Immersion of Self-Intersecting Solids and Surfaces

Yijing Li, Jernej Barbič

University of Southern California

SIGGRAPH2018

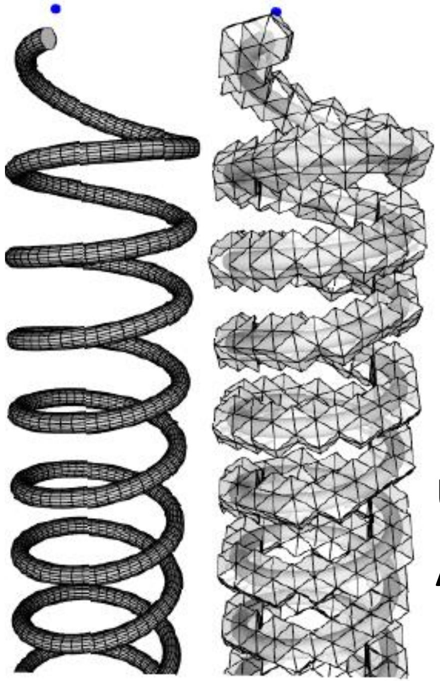
Input: Self-intersecting triangle mesh

zoom

self-intersecting helix

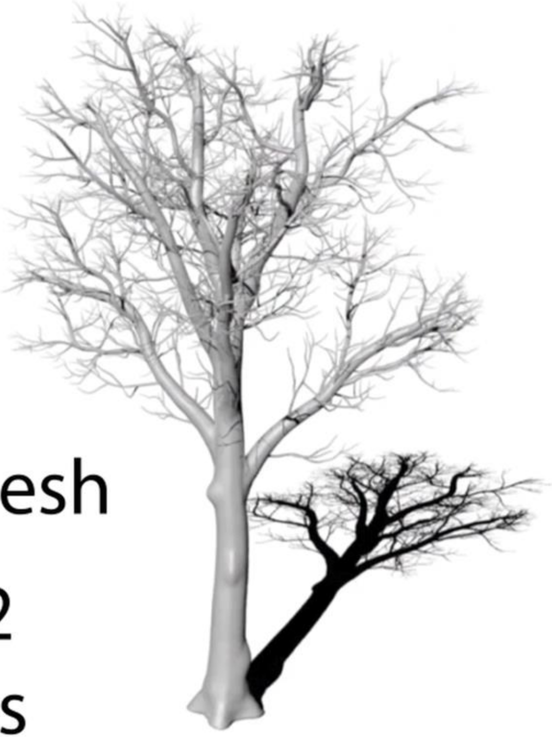
intersections=red

Output: Tet mesh of enclosing volume that respects topology



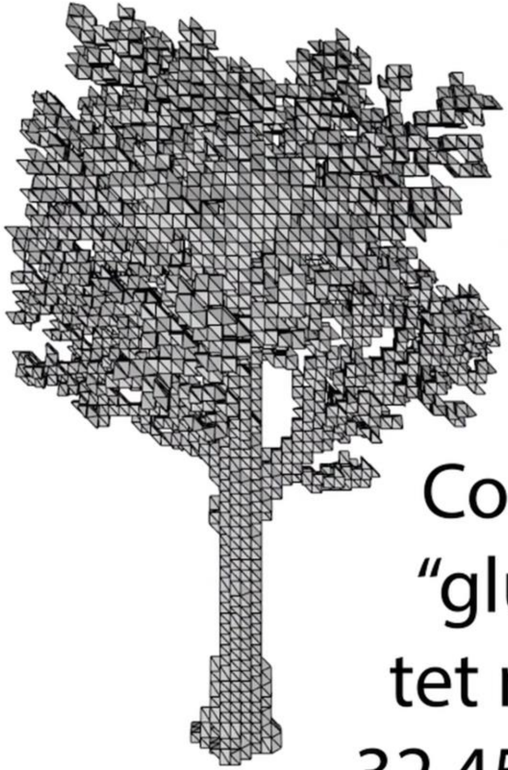
Deformed using ARAP

Application: FEM Volumetric Simulation



Input mesh

328,152 triangles

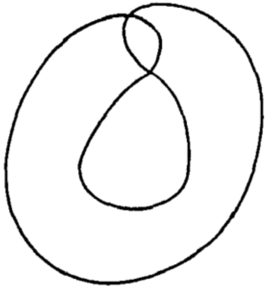


Coarse "glued" tet mesh

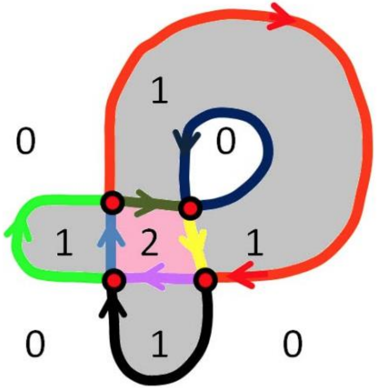
32,457 tets

Related Work: 2D Self-Intersection

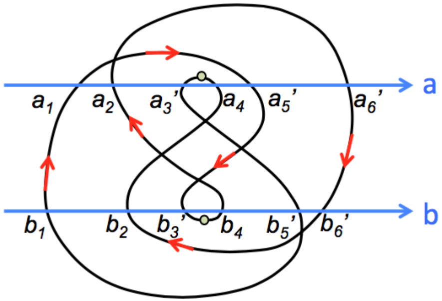
2D disk topology, immersion



[Shor 1989]



[Mukherjee 2011]



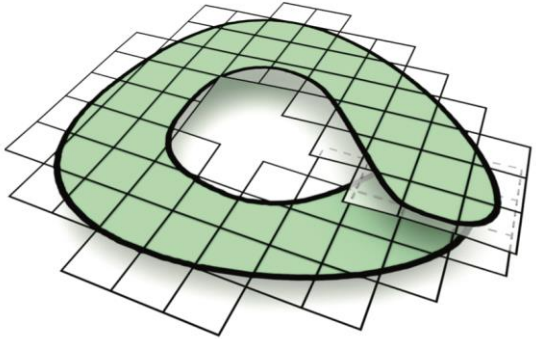
[Mukherjee 2014]

Related Work: 3D Self-Intersection

Sphere-oriented flow

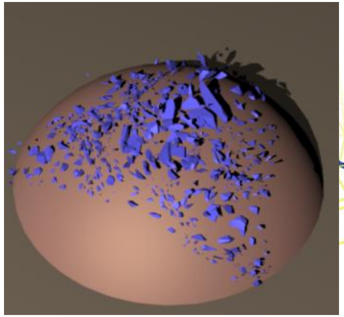
[Sacht 2013]

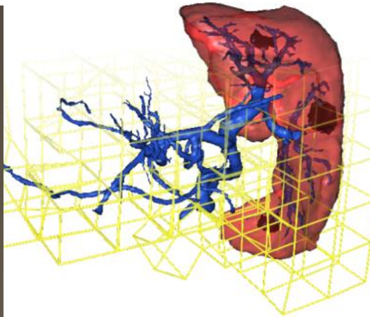
Non-manifold level sets

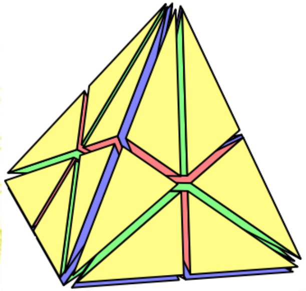


[Mitchell 2015]

Embeddings:



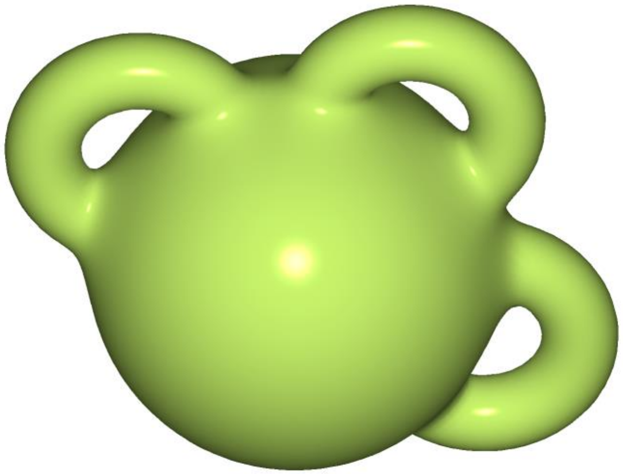




[Molino 2004][Teran 2005] [Sifakis 2007] [Nesme 2009] [Wang 2014]

Geometric topology

A field of mathematics studying manifolds embedded into Euclidean spaces

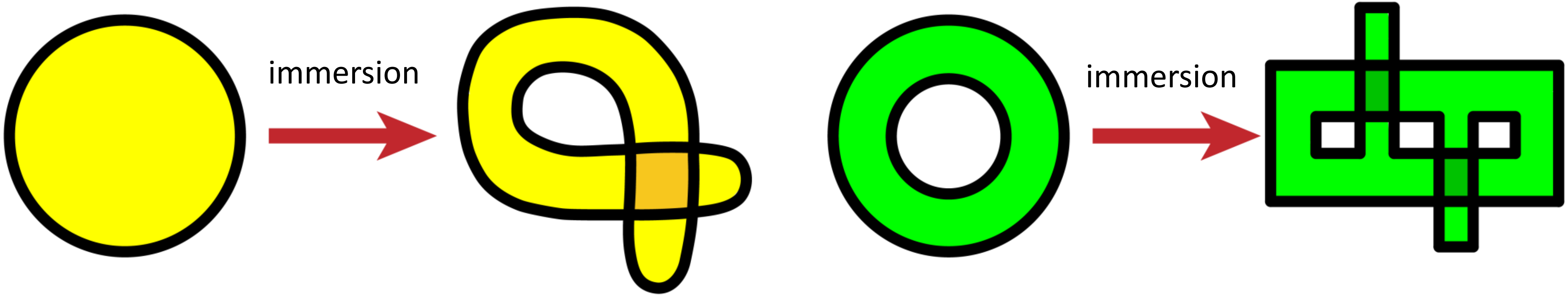


Algebraic topology

A field of mathematics studying shapes by assigning algebraic groups to them

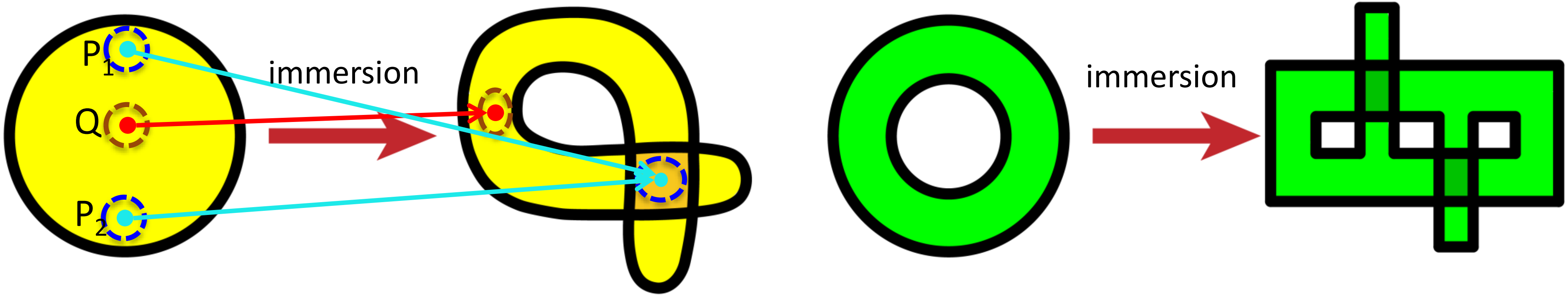
$$\mathbb{Z} \oplus \mathbb{Z} \oplus \mathbb{Z}$$

Immersion



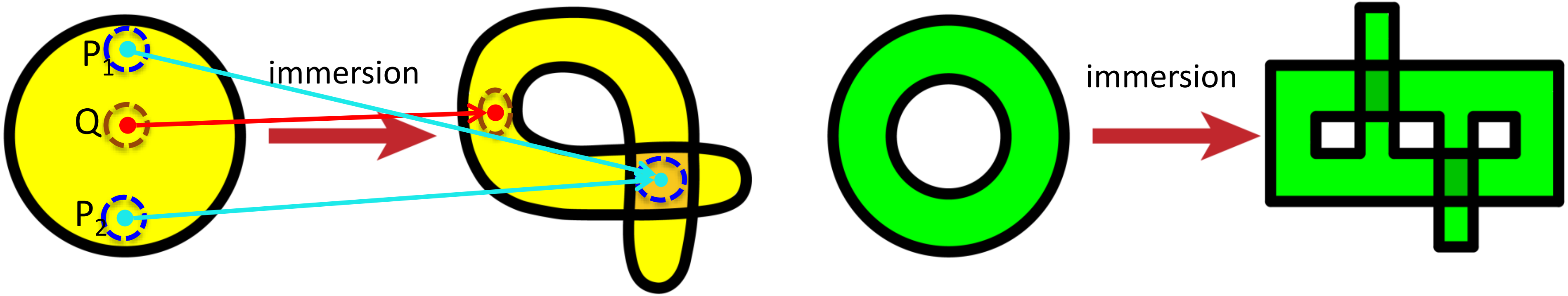
Continuous map between two topological spaces that is locally injective

