PolyCube

Xiao-Ming Fu

Outlines

- Definition
- Deformation-based method
  - All-Hex Mesh Generation via Volumetric PolyCube Deformation
- Voxel-based method
  - Optimizing PolyCube domain construction for hexahedral remeshing
- Cluster-based method
  - PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction
- Generalized PolyCube

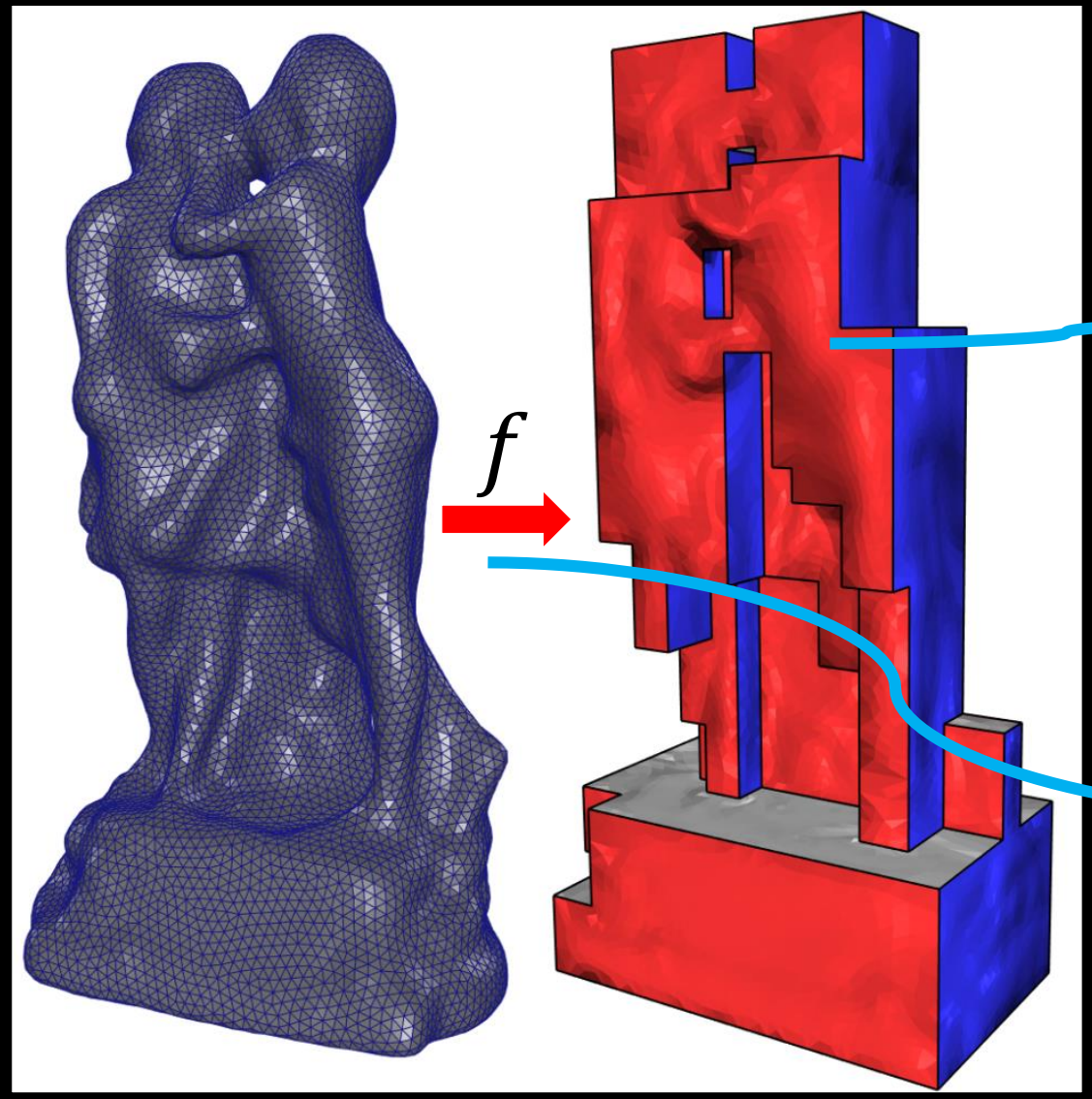
Outlines

- Definition

- Deformation-based method
  - All-Hex Mesh Generation via Volumetric PolyCube Deformation
- Voxel-based method
  - Optimizing PolyCube domain construction for hexahedral remeshing
- Cluster-based method
  - PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction

- Generalized PolyCube

PolyCube



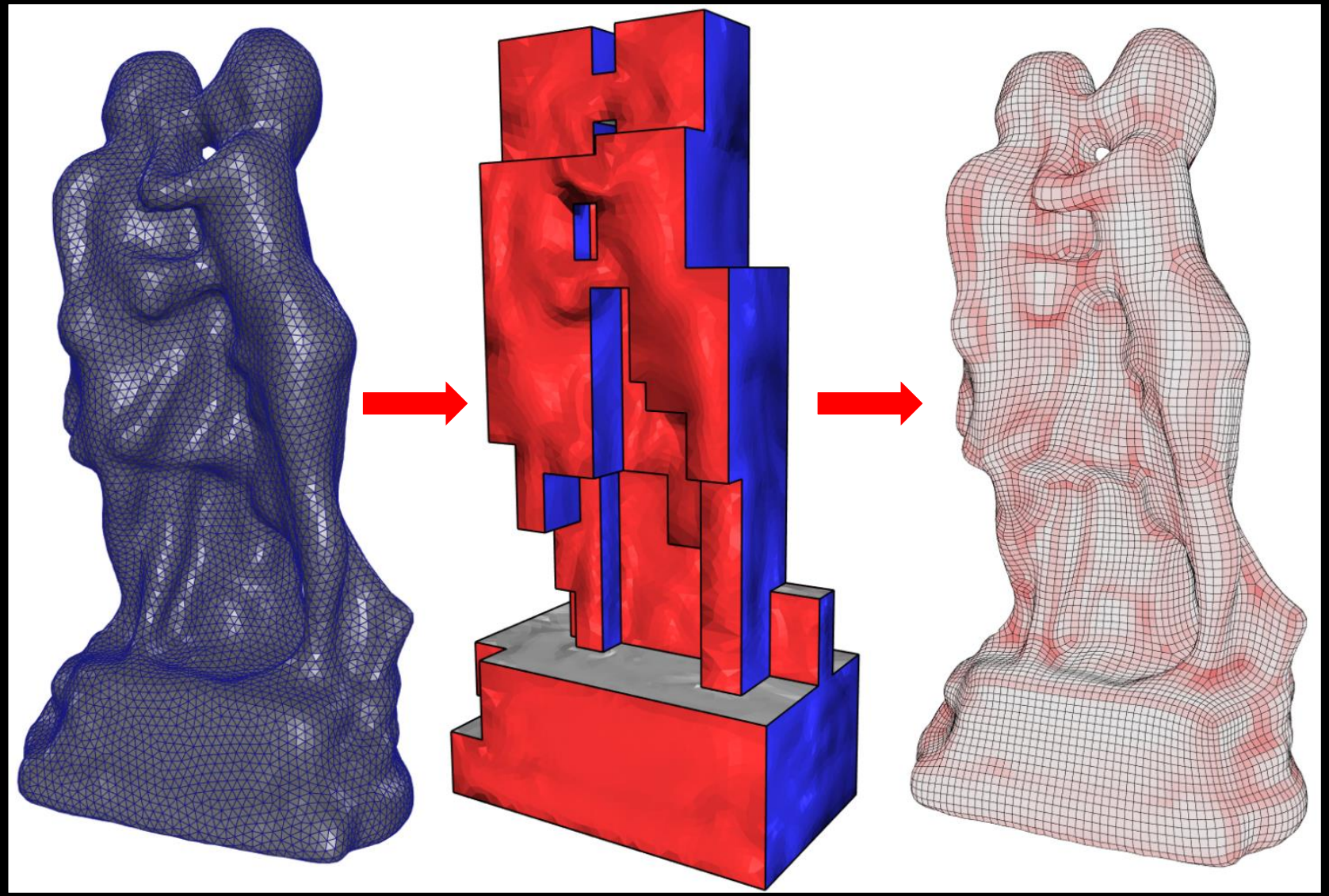
Tetrahedral Mesh PolyCube

PolyCube:
1. Compact representations for closed complex shapes
2. Boundary normal aligns to the axes.
3. Axes: $(\pm 1,0,0)^T$, $(0,\pm 1,0)^T$, $(0,0,\pm 1)^T$.

PolyCube-Map f:
1. A mesh-based map.
2. Foldover-free and low distortion.

Application – All-hex meshing

Tetrahedral Mesh PolyCube All-Hex Mesh



Applications based on PolyCube:
1. All-Hex Mesh generation.
2. Texture Mapping [Tarini et al. 2004].
3. GPU-based subdivision [Xia et al. 2011].
……

Application – Seamless texture mapping

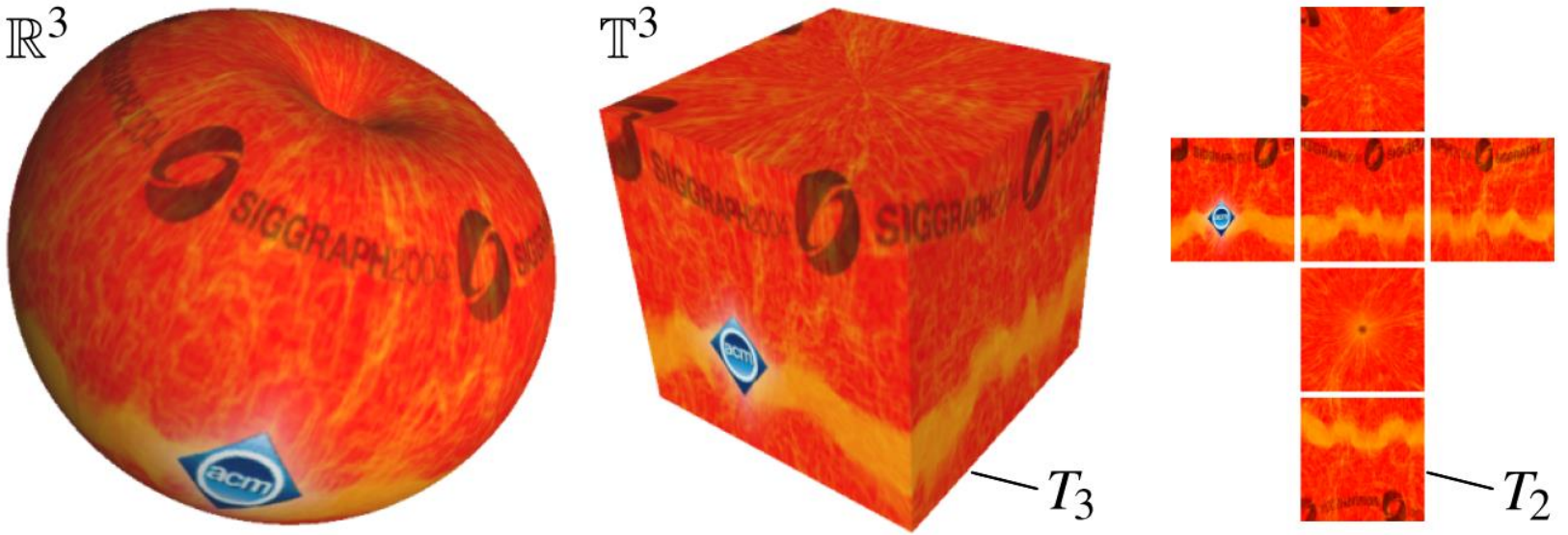


Figure 1: Cube maps can be used to seamlessly texture map an apple (left). In this case, the 3D texture domain T_3 is the surface of a single cube that is immersed in the 3D texture space $\mathbb{T}^3$ (middle) and corresponds to a 2D texture domain T_2 that consists of six square images (right).

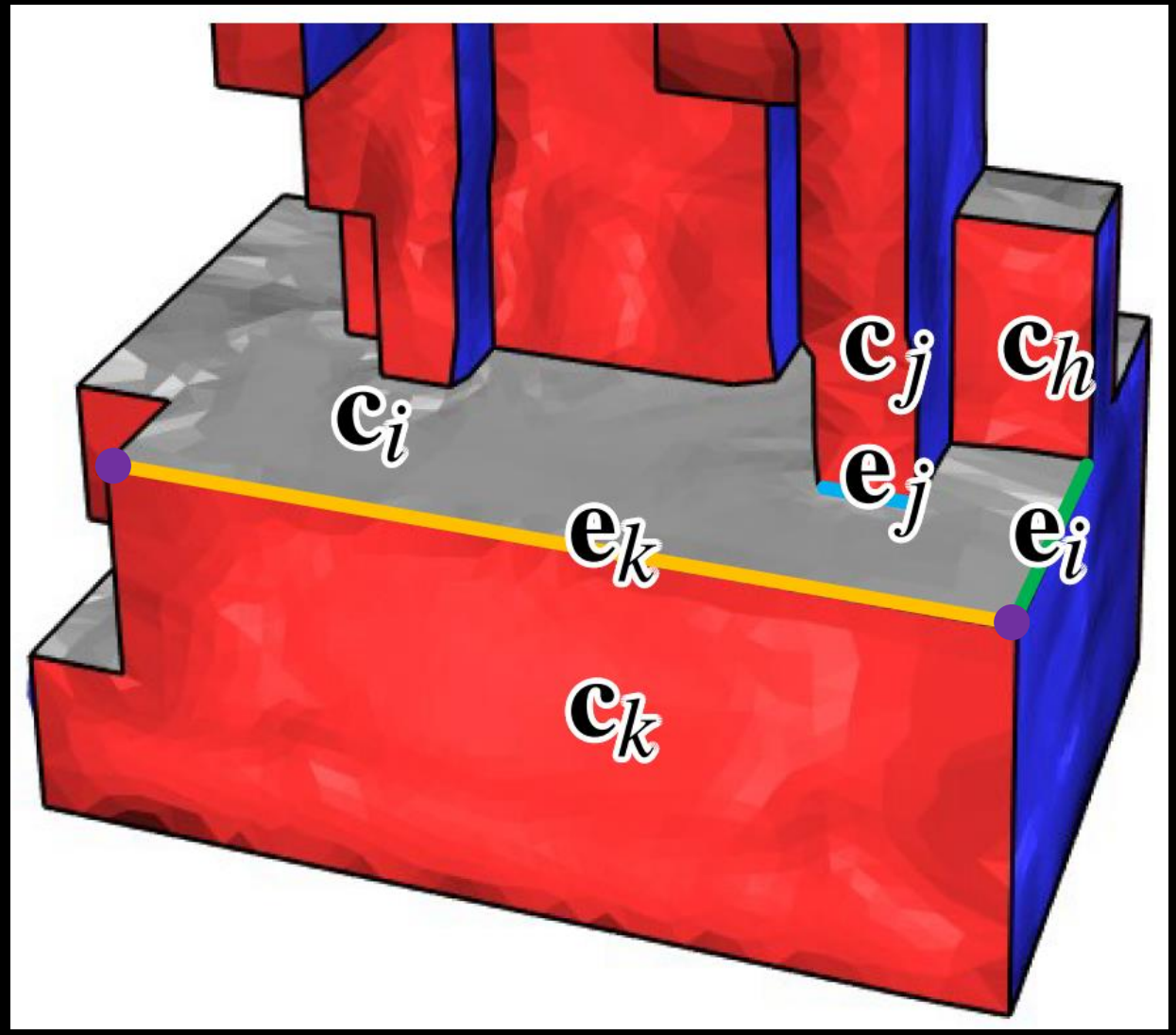
Applications based on PolyCube:
1. All-Hex Mesh generation.
2. Texture Mapping [Tarini et al. 2004].
3. GPU-based subdivision [Xia et al. 2011].
……

