

# Mesh Editing With Poisson-Based Gradient Field Manipulation

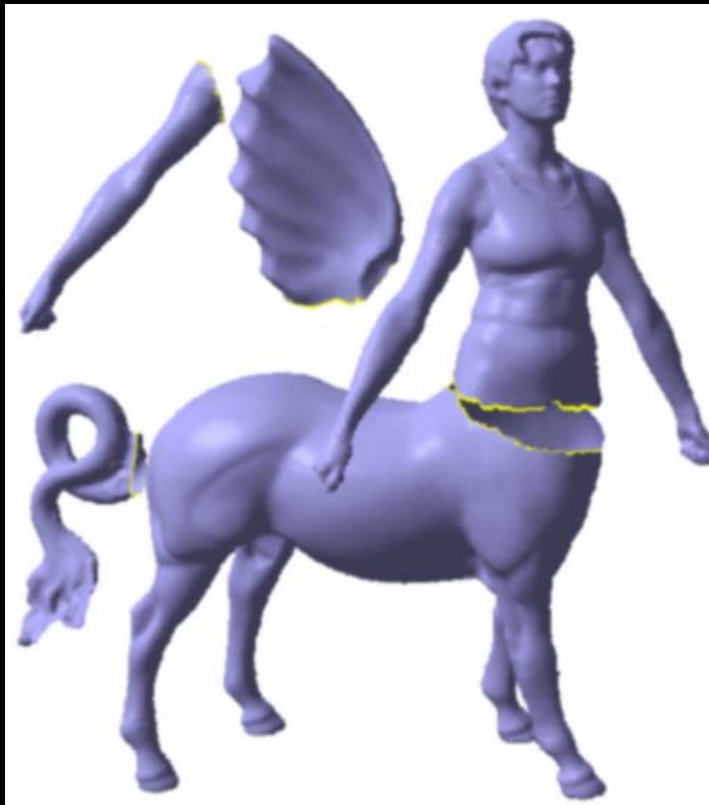
*Yizhou Yu   Kun Zhou   Dong Xu   Xiaohan Shi  
Hujun Bao   Baining Guo   Heung-Yeung Shum*

Univ. Illinois Urbana-Champaign  
Microsoft Research Asia  
Zhejiang University

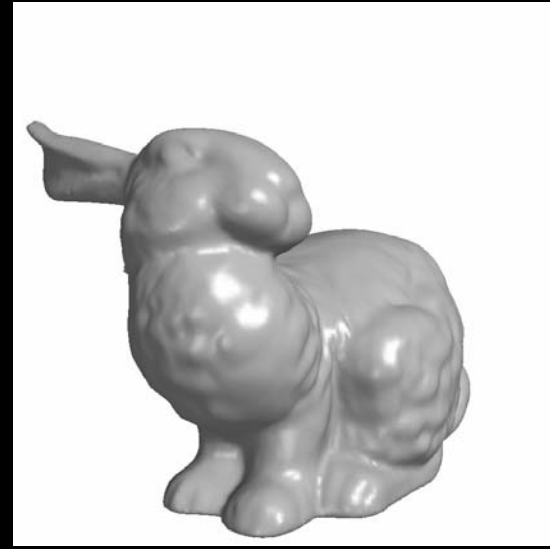
# Introduction

- Implicit vs. Explicit Mesh Editing
  - Explicit methods: responsive but potentially with more user interaction and artifacts
  - Implicit methods: less interaction and artifacts but relatively more expensive
- Motivations for Poisson-Based Editing
  - The gradient is a local differential property that can be modified conveniently
  - Artifacts can be removed during least-squares minimization
- What editing operations are we talking about?

