Computational Geometry 48 (2015) 507–519

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Triangulations from topologically correct digital Voronoi diagrams

Thanh-Tung Cao [a,*], Herbert Edelsbrunner [b], Tiow-Seng Tan [a]

[a] School of Computing, National University of Singapore, Singapore
[b] Institute of Science and Technology, Klosterneuburg, Austria

ARTICLE INFO

Article history:
Received 17 April 2013
Accepted 27 March 2015
Available online 2 April 2015

Keywords:
GPU
GPGPU
Digital geometry
Delaunay triangulation

ABSTRACT

We prove that the dual of the digital Voronoi diagram constructed by flooding the plane from the data points gives a geometrically and topologically correct dual triangulation. This provides the proof of correctness for recently developed GPU algorithms that outperform traditional CPU algorithms for constructing two-dimensional Delaunay triangulations.

1. Introduction

In recent years, the computational power of graphics processing units (GPUs) has surpassed that of central processing units (CPUs). Continuing the trend, the gap between the two is expected to widen in the foreseeable future. With the introduction of programming models, such as CUDA [8] and OpenCL [7], there are now more application areas that benefit from the computational power of GPUs. These areas include scientific computing, games, data mining, and computational finance [9]. In computational geometry, GPUs have been used to solve problems in discrete as well as in continuous space. An example is the digital Voronoi diagram approximating the corresponding Euclidean structure, which has a wide range of applications in image processing, computer vision, and graphics [2]. Another is the Delaunay triangulation, which has applications in mesh generation and scientific computing [5].

In Euclidean space, the Voronoi diagram and the Delaunay triangulation are but different geometric expressions of the same combinatorial structure. In digital geometry, the translation from one to the other is made difficult by the need to approximate. Indeed, it is easy to construct a digital Voronoi diagram, just by coloring the pixels or their higher-dimensional analogues. Early work in this direction uses graphics hardware [6] and the texture unit of the GPU [15]. More recent work takes the vector propagation approach [3], which leads to algorithms whose running time depends solely on the image resolution and not on the number of data points [1,11,13]. In contrast, computing the Delaunay triangulation with GPUs has been more challenging. While Hoff et al. [6] mention the possibility to dualize the digital Voronoi diagram, it was not until recently that a complete GPU algorithm for the Delaunay triangulation has been described [12]. With the tremendous power of GPUs, this algorithm outperforms all traditional CPU algorithms, including the optimized Triangle software of Shewchuk [14]. This work has also been improved and further extended to handle constraints [10]. The reason for the difficulty is the approximate character of digital Voronoi diagrams, which may lead to unwanted artifacts when dualized, such as crossing

* Corresponding author at: School of Computing, National University of Singapore, 13 Computing Drive, Singapore 117417.
E-mail addresses: todd.t.cao@gmail.com (T.-T. Cao), edels@ist.ac.at (H. Edelsbrunner), tants@comp.nus.edu.sg (T.-S. Tan).

http://dx.doi.org/10.1016/j.comgeo.2015.04.001
0925-7721/

edges and missing triangles. However, there is experimental evidence suggesting that a careful implementation can avoid such artifacts.

In this paper, we present a detailed proof that dualizing the digital Voronoi diagram gives a topologically and geometrically correct triangulation. Leaving the correction of non-locally Delaunay diagonals to a postprocessing step, we call the result of the dualization the *digital Delaunay triangulation*. We base our proof on the recent improvement of the GPU Delaunay triangulation algorithm described in [10]. The proof has been used in the latest implementation of that algorithm.[1] After presenting a conceptual version of this algorithm in Section 2, we prove its correctness in Section 3. Specifically, we show that the digital Voronoi diagram obtained by flooding the pixel array can be dualized to give a topologically as well as geometrically valid triangulation in the plane. Our proof takes three steps to establish the validity of the digital Delaunay triangulation. The first step rationalizes the flooding algorithm by proving that the pixels are colored in the order of distance from the data points. The second step exploits this ordering to prove a technical topological result about loops. The third step uses this result to establish the desired properties of the digital Delaunay triangulation. We believe that our approach to proving the correctness of digital geometry algorithms by progressive abstraction is of independent interest.

2. The algorithm

In this section, we formally state the problem and give a conceptual but precise description of the algorithmic solution first presented in [10].

Problem specification The setting is a rectangular array of pixels. To talk about it, we define a *pixel* as the closed unit square centered at an integer point in the plane: $A + [-\frac{1}{2}, \frac{1}{2}]^2$, with $A \in \mathbb{Z}^2$. It has four *sides*: east, north, west, south, and four *corners*: north-east, north-west, south-east, south-west. We say two pixels are *neighbors* if they share a side or a corner. The decomposition of the plane into pixels is denoted by $\mathbb{Z}^2 + [-\frac{1}{2}, \frac{1}{2}]^2$. Since computers are finite, we consider only a finite rectangular piece of the thus decomposed plane and call this piece the *texture* within which all computations are performed. Now suppose we are given a subset of the pixels in the texture. We call the center of each such pixel a *seed point* and write $S = \{x_1, x_2, \ldots, x_n\}$ for the set of seed points. We assume that the points do not all lie on a single straight line. Equivalently, at least three of the seed points span a proper triangle. The goal is to connect the seed points in S with edges and triangles to form a simplicial complex. It will be convenient to add a *dummy* seed point, x_0, which we imagine at infinity and use as an additional vertex when we form the simplicial complex. With this modification, we consider a simplicial complex a *valid solution* to our problem if it satisfies the following three conditions:

I. The set of vertices is $S \cup \{x_0\}$.
II. The simplicial complex triangulates the sphere.
III. Removing x_0 gives a geometric realization in the plane.

Condition I prescribes the relationship between input and output. Condition II summarizes the required topological properties. It includes the local requirements of a 2-manifold, that every edge belongs to two triangles and every vertex belongs to a ring of triangles. It also prescribes the global topology of the simplicial complex to that of the 2-dimensional sphere. Condition III summarizes the required geometric properties, namely that the edges do not intersect other than at shared vertices, and the triangles do not intersect other than along shared edges. Note that Condition III applies only to the finite portion of the simplicial complex, obtained by removing the star of x_0, that is, x_0 together with all edges and triangles that connect to x_0. Since removing a single vertex star from a triangulated sphere keeps the rest connected, the finite portion of the simplicial complex is thus required to be connected.

