Linear Time Simple

POLYGON

Triangulation

Landon Boyd
CPSC 516 (Computational Geometry)
April 15, 2009
Simple Polygon:

- no self-intersections
- no holes

Triangulation:
Why Triangulate?

Computational Geometry Problems:
- Art gallery
- Link distance
- Shortest monotone path

Graphics and Physics-based Simulation
- Quality of triangles matter
- eg. Constrained Delaunay Triangulation (CDT)
Timeline of Polygon Triangulation Algorithms

<table>
<thead>
<tr>
<th>Year</th>
<th>Complexity</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1978</td>
<td>$O(n \log n)$</td>
<td>Garey et al.</td>
</tr>
<tr>
<td>1982</td>
<td>$O(n \log n)$</td>
<td>Chazelle</td>
</tr>
<tr>
<td>1988</td>
<td>$O(n \log \log n)$</td>
<td>Kirkpatrick et al.</td>
</tr>
<tr>
<td>1989</td>
<td>$O(n \log^* n)$ (\text{exp.})</td>
<td>Clarkson et al.</td>
</tr>
<tr>
<td>1990</td>
<td>$O(n \log \log n)$</td>
<td>Kirkpatrick et al.</td>
</tr>
<tr>
<td>1991</td>
<td>$O(n \log^* n)$ (\text{exp.})</td>
<td>Seidel</td>
</tr>
<tr>
<td></td>
<td>$O(n)$</td>
<td>Chazelle</td>
</tr>
<tr>
<td>2000</td>
<td>$O(n)$ (\text{exp.})</td>
<td>Amato et al.</td>
</tr>
</tbody>
</table>
Amato, Goodrich and Ramos

- $O(n)$ expected time
- Simpler than Chazelle 1991
- Implementable?
- Explainable in three slides?

Generate trapezoidal map in 3 phases:
- Sampling
- Bottom-Up
- Top-Down

Generate triangulation from trapezoidal map
Sampling

- Split the polygon into subchains at $k$ levels
- $k = O(\log^* n)$
- Randomly select subchains at each level

$L[0]$: one subchain of length $k$
\[ \lambda_i = \log^2(\lambda_{i-1}) \]

$L[k]$: $n$ subchains of length 1

$K[i]$: random samples from $L[i]$
\[ p_i = 1/\lambda_i^{3/2} \]
Bottom-Up Phase

- For each chain $l$ in $L[i]$ ($i=1..k$), generate:
  - The *conformal chain-trapezoidation* of its *sampled* subchains
  - The *conflict list* for each resulting *chain-trapezoid*

conformal chain-trapezoidation:

conflict list:
  - *all* subchains infringing upon the trapezoid
Top-Down Phase

For every level $i \ (i=1..k)$, compute:
- combined c. c.-t. of all the sampled (K) chains
- conflicts between each c.-t. and all chains in L[$i$]
- output of final iteration is polygon trapezoidal map
- uses data collected in bottom-up phase...
- see my report for more details!
Triangulation from Trapezoidal Map

- O(n) – Fournier, Montuno 1984
- Used by Seidel 1991 and others

(a) trapezoids inside polygon

(b) diagonals between vertices

(c) subpolygons easy to triangulate: 1 edge + y-monotone chain

diagram from Seidel 1991
Implementation

The Plan:
- Implement in C++
- Use Narkhede and Manosha Seidel code for...
  - File I/O
  - Generating trap. maps
  - Triangulating the final trap. map

Reality:
- Completed...
  - Sampling
  - First step of bottom-up phase
  - Visualization of trap. maps
Results

Can you identify this input data?

- 3000 vertices
Results

Some debugging output:

n=2961
k=5
p=0.00199981
p=0.0142668
p=0.0441942
p=0.125
p=1
lambda[0] = 2961
lambda[1] = 63
lambda[2] = 17
lambda[3] = 8
lambda[4] = 4
lambda[5] = 1
L[0].size=1
L[1].size=47
L[2].size=188
L[3].size=517
L[4].size=846
L[5].size=2961
K[0].size=0
K[1].size=1
K[2].size=2
K[3].size=23
K[4].size=103
K[5].size=2961
Results

Level 0 Trapezoidal Map:

- 63 edges
Results

Level 2 Trapezoidal Map:

- 8+1 edges
Level 4 Trapezoidal Map:

- 4 edges
Results

Seidel Triangulation:
- Narkhede Manosha
- 96 ms
Results

Constrained Delaunay Triangulation:
- Shewchuk's *triangle*
- 65 ms
Conclusion

Amato et al. 2000:
  - Implementable? I think so!

I want to triangulate a polygon right now. What should I do?
  - Use triangle.