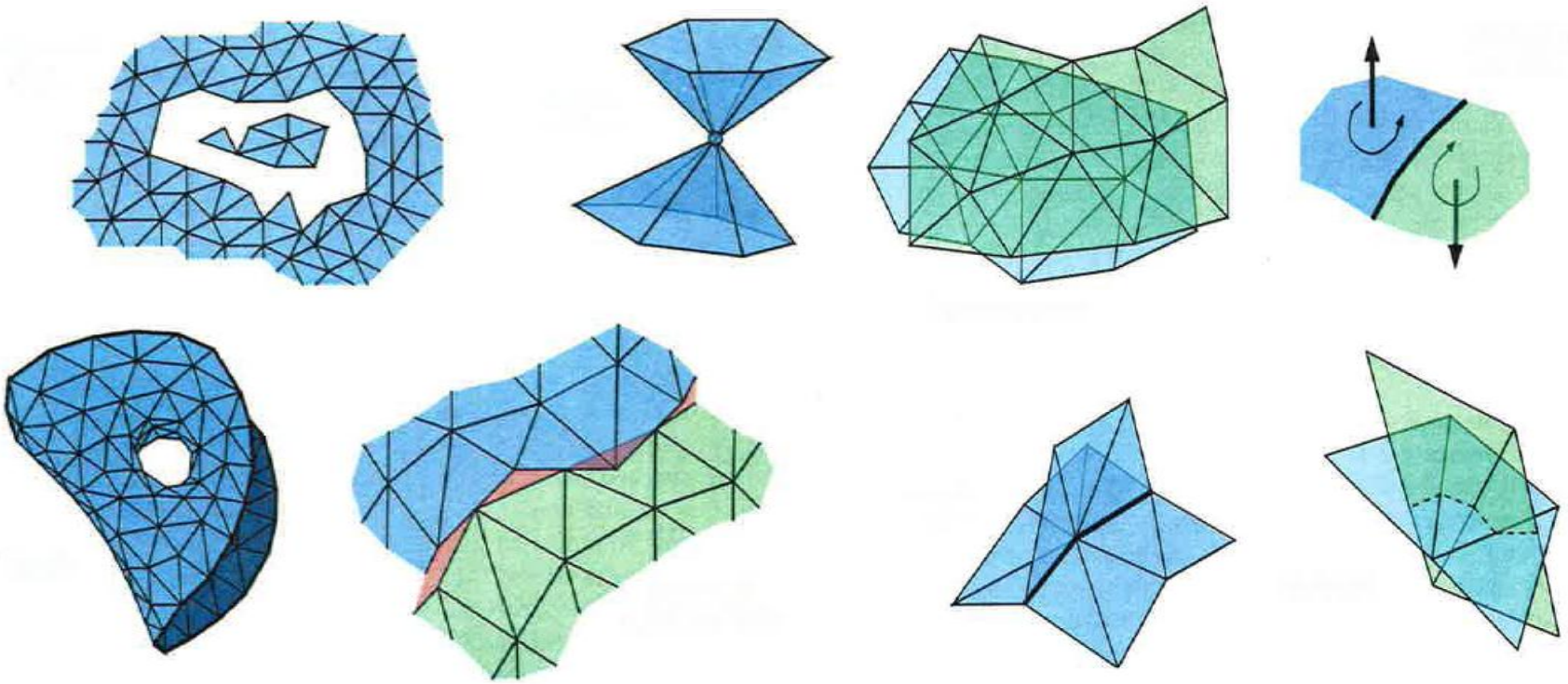
600.657: Mesh Processing

Chapter 8

Handling Mesh Degeneracies



[Botsch *et al*., Polygon Mesh Processing]

Class of Approaches

Geometric:

- Preserve the mesh where it's good.

Volumetric:

- Can guarantee no self-intersection.
- Can handle topology changes.

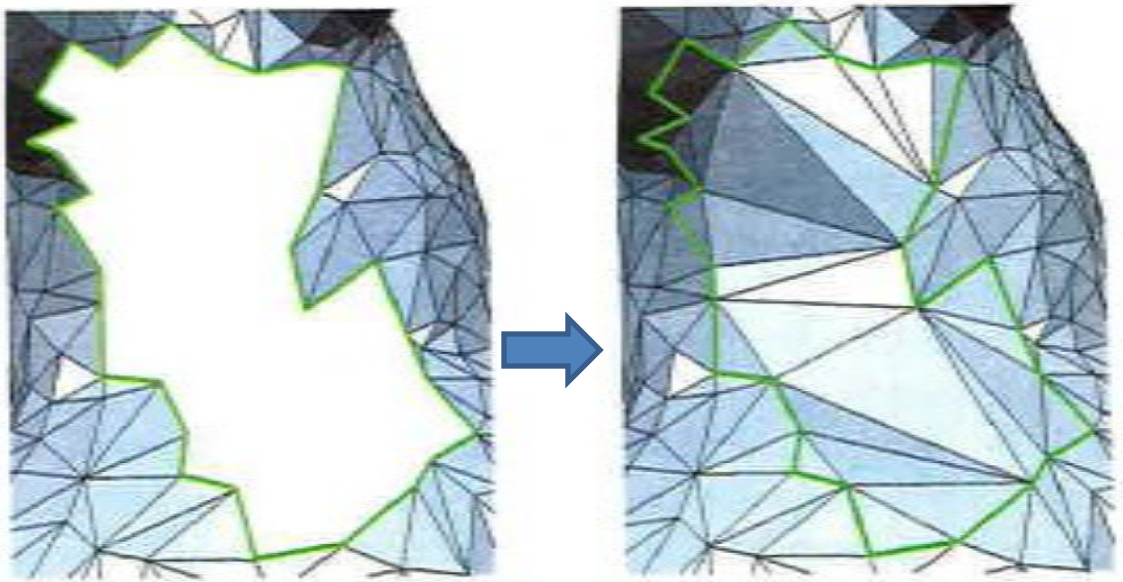
Outline

- Closing Holes [Barequet *et al*., 1998]
- Removing Loops [Guskov and Wood, 2001]
- Sealing Polygon Soups [Murali and Funkhouser, 1997]

Closing Boundary Holes

Goal:

Given a polygon on in 3D, find a triangulation that minimizes area/dihedral-angle/etc.



[Botsch *et al*., Polygon Mesh Processing]

Closing Boundary Holes

Problem Statement:

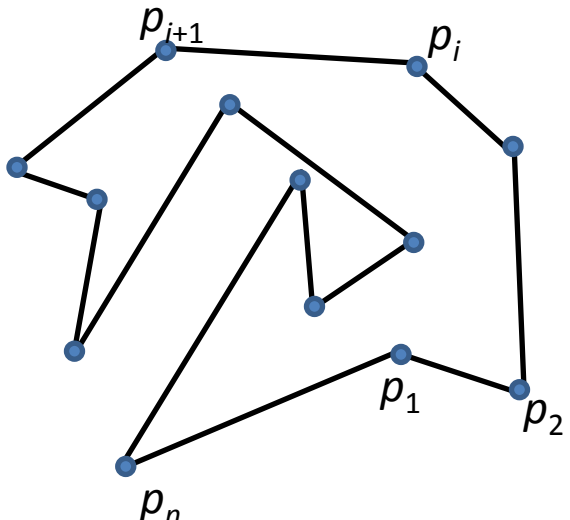
Given a set of points, $P=\{p_1,\ldots,p_n\}$, find the area/dihedral-angle/etc. minimizing set of triangles whose vertices are in P and satisfy:

- The edge $\{p_i,p_{i+1}\}$ appears in exactly one triangle.
- Any other edge is shared by exactly two triangles.

