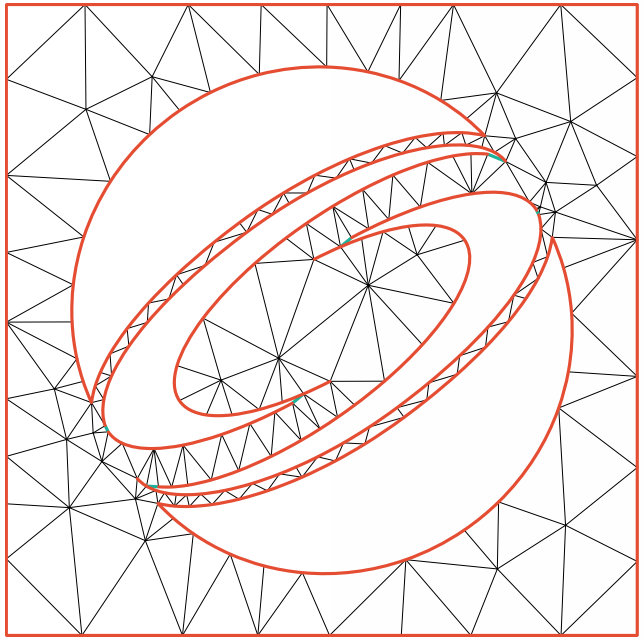
TriWild:
Robust Triangulation with Curved Constraints

Yixin Hu,[1]
Teseo Schneider, [1]
Xifeng Gao,[2]
Qingnan Zhou,[3]
Alec Jacobson,[4]
Denis Zorin,[1]
Daniele Panozzo.[1]

1 NYU COURANT INSTITUTE OF MATHEMATICAL SCIENCES
2 FLORIDA STATE UNIVERSITY
3 Adobe
4 Computer Science UNIVERSITY OF TORONTO

2D Triangulation



Input 2D Boundary

Output Triangle Mesh

Physical Simulation

Linear Triangulation

#V = 3013, #F = 4149

Want a smaller mesh?

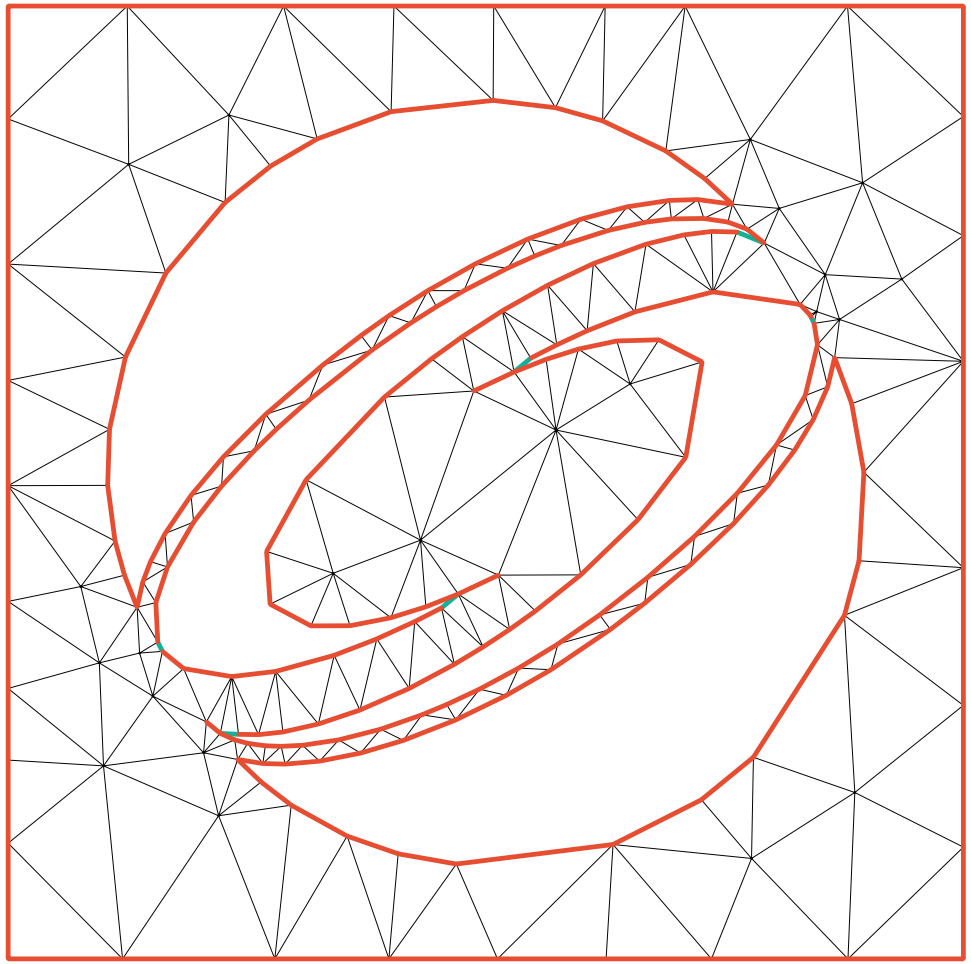
(Generated by Triangle)

Linear Triangulation

~~#V = 3013, #F = 4149~~

#V = 1503

#F = 304

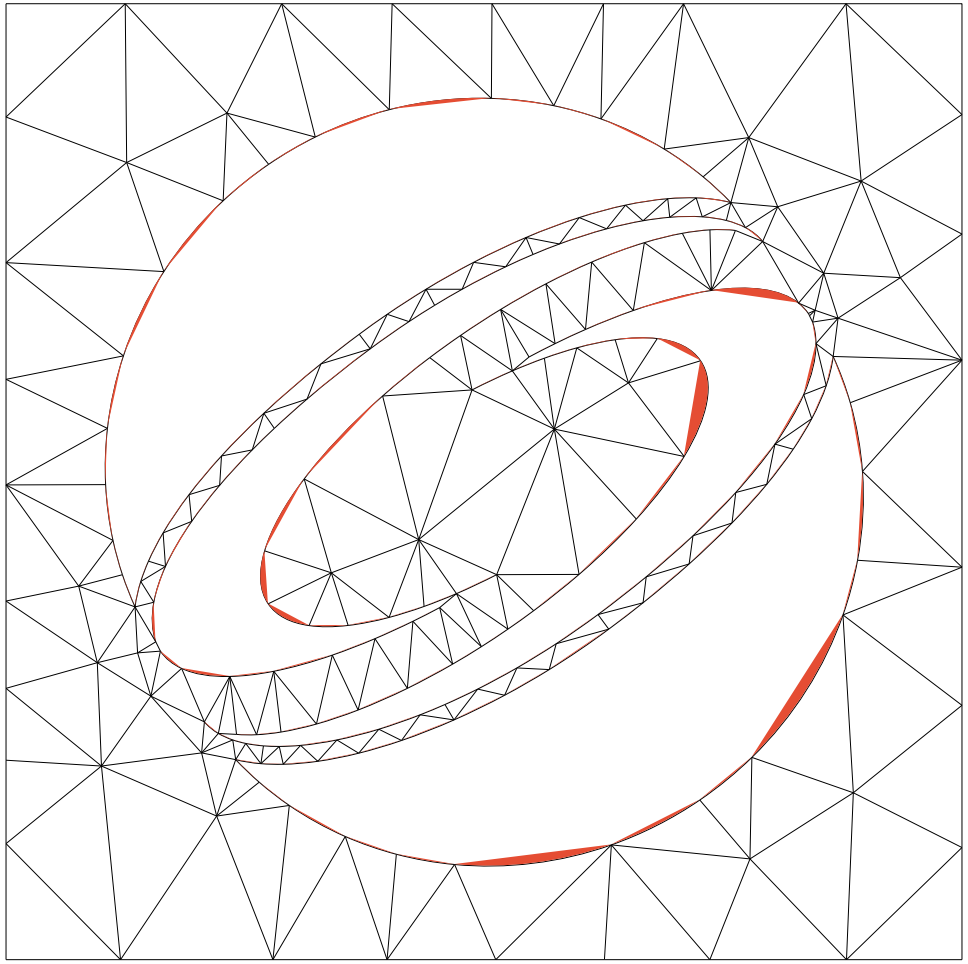


Linear Triangulation

~~#V = 3013, #F = 4149~~

#V = 1503

#F = 304

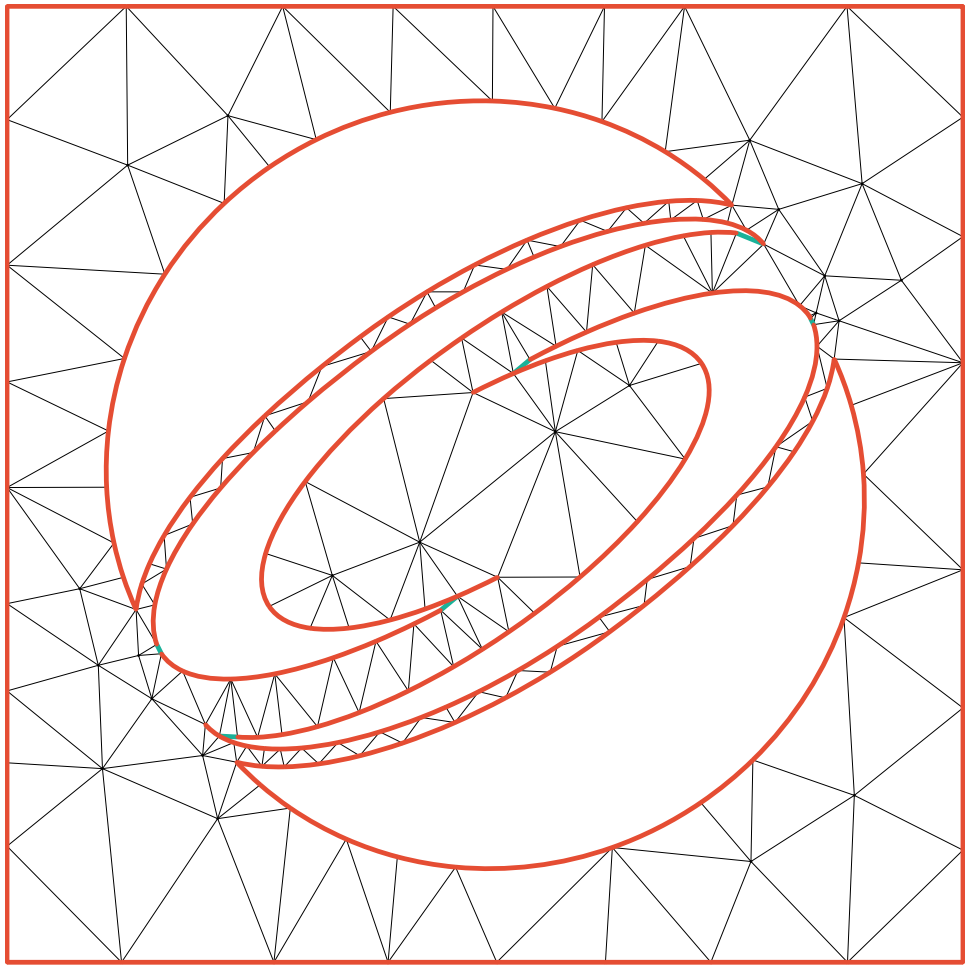


Curved Triangulation

~~#V = 3013, #F = 4149~~

#V = 1503

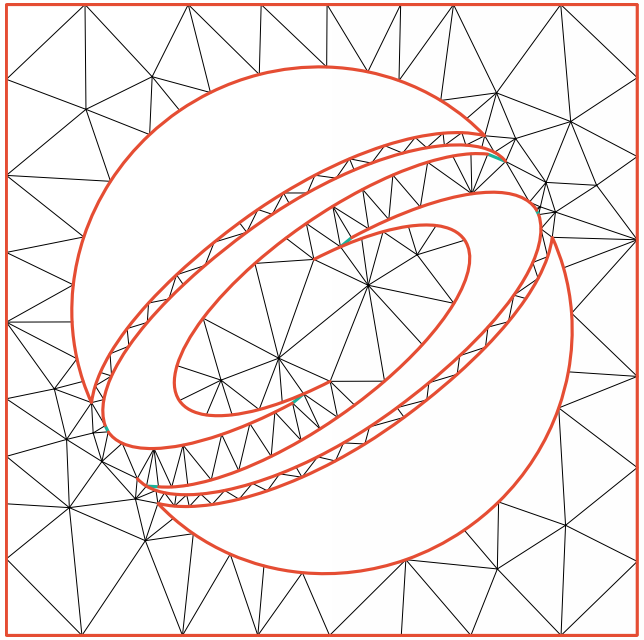
#F = 304



(Generated by TriWild)

Curved mesh!

2D Triangulation



Coarser → Faster

Input 2D Boundary

Output Triangle Mesh

Physical Simulation

Conforming → Accurate

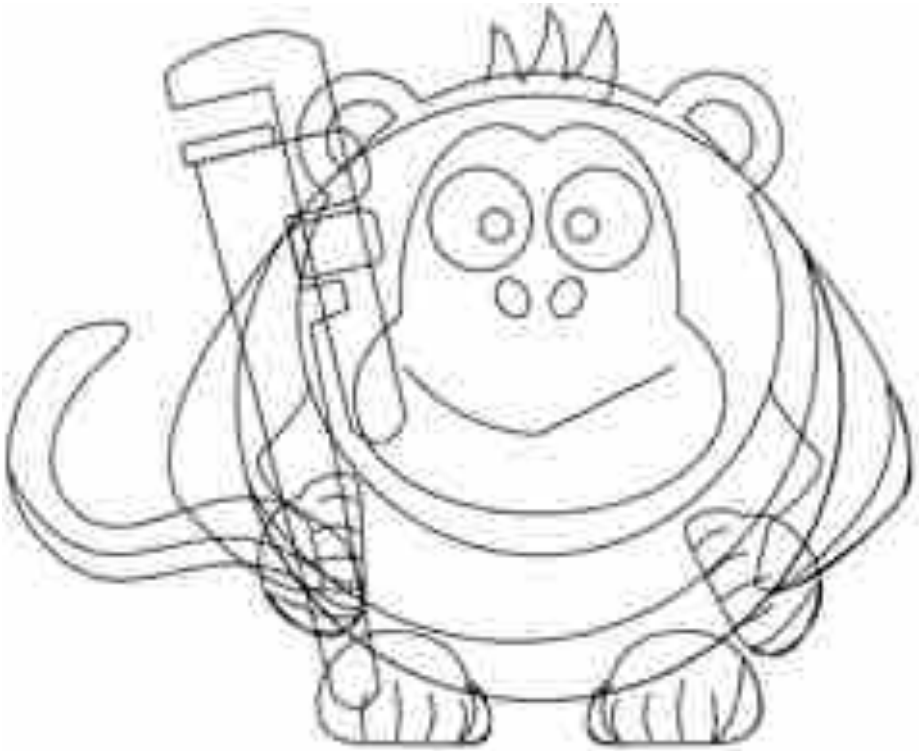
Existing Methods

- **Curved triangle mesh is not a new concept.**
  - Introduced in late 80s.
  - Haven’t been tested on large-scale dataset.
  - Not widely used because no robust tools.

