L20: Mesh Processing

Hao Su

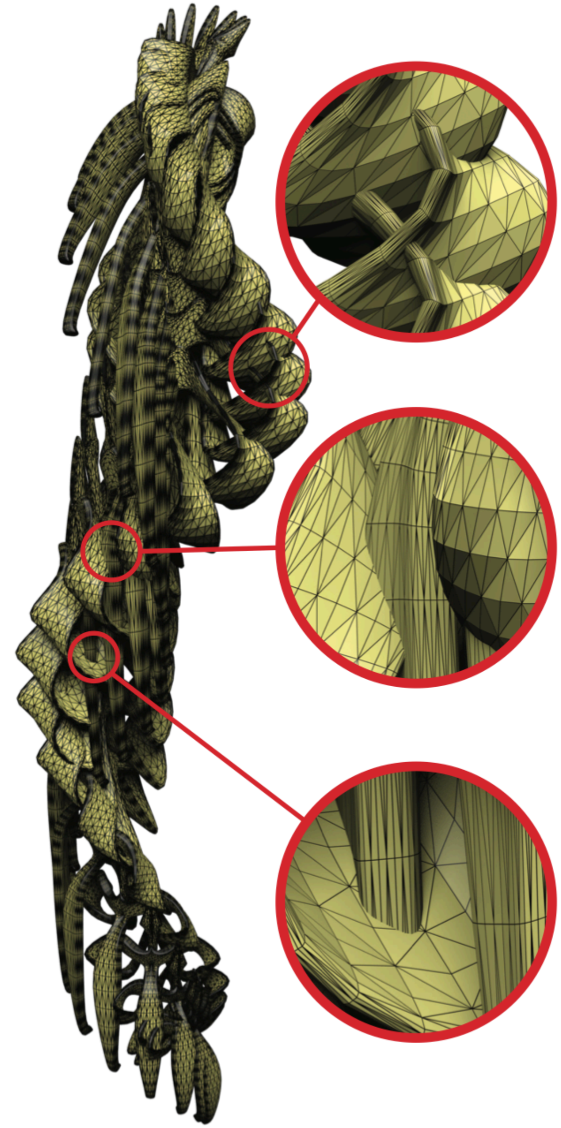
Ack: Minghua Liu helped to prepare slides

A Peek into Various
Mesh Processing Algorithms

Watertight Manifold Surface Generation

- Many downstream tasks require the input mesh to be a watertight manifold.

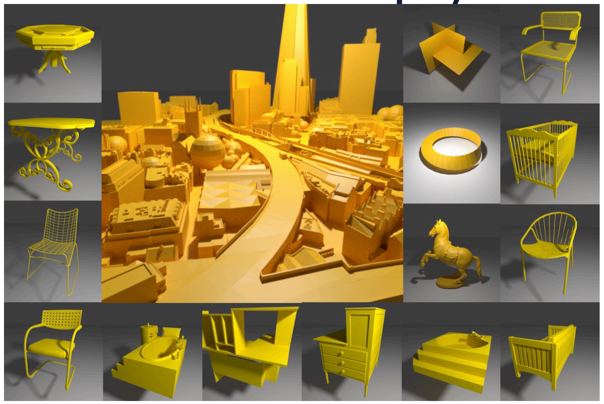
- However, meshes designed by artists or reconstructed by some algorithms are often non-manifold:
  - open boundaries
  - self-intersections
  - incorrect connectivity
  - ambiguous face orientation
  - double surfaces



Hu et al., "**Tetrahedral meshing in the wild.**", *Siggraph 2018*

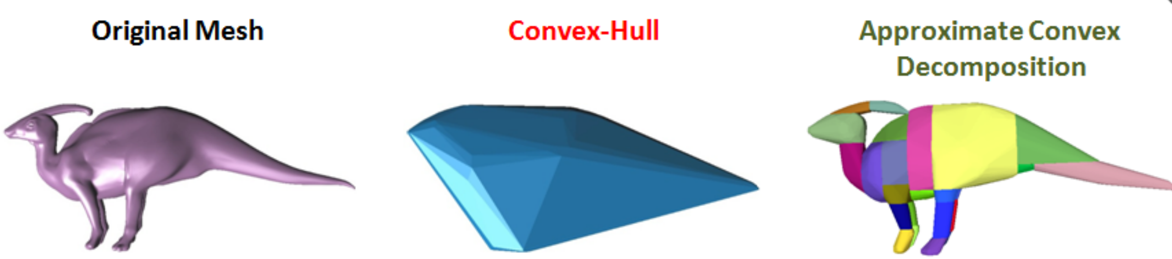
Watertight Manifold Surface Generation

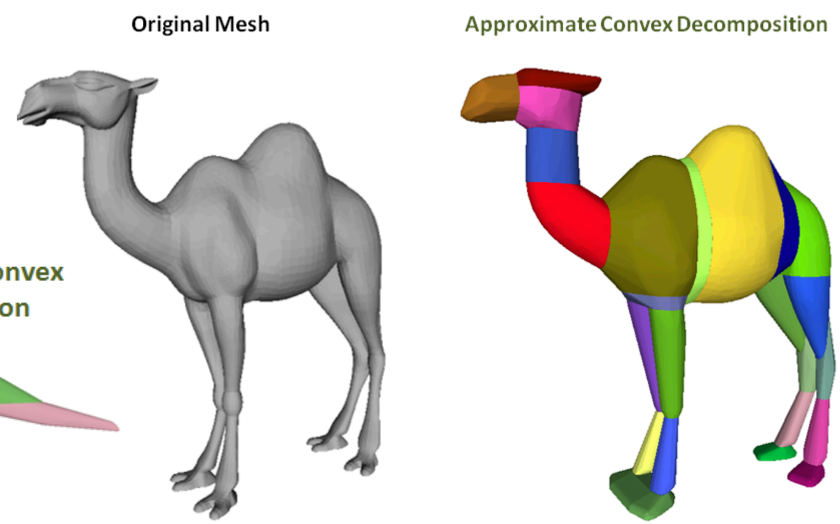
- Huang J, Su H, Guibas L. **"Robust watertight manifold surface generation method for ShapeNet models."**
    - Small artifacts, all surface have volume (https://github.com/hjwdzh/Manifold)
- Huang J, Zhou Y, Guibas L. **"ManifoldPlus: A Robust and Scalable Watertight Manifold Surface Generation Method for Triangle Soups." (https://github.com/hjwdzh/ManifoldPlus)**
    - Less artifacts, may contain zero-volume structures
- Basic steps:
    - Voxelize the surface
    - Extracts exterior faces between occupied voxels and empty voxels
    - Projection-based optimization



Convex Decomposition

- In some applications (e.g., collision detection), convex shapes are an input requirement.
- Directly using convex-hull provides poor approximation for concave surfaces.
- Exact convex decomposition is NP-hard and may produce a high number of primitives.





