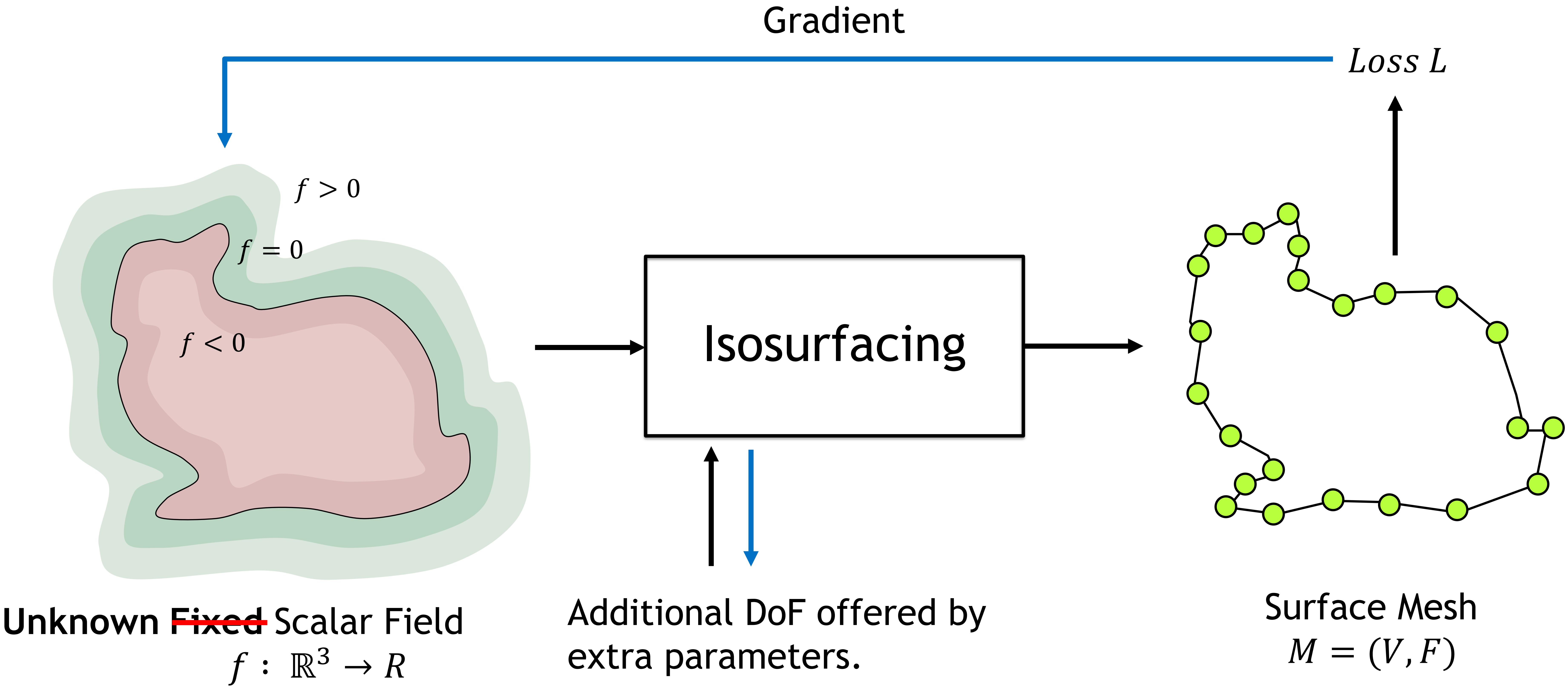
NVIDIA

# Flexible Isosurface Extraction for Gradient-Based Mesh Optimization

Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp*, Jun Gao*

# ISOSURFACING



# GENERATING MESH VIA GRADIENT-BASED **OPTIMIZATION**

Why not directly optimize the implicit field?

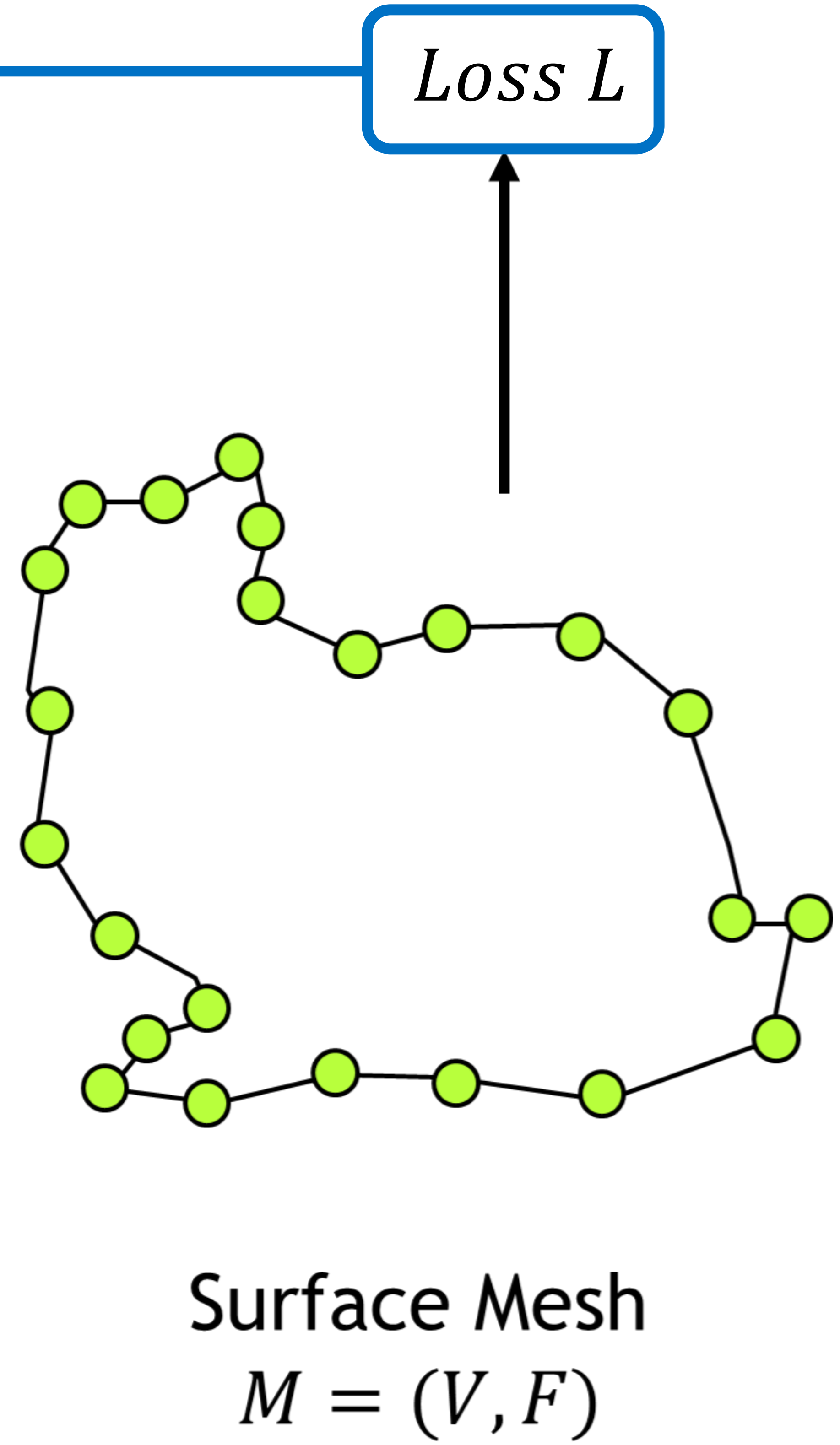
Visual

Geometric

Physical

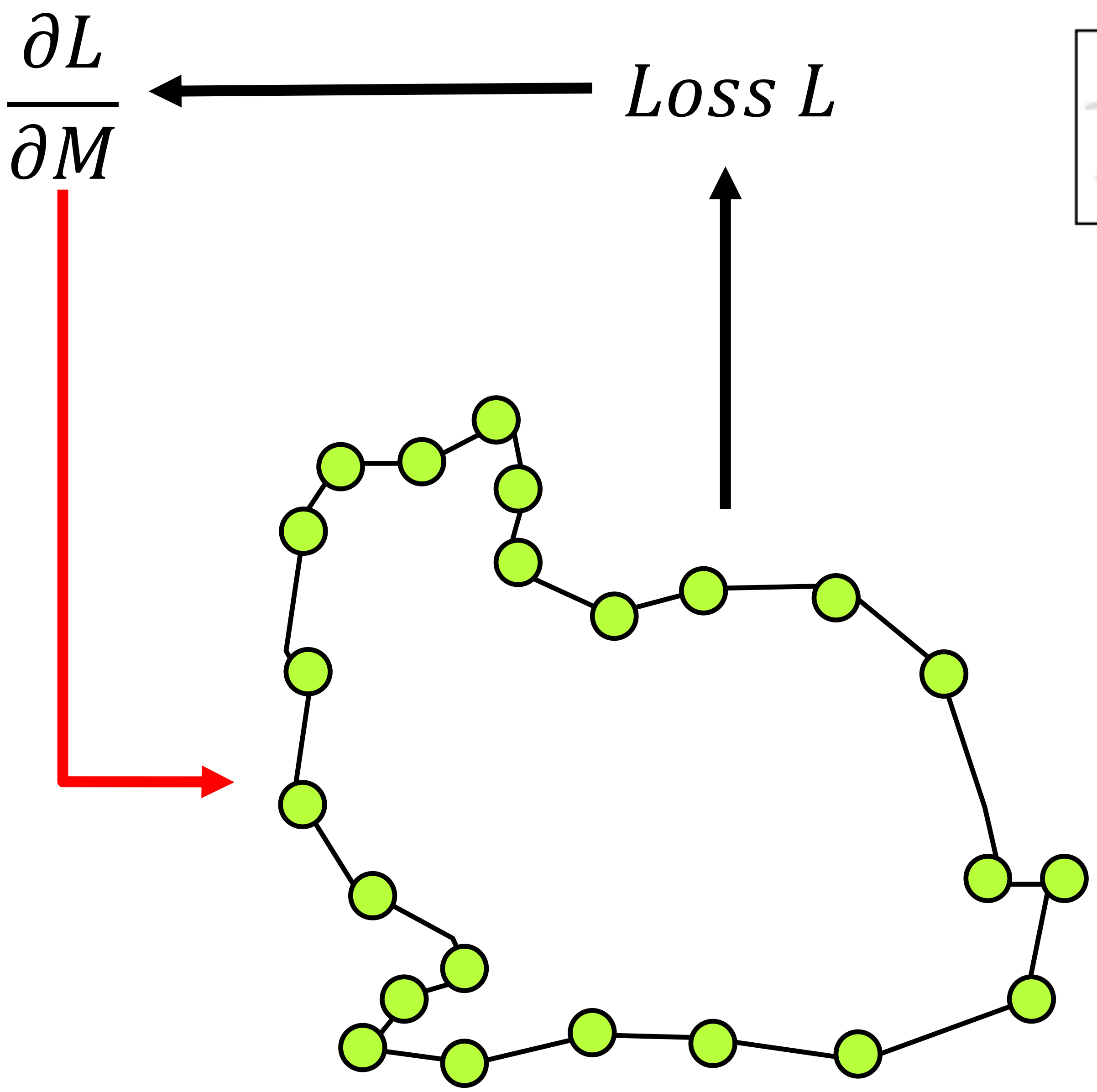
Evaluating objectives on meshes:

- ✓ Efficient, benefit from mesh representation.
- ✓ Ready to use in downstream applications.



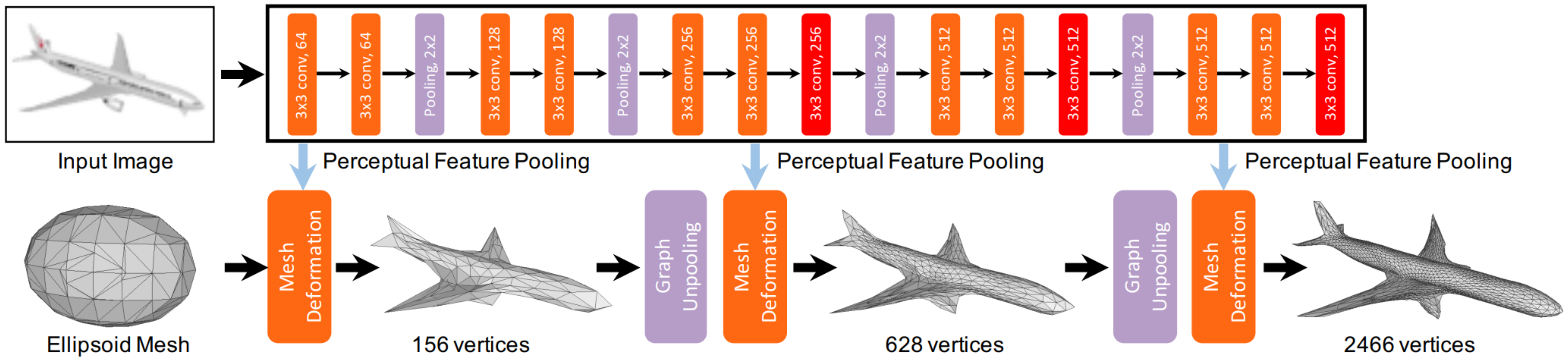
# GENERATING MESH VIA GRADIENT-BASED **OPTIMIZATION**

Why not directly optimizing the vertex positions of a mesh?

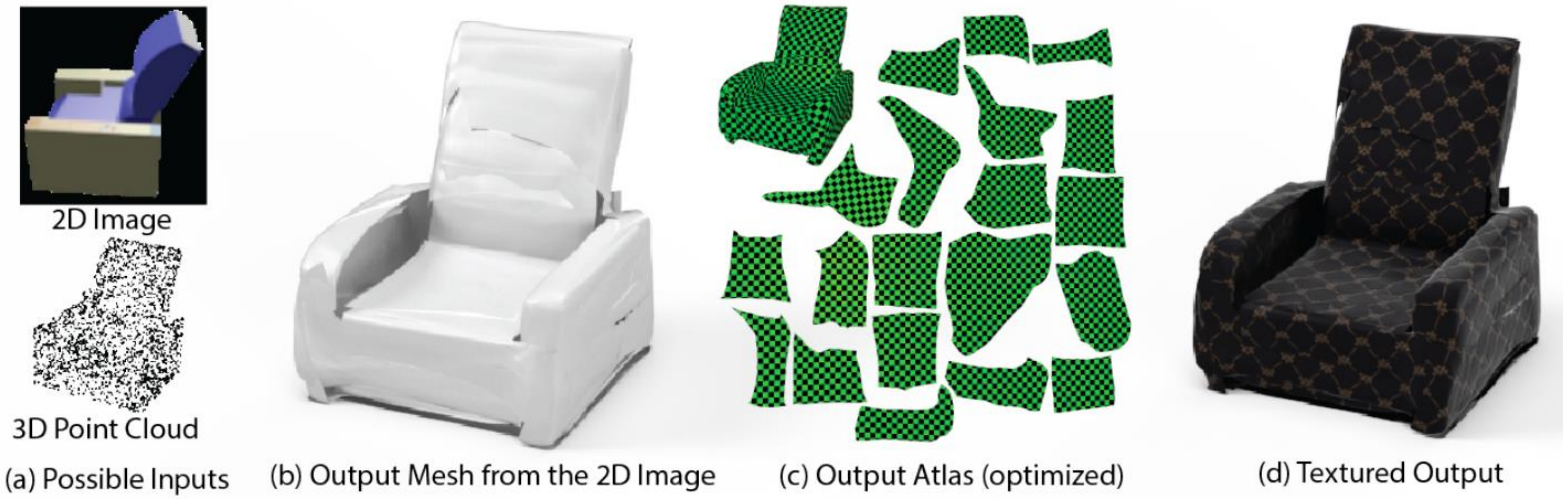



Surface Mesh

$M = (V, F)$

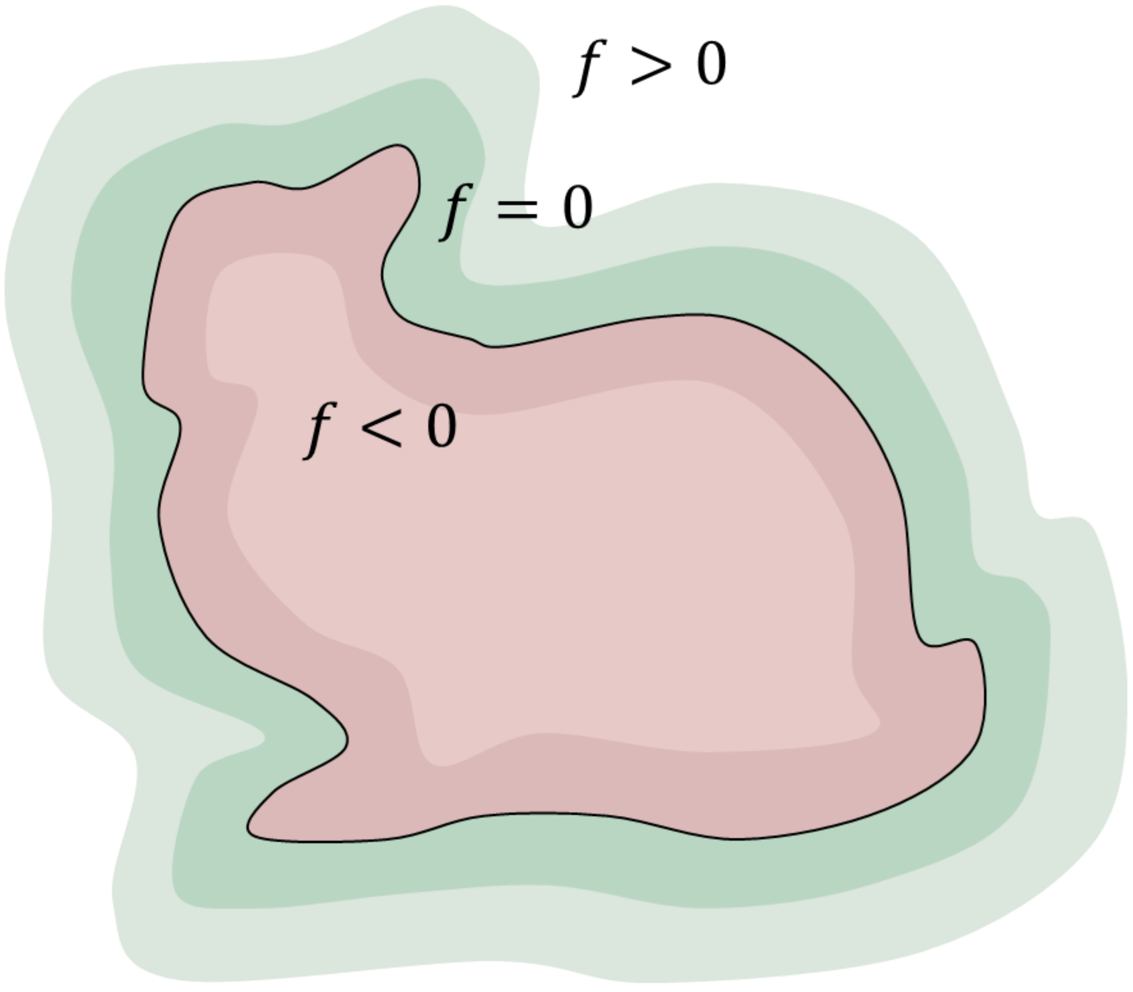



Pixel2Mesh [Wang et al. 2018]




AtlasNet [Groueix et al. 2018]