Tangled Curved Constraints

Superposed Curves

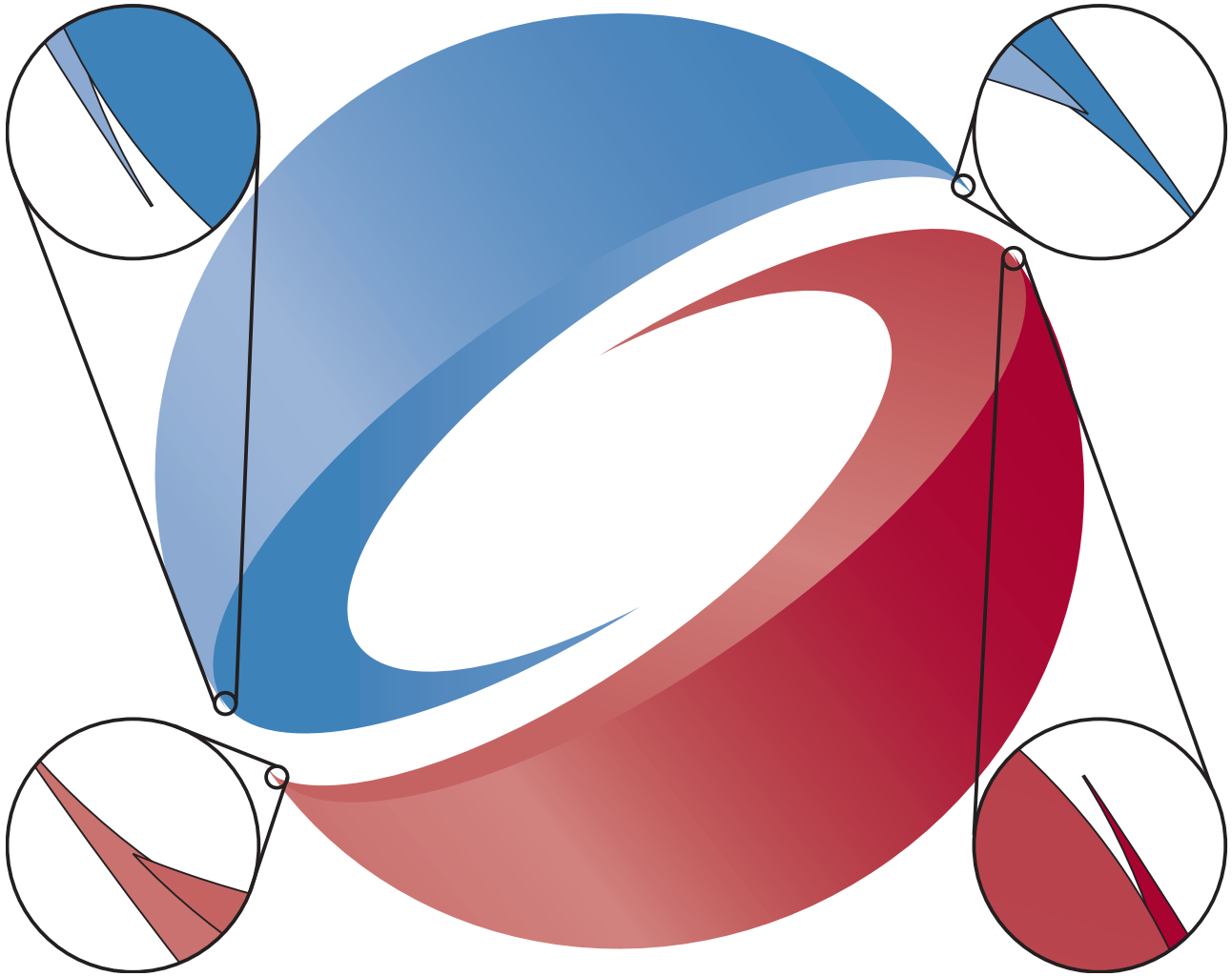
Tangled Curved Constraints

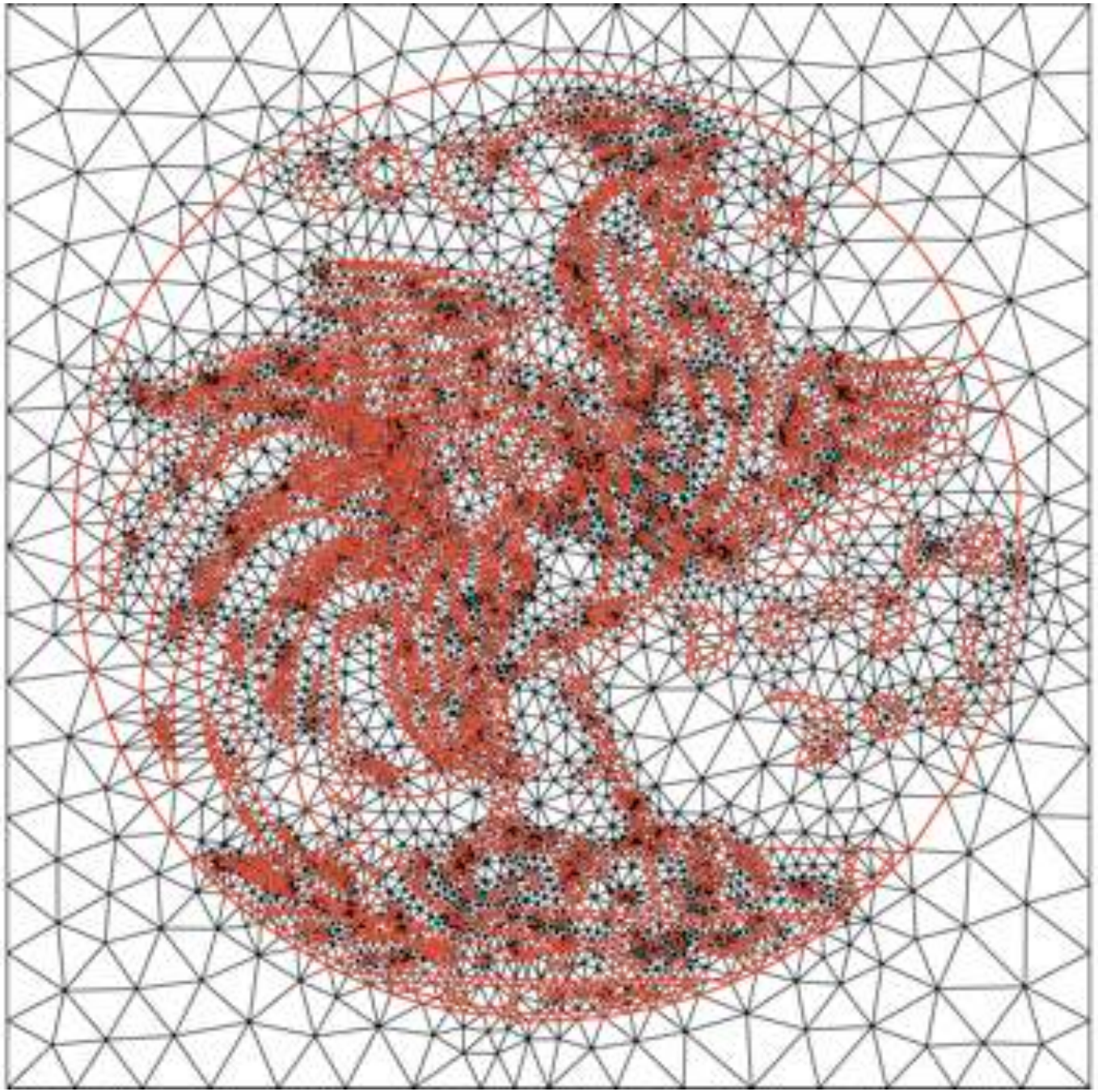


Intersected Curves

Tangled Curved Constraints

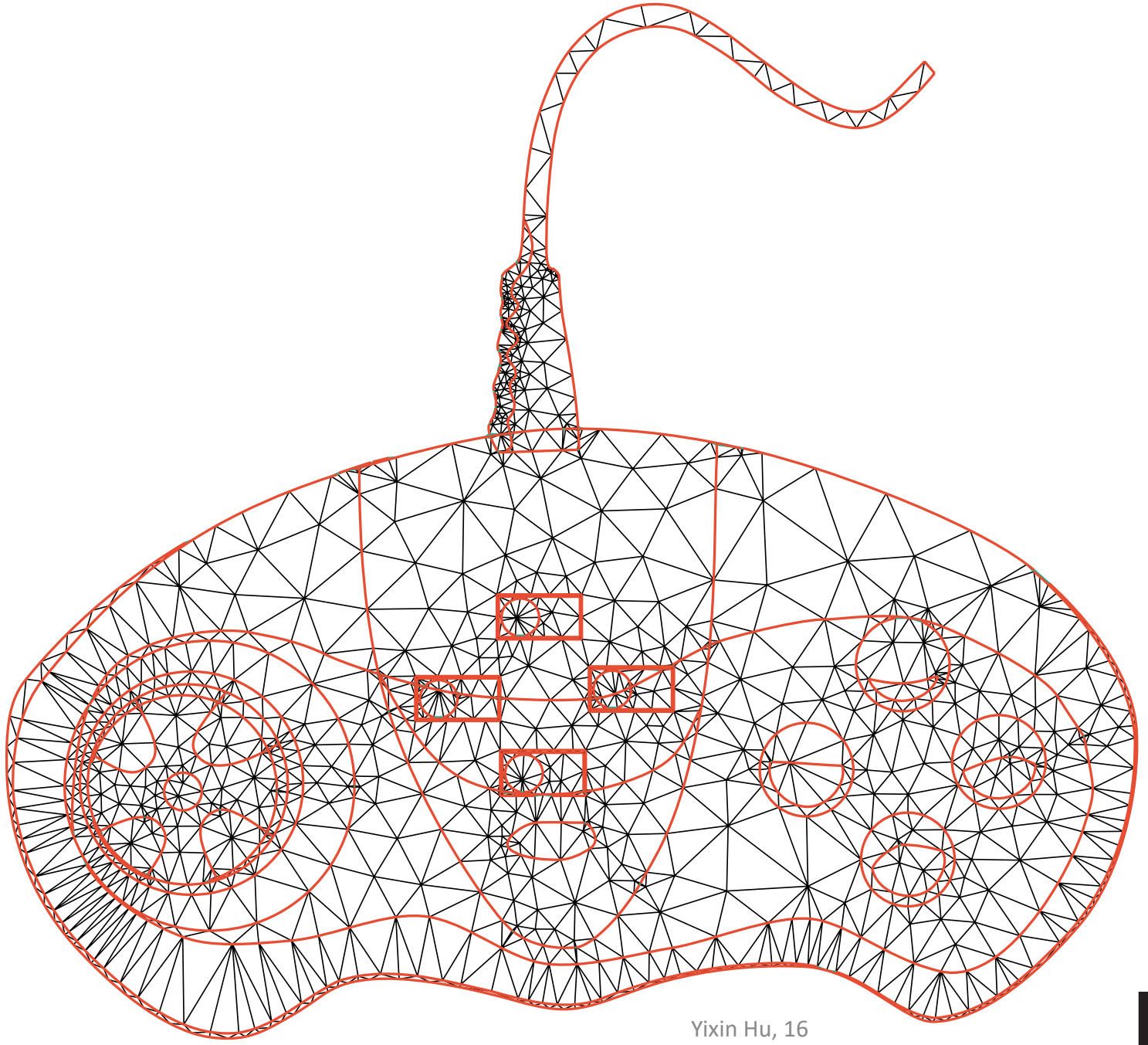
Tangled Curved Constraints





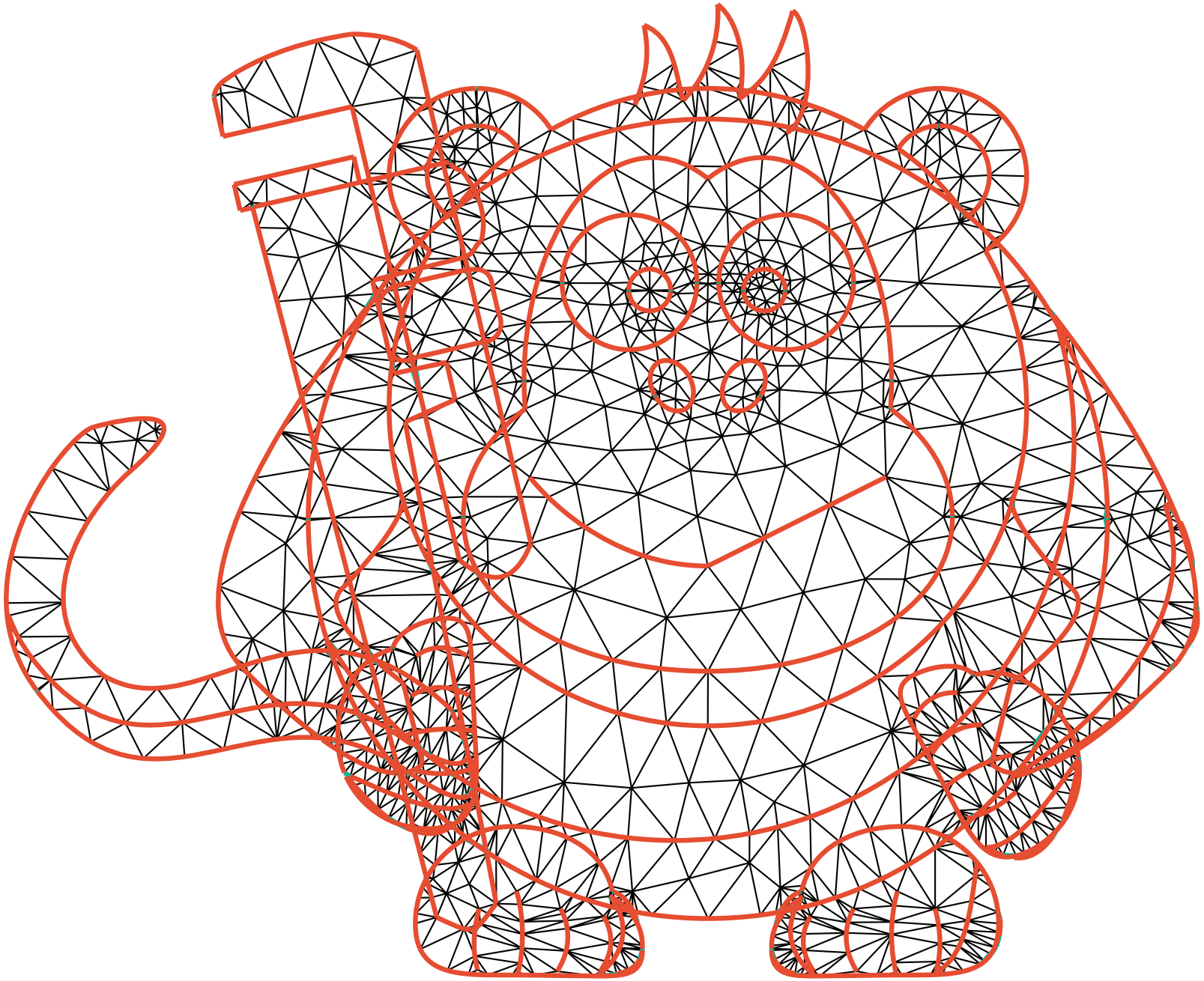
Input:

(Generated by TriWild)



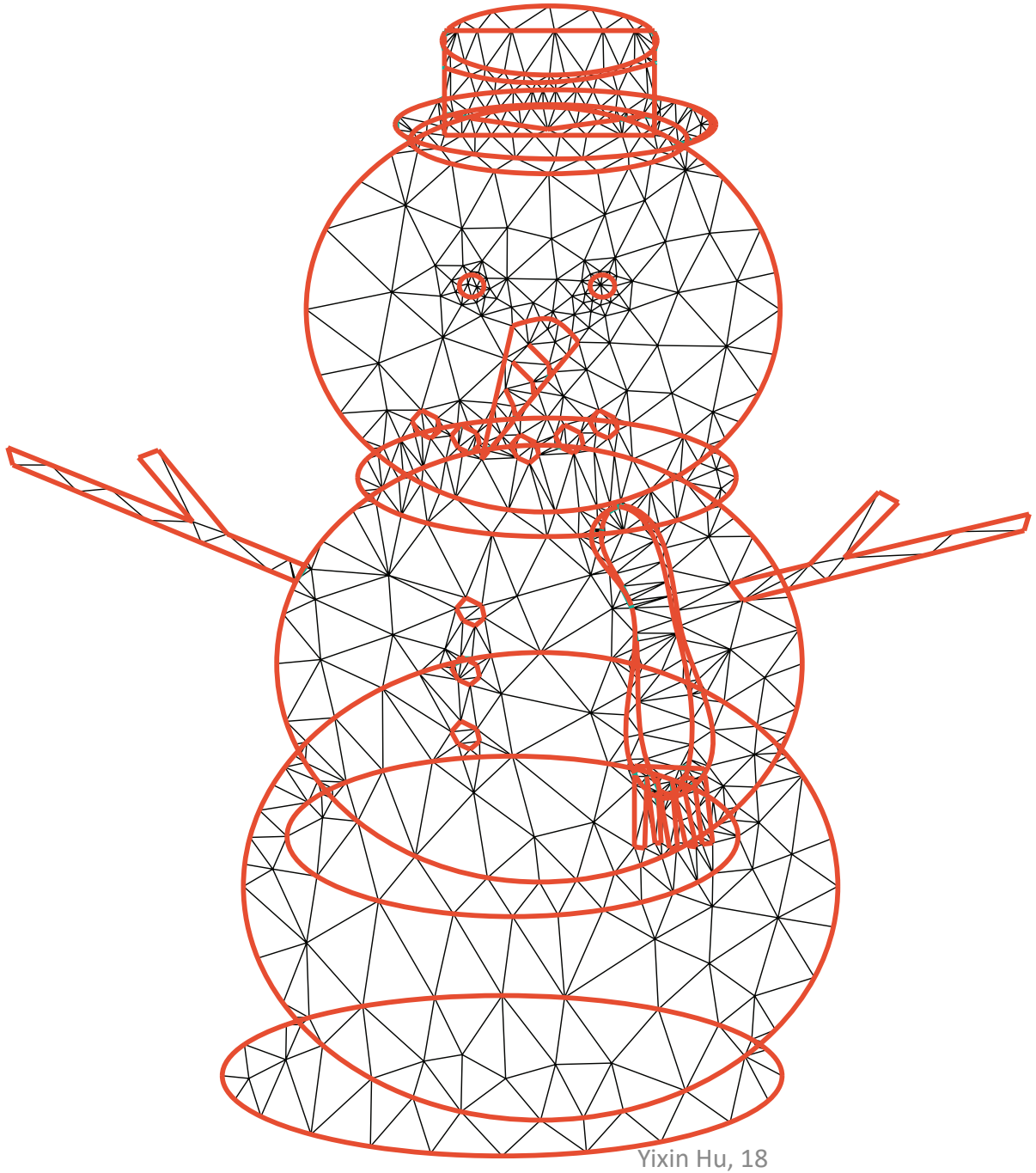
Input:

(Generated by TriWild)



Input:

(Generated by TriWild)

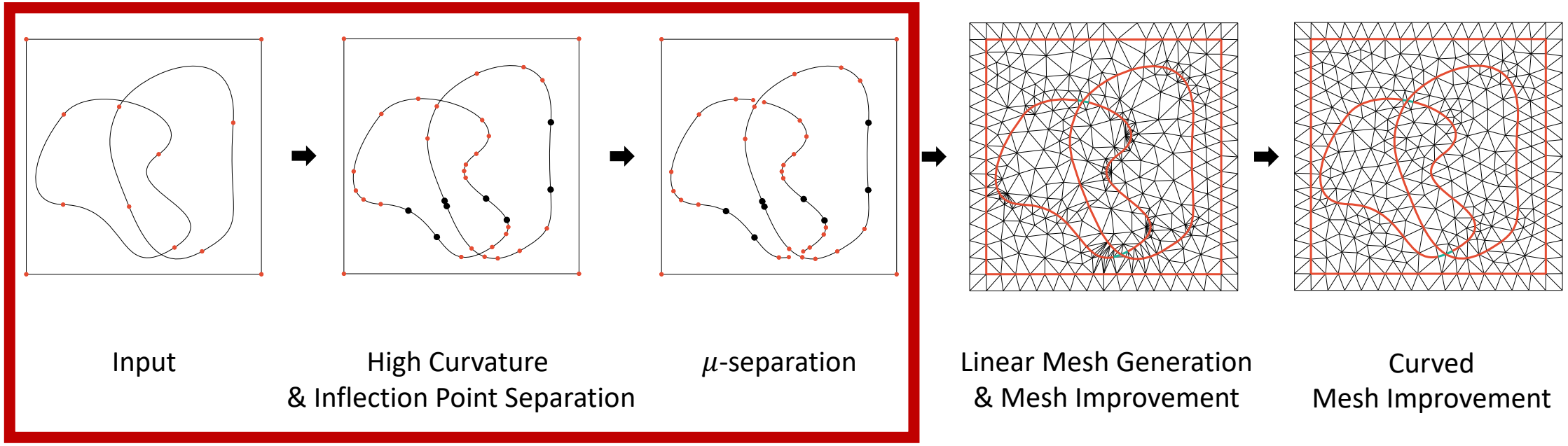


Input:

(Generated by TriWild)

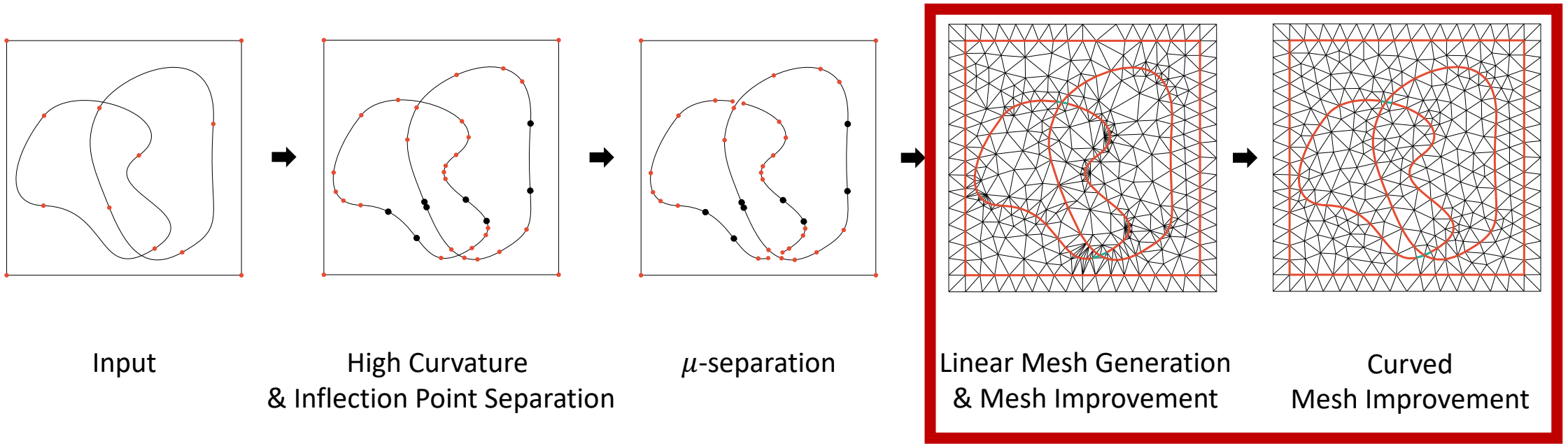
Why TriWild?

"Cleanup" the input curves.



Why TriWild?

Linear mesh for easier curving.



Input

High Curvature
& Inflection Point Separation

μ-separation

Linear Mesh Generation
& Mesh Improvement

Curved
Mesh Improvement

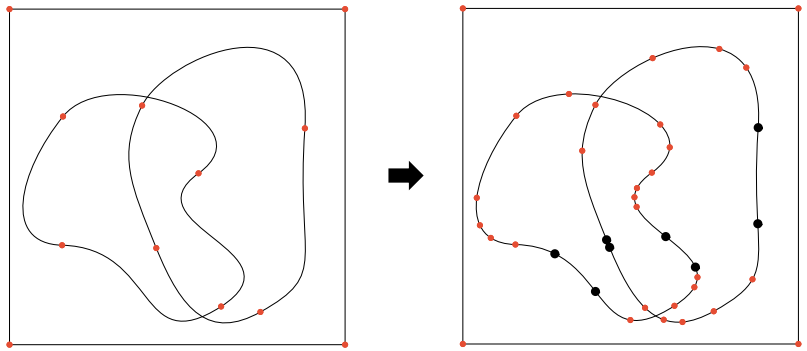
Pipeline - Input & Output

- Input:
  - Bezier curves. (SVG standard, degree 3)
- Output:
  - Curved triangle mesh
    - Conforms to primary features.
    - Approximates secondary features up to ϵ.

Primary features

Secondary features

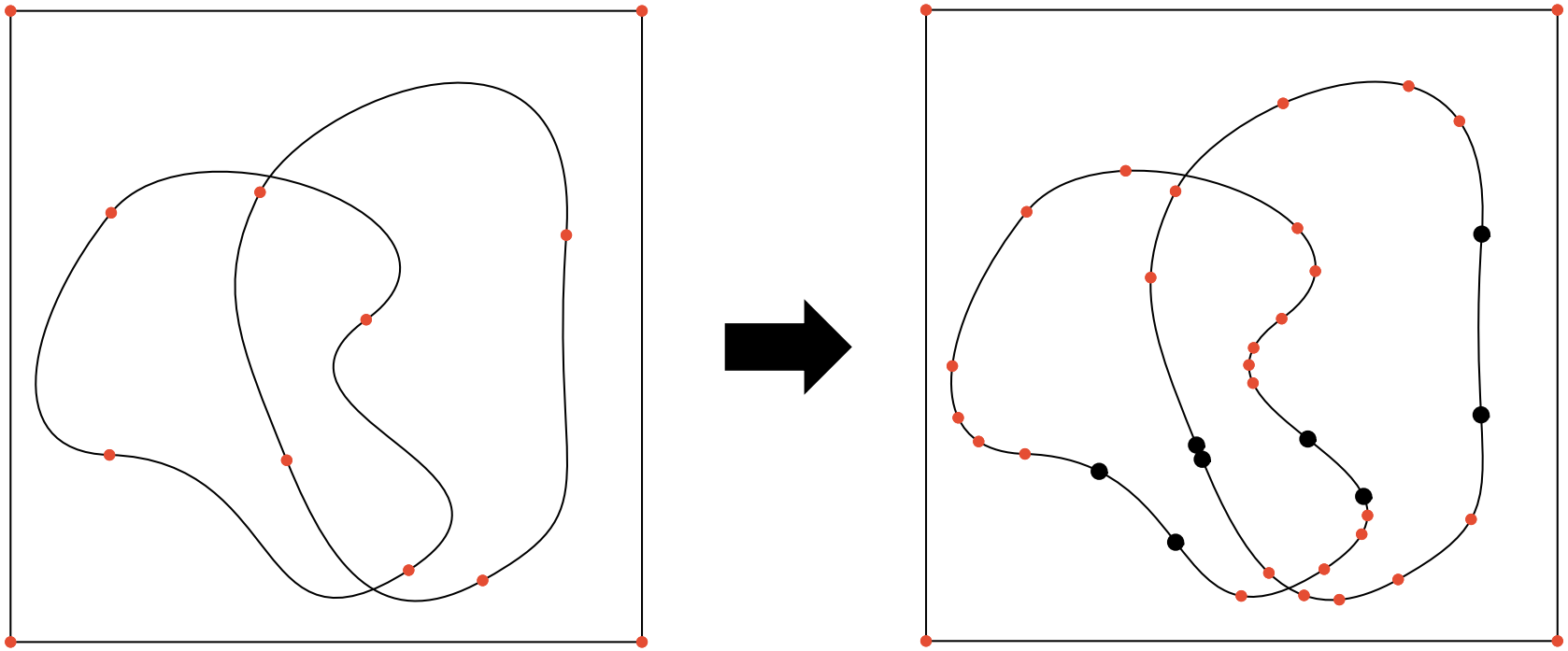
Pipeline - Feature Preprocessing



Input

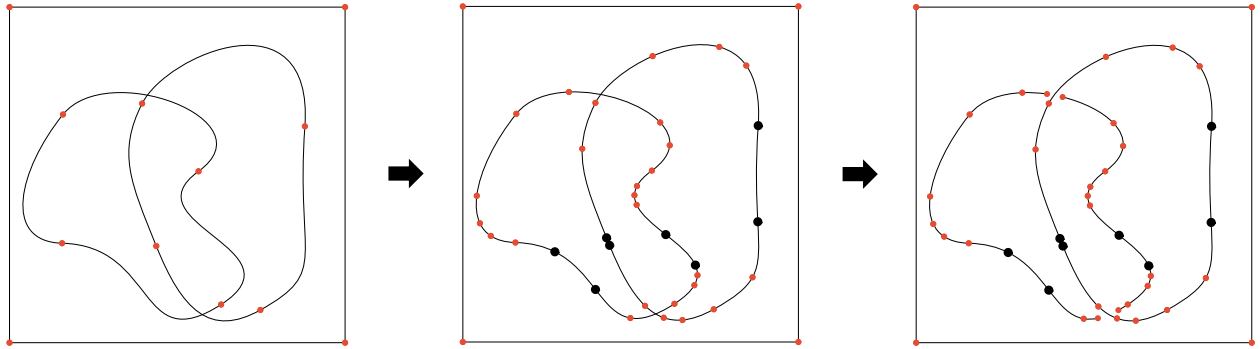
High Curvature
& Inflection Point Separation