Immersion (local injectivity)



Continuous map between two topological spaces that is locally injective

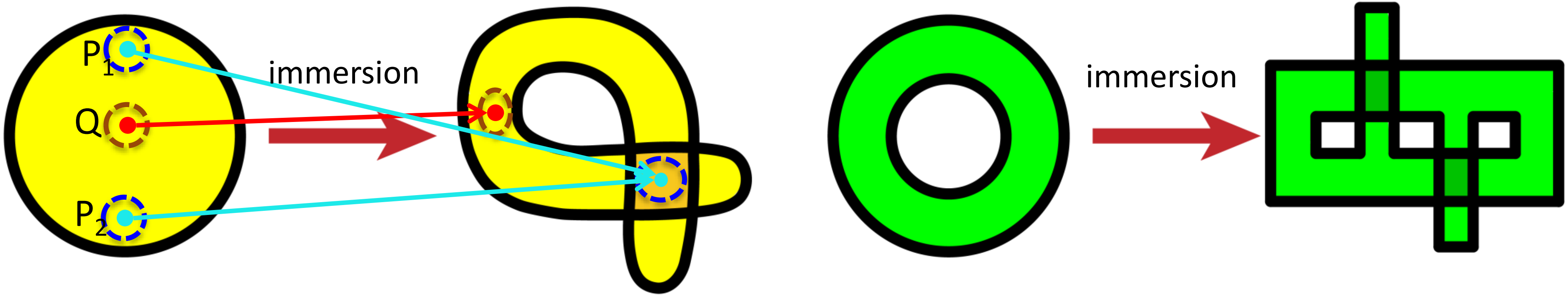
Immersion



Continuous map between two topological spaces that is locally injective

Boundary of immersion → Self-intersecting shape

Immersion

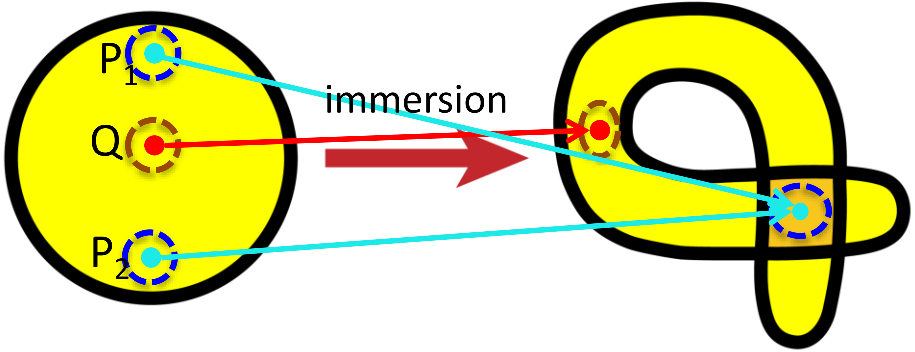


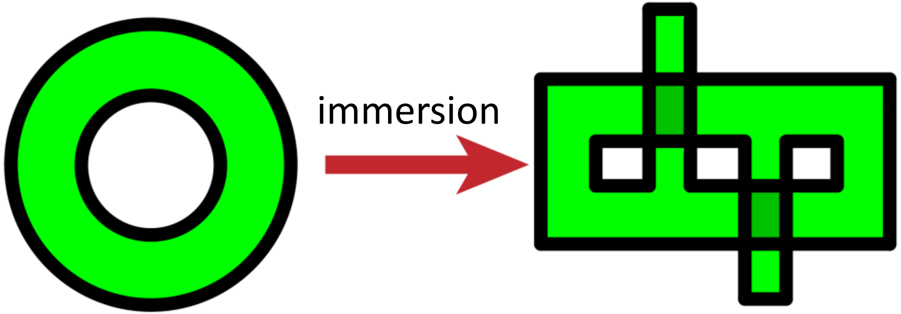
Continuous map between two topological spaces that is locally injective

Boundary of immersion

Self-intersecting shape

Immersion

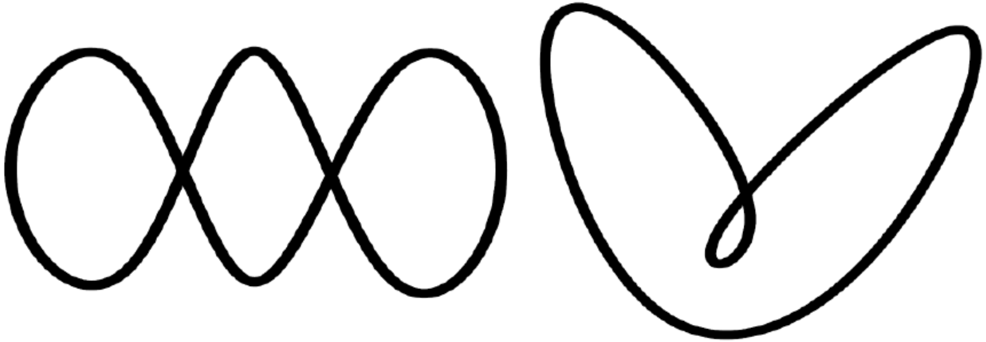




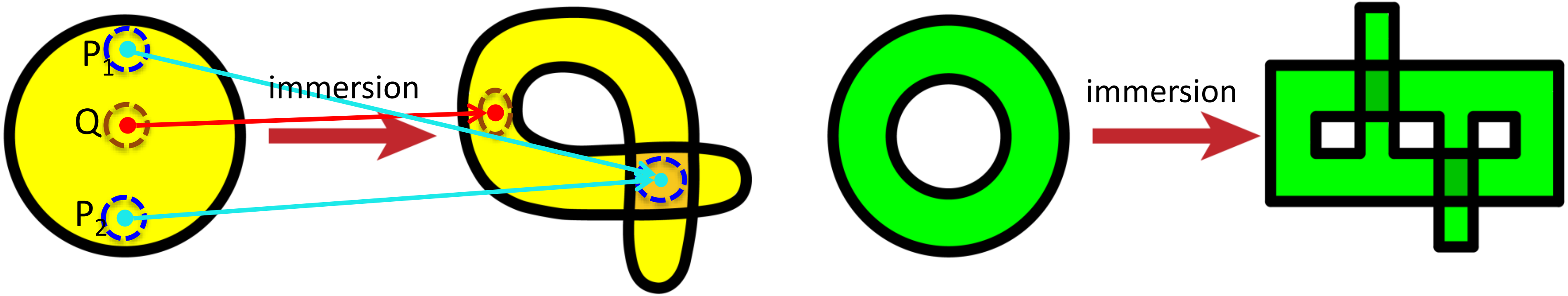
Boundary of immersion → Self-intersecting shape

Boundary of immersion ~~←~~ Self-intersecting shape

Inversion:



Immersion

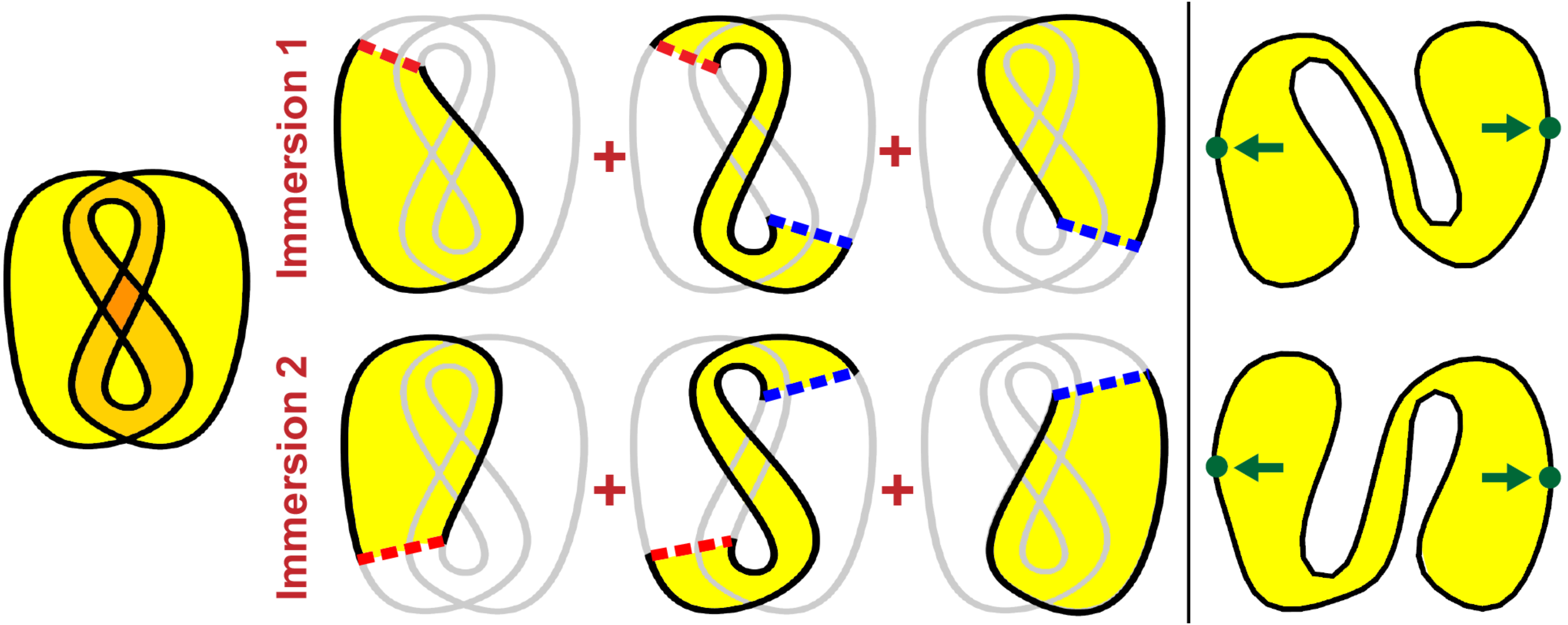


Given input triangle mesh

Find the immersion or report non-existence
(proof in paper via algebraic topology)

