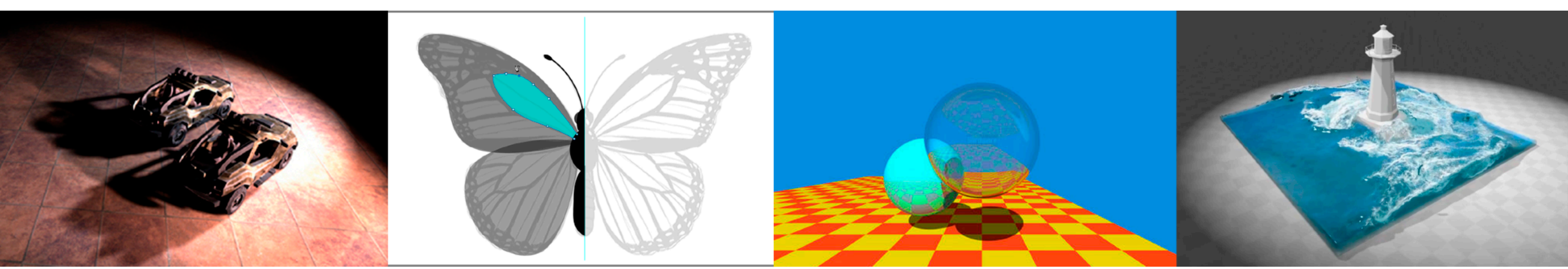
Introduction to Computer Graphics

GAMES101, Lingqi Yan, UC Santa Barbara

Lecture 12:

Geometry 3



http://www.cs.ucsb.edu/~lingqi/teaching/games101.html

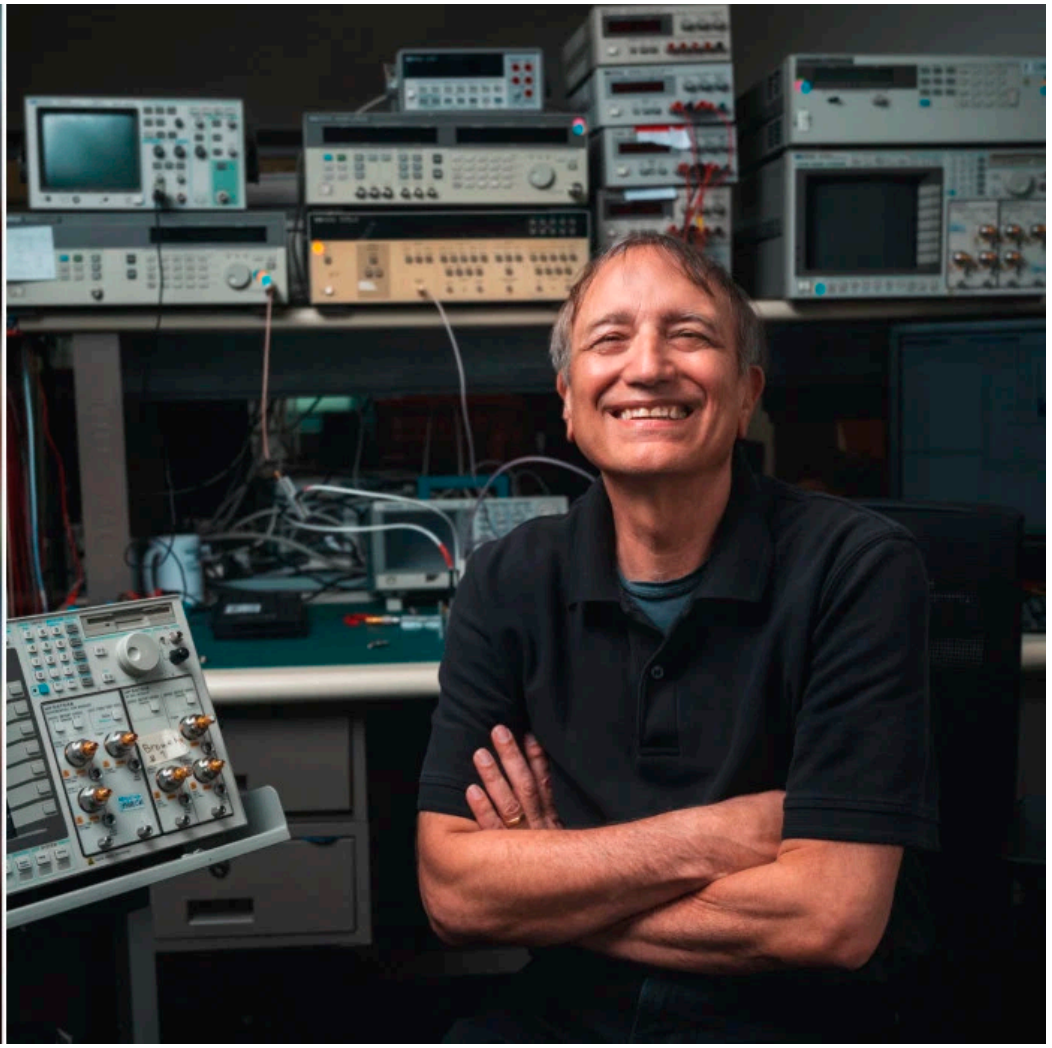
Announcements

- Homeworks
  - Enjoying HW3?
  - HW1 submission window reopened (similar policy applies to later HWs)
- The T/N/B calculation
  - Will be in the next lectures [local shading frame]
- BIG NEWS!
  - Computer Graphics won the Turing Award after 32 years!

Turing Award Winners

- Made Computer Graphics great
- We will soon learn about their work!

Ed Catmull

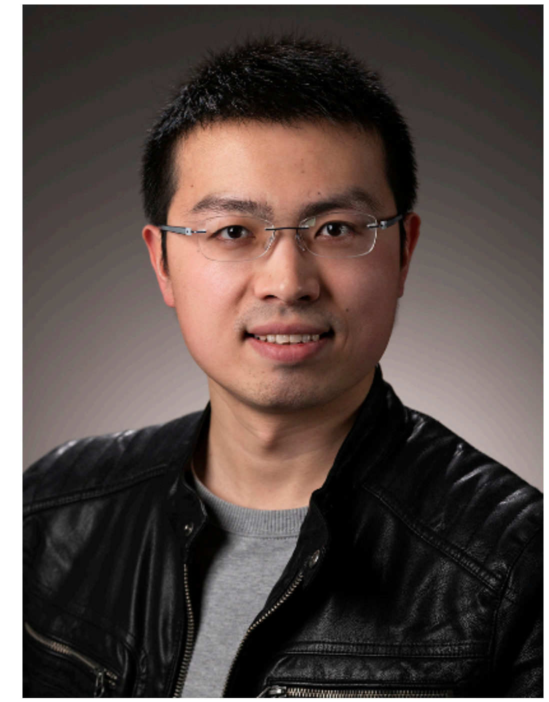


Pat Hanrahan

Academic Family Tree

Pat Hanrahan @ Stanford

Ravi Ramamoorthi @ UCSD



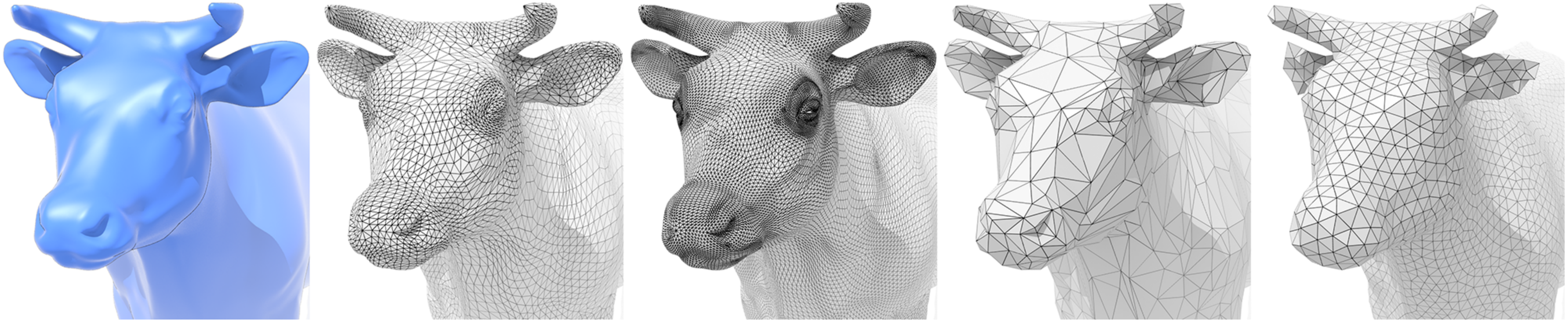
Lingqi Yan **@ UCSB**

Pradeep Sen **@ UCSB**

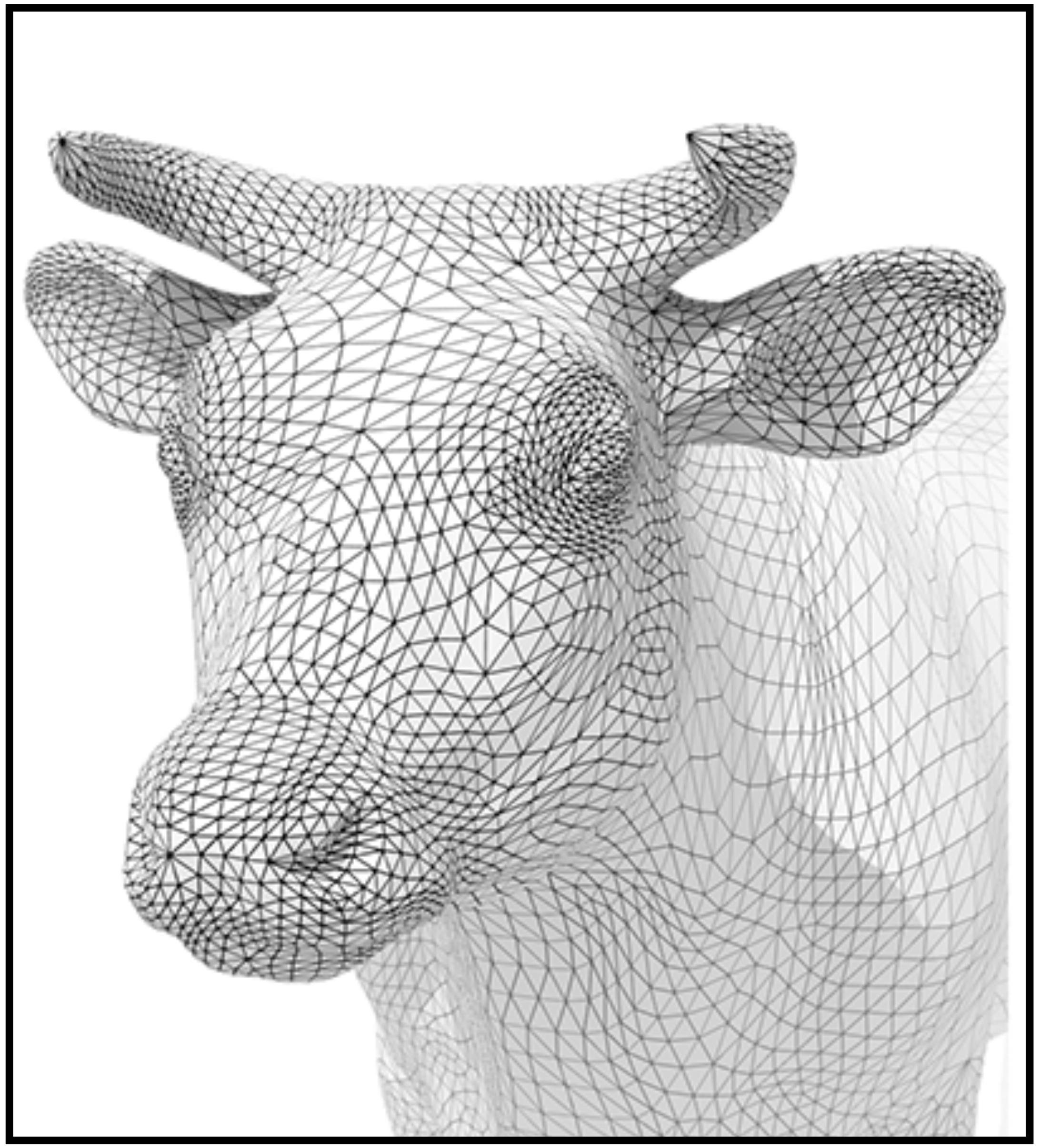
Back to Geometry

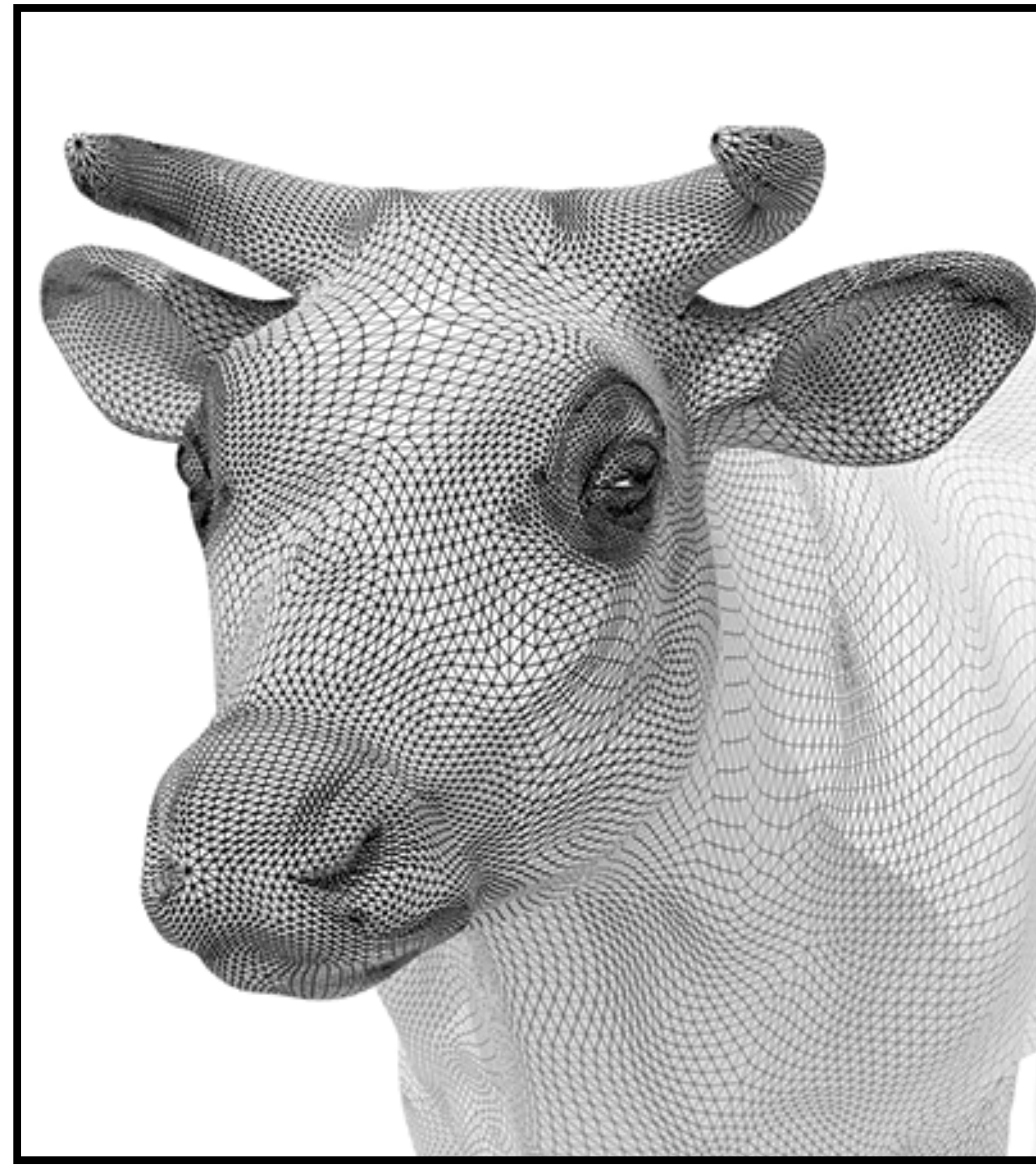
Mesh Operations: Geometry Processing

- Mesh subdivision
- Mesh simplification
- Mesh regularization



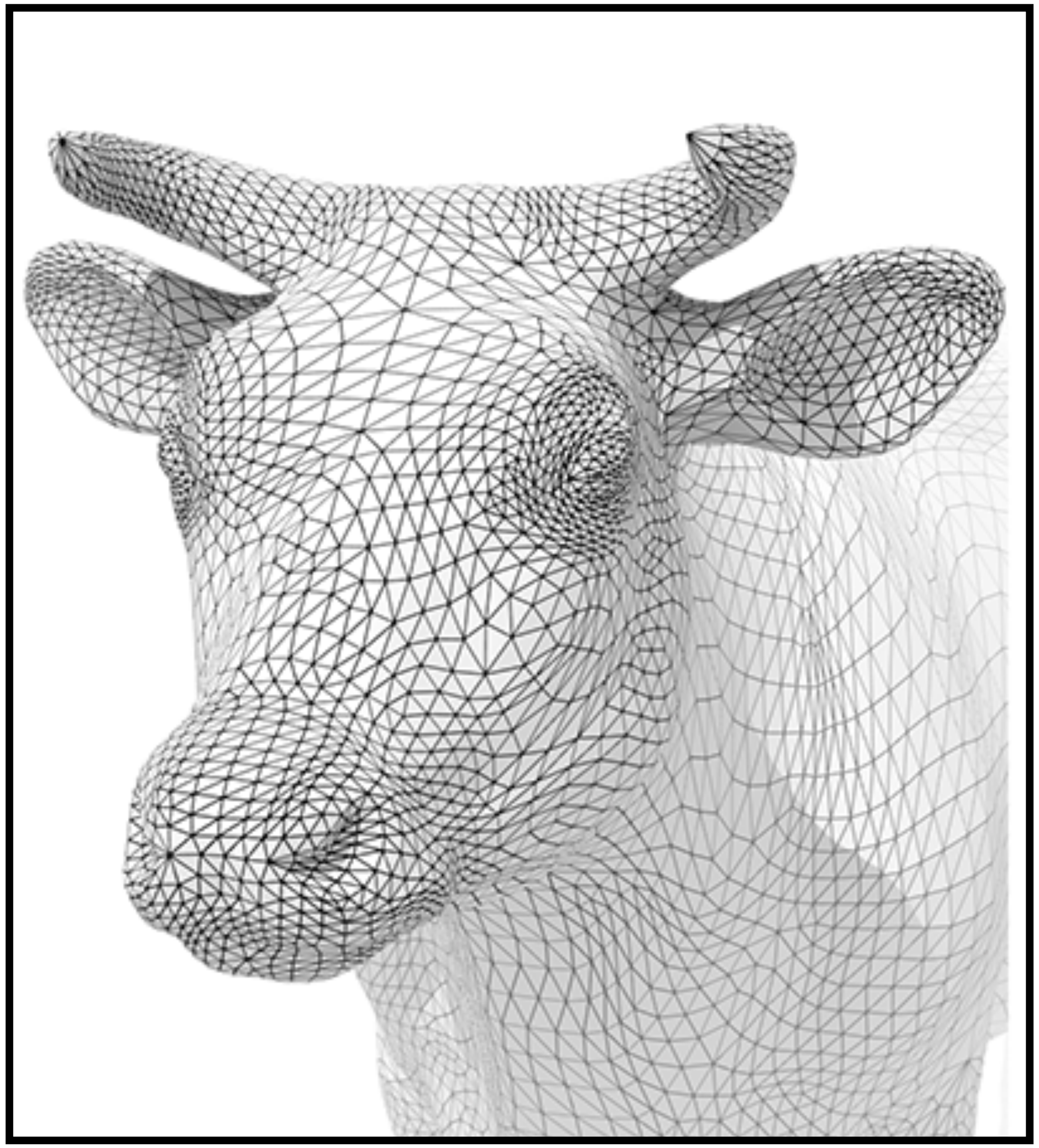
Mesh Subdivision (upsampling)

