

Lecture 20: Introduction to Graphics and 3D Modeling

Yuhao Zhu

<http://yuhaozhu.com>
yzhu@rochester.edu

CSC 259/459, Fall 2025
Computer Imaging & Graphics

The Roadmap

Theoretical Preliminaries

Human Visual Systems

Display and Camera

Modeling and Rendering



3D Modeling

Rasterization and GPU

Ray Tracing

Shading

Rendering in AR/VR



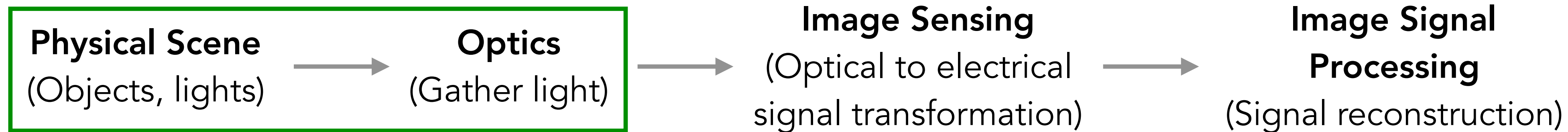






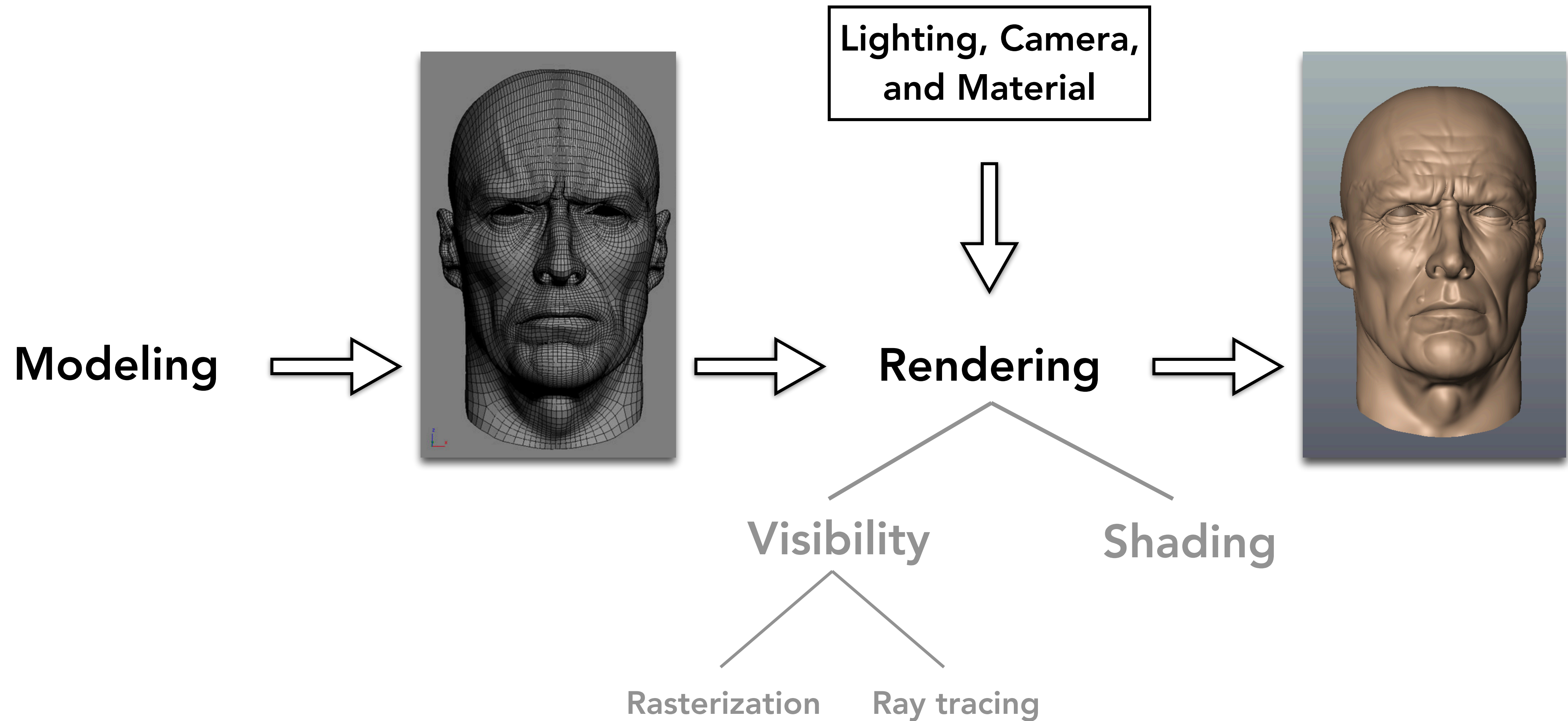


Graphics Rendering



- ▶ Why do those images look “real” to you?
 - ▷ Photorealism: a rendered image that looks like taken by a camera
- ▶ Simulating the camera pipeline (sometimes assuming perfect optical to electrical signal transformation)
 - ▷ Could simulate some desired optical distortions

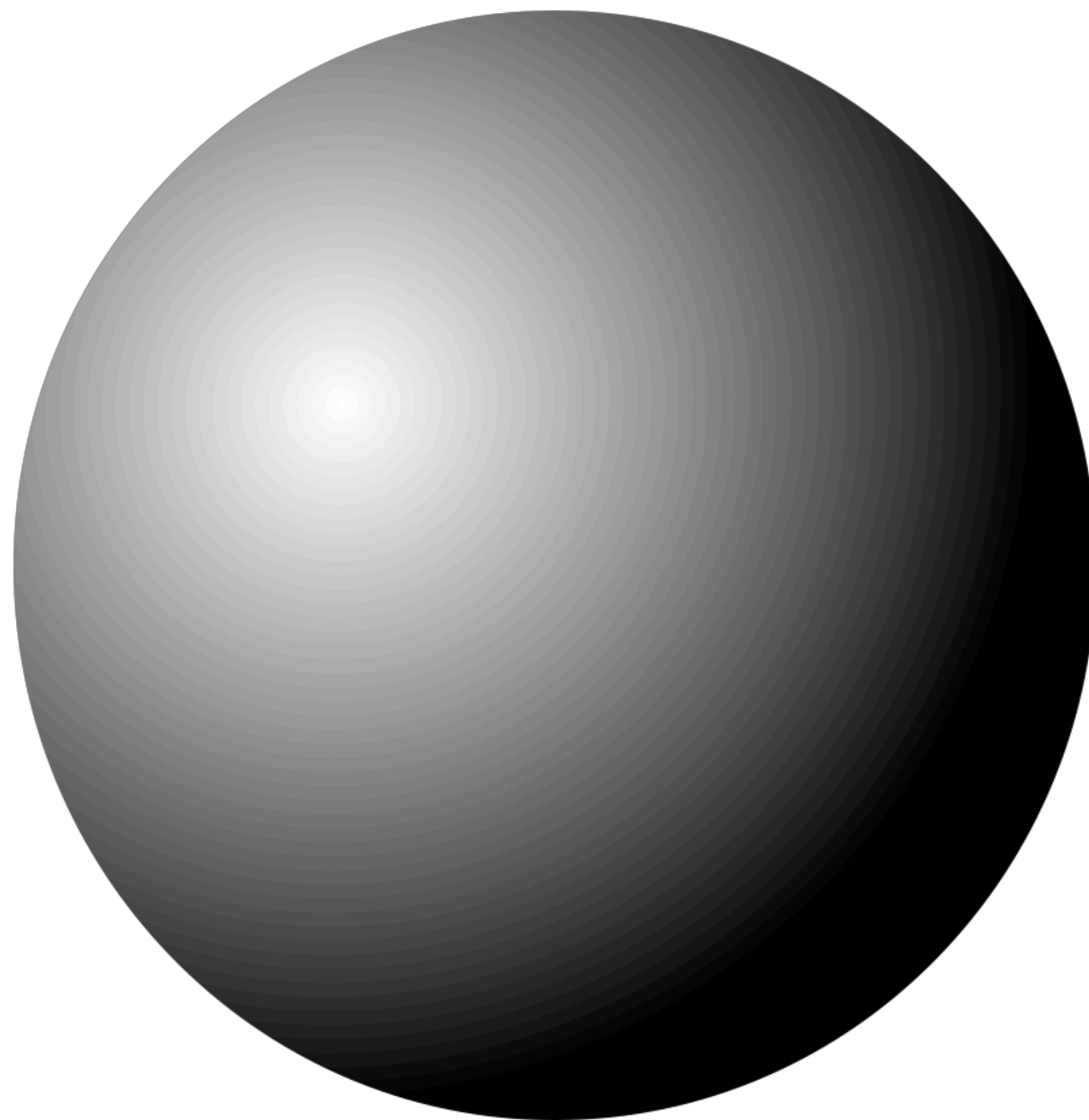
Computer Graphics



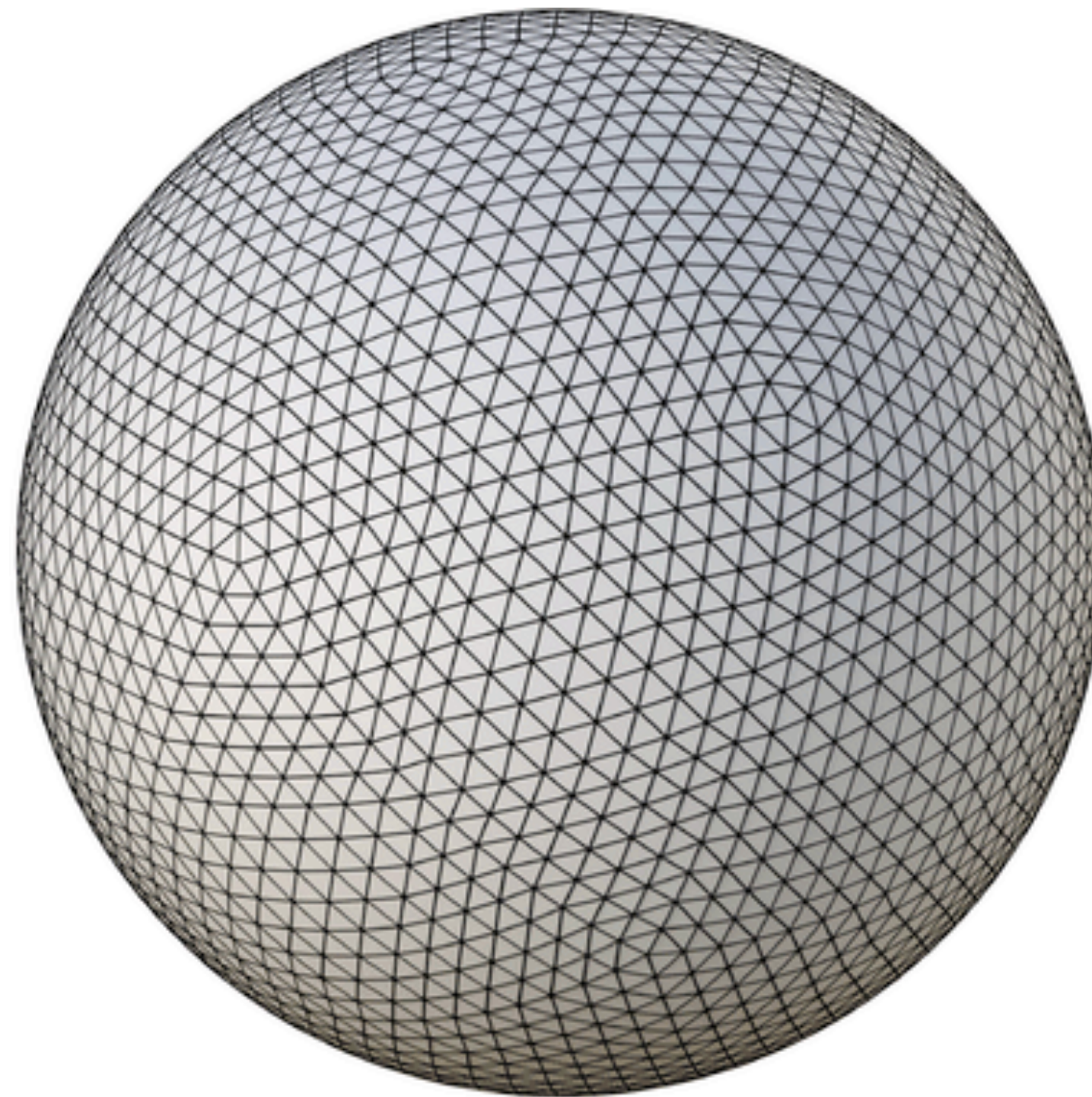
Modeling Objects

3D Modeling

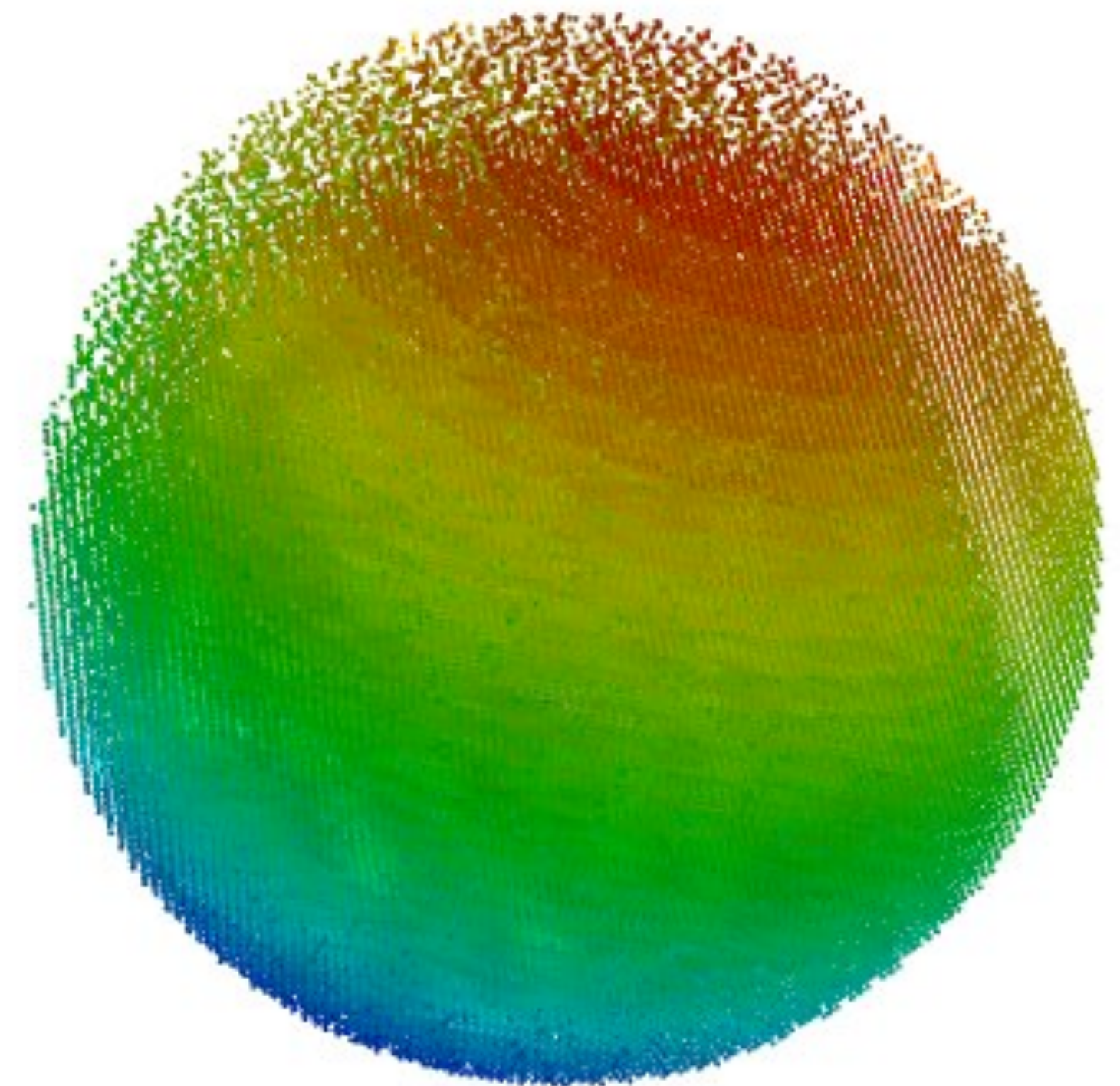
Parametric/Implicit surface



Polygon mesh



Point cloud

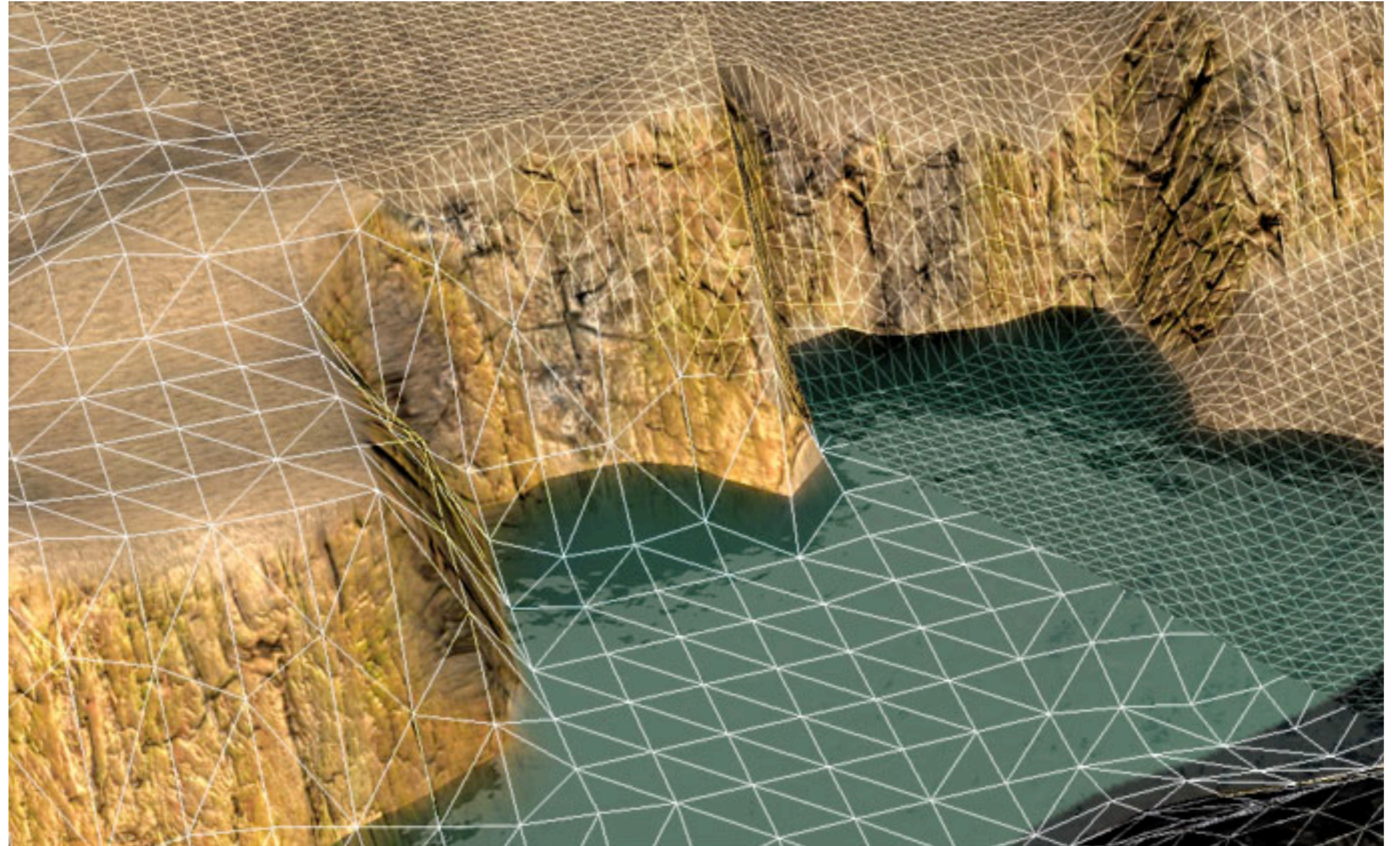
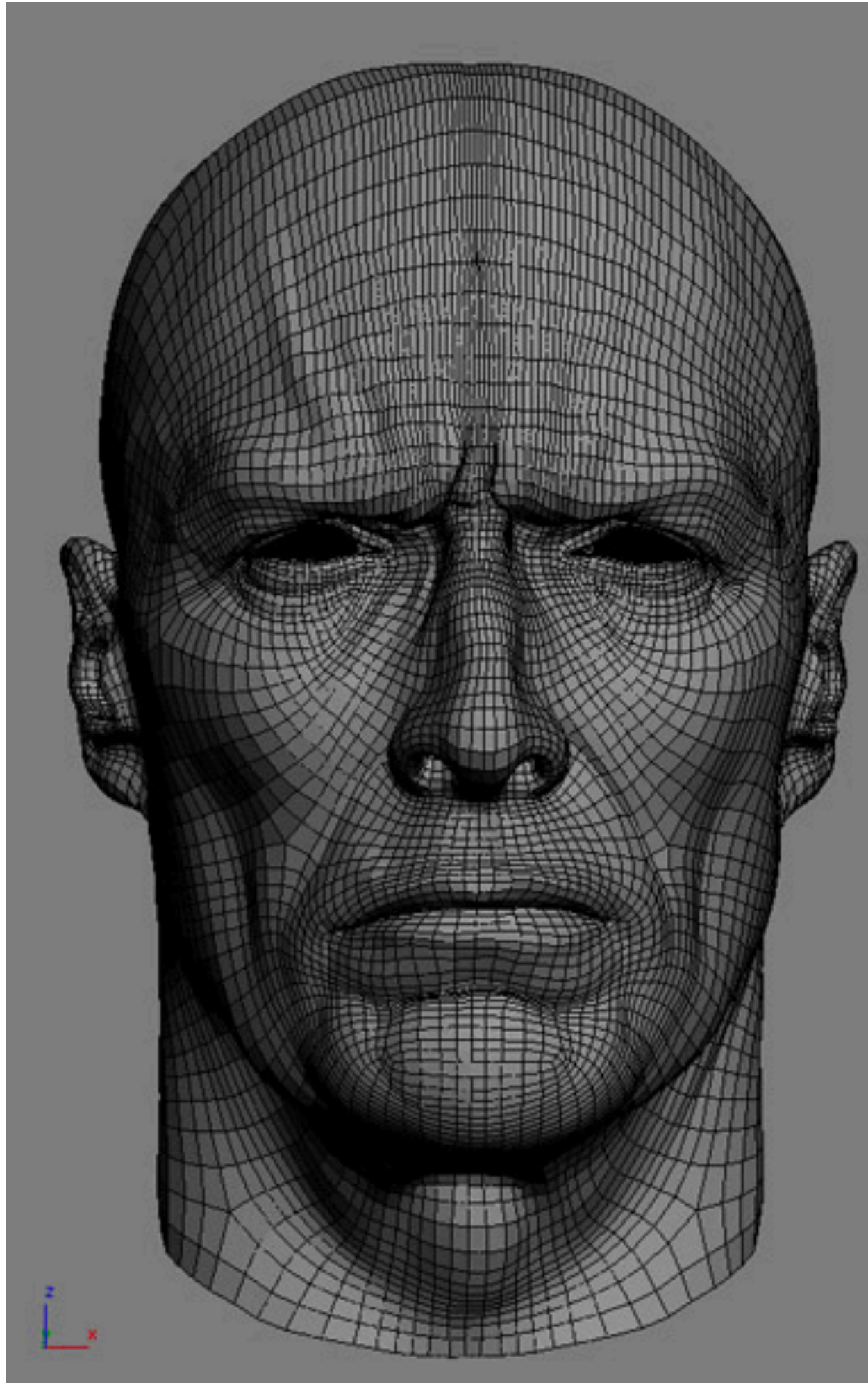


