Skolkovo Institute of Science and Technology

The Geometry Processing Pipeline

Geometric Computer Vision

GCV v2021.1, Module 1

Alexey Artemov, Spring 2021

Lecture Outline

Skolkovo Institute of Science and Technology

§1. The geometry processing pipeline [45 min]

1.1. Goals of 3D/geometric computer vision systems

1.2. Common stages of geometry processing

1.3. Scanning [next video]

1.4. Registration

1.5. Reconstruction and meshing

1.6. Postprocessing [next videos]

Lecture Outline

§2. 3D representations in computer vision/graphics [15 min, Friday]

2.1. Directly measurable: multiple-view images, range-images, point clouds, volumes

2.2. Derived: surface meshes, implicit functions

2.3. Higher-level: CAD, shape programs

§2. 3D representations in vision and graphics

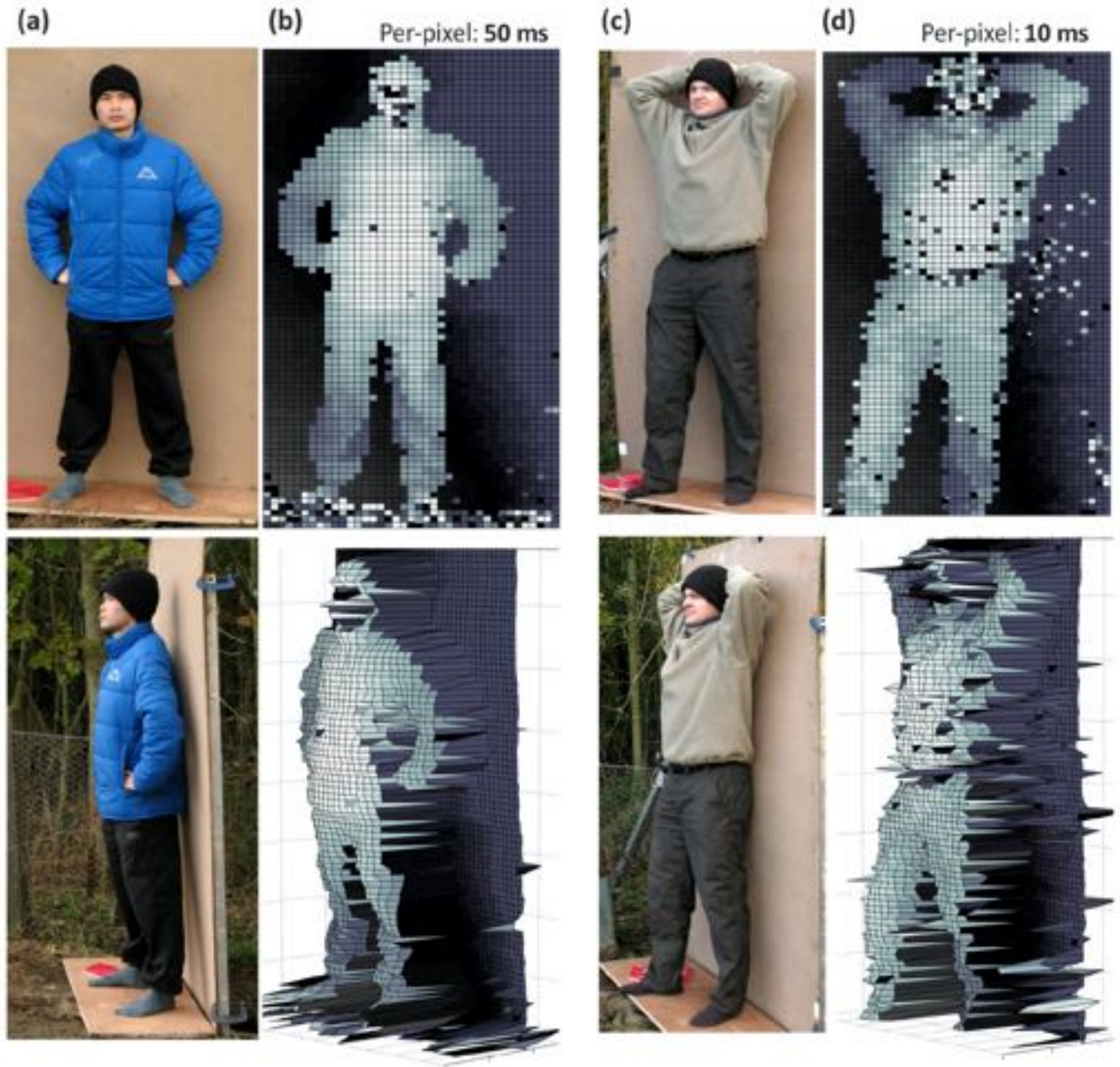
Range-images [Depth maps]

2.1. Range-images [Depth maps]

Skoltech
Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics

- Like conventional camera images, with each pixel storing depth rather than a color
- Likely the most common datatype acquired directly
- We commonly need multiple depth images to capture the full scene or object



Picture Credit: Heriot-Watt University

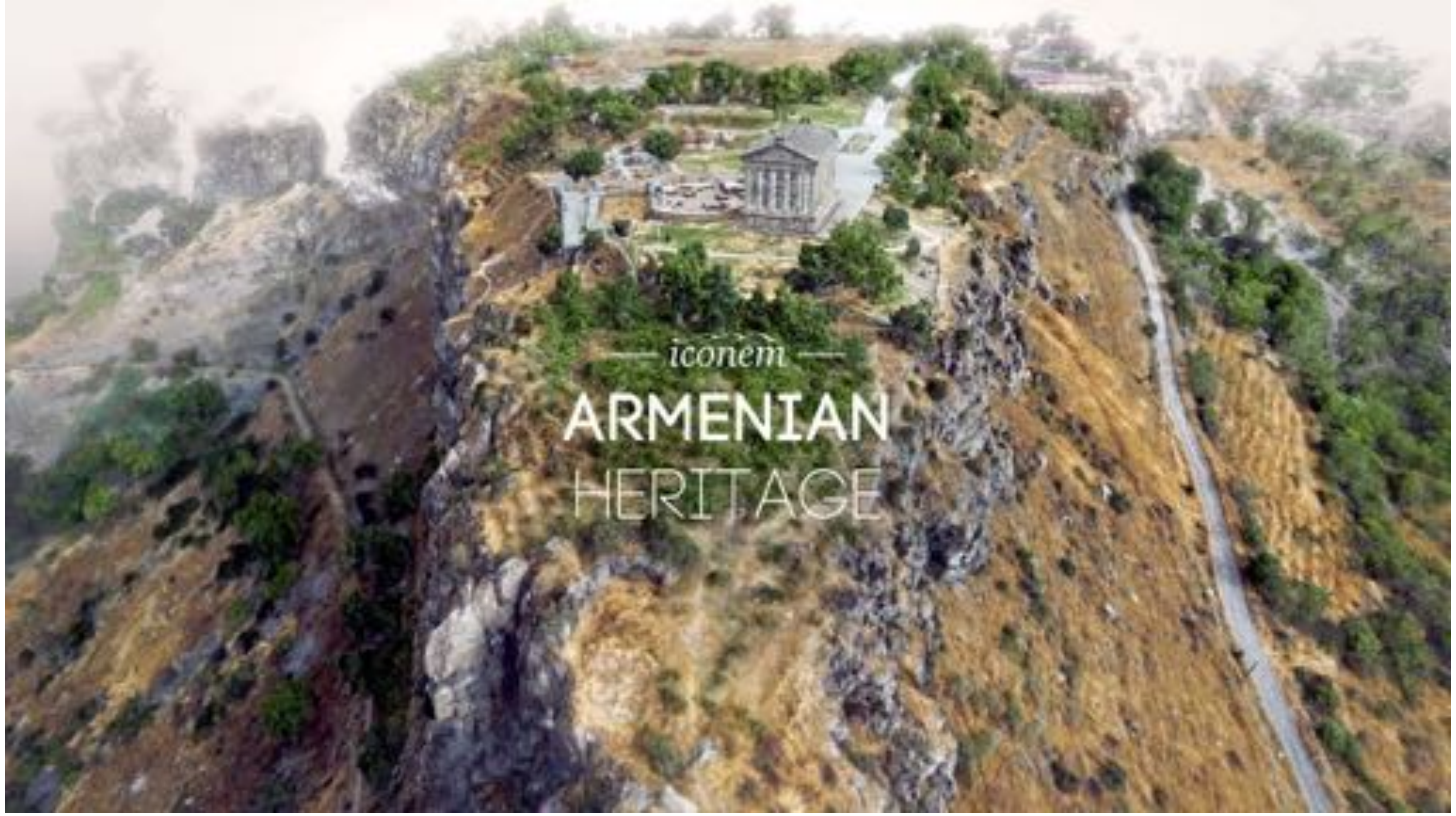
Point clouds [Point sets]

