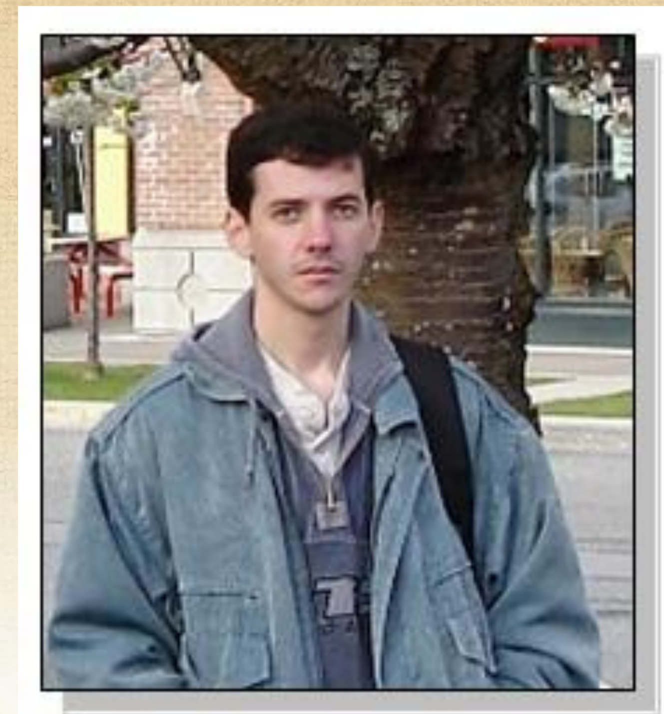
Cross-Parameterization and Compatible Remeshing of 3D Models

Vladislav Kraevoy
Alla Sheffer

University of British Columbia

Authors



Vladislav Kraevoy

Ph.D. Student

Alla Sheffer

Assistant Professor

Outline

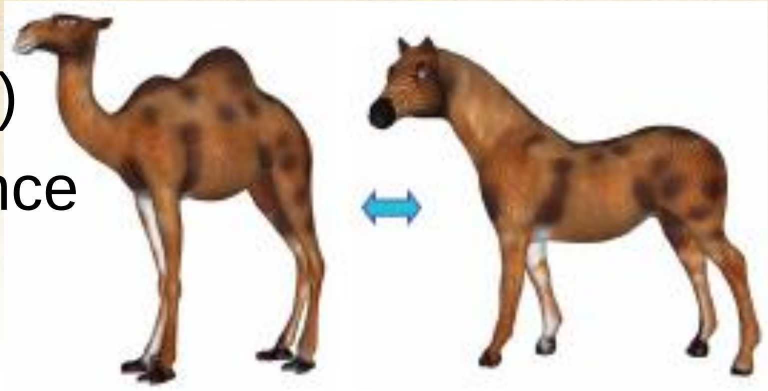
- Introduction
- Previous work
- Goal & Algorithm
- Results
- Conclusion

Introduction (1/3)

- Many geometry processing application require a bijective mapping between models
- What do we need?
  - Cross-parameterization
  - Compatible meshes

Introduction (2/3)

- Cross-parameterization
  - Parameterize the models on a common base domain
  - Requirements
    - Bijectivity (one-to-one)
    - Feature correspondence
      - vertex to vertex
    - Low distortion



Introduction (3/3)

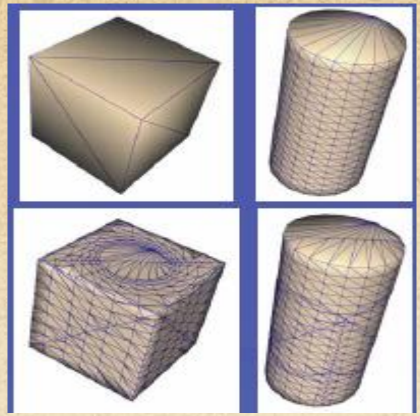
- Compatible meshes
  - Meshes with identical connectivity
  - Required by many applications of cross-parameterization
    - Morphing
    - Editing

Previous work (1/2)

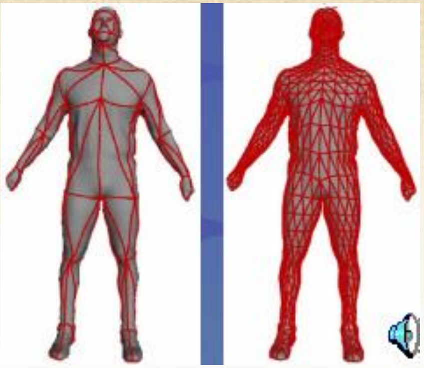
- Cross-parameterization
  - Sphere parameterization
    - Not guarantee a bijective mapping
    - Not always match the features
  - Base mesh
    - Segment meshes into triangular patches (same connectivity)
    - Map patches to base triangles

Previous work (2/2)

- **Compatible remeshing**
  - Mutual tessellation
    - Intersect meshes in parameter domain



  - Regular base mesh refinement
    - Remesh with subdivision connectivity



  - Both methods:
    - output meshes much larger (~x10) than input

Goal

- **Cross-Parameterization**
  – Bijective
  – Exact feature vertex correspondence
  – Low distortion (preserve shape)
  – Minimal user input: models + feature vertices

- **Compatible remeshing**
  – Closely approximate the input models
  – Similar (order of magnitude) number of elements as input