<https://twitter.com/dmitriatsafe/status/1066037693174341632>

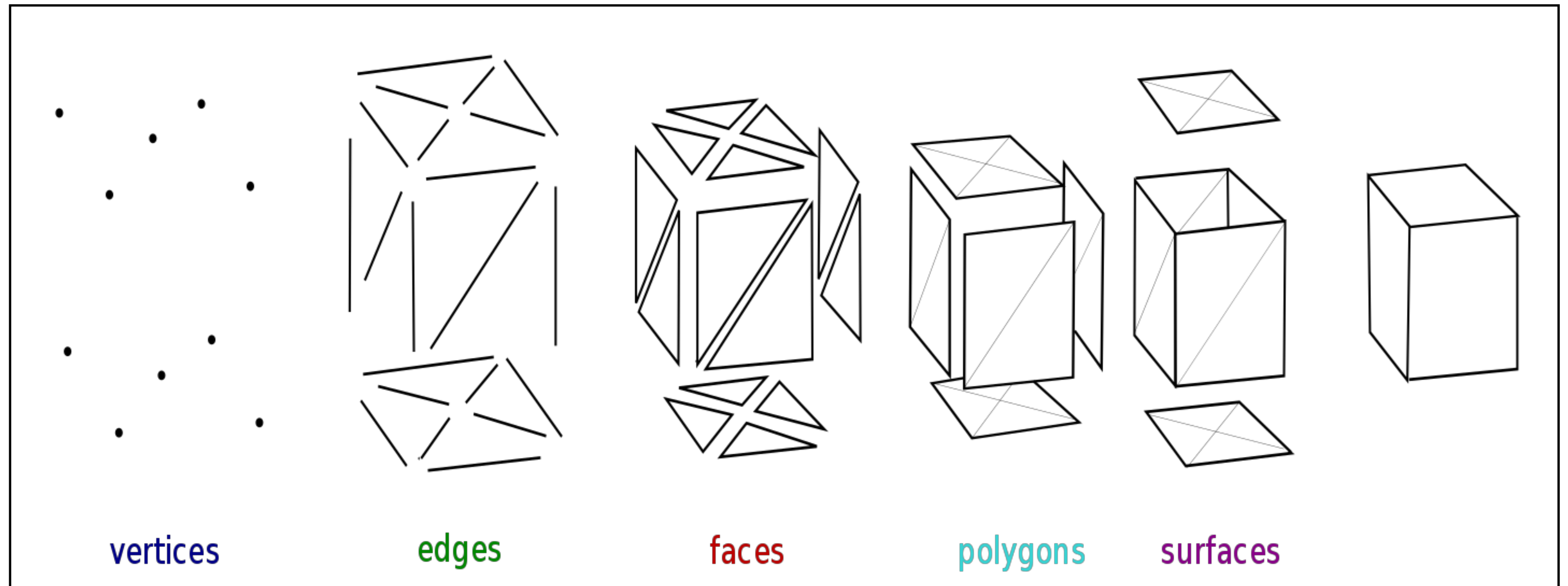
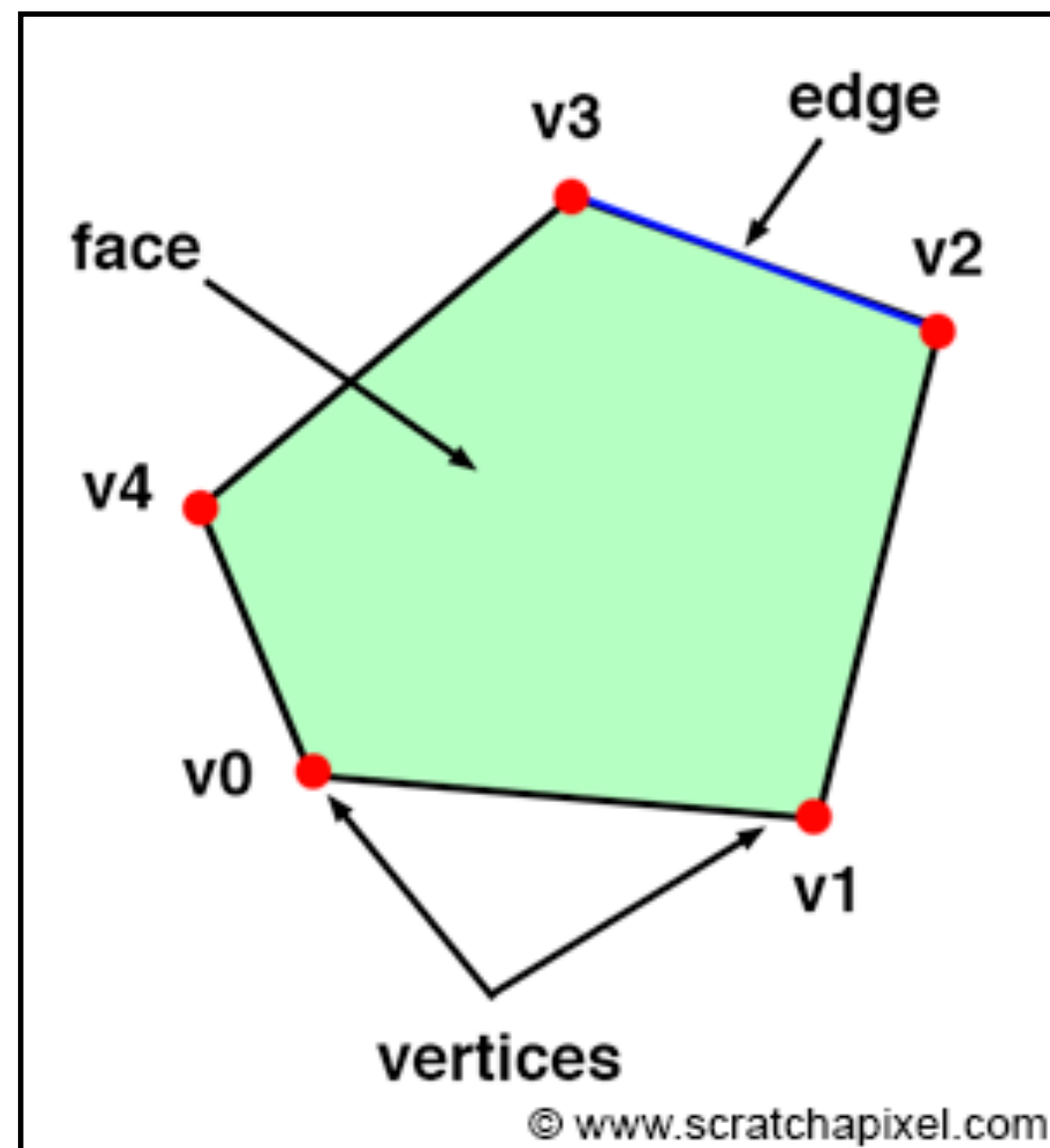
https://commons.wikimedia.org/wiki/File:Sphere_-_monochrome_simple.svg

https://www.researchgate.net/figure/Comparison-between-greedy-and-spectral-reconstructions-of-a-sphere-mesh-with-jittered_fig2_283890246

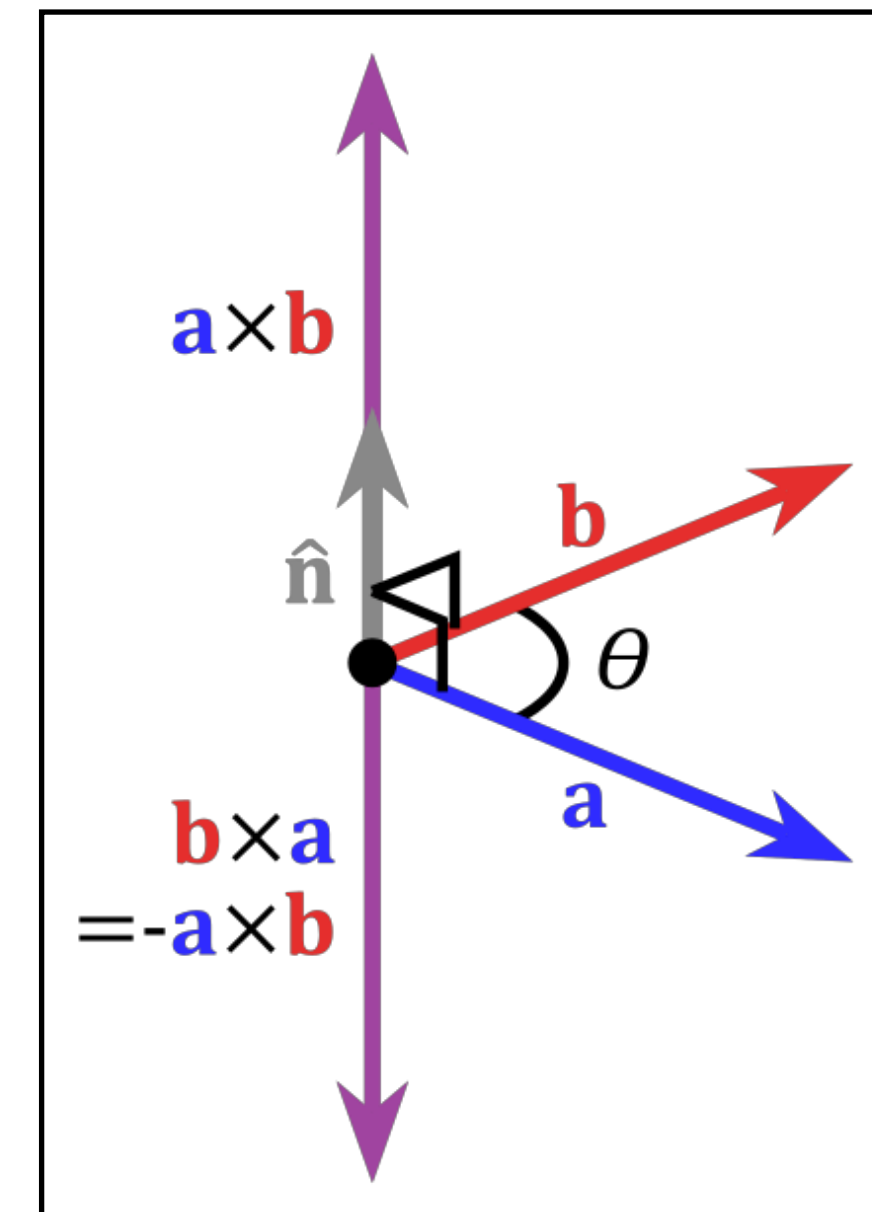
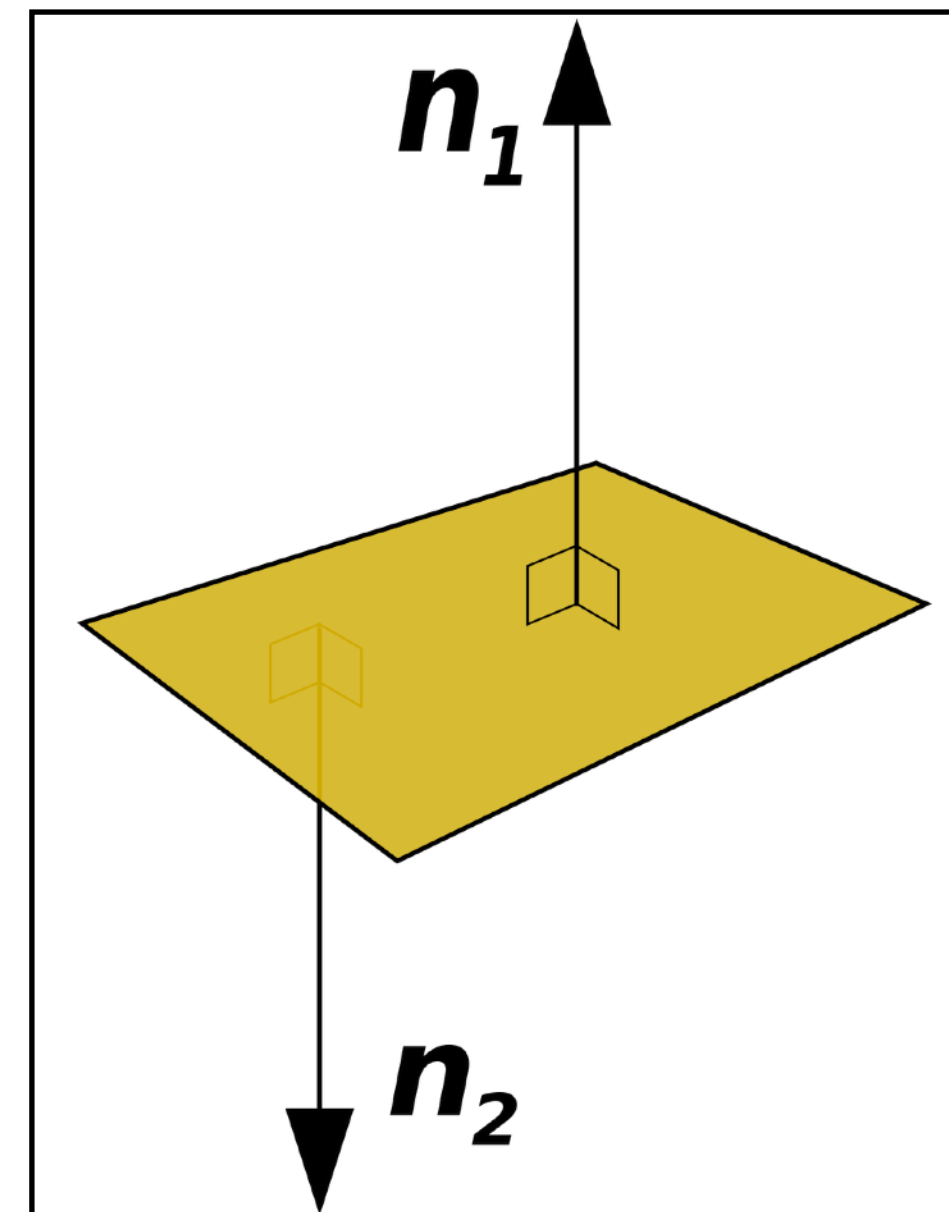
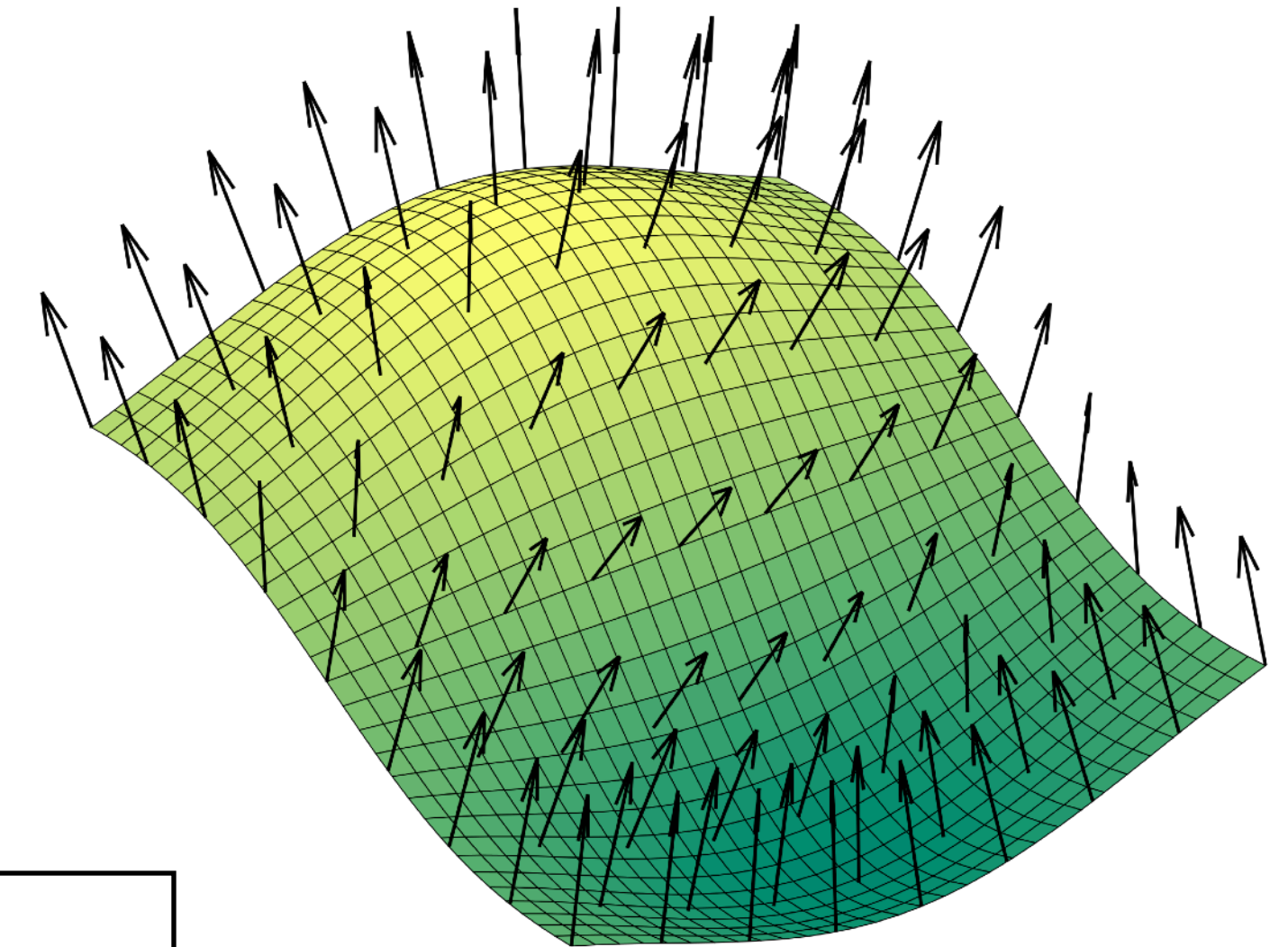
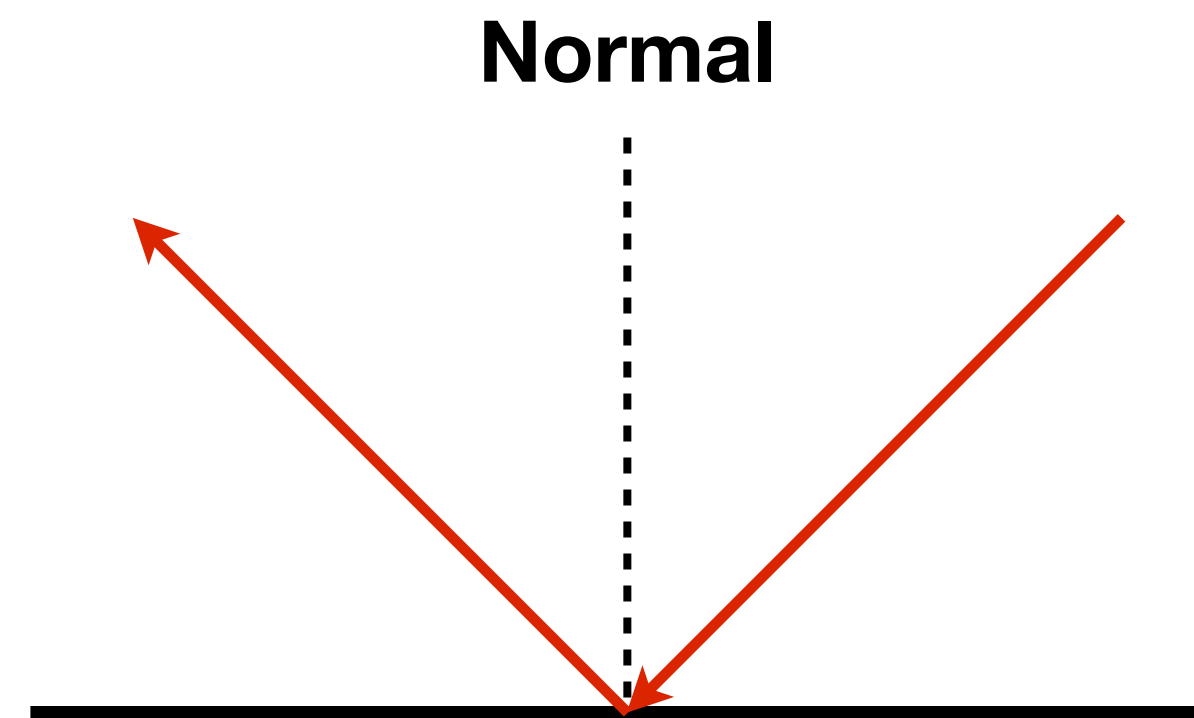
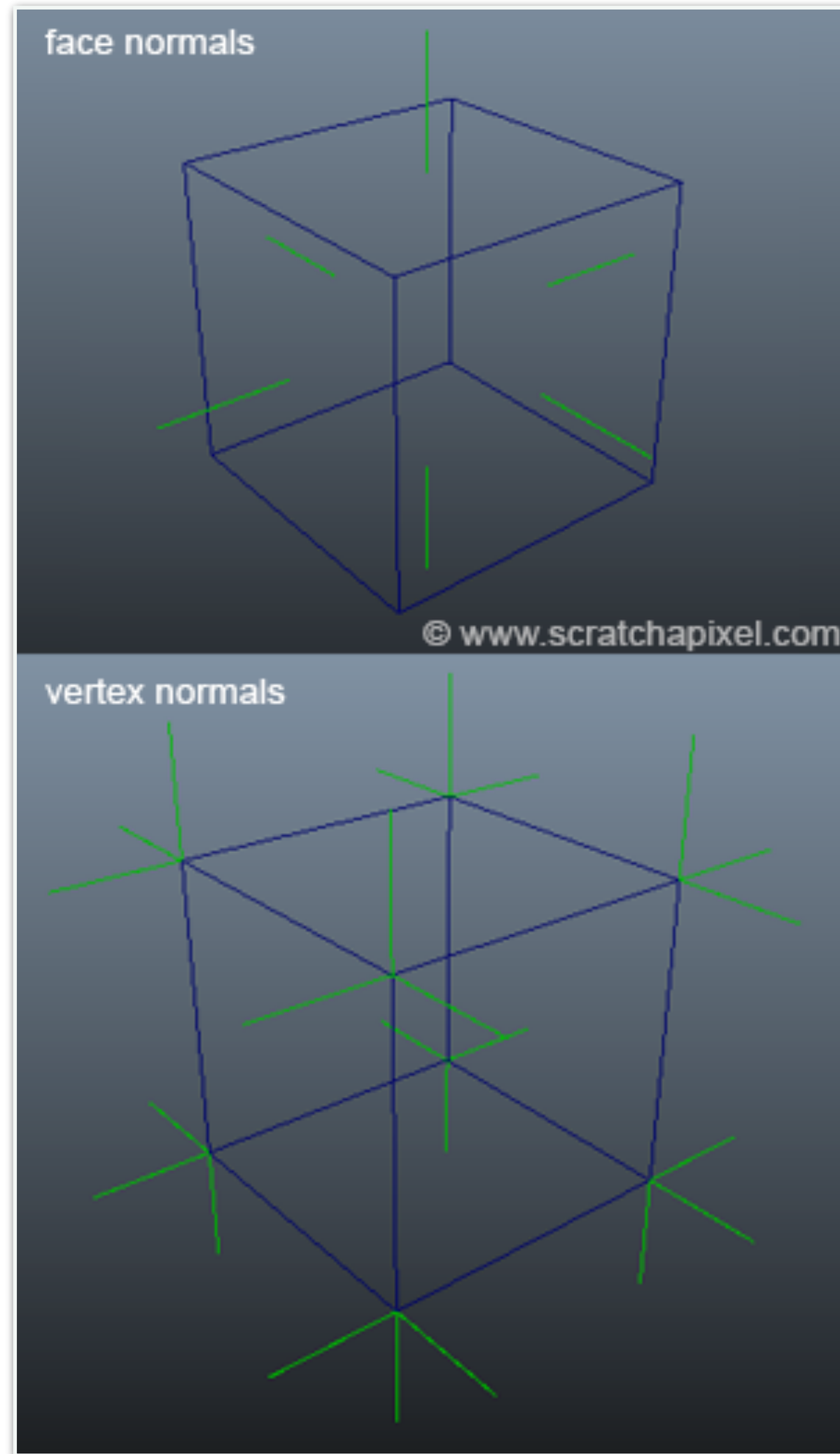
Mesh