2.1. Point clouds [Point sets]

Skoltech
Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics

- A set of data points in space, representing a shape of an object
- Commonly:
  - registered depth maps
  - directly acquired (e.g. LIDAR, ToF)
- Permutation invariance (points are unordered!)

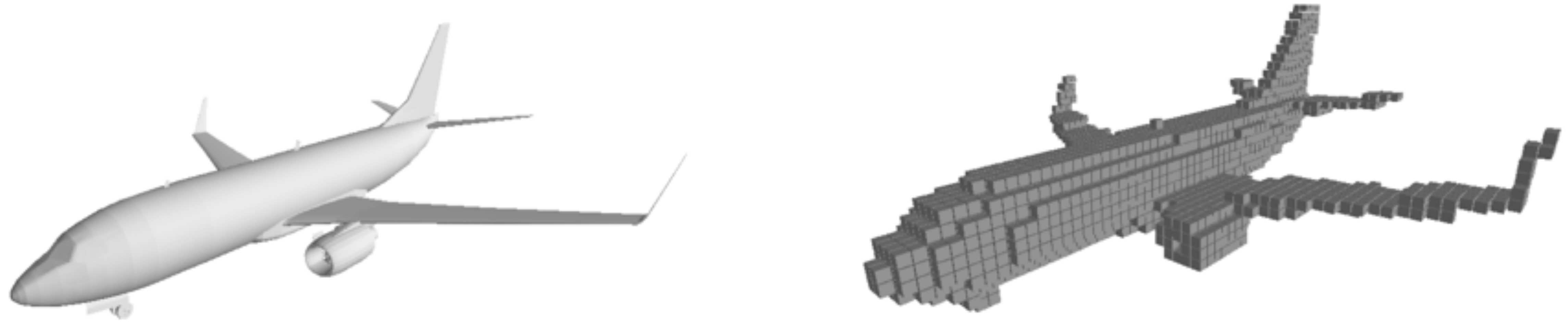


Video credit: Iconem, 3D models of Armenian Heritage

Volumetric pixels [Voxels, volumes]

2.1. Volumetric pixels [Voxels, volumes]

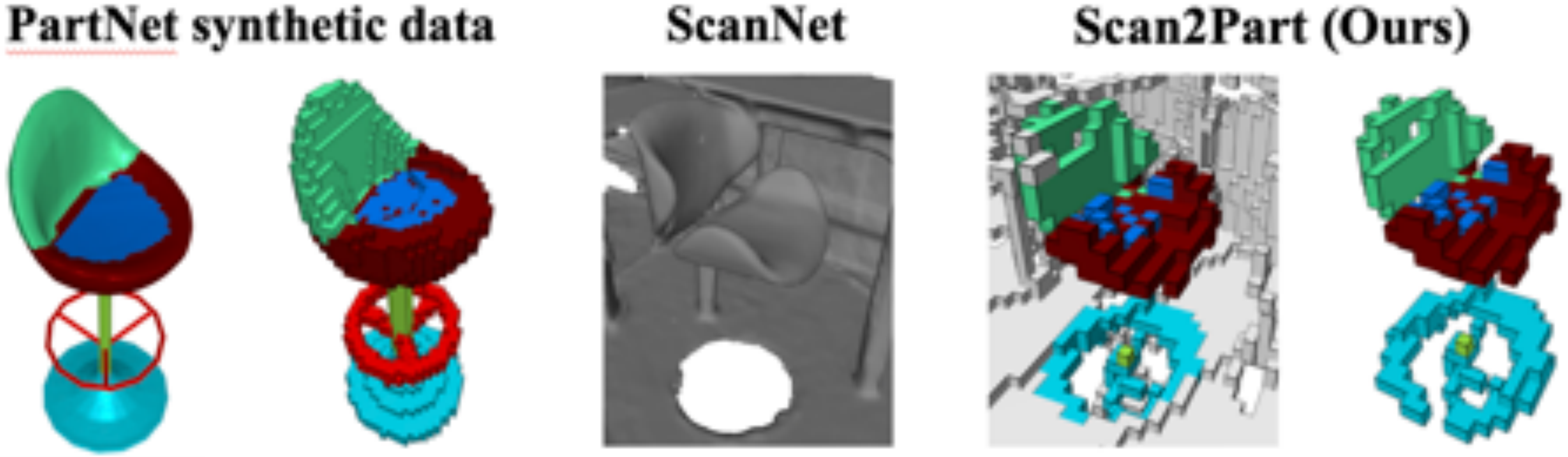
§2. 3D representations in vision and graphics



