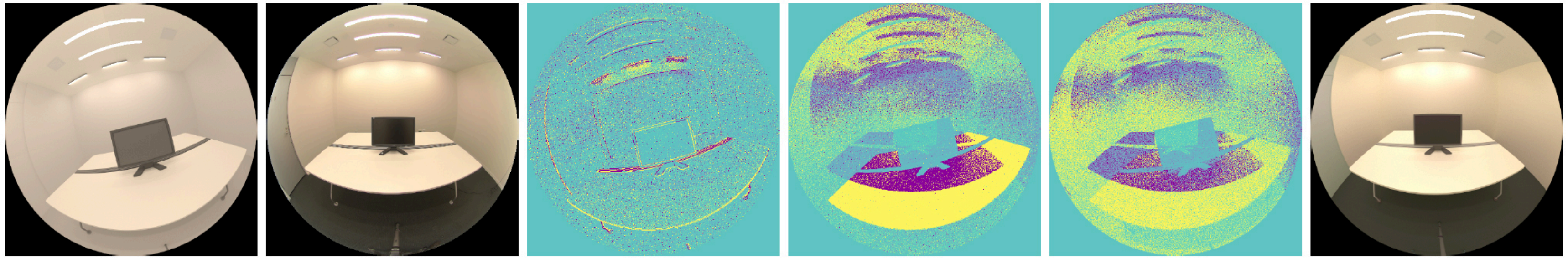
Differentiable Geometry for Optimization

Sanghyun Son

Sep 25th, 2025

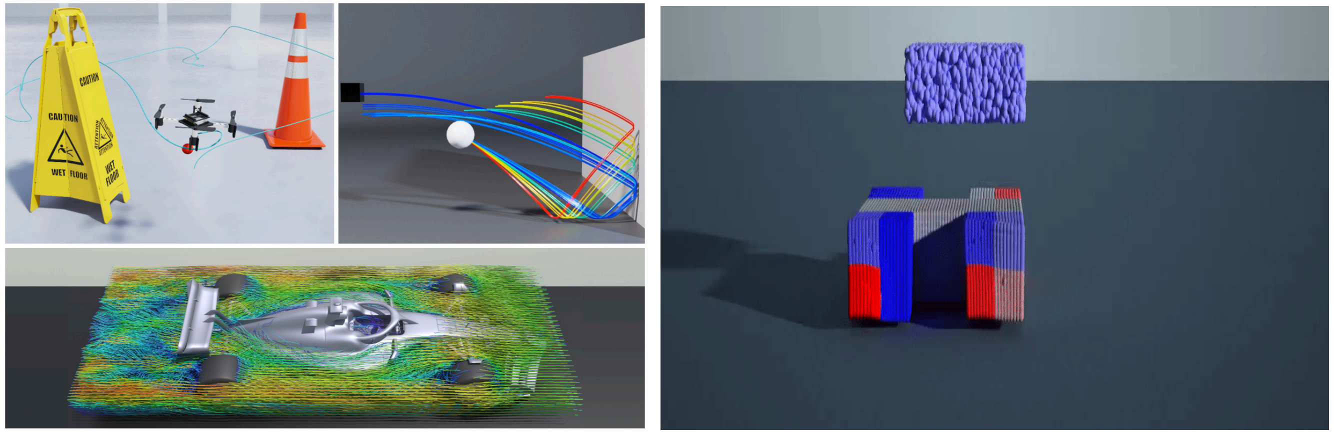
DEPARTMENT OF
COMPUTER SCIENCE

Differentiable systems in the wild

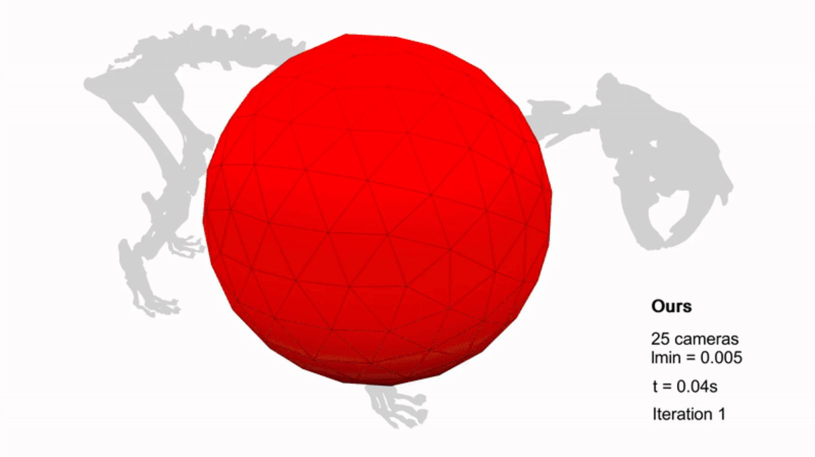


(a) initial guess (b) real photograph (c) camera gradient (per-pixel contribution) (d) table albedo gradient (per-pixel contribution) (e) light gradient (per-pixel contribution) (f) our fitted result

Differentiable Rendering



Differentiable Physics Simulation



Differentiable Geometry

What is Differentiable System?

Input

$\mathbb{X}$

$(x_1, x_2, \ldots, x_n)$

Physical states at time s

(e.g. ↓ position of cha

Physics Simulator

System (Program)

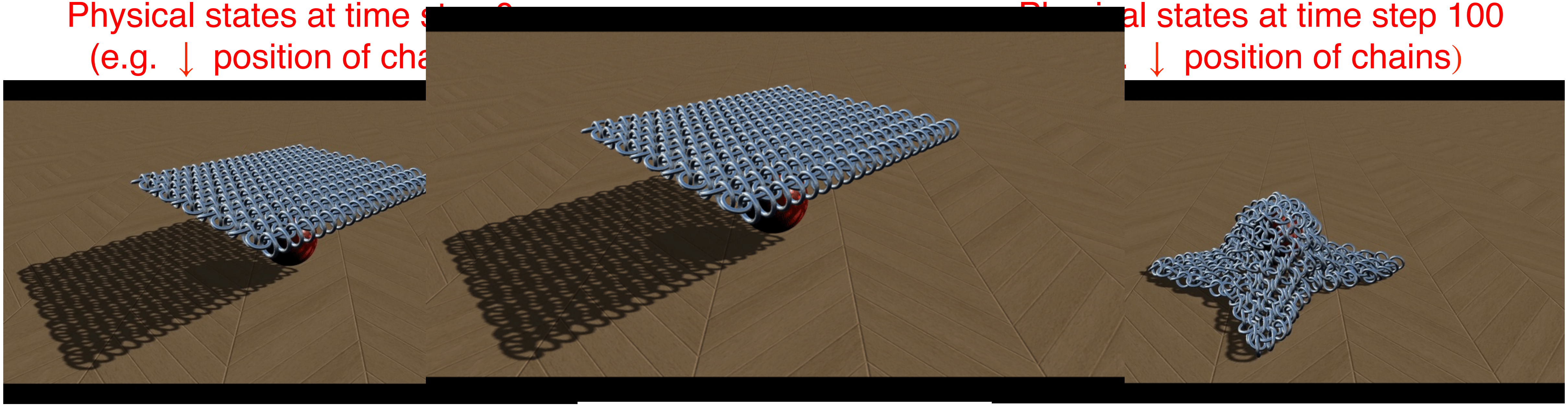
Output

$\mathbb{Y}$

$(y_1, y_2, \ldots, y_m)$

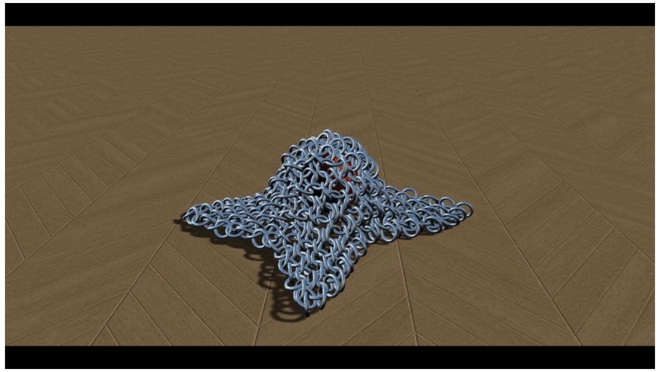
al states at time step 100

↓ position of chains)



What is Differentiable System?

Physical states at time step 100
(e.g. ↓ position of chains)



Output

$$\mathbb{Y}$$

$$(y_1, y_2, ..., y_m)$$

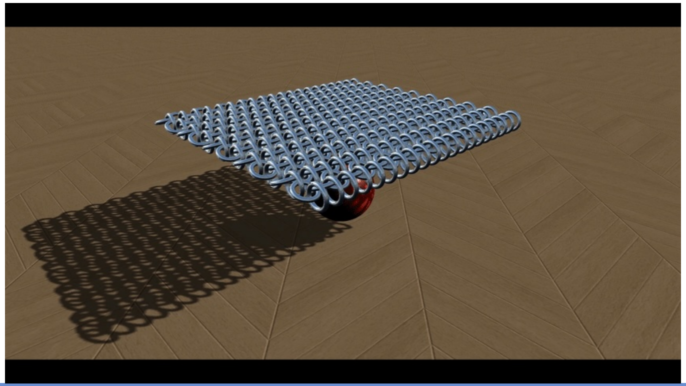
Diff. Physics Simulator

Input

$$\mathbb{X}$$

$$(x_1, x_2, ..., x_n)$$

Physical states at time step 0
(e.g. ↓ position of chains)



$\mathbb{X} \rightarrow$ Differentiable System $\rightarrow$ Output / Gradients

Gradients

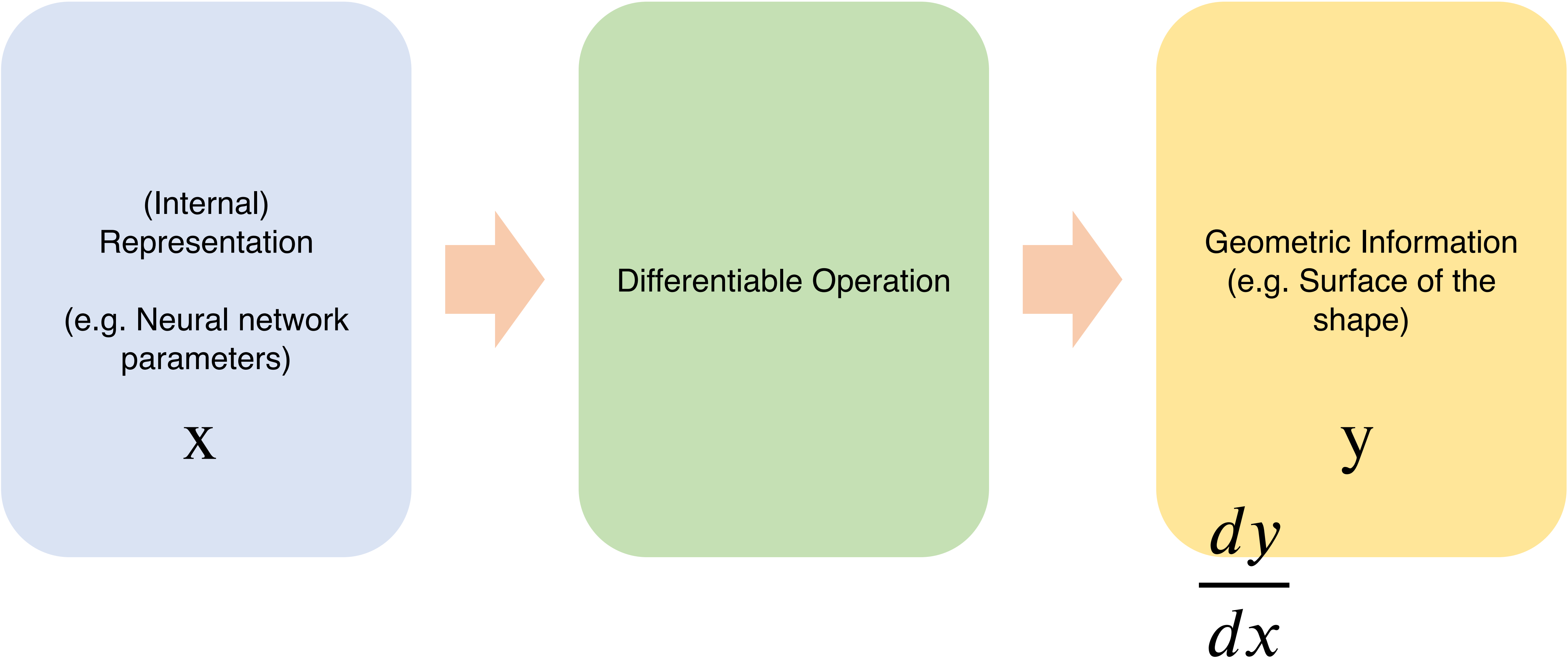
$$\frac{d\mathbb{Y}}{d\mathbb{X}}$$

$$\left(\frac{dy_1}{dx_1}, \frac{dy_1}{dx_2}, \dots \frac{dy_m}{dx_n}\right)$$

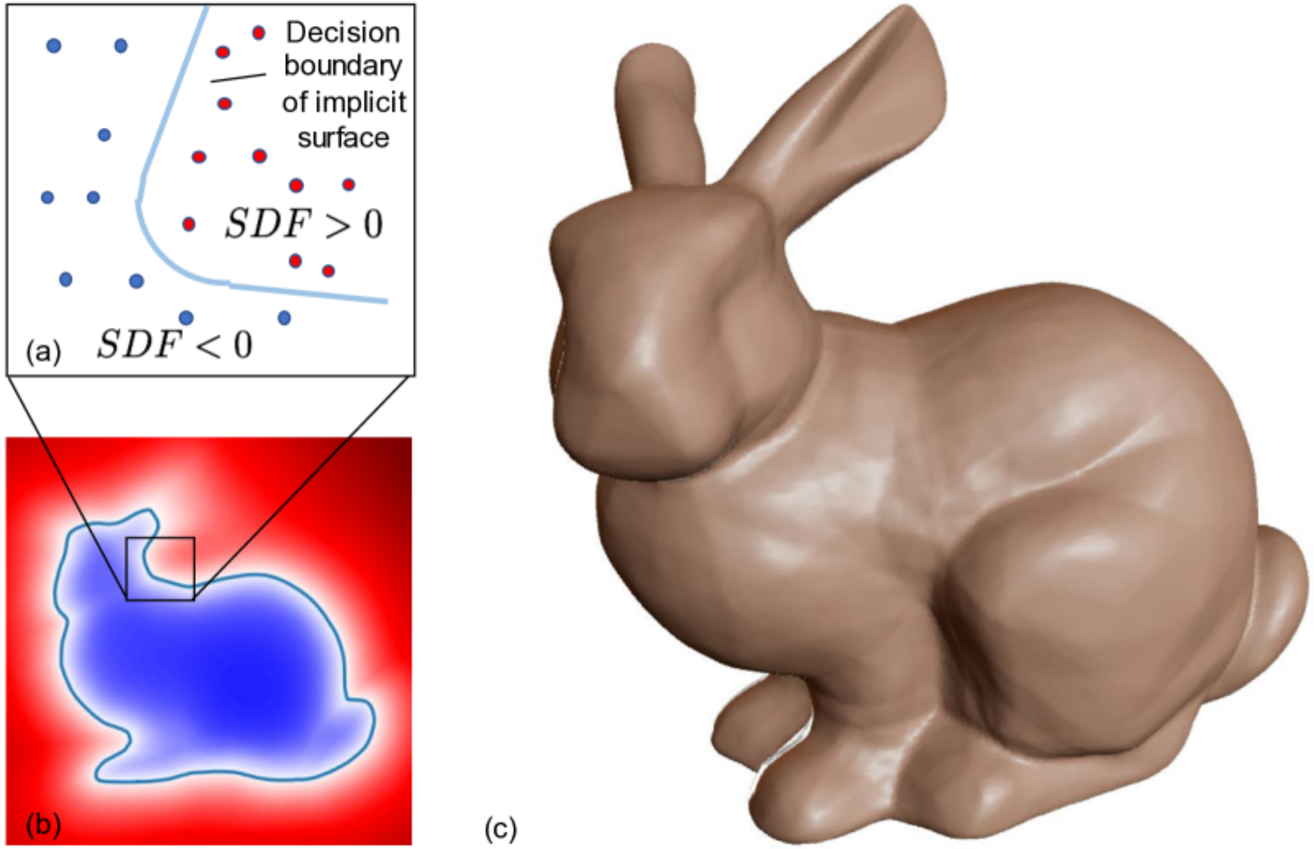
Gradients of Physical States

e.g. How does the chain positions at time step 100 change when those at time step 0 change?

