2m^2)$ time for simple polygons.

Our structure has two major components. One is a standard planar point location structure used when no segments of the query rectangle W intersect any segments in E. The other is an index for line segments that supports range searching. While performance of this may vary, the nature of this problem allows us to avoid having to index, and later modify, the $O(km^2)$ or $O(k^2m^2)$ line segments that bound the planar subdivision. Instead, we simply index the km sides of the original polygons in P. While these km segments may cross and will share endpoints, we reduce the cost of the spatial index.

Each of the km line segments includes a pointer to a secondary structure. The jth side of polygon i has a range index G_i^j, which consists of the coordinates of the side's endpoints, and three dynamic arrays. As illustrated in Figure 7, one array has fractional values from zero to one indicating the distance along the side where intersections with other sides occur. This array is an ordered list of the segments intersecting side j, which is of length $O(m)$ in the convex case and $O(km)$ in the simple polygons case. The other two arrays contain indices for the faces that include each line segment, respectively those internal and external to polygon i. By recording face indices in this manner, every line segment of the planar subdivision is represented once, and the secondary structures require $O(km^2)$ or $O(k^2m^2)$ space in the convex or simple polygons cases, respectively.

By using a sorted list, the n pieces of a line segment can be compared with a query rectangle in $O(\log n + T)$ time, where T is the number of pieces returned. We know that no line segment of a polygon can be divided into more than $O(m)$ pieces if the polygons are convex, and $O(km)$ if they are not.

Our main obstacle is over-reporting. In the worst case, a query rectangle W could intersect all of the segments bounding a face, effectively reporting it $O(km)$ times in the worst case. Unlike a trapezoidal map, every segment belongs to two faces. As Theorem 6 shows, two faces can share at most $O(m+k)$ line segments. If a face with $O(mk)$ segments has all its segments intersecting W, some number of these segments will belong to other faces entirely. Averaging the number of segments over the number of faces returned gives an over-reporting factor of $O(m+k)$.

In the event that a query rectangle neither intersects nor contains any line segments, the entire query rectangle is contained inside a single face. The problem reduces to planar point location; a query that intersects no segments can be effectively reduced to a single point, as we need only locate the face that contains some point from the query region. Edelsbrunner et al's [3] worst case optimal linear space algorithm, for example, can perform planar point location in logarithmic time.

Theorem 7 *Orthogonal range search on a planar subdivision arising from m k–sided overlapping simple polygons can report the T faces in range in $O(mk + (m+k)(\log mk)T)$ worst case time.*

Proof. As illustrated in Algorithm 1, we can perform a linear scan of the mk sides of all polygons $p \in P$ in $O(mk)$ time. For each side, a segment intersection test will determine whether any segment from that side intersects a query rectangle W. If no sides intersect W, point location can be performed using a layered dag [3] in $O(\log mk)$ time. Otherwise, the algorithm continues. The range indices G_i^j are sorted, so appropriate segment information can be located in $O(\log mk)$ time, returning a minimum of one segment. As shown in Theorem 6, no two faces can share more than $O(m+k)$ segments. In the worst case, each of the T faces includes $O(m+k)$ segments intersecting W, where each segment requires $O(\log mk)$ time to find in its range index. □

The time cost of over-reporting cannot be easily avoided; any description of a face would require information at least equal to the number of segments defining it. In practice, we may wish to report any intersected face only once. To accomplish this, Algorithm 1 makes use of a "face list" L. We consider two possibilities. The first is a simple array of boolean flags of size $|R|$; while this requires time $O(|R|)$ to build and scan, we can still avoid requiring all the information from each face, and gain an advantage over the naïve solution. The second is a balanced tree of face indices that have been located; each of the $O((m+k)T)$ indices we locate can be checked against the tree and inserted in $O(\log T)$ time; as $T \leq |R| = O(k^2m^2)$, this can be done in time $O(\log mk)$ per segment which maintains the range search cost of Theorem 7.

For Update Algorithm 2, faces are redefined to consist of multiple disjoint components. Two components are part of the same face if both components are contained in the same set $Q \subseteq P$ of polygons (see e.g. faces

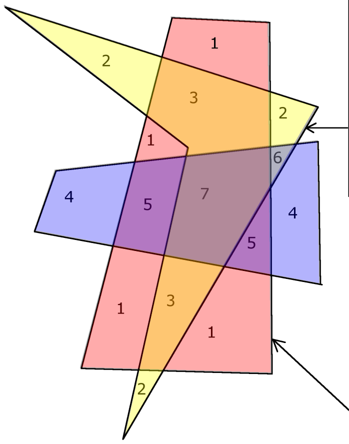
Algorithm 1: Search(W, S)

Input : A query rectangle W, a polygon overlap index structure S
Output: A set of faces L

```
begin
    Set up an empty list of faces L;
    Perform a range search over W on the km sides in S;
    for Each line segment i intersecting W do
        Use the endpoints of i to determine what length interval [a, b] is contained in W;
        Perform a binary search on the secondary structure of i to find the entry at length a;
        for Each entry g of G_p^i between lengths a and b do
            Insert both internal and external regions from g into L;
    if L is empty then
        Use planar point location to determine what face s contains the top left corner of W;
        Return s;
    else
        Return L;
```

1 through 5 in Figure 7). This allows indexing to be simplified; upon adding a polygon q to the structure S, we construct a mapping M from existing face indices to new indices for faces covered by q. We consider either a deterministic algorithm using binary arithmetic (adding 2^x to the index of any face covered by the xth polygon), or an array of size $|R|$ to be filled in incrementally, to ensure that every face index pertains to some portion of the plane. Algorithm 2 makes no assumptions about which mapping is used; only that there exists a method that takes an existing face index number and returns another.

