Surface Remesher

Shayan Hoshyari

December, 2016

Surface Remeshing

Goal:

- Increase triangle quality
- Reduce/increase number of faces
- Increase mesh regularity
- Target based grading, e.g., curvature

Main Constraint:

- Stay close to the initial surface

Based On:

- Vitaly Surazhsky, and Craig Gotsman. "Explicit Surface Remeshing"

Surface Remeshing

Goal:

- Increase triangle quality
- Reduce/increase number of faces
- Increase mesh regularity
- Target based grading, e.g., curvature

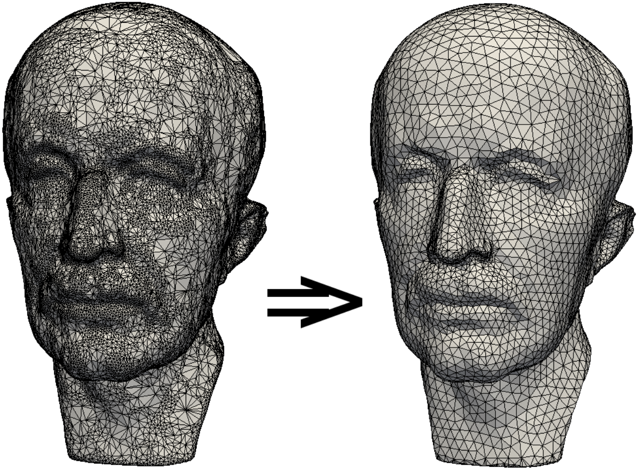
Main Constraint:

- Stay close to the initial surface

Based On:

- Vitaly Surazhsky, and Craig Gotsman. "Explicit Surface Remeshing"

Surface Remeshing



Max Planck model remeshed and coarsened.

Tools

Local operations in “geodesic polar mapped space”:

- Edge collapse
- Edge split
- Edge flip
- Area based vertex relocation
- Laplacian smoothing

Keeping mesh fidelity:

- Fidelity error metrics to prevent certain operations
- Overlapping patchwise parameterization

Fidelity Error Metrics

All created triangles T with vertices $\mathcal{V}(T)$ must satisfy:

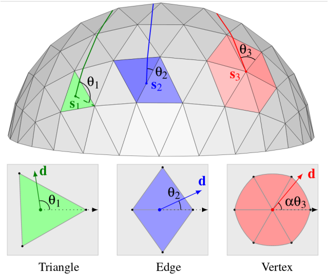
$$\min(\vec{N}_i \cdot \vec{N}_j) > \cos(\theta_1) \quad i, j \in \mathcal{V}(T)$$

$$\min(\vec{N}_i \cdot \vec{N}_T) > \cos(\theta_2) \quad i \in \mathcal{V}(T)$$

$$\theta_1 = \theta_2 = 20 \text{ deg}$$

Geodesic Polar Mapping

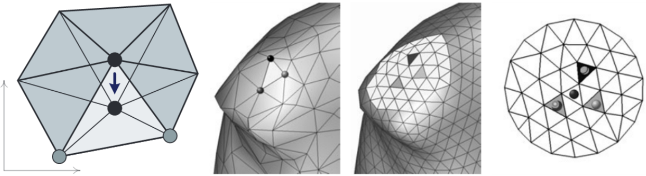
- Map the neighborhood of an edge or vertex to two dimensions
- Vertex:
  - Preserve distances
  - Scale angles to sum to 2π
- Edge:
  - Preserve angles
  - Preserve distances
  - Rotate around common edge



Overlapping Patchwise Parameterization

Output of local operations: $v = \text{Locate}(T, b), \quad T \in M$
Possible Locate() candidates:

- Using current mesh M
  - $v = \text{Interpolate}(T, b)$
- Projection on the initial mesh M_0:
  - $((T_1, b_1), (T_2, b_2), (T_3, b_3)) = \text{Reference}(T)$
  - $\hat{v}_i = \text{Interpolate}(\hat{T}_i, b_i) \quad i = 1 \ldots 3$
  - $\hat{T} = \text{Triangle}(\hat{v}_1, \hat{v}_2, \hat{v}_3)$
  - $\hat{v} = \text{Interpolate}(\hat{T}, b)$
  - Find $\hat{T}_r$ where $\hat{v} = \text{Interpolate}(\hat{T}_r, b_r)$
  - $v = \text{Interpolate}(T_r, b_r)$