Differentiable Geometry

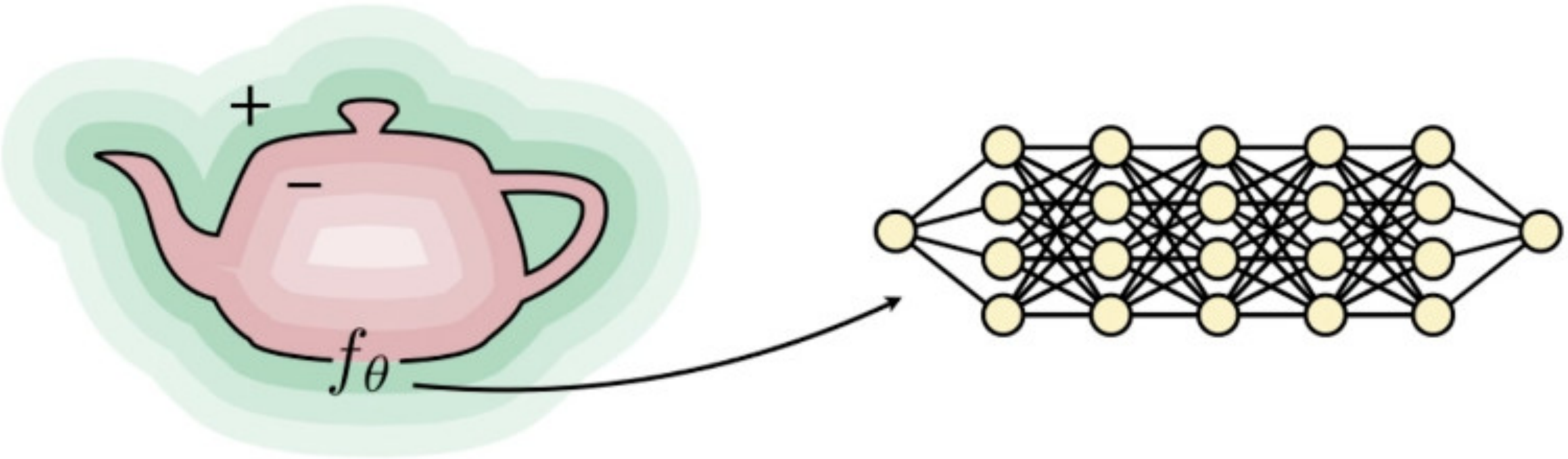


Case Study) Signed Distance Function (SDF)



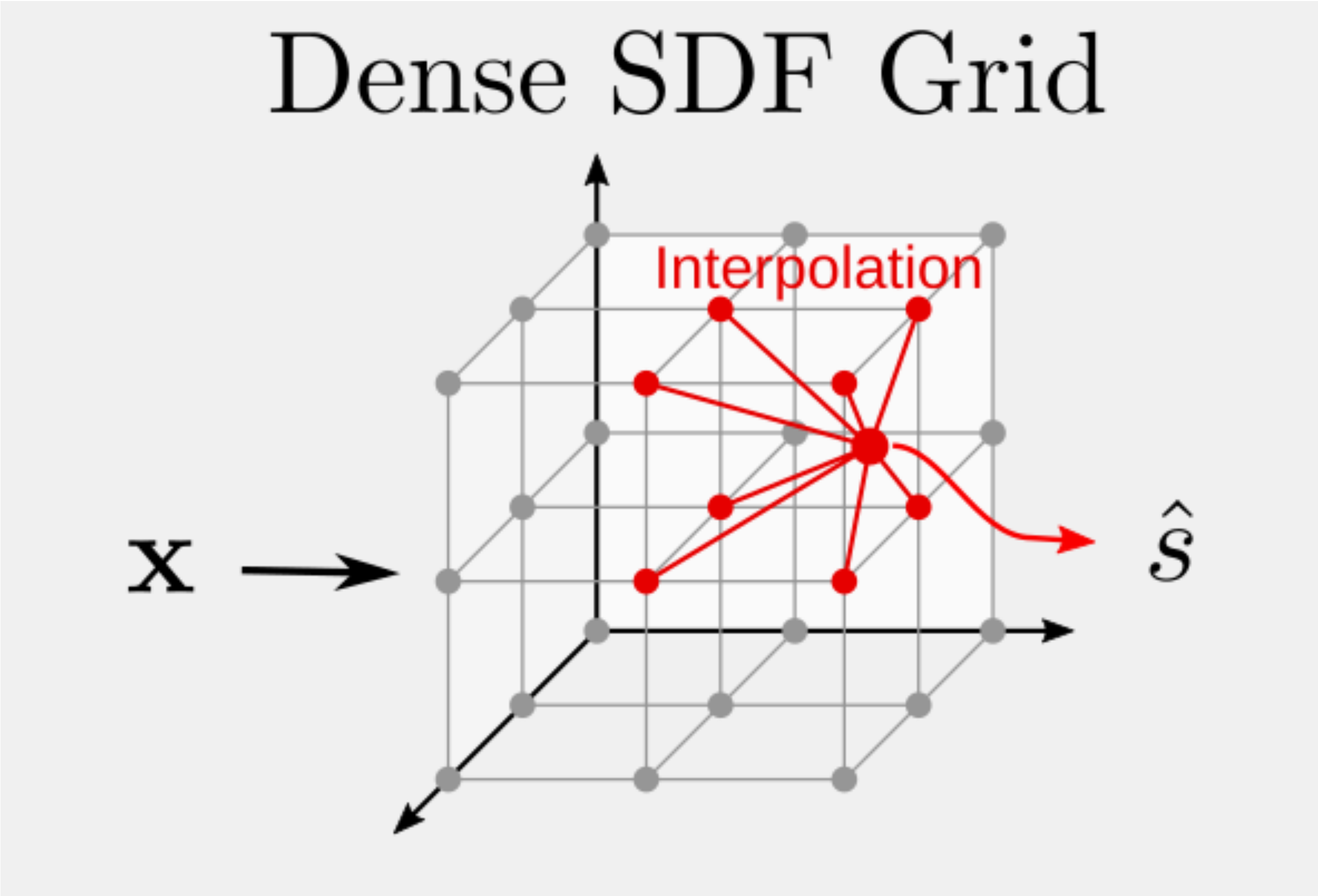
Function that returns signed distance of the given query point

SDF: Representation + Operation

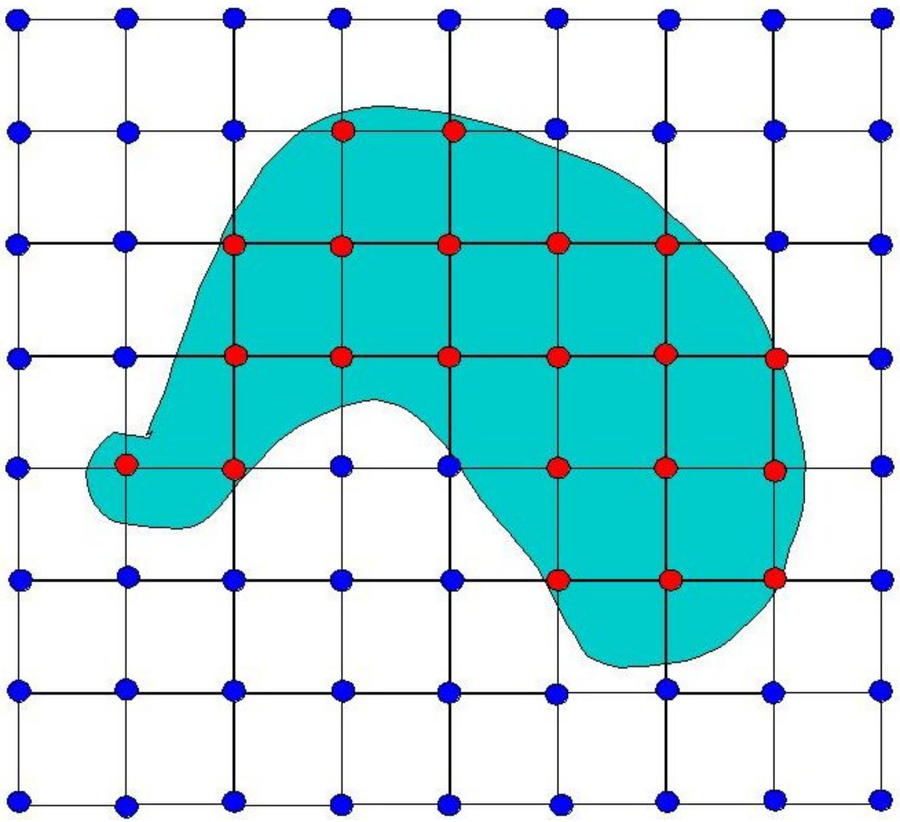


Neural network parameters
(Neural SDF)

SDF: Representation + Operation

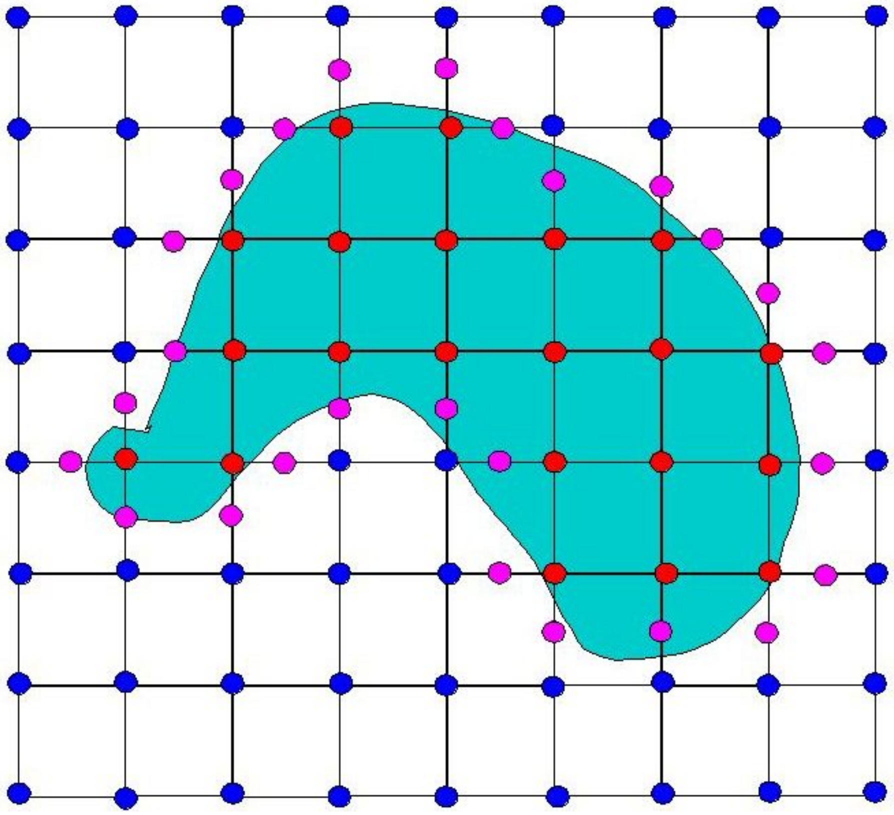


SDF: Operation (Marching Cubes)



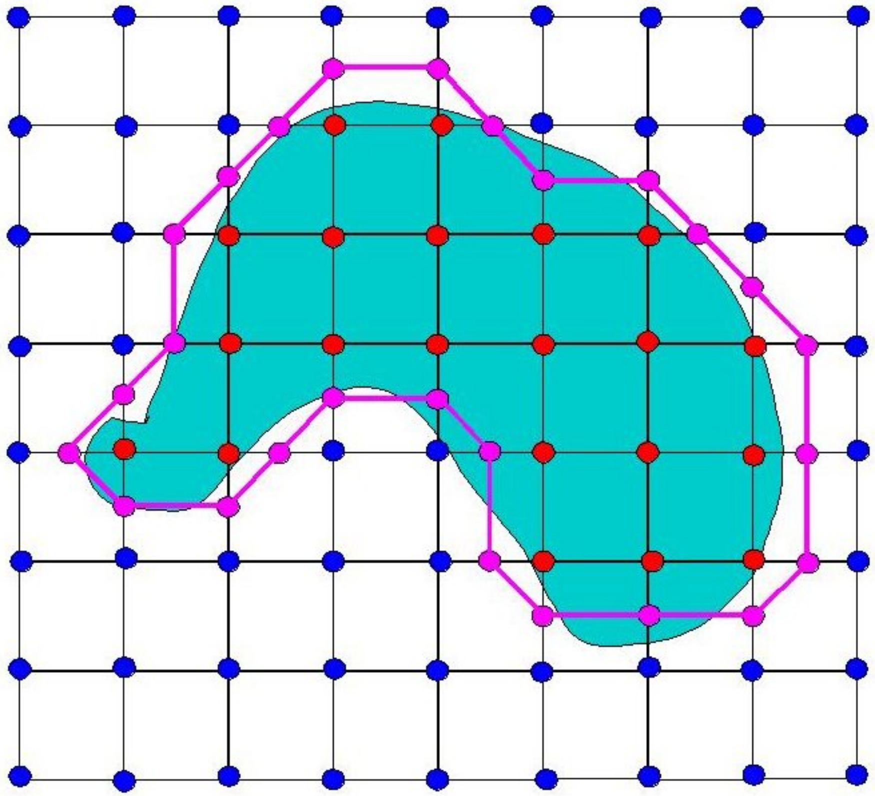
For the given SDF, evaluate signed distance on the grid points
(Red = inside, Blue = outside)

SDF: Operation (Marching Cubes)



For the edges that connect inside and outside,
find the zero-crossing point by interpolation

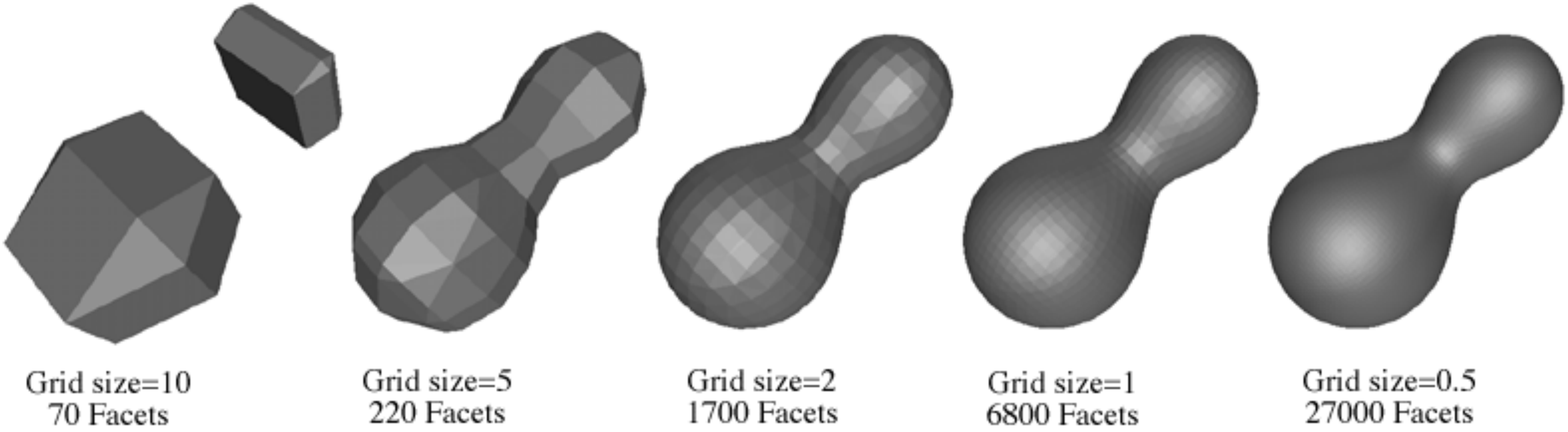
SDF: Operation (Marching Cubes)



Connect zero-crossing points using pre-defined rules

→ Note that the **operations that we used are all differentiable**!

