Direct Triangle Meshes Remeshing using Stellar Operators

Aldo Zang, Fabian Prada

VISGRAF - IMPA

Introduction

Remeshing: Mesh quality improvement

- Sampling density.
- Regularity.
- Size.
- Orientation.
- Alignment.
- Shape.

Remeshing Algorithms

- Variational Remeshing.
- **Incremental Remeshing.**

Introduction

Problem Statement

Propouse a remeshing strategy based on stellar operators to obtain a mesh that satisfactorially meets the following criterias:

- Uniformity: Equilateral Aspect Triangles
- Regularity: Valence 6 vertices at interior and valence 4 at boundary

Constraints

- Preserve Geometry and Features.
- Maintain Resolution.

Papers used for our approach

- *A remeshing approach to multiresolution modeling*. Mario Botsch, Leif Kobbelt.
- *Multiresolution shape deformations for meshes with dynamic vertex connectivity*. Leif Kobbelt, Thilo Bareuther, Hans-Peter Seidel.
- *Stellar mesh simplification using probabilistic optimization*. Antônio Wilson Vieira et al.
- *Difusion tensor weighted harmonic fields for feature classification*. Shengfa Wang et al.
- *Hierarchical feature subspace for structure-preserving deformation*. Submited to GMP 2012

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l
2. Split all edges that are longer than $\frac{4}{3}l$ at their midpoint

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l
2. Split all edges that are longer than $\frac{4}{3}l$ at their midpoint
3. Collapse all edges shorter than $\frac{4}{5}l$ into their midpoint (or collapse over the vertex with higher valence)

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l
2. Split all edges that are longer than $\frac{4}{3}l$ at their midpoint
3. Collapse all edges shorter than $\frac{4}{5}l$ into their midpoint (or collapse over the vertex with higher valence)
4. Flip edges in order to minimize the deviation from valence 6 (or 4 on boundaries)

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l
2. Split all edges that are longer than $\frac{4}{3}l$ at their midpoint
3. Collapse all edges shorter than $\frac{4}{5}l$ into their midpoint (or collapse over the vertex with higher valence)
4. Flip edges in order to minimize the deviation from valence 6 (or 4 on boundaries)
5. Relocate vertices on the surface by tangential smoothing

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l
2. Split all edges that are longer than $\frac{4}{3}l$ at their midpoint
3. Collapse all edges shorter than $\frac{4}{5}l$ into their midpoint (or collapse over the vertex with higher valence)
4. Flip edges in order to minimize the deviation from valence 6 (or 4 on boundaries)
5. Relocate vertices on the surface by tangential smoothing
6. Repeat steps (2)-(5) until satisfy the stop criteria (good edges ratio)

Remeshing pipeline

Remeshing algorithm

1. Get a edge target length l
2. Split all edges that are longer than $\frac{4}{3}l$ at their midpoint
3. Collapse all edges shorter than $\frac{4}{5}l$ into their midpoint (or collapse over the vertex with higher valence)
4. Flip edges in order to minimize the deviation from valence 6 (or 4 on boundaries)
5. Relocate vertices on the surface by tangential smoothing
6. Repeat steps (2)-(5) until satisfy the stop criteria (good edges ratio)
7. Apply area based tangential smoothing to equalize triangles areas

Remeshing algorithm: step by step

1- Get the edge target length l

We compute some statistics for the mesh and set the target length as:

$$l = Mean(edges\ \ lenght) - \lambda Deviation(edges\ \ lenght)$$

with $\lambda \in [0, 1]$.

Remeshing algorithm: step by step

2- Split edges

All edges wich are longer than $\frac{4}{3}l$ are split by inserting a new vertex at its midpoint. The two adjacent triangles are bisected accordingly. The upper and lower bound on the edges lenght are only comptatible if $\epsilon_{max} > 2\epsilon_{min}$.

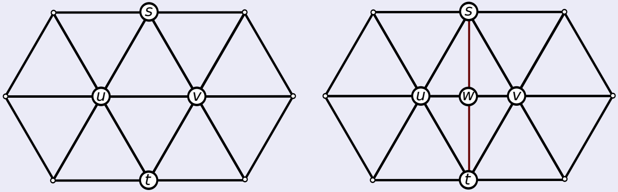


Figure: **Left:** original edge (u, v); **Right:** Split of (u, v) inserting vertex w.

3- Collapse edges

All edges wich are shorter than $\frac{4}{5}l$ are removed by collapsing the two-end vertices. We collpase that-end vertex with lower valence into the one with higher. This prevent accumulation of edges collapses.

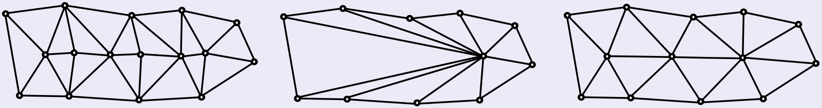


Figure: **Left:** original mesh; **Center:** Accumulation of edges collapses; **Right:** Collapses over the higher valence vertex.

Remeshing algorithm: step by step

4- Flip edges

We perform edge-flipping in order to regularize the connectivity. For every two neighboring triangles $\Delta(A, B, C)$ and $\Delta(C, B, D)$ we maximize the number of vertices with valence six by flipping the diagonal $\bar{BC}$ if the total valence excess is reduced.

$$V(e) = \sum_{p \in A,B,C,D} (valence(p) - 6)^2$$

Remeshing algorithm: step by step

4- Flip edges

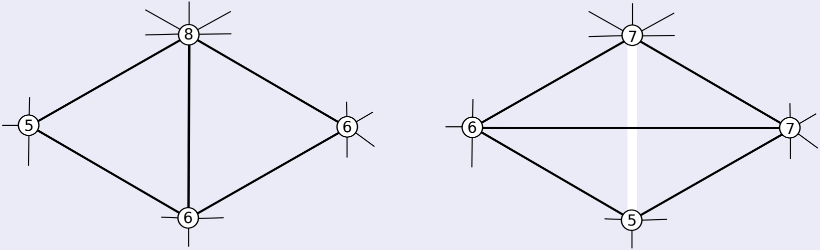


Figure: **Left:** Initial valence condition $V(e) = 2^2 + 1 = 5$; **Right:** Valence condition after flipping edge $V(e) = 1 + 1 + 1 = 3$.