Pipeline - Feature Preprocessing



High Curvature & Inflection Point Separation

Pipeline - Feature Preprocessing

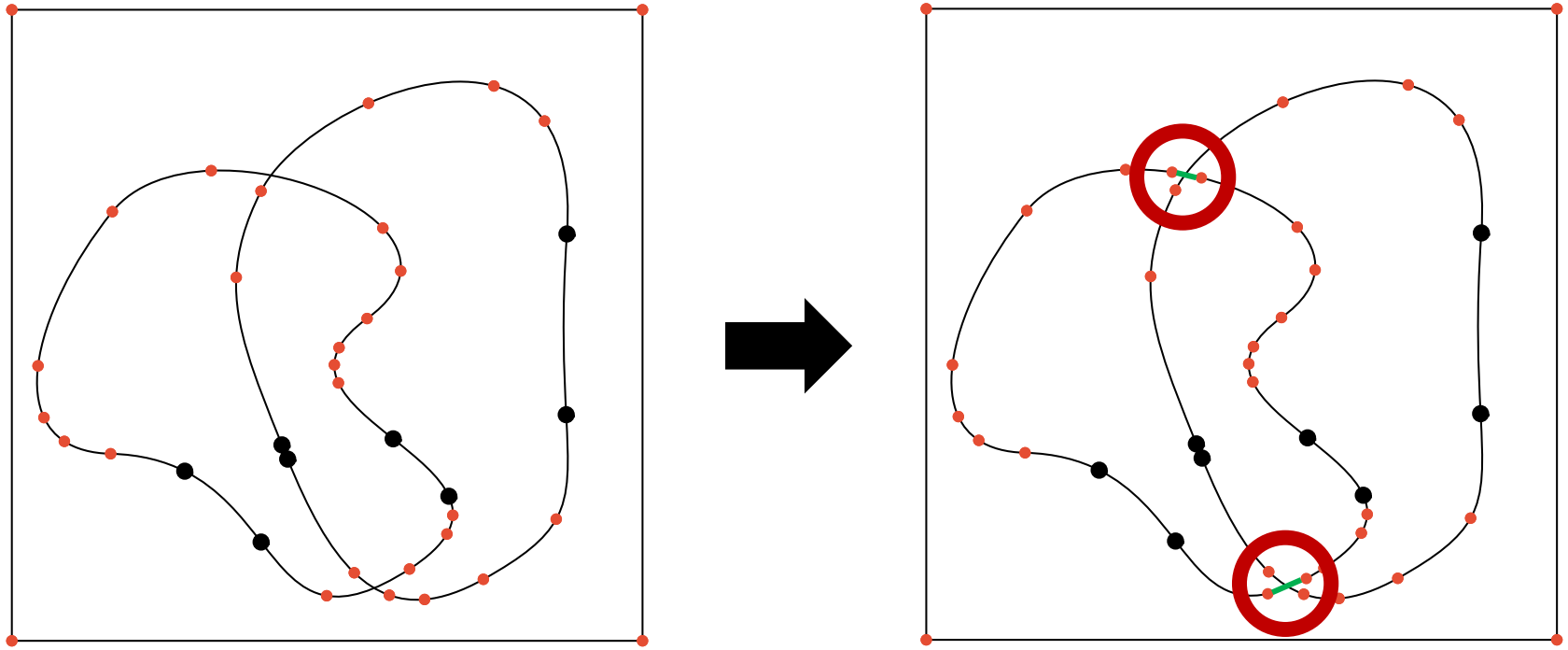


Input

High Curvature
& Inflection Point Separation

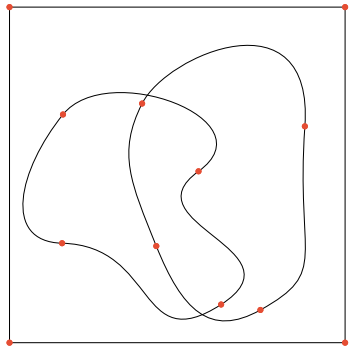
μ-separation

Pipeline - Feature Preprocessing



μ-separation

Pipeline



Input

➡

& I

Tetrahedral Meshing in the Wild

YIXIN HU, New York University
QINGNAN ZHOU, Adobe Research
XIFENG GAO, New York University
ALEC JACOBSON, University of Toronto
DENIS ZORIN, New York University
DANIELE PANOZZO, New York University

ACM Reference Format:

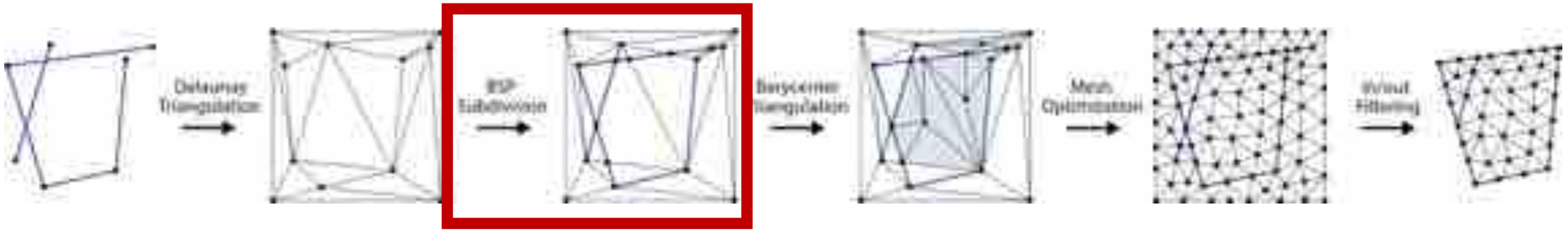
1 INTRODUCTION

Triangulating the interior of a shape is a fundamental subroutine in 2D and 3D geometric computation.

Recap - Linear Mesh Generation (TetWild)

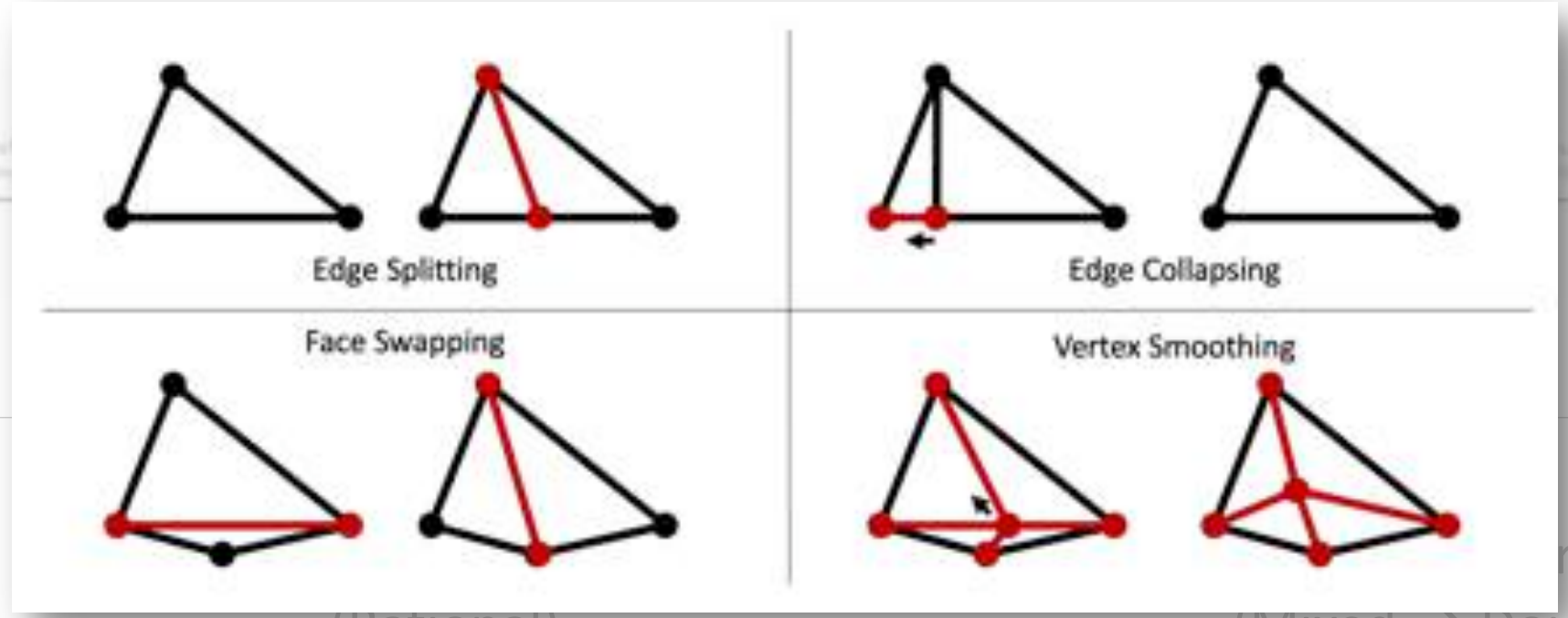
Segment Soup

Linear Triangle Mesh



Stage I: Valid Mesh Generation (Rational)

Recap - Linear Mesh Generation (TetWild)

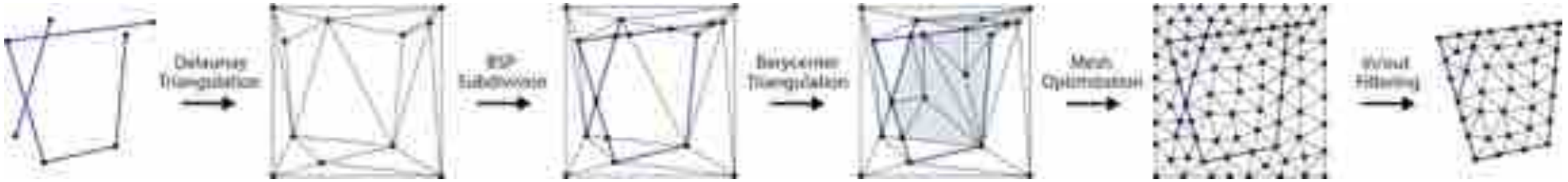


(Rational)

rovement
(Mixed → Double)

Recap - Linear Mesh Generation (TetWild)

100% success rate on 10,000 real-world models.

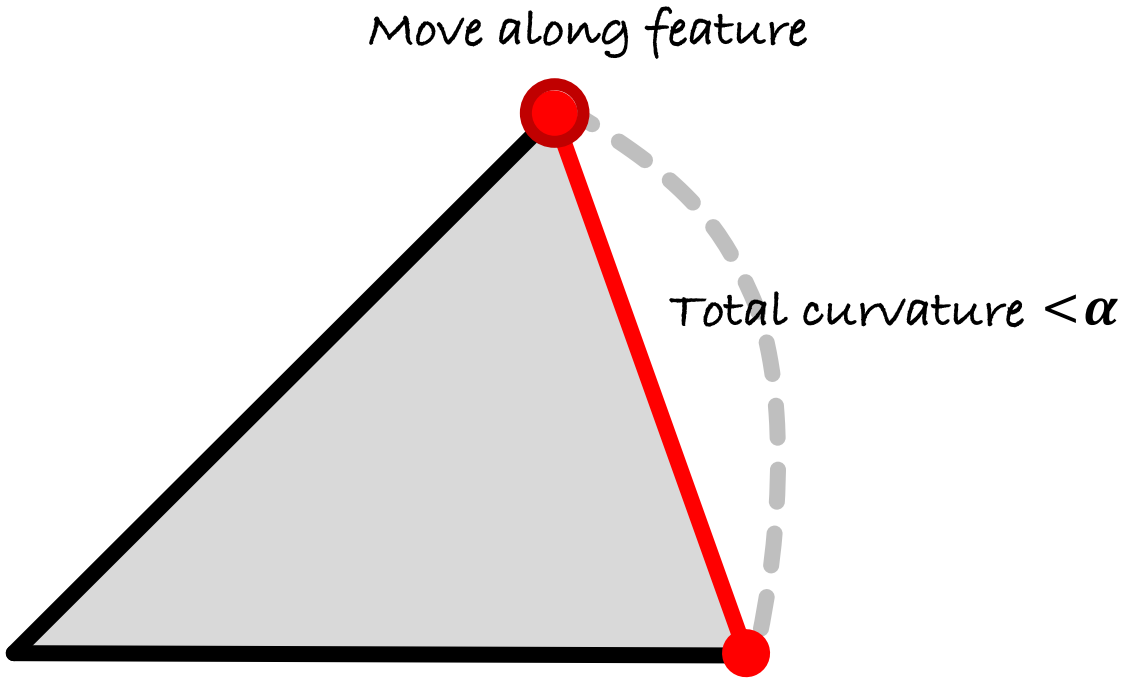


Stage I: Valid Mesh Generation (Rational)

Stage II: Mesh Improvement (Mixed → Double)

Pipeline - Linear Mesh Generation

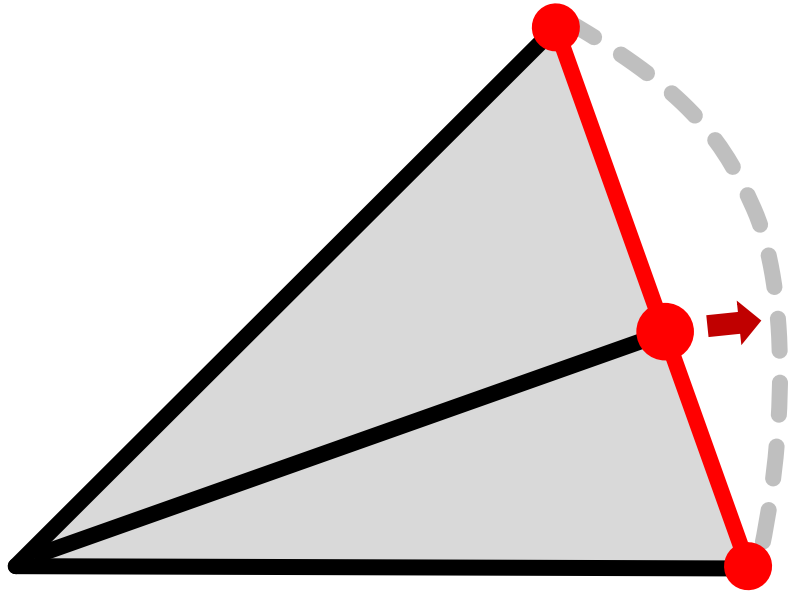
- Feature Invariant:
  - Feature edges' associated curve has total curvature $<\alpha$.
  - Feature vertices can only move along feature curves.



"feature" == "primary feature"

Pipeline - Linear Mesh Generation

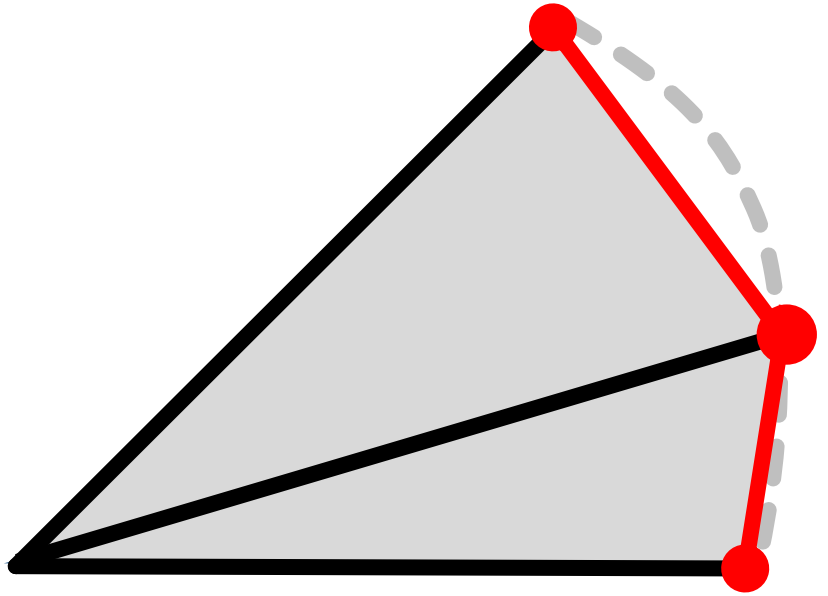
- Feature Invariant:
  - Feature edges' associated curve has total curvature $<\alpha$.
  - Feature vertices can only move along feature curves.



- Vertex Projection

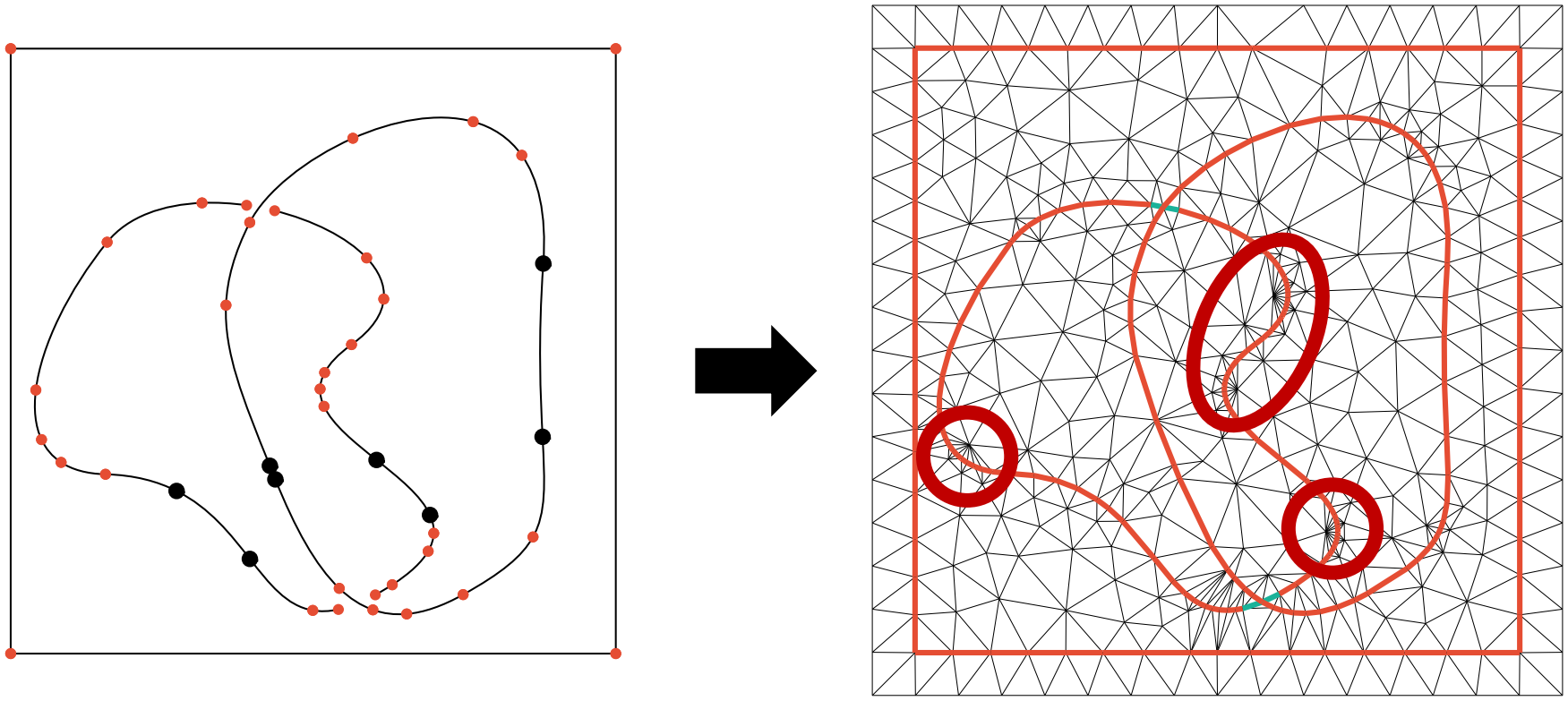
Pipeline - Linear Mesh Generation

- Feature Invariant:
  - Feature edges' associated curve has total curvature $<\alpha$.
  - Feature vertices can only move along feature curves.



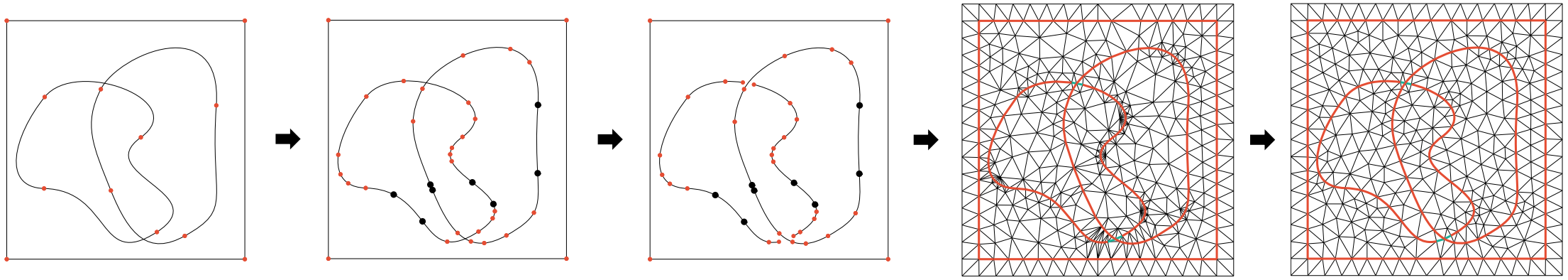
- Vertex Projection

Pipeline - Curved Mesh Generation



Linear Mesh Generation & Mesh Improvement

Pipeline - Curved Mesh Generation



Input

High Curvature
& Inflection Point Separation

μ-separation

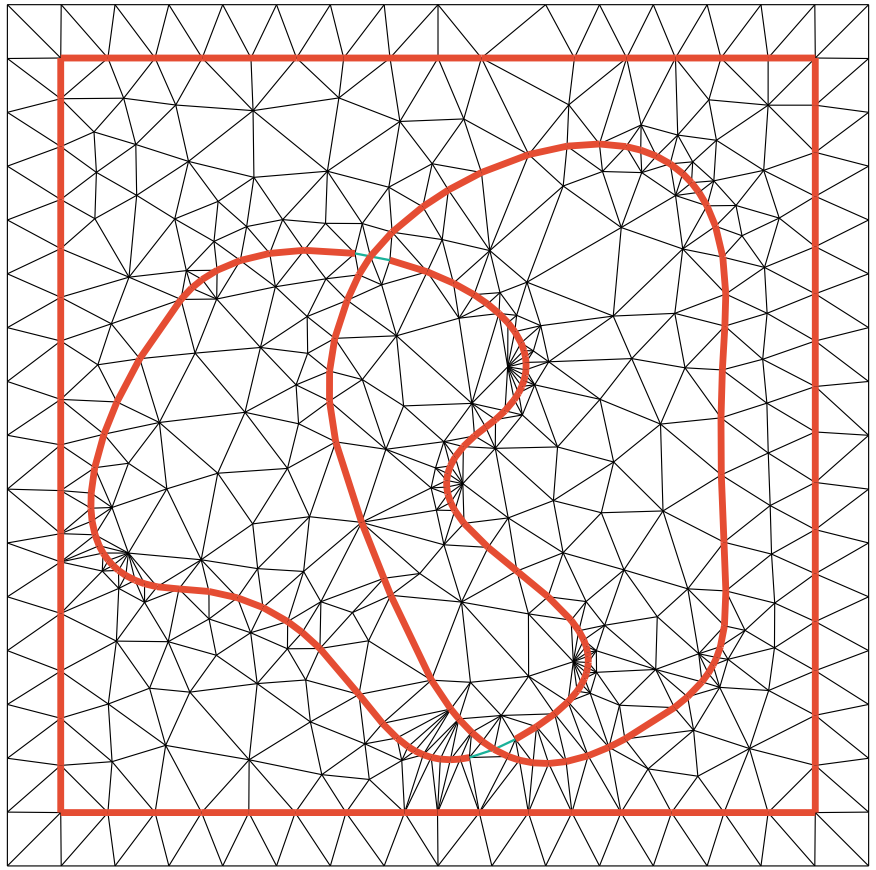
Linear Mesh Generation
& Mesh Improvement

