Multiresolution Approximations of Generalized Voronoi Diagrams

I. Boada, N. Coll, and J.A. Sellarès

Institut Informàtica i Aplicacions, Universitat de Girona, Spain
{imma, coll, sellares}@ima.udg.es

Abstract. A framework to support multiresolution approximations of planar generalized Voronoi diagrams is presented. Our proposal is: (1) A multiresolution model based on a quadtree data structure which encodes approximations of a generalized Voronoi diagram at different levels of detail. (2) A user driven refinement strategy which generates from the quadtree a continuous polygonal approximation of the Voronoi diagram.

1 Introduction

The generalized Voronoi diagram of a set of sites partitions the plane into regions, one per site, such that all points in a region have the same closest site according to some given distance function [3,4,12]. Voronoi diagrams are widely used in many scientific fields and application areas, such as computer graphics, geometric modeling, solid modeling, geographic information systems, ... (see [17]). Although there are different algorithms to compute exact generalized planar Voronoi diagrams, they usually suffer from numerical robustness problems and are time-consuming. To avoid these problems researchers have proposed techniques to compute approximated Voronoi diagram within a predetermined precision. Among the existing techniques, we focus our interest on adaptive Voronoi diagram approximation methods based on hierarchical structures [11,16,15, 14,7,8,9].

One of the main advantages of hierarchical methods relies on their ability to support multiresolution. Multiresolution approaches can effectively control the tradeoff between quality and speed extracting models in which the resolution varies over the domain of the dataset. Multiresolution approximations of Voronoi diagrams are interesting in their own right. They are useful tools to solve problems in robot path planning [10,5], curve and surface reconstruction [2], and region approximation in GIS [1]. Unfortunately, algorithm for obtaining multiresolution approximations of Voronoi diagrams are still scarse. In this paper we propose a method to obtain them. The proposed approach is an extension of the algorithm we presented for the adaptive approximation of generalized planar Voronoi diagrams [7]. In this algorithm we built a quadtree, called the Voronoi Quadtree (VQ), to encode the information of the boundaries of the Voronoi regions in its leaf nodes. Then we group leaf nodes into five different patterns that determine how to generate the polygonal approximation of the part of the

Voronoi diagram contained in the leaf. This approximation is represented by a DCEL structure.

Taking into account the capabilities of quadtrees to support multiresolution, we propose to use the VQ as a multiresolution model, i.e. the model that maintains the diagram approximations at different levels of detail. Our multiresolution approach starts with the construction of a VQ. Once all the information of the diagram is encoded in the VQ leaf nodes we apply a refinement process. The user defines a region of interest (ROI) and a degree of accuracy. Then, the process selects or generates, when it is required, the nodes from which the polygonal approximation of the Voronoi diagram satisfiying user requirements has to be obtained. The method guarantees that the approximation is continuous, the ROI is represented with the user desired degree of accuracy and out of the ROI the representation is as simple as possible.

2 Definitions and Notation

In this section we present the definitions and notation used int he paper. Let $\mathcal{S} = \{s_1, \cdots, s_n\}$ be the set of input sites. Each site s is represented by $s =< G_s, D_s, P_s >$, where G_s defines the geometry of the site s, D_s is the function that gives the distance from any point p to s and P_s (the base point of s) is a point such that $D_s(P_s) = 0$ and $P_s \in K$, where K is a rectangle. Each site $s_i \in \mathcal{S}$ has associated a Voronoi region $VR(s_i) = \{p \mid D_{s_i}(p) \leq D_{s_j}(p) \,\text{for all}\; j \neq i\}$. The generalized Voronoi diagram of $\mathcal{S}$, denoted $VD(\mathcal{S})$, is defined as the partition of the plane induced by the Voronoi regions. Our goal is to obtain a multiresolution approximation of the part of $VD(\mathcal{S})$ included in K.

3 A Multiresolution Framework

Multiresolution representations permit extracting models in which the resolution varies over the domain of the dataset. In this way the user may choose to approximate with highest detail only some parts of the dataset, for example the ones considered of maximal interest. To define a framework able to support multiresolution Voronoi diagram approximations two issues must be resolved. First of all, it has to be defined a model capable of maintaining approximations of the Voronoi diagram at different levels of detail, taking into account that the level of detail may be different in distinct areas of the diagram. Secondly, it has to be defined a strategy able to generate from the information encoded in the multiresolution model the polygonal approximation of the Voronoi diagram that satisfies user requirements. The strategy has also to detect and solve the cracks (i.e discontinuities typical of domain decompositions that are fine in certain regions and coarse in the others).

