

# Voronoi Diagrams of Line Segments Made Easy\*

## (Extended Abstract)

Christopher M. Gold<sup>†</sup>  
Faculté de Foresterie et de Géomatique  
Université Laval  
Québec, Canada

Peter R. Remmele<sup>\*\*</sup>  
Department of Computer Science  
Federal Institute of Technology  
Zürich, Switzerland

**Thomas Roos**  
Department of Computer Science  
Federal Institute of Technology  
Zürich, Switzerland

### Abstract

This paper presents a new method for constructing Voronoi diagrams of  $n$  disjoint line segments. The basic idea is as follows. We construct a point, Voronoi diagram by selecting one of the two endpoints of each line segment. Then we incrementally expand each endpoint to its corresponding line segment, continually maintaining the Voronoi diagram using kinematic Voronoi diagram methods.

The technique is simple and easy to implement and runs in worst-case time  $O((n + t) \log n)$  using  $O(t)$  space, where  $t$  is the number of topological events that appear during all expansions. Experimental results indicate that the number of topological events is roughly linear in the number of line segments with an additive sublinear factor depending on the maximal length of the line segments. In practice our algorithm even shows a linear or slightly superlinear running time.

## 1 Introduction

The Voronoi diagram is a fundamental data structure appearing in many variants in the Computational Geometry literature (cf. [1] for a survey). The focus of this paper is the Voronoi diagram of disjoint line segments in the Euclidean plane (cf. Figure 1), first investigated by Drysdale and Lee [5], and Kirkpatrick [12]. Nearly all computing paradigms apply to the construction of Voronoi diagrams: there are randomized incremental algorithms [3], divide & conquer methods [16], and sweepline techniques [6] which solve this problem in optimal  $O(n \log n)$  time and  $O(n)$  space where  $n$  is the number of line segments. A recent compilation of the experimental behavior of these algorithms can be found in [4]; a numerically robust,  $O(n^2)$  implementation is given in [11].

This paper presents a new technique for constructing Voronoi diagrams of disjoint line segments which is based on kinematic Voronoi diagram methods [2, 9, 13, 14]. We first construct a point

---

<sup>†</sup>The authors gratefully acknowledge the support by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Association de l'Industrie Forestière du Québec, and the Swiss National Science Foundation (SNF) under grant 21-39328.93.

<sup>†</sup>Address: Centre de recherche en géomatique, Pavillon Casault, Université Laval, Ste-Foy, Québec, Canada G1K 7P4. Email: cgold@vm1.ulaval.ca

<sup>\*\*</sup>Address: Dept. Informatik, ETH Zentrum, CH-8092 Zürich, Switzerland. Fax: +41 1 632 1172. Email: (remmele, roos)@inf.ethz.ch

Voronoi diagram by selecting one endpoint of each line segment. Then we maintain the Voronoi diagram while expanding each endpoint to its corresponding line segment. Since only one point is expanding at a time the topological changes of the line segment Voronoi diagram are basically the same as those of Voronoi diagrams of one moving point. [13]. After the expansion of all sites, we obtain the Voronoi diagram of the given line segments. The presented technique is not only simple and efficient, but also easy to implement.

The paper is organized as follows. Section 2 describes the algorithm in more detail. Section 3 discusses some experimental results obtained from all implementation. Finally, some open problems and extensions of this technique can be found in the Conclusions.

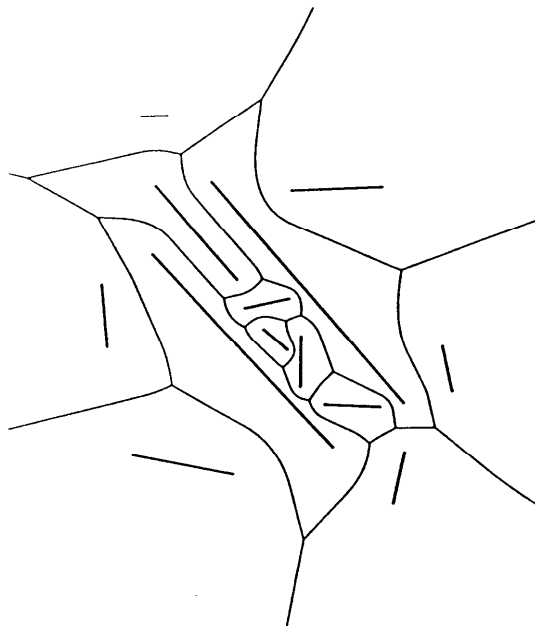


Figure 1: A Voronoi diagram of line segments.