Digital Voronoi diagrams Our approach to solving the problem mimics the computation of the Euclidean case. Recall that the (*Euclidean*) *Voronoi region* of a point x_i is the set of points for which x_i is the closest seed points, that is,

$$V_i = \{x \in \mathbb{R}^2 \mid \|x - x_i\| \leq \|x - x_j\|, \forall j\}.$$

It is easy to see that V_i is convex. The collection of V_i is the (*Euclidean*) *Voronoi diagram* of S. Finally, the (*Euclidean*) *Delaunay triangulation* is the dual of this diagram. Working with integer instead of real coordinates, we can only approximate this Euclidean construction. We do this with two types of *digital Voronoi diagrams*. To construct the first, we color each pixel with the index of the closest seed point:

$$E_i = \{A \in \mathbb{Z}^2 \mid \|A - x_i\| \leq \|A - x_j\|, \forall j\}.$$

We assume a fixed tie-breaking rule so that each pixel receives only one color. The *bulk* of E_i is the component B_i that contains the seed point, x_i. All the other pixels of E_i are *debris*, which exist only inside a sharp corner of the Euclidean Voronoi region; see Fig. 1. The debris is a serious drawback as it makes it difficult to turn the decomposition into a valid

[1] URL: http://www.comp.nus.edu.sg/~tants/delaunay2DDownload.html.

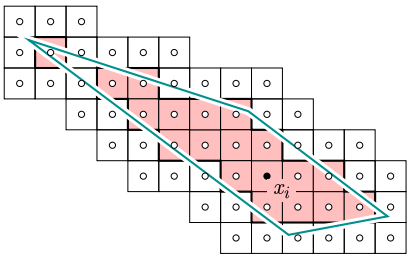

Fig. 1. A region with a sharp corner. Its corresponding digital region consists of the bulk and one debris pixel.

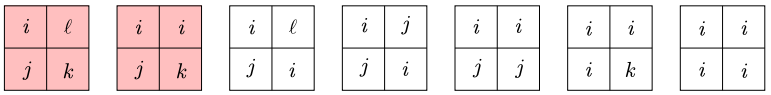

Fig. 2. The seven types of colorings of a 2-by-2 array of pixels. The first two are digital Voronoi vertices.

solution to our triangulation problem.

This difficulty is alleviated by introducing a second kind of digital Voronoi diagram. It is obtained by growing the regions simultaneously until they run into each other. In other words, we run n versions of breadth-first search in parallel, making sure they do not invade each other's territory. To make this process precise, we say a pixel A is *eligible* to be colored i if $A = x_i$ or A has a neighboring pixel of color i. Initially, after $s = 0$ steps, all pixels are uncolored. Write $F_{i,s}$ for the set of pixels colored i after s steps, and Q_s for the set of eligible pixel-color pairs (A, i). Writing $N(A)$ for the set of pixels neighboring A. We implement Q_s as a priority queue and allow it to contain additional pairs whose pixels are already colored. These pairs do not interfere with the proper execution of the algorithm. Let $\min(Q_s)$ be the operation that removes and returns the pair with minimum distance between the pixel center and the seed point of the color, with respect to a tie-breaking mechanism. We now state the algorithm more formally, suppressing the counter for the number of steps, which is implicit. We initialize the regions to $F_i = \emptyset$ and the queue to the set of pairs (x_i, i), for all i.

repeat
  $(A, i) = \min(Q)$;
  if A is not colored **then**
    $F_i = F_i \cup \{A\}$;
    $Q = Q \cup \{(B, i) \mid B \in N(A)\}$;
  end if
until $Q = \emptyset$.

It is easy to see that this algorithm succeeds in coloring all pixels. Writing F_i for the set of pixels colored i after the last step of the algorithm, this is equivalent to saying that the union of the F_i covers the entire texture. Indeed, in any other case there would be an uncolored pixel neighboring a colored pixel and the algorithm could continue coloring.

An important aspect of the algorithm is its tie-breaking mechanism. Any total order of the pixel pairs consistent with the Euclidean distance between pixel centers will do. For example, we may exploit the total order of the integers as follows. A pair of pixels is specified by four coordinates, which we sort into a vector of length four and write $(A, B) <_{\text{lex}} (C, D)$ if the vector obtained from $A, B \in \mathbb{Z}^2$ is lexicographically smaller than the vector obtained from $C, D \in \mathbb{Z}^2$. We say (A, B) has *higher priority* than (C, D), denoted as $\|A - B\| \prec \|C - D\|$, if $\|A - B\| < \|C - D\|$ or $\|A - B\| = \|C - D\|$ and $(A, B) <_{\text{lex}} (C, D)$. Note that $\|A - B\| \prec \|C - D\|$ implies $\|A - B\| \leq \|C - D\|$ but not always $\|A - B\| < \|C - D\|$. This will be important in our analysis of the algorithm.

Digital Delaunay triangulation Similar to the Euclidean case, we dualize the digital Voronoi diagram, and in particular the regions F_i. The key concept is that of a *digital Voronoi vertex*. This is a corner shared by four pixels that have either four different colors or three different colors in which the two pixels sharing the color also share a side. Equivalently, a digital Voronoi vertex is a 2-by-2 array of the type which is shaded in Fig. 2. We note that the third type of array, which is not a digital Voronoi vertex, is called a neck of the region whose color is repeated. We will see in Section 3 that the fourth type, with two crossing necks, does not arise. This is important when we dualize the colored regions as follows:

- For each digital Voronoi vertex with three different colors, i, j, k in counterclockwise order, we add the edges x_ix_j, x_jx_k, x_kx_i to $\mathcal{E}$ and the triangle $x_ix_jx_k$ to $\mathcal{T}$.
- For each digital Voronoi vertex with four different colors, i, j, k, ℓ in counterclockwise order, we add the edges x_ix_j, x_jx_k, x_kx_i and x_ix_k, x_kx_ℓ, $x_\ell x_i$ to $\mathcal{E}$ and the triangles $x_ix_jx_k$, $x_ix_kx_\ell$ to $\mathcal{T}$.

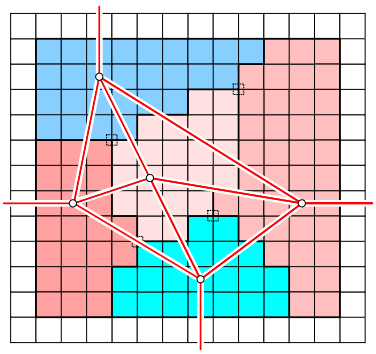

Fig. 3. The digital Delaunay triangulation superimposed on the digital Voronoi diagram obtained by flooding.

Note that in the second case, we have a choice between the two diagonals, which we make arbitrarily. Here, $\mathcal{E}$ and $\mathcal{T}$ are multisets. We will prove that $\mathcal{T}$ is in fact a set and $\mathcal{E}$ contains each edge exactly twice. Identifying the edges in pairs thus gives a simplicial complex; see Fig. 3 for an example. We call this the *digital Delaunay triangulation* of S. Its finite portion is obtained by removing x_0 together with all edges and triangles that share x_0.

3. The proof

In this section, we present the proof that the digital Delaunay triangulation is a valid solution to our triangulation problem. It consists of three consecutive steps, explained in the following three subsections.