SDF: Operation (Marching Cubes)



With Marching Cubes, we can extract **surface mesh** from SDF

As the grid becomes denser, the extracted surface becomes more accurate

Case Study) Signed Distance Function (SDF)

1. Network parameters (Neural SDF)

2. SDF encoded grid points

→

For a given point p, get $\mathrm{SDF}(p)$

1. Giving p as an input to the network

2. Interpolating grid SDF values based on p

→

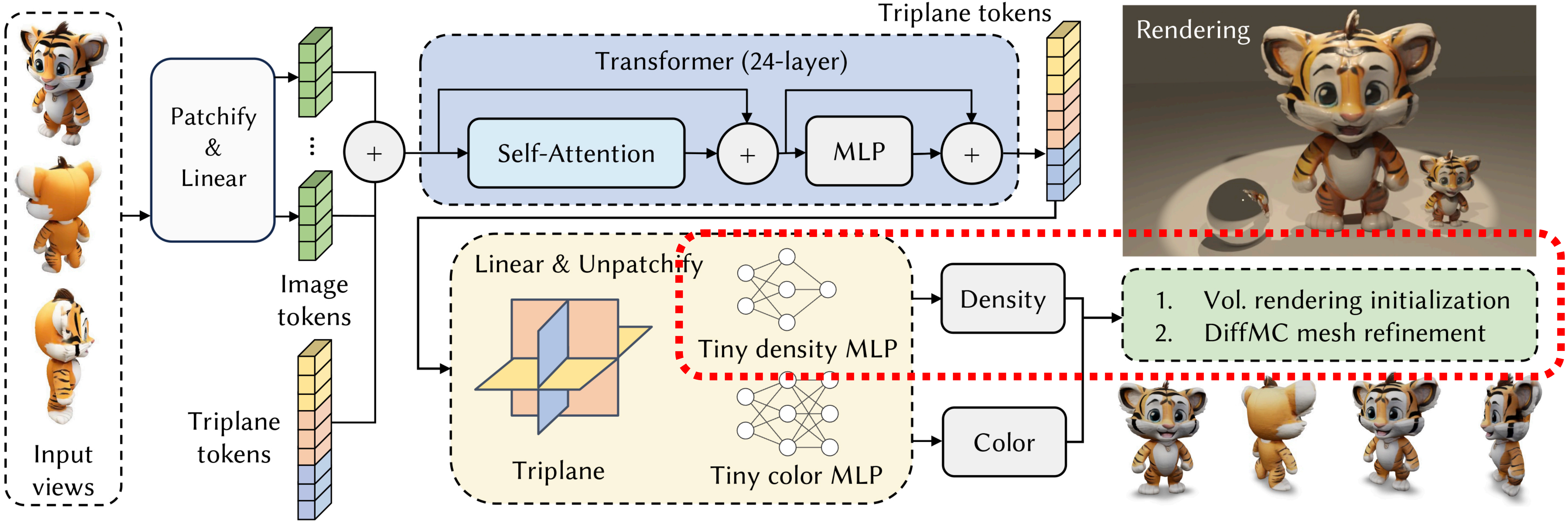
Differentiable Marching Cube

→

Mesh surface

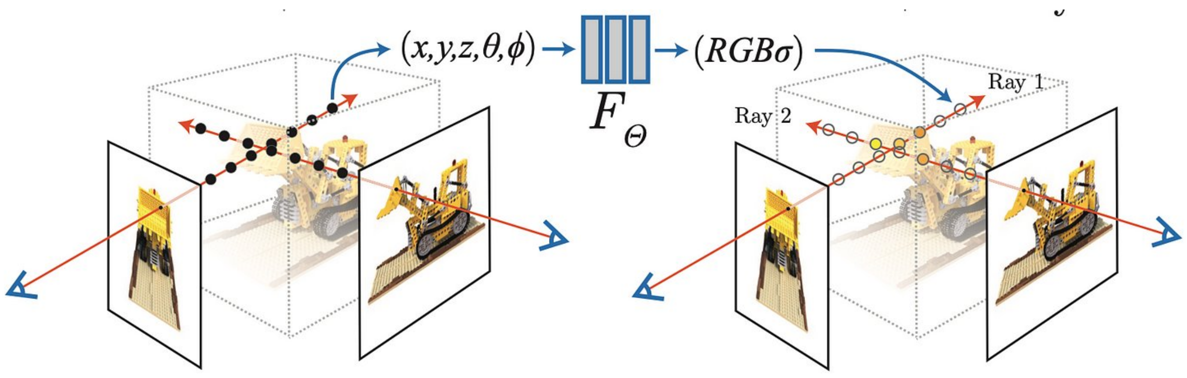
Finally, compute the loss (e.g. rendering loss) on the mesh surface and backpropagate to the representations to update them!

Case Study) Signed Distance Function (SDF)

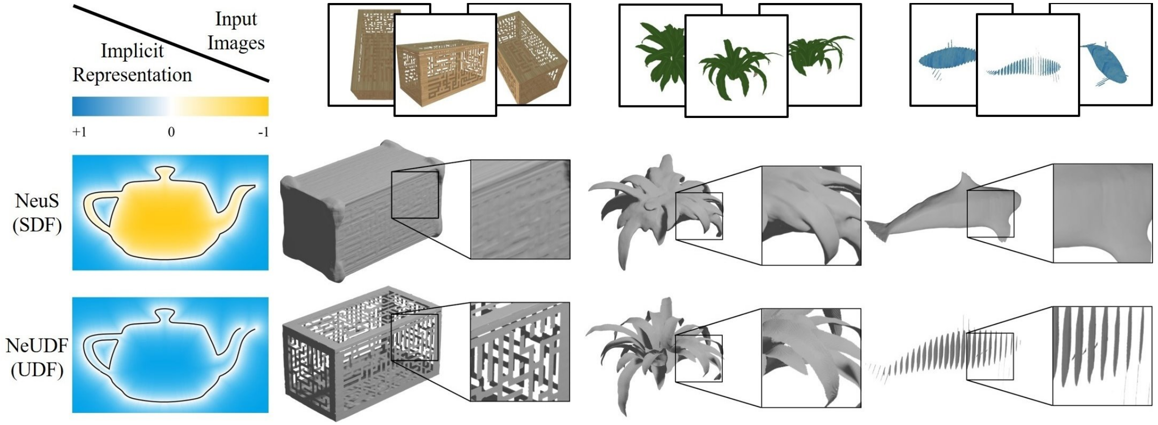


1. Predict SDF from density field
2. Extract surface mesh from SDF
3. Render the surface mesh and compute the rendering loss

Various 3D representations

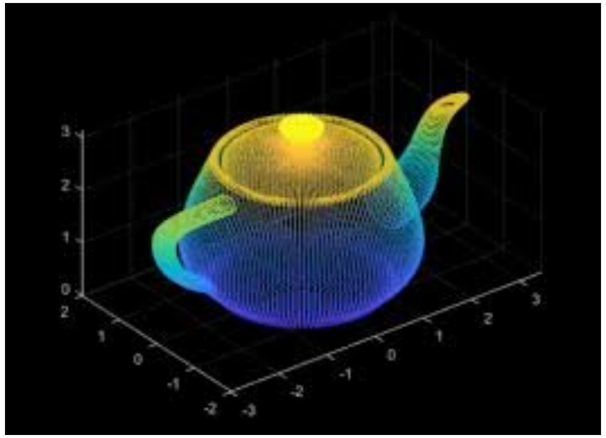


Neural Radiance Fields (NeRF)

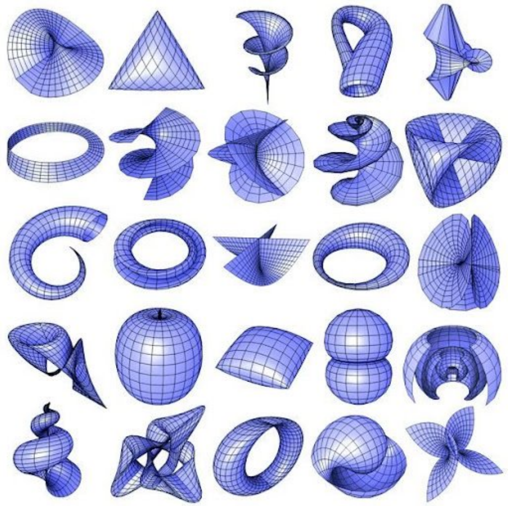


Neural Distance Fields

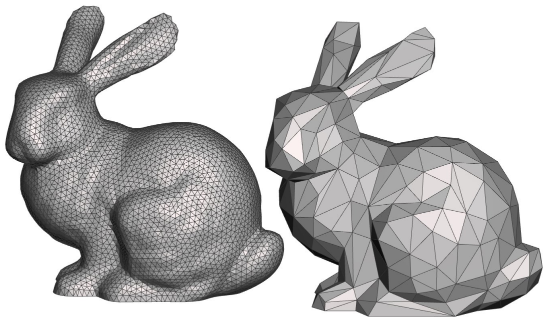
(Neural) Implicit Representations



Point Cloud



Parametric Surfaces



(Triangular) Mesh

3D Gaussian Splatting

Explicit Representations

Various 3D representations

- (Neural) **Implicit representations are preferred** over explicit representations in the current ML pipeline
  - High representation power
  - Differentiability

- Recently, **3D Gaussian Splatting** gained popularity
  - Similar representation power as NeRF
  - Differentiable, but not as differentiable as implicit rep.
  - Much less computational cost than neural implicit rep.

- How about the other explicit representations, especially **Mesh**?

Mildenhall, Ben, et al. "Nerf: Representing scenes as neural radiance fields for view synthesis." *Communications of the ACM* 65.1 (2021): 99-106.
Wang, Peng, et al. "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction." *arXiv preprint arXiv:2106.10689* (2021).
Long, Xiaoxiao, et al. "Neuraludf: Learning unsigned distance fields for multi-view reconstruction of surfaces with arbitrary topologies." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023.
Kerbl, Bernhard, et al. "3D Gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.* 42.4 (2023): 139-1.

DMesh: A Differentiable Mesh Representation

Sanghyun Son, Matheus Gadelha, Yang Zhou, Zexiang Xu, Ming C. Lin, Yi Zhou

NeurIPS 2024

UNIVERSITY OF MARYLAND 1856
DEPARTMENT OF
COMPUTER SCIENCE

Why Mesh?

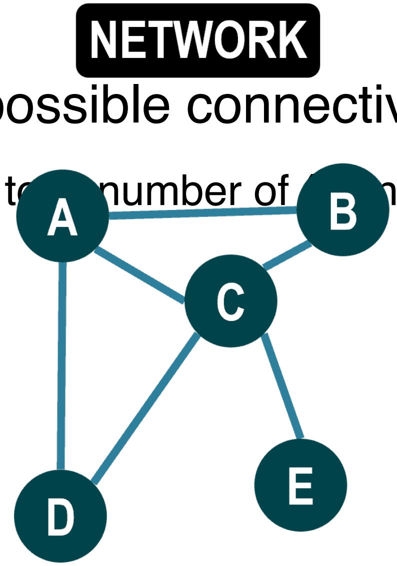
- **Efficient** data structure
  - Vertices
  - Connectivity (Edge, Face)
- Optimized pipeline for **downstream tasks**
  - Rendering
  - Physics simulation
- Easy to **modify & control**
  - 3D modeling tools (e.g. Blender)

Marching Cubes MeshAnything

Chen, Yiwen, et al. "MeshAnything: Artist-Created Mesh Generation with Autoregressive Transformers." *arXiv preprint arXiv:2406.10163* (2024).

Mesh in ML: Obstacles

- **Two obstacles** in using mesh in the current ML pipeline
  - **Discrete connectivity**
    - If two vertices are connected, encoded as 1
    - Otherwise, encoded as 0
  - **Exponential increase** of possible connectivity
    - When number of points is n, t... number of ...ngular) faces is



CONNECTIVITY MATRIX

	A	B	C	D	E
A	0	1	1	1	0
B	1	0	1	0	0
C	1	1	0	1	1
D	1	0	1	0	0
E	0	0	1	0	0

Mesh in ML: Implicit rep. to mesh

- Solution 1) Circumvent the problem by extracting mesh from implicit representations using **iso-surface extraction algorithm**

  - Optimize **neural implicit representations (or 3D GS)** and extract mesh
    - +) Preserve fine detail, can represent various topology
    - -) Excessive computational cost, bad mesh quality

  - Optimize **vertex-wise signed distance values** and extract mesh
    - +) Much more efficient and better mesh quality than neural representations
    - -) Limited topology (mainly volume), self-intersections, hardly extensible to scene-scale

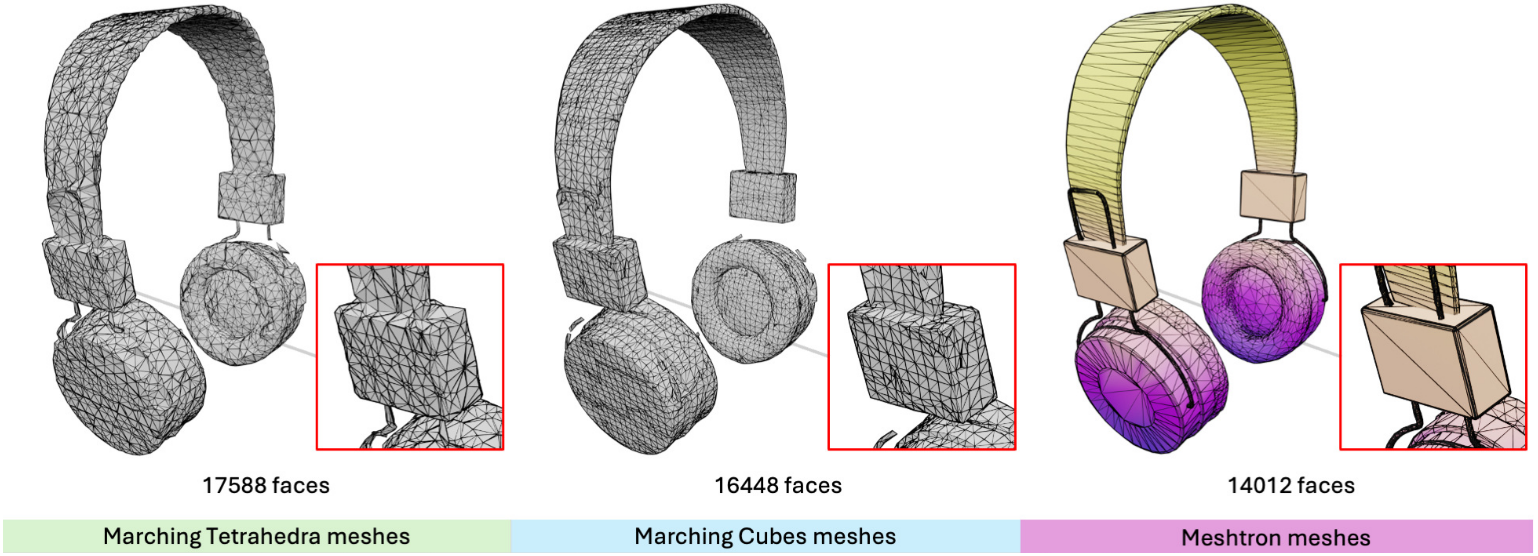
Lorensen, William E., and Harvey E. Cline. "Marching cubes: A high resolution 3D surface construction algorithm." *Seminal graphics: pioneering efforts that shaped the field*. 1998. 347-353.
Ju, Tao, et al. "Dual contouring of hermite data." *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. 2002.
Guillard, Benoit, Federico Stella, and Pascal Fua. "Meshudf: Fast and differentiable meshing of unsigned distance field networks." *European Conference on Computer Vision*. Cham: Springer Nature Switzerland, 2022.

Wang, Peng, et al. "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction." *arXiv preprint arXiv:2106.10689* (2021).
Long, Xiaoxiao, et al. "Neuraludf: Learning unsigned distance fields for multi-view reconstruction of surfaces with arbitrary topologies." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023
Liu, Yu-Tao, et al. "Neudf: Leaning neural unsigned distance fields with volume rendering." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023.
Wei, Xinyue, et al. "Neumanifold: Neural watertight manifold reconstruction with efficient and high-quality rendering support." *arXiv preprint arXiv:2305.17134* (2023).
Guédon, Antoine, and Vincent Lepetit. "Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024.

Shen, Tianchang, et al. "Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis." *Advances in Neural Information Processing Systems* 34 (2021): 6087-6101.
Shen, Tianchang, et al. "Flexible Isosurface Extraction for Gradient-Based Mesh Optimization." *ACM Trans. Graph.* 42.4 (2023): 37-1.
Liu, Zhen, et al. "Ghost on The Shell: An Expressive Representation of General 3D Shapes." *arXiv preprint arXiv:2310.15168* (2023).

Mesh in ML: Autoregressive models

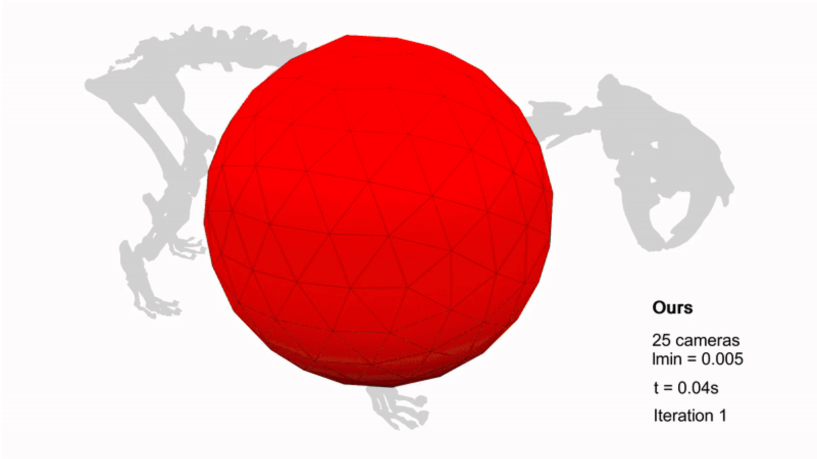
- Solution 2) Use **autoregressive models** to generate mesh by predicting mesh connectivity step by step
  - +) Great mesh quality, almost human-made mesh quality
  - -) Excessive computational cost, little topological guarantee, outlier issue (data-driven)



Siddiqui, Yawar, et al. "Meshgpt: Generating triangle meshes with decoder-only transformers." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024.
Chen, Yiwen, et al. "MeshAnything: Artist-Created Mesh Generation with Autoregressive Transformers." *arXiv preprint arXiv:2406.10163* (2024).
Shen, Tianchang, et al. "SpaceMesh: A Continuous Representation for Learning Manifold Surface Meshes." *SIGGRAPH Asia 2024 Conference Papers*. 2024.
Hao, Zekun, et al. "Meshtron: High-Fidelity, Artist-Like 3D Mesh Generation at Scale." *arXiv preprint arXiv:2412.09548* (2024).