# GENERATING MESH VIA GRADIENT-BASED **OPTIMIZATION**




Scalar Field
$f : \mathbb{R}^3 \to R$

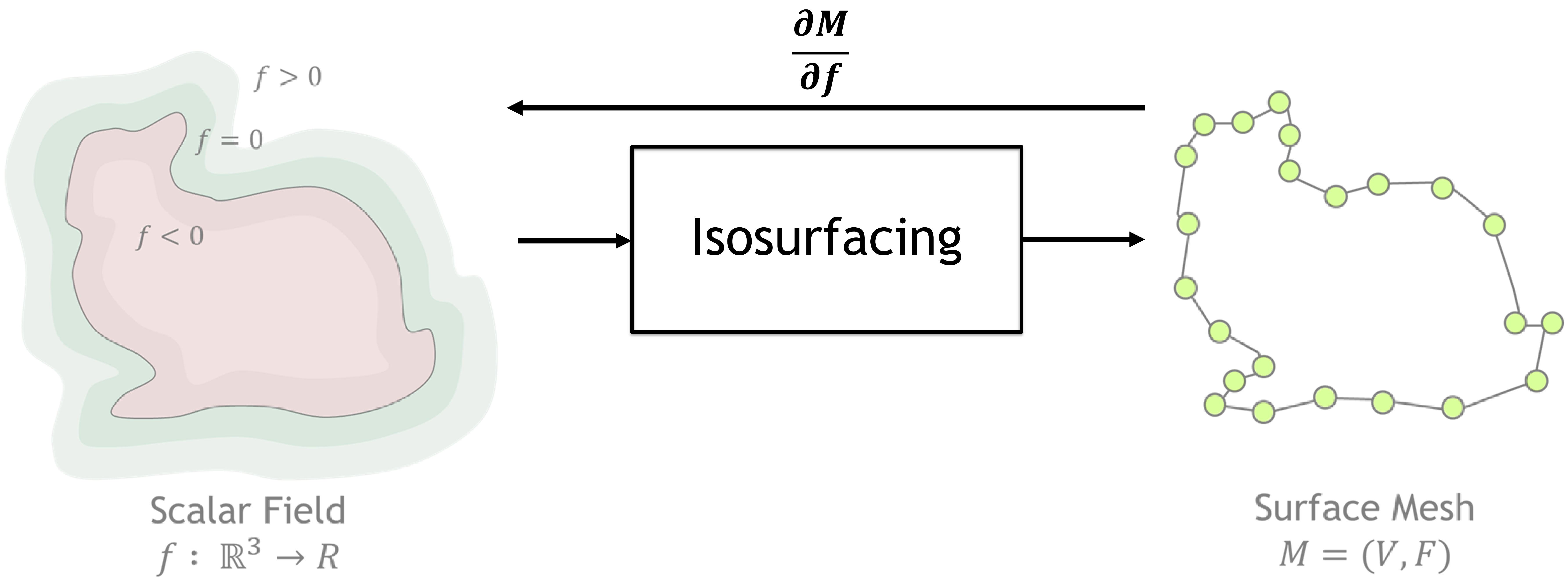
Using implicit field to parametrize surface:

- ✓ Support Arbitrary topology.
- ✓ Avoid topological errors.
- ✓ Easy for machine learning algorithms.

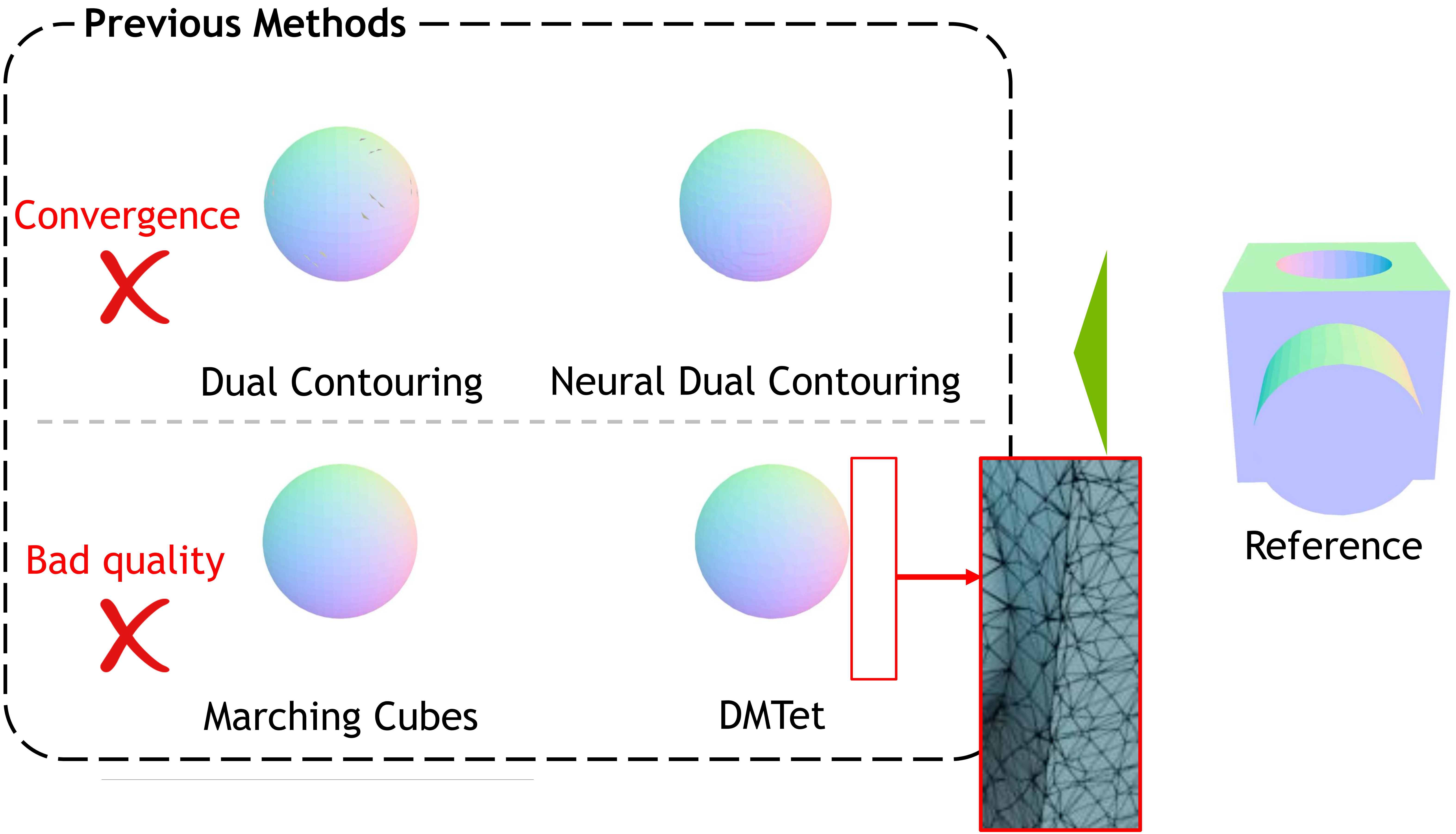
# DESIRED ISOSURFACING METHOD

Accuracy + mesh quality.

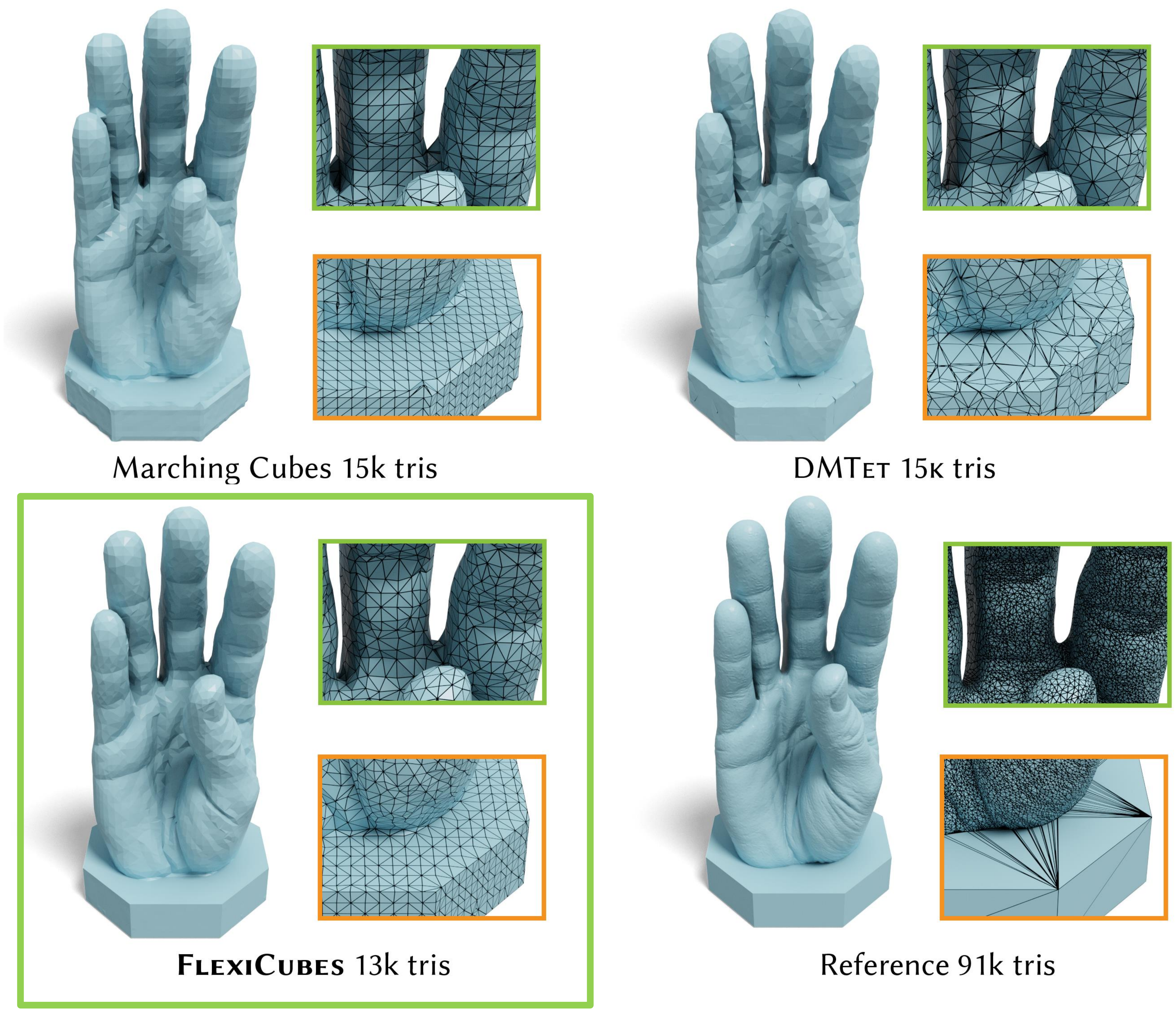
Well-defined gradient differentiation.



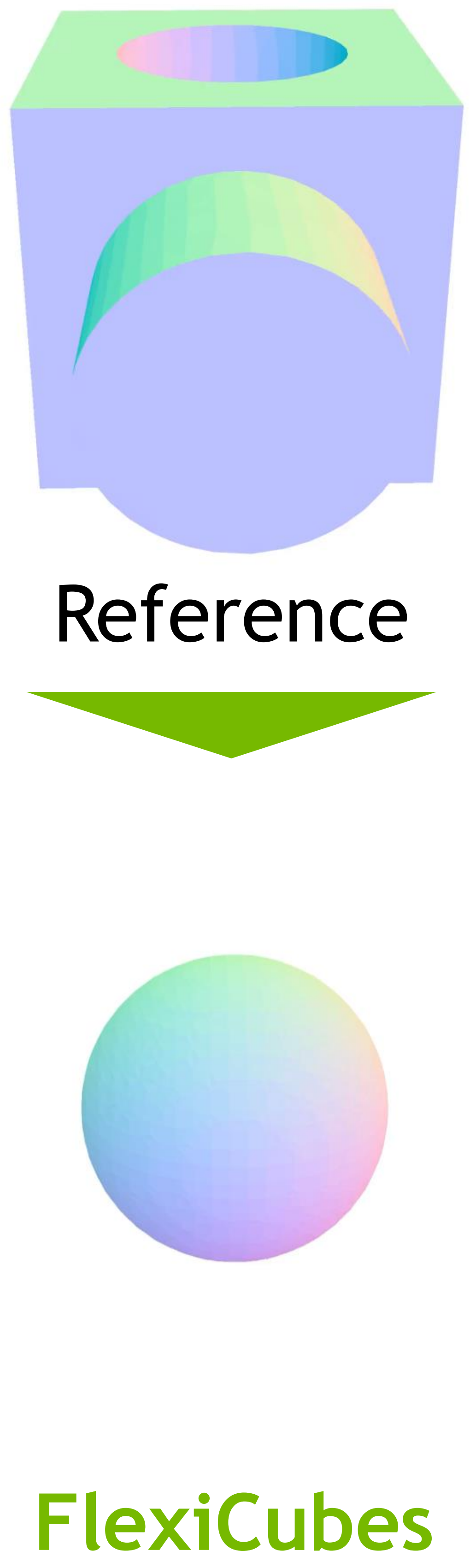
# PREVIOUS METHODS FAIL TO SATISFY BOTH PROPERTIES



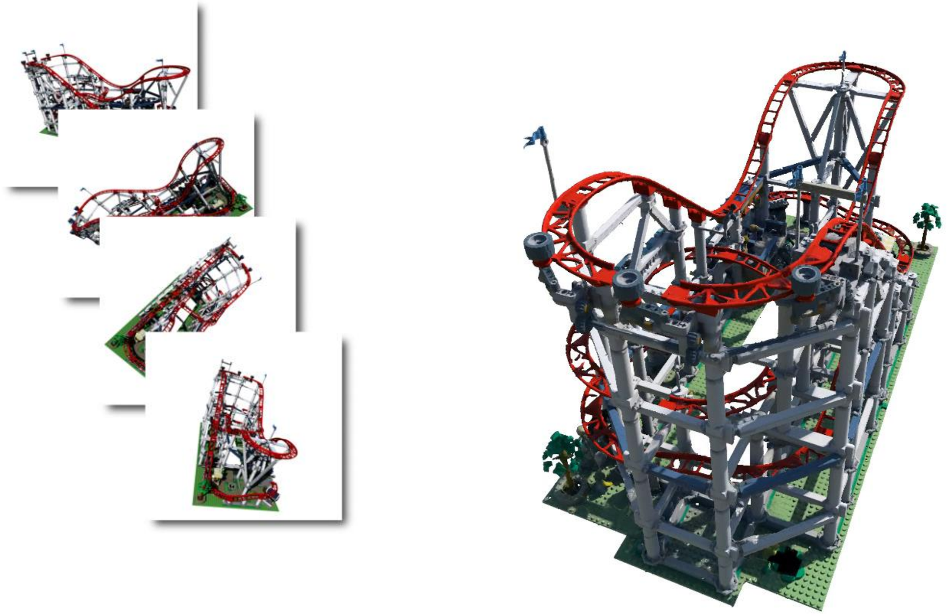
# WE INTRODUCE FLEXICUBES




- High-fidelity
- Good Quality
- Convergence →



# APPLICATIONS W/ FLEXICUBES



3D Reconstruction from Images

Generative 3D Modeling

Mesh Regularizations

Animated 3D Reconstruction

Physics Simulation

… and more!

# WHY IS PRIOR WORK NOT SUFFICIENT?

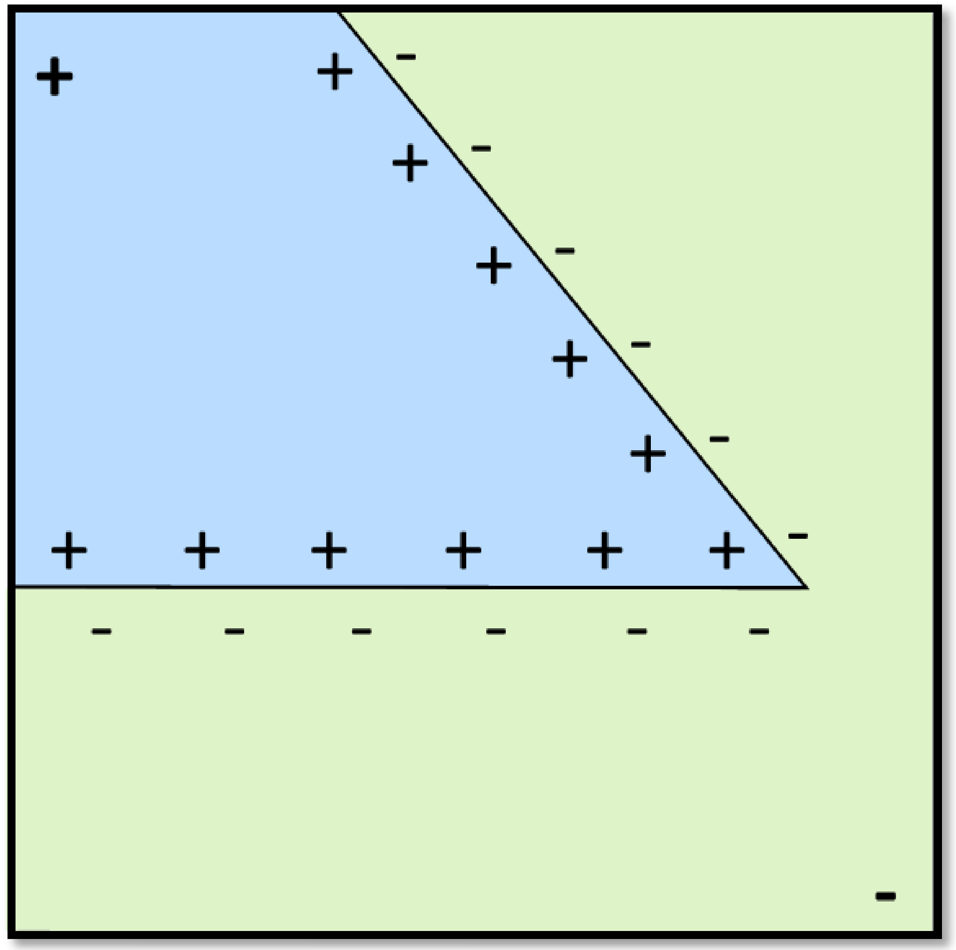
# MARCHING CUBES

*SDF* — *vertex position*

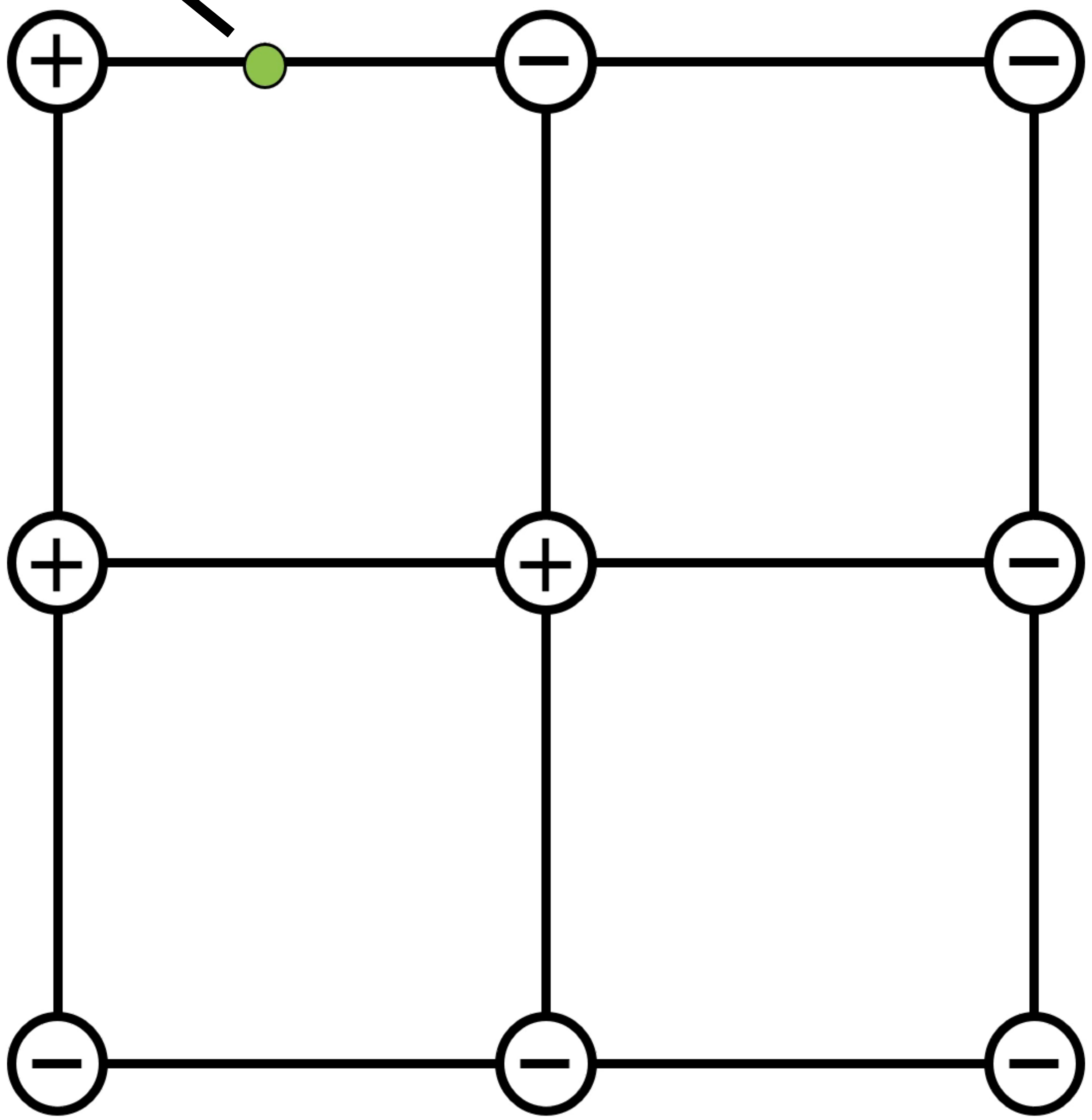
$$\text{zero crossing: } u_e = \frac{s(x_i)x_j - s(x_j)x_i}{s(x_i) - s(x_j)}$$

✗ Mesh vertices are constrained on lattice edges.