https://github.com/kmammou/v-hacd

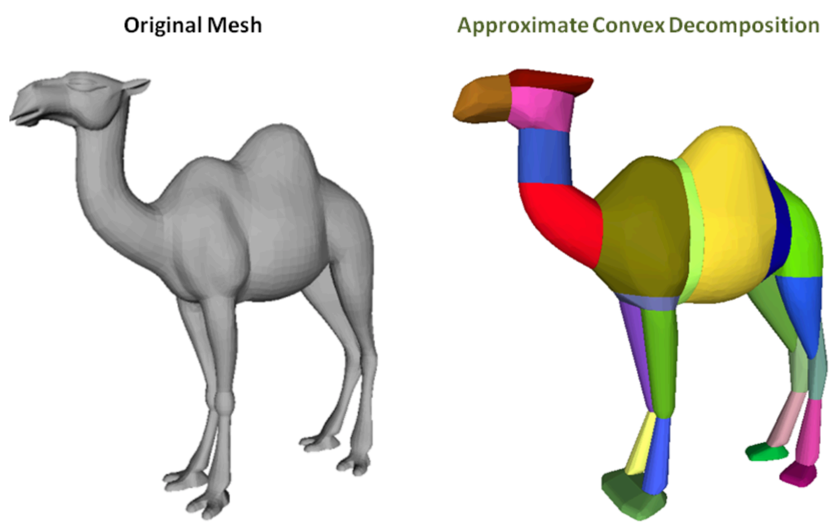
Approximate Convex Decomposition

- Goal: partition the mesh into a minimal number of clusters while ensuring that each cluster has a concavity metric lower than a pre-defined threshold

V-HACD

- First, voxelize the shape
- Then, use a heuristic algorithm (e.g., cut by axis-aligned plane) to segment the shape into parts
- The segmentation optimizes a volume-based concavity measure:

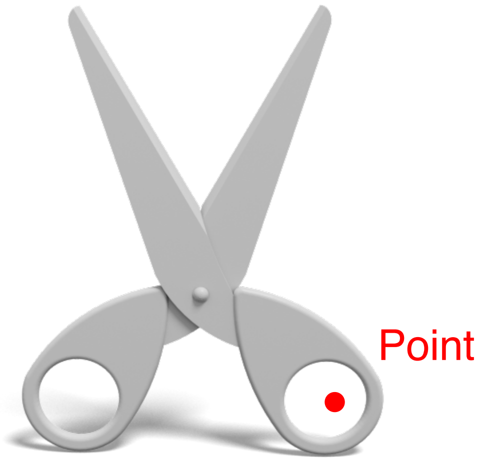
$$\frac{V(ConvexHull(Sub_1)) + V(ConvexHull(Sub_2)) - V(original)}{V(original)}$$



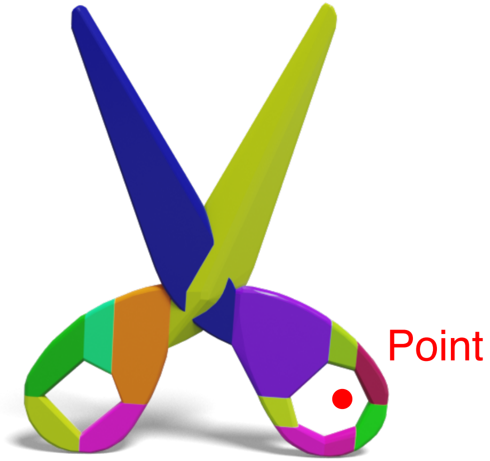
https://github.com/kmammou/v-hacd

Good Concavity Definition

- Preserve Collision Conditions
  - A point: unlikely to collide with input shape $\Longrightarrow$ unlikely to collide with approximation results



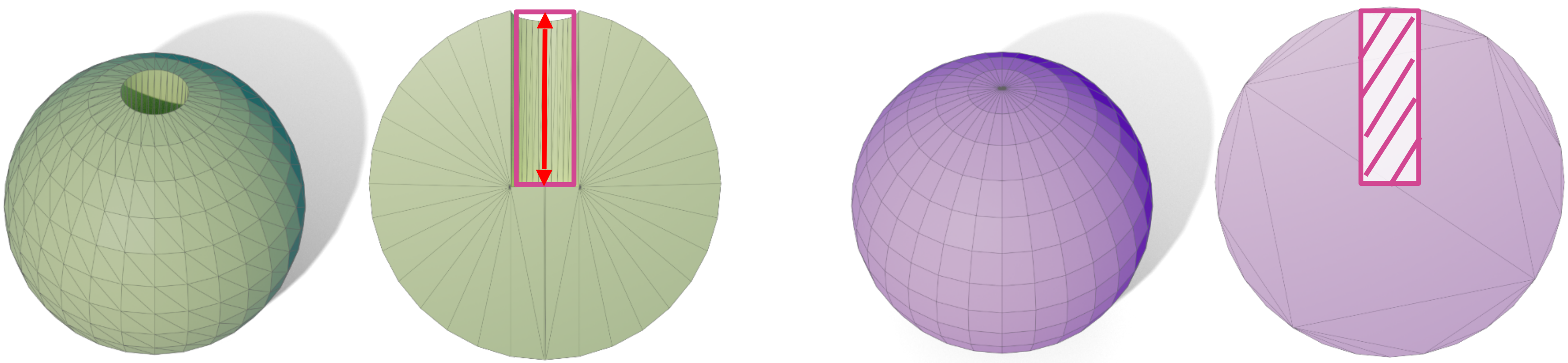
Input Shape



Approximation Result

Issue of V-HACD

- *Volume-based metrics* **alone** are insufficient for preserving collision conditions