Mesh

Voxelized 64^3

Picture Credit: OctNet: Learning Deep 3D Representations at High Resolutions



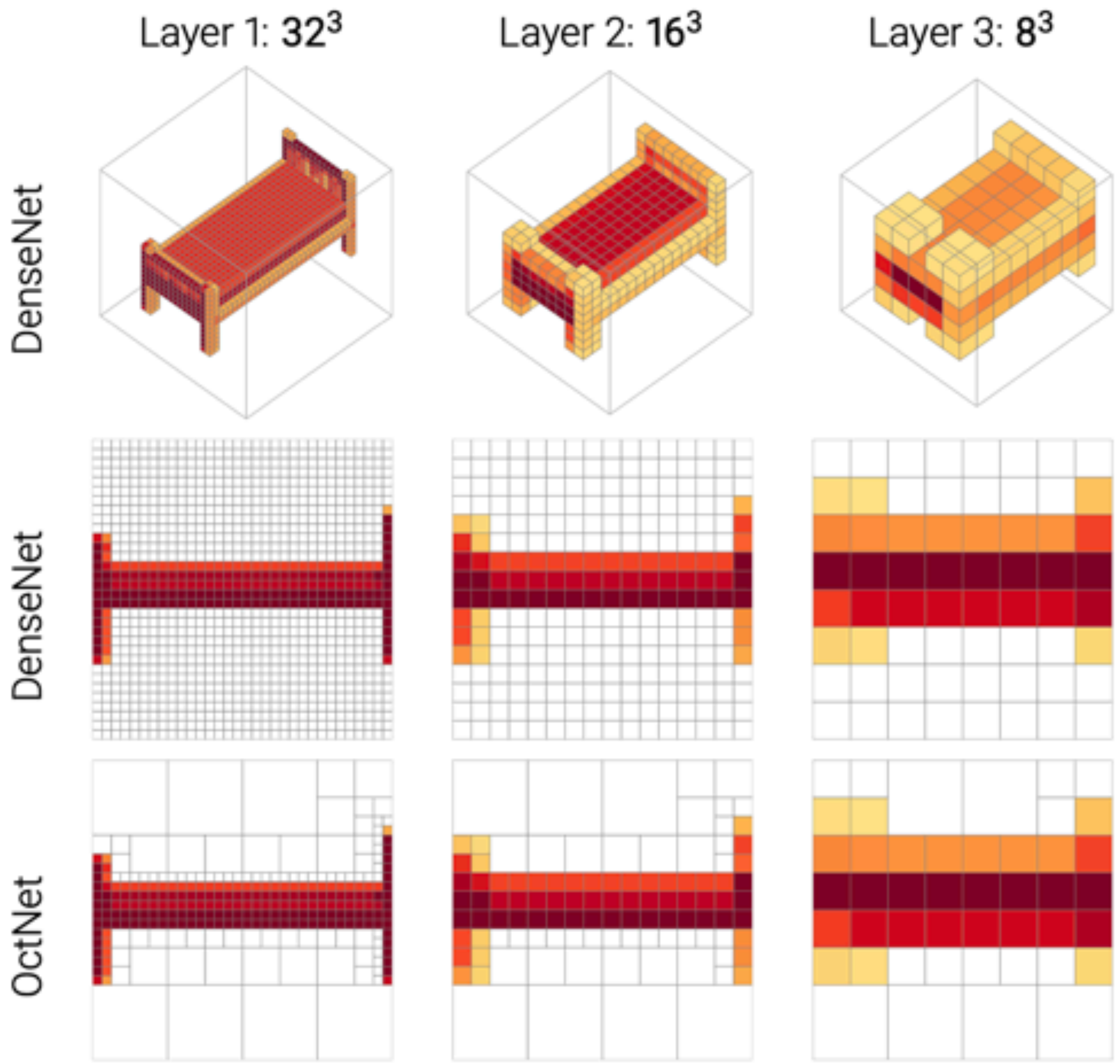
- Define a regular grid in 3D
- Store per-cell values of signed distance function (SDF) or a truncated SDF (TSDF)
- Store occupancy grid: OG(x) = 1 if TSDF(x) < T

2.1. Volumetric pixels [Voxels, volumes]

§2. 3D representations in vision and graphics

- Storing raw voxels is inefficient compared to points or meshes
- Low spatial resolution

- Need adaptive data structures
- Need sparsity enabled methods



Picture Credit: OctNet: Learning Deep 3D Representations at High Resolutions

2.1. Volumetric pixels [Voxels, volumes]

Skoltech
Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics

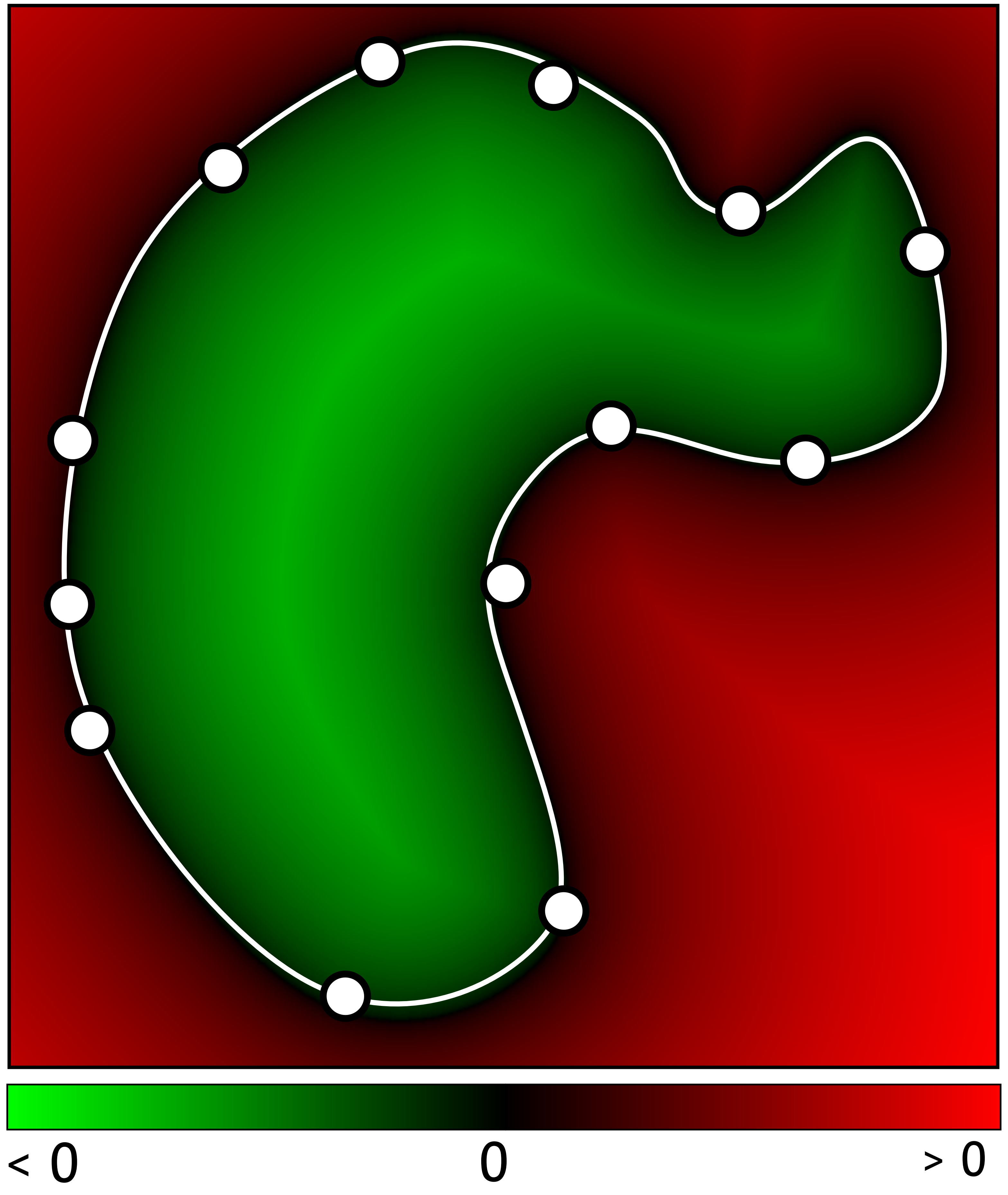
Video Credit: Atomontage

- Volumetric rendering and volumetric graphics (old idea but still impressive)

