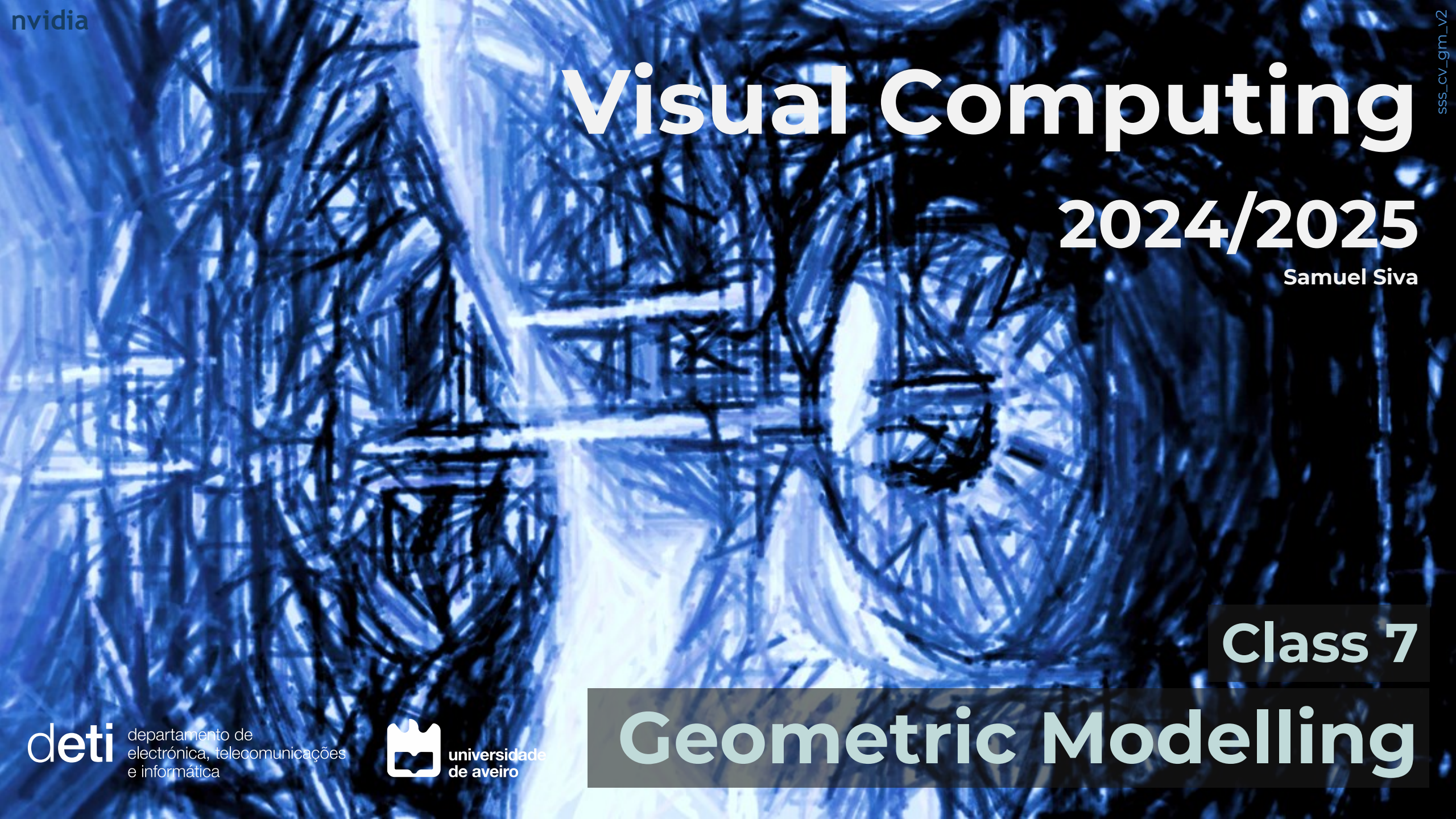


nvidia

Visual Computing

2024/2025

Samuel Siva

sss_cv_gm_v2

Class 7

Geometric Modelling

deti departamento de electrónica, telecomunicações e informática

universidade de aveiro

Agenda

- Definition and Applications of Geometric Modelling
- Polygonal Meshes
- Geometric and Topological Information
- Computational Representation

Geometric Modeling

A **geometric model** describes the **shape** of an object (real or virtual)

How?

- Different mathematical representations?
- Possible operations?
- Compactness ? Robustness ? Efficiency ?
- Data structures?
- Interpolation vs Approximation ?
- …

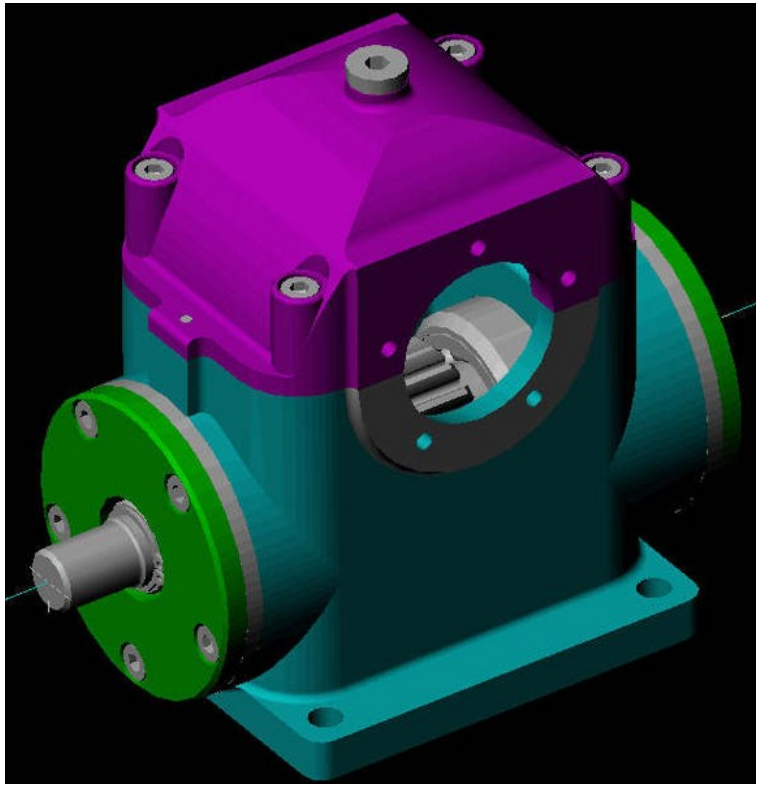
Geometric Modeling

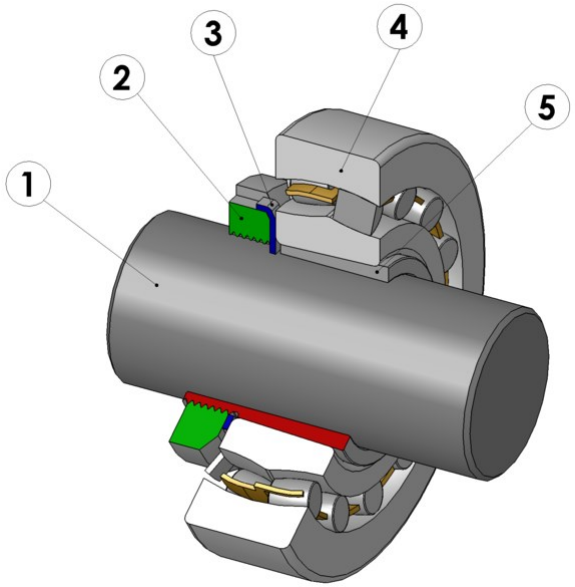
What for?

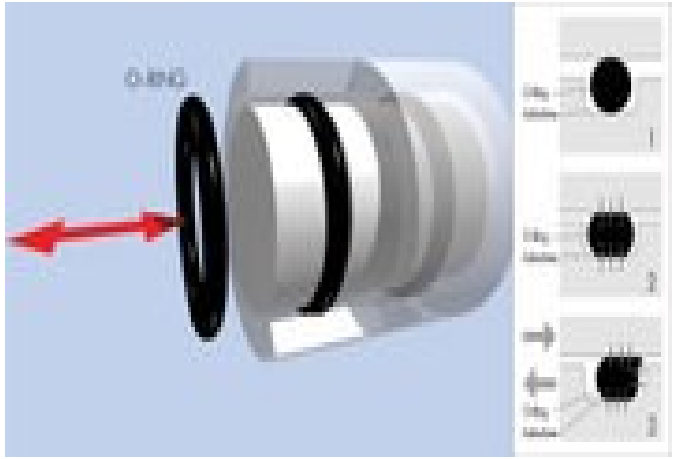
- Distinguish between **inside**, **outside** and **border** of a model
- Compute **properties**
  - Centroid
  - Area / Volume
  - …
- Detect interferences / **collisions**
- Compute **light reflections** and / or transparencies
- …

3D models – Applications

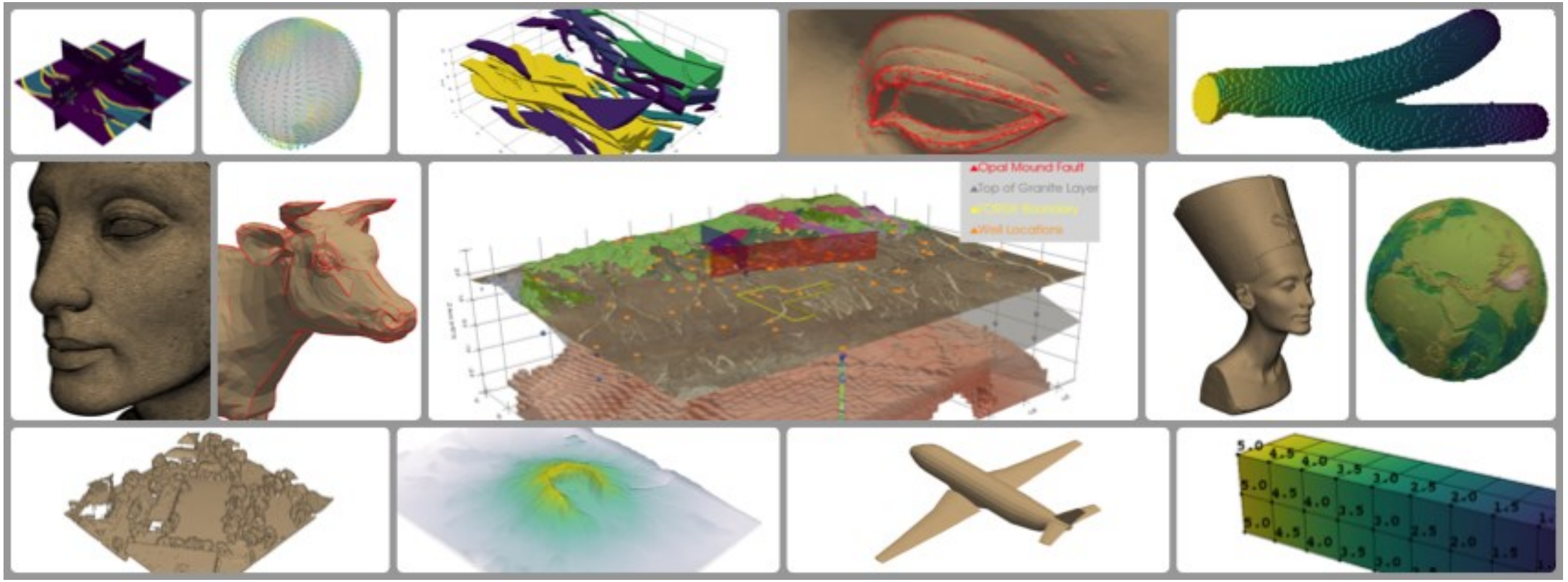
CAD / CAM



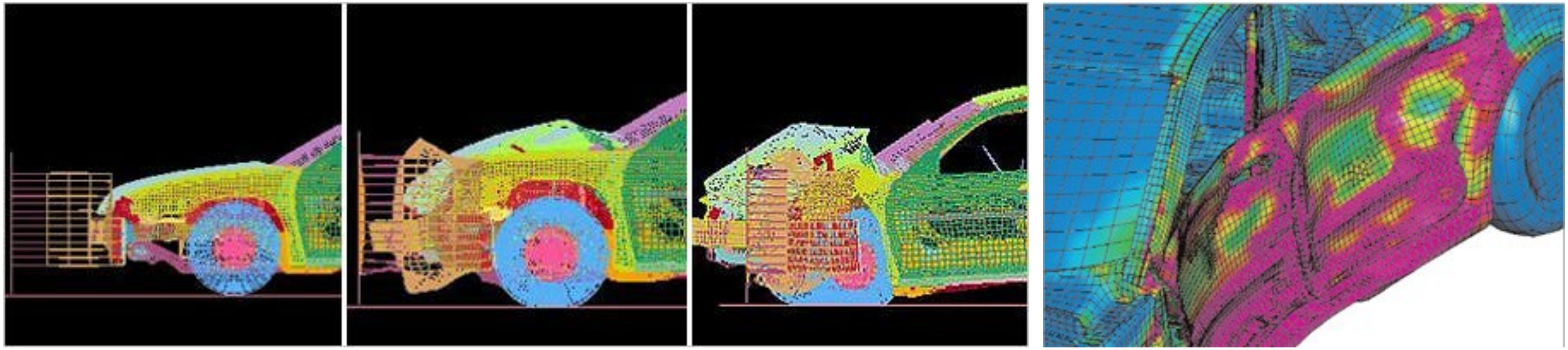




Data Visualization



CAD – Simulation and Visualization



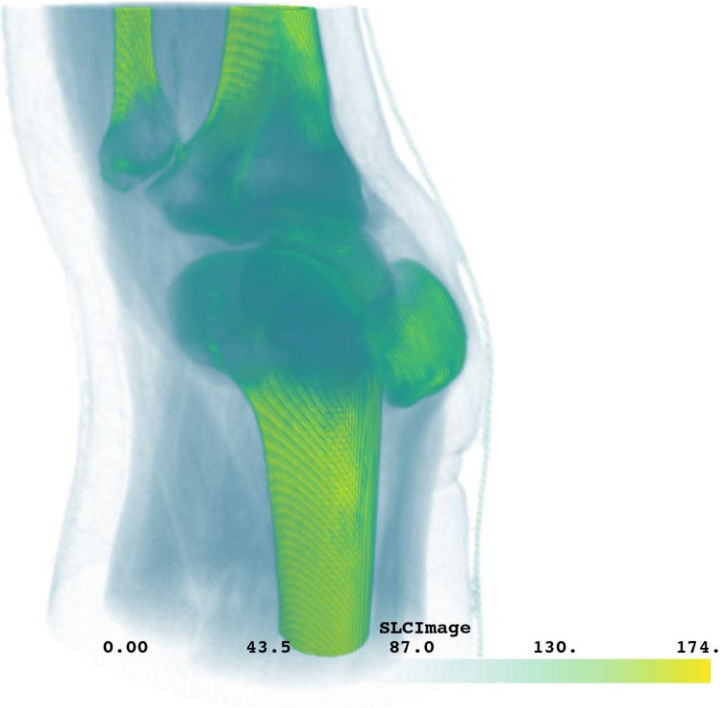
Crashing Cars When They’re Still a Gleam in the Designer’s Eye
NY Times
https://www.nytimes.com/2007/06/17/automobiles/17CRASH.html