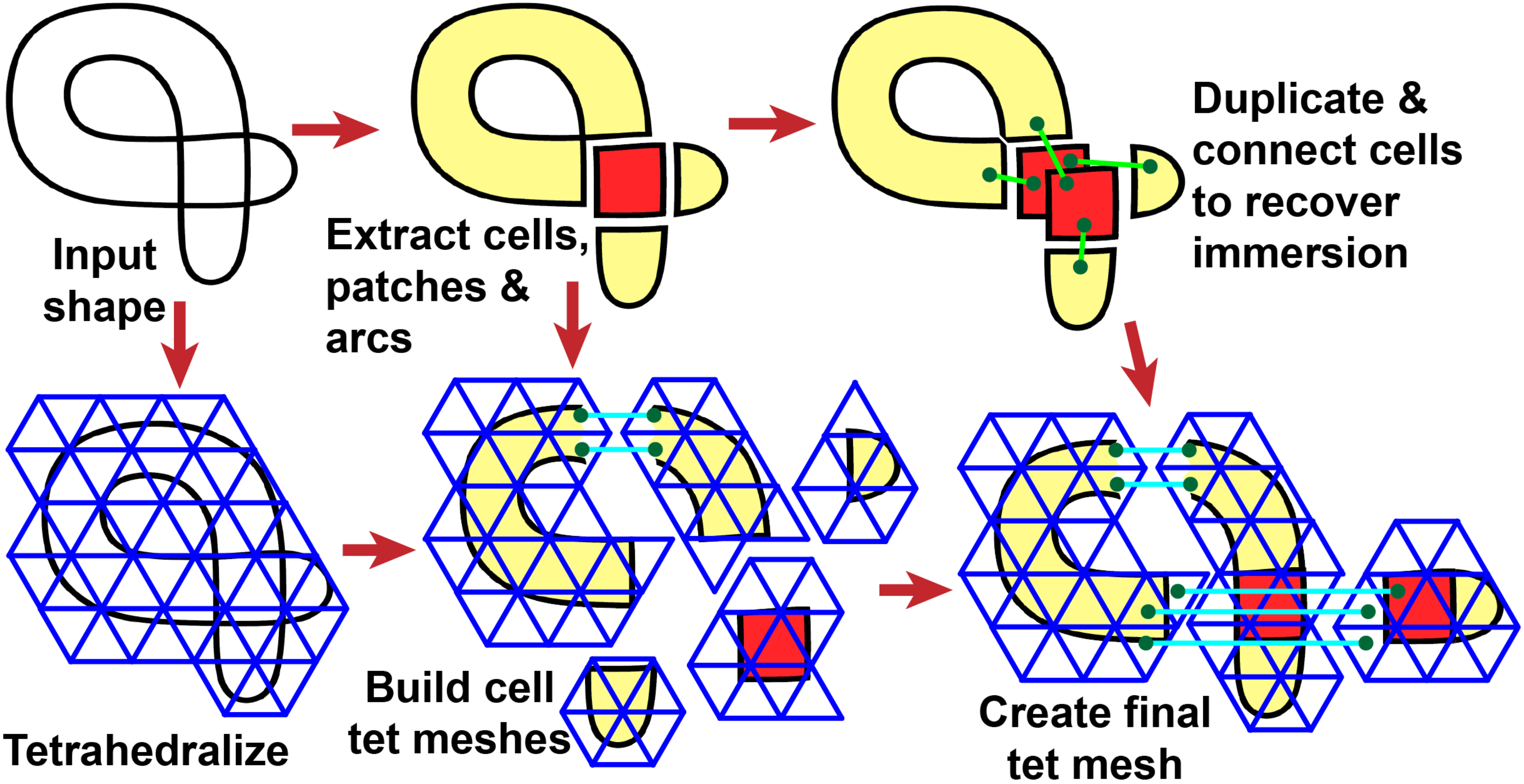
If exists, build volumetric mesh

Ambiguity in immersion



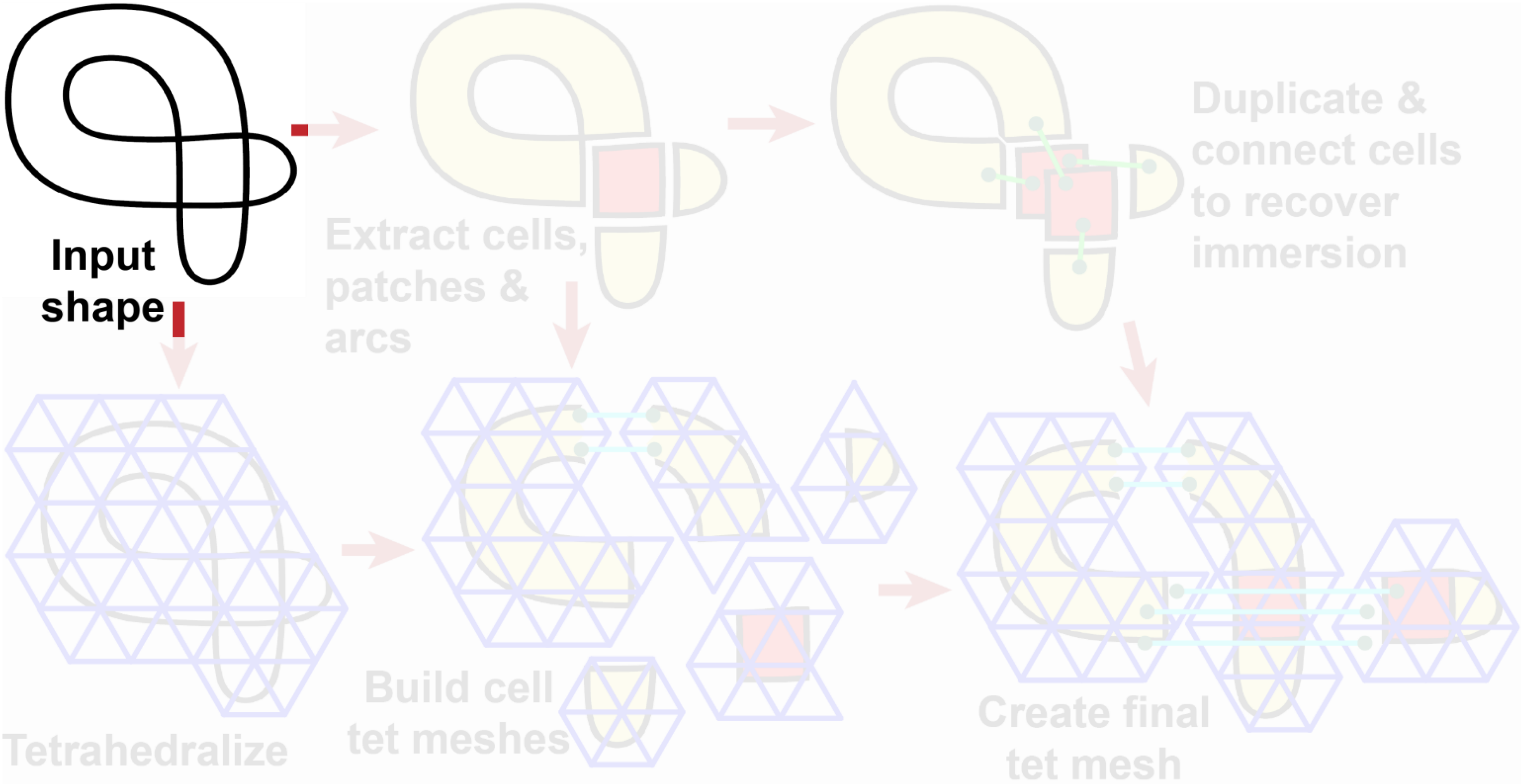
Our method finds all possible immersions

Overview



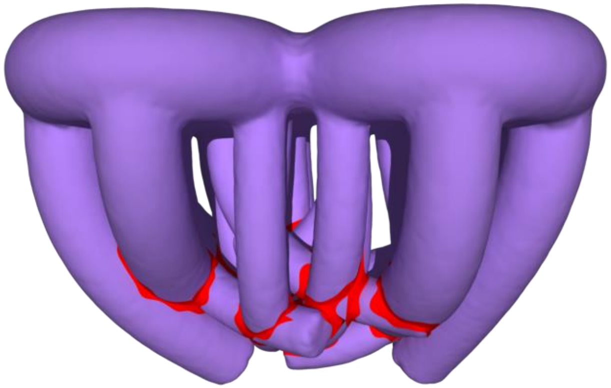
Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

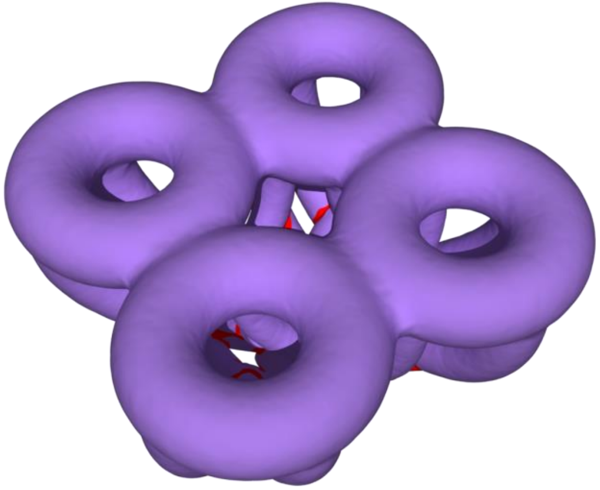
Overview


Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

Input

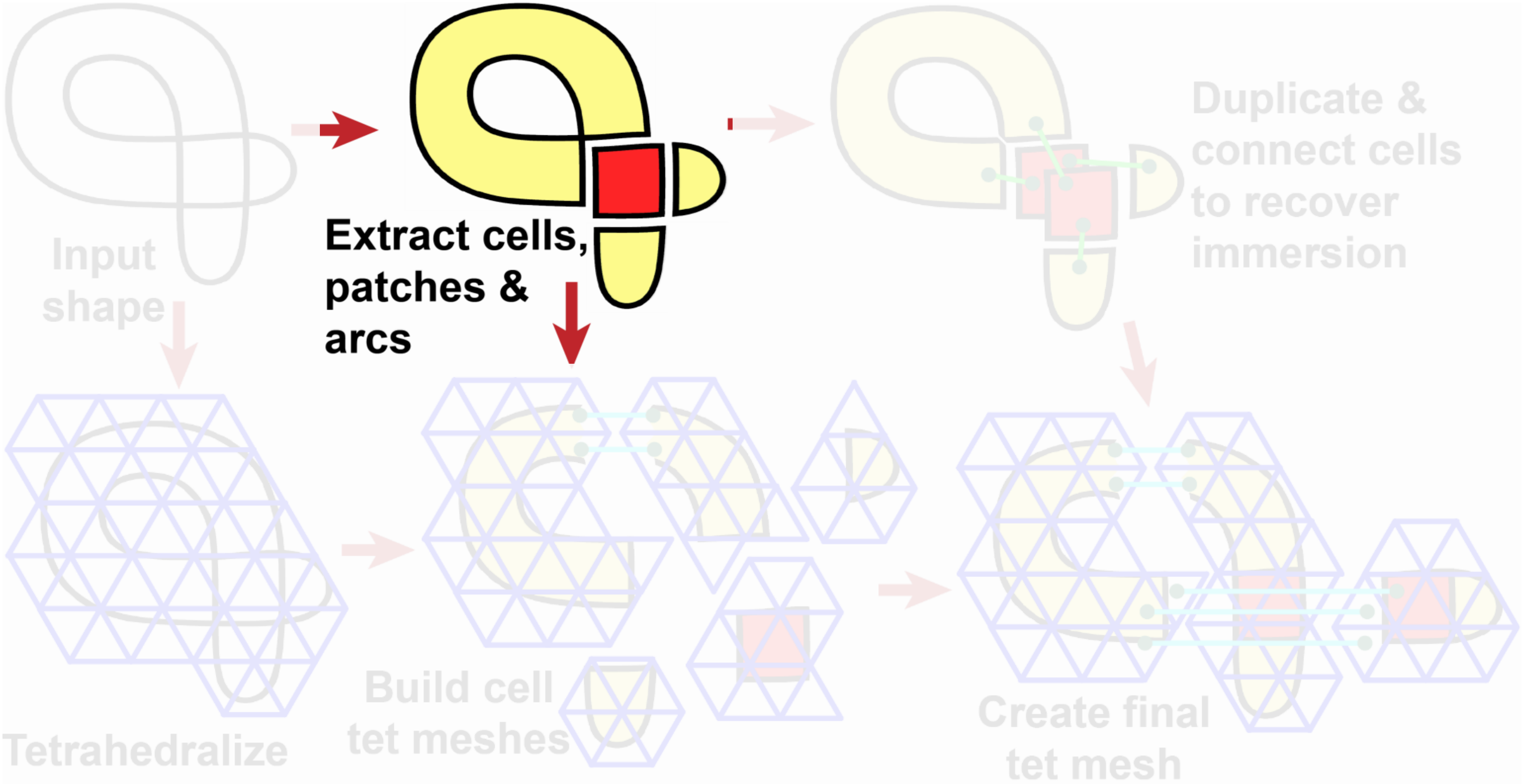
- Manifold
- Orientable
- No boundary
- Non-degenerate
- Arbitrary #holes and #handles





