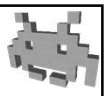
Course Plan



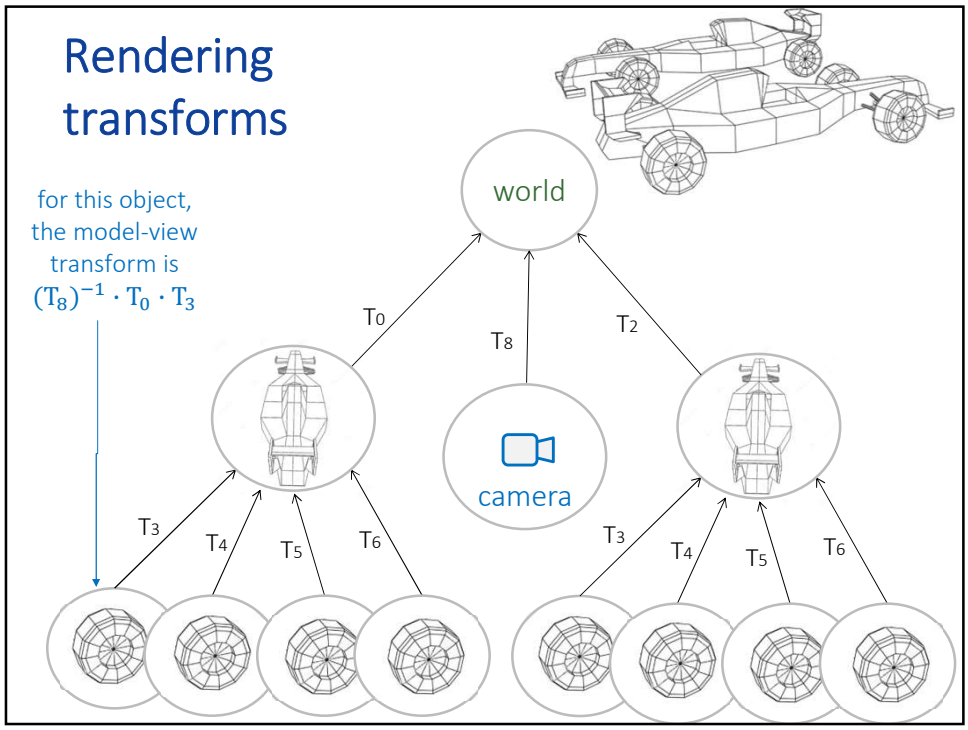
lec. 1: Introduction
lec. 2: Mathematics for 3D Games
lec. 3: Scene Graph
lec. 4: Game 3D Physics +
lec. 5: Game Particle Systems
lec. 6: Game 3D Models
lec. 7: Game Textures
lec. 8: Game 3D Animations
lec. 9: Game 3D Audio
lec. 10: Networking for 3D Games
lec. 11: Artificial Intelligence for 3D Games
lec. 12: Game 3D Rendering Techniques