## 2 The Algorithm

Given a set of  $n$  disjoint line segments  $S = \{s_1, \dots, s_n\}$  in the Euclidean plane, the *Voronoi* diagram  $V(S)$  partitions the plane into Voronoi regions such that all points within a Voronoi region  $V_s$  lie closer to the line segment  $s \in S$  than to all other line segments in  $S - \{s\}$  (see Figure 1). It is well-known that the complexity of the Voronoi diagram  $V(S)$  is  $O(n)$  [1]. Using these elementary definitions, we are ready to describe the basic idea of our algorithm:

- (1) We first select one endpoint  $e_i$  of each line segment,  $s_i \in S$  and construct the Voronoi diagram  $V(\{e_1, \dots, e_n\})$  of these endpoints.
- (2) In the second step, we expand each endpoint  $e_i$  one at a time constructing the original line segment  $s_i$ . While doing this we maintain the Voronoi diagram with the help of kinematic Voronoi diagram methods.

We can perform the first step using any of the existing  $O(n \log n)$  algorithms for point Voronoi diagrams. The second step can be thought of as expanding a rubber band which is first totally contracted in point  $e_i$  and then expanded to the line segment  $s_i$  by dragging the other end (see Figure 2 for an elementary expansion)

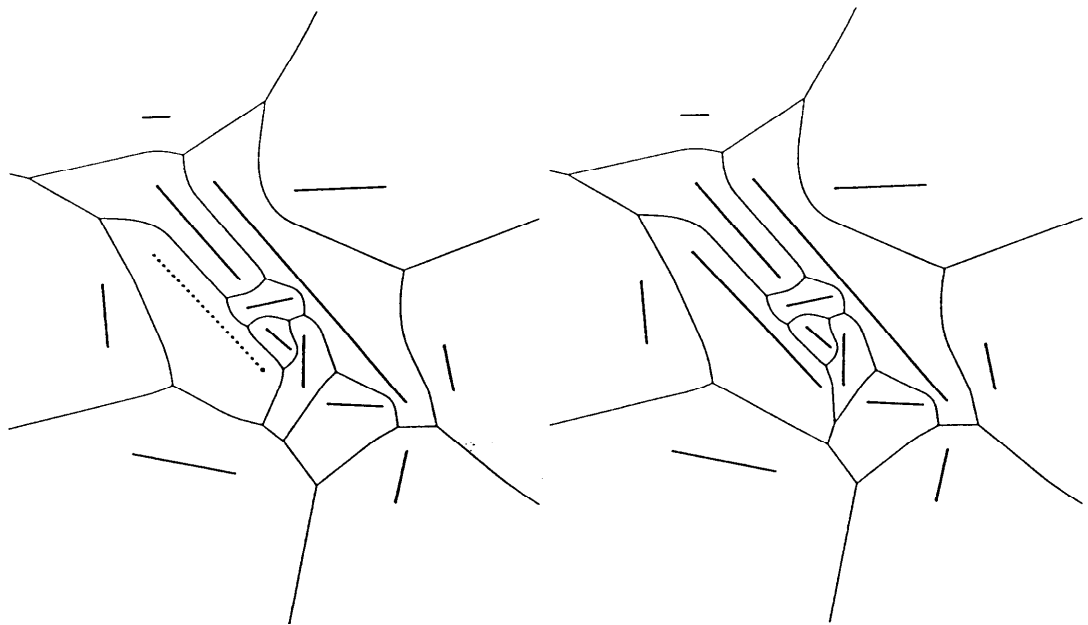


Figure 2: A Voronoi diagram during an expansion.

We can consider the (one-by-one) expansion of each line segment as the motion of one point in a Voronoi diagram of points and already expanded line segments. This problem has been thoroughly investigated (cf. [13]). Since only one line segment is expanding at a time, the topological changes are basically the same as those of Voronoi diagrams of one moving point.

