The L_∞ Hausdorff Voronoi diagram revisited

Evanthia Papadopoulou* Jinhui Xu†

Abstract

We revisit the L_∞ Hausdorff Voronoi diagram of clusters of points, equivalently, the L_∞ Hausdorff Voronoi diagram of rectangles, and present a plane sweep algorithm for its construction, generalizing and improving upon previous results. We show that its structural complexity is $\Theta(n+m)$, where n is the number of given clusters and m is the number of *essential* pairs of *crossing* clusters. The algorithm runs in $O((n+M)\log n)$ time and $O(n+M)$ space, where M is the number of *potentially essential crossings*; m, M are $O(n^2)$, $m \leq M$, but $m = M$, in the worst case. In practice, $m, M << n^2$, as the total number of crossings in the motivating application is typically small, even compared to n. For non-crossing rectangles the algorithm runs in optimal $O(n \log n)$-time and $O(n)$-space.

1 Introduction

Given a set S of point clusters in the plane, the *Hausdorff Voronoi diagram* of S, denoted *HVD(S)*, is a subdivision of the plane into regions such that the *Hausdorff Voronoi region* of a cluster P, denoted *hreg(P)*, is the locus of points *closer* to P than to any other cluster in S, where distance between a point t and a cluster P is measured as the *farthest distance* between t and any point in P, $d_f(t,P) = \max\{d(t,p), p \in P\}$.

$$hreg(P) = \{x \mid d_f(x,P) < d_f(x,Q), \forall Q \in S\}$$

hreg(P) is subdivided into finer regions by the *farthest Voronoi diagram* of P, *FVD(P)*. The farthest distance $d_f(t,P)$ is equivalent to the *Hausdorff distance*[1] between t and P. In the L_∞ metric[2], $d_f(t,P)$ is equivalent to $d_f(t,P')$, where P' is the minimum enclosing axis-aligned rectangle of P. Thus, the L_∞ Hausdorff Voronoi diagram of S is equivalent to the L_∞ Hausdorff Voronoi diagram of the set S' of the minimum enclosing rectangles of all clusters in S. In this abstract the terms cluster and rectangle are used interchangeably. Once the cluster minimum enclosing rectangles are available, individual cluster points have no further influence.

The Hausdorff Voronoi diagram has appeared in the literature under different names having been motivated by different problems [6, 1, 10, 14, 11, 5, 16]. It first appeared in [6] as the *cluster Voronoi diagram*, where several combinatorial bounds were derived. A tight combinatorial bound on its structural complexity in the Euclidean plane was given in [11]. The L_∞ version of the problem was introduced in [10] as a solution to the VLSI *critical area extraction* problem for a defect mechanism on *contact layers*, called a *via-block*. A related, reverse type, of Voronoi diagram has been considered in [15, 2, 3].

In this paper we revisit the Hausdorff Voronoi diagram in the L_∞ metric. We provide a tight bound on its structural complexity and a plane sweep algorithm for its construction, generalizing and improving upon [10]. In particular, we show that the structural complexity of the L_∞ Hausdorff Voronoi diagram is $\Theta(n+m)$, where n is the number of input clusters and m is the number of *essential pairs* of *crossing* clusters (see Def. 1). The algorithm consists of an $O((n+M)\log n)$-time preprocessing step, based on *point dominance* in $\mathbb{R}^3$, followed by the main plain sweep algorithm that runs in $O((n+M)\log n)$-time and $O(n+M)$-space, where M reflects special crossings that are *potentially essential* (see Def. 2); m, M are $O(n^2)$, $m \leq M$, but $m = M$, in the worst case. In practice, typically, $m, M << n^2$, as the total number of possible crossings in our application is small, even compared to n. For non-crossing rectangles the algorithm simplifies to optimal $O(n \log n)$-time and $O(n)$-space; no previous algorithm achieves an optimal bound in case of non-crossing clusters. An output sensitive version of the plane sweep construction, at the expense, however, of advanced data structures that require increased space consumption, is given in [16].

The L_∞ Hausdorff Voronoi diagram finds applications in VLSI design for manufacturability as explained in [10, 12] and it has been integrated in [17]. Due to its simplicity and the absence, in L_∞, of numerical issues, the approach is very well suited for use in applications.

*Faculty of Informatics, Università della Svizzera italiana, evanthia.papadopoulou@usi.ch. Supported in part by SNSF project 200021_127137.