The layered dag [3] is not dynamic, but can be built in time proportional to the number of segments as long as they are sorted by their leftmost points. We can maintain such a sorted list by inserting each of the $O(mk^2)$ new intersections into the sorted list in logarithmic time, for a total insertion cost of $O(mk^2 \log(mk))$; rebuilding the layered dag would then require $O(m^2k^2)$ time. Each of the mk sides can have up to k new indices in the worst case, for a similar insertion cost of $O(mk^2 \log(mk))$. This is worst-case optimal; the insertion algorithm also requires indices covered by the new polygon to be updated. In the worst case, a new polygon could cover all $O(m^2k^2)$ existing faces, requiring all of them to have their indices modified.



0	2	0
.1	6	4
.2	7	5
.5	3	1
.8	2	0

0	1	0
.20	3	2
.36	7	6
.45	5	4
.71	1	0

Figure 7: Example of a set of $m = 3$ $k = 4$-sided polygons, with two polygon sides having their range indices shown. Columns are, from left to right, fractional distance from the top corner, internal face, external face.

5 Conclusions

We present an algorithm for orthogonal range search on a set R of faces arising from m k–sided overlapping simple polygons. The T faces in range are reported in $O(mk+(m+k)(\log mk)T)$ worst case time, and the data structures supporting the algorithm require $O(k^2m^2)$ space. Updating the data structure supporting this algorithm to add a new simple polygon requires $O(k^2m^2)$ time. To our knowledge, ours is the first paper to address the complexity of faces arising from overlapping simple polygons, and to provide an efficient algorithm for orthogonal range search on their planar arrangement.

Algorithm 2: Update(S, p)

Input : A polygon overlap index structure S, a k-sided polygon p
Output: An updated index structure S

begin
  Create a mapping M from existing face indices to unused ones.;
  Create a range index G_p^i for each of the k sides of p, initialized with null indices;
  for *Each side $j \in S$, from existing polygon q* **do**
    if *j does not intersect p* **then**
      Do nothing to j and continue;
    else if *j is entirely contained by p* **then**
      Replace all face indices in G_q^j in accordance with mapping M;
    else
      Determine what sides i of p are intersected by j, and where;
      For each intersection, add new entries to G_q^j, copying from the adjacent entry;
      In all ranges of j contained by p, replace entries with new indices from M;
      Add entries to G_p^i for intersections between i and j, and copy entries from G_q^j to G_p^i;
  for *Each side j of p with null entries* **do**
    Use planar point location to determine which face s an endpoint of p is in;
    Set the external entry of j to s, and the internal side to the mapping M of s;
  Add all k sides of p to S;
  Reconstruct the layered dag supporting planar point location;

References

[1] P. Afshani, L. Arge, and K. D. Larsen. Orthogonal range reporting: query lower bounds, optimal structures in 3-d, and higher-dimensional improvements. In J. Snoeyink, M. de Berg, J. S. B. Mitchell, G. Rote, and M. Teillaud, editors, *Symposium on Computational Geometry*, pages 240–246. ACM, 2010.

[2] P. K. Agarwal and J. Erickson. Geometric range searching and its relatives. *Contemporary Mathematics*, 223:1–56, 1999.

[3] H. Edelsbrunner, L. Guibas, and J. Stolfi. Optimal point location in a monotone subdivision. *SIAM Journal on Computing*, 15(2):317–340, 1986.

[4] D. Kirkpatrick. Optimal search in planar subdivisions. *SIAM Journal on Computing*, 12(1):28–35, 1983.

[5] S. A. Macgillivray and B. G. Nickerson. A data structure supporting exclusion persistence range search. In *Proc. of CCCG*, pages 29–34, 2012.

[6] J. Matoušek. Geometric range searching. *ACM Comput. Surv.*, 26(4):422–461, Dec. 1994.

[7] D. M. Mount. Handbook of discrete and computational geometry, 2nd ed. In J. E. Goodman and J. O'Rourke, editors, *Geometric Intersection*, chapter 38, pages 857–876. Chapman & Hall, 2004.

[8] F. P. Preparata. A new approach to planar point location. *SIAM Journal on Computing*, 10(3):473–482, 1981.

[9] N. Sarnak and R. E. Tarjan. Planar point location using persistent search trees. *Communications of the ACM*, 29(7):669–679, 1986.