Mesh

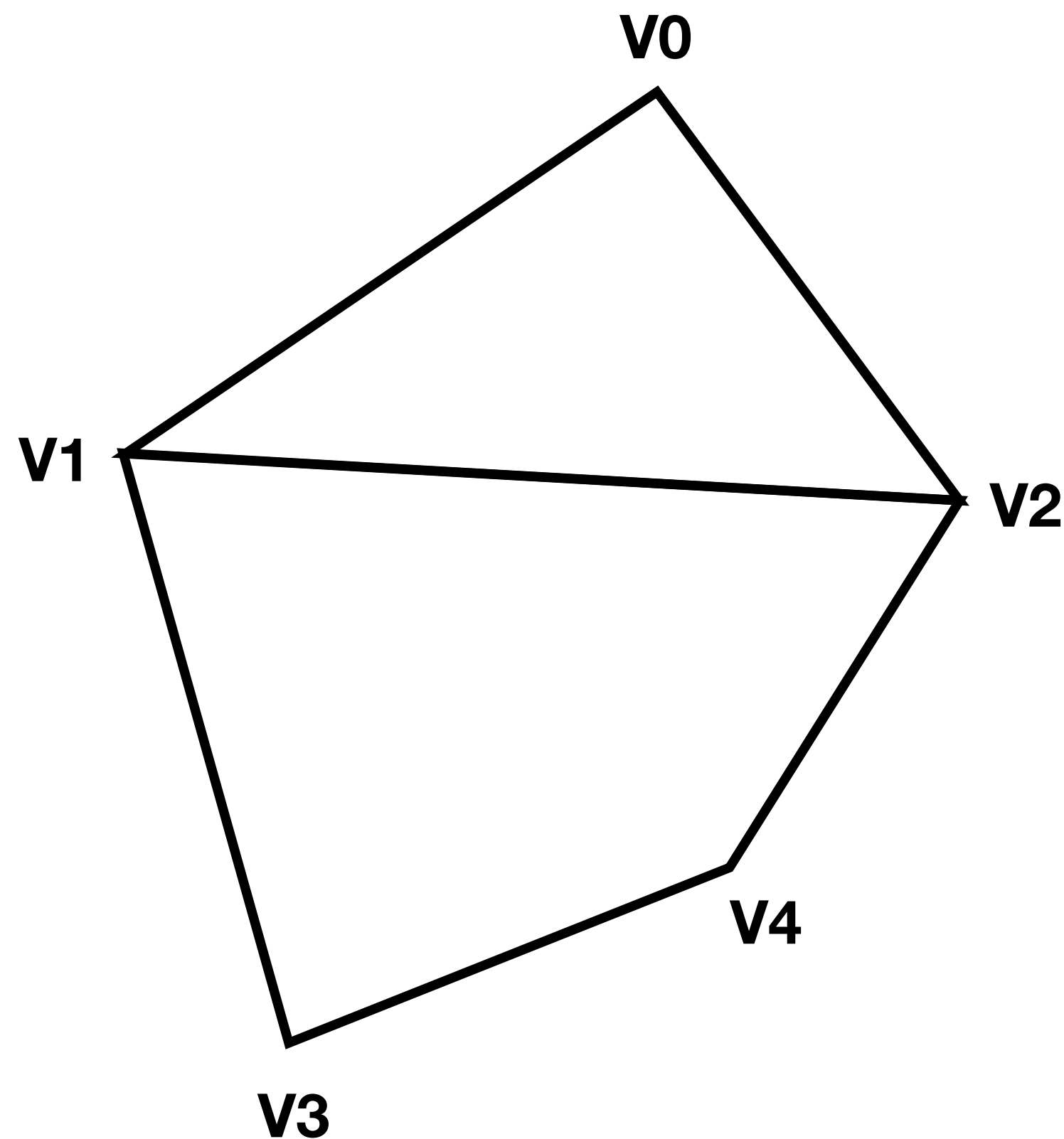


Normals



Calculating the normal of a face by taking the **cross-product** of two vectors on the face.

General Idea of Representing a Mesh



Encoding the coordinates of each vertex

V0 {1, 1, 2}, V1 {-1, 2, 1}, V2 {0, 1, 0}, V3 {-1, -1, 2}, V4 {1, -1, 2}

Encoding vertices of each face

{V0, V2, V1}, {V1, V2, V4, V3}

Could also have a
normal vector list and
various other primitives

RenderMan-like Interface

of vertices
in each face

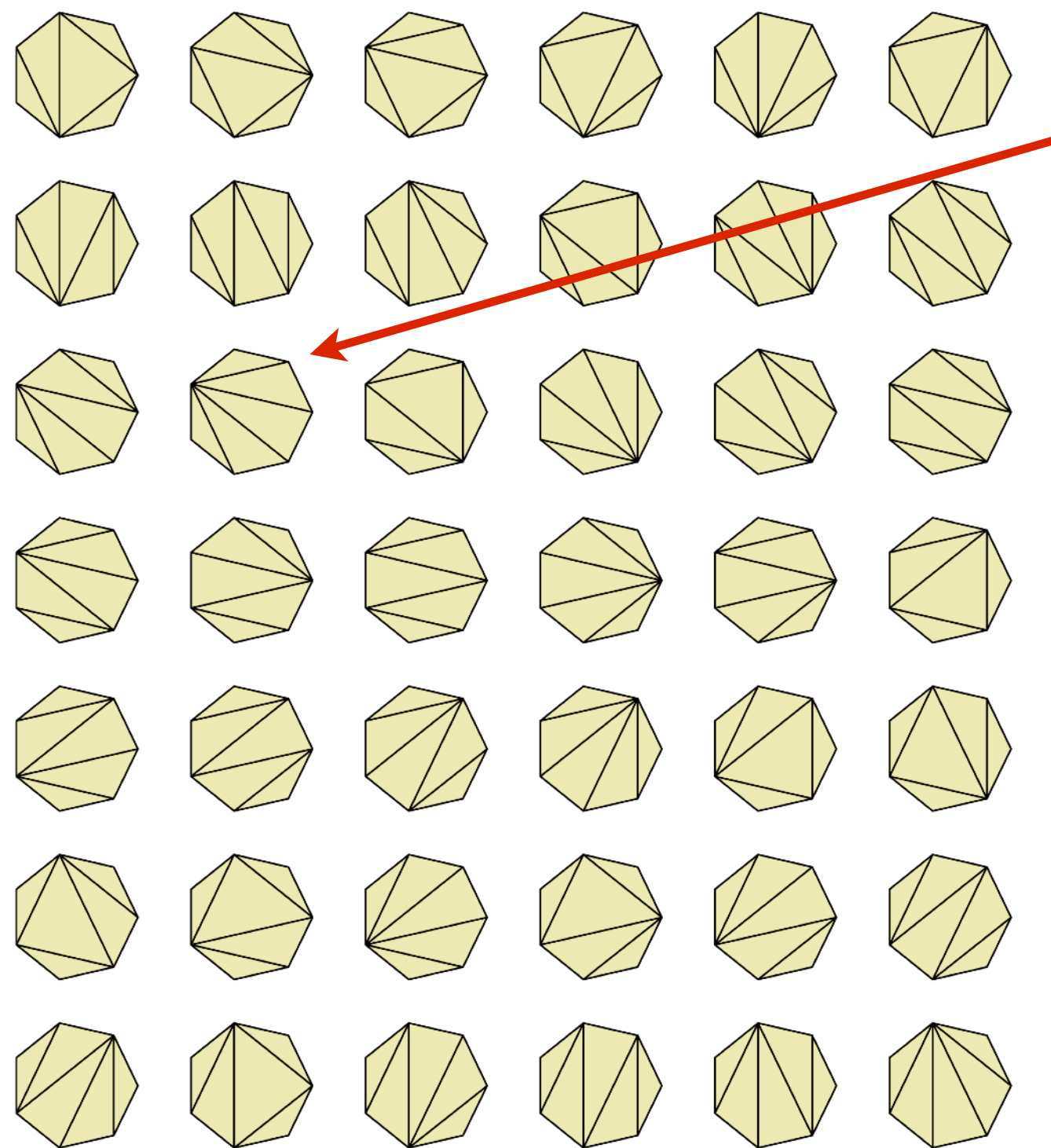
Vertex Index Array

PointsPolygon [3, 4] [0, 2, 1, 1, 2, 4, 3] "P"
[1, 1, 2, -1, 2, 1, 0, 1, 0, -1, -1, 2, 1, -1, 2]

Vertex Coordinates Array

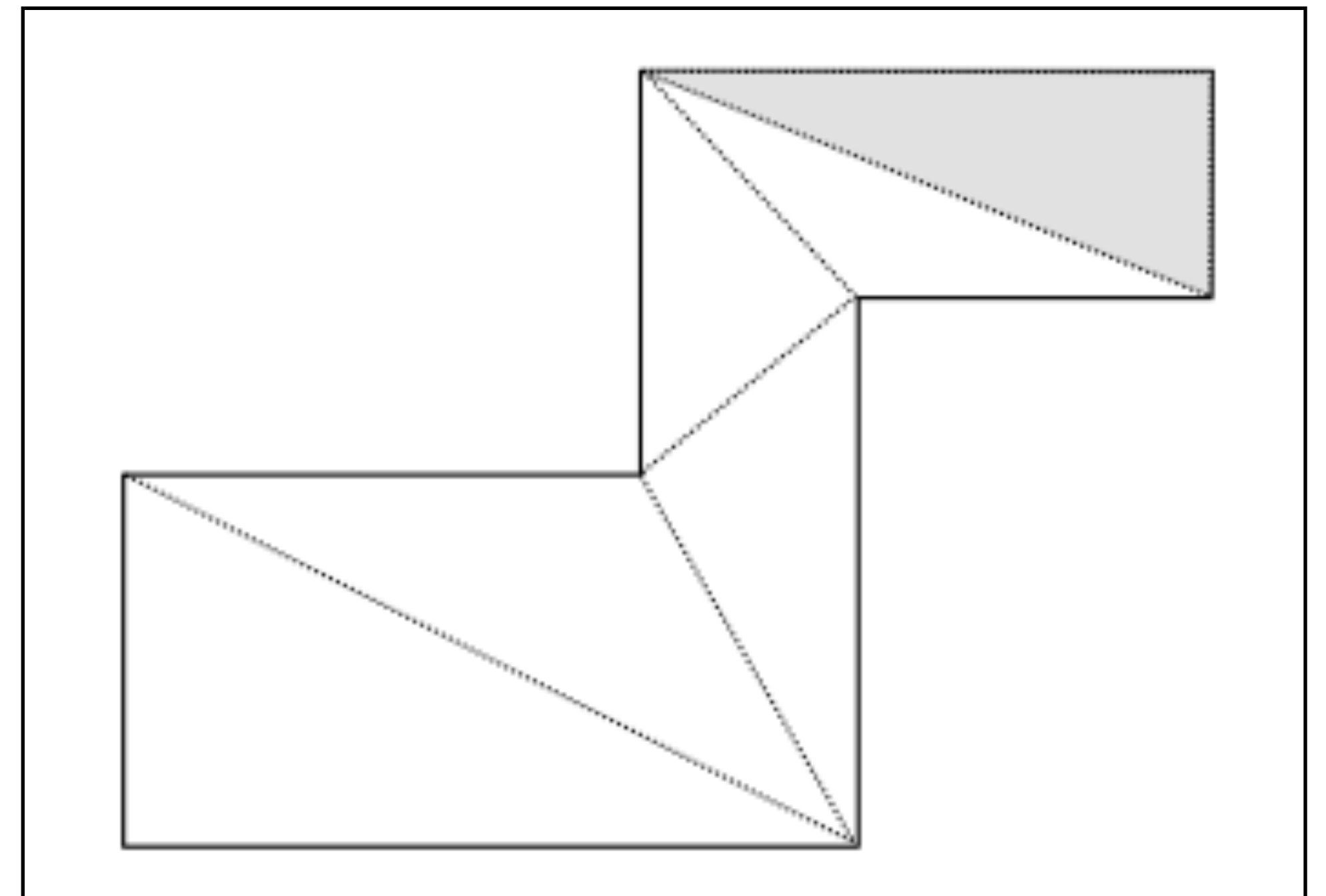
Polygon Triangulation

- Converting a polygon into a set of triangles.
 - ▷ Why? It usually allows more efficient implementations to have just one kind of primitive.



**A "fan triangulation"
that converts a convex
N-gon into N-2 faces.**

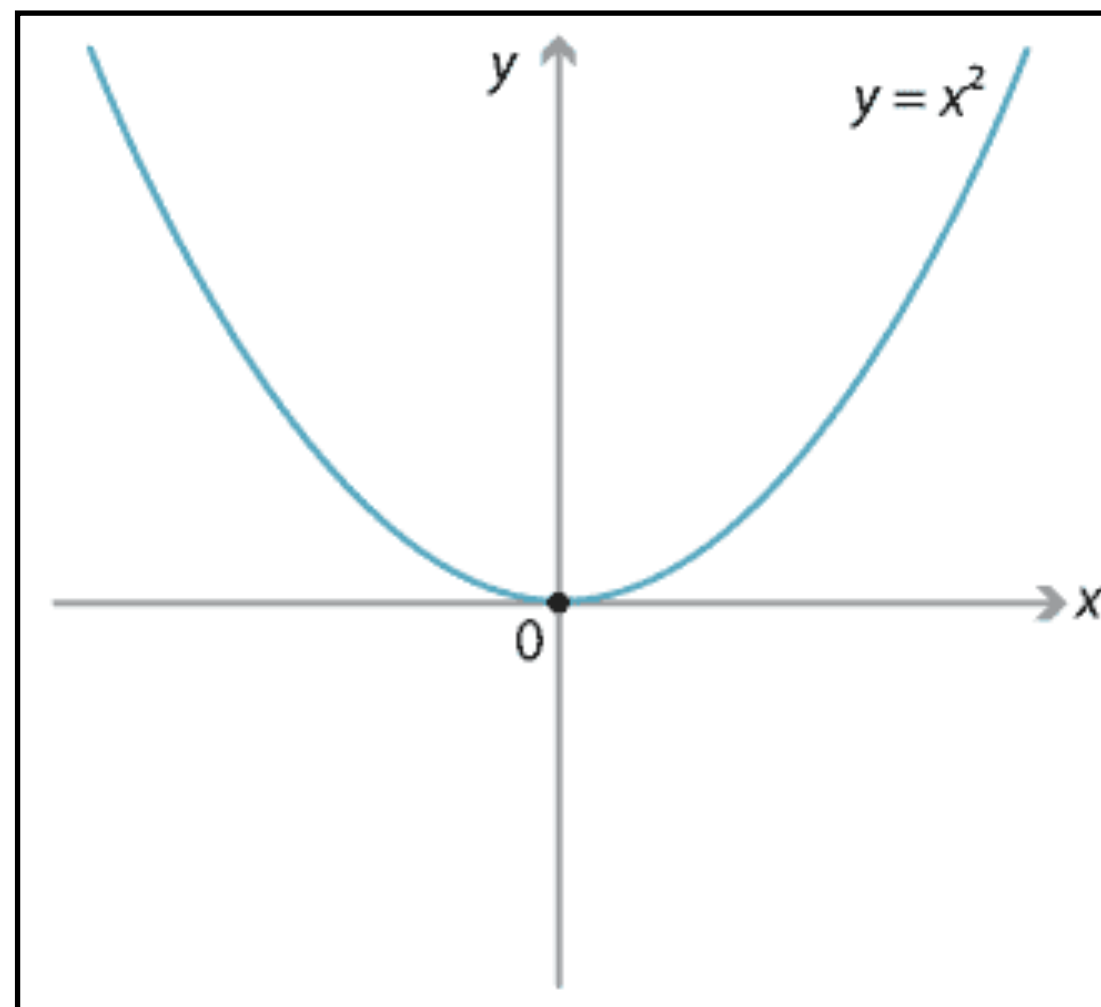
**Ear-clipping
triangulation.**



Parametric Curve

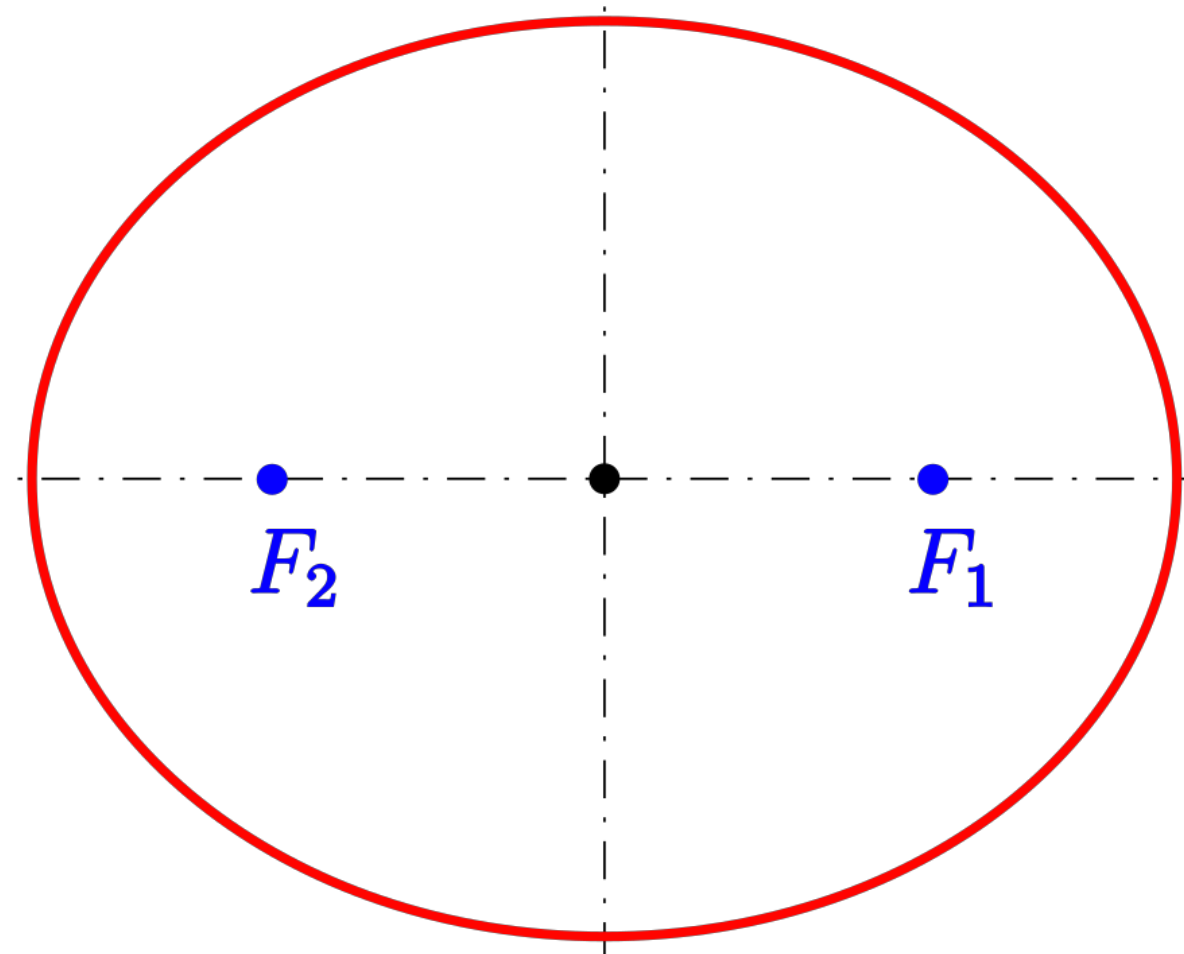
- Parametric curve: defining a curve by evaluating a set of *parametric equations*. Each equation has 1 parameter. $\mathbb{R} \rightarrow \mathbb{R}^2$.