4 A Quadtree-Based Multiresolution Model

Our multiresolution model must allow us to extract approximations of the Voronoi diagram suitable for diverse circumstances and it must also allow us to change the level of detail without excessive overhead. To satisfy all these requirements we propose to exploit the capabilities of the Voronoi quadtree data structure presented in [7]. Thus, the first phase of our approach consist on the creation of a Voronoi quadtree (VQ).

4.1 The Voronoi Quadtree

The VQ make use of the divide-and-conquer power of binary subdivision of quadtrees to encode the information required to obtain a polygonal approximation of a Voronoi diagram. Differently of other adaptive methods, that always consider all the set of sites, in the VQ approach at each step of the process we only take into account the sites related to each node, reducing in this manner the complexity of the diagram approximation with respect to other adaptative related methods.

To construct the VQ a set of basic definitions is introduced. Let N be a node and s a site. We say that: (i) s is a *I-site* with respect to N when $P_s \in N \cap VR(s)$; (ii) s is a *V-site* with respect to N when some vertex v of N verifies $v \in VR(s)$ and (iii) s is a *E-site* with respect to N when it is not a V-site and there exist some edge e of N that verifies $e \cap VR(s) \neq \emptyset$.

A node of the VQ is a leaf node when its level is L_M (the maximal subdivision level) or it is completely contained in a Voronoi Region, i.e. the total number of V-sites, I-sites and E-sites contained in the node is one. The VQ construction process is based on a breadth first traversal of the quadtree which uses a priority queue Q. The process starts with the creation of the root node assigning to it the rectangular region K and all the sites of S as I-sites. Then the V-sites of the root are computed and the root is sent to Q. In a loop over the Q nodes, for every node N we actualize its V-sites with the nearest of its sites. To maintain the coherence of the quadtree we evaluate adjacent nodes of N, modifying the information of its sites when it is required, and sending the nodes to Q if it is convenient. The construction process ends when Q is empty [7].

As the VQ encodes the information of the boundaries of the Voronoi regions in the leaf nodes we consider this phase as the initialization of the multiresolution model.

5 DCEL Based Multiresolution Polygonal Approximations of the Voronoi Diagram

To generate the polygonal approximation of the Voronoi diagram we will use the DCEL data structure [6]. This data structure uses three types of records, vertices, halfedges and faces, to maintain the adjacency between vertices, edges and faces of a planar subdivision. In [7] we describe how to obtain a DCEL based polygonal approximations of the Voronoi Diagram from the information encoded in the VQ leaf nodes. We assign a pattern to each VQ leaf node according to the distribution of its V-sites. This pattern determines the position of the DCEL-vertices and how they have to be connected (see Fig. 1). The accuracy of the diagram approximation obtained from leaf nodes of level L_M is $\frac{\sqrt{a^2+b^2}}{2^{L_M}}$, where a and b are edge lengths of K.

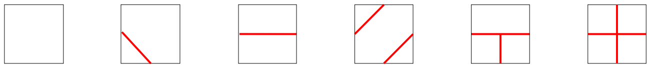
Fig. 1. Leaf node patterns of a Voronoi-Quadtree

To obtain a multiresolution approximation of the Voronoi diagram our idea is to extend the DCEL generation strategy proposed in [7]. The user defines the ROI, by a simple subregion of the dataset domain, and introduces the desired degree of accuracy ϵ. Then, since the VQ obtained in the previous phase has all leaf nodes at level L_M, we apply a refinement process that determines the set of nodes of the VQ from which the polygonal approximation of the Voronoi diagram that satisfies the user requirements has to be obtained.

The refinement process classifies VQ leaf nodes as outer ROI nodes if their four V-sites are out of the ROI and as inner ROI nodes on the contrary. The ϵ parameter determines the level $L_\epsilon = \left\lceil \log_2 \left(\frac{\sqrt{a^2+b^2}}{\epsilon} \right) \right\rceil$ of the quadtree at which nodes of the ROI have to be represented. Obtain the inner ROI representation is straightforward, it is only required the same subdivision process applied for the VQ construction described in the previous section.

The main difficulty of the refinement process is on the representation of outer ROI nodes, its representation has to be simplified as much as possible while preserving the continuity. Hence, the critical point is how to guarantee the continuity of the diagram approximation. To solve this problem we propose the crack solving strategy presented in the next section.