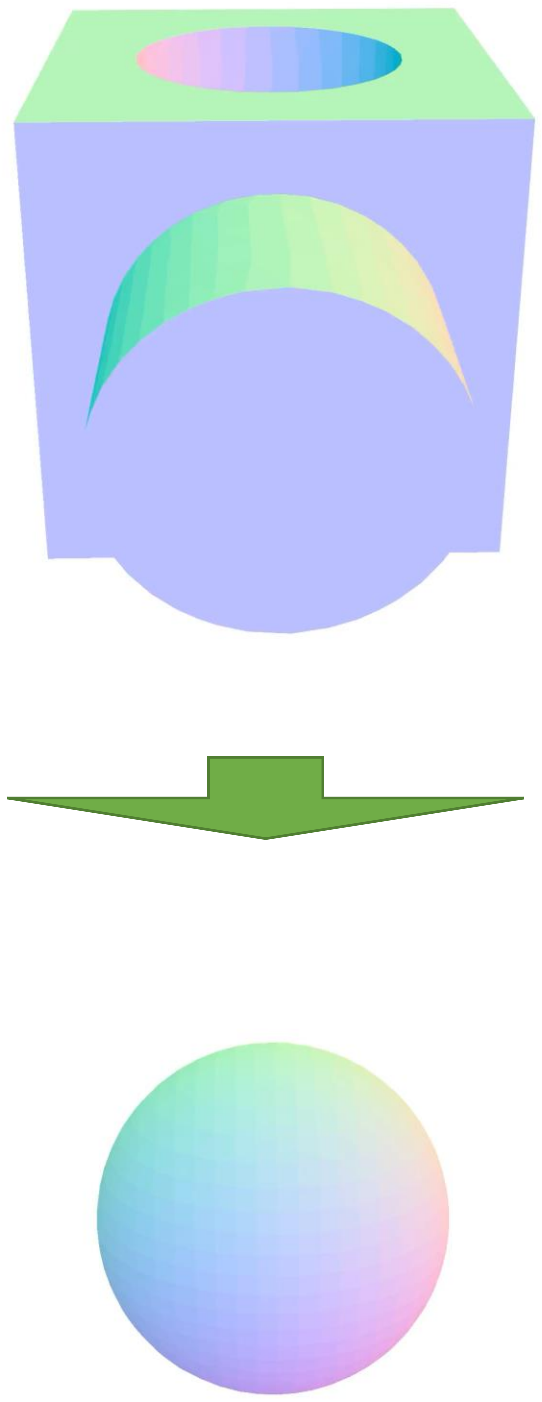
Ground Truth




Signed Distance Field



Marching Cubes

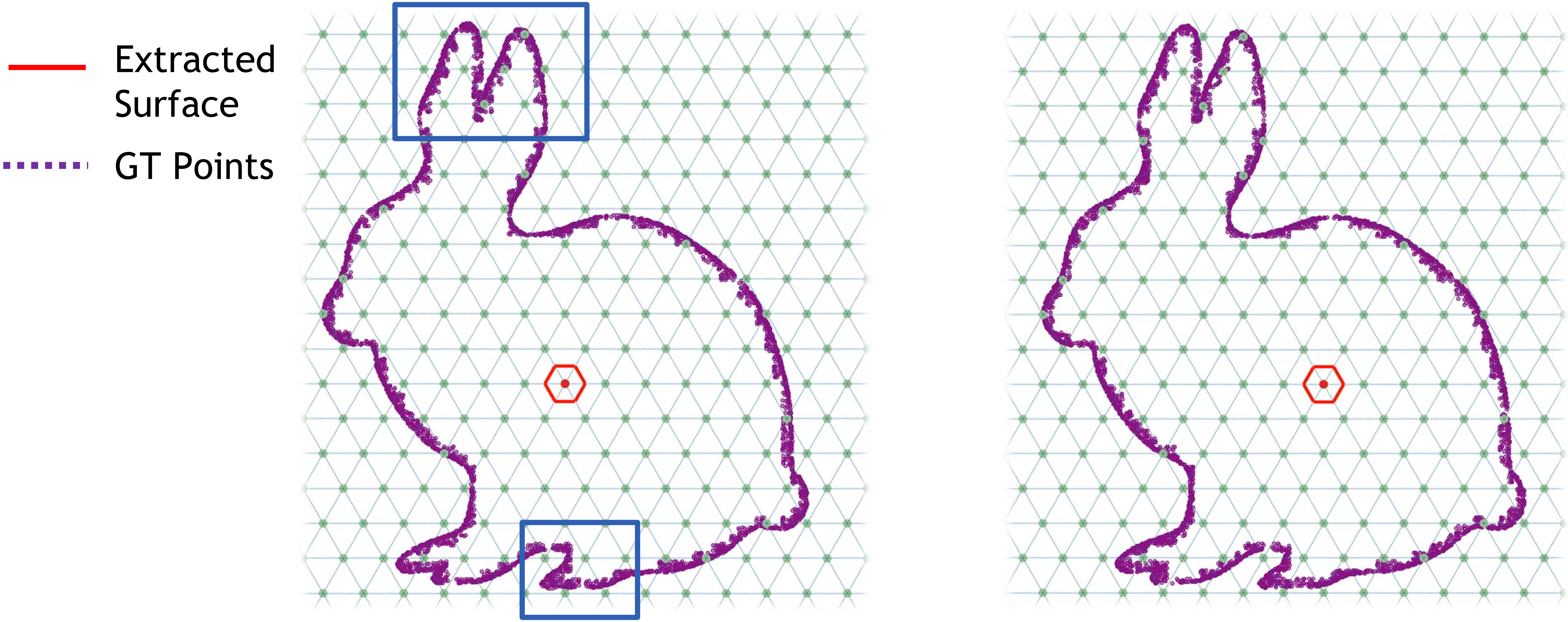


✓ Grad.

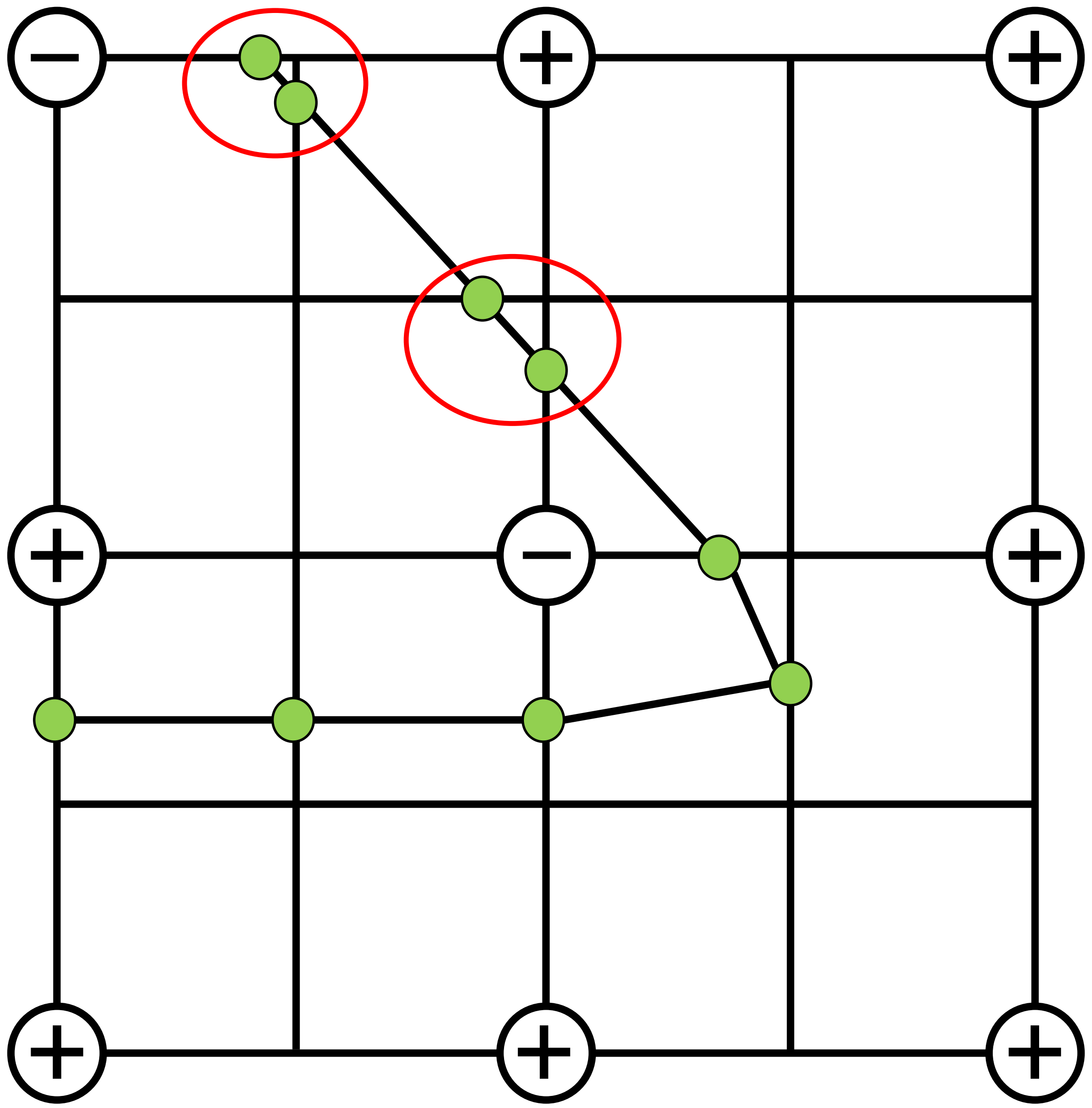
# DEEP MARCHING TETRAHEDRA (DMTET)

*SDF* → $s(x_i)$; *vertex position* → $x_j$

$$zero\ crossing:\ u_e = \frac{s(x_i)x_j - s(x_j)x_i}{s(x_i) - s(x_j)}$$



# MARCHING CUBES



Increase grid resolution

Not ideal for optimization settings.

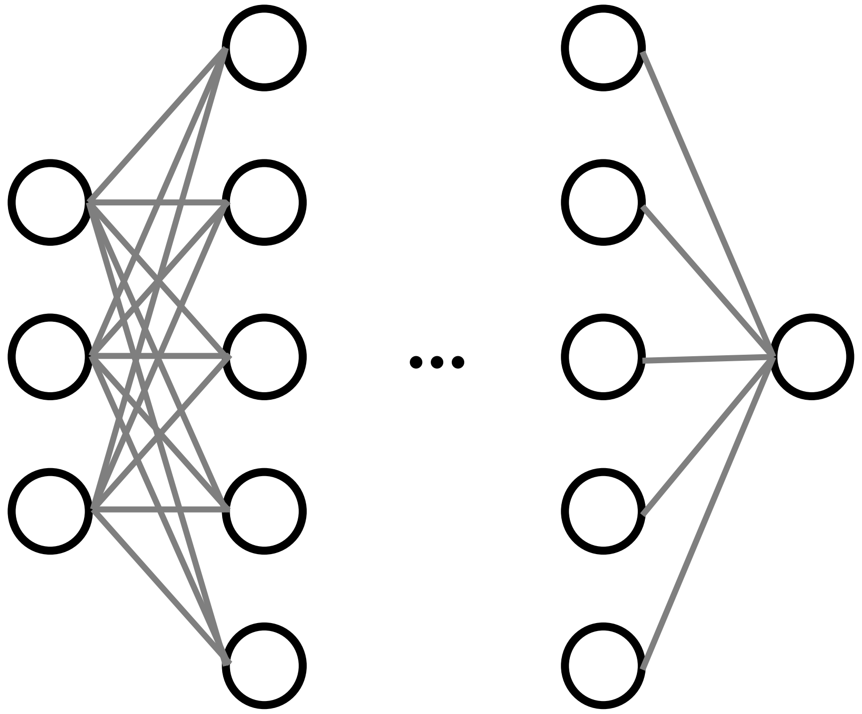
- ❌ Excessive number of triangles
  -> bad for downstream applications.

- ❌ Requires more SDF samples at every optimization step.

Especially for neural implicits s.t.

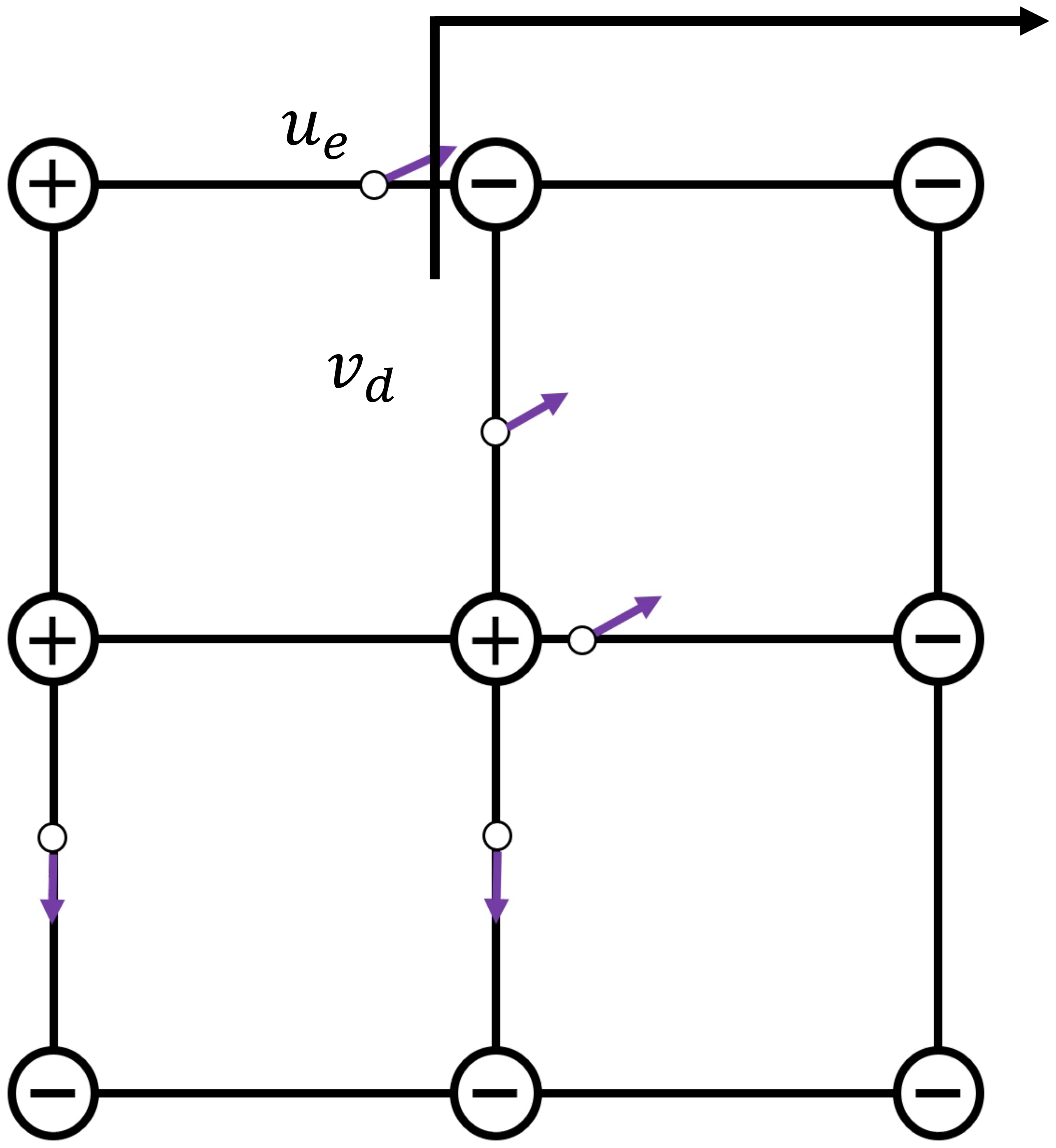
$$s = f_\theta(x)$$

$x \in \mathbb{R}^3$



$s$

# DUAL CONTOURING

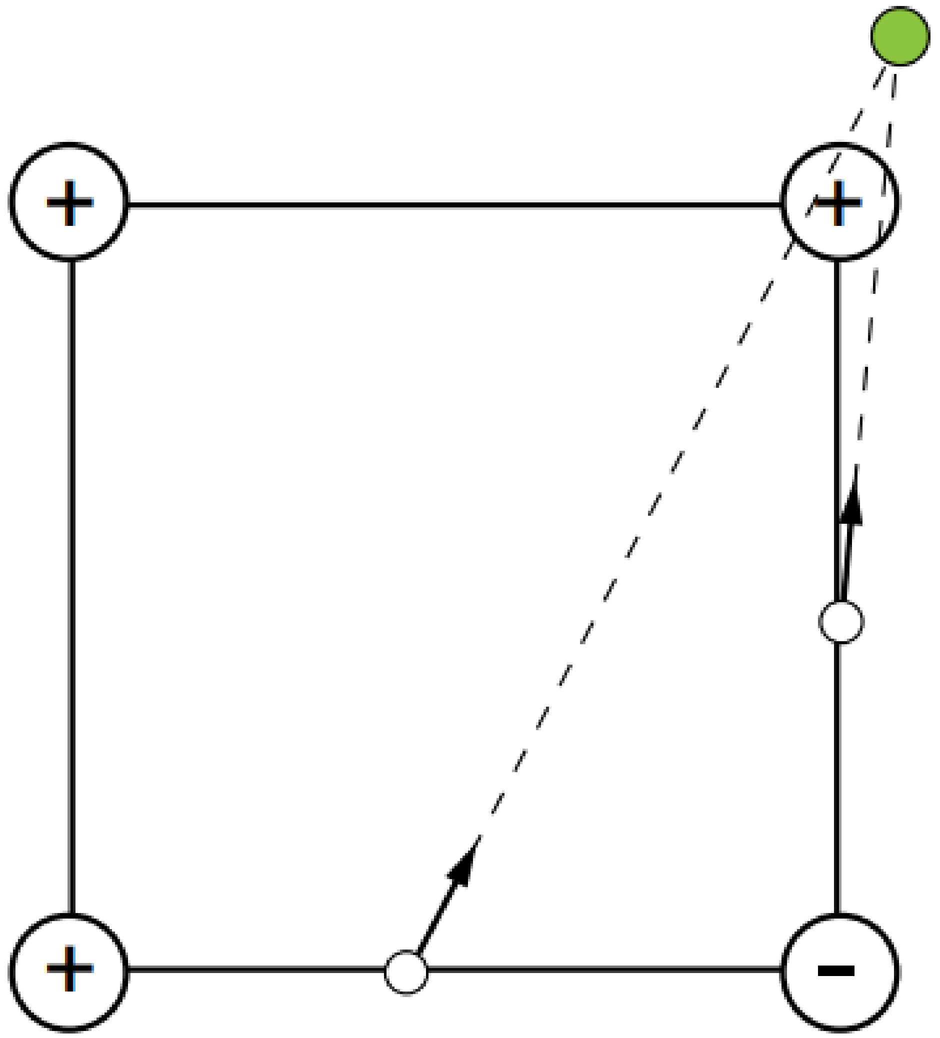



Dual Contouring

$$v_d = \underset{v_d}{\operatorname{argmin}} \sum_{u_e \in \mathcal{Z}_e} \nabla s(u_e) \cdot (v_d - u_e)$$