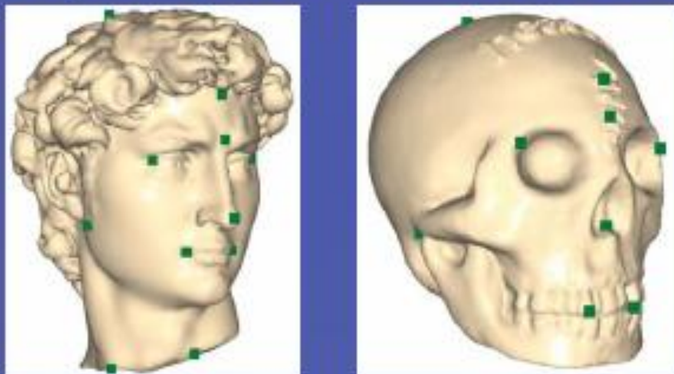
Algorithm

- 3 main stages
  - 1.Construct a common base domain
  - 2.Low distortion cross-parameterization
  - 3.Compatibly remeshes the input models

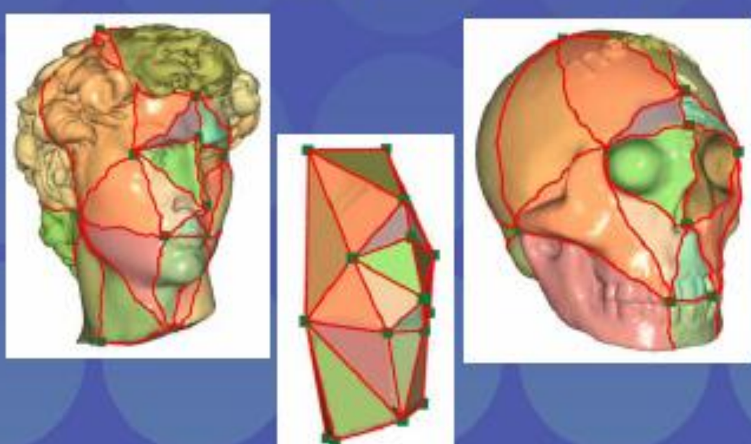
Algorithm Stages

SIGGRAPH2004

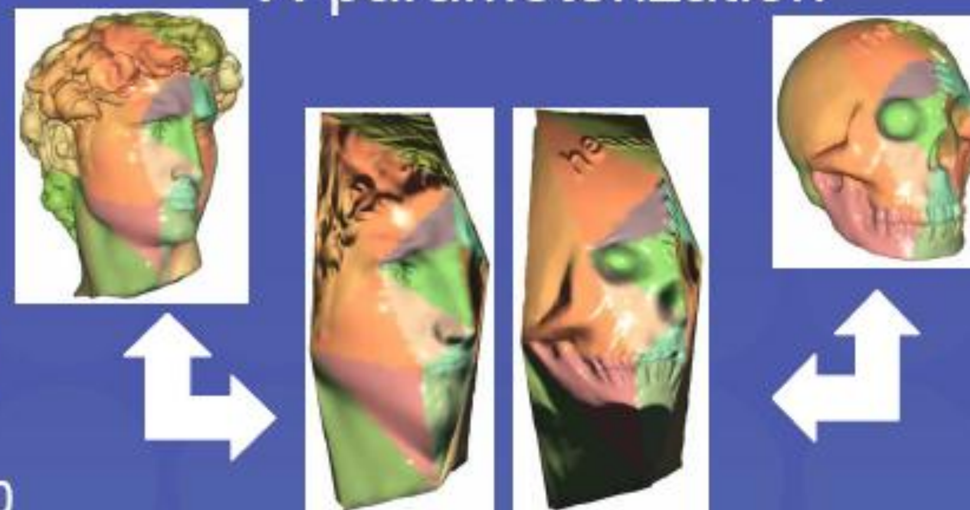
Input: models + features



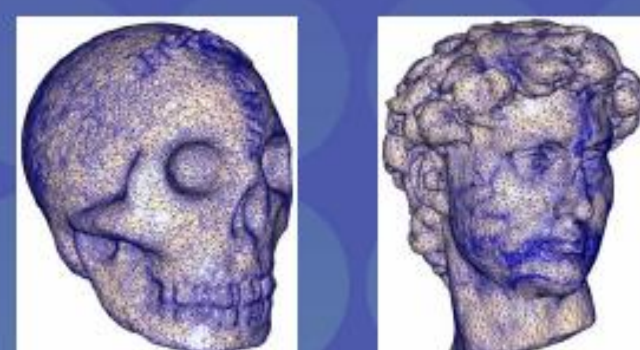
1. Common base mesh construction
   - Provably correct for genus 0