PolyCube facet, edge, and vertex

PolyCube ***facet***:
share the same label

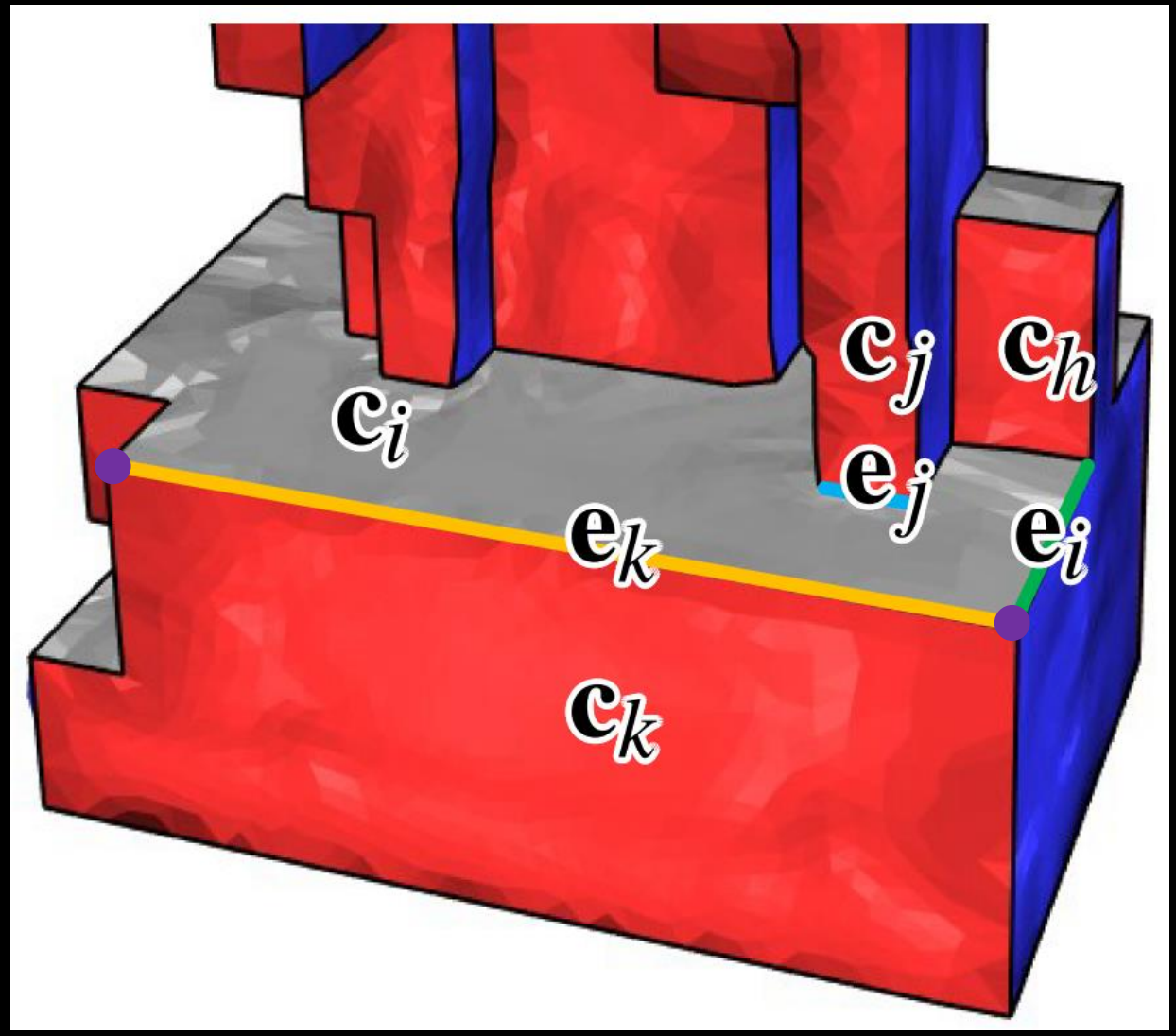
PolyCube ***edge***:
The edges between facets

PolyCube ***vertex***:
sharing by at least three charts



Sufficient topological conditions

- Any PolyCube facet should have at least four neighboring Poly-Cube facets.
- Any two neighboring PolyCube facets should not have opposite labels such as $+X$ and $-X$.
- The valence of each PolyCube vertex is three.



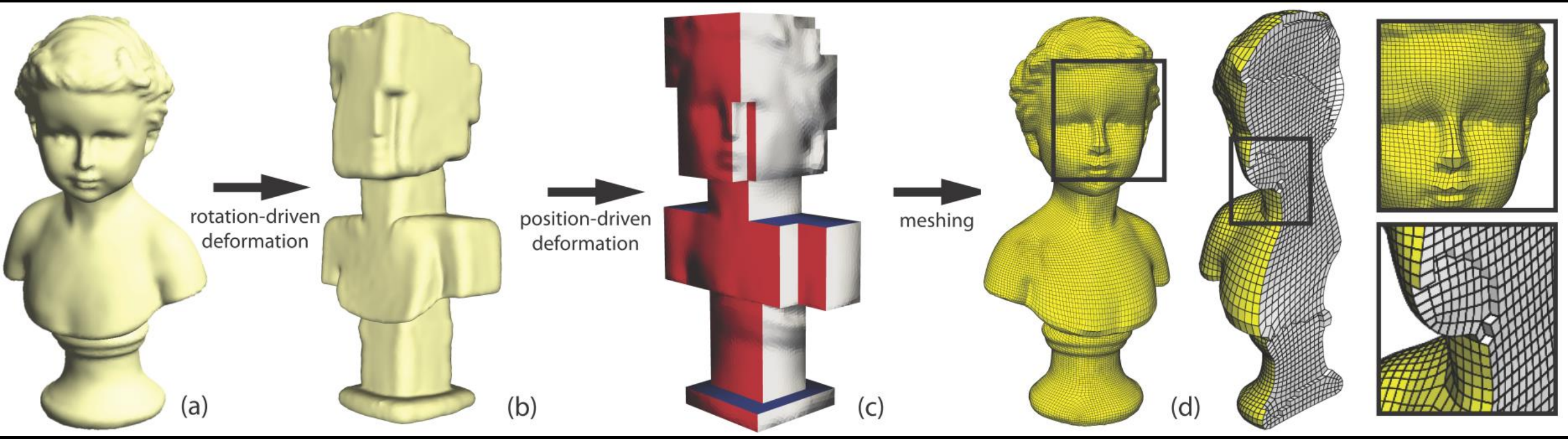
Outlines

- Definition

- Deformation-based method
  - All-Hex Mesh Generation via Volumetric PolyCube Deformation
- Voxel-based method
  - Optimizing PolyCube domain construction for hexahedral remeshing
- Cluster-based method
  - PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction

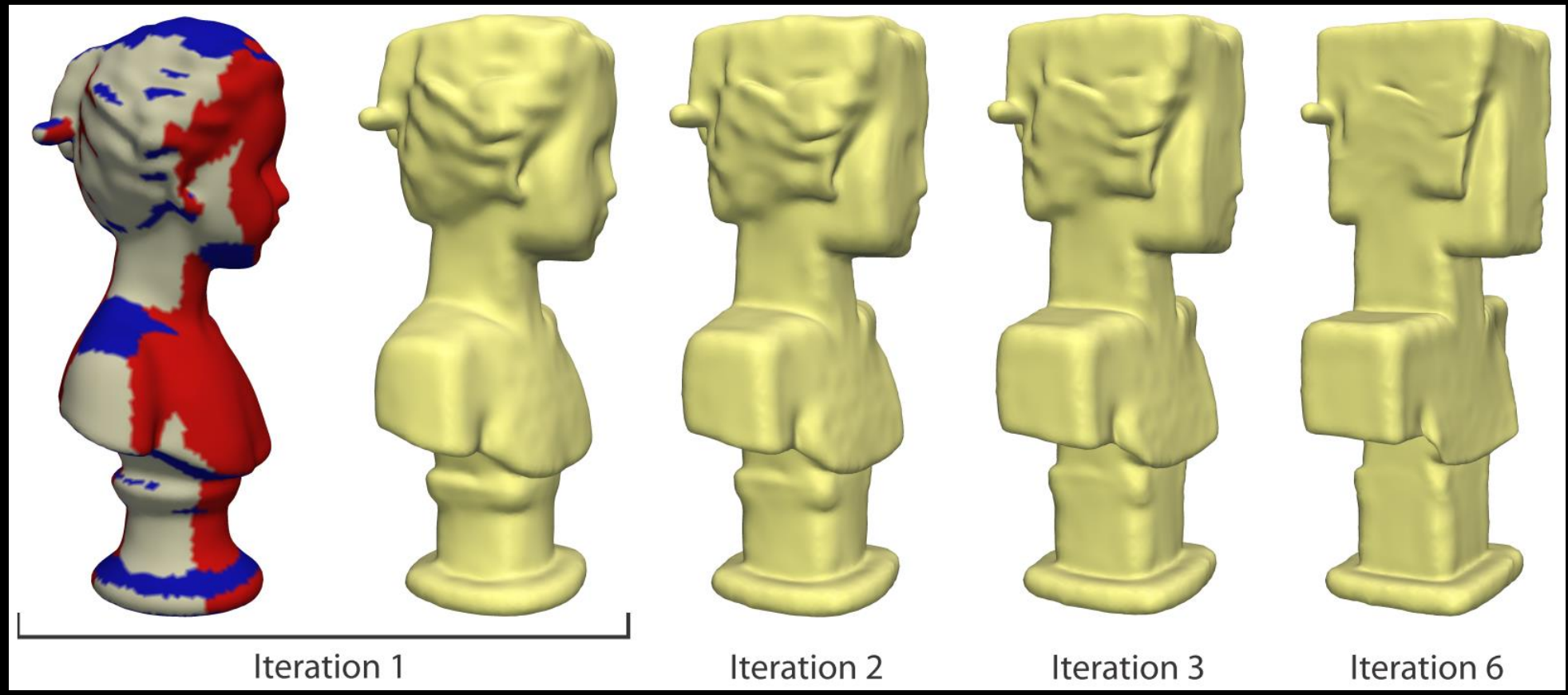
- Generalized PolyCube

Pipeline


rotation-driven
deformation
position-driven
deformation
meshing
(a)
(b)
(c)
(d)

Rotation-driven deformation

- Goal: gradually aligns the model's surface normals with one of the six global axes, preserving shape as much as possible.



Rotation-driven deformation

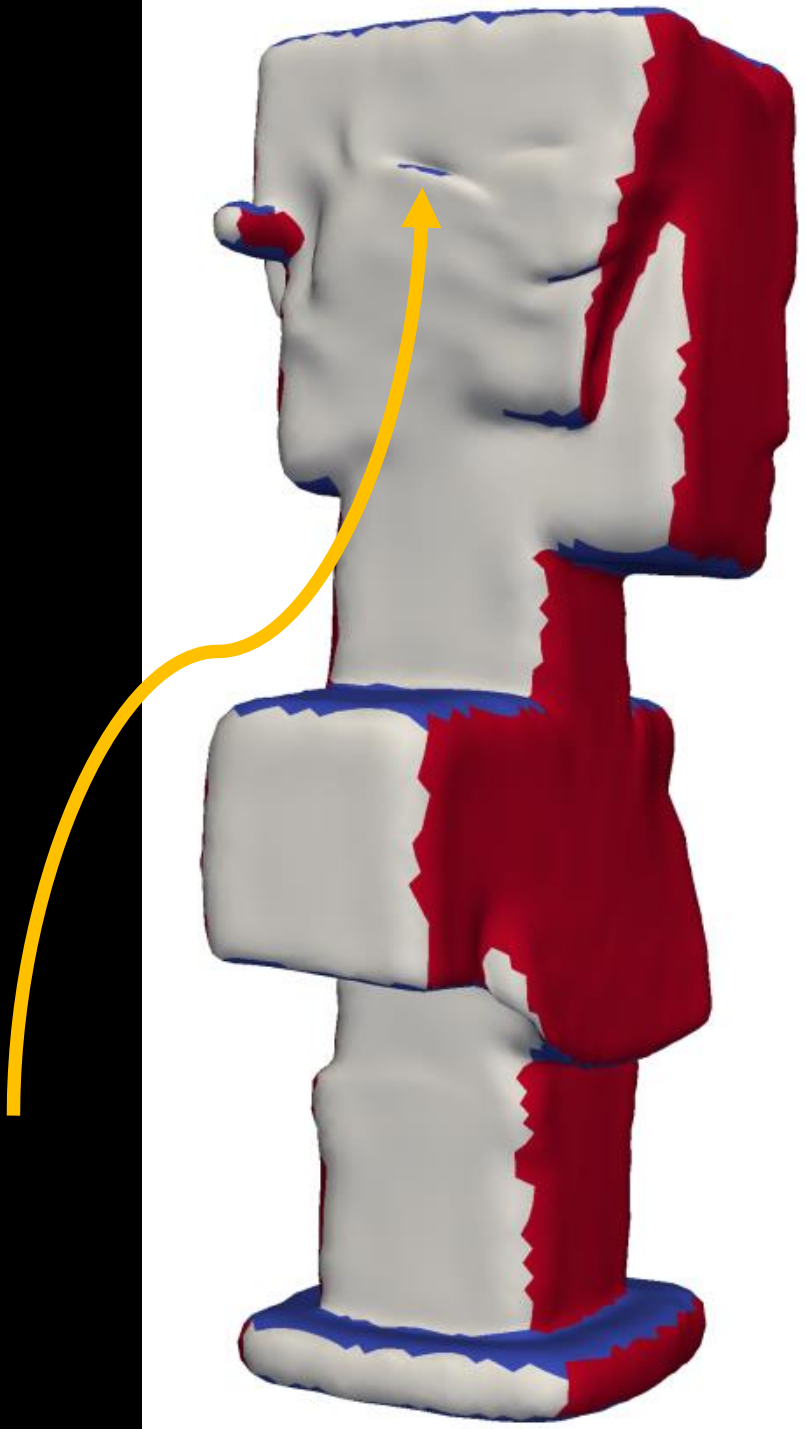
- As-Rigid-As-Possible deformation $E = \sum_{i=1}^{N_v} w_i \sum_{j \in \Omega(i)} w_{ij} \left\| \left(p_i' - p_j'\right) - R_i\left(p_i - p_j\right) \right\|^2$
  - No local step
  - Rotations are determined by axis-alignment constraints.