Mesh in ML: (Limited) Differentiable Mesh

- Solution) Fix the mesh connectivity, and only optimize vertex positions
  - +) Low computational cost
  - -) Limited topology



Differentiable Geometry

→ Topologically, it is still a ball!

Differentiability & Challenges

	Vertex positions	Edge Connectivity
Previous Work	O	X
Ours	O	O

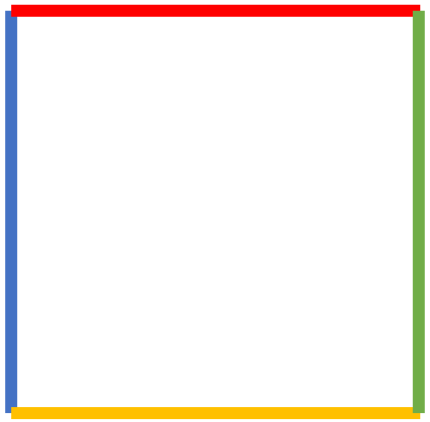
Differentiable connectivity has not been discussed so far

→ **Challenge: Non-differentiability in discrete data structure for connectivity**

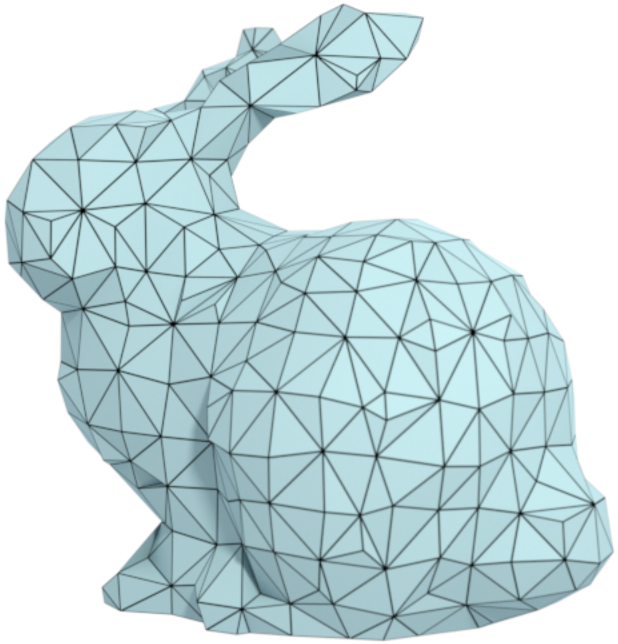
Definitions

Mesh = Set of faces

	Face	Simplex
D=2	Line segment	Triangle
D=3	Triangle	Tetrahedron



2D mesh = Set of line segments



3D mesh = Set of triangles

Toward differentiable mesh: DMesh

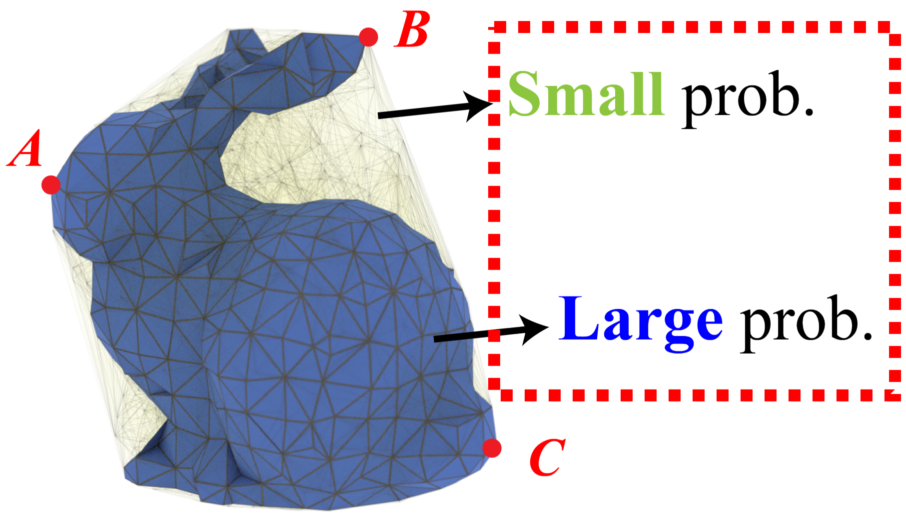
- **DMesh**: Our solution for **truly differentiable mesh**

1) Discrete connectivity → Use **probabilistic approach** for mesh connectivity

$$\text{Faces}\ \left\{\begin{matrix} F_{1,1}, F_{1,2}, F_{1,3}, \boxed{0} \\ F_{2,1}, F_{2,2}, F_{2,3}, \boxed{1} \\ \vdots \\ F_{M,1}, F_{M,2}, F_{M,3}, \boxed{0} \end{matrix}\right\} \qquad \left\{\begin{matrix} F_{1,1}, F_{1,2}, F_{1,3}, \boxed{P(F_1)} \\ F_{2,1}, F_{2,2}, F_{2,3}, \boxed{P(F_2)} \\ \vdots \\ F_{M,1}, F_{M,2}, F_{M,3}, \boxed{P(F_M)} \end{matrix}\right\}$$

Discrete (Original)

Probabilistic (Ours)

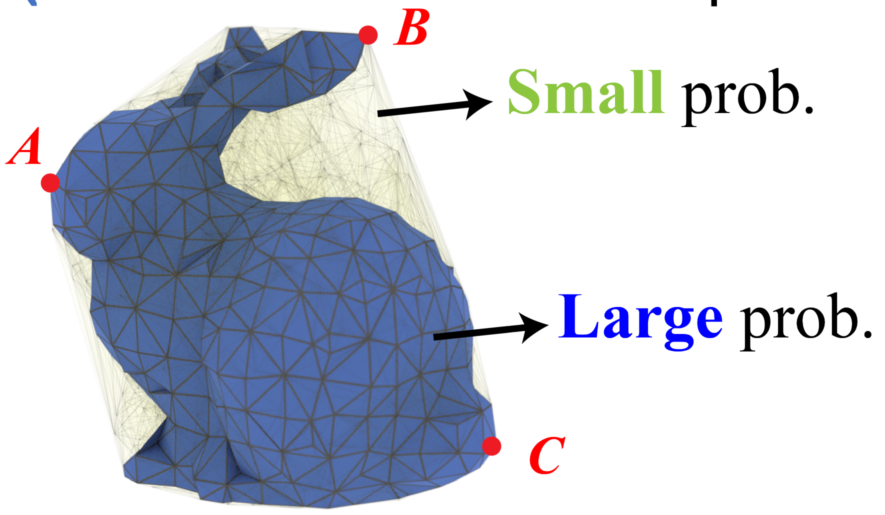


DO NOT have to consider *ABC*, because WDT only connects locally adjacent vertices

Toward differentiable mesh: DMesh

- **DMesh**: Our solution for **truly differentiable mesh**

  2) Exponential connectivity → Based on **Weighted Delaunay Triangulation (WDT)**, exclude most of possible cases



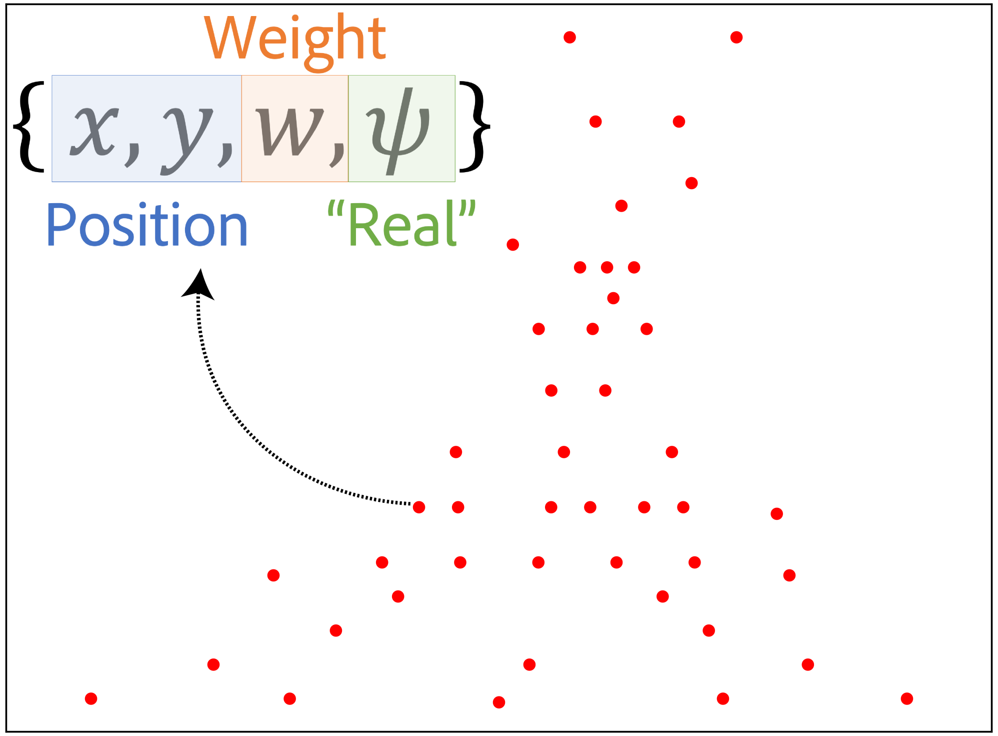
DO NOT have to consider *ABC*, because
WDT only connects locally adjacent vertices

$$O\left(N^3\right) \rightarrow O(N),$$

N = Number of points

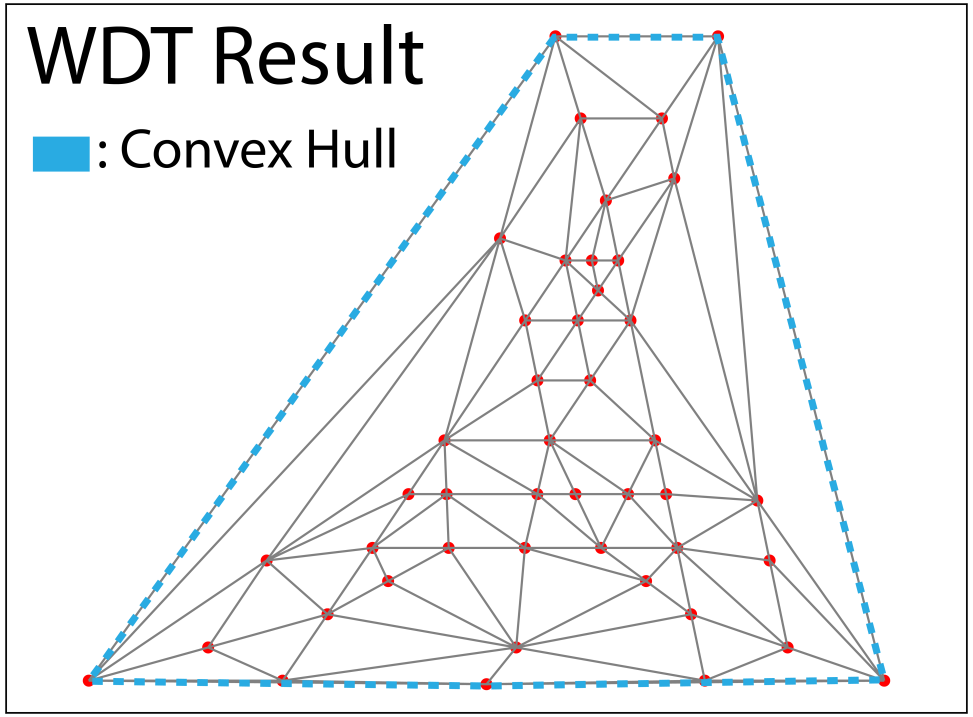
Approach

- DMesh is defined with a **set of featured points**
  - Position
  - Weight: Used for Weighted Delaunay Triangulatio
    - Represents the importance of the point
    - If weight of a point is smaller than that of surrounding points, the point is discarded
  - "Real" value: Used for selecting faces from WDT



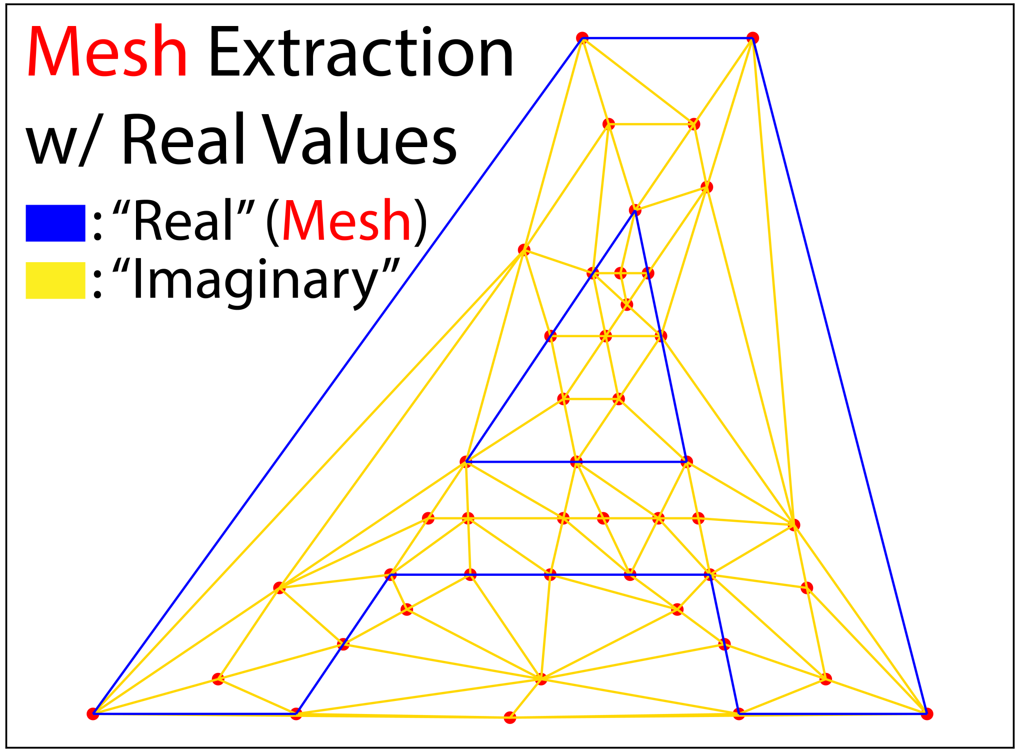
Approach

- Compute **WDT** of the points, which depends on point positions and weights
  - WDT tessellates the **convex hull of the points** **without self-intersections**



Approach

- **Select desirable faces** from WDT using point-wise “real” values
  - **Only when every point on a face has “real” value of 1, the face is selected**



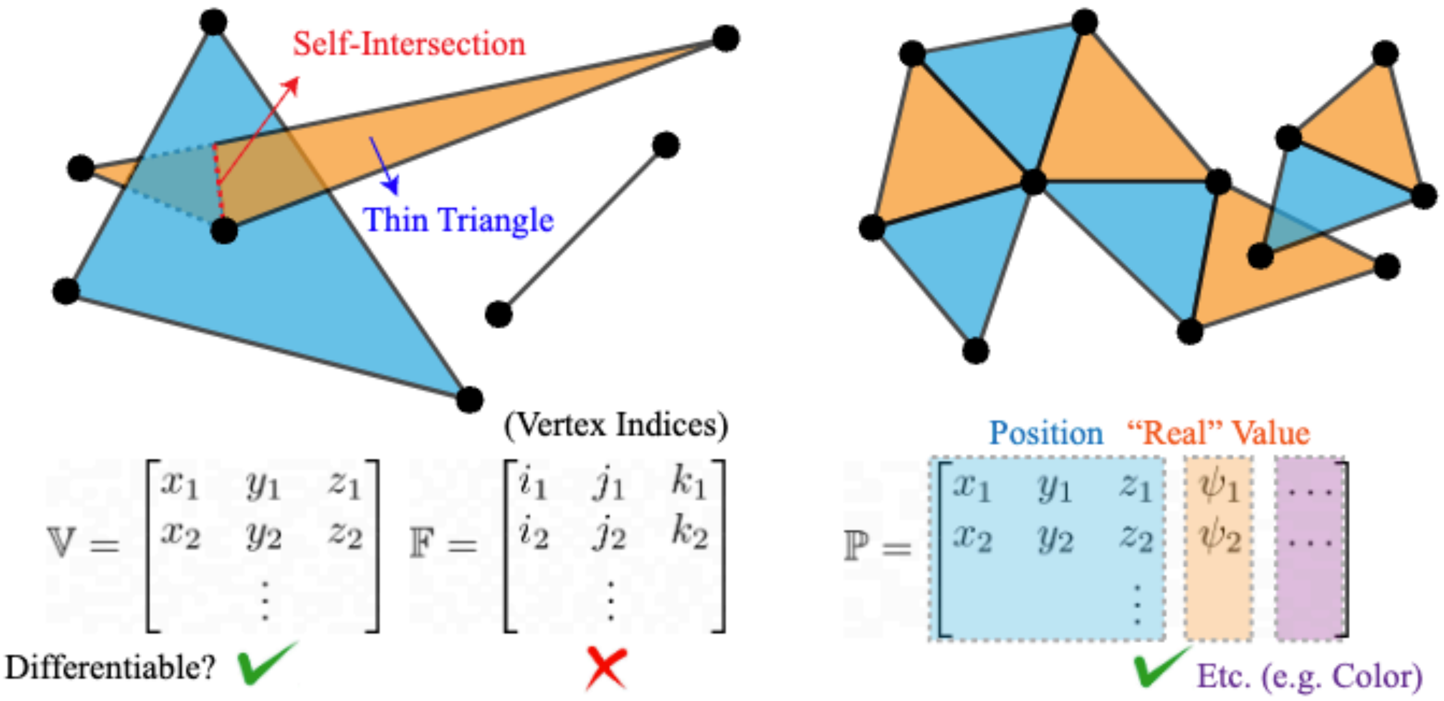
Approach: Sum Up

- **Two conditions** for a face F to be included in the mesh:
  - F should be included in WDT
  - Every points on F should have “real” value of 1