Parabola ($y = x^2$)



$$\begin{aligned}x &= t \\ y &= t^2\end{aligned}$$

Ellipse ($x^2/a^2 + y^2/b^2 = 1$)



$$\begin{aligned}x &= a \cos(t) \\ y &= b \sin(t)\end{aligned}$$

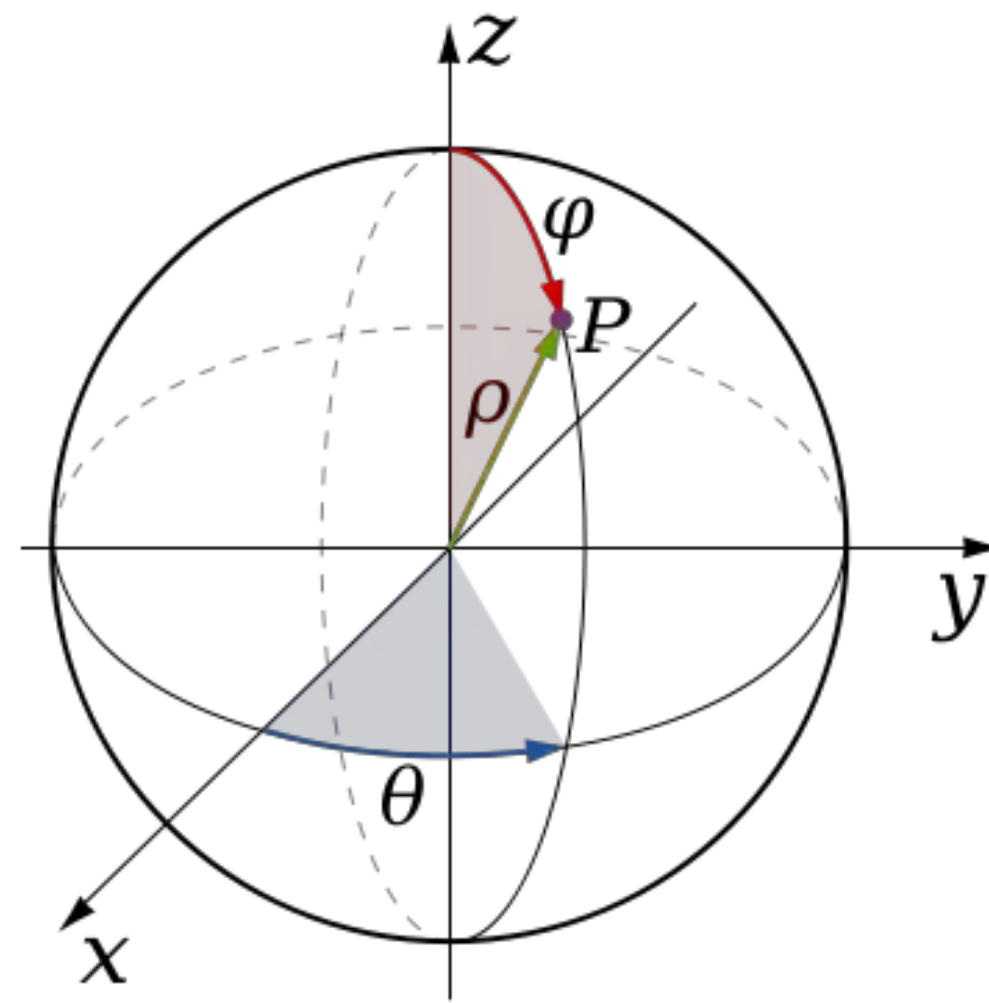
To define a curve, the parametric equations have **1 parameter** (t).

Parametric Surface

- Parametric surface: defining a surface by evaluating a set of *parametric equations*. Each equation have 2 parameters. $\mathbb{R}^2 \rightarrow \mathbb{R}^3$.

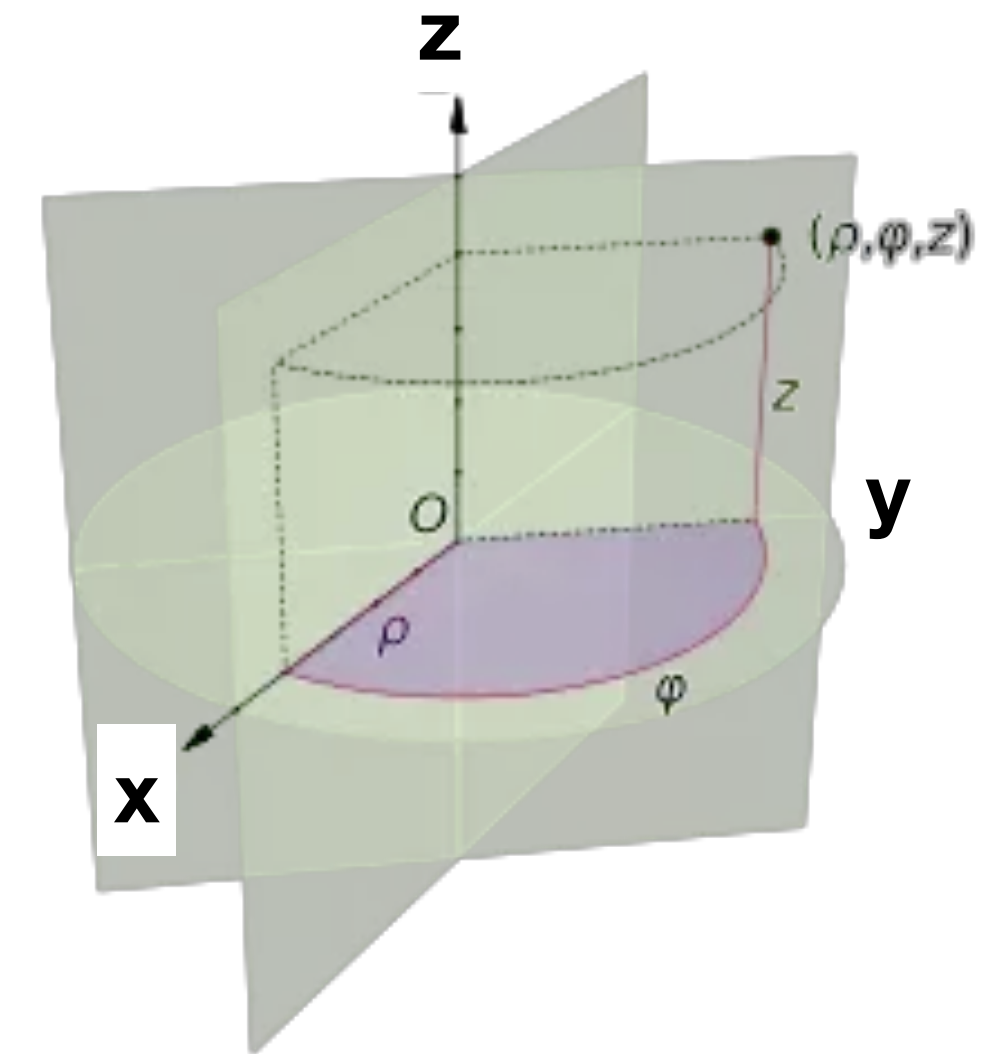
Spherical surface
with a fixed radius ρ

$$\begin{aligned}x &= \rho \sin \phi \cos \theta \\y &= \rho \sin \phi \sin \theta \\z &= \rho \cos \phi\end{aligned}$$



Cylindrical surface
with a fixed radius ρ

$$\begin{aligned}x &= \rho \cos \phi \\y &= \rho \sin \phi \\z &= z\end{aligned}$$

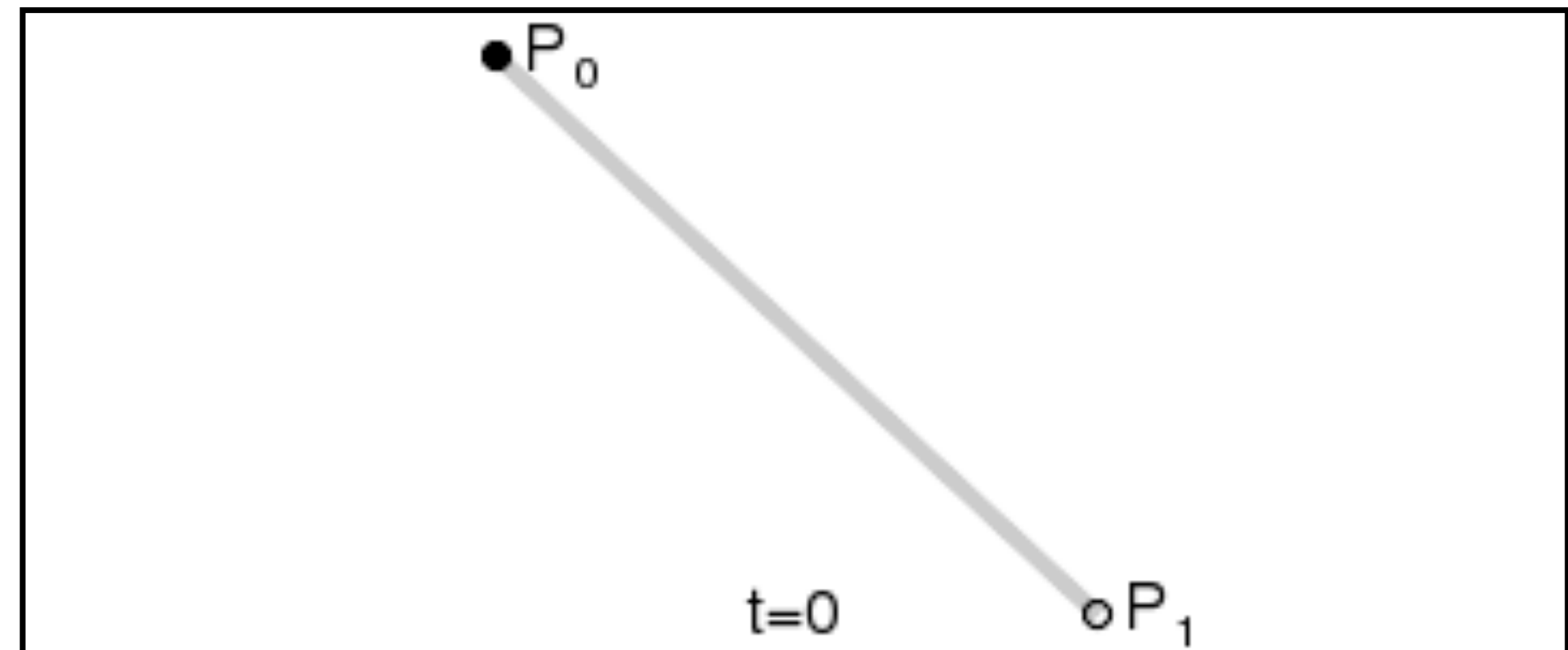
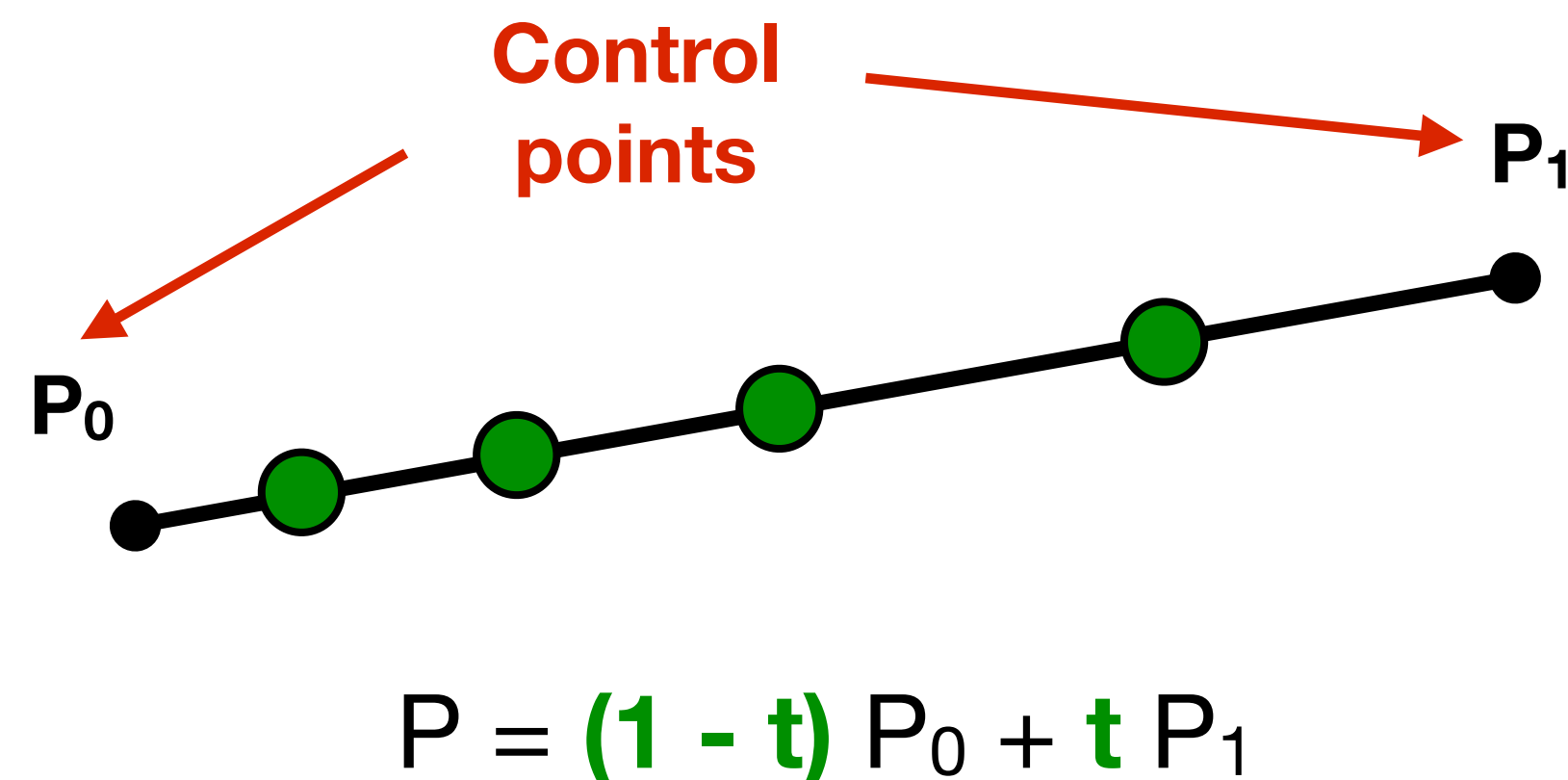


To define a surface, the parametric equations have **2 parameters**.

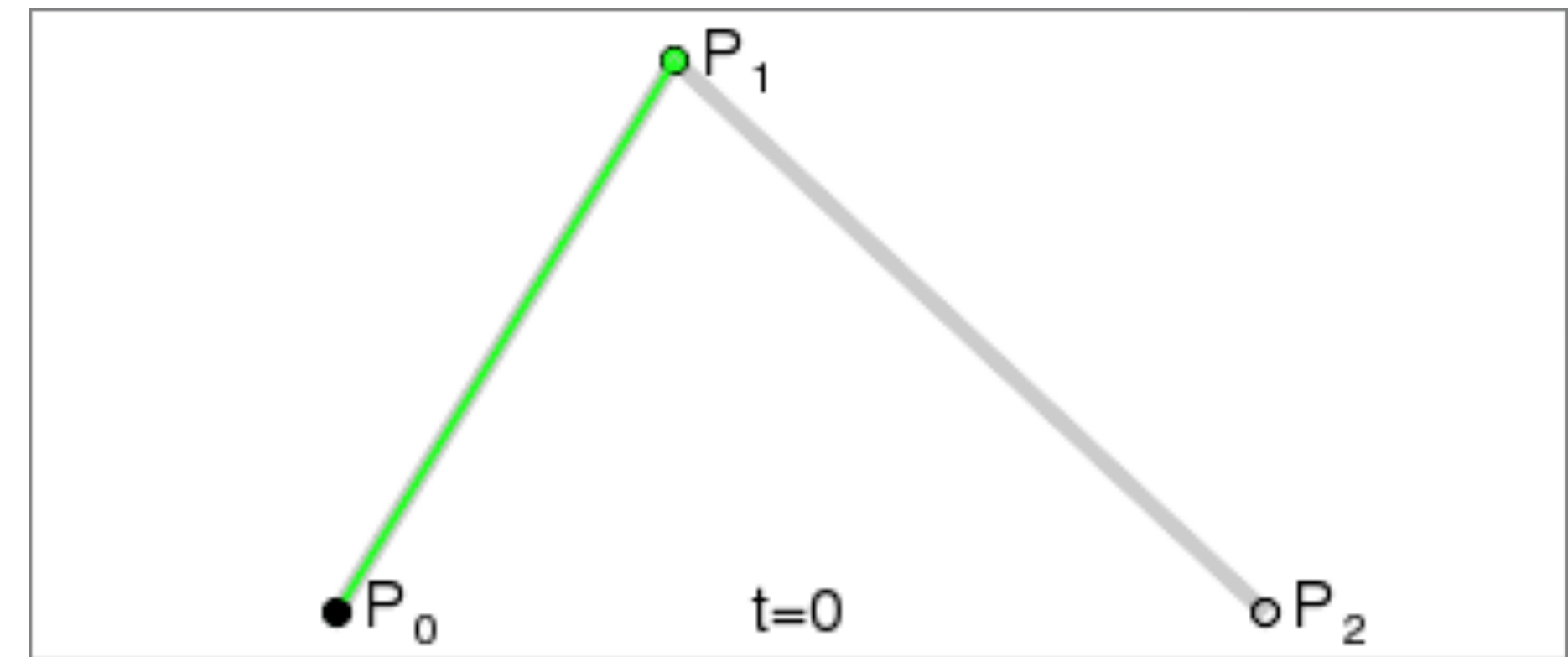
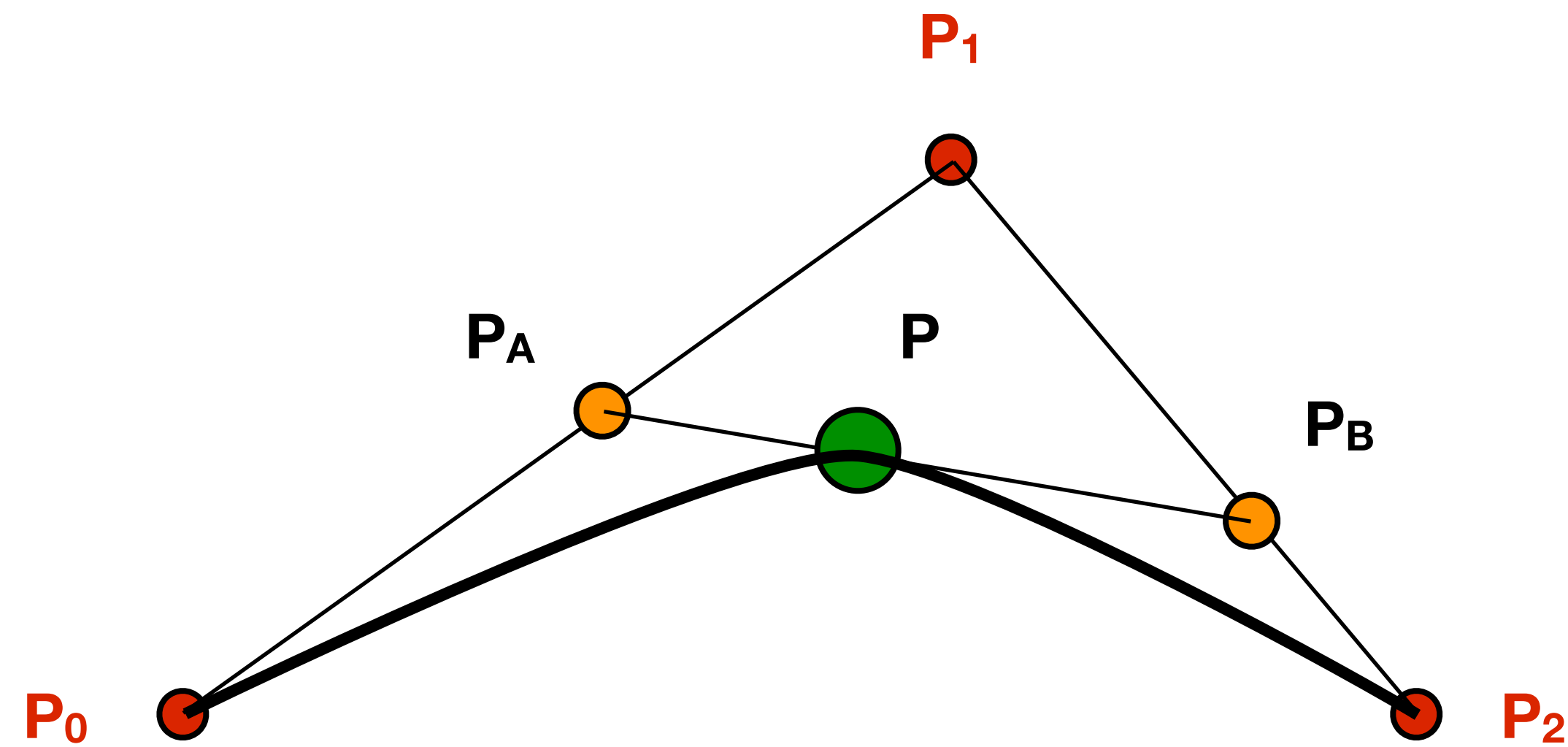
Bezier Curve

- ▶ Bezier curve is a kind of parametric curve expressed as a weighted sum of a set of *control points*.
- ▷ Weights are given by the Bernstein polynomial. Two end control points are on the curve.