- Steps:
  - For every surface vertex (except those on sharp features), the minimal rotation necessary to align each surface vertex normal with one of $\pm X, \pm Y, \pm Z$.
    - quaternion
  - Smoothly propagate to feature and interior vertices.
    - Laplace equation per quaternion component
  - Solve E.
    - Least squares.

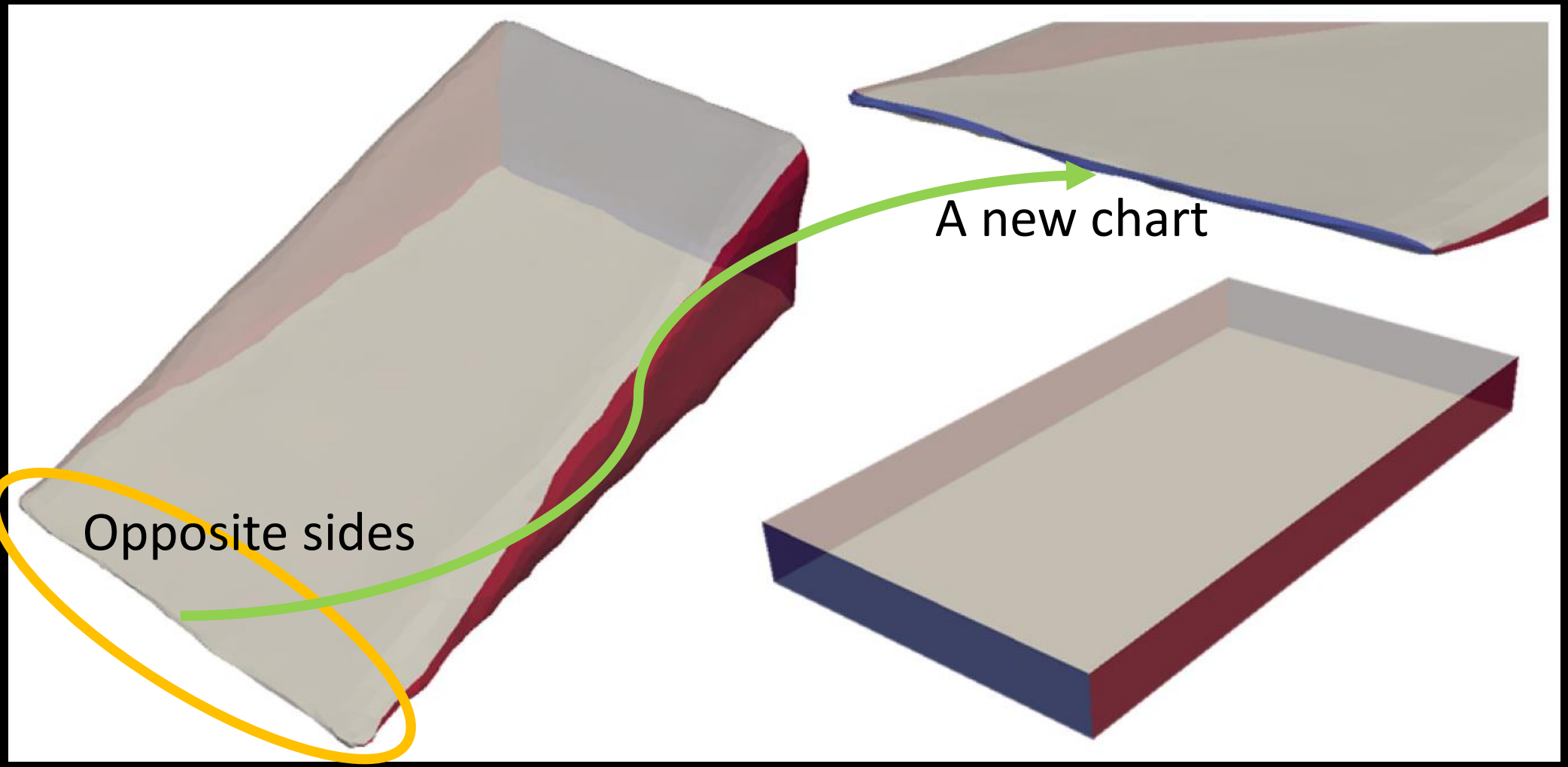
Labeling

- 1. Label surface triangles according to the closest axis
- 2. Group similarly labeled triangles into charts.
- 3. Straighten chart boundaries.

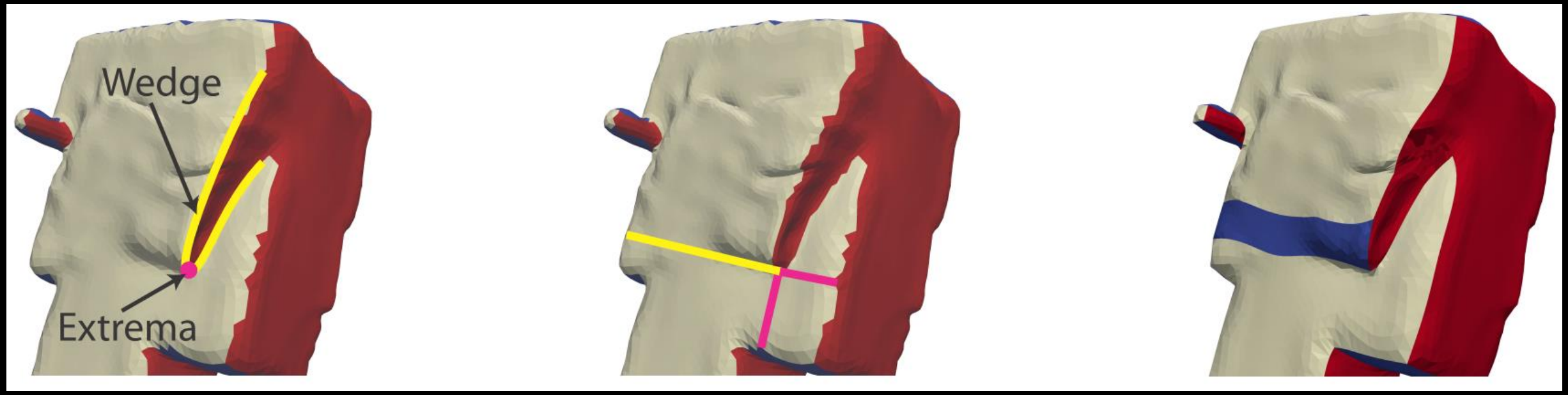
- 4. remove small, spurious charts bounded by at most two edges



Multi-orientation chart



Highly non-planar chart



Detect extrema along the chart boundary

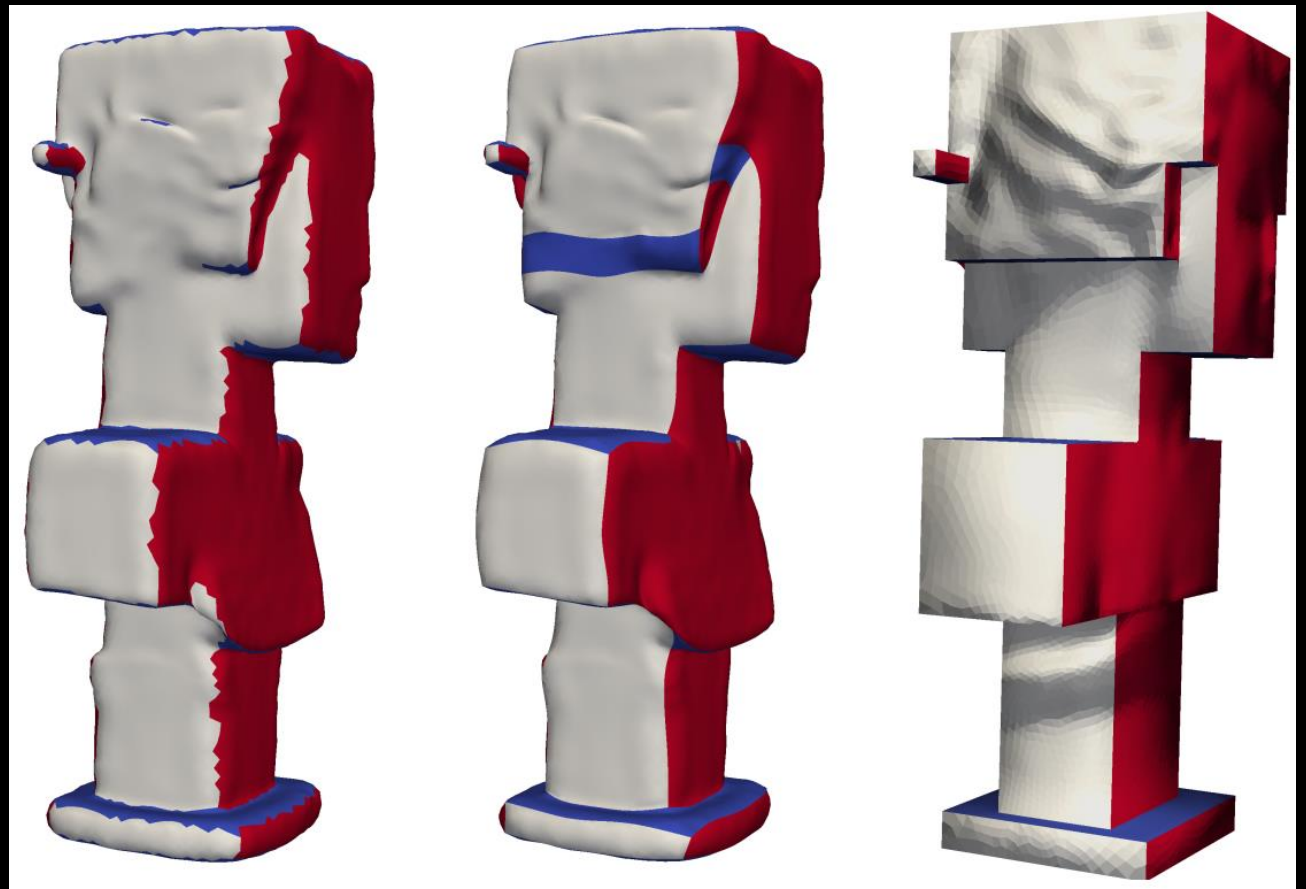
Three possible axis-aligned cut options.

Valid cuts are defined as those that would not introduce new charts with three or fewer neighbors.

Position-driven deformation

- constrain each chart to an axis-aligned plane.
  - the chart coordinate

$$E = \sum_{i=1}^{N_v} w_i \sum_{j \in \Omega(i)} w_{ij} \left\| \left(p_i' - p_j'\right) - R_i\left(p_i - p_j\right) \right\|^2$$



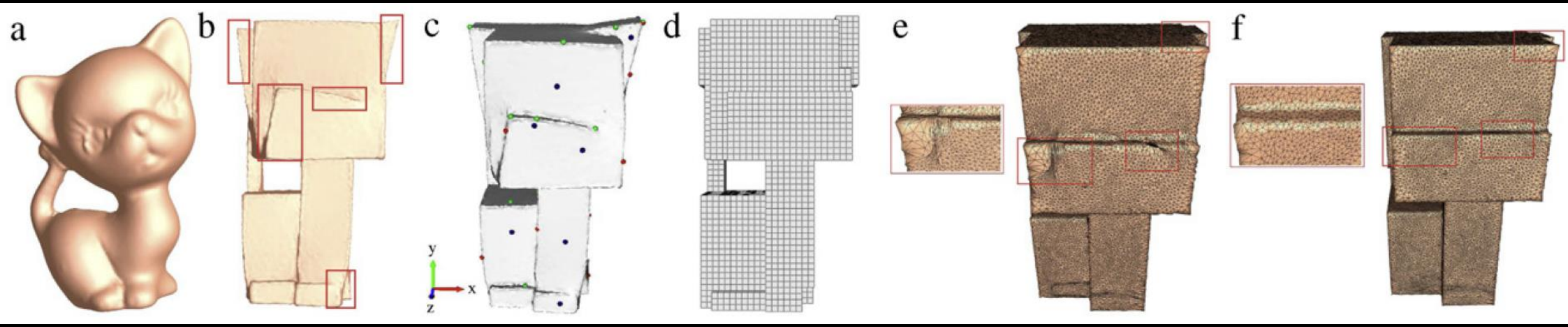
More papers:

- L_1 -based Construction of Polycube Maps from Complex Shapes (2014)
- Efficient Volumetric PolyCube-Map Construction (2016)

Outlines

- Definition
- Deformation-based method
  - All-Hex Mesh Generation via Volumetric PolyCube Deformation
- Voxel-based method
  - Optimizing PolyCube domain construction for hexahedral remeshing
- Cluster-based method
  - PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction
- Generalized PolyCube