2. Low distortion, bijective X-parameterization

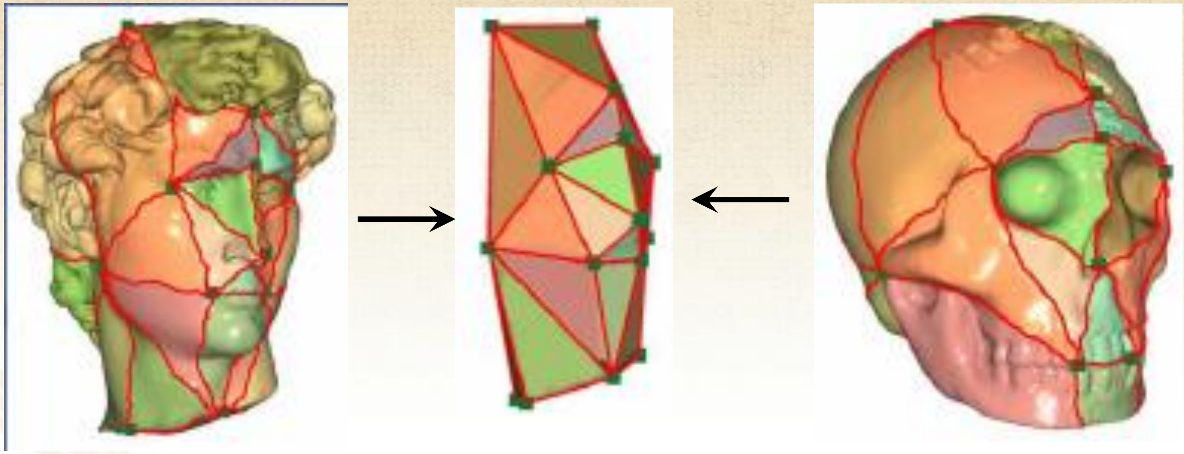


3. Compatible remeshing



Algorithm

- Construct a common base domain
  - Goal: topologically identical triangular patch

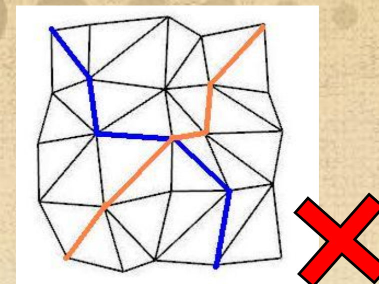


Partition

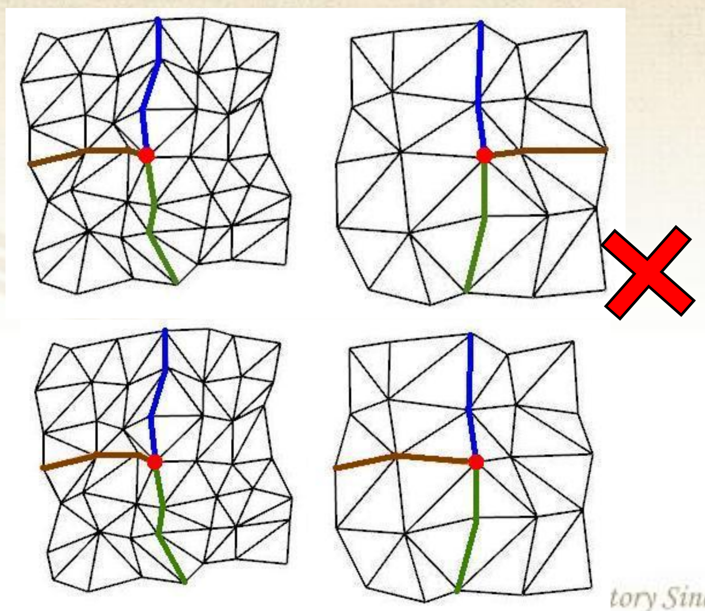
- Assign feature points on both 2 meshes.
- Find the shortest path between each pair of feature vertices. (Dijkstra search)
  - The search is satisfied the legality condition
- Select the best pair of corresponding path and split the mesh.
  - Sort by the sum of path lengths on 2 meshes.
- Adding match path as we can

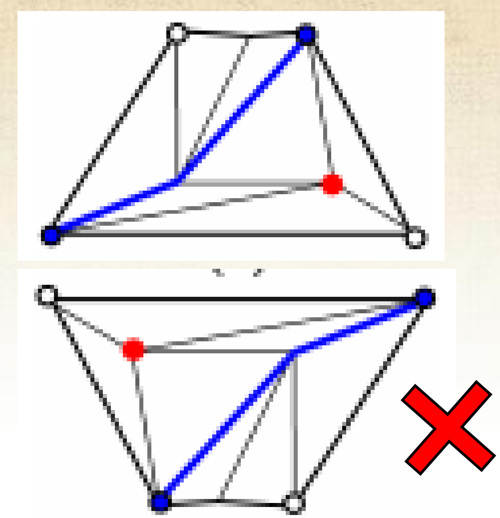
Legality Conditions

- Paths don't intersect



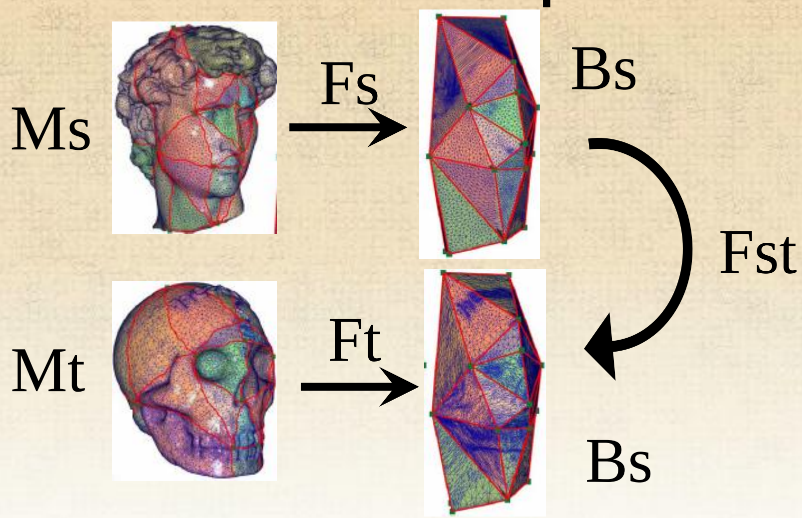
- Consistent neighbor ordering
- Don't blocking