Stellar operators theory

Edge-Flip

Edge-Flip: This operation consists in transforming a two-face cluster into another two-face cluster by swapping its common edge.

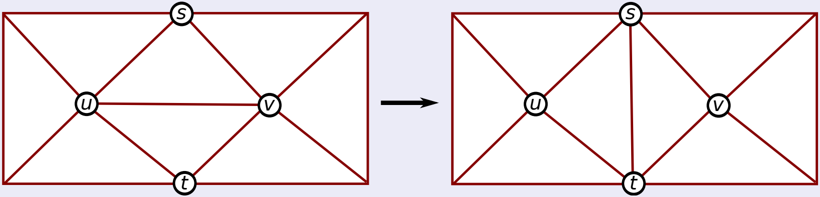


Figure: **Left:** input mesh; **Right:** result after flipping (u, v) to (s, t).

Stellar operators theory

Edge-Split operator

Edge-Split: This operation consists in transforming a two-face cluster into a four-face cluster by inserting a vertex in the interior edge of the cluster.

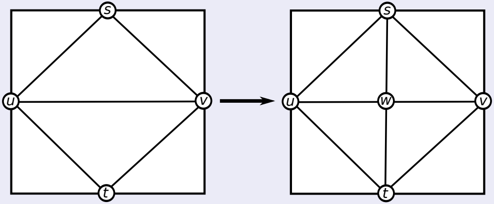


Figure: Split of the edge (u, v) by inserting a midpoint vertex w.

Edge-Flip operator

Lemma (Flip Condition): Let $\mathbf{S}$ a combinatorial 2-manifold. The *flip* of an interior edge that replaces $\mathbf{e} = (\mathbf{u}, \mathbf{v}) \in \mathbf{S}$ by $(\mathbf{s}, \mathbf{t})$ preserves the topology of $\mathbf{S}$ if an only if $(\mathbf{s}, \mathbf{t}) \notin \mathbf{S}$.

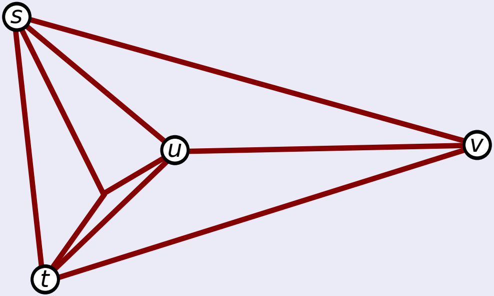


Figure: The edge (u, v) do not satisfies the flip condition because the new edge (s, t) already exists in the mesh.

Stellar operators theory

Edge-Collapse

Edge-Collapse: This operator consists in removing an edge $\mathbf{e} = (\mathbf{u}, \mathbf{v}) \in \mathbf{S}$, identifying its vertices to a unique vertex $\bar{\mathbf{v}}$. From a combinatorial viewpoint, this operator will remove 1 vertex, 3 edges, and 2 faces from the original mehs, thus preserving its Euler characteristic.

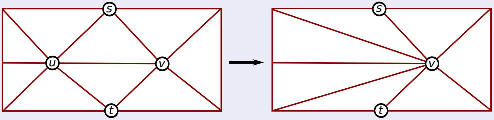


Figure: The edge (u, v) is collpased removing the vertex u from the mesh.

Stellar operators theory

Edge-Collapse

Lemma (Collapse Condition): Let S be a combinatorial 2-manifold . The collapse of an edge $\mathbf{e} = (\mathbf{u}, \mathbf{v}) \in \mathbf{S}$ preserves the topology of **S** if the followin conditions are satisfied:

- $link(\mathbf{u}) \cap link(\mathbf{v}) = link(\mathbf{e})$;
- if **u** and **v** are both boundary vertices, **e** is a boundary edge;
- **S** has more than 4 vertices if neither **u** nor **v** are boundary vertices, or **S** has more than 3 vertices if either **u** or **v** are boundary vertices.

Stellar operators theory

Edge-Collapse condition

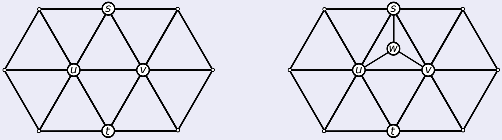


Figure: **Left:** (u, v) satisfies the edge-collpase condition; **Right:** (u, v) do not satisfies the edge-collpase condition because $s \in link(\mathbf{u}) \cap link(\mathbf{v})$ and $s \notin link((\mathbf{u,v}))$;

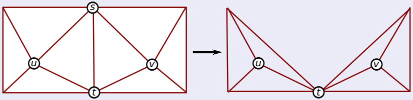


Figure: (s, t) do not satisfies the edge-collpase condition because (s, t) is interior edge but s and t are boundary vertices .

Edge-Weld operator

Edge-Weld: This operation consists in transforming a four-face cluster into a two-face cluster by removing its central vertex.
Corollary: Given a combinatorial 2-manifold $\mathbf{S}$, and a interior vertex $\mathbf{v} \in \mathbf{S}$ with valence 4. The removal of the vertex $\mathbf{v}$ by the Edge-Weld operation (along $(\mathbf{u}, \mathbf{w})$) preserves the topology of $\mathbf{S}$ if and only if there is no edge in $\mathbf{S}$ connecting $\mathbf{u}$ to $\mathbf{w}$.