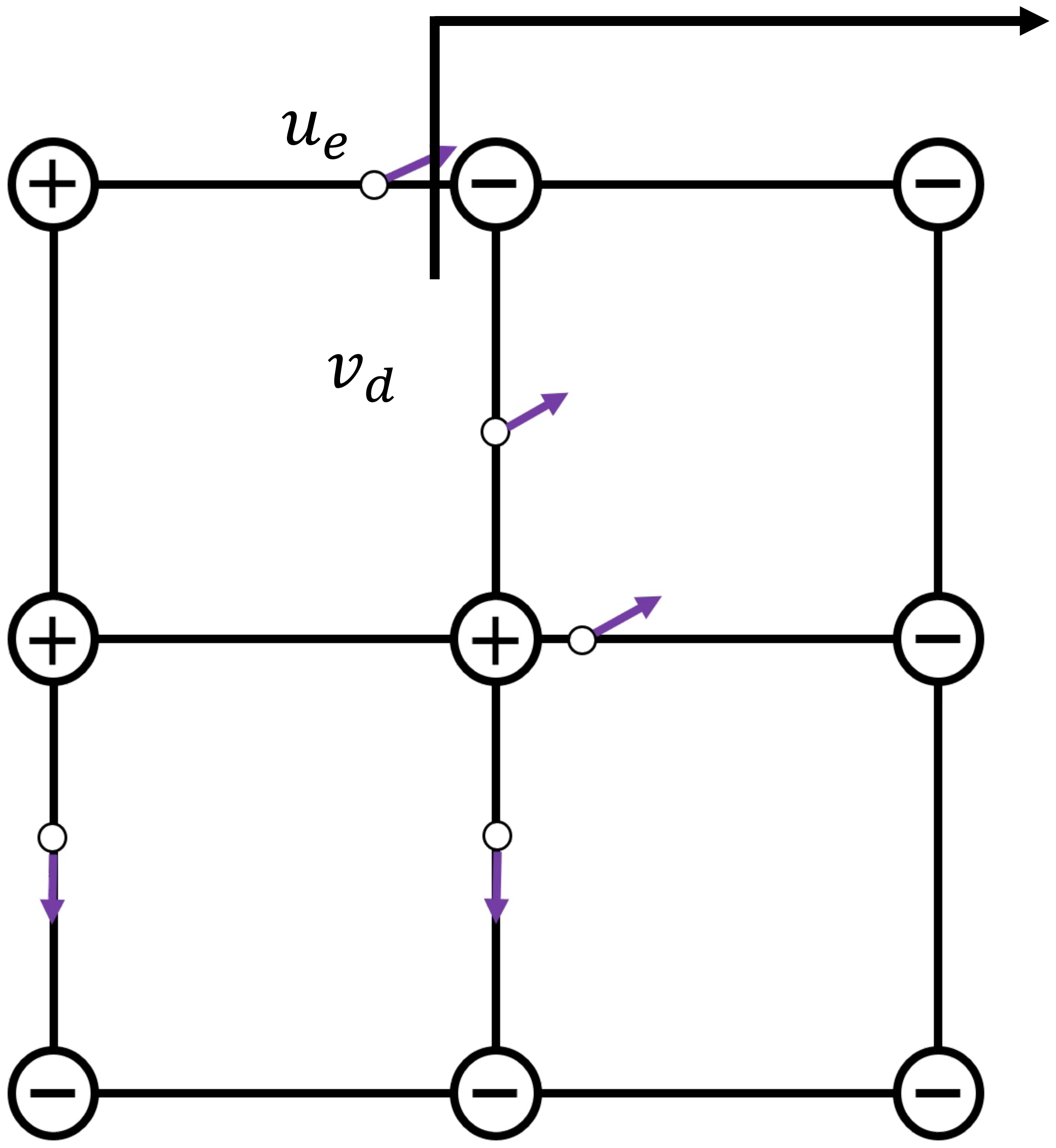
✔ Vertex placement is more adaptive to surface feature.

✘ Grad. issue due to singularity in QEF.



Degenerated config. in DC

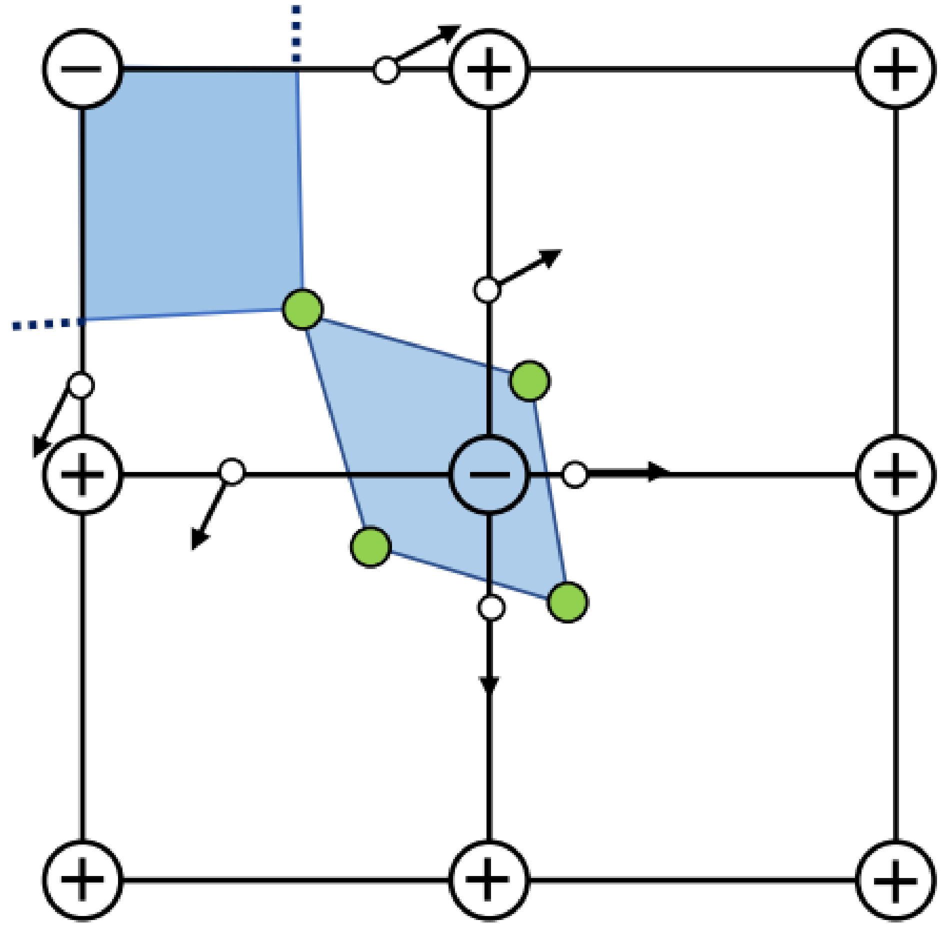
# DUAL CONTOURING




Dual Contouring

$$v_d = \underset{v_d}{\operatorname{argmin}} \sum_{u_e \in \mathcal{Z}_e} \nabla s(u_e) \cdot (v_d - u_e)$$

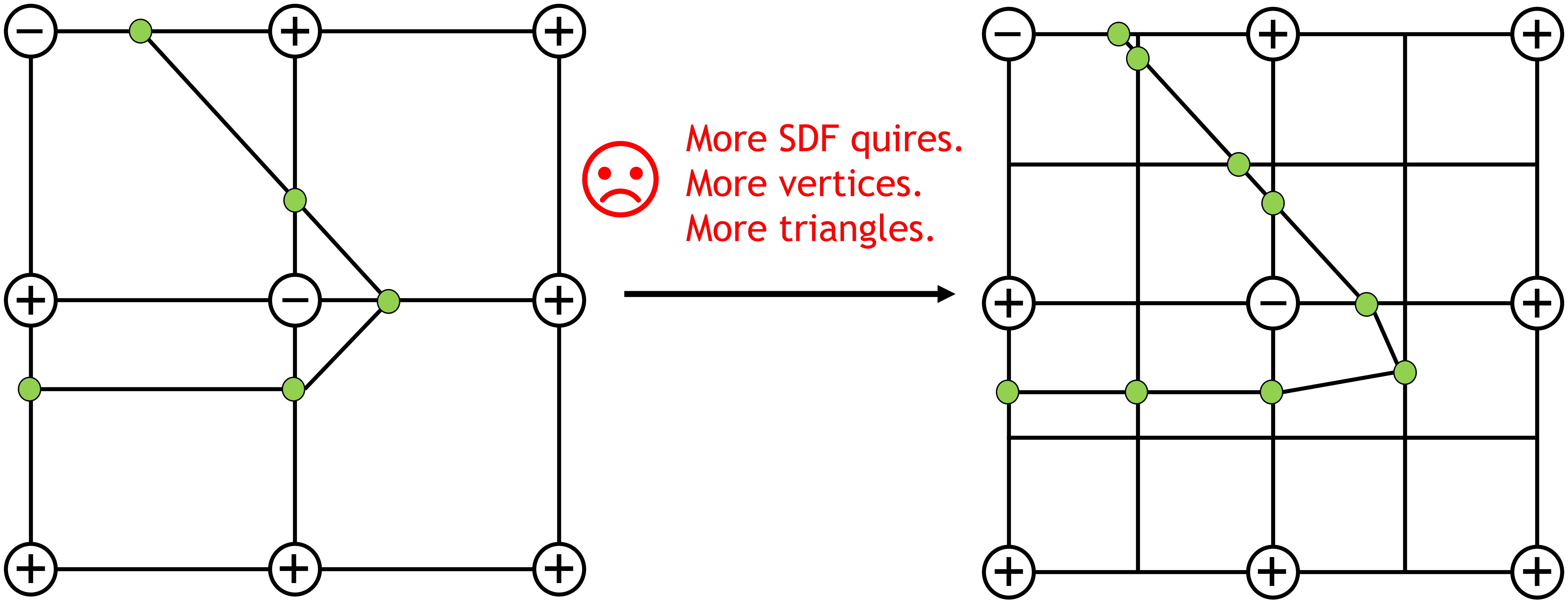
- ✓ Vertex placement is more adaptive to surface feature.
- ✗ Grad. issue due to singularity in QEF.
- ✗ Produce non-manifold results



# OUR FORMULATIONS

# MAIN IDEA

Improve fitting with more elements?

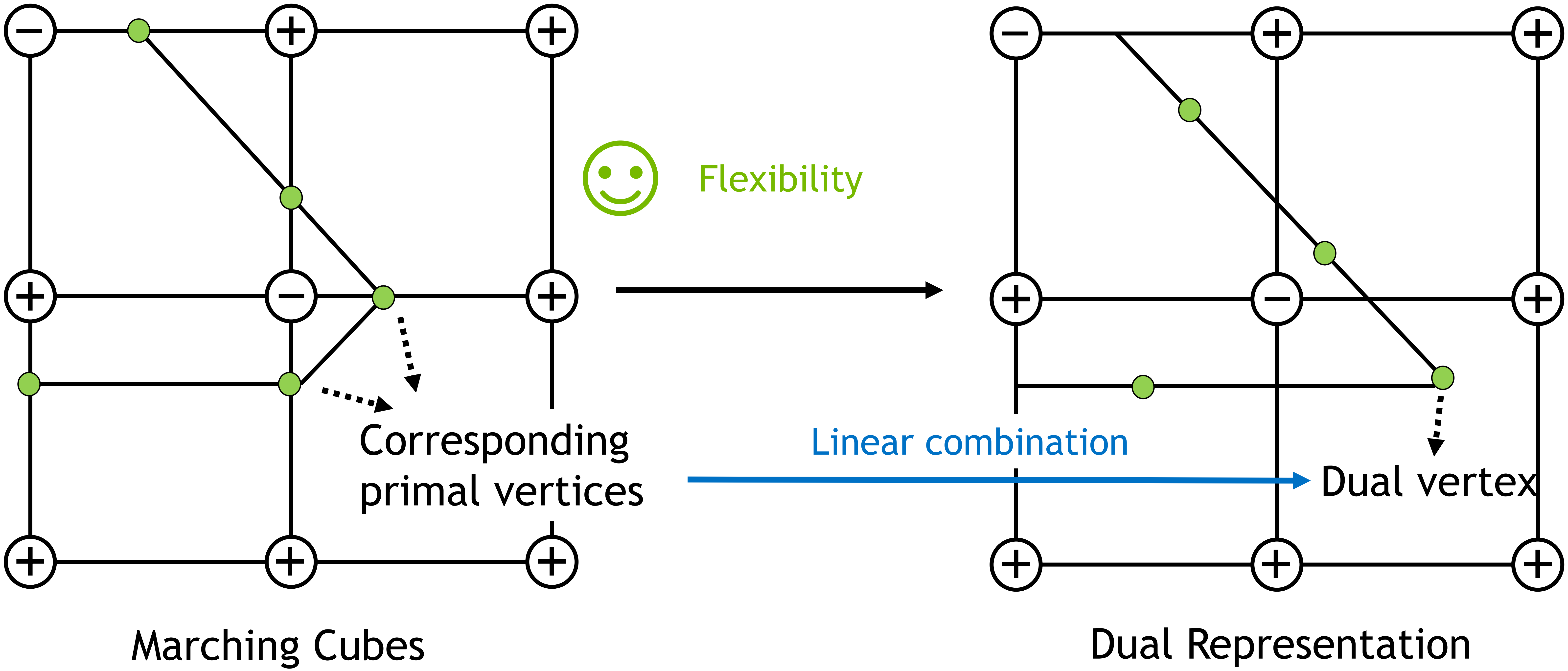



Marching Cubes

# MAIN IDEA

Improve fitting with ~~more elements?~~ **additional flexibility!**

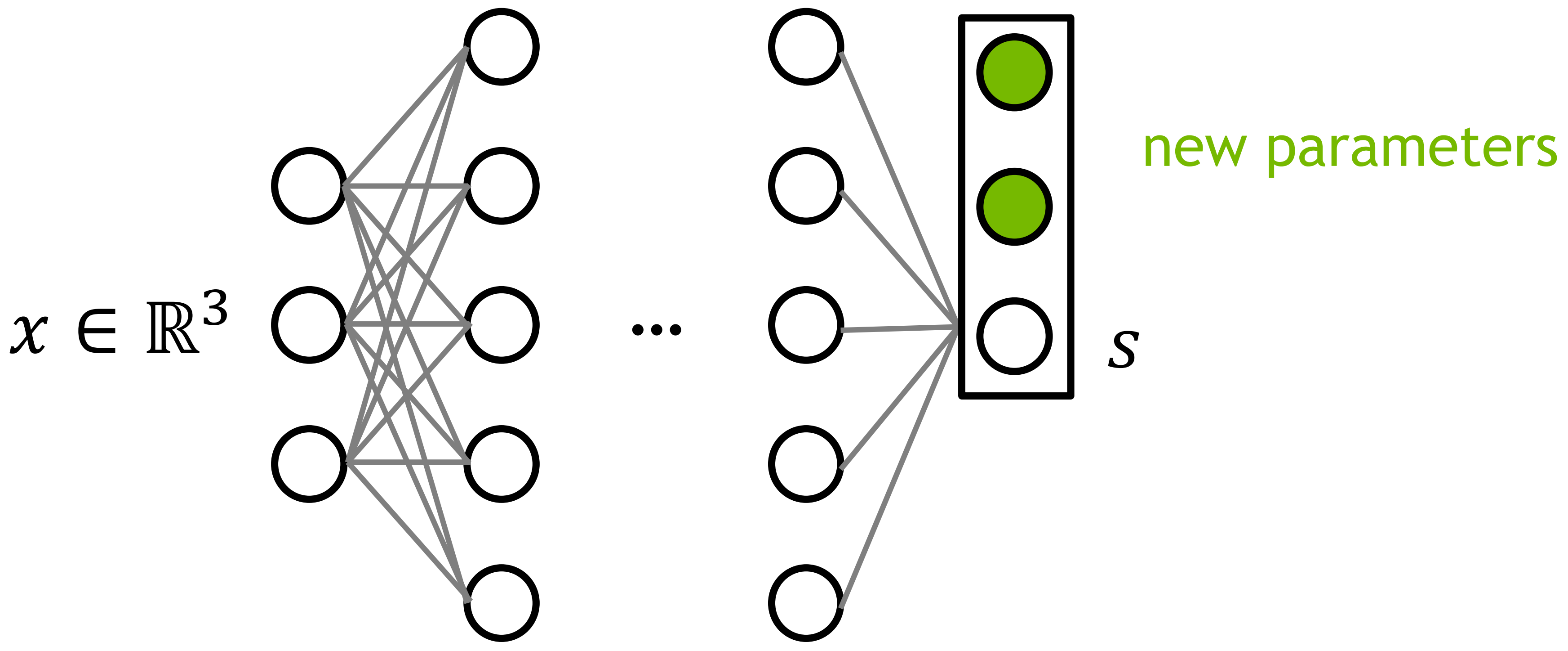
Recall prior dual methods **diverge** during optimization.



# OUR FORMULATIONS

We introduce 3 types of parameters into Dual Marching Cubes:

- Interpolation weights to position dual vertices in space.
- Splitting weights to control how to split quadrilaterals into triangles.
- Deformation vector for spatial alignment.