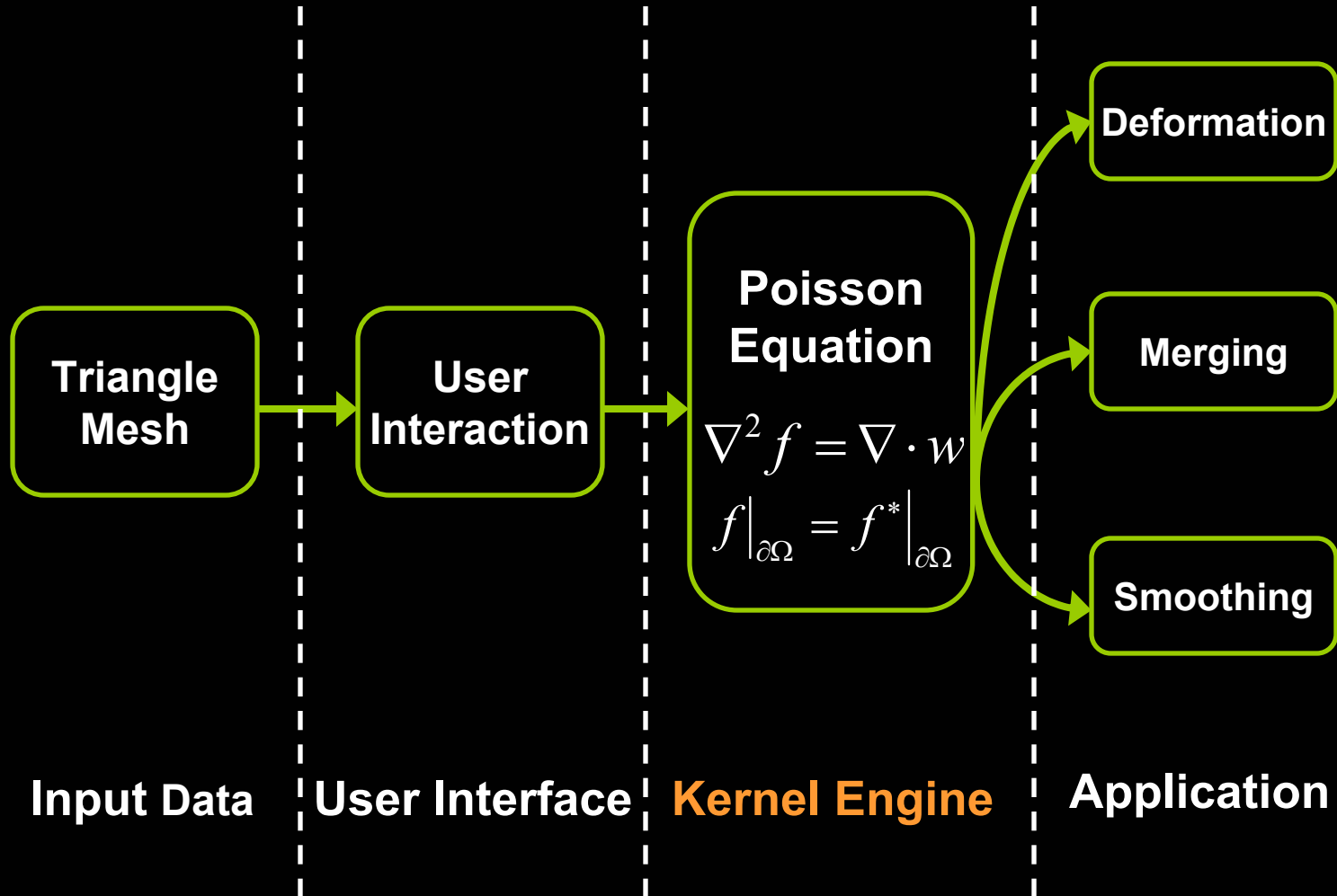
# Mesh Editing: Object Merging



# Mesh Editing: Deformation



# Overview



# Vector Fields and Poisson Equation

- Given a vector field  $\mathbf{w}$ , how can we approximate it using the gradient field of a scalar function?
  - Mathematically, we want to solve this minimization

$$\min_{\phi} \iint_{\Omega} \|\nabla \phi - \mathbf{w}\|^2 dA$$

- The Poisson equation solves the same problem.
  - It recovers an unknown scalar function from a given vector (guidance) field and a boundary condition.

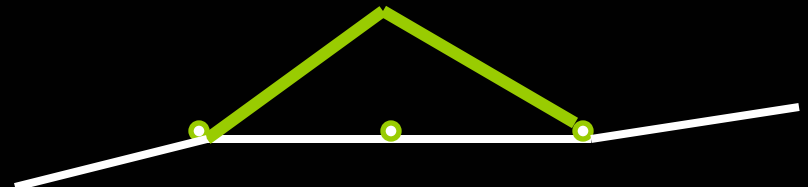
$$\Delta \phi = \nabla \cdot (\nabla \phi) = \nabla \cdot \mathbf{w}, \quad \phi|_{\partial\Omega} = f^*|_{\partial\Omega}$$

$$\Delta \phi = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2}, \quad \nabla \cdot \mathbf{w} = \frac{\partial w_x}{\partial x} + \frac{\partial w_y}{\partial y} + \frac{\partial w_z}{\partial z}$$

# Discrete Fields on Triangle Meshes

[Polthier and Preuss 2000]

- Need new field definitions for discrete irregular grids, such as a triangle mesh.
- Discrete Vector Fields
  - Piecewise constant vector fields, i.e. a constant vector within each triangle. The vector is coplanar with the triangle.
- Discrete Potential Fields
  - Piecewise linear potential fields, i.e. the potential is a linear combination of piecewise-linear basis functions.
  - $$\phi(\mathbf{x}) = \sum_i \phi_i B_i(x)$$
  - where the weights for the bases are defined at the vertices of the grid.