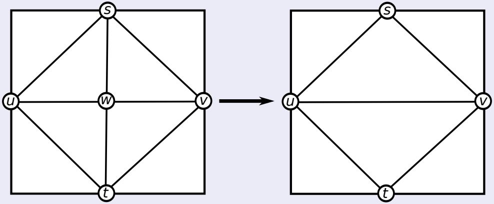


Figure: Edge-weld by removing midpoint vertex w.

Stellar operators theory

Edge-Collapse using basic stellar operators (edge-flip and edge-weld)

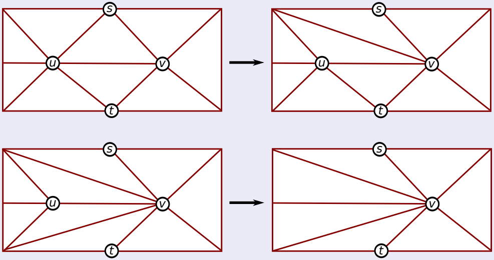


Figure: Edge (u, v) is collpased using 2 edges flips ((u, s) and (u, t)) and one edge-weld for edge (u, v).

Stellar operators of the A48 library

Basic operators

- **flip(** ***halfedge_type *h*** **);**
- **face_weld(** ***halfedge_type *h1, halfedge_type *h2, halfedge_type *h3*** **);**
- **edge_weld(** ***halfedge_type *h1, halfedge_type *h2);***
- **edge_split(halfedge_type *h);**
- **face_split(** ***face_type *f*** **);**

Stellar Operator of the A48 library

New operator and condition tests

- **void edge_collapse(*halfedge_type *h*);**
- **bool can_edge_collapse(*halfedge_type *h*);**
- **bool can_edge_flip(*halfedge_type *h*);**

Data Structure for operations handle

Halfedges Set data structure

```
class RmSetComp {
 public:
   bool operator() ( halfedge_type *hi, halfedge_type *hj ) const {

      if ( hi->l < hj->l )
         return true;
      else if ( hj->l < hi->l )
         return false;
      else return  hi < hj;
   }
};

typedef std::set<halfedge_type *, RmSetComp>  HalfedgeSet;
```

Data Structure for operations handle

Halfedges Set insert and remove functions

```
void remove_from_HSet( HalfedgeSet &RmSet, halfedge_type *he ) {
   if( he > he->opposite() )
      RmSet.erase( he );
   else
      RmSet.erase( he->opposite() );
}

void insert_to_HSet( HalfedgeSet &RmSet, halfedge_type *he ) {
   if( hcurr > hcurr->opposite() )
      RmSet.insert( hcurr );
   else
      RmSet.insert( hcurr->opposite() );
}
```

Vertex Reallocation

Step 1: Unconstrained Vertex Displacement

New position for each vertex is calculated as a weighted sum of its neighbours positions:

$$\hat{p}_i \leftarrow \frac{1}{\sum_{p_j \in N(p_i)} w_j} \Big(\sum_{p_j \in N(p_i)} w_j p_j \Big)$$

Neighbours Weights

- Uniform: $w_j = 1$.
- One Ring Area: $w_j = \sum_{f : p_j \in f} A(f)$.

Vertex Reallocation

Step 2: Reprojection

- Projection on the tangent plane:

$$p_i \leftarrow p_i + (I - n_i n_i^T)(\hat{p}_i - p_i).$$

- Projection on the lowest curvature direction:

$$p_i \leftarrow p_i + \gamma_{min}\gamma_{min}^T(\hat{p}_i - p_i).$$

Vertex Reallocation

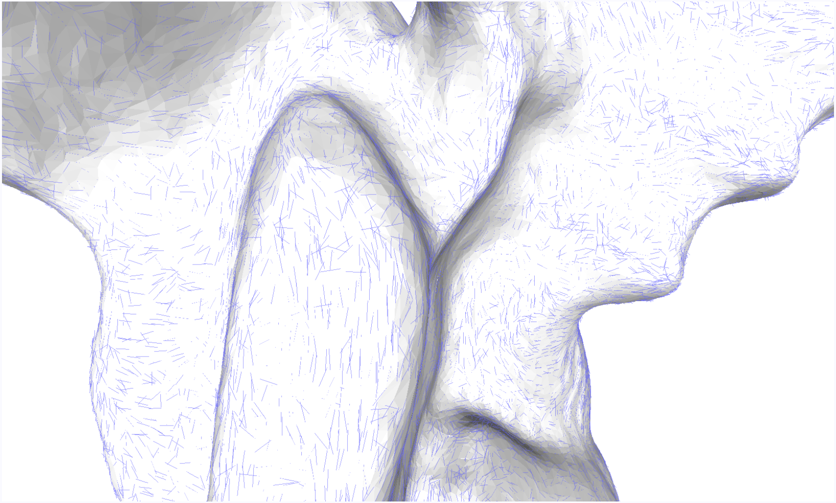
Lowest Curvature Direction

- Discrete Curvature Tensor Estimation:

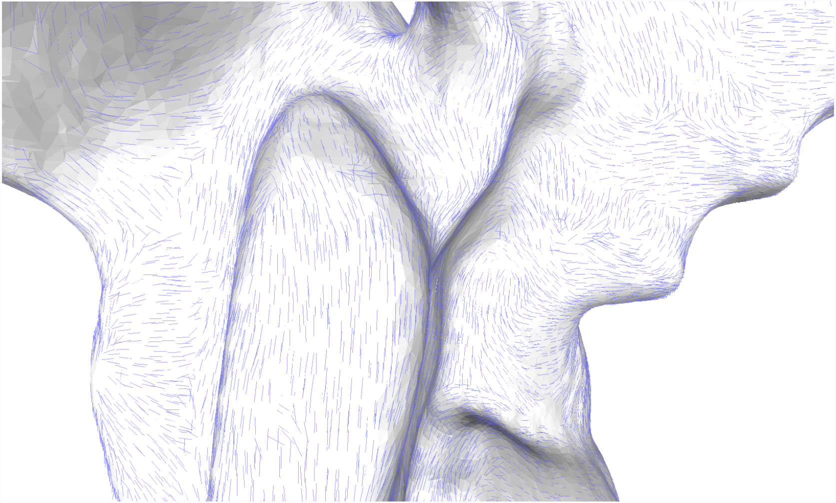
$$T(p) = \frac{1}{|B|} \sum_{e} \beta(e)|e \cap B| ee^T.$$

- Lowest curvature direction is the eigenvector associated to the largest eigenvalue of T(p).