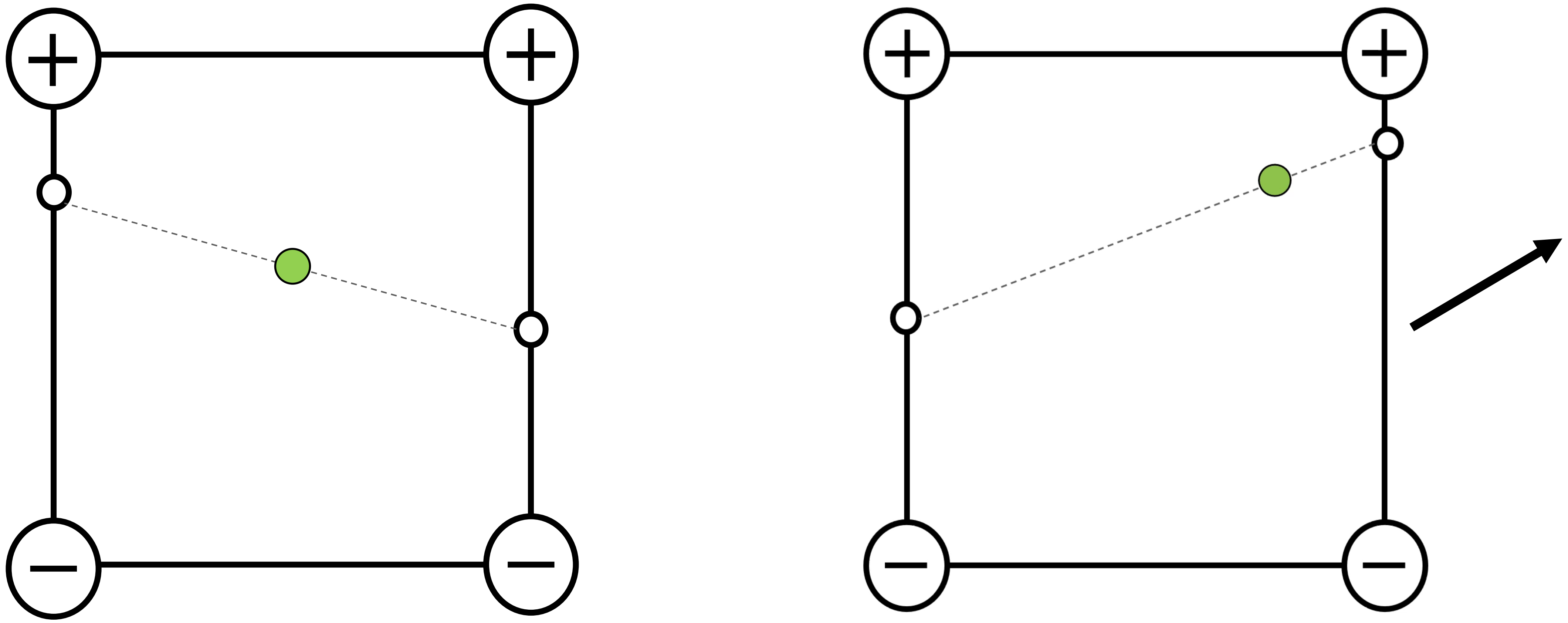



✓ Appended as additional output channel in neural implicit.

# OUR FORMULATION

Interpolation weights:

- $\alpha$ per-cell adjusting interpolation along each edge.



Centroid

With $\alpha$

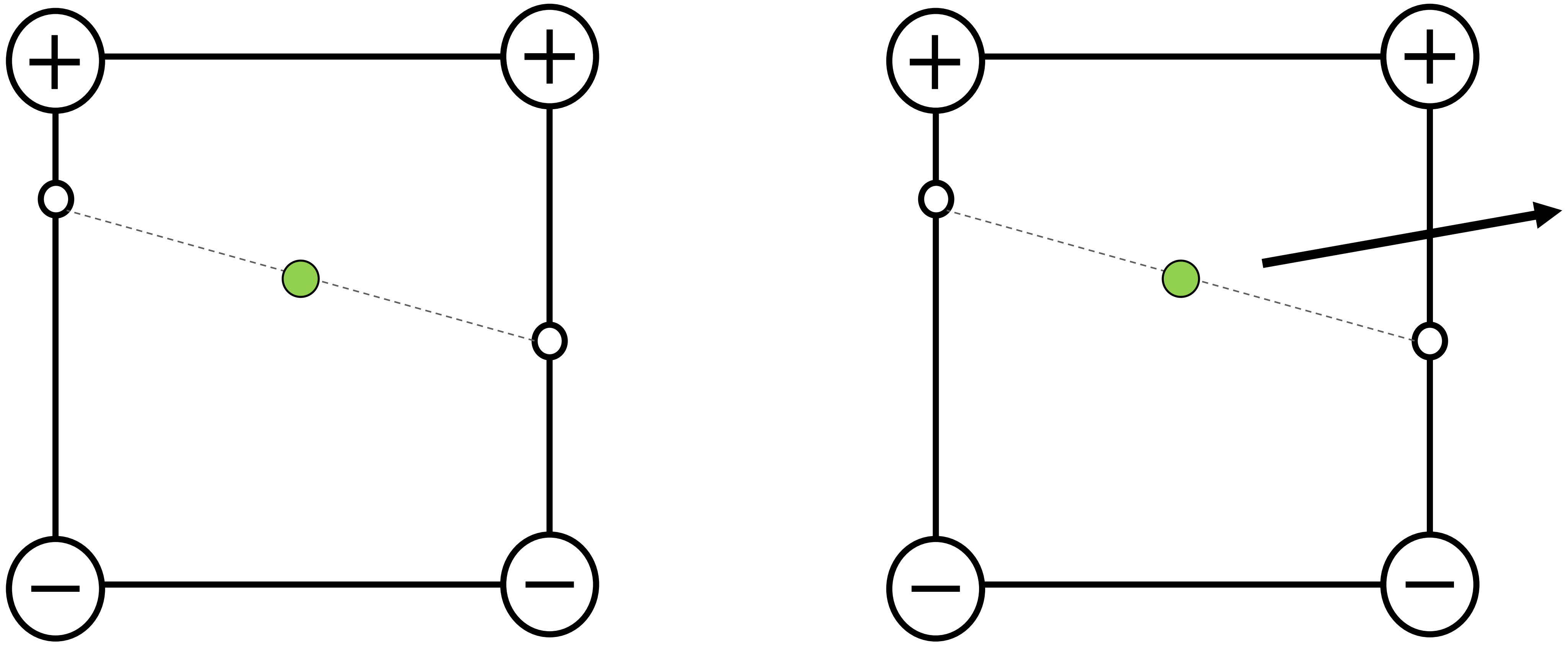
Linear interpolation:

$$u_e = \frac{\boldsymbol{\alpha}_i s(x_i)x_j - \boldsymbol{\alpha}_j s(x_j)x_i}{\boldsymbol{\alpha}_i s(x_i) - \boldsymbol{\alpha}_j s(x_j)}$$

# OUR FORMULATION

Interpolation weights:

- $\alpha$ per-cell adjusting interpolation along each edge.
- $\beta$ per-cell adjusting vertex position within each dual face



Centroid

With $\beta$

Weighted average:

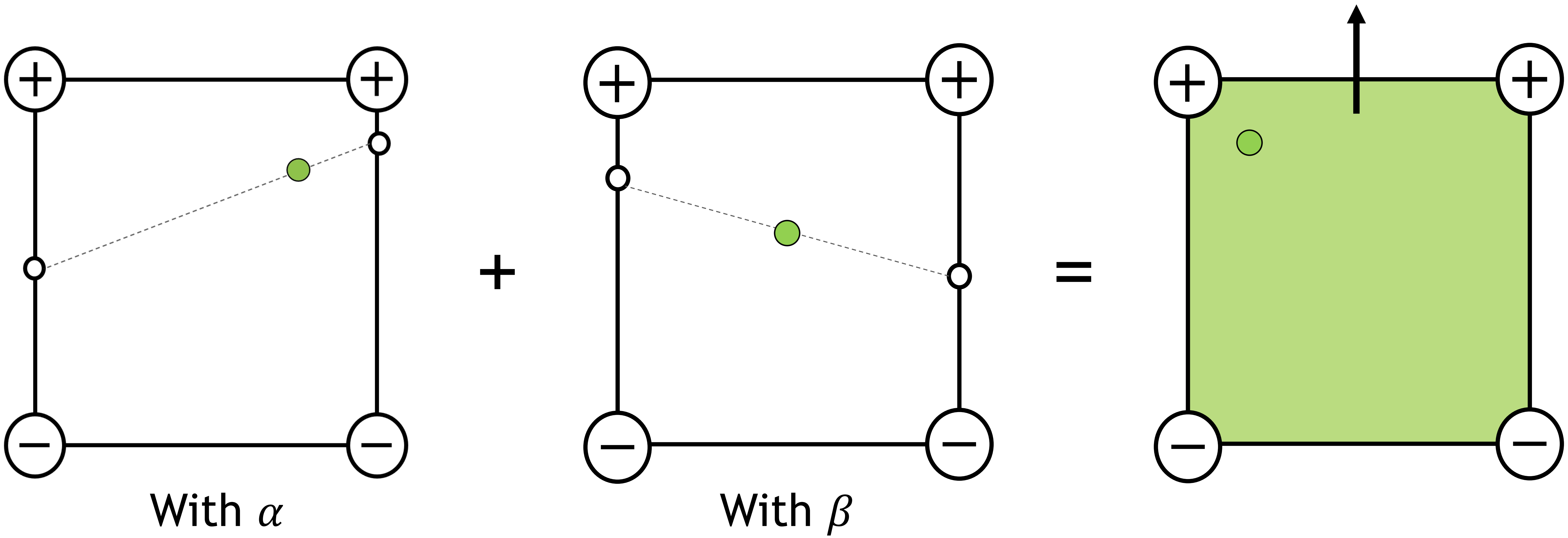
$$v_d = \frac{1}{\sum_{u_e \in V_E} \boldsymbol{\beta}_e} \sum \boldsymbol{\beta}_e u_e$$

# INTERPOLATION WEIGHTS

- $\alpha$ per-cell adjusting **interpolation** along each edge.
- $\beta$ per-cell adjusting vertex position within each dual face

→ Convex combinations

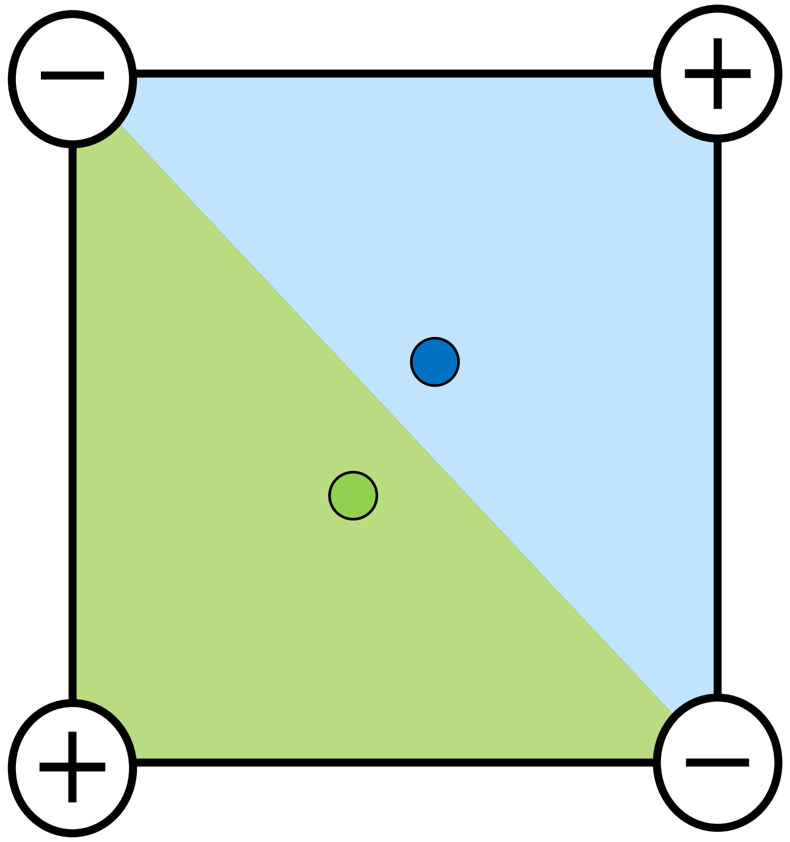
Movable regions of the dual vertex.



# INTERPOLATION WEIGHTS

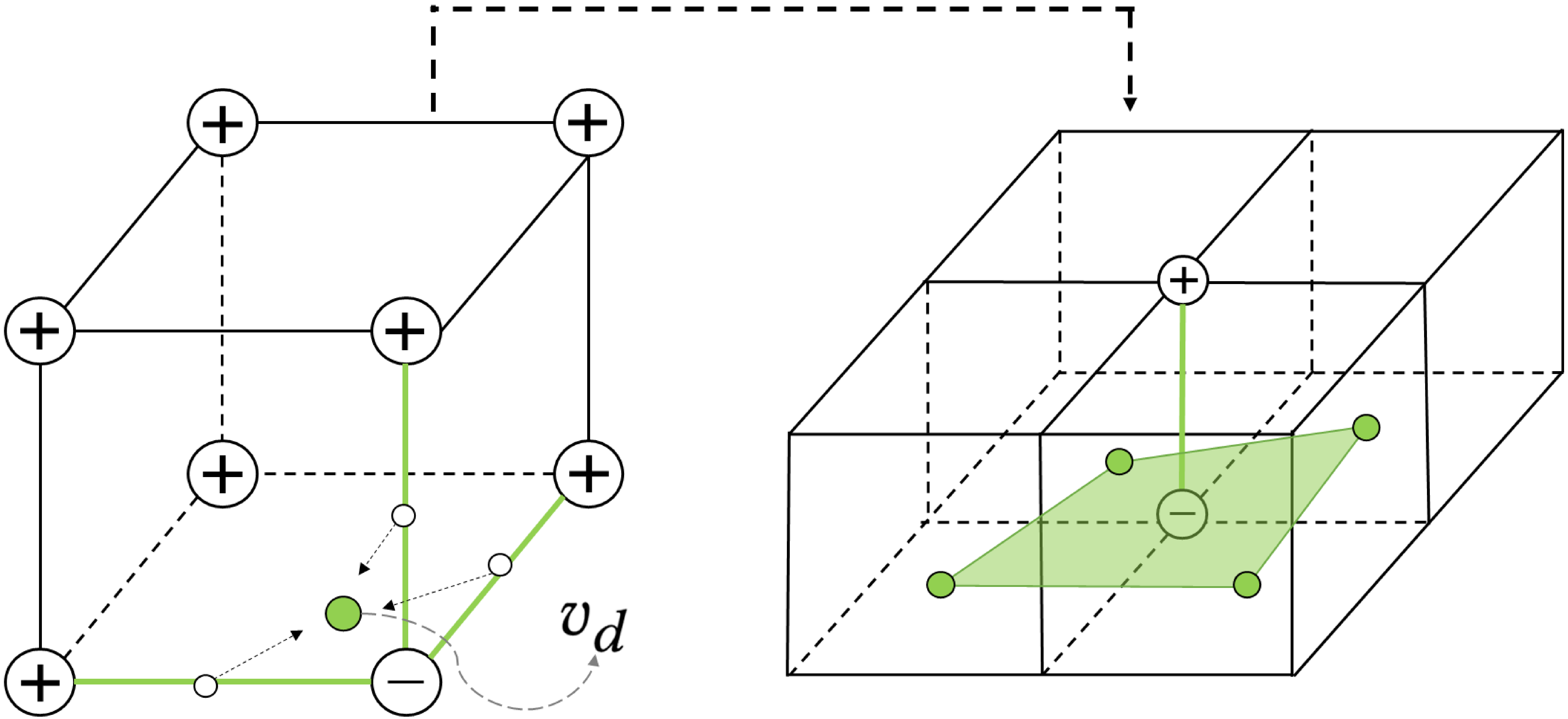
- $\alpha$ per-cell adjusting interpolation along each edge.
- $\beta$ per-cell adjusting vertex position within each dual face

$\alpha, \beta \in \mathbb{R}^+$ → ✓ Preserves grad.




✓ When a cell emits multiple dual vertices, they lies in non-overlapping convex hulls.