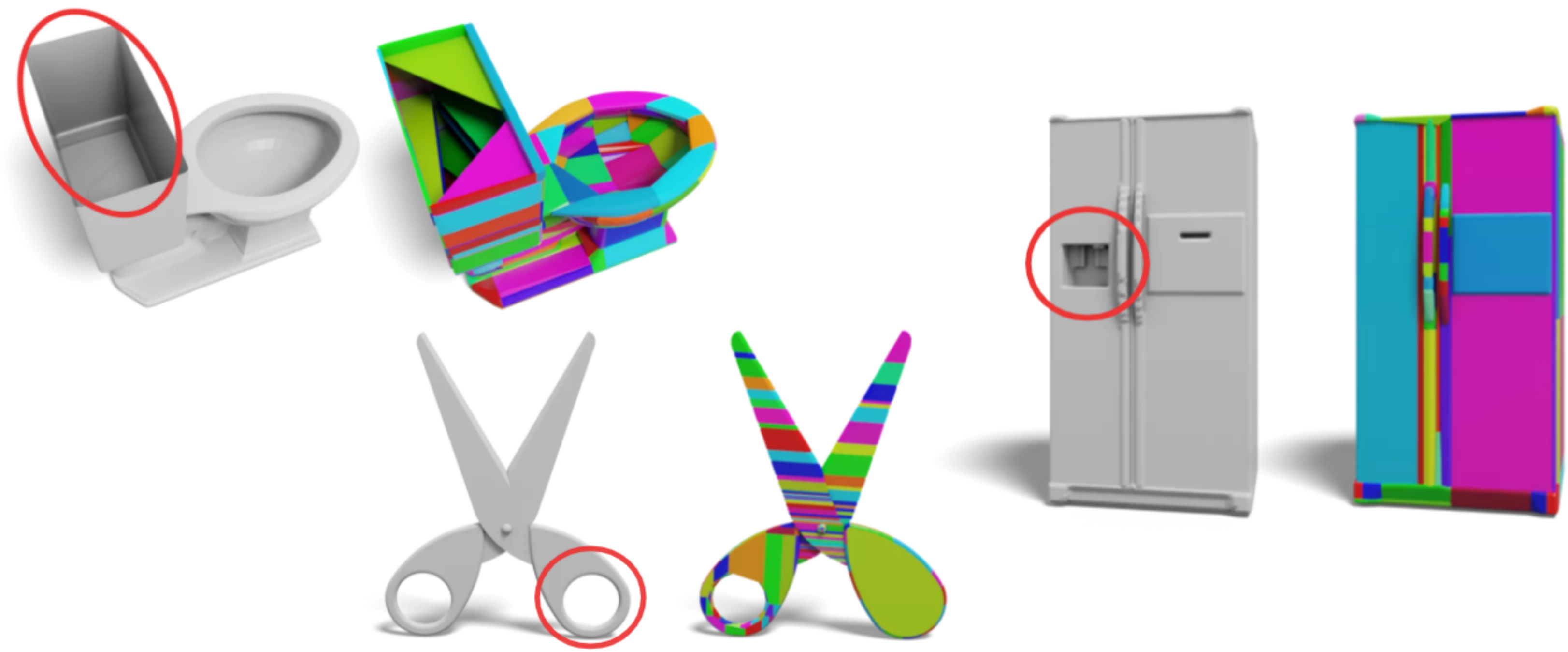


Input Shape ***S*** Convex Hull CH(***S***)

- Filling holes with small volume
  - Volume difference between ***S*** and CH(***S***) is small

Issue of V-HACD

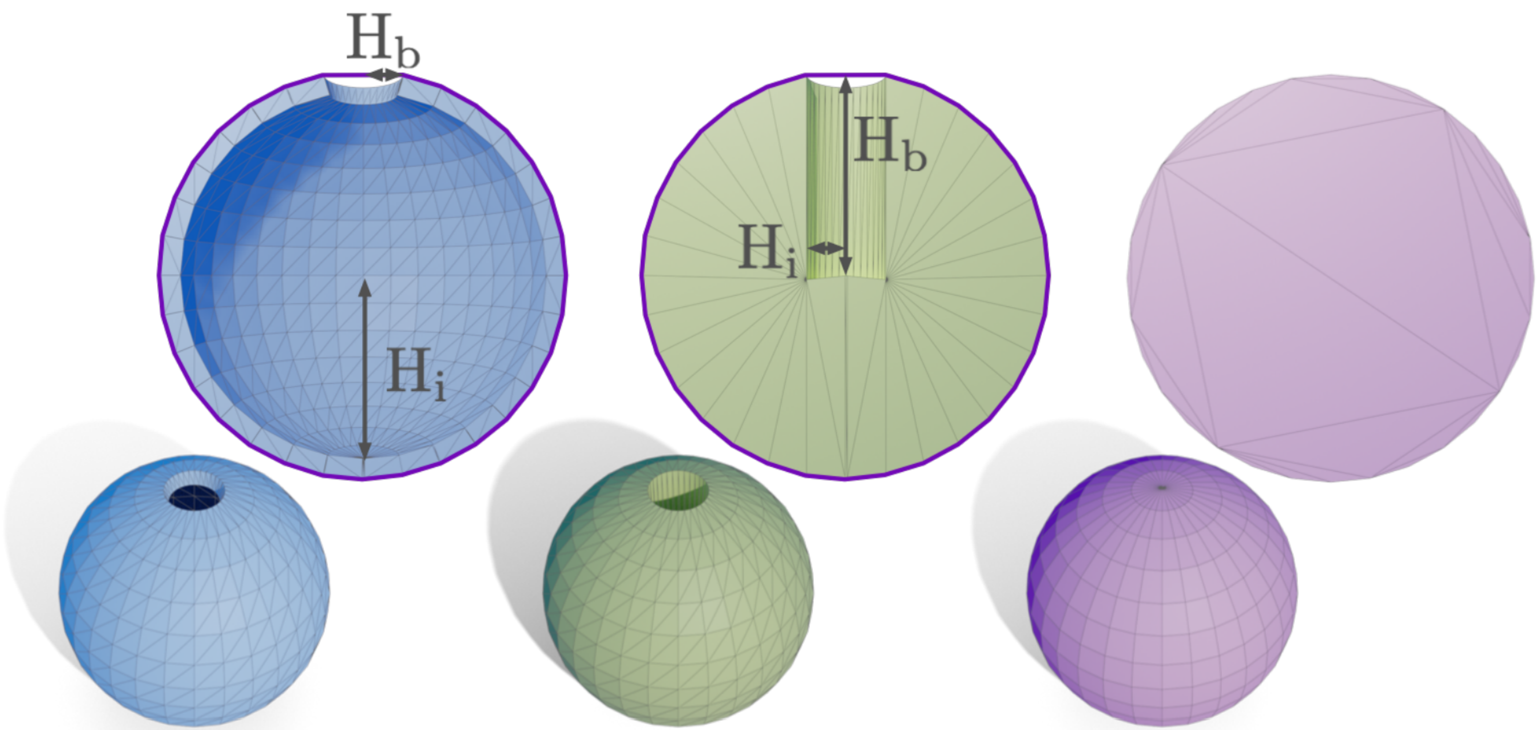
- *Volume-based metrics* **alone** are insufficient for preserving collision conditions



Failure cases from V-HACD

Collision-Aware Concavity

- Boundary + Interior difference
- Sample points + Hausdorff distance



Approximate Convex Decomposition for 3D Meshes with Collision-Aware Concavity and Tree Search, Wei et al., SIGGRAPH2022