# Poisson Equation on Triangle Meshes

[Tong *et al.* 2003]

- A Poisson equation for discrete fields on triangle meshes can be defined.

$$\text{Div} (\nabla \phi) = \text{Div} \mathbf{w},$$

$$(\text{Div} \mathbf{w})(\mathbf{v}_i) = \sum_{T_k \in N(i)} \nabla B_{ik} \cdot \mathbf{w} |T_k|$$

- It essentially has the same properties as the original Poisson equation.
- It is actually a sparse linear system:

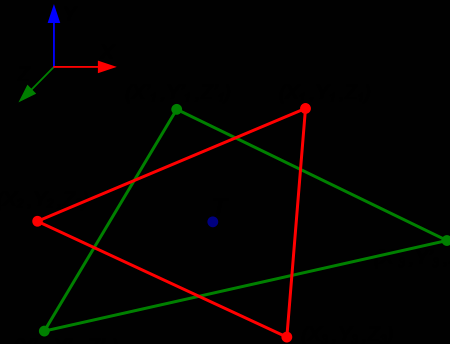
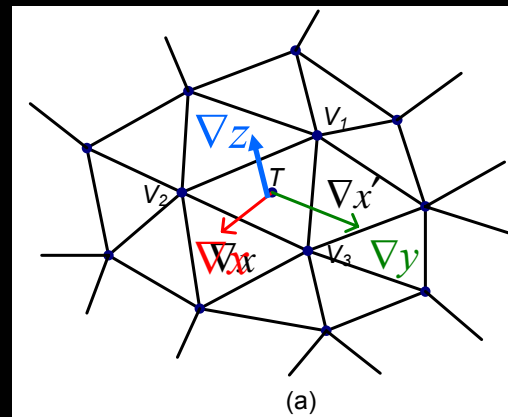
$$\mathbf{Ax} = \mathbf{b}$$

- How can we apply this Poisson equation to mesh geometry?

# A Basic Poisson Mesh Solver

- Each of the  $x$ ,  $y$  or  $z$  coordinates over a mesh is a piecewise-linear function defined over itself.
  - The respective coordinate of the vertices are the weights for the basis functions.
- The gradient of such functions are piecewise constant vector fields.

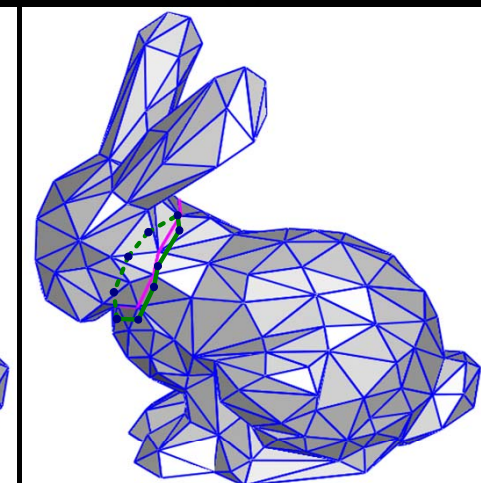
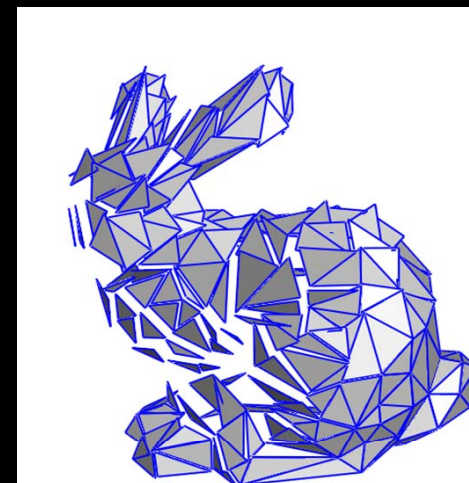
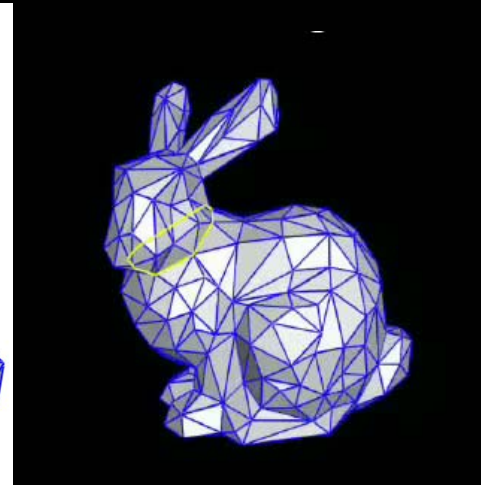
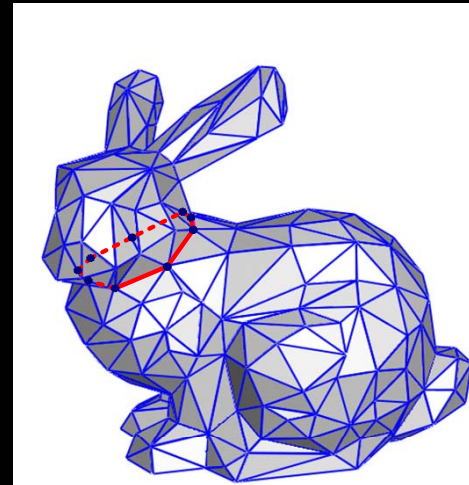
3 vectors per triangle  
They are coplanar  
with the triangle