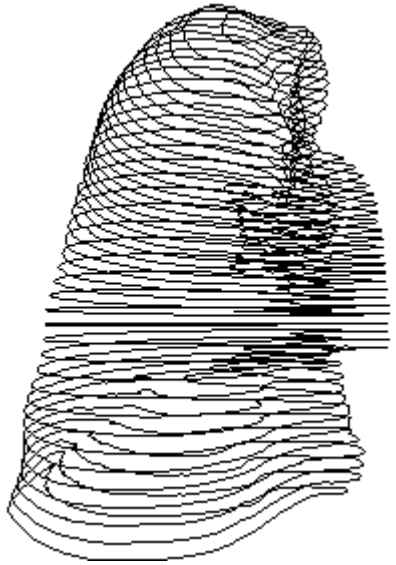
Virtual and Augmented Reality

https://mojoapps.co/wp-content/uploads/2019/03/3D-models-in-business-01-1-thegem-blog-default.jpg

3D models – Applications

Medical Data Processing





Mesh

3D models – Applications

Other application areas

- Computer games
- Geographical information systems (GIS)
- Engineering analysis
- 3D printing / Rapid prototyping
- Medical solid modeling
- …

3D models – Shape

Define from scratch using VRML / X3D, OpenGL, VTK, …

- Tedious; requires skill

Obtain from CAD files or model databases

- Convert to compatible formats
- Use of existing models in manufacturing applications

Create using a 3D digitizer or a 3D scanner

- 3D digitizer : stylus
- 3D scanner : tracker, cameras and laser

3D modeling tools

- Autodesk's **3ds max** and **Maya**
  - http://www.autodesk.com
- **Blender**: Free open-source 3D content-creation suite
  - http://www.blender.org
- **Rhino**: Uninhibited free-form 3D modeling
  - http://www.rhino3d.com
- **SolidWorks**; 3D feature modeling
  - http://www.solidworks.com
- **POV-Ray**: Persistence of Vision Ray-Tracer
  - http://www.povray.org

Polygonal meshes

https://png.pngtree.com/background/20230617/original/pngtree-abstract-reflections-on-glass-surface-3d-rendered-polygonal-mesh-picture-image_3668554.jpg

Geometric Modeling

Main areas

- Curve and surface modeling
  - Computer-Aided Geometric Design (CAGD)
- Solid Modeling
- Volume Modeling

Simplest models

- Curves : Polygonal lines
- Surfaces : Polygonal meshes

Polygonal Meshes

Surface is defined as a collection of neighboring faces (e.g., triangles)

- Geometry + Topology (i.e., connectivity)
- Vertices, edges, faces

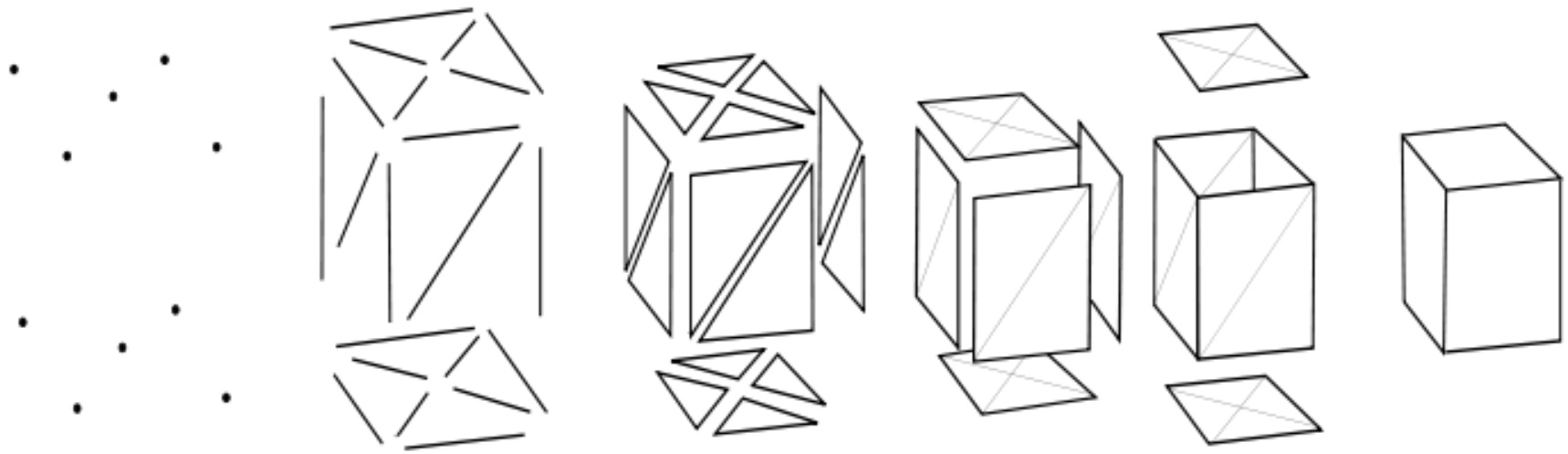
Euler formula for closed surfaces

- V + F – E = 2

Exact vs approximate representations

- Polyhedral models
- Curved surfaces
- Terrain models
- Complex surfaces / models

Polygonal Meshes



Polygonal Meshes

Collection of neighboring **vertices**, **edges** and **polygons**

- Usually, **triangles**

Vertex

- Shared by, at least, 2 edges

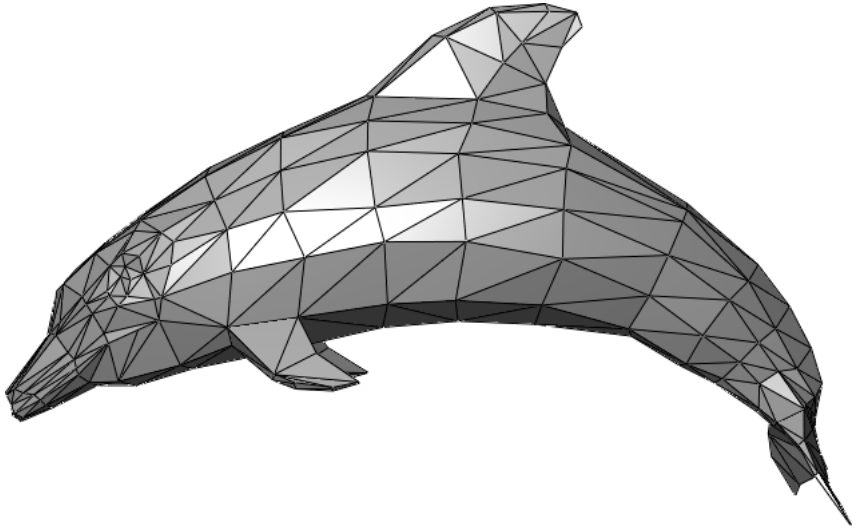
Edge

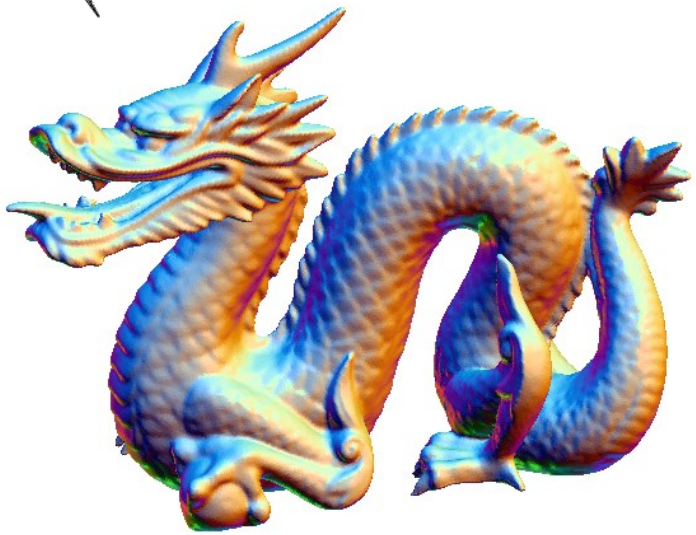
- Connects 2 vertices
- Shared by 2 polygons, if the surface is closed

Polygon

- Sequence of, at least, 3 vertices

Polygonal Meshes





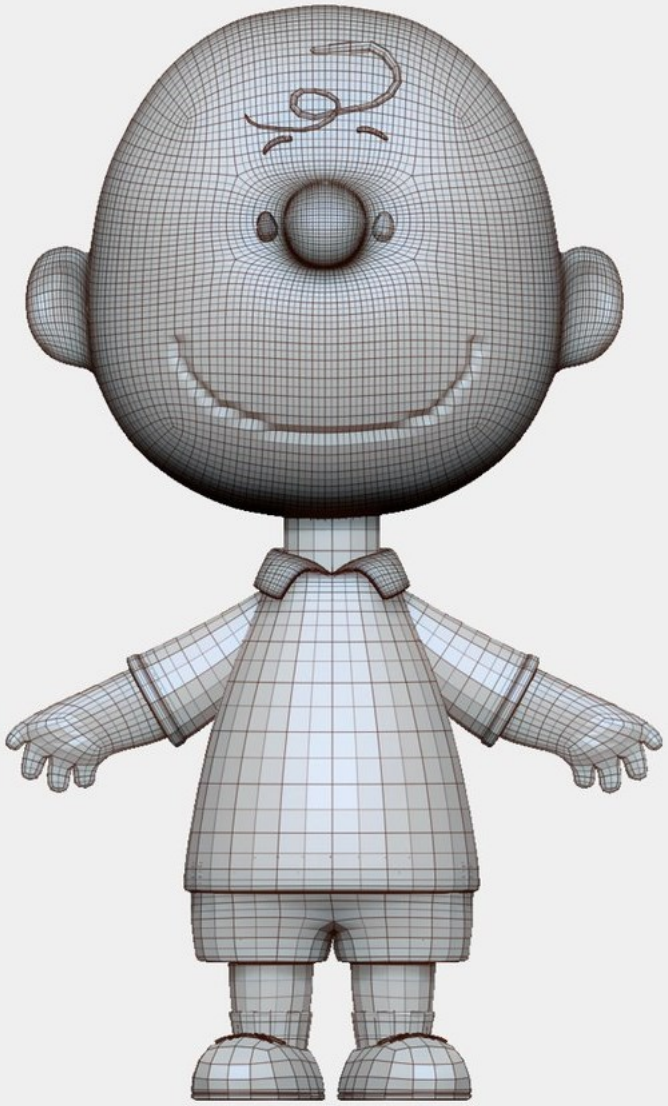
Polygonal Meshes

Supported by most applications

Various **file formats**

Triangle meshes are the most common !!

- Planar faces
- Algorithm simplicity
- Numerical robustness
- Efficient rendering



Polygonal Meshes

Exact vs. approximate rep. – When ?

- Polyhedral models
- Curved surfaces
- Terrain models
- More complex models / surfaces

A “good” approximation might require a large number of faces

- Levels-of-Detail (**LODs**)

Polyhedral models

The same polyhedral model might be represented by **different polygonal meshes** !!