Implicit surface representations

2.2. Implicit surface representations

§2. 3D representations in vision and graphics



- Assumes the existence of a function

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

with value > 0 outside the shape and < 0 inside

- Extract zero-level set

$$\{\mathbf{x} : f(\mathbf{x}) = 0\}$$

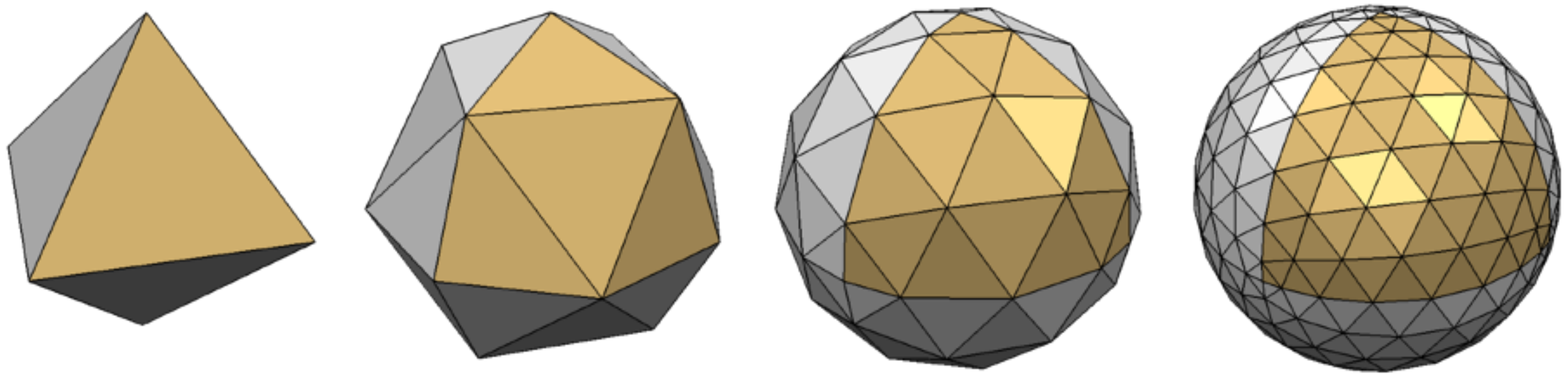
Surface meshes [Polygonal/Triangular meshes, triangulations]