- **Key observation:** If we modify these vector fields, new potential fields (vertex coordinates) can be reconstructed using the Poisson equation. That is, a new mesh is generated!  
**How to modify the vector fields?**

# Editing Gradient Fields Using Local Transforms

- The 3 vectors on each triangle should be modified at the same time to produce geometric meanings.
- Local transforms are applied to the triangles whose gradients produce new vectors
  - Local transforms include rotation and scaling
- Where do the local transforms come from?



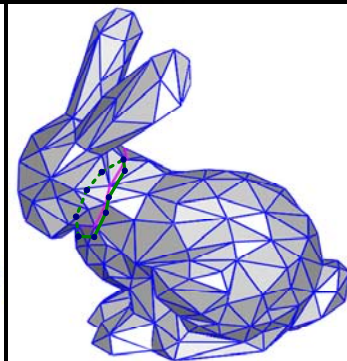
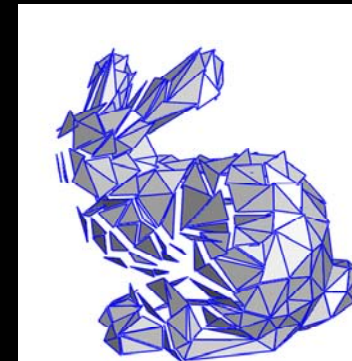
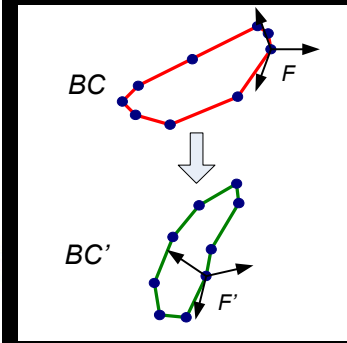
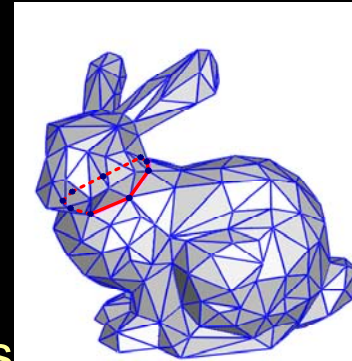
# Boundary Condition Editing

- Editing boundary conditions significantly reduces the amount of user interaction.
  - Boundary conditions only involve a sparse subset of the mesh, yet can influence the solution over the whole mesh.
  - We use curves and vertices as boundary conditions.
- However, only editing boundary conditions does NOT work!
- The gradient fields need to be updated using local transforms.



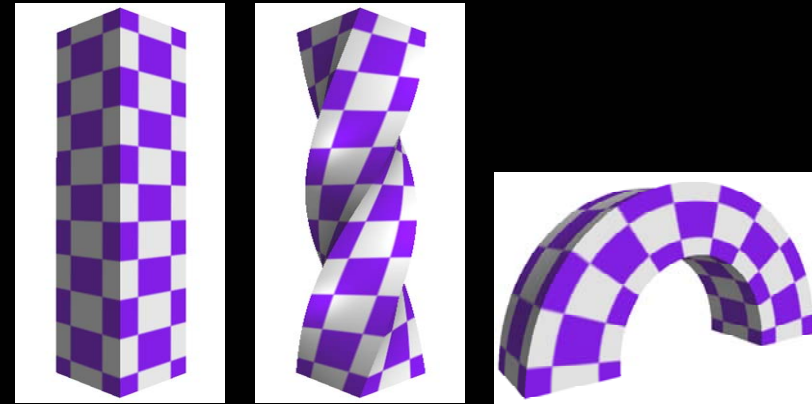
# Propagating Local Transforms from Boundaries