- Useful for shading / rendering

Degrees of freedom

- **Number** of mesh vertices
- **Distribution** of mesh vertices
- **Arrangement** of edges / polygons

Example

- Represent a cube using different polygonal meshes

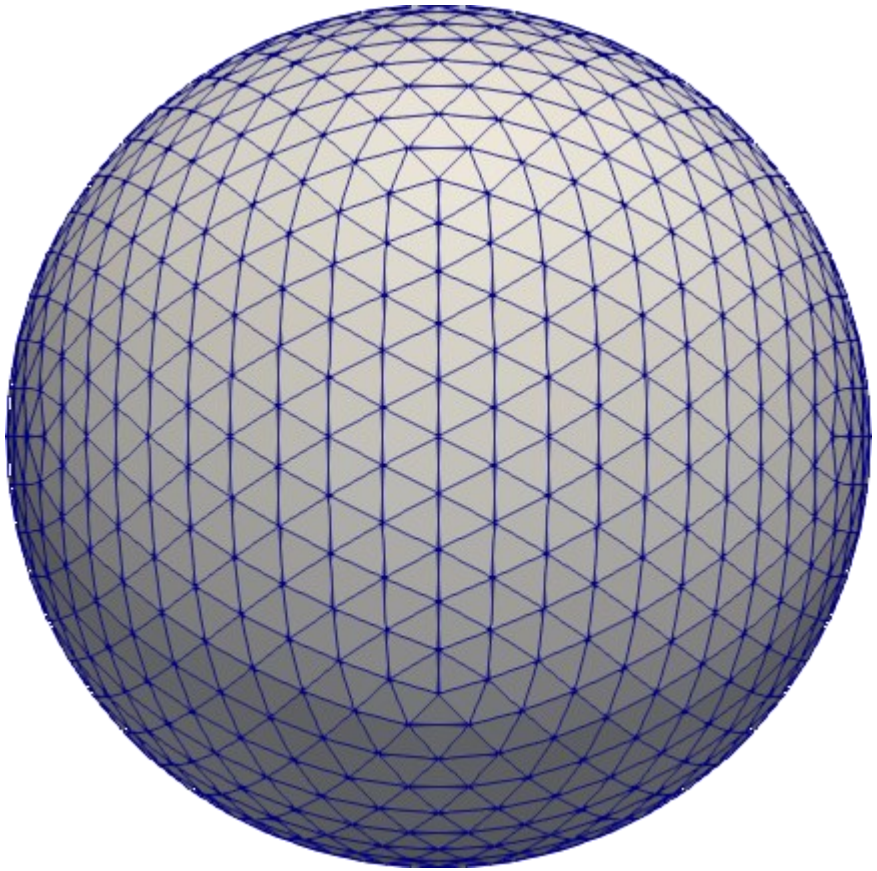
Curved surfaces

Representing the shape of a curved surface is an **approximation** process

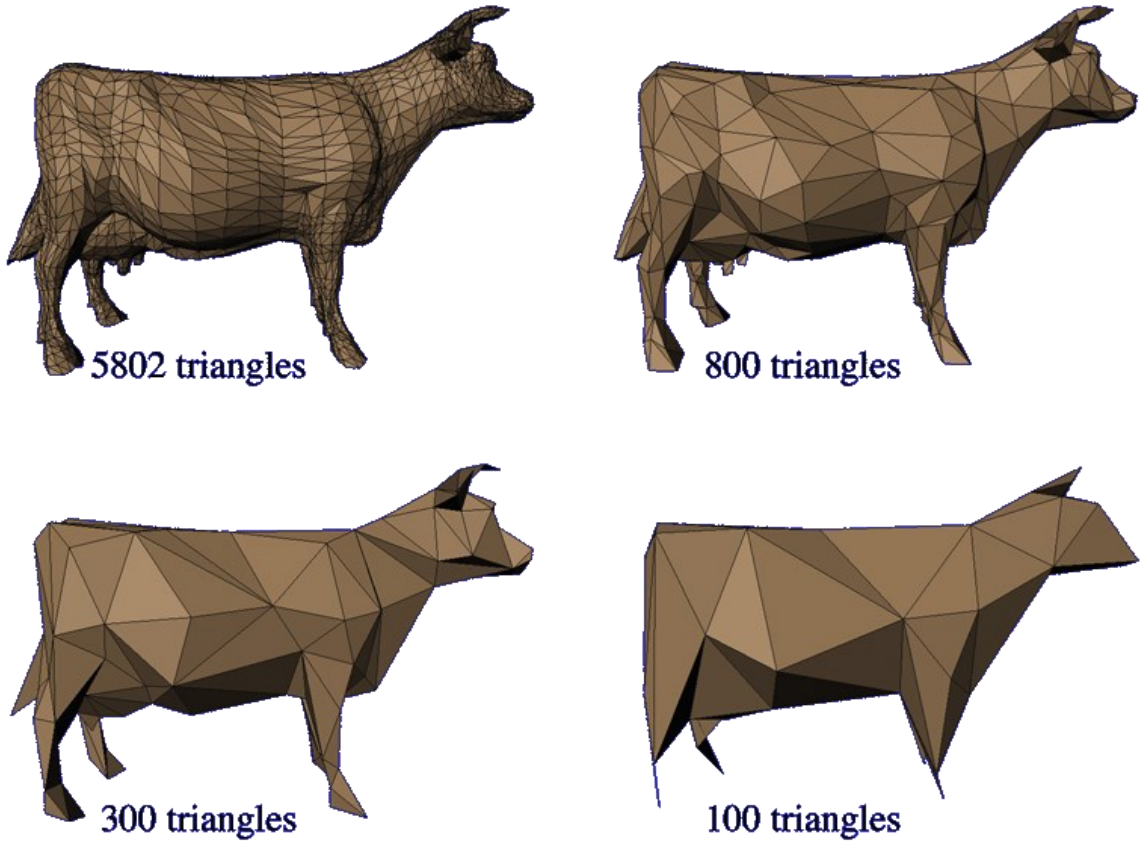
There is no “unique” model

Degrees of freedom

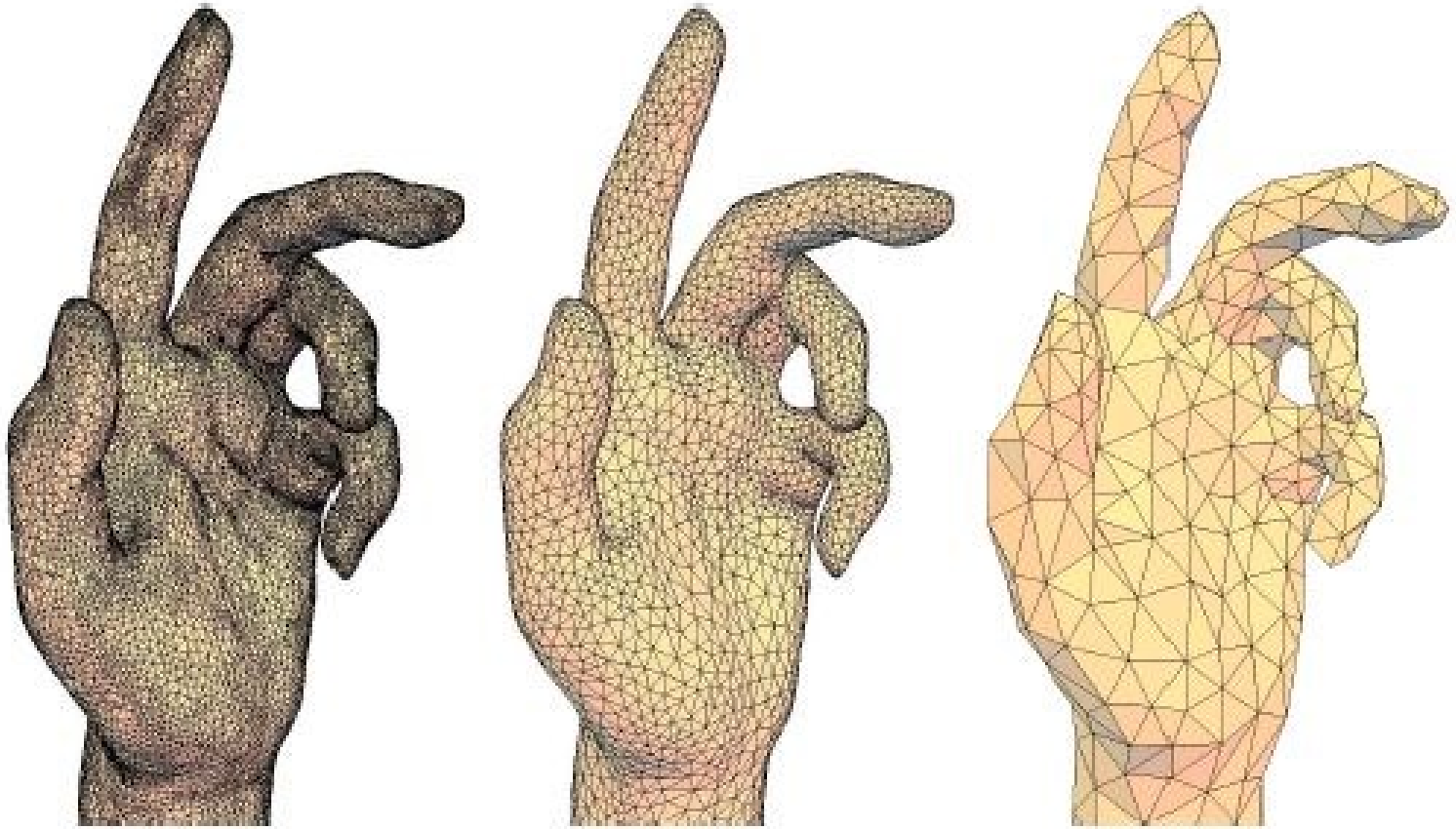
- **Number** of mesh vertices
- **Distribution** of mesh vertices
- **Arrangement** of edges / polygons



How many triangles should be used?



How many vertices should be used?



(a) 25,000 vertices. (b) 5,000 vertices. (c) 500 vertices.

Criteria to Define Number of Vertices

Smoothness

- Differential geometry
- Curvature ?
- Triangle quality ?
- …

Complexity

- Number of vertices / polygons
- Memory space / File size / bandwidth
- Computational cost of usual operations

Criteria – Restrictions

Least admissible smoothness

- Screen resolution ?
- Perception ?
- User studies

Largest admissible complexity

- Processing / rendering speed
- Memory space / File size

Balance ?

How do we adjust mesh density?


Mesh Decimation
and Refinement

How to adjust?

Refinement

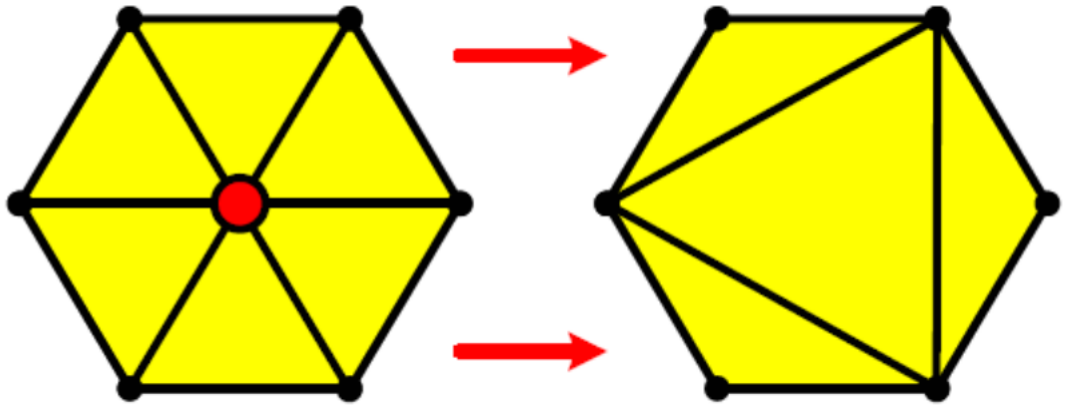
- Increase surface smoothness
- How to compute new vertices and polygons ?
- Where ?

Decimation

- Decrease the number of vertices / polygons !!
- Which edges / polygons should be collapsed ?
- Where ?