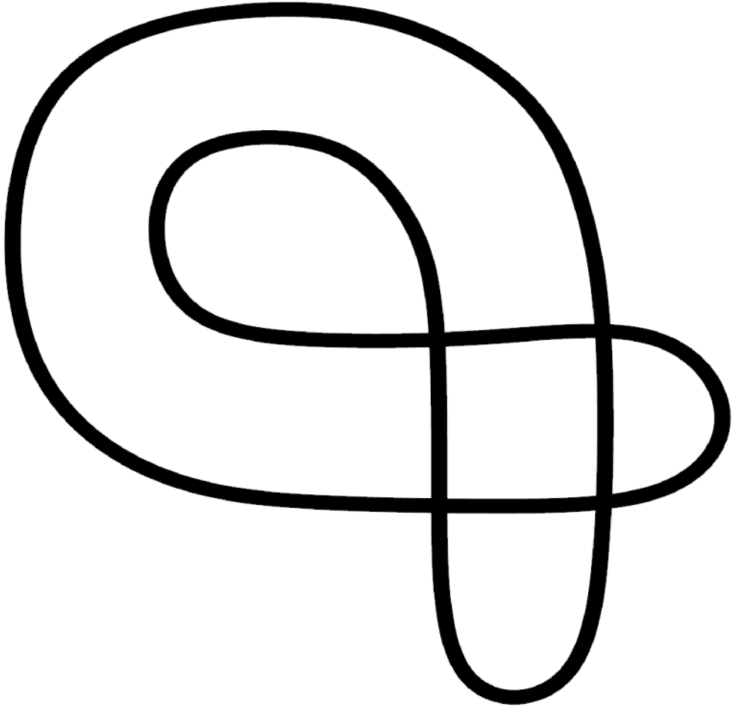
another view

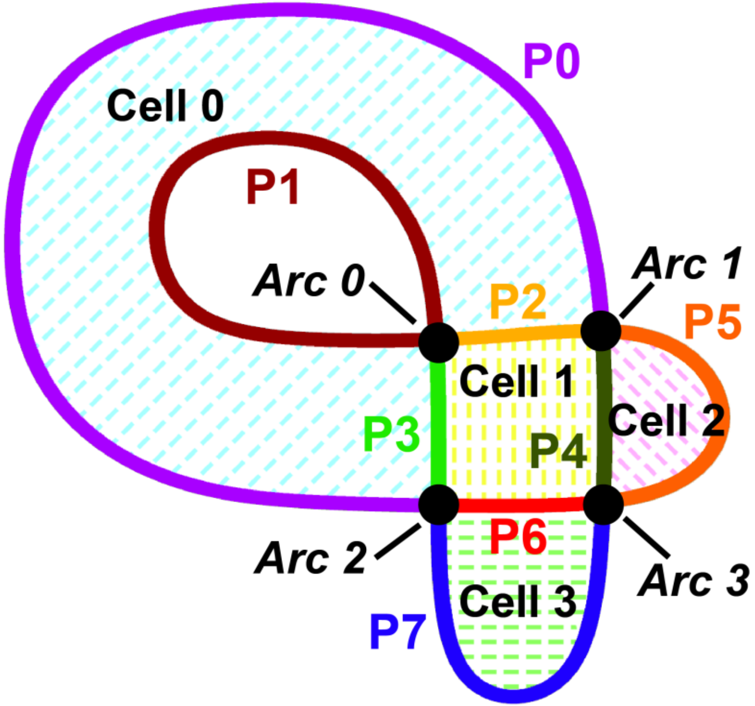
Overview



Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

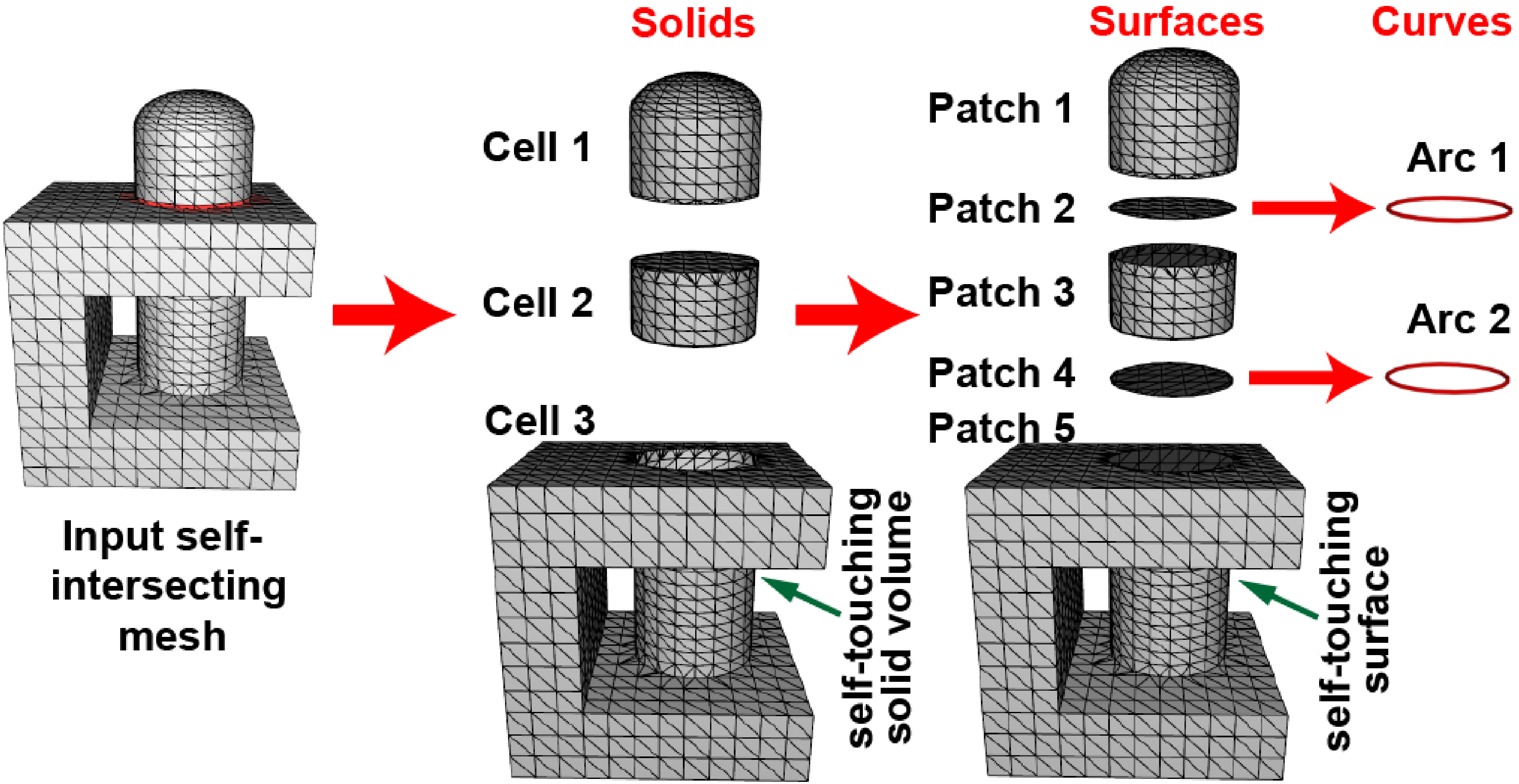
Cell Complex



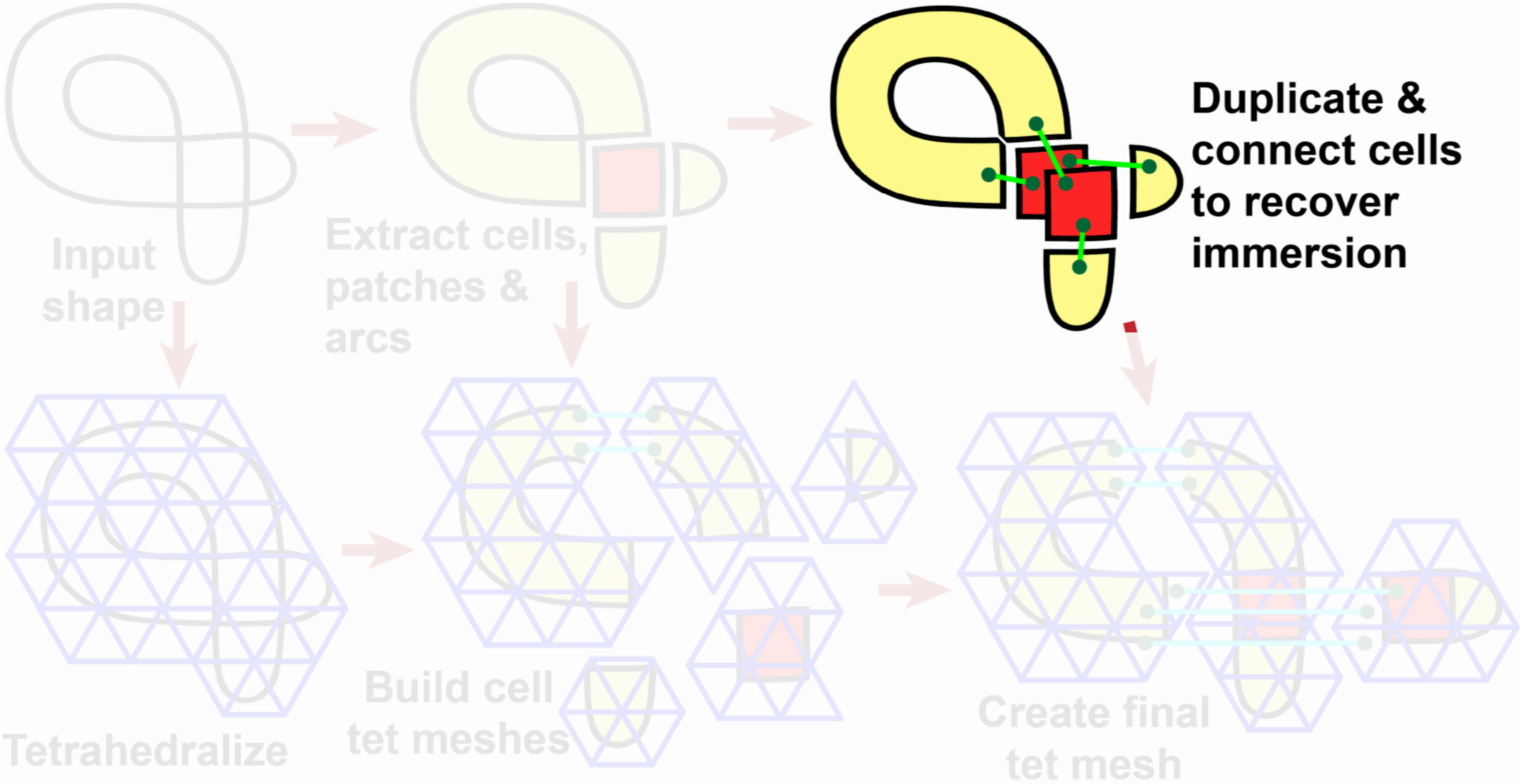


Find cells & patches: [Zhou 2016]

Cell Complex

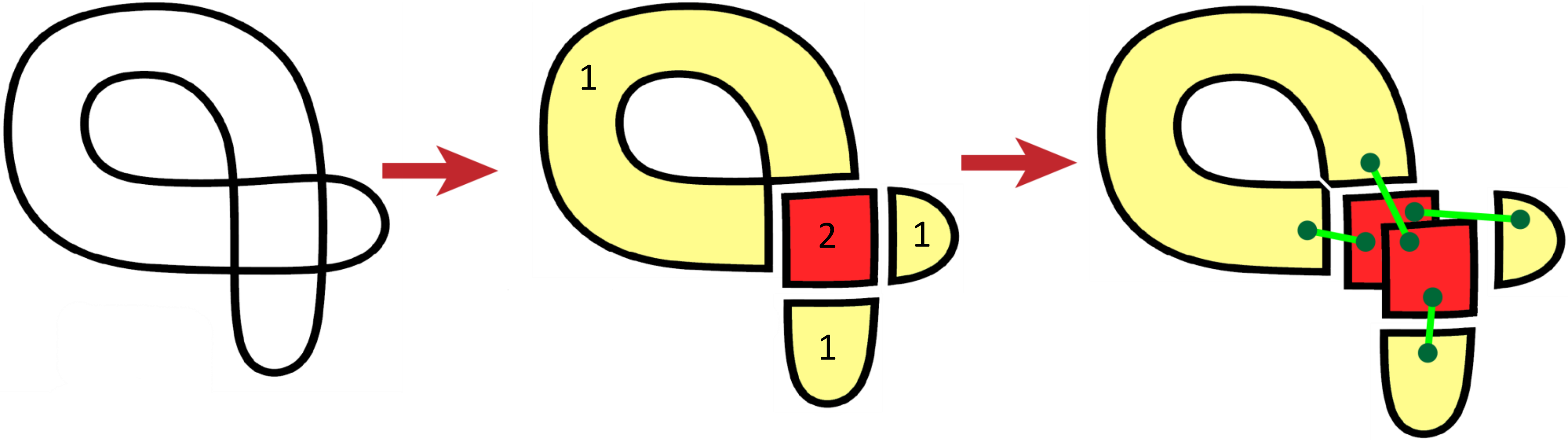


Overview


Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

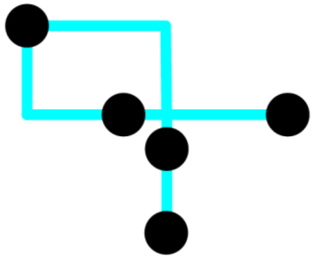
Duplicate Cells

- # copies of a cell = # times the domain of immersion covers the cell = cell winding number