- A single curve:
  - At each vertex on the curve, obtain a local 3D transform.
  - Simplest scheme: for every free vertex in the editable region of the mesh, find the nearest vertex on the curve and “steal” its local transform.
  - Filtering schemes: *Uniform, Linear or Gaussian*
  - The required distance transform is computed by the Level Set method [Sethian 1999].
- Multiple Curves
  - Weighted average of the transforms from all curves. Rotations are represented by quaternions.
  - Weights based on distance: *constant, linearly decreasing, or a cosine wave*



# Mesh Deformation

- Curve- or vertex-driven boundary condition editing
- Simultaneous editing with minimal user interaction
  - Translation, rotation and/or scaling applies to all vertices on the same curve.
  - Simultaneous rotation of all the vertex normals around their respective tangents
- Individual editing
  - Local transforms apply to individual vertices to achieve small scale changes.



4000 triangles, 578ms & 609ms



2480 triangles, 230ms & 240ms

# Detail Editing



# Mesh Deformation Comparison



Original



Naïve  
Poisson



WIRE



Poisson  
Final

# Acceleration for Interactive Deformation

- Build a multiresolution mesh pyramid for large meshes [Guskov *et al.* 1999] and perform mesh editing at a coarse resolution.
- When solving the linear system  $Ax=b$ , the inverse of matrix  $A$  can be pre-computed.
- Detail editing is performed on the finest level, but confined to a small region.