Closing Boundary Holes

Observation:

For any edge $\{p_i,p_{i+1}\}$, there hast to be one triangle in the triangulation containing $\{p_i,p_{i+1}\}$.

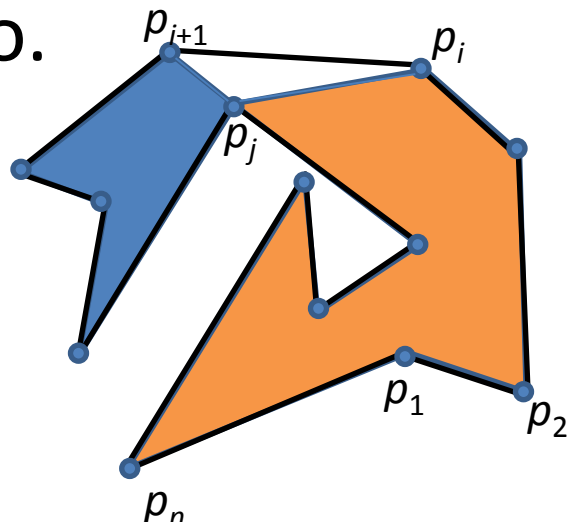


Closing Boundary Holes

Observation:

For any edge $\{p_i,p_{i+1}\}$, there hast to be one triangle in the triangulation containing $\{p_i,p_{i+1}\}$.

Break up into smaller problems by trying all possible triangles. Each choice of p_j breaks the problem into two.

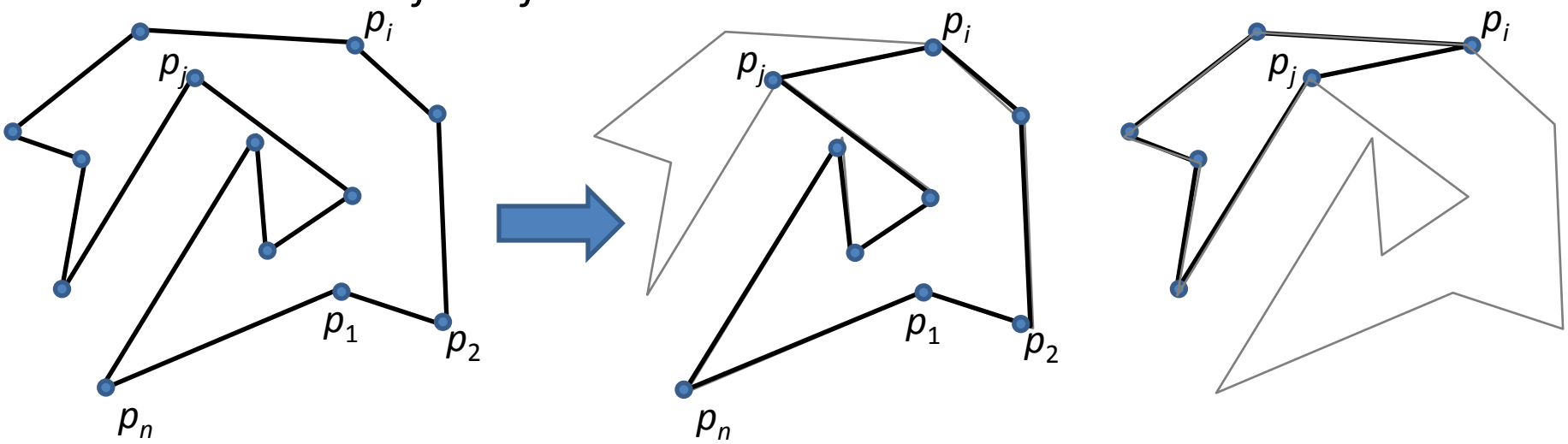


Closing Boundary Holes

Sub-Problem (Area Minimization):

Given $i<j$, find the minimal area triangulation of the points $P_{ij}=\{p_1,\ldots,p_i,p_j,\ldots,p_n\}$.

Given $j<i$, find the minimal area triangulation of the points $P_{ij}=\{p_j, \ldots,p_i\}$.

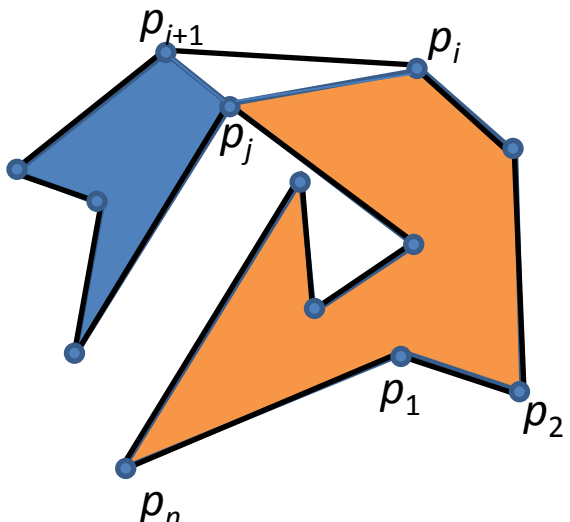


Closing Boundary Holes

Solution:

If we can solve the sub-problem, we can solve the complete problem:

$$\text{Area}(P) = \min_j \left(\text{Area}(p_i, p_{i+1}, p_j) + \text{Area}(P_{i,j}) + \text{Area}(P_{j,i+1}) \right)$$



Closing Boundary Holes

Solution:

If we can solve the sub-problem, we can solve the complete problem:

$$\text{Area}(P) = \min_j \left(\text{Area}(p_i, p_{i+1}, p_j) + \text{Area}(P_{i,j}) + \text{Area}(P_{j,i+1}) \right)$$

- Construct a 2D table $T(i,j)$=Area(P_{ij}).
- Minimal area is $T(i,i+1)$

Closing Boundary Holes

Complexity:

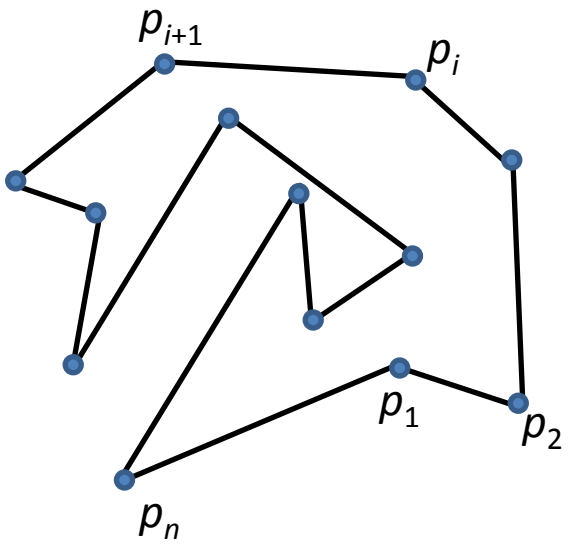
- Space: $O(n^2)$
- Time: ?

Closing Boundary Holes

Complexity:

- Space: O(n^2)
- Time:

  We set $T(i+1,i)=0$.