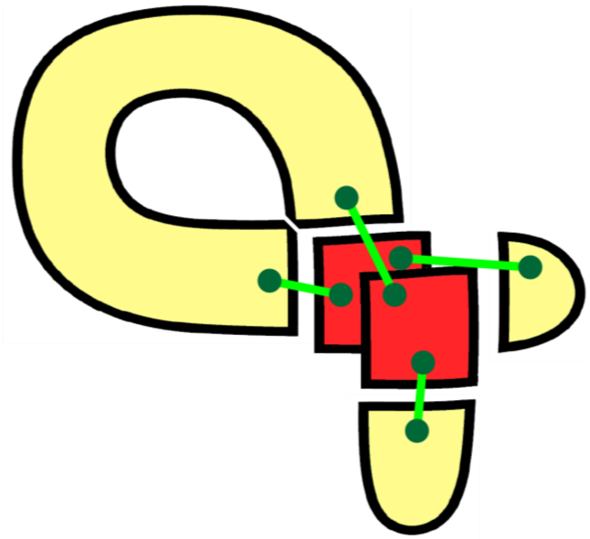


Immersion Graph

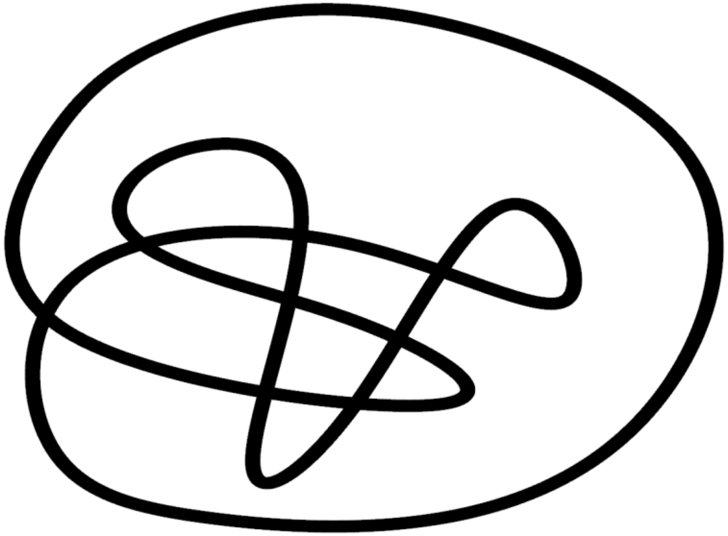
- Nodes: duplicated cells
- Edges: surface patches between cells



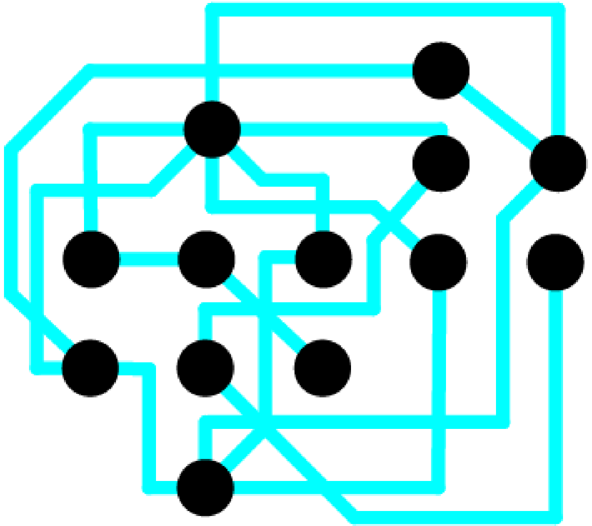
immersion graph



More complex immersion graph example



input shape



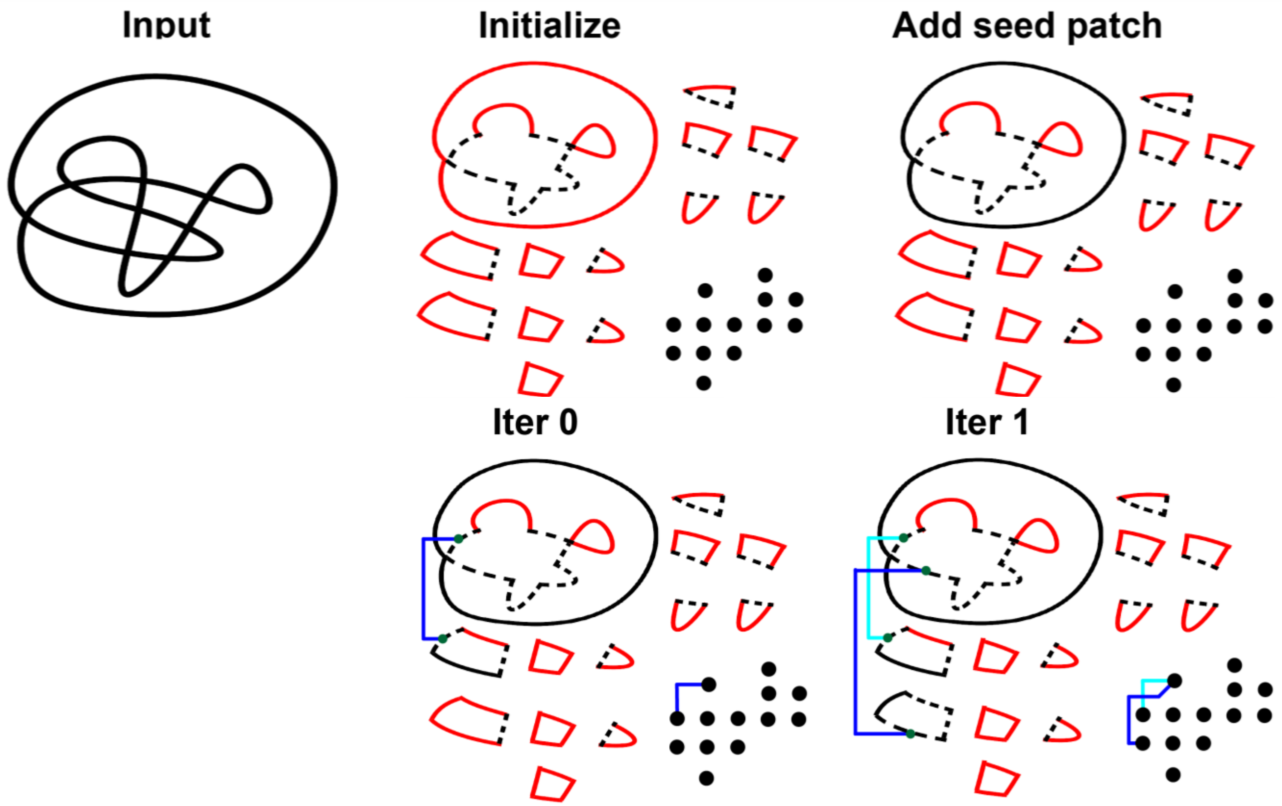
immersion
graph

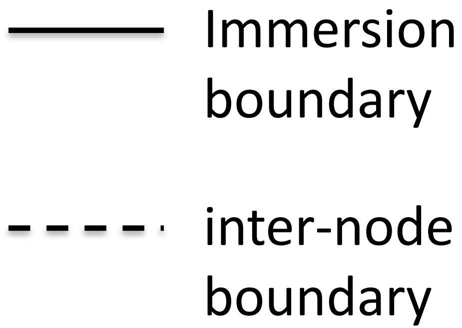
Building the immersion graph

- We state **7 Rules** stemming from the immersion requirement

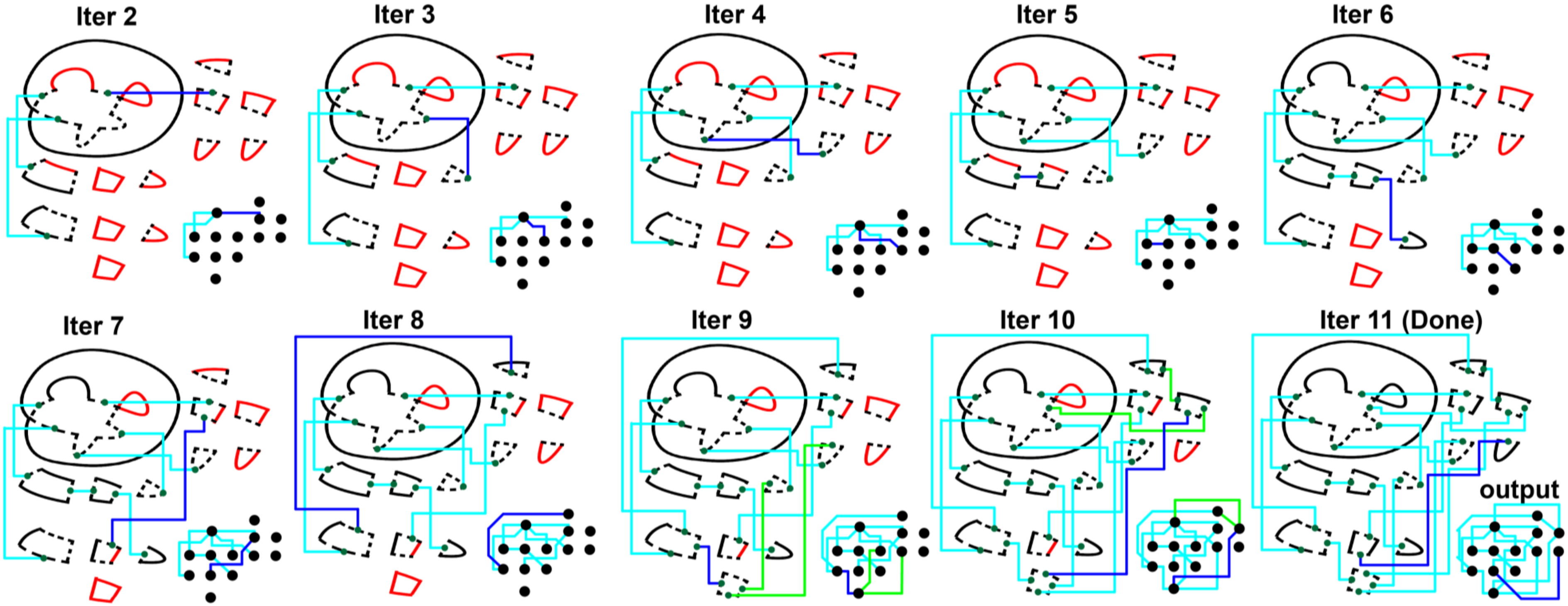
- We prove that the 7 Rules are necessary and sufficient to discover all possible graphs