37

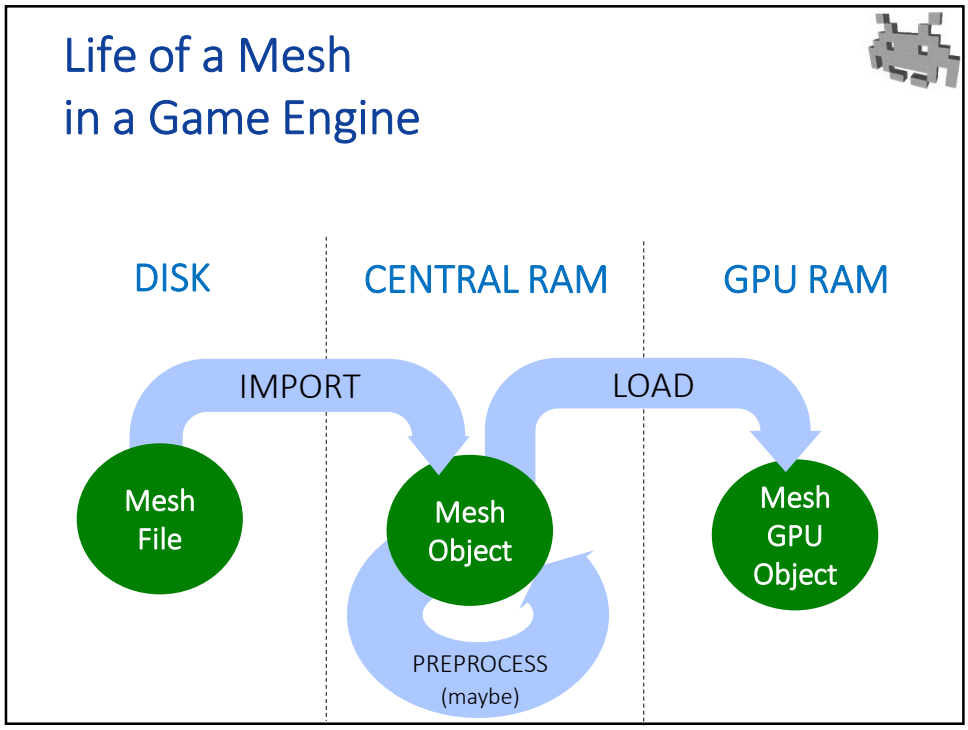
Rendering & Scene graph

- Rendering APIs encode transforms as a 4x4 matrix
  - reason: it is a more flexible, can also express perspective transforms
- To render an object:
  - Combine its Transforms from Object-space to Camera-space ("model-view transform" – in CG terminology)
  - Convert it into a 4x4 matrix
  - Use it during the rendering of the object
  - Note: from world to camera ("view matrix") can be computed and used for all objects
- The model-view matrix is applied to each vertex
  - In the per-vertex processing
  - Combined with the "projection matrix" (from camera space to screen space" is called "model-view-projection" matrix)

38

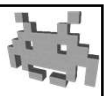

Rendering transforms
for this object, the model-view transform is $(T_8)^{-1} \cdot T_0 \cdot T_3$
world
camera
T_0
T_8
T_2
T_3
T_4
T_5
T_6
T_3
T_4
T_5
T_6

39


Life of a Mesh in a Game Engine
DISK
CENTRAL RAM
GPU RAM
IMPORT
LOAD
Mesh File
Mesh Object
Mesh GPU Object
PREPROCESS (maybe)

42

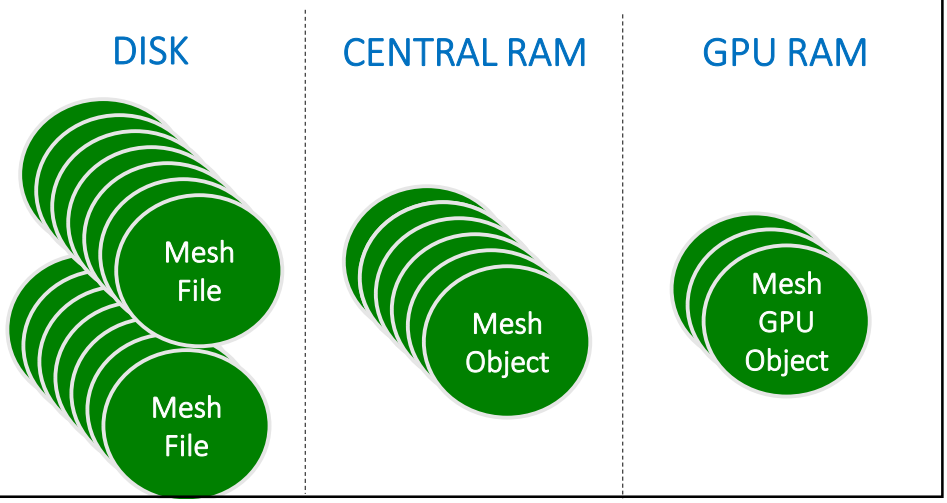
Life of a mesh in a game engine



- Import (from disk)
- Optionally, simple Pre-processing
  - e.g.: Compute Normals (if needed, i.e. rarely)
  - e.g.: Compute Tangent Dirs
  - e.g.: Bake Lighting (sometimes)
- Render (each frame)
  - GPU based
  - Meaning: mesh be loaded in GPU-ram first

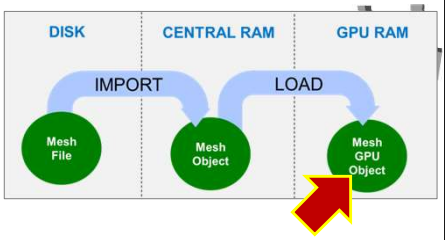
43

Memory Management (during game execution)



44

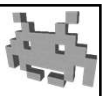
Mesh GPU Object (on Graphic Card)