Linear Bezier curve



Bezier Curve



Quadratic bezier curve has 3 control points.
The curve changes with the control points.

Quadratic Bezier curve

$$P_A = (1 - t) P_0 + t P_1$$

$$P_B = (1 - t) P_1 + t P_2$$

$$P = (1 - t) P_A + t P_B$$



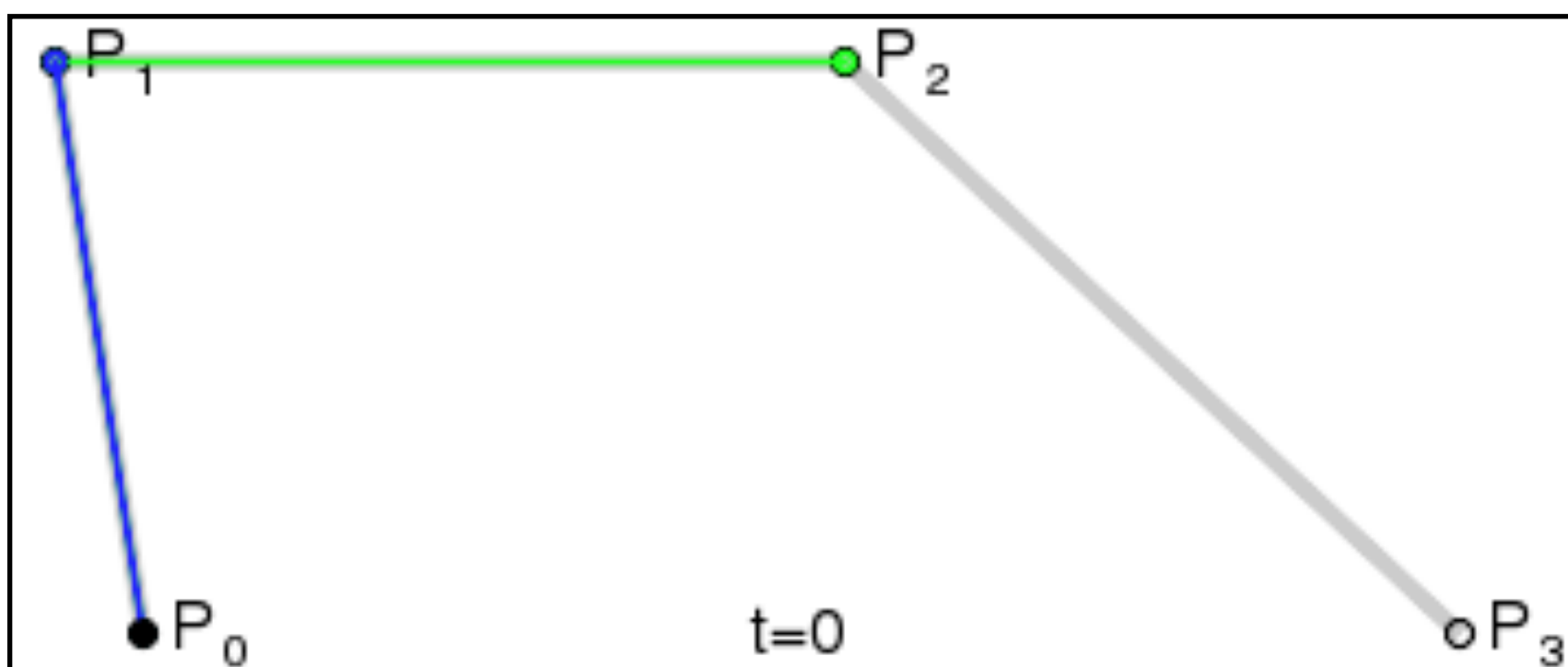
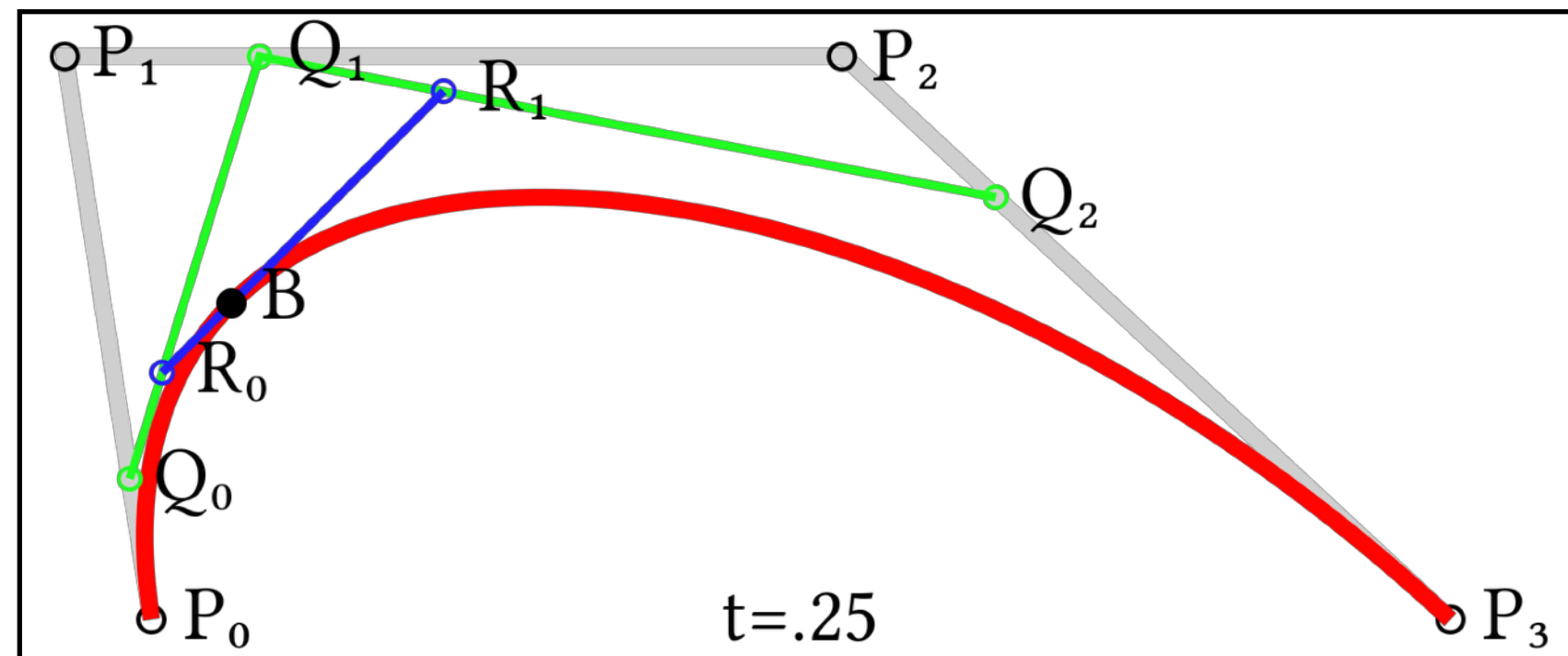
$(1 - t)^2 + 2t(1-t) + t^2 = 1$. P is a weighted sum of P_0 , P_1 , P_2 .

$$P = (1 - t)^2 P_0 + 2t(1-t) P_1 + t^2 P_2$$

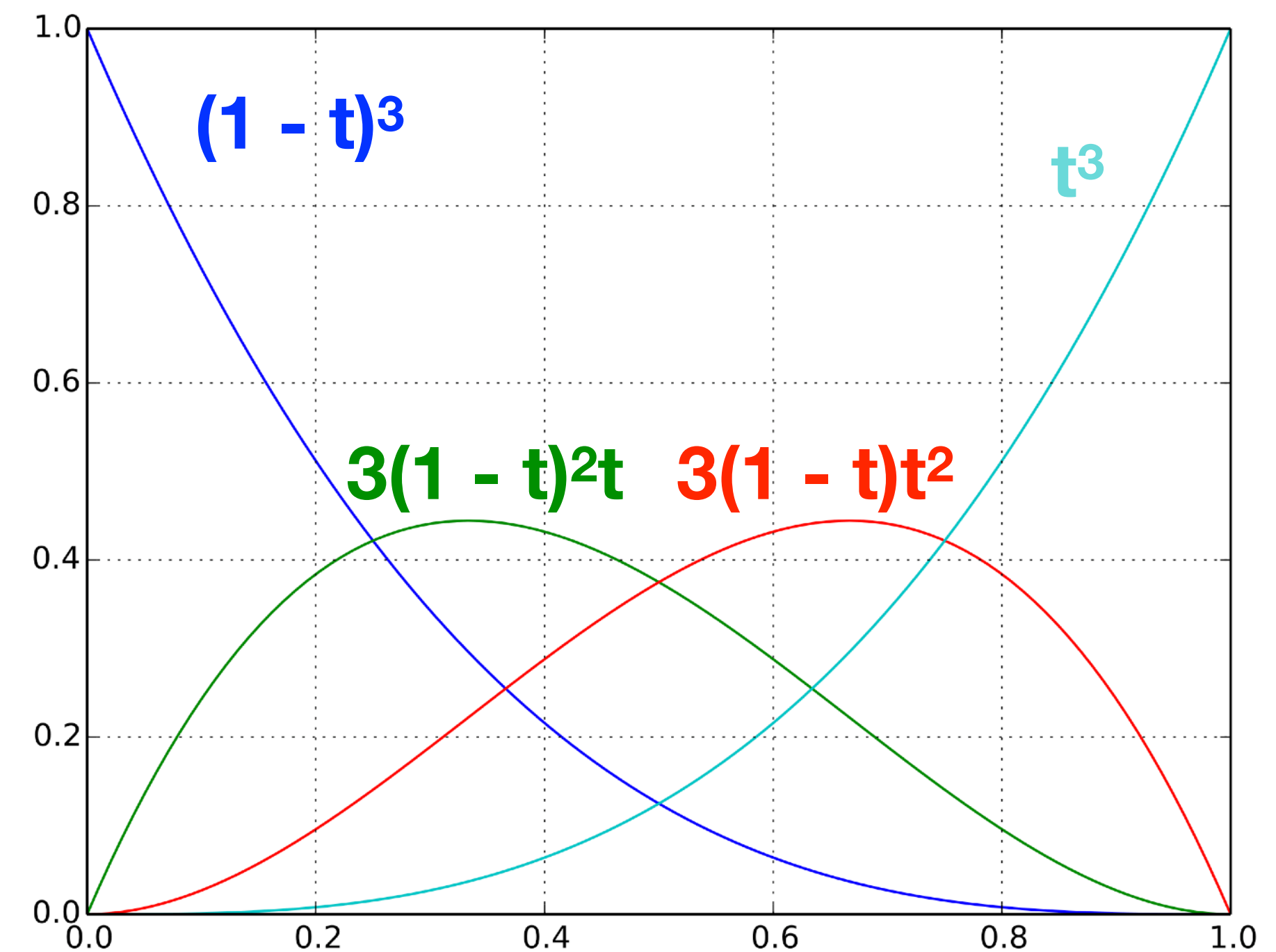
Bezier Curve

- A bezier curve of order n has (n+1) control points.

Third-order Bezier curve



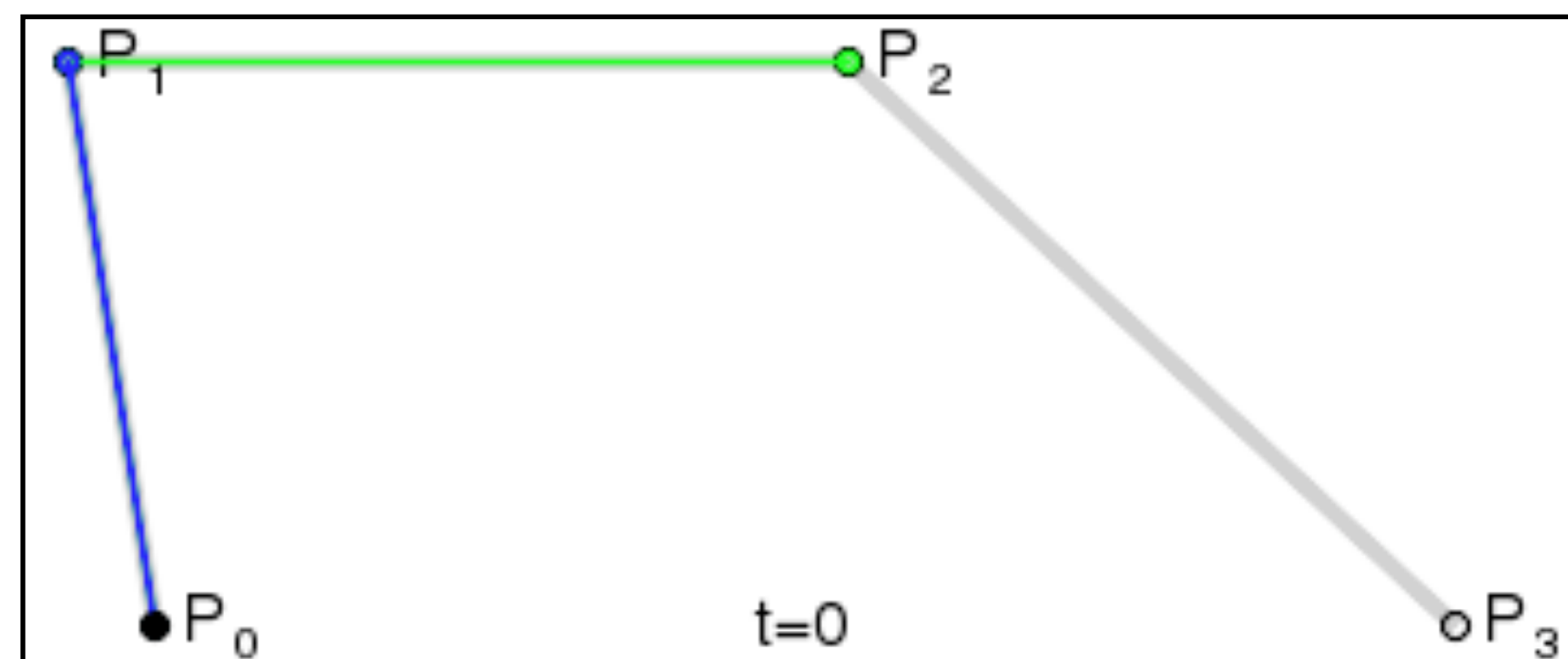
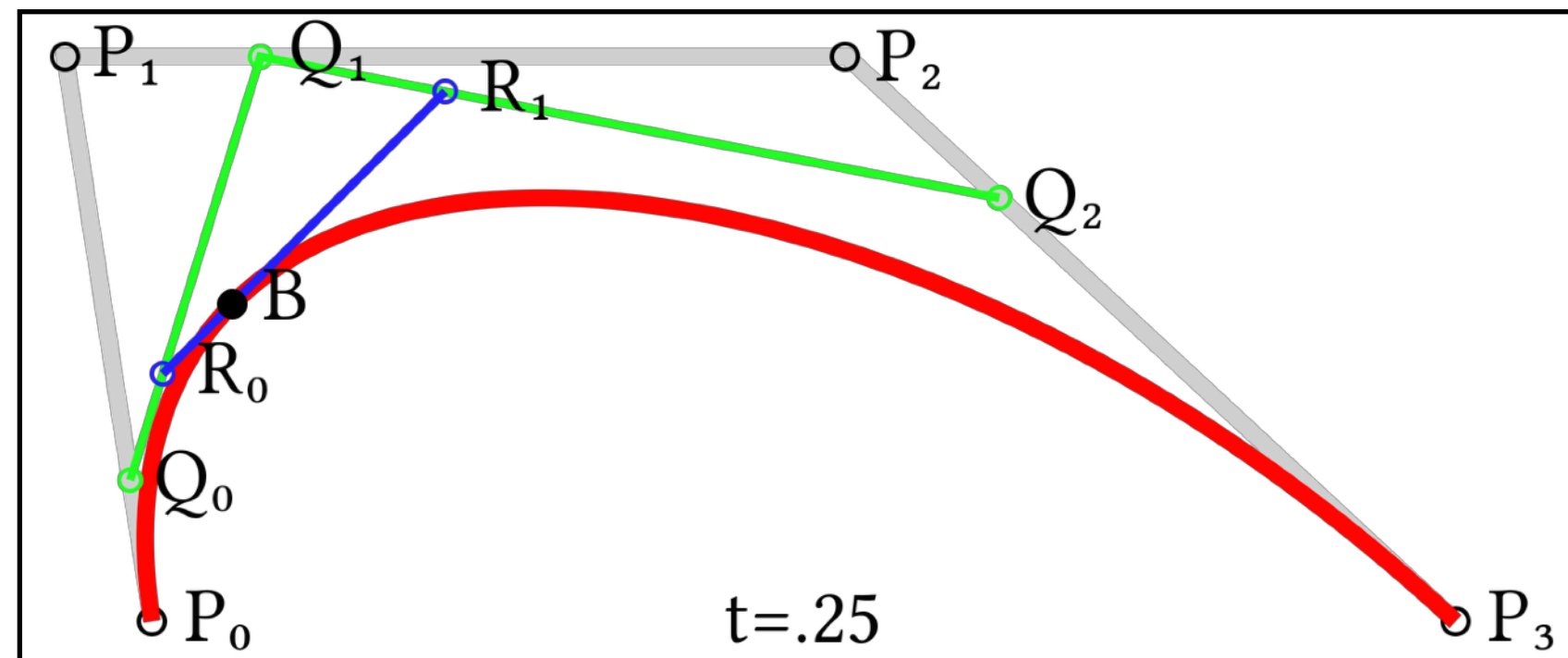
$$P = (1 - t)^3 P_0 + 3(1 - t)^2 t P_1 + 3(1 - t) t^2 P_2 + t^3 P_3$$