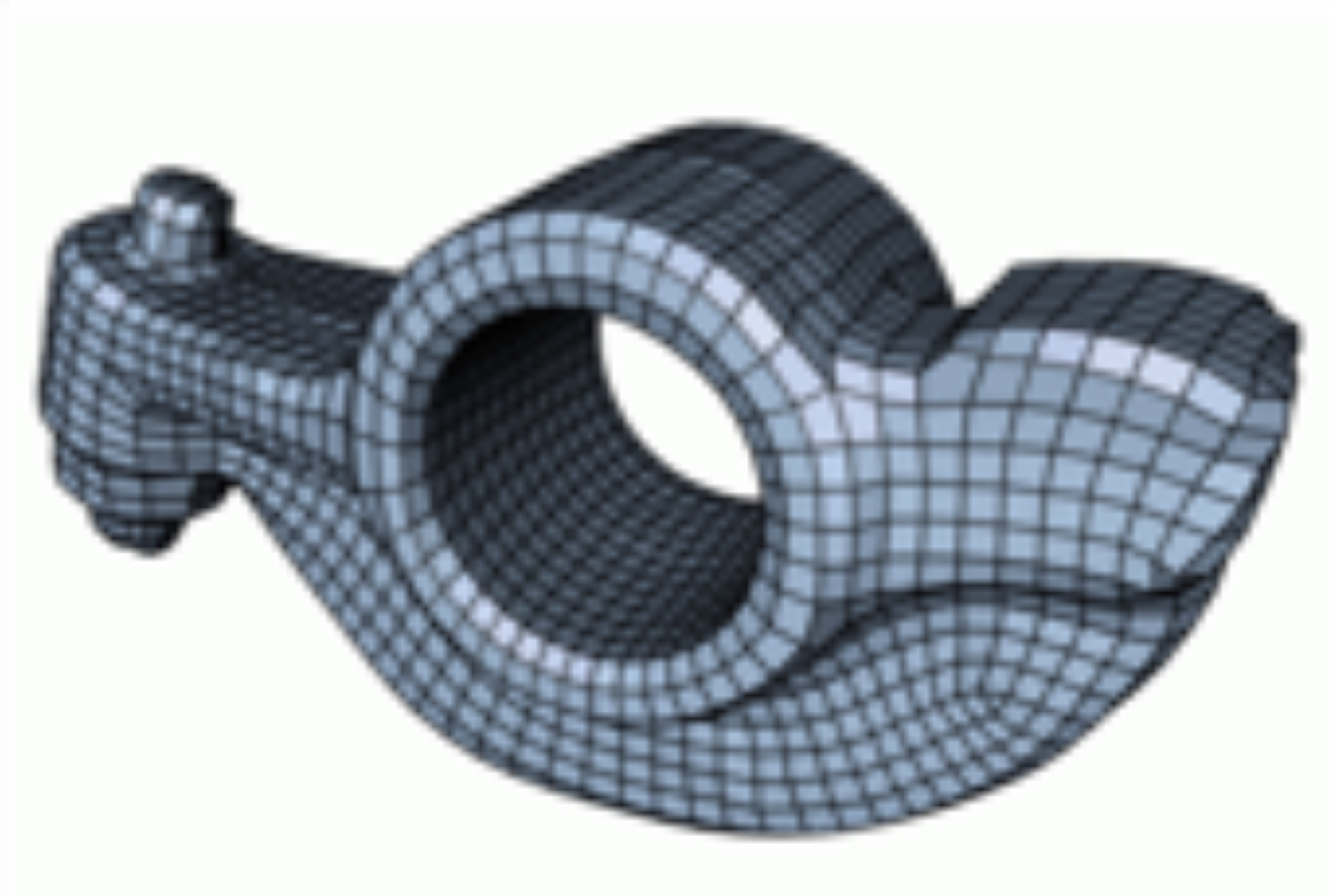
2.2. Polygonal Meshes

Skoltech
Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics

- Boundary representations of objects





Slide Credit: Daniele Panozzo

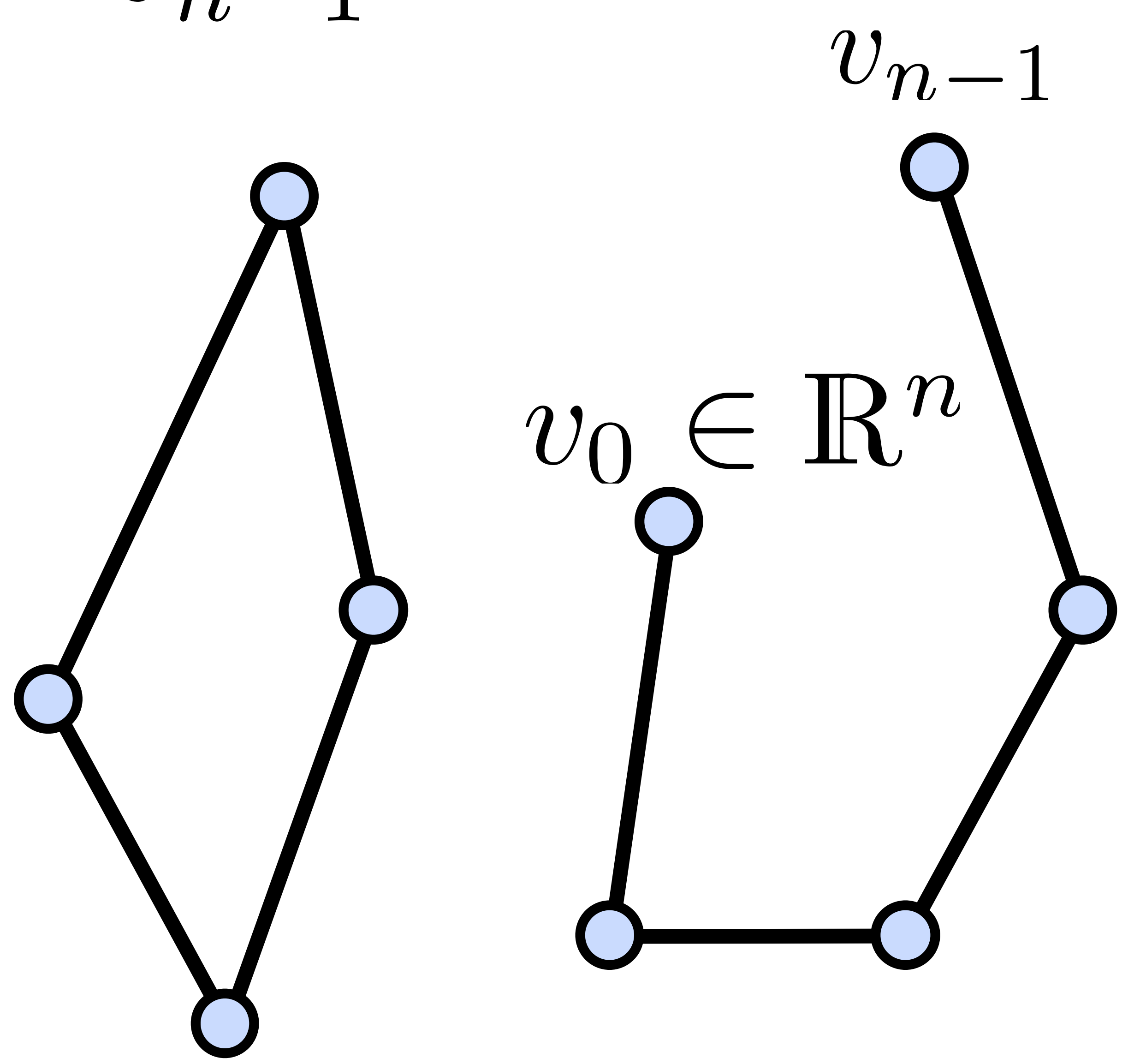
2.2. Polygon

Skoltech
Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics

- Vertices: $v_0, v_1, \ldots, v_{n-1}$
- Edges: $\{(v_0, v_1), \ldots, (v_{n-2}, v_{n-1})\}$

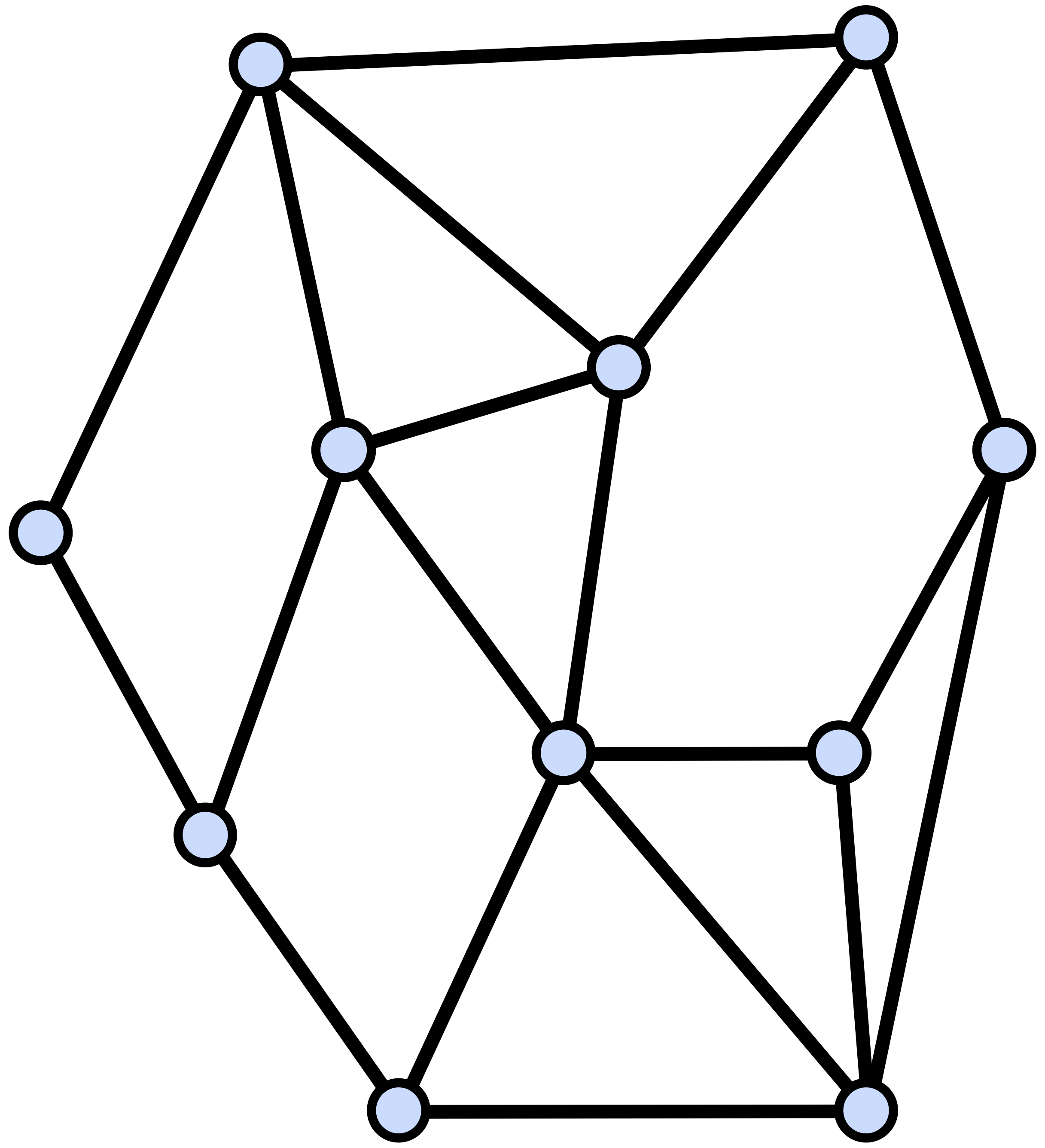
- Closed: $v_0 = v_{n-1}$
- Planar: all vertices on a plane
- Simple: not self-intersecting