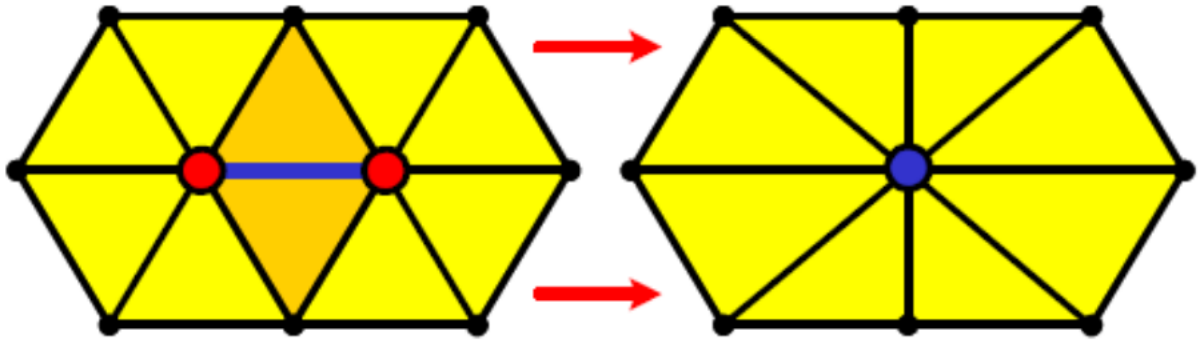
Basic decimation operations

Vertex removal



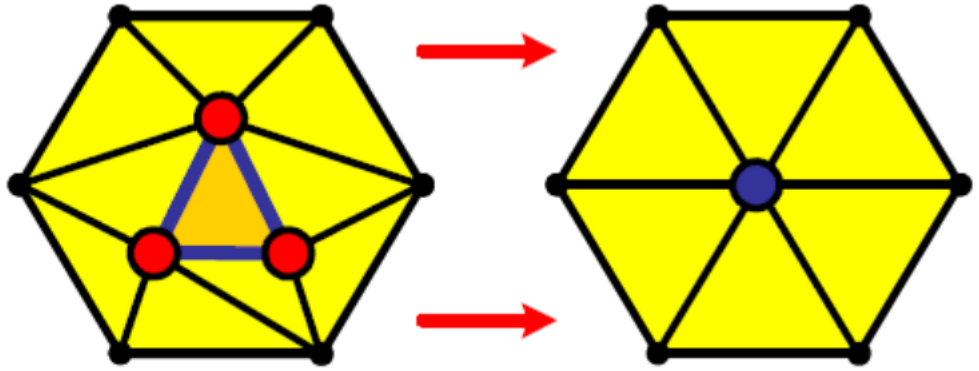
Basic decimation operations

Edge collapse



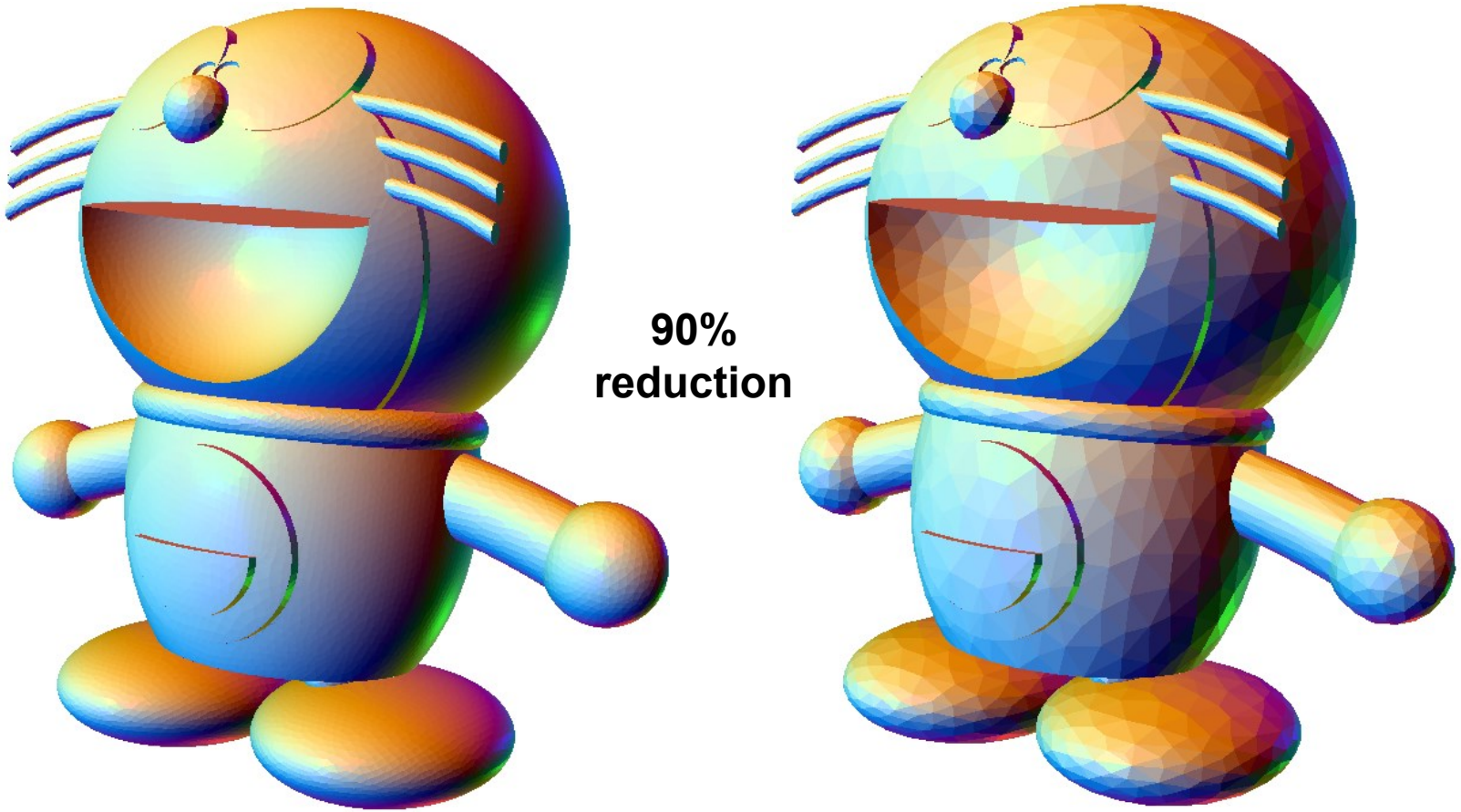
Basic decimation operations

Triangle collapse



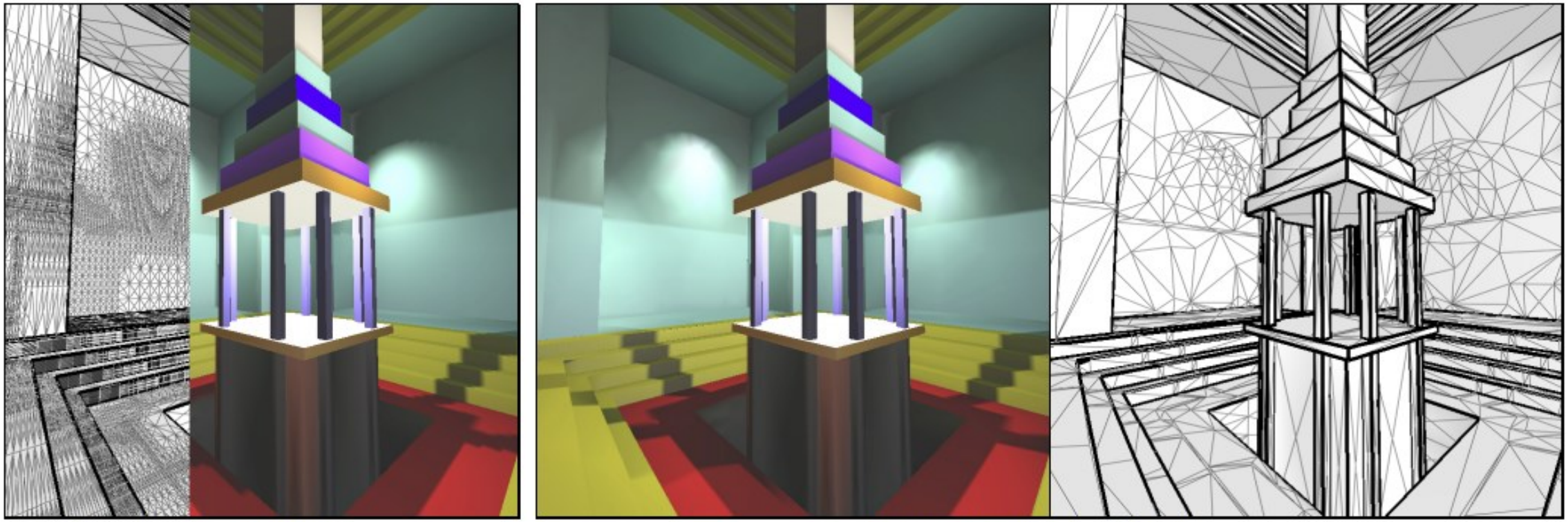
http://staff.ustc.edu.cn/~lgliu/Courses/DGP_2012_spring-summer/Courseware/DGP10_MeshSimplification.pdf

Mesh decimation



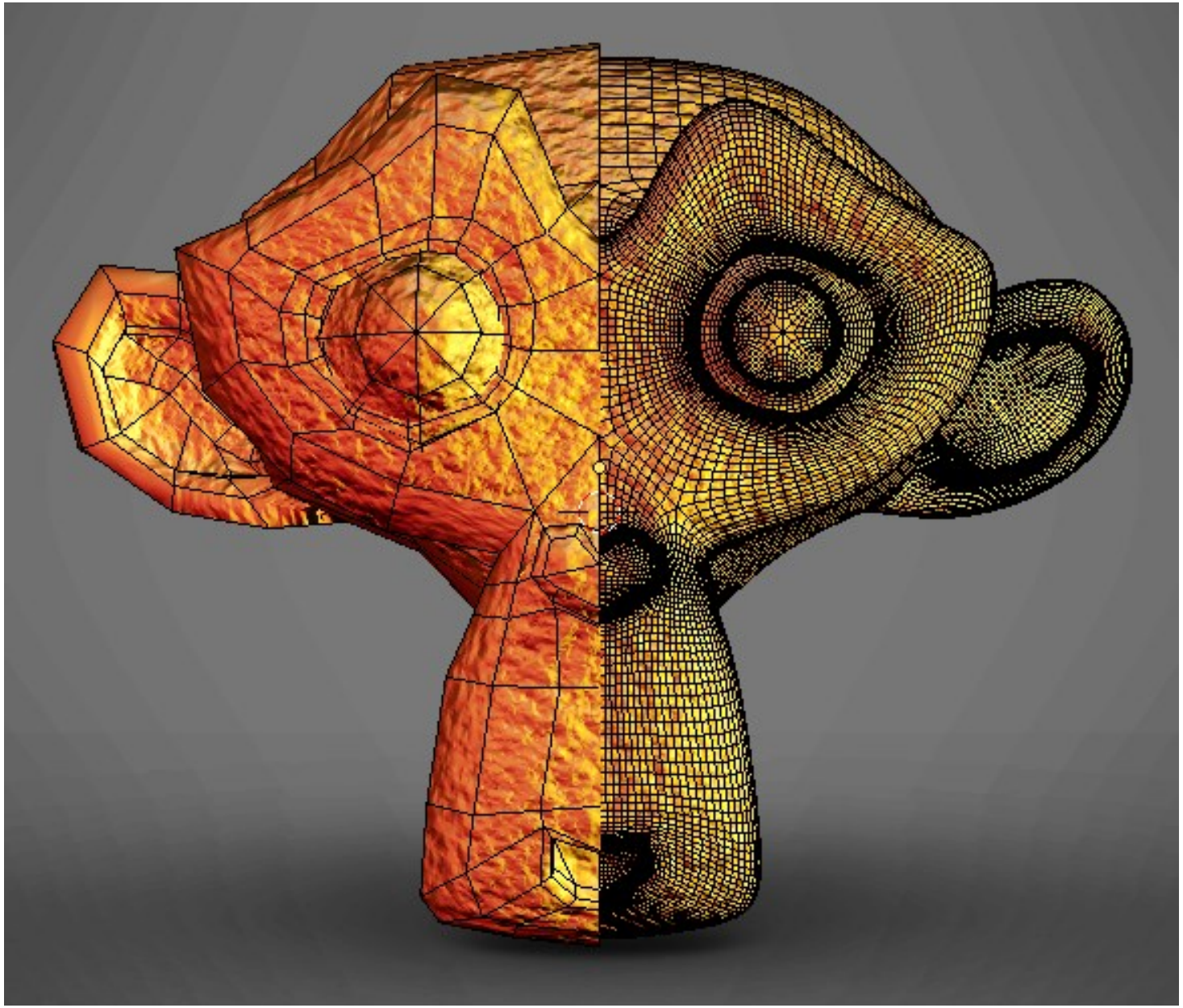
[Seidel and Belyaev, 2006]