Algorithm

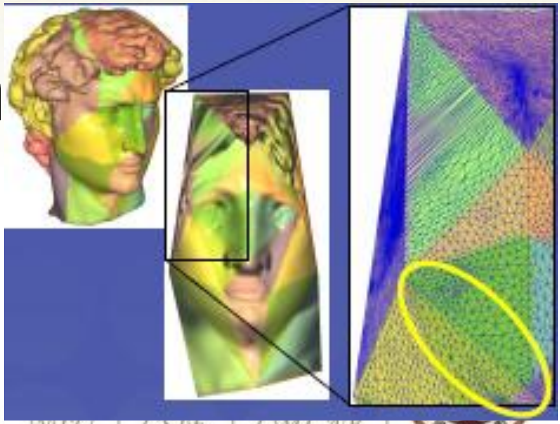
- Initial cross-parameterization



$$M_s \rightarrow M_t$$

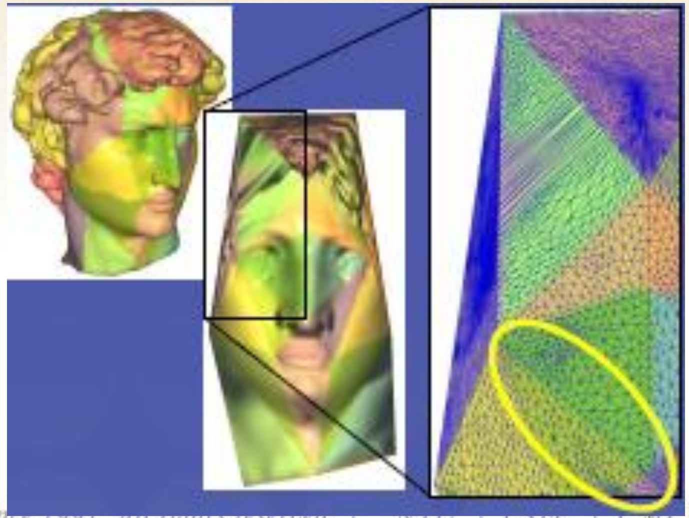
$$F = F_t^{-1} \cdot F_{st} \cdot F_s$$

- Bad patch = high distortion
  - Solution: smoothing

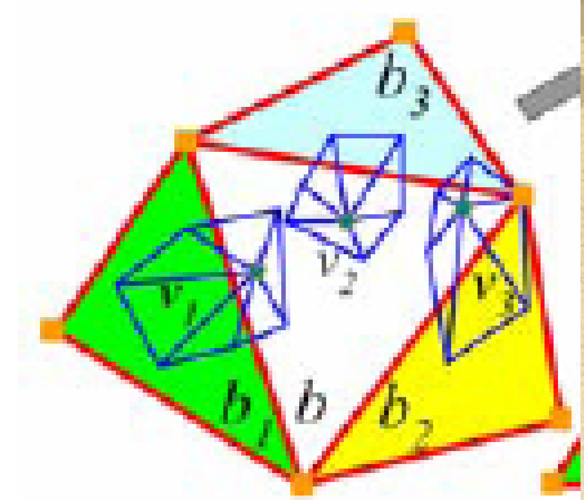


Algorithm

- Smoothing
  - All edges should satisfies adjacency constraint (guarantee a bijective mapping)
    - End vertices belong to the same patch
    - or to 2 adjacent patches (share a common boundary path)

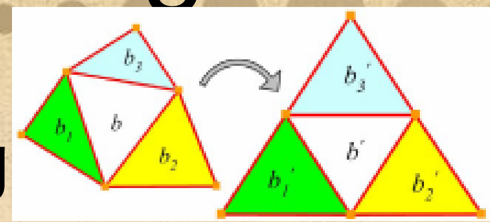
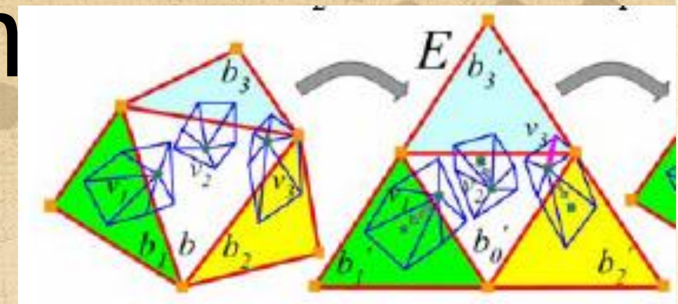


Algorithm



- **Smoothing**
  - For each mesh vertices
    - repeatedly modifying their locations on the base mesh
  - The base mesh location of a vertex v
    - <b , v_b>
    - b : base triangle index
    - v_b : the barycentric coordinates defined with respect to the base triangle
  - Relocate the vertex based on the locations of the neighboring vertices