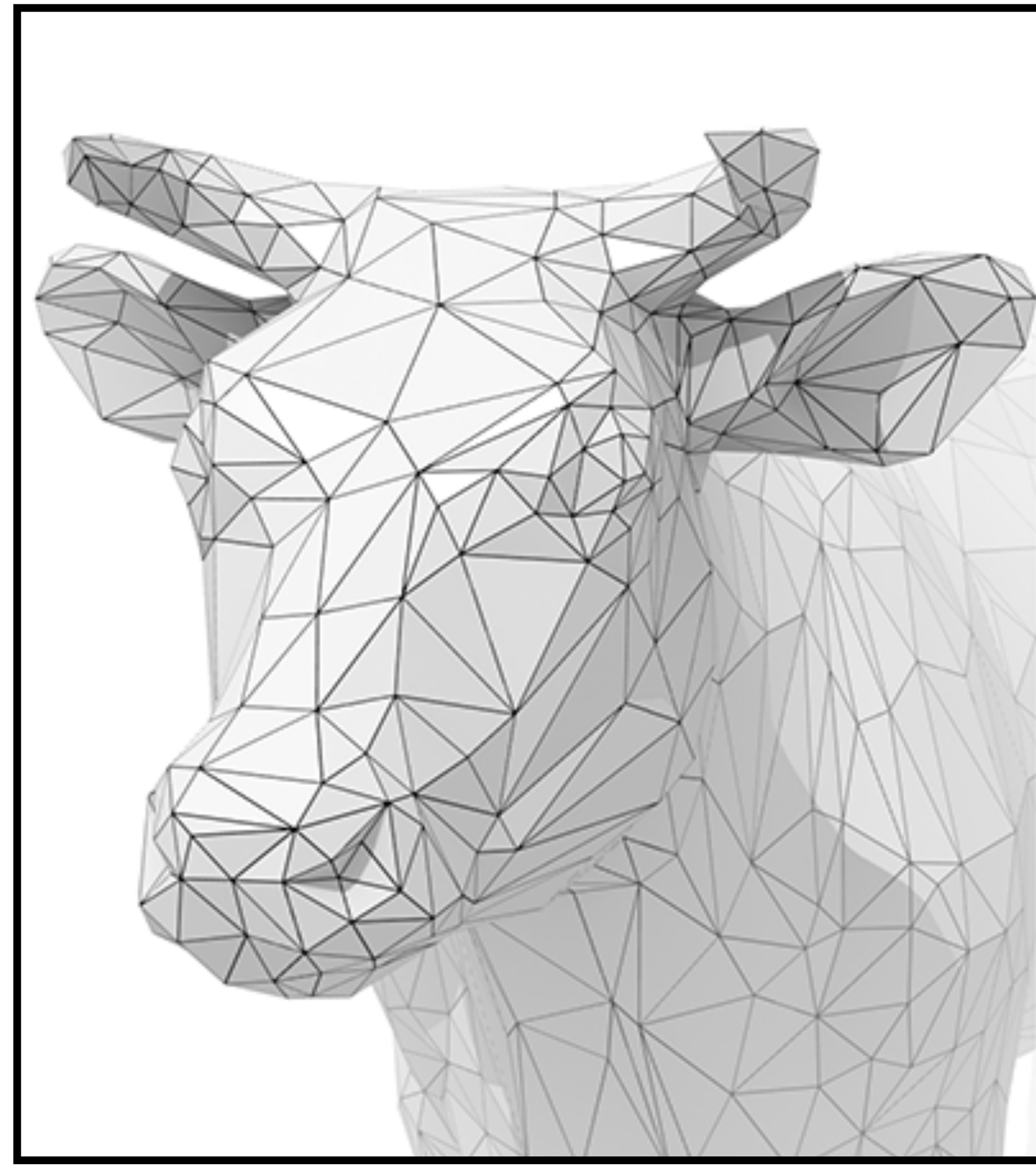




Increase resolution

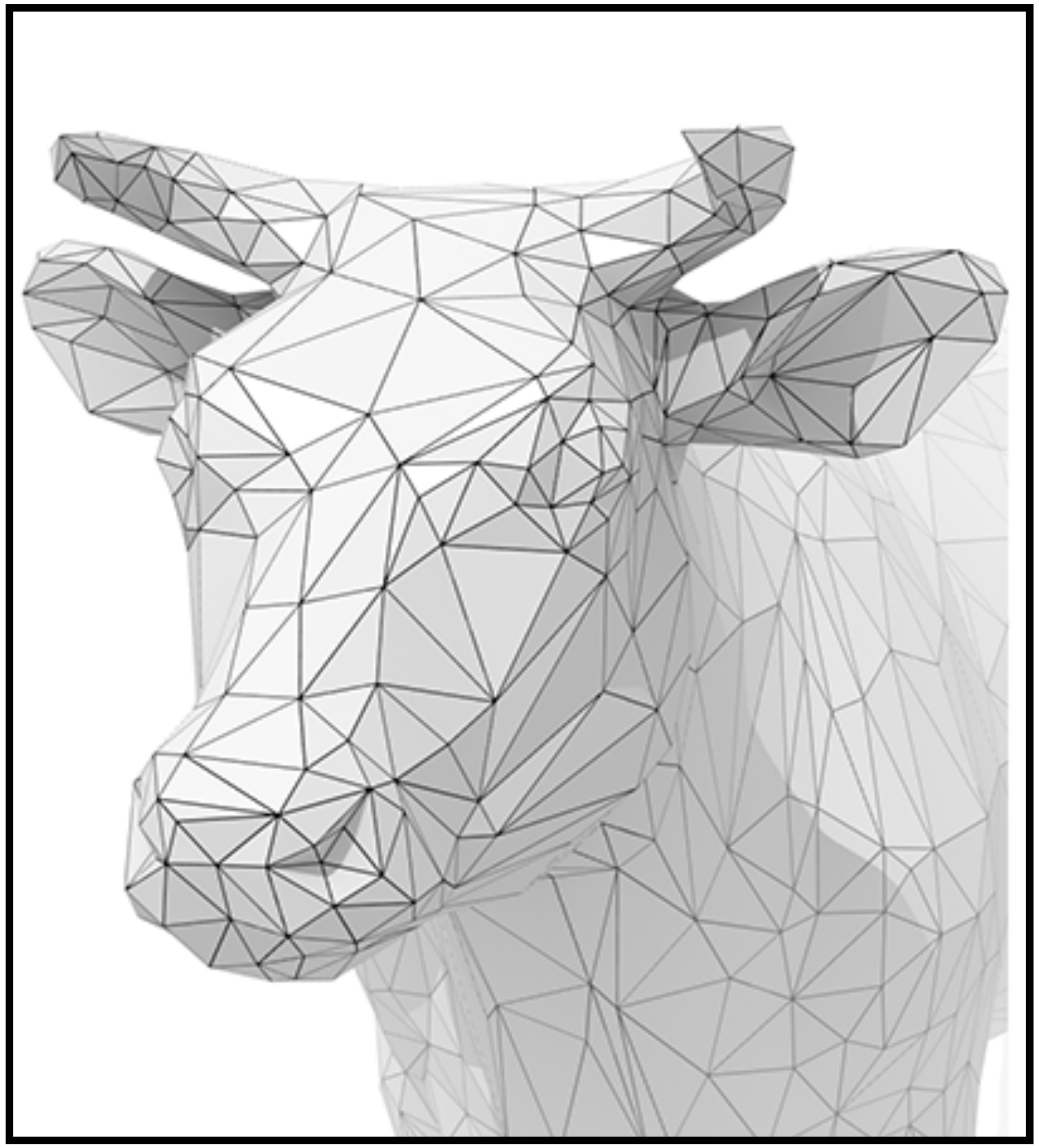
Mesh Simplification (downsampling)

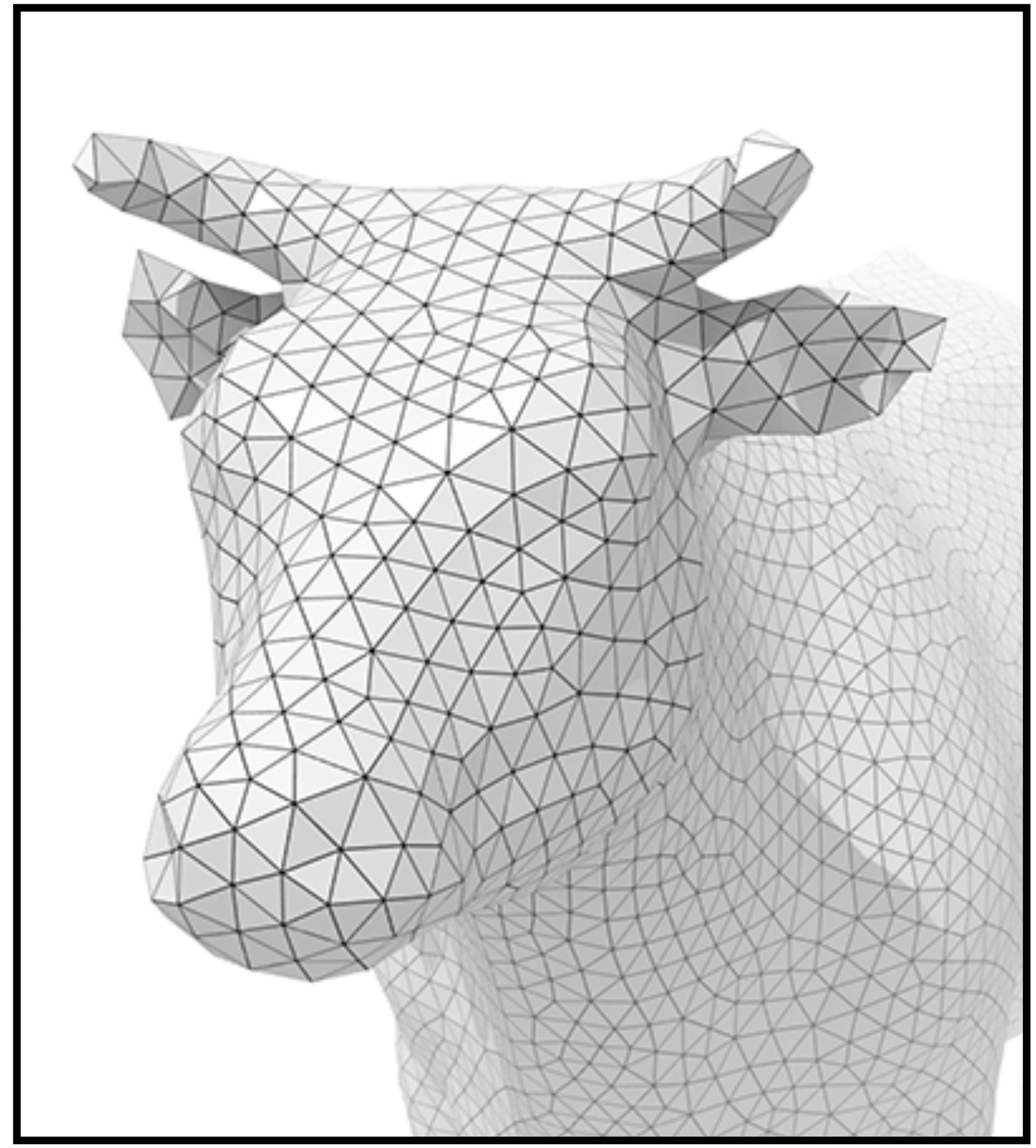




Decrease resolution; try to preserve shape/appearance

Mesh Regularization (same #triangles)





Modify sample distribution to **improve quality**

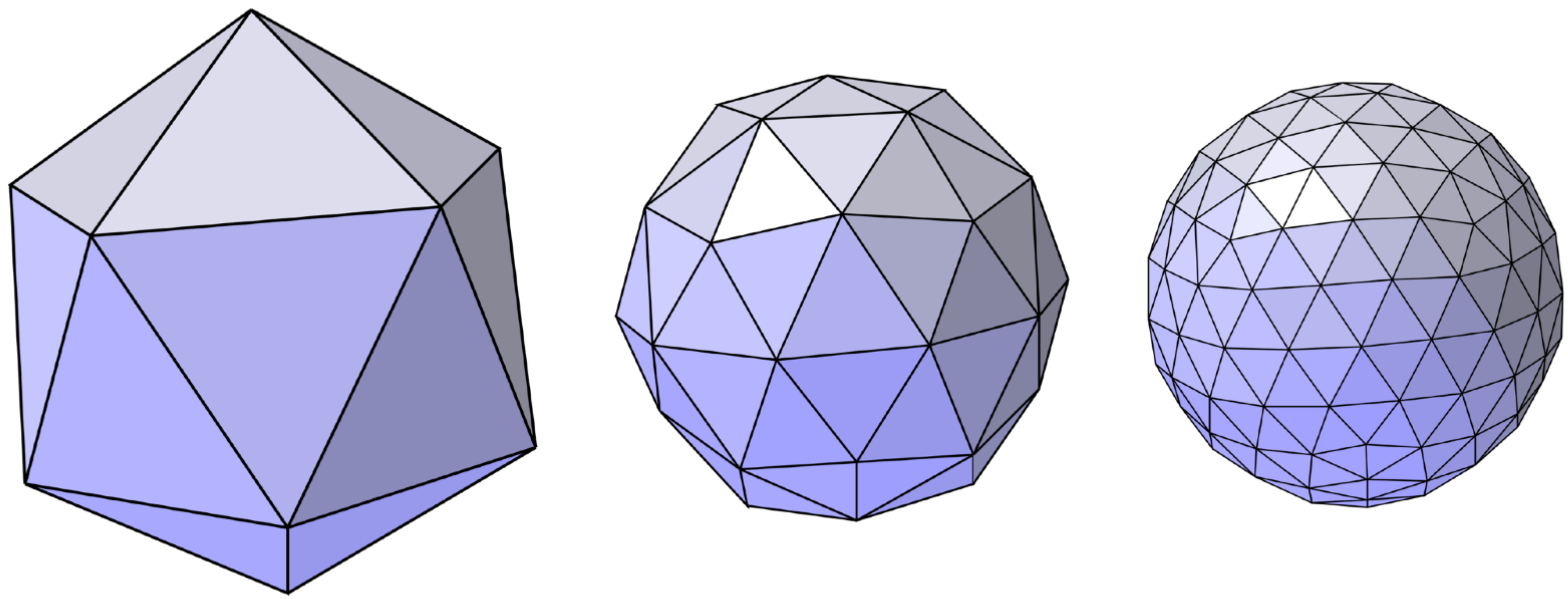
Subdivision

Loop Subdivision

Common subdivision rule for triangle meshes

First, create more triangles (vertices)