†Department of Computer Science and Engineering, State University of New York at Buffalo, {jinhui}@buffalo.edu

[1] The (directed) Hausdorff distance from a set A to a set B is $h(A,B) = \max_{a \in A} \min_{b \in B}\{d(a,b)\}$. The (undirected) Hausdorff distance between A and B is $d_h(A,B) = \max\{h(A,B), h(B,A)\}$.

[2] The L_∞ distance between two points p, q is $d(p,q) = \max\{|x_p - x_q|, |y_x - y_p|\}$.

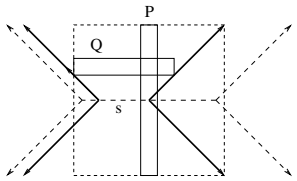

Figure 1: The L_∞ Hausdorff bisector of crossing rectangles.

2 Structural complexity and Definitions

Let S be a set of n rectangles, or a set of n point clusters in the plane, where each cluster has been substituted by its minimum enclosing rectangle. A pair of rectangles (P, Q) is called *crossing* if P and Q intersect in the shape of a cross. Given a crossing pair (P, Q), P is assumed to be at least as long as Q. For a rectangle P, let P^n, P^s, P^e, and P^w denote the north, south, east, and west edge of P respectively. P is called a horizontal (resp. vertical) if P^n is longer (resp. shorter) then P^e. The axis parallel line through edge P^i, $i = n, s, e, w$, is denoted as $l(P^i)$. The term P^i is also used to denote the main coordinate of edge P^i. The *core segment* of P is the locus of centers of all minimum enclosing squares of P, and it is given by the axis-parallel line segment of the L_∞ *farthest Voronoi diagram* of P. It can be viewed as an ordinary line segment s additively weighted with $w(s) = d_f(s, P)$. In Fig. 1, *FVD(P)* is illustrated in dashed lines and the core segment is indicated by s.

The Hausdorff bisector between two clusters P, Q is $b_h(P, Q) = \{y \mid d_f(y, P) = d_f(y, Q)\}$ and it is a subgraph of $FVD(P \cup Q)$ [14]. For a rectangle Q strictly enclosed in the interior of a minimum enclosing square of P, $b_h(P, Q)$ consists of either one (if P and Q are non-crossing) or two (if P and Q are crossing) chains, each one forming a V-shape out of the ± 1-slope rays of $FVD(P \cup Q)$; the apex of each chain is called a *V-vertex*. A V-vertex v is incident to the core segment of P and its $90°$ angle faces the portion of the plane closer to P. It is characterized as *up*, *down, right*, or *left*, depending on whether its $90°$ angle is facing north, south, east, or west respectively. In addition, it is characterized as *crossing*, if Q is crossing P, and *non-crossing*, otherwise. The minimum enclosing square of P centered at V-vertex v is also enclosing Q and it is denoted as $square(P, v)$. It is also denoted as $square(P, Q^i)$, where Q^i, is the non-crossing edge of Q that delimits one of its edges. Fig. 1 illustrates $b_h(P, Q)$ consisting of two crossing V-vertices, one right and one left; $square(P, Q^w)$ is illustrated dashed. $square(P, Q^i)$ is referred to as an *extremal minimum enclosing square* of P and Q. The V-vertices of *HVD(S)* are referred to as *Voronoi V-vertices*.

Definition 1 *A pair of crossing rectangles (P, Q) is called* essential *if there is an extremal minimum enclosing square of P and Q, $square(P, Q^i)$, that is empty of any other rectangle.*

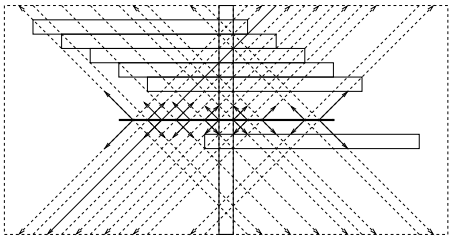

Figure 2: Collection of essential crossings.

Lemma 1 *A pair of crossing rectangles (P, Q) induces a Voronoi V-vertex in HVD(S) if and only if (P, Q) is an essential crossing.*

Combining Lemma 1 with the structural complexity results of [14] we derive the following bound.