Curved
Mesh Improvement

Pipeline - Curving

- Curving:
  - All feature elements
    - With primary feature edges

      (thickened in red)



Pipelir

- Inversion

The Generation of Valid Curvilinear Meshes

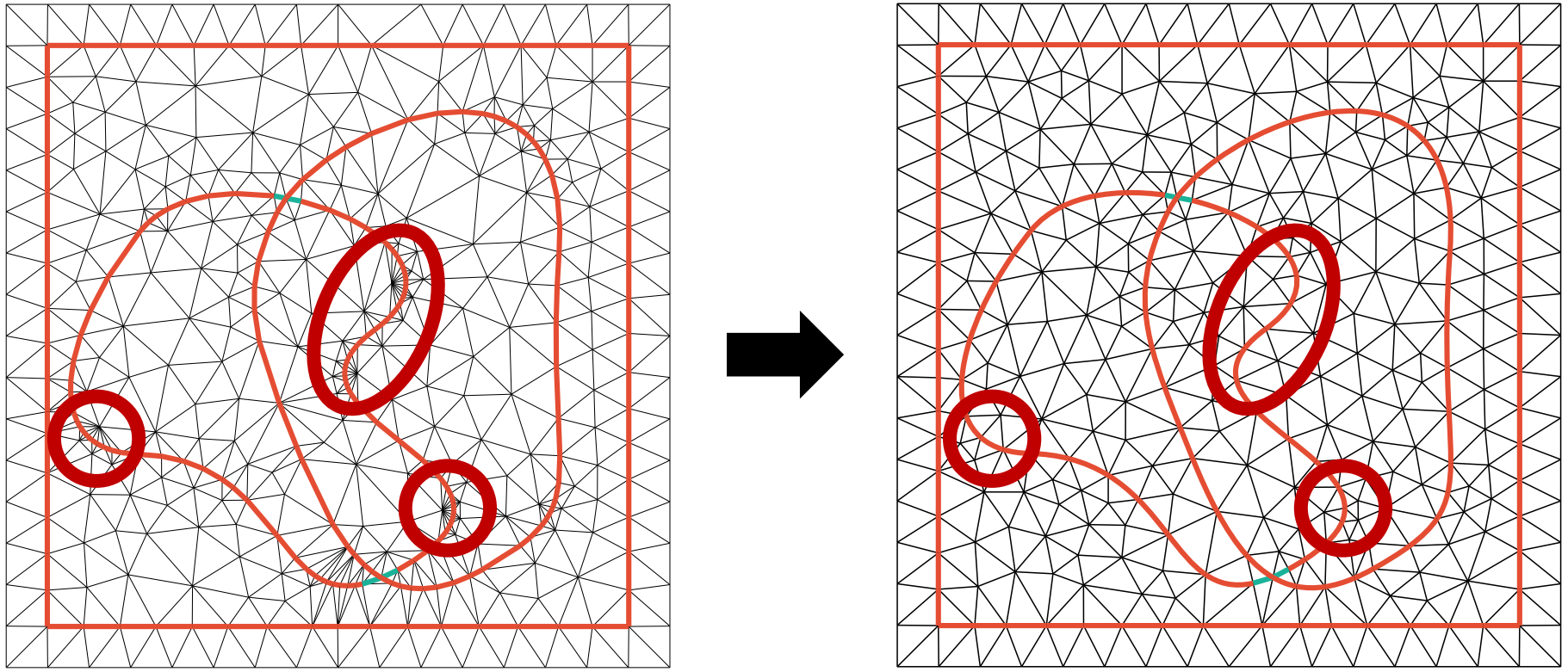
Christophe Geuzaine[1], Amaury Johnen[1],
Jonathan Lambrechts[2,3], Jean-François Remacle[2], Thomas Toulorge[2,3]

[1] Université de Liège, Dept. of Electrical Engineering and Computer Science, B-4000 Liège, Belgium
{a.johnen,cgeuzaine}@ulg.ac.be,
[2] Université catholique de Louvain, Institute of Mechanics, Materials and Civil Engineering (iMMC), B-1348 Louvain-la-Neuve, Belgium
{thomas.toulorge,jonathan.lambrechts,jean-francois.remacle}@uclouvain.be,
[3] Fonds National de la Recherche Scientifique, B-1000 Brussels, Belgium

Abstract. It is now well-known that a curvilinear discretization of the geometry is most often required to benefit from the computational efficiency of high-order numerical schemes in simulations. In this article, we explain how appropriate curvilinear meshes can be generated. We pay particular attention to the problem of invalid (tangled) mesh parts created by curving the domain boundaries. An efficient technique that computes provable bounds on the element Jacobian determinant is used to characterize the mesh validity, and we describe fast and robust techniques to regularize the mesh. The methods presented in this article are thoroughly discussed in Ref. [1,2], and implemented in the free mesh generation software Gmsh [3,4].

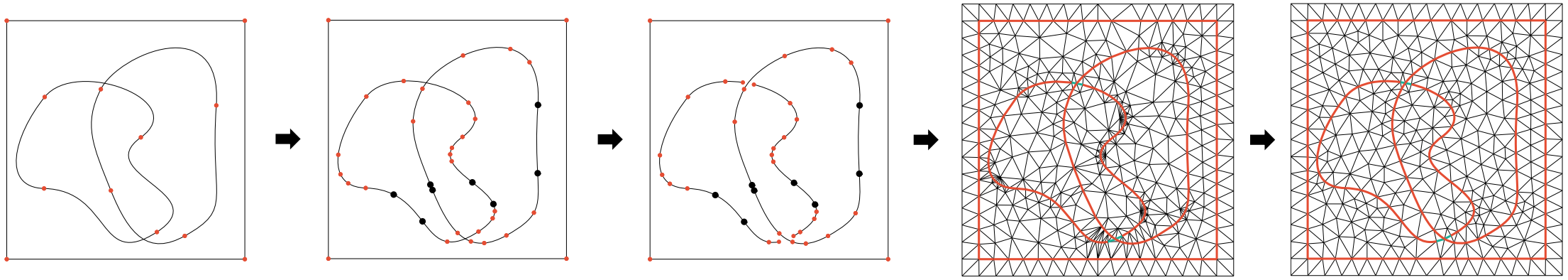
Keywords: High-order mesh, curvilinear mesh, geometry discretization, mesh validity, element Jacobian

Pipeline - Curved Mesh Generation



Curved Mesh Improvement

Pipeline - Curved Mesh Generation



Input

High Curvature
& Inflection Point Separation

μ-separation

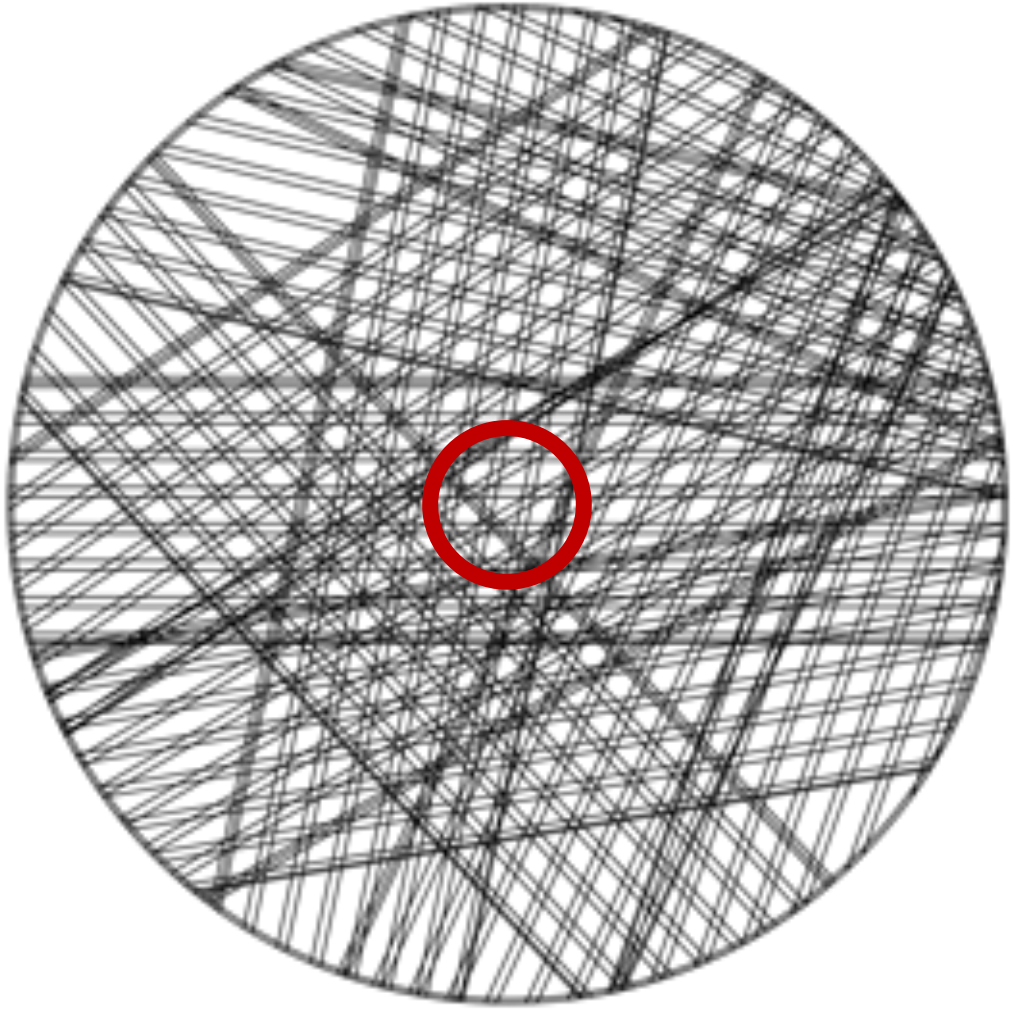
Linear Mesh Generation
& Mesh Improvement

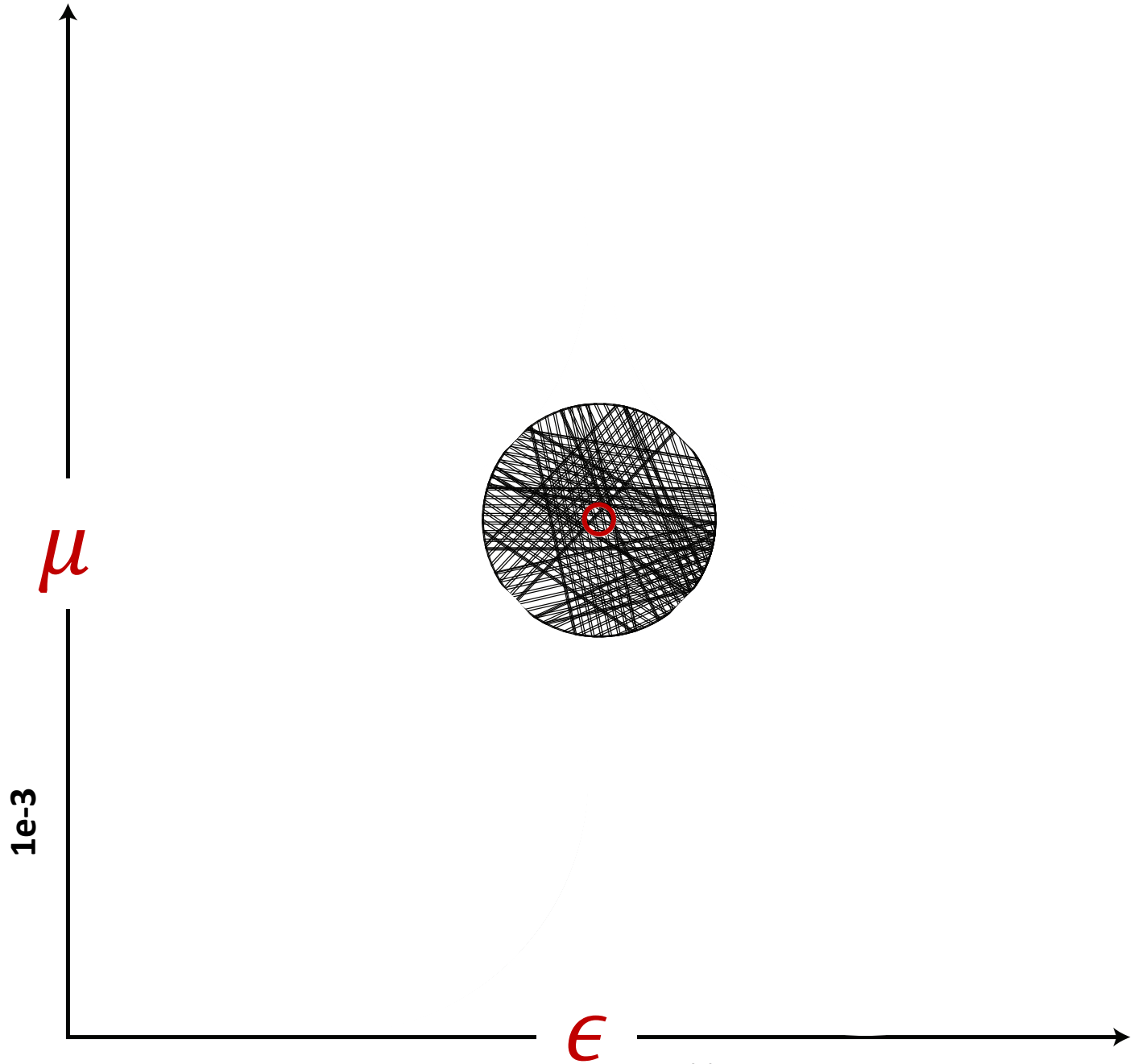
Curved
Mesh Improvement

Parameters – Envelope Size

- μ
  - Used in μ-separation for separating primary features and secondary features.
- ϵ (same parameter used in TetWild)
  - Maximum distant that secondary feature edges can deviate from input.