To understand the **topological events**, i.e. the changes of the topology of the Voronoi diagram during an expansion, we consider the **topological structure**, i.e. the (extended) Delaunay graph. It is well-known that topological events can be described as **swaps** [9] of adjacent triangles in the topological structure. Each topological event can be treated in worst-case optimal  $O(\log n)$  time. In the average this reduces to  $O(1)$  time since the average number of Voronoi neighbors at each instant of time is constant. In our setting the topological events are even simpler since the expansion of a line segment creates only a certain type of topological event. This is, because the expanding line segments only run into (but never out of) circumcircles corresponding to existing Delaunay triangles. With that, we obtain the following theorem:

**Theorem 1** *A Voronoi diagram of  $n$  disjoint line segments can be constructed in worst-case time  $O((n + t) \log n)$  and  $O(n)$  space where  $t$  is the number of topological events that happen during the expansion of all line segments.*

Our algorithm runs in worst-case optimal  $O(n \log n)$  time if  $t \in O(n)$ . This is true for sufficiently short line segments due to continuity arguments from the point situation. In fact, during all our experiments we even observed a linear or slightly superlinear running time.

The **robustness** of the presented algorithm depends on the correctness of the topological structure over time. However, changes in this structure only take place during a topological event, caused by the expansion of a line segment. The topological swap itself is not numerically unstable, but the calculation of the precise instances of these events is crucial to the robustness of the algorithm. This calculation can be performed by computing the zeros of the Incircle determinant [10]. For a single moving point this gives us a quadratic equation to solve which is defined by the **incircle** determinant. Thus, we can achieve numerical robustness of the algorithm by making these computations numerically stable. For that, we apply recent results by Sugihara and Iri [15] that show how to improve the numerical robustness of this determinant.

### 3 Experimental Results

We have implemented our algorithm on the basis of the quad-edge data structure by Guibas and Stolfi [IO]. First, we select one endpoint of each line segment at random and use a divide-&conquer technique for computing the Voronoi diagram of these endpoints in worstcase time  $O(n \log n)$  and  $O(n)$  space. Then we incrementally expand the line segments according to the lexicographical order of their selected endpoints. We use a balanced binary tree to maintain the order of the topological events in worst-case time  $O(\log n)$  per event. Each swap operation can be performed in additional  $O(1)$  time.

We have implemented our algorithm in C on a Sun SPARCstation 20. For each problem size,  $n = 100, 200, 500, 1000$ , ten random test inputs were generated (see Figure 3 for a Voronoi diagram of 500 line segments and points). These inputs were generated in two different ways. Method A chooses for each line segment a starting point and an endpoint uniformly distributed over the unit square  $\{(x, y) \mid 0 \leq x, y \leq 1\}$  until the new line segment intersects with no other line segment. Opposite to that, Method B first chooses a starting point uniformly distributed over the unit square for each line segment. Then it randomly selects an endpoint from the unit square until there is no intersection between the new line segment and any other already inserted line segment.

The following tables show the experimental results. We measured the average running time and the average number of processed swaps over ten randomly generated test inputs. The first group of tables (Table 1A and 1B) lists the results over varying problem sizes and in the second group of tables (Table 2A and 2B) we vary the maximal length of the generated line segments for constant problem size. Each of the two groups consists of a Table A and a Table B according to the method of input generation.

Table 1A and 1B both show a pairwise linear or slightly superlinear correlation between the number of processed topological events ( $t$ ) and the running time ( $time$ ) with respect to the number of line segments ( $n$ ). This is due to the fact that topological events can be treated in  $O(1)$  time in the average as noted earlier.

Table 1A		
<b>n</b>	<b>t</b>	<b>time</b>
100	130	0.94 sec
200	342	2.16 sec
500	1080	5.87 sec
1000	2249	11.59 sec

Table 1B		
<b>n</b>	<b>t</b>	<b>time</b>
100	128	0.93 sec
200	360	2.13 sec
500	1121	5.75 sec
1000	2275	11.22 sec

Tables 1 A, 1B: Topological events and running time over varying problem size.

The number of topological events is also influenced by the length of the line segments. Table 2A and 2B indicate a sublinear behavior of the number of topological events ( $t$ ) and the running time (**time**) with respect to the maximal length (**maxl**) of the line segments. The number of line segments has been fixed to  $n = 100$ .