Comparison of Projection Methods

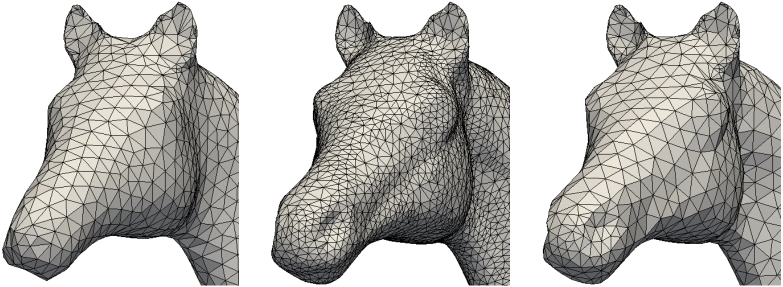
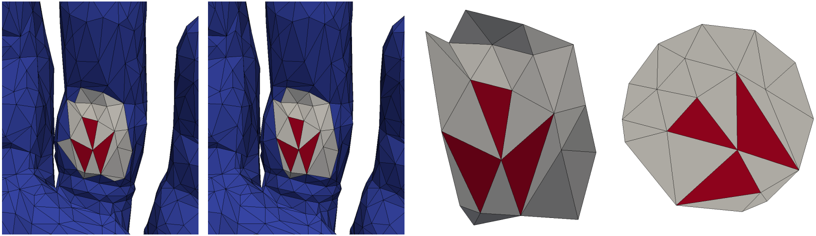


Figure: Comparison of projection techniques

Patch Creation

- Find the patch using BFS search
- Check topology
- Trim ears
- Map to unit disk (CGAL)



Area Based Vertex Relocation

Area Based Vertex Relocation: Minimize
$\sum(A_i(x,y) - \frac{1}{N}\sum A_i)^2 = 0$

Laplacian Smoothing: $v = \frac{1}{N}\sum v_i$

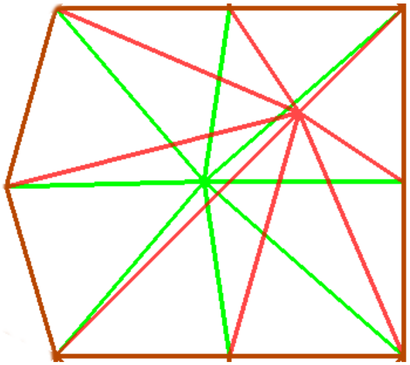


Figure: Area Based Vertex Relocation

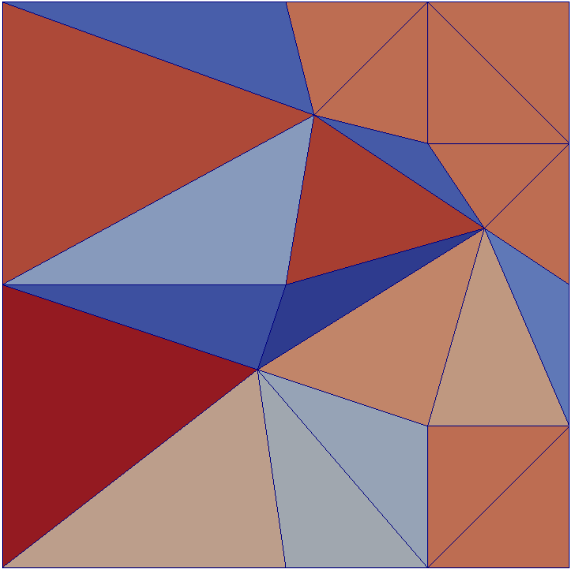
Global Algorithm

Algorithm 1 Gluing the primitive operations together

```
while Target # of vertices is reached do
    Sort the edges according to ascending/descending adjacent triangle quality.
    Split/collapse the edges in the mentioned order.
    Do not collapse or split edges that share an adjacent triangle.
    Perform 3 rounds of area based vertex relocation.
    Perform Dalauany edge flips.
end while
Optionally split all edges facing obtuse angles (not recursively).
Do the following 10 times: 3 rounds of area based vertex relocation followed by Delaunay edge flips.
Perform 10 rounds of Laplacian smoothing.
```