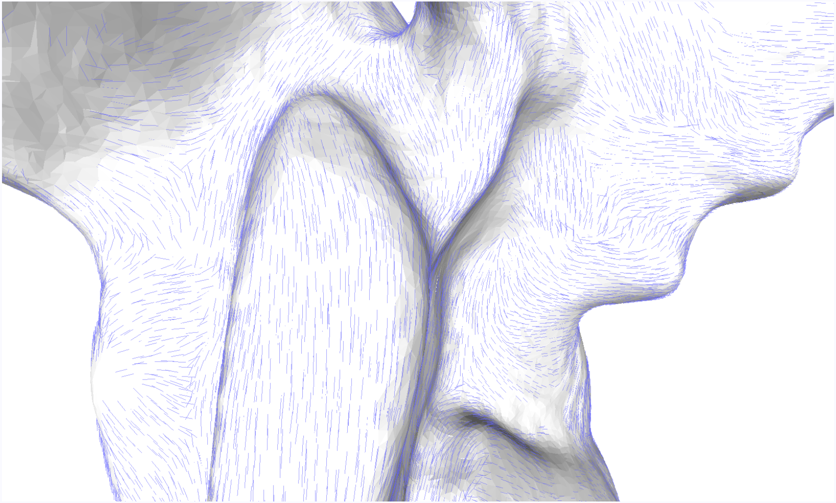
Min Curvatures Estimation: 1 Ring



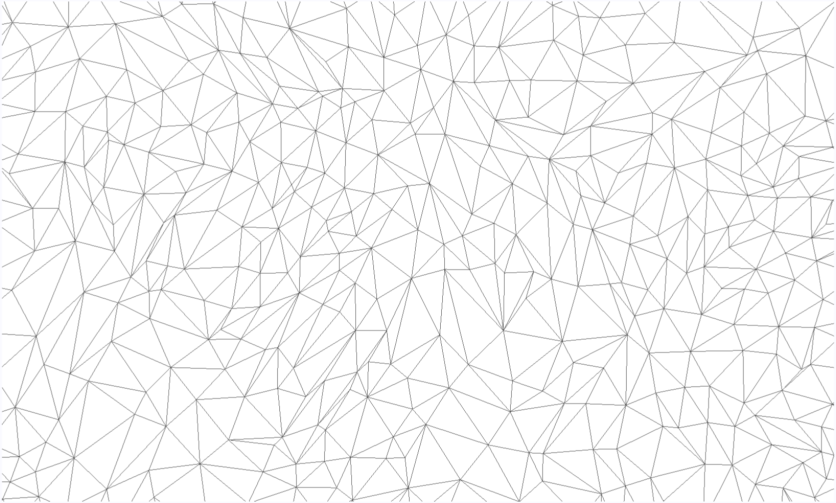
Min Curvatures Estimation: 3 Ring



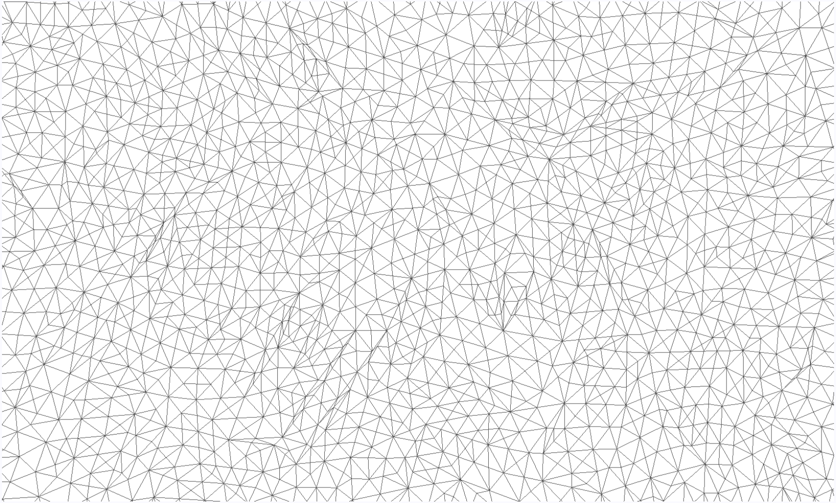
Min Curvatures Estimation: 5 Ring



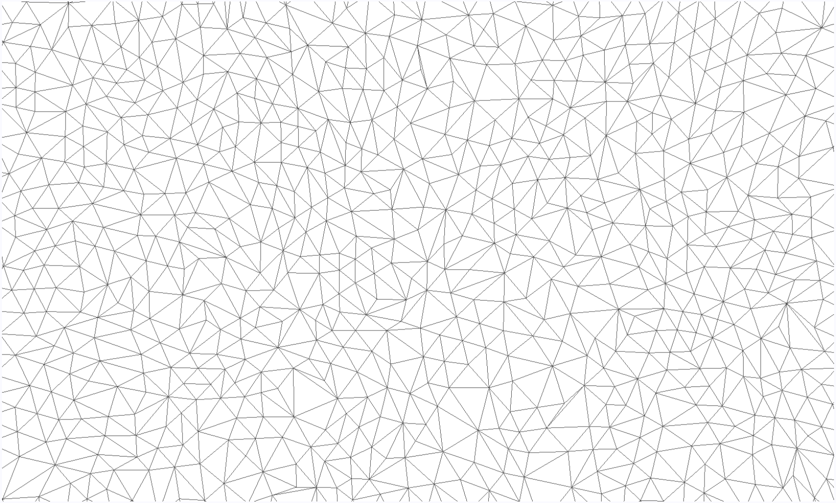
Remeshing Sequence : input mesh