Table 2A		
<b>maxl</b>	<b>t</b>	<b>time</b>
0.06	46	0.78 sec
0.12	81	0.83 sec
0.24	167	1.04 sec
0.48	249	1.19 sec

Table 2B		
<b>maxl</b>	<b>t</b>	<b>time</b>
<b>0.06</b>	51	0.79 sec
0.12	96	0.85 sec
0.24	175	1.07 sec
0.48	248	1.15 sec

Tables 2A, 2B: Topological events and running time over varying segment, length.

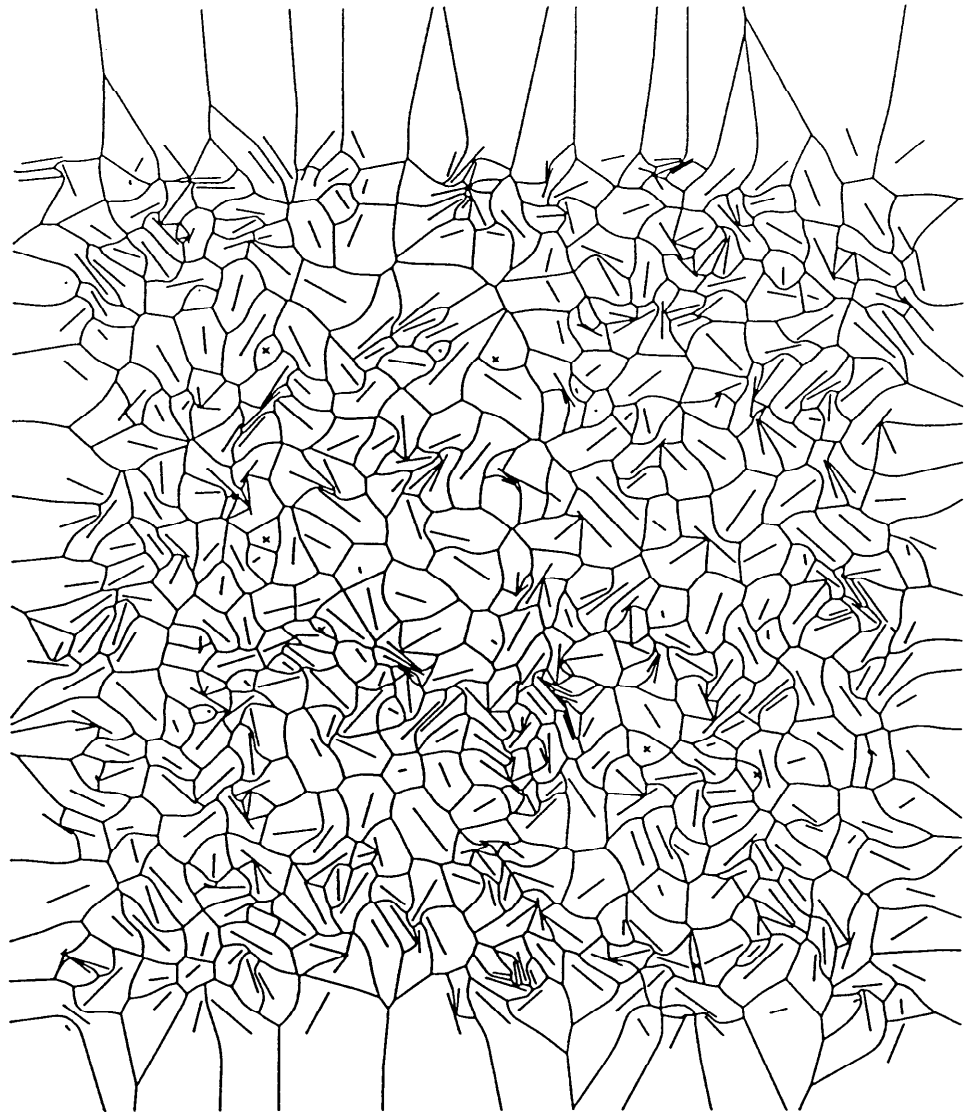


Figure 3: A Voronoi diagram of 500 line segments and points.