- For each condition, compute probability to satisfy it
  - $\Lambda_{wdt}(F)$
  - $\Lambda_{real}(F)$

- Then, the final existence probability for F is $\Lambda_{wdt}(F) \times \Lambda_{real}(F)$

Approach: Sum Up

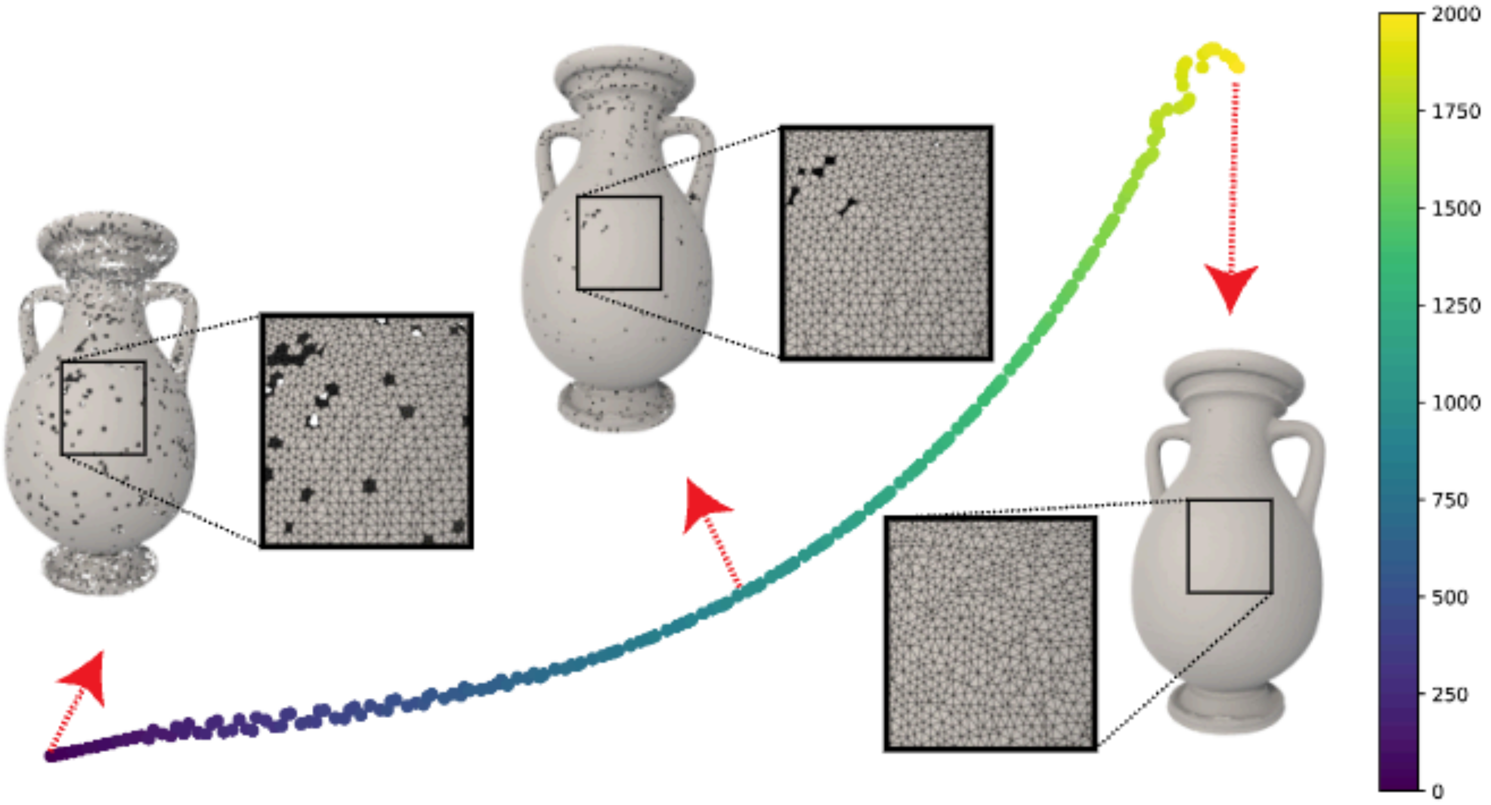


(a) (Traditional) Mesh

(b) DMesh, DMesh++

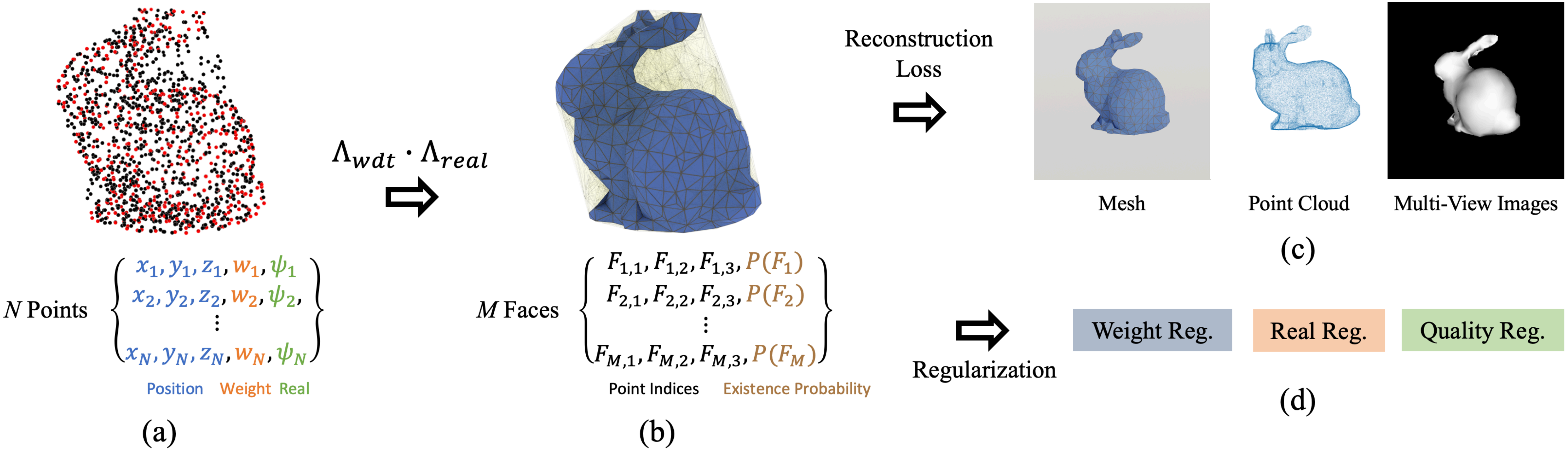
DMesh is free from **1) self-intersections** and **2) ill-formed triangles**, because of (weighted) Delaunay Triangulation

Approach: Sum Up



While DMesh features change **continuously**,
discrete topological changes take place

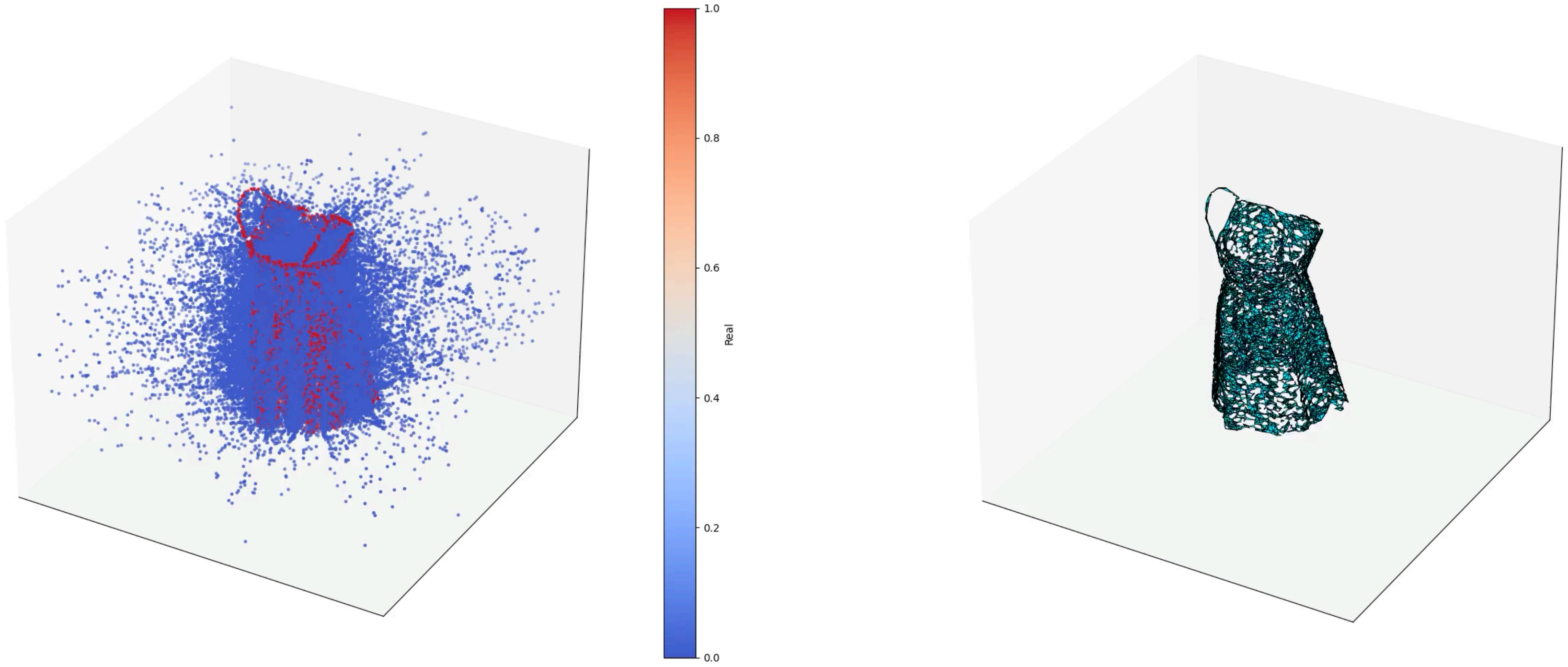
Pipeline



In this work, we proposed **differentiable reconstruction loss formulations** for probabilistic mesh.
Backpropagate the loss, and **update per-point features to optimize mesh**.

Experimental Results (1)

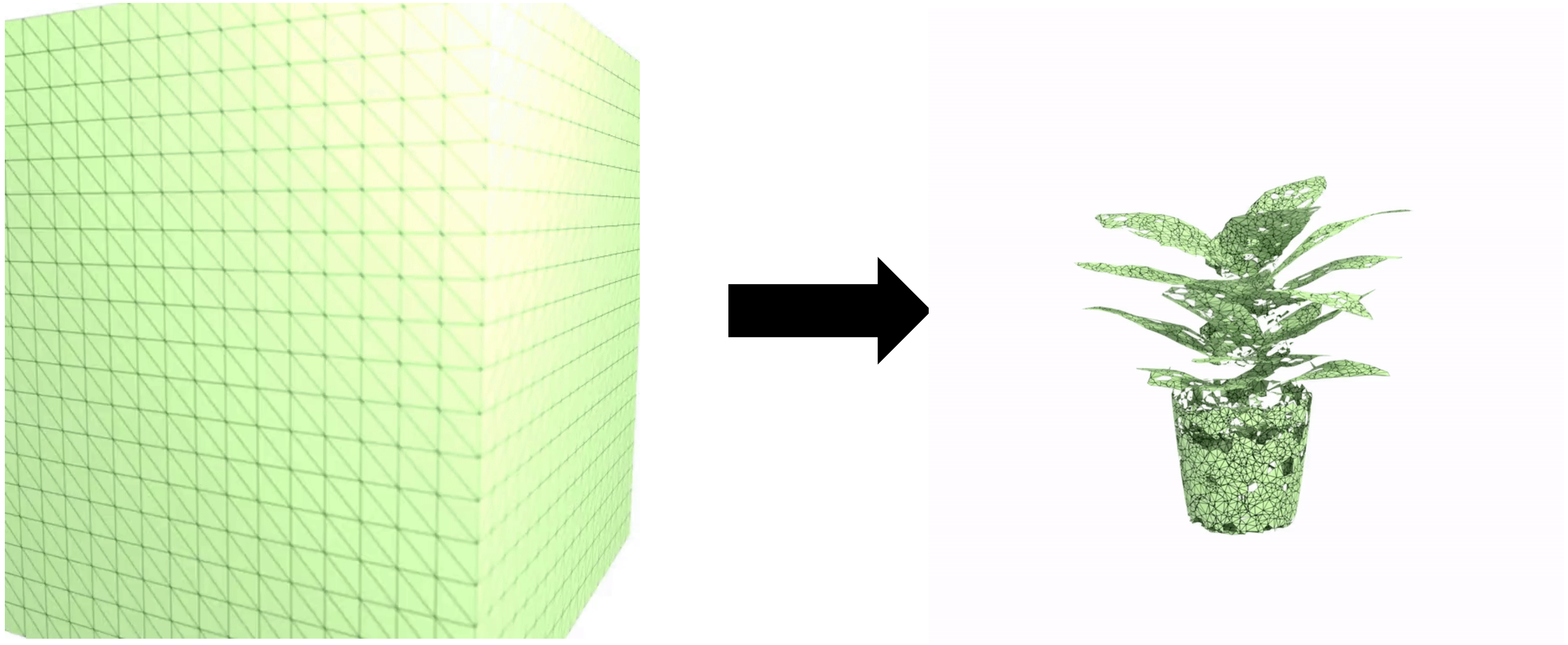
- 3D mesh reconstruction from **point clouds**
  - **Input**: 100K points uniformly sampled from ground truth mesh
  - **Reconstruction loss**: Chamfer Distance (CD) loss



Experimental Results (1)

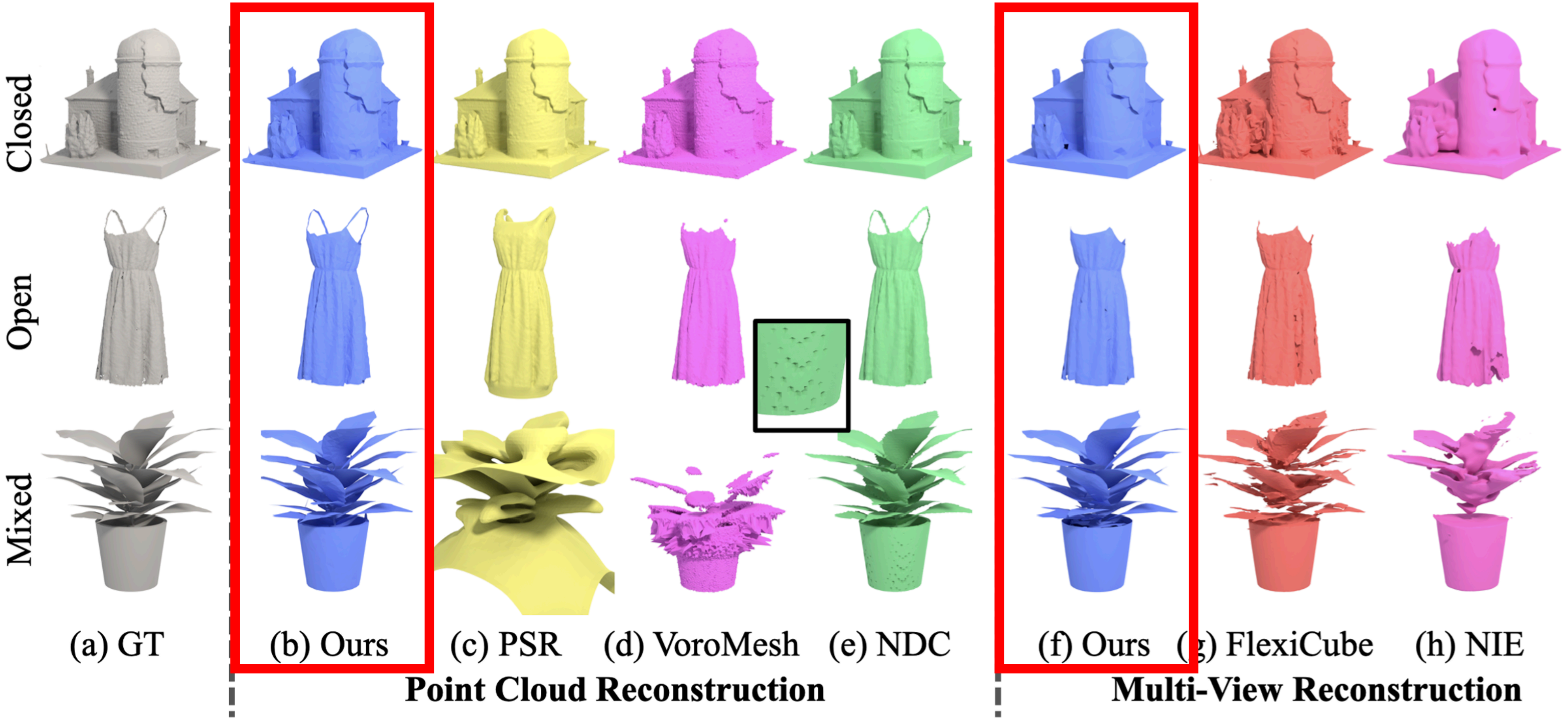
Experimental Results (2)