Building the immersion graph

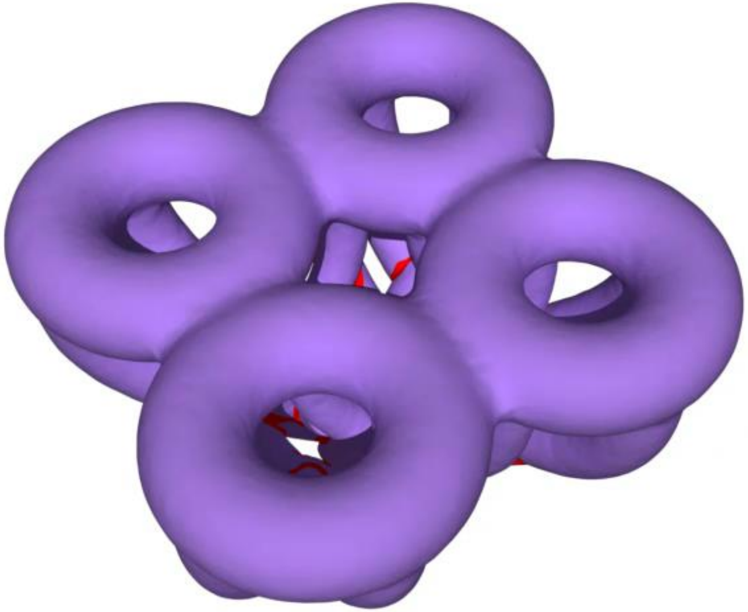




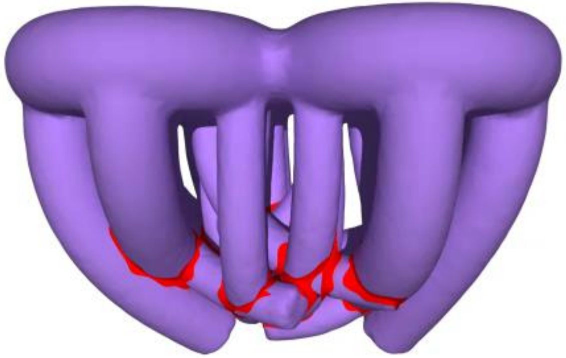
Building the immersion graph



Examples

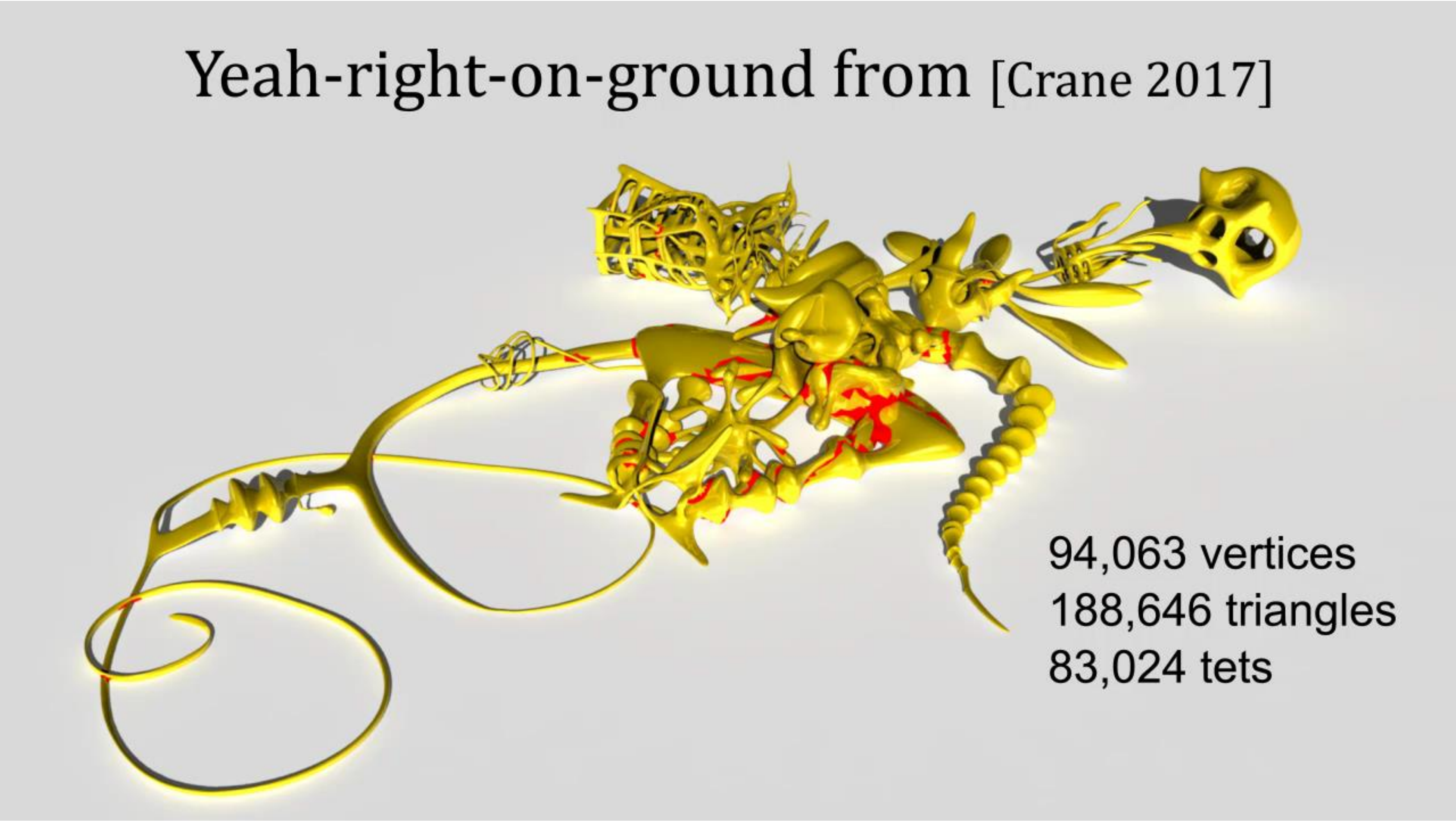


Side view



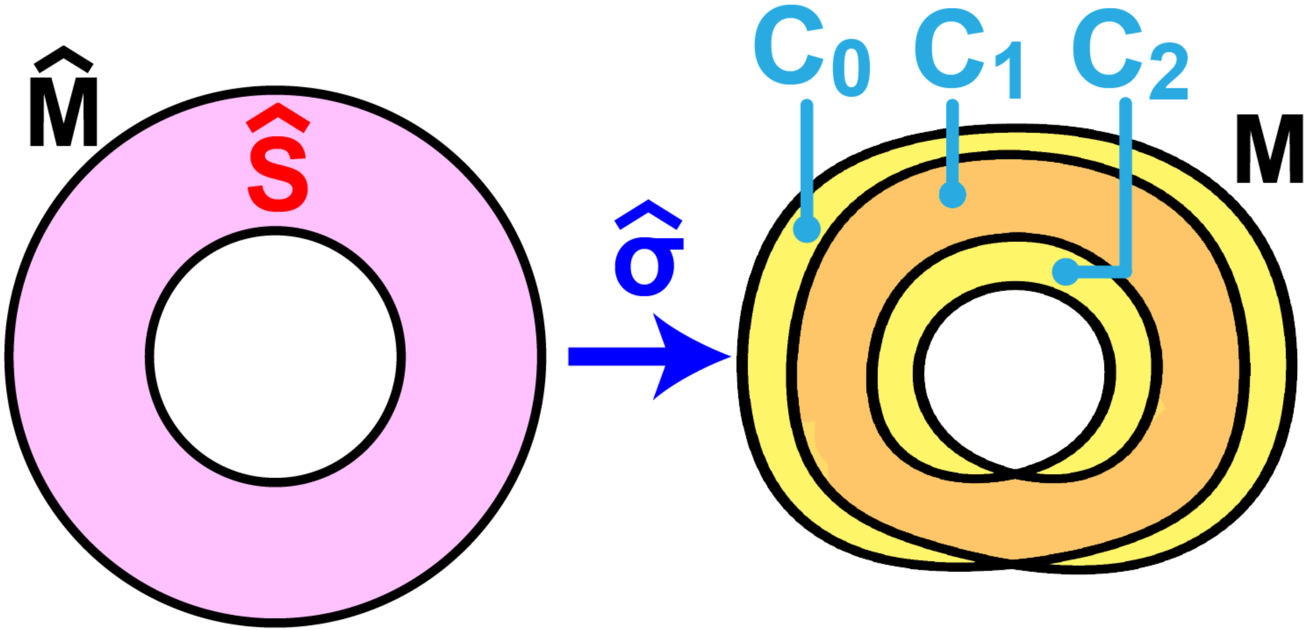
15,211 vertices
30,438 triangles
84,905 tets

Examples

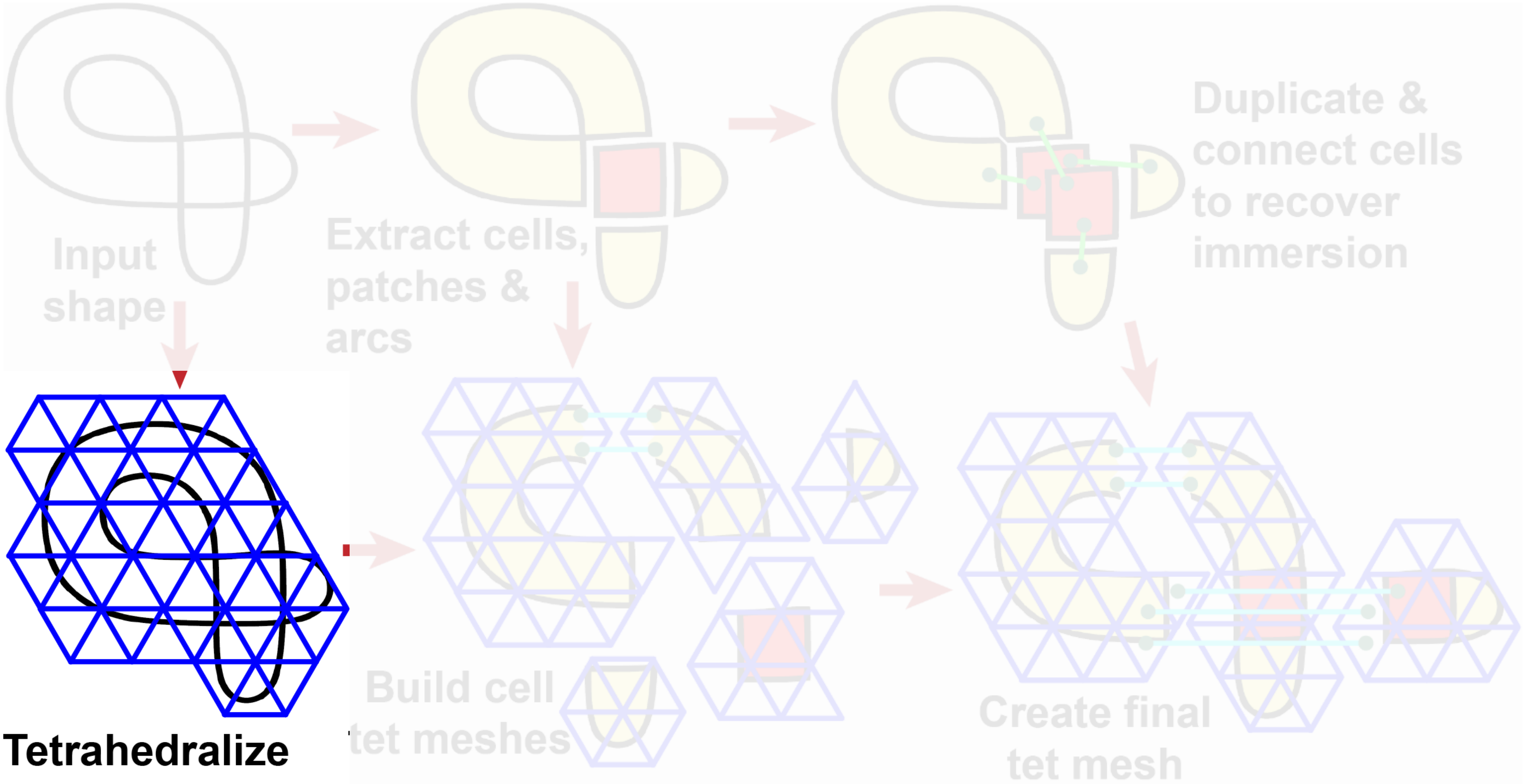

Yeah-right-on-ground from [Crane 2017]
94,063 vertices
188,646 triangles
83,024 tets

Non-simple Immersions

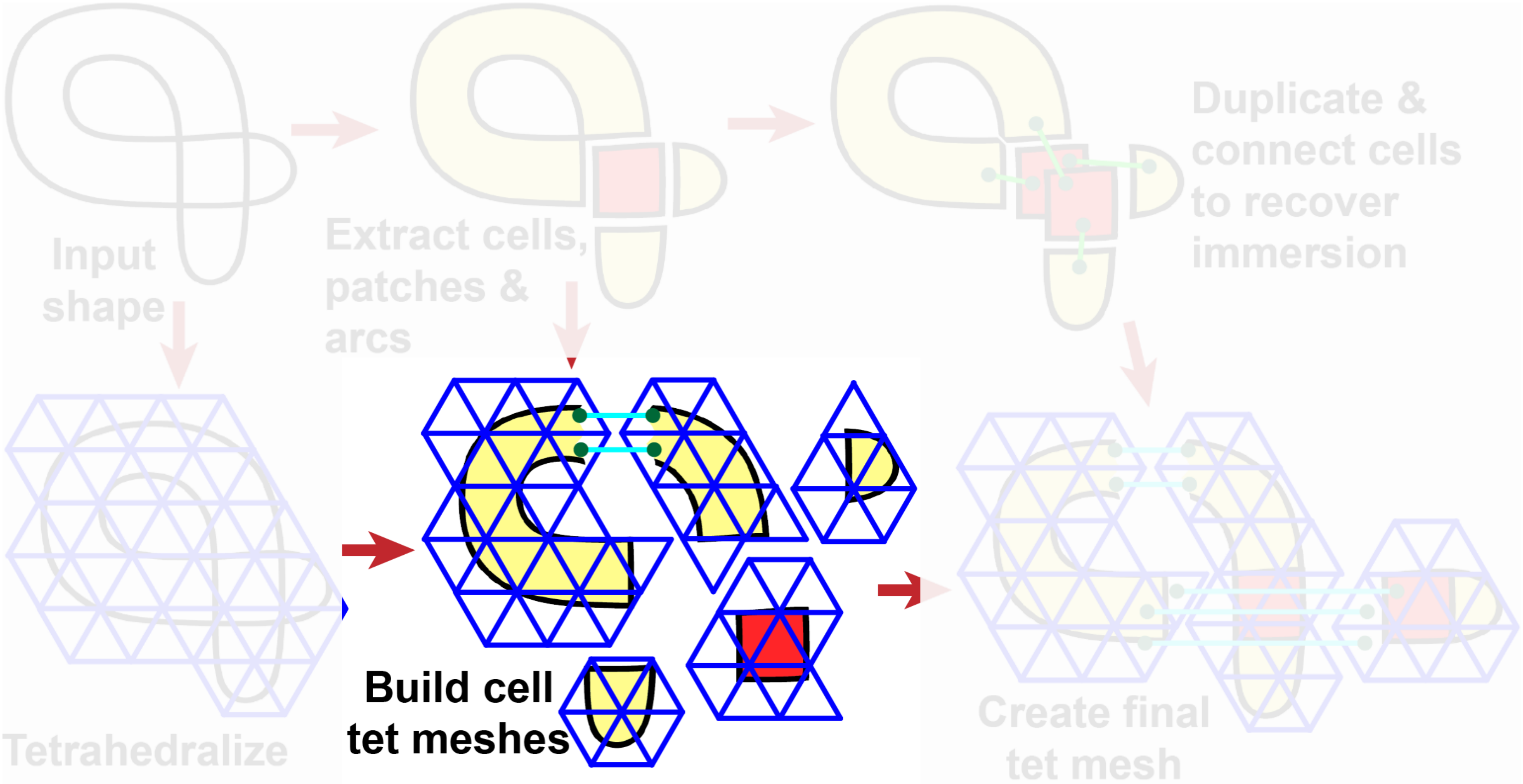
- Requires nodes to connect to another node of the *same* cell