Our Collision-Aware Concavity

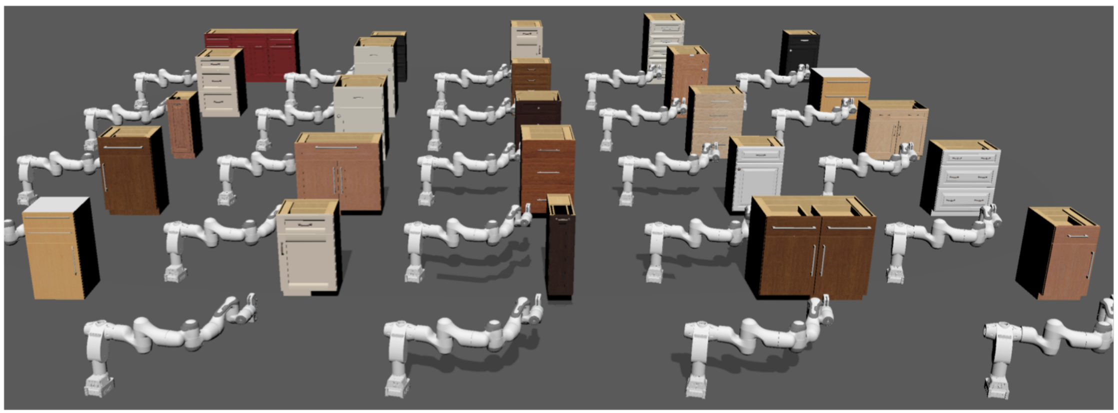
- Fast calculation + theoretical guarantee
  - Surrogate term $\mathrm{R_v}(\mathcal{S})$ for $\mathrm{H_i}(\mathcal{S})$:

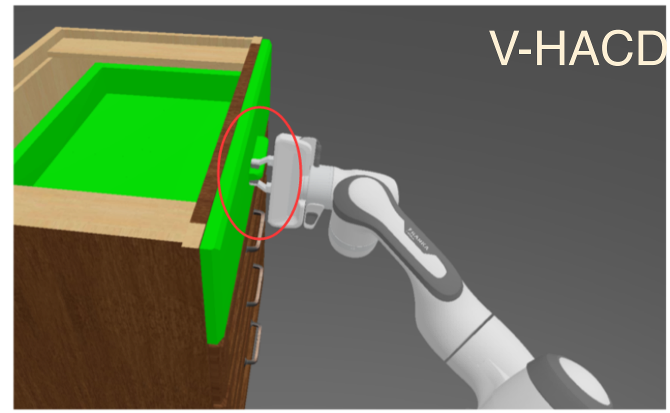
$$\mathrm{R_v}(\mathcal{S}) = \sqrt[3]{\frac{3(\mathrm{Vol}(\mathrm{CH}(\mathcal{S})) - \mathrm{Vol}(\mathcal{S}))}{4\pi}}$$

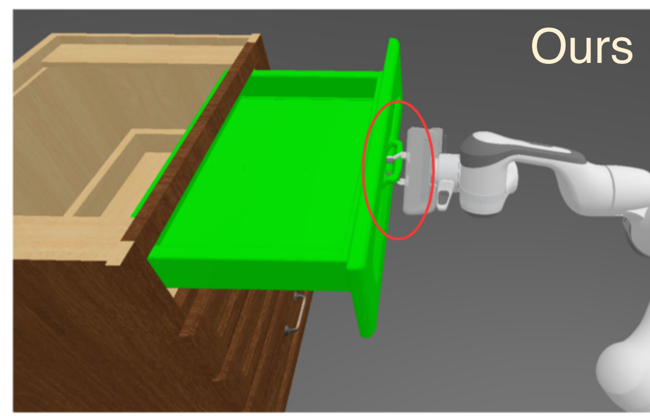
  - THEOREM: For every solid shape ***S***, we have:

$$\sqrt{2}\max(\mathrm{H_b}(\mathcal{S}), \mathrm{R_v}(\mathcal{S})) \geq \max(\mathrm{H_b}(\mathcal{S}), \mathrm{H_i}(\mathcal{S}))$$

Co-ACD: Collision-aware ACD





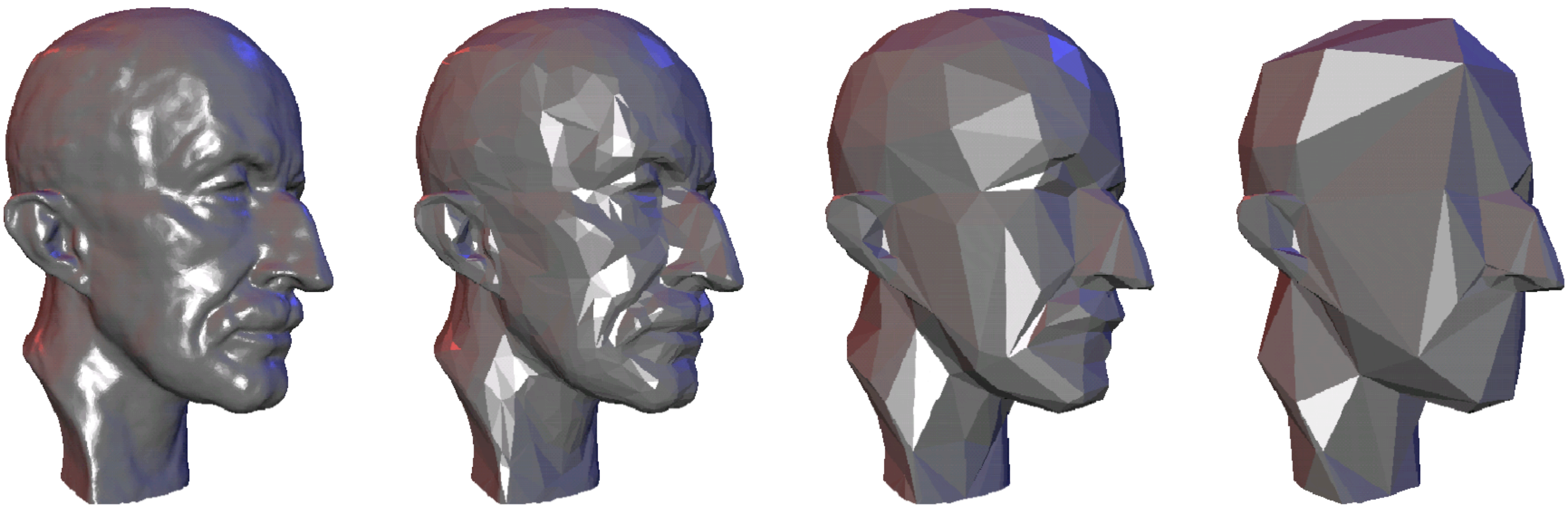


	V-HACD	Ours
Successfully Opened Drawer	49%	**80%**

Approximate Convex Decomposition for 3D Meshes with Collision-Aware Concavity and Tree Search, Wei et al., SIGGRAPH2022