3.1. Flooding sorts

In this subsection, we rationalize flooding by proving some basic properties. Recall our tie-breaking mechanism and that $\|A-B\| \prec \|C-D\|$ implies $\|A-B\| \leq \|C-D\|$.

Order by distance The main technical result in this section is a proof that flooding colors the pixels in the order of their distance from the seed points. This is plausible but difficult to establish. We consider two versions of this claim, a weak version that claims the ordering separately within each color, and a strong version that claims one order for the entire collection of pixels. The proof is inductive, moving from the weak version after s steps to the strong version after s steps to the weak version after $s+1$ steps.

Ordered Coloring Lemma For every $s \geq 0$ and every two colors i and j, we have $\|A-x_i\| \prec \|Y-x_j\|$ for all $A \in F_{i,s}$ and all uncolored pixels Y that are eligible to be colored j after s steps.

Proof. This is the strong version of the claim and we get the weak version as a special case, when $i=j$. We begin by proving that the weak version implies the strong version. The only reason the latter is not trivial is that at step s a pixel A is colored with i. Since we use a priority queue to select A, all other existing pixel-color pairs in Q have priorities lower than the priority of (A,i). On the other hand, for a pixel $Y \in N(A)$ newly eligible to be colored i, by the weak version after s steps, (Y,i) also has priority lower than that of (A,i). By the strong version after $s-1$ steps, the priority of (A,i) is lower than that of all colored pixels we have seen so far. This implies the strong version after s steps.

We now prove that the strong version after s steps implies the weak version after $s+1$ steps. To get a contradiction, we let Y_0 be the first uncolored pixel that violates the claimed inequality for the weak version and we let s_0 be the number of steps after which this violation arises. We define a *predecessor* of a colored pixel as a neighboring pixel that received the same color earlier. By assumption, after $s < s_0$ steps, all predecessors of a pixel colored i are at least as close to x_i as this pixel. After s_0 steps, Y_0 has a neighbor A with color i that satisfies $\|Y_0-x_i\| \prec \|A-x_i\|$. Note that A is the only neighbor with color i, else Y_0 would have contradicted the inequality before the s_0 steps or it would have been colored before A. There are two cases to consider: when A and Y_0 share a side and when they share only a corner. Both cases are further decomposed into subcases, and for each subcase, we either derive a contradiction directly, or we reduce it to another subcase working our way up a path of predecessors one pixel closer to x_i. Since this path is finite, we get a contradiction eventually. To discuss the two cases, we assume the positions of A and Y_0 are as depicted in Fig. 4.

Case 1. A is the neighbor to the west of Y_0. Without loss of generality, assume that x_i lies in the lower right quadrant of Y_0. A has a predecessor at least as close to x_i but not neighboring Y_0. The only possibility is the south-west neighbor B. This further constrains the location of x_i to within a 45° wedge. The neighbor U to the south of A must have been colored before B, but with a different color k, else it would have violated the claimed inequality before Y_0 did. By inductive assumption, $\|U-x_k\| \prec \|B-x_i\| \prec \|B-x_k\|$. Similarly, $\|U-x_k\| \prec \|A-x_i\| \prec \|A-x_k\|$.

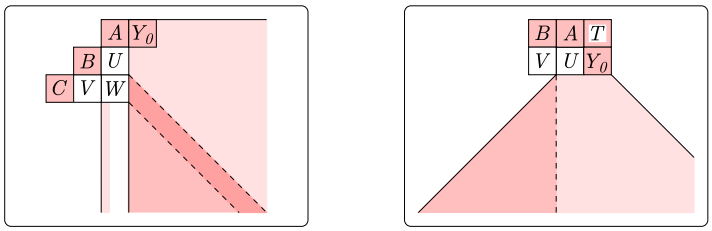

Fig. 4. Illustration of Case 1 on the left and Case 2 on the right. In both cases, the shaded right-angled wedge represents the possible locations of x_i.

We now have two constraints that express that A and B lie on the same side of the perpendicular bisector of x_i and x_k, namely

$$\|A - x_i\| \leq \|A - x_k\|,$$
$$\|B - x_i\| \leq \|B - x_k\|.$$

Since U lies inside the triangle x_iAB but not on the edge AB, the two inequalities imply $\|U - x_i\| < \|U - x_k\|$ and therefore $\|U - x_i\| \prec \|U - x_k\|$. But this is only possible if U has no neighbor of color i at the time it is colored. We consider three subcases:

Case 1.1. The south-east neighbor W of B is a predecessor of B. Therefore, U must have been colored before W. By inductive assumption, we have $\|U - x_i\| \prec \|U - x_k\| \prec \|W - x_i\|$. But this contradicts the inequality $\|W - x_i\| \prec \|U - x_i\|$, which we get from the restriction of x_i to the 45° wedge.

Case 1.2. The neighbor V to the south of B is a predecessor of B, and W is not. Therefore, we get $\|U - x_i\| \prec \|U - x_k\| \prec \|V - x_i\|$ as U must have been colored before V. This implies $\|U - x_i\| \leq \|V - x_i\|$, which further limits x_i to within a narrow diagonal strip, as indicated in Fig. 4 on the left. Let the color of W be $\ell \neq i$, and observe that W must have been colored before V, else it would have violated the claimed inequality before Y_0 did. Therefore, $\|W - x_\ell\| \prec \|V - x_i\| \prec \|V - x_\ell\|$, and similarly, $\|W - x_\ell\| \prec \|B - x_i\| \prec \|B - x_\ell\|$, which implies

$$\|B - x_i\| \leq \|B - x_\ell\|,$$
$$\|V - x_i\| \leq \|V - x_\ell\|.$$

Recalling that x_i lies inside the diagonal strip, we observe that W lies inside the triangle x_iBV but not on the edge BV. Hence, the two inequalities imply $\|W - x_i\| \prec \|W - x_\ell\|$. We now repeat the analysis of Case 1, substituting V, W for B, U.

Case 1.3. The south-west neighbor C of B is a predecessor of B, and V and W are not. The neighbor V to the south of B must have been colored before C, else it would have contradicted the claimed inequality before Y_0 did. We repeat the analysis of Case 1, substituting C, V, B for B, U, A.

We will shortly relax the condition on x_i by adding the column of pixels below A to the quadrant that contains x_i. Indeed, the only reason for not doing so right from the start is the condition $\|Y_0 - x_i\| \prec \|A - x_i\|$, which is violated for pixels in that column.

Case 2. A is the north-west neighbor of Y_0. Without loss of generality assume that x_i lies in the lower quadrant of Y_0; see Fig. 4 on the right. Recall that A is the only neighbor of Y_0 colored i. The neighbor U to the south of A must therefore have been colored before A, with a color k different from i, else it would have violated the claimed inequality before Y_0 did. Using the inductive assumption, we therefore get $\|U - x_k\| \prec \|A - x_i\| \prec \|A - x_k\|$. There are only two possible predecessors of A.