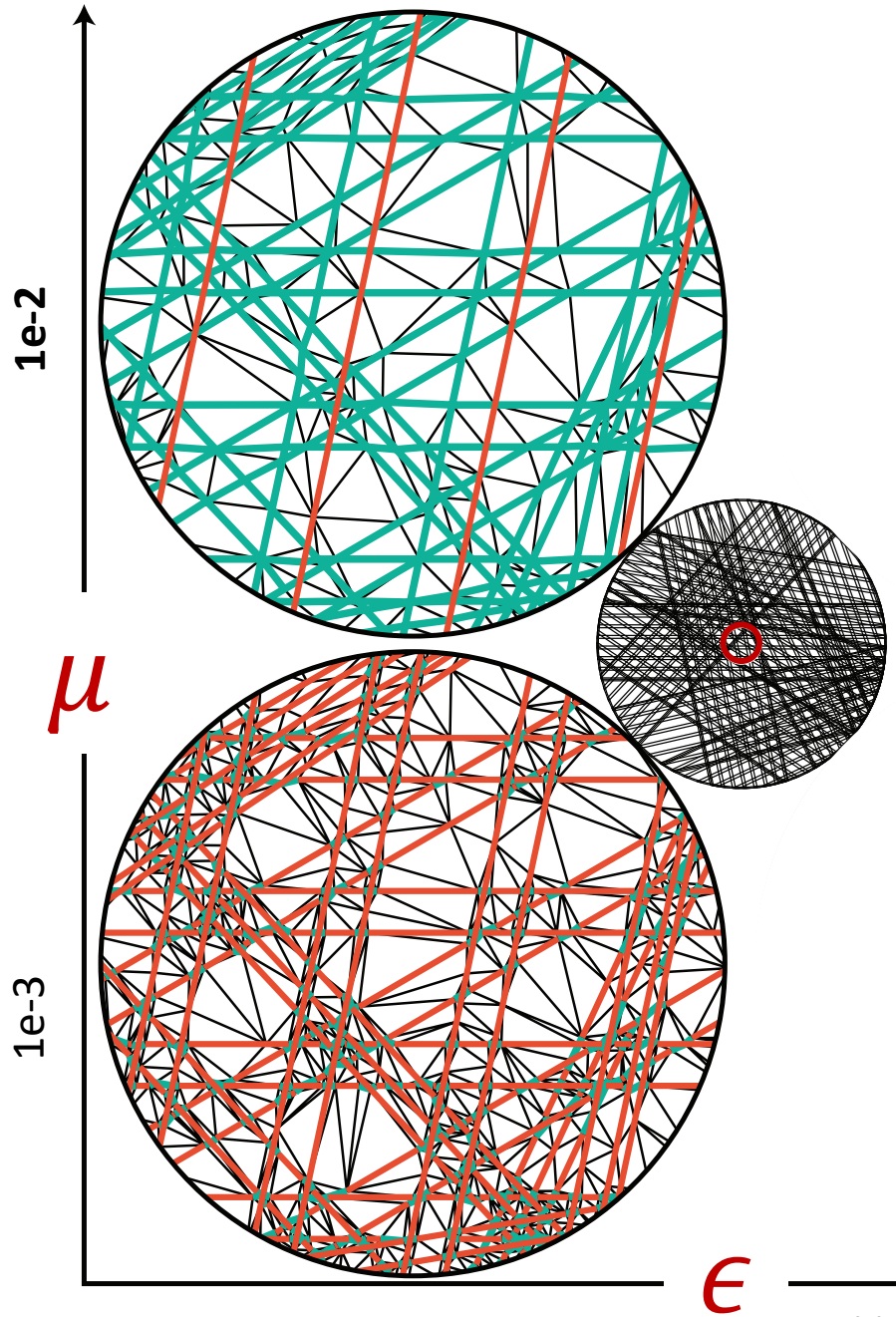
Parameters – Envelope Size





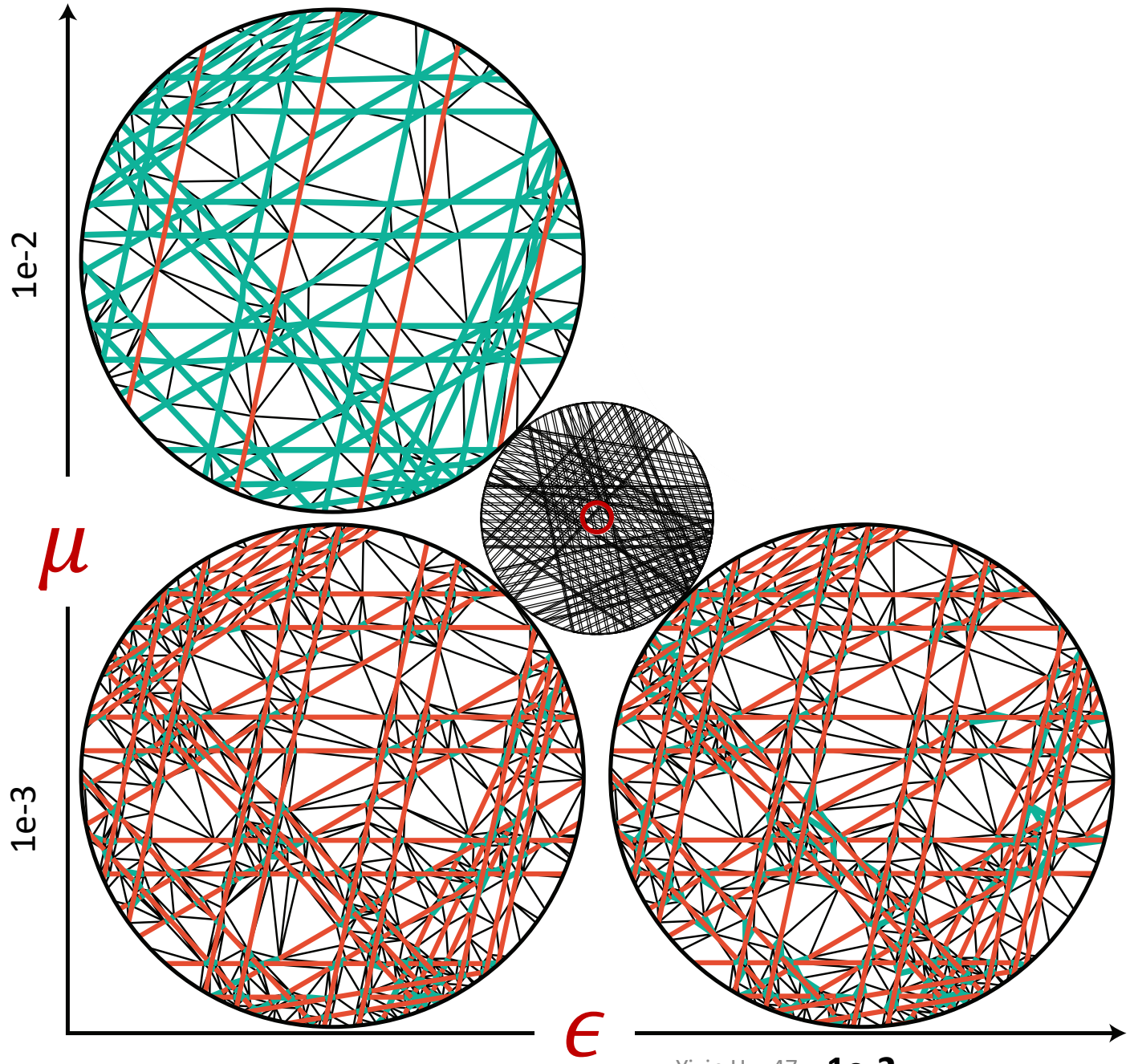
Red:
primary feature edges

Green:
secondary feature edges



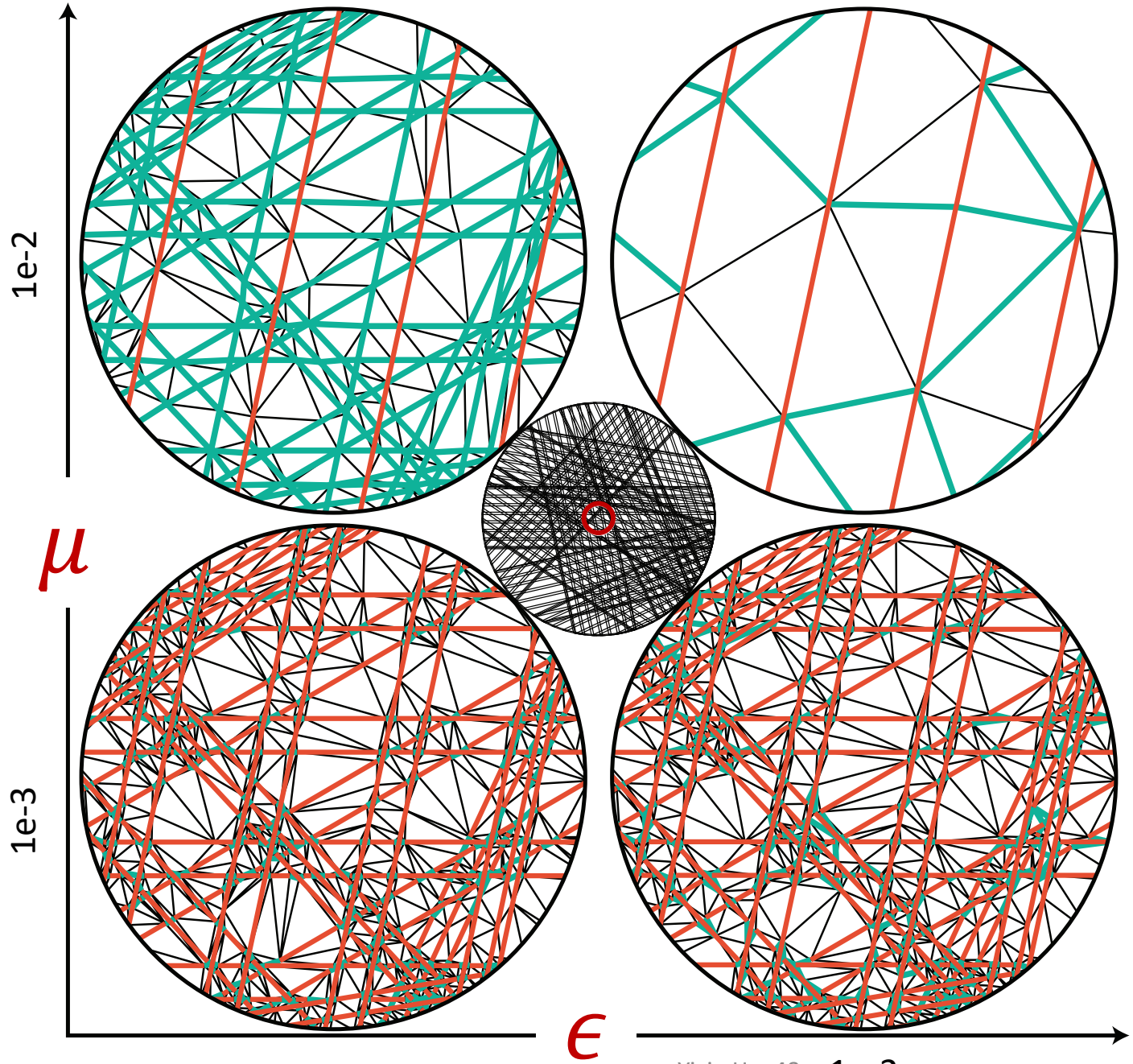
Red:
primary feature edges

Green:
secondary feature edges



Red:
primary feature edges

Green:
secondary feature edges

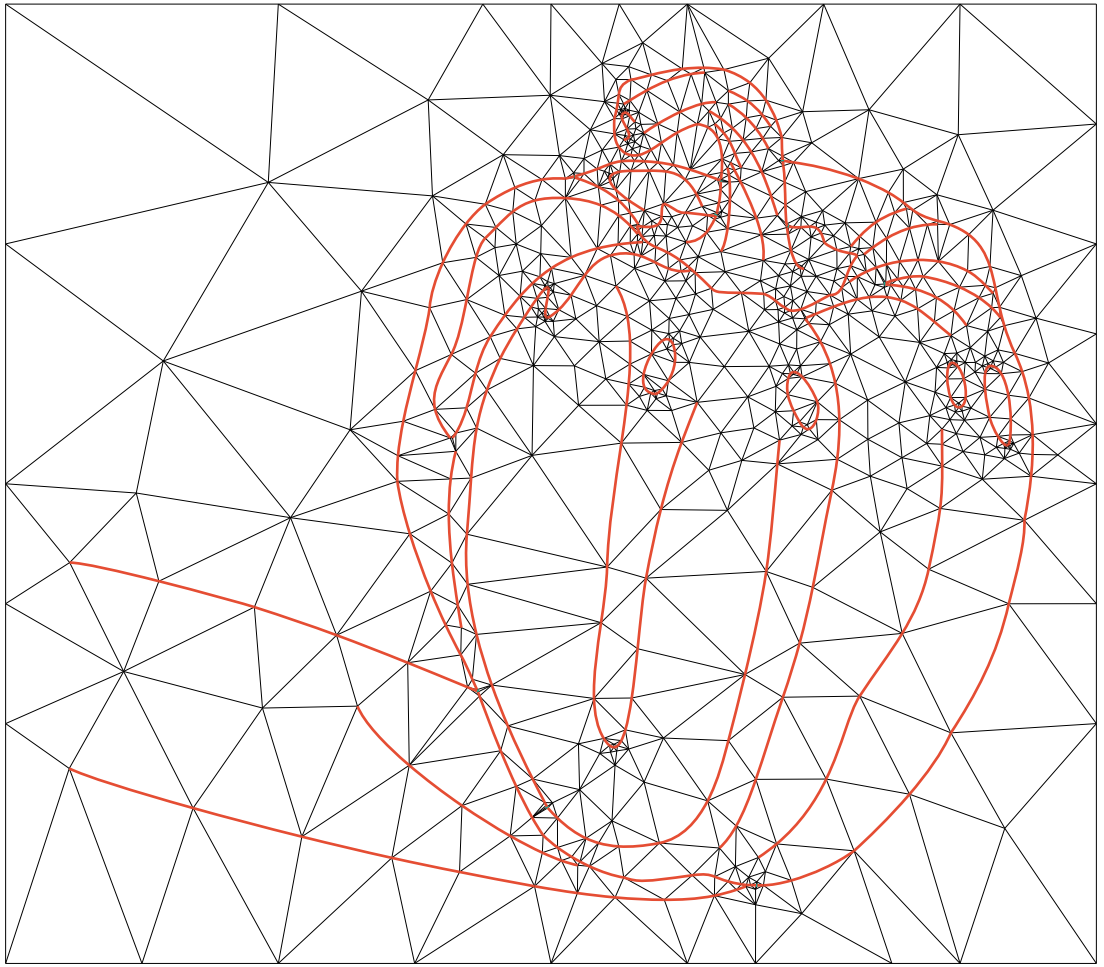


Red:
primary feature edges

Green:
secondary feature edges

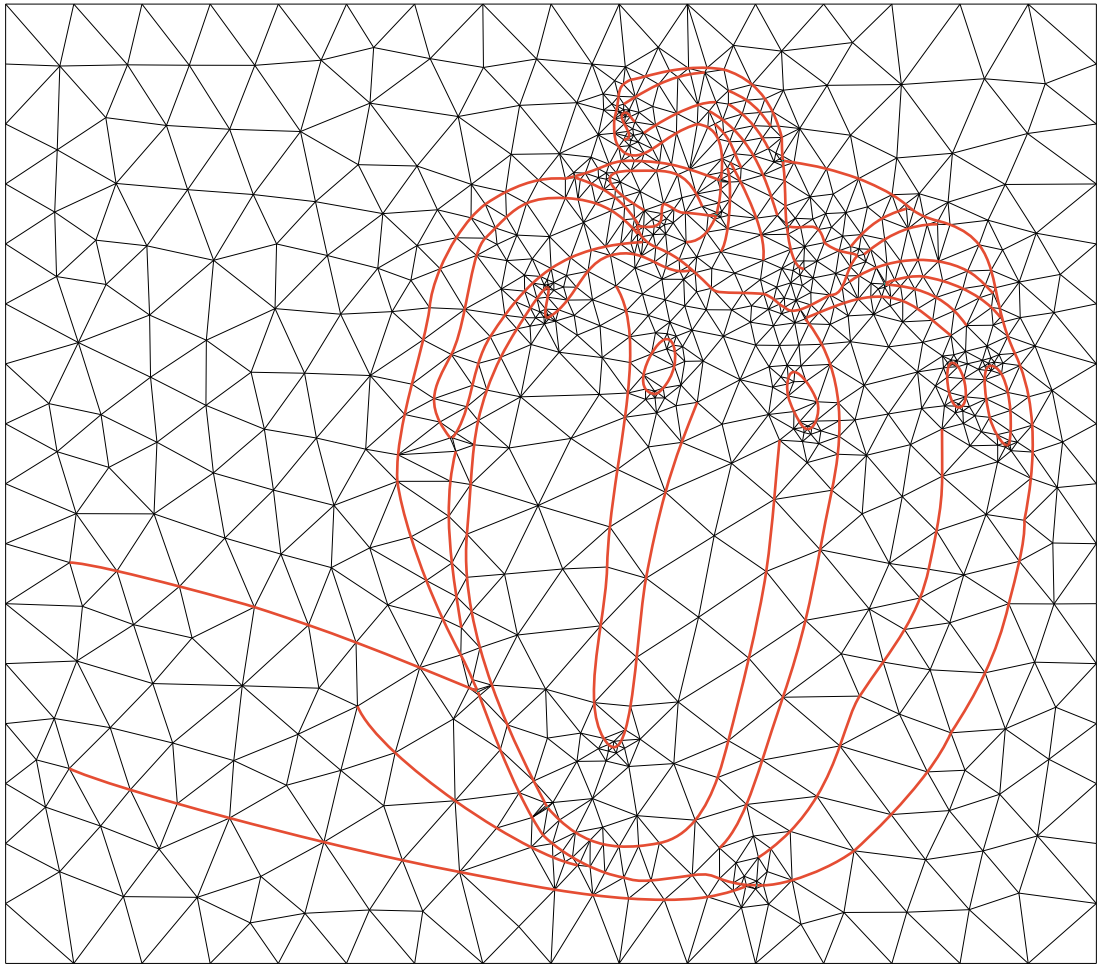
Parameters – Targeted Edge Length

Edge_length = diag_bbox/5



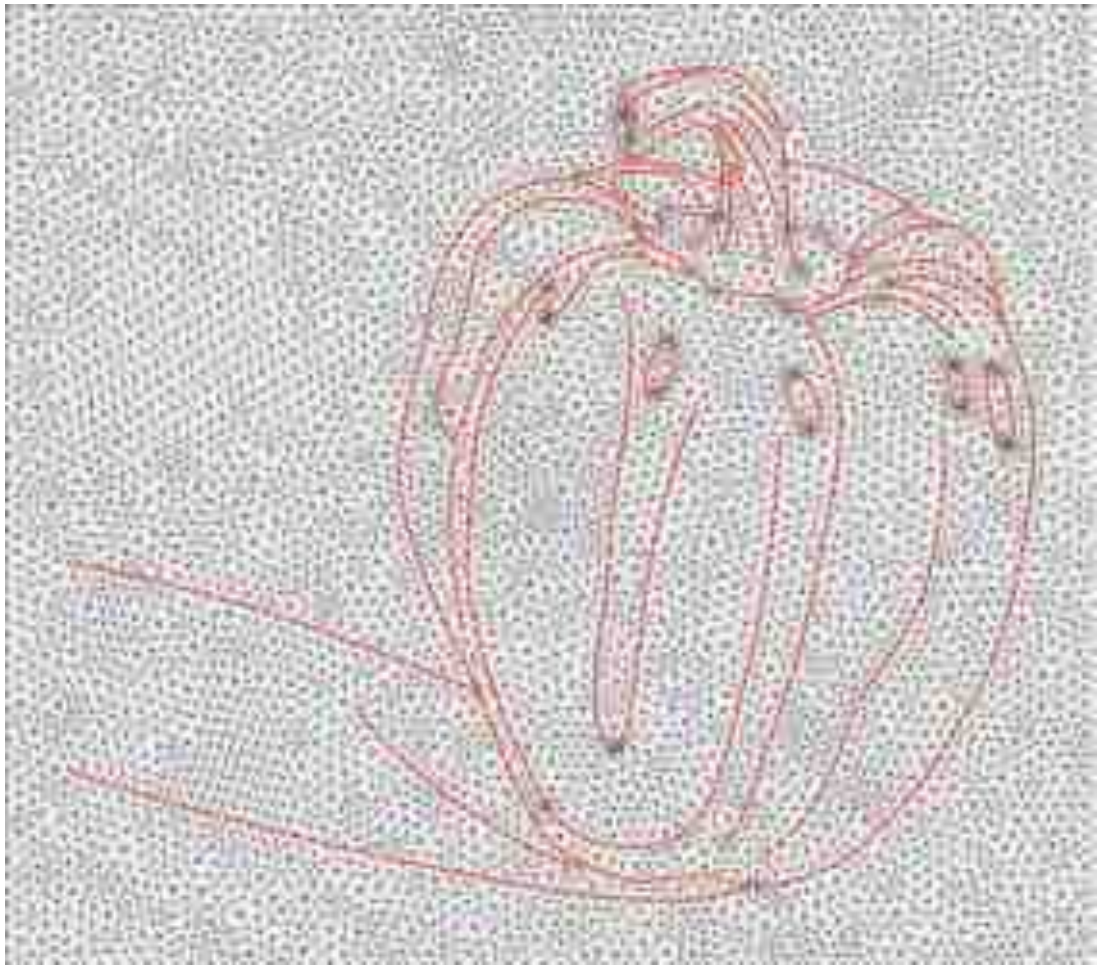
Parameters – Targeted Edge Length

Edge_length = diag_bbox/20

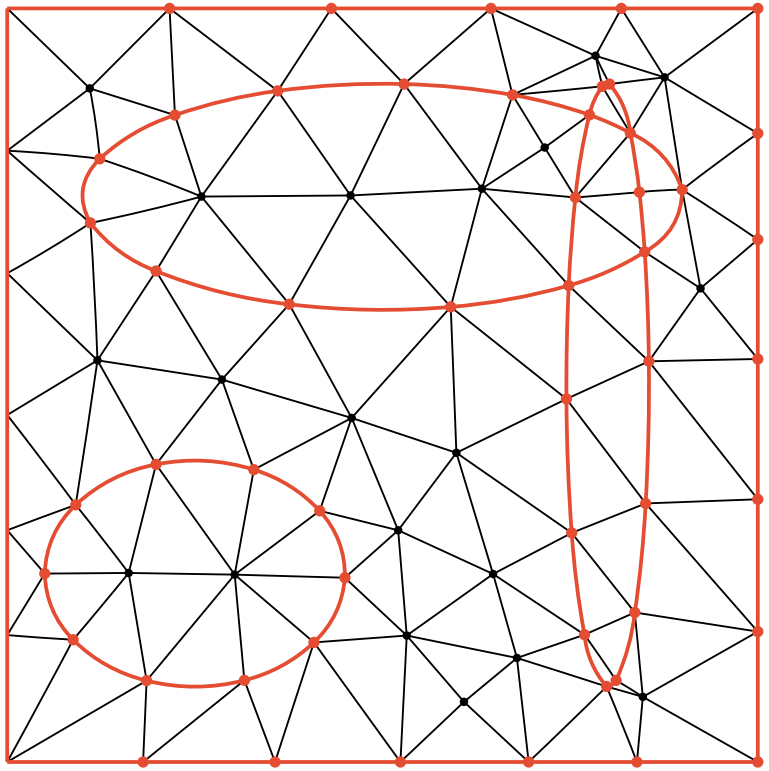


Parameters – Targeted Edge Length

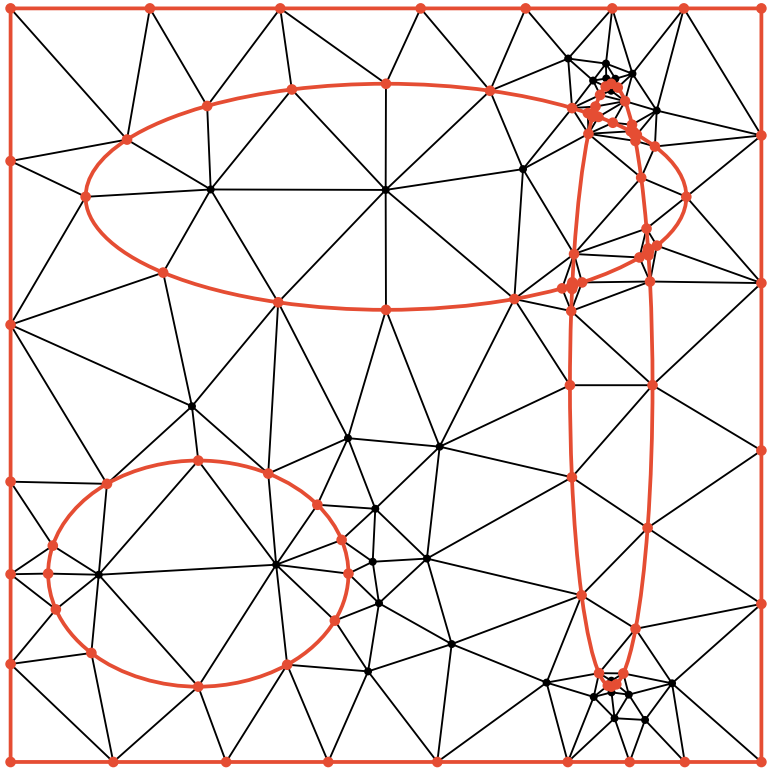
Edge_length = diag_bbox/100



MATLAB vs TriWild

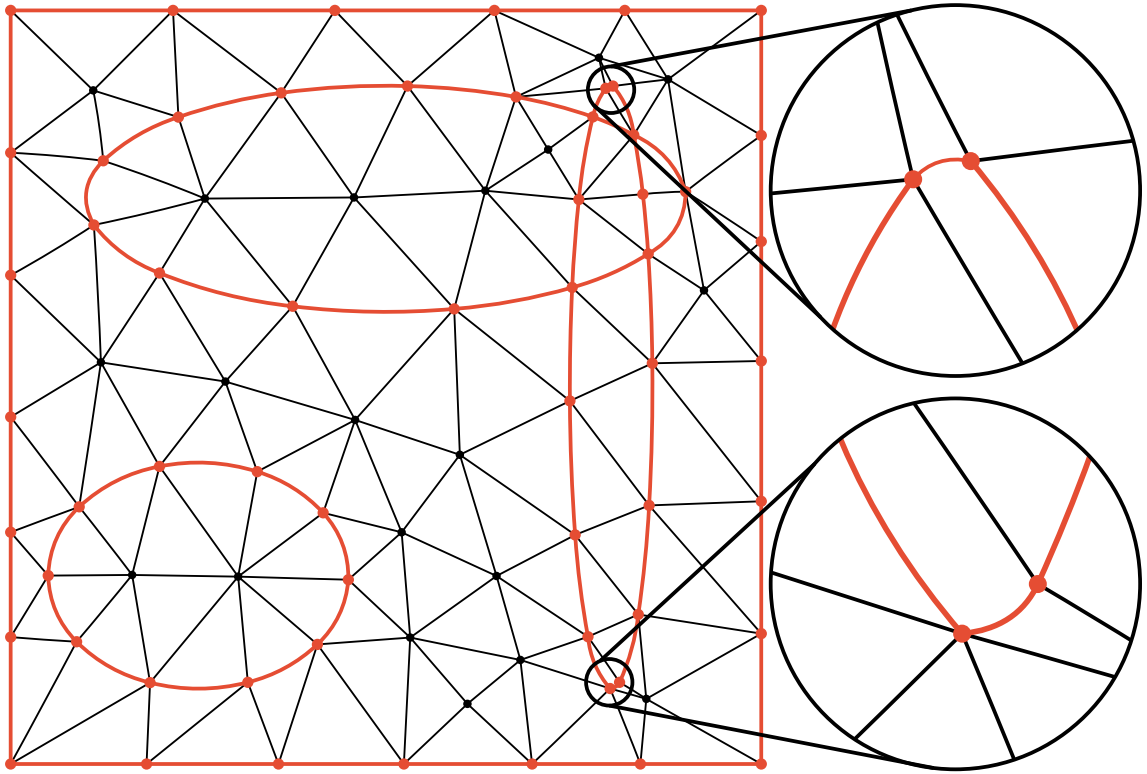


(Generated by MATLAB)

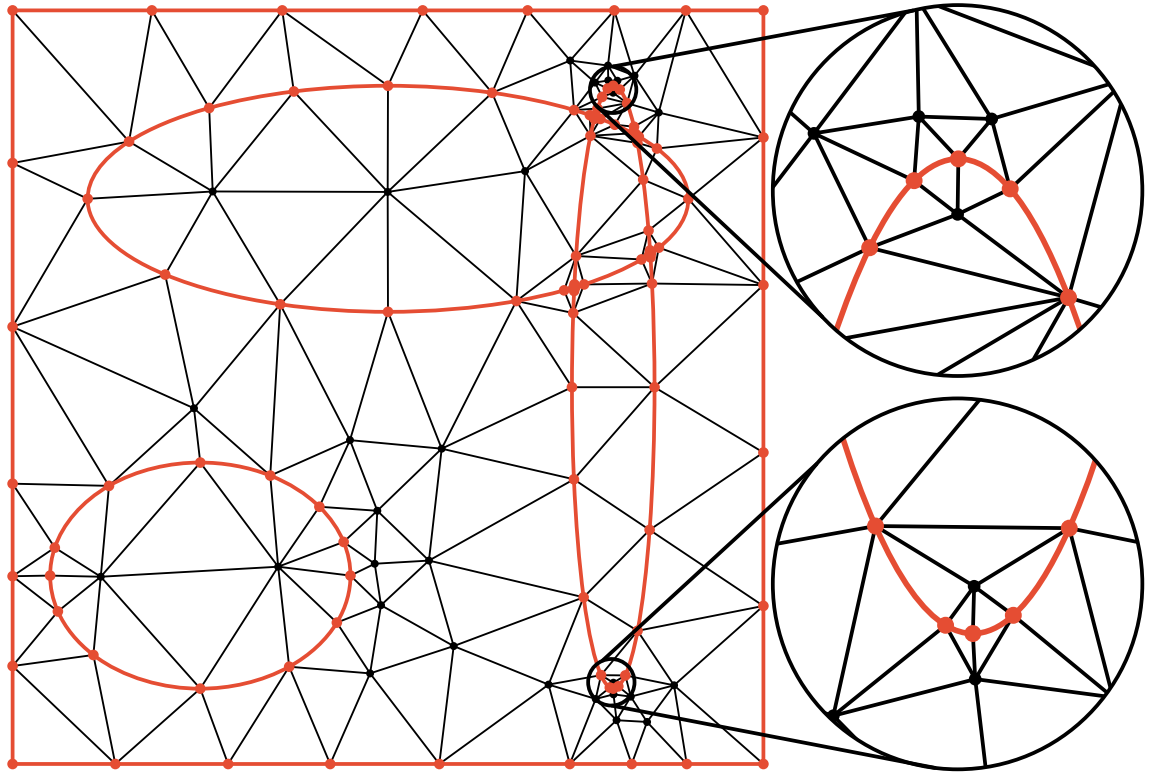


(Generated by TriWild)