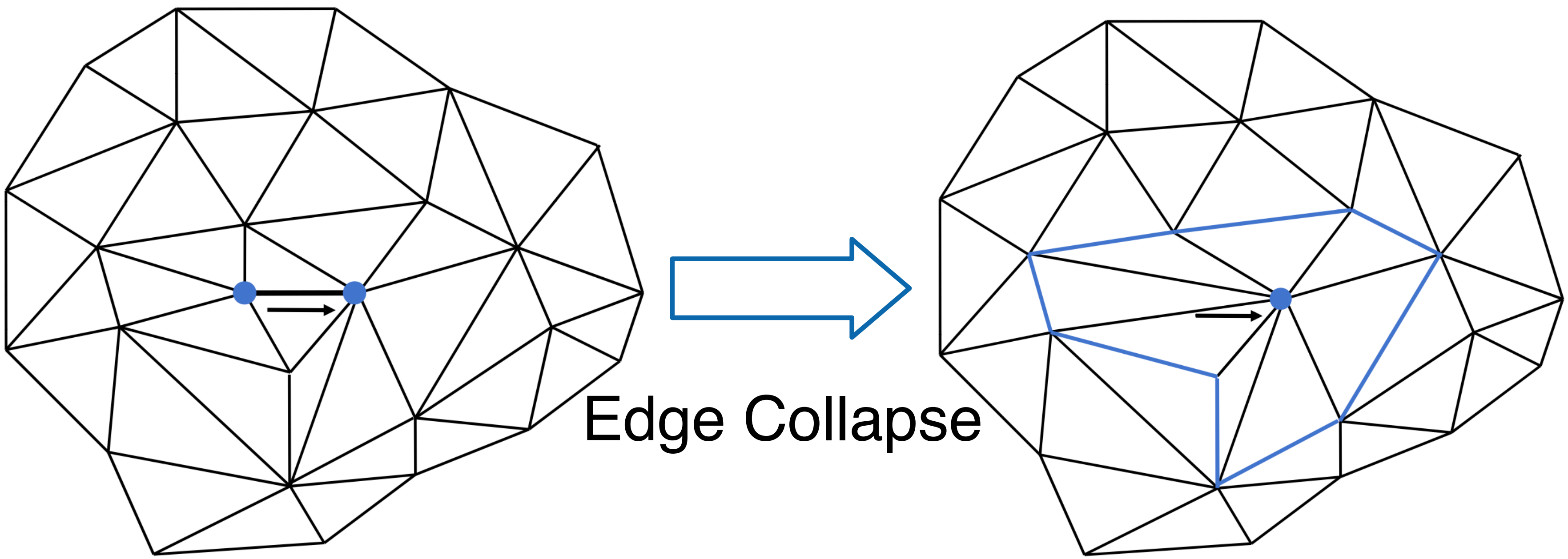
Mesh Simplification (Downsampling)

- Represent a mesh with fewer vertices
- Coarse mesh: memory efficient
- Fine-grained mesh: smooth, details



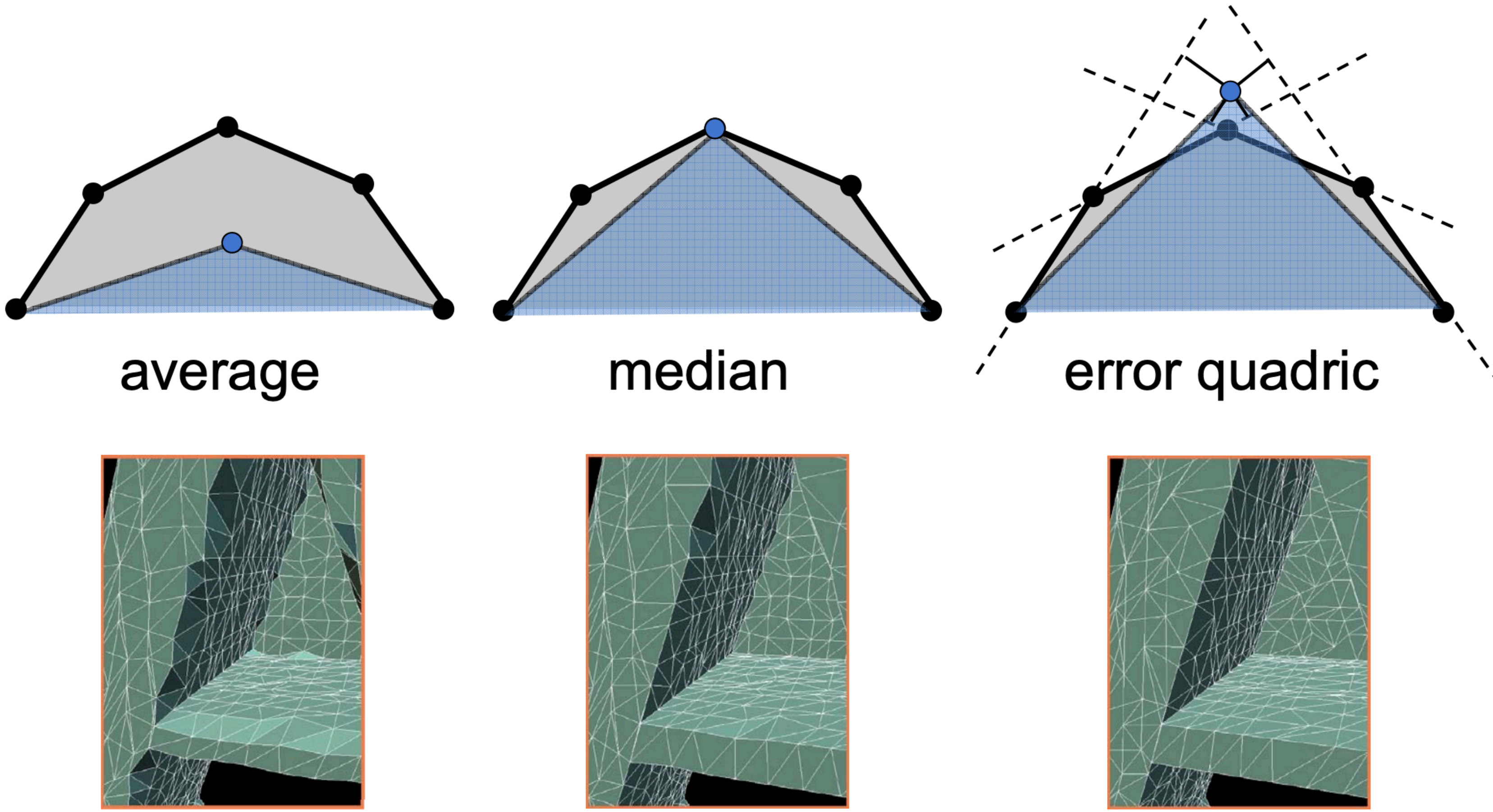
http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/08_Simplification.pdf

Basic Procedure



http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/08_Simplification.pdf

Adiustment of New Node Position



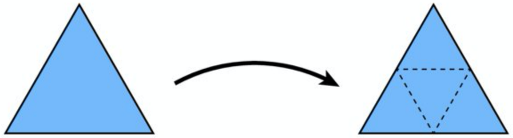
$$\min_p \sum_i \mathrm{dist}(q_i, p)^2$$