Case 2.1. The south-west neighbor V of A is a predecessor of A. We consider two subcases:

Case 2.1.1. $\|U - x_i\| \prec \|V - x_i\|$. Here U must have been colored before V, else it would have violated the claimed inequality before Y_0 did. We therefore get $\|U - x_k\| \prec \|V - x_i\| \prec \|V - x_k\|$. Since x_i lies in the lower right quadrant of A, we can now apply the analysis of Case 1, substituting V, U, A here for B, U, A there. Indeed, all steps of the analysis in Case 1 are valid for x_i in that quadrant, except for $\|Y_0 - x_i\| \prec \|A - x_i\|$, which now holds because Y_0 is the south-east rather than the east neighbor of A, as it was in the original description of Case 1.

Case 2.1.2. $\|V - x_i\| \prec \|U - x_i\|$. Here x_i lies in the lower left quadrant of A. Notice that $\|A - x_i\| \prec \|Y_0 - x_k\|$, else Y_0 would have been colored before A. Together with $\|Y_0 - x_i\| \prec \|A - x_i\|$ and $\|A - x_i\| \prec \|A - x_k\|$, we have

$$\|Y_0 - x_i\| \leq \|Y_0 - x_k\|,$$
$$\|A - x_i\| \leq \|A - x_k\|.$$

Since U lies inside the triangle $x_i A Y_0$ but not on the edge AY_0, the two inequalities imply $\|U - x_i\| \prec \|U - x_k\|$. Hence, U must have been colored before V, else color i would have taken precedence over color k. But this implies $\|U - x_k\| \prec \|V - x_i\|$ and therefore $\|U - x_i\| \prec \|V - x_i\|$, a contradiction to the assumption.

Case 2.2. The west neighbor B of A is a predecessor of A, and V is not. This constrains x_i to lie within the same 45° wedge considered in Case 2.1.2; see Fig. 4 on the right. Here, U must have been colored before B, else it would have violated the claimed inequality before Y_0 did. We can now repeat the analysis of Case 2, substituting B, V for A, U.

An amendment to Case 2.1 is in order. The reason is that we may encounter Case 2.2 first, one or more times, and then reach Case 2.1. If we do, Y_0 is no longer neighbor of the new A, which is equal to the original B or one of its predecessors. The analysis in Case 2.1.1 is unaffected by this difference, regardless of the actual position of Y_0, as Case 2.1.1 can reduce to Case 1 as long as x_i lies in the lower right quadrant of A. However, in Case 2.1.2, we need to find a new argument for $\|U - x_i\| \prec \|U - x_k\|$. To this end, let T be the neighbor to the east of the new A. Noting that T lies on a path of predecessors from the original pixel A back to x_i, we get $\|T - x_i\| \prec \|T - x_k\|$, else color k would have taken precedence over color i. Together with $\|A - x_i\| \prec \|A - x_k\|$, this gives

$$\|T - x_i\| \leq \|T - x_k\|,$$
$$\|A - x_i\| \leq \|A - x_k\|.$$

Since U lies in the triangle $x_i A T$ but not on AT, the above two inequalities imply $\|U - x_i\| \prec \|U - x_k\|$, as desired.

Let us reflect on the recursive structure of the inductive argument. Cases 1.2 and 1.3 reduce to Case 1, but one pixel closer to the end along the path back to x_i. Similarly, Case 2.2 reduces to Case 2, also one pixel closer to x_i. In contrast, Case 2.1 either leads to a contradiction or reduces to Case 1 without getting closer to x_i. But this reduction can happen only once throughout the recursive argument. We thus see that there is no cycle and each sequence of recursive arguments must end with a contradiction. This completes the case analysis and shows that the uncolored pixels have lower priority than the colored pixels throughout the algorithm. The claim follows. □

Recall the weak version of the claim, namely that flooding colors the pixels in a region F_i in the order of their distance from x_i. This implies that we can find a *monotonic* path from x_i to each pixel in F_i. All pixels on the path have color i, and the distance to x_i does not decrease along the path. We get such a path to A by tracing backward, from A to a predecessor to a predecessor of the predecessor and so on. We summarize:

Monotonic Path Lemma For each $A \in F_i$, there is a monotonic path from x_i to A within the region F_i.

Necks A *neck* is a pair of diagonally adjacent pixels of the same color whose two common neighbors both have colors that are different from that of the pair. We claim that the colors of the two common neighbors are also different from each other. In other words, a square of four pixels cannot have two necks. This property is useful because it implies that curves drawn within different color regions do not cross. Contradicting two necks is easy for Euclidean coloring since the bisector of the two seed points cannot separate the four pixel centers according to their color. For flooding, we need a proof, which we now present.

One Neck Lemma Flooding produces a coloring in which every square of four pixels has at most one neck.

Proof. Label the four pixels A, B, U, and V. Assuming two necks, we have that the algorithm colors the diagonally adjacent pixels A and B with i and the other two diagonally adjacent pixels U and V with $j \neq i$; see the fourth type in Fig. 2. Consider the following two right-angled wedges,

$$W_A = \{x \in \mathbb{R}^2 \mid \|A - x\| \leq \min\{\|U - x\|, \|V - x\|\}\},$$
$$W_B = \{x \in \mathbb{R}^2 \mid \|B - x\| \leq \min\{\|U - x\|, \|V - x\|\}\}.$$

Without loss of generality, we may assume that A gets colored first. Using the Ordered Coloring Lemma, we have $\|A - x_i\| \prec \|U - x_j\| \prec \|U - x_i\|$, else U would be colored i. Similarly, $\|A - x_i\| \prec \|V - x_j\| \prec \|V - x_i\|$, which implies $x_i \in W_A$. If B gets colored second, before U and V, then we also have $x_i \in W_B$. But W_B intersects W_A in a single point, namely the corner shared by the four pixels, thus we get a contradiction because this point does not have integer coordinates. So we may assume that U gets colored before B and before V. Therefore, $\|B - x_i\| \prec \|B - x_j\|$, else B would be colored j. Together

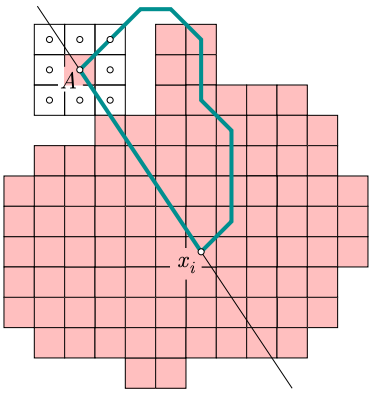

Fig. 5. The bulk of E_i after coloring the first s pixels by flooding.

with $\|U - x_j\| \prec \|U - x_i\|$ and $\|V - x_j\| \prec \|V - x_i\|$, this implies that the perpendicular bisector of x_i and x_j separates B from U and V. Because the bisector is constrained to pass between B on one side and U, V on the other, the half-plane that contains x_i and B intersects W_A in at most one point, namely the shared corner of the four pixels, which is again a contradiction. □

Bulk When we remove a point from S, the Euclidean Voronoi region of a remaining point either stays the same or it grows. The same is true for the regions E_i obtained by Euclidean coloring. Similarly, it is true for the bulk of E_i. However, it is not necessarily true for the regions obtained by flooding. For example, because of two sites x_j and x_k, the region E_i of x_i contains a debris, which during flooding is colored by j. After the removal of x_k, the debris becomes connected with the bulk of E_i and thus can be colored i. As a result, the region of color j obtained by flooding shrinks at a pixel. We prove a weaker statement, namely that the regions obtained by flooding contain the bulks of Euclidean coloring. It follows that the deletion of a seed point can shrink a region only by debris pixels, of which there are generally few.