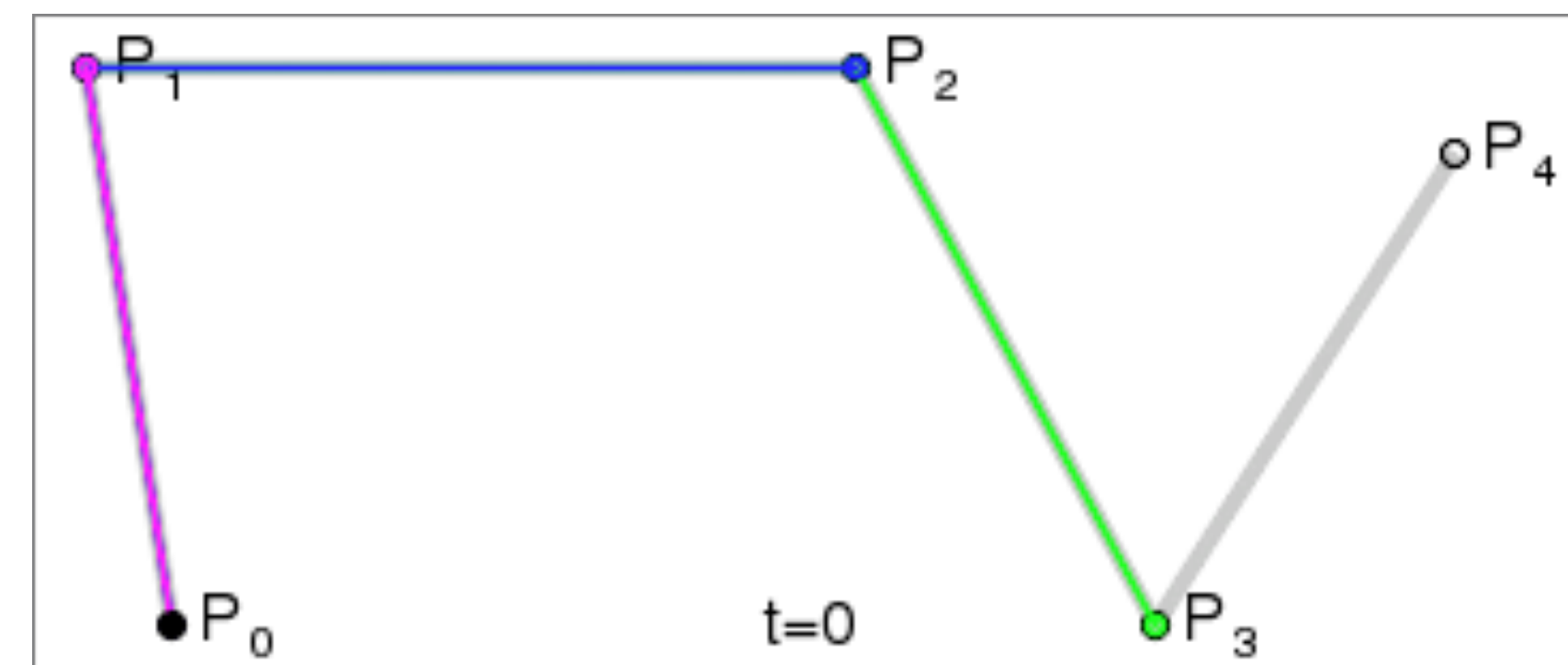
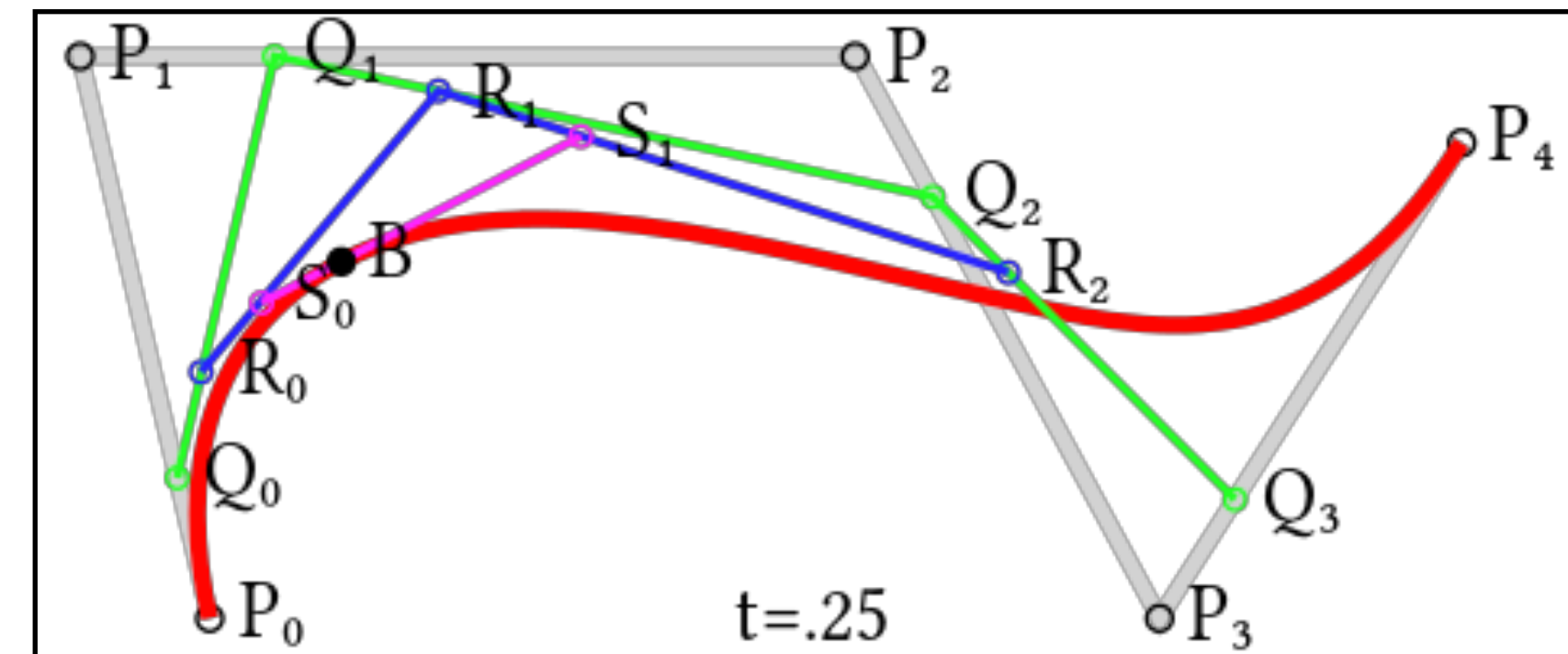
Bezier Curve

- ▶ A bezier curve of order n has $(n+1)$ control points.

Third-order Bezier curve



Fourth-order Bezier curve



Bezier Curve

Weights. Sum of weights is 1. So any point is a weighted sum of all the control points.

$$B(t) = \sum_{i=0}^n b_{i,n}(t) P_i$$

Control points

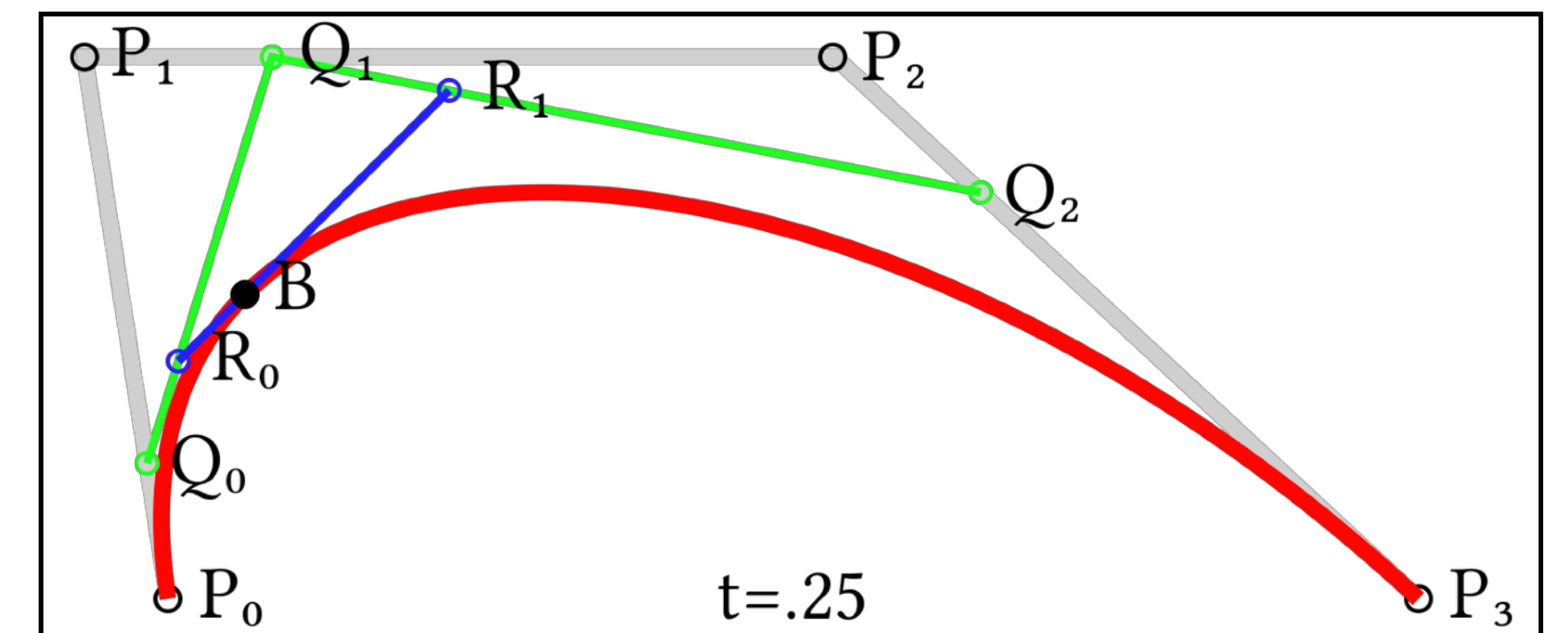
Parameter

Basis functions:
Bernstein
polynomials

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Binomial
coefficients

$$\binom{n}{t} = \frac{n!}{t!(n-t)!}$$

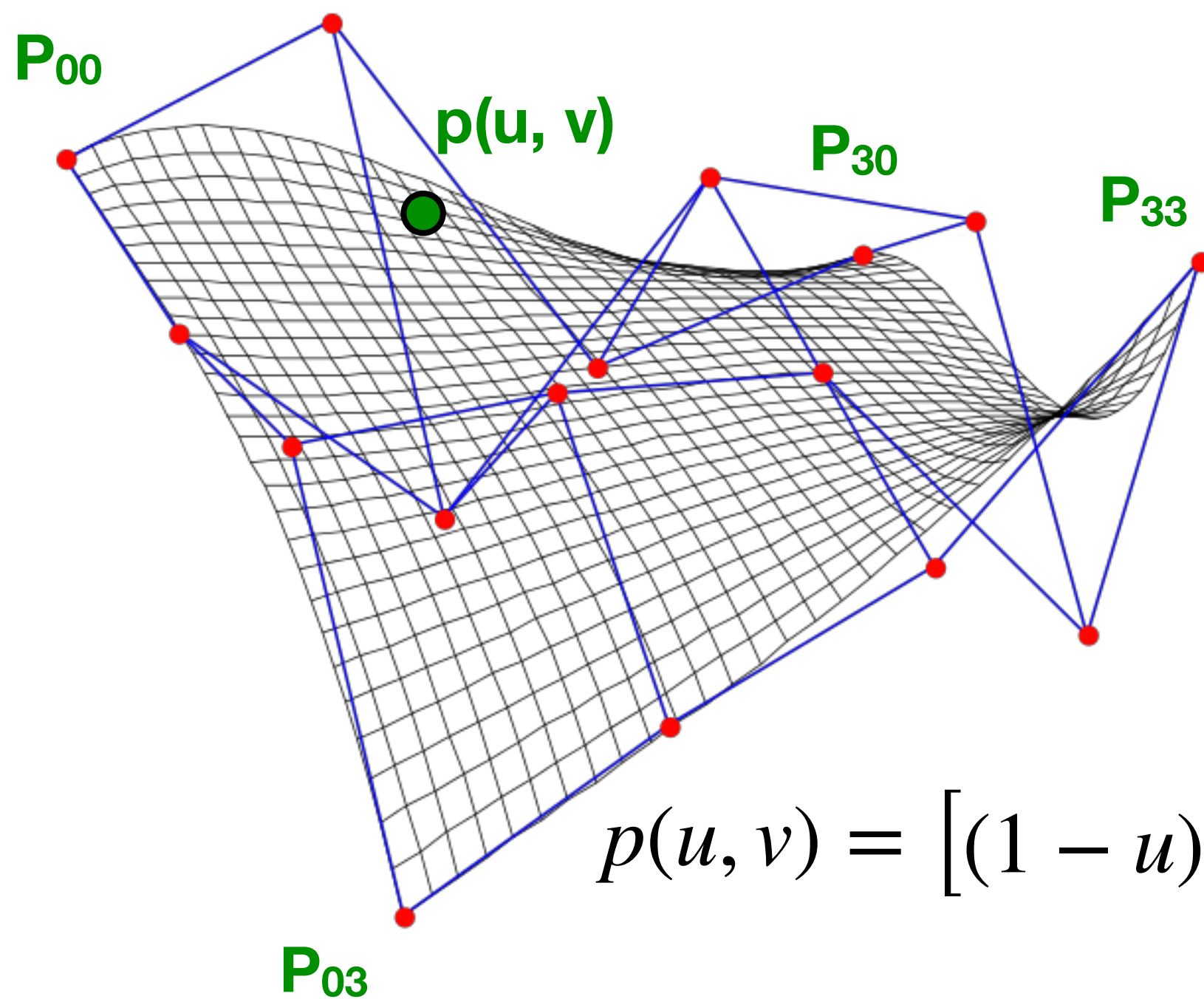


The first and last control points are also the end points of the generate curve.

Bezier Surface

- A (m, n) bezier surface defines a surface where every point is a weighted sum of (m+1) x (n+1) control points.

A (3, 3) Bezier surface/patch



$$p(u, v)$$

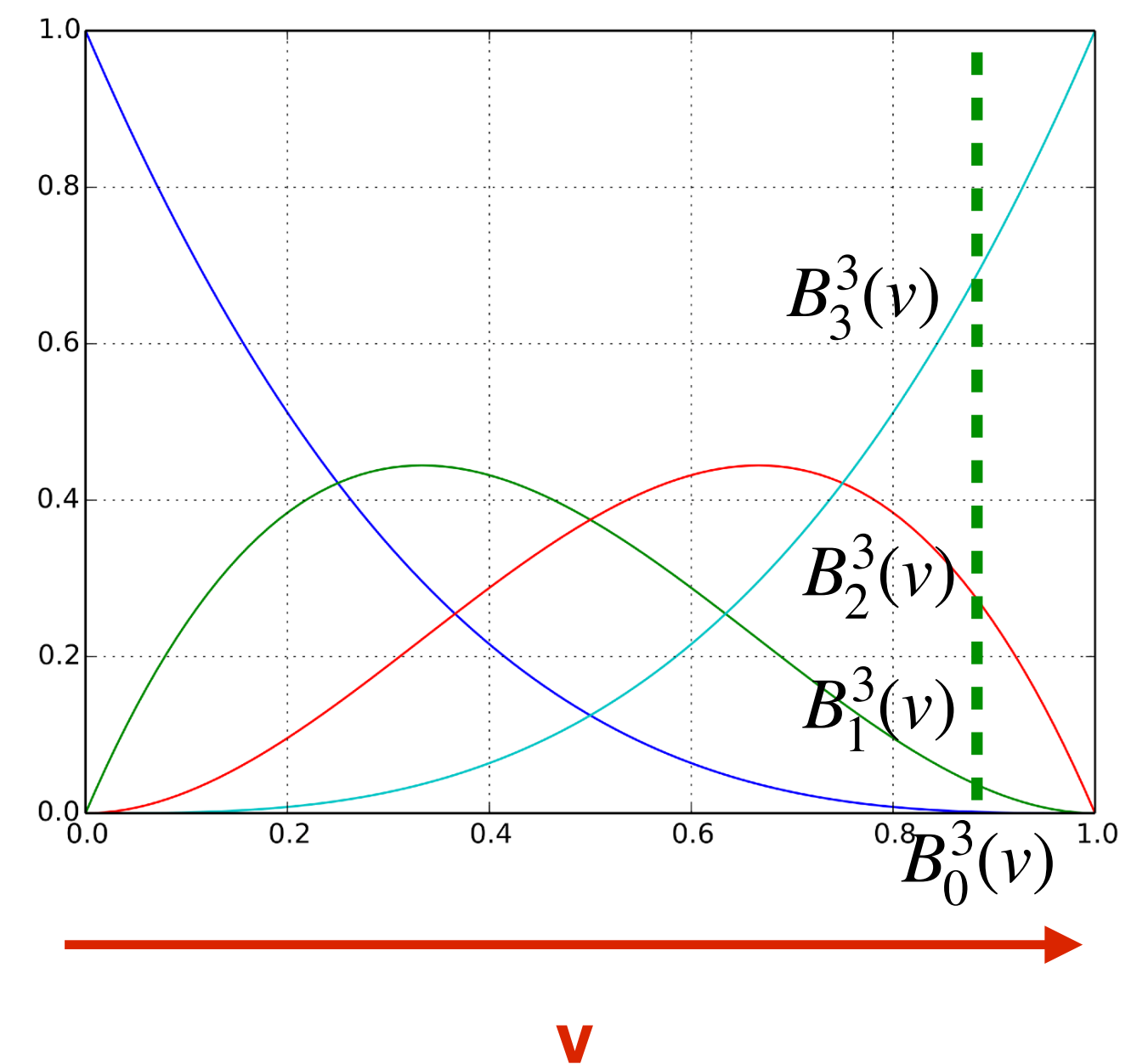
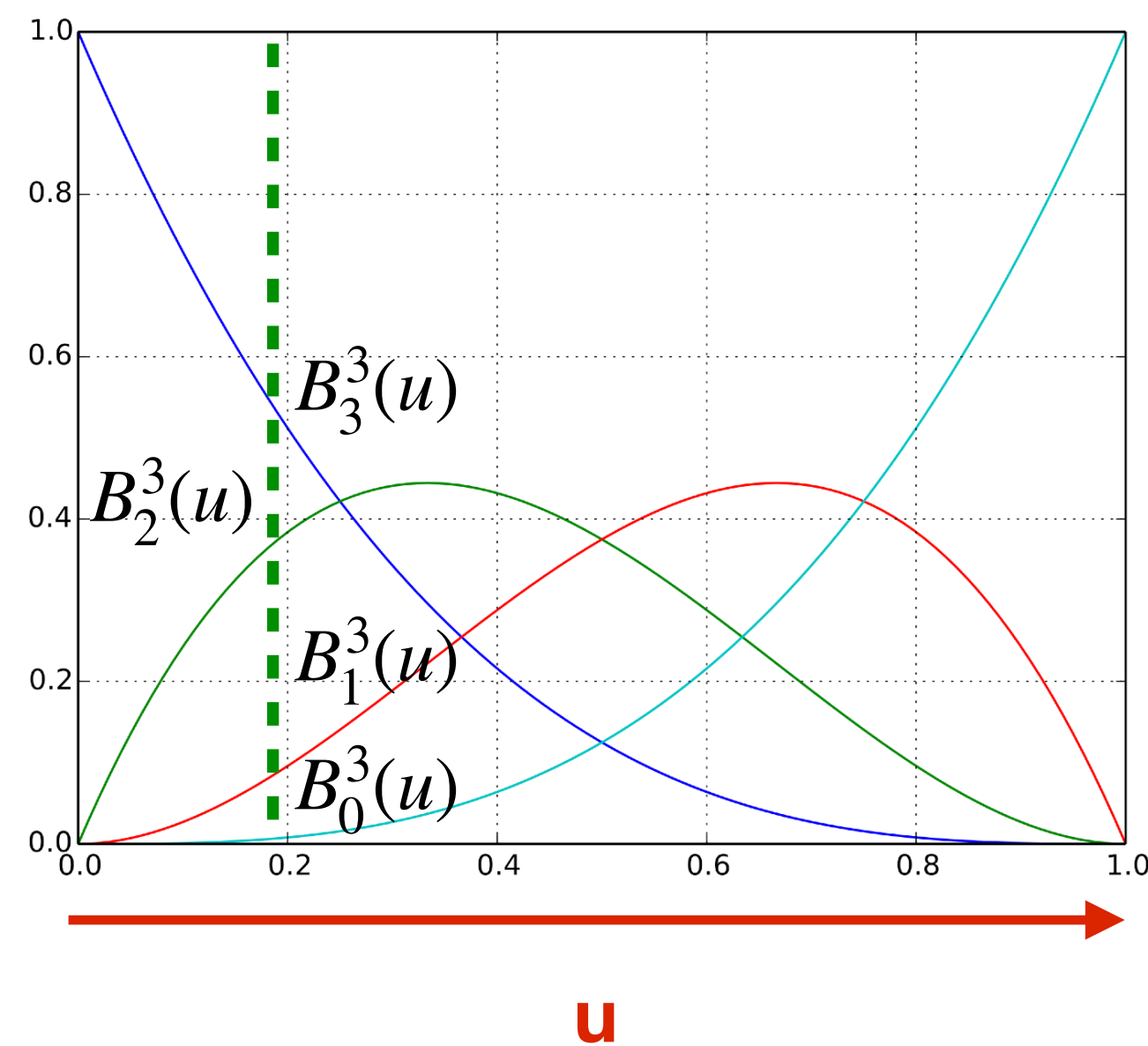
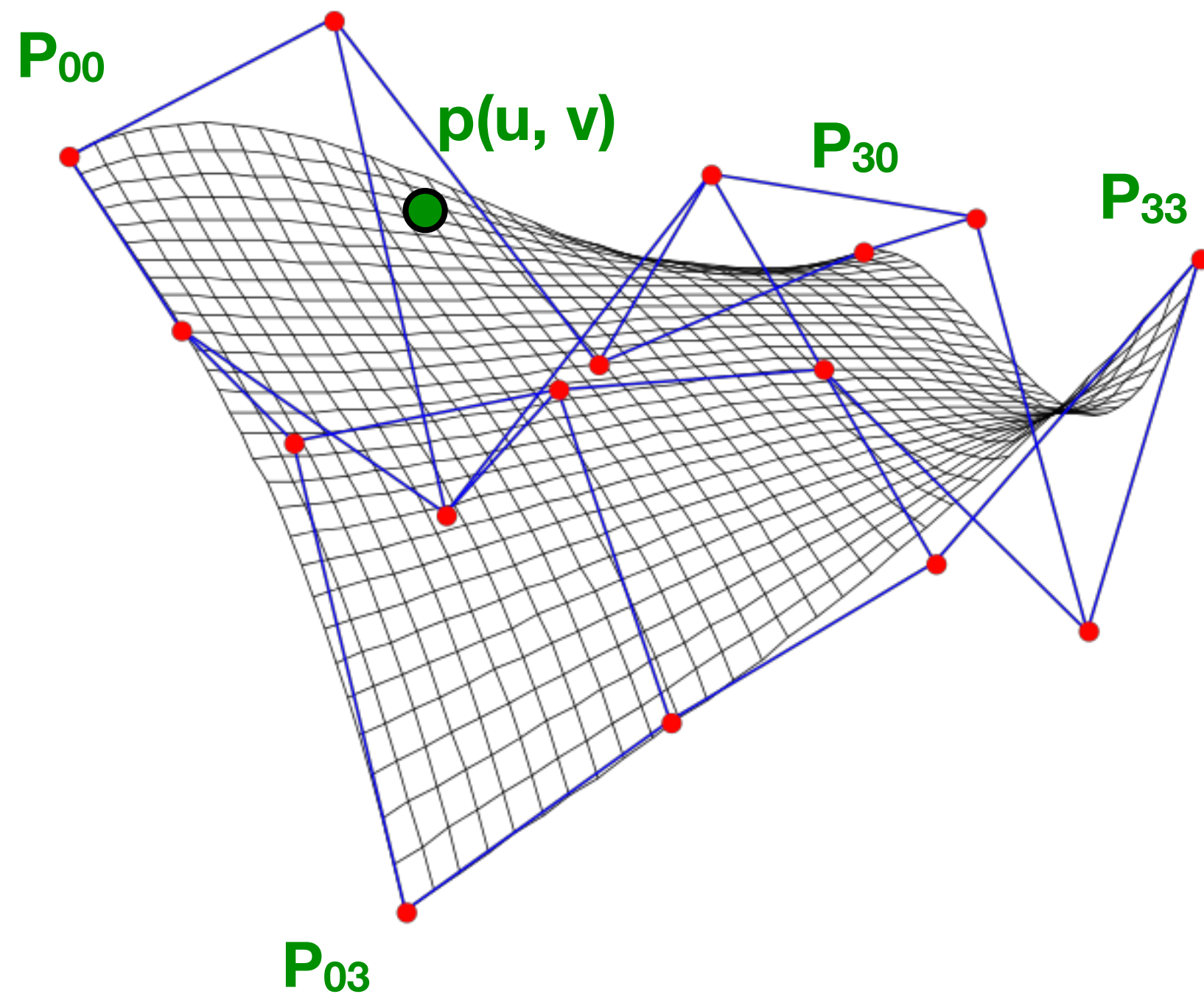
$$\begin{aligned} &= B_0^3(u)B_0^3(v)P_{00} + B_1^3(u)B_0^3(v)P_{10} + B_2^3(u)B_0^3(v)P_{20} + B_3^3(u)B_0^3(v)P_{30} \\ &+ B_0^3(u)B_1^3(v)P_{01} + B_1^3(u)B_1^3(v)P_{11} + B_2^3(u)B_1^3(v)P_{21} + B_3^3(u)B_1^3(v)P_{31} \\ &+ B_0^3(u)B_2^3(v)P_{02} + B_1^3(u)B_2^3(v)P_{12} + B_2^3(u)B_2^3(v)P_{22} + B_3^3(u)B_2^3(v)P_{32} \\ &+ B_0^3(u)B_3^3(v)P_{03} + B_1^3(u)B_3^3(v)P_{13} + B_2^3(u)B_3^3(v)P_{23} + B_3^3(u)B_3^3(v)P_{33} \end{aligned}$$