Algorithm

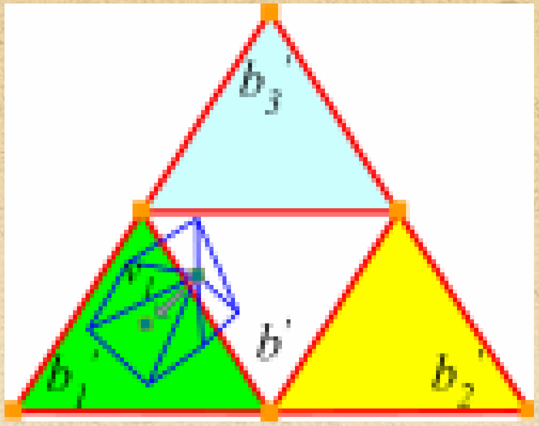



- **Smoothing**
  - 1. Set the three triangles adjecent to b
  - 2. Map to a planar equilateral triangle E
  - 3. mapping is defined
    - compute v to b using the barycentric coor.
    - compute the neighbor of v to E
  - 4. compute the new location of v_E

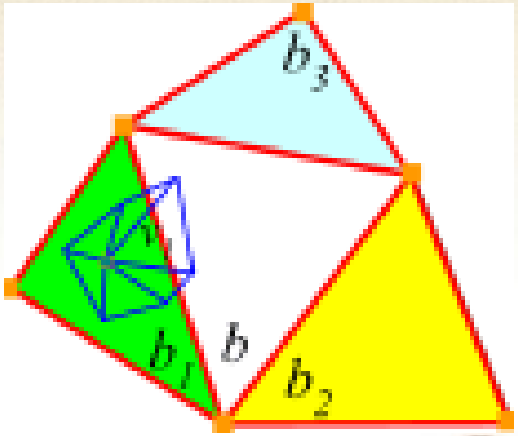
$$v_E = \frac{1}{|Nei(v)|} \sum_{u \in Nei(v)} w_{uv} u_E$$

  - *Weight : mean value edge weight computed on the original mesh*

Algorithm

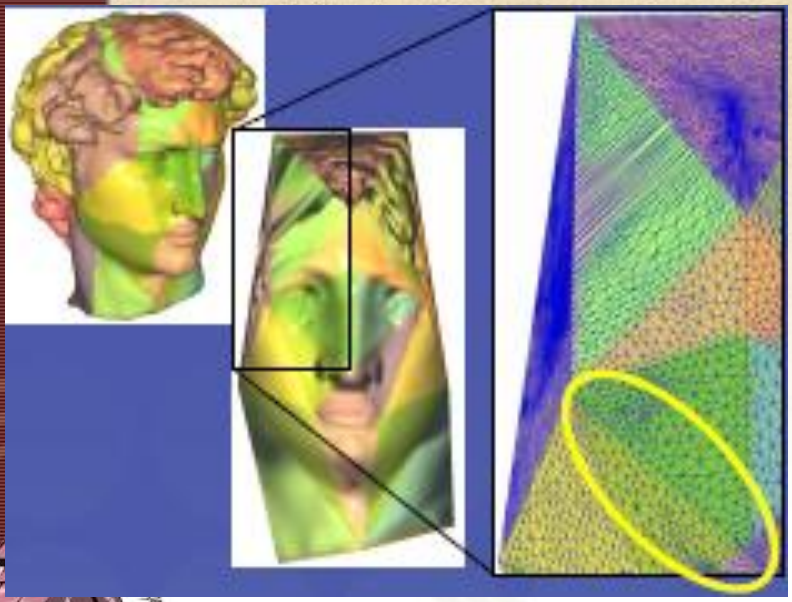


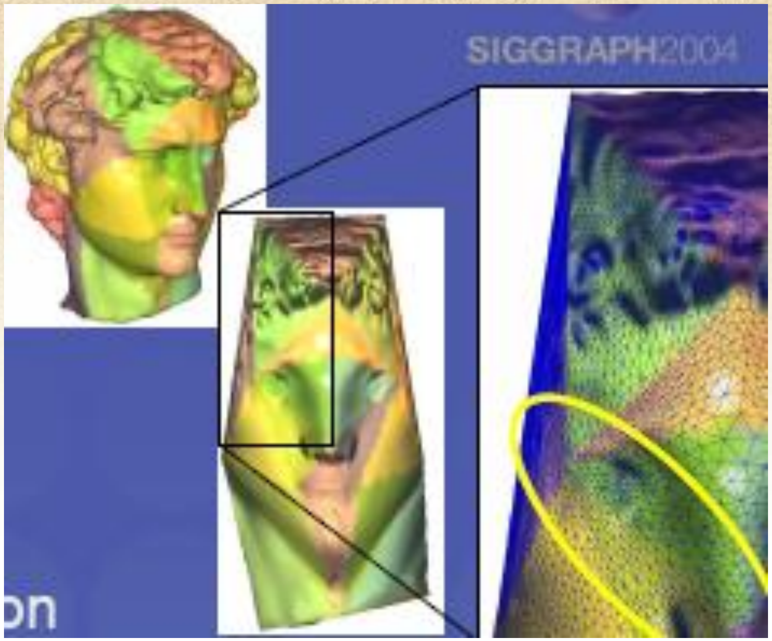
- Smoothing
  - 5. Find the new <b’ ,v’>
  - 6. Check <b’,v’> if is satisfied the constraint
    - Satisfied : update the location of v to <b’,v’>