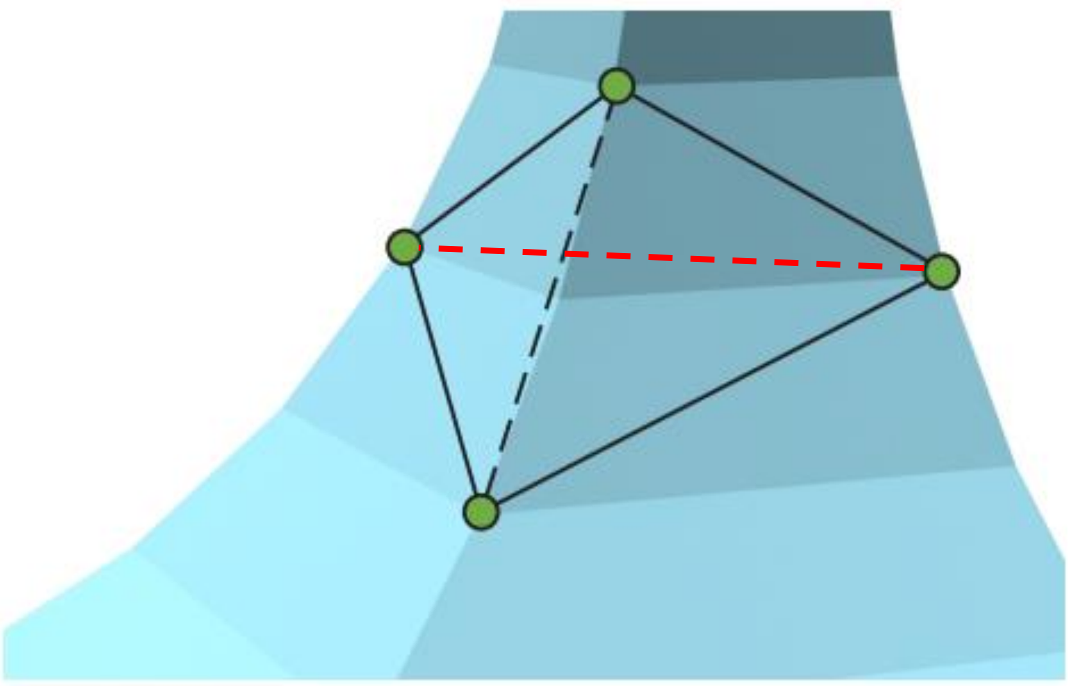
# EXTRACTING FACES

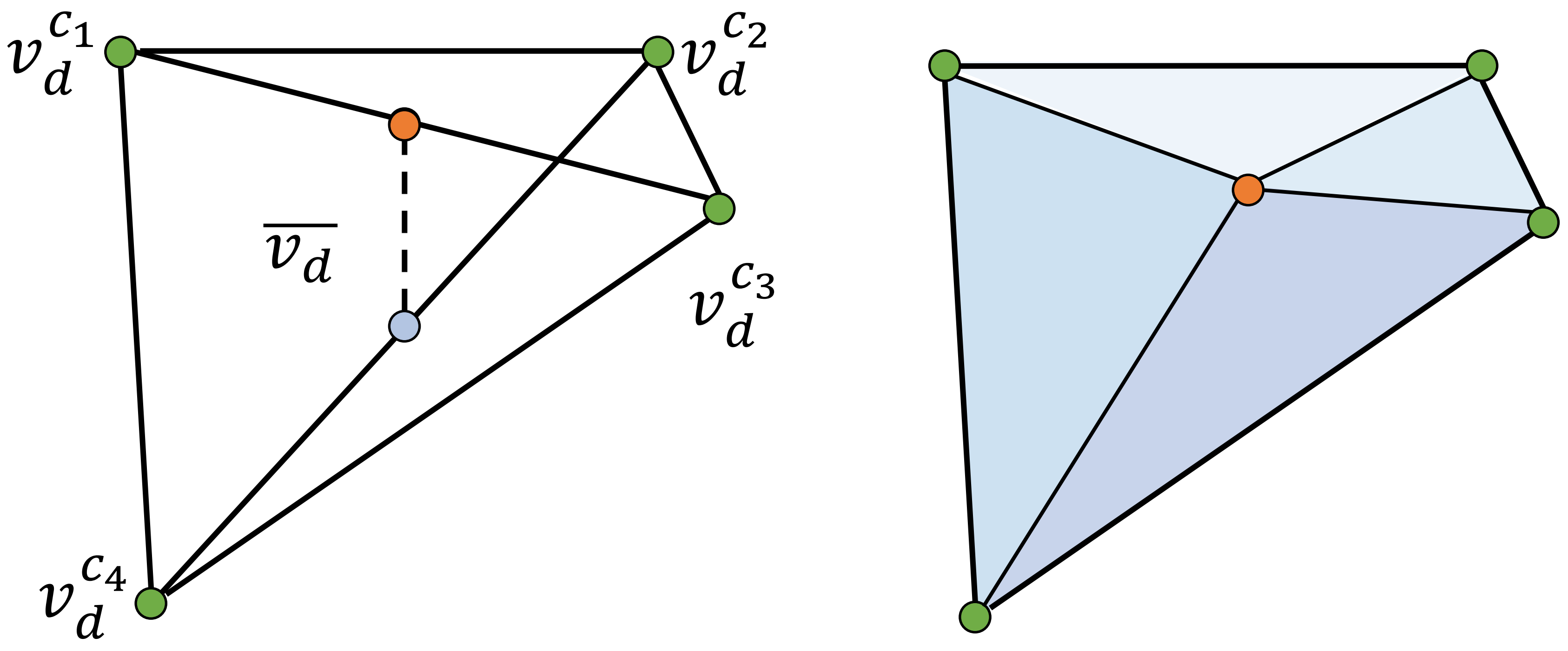


# SPLITTING WEIGHTS

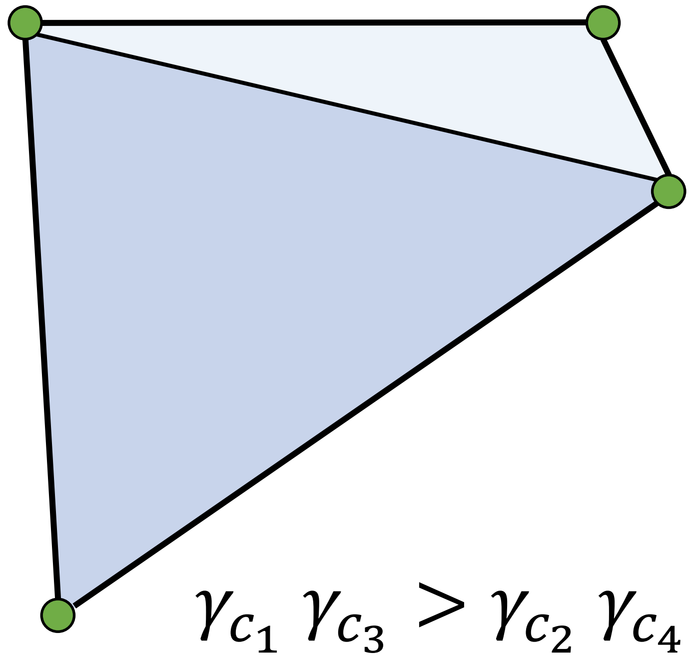
Raw output from Dual Contouring are quadrilateral faces.

- Quad-split weights $\gamma$ controlling how quads get split to tris.



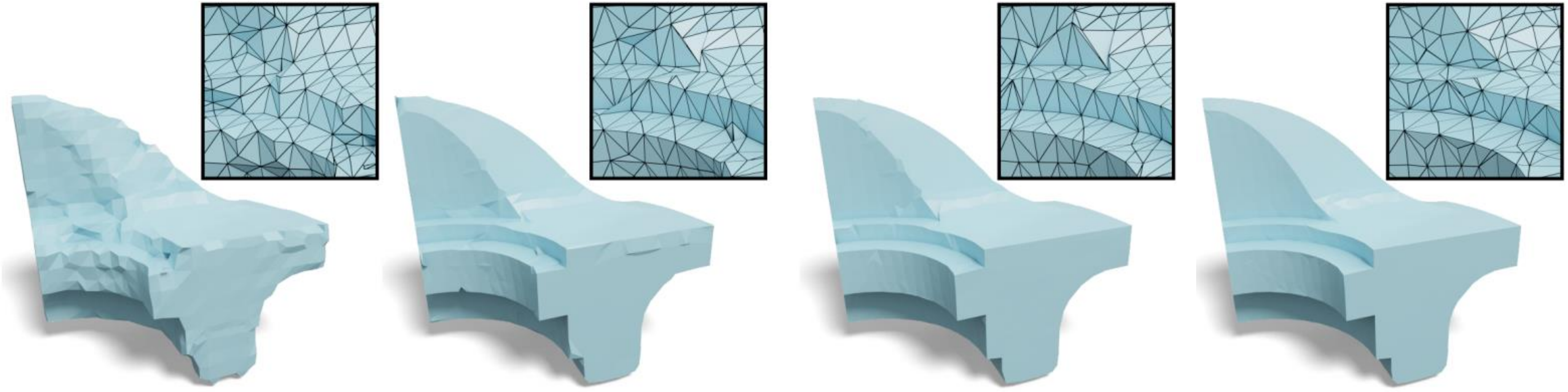



During **optimization**, $\gamma$ interpolates the surface between two possible splits.




At **inference**, we split along the diagonal with dominant $\gamma$

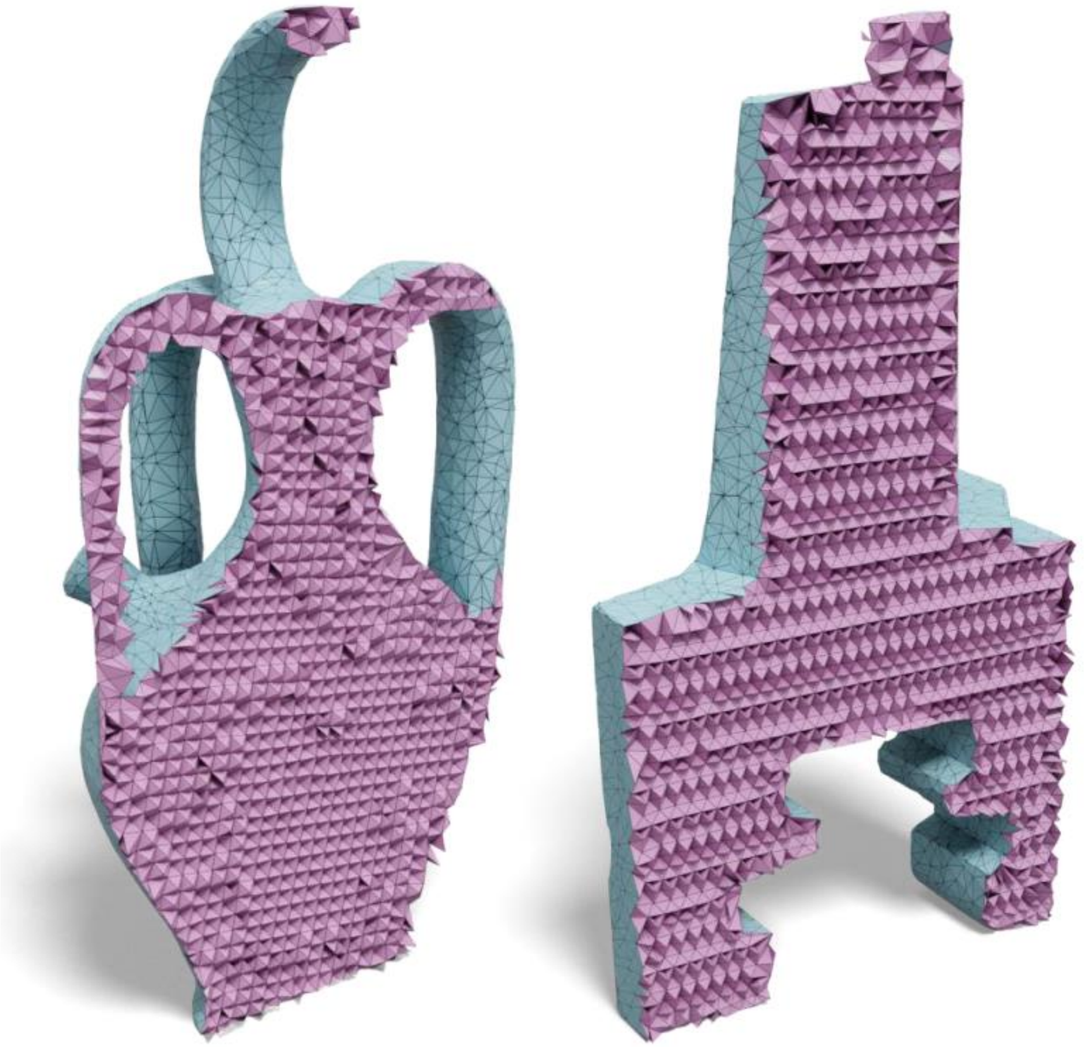
# ABLATING PARAMETERS

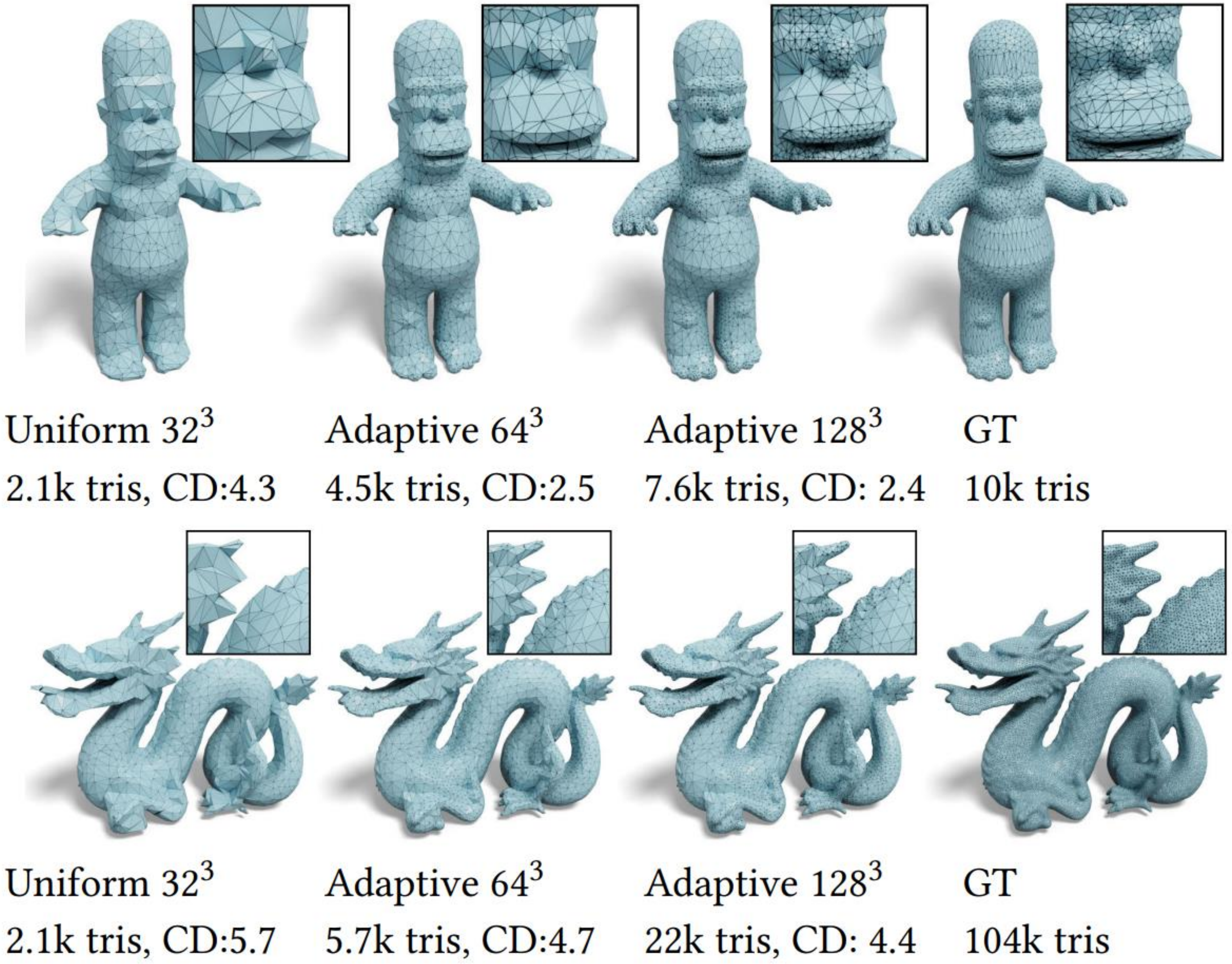


Baseline | + interp. weights | + grid deform | + split weights

# EXTENSIONS

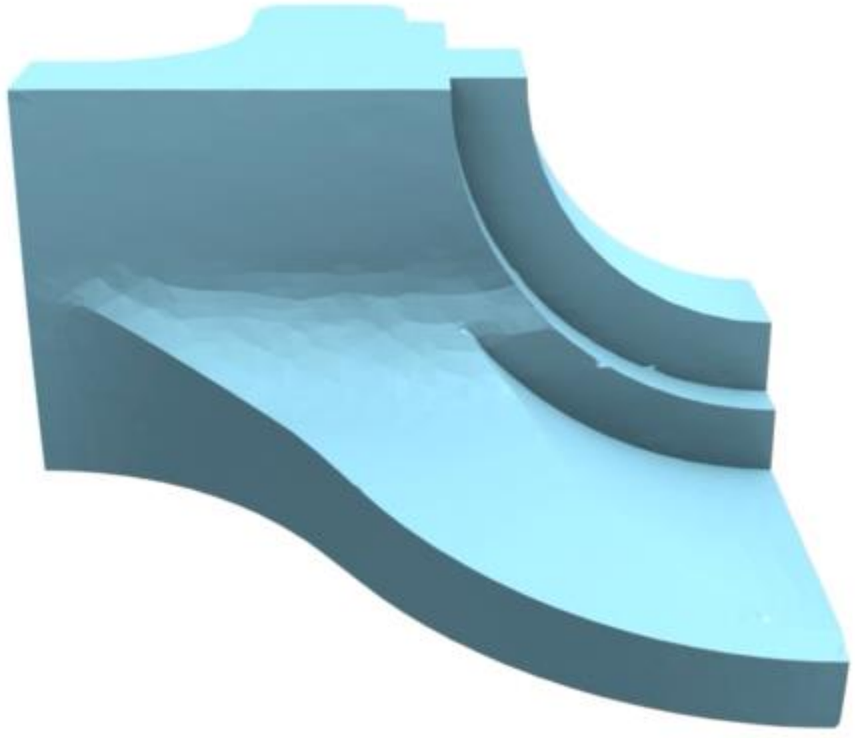
- Tetrahedral mesh extraction.
- Hierarchically adaptive meshing



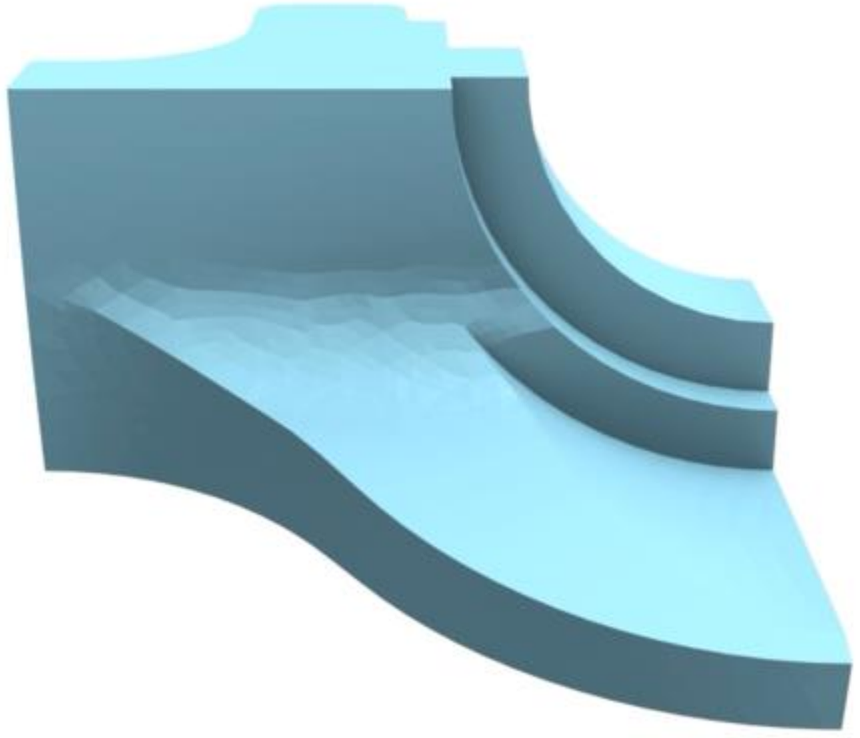


# VALIDATIONS

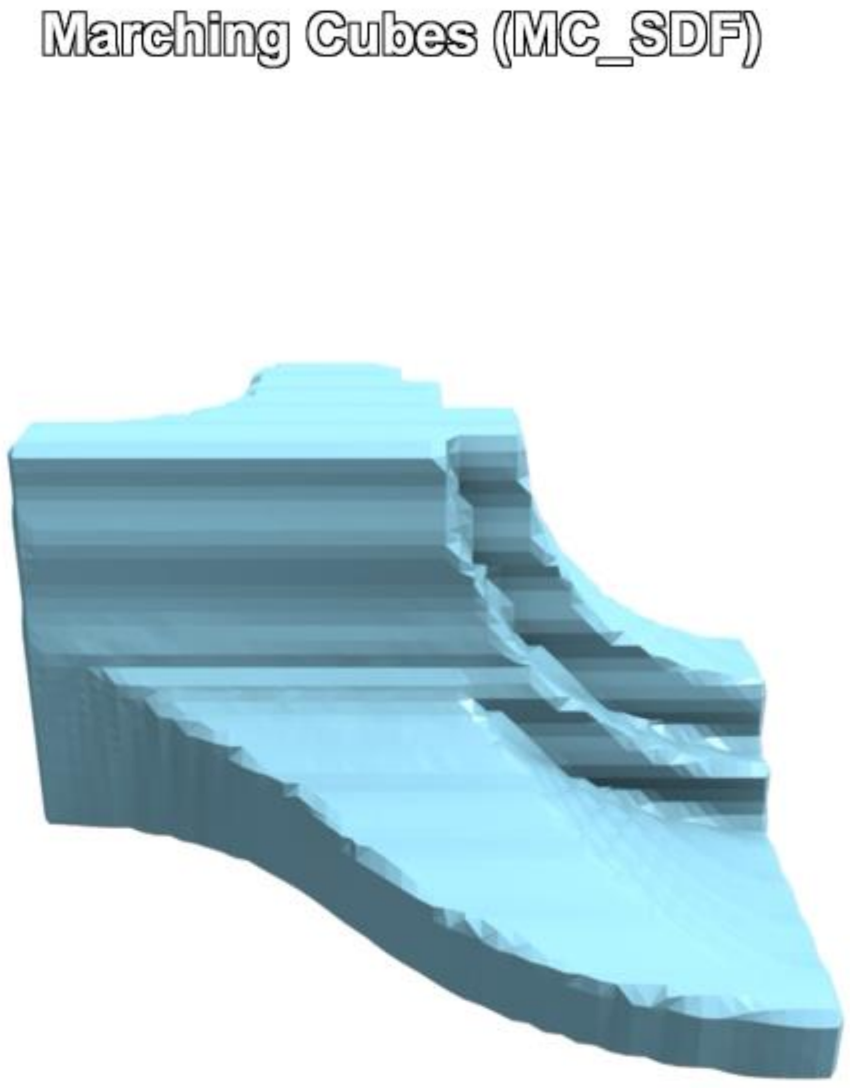
# VISUAL COMPARISON



FlexiCubes



Reference




Baselines

FlexiCubes @ 64 resolution

# QUANTITATIVE COMPARISONS OF RECONSTRUCTION ERROR

Evaluated on dataset collected by Myles et al. [2014] which contains 79 highly-detailed, diverse 3D shapes.