Closing Boundary Holes

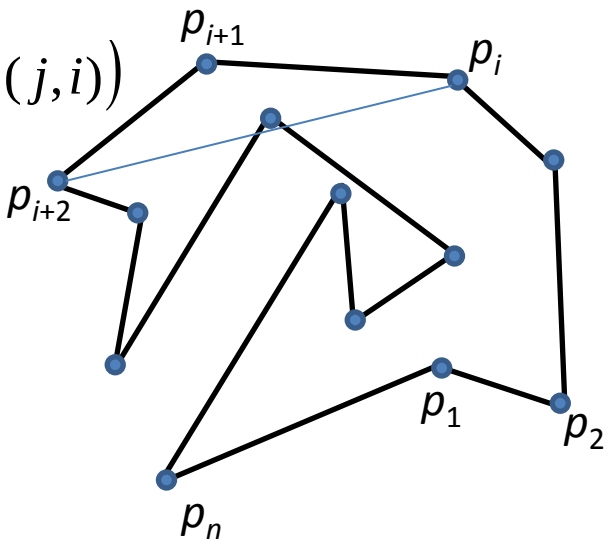
Complexity:

- Space: O(n^2)
- Time:

  We set $T(i+1,i)=0$.

  Next, we set $T(i+2,i)$:

$$T(i+2,i) = \min_{i+1 \le j \le i+1} \left(\text{Area}(p_i, p_{i+2}, p_j) + T(i+2, j) + T(j, i) \right)$$



Closing Boundary Holes

Complexity:

- Space: O(n^2)
- Time:

  We set $T(i+1,i)=0$.

  Next, we set $T(i+2,i)$.

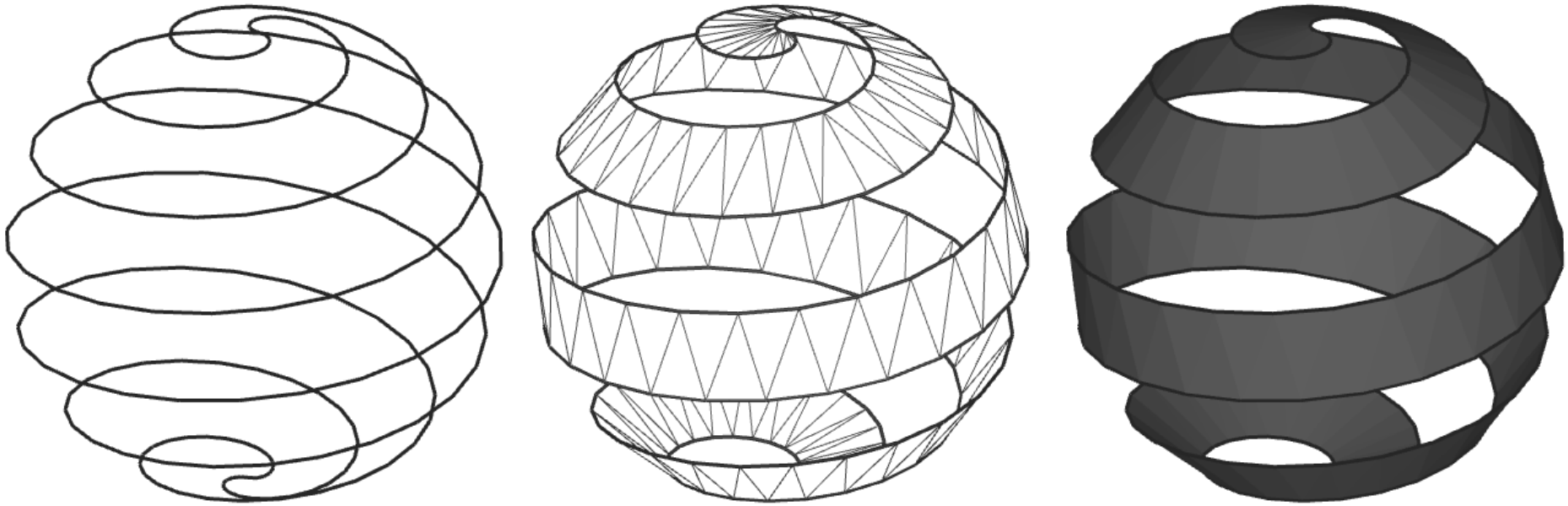
  And then $T(i+3,i)$:

$$T(i+3,i) = \min_{i+1 \le j \le i+2} \left(\text{Area}(p_i, p_{i+3}, p_j) + T(i+3, j) + T(j, i) \right)$$

Closing Boundary Holes

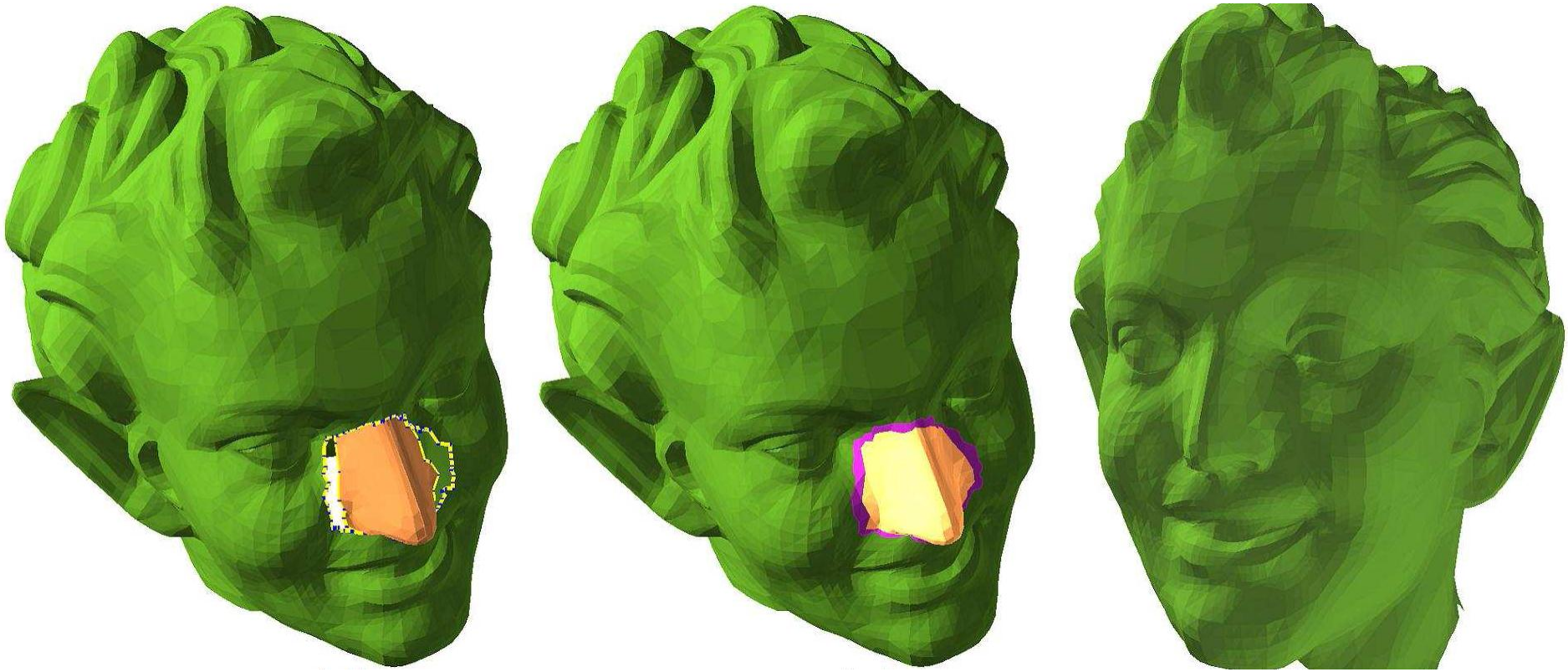
Complexity:

- Space: $O(n^2)$
- Time: $O(n^3)$



Closing Boundary Holes

Q: Can we use this to merge two loops?