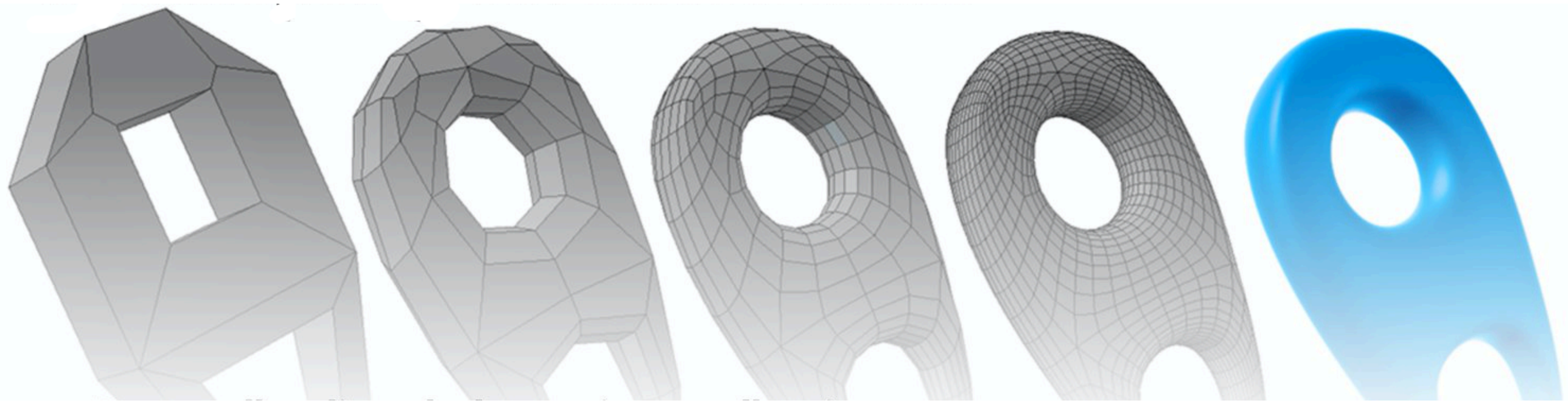
q_i: planes in the cell

http://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/08_Simplification.pdf

Mesh Upsampling via Subdivision

- Recursive upsampling:
  - Divide mesh element (one triangle becomes four)
  - Calculate new positions of the mesh vertices

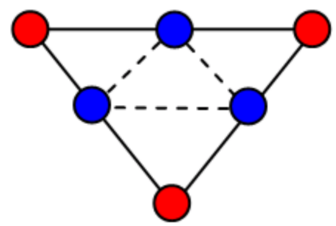




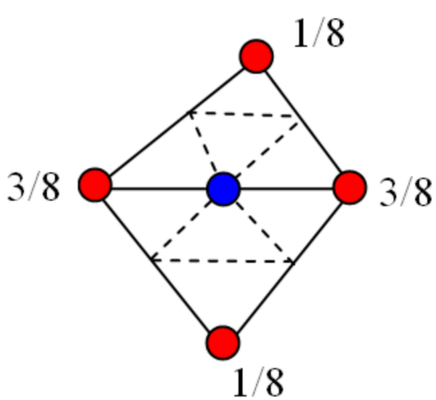
 http://cs248.stanford.edu/winter21/lecture/geometryprocessing/slide_37

Loop Subdivision

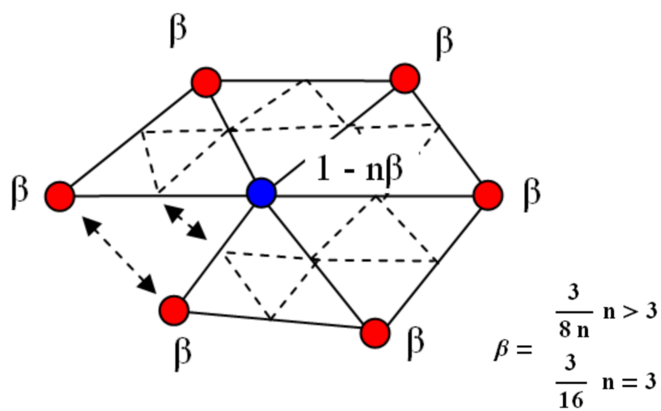
- Divide mesh element (one triangle becomes four)



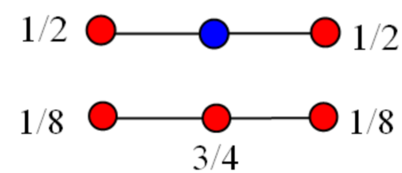
- Calculate new positions of the mesh vertices (linear combinations of original positions)



new vertices



original vertices

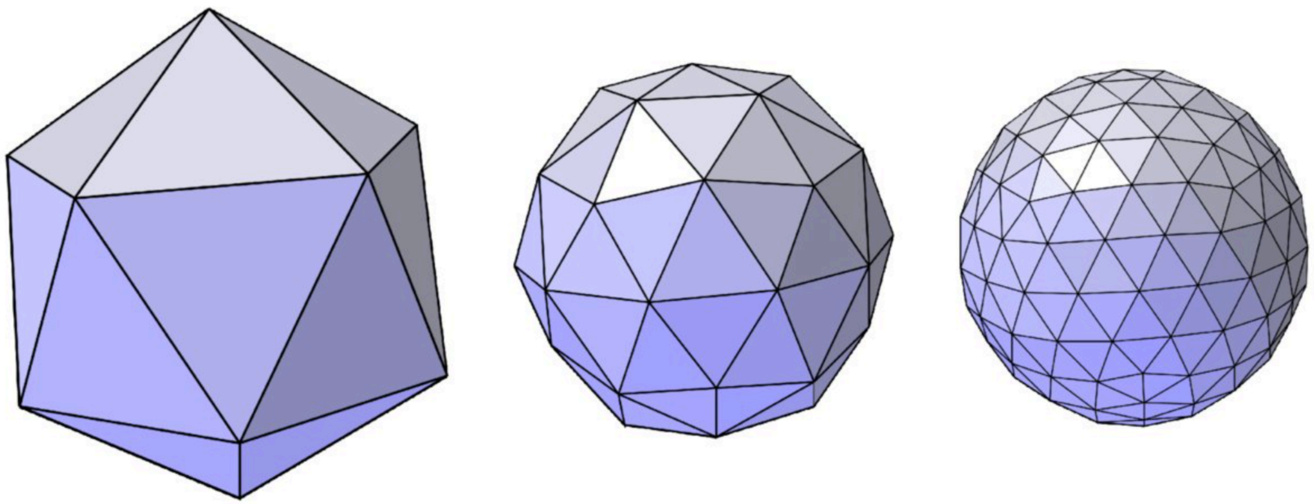


boundary cases

https://graphics.stanford.edu/~mdfisher/subdivision.html

Loop Subdivision

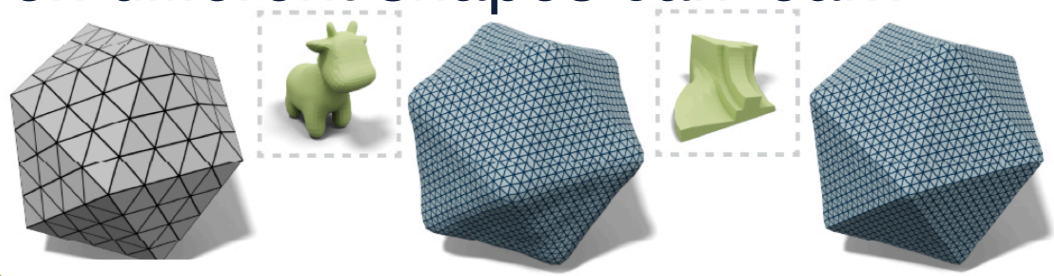
- Common subdivision rule
- “C2” smoothness of its limit shape
- Approximating, not interpolating