- 3D mesh reconstruction from **multi-view images**
  - **Input**: Diffuse and depth images captured from 64 viewpoints
  - **Reconstruction loss**: L1 loss on rendered images



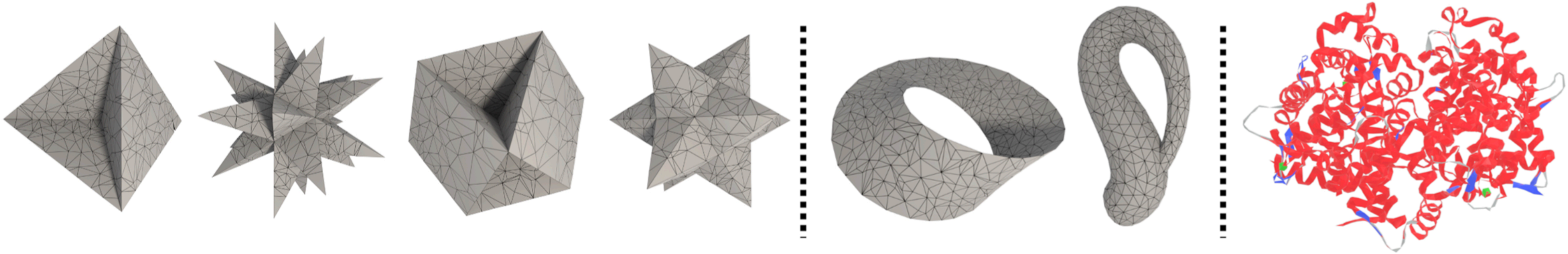
Epoch 1 Epoch 4

Experimental Results



DMesh yields more accurate and efficient mesh than the other methods

Experimental Results

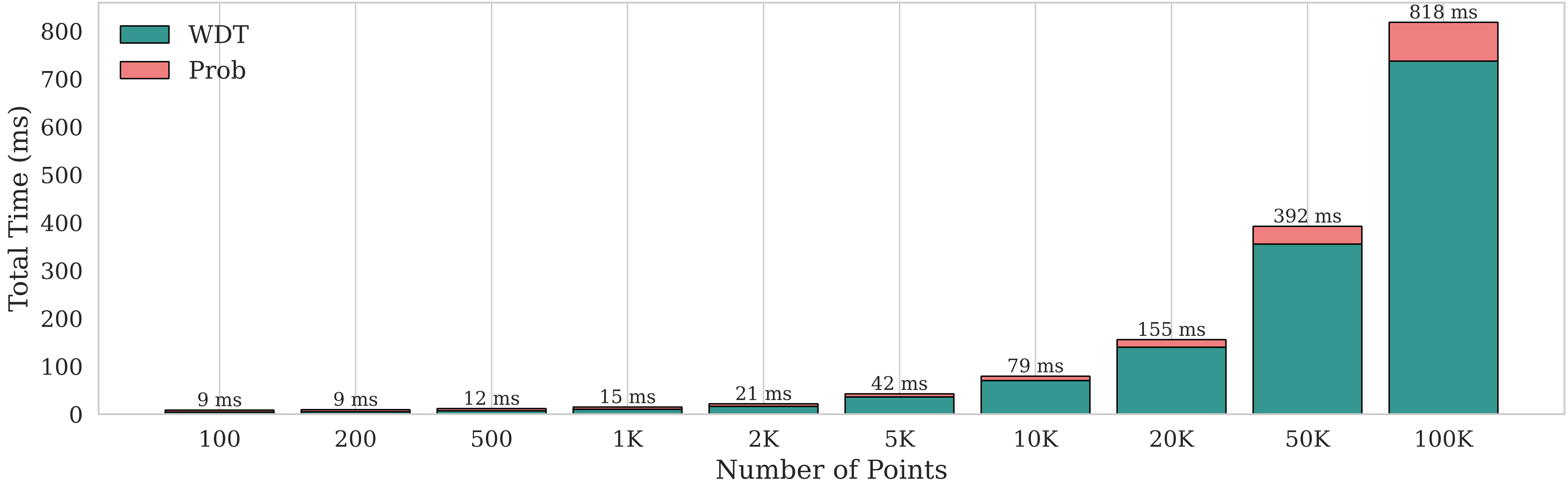


(a) Non-convex polyhedra (b) Non-orientable geometry (c) Protein

While every other representation has topological limitation (e.g. closed surface), DMesh can represent a **shape of any topology**

Limitations of DMesh

- Computational cost
  - Linear computational cost of $O(N)$, where N is the number of points
  - **Cannot handle complex shapes** that require more than 100K points (*800ms* per



DMesh++: An Efficient Differentiable Mesh for Complex Shapes

Sanghyun Son, Matheus Gadelha, Yang Zhou, Matthew Fisher, Zexiang Xu, Yi-Ling Qiao, Ming C. Lin, Yi Zhou

ICCV 2025

DEPARTMENT OF
COMPUTER SCIENCE

Minimum Ball Condition

- **Two conditions** for a face F to be included in the mesh

DMesh		DMesh++	
is in WDT		satisfies the Minimum-Ball condition	
Vertices on have real value of 1		Vertices on have real value of 1	

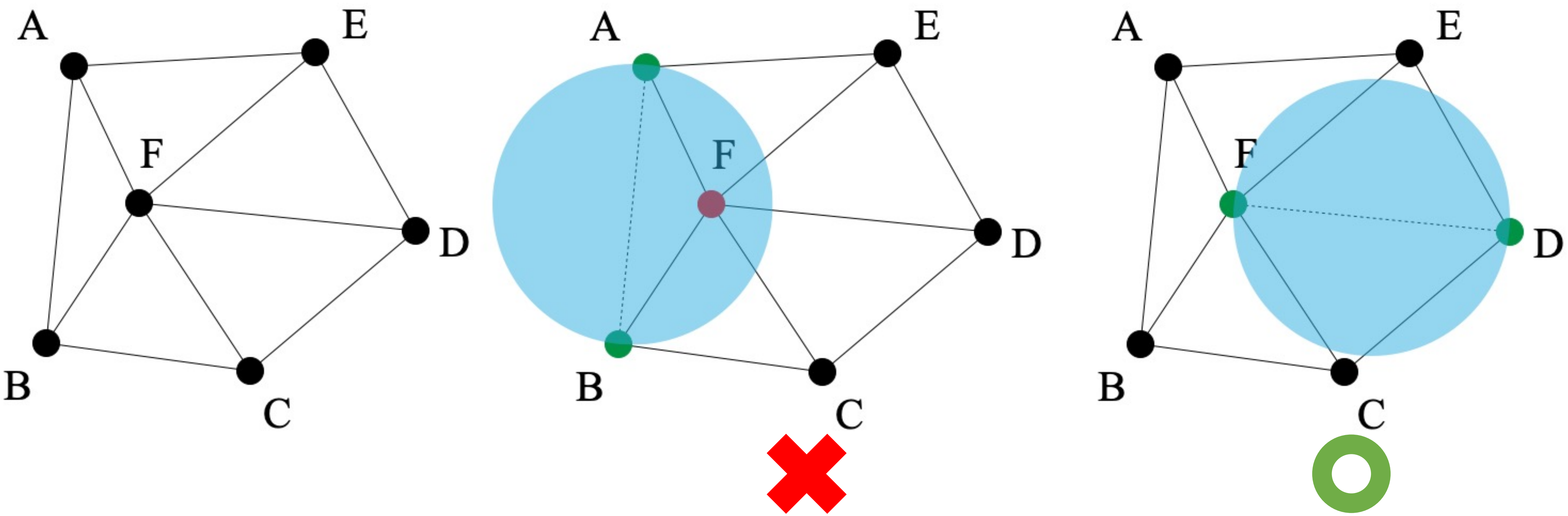
By changing definition of F, we can reduce computational cost:

$$O(N) \rightarrow O(\log N)$$

Minimum Ball Condition

A **face** has an infinite number of **bounding balls**
→ The Minimum Ball is the smallest of those bounding balls

Minimum Ball Condition



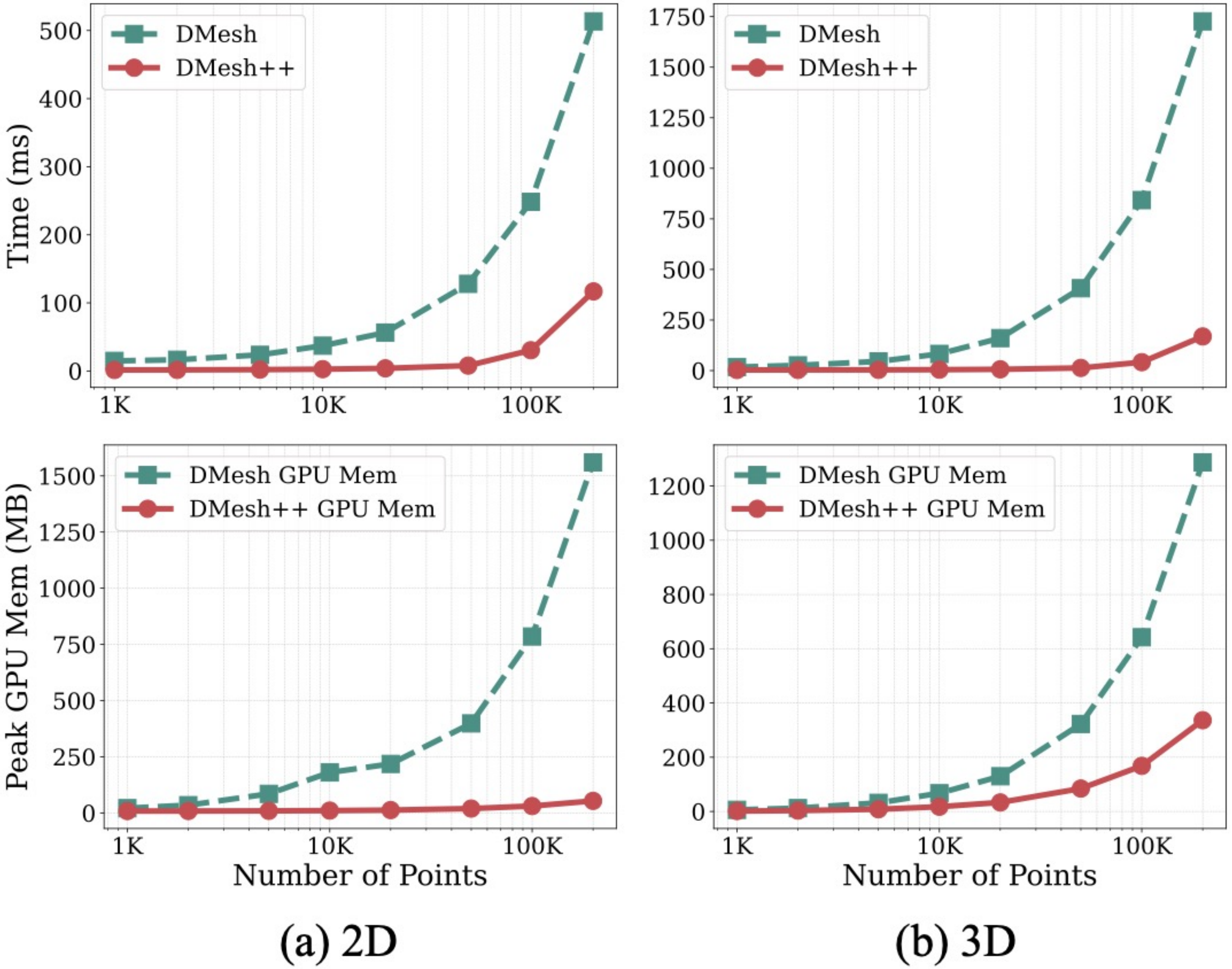
If the Minimum Ball of a face does not have any other points inside it, the face satisfies the Minimum Ball condition

Minimum Ball Condition

Lemma. If a face F satisfies the Minimum Ball condition, it exists in Delaunay Triangulation (DT)

- Therefore, DMesh++ inherits nice properties of DMesh
  - Free from self-intersections
  - Minimizes the number of ill-formed triangles

Computational Cost

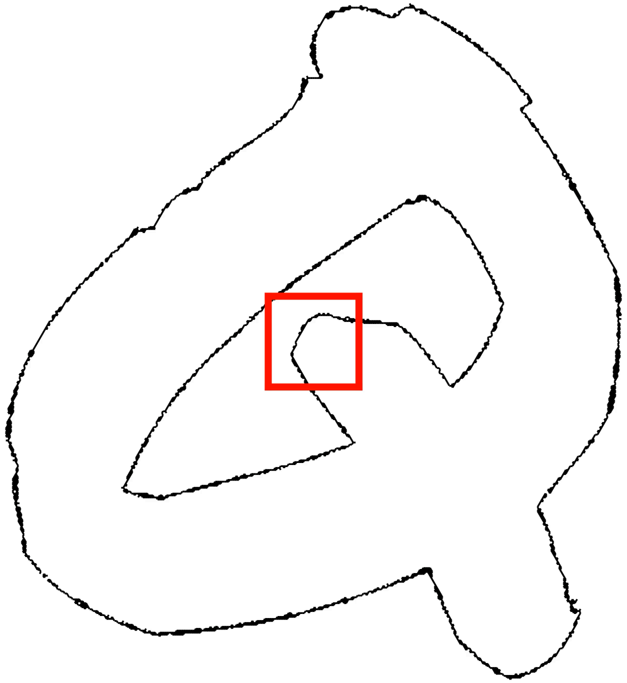


Experimental Results

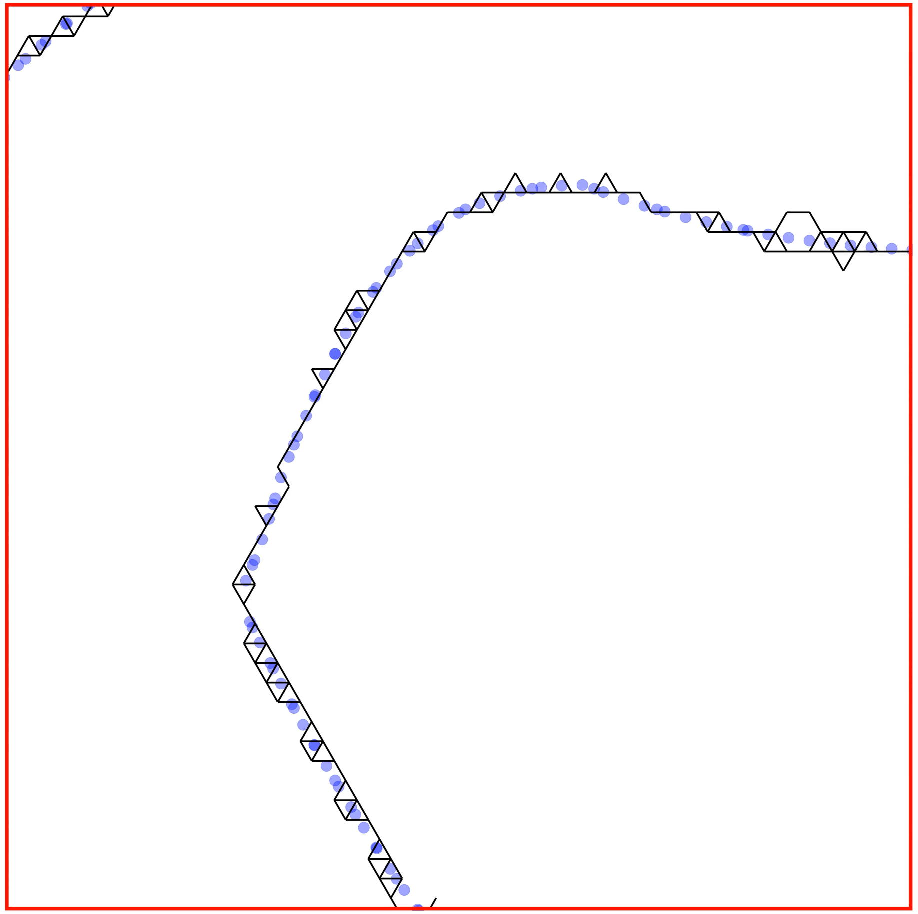
- 2D & 3D mesh reconstruction from point clouds
  - Input: 200K points sampled from ground truth geometry
  - Loss: Chamfer Distance loss

- 3D mesh reconstruction from multi-view images
  - Input: 64 diffuse and depth images captured from ground truth geometry
  - Loss: L1 loss on rendered images

2D Point Cloud Reconstruction



Step 0



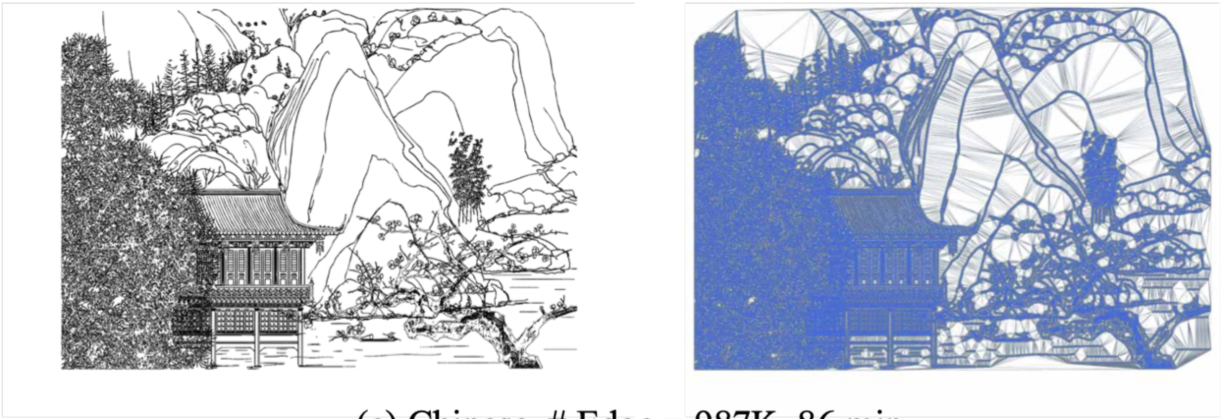
2D Point Cloud Recon.