2.2. Polygonal Mesh

§2. 3D representations in vision and graphics

- A finite set M of closed, simple polygons Q_i is a polygonal mesh
- The intersection of two polygons in M is either empty, a vertex, or an edge



$$M = \langle V, E, F \rangle$$

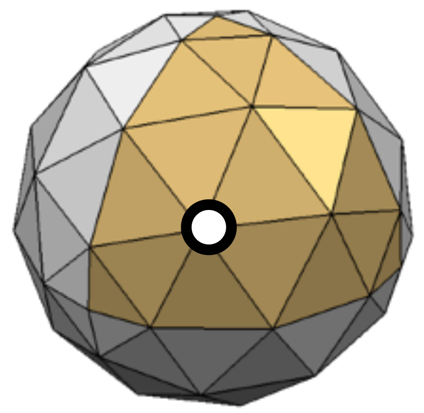
vertices edges faces

2.2. Triangle Meshes

Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics

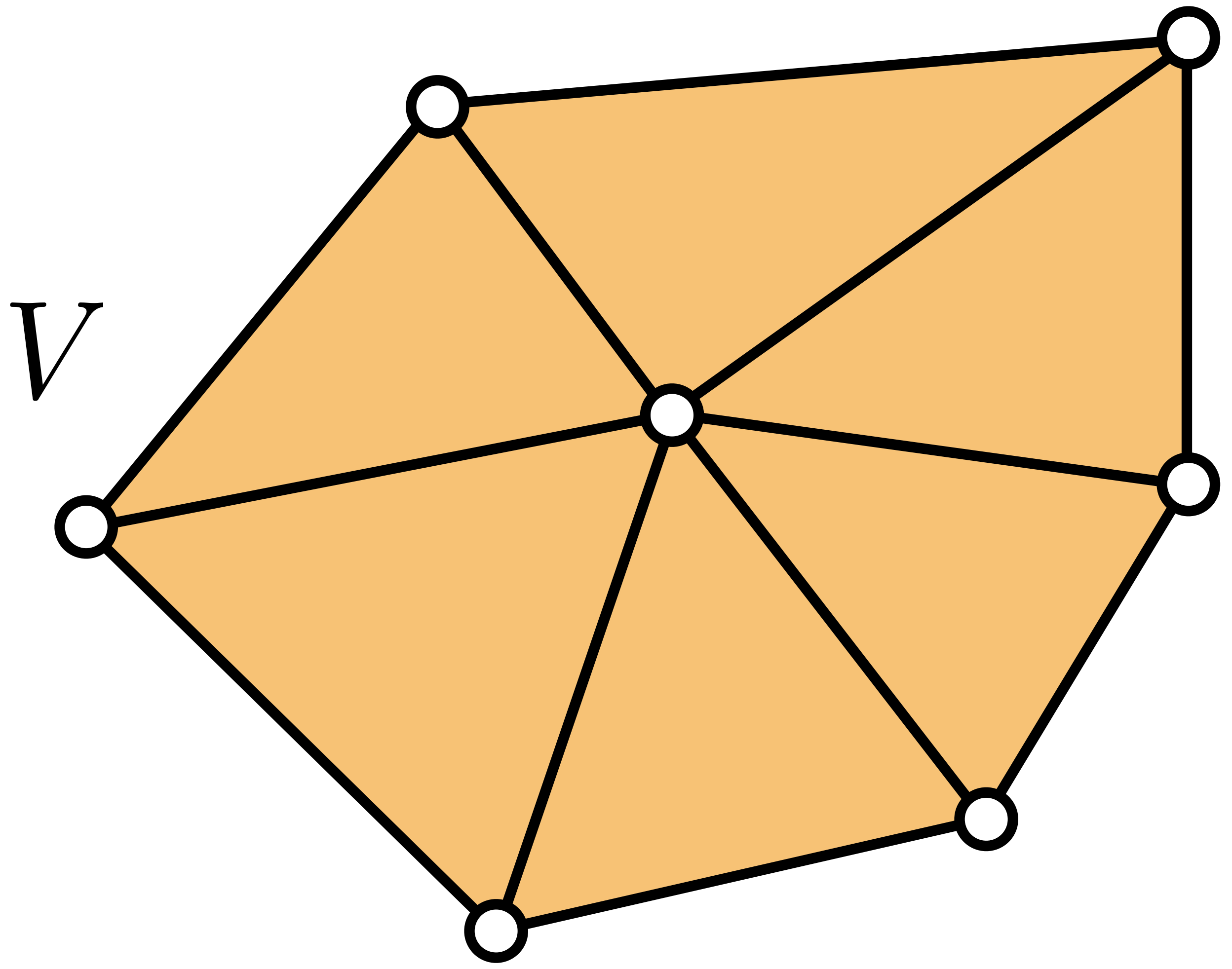
- Connectivity: vertices, edges, triangles
- Geometry: vertex positions