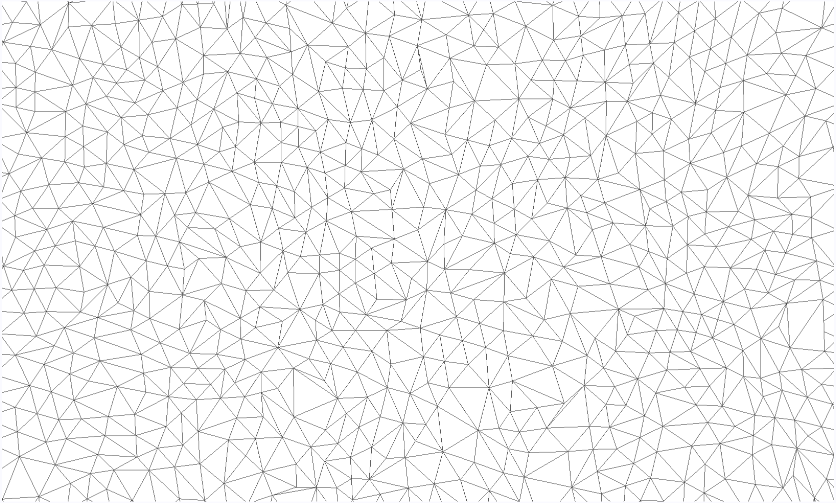
Remeshing Sequence : 1- Split edges



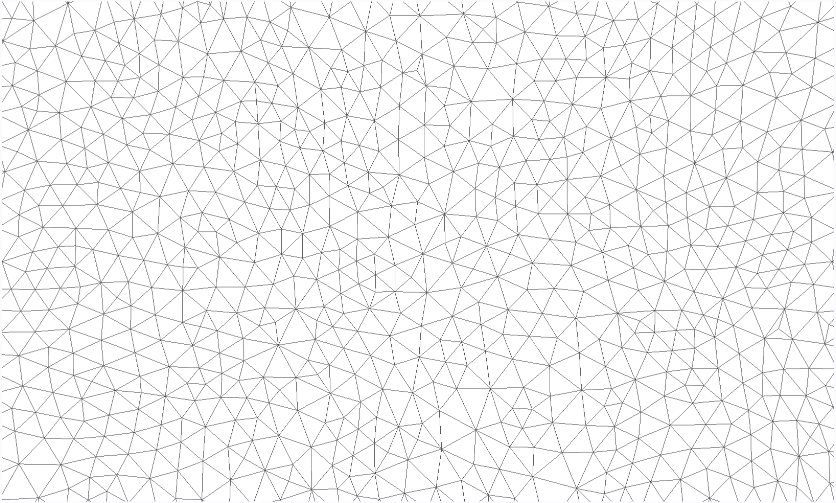
Remeshing Sequence : 2- Collapse edges



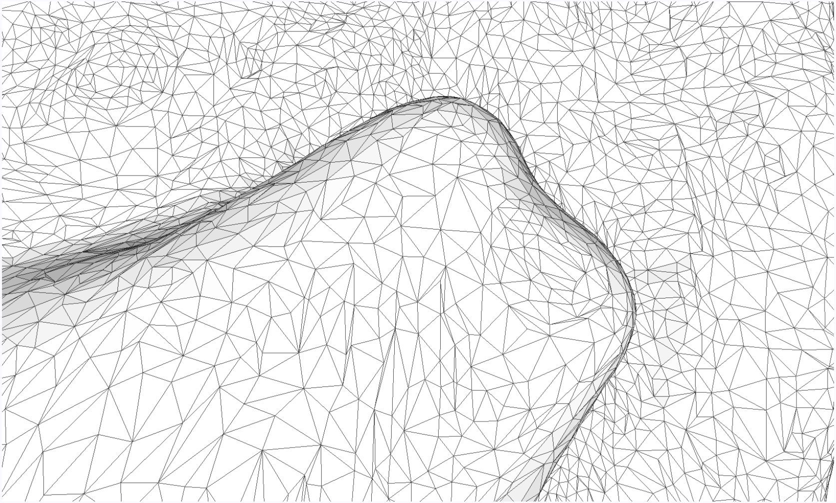
Remeshing Sequence: 3- Flip edges



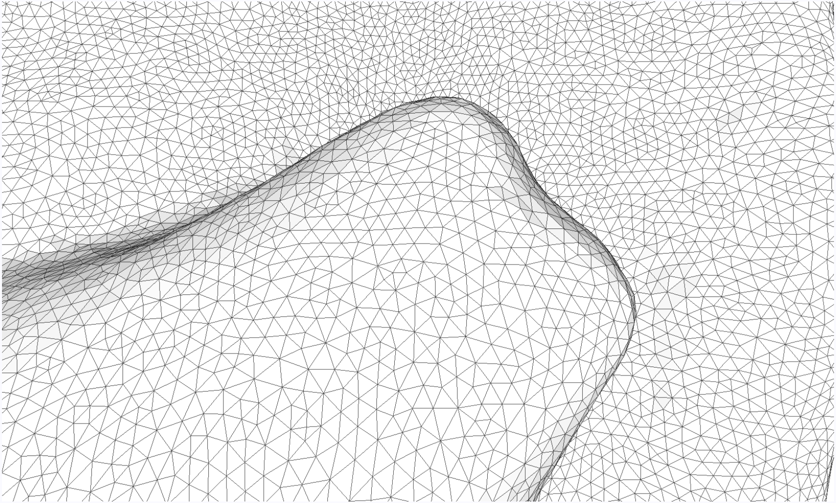
Remeshing Sequence: 4- Vertex reallocation