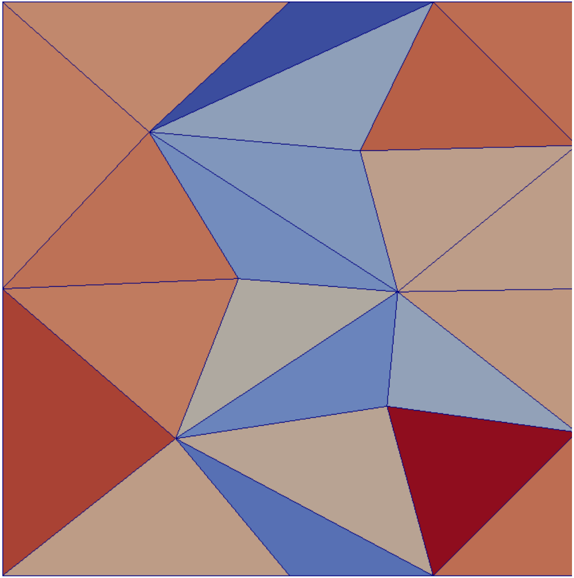
Sample Input

Input



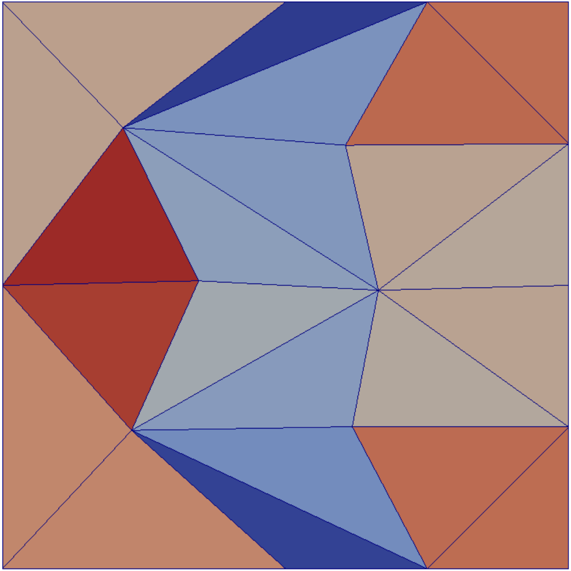
Sample Input

Area based vertex relocation - iteration 1



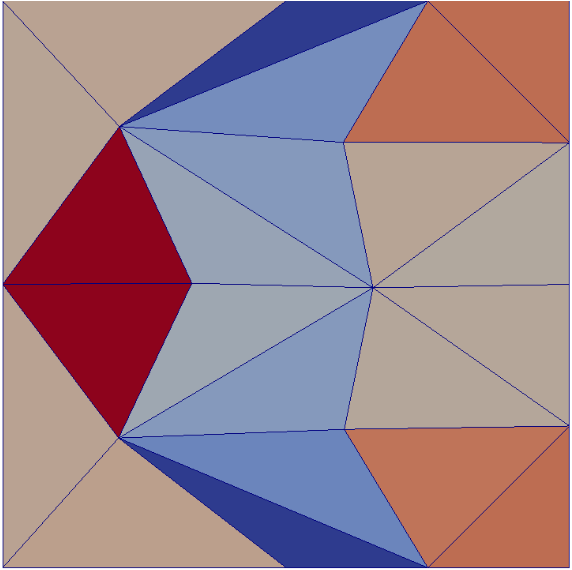
Sample Input

Area based vertex relocation - iteration 2



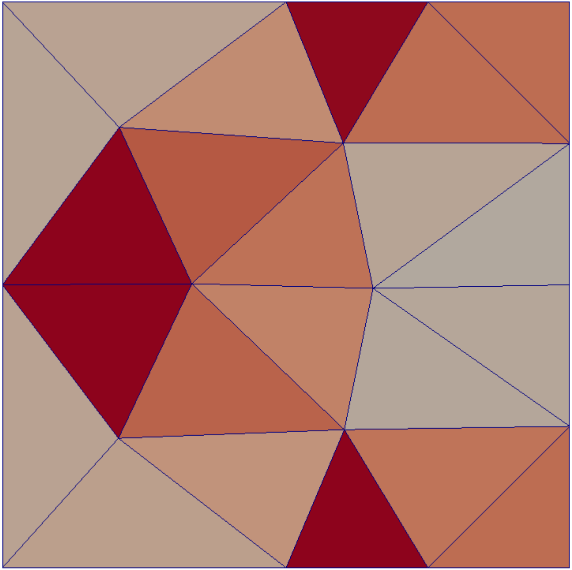
Sample Input

Area based vertex relocation - iteration 3



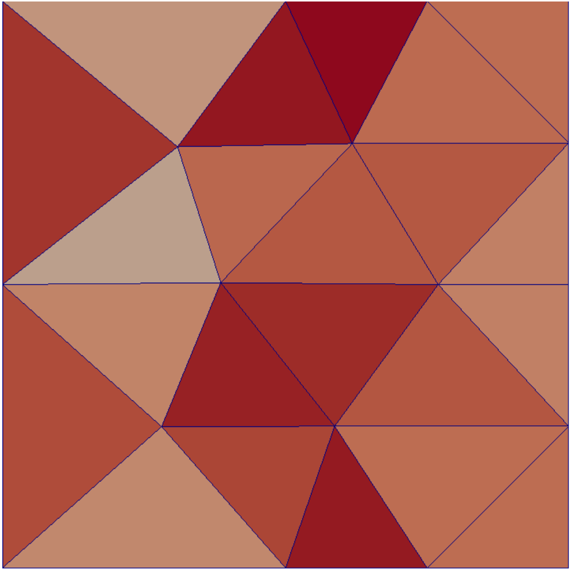
Sample Input

Delaunay flips



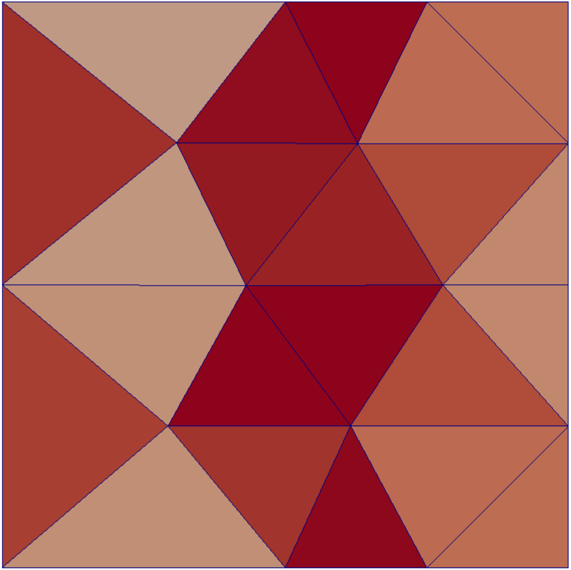
Sample Input

Laplacian smoothing - iteration 1



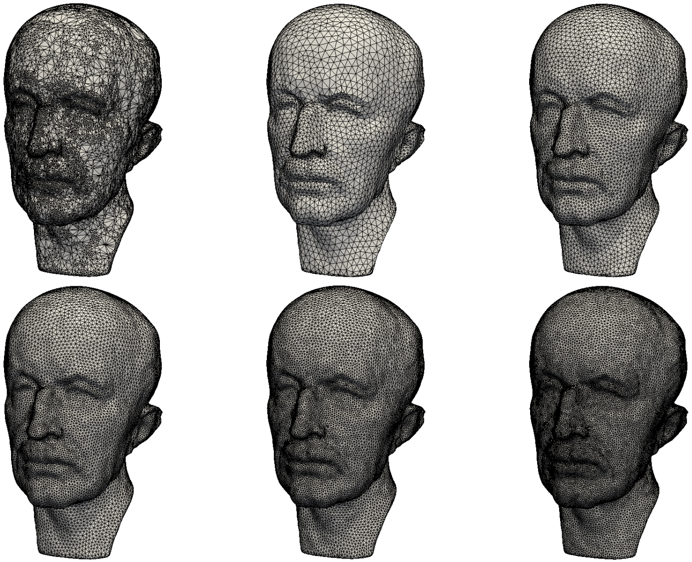
Sample Input