[Funkhouser *et al.*, 2004]

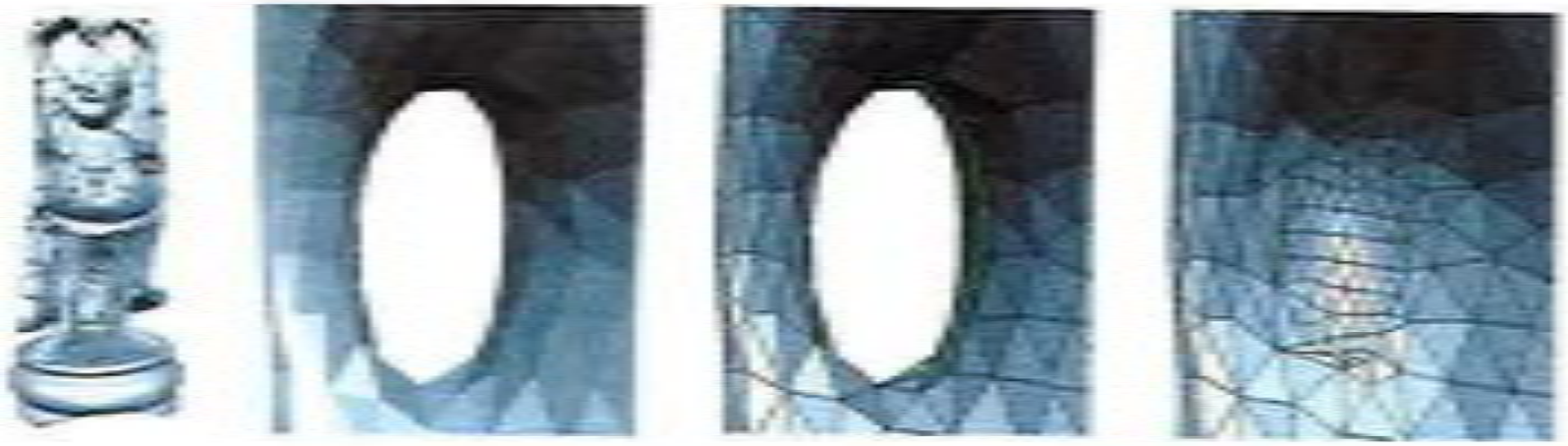
Outline

- Closing Holes [Barequet *et al*., 1998]
- Removing Loops [Guskov and Wood, 2001]
- Sealing Polygon Soups [Murali and Funkhouser, 1997]

Removing Loops

Goal:

Given a water-tight mesh in 3D, remove the small topological handles.



[Botsch *et al*., Polygon Mesh Processing]

Removing Loops

Approach:

- Identify small topological handles.
- Cut them open to get a boundary.
- Close the boundary with two surface patches.

Removing Loops

Approach:

- Identify small topological handles.
- Cut them open to get a boundary.
- Close the boundary with two surface patches.

Q1: How do we identify the small handles?

Q2: How do we cut them open?

Removing Loops

How do identify small handles?

For each point *p* on the mesh *M*, grow a small geodesic region around the point:

$$R_\varepsilon(p) = \{q \in M \mid d(p,q) < \varepsilon\}$$

and evaluate the topology of the region.

Removing Loops

How do identify small handles?

- How do we evaluate the topology?
- How do we grow a small region?

Removing Loops

How do we evaluate the topology?

Use the Euler characteristic:

$$|V|-|E|+|F|+|B|=2-2g$$

where:

- V is the set of vertices
- E is the set of edges
- F is the set of faces
- B is the set of boundary loops
- g is the genus

Removing Loops

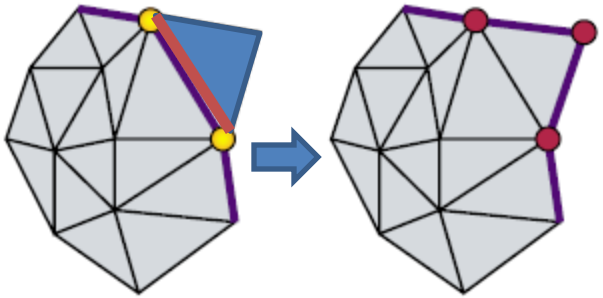
How do we grow a small region?

Start with a seed triangle and use Dijkstra's Algorithm (or fast marching) to add the next triangle across the boundary of the region.

Removing Loops

How do we grow a small region?

In general, when growing by one triangle, we add one face, two edges, and one vertex:

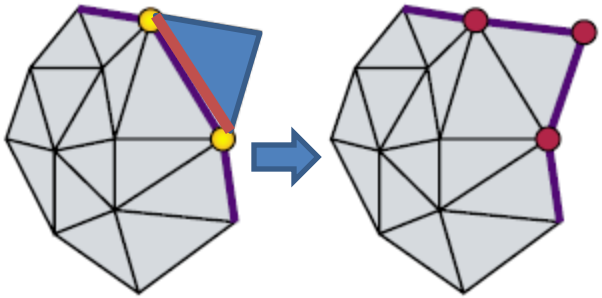


[Guskov and Wood, 2001]

Removing Loops

How do we grow a small region?

In general, when growing by one triangle, we add one face, two edges, and one vertex:



[Guskov and Wood, 2001]

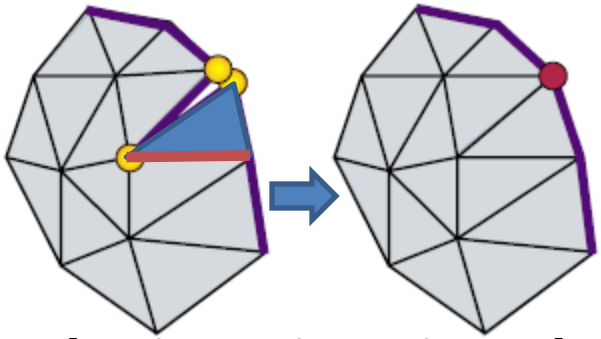
In this case the new Euler characteristic is:

$$\begin{aligned} 2-2g_{new} &= (|V_{old}|+1)-(|E_{old}|+2)+(|F_{old}|+1)+|B_{old}| \\ &= |V_{old}|-|E_{old}|+|F_{old}|+|B_{old}| \\ &= 2-2g_{old} \end{aligned}$$

Removing Loops

How do we grow a small region?

If one of the new edges already exists, we only add a new face and a new edge:

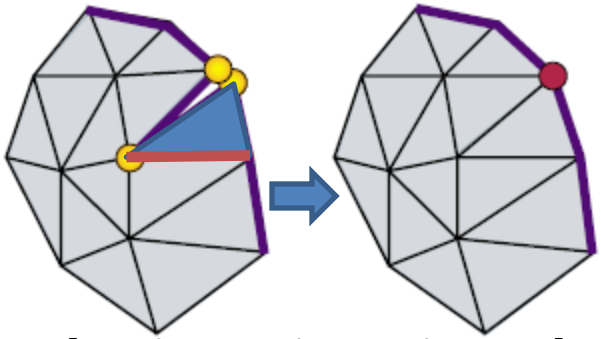


[Guskov and Wood, 2001]

Removing Loops

How do we grow a small region?