Algorithm

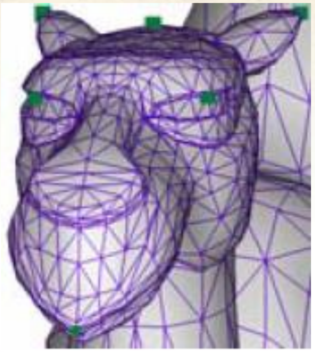
- After Smoothing



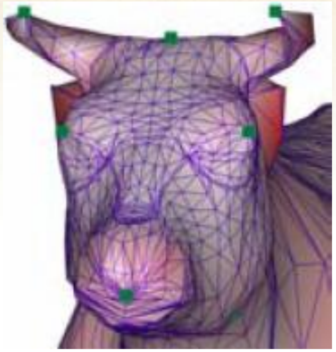


Algorithm

- Compatible remeshing
  - Use connectivity of one model ("source") as basis
  - Map to second model ("target") using X-parameterization



$M_s \rightarrow M_{st}$



Algorithm

- Compatible remeshing
  - 1. Compute the distance approximation error
  - 2. error > threshold :
    - Relocate the vertices of M_{st} using smoothing
    - Refine the meshes
  - 3. Compute the normal approximation error and perform “pseudo” edge-flip refinement

Algorithm

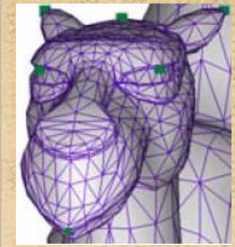
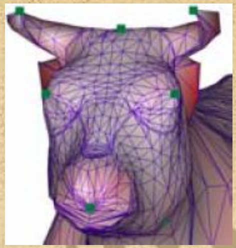
- Compatible remeshing
  - 1. Compute the distance approximation error
  - Measure the distance between the vertices of M_t and the approximation surface

$$e(v) = (F' F^{-1}(v) - v)$$

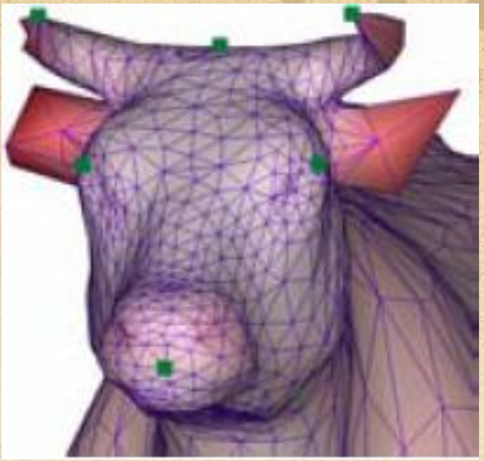
$$e_{uv} = \left(\frac{e(u) + e(v)}{2} + e(\frac{u+v}{2}) / 2 \right)$$

$$w_{uv} = (e_{uv} + w_{uv}) / 2$$

Algorithm

$M_s \rightarrow M_{st}$

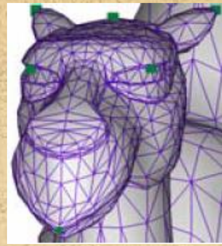
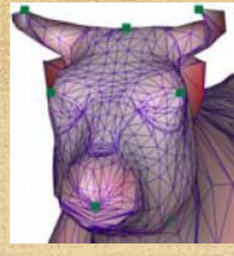


- Compatible remeshing
  - 2-1. smoothing
  - Modify the mapping F
  - The smoothing is same but weight is different

$$v_E = \frac{1}{|Nei(v)|} \sum_{u \in Nei(v)} w_{uv} u_E$$

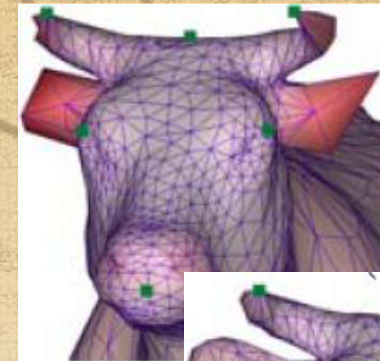
$$e_{uv} = \left(\frac{e(u)+e(v)}{2} + e\left(\frac{u+v}{2}\right) / 2 \right)$$