Pipeline

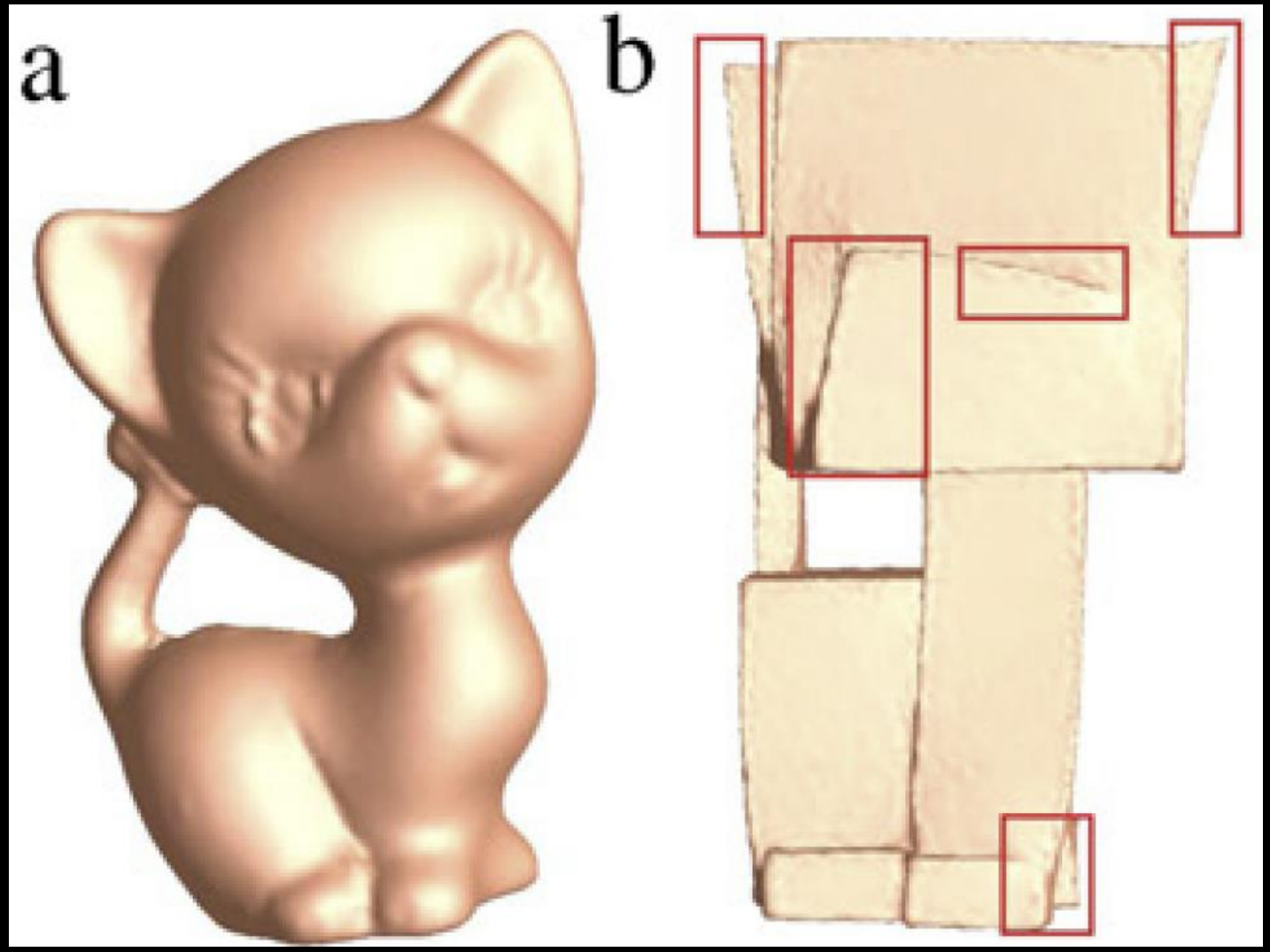


1. Pre-deformation

2. PolyCube construction and optimization

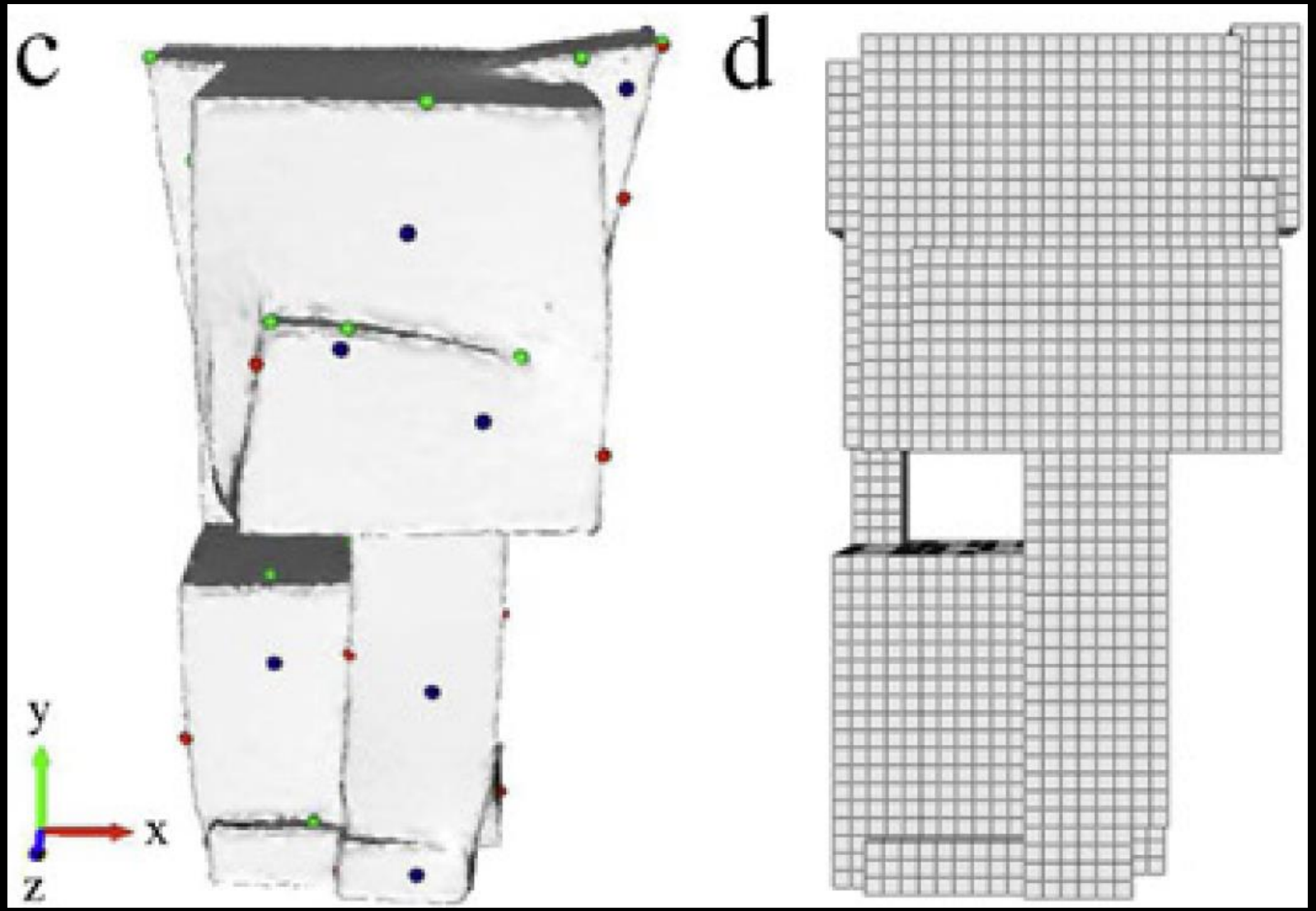
3. Mapping computation

Wedge regions



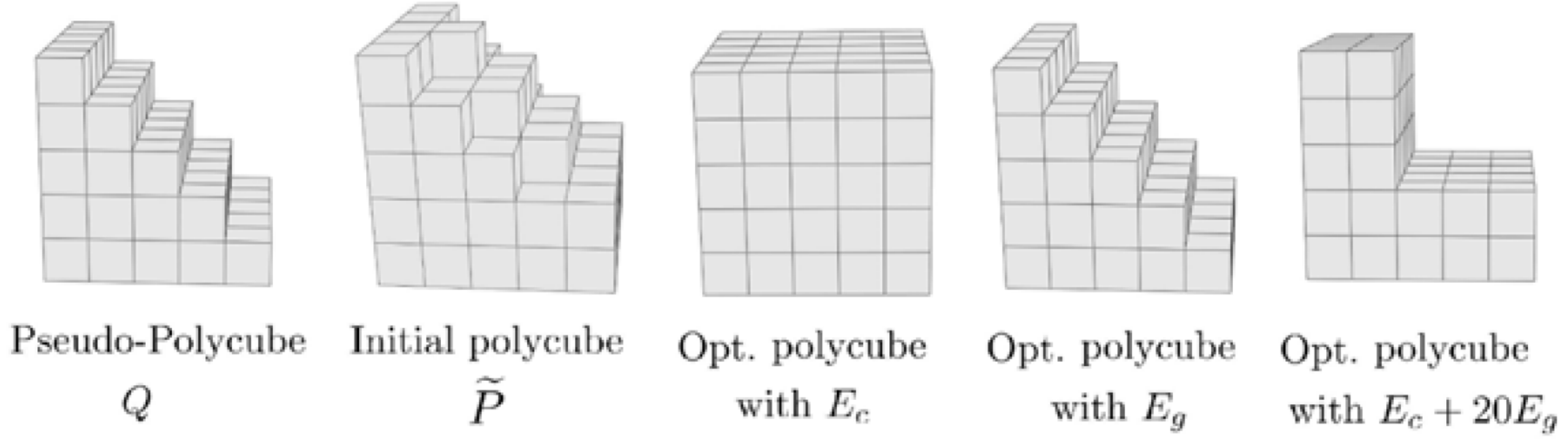
Wedge regions are hard to avoid

Voxelization



Length of cube

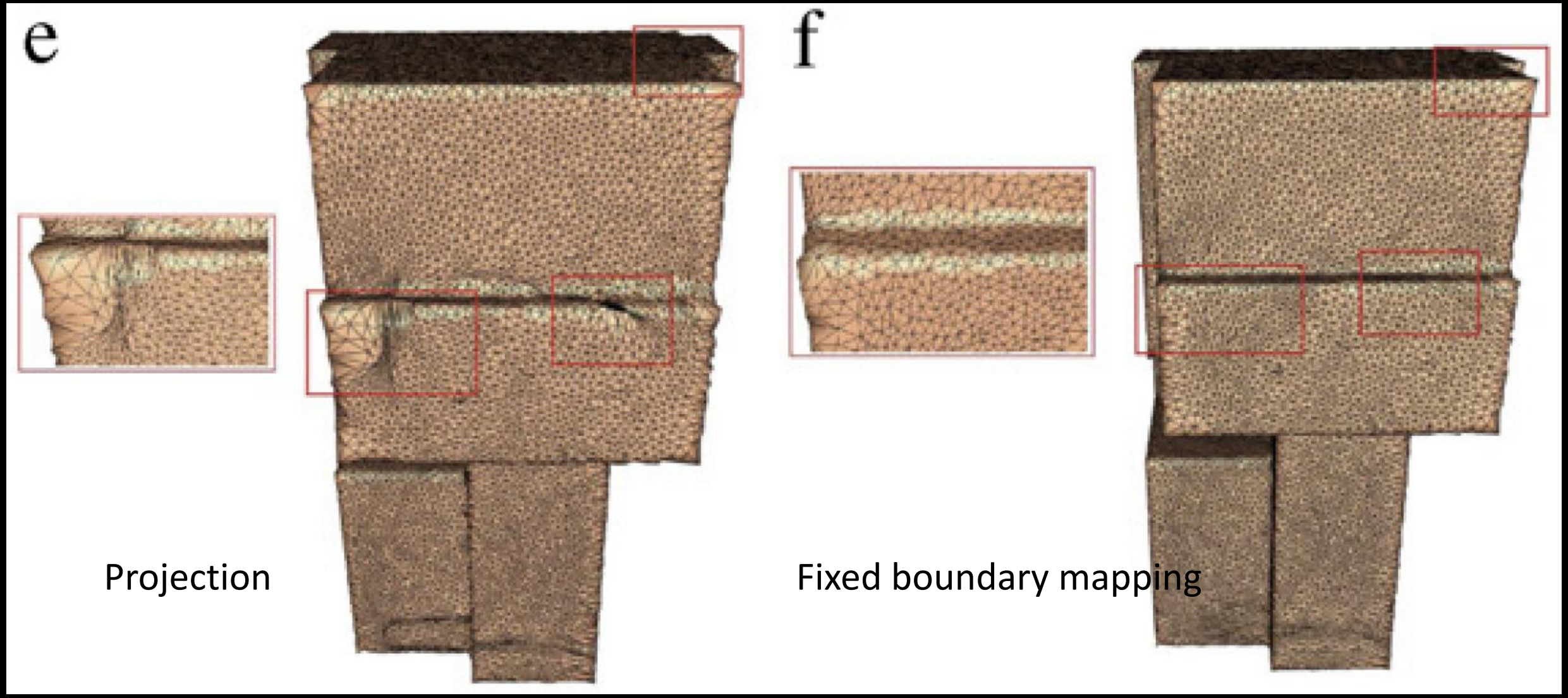
Optimization



Domain simplicity E_c + α Geometric deviation E_g

Morphological operations

PolyCube volumetric parameterization



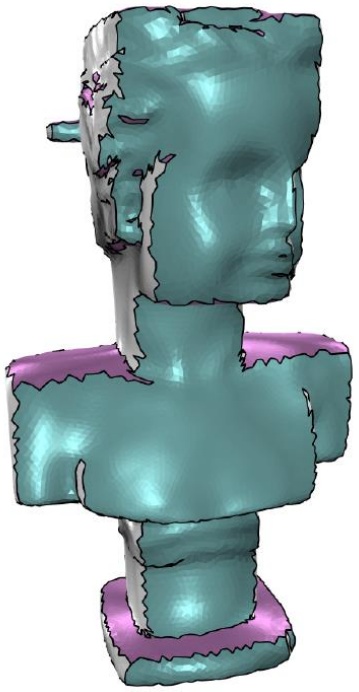
More papers based on construction

- Computing Surface PolyCube-Maps by Constrained Voxelization (PG 2019)