$$V = \{v_1, \dots, v_n\}$$

$$E = \{e_1, \dots, e_k\}, \quad e_i \in V \times V$$

$$F = \{f_1, \dots, f_m\}, \quad f_i \in V \times V \times V$$

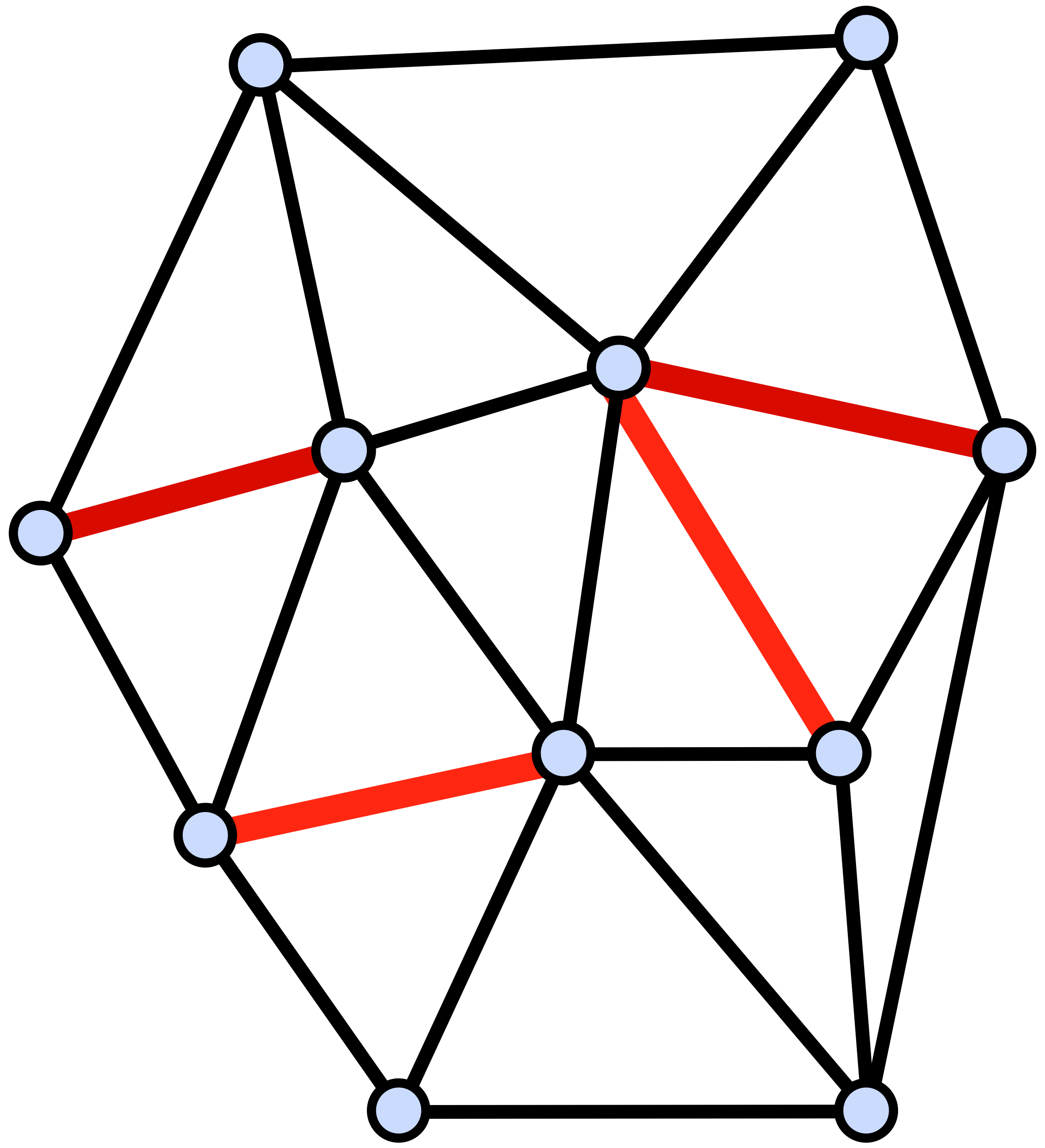


$$P = \{\mathbf{p}_1, \dots, \mathbf{p}_n\}, \quad \mathbf{p}_i \in \mathbb{R}^3$$

Slide Credit: Daniele Panozzo

2.2. Triangulation

§2. 3D representations in vision and graphics



- Polygonal mesh where every face is a triangle

- Simplifies data structures
- Simplifies rendering
- Simplifies algorithms
- Each face planar and convex
- Any polygon can be triangulated