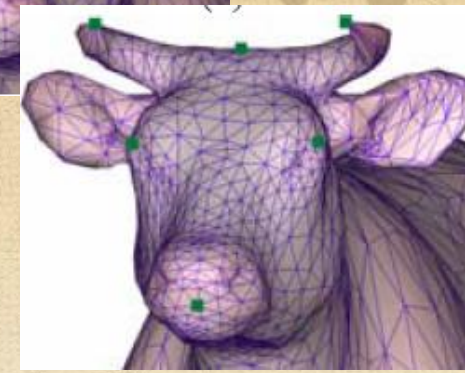
$$w_{uv} = (e_{uv} + w_{uv}) / 2$$

Algorithm

$M_s \rightarrow M_{st}$





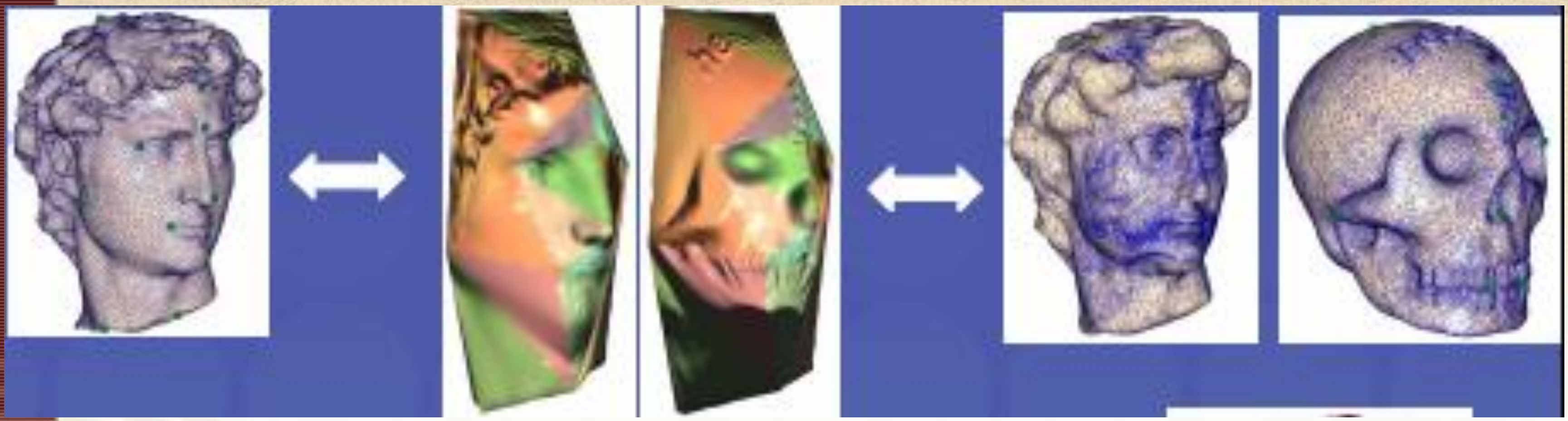
- Compatible remeshing
  - 2-2. Refinement
  - Smoothing alone can't approximated the geometry accurately
  - Refinement is performing edge split operation
  - A edge is refined if $e_{uv} > \max(\frac{1}{2}\max_{(u',v')}(e_{u'v'}), \varepsilon)$
    - ε: user defined
    - Edge is split by placing a vertex at its midpoint

Algorithm

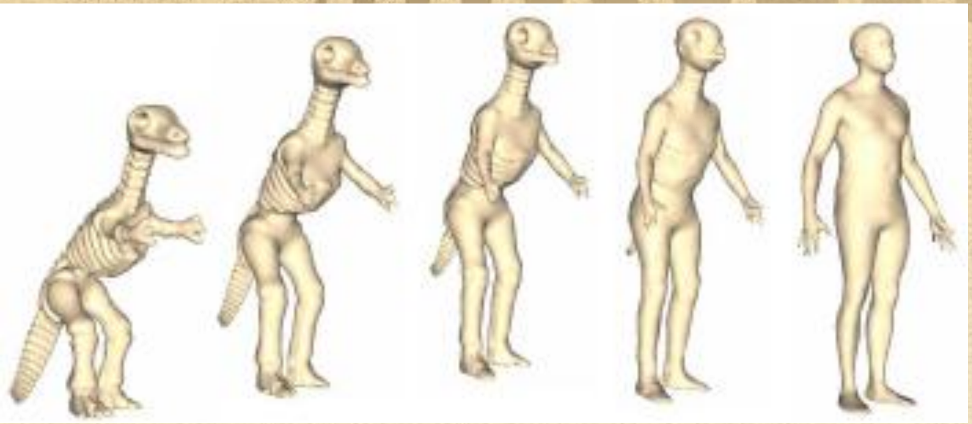
- Compatible remeshing
  - 3. Pseudo edge flips
  - May be large deviation between the face normal of M_t and M_{st}
  - Resloved by edge flips
  - Reproduces the change in normals achieve by a flip