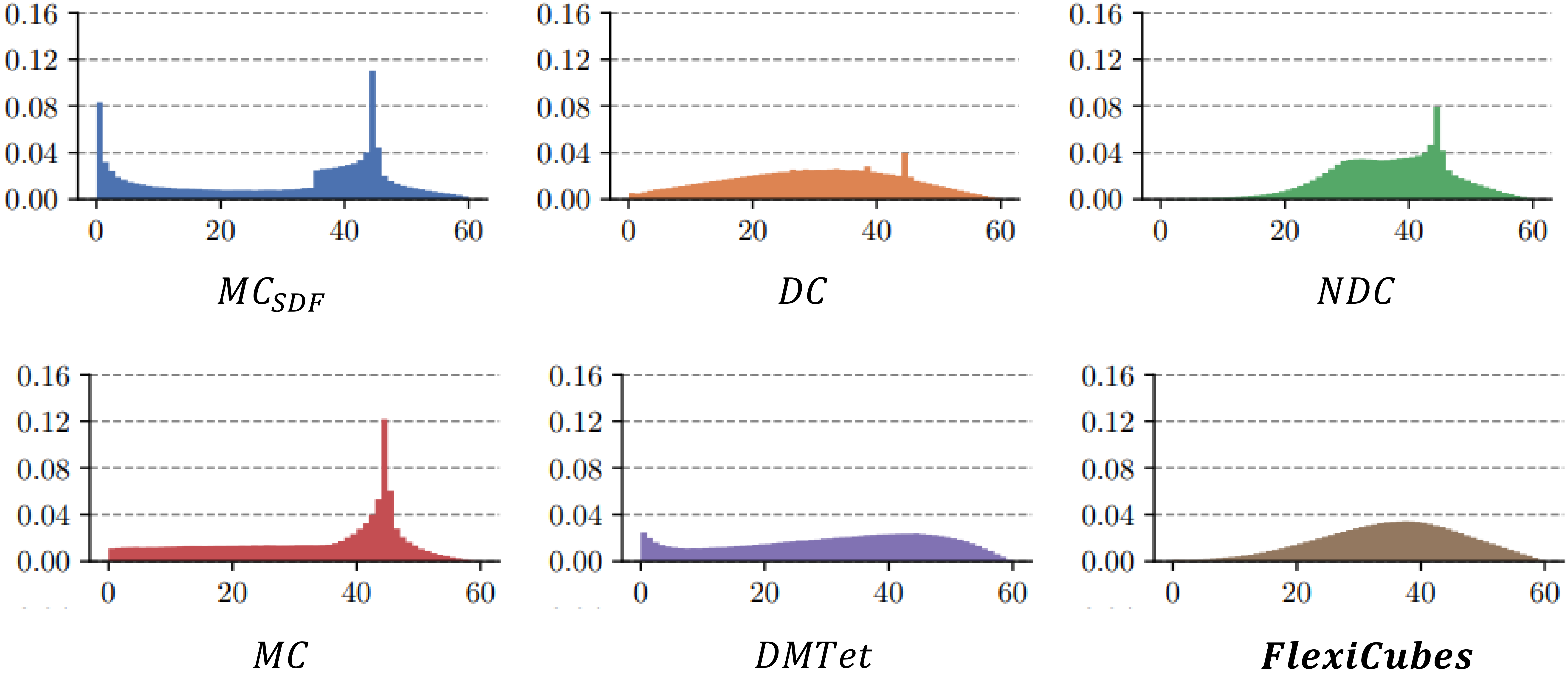
Extracting mesh from ground truth SDF: $MC_{SDF}$, $DC_{hermite}$, $NDC_{SDF}$

Reconstruct mesh via optimization: MC, DMTet, FlexiCubes

| $64^3$ | CD$(10^{-5})\downarrow$ | F1$\uparrow$ | ECD $(10^{-2})\downarrow$ | EF1$\uparrow$ |
|---|---|---|---|---|
| $MC_{SDF}$ | 6.84 | 0.55 | 2.55 | 0.14 |
| $DC_{hermite}$ | 5.90 | 0.61 | 3.80 | 0.23 |
| $NDC_{SDF}$ | 6.16 | 0.57 | 1.22 | 0.26 |
| MC | 6.33 | 0.66 | 1.25 | 0.25 |
| DMTet(64) | 7.50 | 0.66 | 3.77 | 0.28 |
| DMTet(80) | 5.17 | 0.66 | 3.59 | 0.29 |
| FlexiCubes | **4.87** | **0.70** | **0.71** | **0.43** |
| $128^3$ | CD$(10^{-5})\downarrow$ | F1$\uparrow$ | ECD $(10^{-2})\downarrow$ | EF1$\uparrow$ |
| $MC_{SDF}$ | 4.72 | 0.68 | 1.13 | 0.33 |
| $DC_{hermite}$ | 4.59 | 0.69 | 3.82 | 0.40 |
| $NDC_{SDF}$ | 5.04 | 0.65 | 0.79 | 0.43 |
| MC | 4.51 | 0.72 | 1.32 | 0.44 |
| DMTet(128) | 4.98 | 0.74 | 1.50 | 0.39 |
| FlexiCubes | **4.31** | **0.71** | **0.42** | **0.51** |

# BETTER TRIANGLE QUALITY

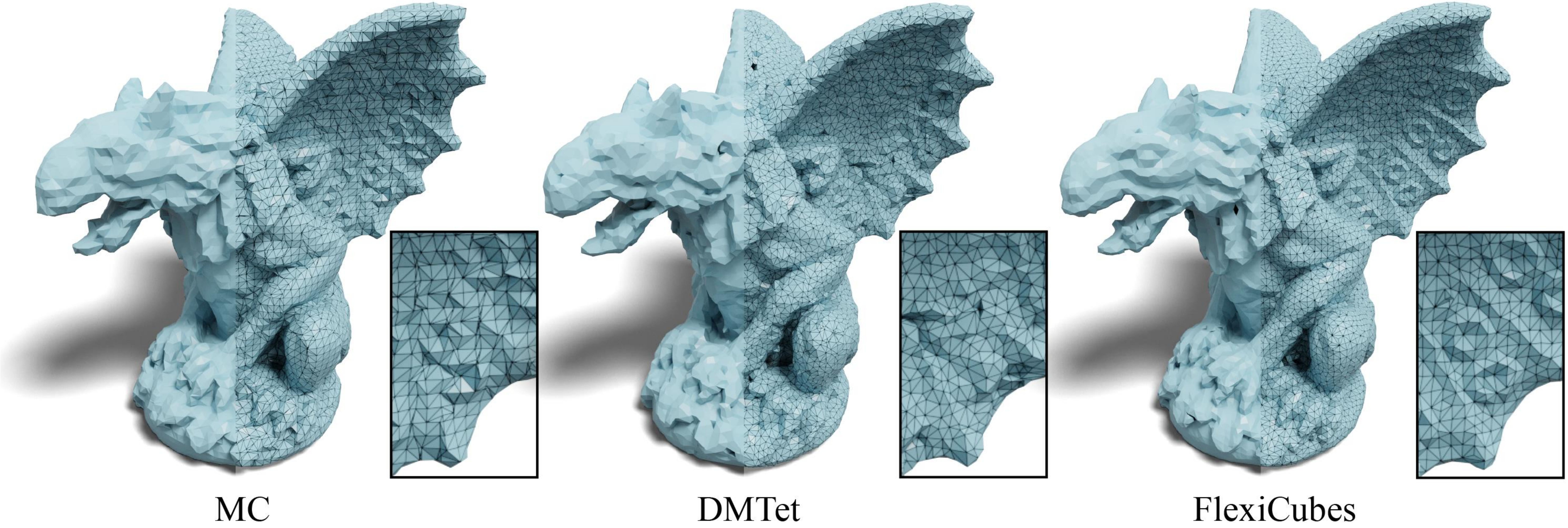
Min angle of extracted triangles.



# MESH OPTIMIZATION WITH REGULARIZATIONS

We can further improve triangle quality by adding additional regularizers.

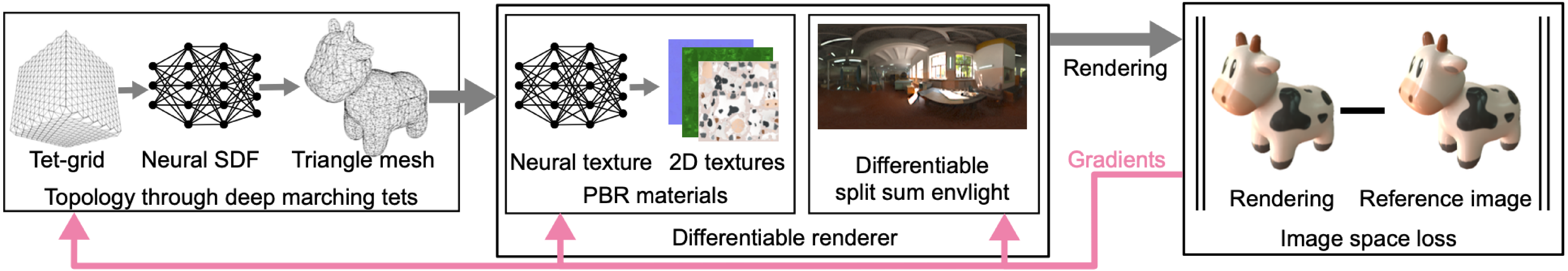
- Equilateral Edge Length



# APPLICATIONS

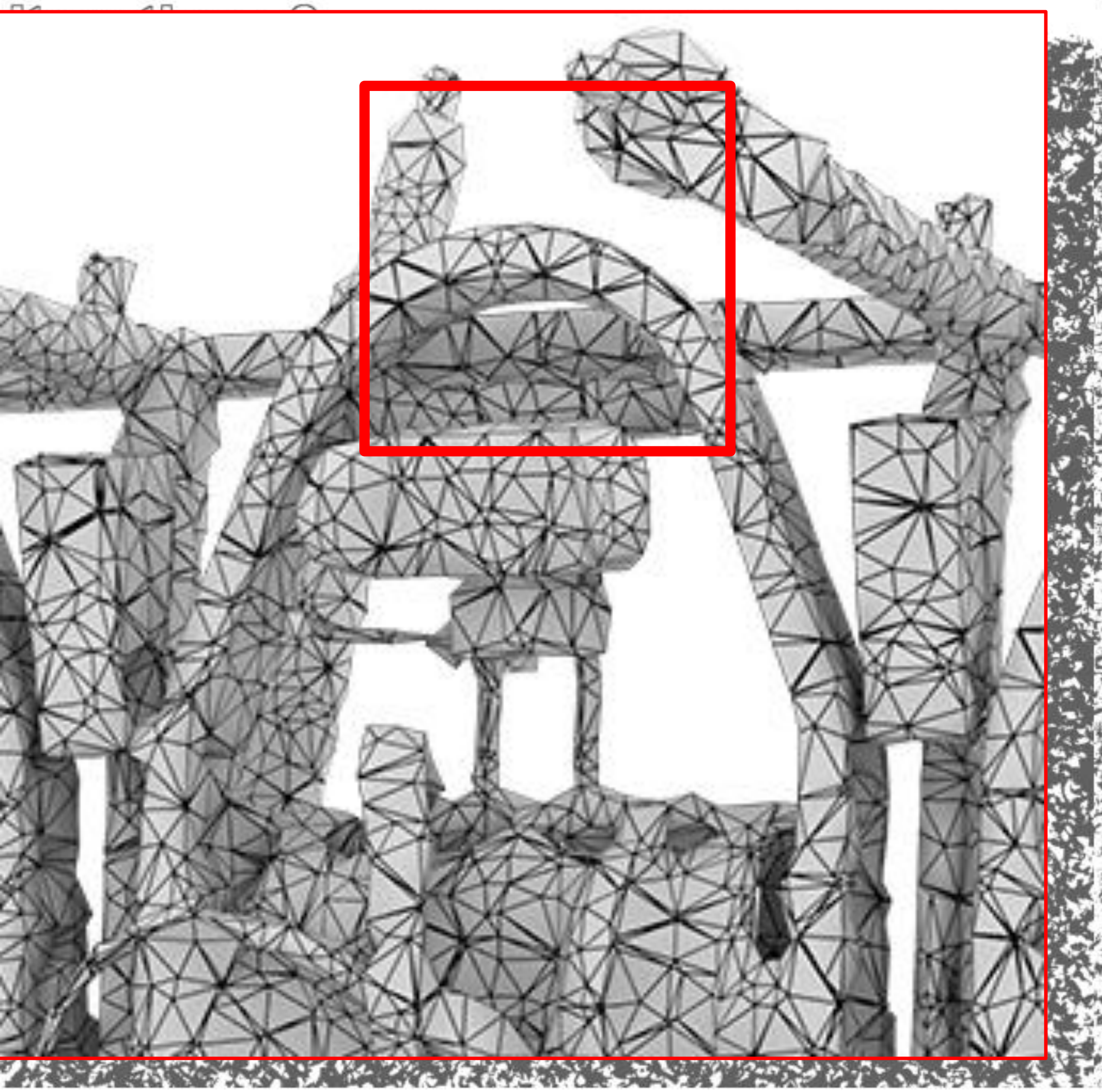
# PHOTOGRAMMETRY THROUGH DIFFERENTIABLE RENDERING

**Nvdiffrec** jointly optimizes shape, materials, and lighting from images.

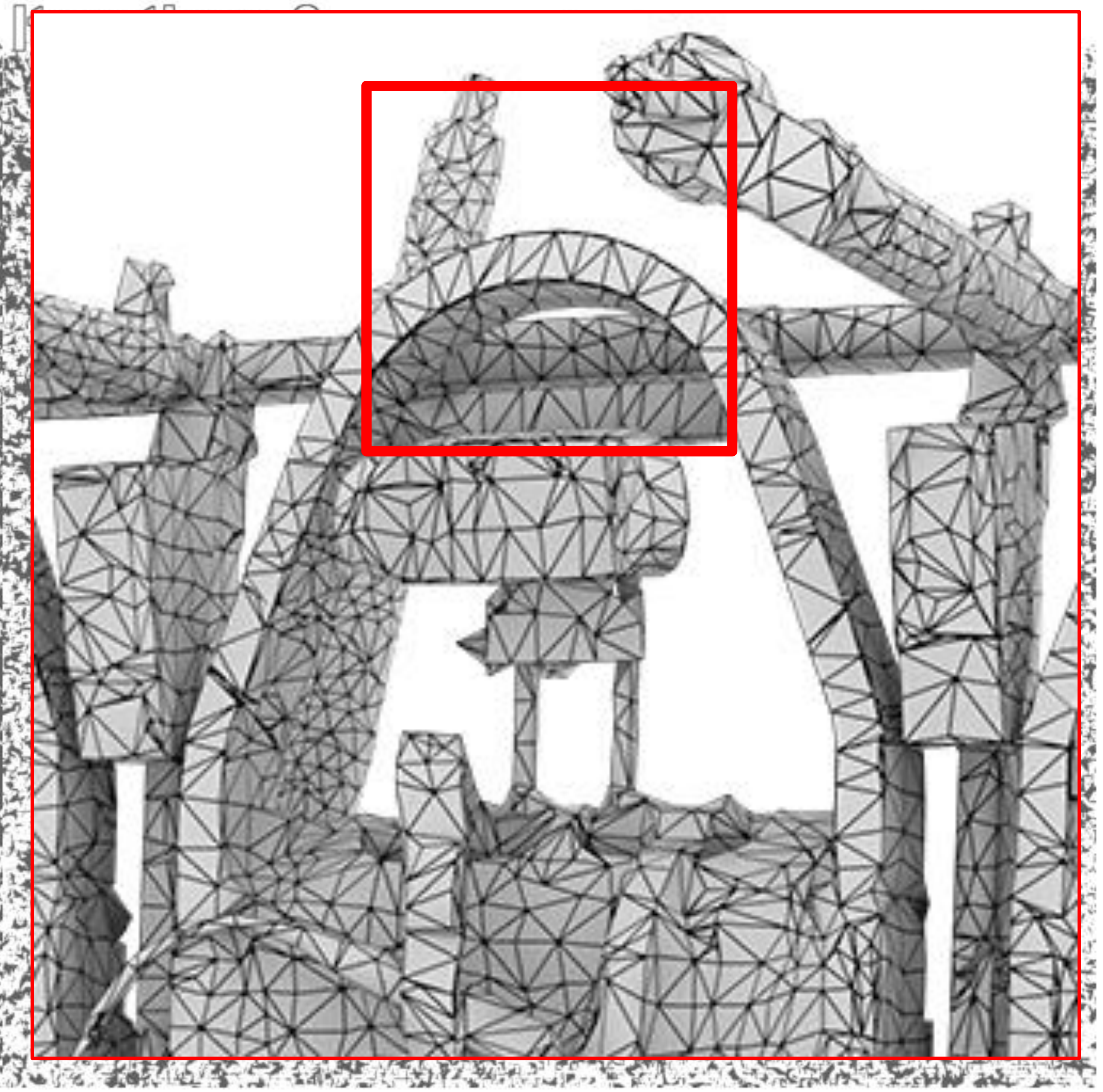


# PHOTOGRAMMETRY THROUGH DIFFERENTIABLE RENDERING

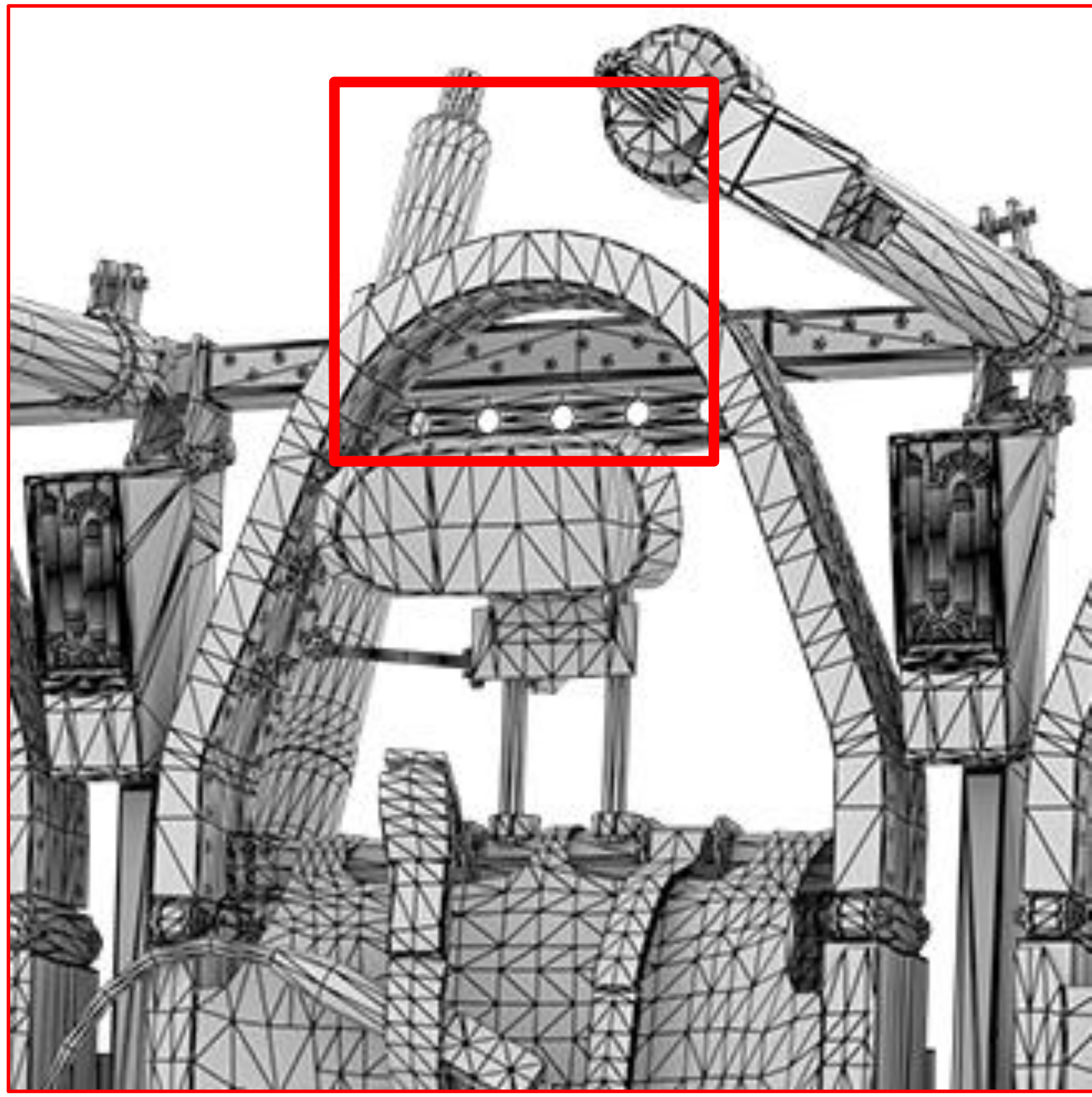
FlexiCubes improves geometric fidelity and mesh quality.