Theorem 2 *The structural complexity of the L_∞ Hausdorff Voronoi diagram of a set S of n point clusters, equivalently n rectangles, is $\Theta(n + m)$, where m is the total number of essential crossings.*

Definition 2 *A collection of crossings for a vertical rectangle P, (P, Q_i), $i = 1, \ldots, k$, is called* a staircase, *if $Q_i^w < Q_{i+1}^w$ and $Q_i^e < Q_{i+1}^e$, $i = 1, \ldots, k$. If in addition, $square(P, Q_i^w)$ is empty of $Q_j \neq Q_i$, the staircase and its crossings are called* potentially essential. *The maximum size of a potentially essential staircase for P is the number of potentially essential crossings for P. Let M denote the total number of potentially essential crossings for all vertical rectangles in S, plus the number of essential crossings for all horizontal rectangles.*

Fig.2 shows a collection of potentially essential crossings that are essential i.e., they all induce Voronoi V-vertices.

3 A refined plane sweep construction

In this section we revisit the plane sweep construction of the L_∞ Hausdorff Voronoi diagram [10], generalize it to crossing rectangles, and improve its time complexity to optimal in the non-crossing case. It is based on the standard plane sweep paradigm for Voronoi diagrams [7, 4] and its adaptation for line segments in L_∞ [13]. Assume a vertical sweep-line l_t sweeping the entire plane from left to right. At any instant t of the sweeping process we compute $HVD(S_t \cup l_t)$, for $S_t = \{P \in S \mid l(P^e) < t\}$. The boundary of the Voronoi region of l_t is the *wavefront* at time t. Voronoi edges and core segments incident to the wavefront are called *spike bisectors* and *spike core segments* respectively. The combinatorial structure of the wavefront changes at specific *events*

organized in a priority queue. We have four types of *site events*: *start-vertical-rectangle*, *end-vertical-rectangle*, *V-vertex* events (for brevity *V-events*), and *horizontal-rectangle* events. Site events are partially similar to those for ordinary line segments [13] enhanced with additional functions to handle V-vertices and disconnected Voronoi regions. *Spike events* are caused by the intersection of incident spike bisectors, and they remain the same as in the ordinary plane sweep paradigm.

The *wave-curve* of a rectangle R is the bisector between R and the sweep line l_t, at time t, $b(R, l_t) = \{y \mid d_f(y, R) = d(y, l_t)\}$, where $d(y, l_t)$ is the ordinary distance between y and l_t. In L_∞, it consists of two or three *waves*: a ray of slope -1, corresponding to $b(R^s, l_t)$, a ray of slope $+1$, corresponding to $b(R^n, l_t)$, and possibly a vertical line segment corresponding to $b(R^w, l_t)$, if appropriate. The *wavefront*, at time t, is the lower envelope, with respect to the sweep line, of the *wave-curves* of all rectangles in S_t. In L_∞, the wavefront is monotone with respect to any line of slope ± 1.

The wavefront is typically maintained as a height balanced binary tree, $\mathcal{T}$, ordered from bottom to top. Leaf nodes correspond to waves, while internal nodes correspond to spike bisectors and spike core segments revealing *breakpoints* between incident waves. In order to answer queries regarding V-vertices efficiently, we augment $\mathcal{T}$ with additional information. Each node x is augmented with a *w-max* value representing the rightmost west edge of all rectangles contributing a wave to the portion of the wavefront rooted at x, and an *x-min* value representing the minimum x-coordinate of the portion of the wavefront rooted at x. In particular, for a leaf node representing a wave of rectangle R, the *w-max* value is R^w and the *x-min* value is $+\infty$. For an internal node x, *w-max* is the maximum between the *w-max* values of its children; the *x-min* value points to the breakpoint of minimum x-coordinate among its own breakpoint and the *x-min* values of its children. These values remain the same unless a combinatorial change in the wavefront (i.e., an event) takes place.

Any Voronoi point in *HVD(S)* enters the wavefront at the time of its *priority*. The *priority* of a point v is the rightmost x-coordinate of the smallest square centered at v that is entirely enclosing the rectangle P that induces v. The priority of a rectangle P is denoted as *priority(P)* and corresponds to the x-coordinate of P^e.

Let us now discuss the handling of various types of events of a rectangle P of core segment s. At time t, let $r_1(t)$ and $r_2(t)$ be the rays of slope $+1$ and -1 respectively, emanating from $l(P^n) \cap l_t$, and $l(P^s) \cap l_t$ respectively, extending towards the left of l_t. Let $a(t)$ and $b(t)$ be the intersection points of $r_1(t)$ and $r_2(t)$ with the wavefront, respectively, at time t. In case of a wave collinear with $r_1(t)$ or $r_2(t)$, the rightmost endpoint is assigned to $a(t), b(t)$.