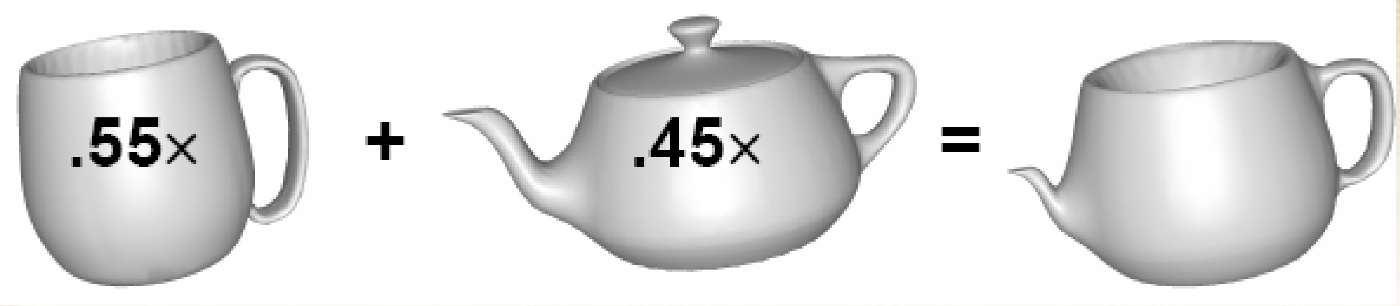
Algorithm

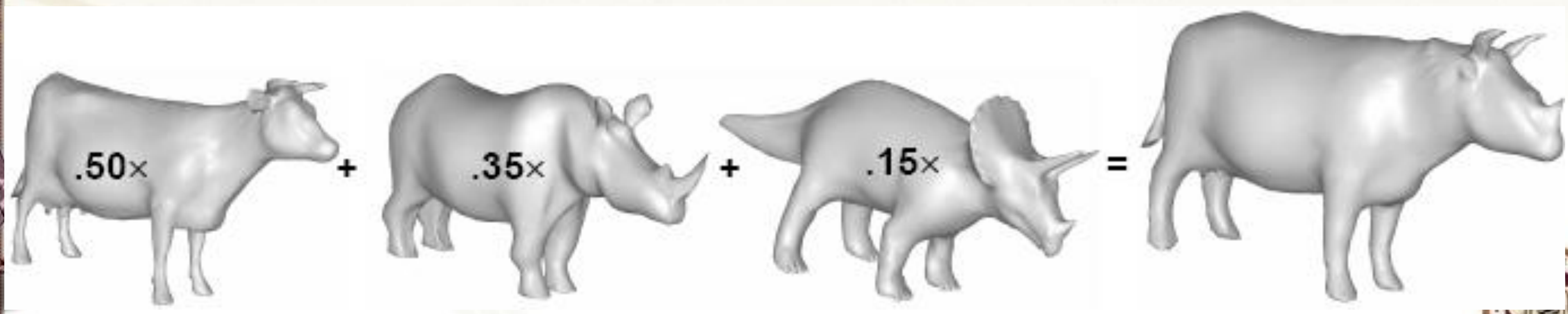
- After Compatible remeshing



Results



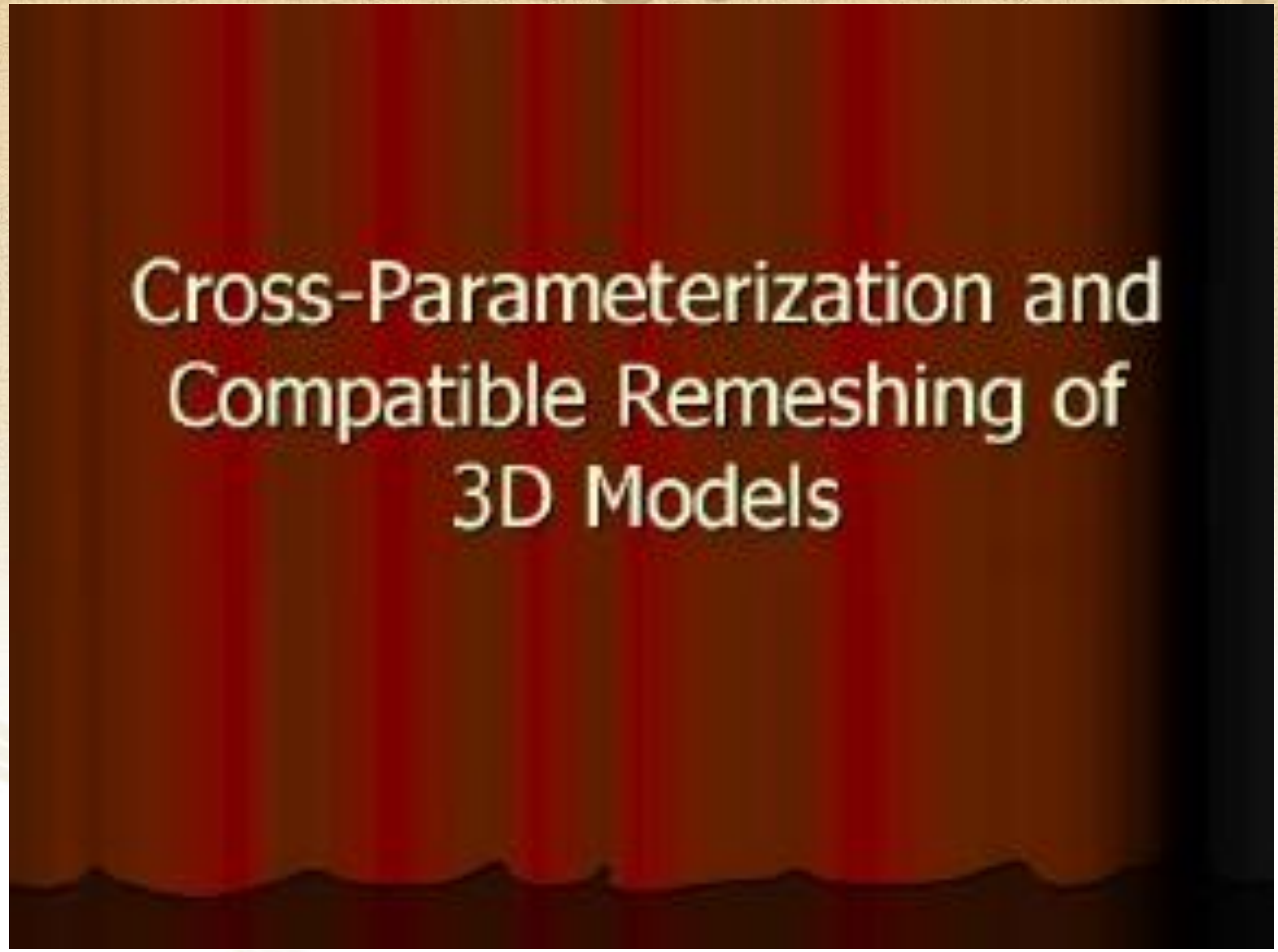




Conclusion

- Robust method for constrained X-parameterization
- New framework for low-distortion parameterization on base mesh
- New compatible remeshing scheme
  - Closely approximate input
  - Small output mesh
    - 20% more triangles

Movie


Cross-Parameterization and
Compatible Remeshing of
3D Models

CMLAB NATIONAL TAIWAN UNIVERSITY

~ The End~