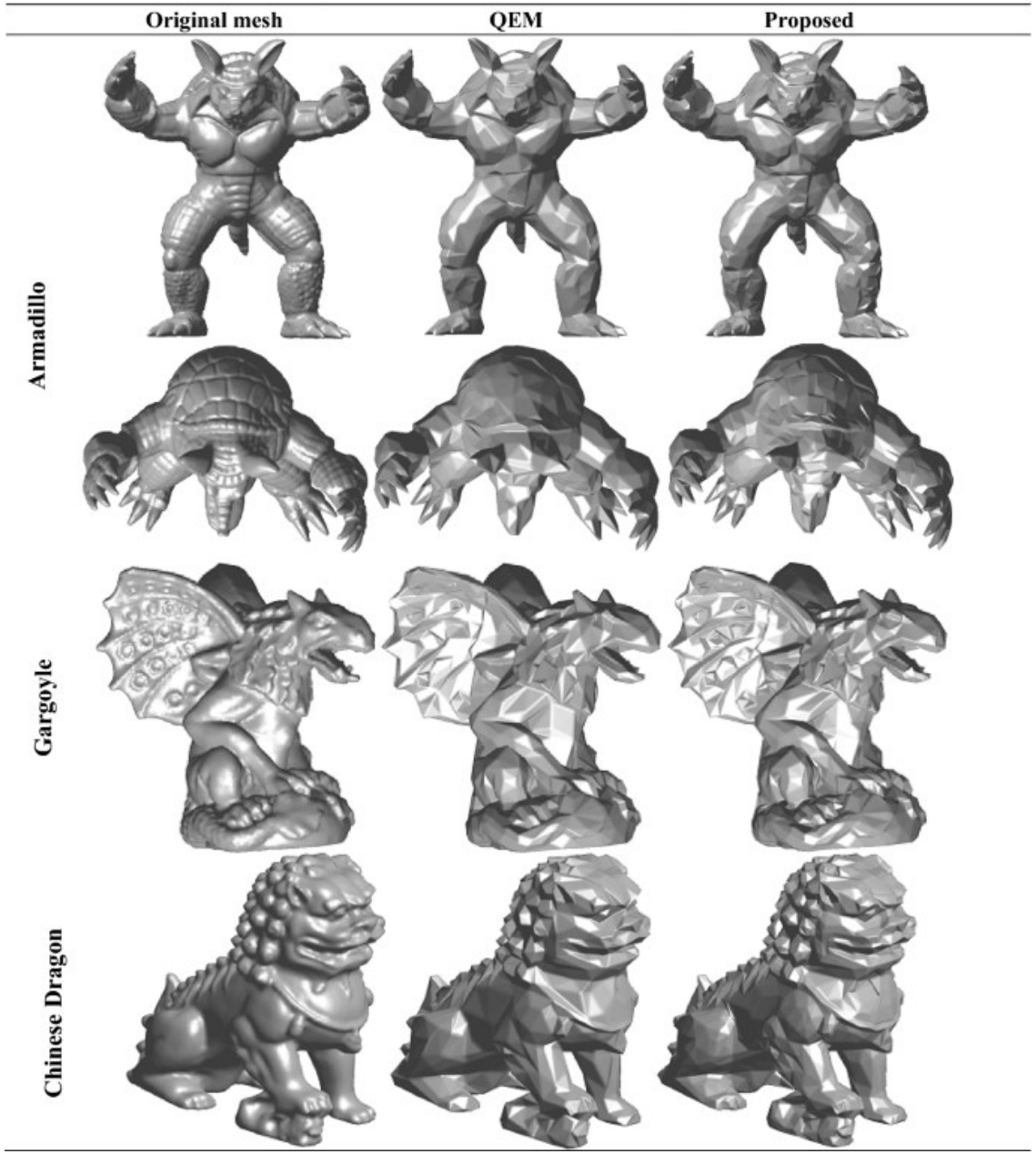
Mesh decimation

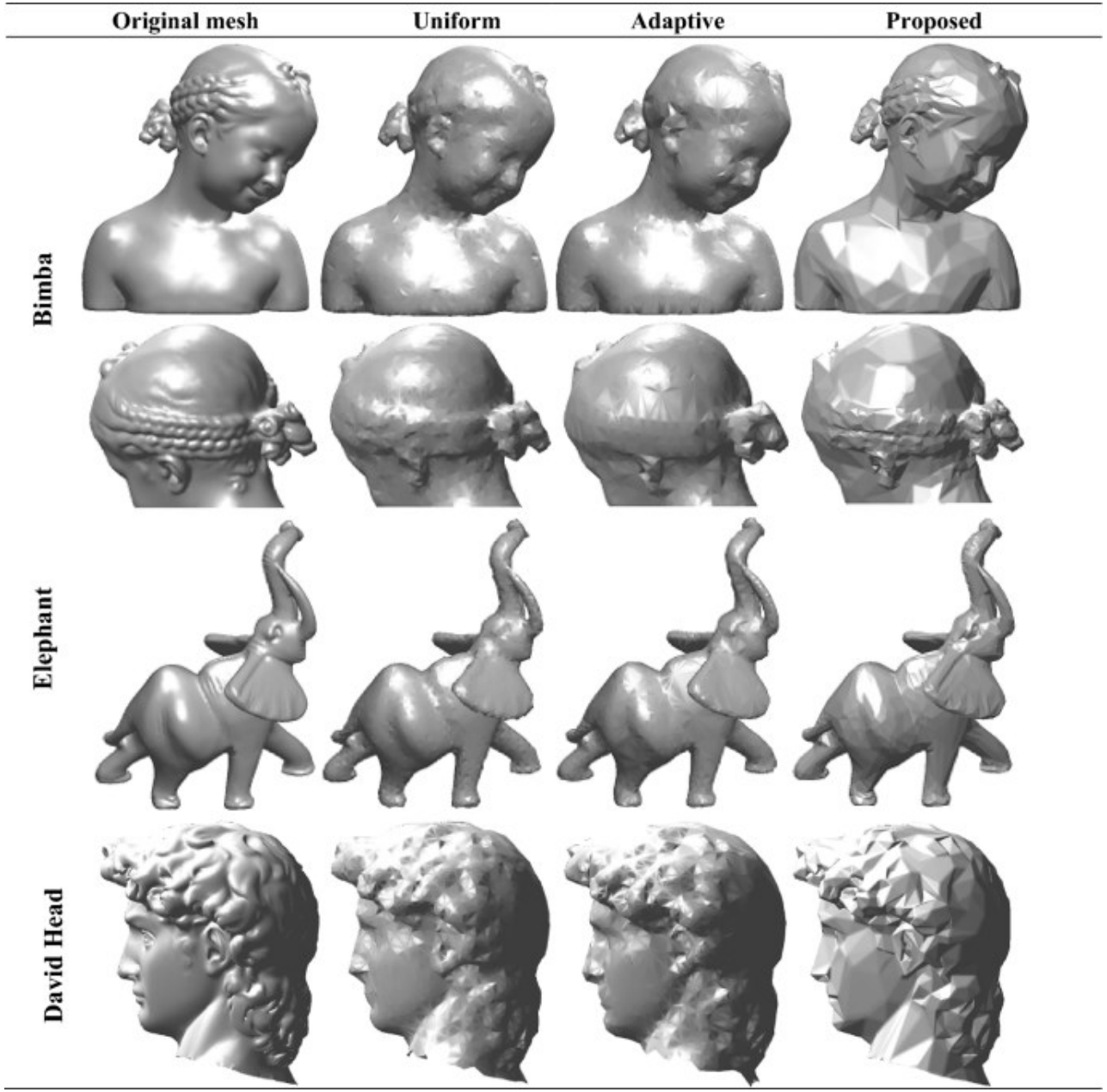


(a) Original mesh (298,468 faces)

(b) Simplified mesh (5,000 faces)



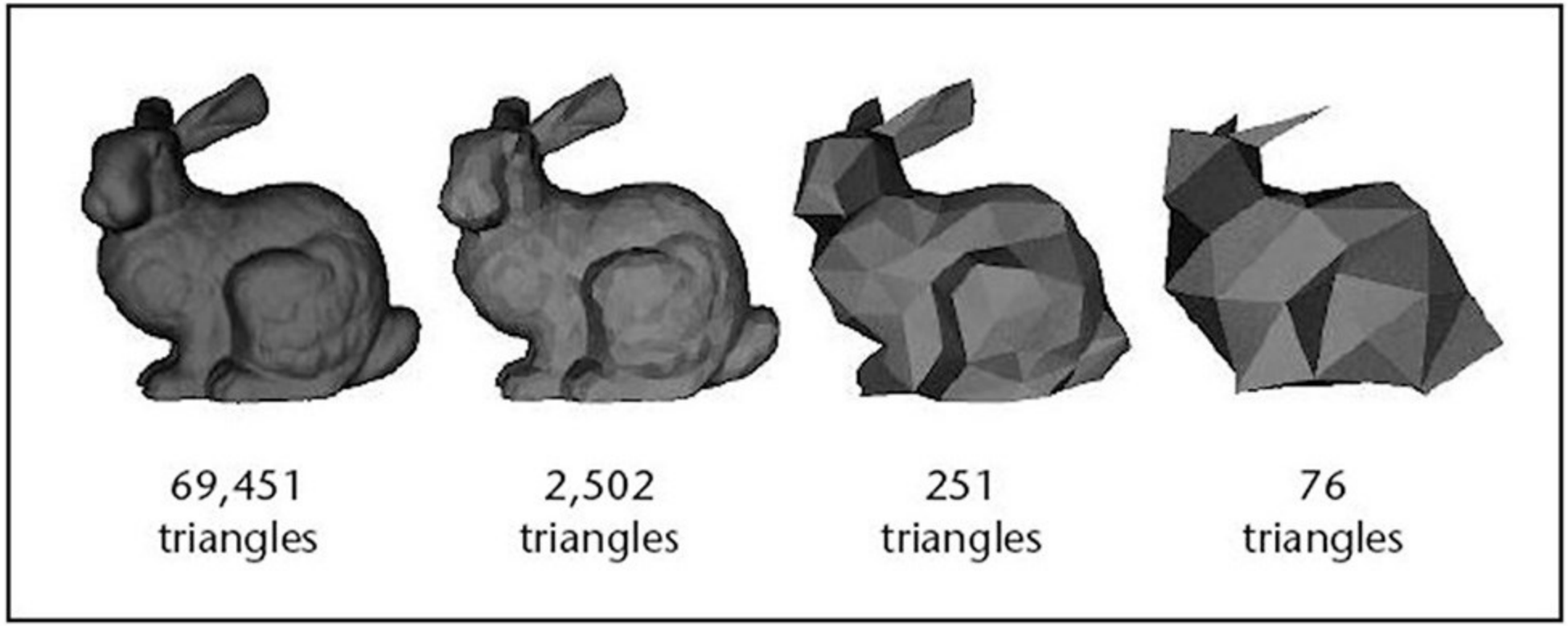


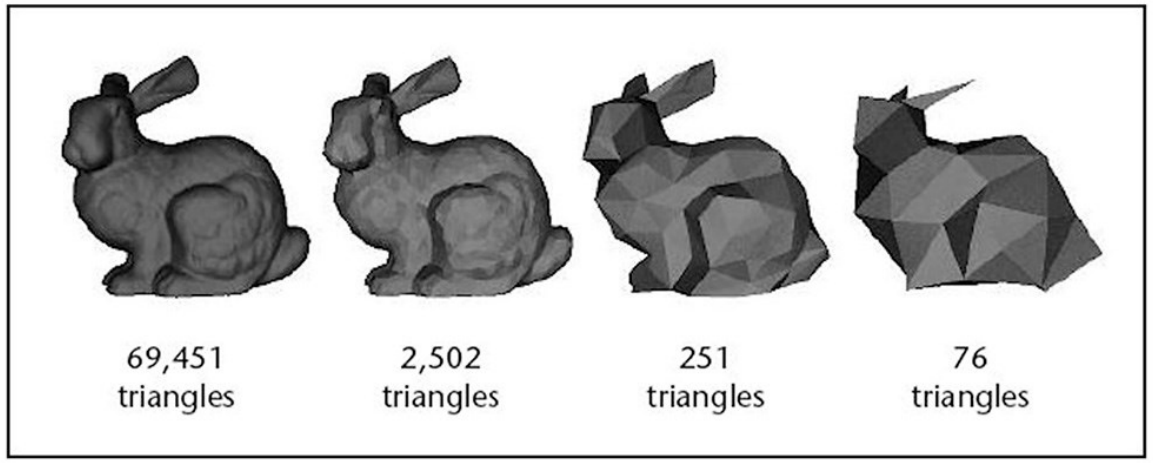


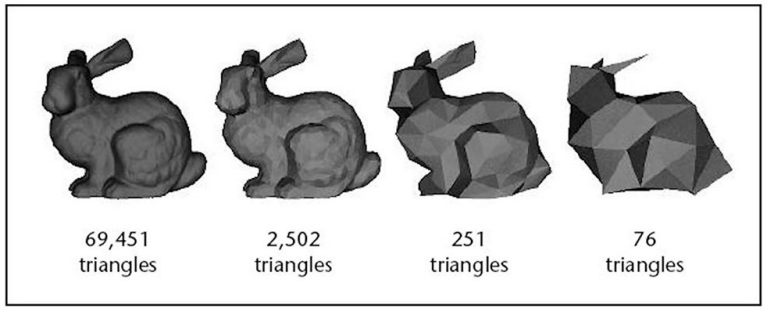
https://media.springernature.com/lw685/springer-static/image/art%3A10.1007%2Fs12650-022-00828-9/MediaObjects/12650_2022_828_Fig12_HTML.png

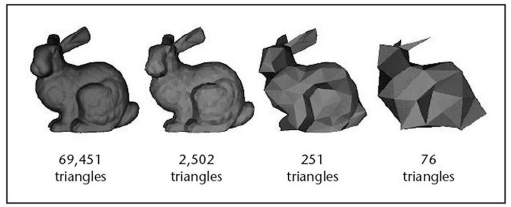
Level-of-Detail

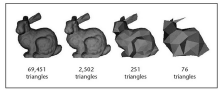
https://3dstudio.co/3d-lod-level-of-detail/


69,451
triangles
2,502
triangles
251
triangles
76
triangles







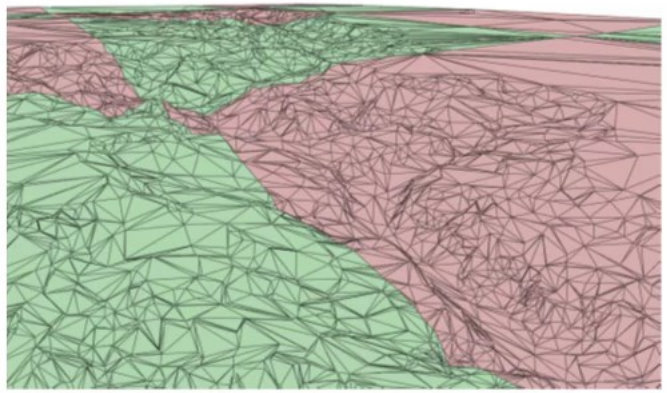


Our detail perception for distant (or peripheral) objects is poor

So, the polygonal complexity may be adjusted for those cases

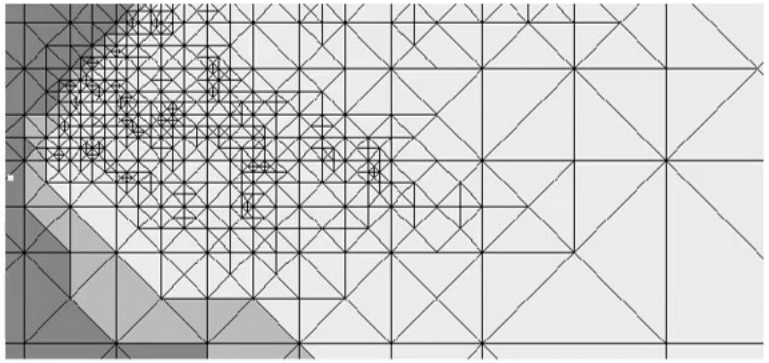
Mesh LOD techniques

Irregular meshes



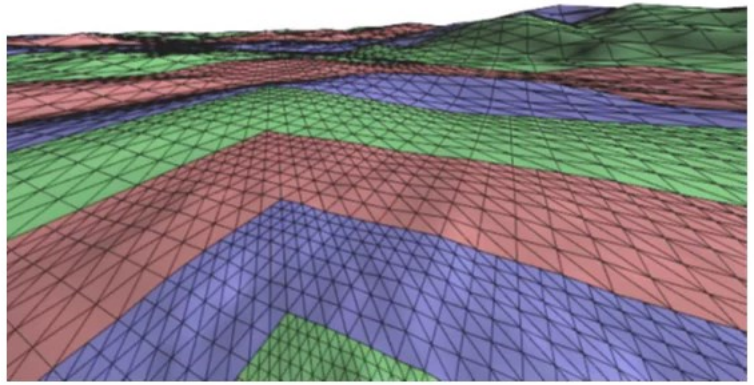
Smooth view-dep LOD [Vis 1998]

Semi-regular meshes



ROAM [Vis 1997]