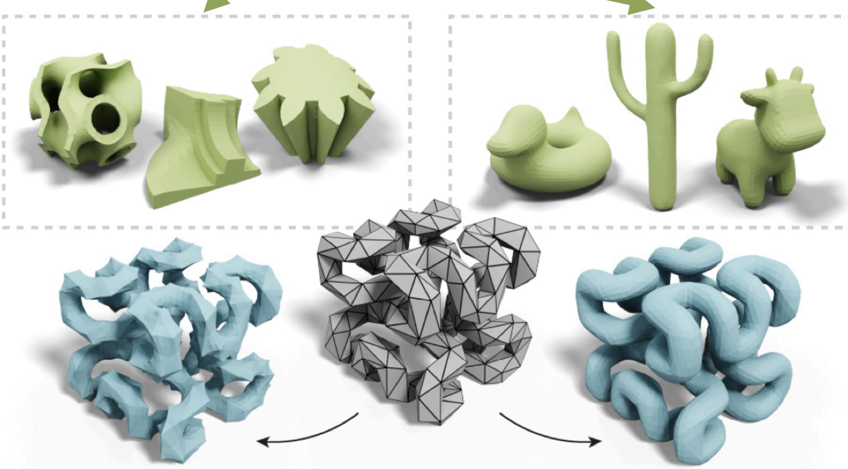
http://cs248.stanford.edu/winter21/lecture/geometryprocessing/slide_37

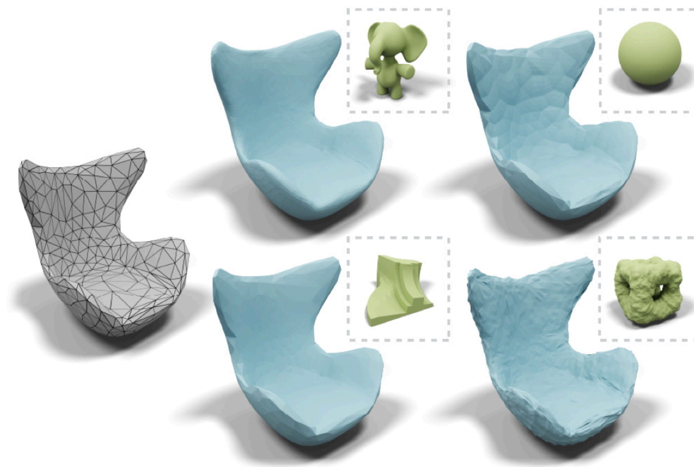
Neural Subdivision

- Using network to predict vertex positions conditioned on the local geometry enables us to learn complex non-linear subdivision scheme (more than smoothing).
- Data-driven: training on different shapes can learn different styles



Two modalities of training data

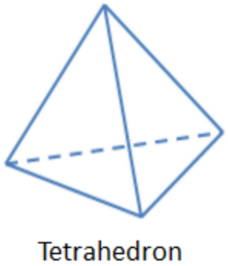


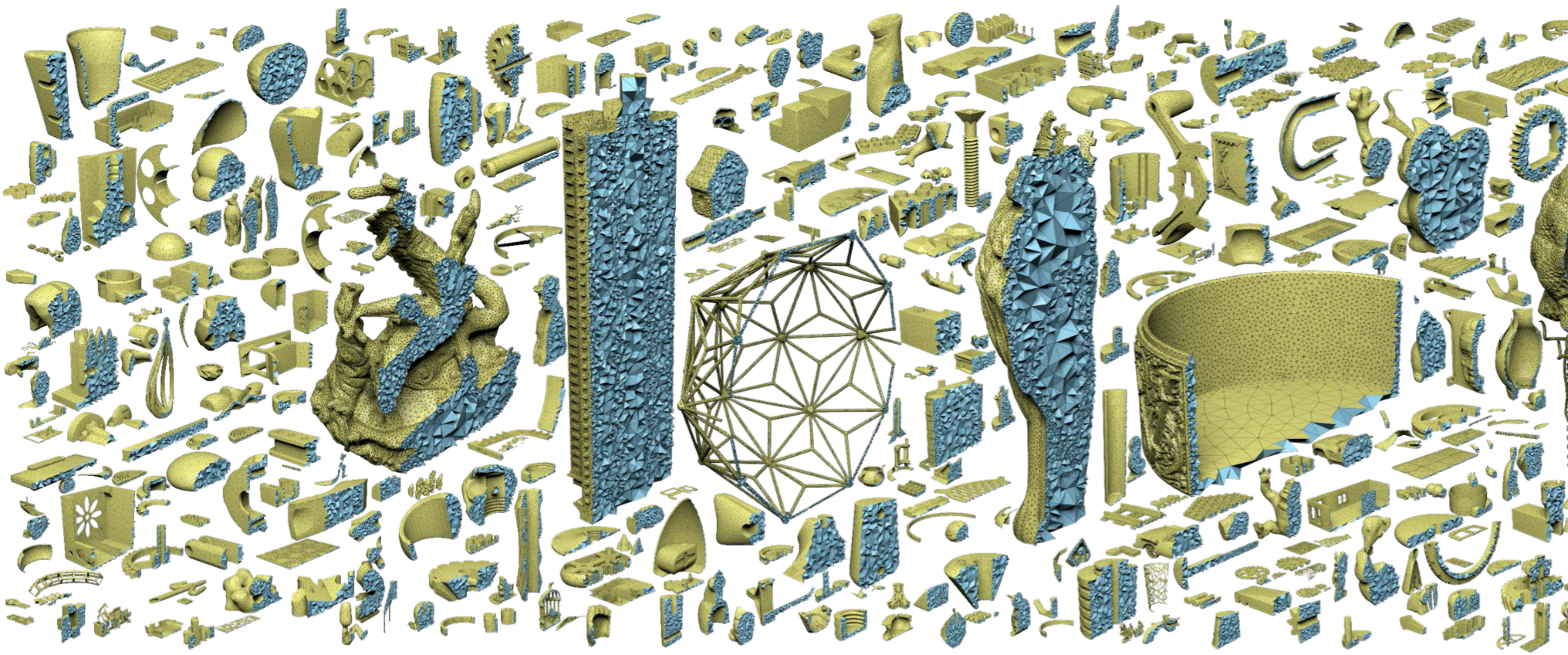


Liu et al., "**Neural subdivision.**", *Siggraph 2020*

Tetrahedral Meshing

- Triangle mesh -> tetrahedral mesh
- Surface (triangle) -> Volume (tetrahedron)
- Finite element method