Bulk Lemma Each region F_i constructed by flooding contains the bulk of E_i.

Proof. We prove that the prefixes of the bulk of E_i obtained by adding the pixels in order from x_i are connected. It follows that the pixels of the bulk are colored in this same order and with the same color. Let $B_{i,s}$ contain the s pixels of the bulk of E_i closest to the seed point. Thus

$$\{x_i\} = B_{i,1} \subseteq B_{i,2} \subseteq \ldots \subseteq B_{i,m} = B_i.$$

Suppose not all of the prefixes are connected and let $B_{i,s} = B_{i,s-1} \cup \{A\}$ be the first that is not connected. Draw the line that passes through A and x_i, as in Fig. 5. There are neighbors of A on both sides of the line whose distance from x_i is less than $\|A - x_i\|$. On the other hand, A belongs to the bulk of E_i, which is connected, so we can find a path within the bulk that connects A with x_i. Drawing it from pixel center to pixel center with straight edges in between, the path belongs to the Euclidean Voronoi region, V_i, by convexity. Hence, the region bordered by the path and the straight segment from A to x_i also belongs to V_i by convexity, and thus all pixel centers enclosed in this region belong to E_i. The region contains the pixel center of at least one neighbor B of A with $\|B - x_i\| \prec \|A - x_i\|$. This neighbor precedes A in the ordering of the pixels in the bulk of E_i and thus belongs to $B_{i,s-1}$, a contradiction to A being separated from $B_{i,s-1}$. □

3.2. Lassos go empty

In this subsection, we prove two technical results about monotonic paths which are instrumental in proving the lemmas needed for the validity of the digital Delaunay triangulation.

Lassos Given a seed point x_i and a pixel A colored i, a *lasso* consists of a monotonic path from x_i to A and the line segment from A back to x_i. We call x_i and A the *base points* and the line oriented from A to x_i the *base line* of the lasso. The *size* of the lasso is the distance between the two base points. When we compare the sizes of two lassos, we use the same tie-breaking mechanism as for the pairs of pixels. The lasso decomposes the plane into two components, an *inside* and an *outside*. The inside includes all pixels of the lasso.

Lasso Lemma There is no seed point inside a lasso.

Proof. To get a contradiction, assume the opposite. Let L_i be the smallest lasso that encloses one or more seed points, and let x_i and A be its base points. Let x_j be the enclosed seed point that is furthest from the base line; see Fig. 6. Given an

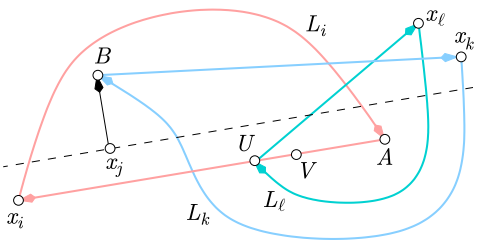

Fig. 6. The first lasso consists of a monotonic path from x_i to A and the line segment from A back to x_i. Inside the first lasso, we see another seed point, x_j. The second lasso defined by x_k and B encloses A. The third lasso defined by x_ℓ and U also encloses A.

oriented line, a pixel belongs to the corresponding *staircase* if its interior intersects the line or its boundary intersects the line and its center lies to the right of the line. The orientation of the line induces an ordering of the pixels along the line. Consider the staircase defined by the line that passes through x_j and is orthogonal to the base line of L_i. Traversing it from x_j and moving away from the base line, we let B be the first pixel not colored j. Because of the choice of B on the staircase orthogonal to the base line of L_i, we have $\|B - x_j\| \prec \|B - x_i\|$. It follows that B does not have color i either. Let $k \notin \{i, j\}$ be the color of B. By the Monotonic Path Lemma, there is a monotonic path from x_k to B. We do a case analysis in which every possibility leads to a contradiction. It will follow that x_j cannot exist.

Case 1. x_k and B lie on opposite sides of the line parallel to the base line of L_i and passes through x_j. Then we get $\|B - x_j\| \prec \|B - x_k\|$. On the other hand, by the choice of B, it has a neighbor with color j and with smaller distance from x_j, thus B would have been colored j instead of k, a contradiction.

Case 2. x_k and B lie on the same side of that line, as in Fig. 6. By the extremality of the choice of x_j, the seed point x_k cannot lie inside the lasso L_i. Let L_k be the lasso consisting of a monotonic path from x_k to B and the line segment from B back to x_k. By the One Neck Lemma, the monotonic path cannot cross the path from x_i to A and therefore must cross the base line of L_i. Hence L_k encloses either x_i or A. For the former, we have

$$\|B - x_k\| \prec \|B - x_j\| \prec \|B - x_i\|,$$

and by drawing a line from x_i through B, we get a pixel on the path from x_i to A that is even further from x_i than B. It follows that $\|B - x_i\| \prec \|A - x_i\|$, thus $\|B - x_k\| \prec \|A - x_i\|$. Hence, the size of L_k is less than the size of L_i, and by the extremality of the choice of L_i, L_k cannot enclose x_i. The only remaining possibility is that L_k encloses A, as in Fig. 6. The final contradiction will rest on the properties of two particular pixels which we now describe. Traverse the staircase from A to x_i and let U be the first pixel whose color is not i. Let V be the predecessor of U in the staircase. Such pixels U and V exist because the staircase crosses the monotonic path from x_k to B, where pixels have color $k \neq i$. However, we may reach U before crossing the path. Let $\ell \neq i$ be the color of U. We have U colored before V; otherwise, the coloring of V would result in putting (U, i) into the priority queue, a contradiction to the Ordered Coloring Lemma. So we have $\|U - x_\ell\| \prec \|V - x_i\|$. This implies $\|U - x_\ell\| \prec \|A - x_i\|$. Let L_ℓ be a lasso with base points x_ℓ and U. Because of the extremal choice of the first lasso, L_ℓ cannot enclose any seed points. There are three cases to consider.