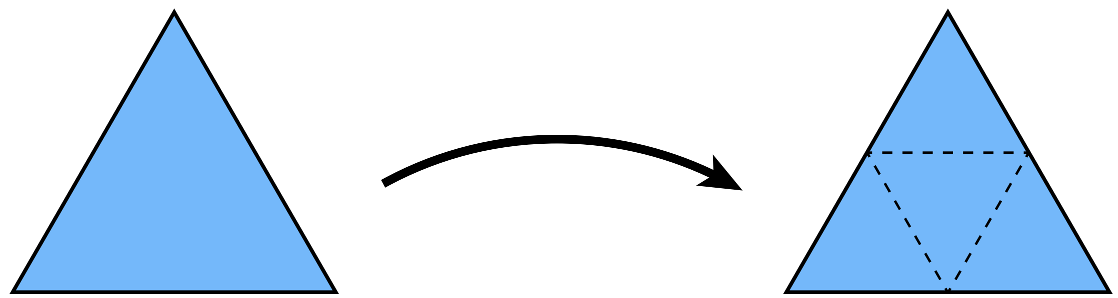
Second, tune their positions



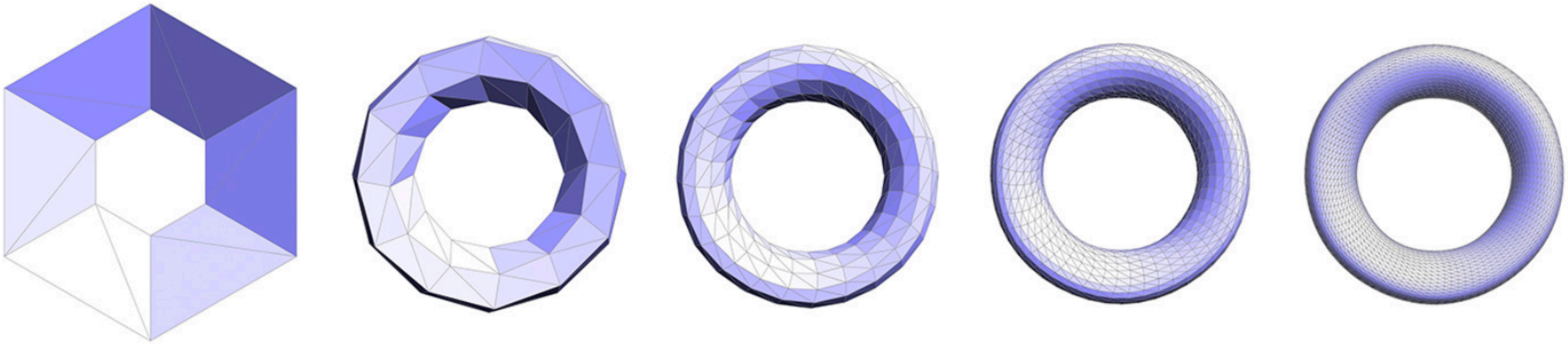
Simon Fuhrman

Loop Subdivision

- Split each triangle into four

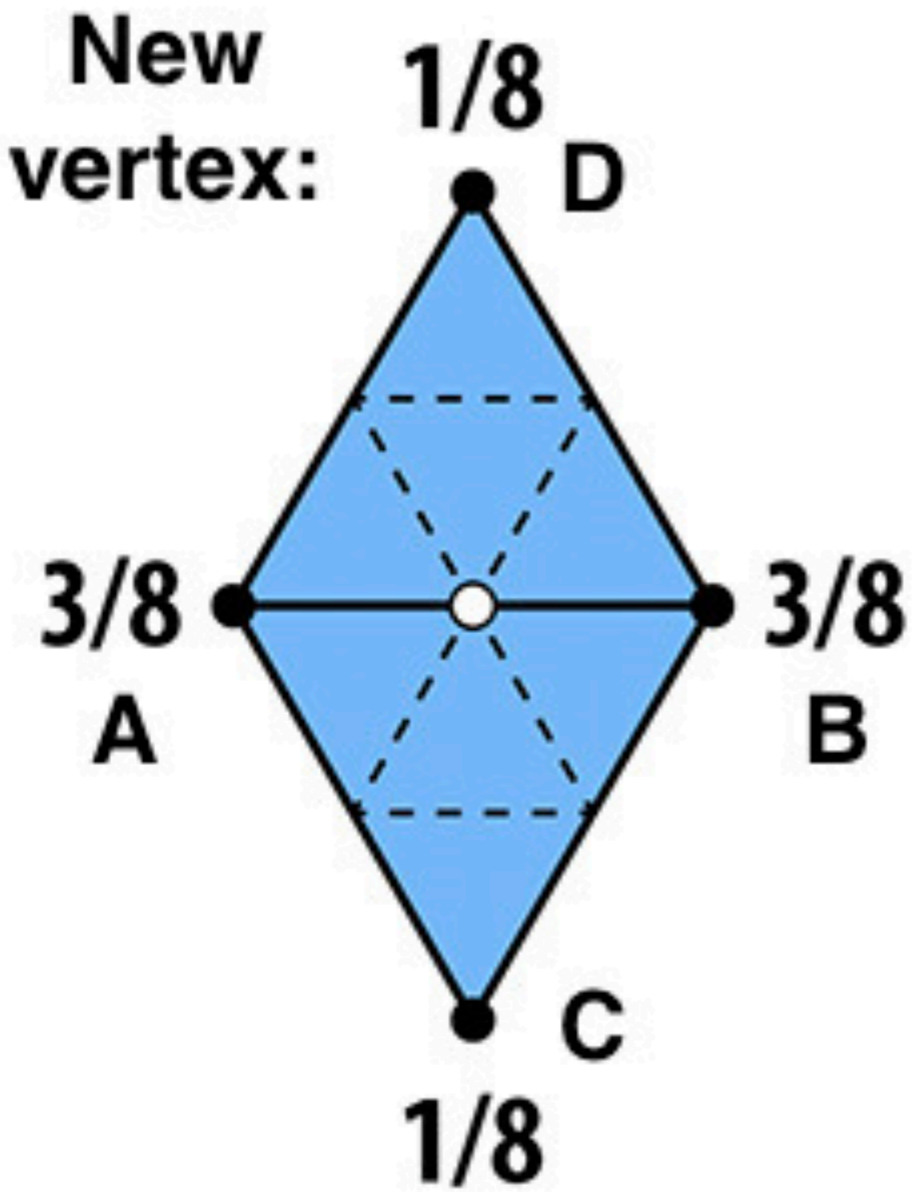


- Assign new vertex positions according to weights
  - New / old vertices updated differently



Loop Subdivision — Update

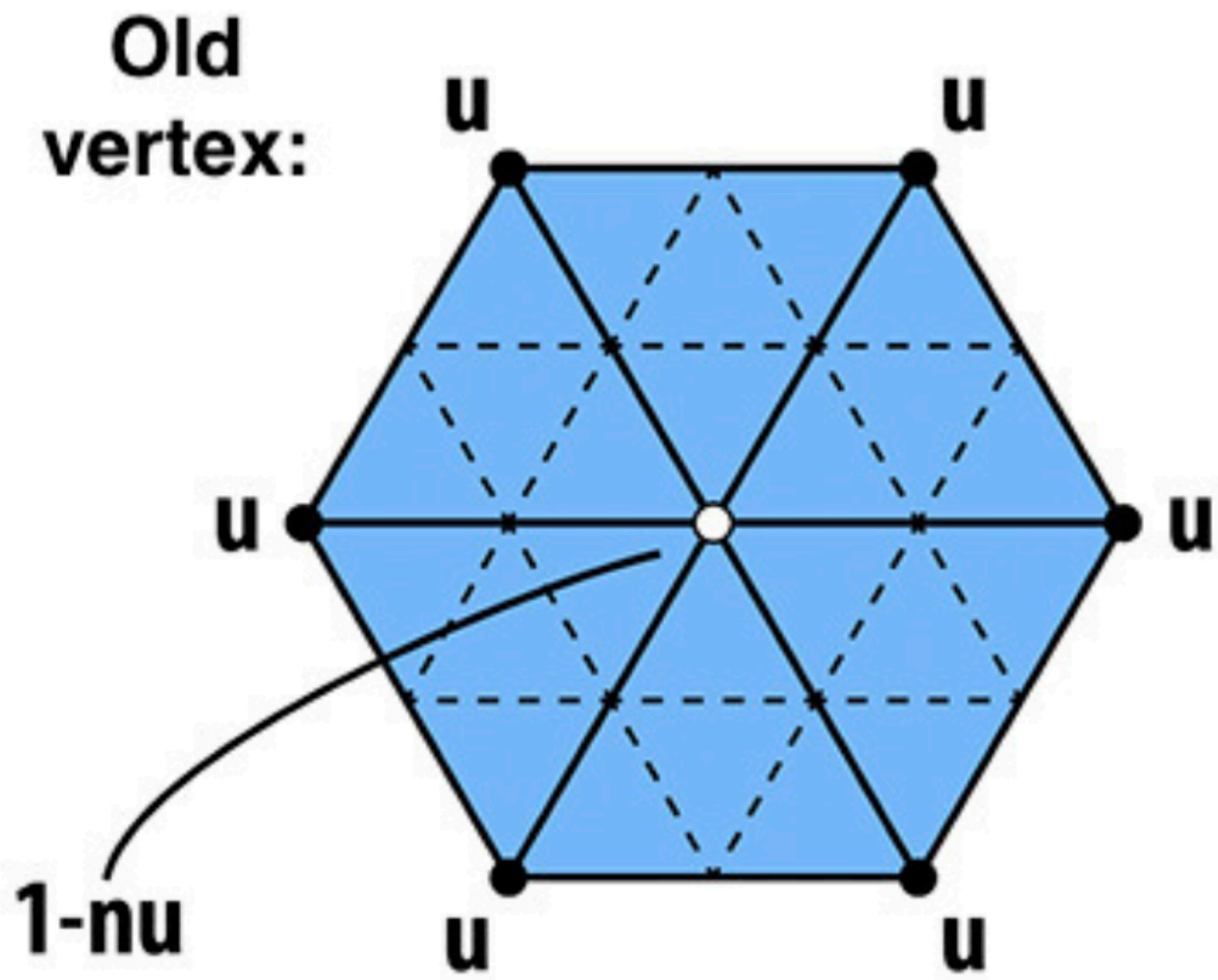
For new vertices:



Update to:
3/8 * (A + B) + 1/8 * (C + D)

Loop Subdivision — Update

For old vertices (e.g. degree 6 vertices here):

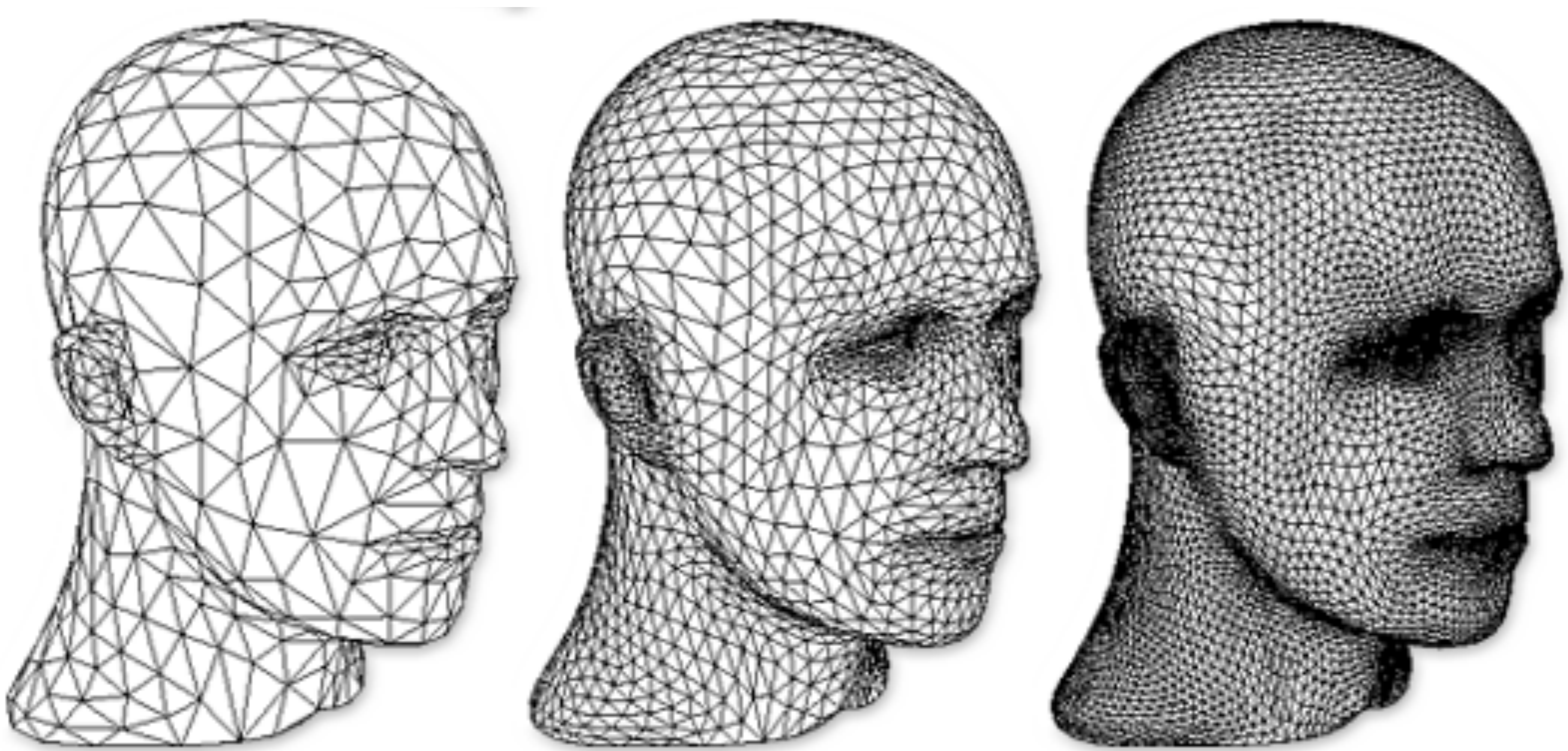


Update to:
(1 - n*u) * original_position +
u * neighbor_position_sum

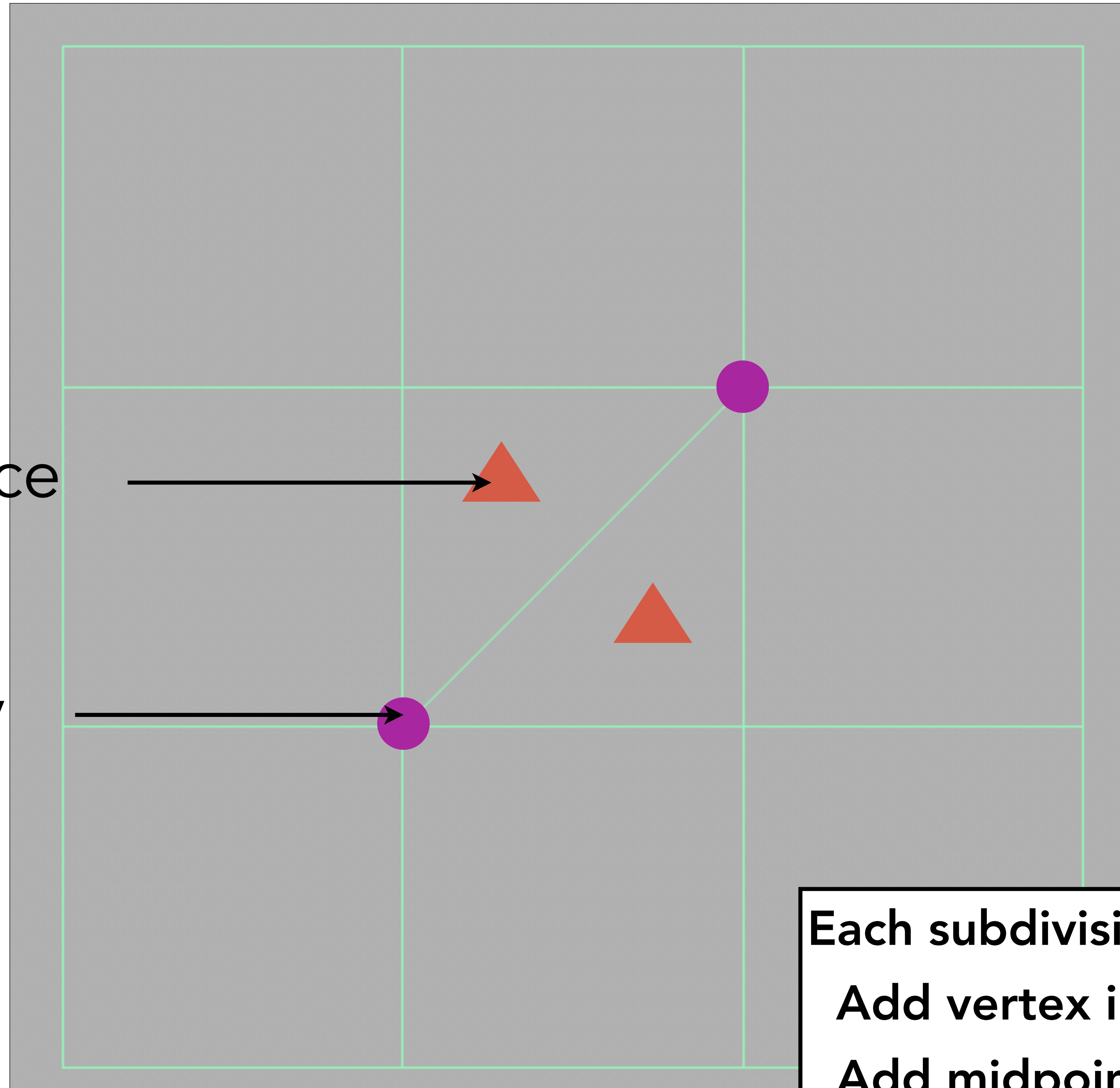
n: vertex degree

u: 3/16 if n=3, 3/(8n) otherwise

Loop Subdivision Results



Catmull-Clark Subdivision (General Mesh)



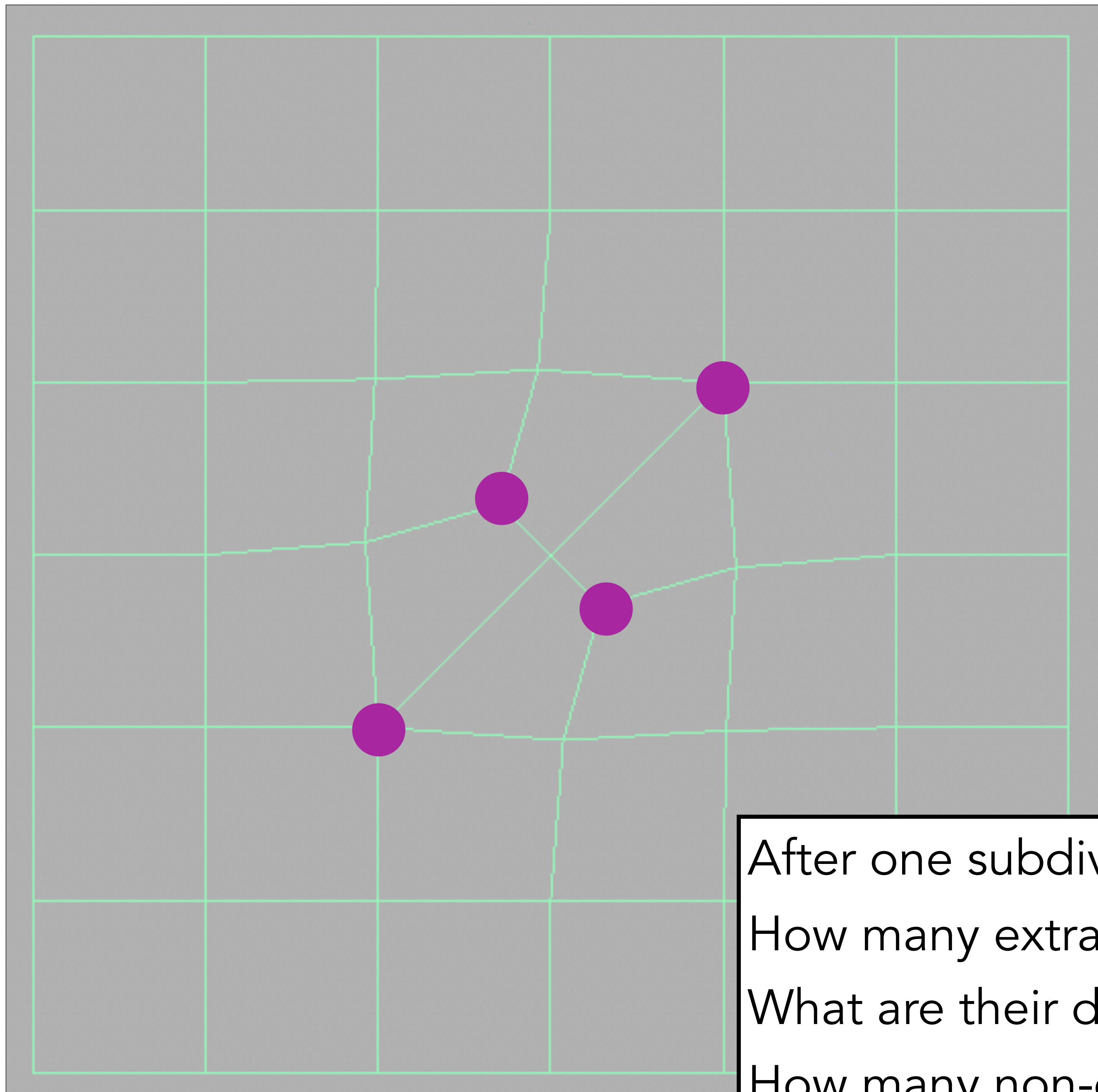
Non-quad face

Extraordinary vertex (奇异点) (degree != 4)