Consider time $t = \mathit{priority}(P)$. If the wavefront has not reached s yet, then the handling of the corresponding event (a start-vertical-rectangle or a horizontal-rectangle event) is similar to processing an ordinary line segment event [13, 10]: The portion of the wavefront between $a(t)$ and $b(t)$ is finalized and gets substituted by the wave-curve of P. There is one new action to take: For any crossing V-vertex on the finalized portion of the wavefront, induced by a rectangle Q, generate a V-event for the right V-vertex of $b_h(P, Q)$. If the wavefront has already covered portion of s, and P is vertical (start-vertical rectangle event) a V-event is generated to predict the first right V-vertex along s (if any). For this purpose, perform a binary search in the augmented wavefront to determine the wave between $a(t)$ and $b(t)$ with the rightmost w-max value as induced by a rectangle Q, and generate a V-event for the right V-vertex of $b_h(P, Q)$.

A V-event v is processed similarly. If at time $t = \mathit{priority}(v)$ the event is *invalid* i.e., the wavefront has already covered v, generate a new V-event. End-rectangle events are similar to right-events of [10].

A horizontal-rectangle event is processed at time $t = \mathit{priority}(P)$. The problem is to identify the intersections (V-vertices) of the wavefront with the vertical core segment s. To identify V-vertices efficiently we use the x-min value of the augmentation. Let r be the breakpoint of minimum *x-min* value between $a(t)$ and $b(t)$. If r is to the right of s then s must be entirely covered by the wavefront and there can be no intersections; $reg(P) = \emptyset$. Otherwise, trace the wavefront sequentially, starting at r, until the first intersections above and below r are determined. The intersection above (resp. below) corresponds to a *down* (resp. *up*) V-vertex v (resp. u). Repeat the process for the portions of the wavefront above v and below u until all intersections are determined.

The time complexity of the plane sweep algorithm, as presented, is $O(n + m + E) \log n)$, where E is the number of invalid V-events. There are two reasons for invalid V-events: 1. Potentially essential staircases of vertical rectangles whose crossings are not all essential. 2. Sequences of *strongly dominating* vertical rectangles, even in the case of non-crossing rectangles. Given a pair of vertical rectangles (P, Q), P is said to *dominate* Q if $Q^w < P^w$ and $Q^n < P^n$, $Q^s > P^s$. If in addition, there is a minimum enclosing square of Q that is crossing P, P is said to *strongly dominate* Q.

To eliminate source-2 of invalid V-vertex events, a preprocessing step may be added to the algorithm, which computes for every vertical rectangle P its *dominating sequence* of rectangles: The *dominating-sequence* of P is a maximal collection of vertical rectangles $R_i, i = 1, \ldots, k$, such that every R_i is entirely enclosed in a minimum enclosing square of P, R_1 is

the rectangle with the rightmost west edge among all rectangles dominated by P; if R_{i-1}, $i > 1$, is crossing P, let R_i be the rectangle with the righmost west edge dominated by $square(P, R^e_{i-1})$.

Computing dominating sequences of vertical rectangles can be transformed into a *point dominance* problem in $\mathbb{R}^3$ and it can be solved by plane sweep in $O((n+M)\log n)$-time, as sketched in the following section. Given the dominating sequences of vertical rectangles, the number of invalid V-events is bounded by $O(n+M)$. Thus, we conclude.

Theorem 3 *HVD(S) can be computed by plane sweep in $O((n+M)\log n)$ time, $O(n+M)$-space. In the case of non-crossing rectangles this results in optimal $O(n\log n)$-time and $O(n)$-space.*

4 Preprocessing step – Point dominance

For any vertical rectangle P, map P^w into point $p = (P^n, -P^s, P^w)$ in $\mathbb{R}^3$. A point $p = (p_1, p_2, p_3)$ is said to *dominate* a point $q = (q_1, q_2, q_3)$ if $p_i \geq q_i$ for all $i, 1 \leq i \leq 3$. For any point p, determine the topmost point r dominated by p, which is R_1.