If one of the new edges already exists, we only add a new face and a new edge:



[Guskov and Wood, 2001]

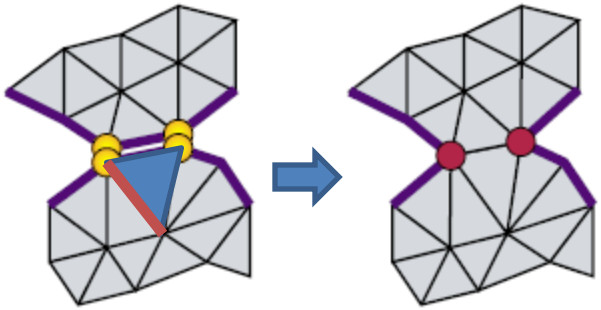
In this case the new Euler characteristic is:

$$\begin{aligned} 2-2g_{new} &= |V_{old}| - (|E_{old}|+1) + (|F_{old}|+1) + |B_{old}| \\ &= |V_{old}| - |E_{old}| + |F_{old}| + |B_{old}| \\ &= 2-2g_{old} \end{aligned}$$

Removing Loops

How do we grow a small region?

If both edges already exist, we add a new face and remove two duplicate vertices:

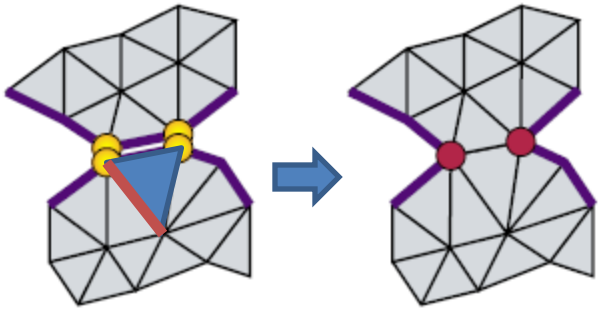


[Guskov and Wood, 2001]

Removing Loops

How do we grow a small region?

If both edges already exist, we add a new face and remove two duplicate vertices:



[Guskov and Wood, 2001]

In this case we are either splitting one boundary or merging two different boundaries.

Removing Loops

How do we grow a small region?

If we are splitting one boundary, the new Euler characteristic is:

$$\begin{aligned} 2-2g_{new} &= (|V_{old}|-2)-|E_{old}|+(|F_{old}|+1)+(|B_{old}|+1) \\ &= |V_{old}|-|E_{old}|+|F_{old}|+|B_{old}| \\ &= 2-2g_{old} \end{aligned}$$

Removing Loops

How do we grow a small region?

If we are merging two boundaries, the new Euler characteristic is:

$$
\begin{aligned}
2-2g_{new} &= \left(|V_{old}|-2\right)-|E_{old}|+\left(|F_{old}|+1\right)+\left(|B_{old}|-1\right) \\
&= |V_{old}|-|E_{old}|+|F_{old}|+|B_{old}|-2 \\
&= 2-2\left(g_{old}+1\right)
\end{aligned}
$$

Removing Loops

How do we grow a small region?

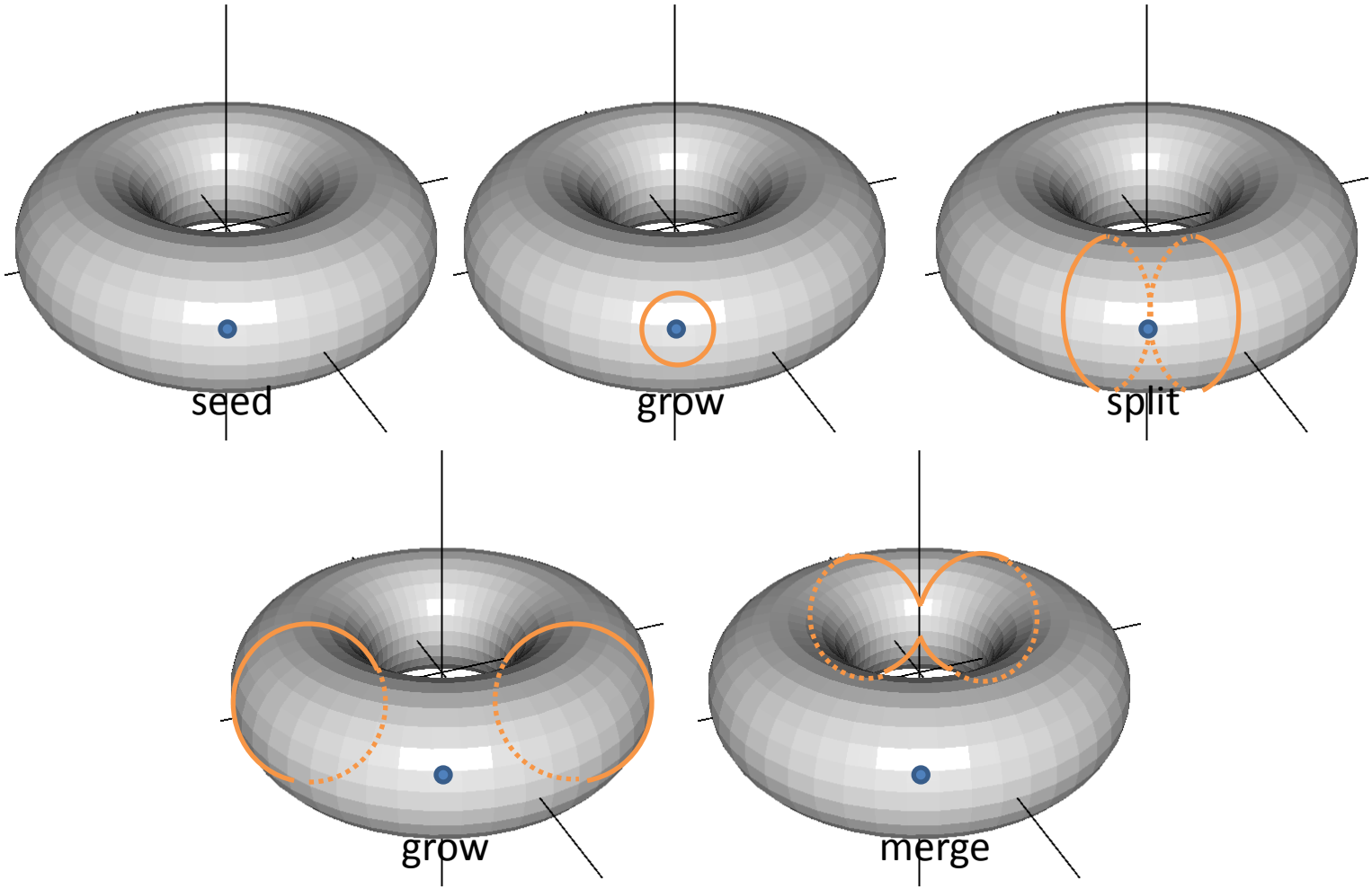
If we are merging two boundaries, the new Euler characteristic is:

$$\begin{aligned} 2-2g_{new} &= (|V_{old}|-2)-|E_{old}|+(|F_{old}|+1)+(|B_{old}|-1) \\ &= |V_{old}|-|E_{old}|+|F_{old}|+|B_{old}|-2 \\ &= 2-2(g_{old}+1) \end{aligned}$$

This implies that the region now has a loop in it.

