Fast Computation of Generalized Voronoi Diagrams Using Graphics Hardware

Kenneth E. Hoff III, Tim Culver, John Keyser, Ming Lin, and Dinesh Manocha

University of North Carolina at Chapel Hill
SIGGRAPH ‘99
What is a Voronoi Diagram?

Given a collection of geometric primitives, it is a subdivision of space into cells such that all points in a cell are closer to one primitive than to any other.
Ordinary
• Point sites
• Nearest Euclidean distance

Generalized
• Higher-order site geometry
• Varying distance metrics
Why Should We Compute Them?

It is a fundamental concept

<table>
<thead>
<tr>
<th>Name</th>
<th>Field</th>
<th>Year</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Descartes</td>
<td>Astronomy</td>
<td>1644</td>
<td>“Heavens”</td>
</tr>
<tr>
<td>Dirichlet</td>
<td>Math</td>
<td>1850</td>
<td>Dirichlet tessellation</td>
</tr>
<tr>
<td>Voronoi</td>
<td>Math</td>
<td>1908</td>
<td>Voronoi diagram</td>
</tr>
<tr>
<td>Boldyrev</td>
<td>Geology</td>
<td>1909</td>
<td>area of influence polygons</td>
</tr>
<tr>
<td>Thiessen</td>
<td>Meteorology</td>
<td>1911</td>
<td>Theissen polygons</td>
</tr>
<tr>
<td>Niggli</td>
<td>Crystallography</td>
<td>1927</td>
<td>domains of action</td>
</tr>
<tr>
<td>Wigner &amp; Seitz</td>
<td>Physics</td>
<td>1933</td>
<td>Wigner-Seitz regions</td>
</tr>
<tr>
<td>Frank &amp; Casper</td>
<td>Physics</td>
<td>1958</td>
<td>atom domains</td>
</tr>
<tr>
<td>Brown</td>
<td>Ecology</td>
<td>1965</td>
<td>areas potentially available</td>
</tr>
<tr>
<td>Mead</td>
<td>Ecology</td>
<td>1966</td>
<td>plant polygons</td>
</tr>
<tr>
<td>Hoofd et al.</td>
<td>Anatomy</td>
<td>1985</td>
<td>capillary domains</td>
</tr>
<tr>
<td>Icke</td>
<td>Astronomy</td>
<td>1987</td>
<td>Voronoi diagram</td>
</tr>
</tbody>
</table>
Why Should We Compute Them?

Useful in a wide variety of applications

Collision Detection
Surface Reconstruction
Robot Motion Planning
Non-Photorealistic Rendering
Surface Simplification
Mesh Generation
Shape Analysis
Outline

• Generalized Voronoi Diagram Computation
  – Exact and Approximate Algorithms
  – Previous Work
  – Our Goal

• Basic Idea
• Our Approach
• Basic Queries
• Applications
• Conclusion
Generalized Voronoi Diagram Computation

“Exact” Algorithms

Previous work
- Lee82
- Chiang92
- Okabe92
- Dutta93
- Milenkovic93
- Hoffmann94
- Sherbrooke95
- Held97
- Culver99
Previous Work: “Exact” Algorithms

• Compute analytic boundaries

but...

• Boundaries composed of high-degree curves and surfaces and their intersections
• Complex and difficult to implement
• Robustness and accuracy problems
Generalized Voronoi Diagram Computation

**Exact Algorithm**

**Approximate Algorithms**

Analytic Boundary  
Discretize Sites  
Discretize Space

Previous work
Lavender92, Sheehy95, Vleugels 95 & 96, Teichmann97
Previous Work: Approximate Algorithms

• Provide practical solutions

but...

• Difficult to error-bound
• Restricted to static geometry
• Relatively slow
Our Goal

Approximate generalized Voronoi diagram computation that is:

- Simple to understand and implement
- Easily generalized
- Efficient and practical

with all sources of error fully enumerated
Outline

• Generalized Voronoi Diagram Computation
• Basic Idea
  – Brute-force Algorithm
  – Cone Drawing
  – Graphics Hardware Acceleration
• Our Approach
• Basic Queries
• Applications
• Conclusion
Brute-force Algorithm

Record ID of the closest site to each sample point

Coarse point-sampling result

Finer point-sampling result
Graphics Hardware Acceleration

Simply rasterize the cones using graphics hardware

Our 2-part discrete Voronoi diagram representation

Color Buffer

Depth Buffer

Site IDs

Haeberli90, Woo97
Cone Drawing

To visualize Voronoi diagram for points in 2D…

Perspective, 3/4 view

Parallel, top view

Dirichlet 1850 & Voronoi 1908
Outline

- Generalized Voronoi Diagram Computation
- Basic Idea
- Our Approach
  - Meshing Distance Function
  - Generalizations
  - 3D
  - Sources of Error
- Basic Queries
- Applications
- Conclusion
The Distance Function

Evaluate distance at each pixel for all sites
Accelerate using graphics hardware
Approximating the Distance Function

Avoid per-pixel distance evaluation
Point-sample the distance function
Reconstruct by rendering polygonal mesh
Meshing the Distance Function

Shape of distance function for a 2D point is a cone

Need a bounded-error tessellation of the cone
Shape of Distance Functions

Sweep apex of cone along higher-order site to obtain the shape of the distance function
Example Distance Meshes
Tessellate curve into a polyline
Tessellation error is added to meshing error
Real-time Motion Planning: Static Scene

Plan motion of piano (arrow) through 100K triangle model

Distance buffer of floorplan used as potential field
Real-time Motion Planning: Dynamic Scene

Plan motion of music stand around moving furniture

Distance buffer of floor-plan used as potential field
Conclusion

Meshing Distance Functions
Graphics Hardware Acceleration

+ Brute-force Approach

Fast and Simple, Approximate
Generalized Voronoi Diagrams
Bounded Error