5.1 Crack Solving

To generate the DCEL approximation we apply the policy based on a set of predefined patterns (see Fig. 1) and the location of DCEL-vertices always on the midpoint of the intersected edges of the node or on the node's center. When an intersected edge is common to nodes of different level a crack situation arises. To define the crack solving strategy we have analyzed the possible crack situations. We detect three different cases, each one characterized by the distribution of the V-sites onto the intersected edge that contains the crack.

Case 1. The first case is given when the V-sites of the edge from one side are the same that the V-sites of the other one, the edge has a unique DCEL-vertex and the nodes that contain this DCEL-vertex have different levels. This case has been illustrated in Fig. 2(a.1). In this case to solve the crack we force the coarse

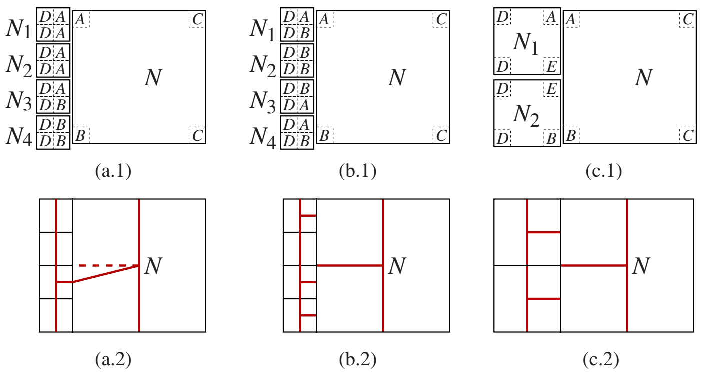

Fig. 2. (a.1) If the DCEL vertex is computed with respect to N has a different position that if it is computed with respect to N_3. (a.2) To solve the crack we always force the coarse leaf node representation to meet with the finer one.(b.1) Observe that all the V-sites located on the intersected edge are A or B. (b.2) Onto the common edge node there are different DCEL-vertices, one if we consider the edge from the N side, another another for the N_1 side, another for N_3 and another for N_4. To solve this crack node N has to be subdivided. (c.1) Node N has V-sites A and B while nodes N_1 and N_2 have V-sites A, E and B. (c.2) Onto the common edge there are three DCEL-vertices. To solve the crack the V-site E has to be introduced as an E-site of N, and N has to be subdivided

leaf node representation to meet with the finer one (see Fig. 2(a.2)).
Case 2. The second situation is given when, although the set of V-sites located on a common edge are the same, we identify more than one intersection point. This case has been illustrated in Figs. 2(b.1), 2(b.2). The correct approximation of the DCEL only can be obtained if the node of upper level is subdivided.
Case 3. The last case is given when the set of V-sites located on the common edge are different from one side to the other. This case has been illustrated in Fig. 2(c.1). To solve the discontinuity generated by this configuration (see Fig. 2(c.2)) the V-site not common to the two nodes has to be introduced as an E-site and the node has to be subdivided.

5.2 VQ Refinement Process

Once the situations of crack have been detected and we know how to solve them we define the refinement strategy. This refinement strategy uses a new leaf node criterion which varies according the position of the node with respect to the ROI. If the node is inside the ROI the criterion is the same used in the VQ construction phase, i.e. the node is a leaf if its number of sites is one. A node outside the ROI is a leaf if it has no E-sites.

Driven by this new leaf node criterion the refinement process starts with an initialization phase that detects all the VQ leaf nodes contained in the ROI and sends them to the queue Q. In a loop over Q nodes, for every node N its V-sites are actualized with the nearest of its sites. Then: (i) if N is a leaf node we apply to it the propagation procedure. This procedure sends to Q the adjacent nodes of each vertex v of N that have to be reprocessed. Let N' be one of these nodes. N' has to be reprocessed if one of three situations represented in Fig. 3 is given. At the end of the propagation procedure if N and its brothers contain only one site, they are pruned. (ii)If N is not a leaf its four son nodes are created, the I-sites and E-sites of N are properly distributed to them and the V-sites of the sons are computed considering the sites of N. The propagation procedure is applied to each one of the descendant nodes and if a son is not a leaf either it is send to Q. The process ends when Q is empty.