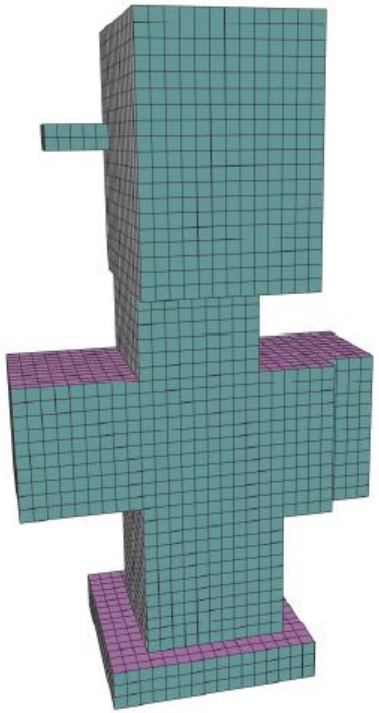
Input:
A source mesh &
a pre-axis-aligned shape

Algorithm workflow

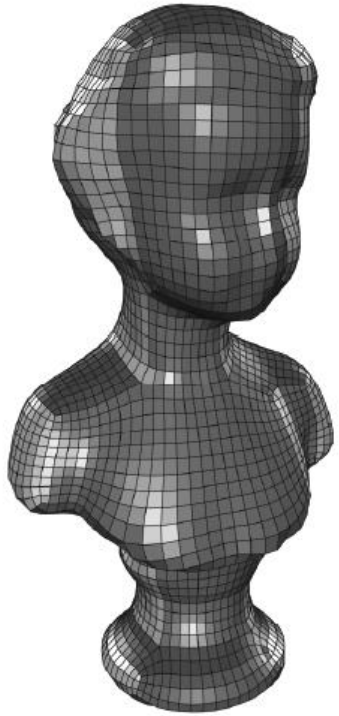
Output:
PolyCube-map



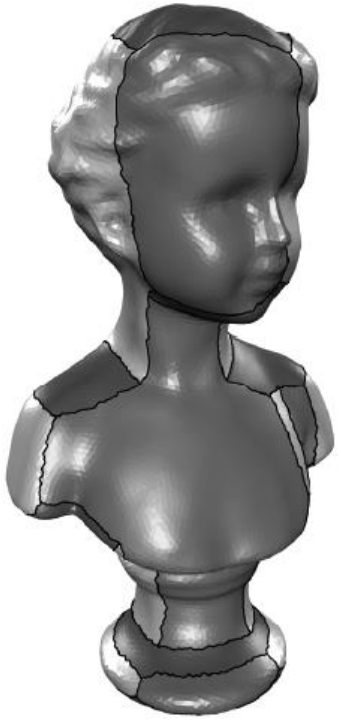
Pre-axis-aligned shape A



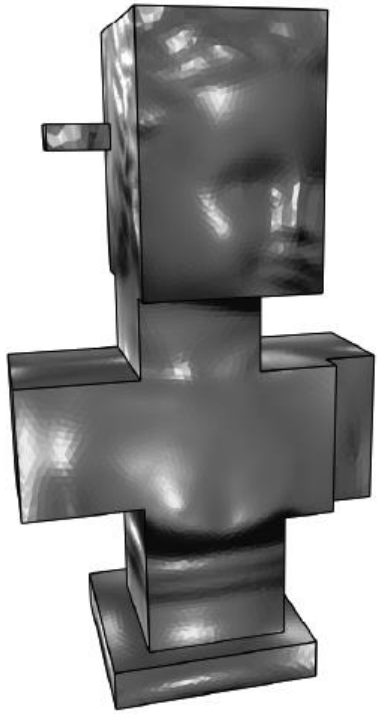
Constructed PolyCube C



Optimized quad mesh Q



Segmentation S



PolyCube-map f

I. Constrained voxelization

II. Computing surface PolyCube-Map

Constrained voxelization: Formulation

$$\min_{\mathcal{C}} \; N(\mathcal{C})$$

The number of corners of the PolyCube

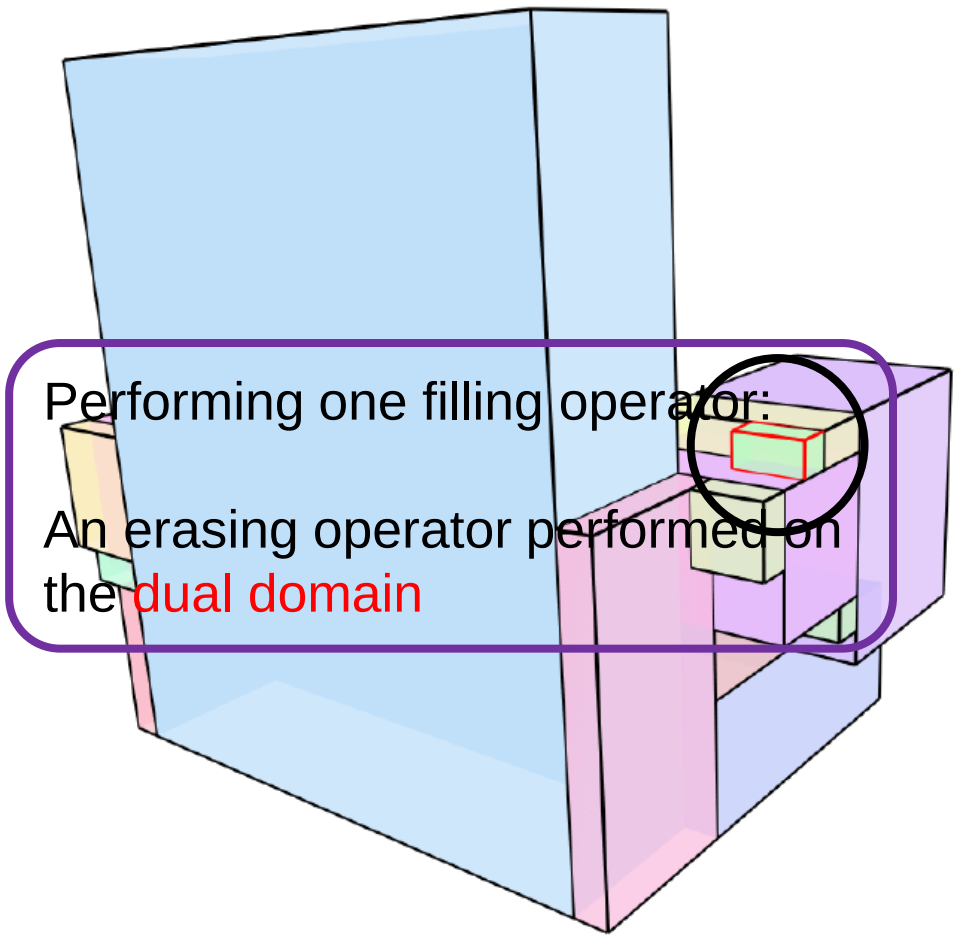
$$\text{s.t.} \quad d_h(\mathcal{C}, \mathcal{A}) \leq K_h,$$

The error-bounded constraint

$$\mathcal{C} \in \mathcal{T}$$

The topological constraint

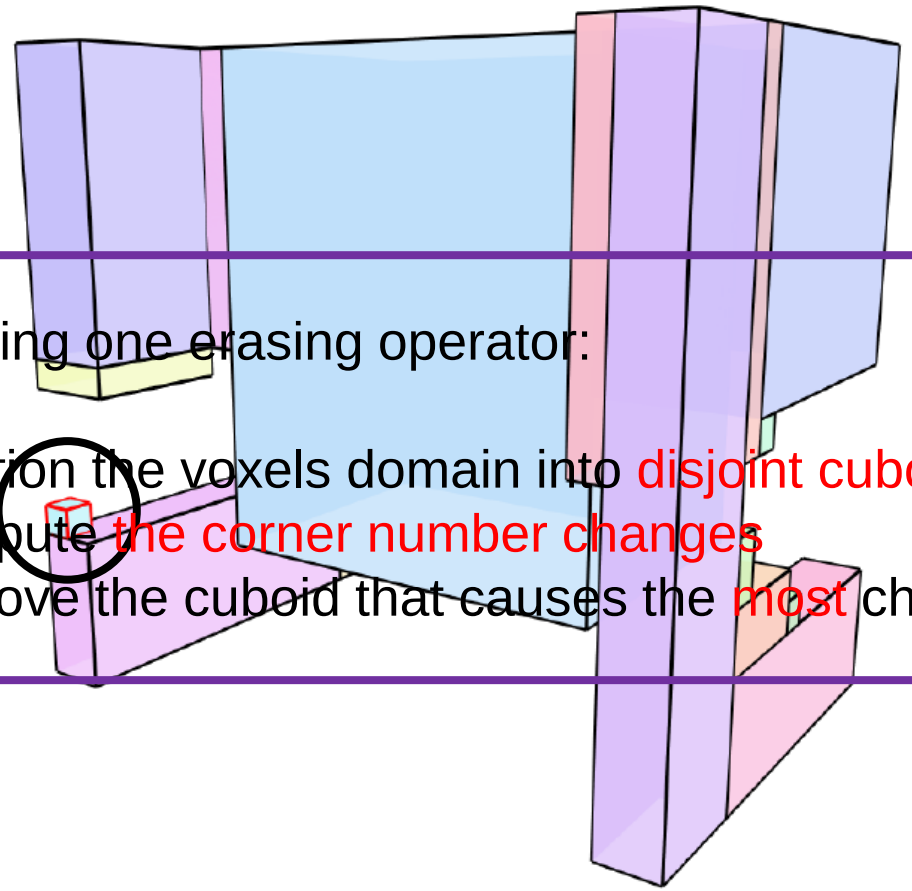
Two operators



Performing one filling operator:

An erasing operator performed on the dual domain

Erasing operator

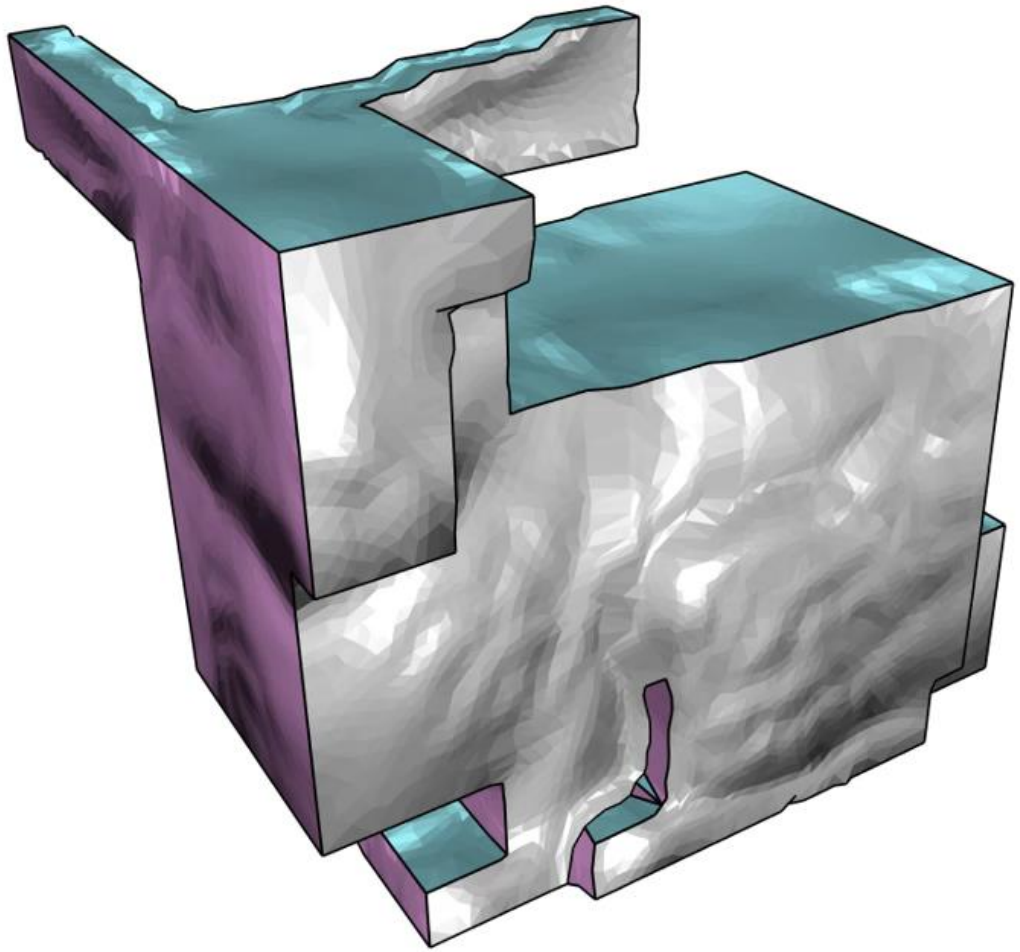


Performing one erasing operator:

1. Partition the voxels domain into disjoint cuboids
2. Compute the corner number changes
3. Remove the cuboid that causes the most changes

Filling operator

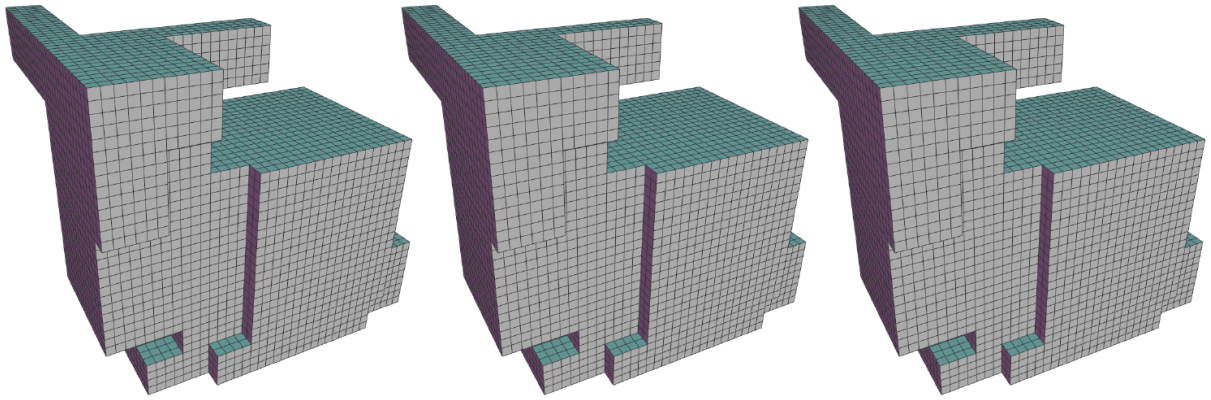
Erasing-and-filling Strategy



Input

1. Generate the initialization, k = 1
2. Perform one erasing operator without violating the constraints
3. Perform one filling operator without violating the constraints
4. k = k + 1, go to step 2