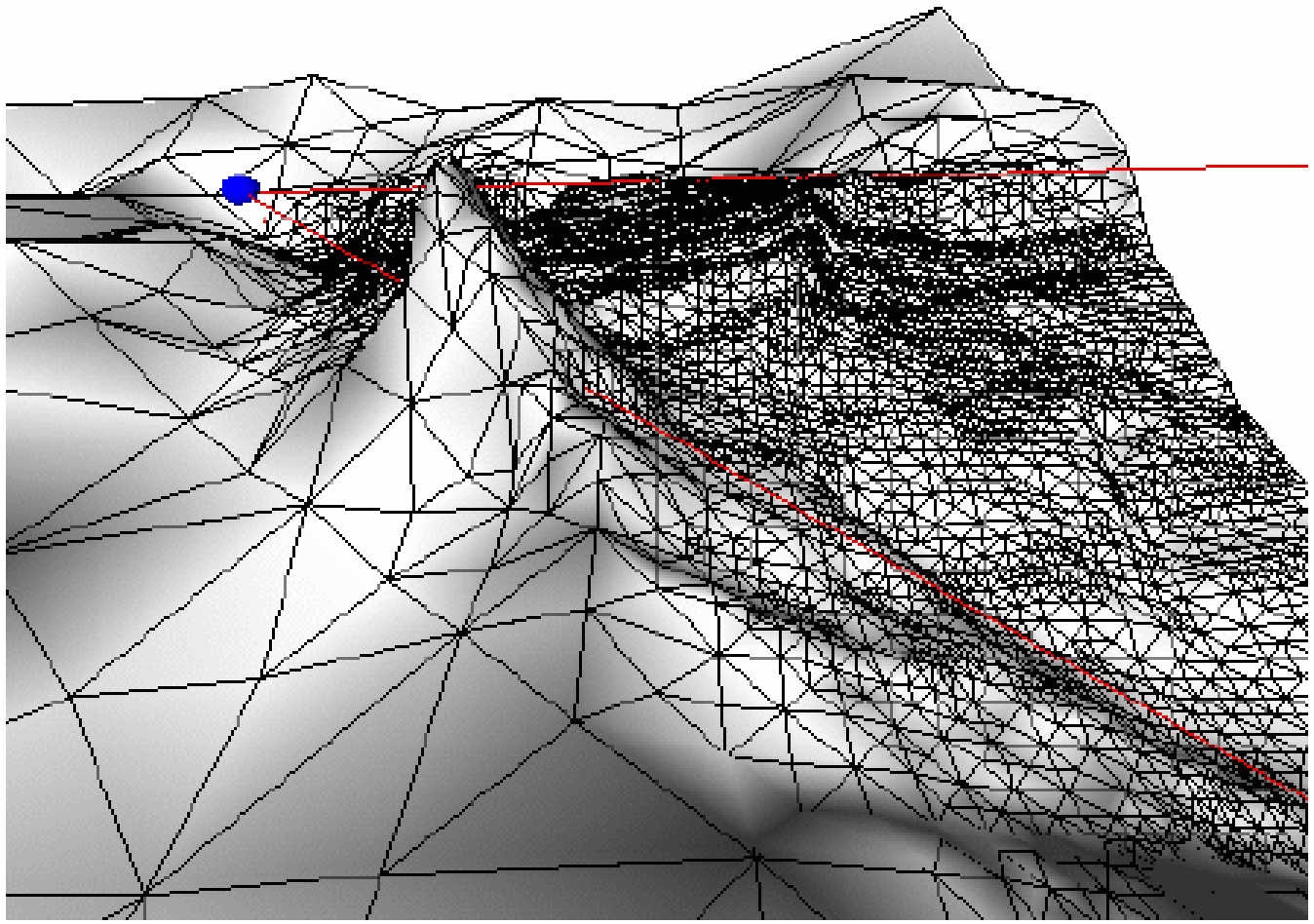
Regular meshes



Geometry clipmaps [2004]

https://www.youtube.com/watch?v=EoQY9H7T0IU&ab_channel=HuguesHoppe

View dependent simplification



Nanite

Artistic freedom in using the best quality possible assets.
Move optimization into the engine...

Standard Practice

To achieve visual quality reducing complexity

Have a high poly count model

Generate normal maps / light maps / etc.

Create low poly model and apply normal maps

Ever heard of baking?...

With Nanite

Choose the asset I want

The end

(for the designer/developer... actually, the engine will take care of the rest)

Nanite

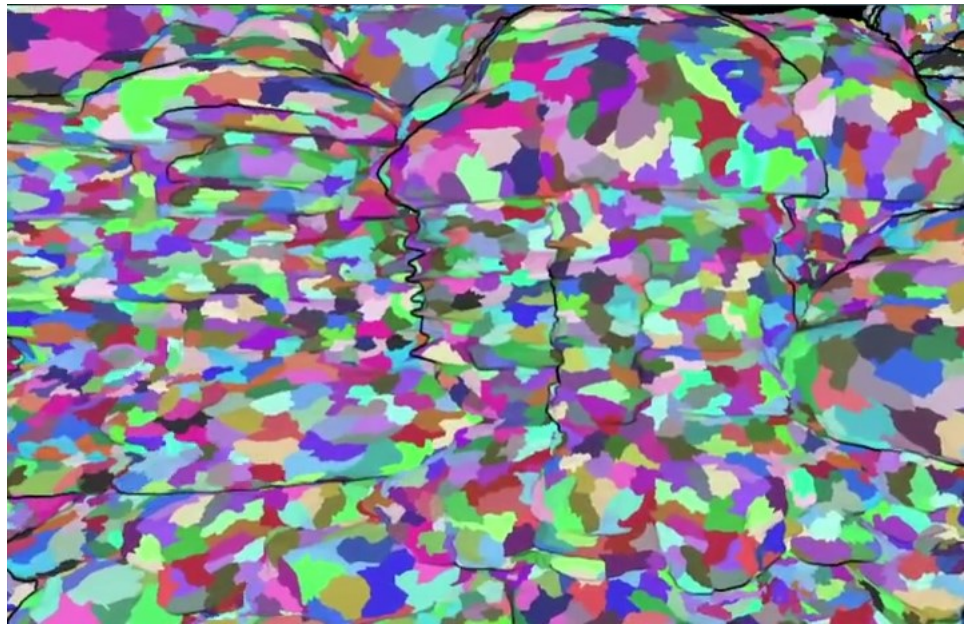
There is a fixed amount of pixels on screen

Draw the same number of clusters every frame regardless of scene complexity

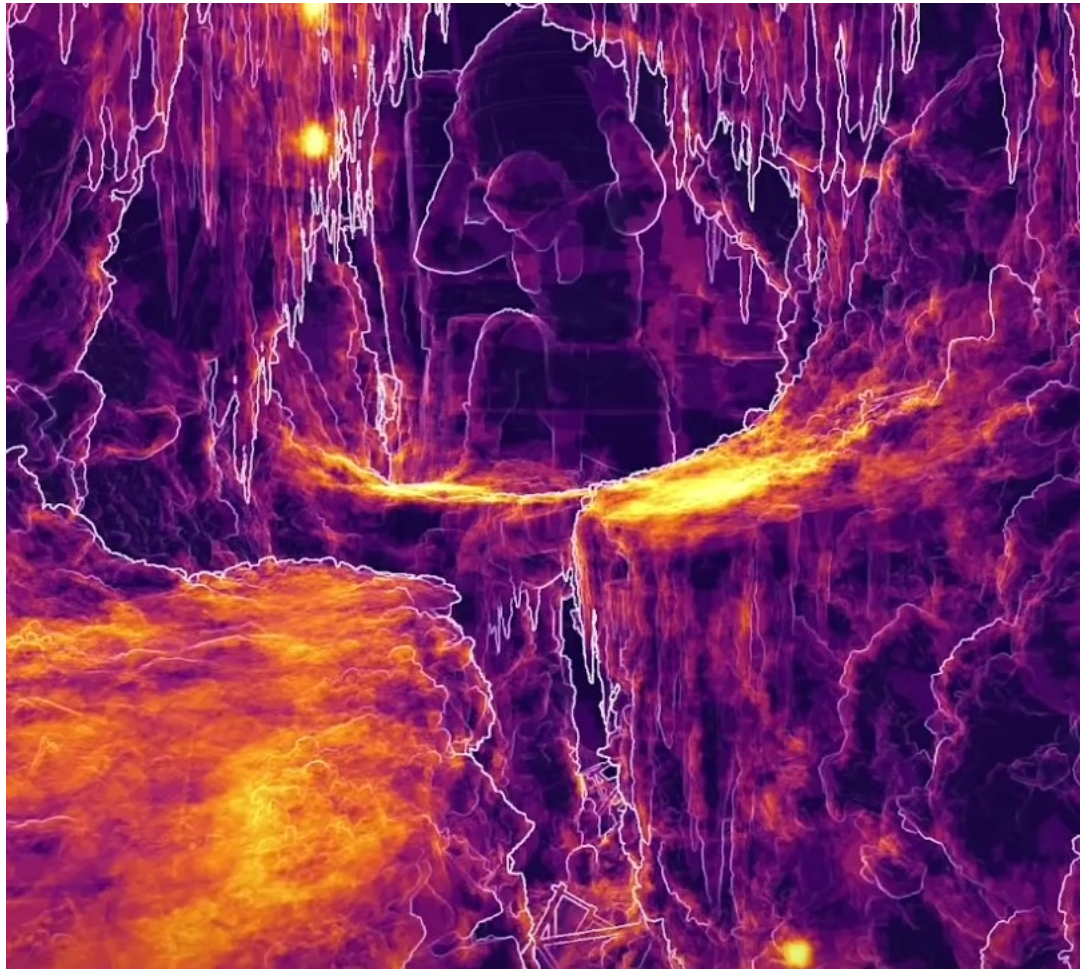
The cost of rendering geometry should scale by screen resolution and not scene complexity

Nanite

cluster culling



overdraw

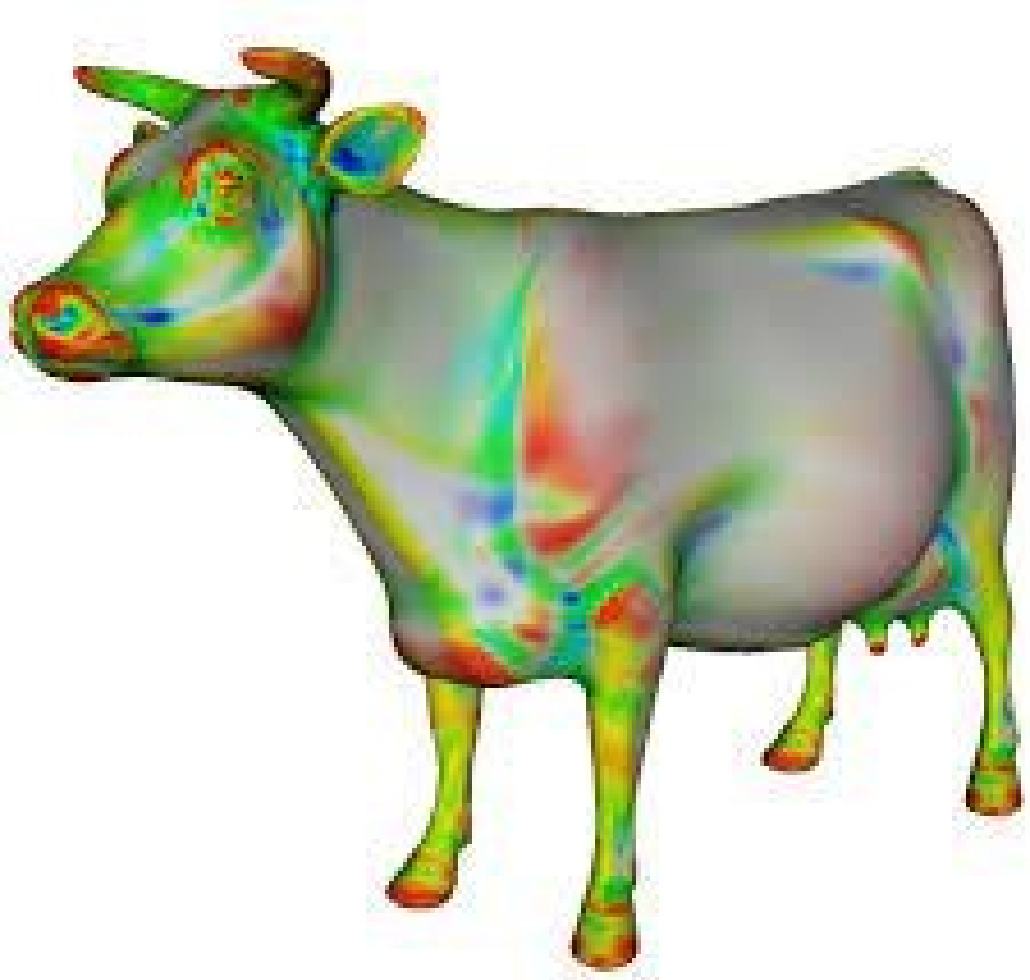


https://www.youtube.com/watch?v=eviSykqSUUw&ab_channel=SIGGRAPHAdvancesinReal-TimeRendering
Theory: https://d-nb.info/997062789/34