Overview

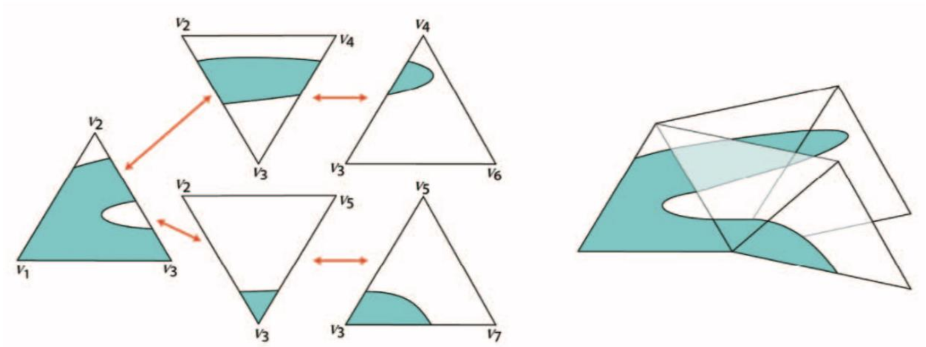

Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

Overview

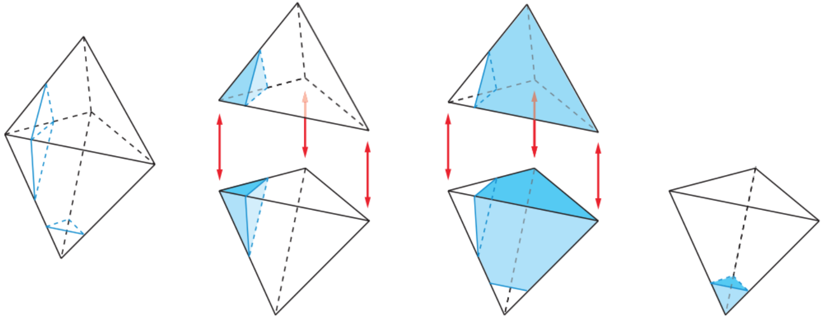

Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

Create Cell Volumetric Meshes

- Should not *glue* proximity geometry
- Duplicate and connect tets
- Prior work used CSG (slow)



[Teran 2005]

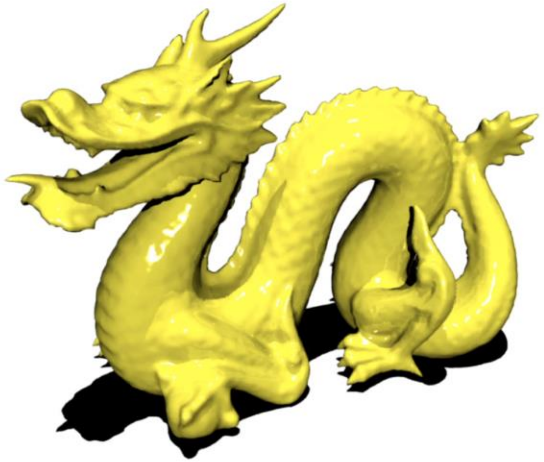


[Sifakis 2007]

We give a new faster method to generate virtualized tet meshes for nearly self-intersecting geometry

30x faster than prior work

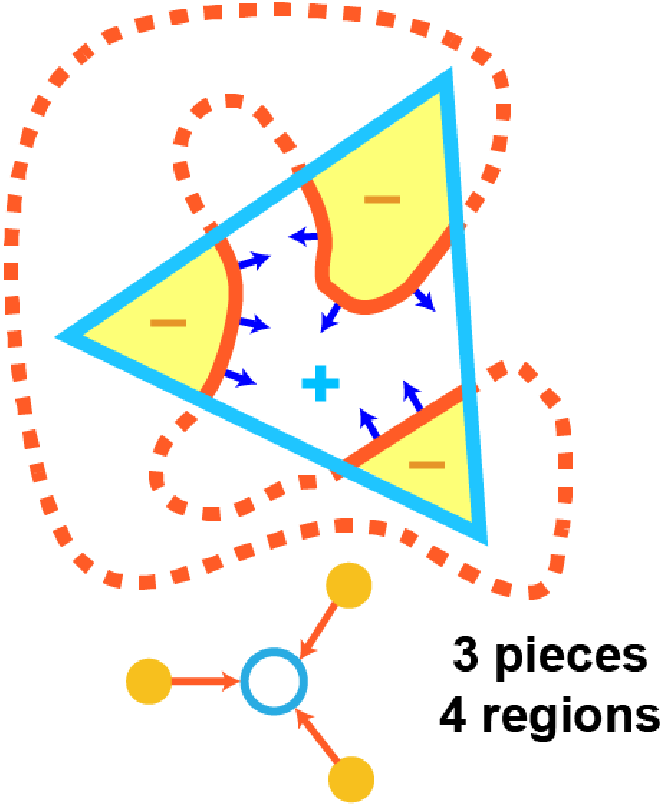
- Sutherland-Hodgman clipping
- Pseudo-normal tests
- no CSG



nearly self-intersecting geometry

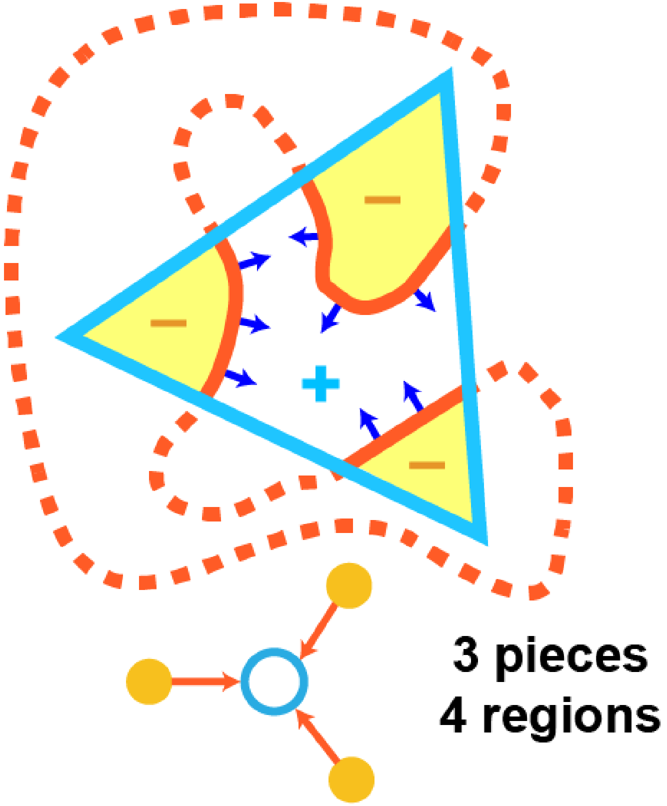
Our "Virtual Tets" method

- Build region graph
- Check orientations to find "-" regions
- Each "-" region assigned one tet copy



Our Cell Tet Meshes: Virtual Tets

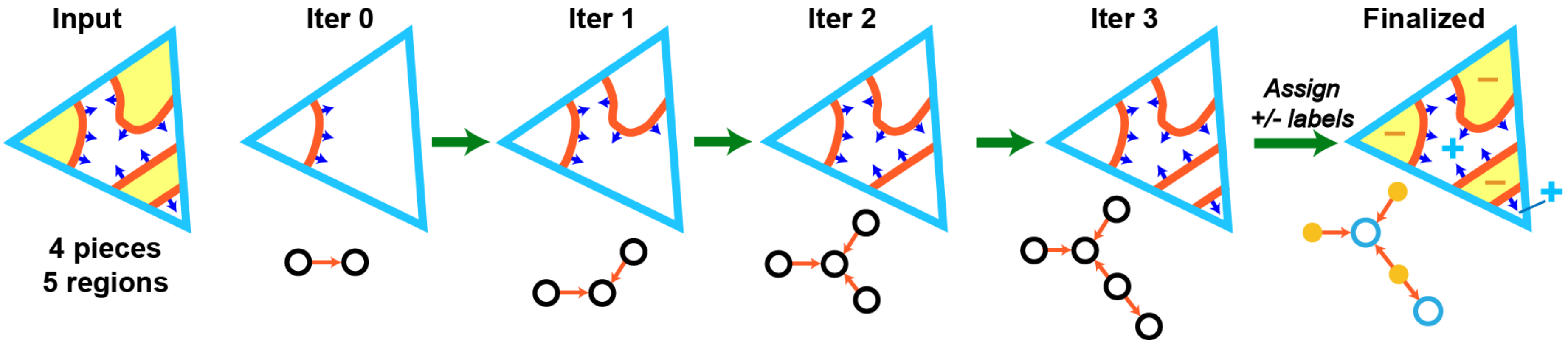
- Clip mesh against tet
- Pieces: clipped triangles
- k pieces divide tet into k+1 regions



Build region graph

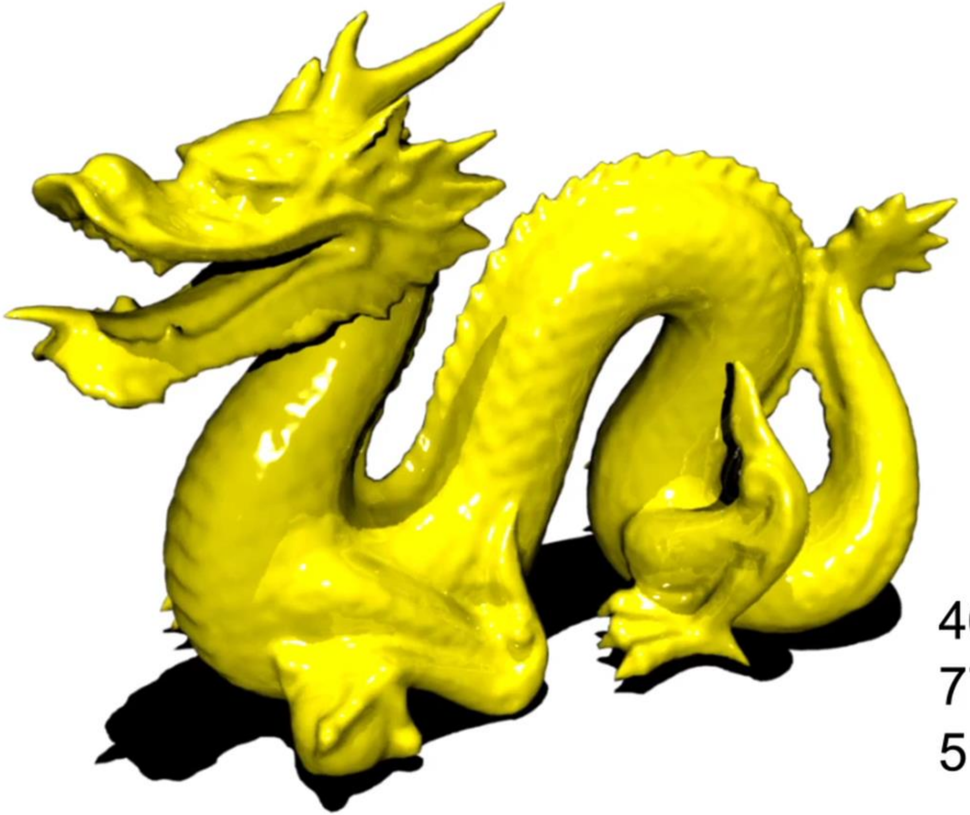
- Pseudo-normal test
  - Decide whether a region is inside/outside of a piece