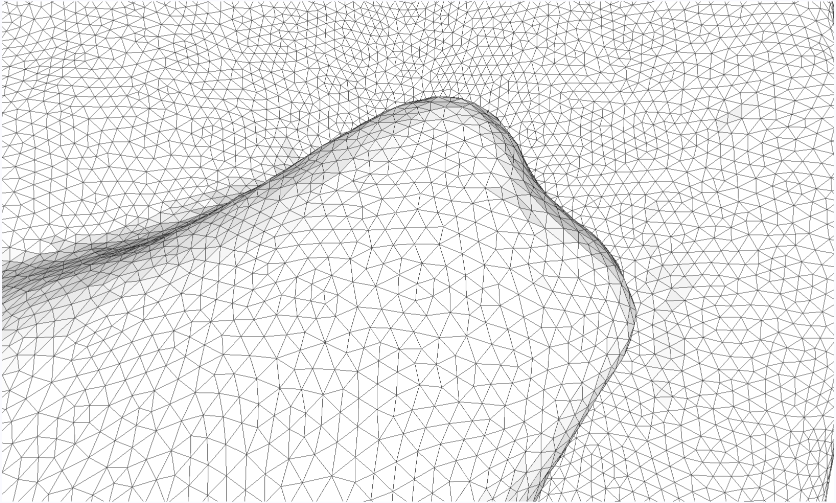
Remeshing iterations : input mesh



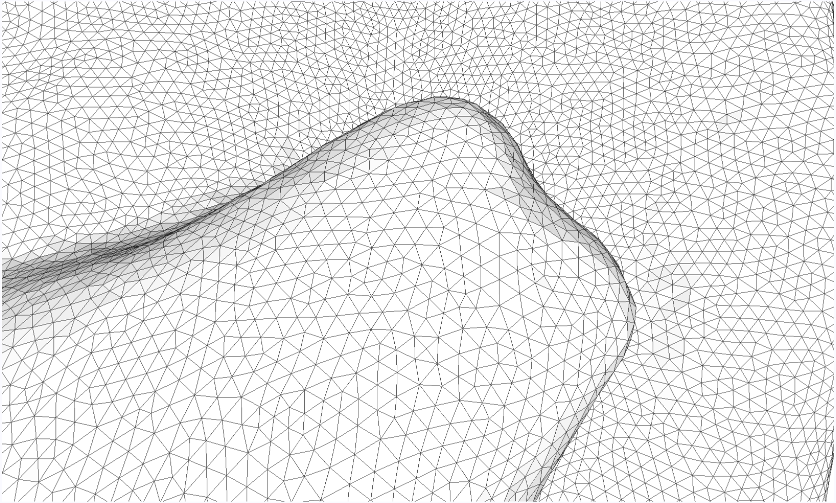
Remeshing iterations : Iteration 1



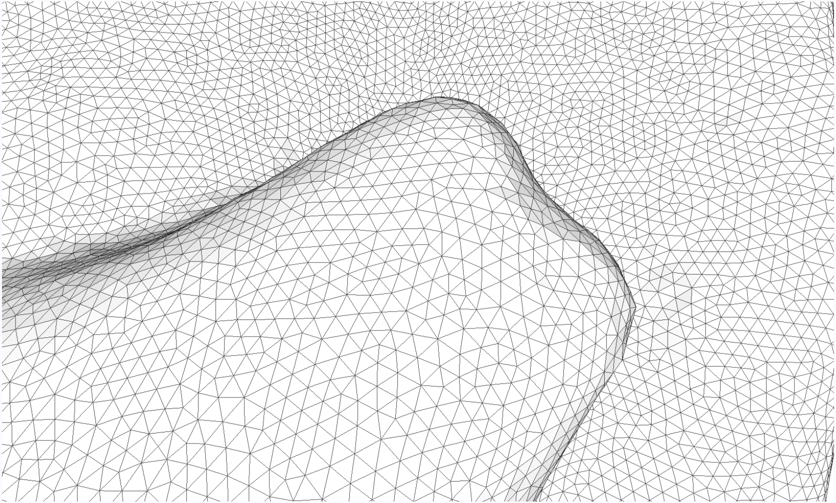
Remeshing iterations : Iteration 2



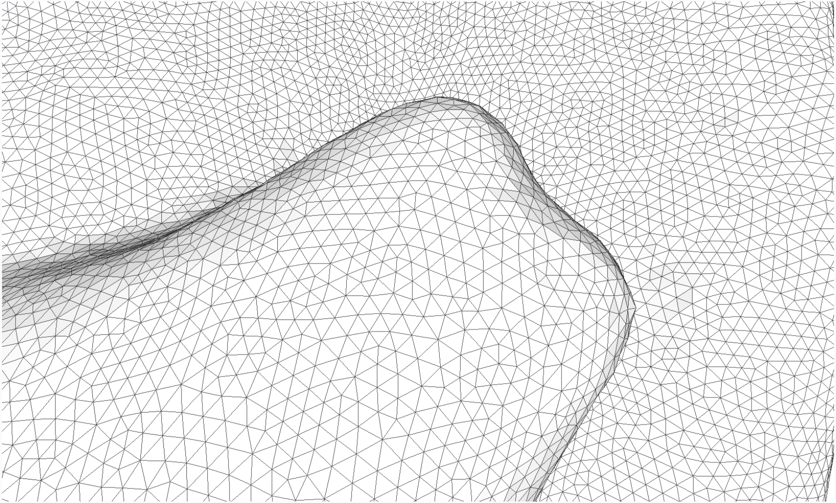
Remeshing iterations : Iteration 5



Remeshing iterations : Iteration 10



Remeshing iterations : Iteration 20



Some results

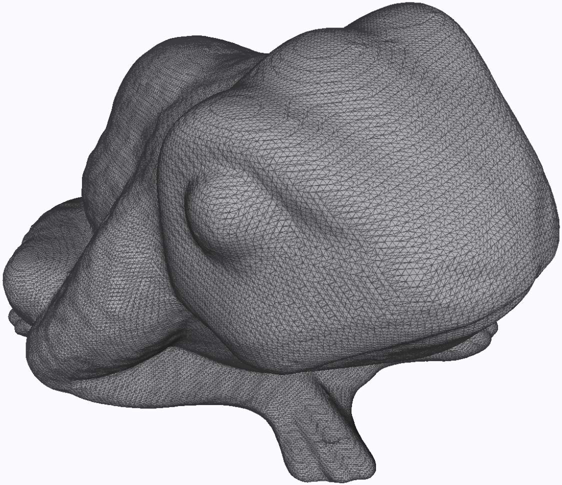


Figure: Input mesh.