MATLAB vs TriWild

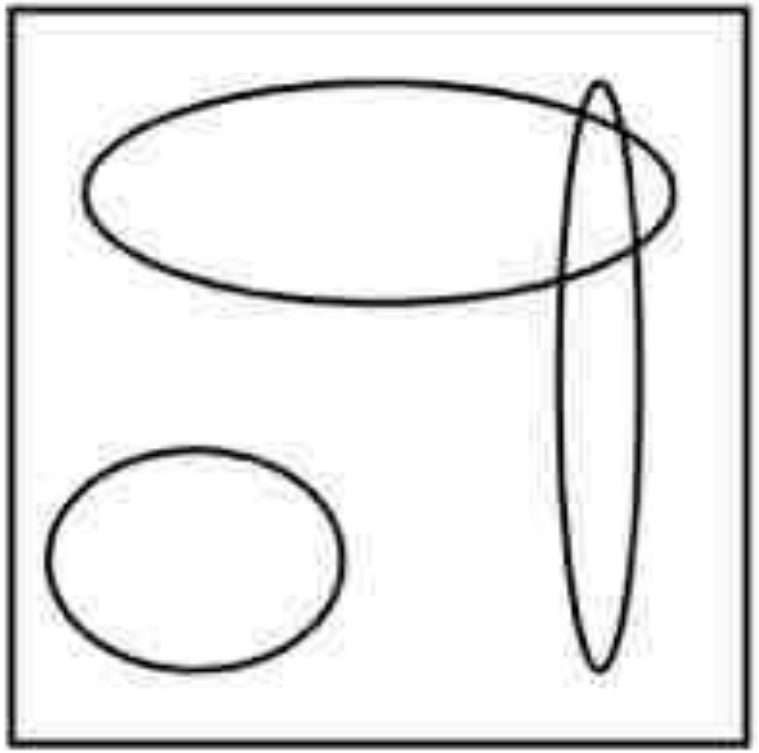


(Generated by MATLAB)

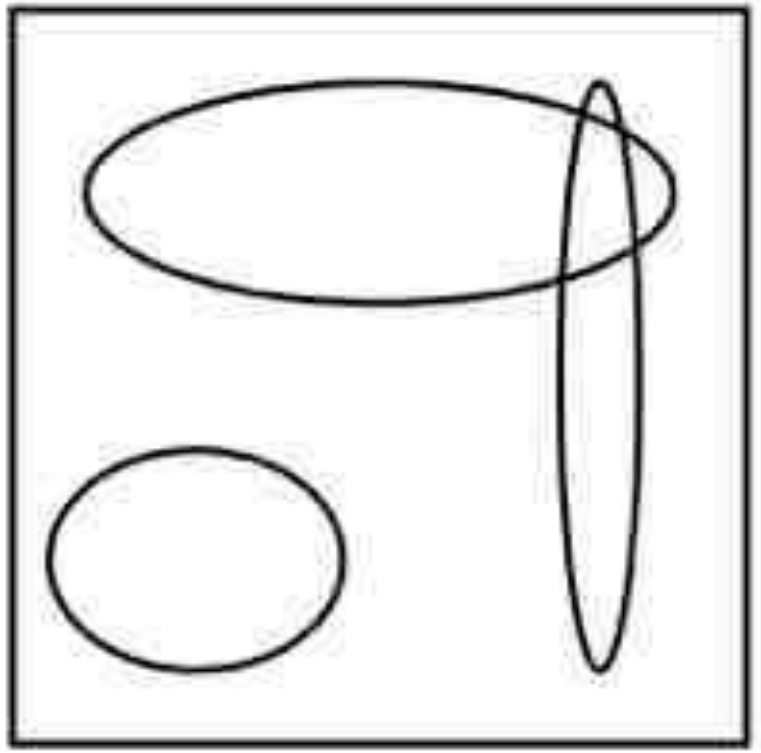


(Generated by TriWild)

MATLAB vs TriWild

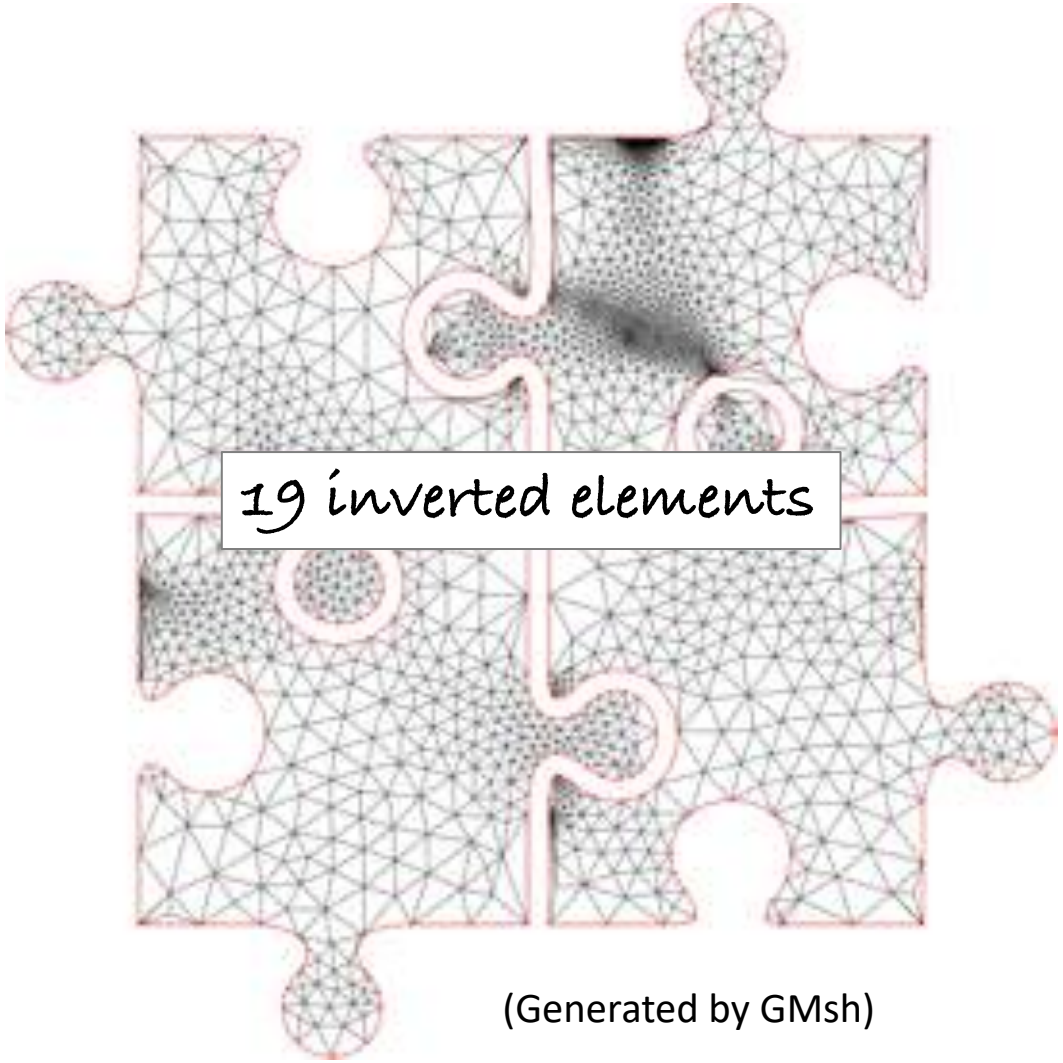


Original Input

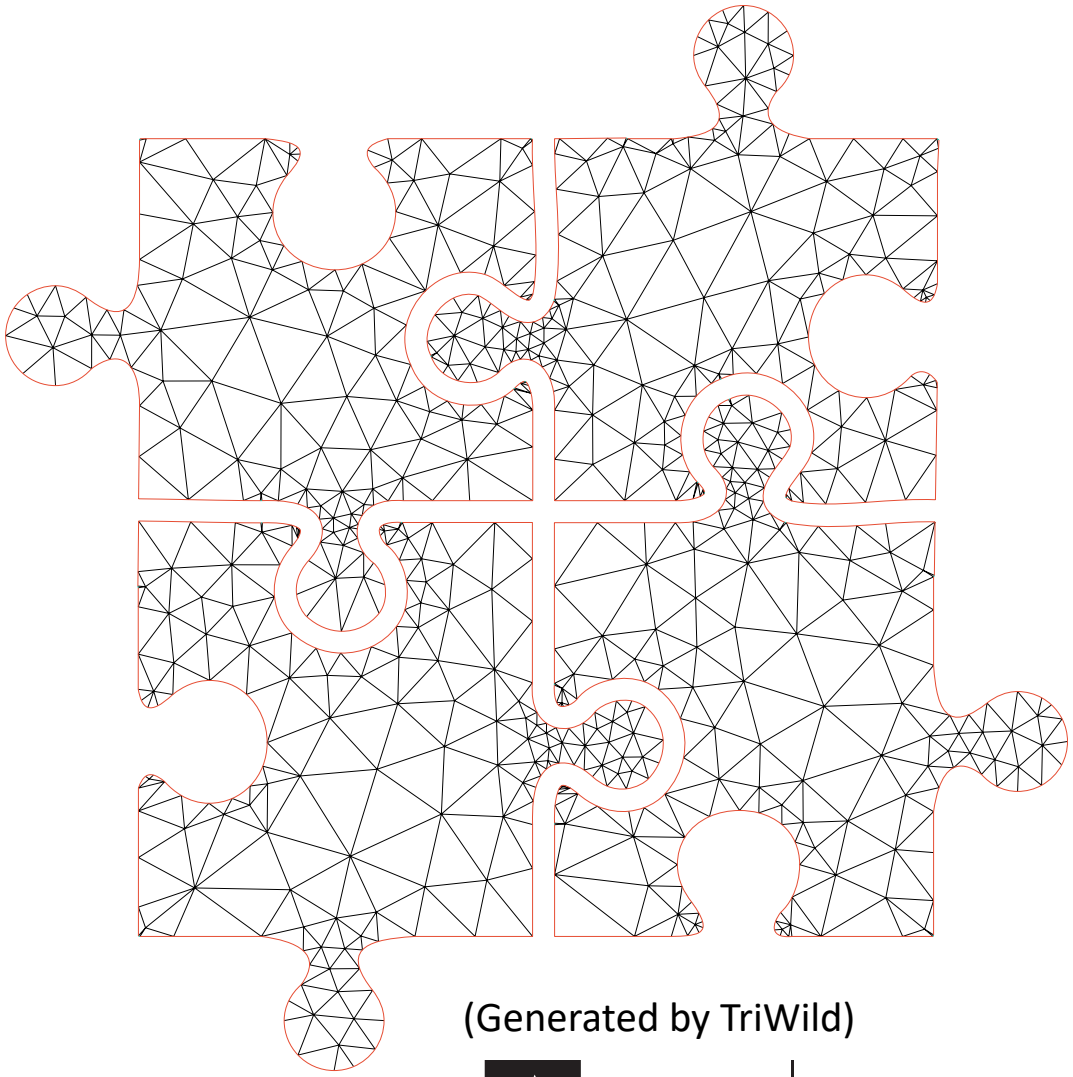


Failure Input

GMsh vs TriWild



(Generated by GMsh)



(Generated by TriWild)

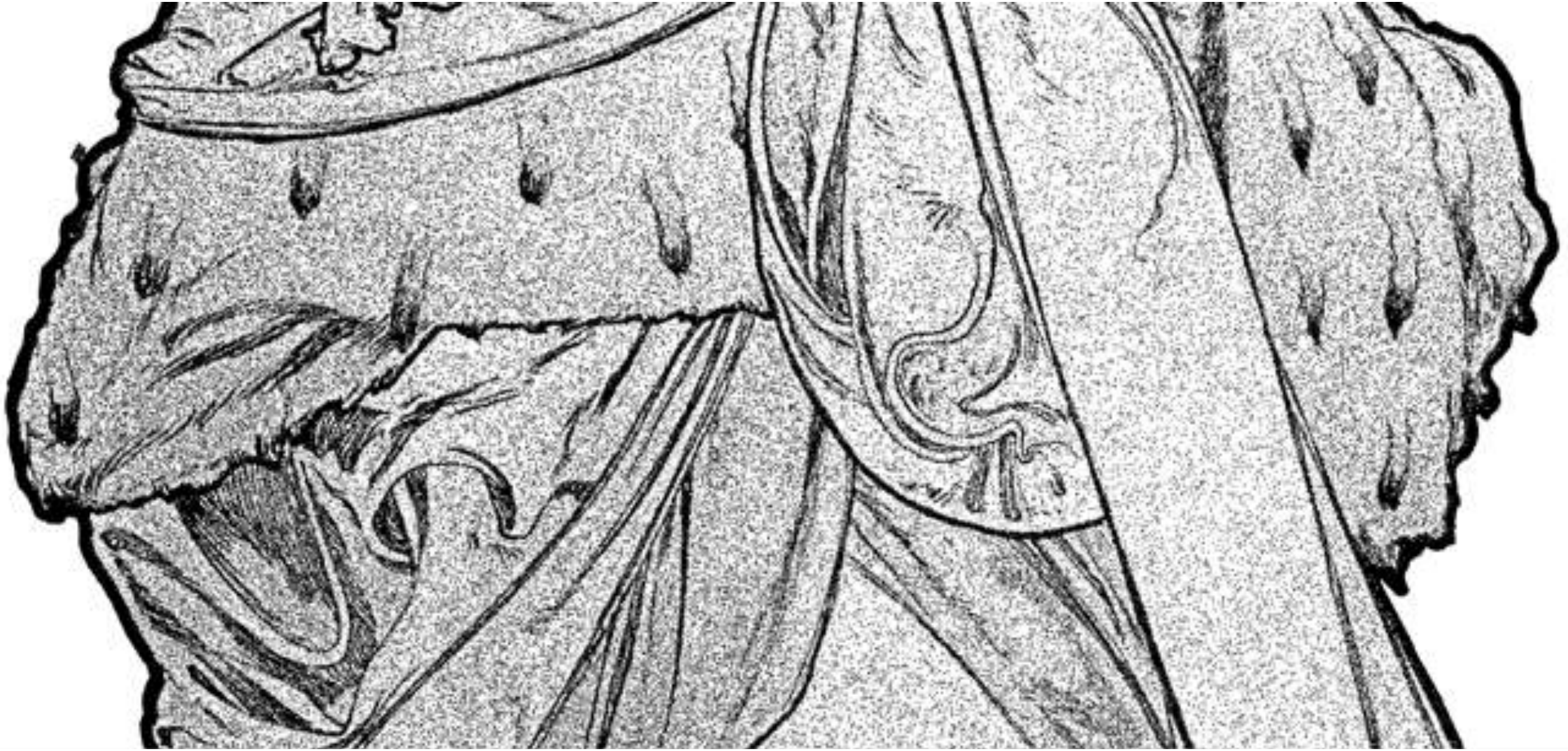
Large-scale Test

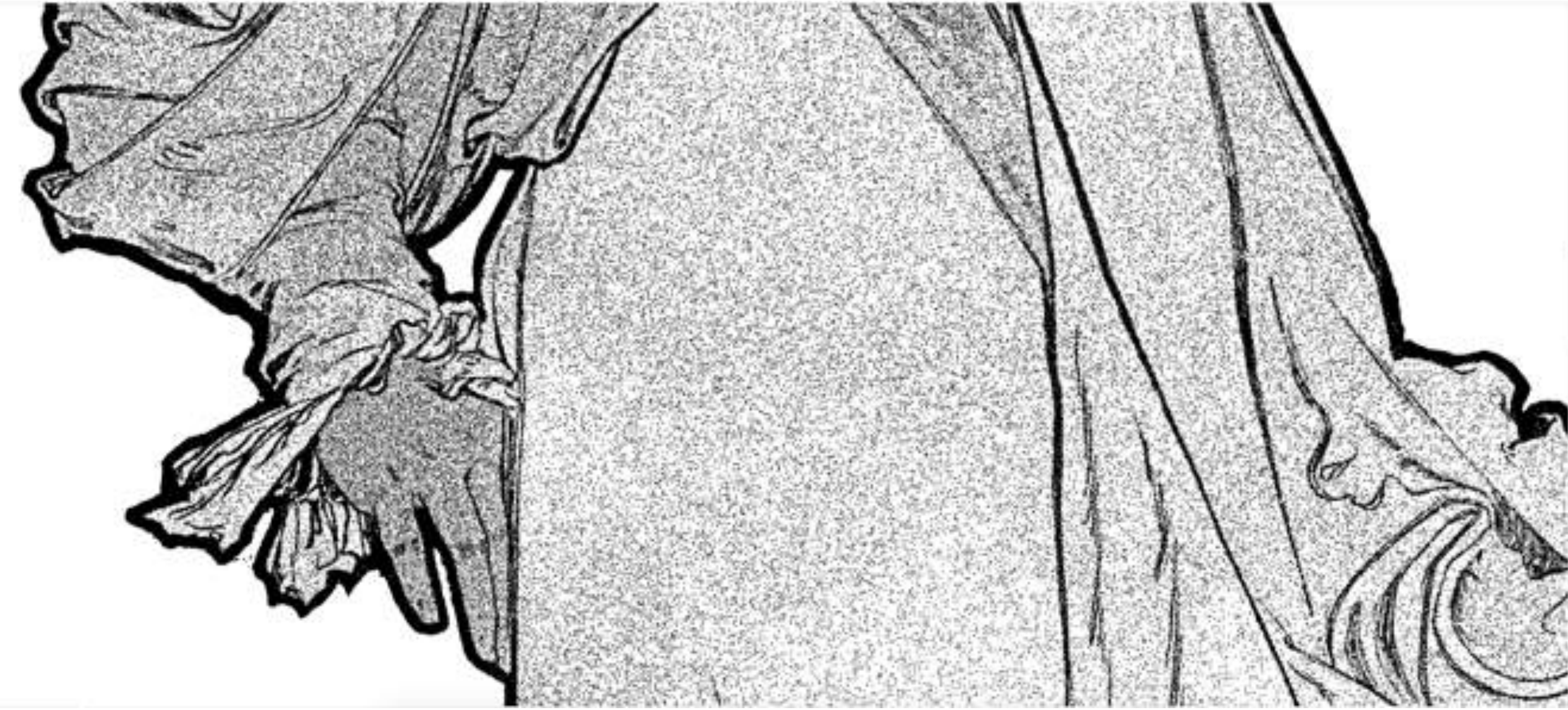
- Dataset: **19,686** real-world SVG images from openclipart.org.
- Success rate: **19,685** output meshes
  - The only failure is due to large input size (1.5GB).

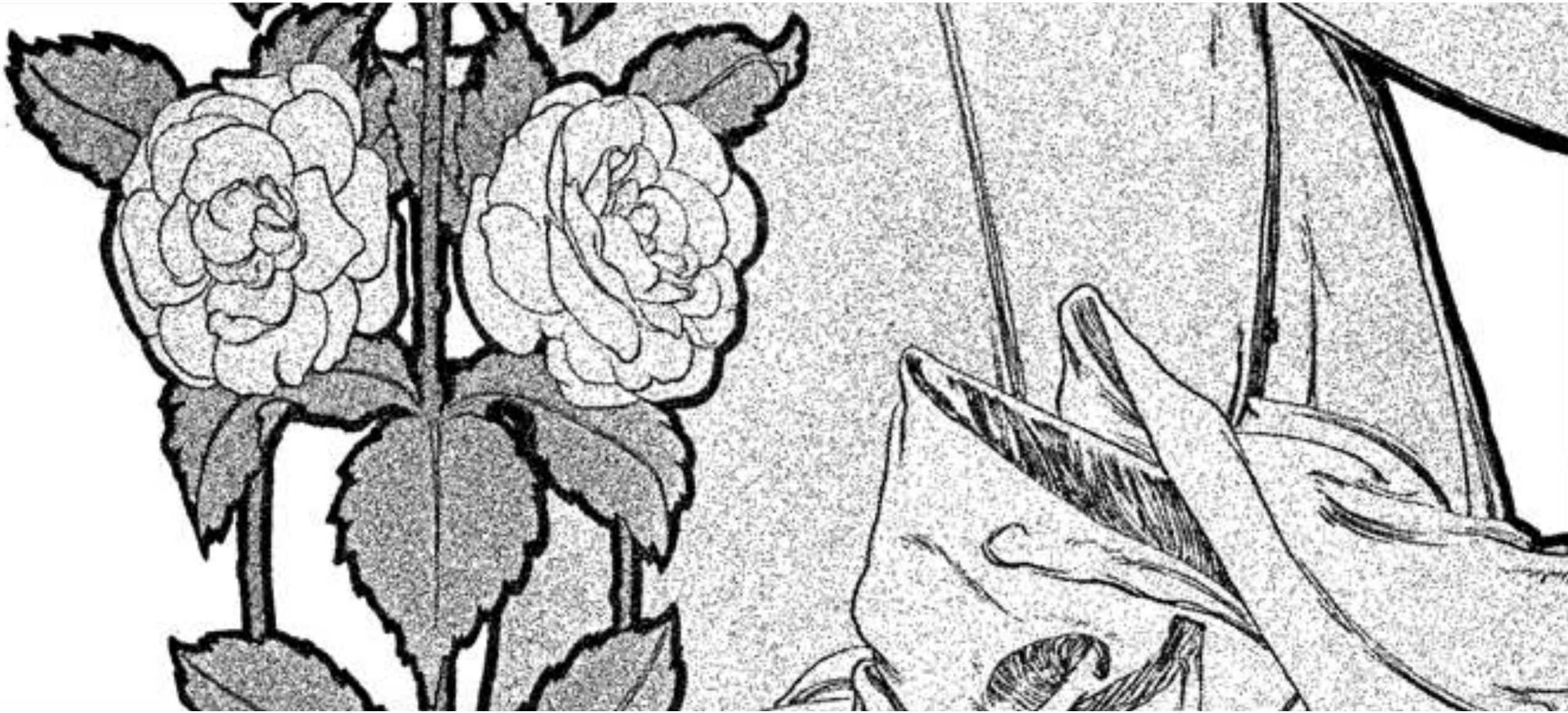
Success:

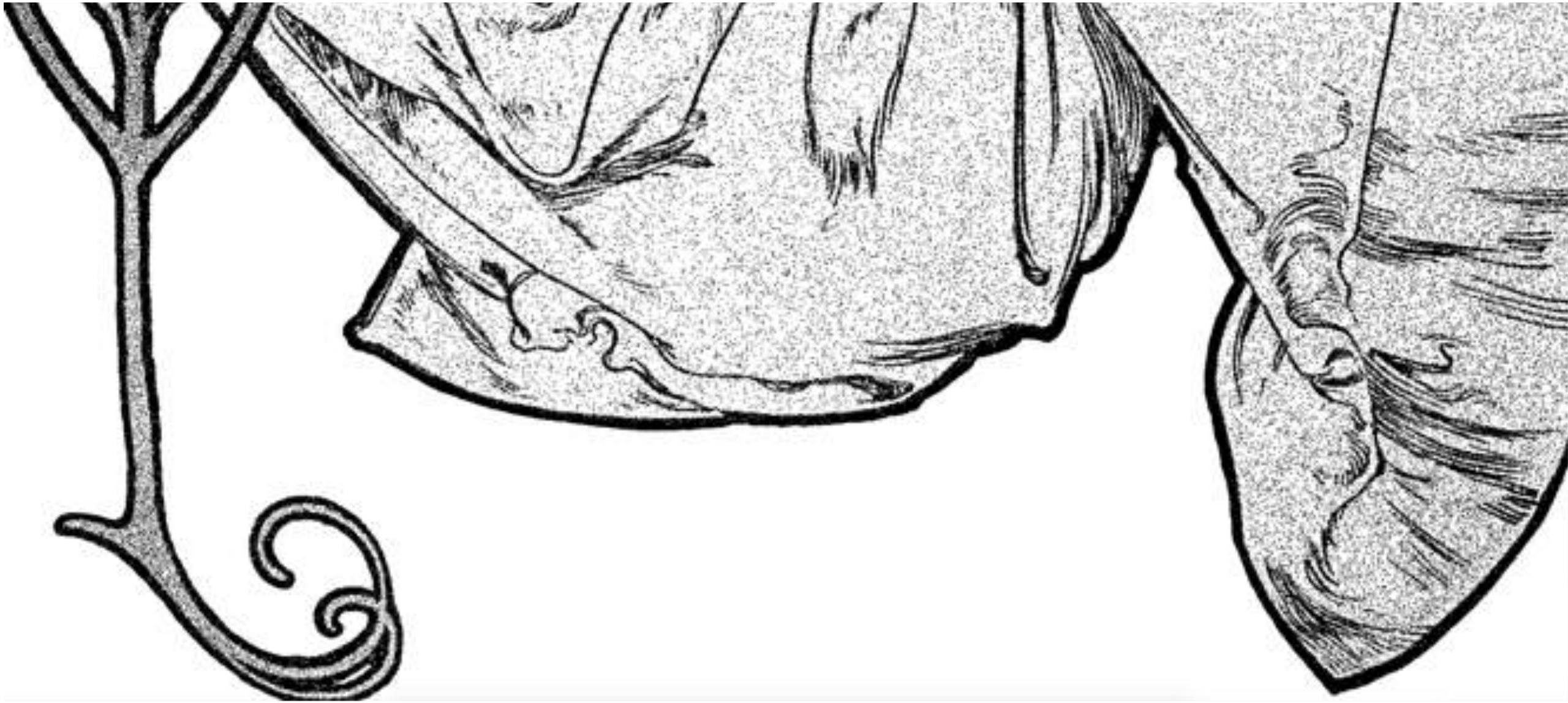
<6 hours

<64GB memory



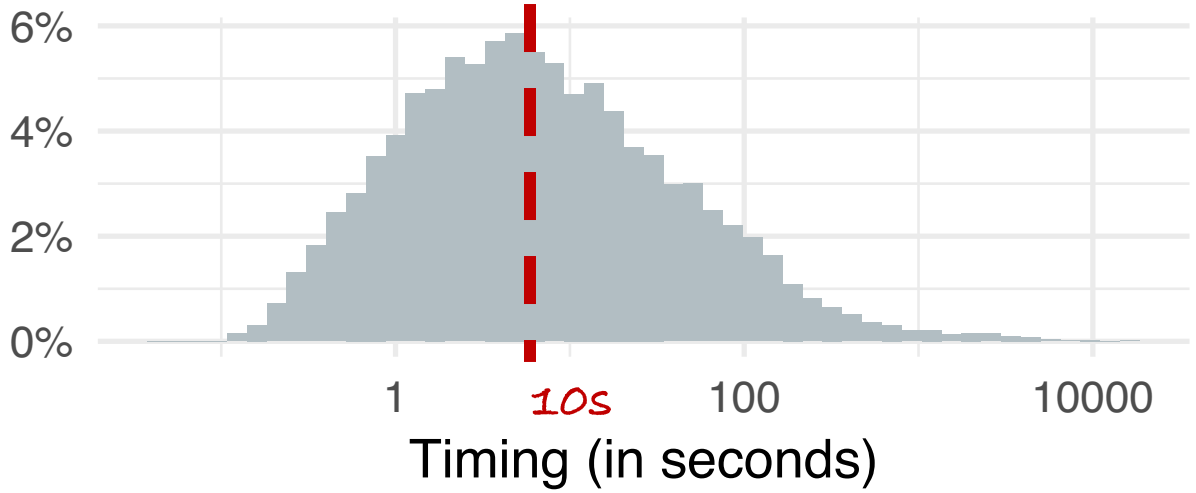






Large-scale Test

- Dataset: **19,686** real-world SVG images from openclipart.org.
- Success rate: **19,685** output meshes
  - The only failure is due to large input size (1.5GB).



Applicat

Diffusion Curves: A Vector Representation for Smooth-Shaded Images

Alexandrina Orzan
Inria - LJK (U. Grenoble - CNRS)

Adrien Bousseau
Inria Sophia Antipolis

Holger Winnemöller
Adobe Systems

Pascal Barla
Inria - U. Bordeaux - IOGS - CNRS

Joëlle Thollot
Inria - LJK (U. Grenoble - CNRS)

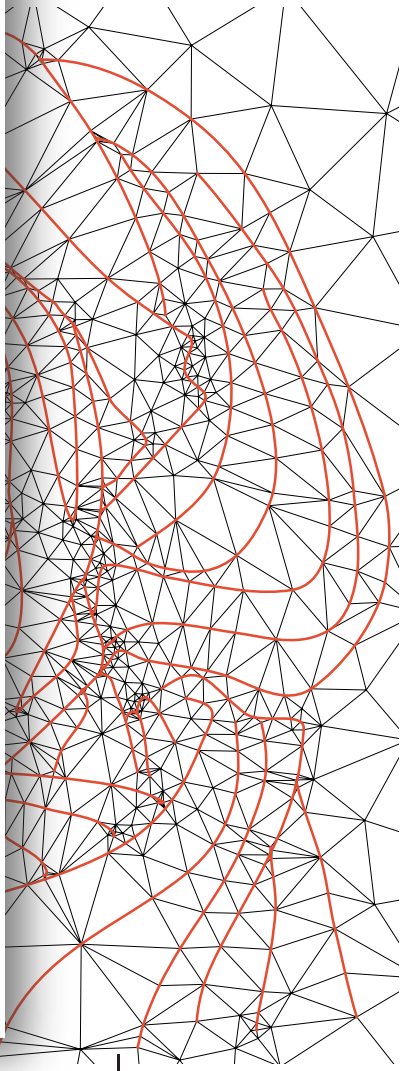
David Salesin
Adobe Systems

ABSTRACT

We describe a new vector-based primitive for creating smooth-shaded images, called the *diffusion curve*. A diffusion curve partitions the space through which it is drawn, defining different colors on either side. These colors may vary smoothly along the curve. In addition, the sharpness of the color transition from one side of the curve to the other can be controlled. Given a set of diffusion curves, the final image is constructed by solving a Poisson equation whose constraints are specified by the set of gradients across all diffusion curves. Like all vector-based primitives, diffusion curves conveniently support a variety of operations, including geometry-based editing, keyframe animation, and ready stylization. Moreover, their representation is compact and inherently resolution independent. We describe a GPU-based implementation for rendering images defined by a set of diffusion curves in real time. We then demonstrate an interactive drawing system for allowing artists to create artworks using diffusion curves, either by drawing the curves in a freehand style, or by tracing existing imagery. Furthermore, we describe a completely automatic conversion process for taking an image and turning it into a set of diffusion curves that closely approximate the original image content.

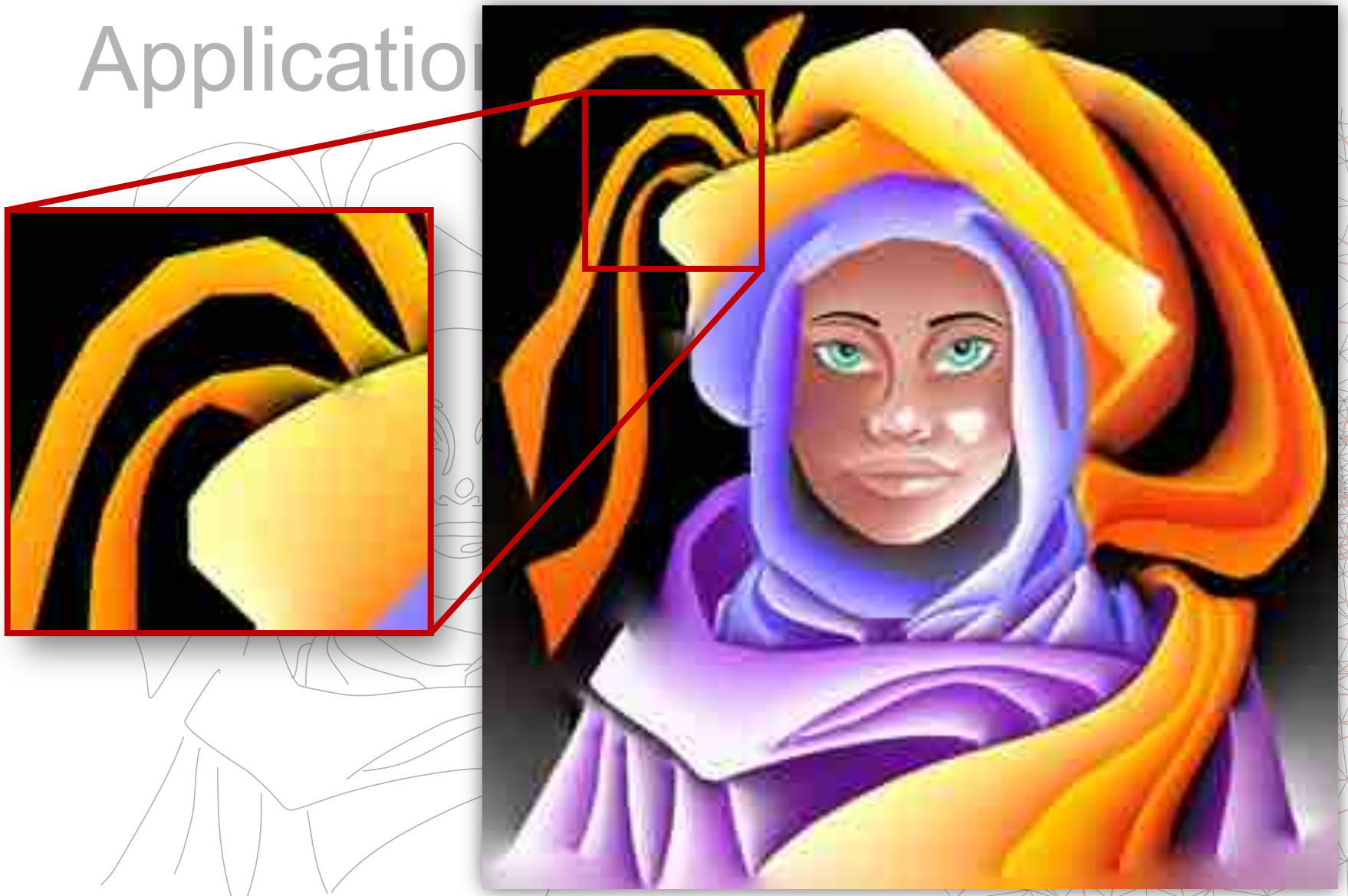
Figure 1: Diffusion curves (left), and the corresponding color image (right). Note the complex shading on the folds and blur on the face.

tor graphics — primarily, a more direct mapping between the hardware devices used for acquisition and display of images, and their internal representation — vector graphics continues to provide certain benefits as well. Most notably, vector graphics offers a more compact representation, geometric editability, and resolution independence (allowing scaling of images while retaining sharp edges, see Figure 2). Vector-based images are also more easily animated through keyframe animation of their underlying geometric primi-

