Automatically repairing invalid polygons with a constrained triangulation

Hugo Ledoux    Ken Arroyo Ohori    Martijn Meijers

Agile 2012 (Avignon, France)
2012–04–25
Validation of polygon = a solved problem

OGC Simple Features and ISO19107 rules:

1. no self-intersection
2. closed boundaries
3. rings can touch but not overlap
4. no duplicate points
5. no dangling edges
6. connected interior
7. etc
If it’s broken then repair it. But how?

Errors are highlighted, but not repaired. One has to manually fix them.

The problem:

Given an invalid polygon, how to *automatically* repair it?
Surprisingly very little written on the topic:

1. “cleaning” for display purposes
2. Practitioners say: “buffer-by-0”
3. PostGIS 2.0’s ST_MakeValid
Related work: cleaning for display purposes

GRASS

ArcGIS
Related work: “buffer-by-0”
Related work: “buffer-by-0”
Related work: “buffer-by-0”

planar graph is constructed
Related work: “buffer-by-0”
Related work: PostGIS 2.0’s ST_MakeValid

- high-level automatic repair function
- diff functions called depending on geometric and topological configurations of rings
- based on construction of planar graph (GEOS is used)
- not documented (‘‘read the code’’) = predicting behaviour is difficult
- very slow for big polygons
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:

1. CT of input polygon
2. Labelling of triangles (*outside* or *inside*)
3. Reconstruction of the rings by depth-first search on the dual graph

![Diagram of a polygon with a dangling piece and a ring not closed.](image-url)
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:

1. CT of input polygon
2. Labelling of triangles (*outside* or *inside*)
3. Reconstruction of the rings by depth-first search on the dual graph
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:

1. CT of input polygon
2. Labelling of triangles (outside or inside)
3. Reconstruction of the rings by depth-first search on the dual graph
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:

1. CT of input polygon
2. Labelling of triangles (*outside* or *inside*)
3. Reconstruction of the rings by depth-first search on the dual graph
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:

1. CT of input polygon
2. Labelling of triangles (outside or inside)
3. Reconstruction of the rings by depth-first search on the dual graph
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:
1. CT of input polygon
2. labelling of triangles (outside or inside)
3. reconstruction of the rings by depth-first search on the dual graph
Our approach = constrained triangulation (CT)

Repairing = 3 simple steps:

1. CT of input polygon
2. Labelling of triangles (outside or inside)
3. Reconstruction of the rings by depth-first search on the dual graph
Examples of polygons automatically repaired

\[ p_1 = \]

\[ p_3 = \]

\[ p_4 = \]

\[ p_7 = \]

\[ p_8 = \]

\[ p_9 = \]

\[ p_{10} = \]

\[ p_{12} = \]

\[ p_{13} = \]
Experiments with big polygons: CORINE2006

- 32,473 points
- 346 rings
- 2,412 points
- 10 rings
- 102,272 points
- 299 rings
Experiments with big polygons: CORINE2006
Experiments with big polygons: CORINE2006

<table>
<thead>
<tr>
<th>Points</th>
<th>Rings</th>
<th>prepair</th>
<th>ST_MakeValid</th>
</tr>
</thead>
<tbody>
<tr>
<td>EU-47552</td>
<td>32 473</td>
<td>346</td>
<td>2 412</td>
</tr>
<tr>
<td>EU-47997</td>
<td>32 473</td>
<td>346</td>
<td>2 412</td>
</tr>
<tr>
<td>EU-180927</td>
<td>102 272</td>
<td>299</td>
<td>102 272</td>
</tr>
</tbody>
</table>
The code is freely available: help us improve it!

- http://tudelft-gist.github.com/prepair
- only around 300 lines of code
- BSD license
- (CGAL is GPL)
Future work: alternative repair paradigms

Point-set topology: $p = \text{outerring} \setminus \{\text{innerrings}\}$
Future work: alternative repair paradigms

Point-set topology: $p = \text{outerring} \setminus \{\text{innerrings}\}$
Future work: alternative repair paradigms

Point-set topology: $p = \text{outerring} \setminus \{\text{innerrings}\}$
Point-set topology: $p = \text{outerring} \setminus \{\text{innerrings}\}$
Future work: validation with a tolerance
Thanks for your attention

Hugo Ledoux
h.ledoux@tudelft.nl

Ken Arroyo Ohori
g.a.k.arroyoohori@tudelft.nl

Martijn Meijers
b.m.meijers@tudelft.nl

tudelft-gist.github.com/prepair