Some results

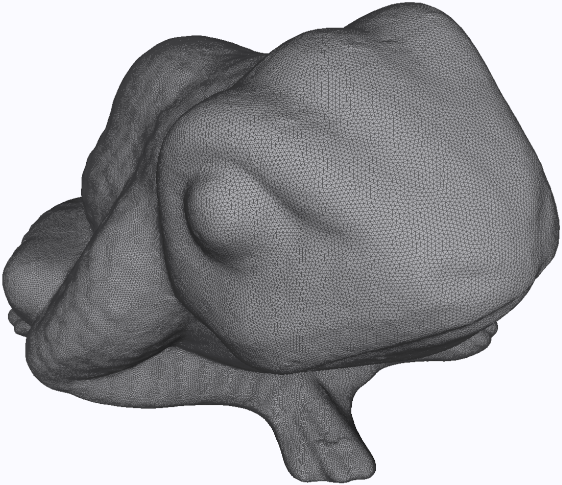


Figure: Random remeshing using tangential area smoothing.

Some results

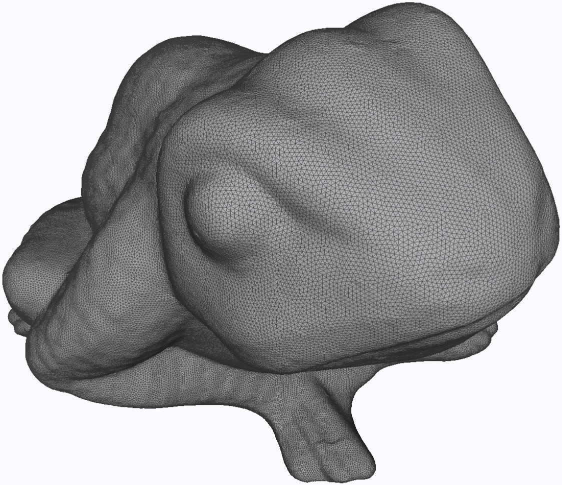


Figure: Sequential remeshing using tangential area smoothing.

Some results

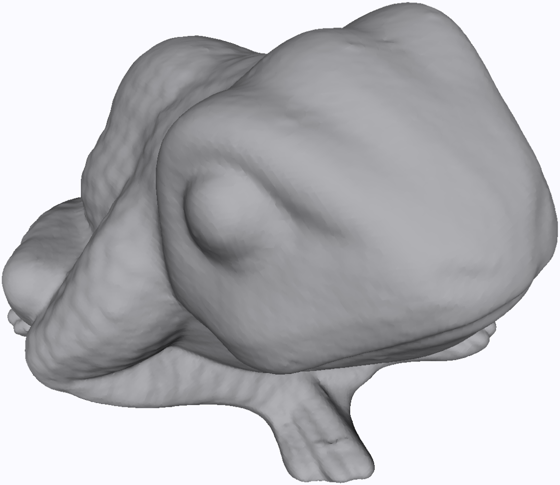


Figure: Input mesh

Some results

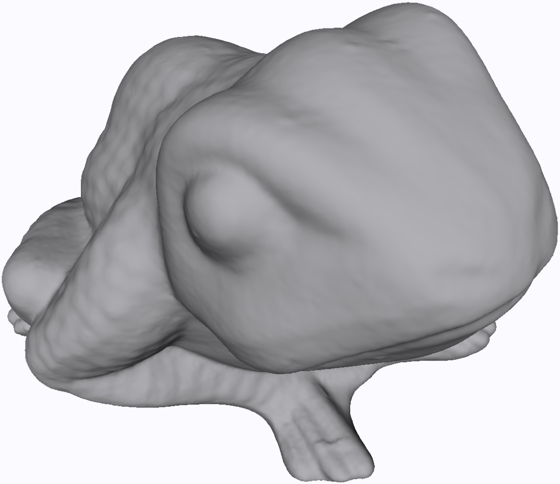


Figure: Sequential remeshing using tangential area smoothing.

Some results

Figure: Input mesh.

Some results

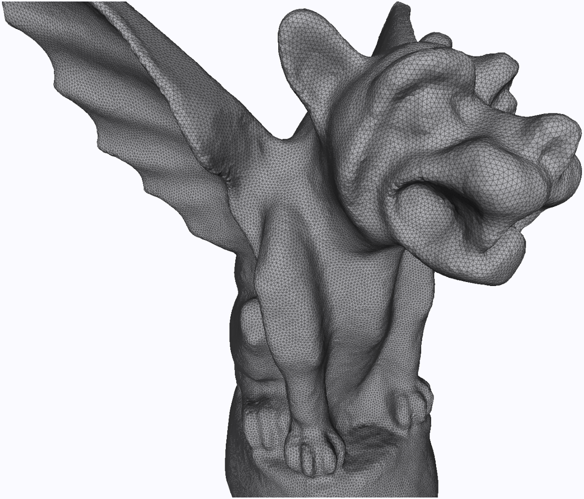


Figure: Sequential remeshing using tangential area smoothing.

Some results

Figure: Input mesh.