This can be done by sweeping points in decreasing z-coordinate, while maintaining their *minima*, $M(t)$, i.e., the set of points above the sweeping plane at time t that do not dominate anything so far, projected on the xy-plane. $M(t)$ can be organized in a *priority search tree* $T_{M(t)}$ [9]. Let $r = (r_1, r_2, r_3)$ be the point to be considered at time $t = r_3$ corresponding to a rectangle R. Perform a range query on $T_{M(t)}$ for the NE quadrant of (r_1, r_2), i.e., range $J(r) = [r_1, \infty] \times [r_2, \infty]$. For any point p within $J(r)$, report (p, r). Delete all points in $M(t) \cap J(r)$ from $T_{M(t)}$ and insert r. Point r can be deleted from $T_{M(t)}$ when the west edge of its leftmost minimum enclosing rectangle, referred to as its *expiration time*, is reached. Since priority search trees are fully dynamic this is an $O(n\log n)$-time $O(n)$-space algorithm.

For any point p within $J(r)$ corresponding to a crossing (P, R), consider $square(P, R^e)$ and its corresponding point p' which is assigned the expiration time of P, to be processed normaly, in order to determine R_i following R in the dominating sequence of P. This process is repeated by considering $square(P, R^e_i)$, until either a rectangle R_k non-crosssing with P is determined or the expiration time of P is reached. Since all crossings in a dominating sequence are potentially essential, the time complexity is $O((n+M)\log n)$.

References

[1] M. Abellanas, G. Hernandez, R. Klein, V. Neumann-Lara, and J. Urrutia. A combinatorial property of convex sets. *Discrete Computat. Geometry*, 17:307–318, 1997.

[2] M. Abellanas, F. Hurtado, C. Icking, R. Klein, E. Langetepe, L. Ma, B. Palop, and V. Sacristán. The farthest color Voronoi diagram and related problems. *Abstracts 17th Euro. Workshop Comput. Geom. 2001*, 113–116.

[3] O. Cheong, H. Everett, M Glisse, J Gudmundsson, S. Hornus, S. Lazard, M. Lee, and H.-S. Na. Farthest-Polygon Voronoi Diagrams. arXiv:1001.3593v1 [cs.CG], 2010.

[4] F. Dehne and R. Klein. The big sweep": On the power of the wavefront approach to Voronoi diagrams. *Algorithmica*, 17:19–32, 1997.

[5] F. Dehne, A. Maheshwari, and R. Taylor. A coarse grained parallel algorithm for Hausdorff Voronoi diagrams. In *Proc. 2006 Int. Conf. on Parallel Processing*, 497–504, 2006.

[6] H. Edelsbrunner, L. Guibas, and M. Sharir. The upper envelope of piecewise linear functions: Algorithms and applications. *Discrete Computat. Geometry*, 4:311–336, 1989.

[7] S. J. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

[8] R. Klein, K. Mehlhorn, S. Meiser, "Randomized incremental construction of abstract Voronoi diagrams", *Computational Geometry: Theory and Applications*, 3, 1993, 157-184

[9] E. M. McCreight. Priority Search Trees. *SIAM J. Comput.*, 14:257–276, 1985.

[10] E. Papadopoulou. Critical area computation for missing material defects in VLSI circuits. *IEEE Trans. Comp.-Aided Design*, 20(5):583–597, 2001.

[11] E. Papadopoulou. The Hausdorff Voronoi diagram of point clusters in the plane. *Algorithmica*, 40:63–82, 2004.

[12] E. Papadopoulou. Net-aware critical area extraction for opens in VLSI circuits via higher-order Voronoi diagrams. *IEEE Trans. Comp.-Aided Design*, 30(5), 2011. Prel. version *LNCS* vol 4835, 716–727, 2007.

[13] E. Papadopoulou and D. T. Lee. The L_∞ Voronoi diagram of segments and VLSI applications. *Int. J. Computat. Geometry and Applications*, 11(5):503–528, 2001.

[14] E. Papadopoulou and D. T. Lee. The Hausdorff Voronoi diagram of polygonal objects: A divide and conquer approach. *Int. J. Comput. Geometry and Applications*, 14(6):421–452, 2004.

[15] D. P.Huttenlocher, K. Kedem, and M. Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete Computat. Geometry*, 267–291, 1993.

[16] J. Xu, L. Xu, and E. Papadopoulou. Computing the map of geometric minimal cuts. In *Proc. 20th Int. Symp. Algorithms and Computation*, *LNCS* vol. 5878, 244–254, 2009.

[17] "Voronoi CAA: Voronoi Critical Area Analysis", IBM CAD Tool, Dept. Electronic Design Automation, IBM Microelectronics, Burlington, VT. Initial patents: US6178539, US6317859. Distributed by Cadence.