Removing Loops

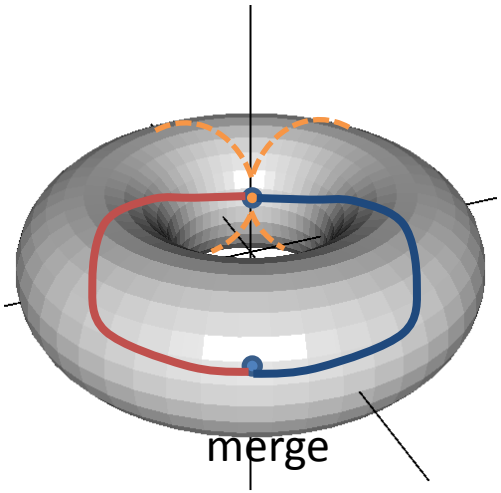
Region Growing:



Removing Loops

How do we cut open the handles?

Since we are expanding the region geodesically, when two different regions merge, we have two different paths back to the seed.

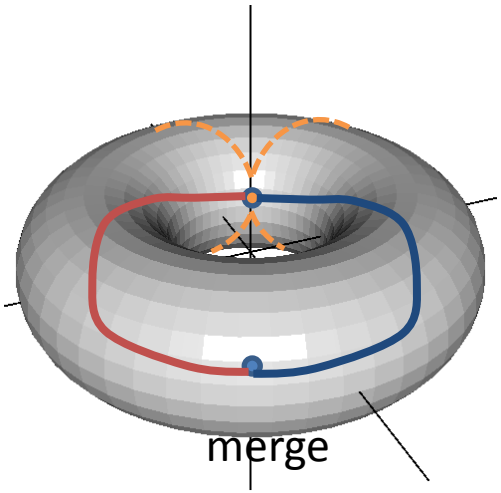


Removing Loops

How do we cut open the handles?

Since we are expanding the region geodesically, when two different regions merge, we have two different paths back to the seed.

Combining these two paths we get a closed loop around the handle.



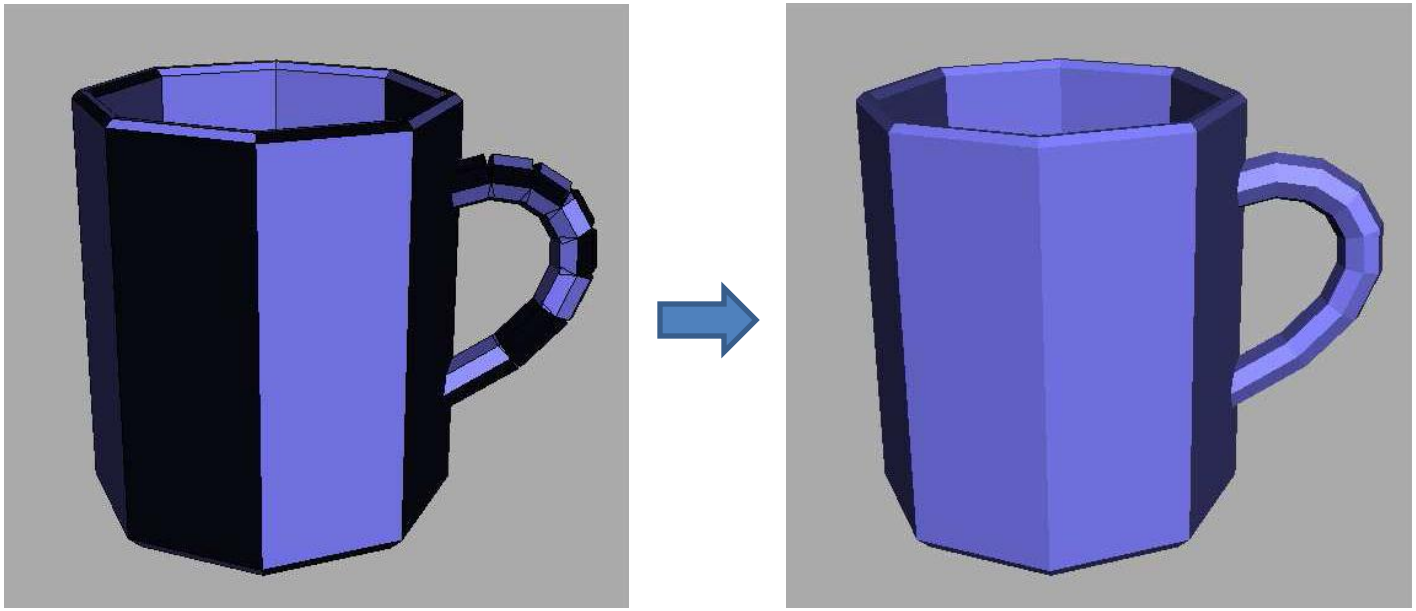
Outline

- Closing Holes [Barequet *et al*., 1998]
- Removing Loops [Guskov and Wood, 2001]
- Sealing Polygon Soups [Murali and Funkhouser, 1997]

Sealing Polygon Soups

Goal:

Given a polygon soup, fit a water-tight model to the soup.

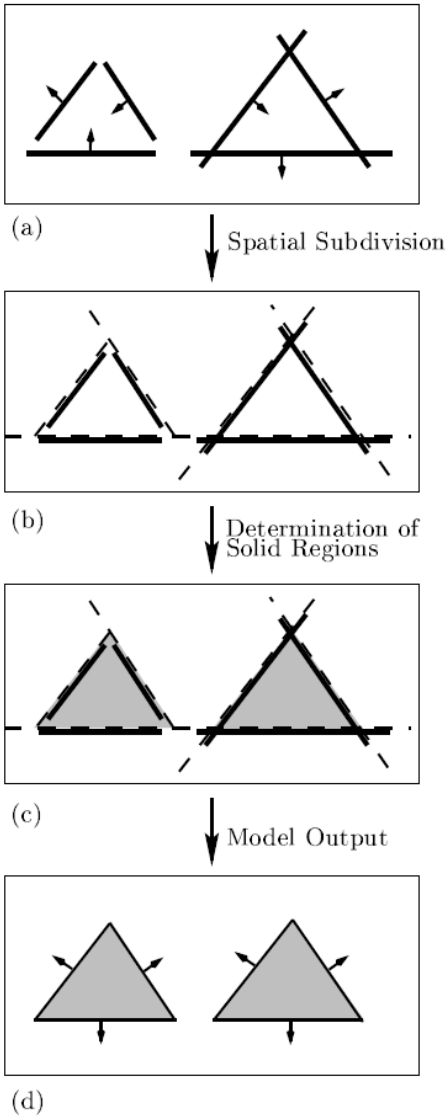


[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Approach:

- Use the polygon soup to define a partition of 3D space into disjoint cells.
- Label each cell as interior or exterior.
- Extract the boundary between interior and exterior cells.