Each subdivision step:

- **Add vertex in each face**
- **Add midpoint on each edge**
- **Connect all new vertices**

Catmull-Clark Subdivision (General Mesh)



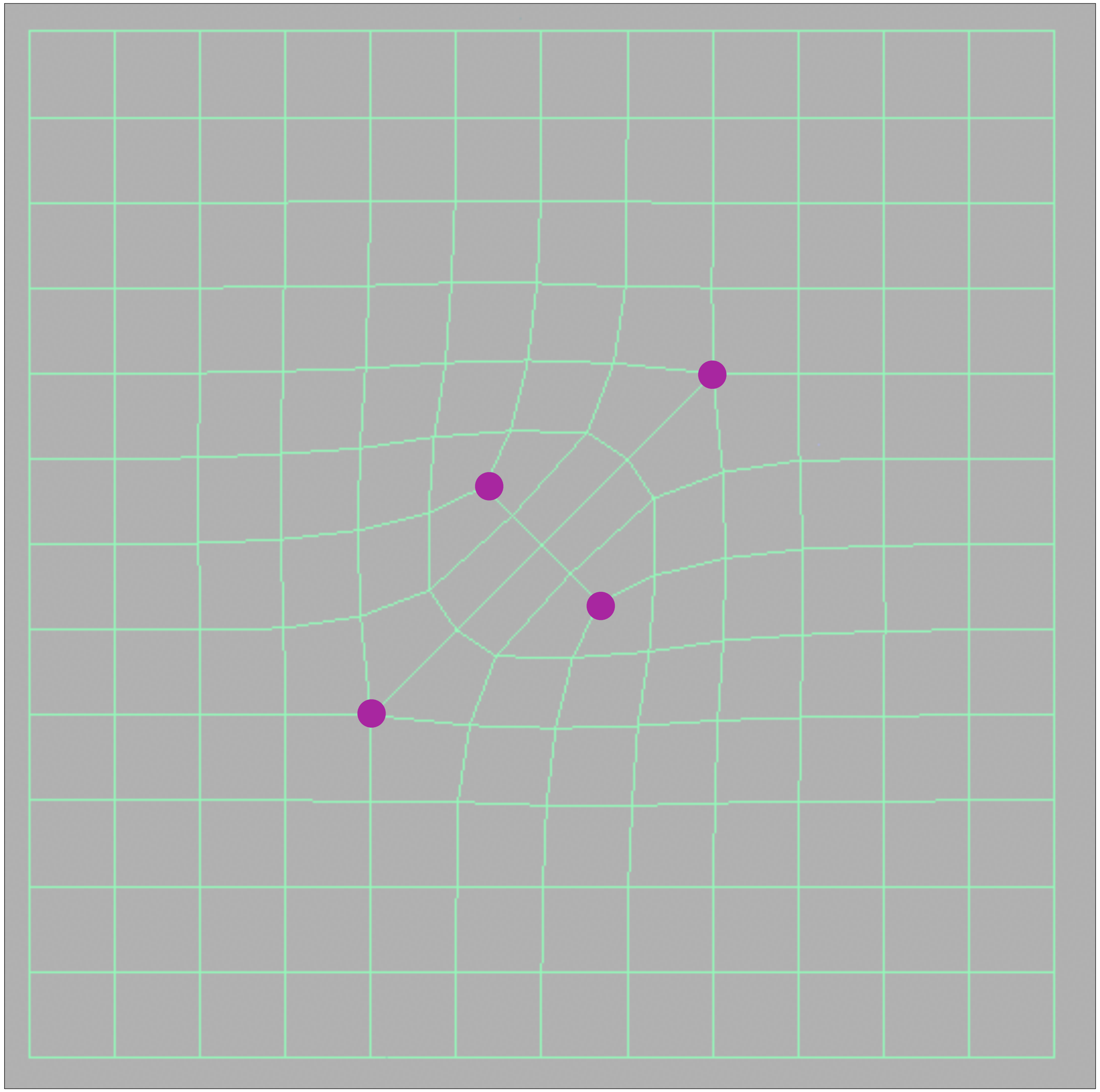
After one subdivision:

How many extraordinary vertices?

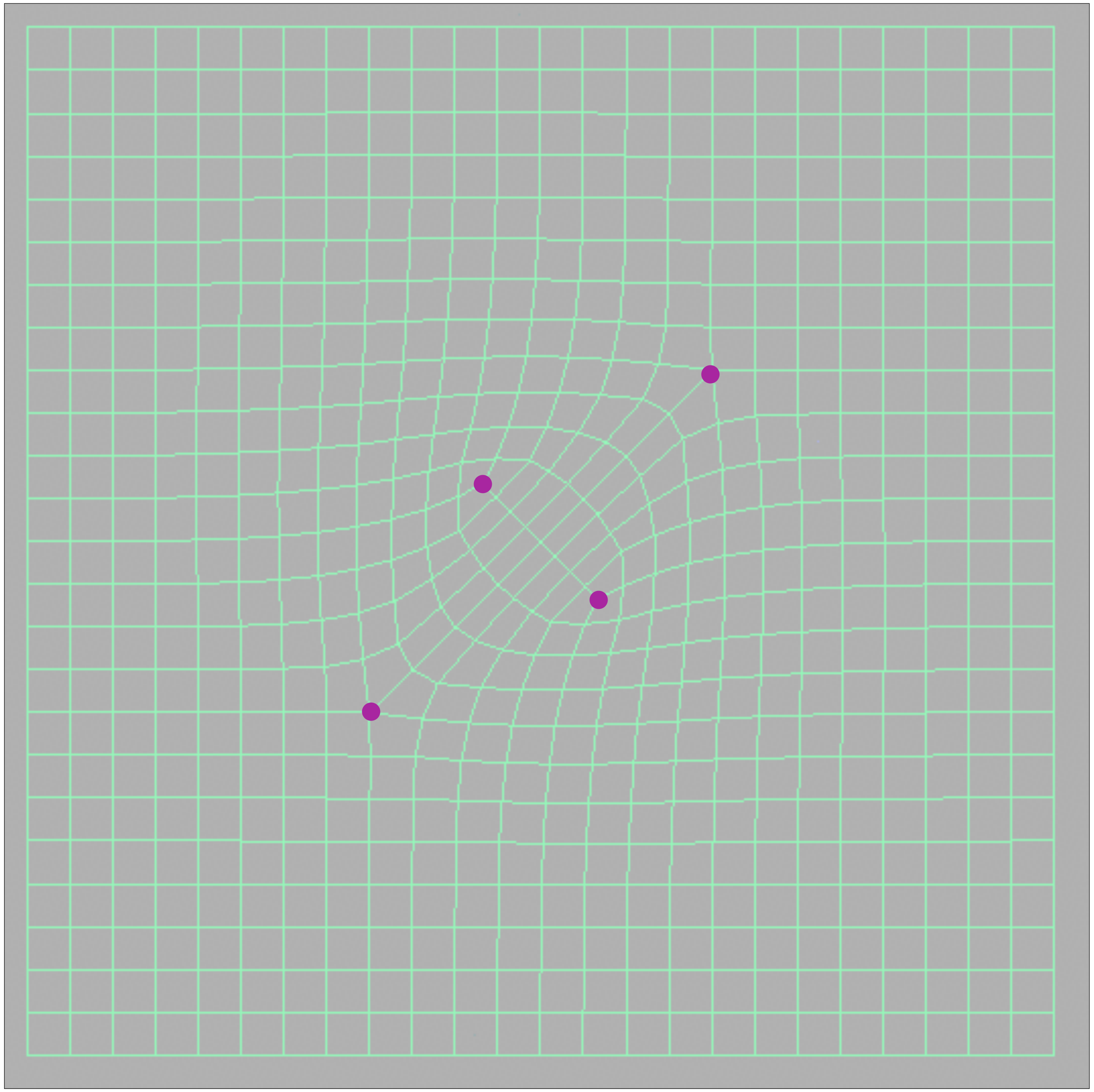
What are their degrees?

How many non-quad faces?

Catmull-Clark Subdivision (General Mesh)

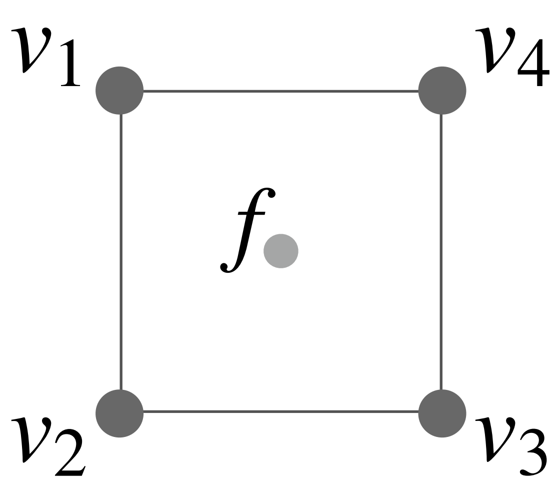


Catmull-Clark Subdivision (General Mesh)



FYI: Catmull-Clark Vertex Update Rules (Quad Mesh)

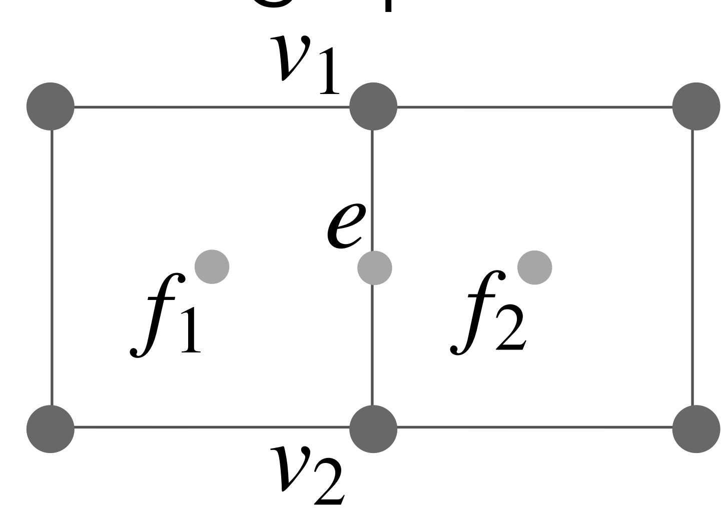
Face point

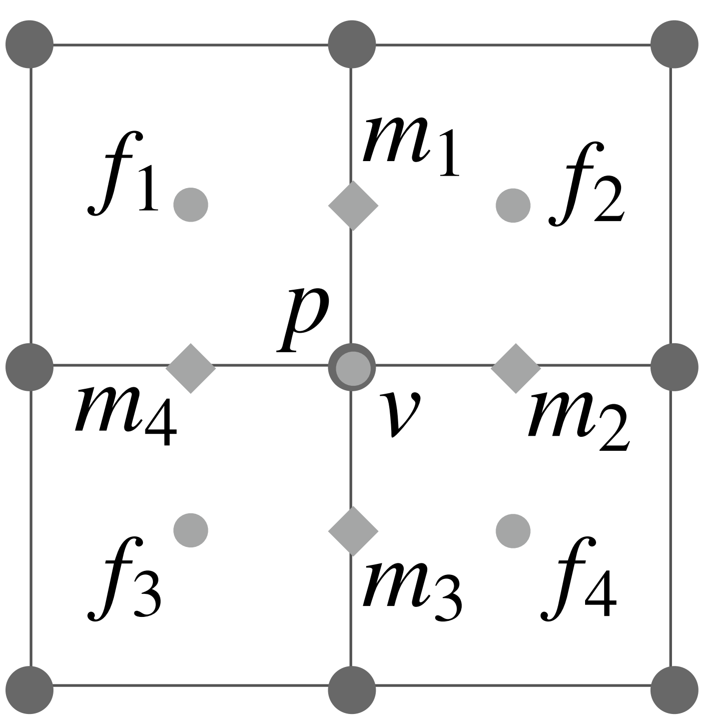


$$f = \frac{v_1 + v_2 + v_3 + v_4}{4}$$

$$e = \frac{v_1 + v_2 + f_1 + f_2}{4}$$

Edge point





Vertex point

$$v = \frac{f_1 + f_2 + f_3 + f_4 + 2(m_1 + m_2 + m_3 + m_4) + 4p}{16}$$

m **midpoint of edge**

p **old "vertex point"**

Convergence: Overall Shape and Creases

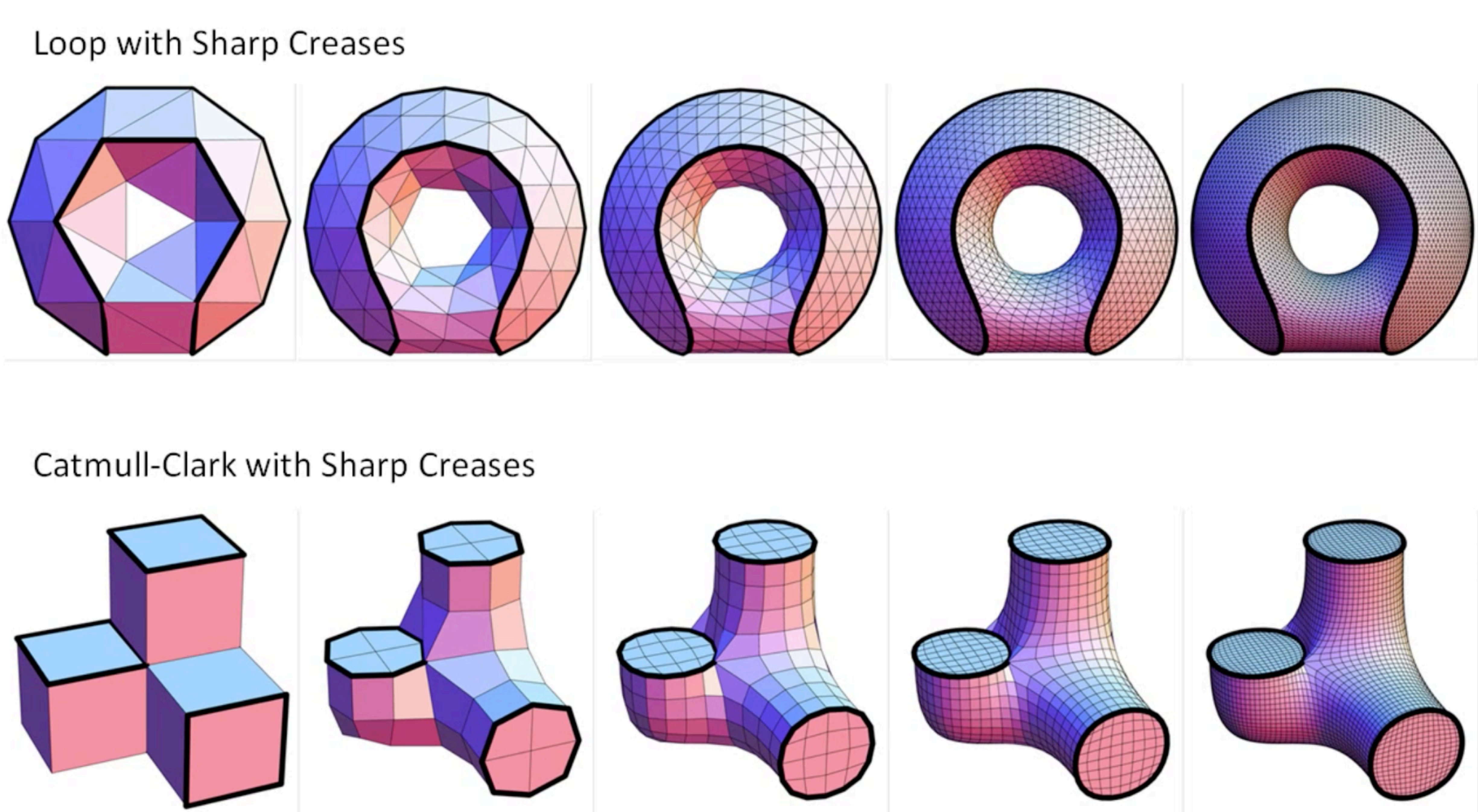


Figure from: Hakenberg et al. Volume Enclosed by Subdivision Surfaces with Sharp Creases

Subdivision in Action (Pixar's "Geri's Game")

Mesh Simplification

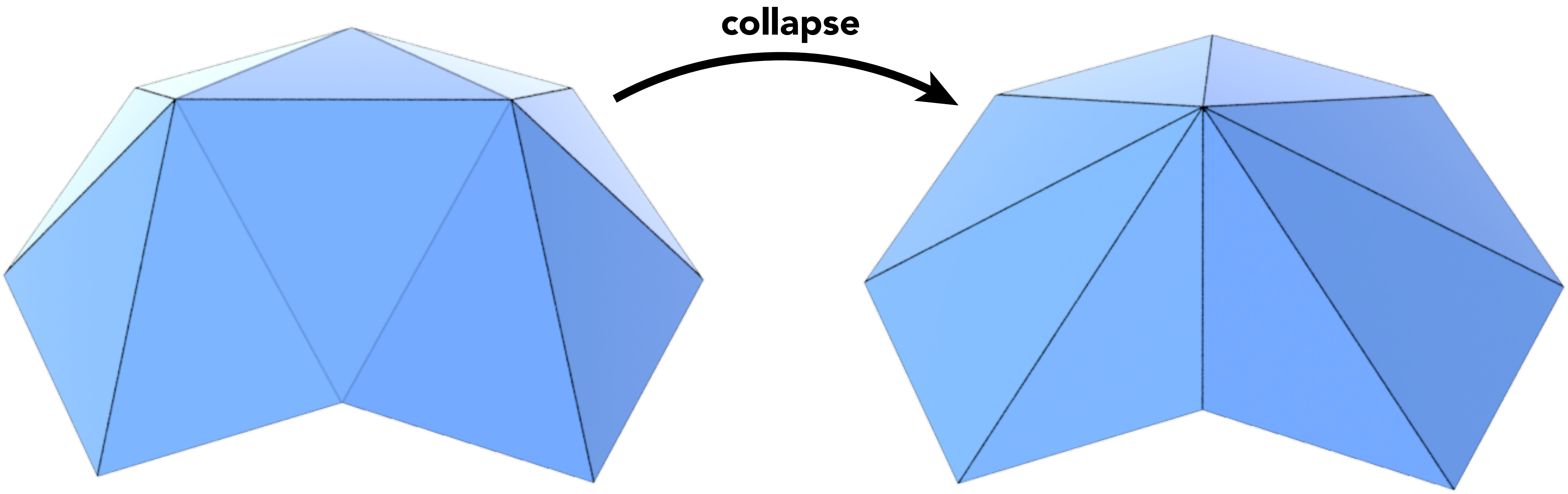
Mesh Simplification

Goal: reduce number of mesh elements while maintaining the overall shape

How to compute?