Nvdiffrec w/ DMTet



Nvdiffrec w/ FlexiCubes

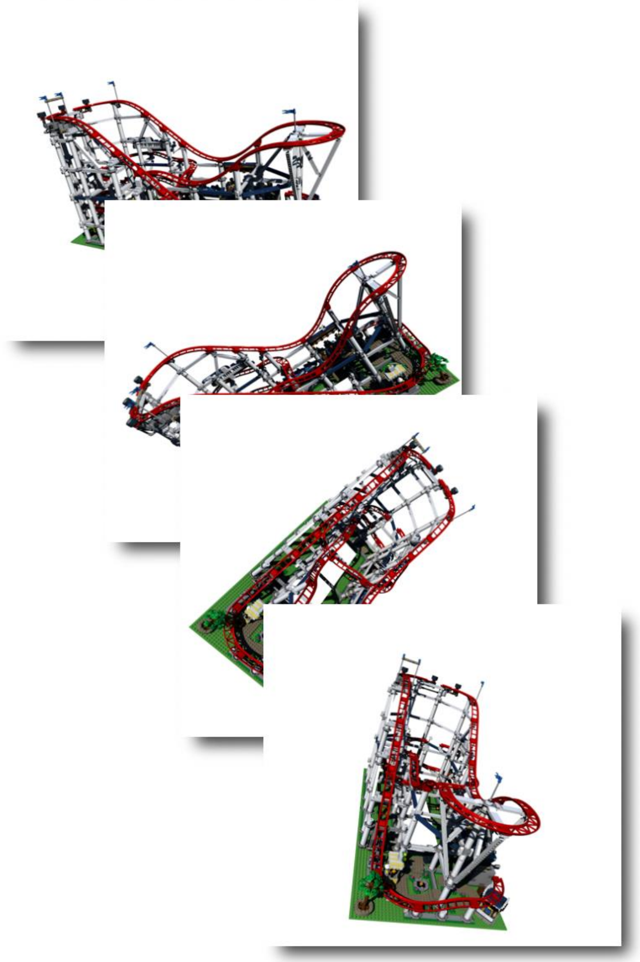


Reference

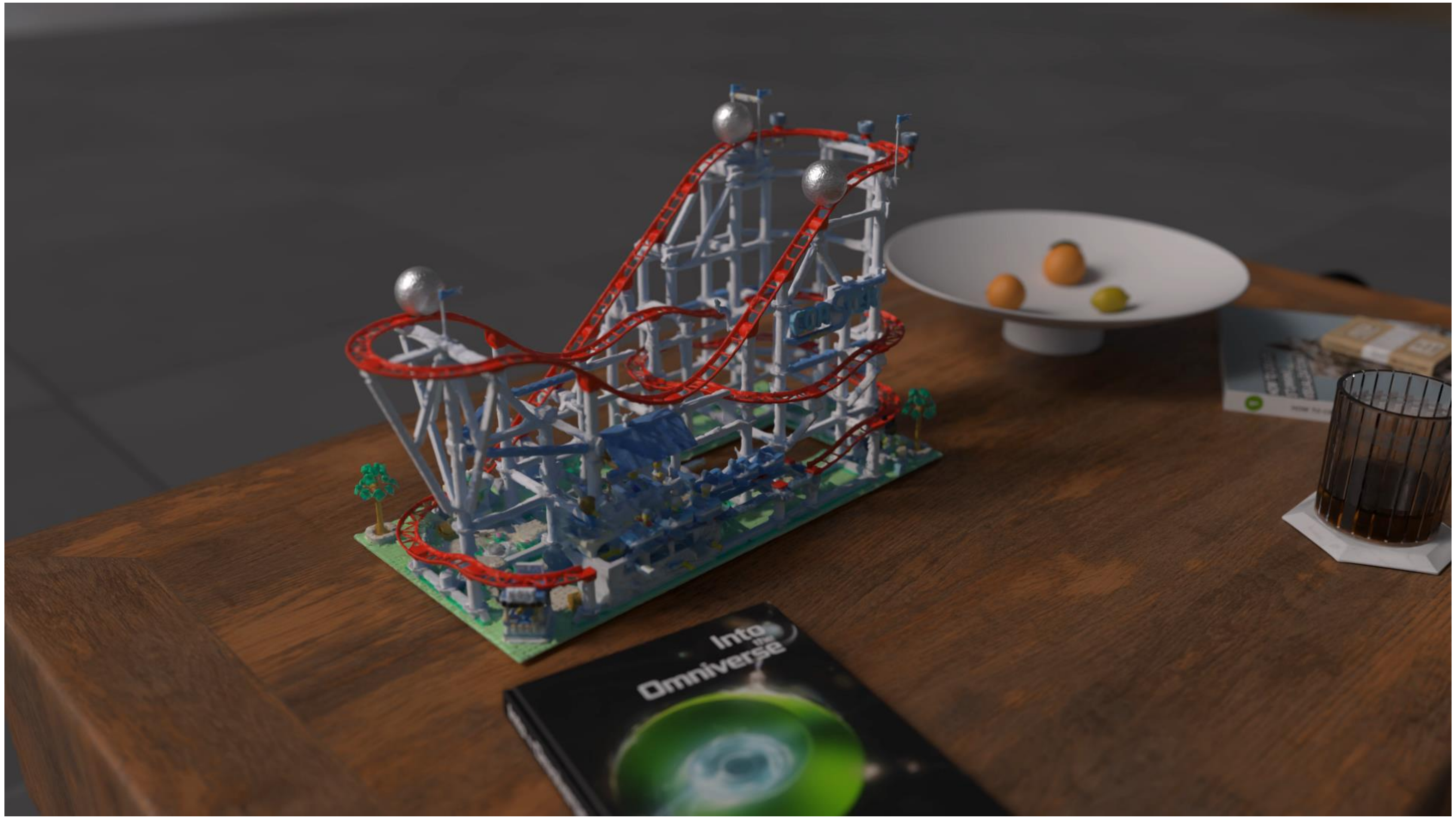
**Extracting Triangular 3D Models, Materials, and Lighting From Images**
Munkberg et. al. CVPR 2022

# PHOTOGRAMMETRY THROUGH DIFFERENTIABLE RENDERING

The outputs are compatible with standard graphics workflow.



Multiview inputs




Simulating reconstructed asset with physics in Omniverse

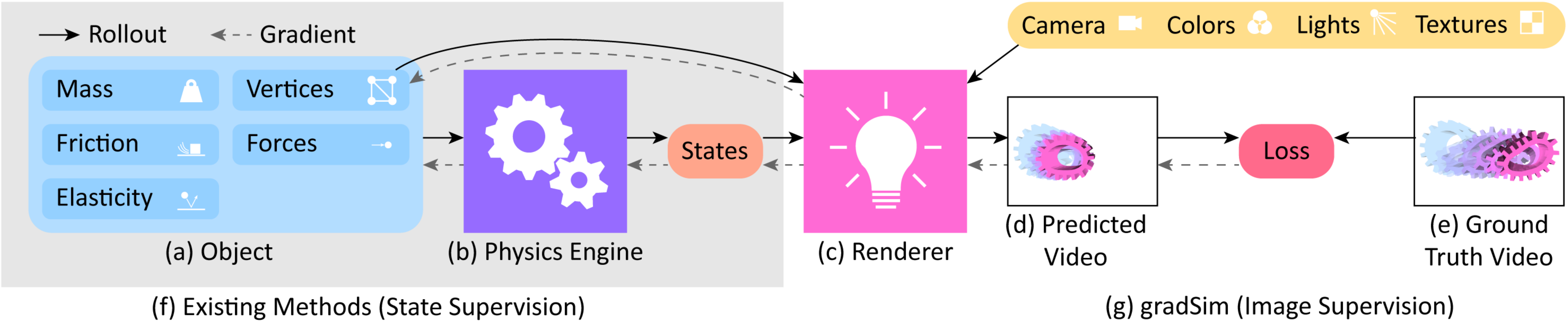
# MESH SIMPLIFICATION OF ANIMATED OBJECTS

End-to-end optimization w/ **FlexiCubes** avoids mesh stretching.

**End-to-end** optimization | Reference | T-pose optimization

**Appearance-Driven Automatic 3D Model Simplification**
Hasselgren et. al. Eurographics Symposium on Rendering. 2021

# DIFFERENTIABLE PHYSICS SIMULATION WITH TET MESH



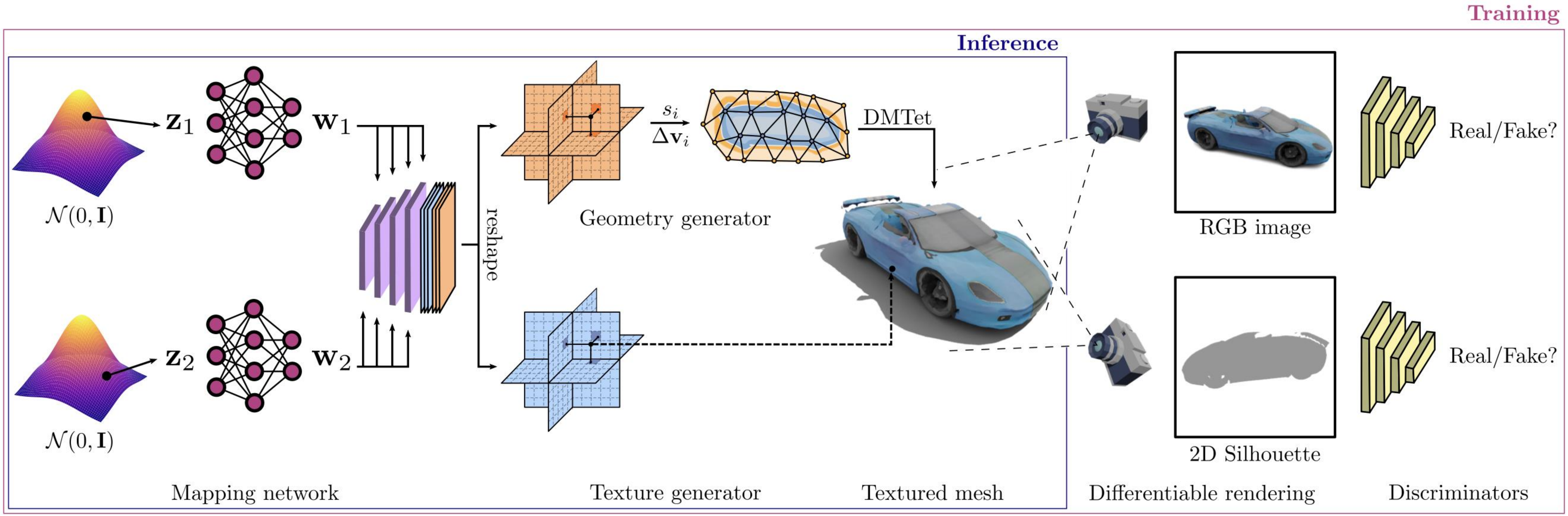

**gradSim: Differentiable simulation for system identification and visuomotor control**
Jatavallabhula and Macklin et. al. ICLR 2021

# DIFFERENTIABLE PHYSICS SIMULATION WITH TET MESH

Initialization | Optimized Results w/ **FlexiCubes** | Reference

**gradSim: Differentiable simulation for system identification and visuomotor control**
Jatavallabhula and Macklin et. al. ICLR 2021

# 3D GENERATIVE MODELING FOR MESHES W/ GET3D




**GET3D: A Generative Model of High Quality 3D Textured Shapes Learned from Images**
Gao, Shen, Wang, Chen, Yin, Li, Litany, Gojcic, Fidler, NeurIPS 2022

# 3D GENERATIVE MODELING FOR MESHES W/ GET3D

GET3D w/ FlexiCubes generates meshes with better details and tessellation.

DMTet

FlexiCubes

Motorbike

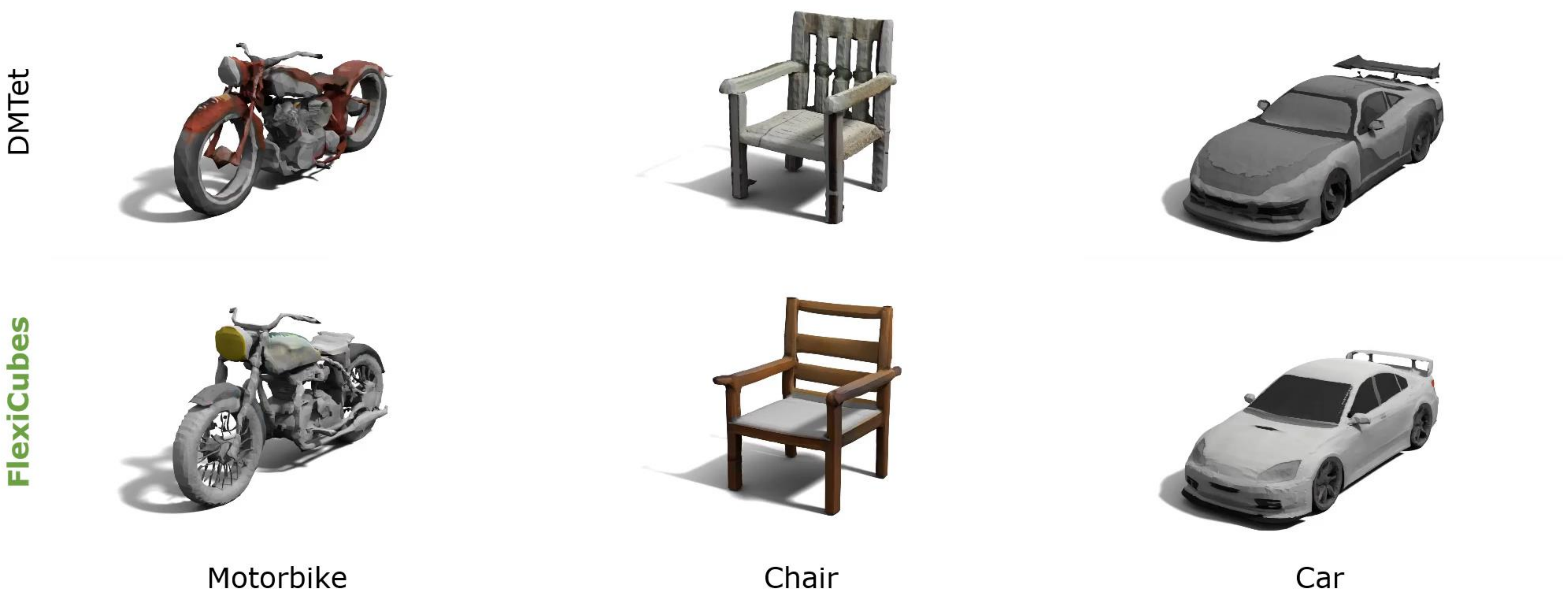
Chair

Car

**GET3D: A Generative Model of High Quality 3D Textured Shapes Learned from Images**
Gao, Shen, Wang, Chen, Yin, Li, Litany, Gojcic, Fidler, NeurIPS 2022

# 3D GENERATIVE MODELING FOR MESHES W/ GET3D

GET3D w/ **FlexiCubes** generates meshes with better details and tessellation.




**GET3D: A Generative Model of High Quality 3D Textured Shapes Learned from Images**
Gao, Shen, Wang, Chen, Yin, Li, Litany, Gojcic, Fidler, NeurIPS 2022

# ADDING MESH REGULARIZATIONS

**FlexiCubes** supports optimizing regularizations defined on meshes.

Marching Cubes

FlexiCubes

**Developability of triangle meshes**
Stein et. al. SIGGRAPH 2018

# PERFORMANCE

Performance of Isosurfacing operations

| $128^3$ | Forward Time (ms) | Backward Time (ms) | Memory (MB) |
|---|---|---|---|
| MC | 5.08 | 0.58 | 72.85 |
| DMTet | 6.94 | 1.39 | 168.27 |
| $DMC_{centroid}$ [Nielson 2004] | 7.34 | 1.74 | 150.75 |
| FlexiCubes | 14.06 | 9.53 | 816.17 |

Performance of various application (1 iteration)

| Applications ($96^3$) | NVDIFFRECMC | | GET3D | |
|---|---|---|---|---|
| Isosurface | DMTet | FlexiCubes | DMTet | FlexiCubes |
| Time per iter. (ms) | 307 | 315 | 510 | 610 |
| Memory(GiB) | 13.1 | 15.3 | 11.6 | 11.1 |

# LIMITATIONS AND FUTURE WORK

Limitations:

- Self-intersections
- Weaker guarantees of topological correctness in adaptive and tetrahedral meshing.

Future work:

- Integrate volumetric rendering with mesh-based representation.
- Extend to 4D spatiotemporal meshing.
- Integrate adaptive hierarchical meshing into generative modeling pipelines.

# THANK YOU FOR LISTENING!

*Takeaways:*

- FlexiCubes is designed for gradient-based mesh optimization.
- Incorporate additional DoFs into mesh extraction.
- Drop-in replacement for better mesh quality and geometric fidelity!

Populate your 3D world with assets generated w/ FlexiCubes!

Visit our project page to explore additional results and learn more details!