Erasing-and-filling Strategy



Order I Order II Order III

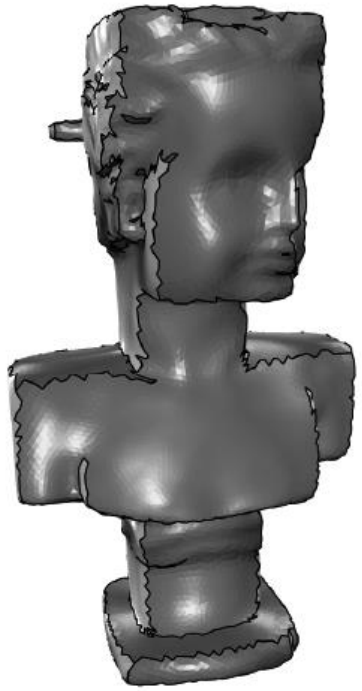
$N(C) = 72$ $N(C) = 72$ $N(C) = 72$

Our developed erasing and filling operators are effective and robust

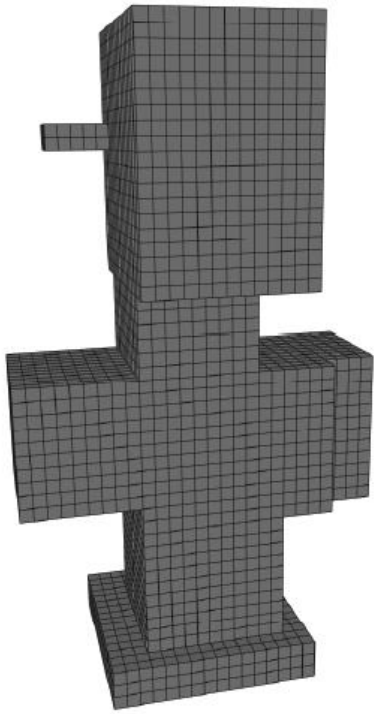
Input:
A source mesh &
a pre-axis-aligned shape

Algorithm workflow

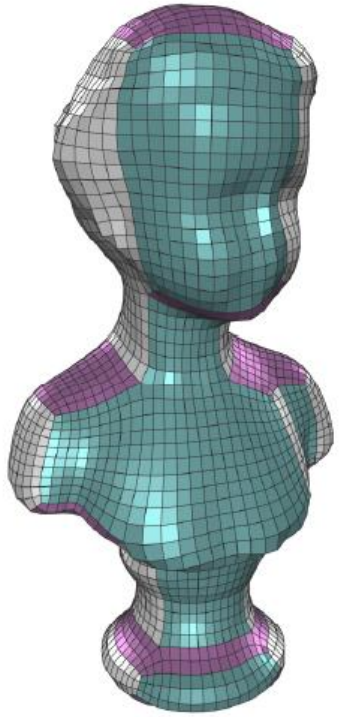
Output:
PolyCube-map



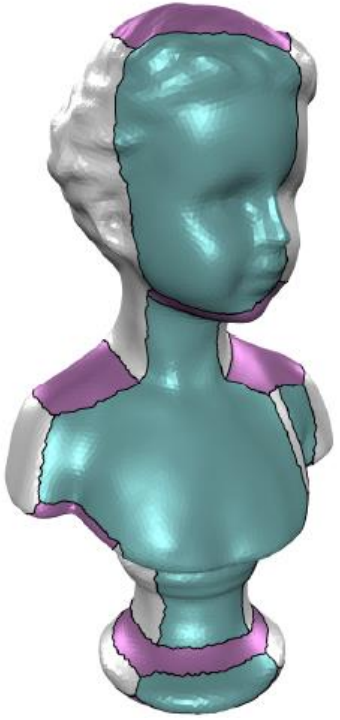
Pre-axis-aligned shape A



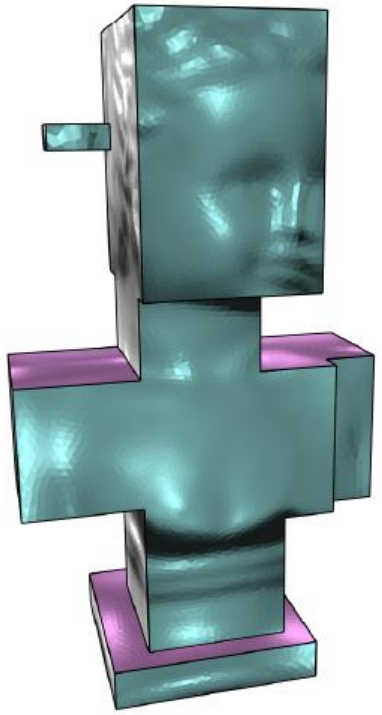
Constructed PolyCube C



Optimized quad mesh Q



Segmentation S



PolyCube-map f

I. Constrained voxelization

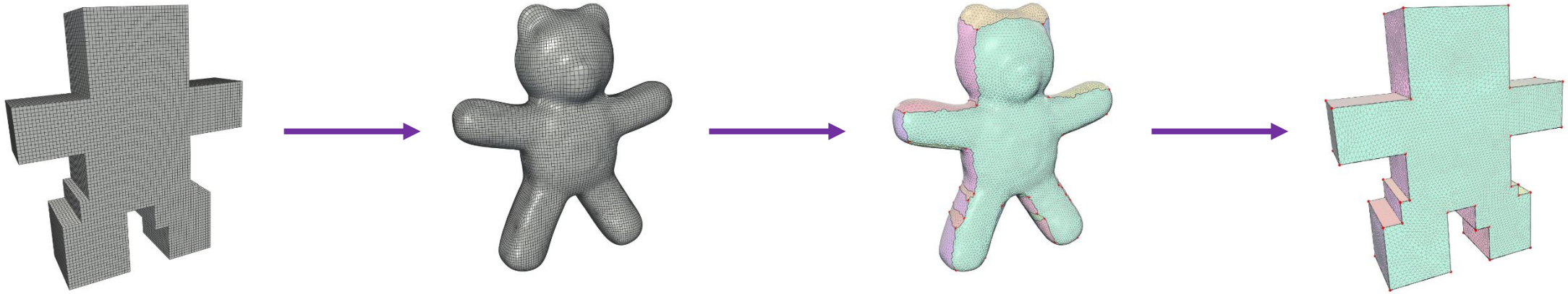
II. Computing surface PolyCube-Map

PolyCube-Maps computation

1. Optimize a new quad mesh

2. Segment the triangular surface into a set of submeshes

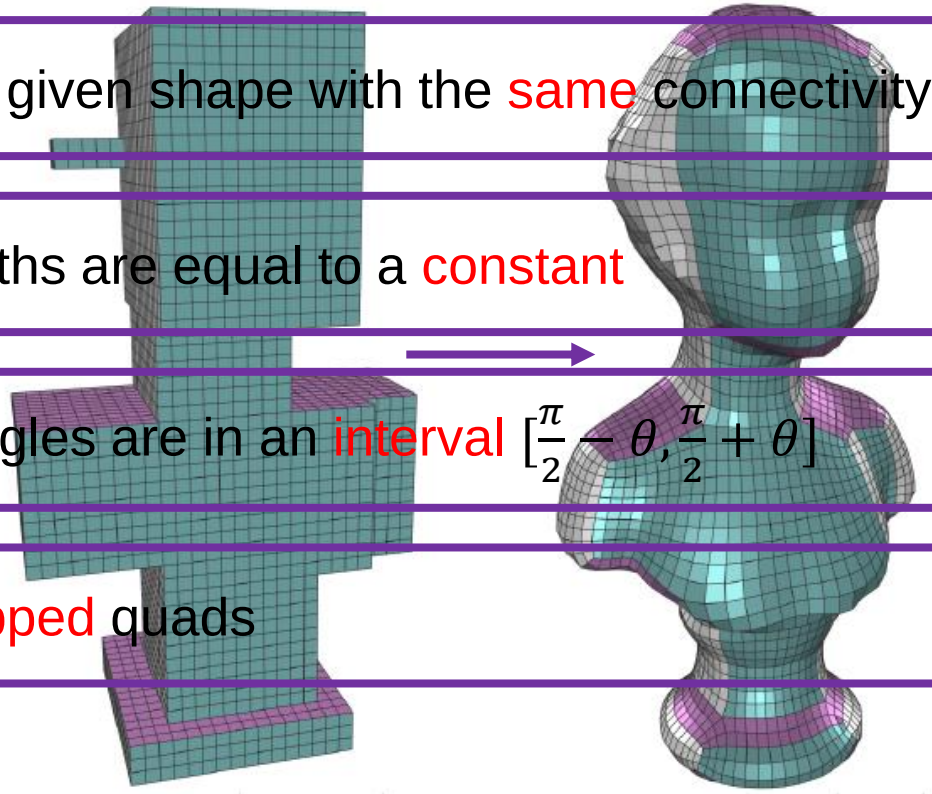
3. Map each submesh onto one PolyCube chart



Quad mesh optimization

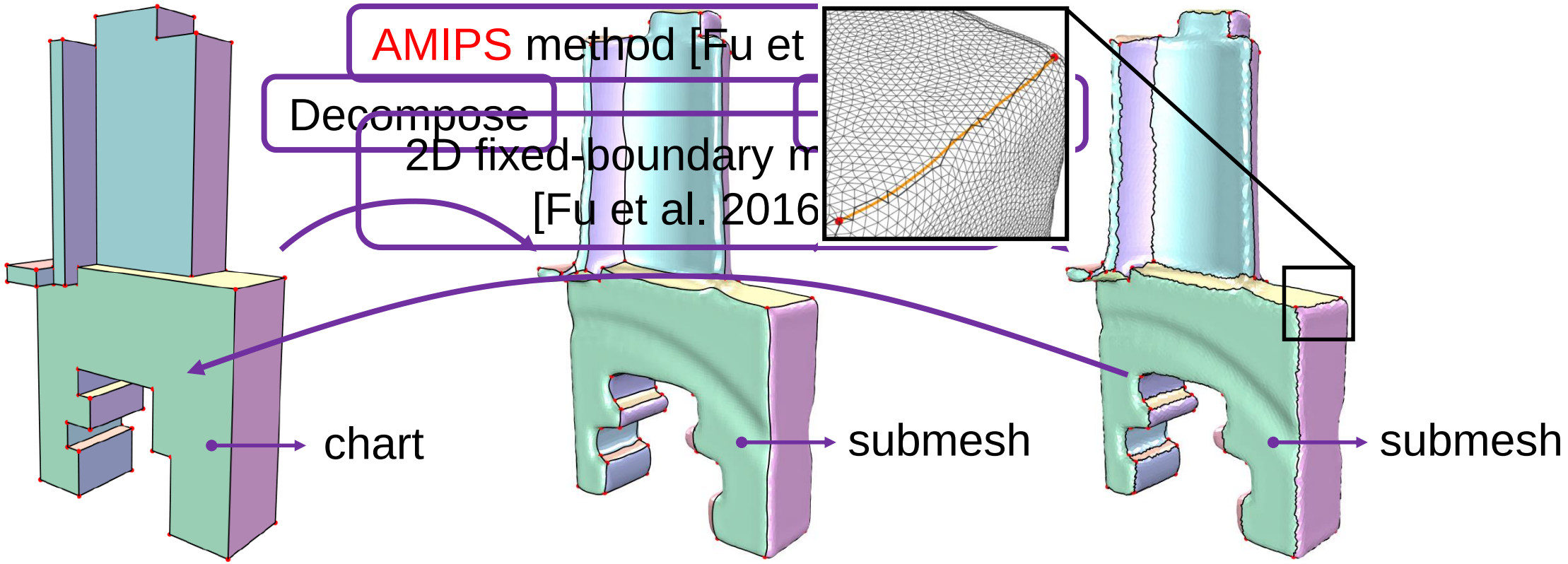
Requirements

1. Preserve the given shape with the same connectivity of the Polycube
2. All edge lengths are equal to a constant
3. All interior angles are in an interval $[\frac{\pi}{2} - \theta, \frac{\pi}{2} + \theta]$
4. Almost no flipped quads



Anderson acceleration [Peng et al. 2018]

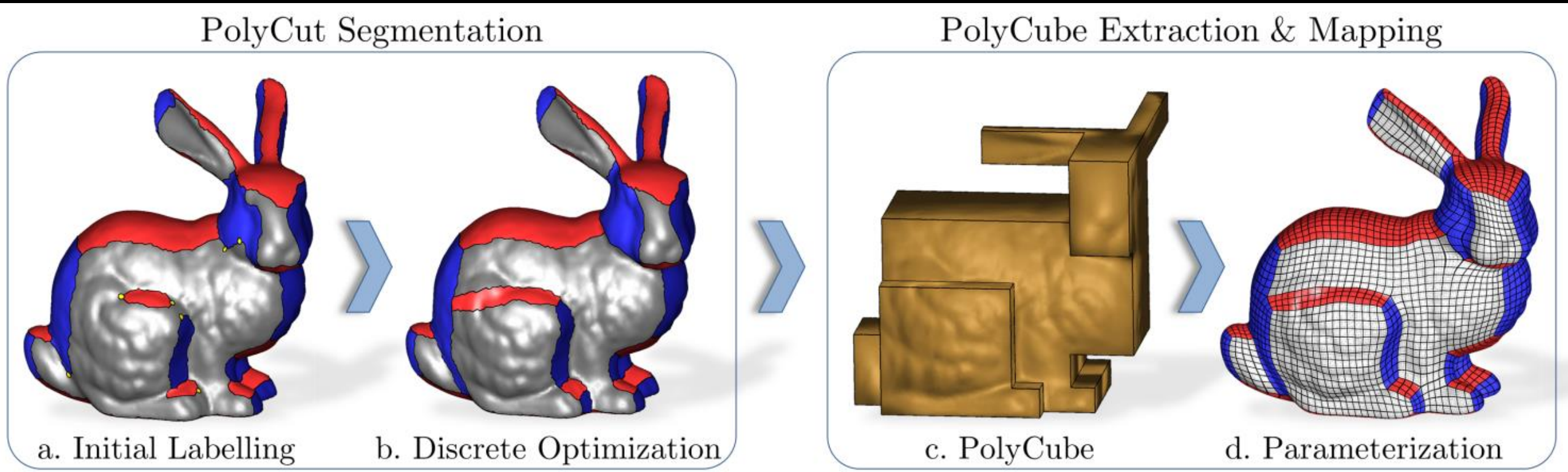
Segmentation and map computation



Outlines

- Definition
- Deformation-based method
  - All-Hex Mesh Generation via Volumetric PolyCube Deformation
- Voxel-based method
  - Optimizing PolyCube domain construction for hexahedral remeshing
- Cluster-based method
  - PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction
- Generalized PolyCube