Case 2.1. $\ell = k$. We have $\|U - x_k\| \prec \|V - x_k\|$, else V would have been colored k. But this also implies that x_k and U lies on the same side of the perpendicular bisector of U and V, thus L_ℓ encloses V. Draw a half-line from x_k through V, we get a pixel on the monotonic path from x_k to U whose distance from x_k exceeds $\|U - x_k\|$. This contradicts the monotonicity of the path.

Case 2.2. $\ell \neq k$ and x_ℓ and B are on opposite sides of the line parallel to the base line of L_i and passes through x_j. The lasso L_ℓ thus either encloses x_k or B, since x_ℓ must lie outside L_k. If it encloses B, it also encloses x_j because the two pixels are connected by a piece of a staircase of color $j \neq \ell$. As mentioned earlier, L_ℓ cannot enclose a seed point so we have a contradiction in either case.

Case 2.3. $\ell \neq k$ and x_ℓ and B are on the same side of the line passing through x_j. Avoiding to enclose x_i, the only possibility for the monotonic path from x_ℓ to U is as shown in Fig. 6. Here we use $\|U - x_\ell\| \prec \|V - x_i\| \prec \|V - x_\ell\|$. But V is enclosed by L_ℓ, so we can draw a directed line from x_ℓ through V, as in Case 2.1. This gives a pixel on the path from x_ℓ to U whose distance from x_ℓ exceeds $\|U - x_\ell\|$, again a contradiction.

This implies that x_j does not exist, which proves the claim. □

Spliced lassos We extend the lasso to a slightly more elaborate construction. Let x_i and x_ℓ be two different seed points and $A \in F_i$, $D \in F_\ell$ two pixels. For the construction, we require that A and D are neighbors sharing a common side or at least a common corner, but in the latter case the remaining two pixels sharing the same corner must have colors that are different

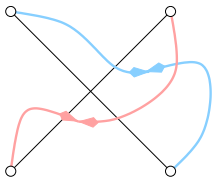

Fig. 7. The two edges cross and so do the two spliced paths.

from each other. Drawing a monotonic path of color i from x_i to A, another monotonic path of color ℓ from x_ℓ to D, and the line segment between x_i and x_ℓ, we get what we call a *spliced lasso*. We refer to x_i and x_ℓ as its *base points* and the connecting line as its *base line*. Similar to the lasso, a spliced lasso decomposes the plane into an inside and an outside. Another monotonic path cannot cross the spliced paths. Indeed, the only weak point is the corner at which the two paths are spliced, but we cannot cross there either if that corner is a digital Voronoi vertex. The only possibility to go from outside to inside the spliced lasso is therefore to cross the base line.

Spliced Lasso Lemma There is no seed point inside a spliced lasso.

Proof. Assume to the contrary that we have a spliced lasso with base points x_i and x_ℓ that encloses other seed points. Let x_j be the seed point that maximizes the distance from the base line. Starting at x_j, construct a piece of a staircase orthogonal to and moving away from the base line. Let B be the first pixel not colored j. As before, we argue that the color of B is $k \notin \{i, j, \ell\}$ and we construct a lasso L_k connecting x_k to B and back. If x_k and B lie on opposite sides of the line passing through x_j and parallel to the base line of the spliced lasso, then we get $\|B - x_j\| \prec \|B - x_k\|$, contradicting the Ordered Coloring Lemma. On the other hand, if x_k and B lie on the same side of the line then the extremal choice of x_j prohibits that x_k be inside the spliced lasso. Hence L_k encloses either x_i or x_ℓ, contradicting the Lasso Lemma. □

3.3. The triangulation is valid

In this subsection, we use the two lasso lemmas to show that the digital Delaunay triangulation is a valid solution to our triangulation problem.

No crossing edges Recall that $\mathcal{E}$ is the multiset of edges identified when we dualize the digital Voronoi diagram obtained by flooding. We say two edges *cross* each other if the endpoints of each lie on opposite sides of the line spanned by the other. Here we require implicitly that no three of the four endpoints lie on a common line. In particular, two copies of the same edge are not considered to cross each other.

No Crossing Lemma No two edges in $\mathcal{E}$ cross each other.

Proof. Assume the opposite and let x_ix_j and x_px_q be two edges that cross. The four seed points thus form a convex quadrilateral whose diagonals are the two edges; see Fig. 7 on the left. Let A, B and C, D be the corresponding pixels in the two digital Voronoi vertices identifying the two edges. Connecting x_i to A and x_j to B by monotonic paths, we get a first spliced lasso, which we denote as L_{ij}. Similarly, we construct a second spliced lasso, L_{pq}. By the Spliced Lasso Lemma, the two cannot enclose any seed points. This implies that the spliced paths of L_{ij} cross the edge x_px_q an odd number of times, and the spliced paths of L_{pq} cross x_ix_j an odd number of times. But then the two paths must cross, which is prohibited by the One Neck Lemma. □

Consistent orientation Recall that $\mathcal{T}$ is the multiset of triangles in the digital Delaunay triangulation. As we will see shortly, $\mathcal{T}$ is in fact a set. The sequence of the three vertices of a triangle implies an *orientation*, which is either clockwise or counterclockwise. Assuming $x_ix_jx_k$ is in $\mathcal{T}$, it has a dual digital Voronoi vertex that contains three pixels colored i, j, and k. By definition of digital Voronoi vertex, the color of the fourth pixel is different from the color of the diagonally opposite pixel in the 2-by-2 array; see Fig. 2. We say the orientation of $x_ix_jx_k$ is *consistent* with the three corresponding pixels if both are clockwise or both are counterclockwise.

Consistent Orientation Lemma The orientation of each triangle in $\mathcal{T}$ is consistent with the orientation of the corresponding pixels in the dual digital Voronoi vertex.

Proof. Let $x_ix_jx_k$ be a triangle in $\mathcal{T}$ with corresponding pixels A, B, C in the dual digital vertex. Note that in a digital Voronoi vertex, two diagonal pixels cannot have the same color. Without loss of generality, we assume that horizontally x_k is between x_i and x_j, and the triangle has clockwise orientation. That is, x_k lies between the two vertical lines through x_i

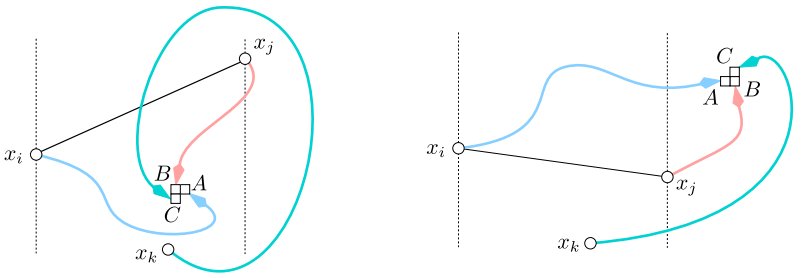

Fig. 8. If the orientation of $x_i x_j x_k$ is not consistent with ABC, either a lasso (left) or a spliced lasso (right) encloses a seed point.