[Murali and Funkhouser, 1997]

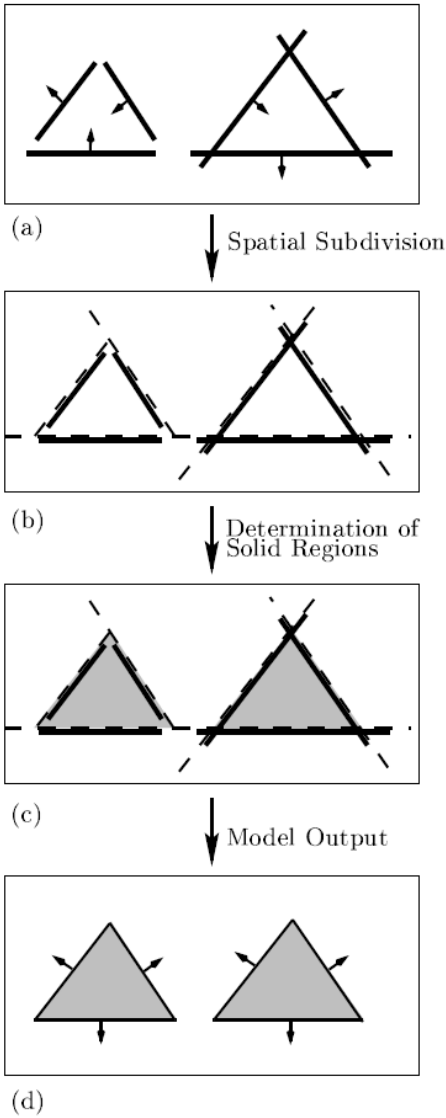
Sealing Polygon Soups

Approach:

- Use the polygon soup to define a partition of 3D space into disjoint cells.
- Label each cell as interior or exterior.
- Extract the boundary between interior and exterior cells.

Q1: How do we partition space?

Q2: How we assign labels?

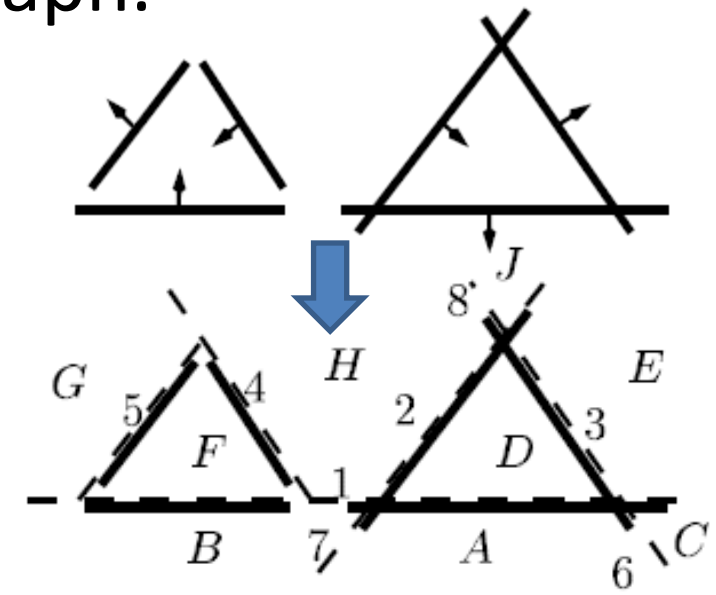


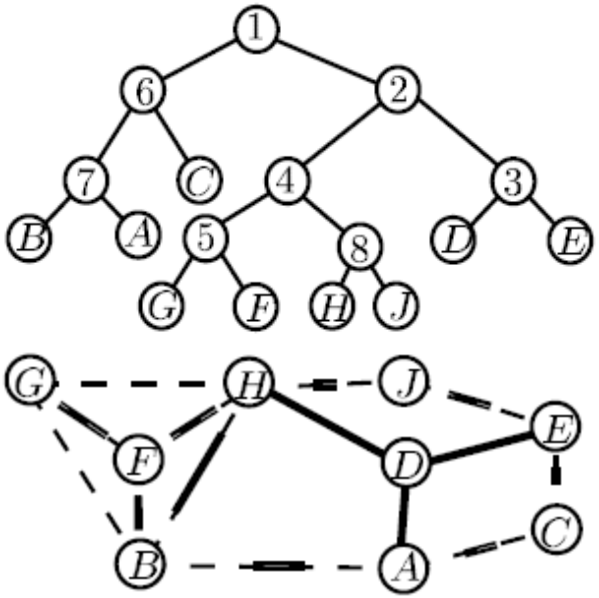
[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Partitioning Space:

Construct a BSP tree, using the polygons in the polygon soup to define the splitting planes (prioritized by area) and track the adjacency graph.





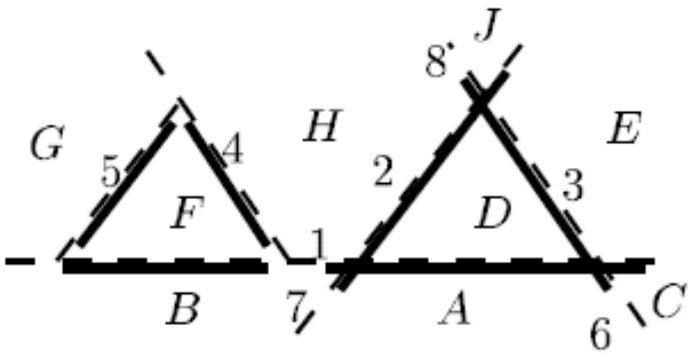
[Murali and Funkhouser, 1997]

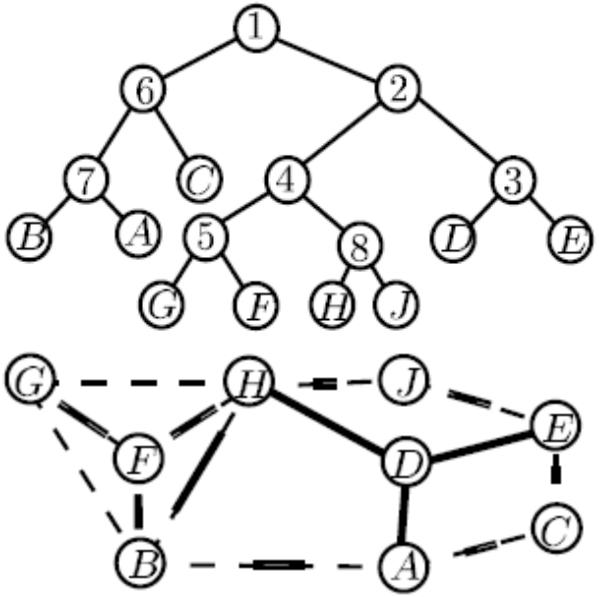
Sealing Polygon Soups

Partitioning Space:

Leaves of the BSP tree correspond to the (convex) cells of the partition.