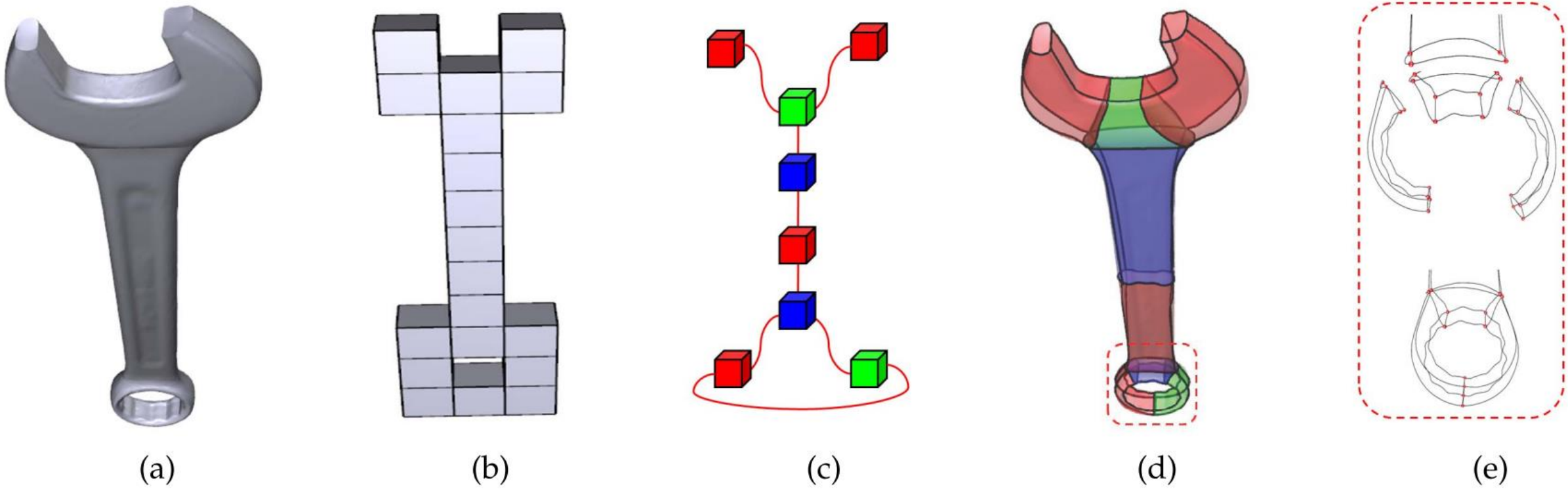
Pipeline


PolyCut Segmentation
PolyCube Extraction & Mapping
a. Initial Labelling
b. Discrete Optimization
c. PolyCube
d. Parameterization

Outlines

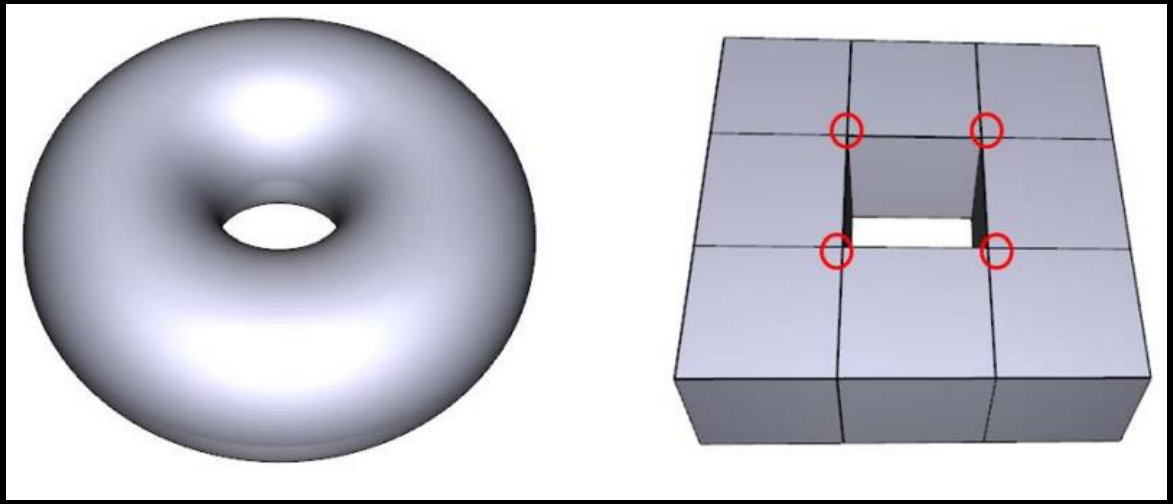
- Definition
- Deformation-based method
  - All-Hex Mesh Generation via Volumetric PolyCube Deformation
- Voxel-based method
  - Optimizing PolyCube domain construction for hexahedral remeshing
- Cluster-based method
  - PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction
- Generalized PolyCube

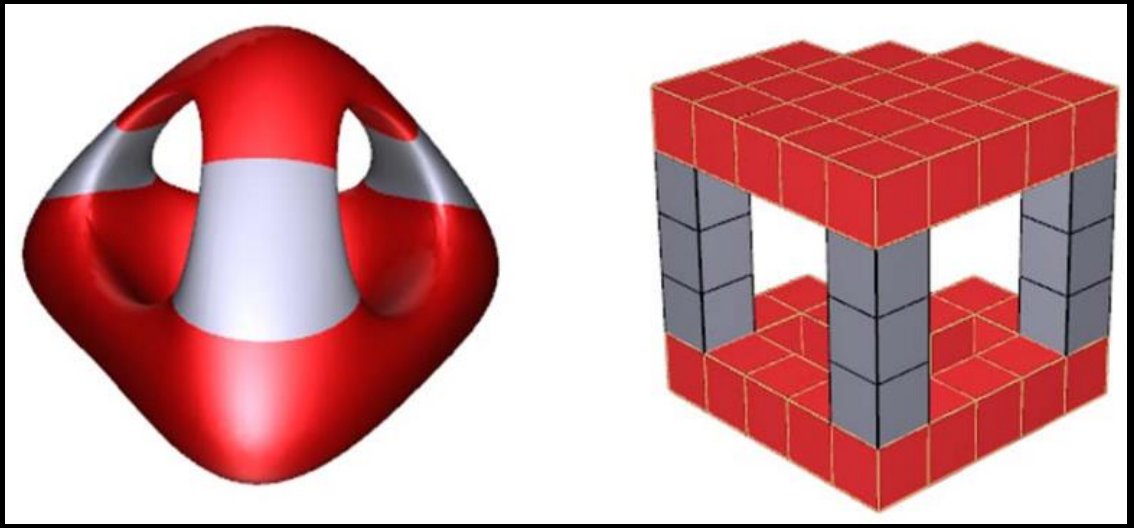
Definitions – Thinking from topology

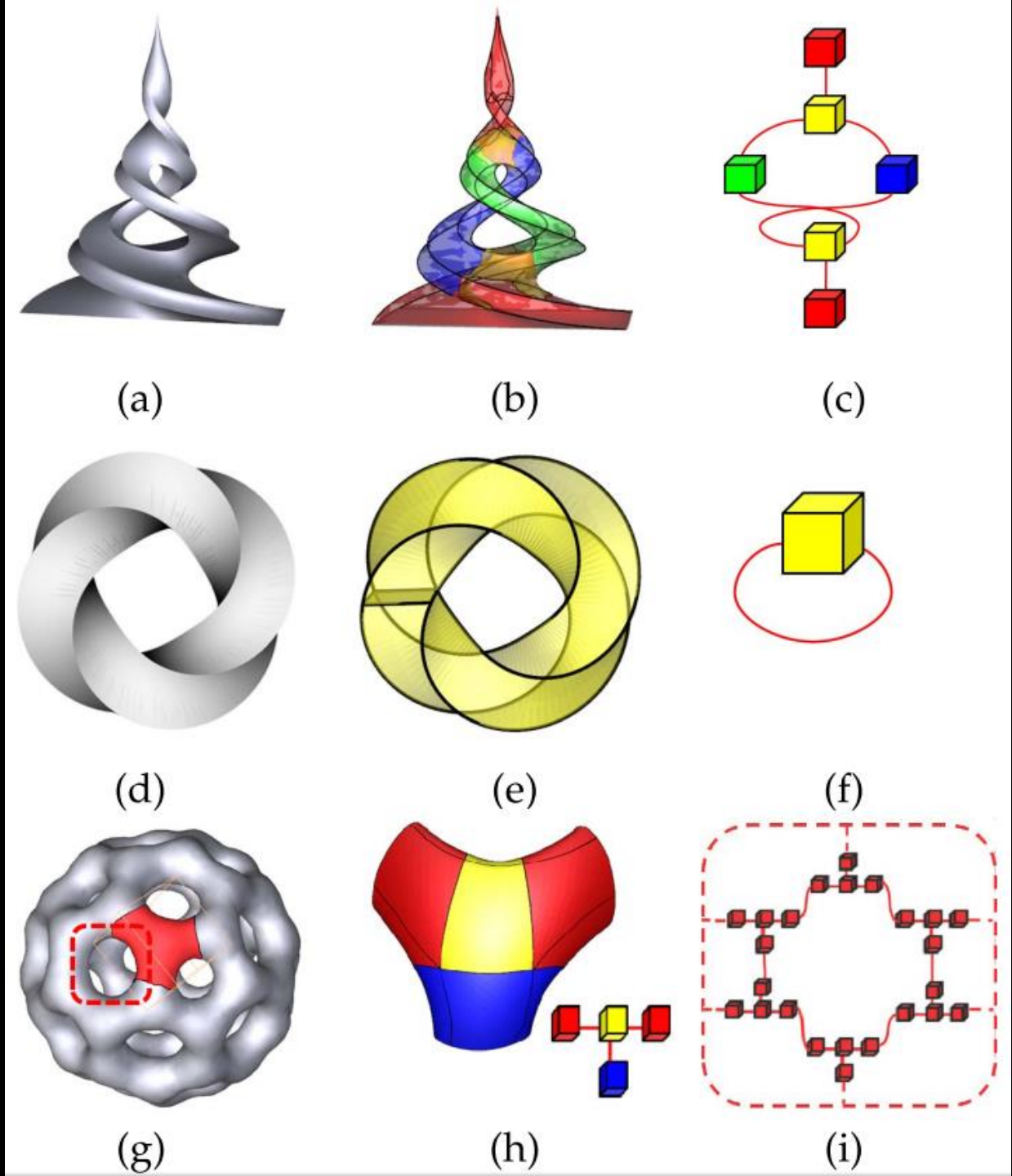


Generalized poly-cubes: (a) The wrench model; (b) The conventional poly-cube (CPC); (c) The generalized poly-cube (GPC) as a topological graph; (d-e) The cuboid edges are overlaid onto the model to visualize the GPC global structure.

More examples





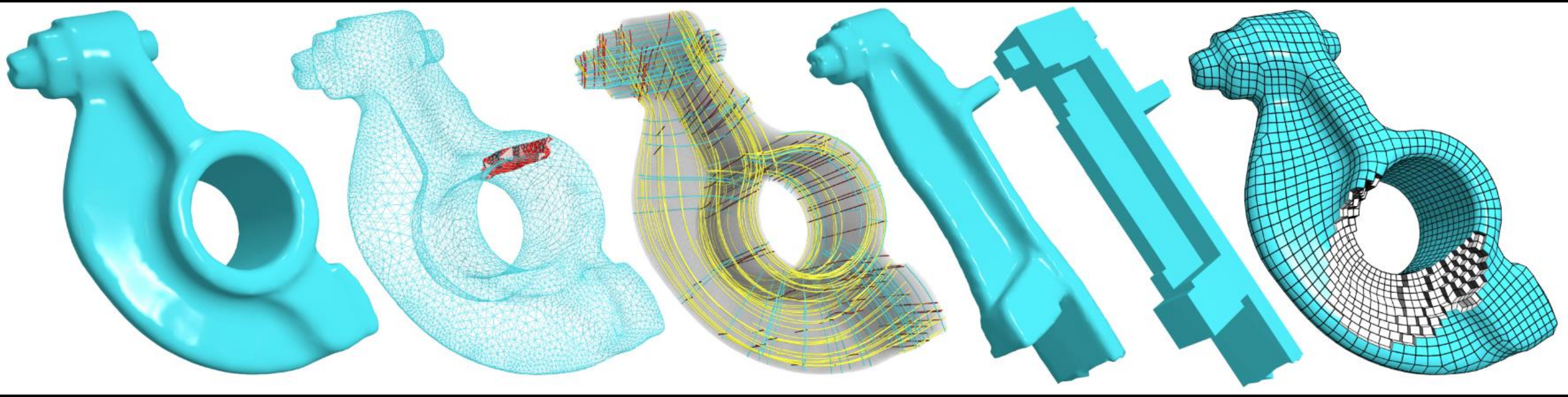


Difference

- Conventional PolyCube:
  - A shape composes of axis-aligned unit cubes that abut with each other.
  - Unit cubes as the building block.
  - All cubes are glued together and embedded in the 3D space.

- Generalized PolyCube:
  - A shape composes of a set of cuboids glued together topologically.
  - Topological simplicity and elegance.