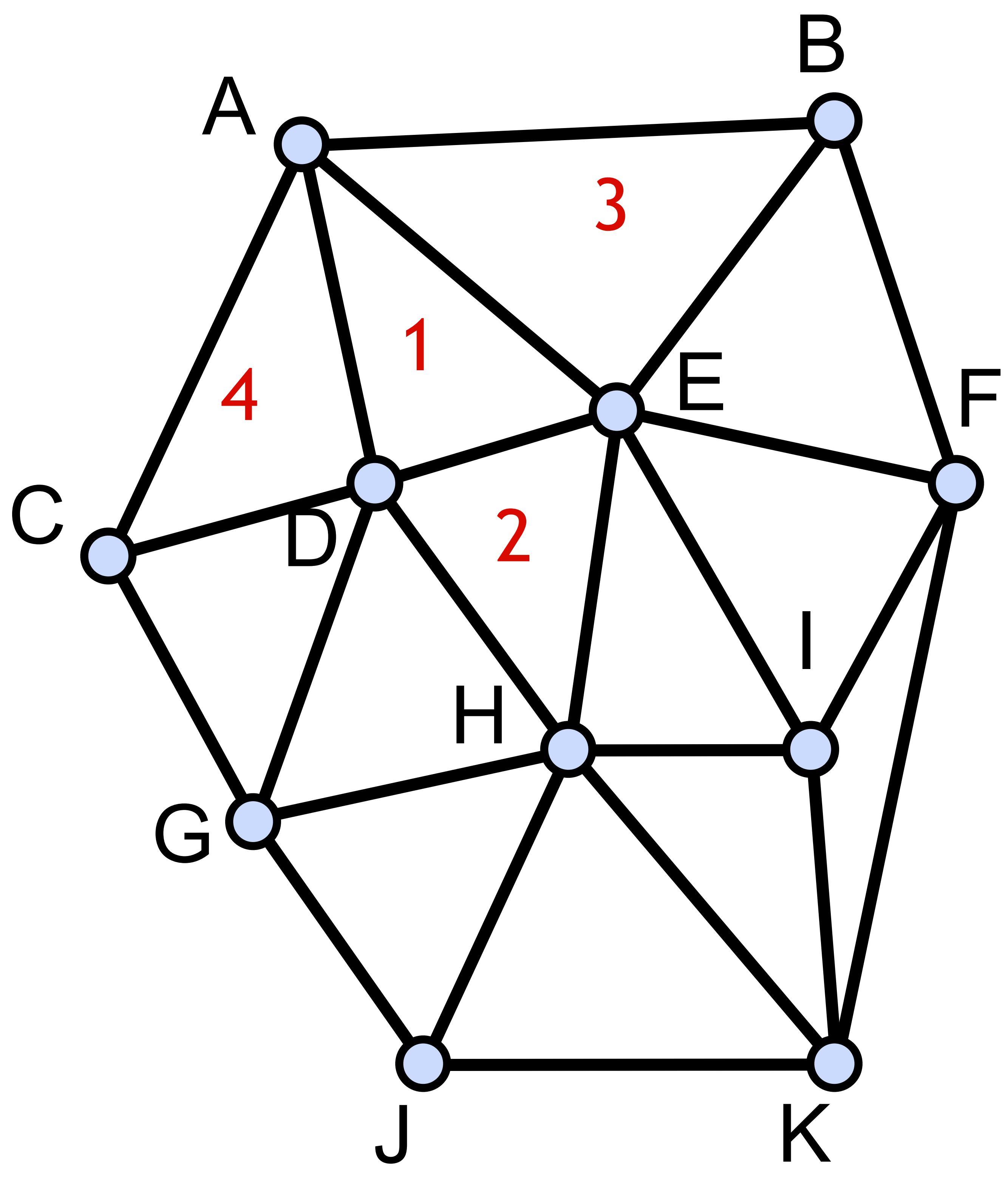
2.2. Data Structures

§2. 3D representations in vision and graphics

- What should be stored?
  - Geometry: 3D coordinates
  - Connectivity
    - Adjacency relationships
  - Attributes
    - Normal, color, texture coordinates
    - Per vertex, face, edge

2.2. Data Structures

§2. 3D representations in vision and graphics

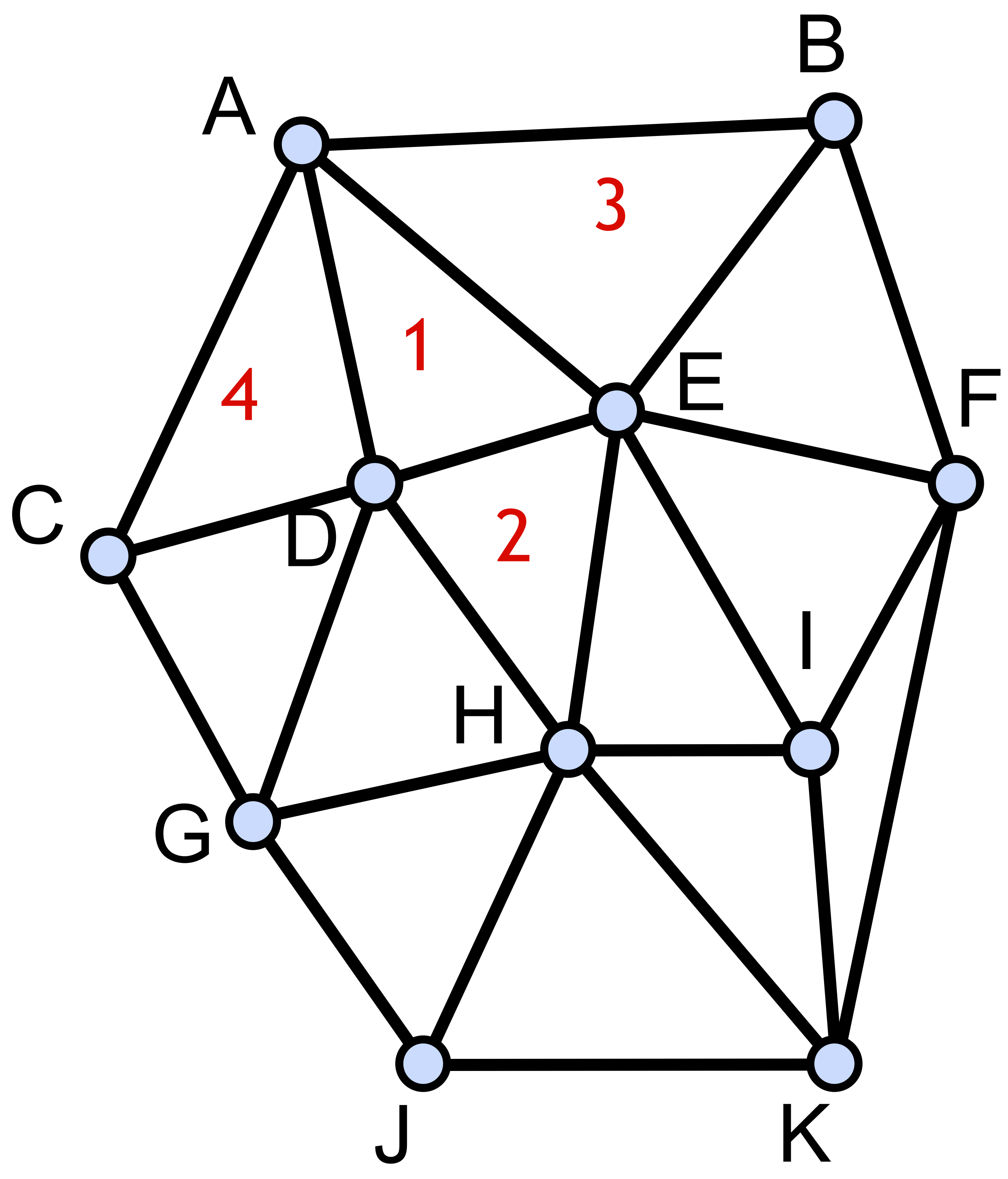


- What should be supported?
  - Rendering
  - Geometry queries
    - What are the vertices of face #2?
    - Is vertex *A* adjacent to vertex *H*?
    - Which faces are adjacent to face #1?
  - Modifications
    - Remove/add a vertex/face
    - Vertex split, edge collapse

2.2. Data Structures

Skoltech
Skolkovo Institute of Science and Technology

§2. 3D representations in vision and graphics



- How good is a data structure?
  - Time to construct
  - Time to answer a query
  - Time to perform an operation
  - Space complexity
  - Redundancy
- Criteria for design
  - Expected number of vertices
  - Available memory
  - Required operations
  - Distribution of operations

2.2. Triangle List

§2. 3D representations in vision and graphics

- STL format (used in CAD)
- Storage
  - Face: 3 positions
  - 4 bytes per coordinate
  - 36 bytes per face
    - Euler: $f = 2v$
    - $72*v$ bytes for a mesh with v vertices
- No connectivity information

Triangles			
0	x0	y0	z0
1	x1	x1	z1
2	x2	y2	z2
3	x3	y3	z3
4	x4	y4	z4
5	x5	y5	z5
6	x6	y6	z6
...	...	...	...

2.2. Indexed Face Set

§2. 3D representations in vision and graphics

- Used in formats

  OBJ, OFF, WRL
- Storage
  - Vertex: position
  - Face: vertex indices
  - 12 bytes per vertex
  - 12 bytes per face
  - 36*v bytes for the mesh

- No *explicit* neighborhood info

Vertices			
v0	x0	y0	z0
v1	x1	x1	z1
v2	x2	y2	z2
v3	x3	y3	z3
v4	x4	y4	z4
v5	x5	y5	z5
v6	x6	y6	z6
...	...	...	...

Triangles			
t0	v0	v1	v2
t1	v0	v1	v3
t2	v2	v4	v3
t3	v5	v2	v6
...	...	...	...

References

1. Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., & Lévy, B. (2010). *Polygon mesh processing*. CRC press.