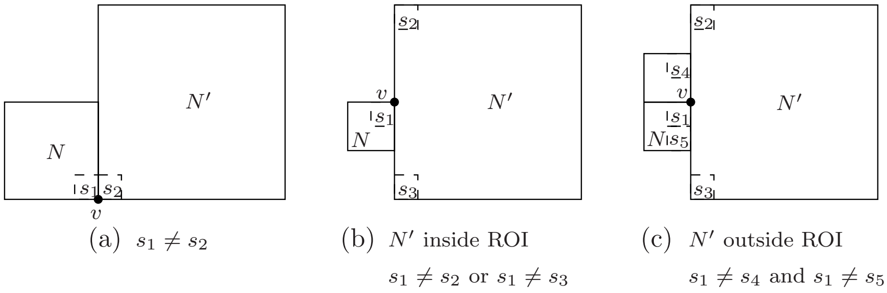

Fig. 3. N' has to be reprocessed if: (a) N' has a different V-site in v; (b) v lies on an edge of N', N' is inside the ROI, and the N V-site of v is different to one of the N' V-sites on the edge; (c) v lies on an edge of N', N' is outside the ROI, and the N V-site of v is different to each one of the nearest V-sites to v on the edge

At the end of the refinement process, leaf nodes of the ROI are all at level L_ϵ while nodes outside the ROI may be distributed at different levels of the quadtree. The different distribution of leaf nodes affects the DCEL obtainment. Now when a DCEL-vertex is common to leaf nodes of different level nodes we always force the coarse leaf node representation to meet with the finer one.

6 Computational Cost

In this section the computational costs of the processes related with the proposed method are summarized.

Let n be the number of sites, n_{ROI} be the number of sites whose Voronoi region intersects the ROI, $\overline{\partial VD(\mathcal{S})_K}$ be the length of the part boundary of $VD(\mathcal{S})$ included in K, and $\overline{\partial VD(\mathcal{S})_{ROI}}$ be the length of the part boundary of $VD(\mathcal{S})$ included in the ROI. There are some important considerations to be observed in order to obtain the computational cost of the VQ construction algorithm and the refinement process: (i) The algorithm applies the subdivision process to nodes that contain a piece of $VD(\mathcal{S})$. (ii)A curve of length $\overline{C}$ generates $O(\overline{C}\,2^l)$ nodes of level l in a quadtree, and $O(\overline{C}\,2^{L_M+1})$ nodes in a quadtree of maximum level L_M [13]. (iii) For each level we distribute the n sites to some nodes as I-sites. (iv) For each node we need to locate its neighbor nodes. This can be done in L_M worst time, but the expecting time for locating neighbors is approximately $O(4)$ [13].

According to the previous considerations we have the next results. The number of nodes generated by the VQ construction algorithm is: $O(\overline{\partial VD(\mathcal{S})}_K 2^{L_M+1})$. The running time of the VQ construction algorithm is: $o(nL_M + \overline{\partial VD(\mathcal{S})}_K 2^{L_M+1})$. The number of nodes generated by the refinement process in a the ROI with accuracy ϵ is: $O(\overline{\partial VD(\mathcal{S})}_{ROI} 2^{L_\epsilon - L_M})$. The running time of the refinement process in the ROI with accuracy ϵ is: $o(n_{ROI}(L_\epsilon - L_M) + \overline{\partial VD(\mathcal{S})}_{ROI} 2^{L_\epsilon - L_M})$.

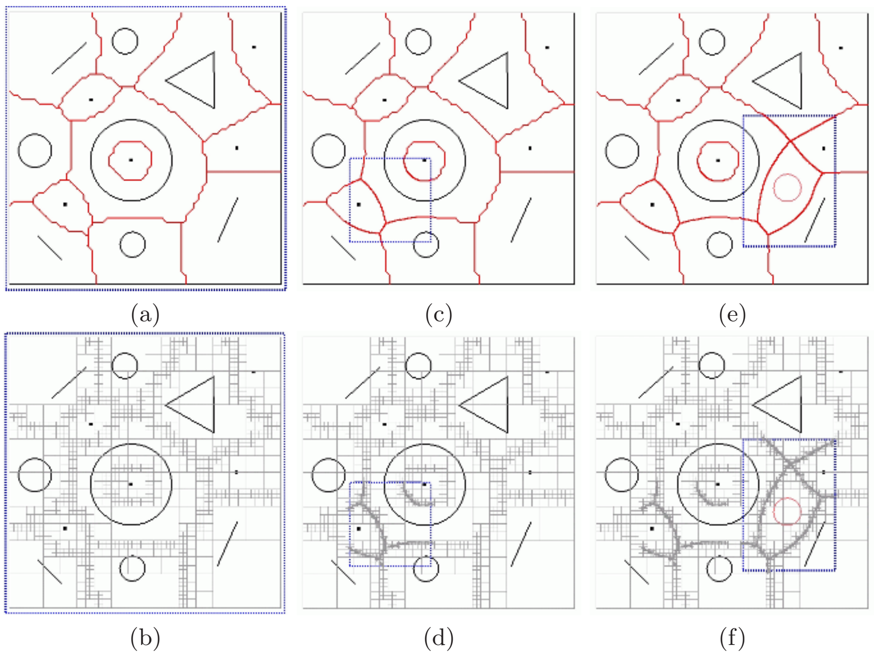

Fig. 4. Voronoi diagrams approximations obtained with the proposed approach are represented in the first row. The last row represents the distribution of nodes used to generate these polygonal approximation

7 Results

In this section we present the results obtained with our proposed approach when experimenting with a set of 13 sites. All these results have been represented in Fig. 4. The first image (see Figs. 4(a)) corresponds to the polygonal approximation of the Voronoi diagram obtained at the end of the VQ initialization phase.