Paper: All-Hex Meshing using Closed-Form Induced Polycube



Cut faces

Frame field

Deformed cut mesh

PolyCube

Rotation

https://en.wikipedia.org/wiki/Rotation

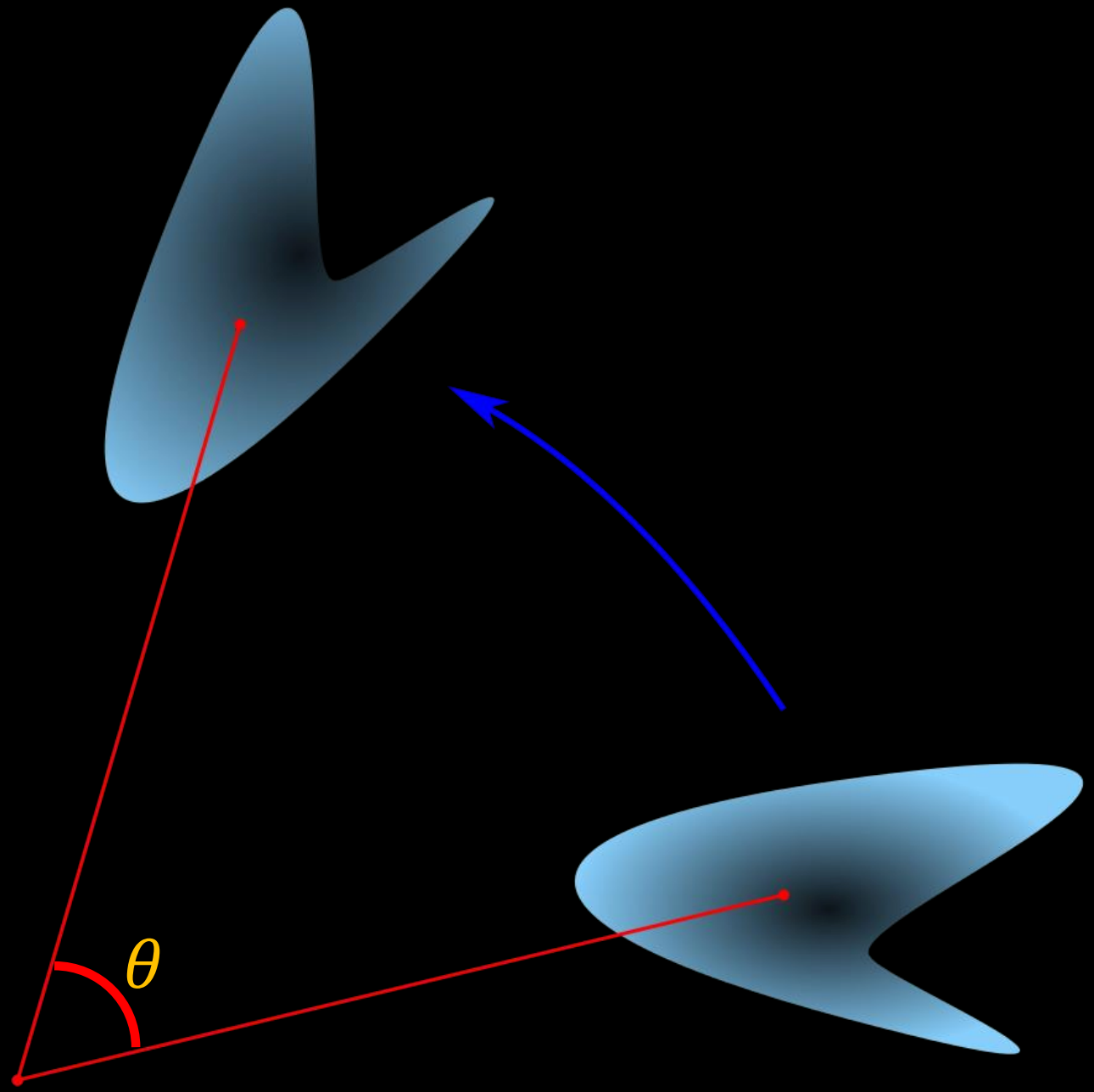
- A **rotation** is a circular movement of an object around a center (or point) of rotation.

- A three-dimensional object can always be rotated around an infinite number of imaginary lines called *rotation axes.*

- A rotation is a **rigid** body movement.

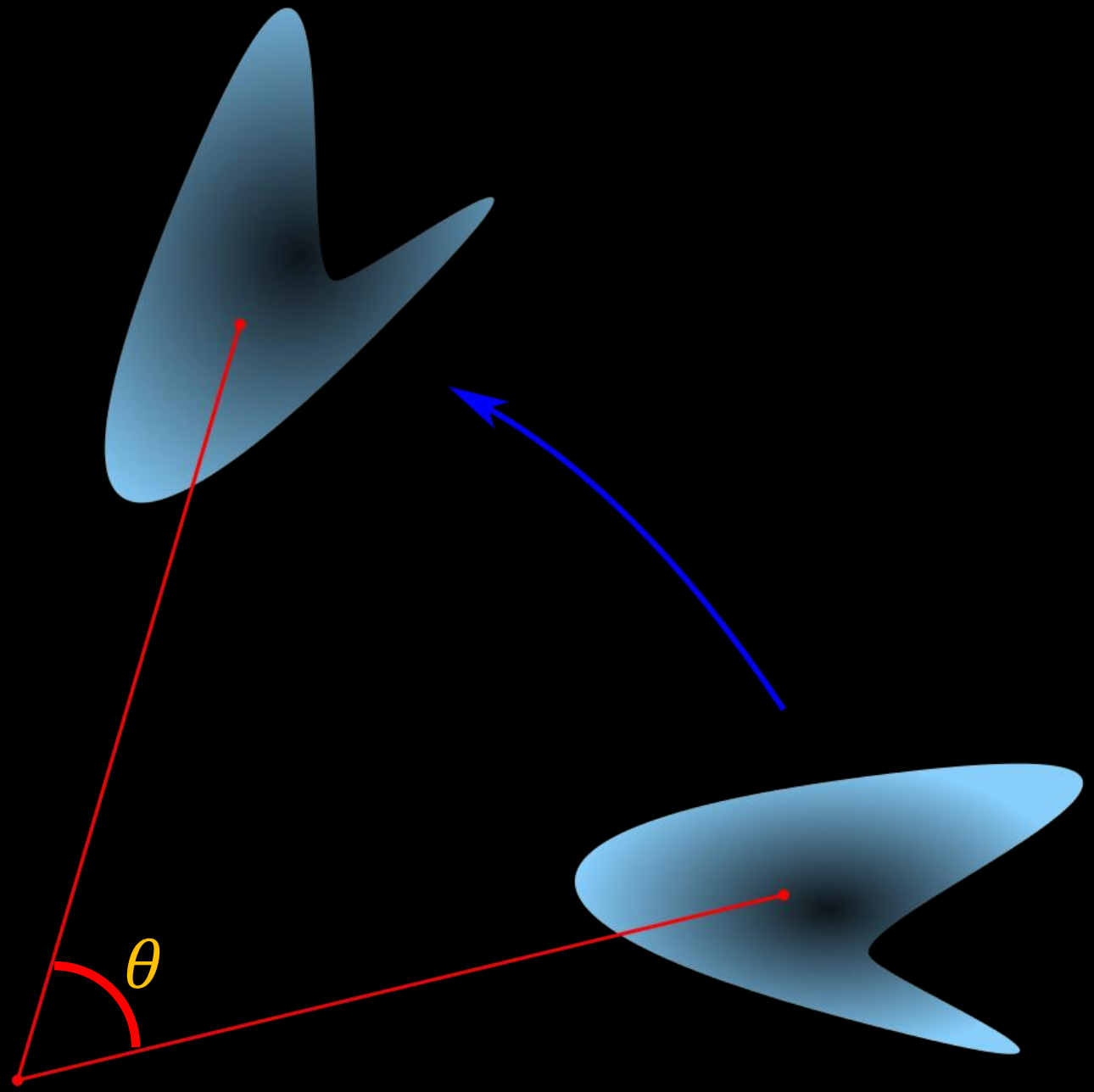
2D rotation

- $R = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix}$



Complex view

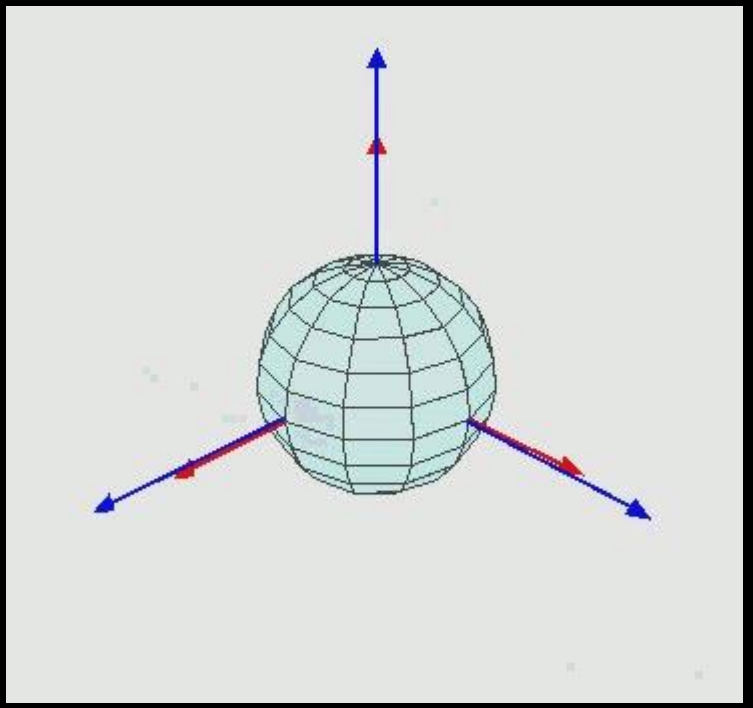
- $z' = e^{i\theta} z$
- z' has the same length as z.
- z' is rotated by θ degrees.



Euler angles

- The Euler angles are three angles introduced by Leonhard Euler to describe the orientation of a rigid body with respect to a fixed coordinate system.

- Any orientation can be achieved by composing three elemental rotations, i.e. rotations about the axes of a coordinate system. Euler angles can be defined by three of these rotations.



Quaternion

https://en.wikipedia.org/wiki/Quaternion

- Form: $q = a + b\boldsymbol{i} + c\boldsymbol{j} + d\boldsymbol{k}$
  - a, b, c, d are real numbers
  - $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ are the fundamental *quaternion units*
  - $\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1$
- Componentwise addition
  - $(a_1 + b_1\boldsymbol{i} + c_1\boldsymbol{j} + d_1\boldsymbol{k}) + (a_2 + b_2\boldsymbol{i} + c_2\boldsymbol{j} + d_2\boldsymbol{k}) = (a_1 + a_2) + (b_1 +$

Quaternion

- $\boldsymbol{ij} = \boldsymbol{k}, \boldsymbol{ji} = -\boldsymbol{k}; \boldsymbol{jk} = \boldsymbol{i}, \boldsymbol{kj} = -\boldsymbol{i}; \boldsymbol{ki} = \boldsymbol{j}, \boldsymbol{ik} = -\boldsymbol{j}.$

- Multiplication *(Hamilton product)*

$$(a_1 + b_1\boldsymbol{i} + c_1\boldsymbol{j} + d_1\boldsymbol{k})(a_2 + b_2\boldsymbol{i} + c_2\boldsymbol{j} + d_2\boldsymbol{k})$$
$$= a_1(a_2 + b_2\boldsymbol{i} + c_2\boldsymbol{j} + d_2\boldsymbol{k}) + b_1\boldsymbol{i}(a_2 + b_2\boldsymbol{i} + c_2\boldsymbol{j} + d_2\boldsymbol{k})$$
$$+ c_1\boldsymbol{j}(a_2 + b_2\boldsymbol{i} + c_2\boldsymbol{j} + d_2\boldsymbol{k}) + d_1\boldsymbol{k}(a_2 + b_2\boldsymbol{i} + c_2\boldsymbol{j} + d_2\boldsymbol{k})$$
$$= a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2 + (a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2)\boldsymbol{i}$$
$$+ (a_1c_2 - b_1d_2 + c_1a_2 + d_1b_2)\boldsymbol{j} + (a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2)\boldsymbol{k}$$

noncommutative

Conjugation, the norm, and reciprocal

- Conjugation:
  - $q^* = a - b\boldsymbol{i} - c\boldsymbol{j} - d\boldsymbol{k}$

- Norm:
  - $\|q\| = \sqrt{qq^*} = \sqrt{a^2 + b^2 + c^2 + d^2}$

- Reciprocal:
  - $q^{-1} = \frac{q^*}{\|q\|^2}$

Unit quaternion

- Because the vector part of a quaternion is a vector in R^3, the geometry of R^3 is reflected in the algebraic structure of the quaternions.

- A single rotation by a given angle θ about a fixed axis $u = x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}$
- An extension of Euler's formula:
  - $q = e^{\frac{\theta}{2}(x\boldsymbol{i}+y\boldsymbol{j}+z\boldsymbol{k})} = \cos\frac{\theta}{2} + (x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k})\sin\frac{\theta}{2}$
- The desired rotation can be applied to an ordinary vector $p = p_x\boldsymbol{i} + p_y\boldsymbol{j} + p_z\boldsymbol{k}$
  - $p' = qpq^{-1}$
  - $q^{-1} = q^* = \cos\frac{\theta}{2} - (x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k})\sin\frac{\theta}{2}$

Unit quaternion ⟺ Rotation Matrix

$q = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} \Leftrightarrow M$

- Unit quaternion ⟹ Rotation Matrix

$$M = \begin{pmatrix} 1-2y^2-2z^2 & 2xy+2zw & 2xz-2yw \\ 2xy-2zw & 1-2x^2-2z^2 & 2yz+2xw \\ 2xz+2yw & 2yz-2xw & 1-2x^2-2y^2 \end{pmatrix}$$

- Unit quaternion ⟸ Rotation Matrix

Suppose $M = \begin{pmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{pmatrix}$

$m_{00} + m_{11} + m_{22} = 3 - 4(x^2 + y^2 + z^2) = 3 - 4(1 - w^2) = -1 + 4w^2$ **(ambiguity)**

$m_{01} - m_{10} = 4zw; m_{20} - m_{02} = 4yw; m_{12} - m_{21} = 4xw$