Collapsing An Edge

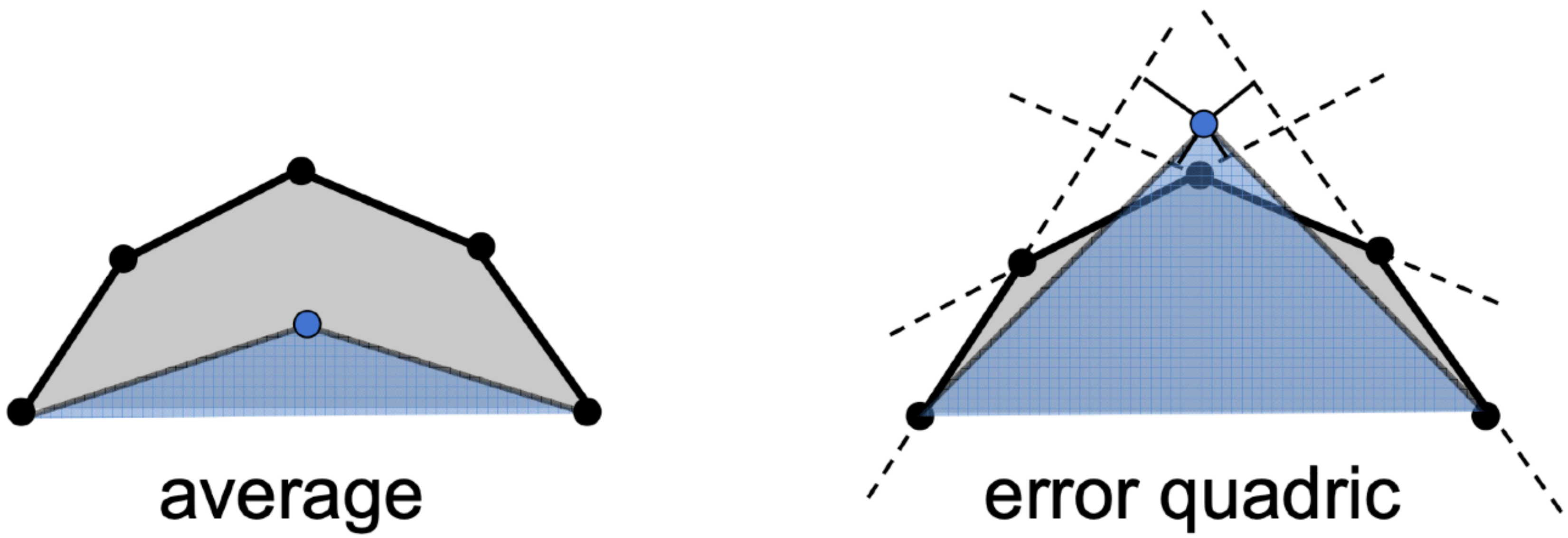
- Suppose we simplify a mesh using edge collapsing



Quadric Error Metrics

（二次误差度量）

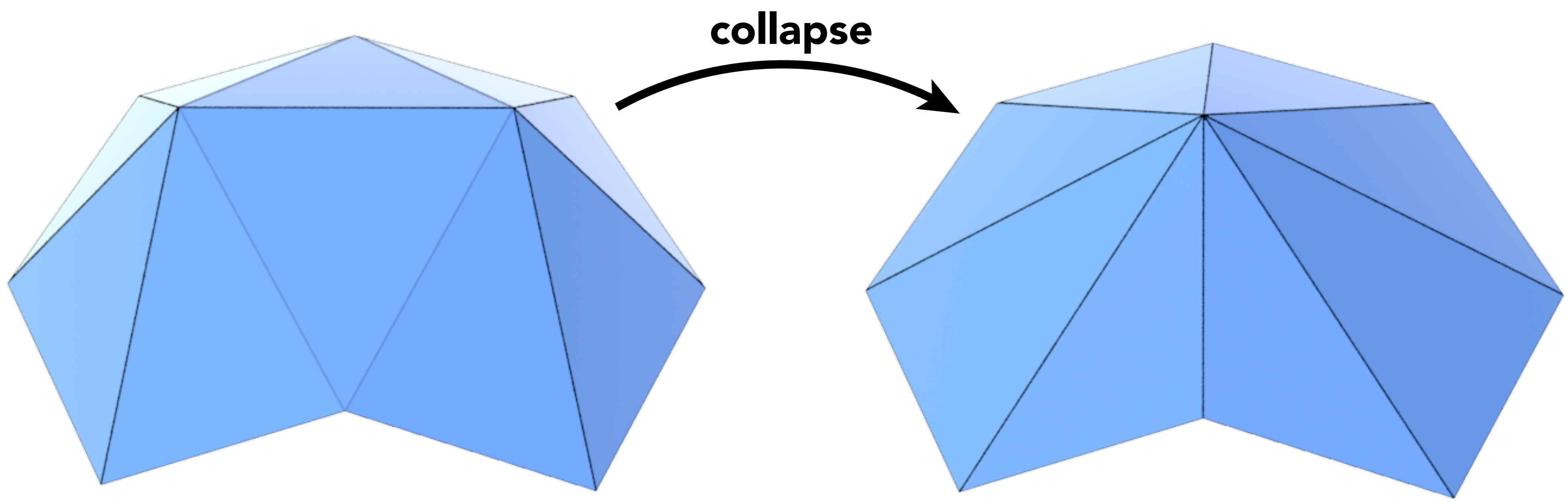
- How much geometric error is introduced by simplification?
- Not a good idea to perform local averaging of vertices
- Quadric error: new vertex should minimize its sum of square distance (L2 distance) to previously related triangle planes!



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

Quadric Error of Edge Collapse

- How much does it cost to collapse an edge?
- Idea: compute edge midpoint, measure quadric error



- Better idea: choose point that minimizes quadric error
- More details: Garland & Heckbert 1997.

Simplification via Quadric Error

Iteratively collapse edges

Which edges? Assign score with quadric error metric*

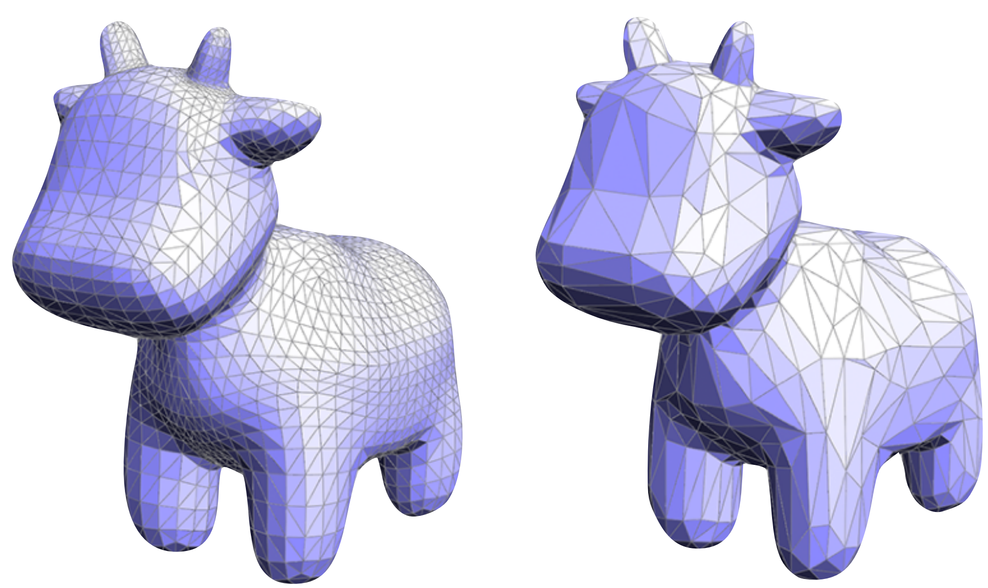
- approximate distance to surface as sum of distances to planes containing triangles
- iteratively collapse edge with smallest score
- greedy algorithm… great results!

* (Garland & Heckbert 1997)

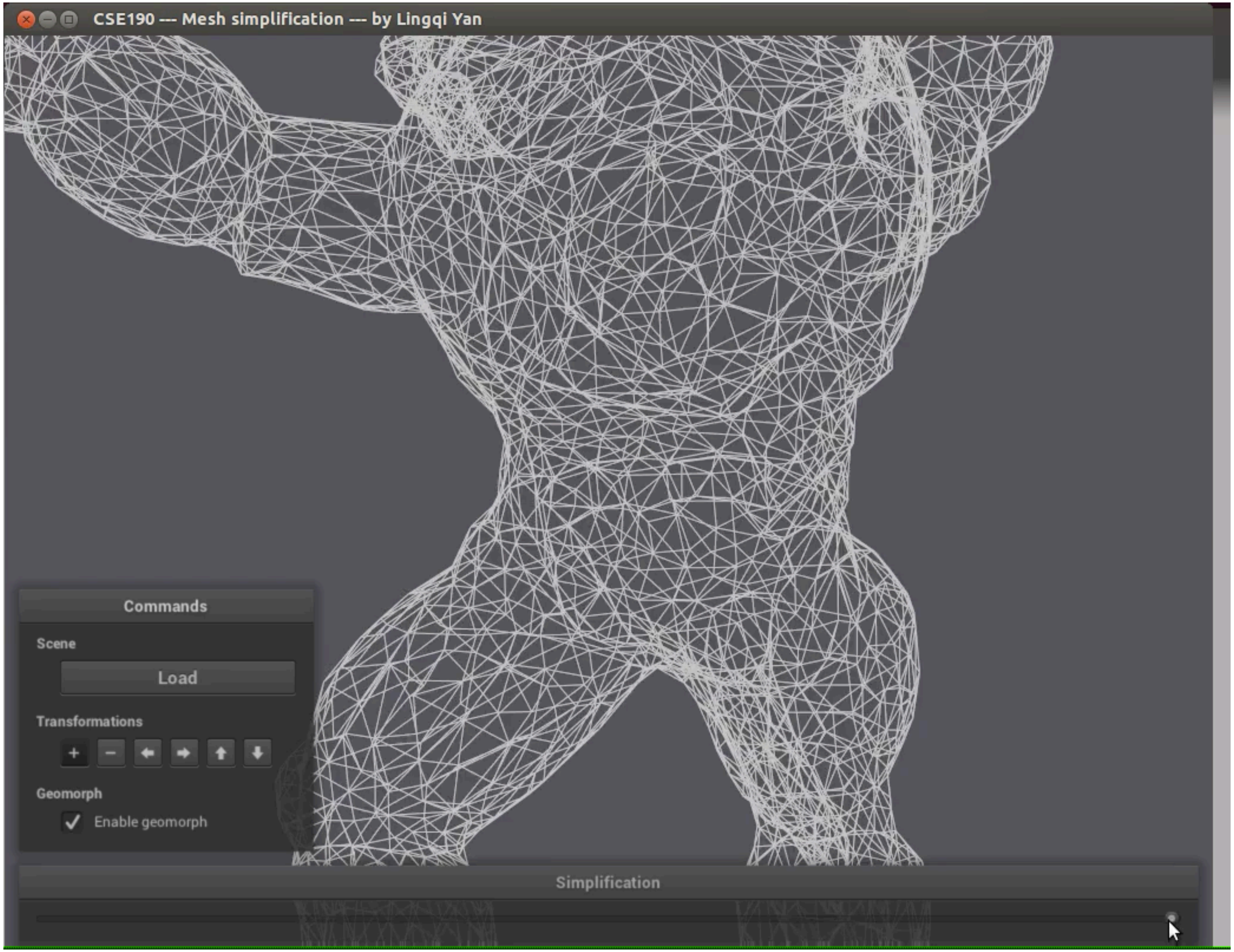
Quadric Error Mesh Simplification

5,804 **994** **532** **248** **64**

Garland and Heckbert '97



Quadric Error Mesh Simplification



Before we move on…

- Shadows
  - How to draw shadows using rasterization?
  - **Shadow mapping**!

Shadow of the Tomb Raider, 2018

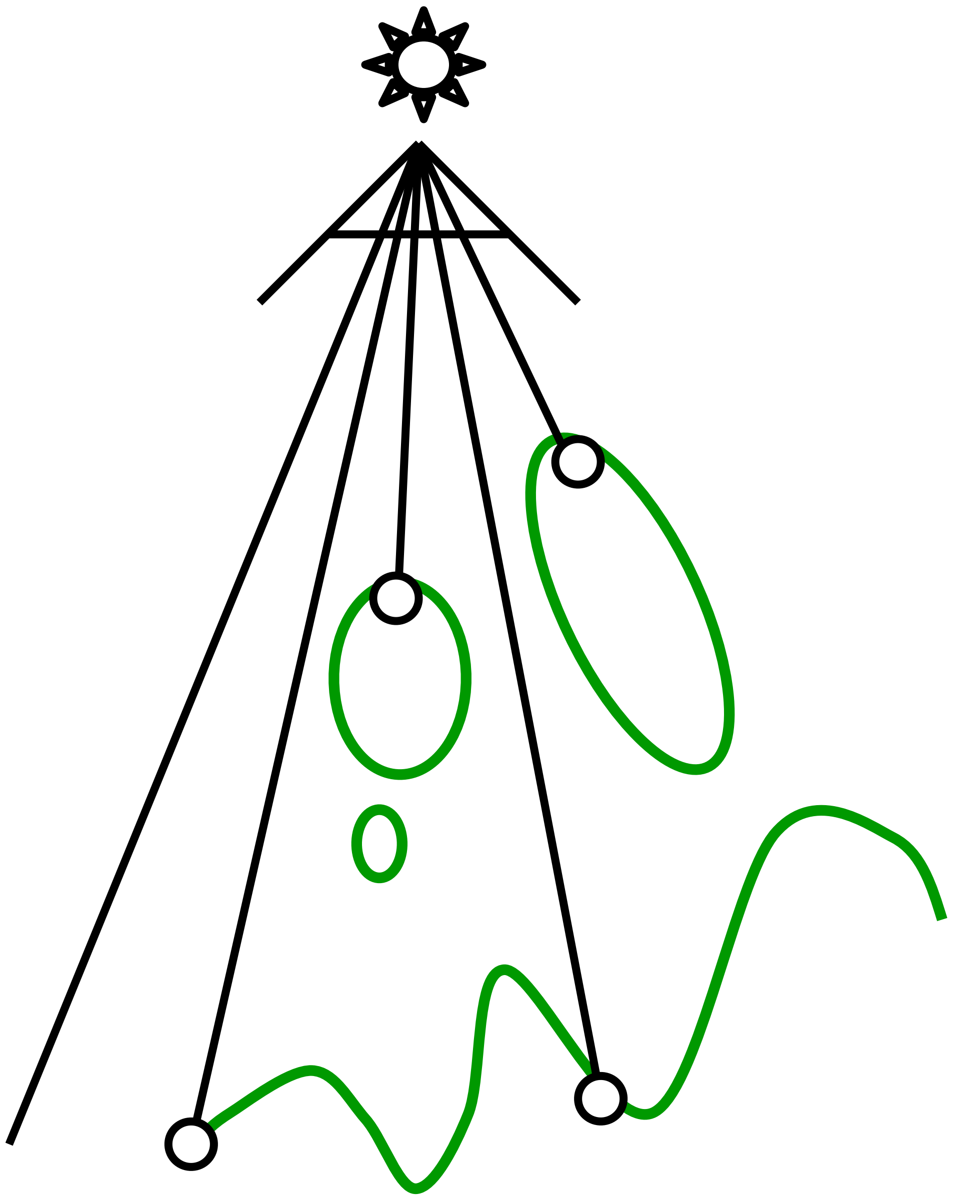
Shadow Mapping

- An Image-space Algorithm
  - no knowledge of scene's geometry during shadow computation
  - must deal with aliasing artifacts

- Key idea:
  - the points NOT in shadow must be seen both **by the light** and **by the camera**