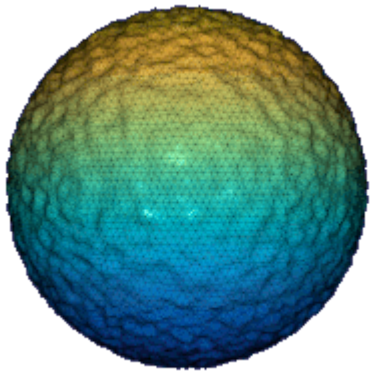
Mesh Processing

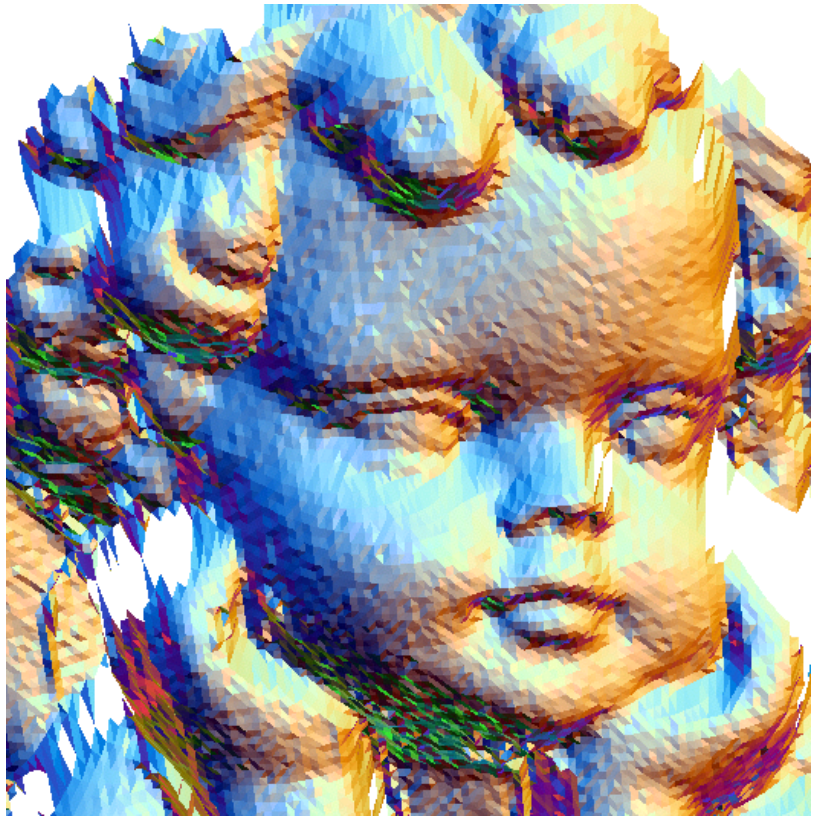
HÅKON ANTON FAGERÅS

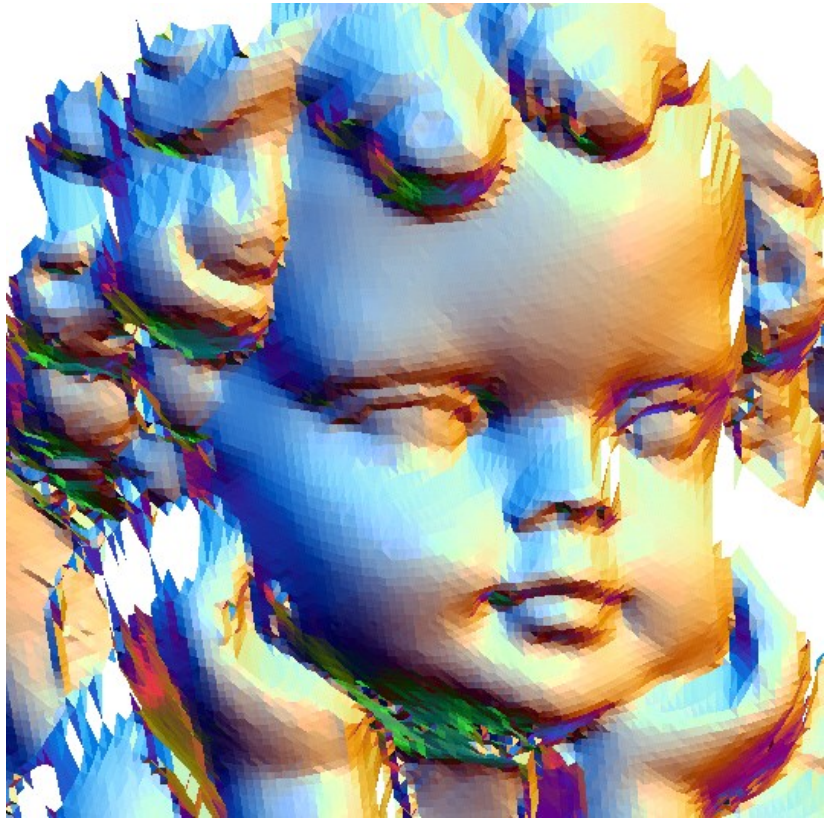
Random slide with a cow



Mesh smoothing







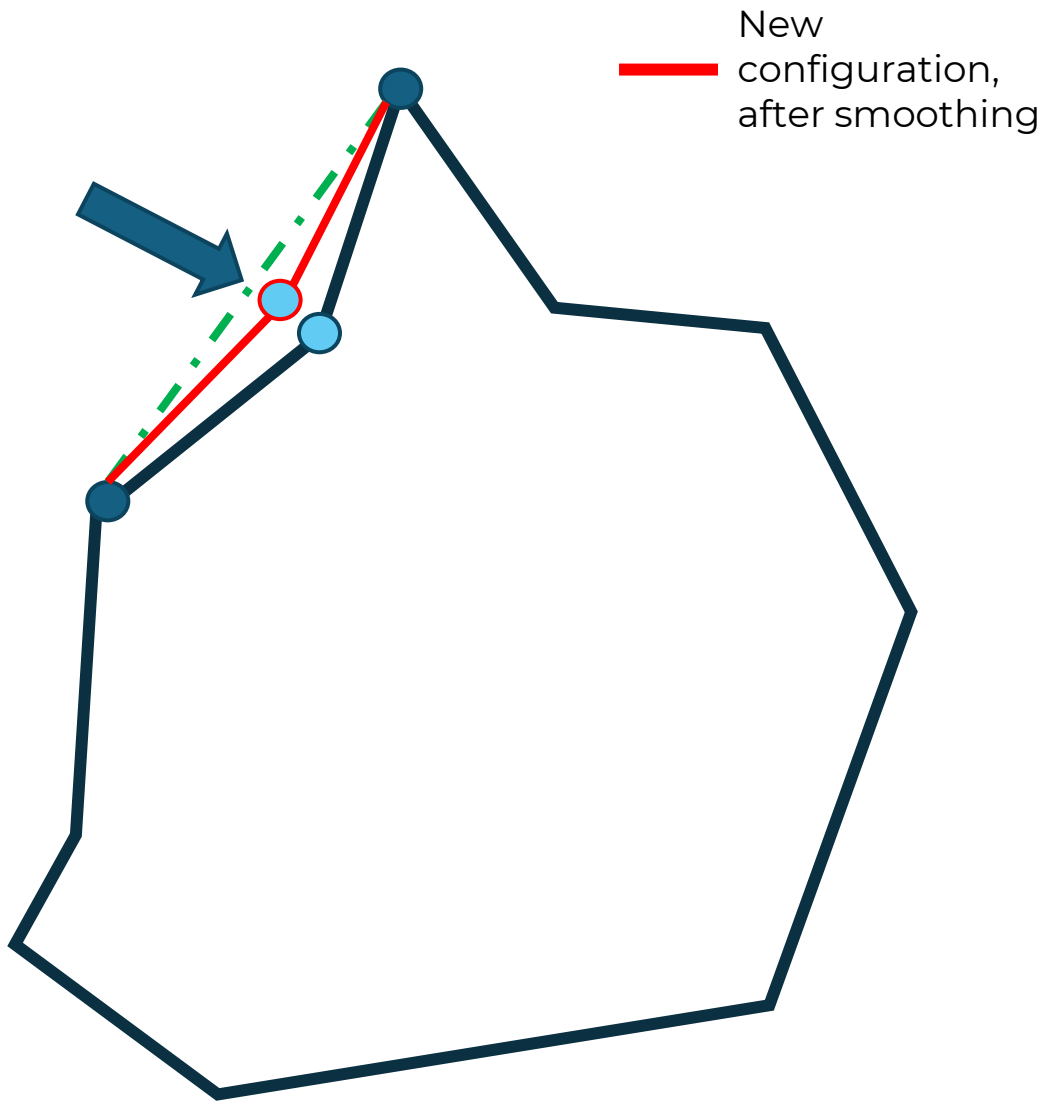
[Seidel and Belyaev, 2006]

What do We Need to Perform Smoothing?

Example of Gaussian Smoothing

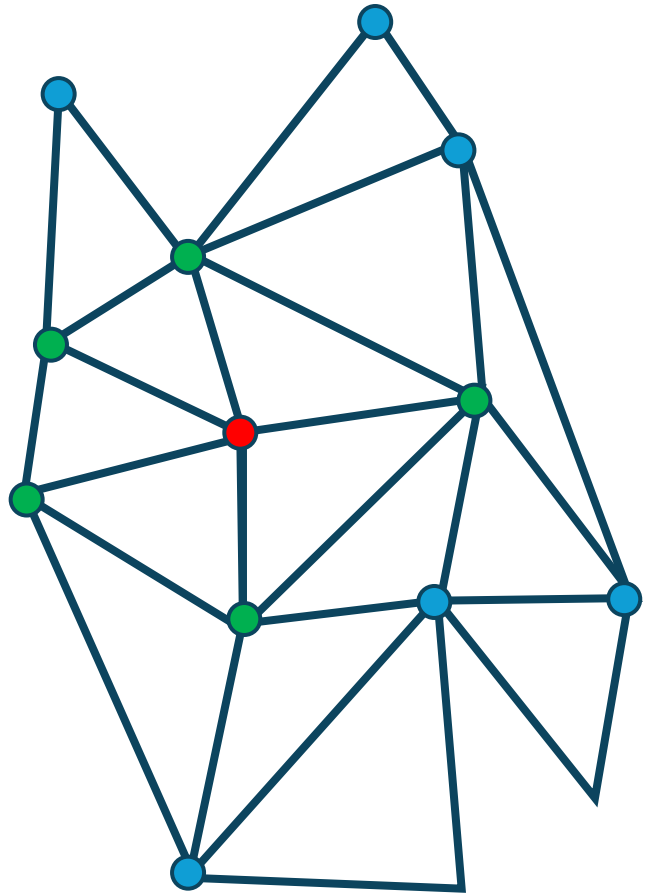
- Average position of a vertex considering its neighbourhood

So, we need to have more than vertex position…



Some basic operations

- Find the vertices defining an edge
- Find the edges incident in a vertex
- Find all polygons sharing
  - A vertex
  - An edge
- Identify mesh errors. I.e., the lack of
  - A vertex / an edge / a face
- Rendering a mesh




Computational
Representation

Polygonal Mesh Representation

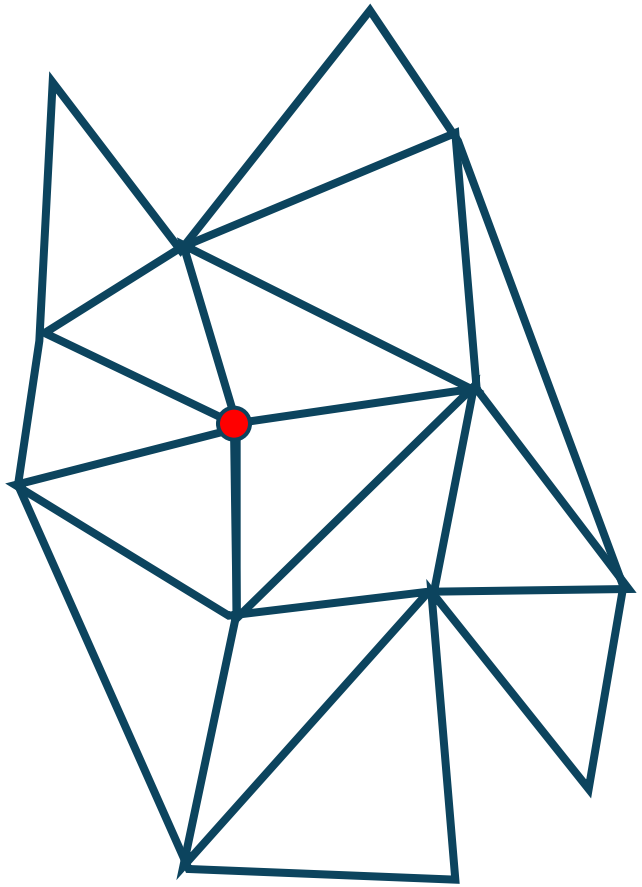
- Which **geometric information** ?
  - Vertex coordinates
- Which **topological information** (i.e., connectivity) ?
  - How are edges and faces arranged ?
  - How to identify **neighboring / incident /adjacent** entities ?
- Which **additional properties**?
  - Normal vector to each face / vertex
  - Texture coordinates
- How to check the **validity** of a model ?
  - 2-manifolds
  - Euler Formulae

Face-Vertex List

- Vertices list / array
  - Store just once the coordinates of each vertex !
  - Easy to edit / modify one vertex
- Each polygon is described by its vertices sequence
  - Pointer / index
  - Usage : storing in a file (e.g., **OBJ** format)
- **Inefficient**
  - Hard to detect which polygons share a given edge !!
  - Rendering : edges are drawn twice !!

Topological information

- How to store incidence and adjacency information ?
- How to answer **basic queries** fast ?
  - Which are the **end vertices** of an edge ?
  - Which are the **adjacent polygons** of an edge ?
  - Which are the **neighboring vertices** of a vertex ?
  - …
- Efficiency
  - Time ?
  - Space ?



Data Structures for Mesh Representation

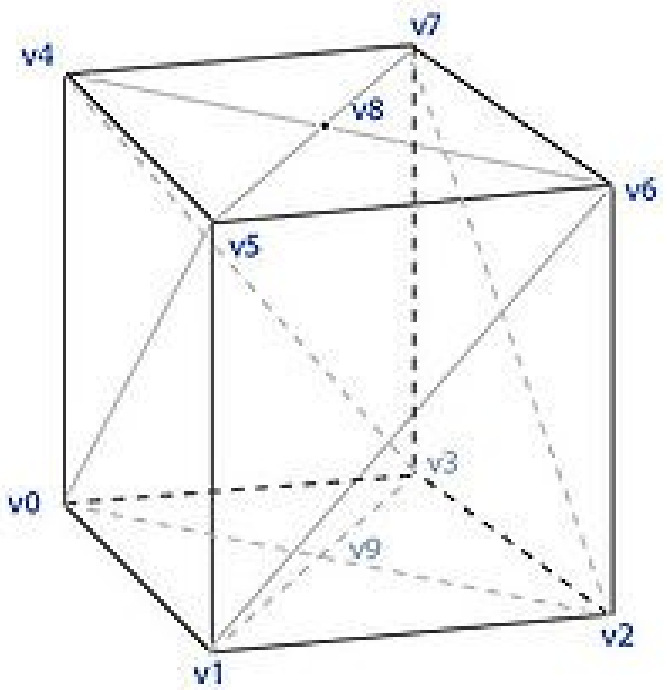
What do we need to store?