- Each “-” region assigned one tet copy



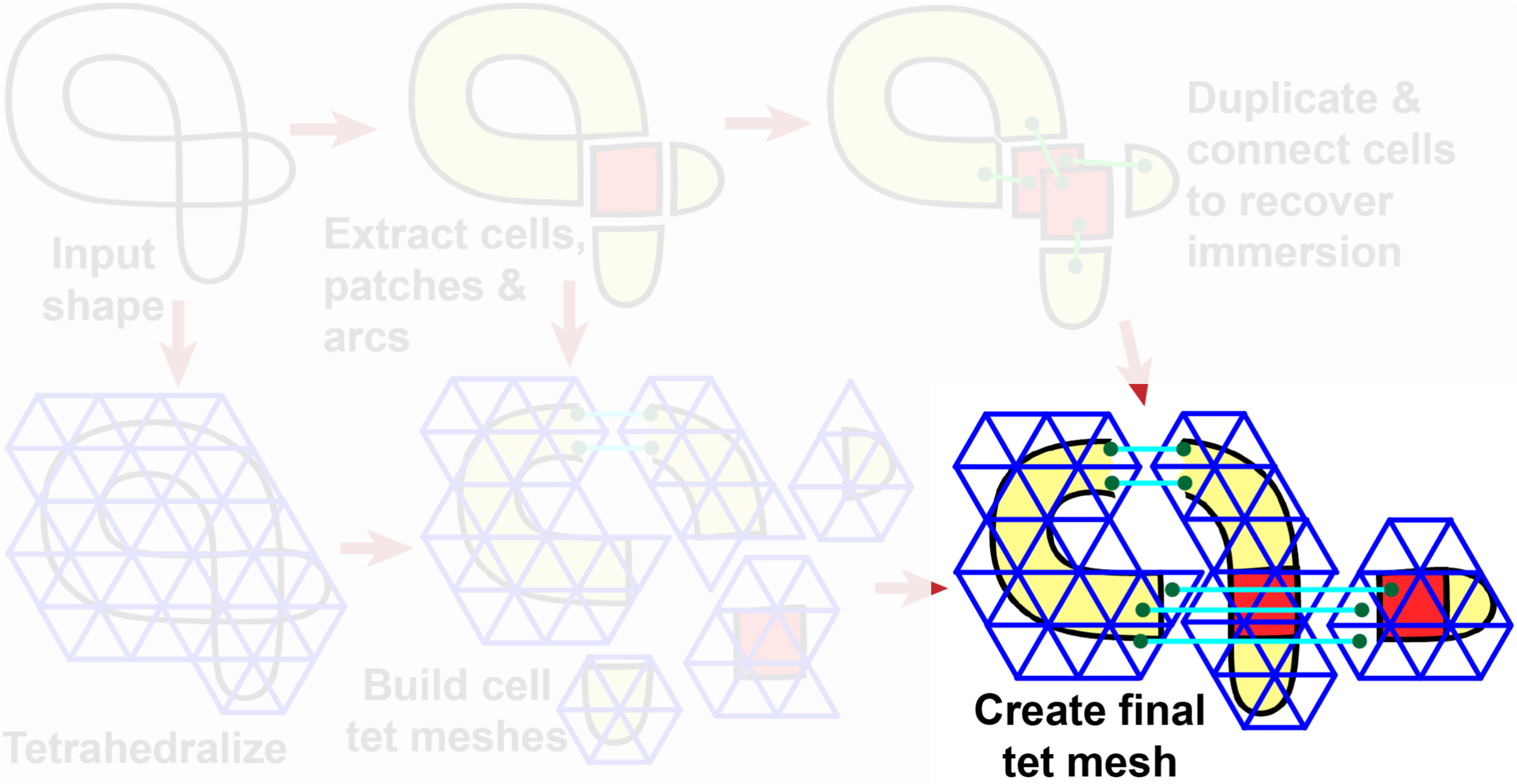
Virtual Tets Examples

Nearly self-intersecting dragon

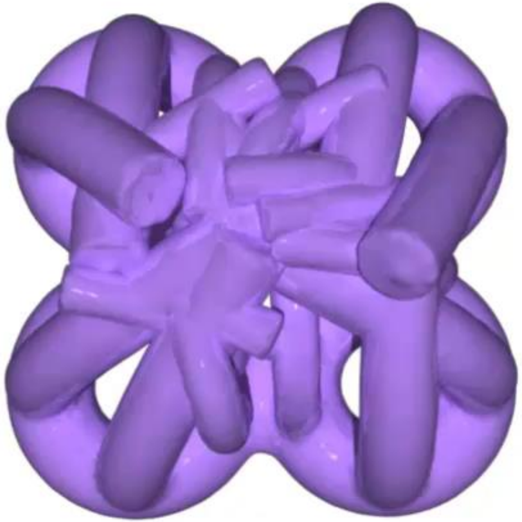


46,736 vertices
77,250 triangles
512 tets

Overview

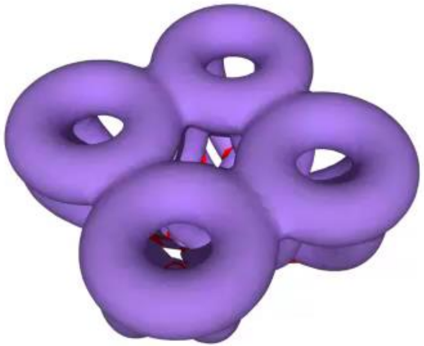

Input shape
Extract cells, patches & arcs
Duplicate & connect cells to recover immersion
Tetrahedralize
Build cell tet meshes
Create final tet mesh

Application: Volume-Based Collision Resolution

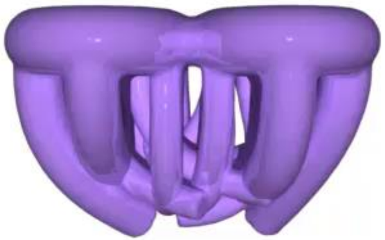


Bottom view

Elastic rest state =
input self-intersecting shape



Input shape

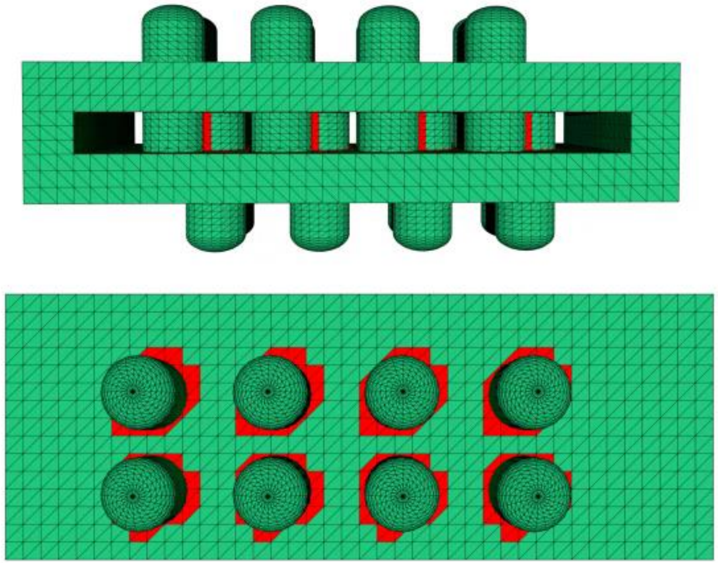


Side view

Comparison to [Sacht 2013] (we are 100x faster)

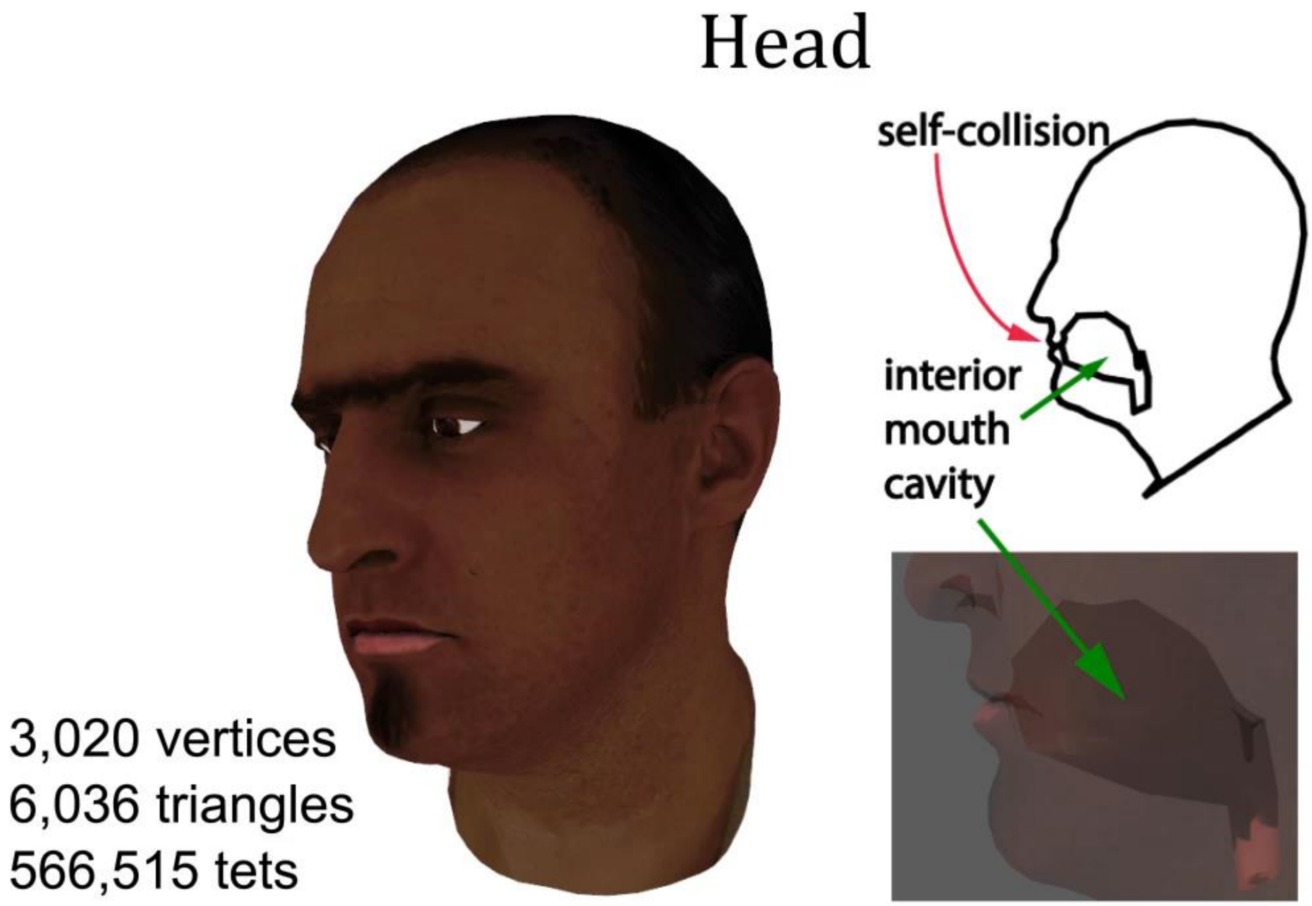
Side view

Top view



12,448 vertices
24,896 triangles
112,554 tets

Application: FEM Anatomical Simulation



Conclusion

- 3D intersection-aware volumetric meshing (also works in 2D)
- Immersion construction algorithm & proof via algebraic topology
- Fast virtual tets algorithm for nearly self-intersecting inputs

Limitation and Future Work

- Inverted shapes
- Degenerate inputs
- Mesh exact boundary of input meshes
- Non-simple immersions

Acknowledgments

- NSF (CAREER-1055035, IIS-1422869)

- USC Annenberg Fellowship to Yijing Li

- Bohan Wang, Danyong Zhao, Sitao Xiang

- Anonymous reviewers