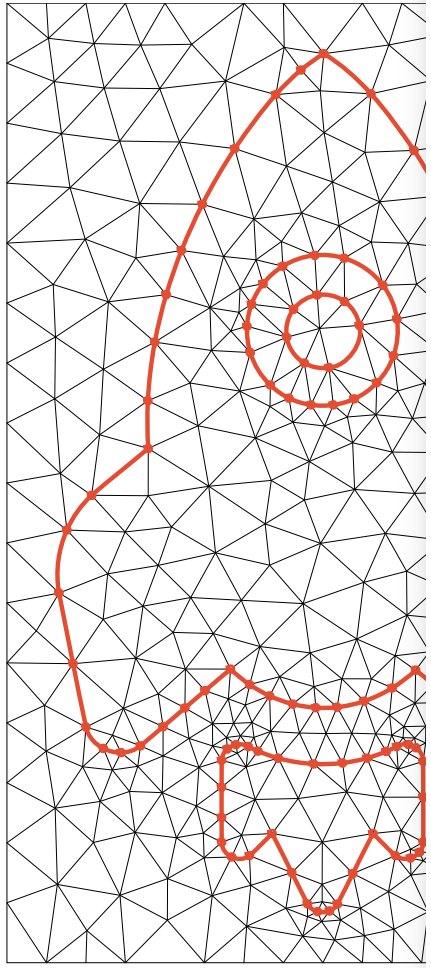


NYU COURANT INSTITUTE OF MATHEMATICAL SCIENCES

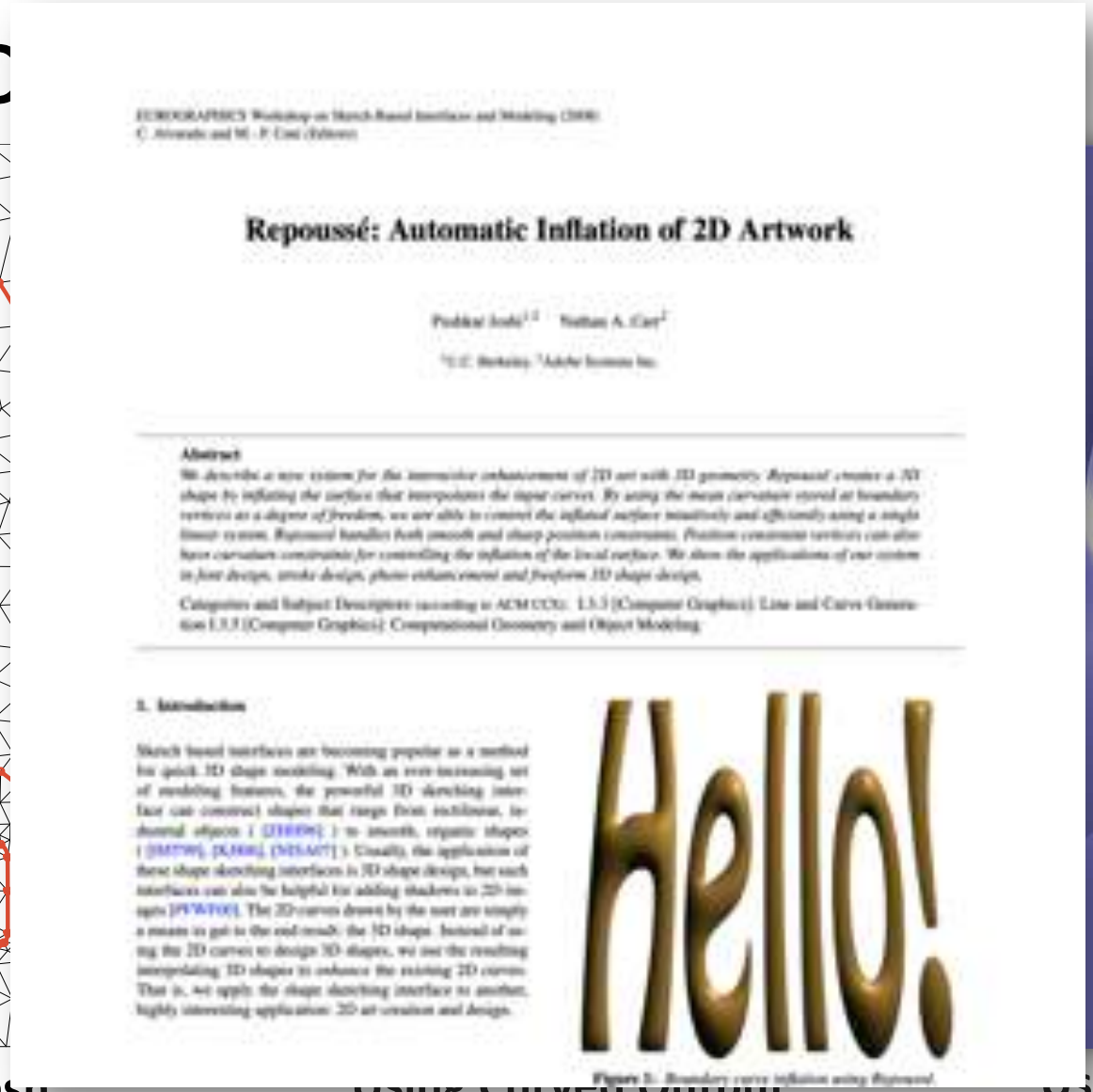
Applicatio



Applicatic



Curved Mesh

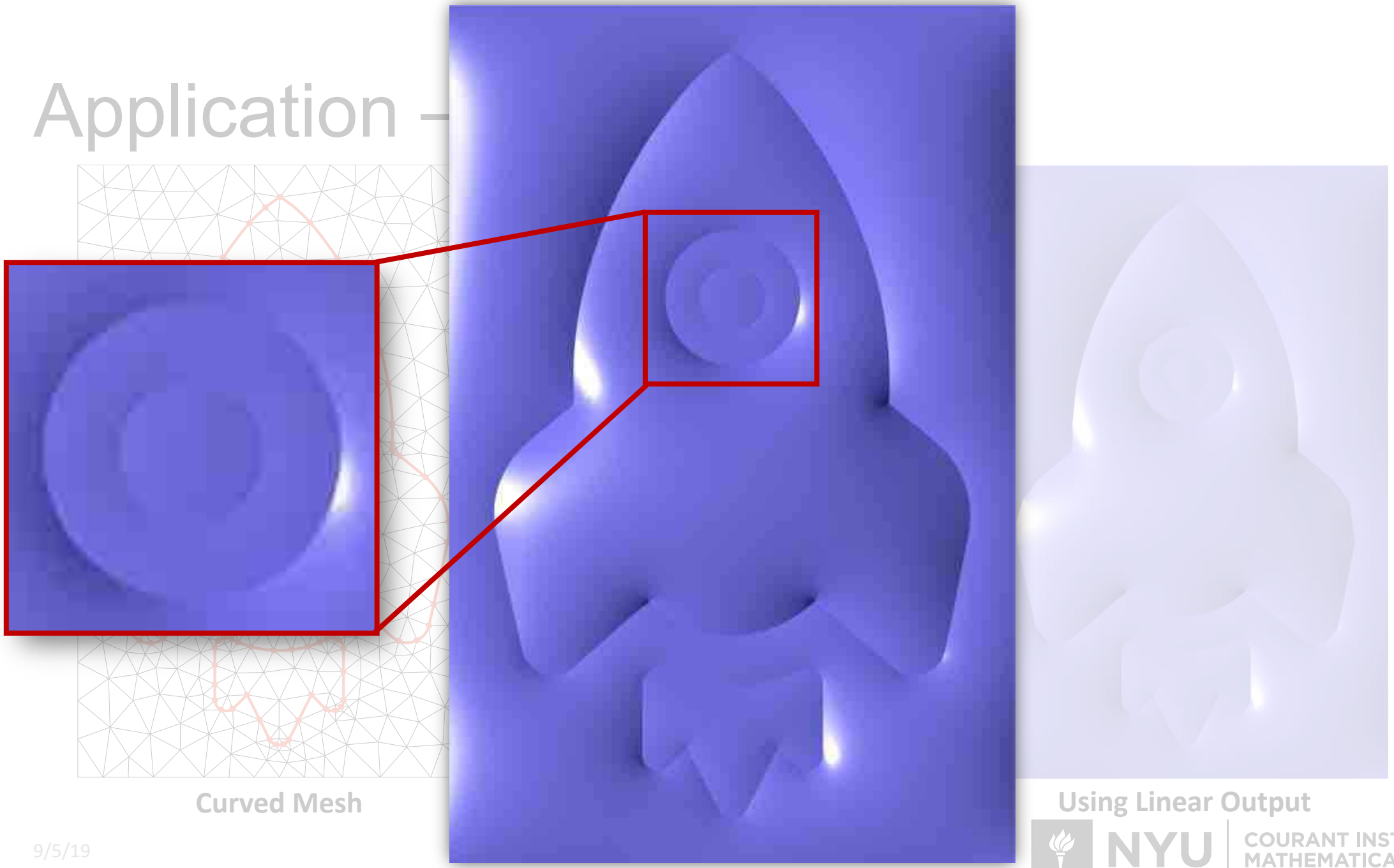


Repoussé: Automatic Inflation of 2D Artwork

Using Curved Output

Using Linear Output

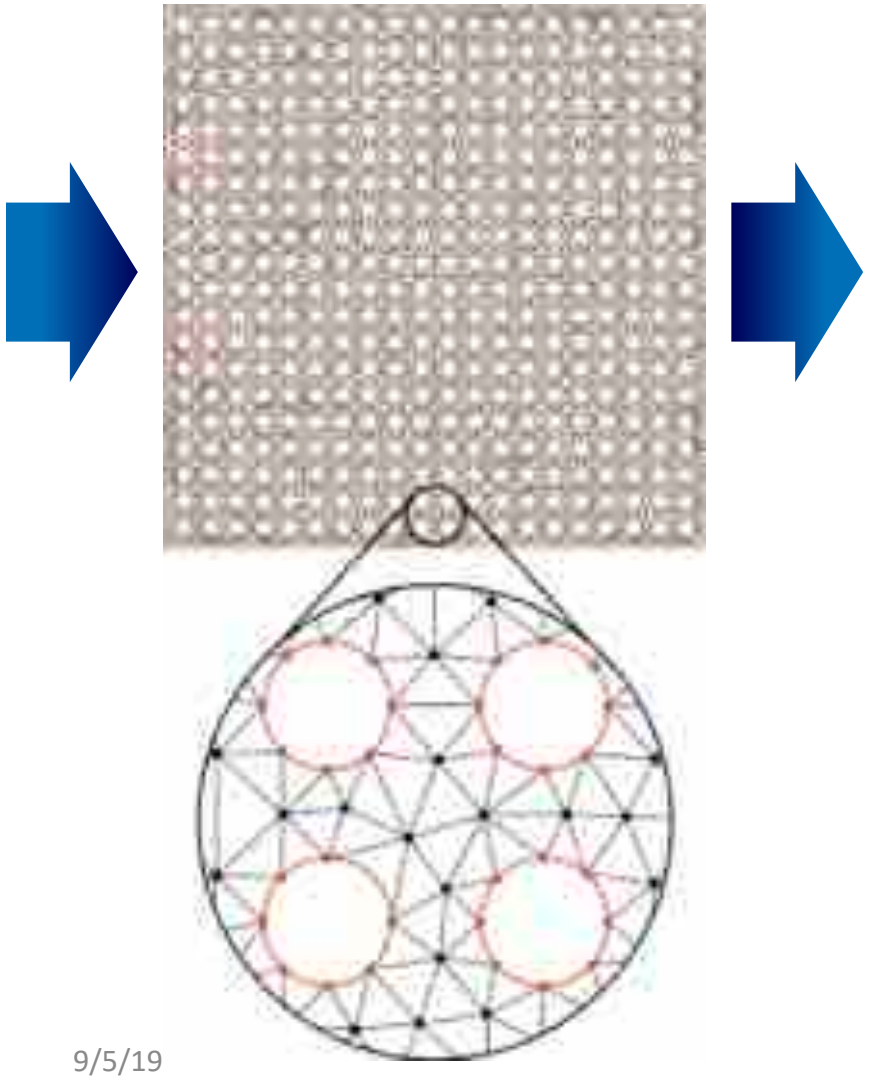
Application –

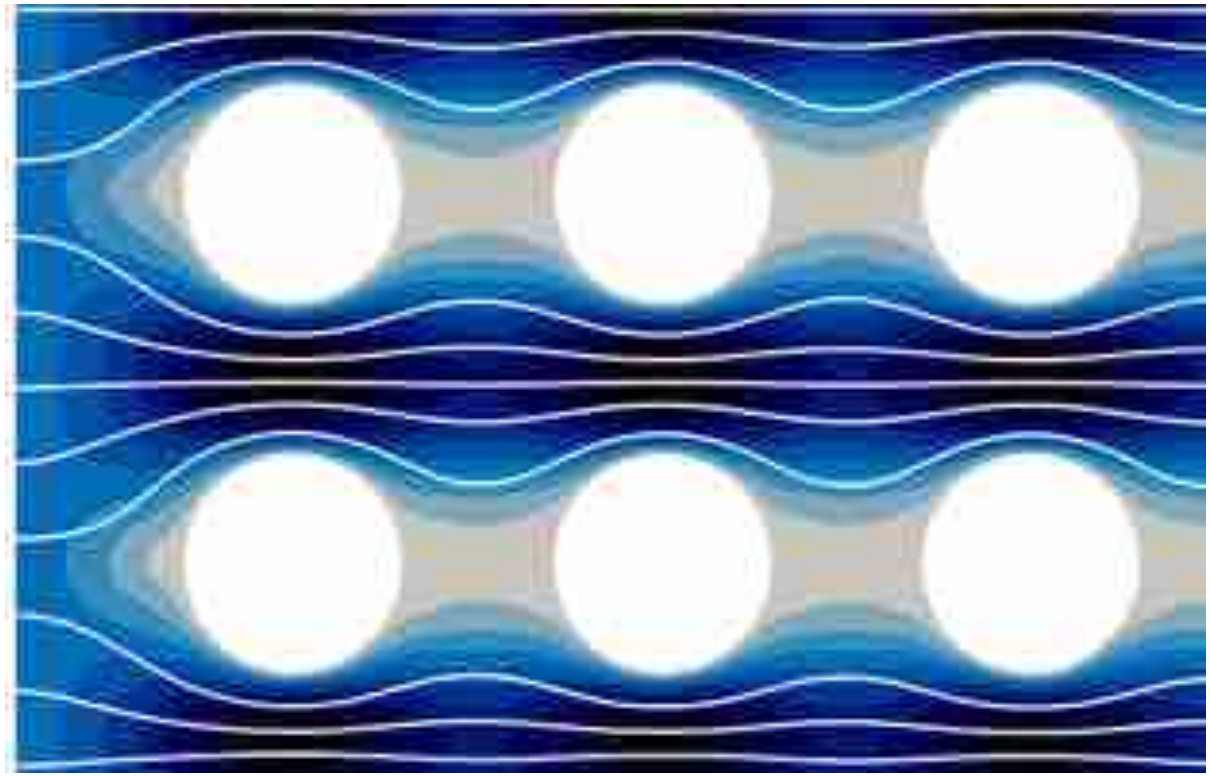


Curved Mesh

Using Linear Output

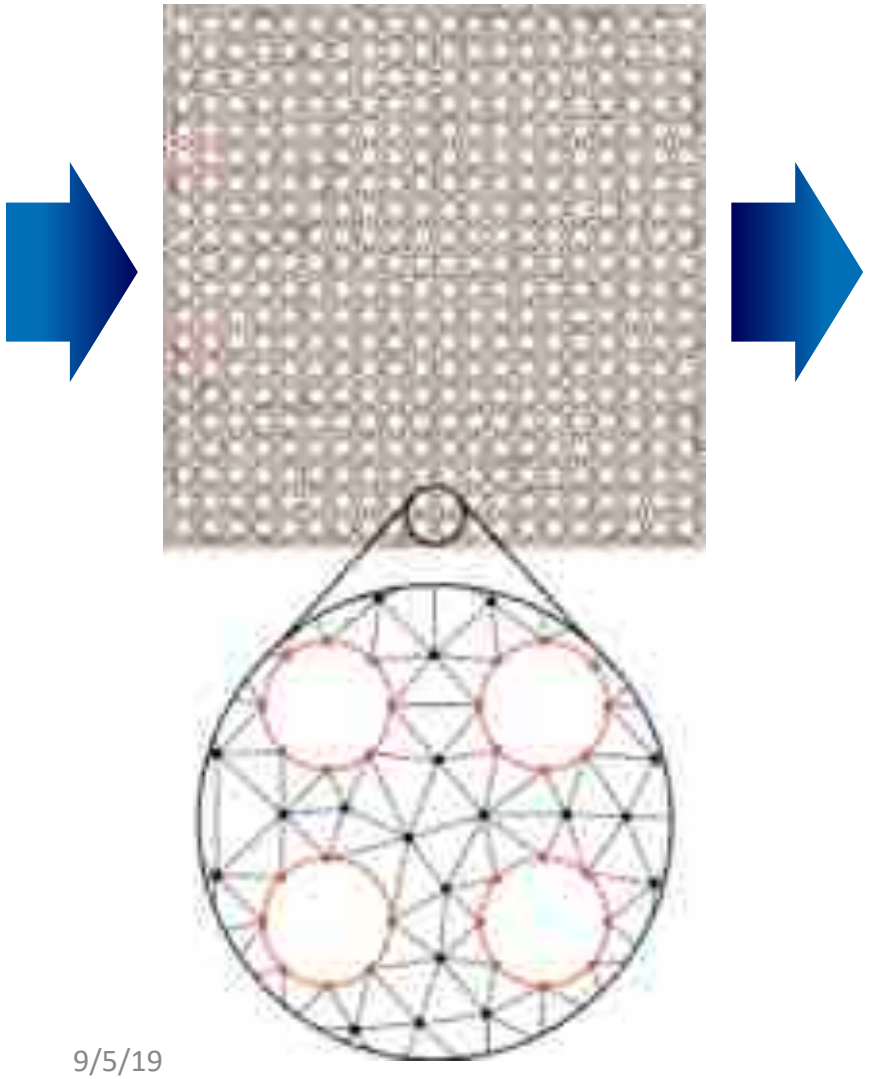
Application – Stokes

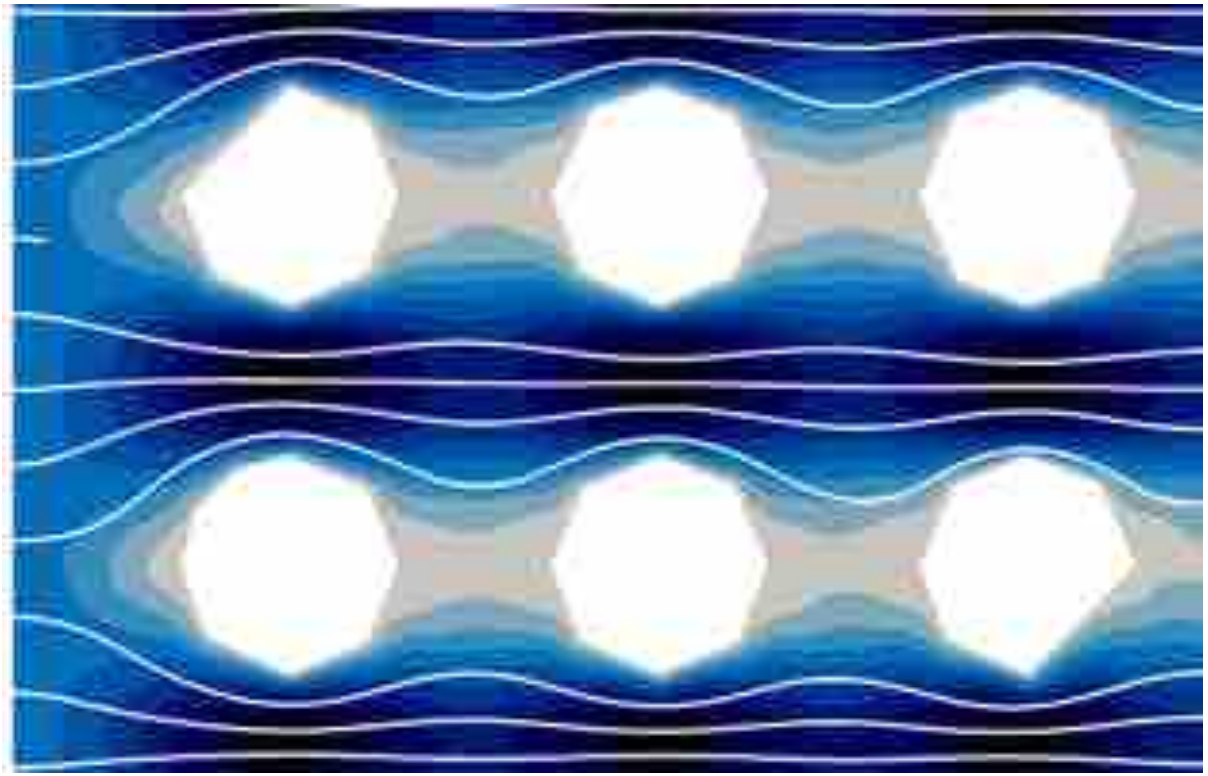




Using Curved Mesh

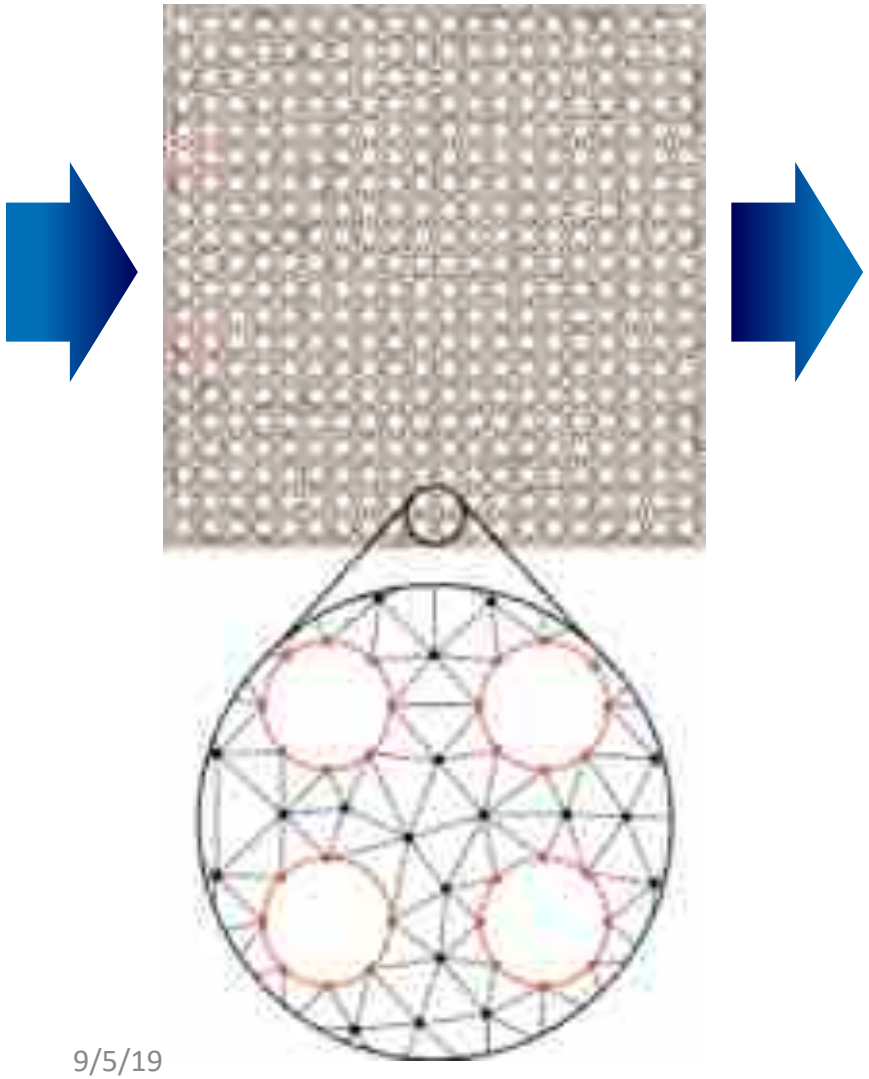
Application – Stokes

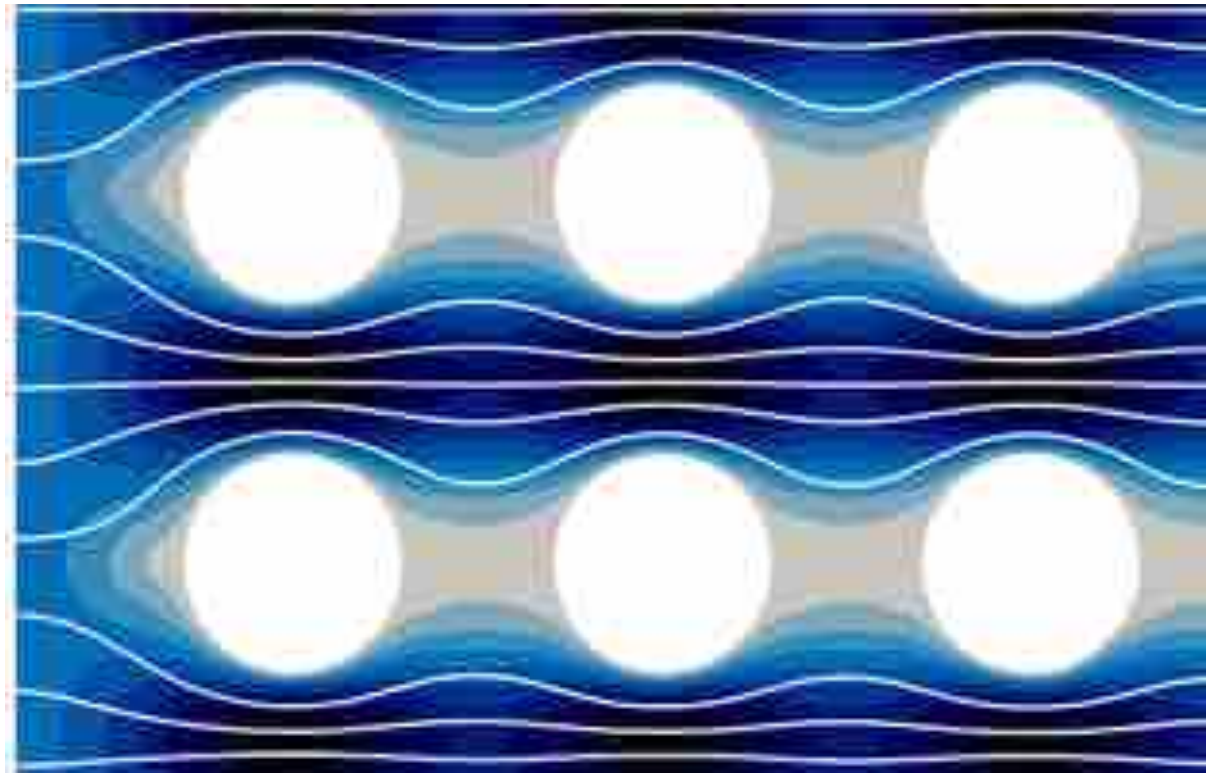




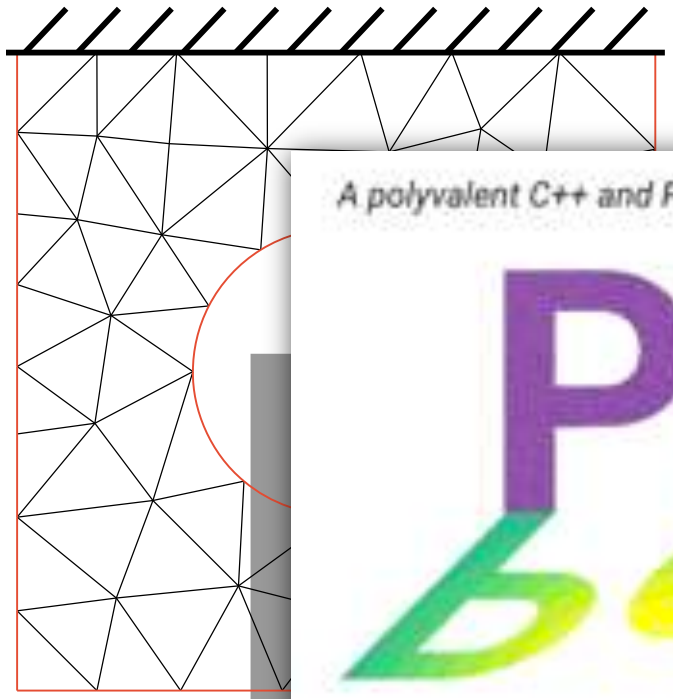
Using Linear Mesh

Application – Stokes





Application – Elasticity



Gravity

Curved Mesh

0.09s

115s

Using Curved Mesh

Using Dense Linear Mesh

A polyvalent C++ and Python FEM library.

PolyFEM

BOOLEAN

Linear Pipeline

Dataset 1: Raw OpenClip20k

- Piece-wise linear approximations of 19,686 SVG images.

Dataset 2: Cleaned OpenClip20k

- Raw dataset with duplication and degeneracy removed.

Dataset 3: Snapped OpenClip20k

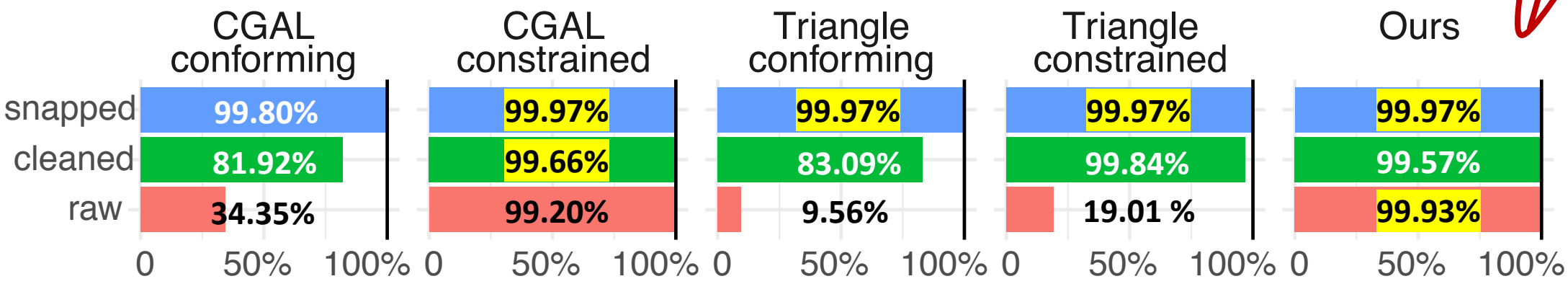
- Raw dataset with iteratively snap rounding using ϵ as pixel size.

Linear Pipeline

Success:
<1 hour
<16GB memory

Compared with:

1) CGAL Conforming/Constrained Delaunay Triangulation
2) Triangle Conforming/Constrained Delaunay Triangulation

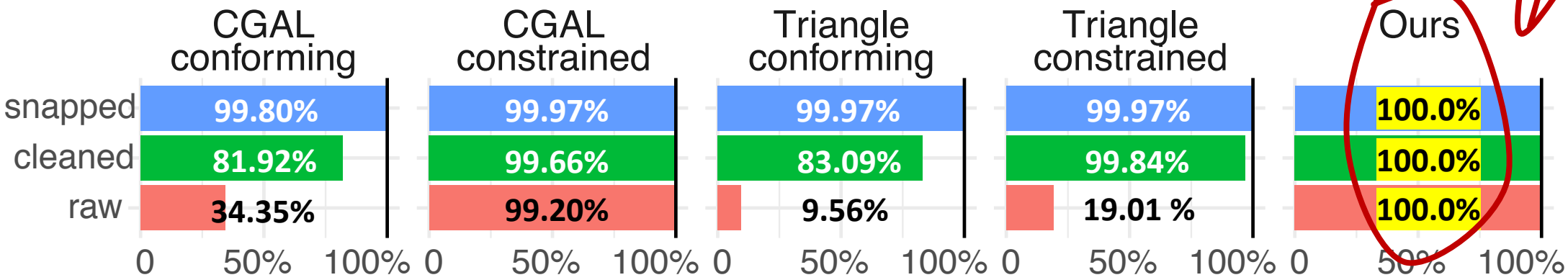


Linear Pipeline

Success:
<1 hour
<~~16~~GB memory
64GB

Compared with:

1) CGAL Conforming/Constrained Delaunay Triangulation
2) Triangle Conforming/Constrained Delaunay Triangulation



Thank You!

https://github.com/wildmeshing/TriWild

ACKNOWLEDGMENTS

- This work was supported in part through the NYU IT High Performance Computing resources, services, and staff expertise.

- This work was partially supported by the NSF CAREER award under Grant No. 1652515, the NSF grant IIS-1320635, the NSF grant DMS-1436591, the NSF grant 1835712, the SNSF grant P2TIP2_175859, NSERC Discovery Grants (RGPIN-2017-05235 & RGPAS-2017-507938), Canada Research Chair award, Connaught Fund, a gift from Adobe Research, and a gift from nTopology.

Thank You!

https://github.com/wildmeshing/TriWild

- **Please scan the QR code for reference implementation in C++ and data.**
- **Python wrapper via Conda:**

```
conda config --add channels conda-forge
conda install wildmeshing
```

- **Contact me if you have any questions!**

  yixin.hu@nyu.edu