# Interactive Deformation

11.5 FPS



Coarse resolution: ~2000 triangles

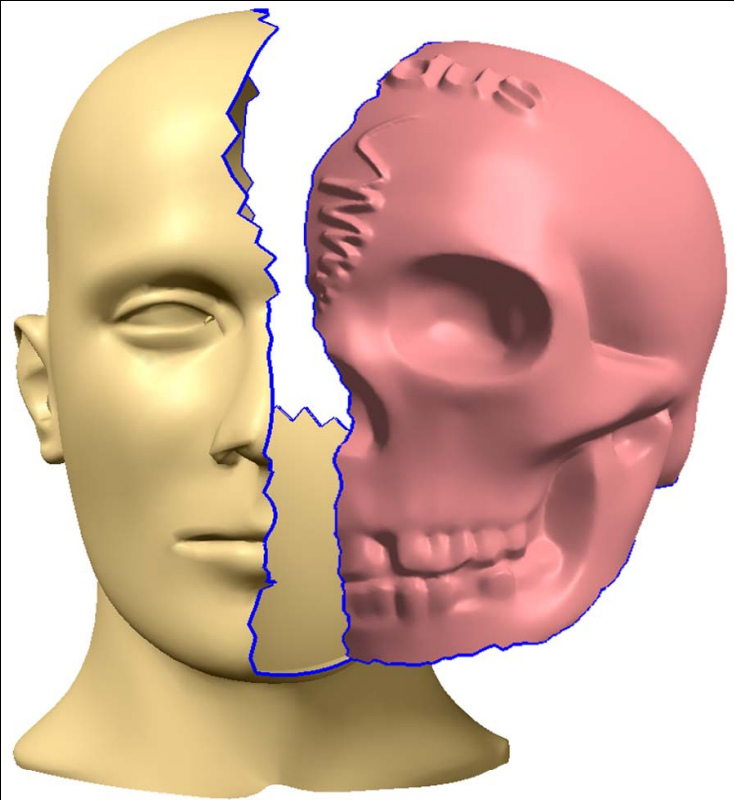
Fine resolution: ~70,000 triangles

Processor: Intel Xeon 1.5GHz

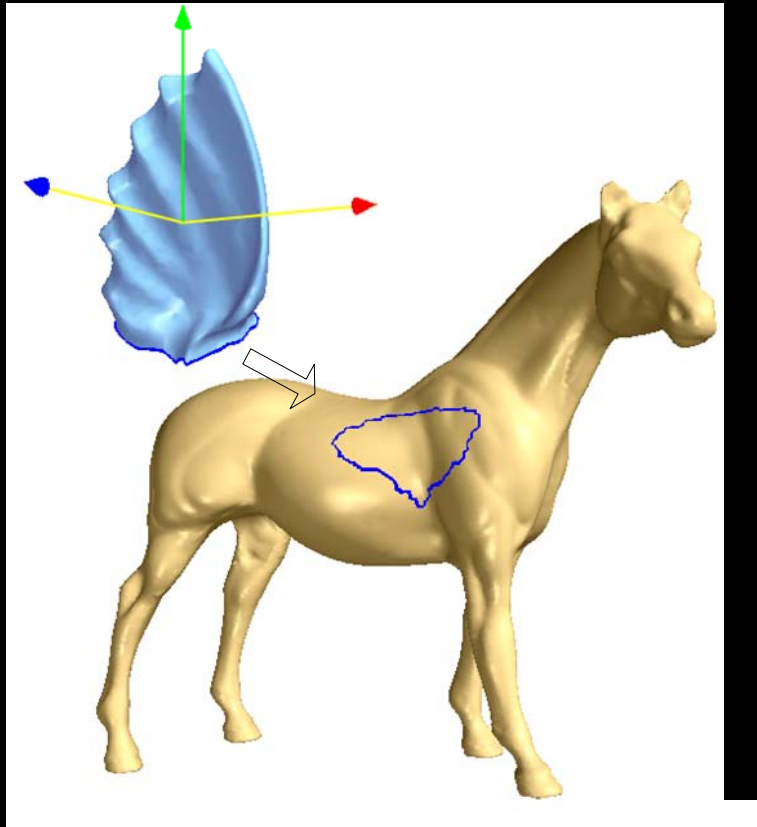
# Object Merging

- Steps for merging two partial meshes at their open boundaries
  - Define local frames everywhere on the two boundaries;
  - Define vertex correspondence between them;
  - Generate an intermediate mesh boundary including interpolated vertex positions and local frames;
  - Consider the intermediate mesh boundary as the new boundary condition for both meshes, and perform boundary condition editing on both.
- Advantages of this Poisson-based method
  - The original mesh boundaries can be very different.
  - Local transform propagation globally adjusts the meshes to make them compatible with each other.

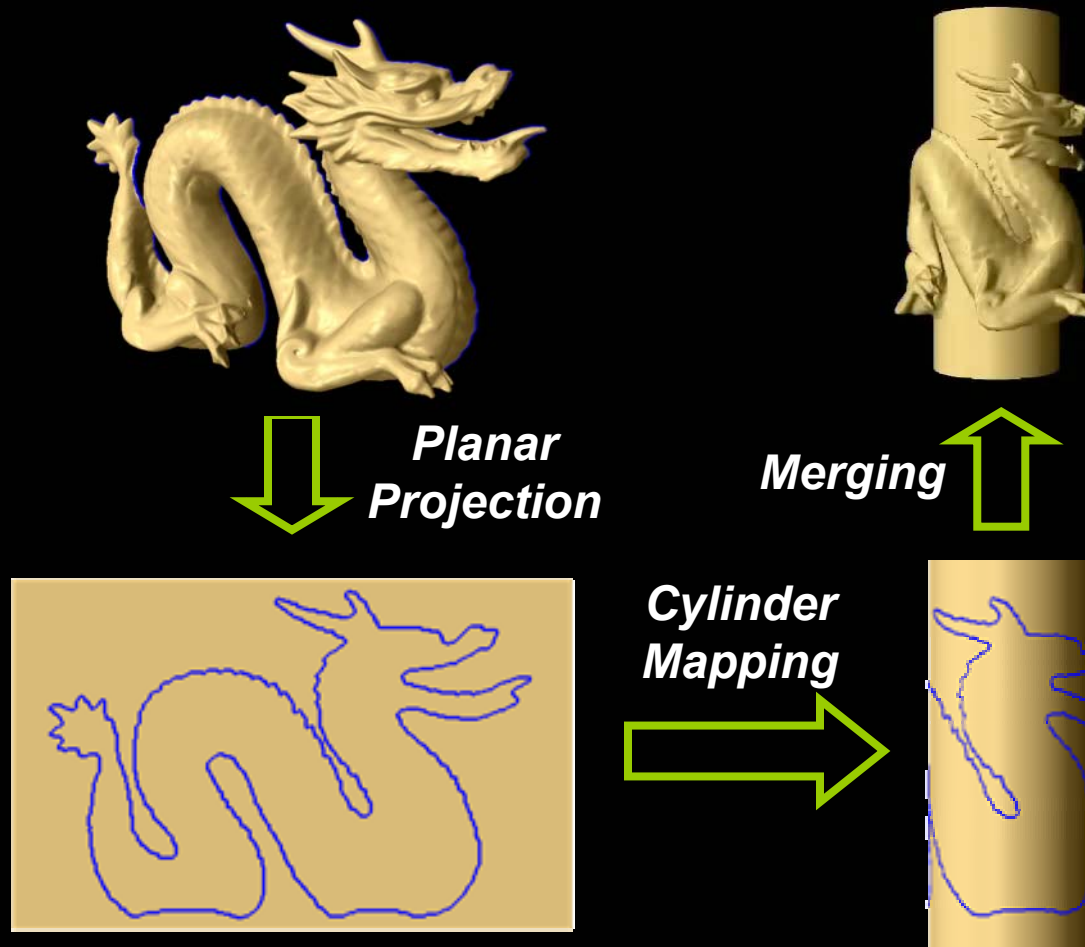
# Object Merging Using Sparse Correspondences