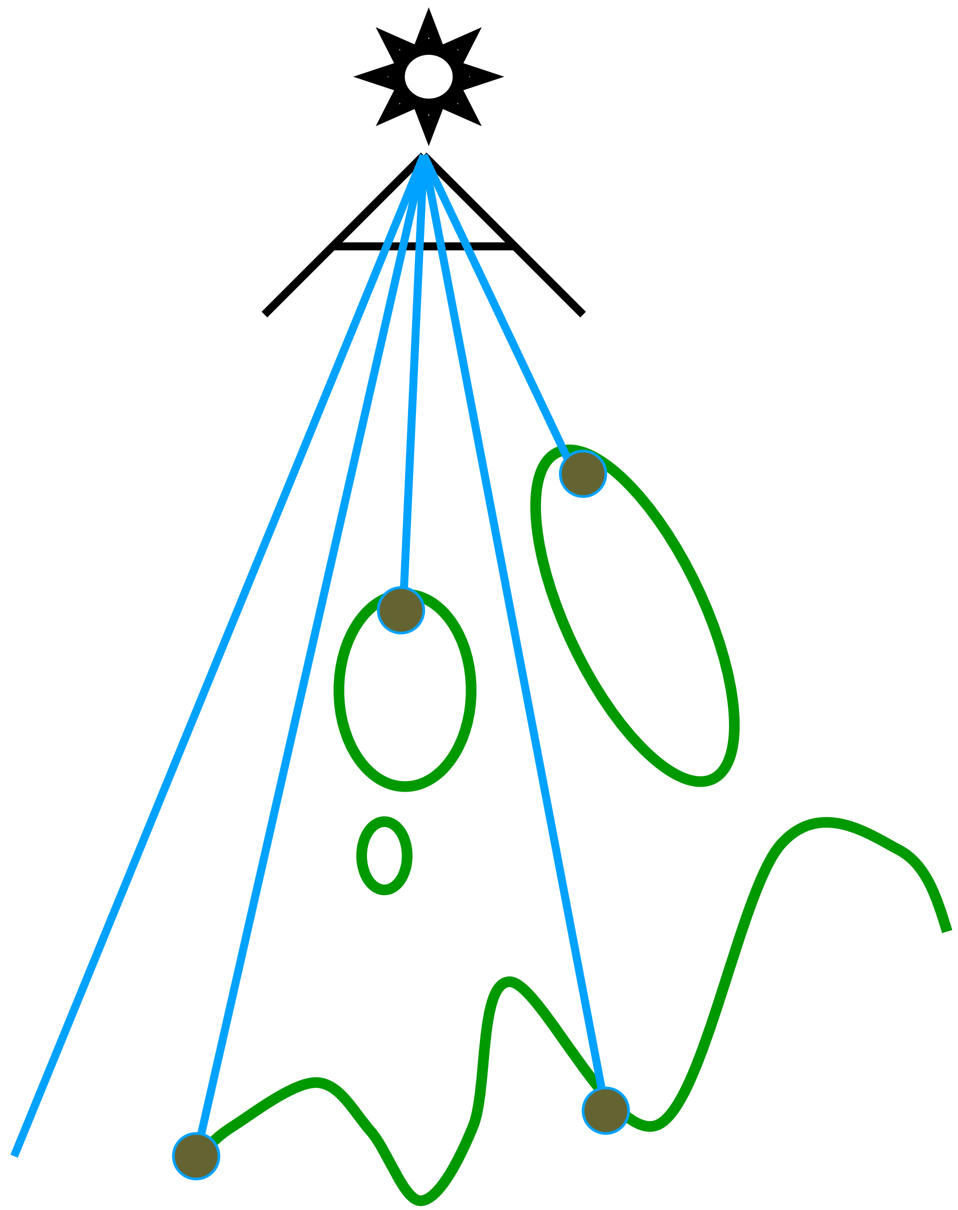
Pass 1: Render from Light

- Depth image from light source



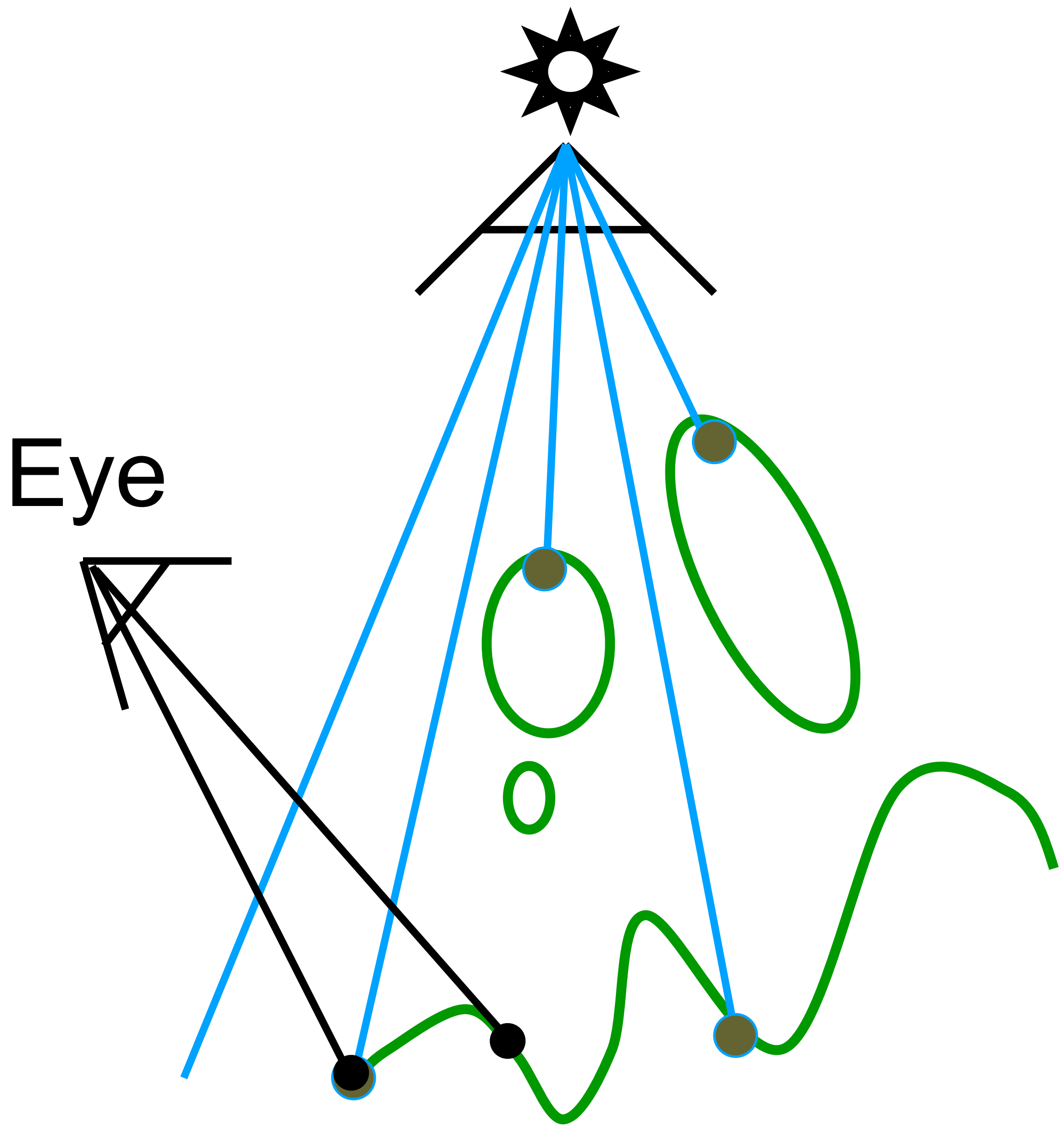
Pass 1: Render from Light

- Depth image from light source



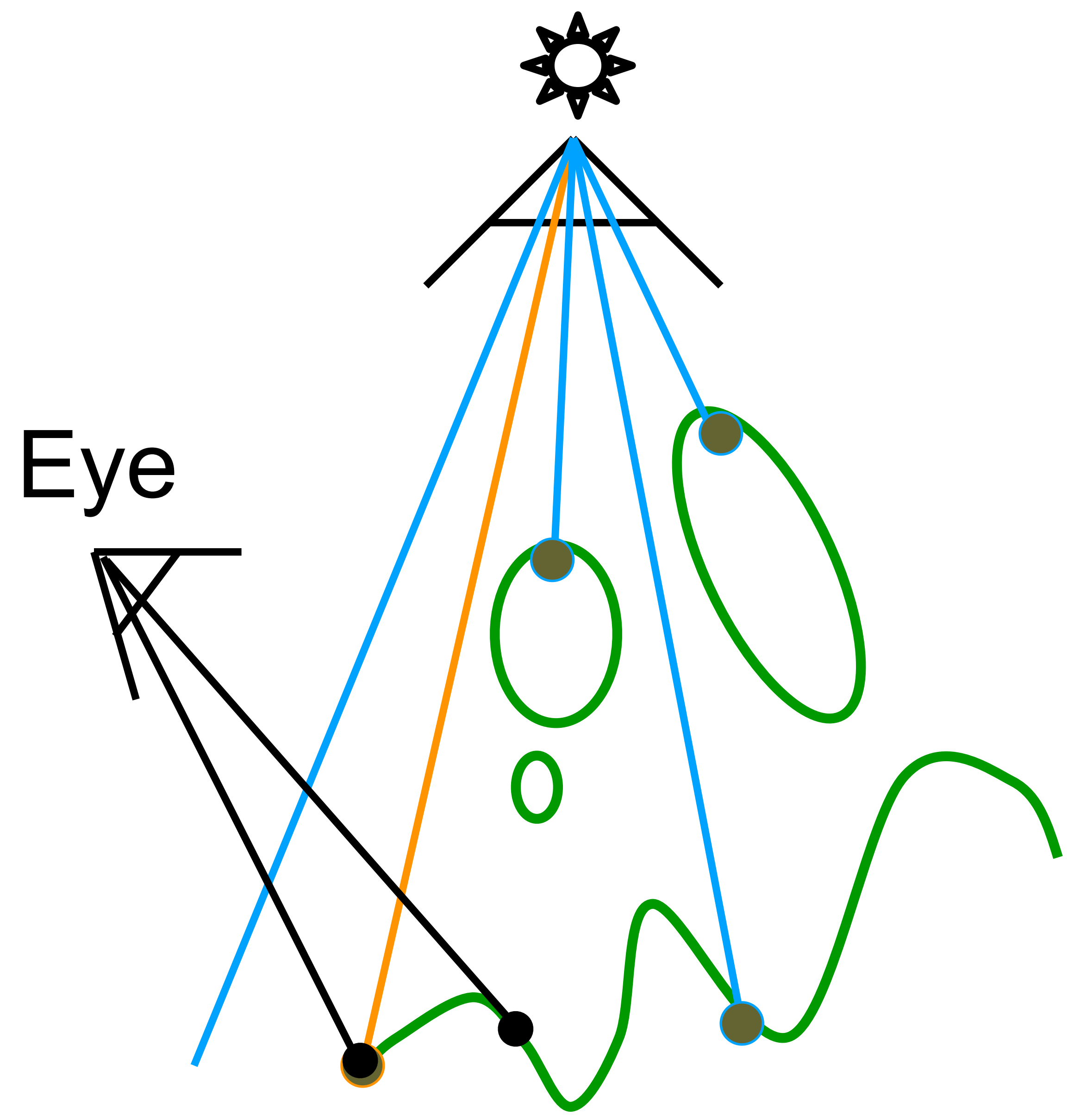
Pass 2A: Render from Eye

- Standard image (with depth) from eye



Pass 2B: Project to light

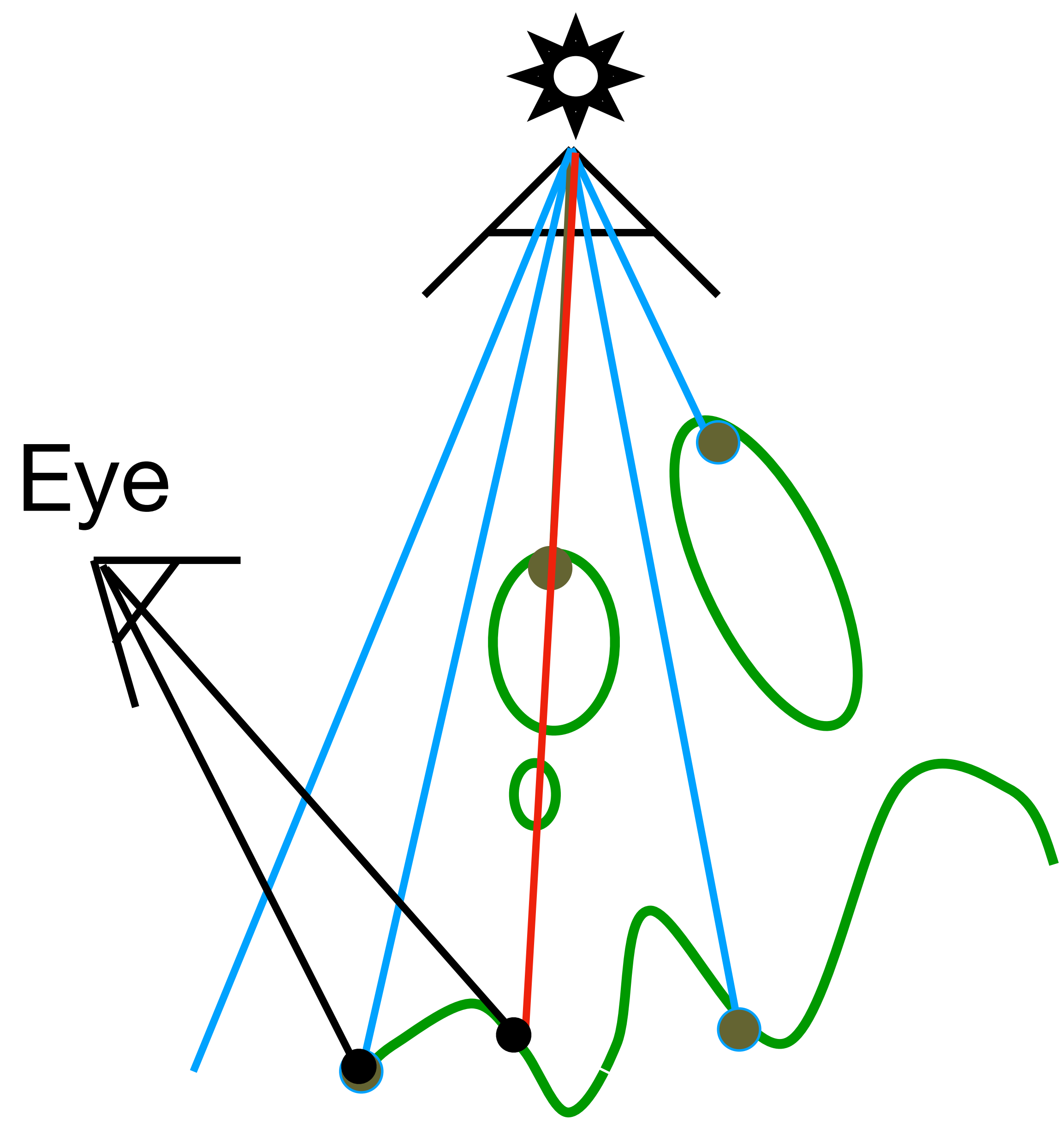
- Project visible points in eye view back to light source



(Reprojected) depths match for light and eye. VISIBLE

Pass 2B: Project to light

- Project visible points in eye view back to light source

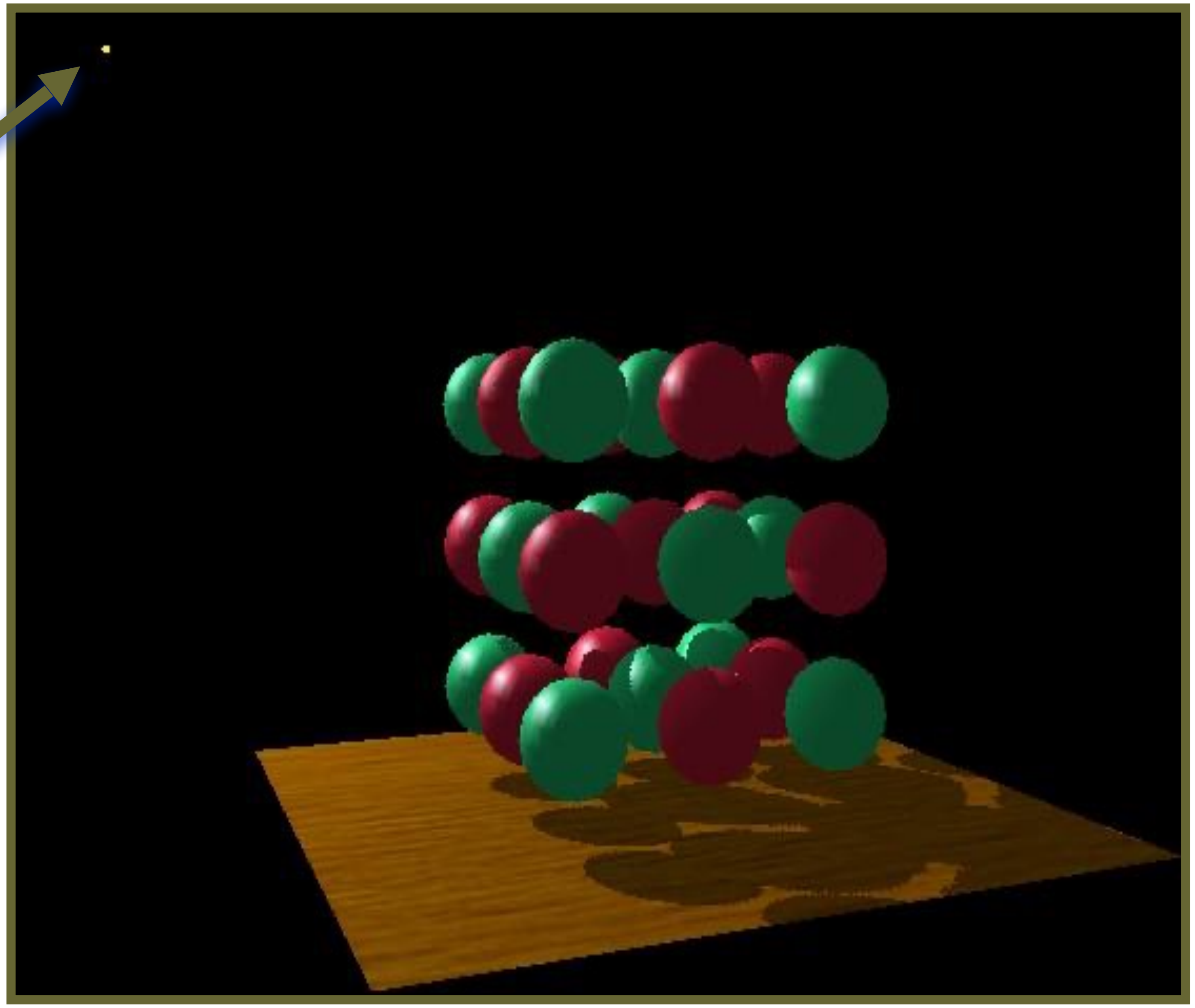


(Reprojected) depths from light and eye are not the same. BLOCKED!!