$$p(u, v) = \begin{bmatrix} (1-u)^3 & 3u(1-u)^2 & 3u^2(1-u) & u^3 \end{bmatrix} \times \begin{bmatrix} P_{00} & P_{01} & P_{02} & P_{03} \\ P_{10} & P_{11} & P_{12} & P_{13} \\ P_{20} & P_{21} & P_{22} & P_{23} \\ P_{30} & P_{31} & P_{32} & P_{33} \end{bmatrix} \times \begin{bmatrix} (1-v)^3 \\ 3v(1-v)^2 \\ 3v^2(1-v) \\ v^3 \end{bmatrix}$$

Bezier Surface

- A (m, n) bezier surface defines a surface where every point is a weighted sum of $(m+1) \times (n+1)$ control points.

A (3, 3) Bezier surface/patch



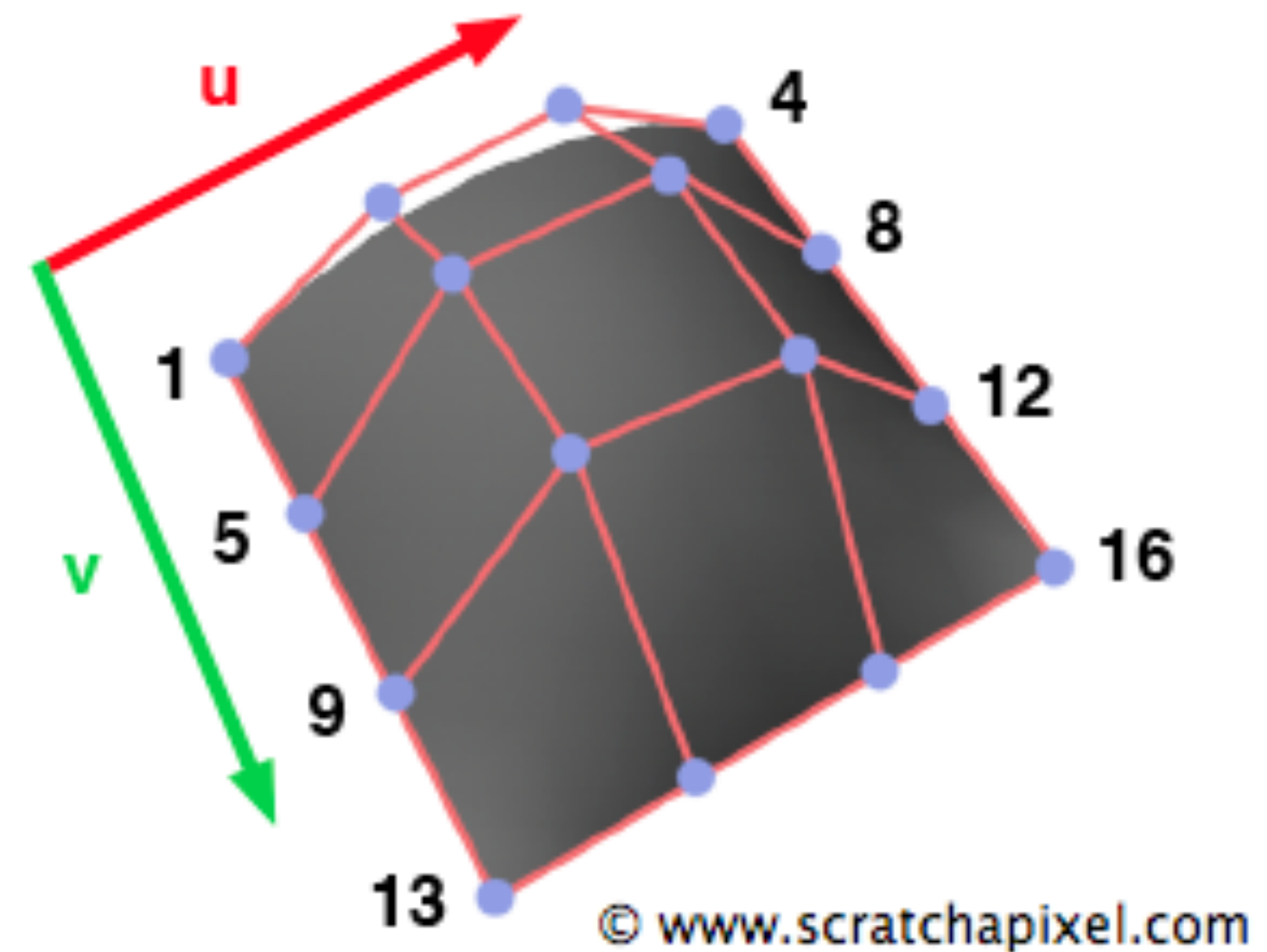
Bezier Surface

- ▶ A (m, n) bezier surface defines a surface where every point is a weighted sum of (m+1) x (n+1) control points.

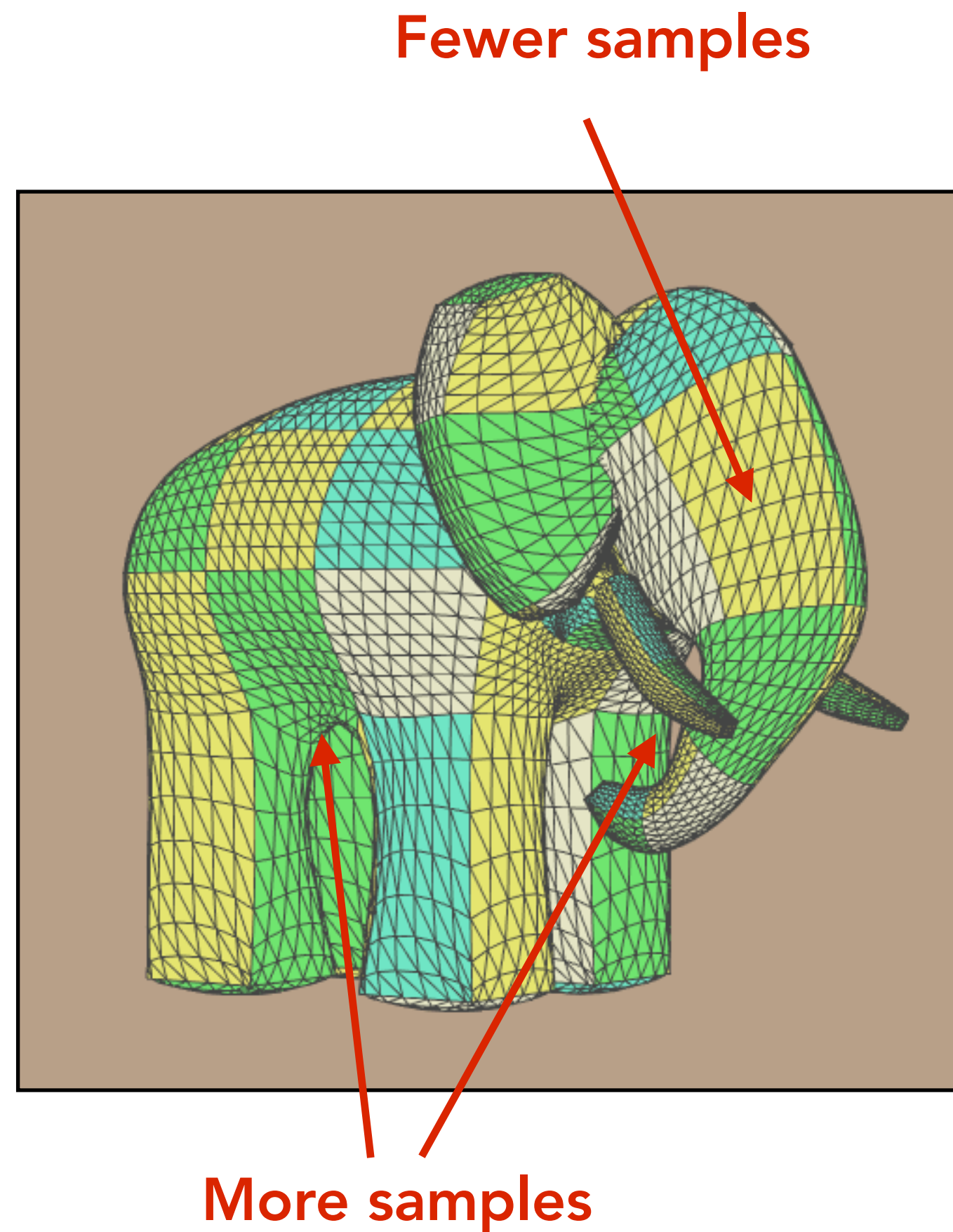
$$p(u, v) = \sum_{i=0}^n \sum_{j=0}^m B_i^n(u) B_j^m(v)$$

$$B_i^n(u) = \binom{n}{i} u^i (1 - u)^{n-i}$$

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}$$

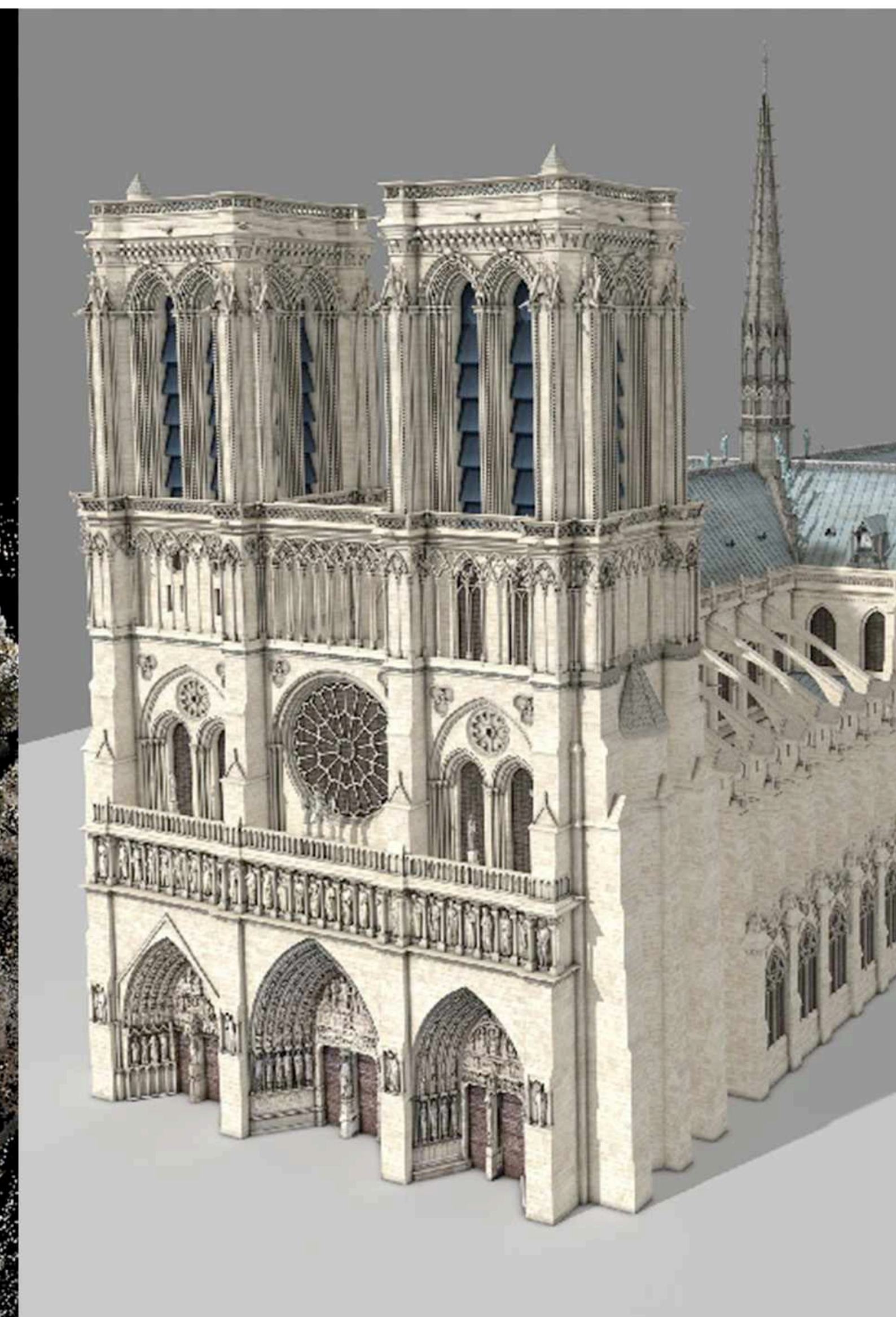
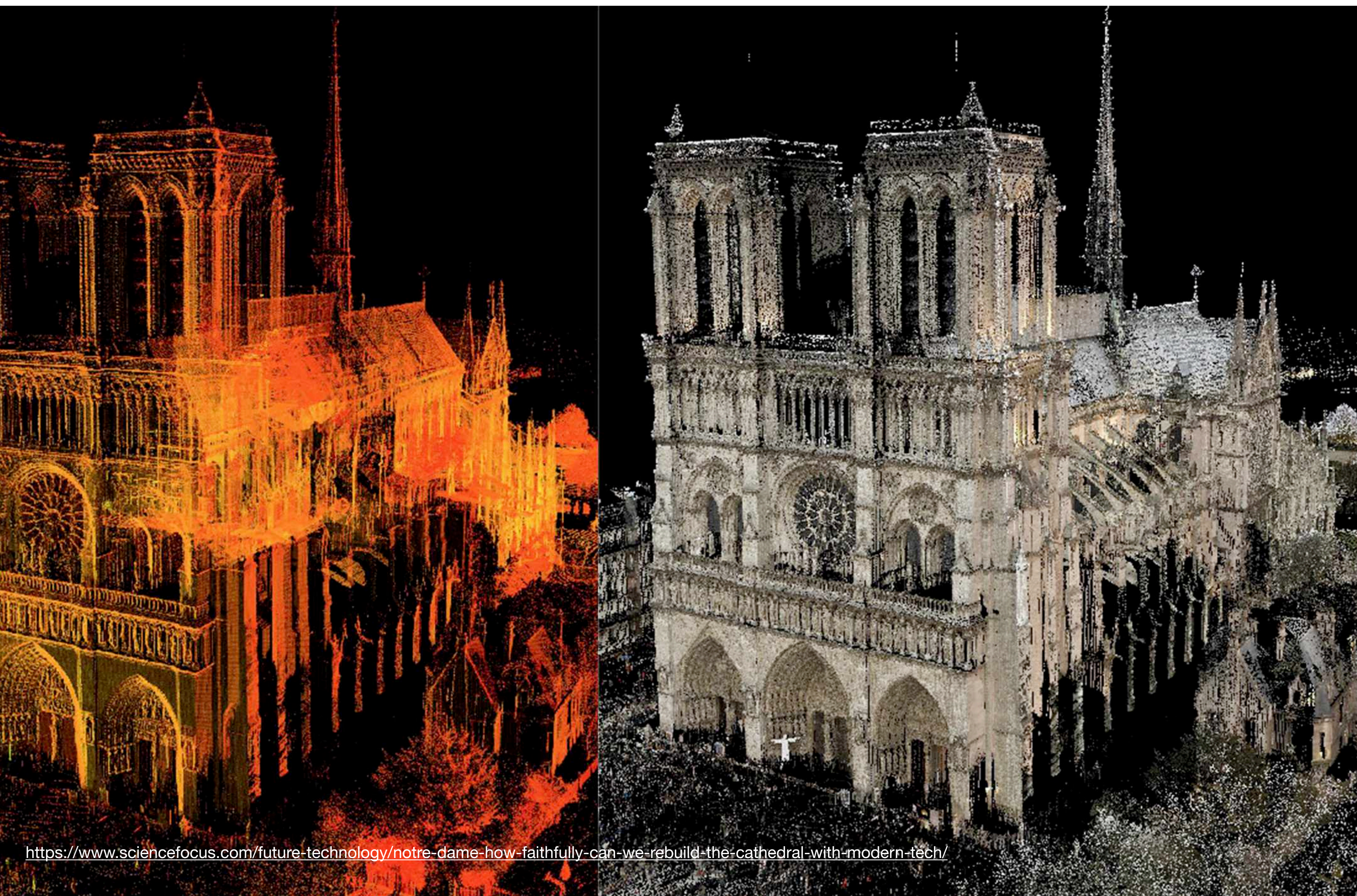


Bezier Surface to Mesh



- ▶ A Bezier surface could be used directly for rendering, but it is usually converted to a mesh (tessellation), which is easier to render.
- ▷ Sample the Bezier surface by calculating a set of points, which become mesh vertices.
- ▷ Connect these vertices generates a mesh.
- ▷ Sampling could be uniform, or could be non-uniform (e.g., more sample when curvature is higher).

Point Cloud/Point Set



Laser Scanner



Laser scanner.



From Mike Jarvis, Prof. @ History Dept. Scanning Elmina Castle, a UNESCO World Heritage Site at Ghana.