## Conclusions and Remarks

In this paper, we have presented a new technique for constructing Voronoi diagrams of line segments by combining Voronoi diagrams of points and kinematic Voronoi diagrams. Our algorithm is not **only** simple and efficient, but also easy to implement and **can** be made numerically robust, as it performs only local operations on the topological structure. One major advantage is the absence of a point location structure which is inherent to all incremental algorithms. Another advantage is the simplicity of the expansion step during which only simple topological events have to be processed. Although the theoretical worst-case running time of our algorithm is  $O(n^2 \log n)$  (due to a quadratic bound on the number of topological events given in [13]), we are confident that we will be able to improve the expected time in a randomized setting, when randomizing over the sequence of expansions.

There are applications of this technique, e.g., in GIS (cf. [7, 8]). Our technique can easily be extended to more general objects such as polygons and, with some care, even to general subdivisions. Finally, we are confident that Voronoi diagrams of “complex” 3-dimensional objects can be constructed in the near future, as the basic ingredients are already available [2].

## References

- [1] F. Aurenhammer, *Voronoi diagrams: a survey of a fundamental geometric data structure*, ACM Comput. Surv., Vol. 23, pp 345-405, 1991
- [2] G. Albers and T. Roos, **Voronoi diagrams of moving points in higher dimensional spaces** Proc. 3<sup>rd</sup> Scandinavian Workshop on Algorithm Theory SWAT'92, Helsinki, Finland LNCS 621, pp 399-409, 1992
- [3] J.-D. Boissonnat, O. Devillers, R. Schott, M. Teillaud, and M. Yvinec, *Applications of random sampling to on-line algorithms in computational geometry*, Discrete & Computational Geometry, Vol. 8, pp 51-71, 1992
- [4] C. Burnikel, K. Melhorn, and S. Schirra, *How to compute the Voronoi diagram of line segments: theoretical and experimental results* Proc. 2<sup>nd</sup> Annual European Symposium on Algorithms ESA'94, LNCS 855, pp 227-239, 1994
- [5] R.L.S. Drysdale and D.T. Lee, **Generalized Voronoi diagrams in the plane** Proc. 16<sup>th</sup> Annual Allerton Conf. Commun. Control Comput., pp 833-842, 1978
- [6] S.J. Fortune, **A sweepline algorithm for Voronoi diagrams**, Algorithmica, Vol. 2, pp 153-174, 1987
- [7] C.M. Gold, **Spatial data structures: the extension from one to two dimensions**, in: L.F. Pau (Ed.), Mapping and spatial modelling for navigation, NATO ASI Series F, No. 65, Springer, Berlin, pp 11-39, 1990
- [8] C.M. Gold and T. Roos, **Surface modelling with guaranteed consistency - an object-based approach**, in J. Nievergelt et al (Eds.): Proc. International Workshop on Advanced Research in Geographic Information Systems IGIS'94, LNCS 884, pp 70-87, 1994
- [9] L.J. Guibas, J.S.B. Mitchell, and T. Roos, **Voronoi diagrams of moving points in the plane** Proc. 17<sup>th</sup> International Workshop on Graphtheoretic Concepts in Computer Science. Fischbachau, Germany, LNCS 570, pp 113-125, 1991
- [10] L.J. Guibas and J. Stolfi, **Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams**, ACM Transactions on Graphics, Vol. 4, No. 2, pp 74-123, 1985
- [11] T. Imai and K. Sugihara, **A failure-free algorithm for constructing Voronoi diagrams of fine segments**, Transaction of the Information Processing Society of Japan, Vol. 35, No. 10, pp 1966-1977, 1994
- [12] D. Kirkpatrick, **Efficient computation of continuous skeletons**, Proc. 20<sup>th</sup> Ann. IEEE Symp. Foundations of Computer Science, pp 18-27, 1979
- [13] T. Roos, **Dynamic Voronoi diagrams**, PhD thesis, University of Würzburg, Germany, 1991
- [14] T. Roos, **Tighter bounds on Voronoi diagrams of moving points**, 5<sup>th</sup> Canadian Conference on Computational Geometry CCCG'93, 1993
- [15] K. Sugihara and M. Iri, **Construction of the Voronoi diagram for one million generators in single-precision arithmetic**, Proceedings of the IEEE, Vol. 80, No. 9, pp 1471-1484, 1992
- [16] C.K. Yap, **An  $O(n \log n)$  algorithm for the Voronoi diagram of a set of simple curve segments**. Discrete & Computational Geometry, Vol. 2, pp 365-393, 1987