(a) Flower, # Edge = 99K, 6 min.

(b) Eagle, # Edge = 179K, 11 min.

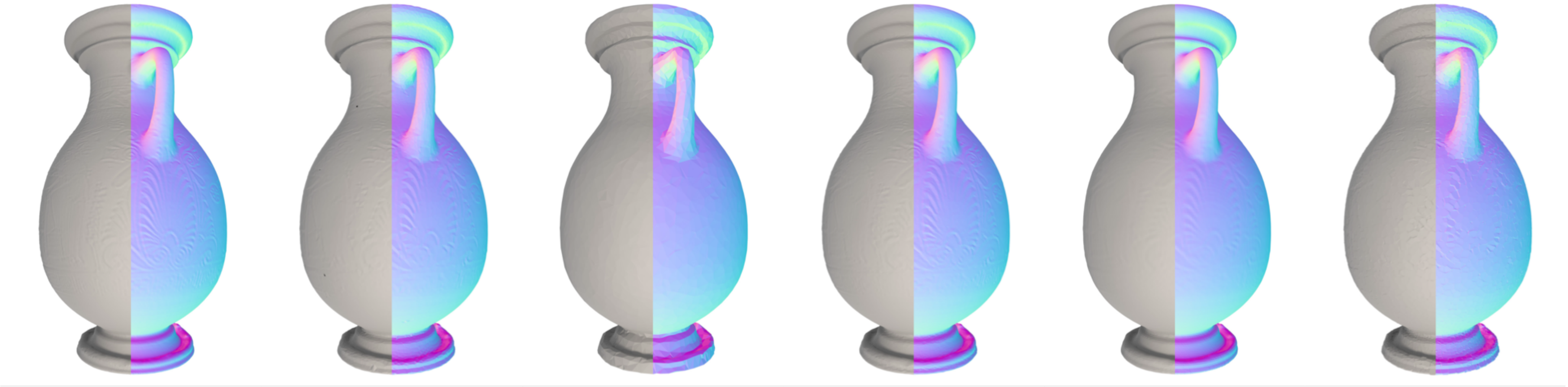
(c) Picasso, # Edge = 159K, 8 min.

(d) Egyptian, # Edge = 227K, 19 min.



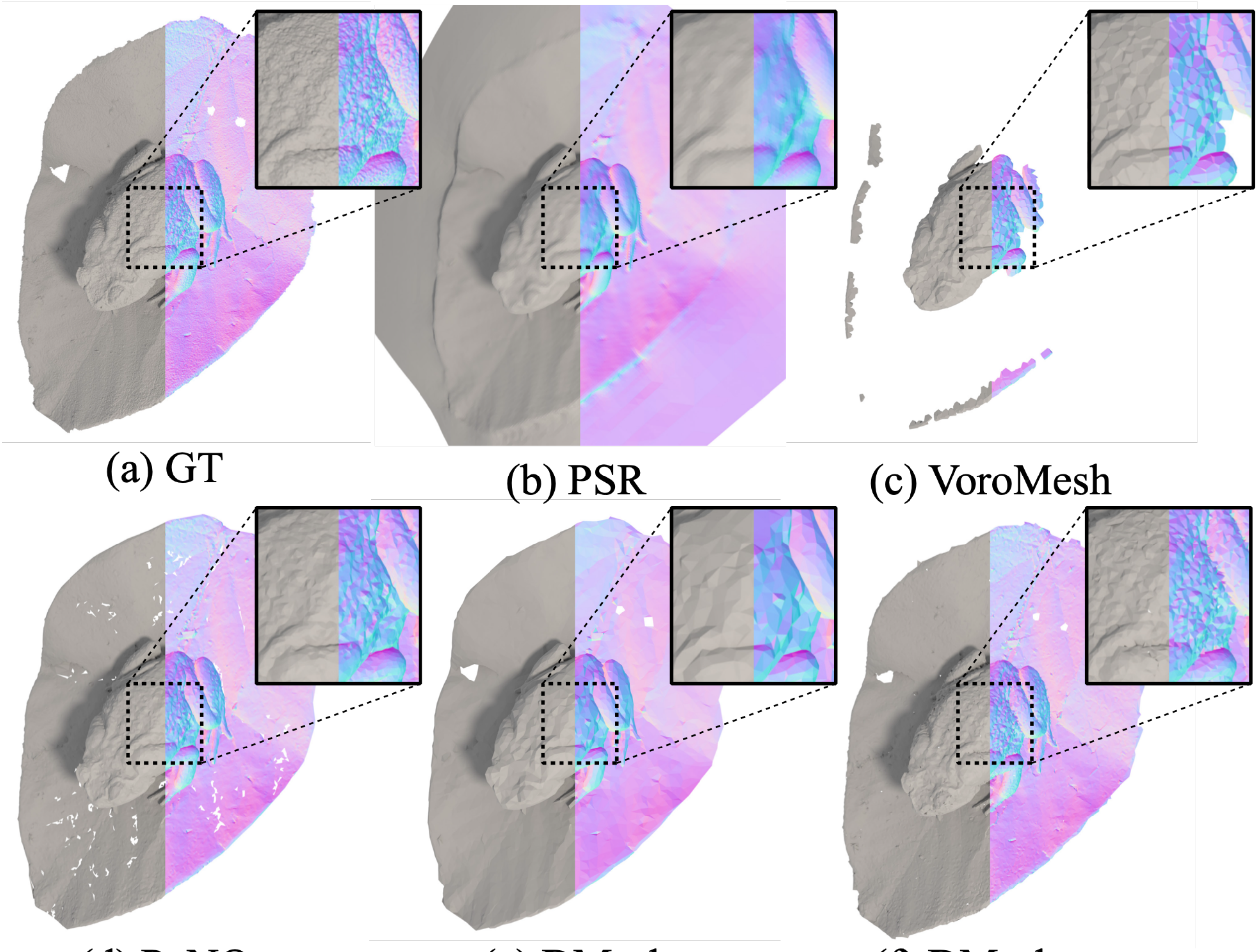
(e) Chinese, # Edge = 987K, 86 min.

3D Point Cloud Recon. (Closed Surface)



(a) Ground Truth (b) DMesh++ (c) DMesh (d) PoNQ (e) PSR (f) VoroMesh

3D Point Cloud Recon. (Open Surface)



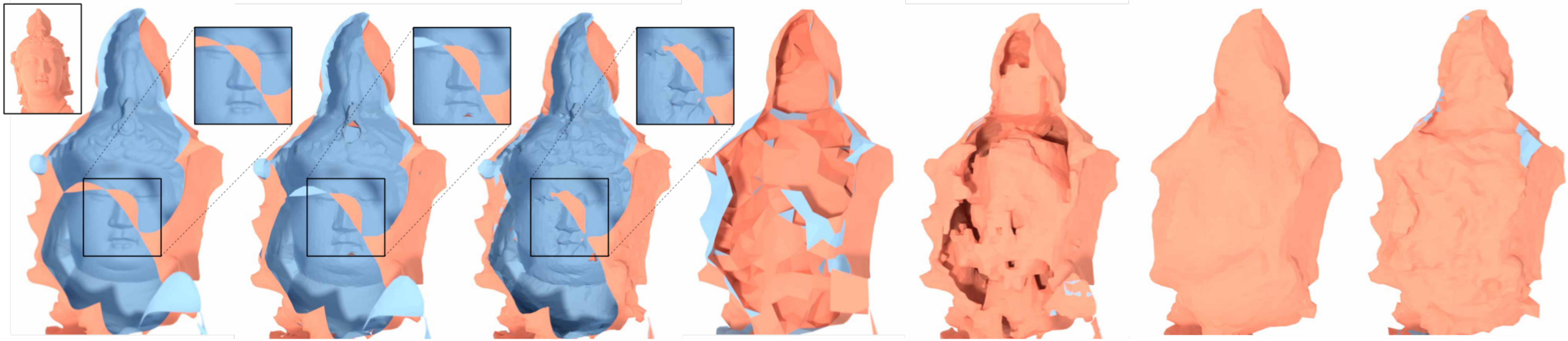
(a) GT (b) PSR (c) VoroMesh

(d) PoNQ (e) DMesh (f) DMesh++

3D Multi-View Recon. (Closed Surface)

(a) Ground Truth (b) DMesh++ (c) GShell (d) DMesh (e) FlexiCubes (f) DMTet (g) Remeshing

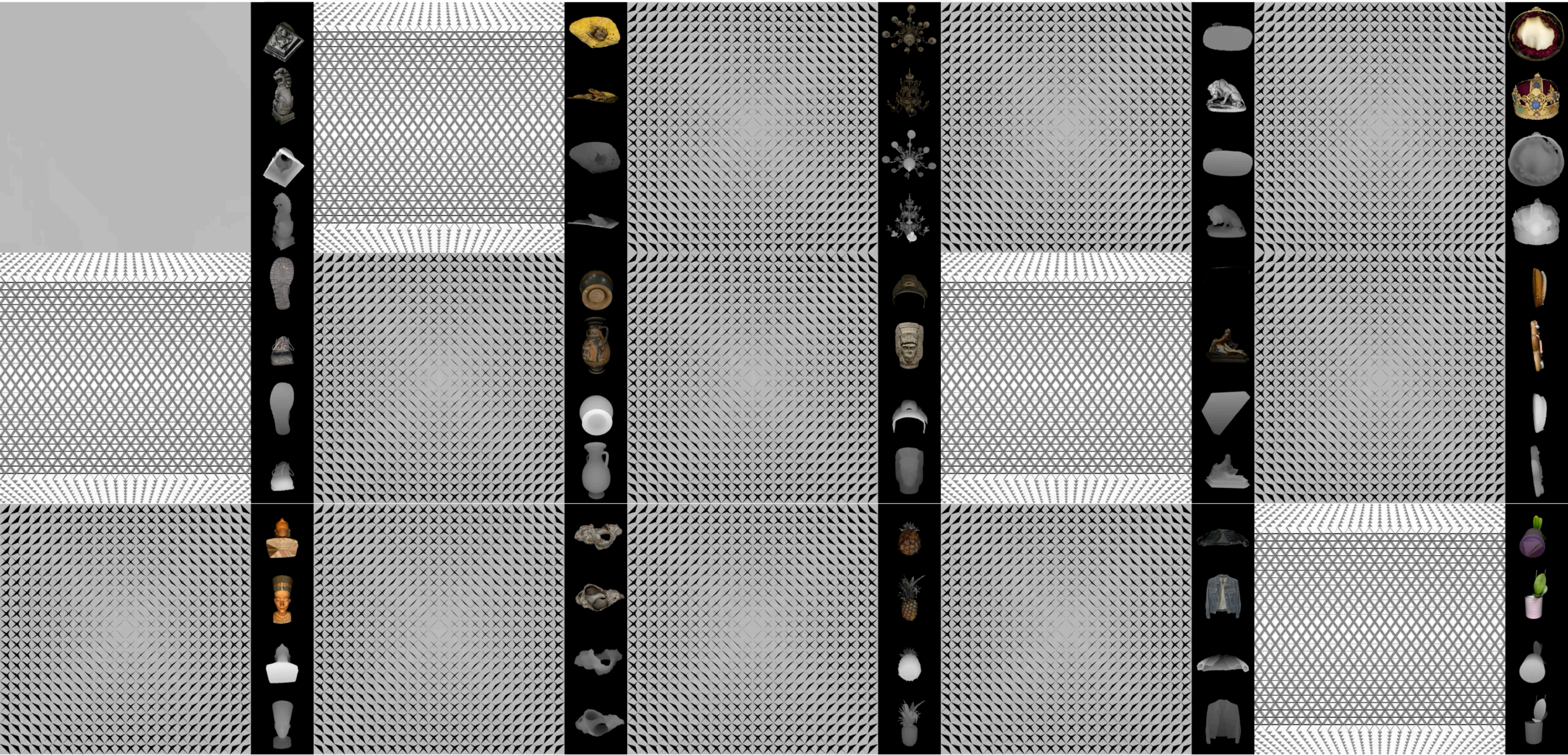
3D Multi-View Recon. (Open Surface)



(a) Ground Truth (b) DMesh++ (c) GShell (d) DMesh (e) FlexiCubes (f) DMTet (g) Remeshing

3D Reconstruction from Multi-View Images (Color, Depth)

Progress: 0%



Conclusion

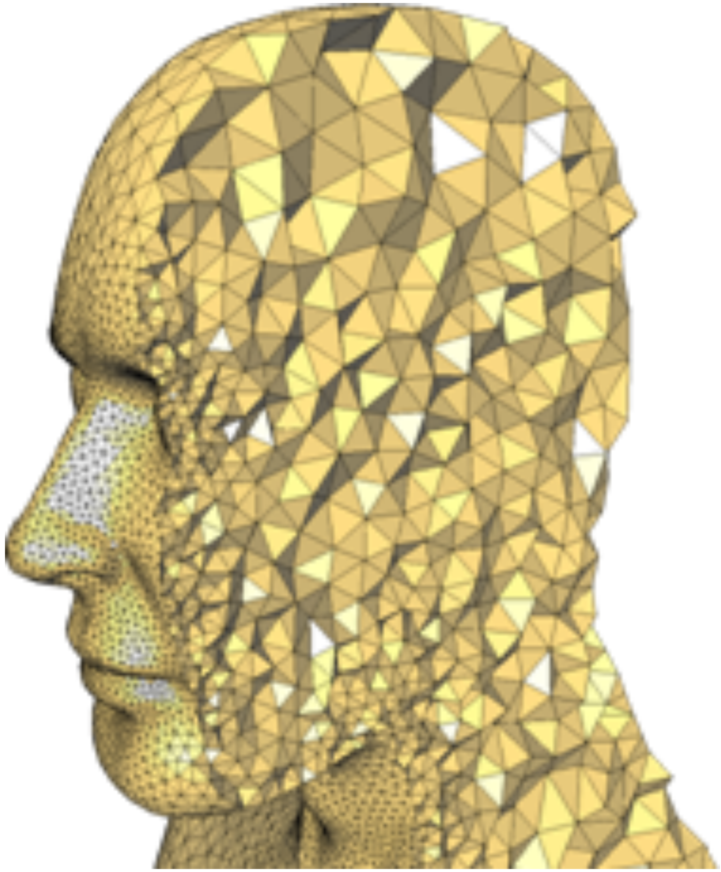
- DMesh and DMesh++ is a **differentiable, probabilistic** approach for mesh
- Compared to the other baseline methods, it has **advantages** in
  - **Computational cost** (vs. Neural implicit methods)
  - **Representation power** (vs. Methods based on iso-surface extraction)
  - **Ready for downstream application** (vs. 3DGS)
  - **Lower-level method that is not data-driven** (vs. Autoregressive models)