LiDAR

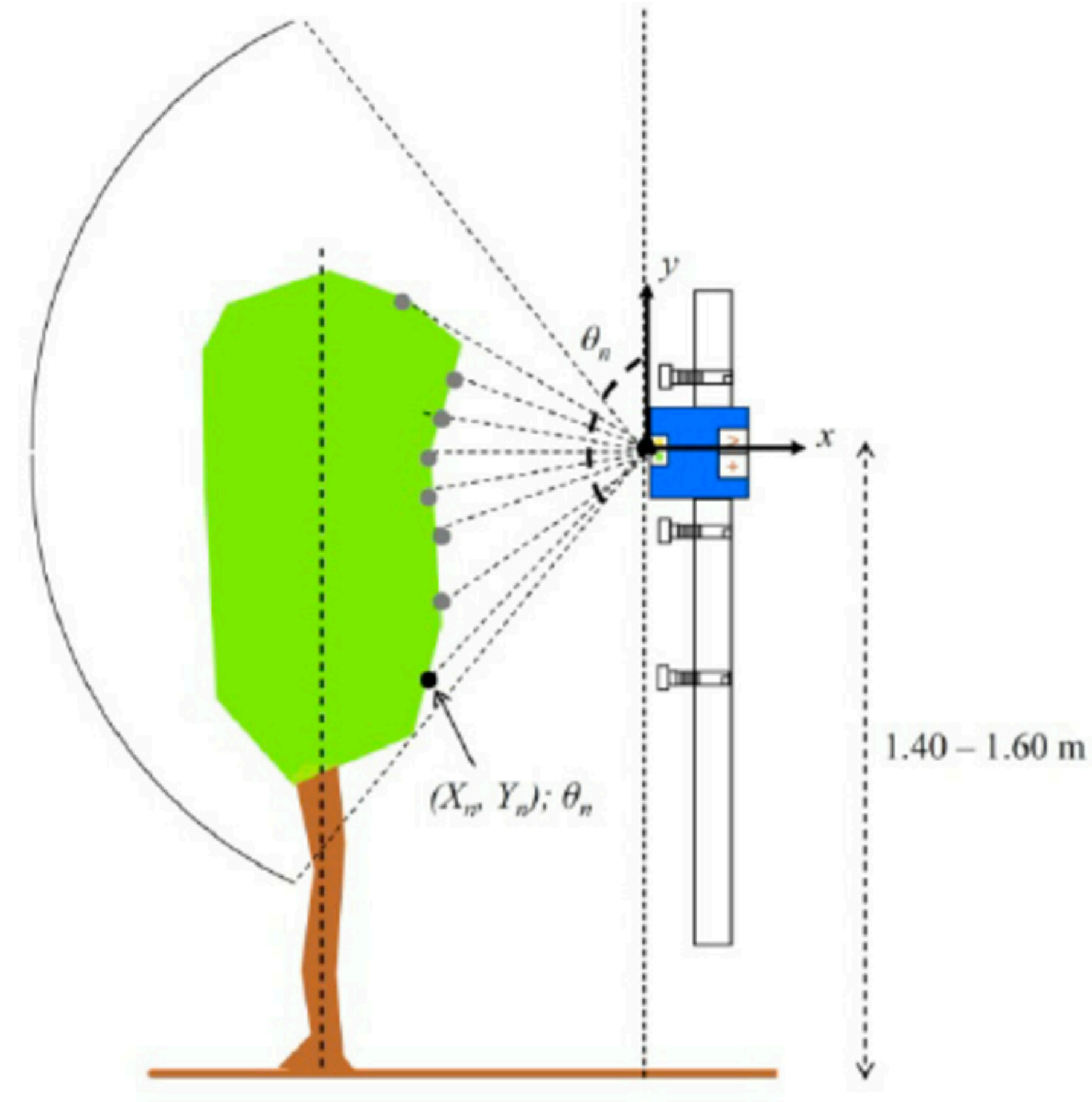
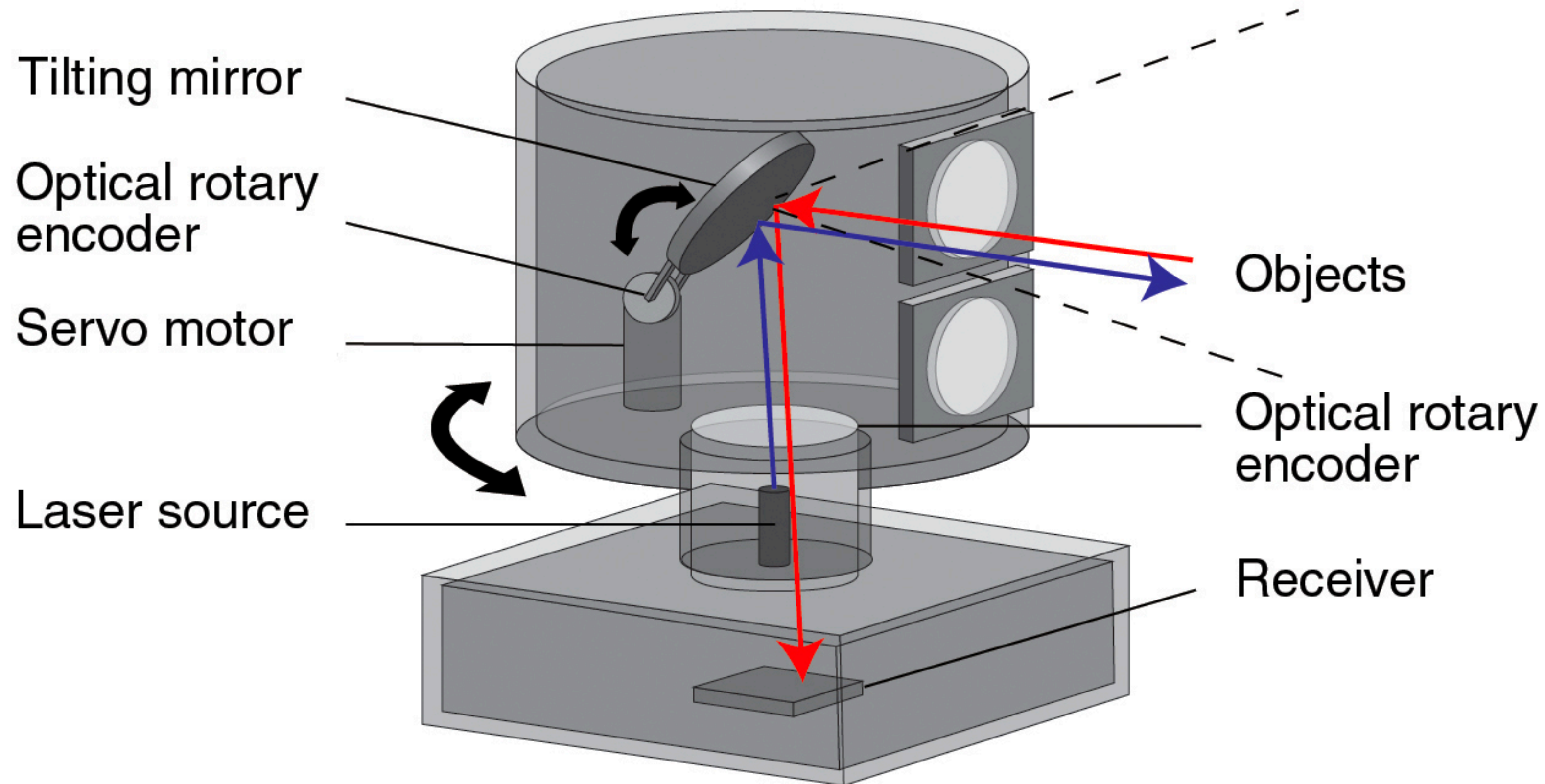


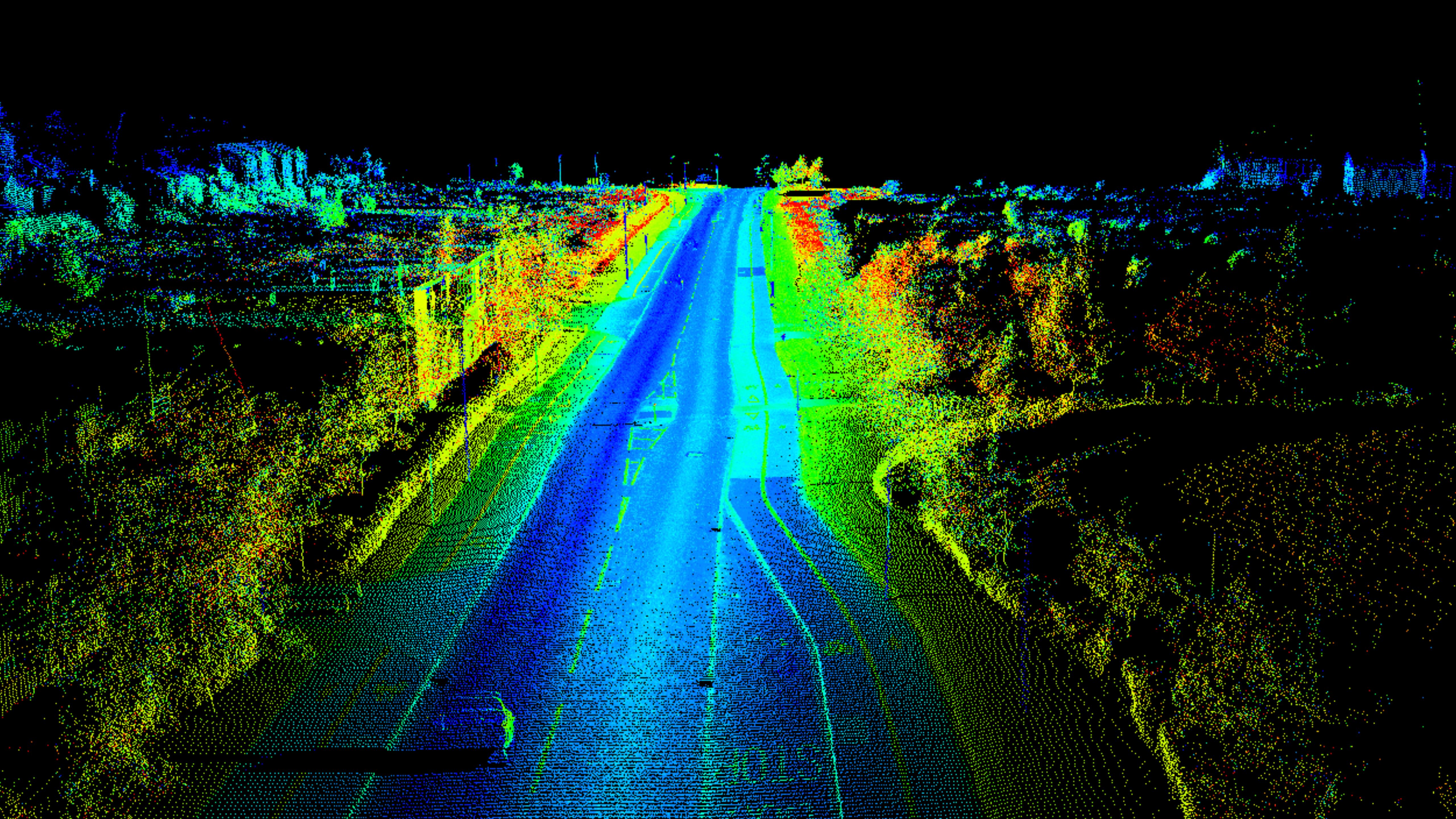
**Velodyne LiDAR (Light
Detection and Ranging)**

Uber's self-driving car equipped with LiDARs



LiDAR





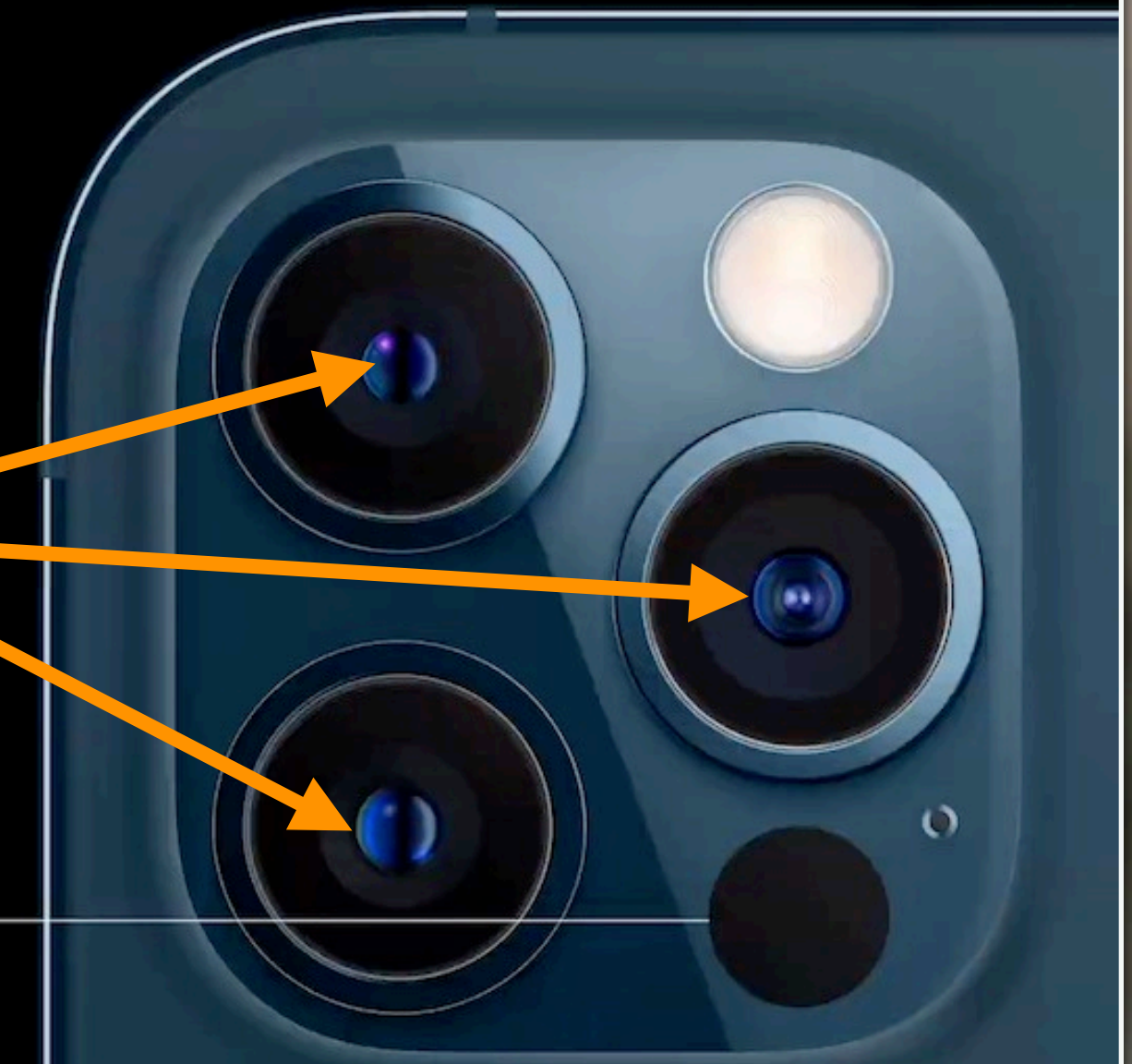


**iPhone 12 Pro LiDAR sensor allows for
6x faster low-light autofocus, instant AR**



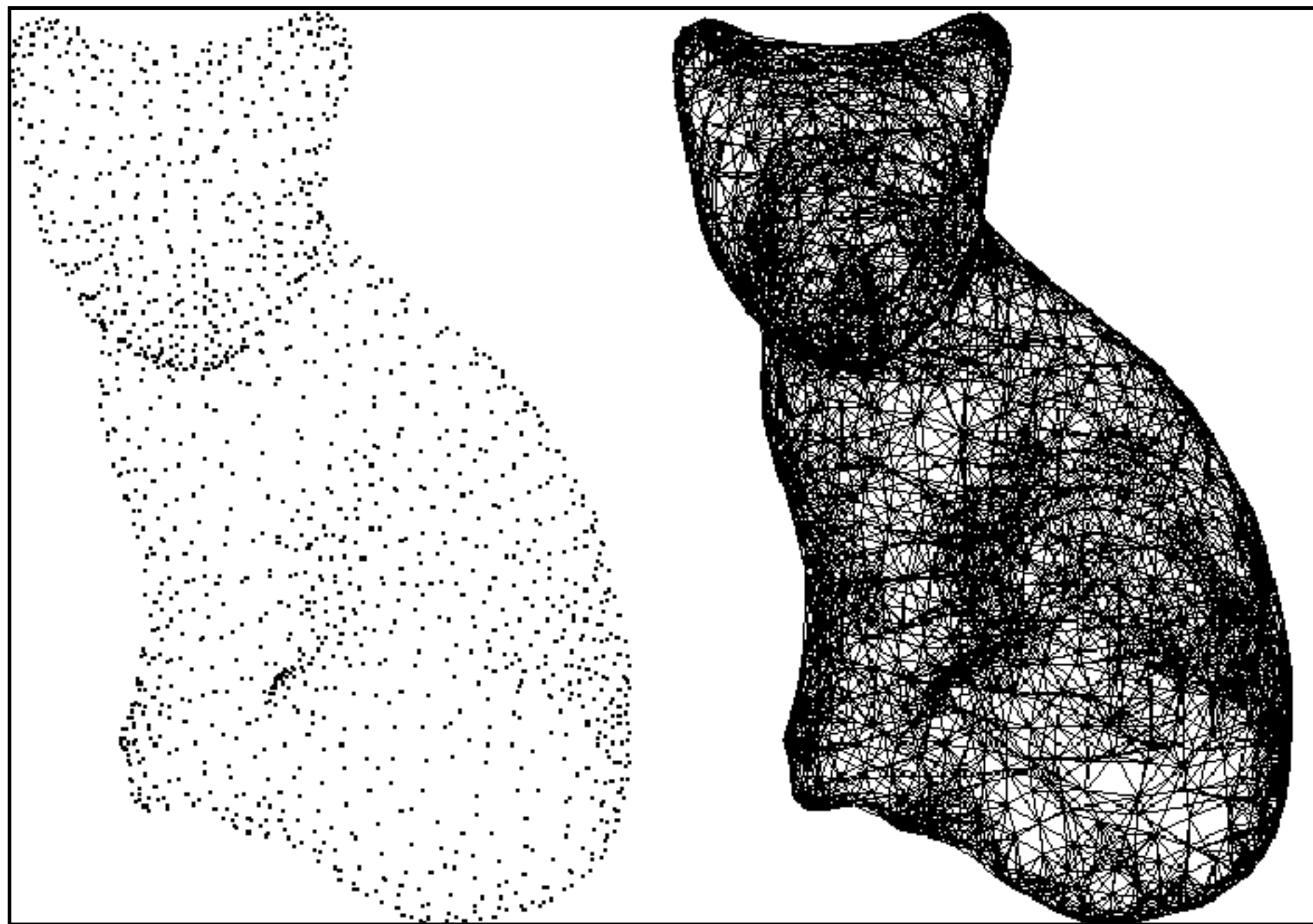
**What are these
cameras?**

LiDAR Scanner

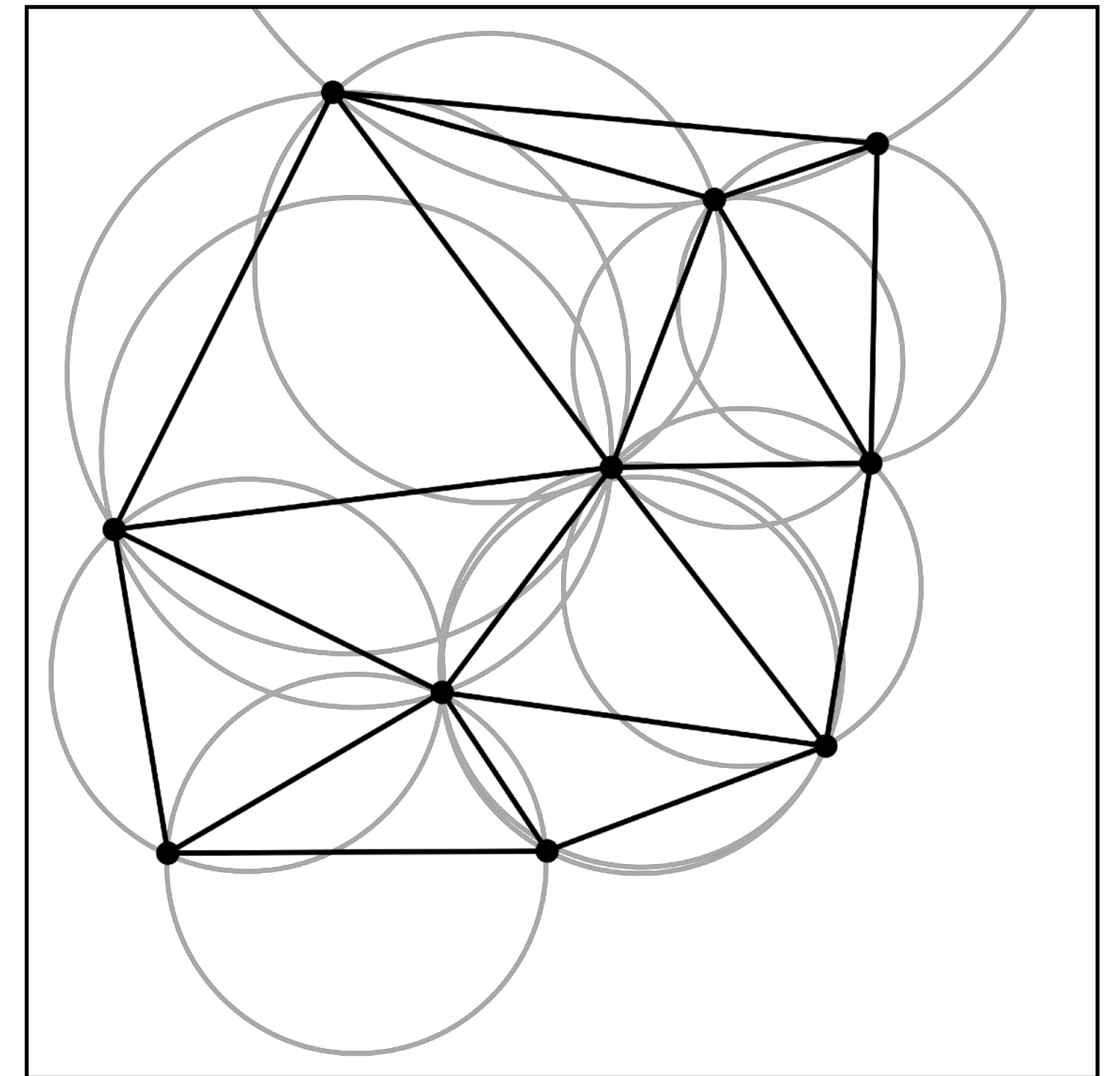


Point Set Triangulation

- Point set could be used directly for rendering (point-based graphics), but it is usually converted to a mesh, which is easier to render.



Delaunay triangulation: no point is inside the circumcircle of any triangle.



Additional Readings

- ▶ Tessellation of 4x4 Bezier Patches for the Intel Pentium III Processor
- ▶ Bézier Curves and Surfaces: the Utah Teapot. A series of articles from scratchapixel on Bezier curves and surfaces.