# Object Merging Using Projection



# Object Merging Using Mapping

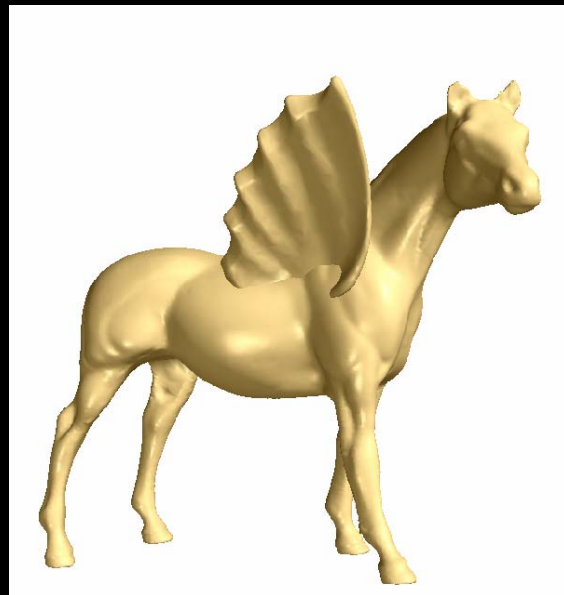


DRAGON: 18K triangles, CYLINDER: 60K triangles, Running Time: 5 seconds

# Object Merging Comparison



Poisson



Boolean

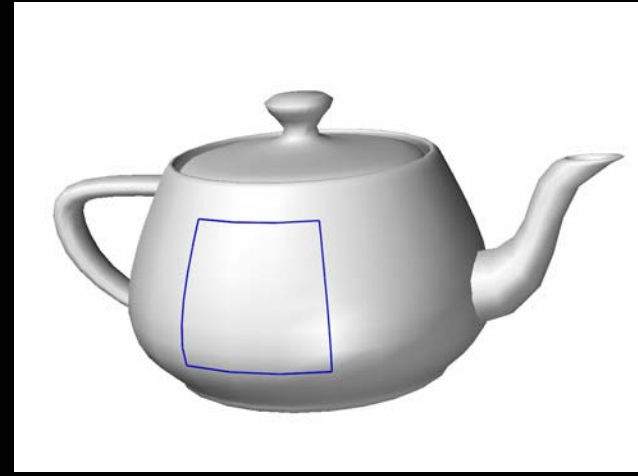
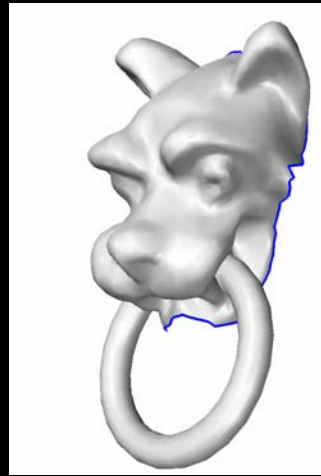


WIRE

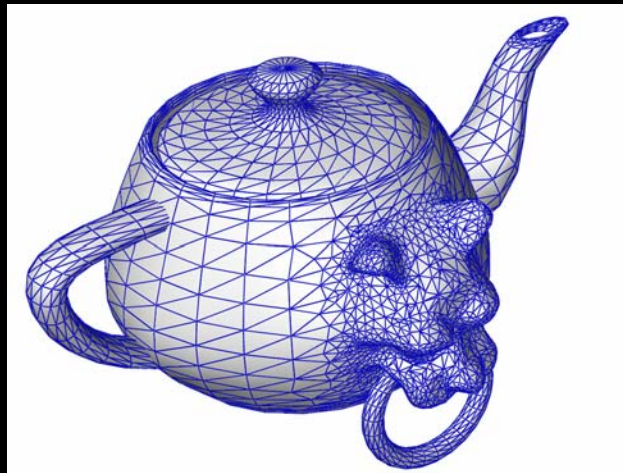
WING: 2000 triangles, HORSE: 100K triangles, Running Time: 400 ms