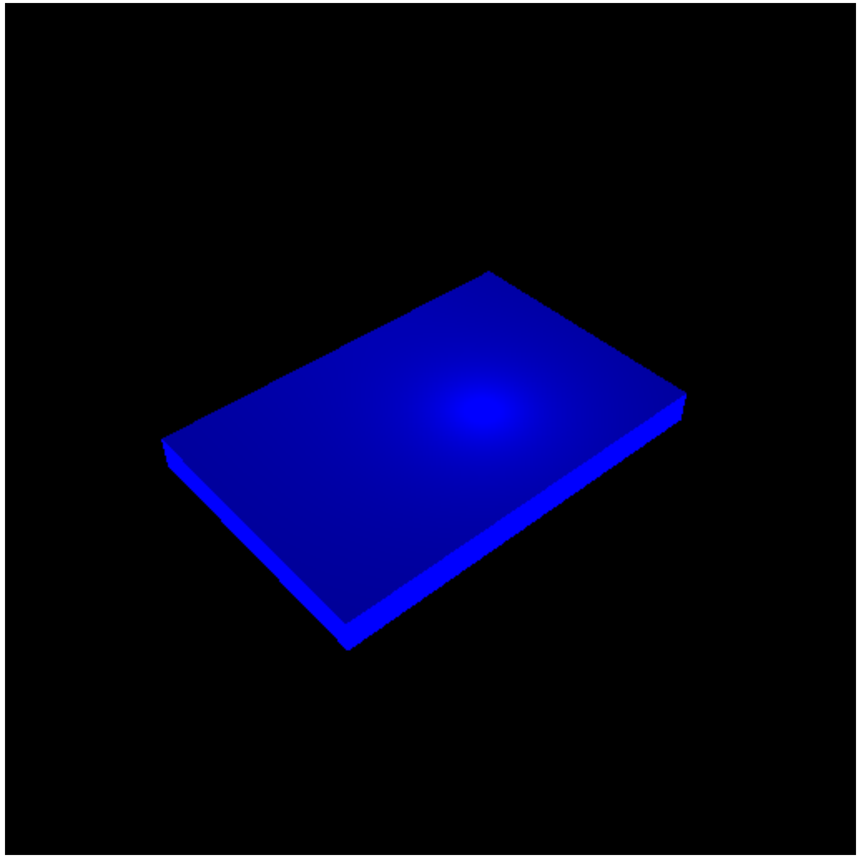
WIP: 3D Scene Reconstruction

- DMesh and DMesh++ could not do 3D recon. from real-world images

- Triangle Splatting+ is using a similar formulation of DMesh to extract opaque triangles and define mesh

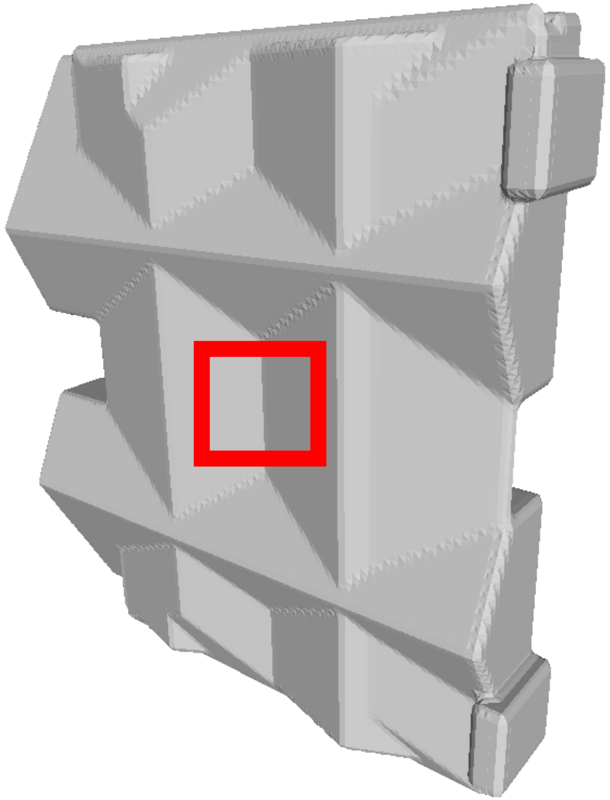
WIP: Differentiable Tetrahedral Mesh

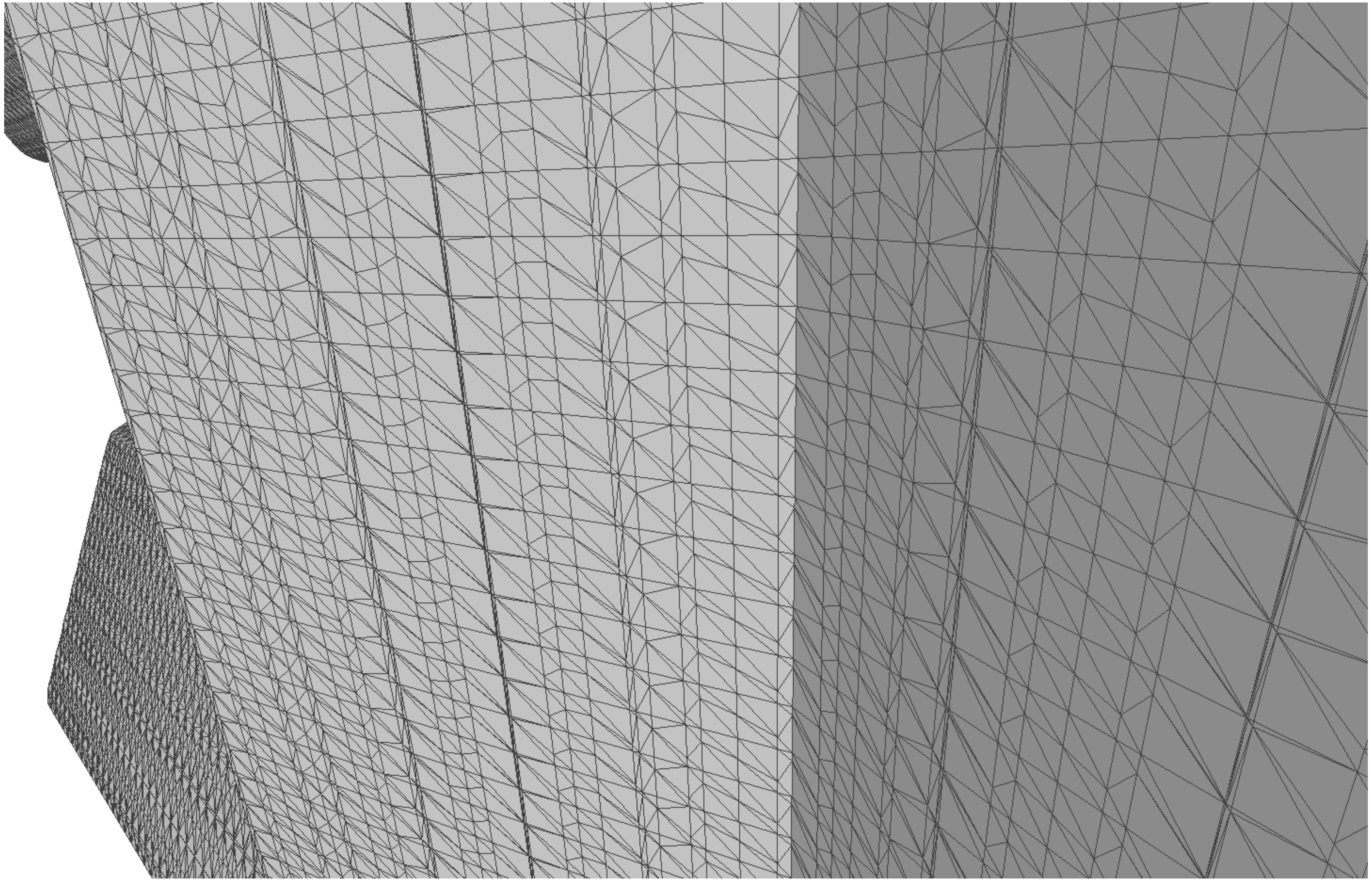




For physics simulation, we need tetrahedral mesh of the volumetric shape
→ Quality of the tet. mesh is critical for the simulation quality

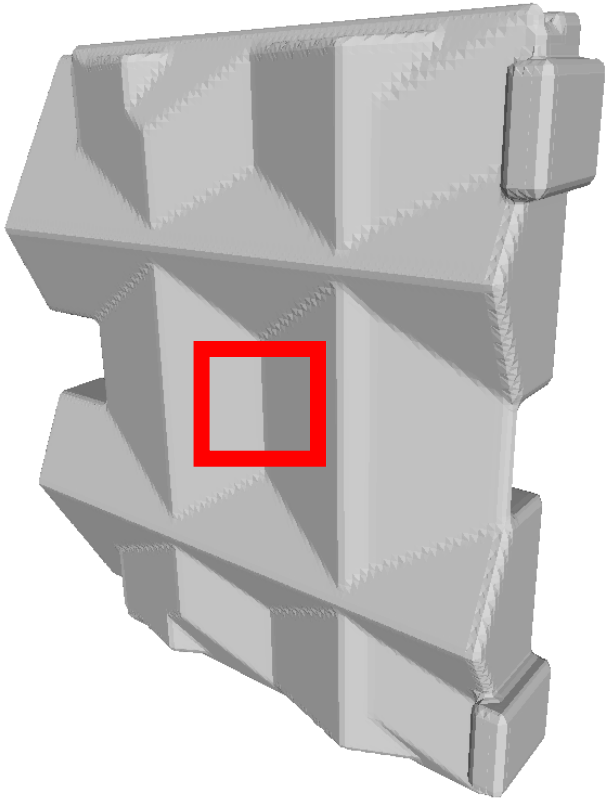
WIP: Differentiable Tetrahedral Mesh

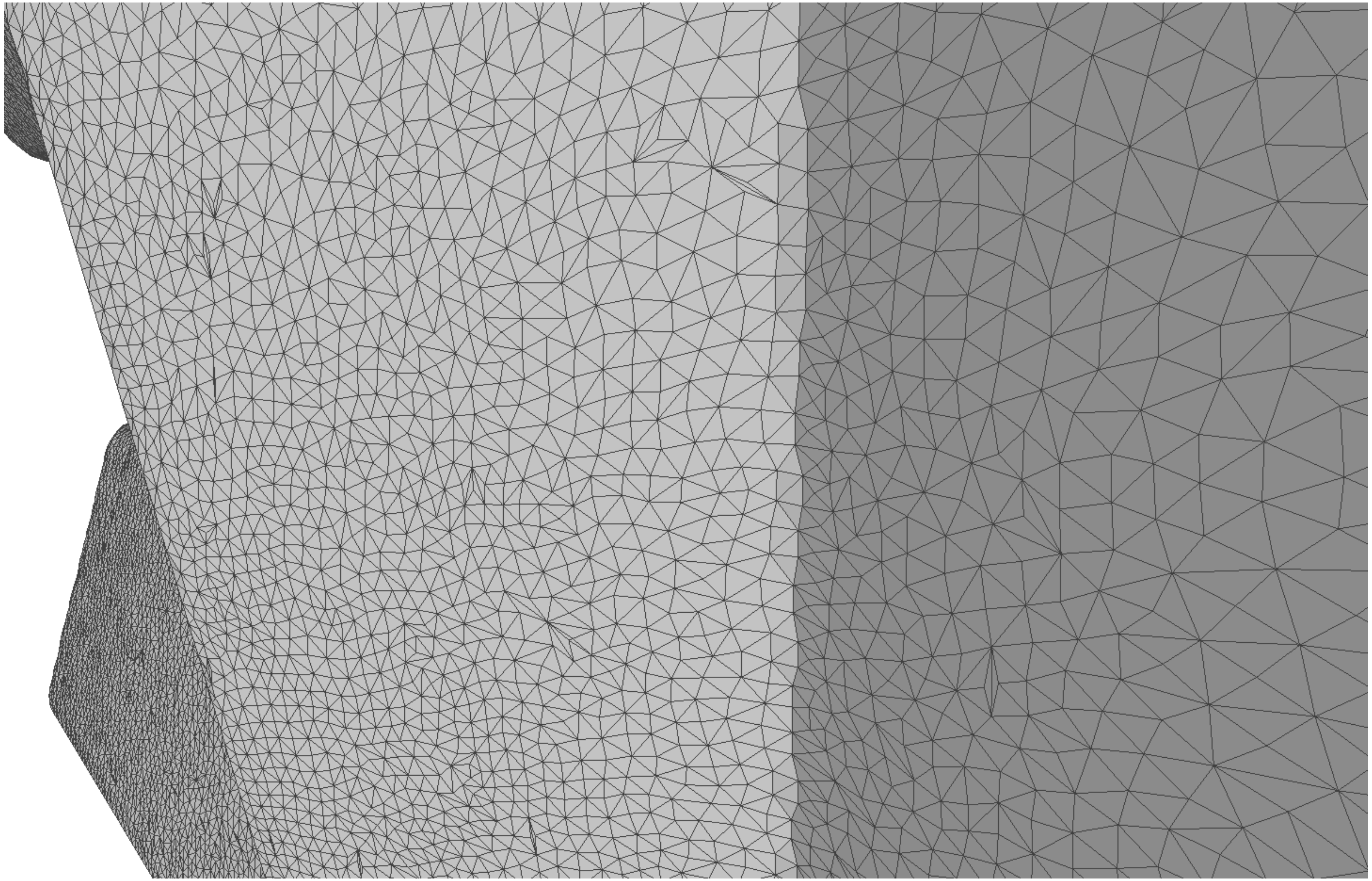




There are skinny triangles (tetrahedra) in this mesh, which is undesirable

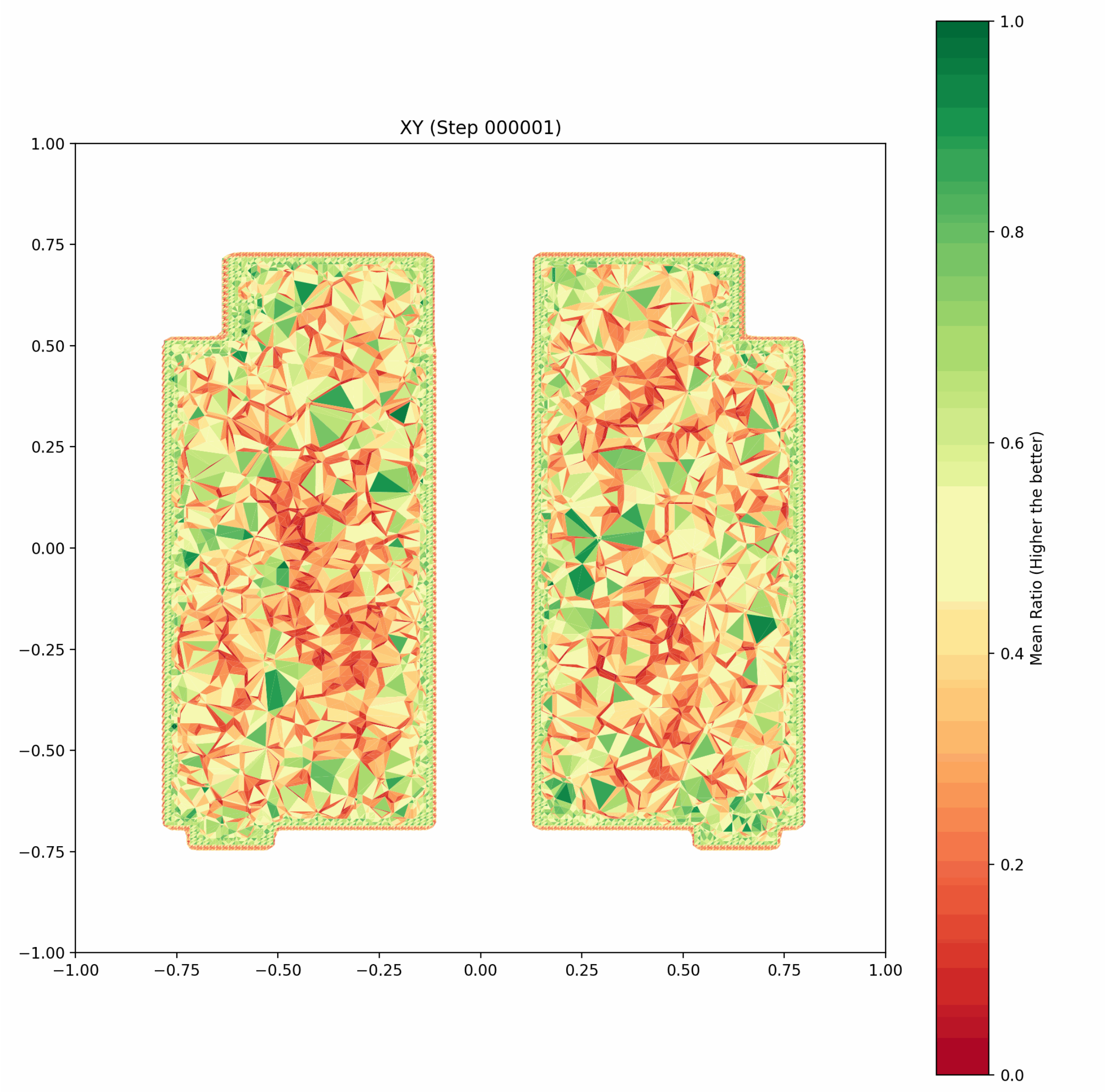
WIP: Differentiable Tetrahedral Mesh





Triangle (Tetrahedra) quality becomes much better after optimization

WIP: Differentiable Tetrahedral Mesh



THANK YOU!