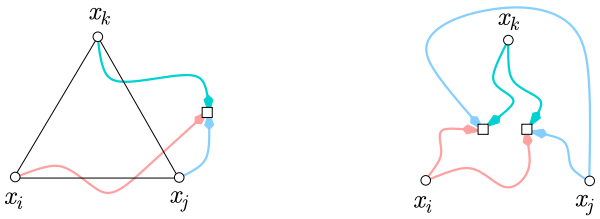

Fig. 9. Left: the insides of the three spliced lassos cover the triangle. Right: Planar drawing of a complete bipartite graph.

and x_j respectively, and is below the line through x_i and x_j. To get a contradiction, we suppose the orientation of ABC is counterclockwise. We analyze two possible cases of the position of C.

Case 1. C is below the line through x_i and x_j; see Fig. 8 on the left. The spliced lasso from x_i to x_j encloses C, else either the lasso from x_i to A encloses x_j or the lasso from x_j to B encloses x_i, contradicting the Lasso Lemma. On the other hand, x_k is outside this spliced lasso. As such, for the monotonic path from x_k to reach C, it has to enclose either x_i or x_j, contradicting the Lasso Lemma.

Case 2. C is above the line through x_i and x_j; see Fig. 8 on the right. Also, the lasso from x_i to A cannot enclose x_j, and that from x_j to B cannot enclose x_i. As such, C always lies outside the spliced lasso between x_i and x_j. Since x_k is below the line through x_i and x_j while C is above this line, either the spliced lasso from x_k to x_i encloses x_j, or the spliced lasso from x_k to x_j encloses x_i, contradicting the Spliced Lasso Lemma. □

No nesting triangles Using the consistent orientation between the triangles and the pixels in the dual digital Voronoi vertices, we now show that no two triangles in $\mathcal{T}$ are nested. This includes the case in which the two triangles are the same.

No Nesting Lemma No two triangles in $\mathcal{T}$ are nested or the same.

Proof. Let $x_i x_j x_k$ and $x_p x_q x_r$ be two triangles with different dual digital Voronoi vertices. We first consider the case in which the second triangle has at least one new seed point, $x_r \notin \{x_i, x_j, x_k\}$. To show that the triangles are not nested, it suffices to prove that x_r does not lie inside the triangle $x_i x_j x_k$. As usual, we draw the monotonic paths from the seed points to the dual vertex; see Fig. 9 on the left. Splicing the paths in pairs, we get three spliced lassos. To complete the proof, we think of the insides of the three spliced lassos as the projection of the three faces of a tetrahedron built on top of the triangle. Observe that the projections cover the triangle. If x_r lies inside the triangle then it is enclosed by one of the spliced lassos, a contradiction to the Spliced Lasso Lemma.

We consider second the case in which the two triangles are the same. Connecting the seed points with the dual vertices, we get a complete bipartite graph with five vertices and six edges; see Fig. 9 on the right. We may assume that the drawing of the graph is plane, that is, the paths do not cross. Indeed, two paths can only cross if they have the same color, and in this case we can cut and splice the pieces to remove the crossing. Now observe how the three seed points connect to the two Voronoi vertices. If x_i, x_j, and x_k connect in a clockwise order to one vertex then they connect in a counterclockwise order to the other vertex. It follows that one of the two triangles contradicts the Consistent Orientation Lemma. □

We have now completed the proof that the digital Delaunay triangulation satisfied Condition III. It remains to prove that it has the right topology, that is, it satisfies Condition II.

Connectivity If an edge in $\mathcal{E}$ belongs to three or more triangles then there are two that are nested, the same, or have crossing edges. This would contradict the No Crossing Lemma or the No Nesting Lemma, implying that each edge in $\mathcal{E}$ belongs to either one or two triangles in $\mathcal{T}$. In the latter case, the two triangles lie on opposite sides of the edge. We argue that each edge belongs to exactly two triangles. In other words, there are no holes in the triangulation. We begin by proving that each region F_i is simply connected. In the plane this is equivalent to being connected and having no holes.

Simply Connectedness Lemma All digital Voronoi regions constructed by flooding are simply connected.

Proof. Suppose there is a region F_i that is not simply connected. By construction, F_i is connected, so we can splice two monotonic paths to form a loop going around one of the holes. If possible, we do the splicing along a shared side of two pixels. If this is not possible, we splice the two paths at a shared pixel corner and recall from the One Neck Lemma that the other two pixels sharing that corner have colors different from each other. The two spliced paths both originate at the seed point, so we have a spliced lasso with the base line being a single point. Drawing the piece of a staircase emanating from a presumed seed point, x_j, inside the lasso away from that pixel, the proof of the Spliced Lasso Lemma still applies. It follows that the hole contains no seed points. But then the hole must be empty, else we could construct a monotonic path connecting the hole to the outside. □

Write ∂F_i for the set of sides and corners shared between pixels in F_i and pixels not in F_i. We can orient the sides so that F_i lies locally to the left and the resulting curve is connected and goes around the region in a counterclockwise order. This construction is unambiguous except in one important special case. If F_i has a neck, there is a corner shared by four sides. In this case, we duplicate the corner and we connect the sides in pairs so that the curve does not cross the neck. This gives a cyclic sequence in which each corner appears only once. Replacing every corner by the four pixels around it, we get a cyclic sequence of 2-by-2 arrays. Each such array contains at least one and at most three pixels from F_i. Since every side is either vertical or horizontal, any two contiguous arrays overlap in exactly two pixels, one colored i and the other $j \neq i$. As we walk along the sequence, the color j can only change when we pass through a digital Voronoi vertex. Indeed, the only other array with at least three different colors is the neck, but it shares the color j with both its predecessor and its successor along the sequence.

This observation allows us to interpret the information as we read along the sequence of arrays. Specifically, the digital Voronoi vertices decompose the cycle into *segments* within which the color $j \neq i$ of the shared pixels is constant. It follows that two contiguous digital Voronoi vertices share one pair of colors. In other words, the segment gives rise to two triangles sharing a common edge. It follows that each edge in $\mathcal{E}$ belongs to at least two triangles in $\mathcal{T}$. Because of the No Nesting Lemma, this number is at most two and therefore exactly two. Because the regions are simply connected, we get exactly one cycle of arrays for each F_i, which implies that the triangles incident to a vertex x_i form a ring around the seed point. Hence, the triangulation has the topology of a 2-manifold. To conclude the argument, we use the Nerve Theorem [4, Section III.2], which applies because each region is simply connected and it intersects each other region in a point or a connected segment, if at all. This theorem implies that the triangulation has the same homotopy type as the union of regions. The outside region, F_0, complements the texture to form a 2-dimensional sphere, so the only remaining possibility is that we have a triangulation of the 2-sphere.

This completes the proof that the digital Delaunay triangulation satisfies Condition II and is therefore a valid solution to our triangulation problem.