Links in the adjacency graph represents shared (convex) boundaries



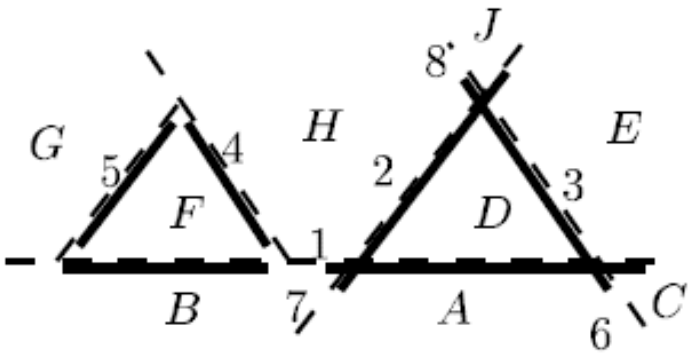


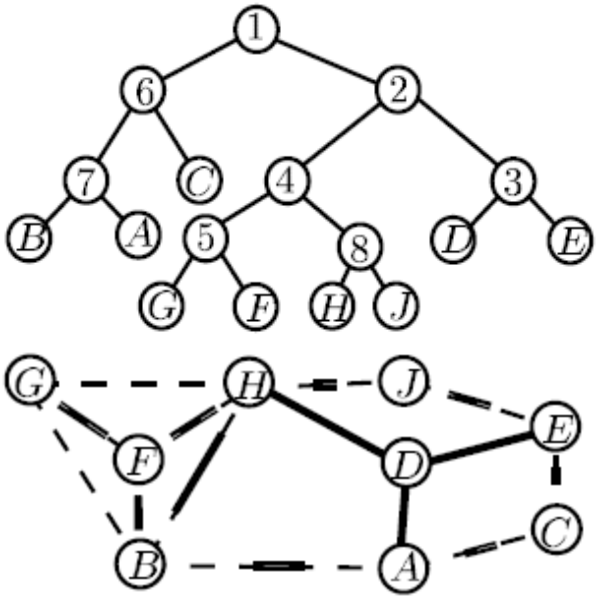
[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Partitioning Space:

To each (bounded) link we can associate the measure of opacity/transparency.



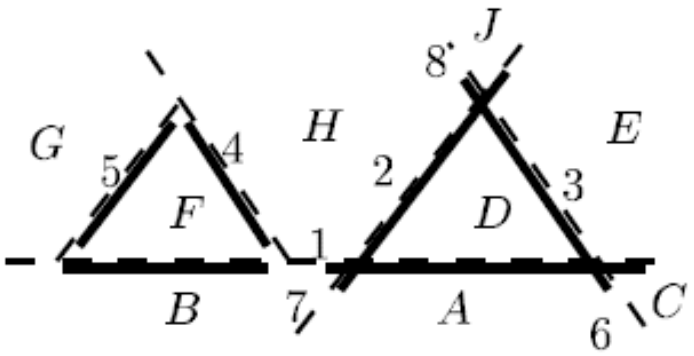


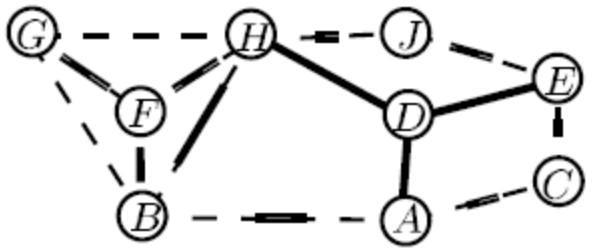
[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Assigning Labels:

- Any BSP leaf node with an unbounded face (link) must be exterior to the surface.



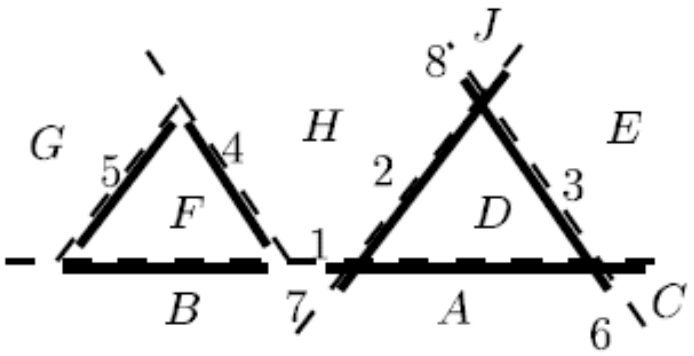


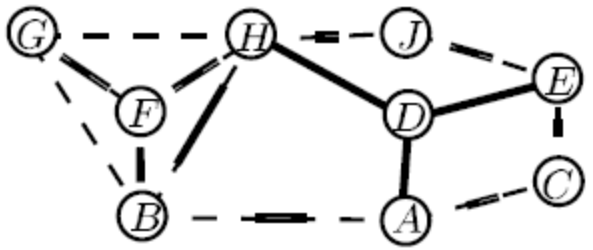
[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Assigning Labels:

- Any BSP leaf node with an unbounded face (link) must be exterior to the surface.
- If two cells share a boundary that is transparent, they are likely to be assigned the same labels.
- If two cells share a boundary that is opaque, they are likely to be assigned different labels.



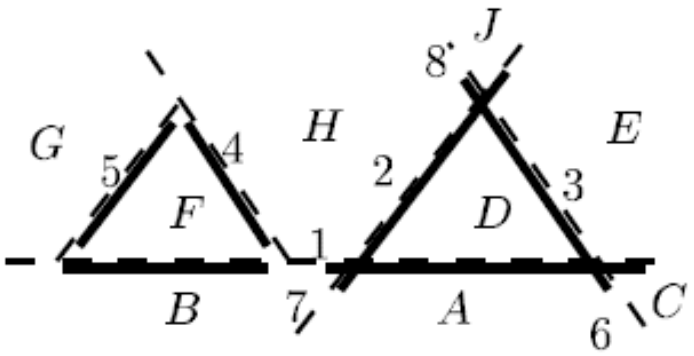


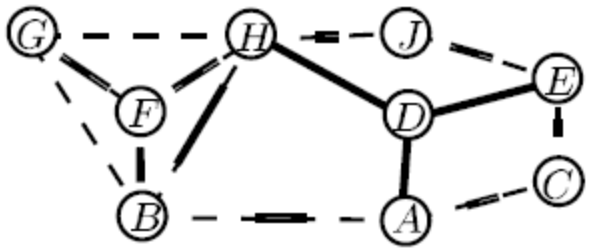
[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Goal:

Solve for soft assignments which are consistent with opacity.





[Murali and Funkhouser, 1997]

Sealing Polygon Soups

Linear System:

That is, s_i should be the area-weighted combination of its neighbors, with positive weights if the boundary is mostly transparent and negative weights if its mostly opaque.

Sealing Polygon Soups

Linear System:

That is, s_i should be the area-weighted combination of its neighbors, with positive weights if the boundary is mostly transparent and negative weights if its mostly opaque.

Formally, if $s_i \in [-1,1]$ is the assignment of leaf node i, and A_i is its surface area, then it should satisfy the property:

$$s_i = \frac{1}{A_i} \sum_j \left(t_{ij} - o_{ij} \right) s_j$$

Sealing Polygon Soups

Linear System:

This is a sparse linear system, which we can solve to get the assignments.

Sealing Polygon Soups

Reconstruction:

Label all leaf nodes with assignment $s_i > 0$ as interior and $s_i \leq 0$ as exterior.

Sealing Polygon Soups

Reconstruction:

Label all leaf nodes with assignment $s_i > 0$ as interior and $s_i \leq 0$ as exterior.

Define the water-tight surface to be the union of boundaries between interior and exterior cells.

Sealing Polygon Soups

Reconstruction:

Label all leaf nodes with assignment $s_i > 0$ as interior and $s_i \leq 0$ as exterior.

Define the water-tight surface to be the union of boundaries between interior and exterior cells.

This approach has a very similar feel to the approaches we have seen for harmonic maps, only the weights sum to a value in the range [-1,1].