Visualizing Shadow Mapping

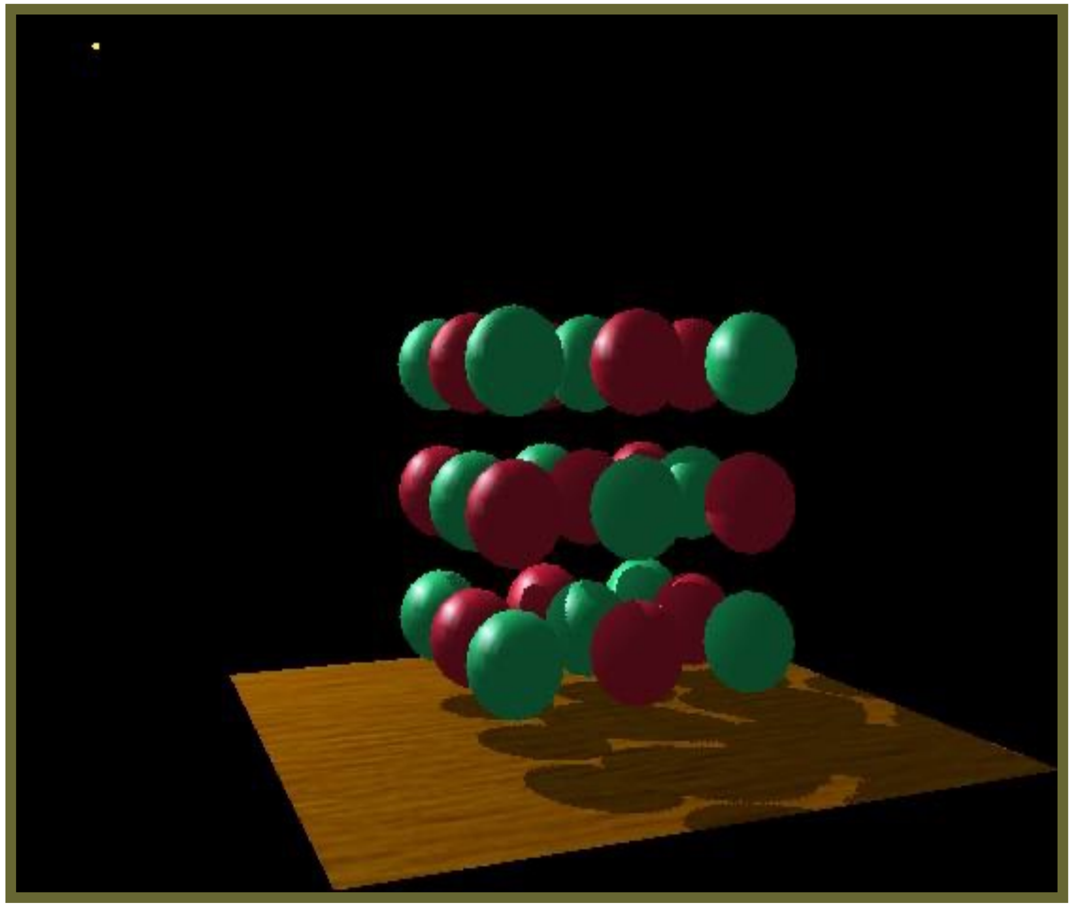
- A fairly complex scene with shadows

the point
light source

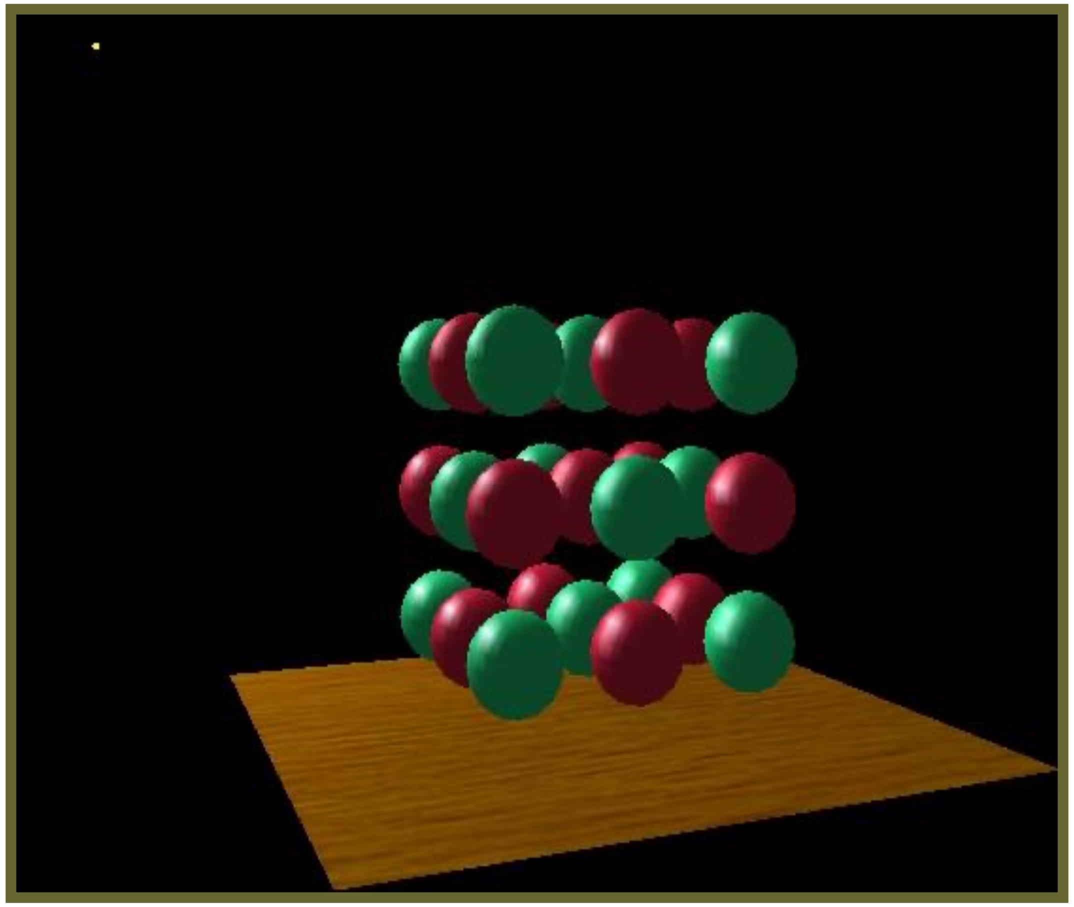


Visualizing Shadow Mapping

- Compare with and without shadows



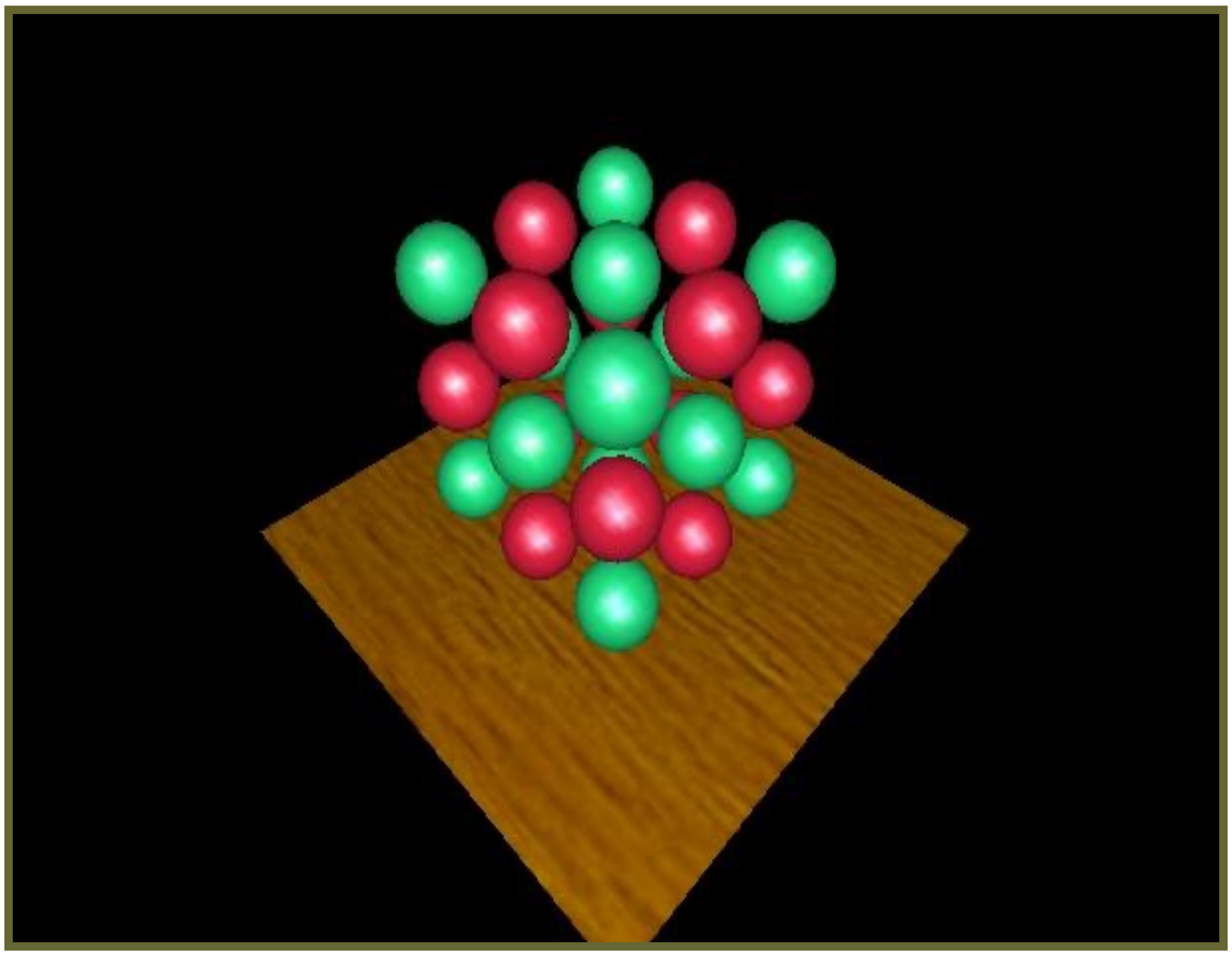
with shadows

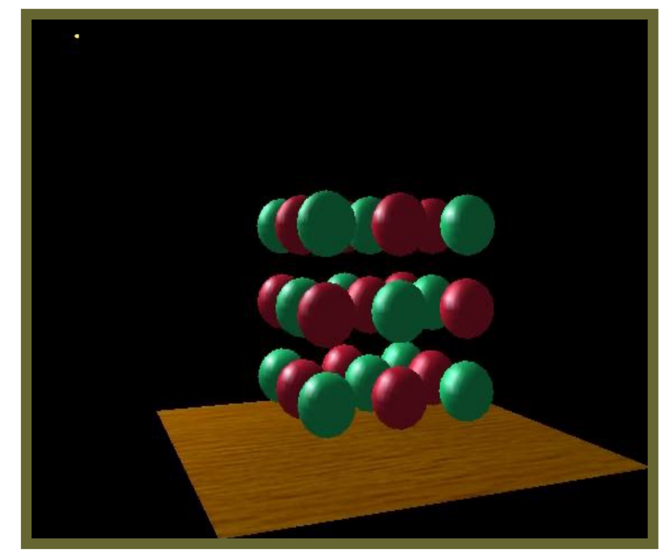


without shadows

Visualizing Shadow Mapping

- The scene from the light's point-of-view

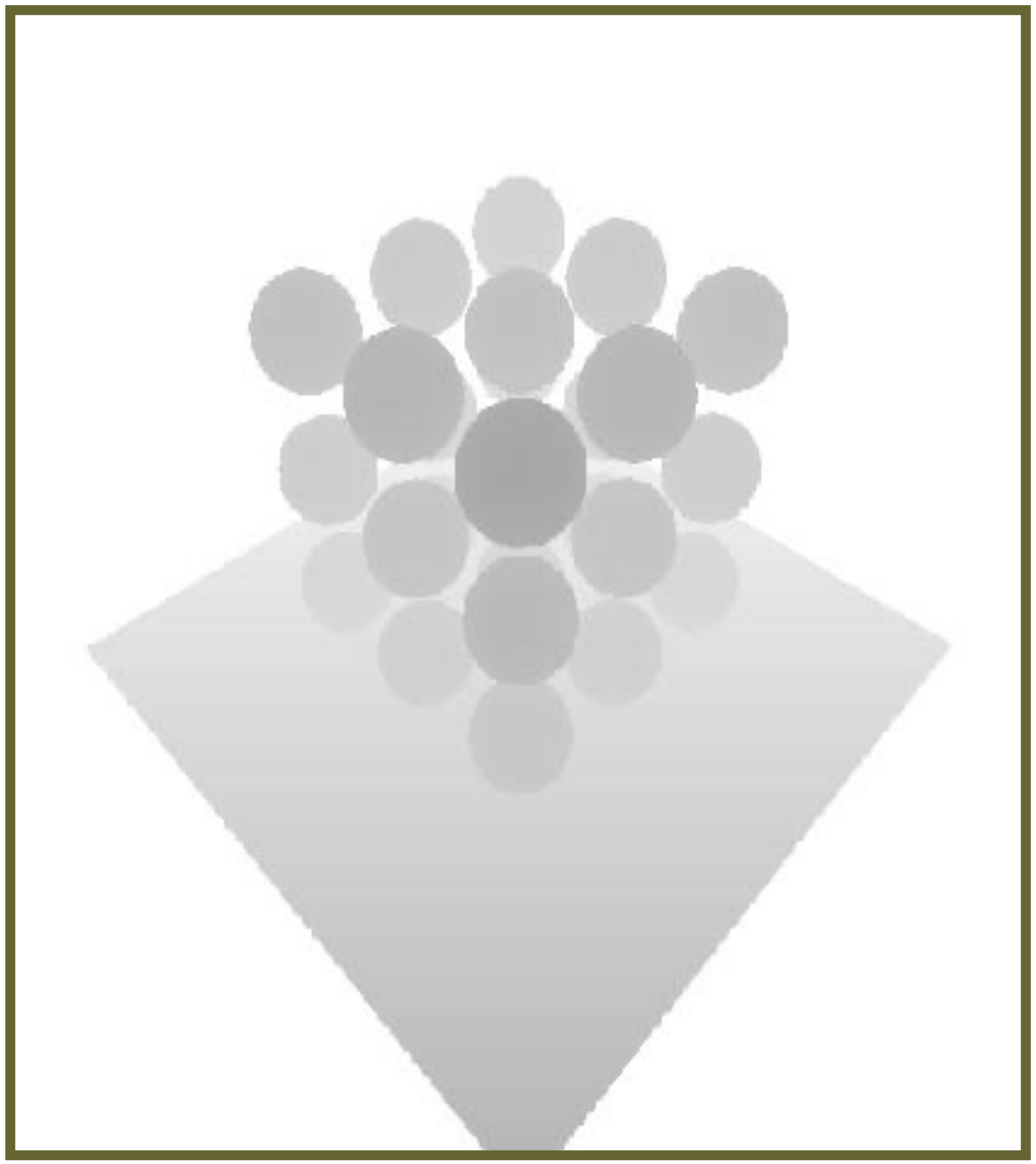


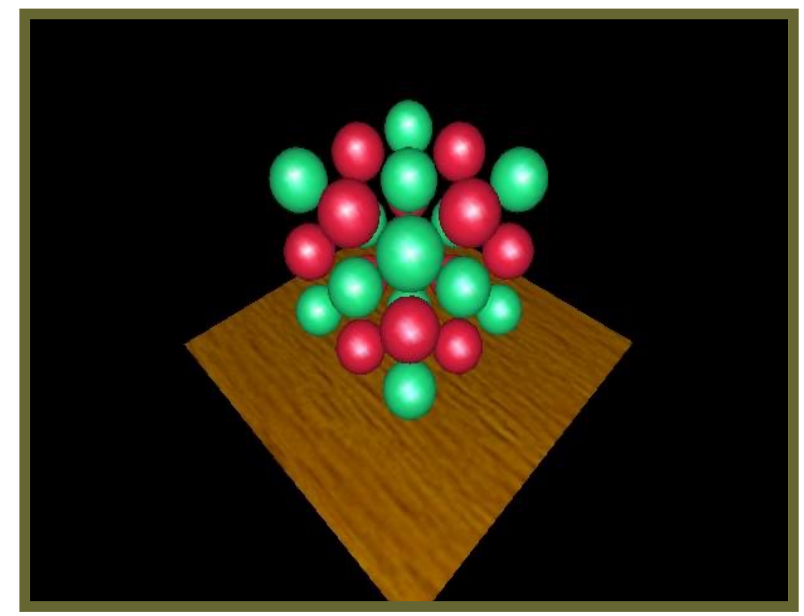


FYI: from the eye's point-of-view again

Visualizing Shadow Mapping

- The depth buffer from the light's point-of-view



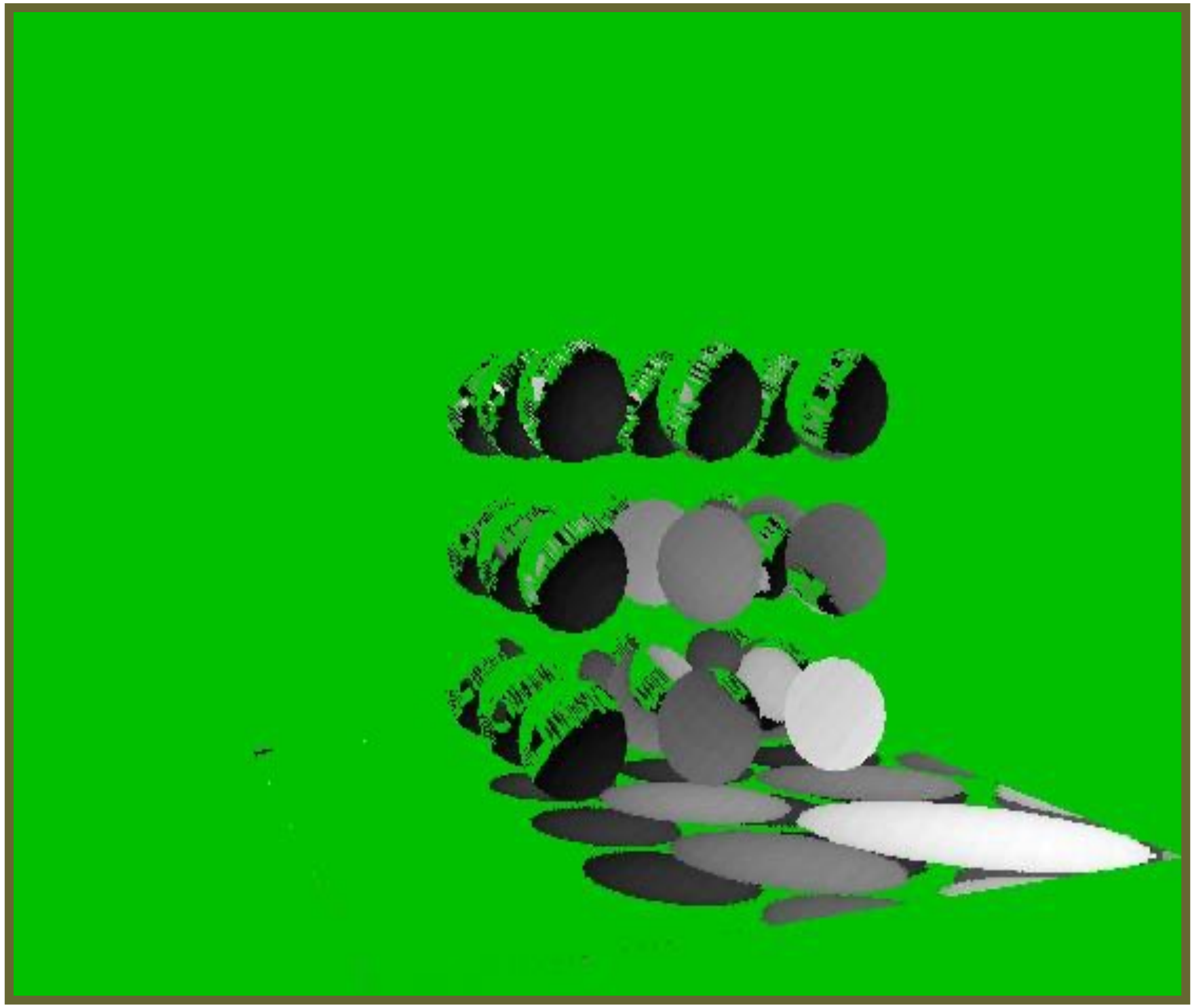


FYI: from the light's point-of-view again

Visualizing Shadow Mapping

- Comparing Dist(light, shading point) with shadow map

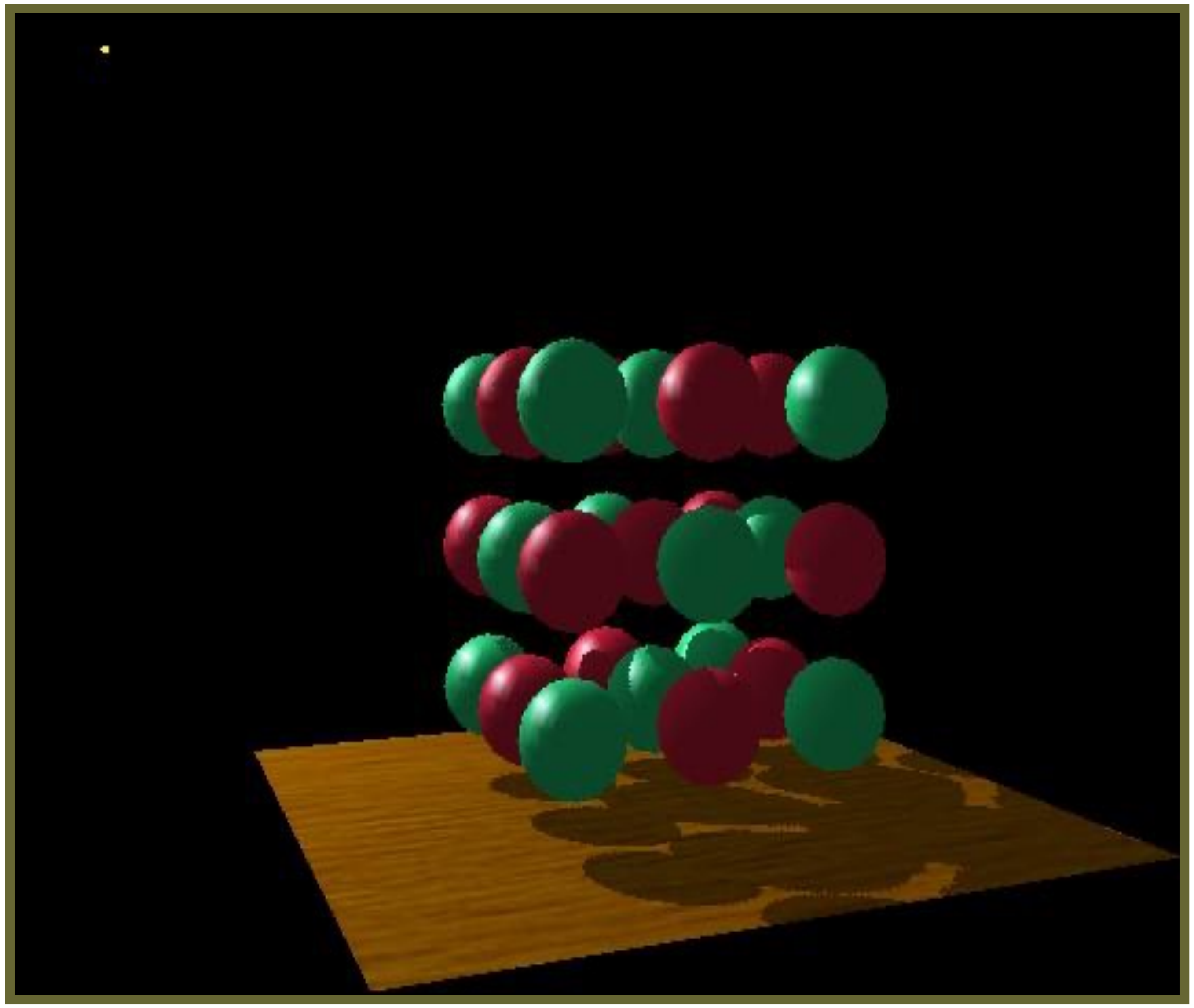
Green is where the distance(light, shading point) ≈ depth on the shadow map



Non-green is where shadows should be

Visualizing Shadow Mapping