4. Discussion

The main contribution of this paper is a proof that the digital Delaunay triangulation has the geometric and topological properties we usually expect from a triangulation in the plane: its edges do not cross and after connecting the boundary to a dummy vertex at infinity, we get a triangulation of the 2-dimensional sphere. We get these properties if we dualize the collection of digital Voronoi regions colored by flooding. In contrast to coloring by Euclidean distance to the seed points, flooding forms regions that are connected. We can therefore think of flooding as a method to remove the topological noise caused by the digital approximation of the real plane. The most interesting next question is the extension of our correctness proof to 3-dimensional voxel arrays.

Acknowledgement

We would like to thank the anonymous reviewer for the valuable feedback to improve the paper. The research of the first and the third author is partially supported by NUS under grant R-252-000-337-112. The research of the second author is partially supported by NSF under grant DBI-0820624 and by DARPA under grants HR011-05-1-0057 and HR0011-09-0065.

Appendix A. Overview of proof steps

The proof consists of three major steps, each consisting of a small number of lemmas. In Fig. 10, we show the steps as three dashed boxes with implications in sequences; they correspond to Sections 3.1, 3.2, and 3.3. Each smaller, shaded

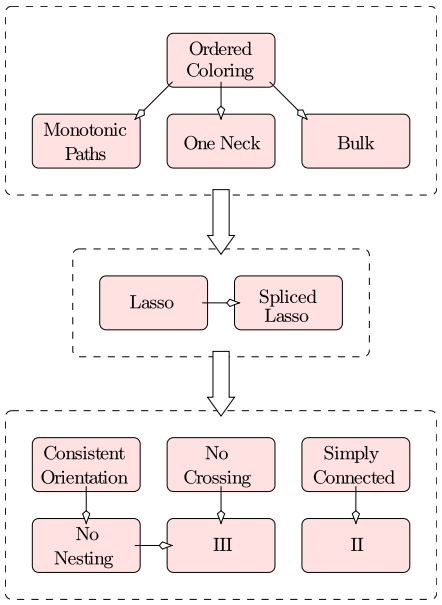

Fig. 10. Steps in the proof and their dependencies.

box is a lemma. The most important are the Ordered Coloring Lemma, which encapsulates most of the digital geometry reasoning, and the Lasso Lemma, which forms the primary topological tool used to prove the rest.

Appendix B. Notation

Table 1 provides a list of notation used in this paper.

Table 1
Notation for geometric concepts, sets, functions, vectors, variables.

$S = \{x_1, \ldots, x_n\}$	data set, seed points
$x_i x_j, x_i x_j x_k$	edges, triangles
A, B, C, U, V, Y_0	pixels
V_i, E_i, F_i	Euclidean, digital Voronoi regions
B_i	bulk
$F_{i,s}, B_{i,s}, Q_s$	region, bulk, queue after s steps
L_i, L_{ij}	lasso, spliced lasso
$S, \mathcal{E}, \mathcal{T}$	Delaunay vertices, edges, triangles

References

[1] T.T. Cao, K. Tang, A. Mohamed, T.S. Tan, Parallel banding algorithm to compute exact distance transform with the GPU, in: I3D '10: Proc. ACM Symp. Interactive 3D Graphics and Games, ACM, New York, NY, USA, 2010, pp. 83–90.
[2] O. Cuisenaire, Distance transformations: fast algorithms and applications to medical image processing, Ph.D. thesis, Universite Catholique de Louvain (UCL), Louvain-la-Neuve, Belgium, 1999.
[3] P.E. Danielsson, Euclidean distance mapping, Comput. Graph. Image Process. 14 (1980) 227–248, http://dx.doi.org/10.1016/0146-664X(80)90054-4.
[4] H. Edelsbrunner, J. Harer, Computational Topology: An Introduction, American Mathematical Soc., 2009.
[5] S. Fortune, Voronoi diagrams and Delaunay triangulations, in: J.E. Goodman, J. O'Rourke (Eds.), Handbook of Discrete and Computational Geometry, CRC Press, Inc., Boca Raton, FL, USA, 1997, pp. 377–388.
[6] K.E. Hoff III, J. Keyser, M. Lin, D. Manocha, T. Culver, Fast computation of generalized Voronoi diagrams using graphics hardware, in: Proc. ACM SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999, pp. 277–286.
[7] J. Lee, J. Kim, S. Seo, S. Kim, J. Park, H. Kim, T.T. Dao, Y. Cho, S.J. Seo, S.H. Lee, S.M. Cho, H.J. Song, S.B. Suh, J.D. Choi, An OpenCL framework for heterogeneous multicores with local memory, in: PACT '10: Proc. Intern. Conf. Parallel Architectures and Compilation Techniques, ACM, New York, NY, USA, 2010, pp. 193–204.
[8] J. Nickolls, I. Buck, M. Garland, K. Skadron, Scalable parallel programming with CUDA, Queue 6 (2) (2008) 40–538, http://dx.doi.org/10.1145/1365490.1365500.
[9] J.D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krüger, A. Lefohn, T.J. Purcell, A survey of general-purpose computation on graphics hardware, in: Eurographics 2005, in: State of the Art Reports, 2005, pp. 21–51.
[10] M. Qi, T.T. Cao, T.S. Tan, Computing 2D constrained Delaunay triangulation using the GPU, in: I3D '12: Proc. Symp. Interactive 3D Graphics and Games, ACM, New York, NY, USA, 2012, pp. 39–46.

[11] G. Rong, T.S. Tan, Jump flooding in GPU with applications to Voronoi diagram and distance transform, in: I3D '06: Proc. Symp. Interactive 3D Graphics and Games, ACM, New York, NY, USA, 2006, pp. 109–116.
[12] G. Rong, T.S. Tan, T.T. Cao, Stephanus, Computing two-dimensional Delaunay triangulation using graphics hardware, in: I3D '08: Proc. Symp. Interactive 3D Graphics and Games, ACM, New York, NY, USA, 2008, pp. 89–97.
[13] J. Schneider, M. Kraus, R. Westermann, GPU-based real-time discrete Euclidean distance transforms with precise error bounds, in: Intern. Conf. Computer Vision Theory and Applications (VISAPP), Springer, Berlin/Heidelberg, 2009, pp. 435–442.
[14] J. Shewchuk, Triangle: engineering a 2D quality mesh generator and Delaunay triangulator, in: M. Lin, D. Manocha (Eds.), Applied Computational Geometry Towards Geometric Engineering, in: Lecture Notes in Computer Science, vol. 1148, Springer, Berlin/Heidelberg, 1996, pp. 203–222.
[15] A. Sud, N. Govindaraju, R. Gayle, D. Manocha, Interactive 3D distance field computation using linear factorization, in: I3D '06: Proc. Symp. Interactive 3D Graphics and Games, ACM, New York, NY, USA, 2006, pp. 117–124.