In Fig. 4(b) we have also illustrated the distribution of leaf nodes. Next figures (see Figs. 4(c)(d)) correspond to the approximations of the diagram once a ROI has been defined. Observe that the refinement process only affects nodes of the ROI and some on the boundary of the ROI. It can be seen that there are no cracks on the polygonal approximation. We want to remark that our multiresolution approach supports dynamic maintenance, under insertion and deletion of sites, by using the strategy we describe in [9]. To illustrate this property we have introduced a new site to the diagram and once the method has properly updated the Voronoi diagram approximation we have selected a ROI around the new site. The obtained diagrams are represented in Figs. 4(e)(f)).

Acknowledgments. This work has been funded in part with grant numbers TIC2000-1009, TIC2001-2226-C02-02, TIC2001-2392-C03-01 and 2001-SGR-00296.

References

1. Alani, H., Jones, C.B., Tudhope, D.: Voronoi-based region approximation for geographical information retrieval with gazetteers. Int. J. Geographical Information Science, 15(4). (2001) 287-306
2. Amenta, N., Bern, M. Kamvysselis, M.: A new Voronoi-based surface reconstruction algorithm. Proceedings of Siggraph '98. ACM (1998) 415-421
3. Aurenhammer, F.: Voronoi diagrams: A survey of a fundamental geometric data structure. ACM Computer Surveys, 23(3). (1991) 686-695
4. Aurenhammer, F. Klein, R.: Voronoi diagrams. In: Sack, J.R., Urrutia, J. (eds.): Handbook of Computational Geometry. Elsevier (2000) 201-290
5. Behnke, S.: Local Multiresolution Path Planning. Proceedings of RoboCup 2003 International Symposium. (2003)
6. de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry; Algorithms and applications. 2nd edn. Springer-Verlag, Berlin Germany (2000)
7. Boada, I., Coll, N., Sellarès, J.A.: The Voronoi-Quadtree: construction and visualization. Eurographics 2002 Short Presentations. (2002) 349-355
8. Boada, I., Coll, N., Sellarès, J.A.: Hierarchical Planar Voronoi Diagram Approximations. Proceedings of 14th Canadian Conference on Computational Geometry. (2002) 40-45
9. Boada, I., Coll, N., Sellarès, J.A.: Dynamically maintaining a hierarchical planar Voronoi diagram approximation. In: Kumar, V. et al. (eds.): ICCSA 2003, Lecture Notes in Computer Science, 2669. Springer-Verlag (2003) 836-846
10. Kambhampati, S., Davis, L.S.: Multiresolution Path Planning for Mobile Robot's, IEEE Journal of Robotics Automation, RA-2(3). (1986) 135-145
11. Lavender, D., Bowyer, A., Davenport, J., Wallis, A., Woodwark, J.: Voronoi diagrams of set-theoretic solid models. IEEE Computer Graphics and Applications, 12(5). (1992) 69-77
12. Okabe, A., Boots, B., Sugihara, K., Chiu, S.N.: Spatial Tessellations: Concepts and Application of Voronoi Diagrams. John Wiley (2000)
13. Samet, H.: Applications of Spatial Data Structures: computer graphics, image processing, and GIS. Addison-Wesley (1993)

14. Teichmann, T., Teller, S.: Polygonal approximation of Voronoi diagrams of a set of triangles in three dimensions. Technical Report 766. Laboratory of Computer science, MIT (1997).
15. Telea, A.C., van Wijk, J.J.: Visualization of Generalized Voronoi Diagrams. Proceedings of IEEE VisSym '01. Springer (2001) 165-174
16. Vleugels, J., Overmars, M.: Approximating Generalized Voronoi Diagrams in Any Dimension. Int. J. on Computational Geometry and Applications, 8. (1998) 201-221
17. Gold, C.: Voronoi Diagrams page on the Web: Applications. http://www.voronoi.com/section_1.htm