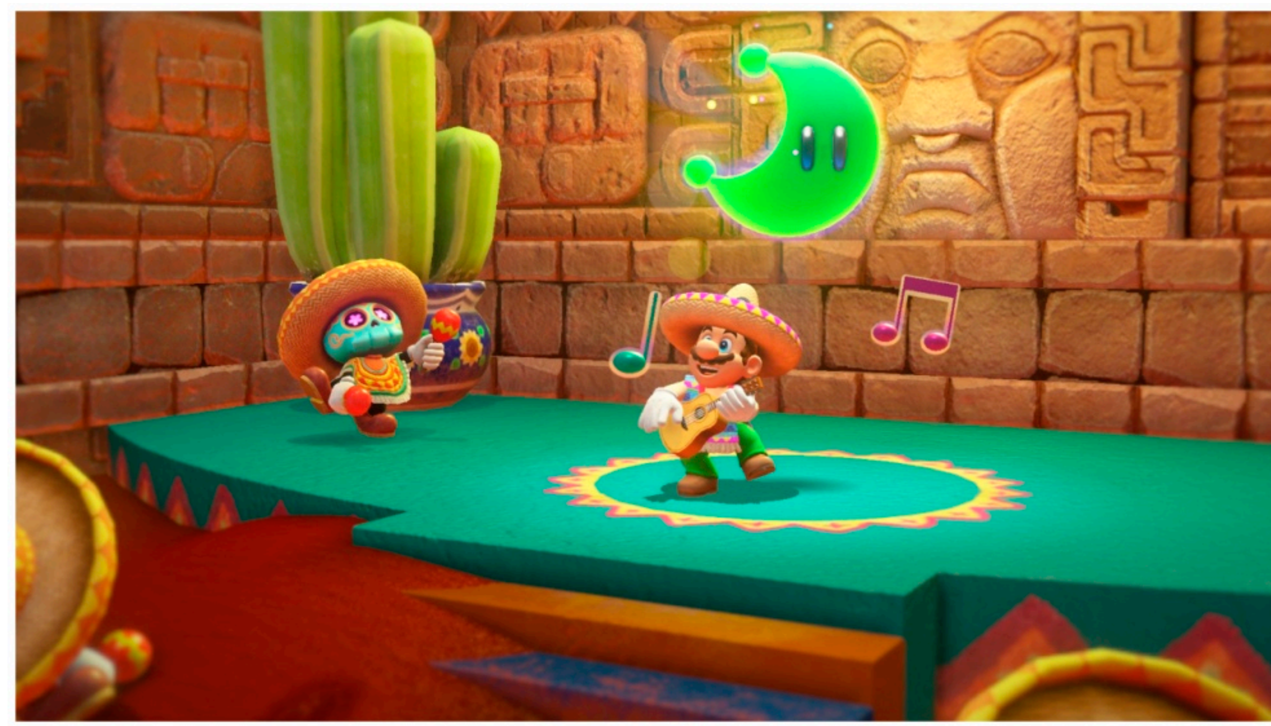
- Scene with shadows



Shadow Mapping

- Well known rendering technique
  - Basic shadowing technique for early animations (Toy Story, etc.) and in EVERY 3D video game

Zelda: Breath of the Wild



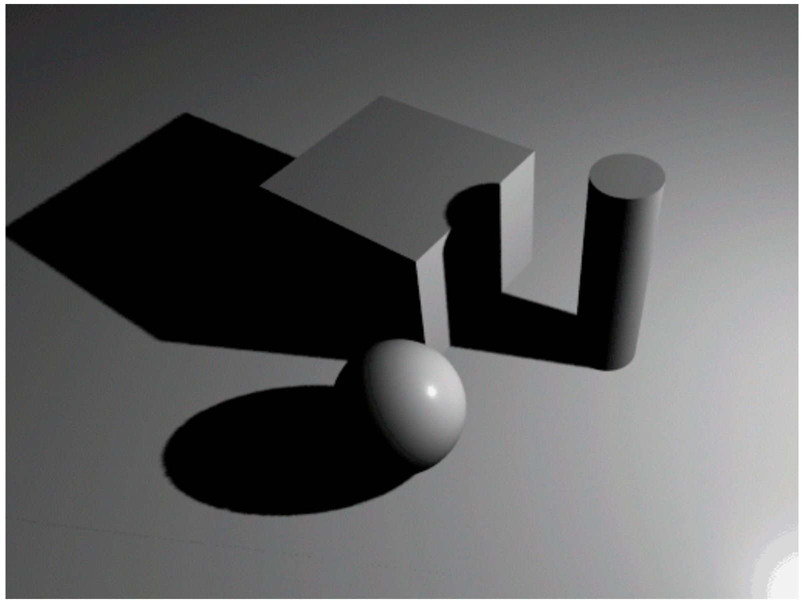
Super Mario Odyssey

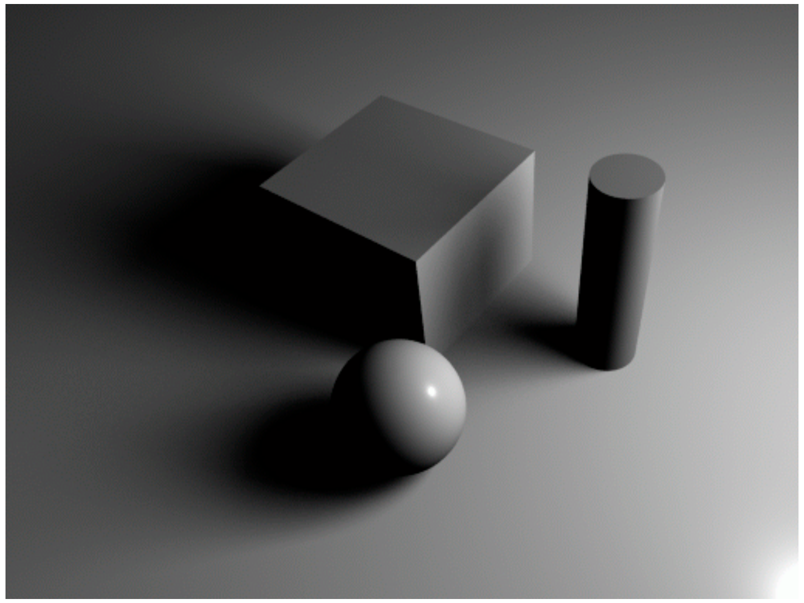
Problems with shadow maps

- Hard shadows (point lights only)
- Quality depends on shadow map resolution
(general problem with image-based techniques)
- Involves equality comparison of floating point depth values means issues of scale, bias, tolerance

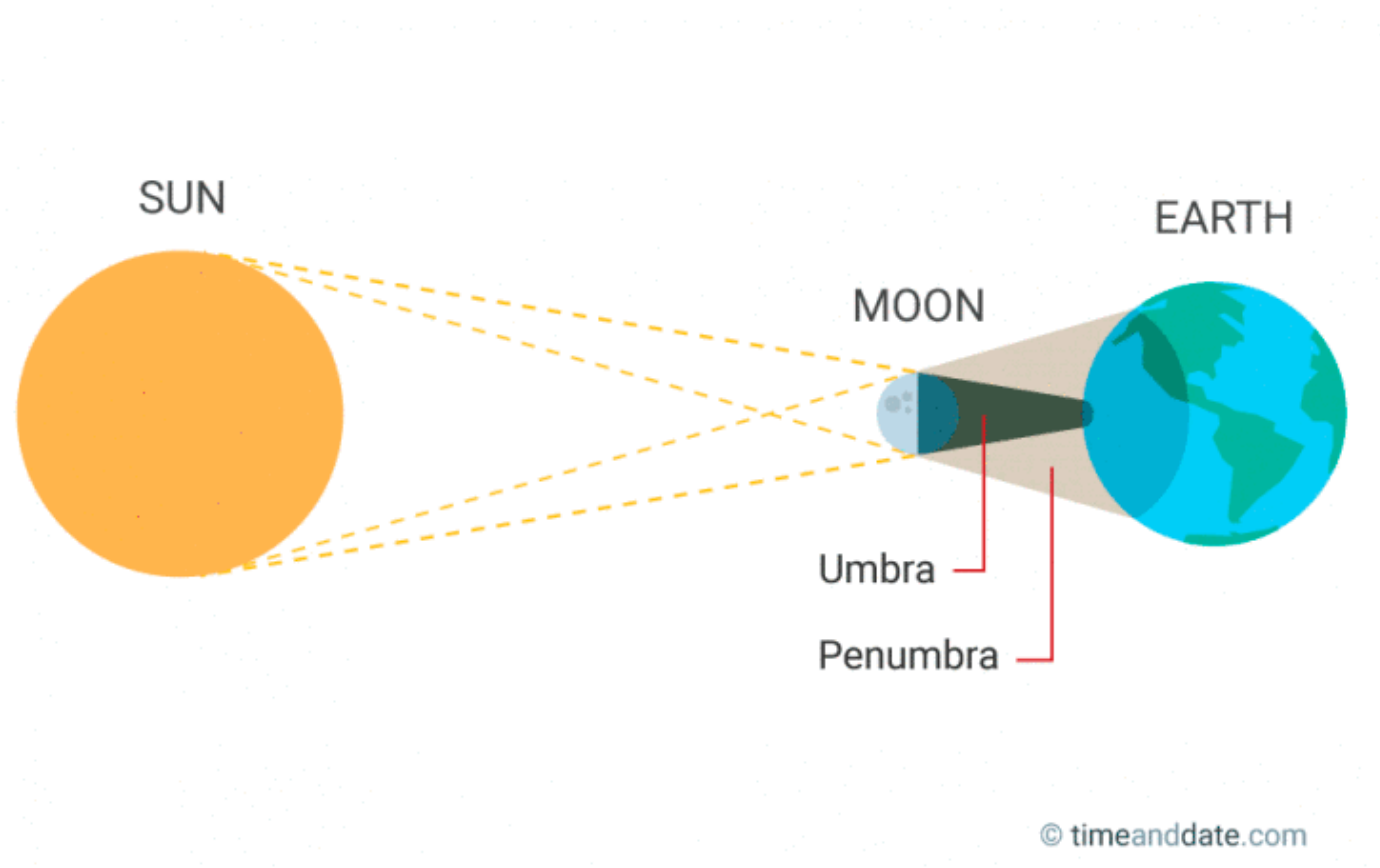
Problems with shadow maps

- Hard shadows vs. soft shadows





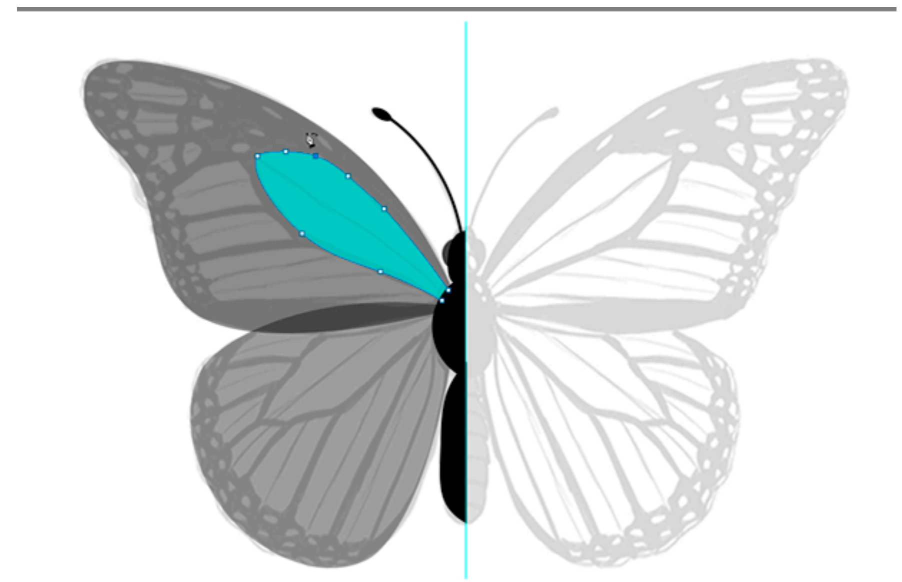
[RenderMan]



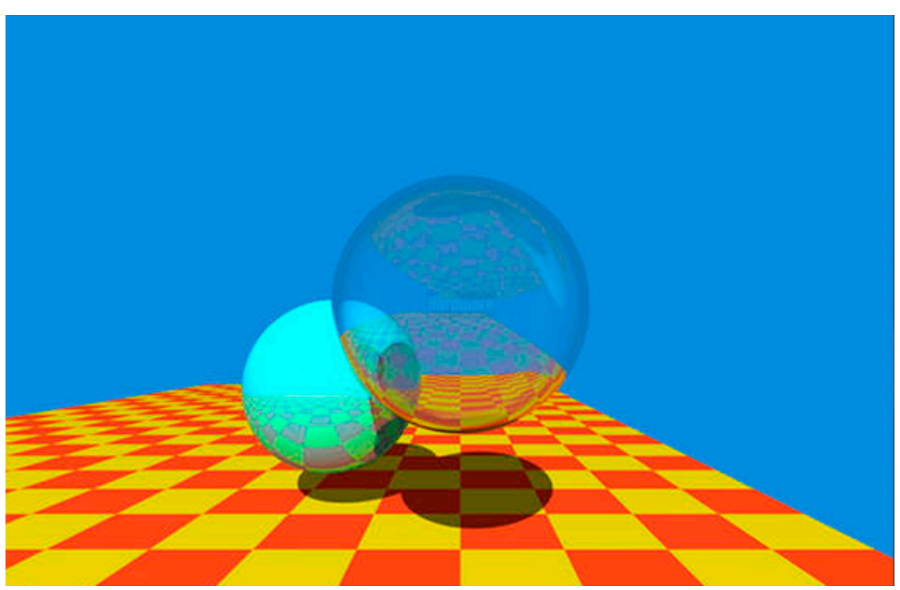
[https://www.timeanddate.com/eclipse/umbra-shadow.html]

Course Roadmap

Rasterization



Geometry



Ray tracing

Animation / simulation

Thank you!

(And thank Prof. Ravi Ramamoorthi and Prof. Ren Ng for many of the slides!)