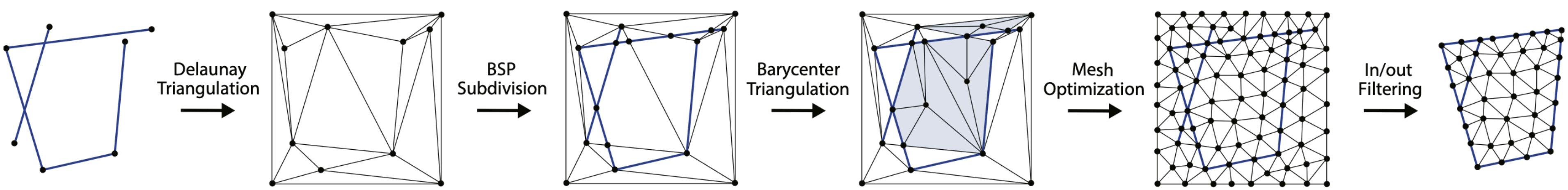




Hu et al., “**Tetrahedral meshing in the wild.**”, *Siggraph 2018*

Tetrahedral Meshing

- Hu Y, Zhou Q, Gao X, Jacobson A, Zorin D, Panozzo D. “**Tetrahedral meshing in the wild**”. Siggraph 2018
- Hu Y, Schneider T, Wang B, Zorin D, Panozzo D. “**Fast tetrahedral meshing in the wild.**” Siggraph 2020



2D pipeline

Hu et al., “**Tetrahedral meshing in the wild.**”, *Siggraph 2018*

Intro to CSE291-D for Spring, 2021

Machine Learning for Robotics

Why Are We Interested In Robotics?

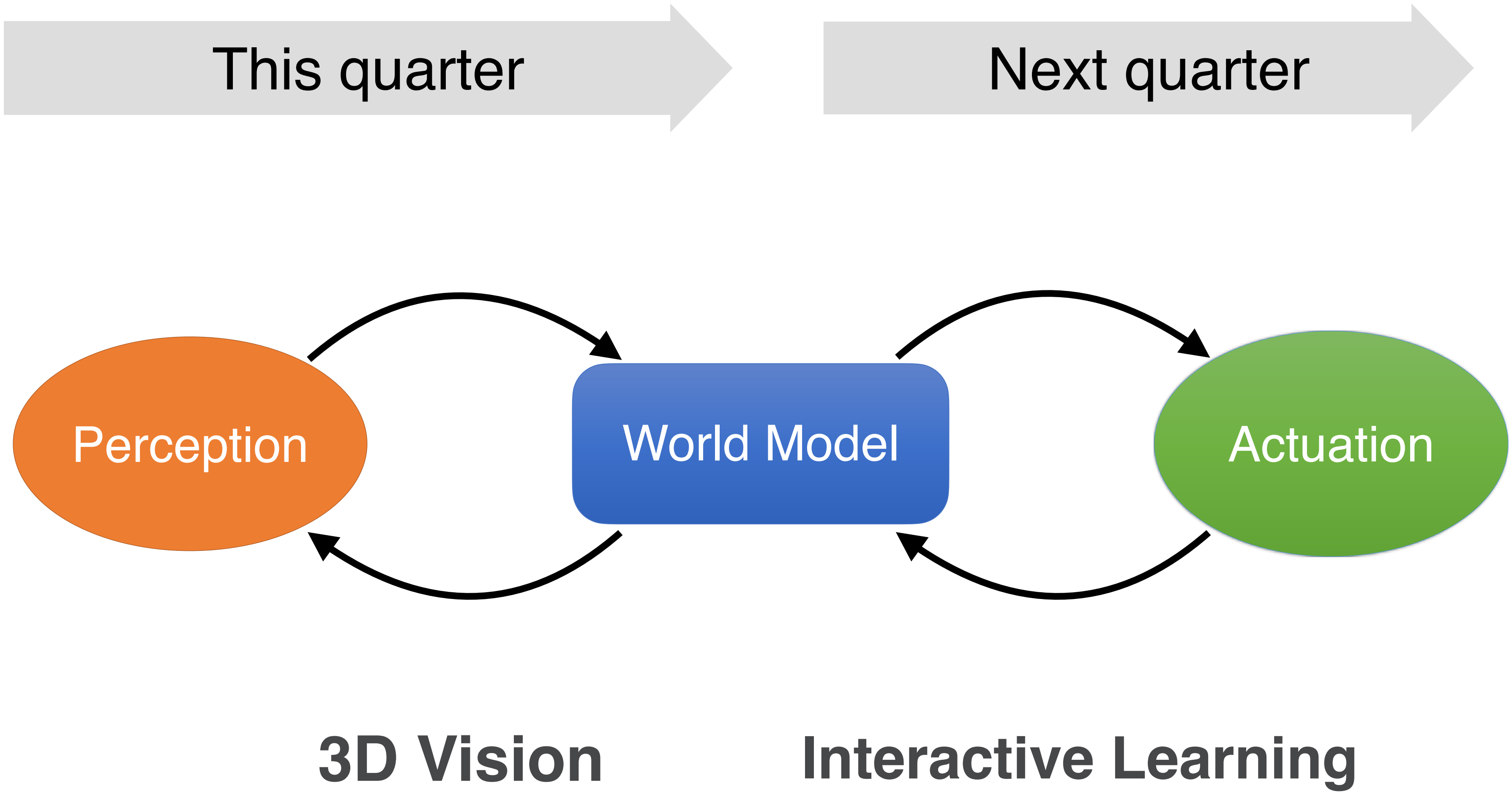
Passive AI

- We know how to fit data well (by “deep learning”)
  - e.g., computer vision, natural language processing

Why Are We Interested In Robotics?

Passive AI → Active AI

- We aspire that autonomous agents can perform tasks and “grow” through interaction experiences
  - Need the ability to **interact**


This quarter
Next quarter
Perception
World Model
Actuation
3D Vision
Interactive Learning

Syllabus

- Part I: Geometry and Physics of Robots
  - Robot Body
  - Robot Motion
  - Robot-Object Interaction
  - Physical Simulation
  - Classical Robotics Pipeline
- Part II: Reinforcement Learning
  - RL Concepts
  - RL as an Optimization Problem
  - Long-horizon RL
  - Generalizable RL

Thank you for the efforts of the the quarter!