Adjacent Vertices List

Vertex-Vertex Meshes (VV)

Vertex List

v0	0,0,0	v1 v5 v4 v3 v9
v1	1,0,0	v2 v6 v5 v0 v9
v2	1,1,0	v3 v7 v6 v1 v9
v3	0,1,0	v2 v6 v7 v4 v9
v4	0,0,1	v5 v0 v3 v7 v8
v5	1,0,1	v6 v1 v0 v4 v8
v6	1,1,1	v7 v2 v1 v5 v8
v7	0,1,1	v4 v3 v2 v6 v8
v8	.5,.5,0	v5 v6 v7 v8
v9	.5,.5,1	v0 v1 v2 v3



Vertices List + Faces List

Face-Vertex Meshes

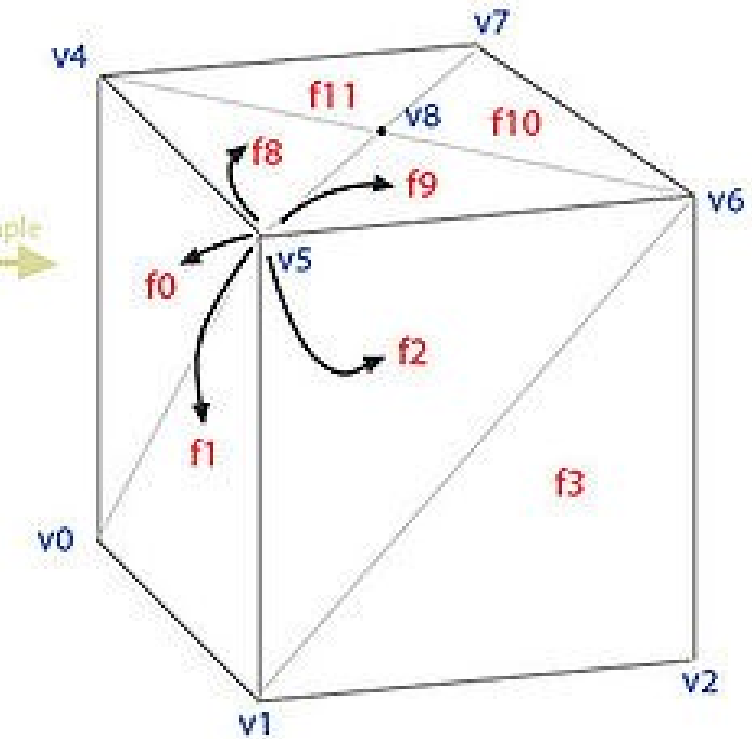
Face List

f0	v0 v4 v5
f1	v0 v5 v1
f2	v1 v5 v6
f3	v1 v6 v2
f4	v2 v6 v7
f5	v2 v7 v3
f6	v3 v7 v4
f7	v3 v4 v0
f8	v8 v5 v4
f9	v8 v6 v5
f10	v8 v7 v6
f11	v8 v4 v7
f12	v9 v5 v4
f13	v9 v6 v5
f14	v9 v7 v6
f15	v9 v4 v7

Vertex List

v0	0,0,0	f0 f1 f12 f15 f7
v1	1,0,0	f2 f3 f13 f12 f1
v2	1,1,0	f4 f5 f14 f13 f3
v3	0,1,0	f6 f7 f15 f14 f5
v4	0,0,1	f6 f7 f0 f8 f11
v5	1,0,1	f0 f1 f2 f9 f8
v6	1,1,1	f2 f3 f4 f10 f9
v7	0,1,1	f4 f5 f6 f11 f10
v8	.5,.5,0	f8 f9 f10 f11
v9	.5,.5,1	f12 13 14 15

example



Vertices, Edges and Faces Lists

Vertices list / array

Edges list / array, which references

- The respective end vertices
- The respective polygons

Polygons list / array, which references

- The respective edges

Rendering : draw edges!!

Issue

- How to identify the edges incident to a vertex ?

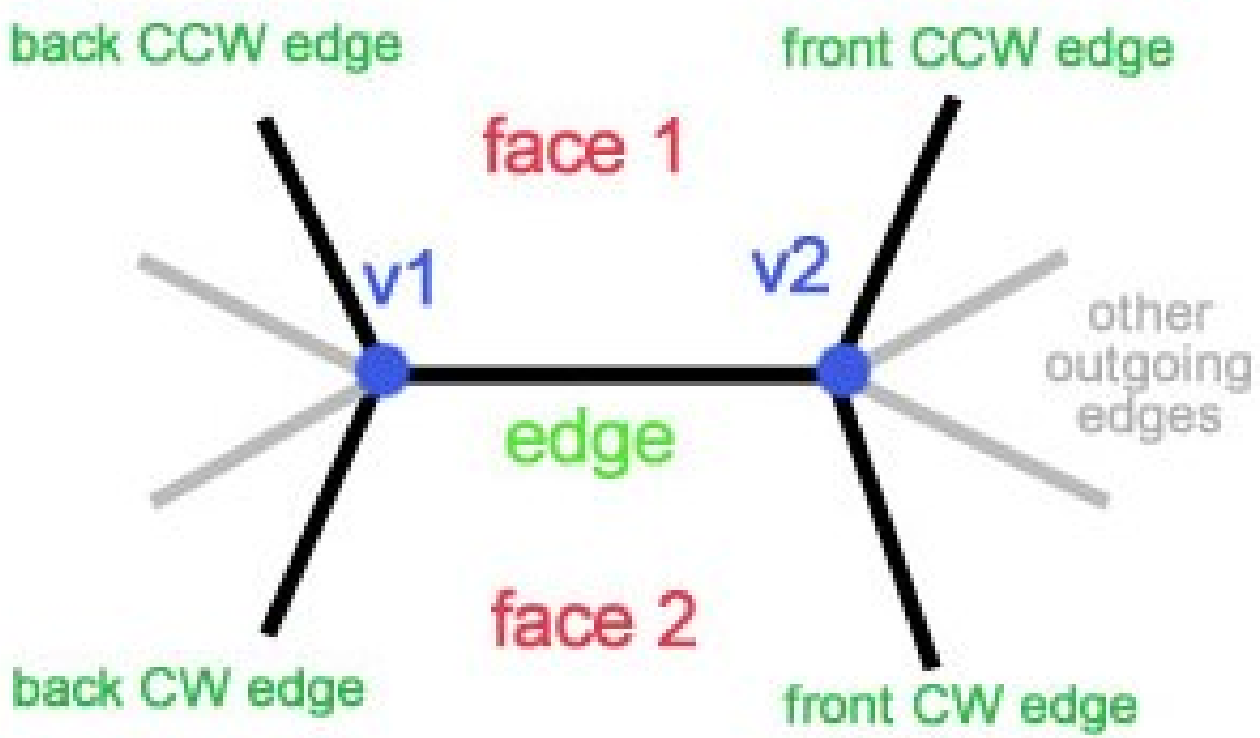
The “Winged-Edge” data structure

Explicit representation of vertices, edges and faces

Allows

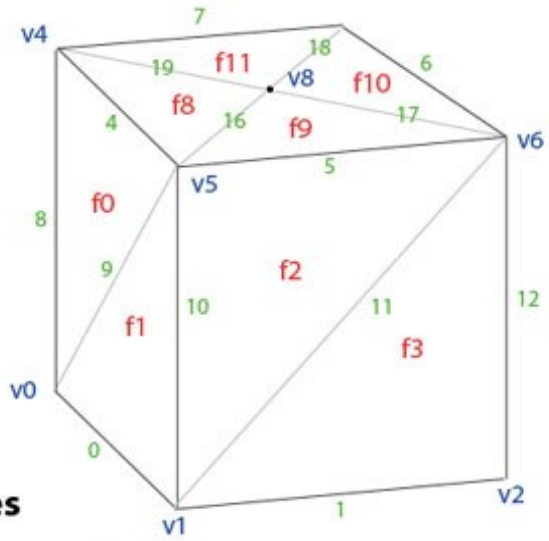
- Dynamic mesh modification
- Efficient answer to some “queries”

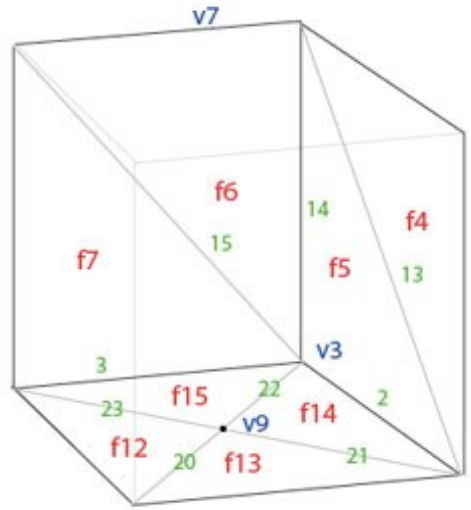
The “Winged-Edge” data structure



Winged Edge Structure

The “Winged-Edge” data structure





Winged-Edge Meshes

Face List

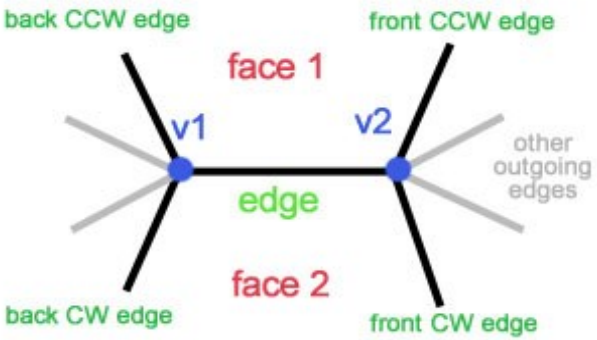
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



Winged Edge Structure

The “Winged-Edge” data structure

Face List

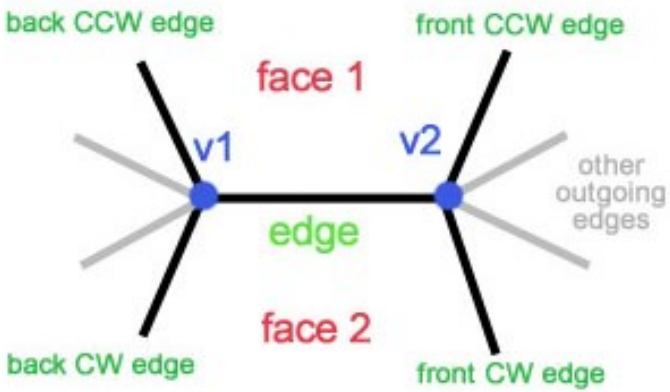
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

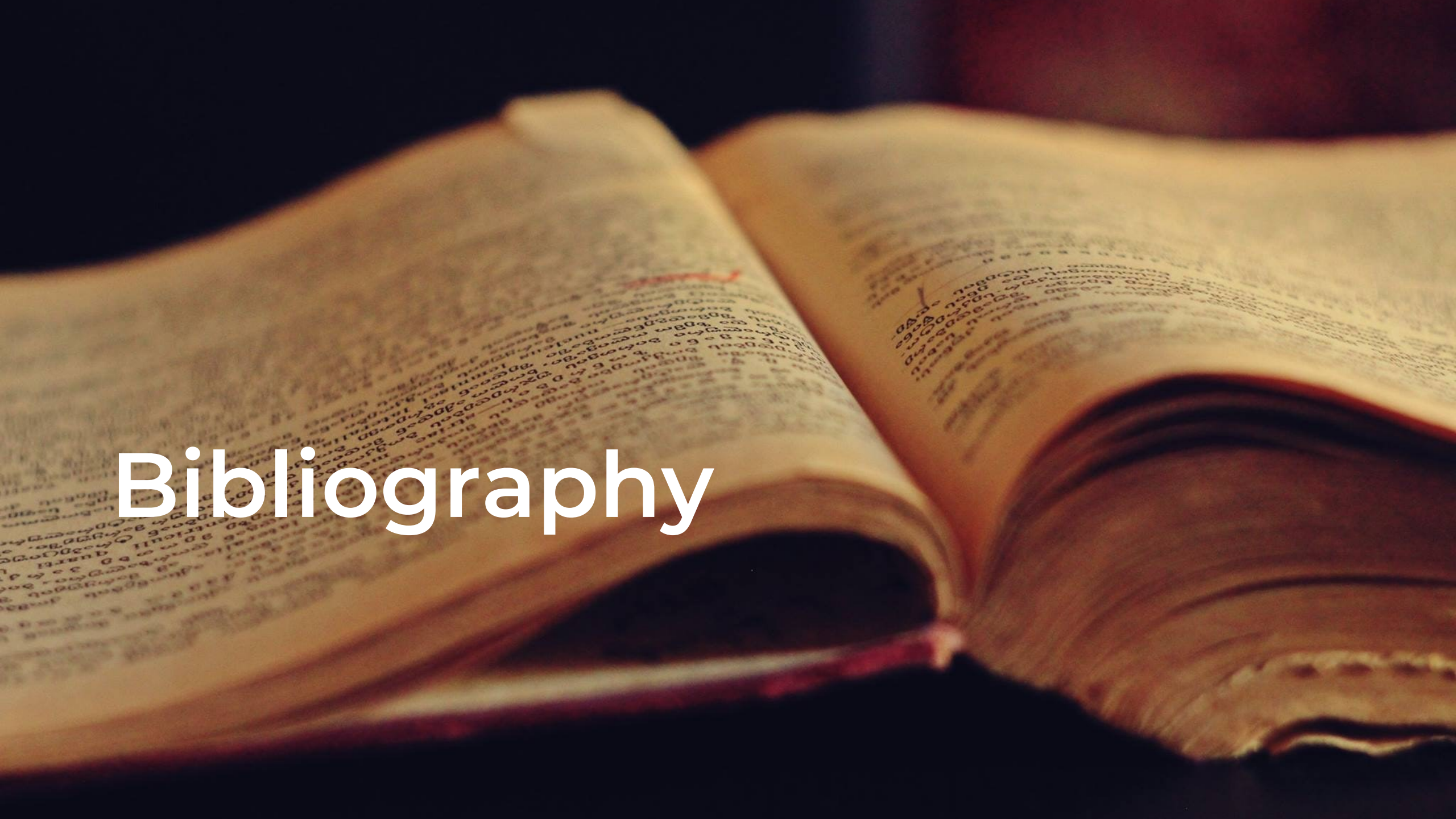
v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



Winged Edge Structure

Mesh libraries / toolboxes

- OpenMesh
  - https://www.graphics.rwth-aachen.de/software/openmesh/
- OpenFlipper
  - https://www.graphics.rwth-aachen.de/software/openflipper/
- CGAL – Comp. Geometry Algorithms Library
  - https://www.cgal.org/
- MeshLab
  - https://www.meshlab.net/
- …



Bibliography

Bibliography

- Botsch, M., & Pauly, M. (Eds.). (2007). Session details: Course 23: Geometric modeling based on polygonal meshes. In *ACM SIGGRAPH 2007 courses*. https://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.132.4099

- Luebke et al., “Level of Detail for 3D Graphics”, Morgan Kaufman, 2002, https://learning.oreilly.com/library/view/level-of-detail/9780080510118/

Hands On!