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.