# Merging High Genus Meshes

Input

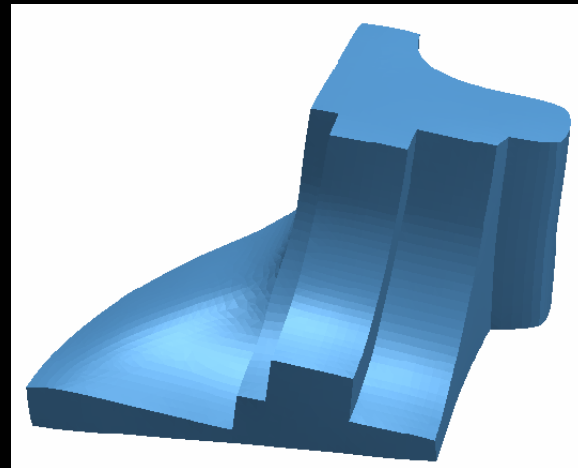
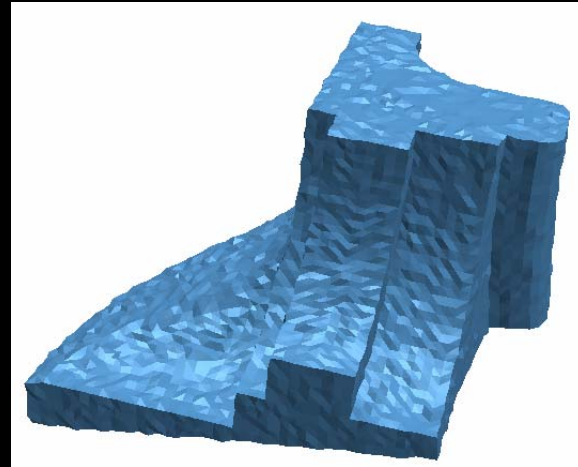


Output

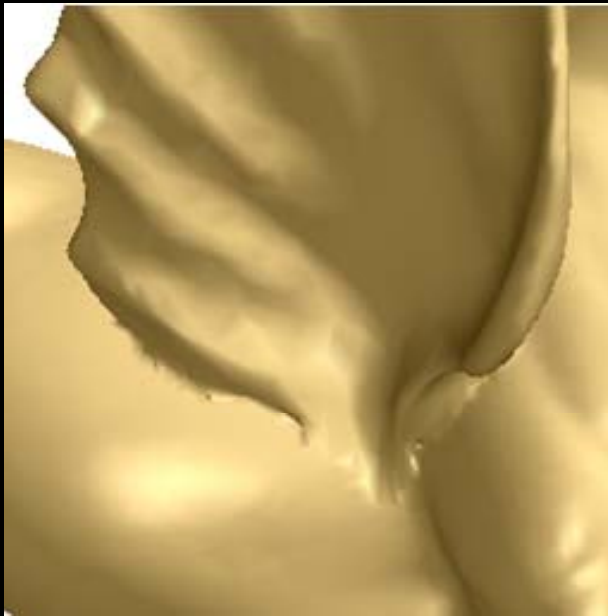


GARGOYLE: 4000 triangles, TEAPOT: 2000 triangles, Running Time: 890ms

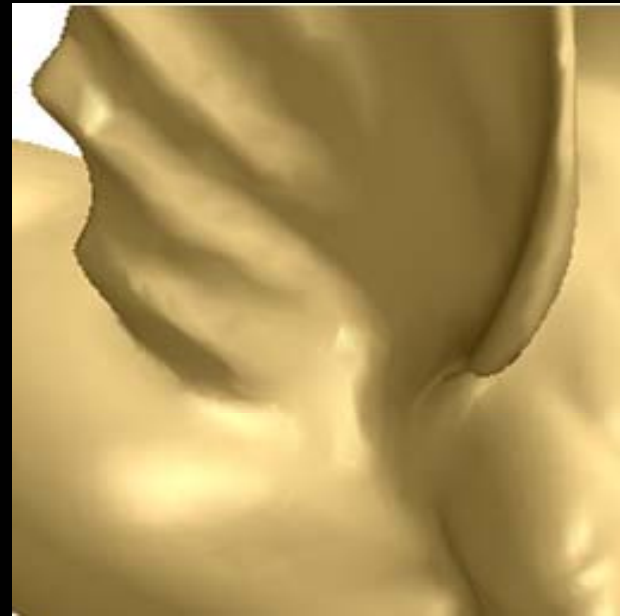
# Mesh Denoising by Feature-Preserving Normal Filtering



# Smoothing at the Merging Boundary



Before Smoothing



After Smoothing

# Contributions

- A Poisson-based mesh editing framework featuring local gradient manipulation and local transform propagation
- Techniques for applying this framework to mesh deformation, merging and smoothing
- Interactive tools for mesh editing
- Acceleration schemes for interactive deformation

# Acknowledgments

- Karan Singh, Univ. of Toronto
- Lin Shi, UIUC
- Yiyong Tong, USC
- Ran Zhou, Bo Zhang and Steve Lin, MSRA

*The End*