Laplacian smoothing - iteration 2

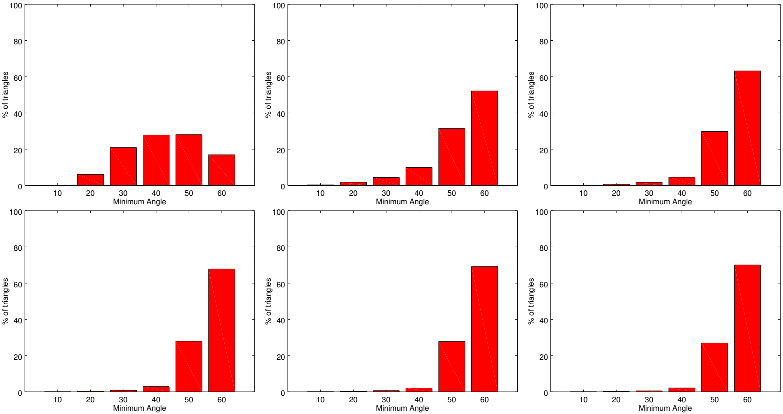


Results

$N_v = 19132, 5000, 10000, 150000, 20000, 30000$



Results



Results

Name	Vertex #	Run Time	Patch #
Initial	19132	—	—
Case a	5000	7.94 sec	4568
Case b	10000	9.84 sec	4210
Case c	15000	11.14 sec	3984
Case d	20000	13.89 sec	3542
Case e	30000	22.32 sec	3520

Possible Improvements

- Using Bezier patches (PN triangles) to represent initial surface
- Adaptively subdividing areas with high initial fidelity error
- Goal based insertion and collapsing, e.g. regularization
- Curvature based grading