Some results

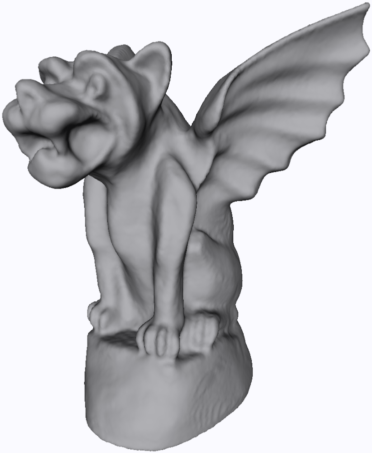


Figure: Sequential remeshing using tangential area smoothing.

Some results

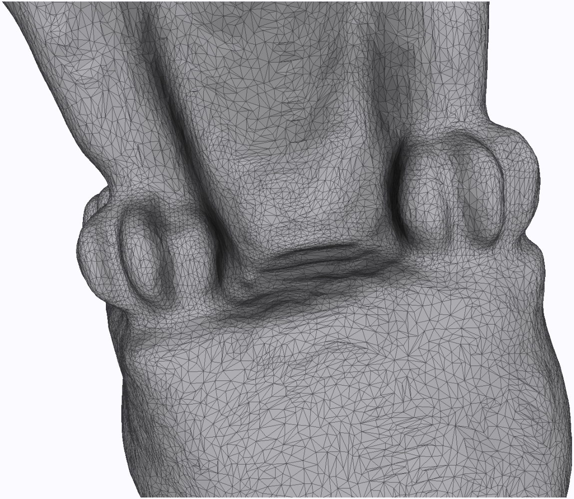


Figure: Input mesh.

Some results

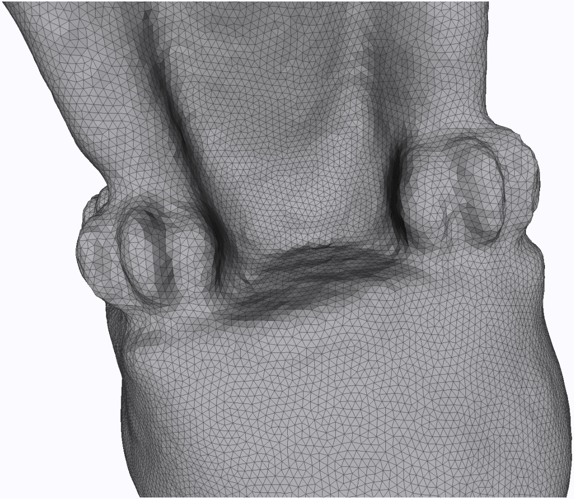


Figure: Sequential remeshing using tangential area smoothing.

Features guided remeshing

Difusion tensor

$$T(v_i) = \sum_{t_j \in N_t(v_i)} \mu_j n_{t_j} n_{t_j}^T$$

where t_j is a triangle, $N_t(v_i)$ denote the set of neighboring triangles of v_i, n_{t_j} is the normal of triangle t_j, and μ_j is the weight coefficient (we use $\mu = 1$).

Features guided remeshing

Difusion tensor

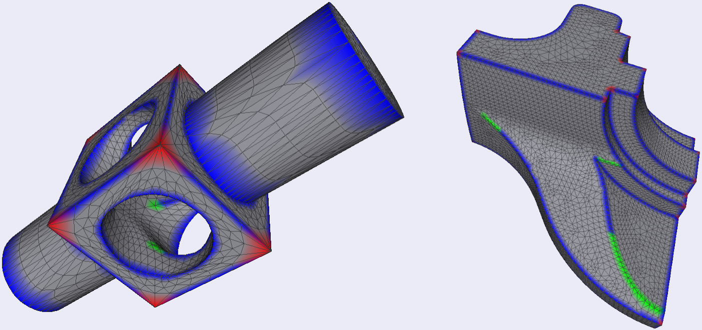
For each vertex of the mesh, $\lambda_1, \lambda_2, \lambda_3 \geq 0$ are eigenvalues of the corresponding structure tensor, then the feature analysis is documented as

- Face: if $\lambda_1 > 0.1$, $\lambda_2 < 0.02$
- Corner: if $\lambda_3 > 0.1$
- Strong: if $\lambda_2 > 0.1$, $\lambda_3 < 0.02$
- Weak: if $\lambda_1 > 0.1$, $0.1 \geq \lambda_1 \geq 0.02$, $\lambda_3 < 0.02$

The eigenvector corresponding to λ_3 is the difusion direction, used to define the neighboring vertex coincidence (NVC) condition.

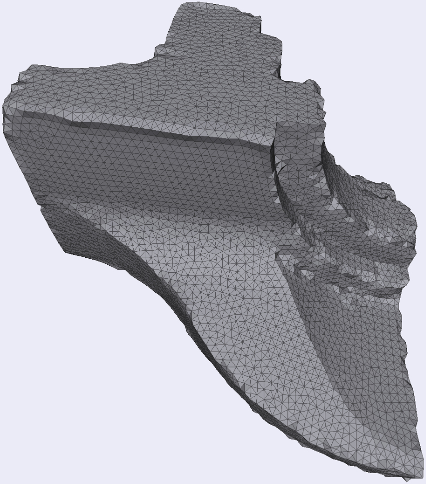
Features guided remeshing

Features detection



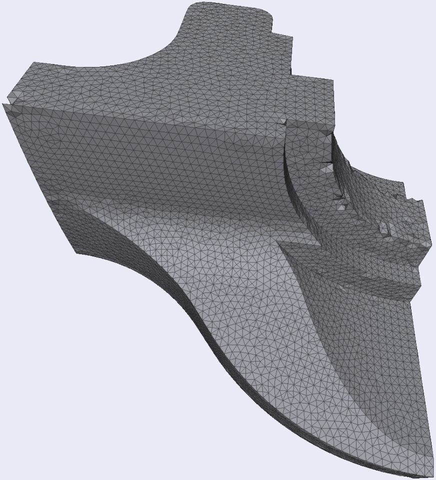
Features guided remeshing

Remeshing without using features constraints



Features guided remeshing

Remeshing using features constraints



THANKS!