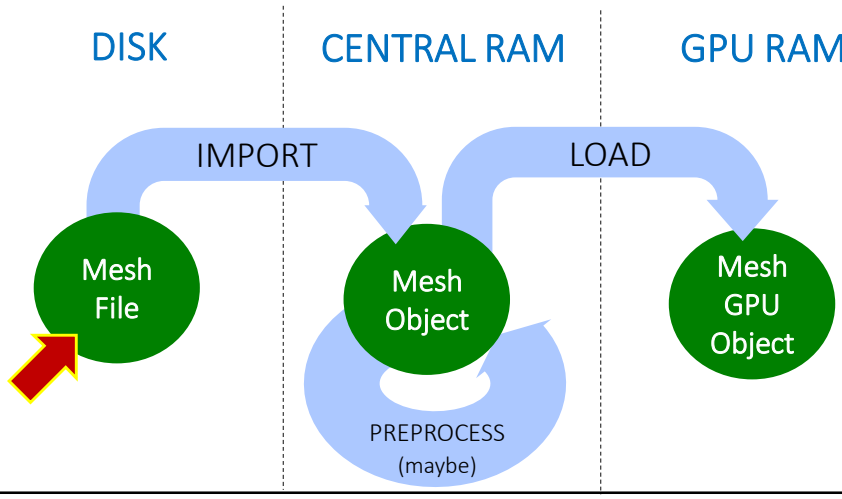


- Buffers storing the mesh
  - GPU APIs call them: Vertex Buffer Object or Vertex Arrays
- They are stored in GPU RAM
  - *The scarcest one !*
- Ready to render!
- Choices for a Game Engine:
  - storage formats, including precisions
  - trade-off between storage cost / accuracy
  - e.g.
    - color? 8 bit per channel
    - position? 16 bit per coordinate

45

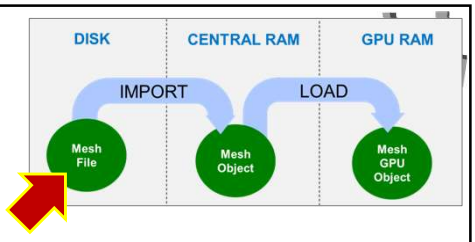
Life of a Mesh in a Game Engine





46

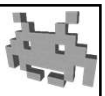
Mesh as an asset

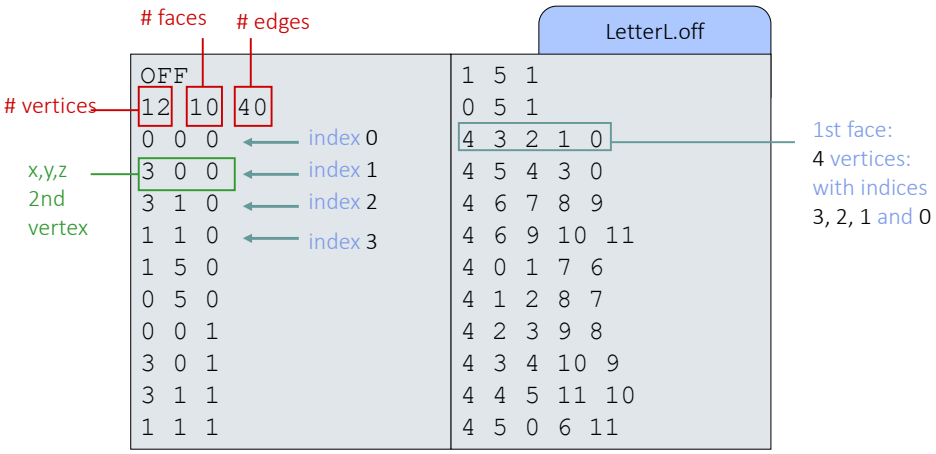


- A file of a given format sitting on the disk
- Choices for the game engine:
  - which formats(s) to import?
    - proprietary, standard…
  - storing which attributes?
- Issues:
  - storage cost
  - loading time

47

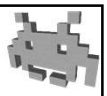
Example of file format for indexed meshes: OFF format





48

File formats for meshes
(a Babel tower!)



- 3DS - 3D Studio Max file format
- OBJ - Another file format for 3D objects
- MA, MB - Maya file formats
- 3DX - Rinoceros file format
- BLEND - Blender file format
- STL - Very used for 3D Printing
- FBX - Autodesk interchange file format
- X - Direct X object
- SMD - good for animations (by Valve)
- MD3 - quake 3 vertex animations
- DEM - Digital Elevation Models
- DXF - exchange format, Autodesk's AutoCAD)
- FIG - Used by REND386/AVRIL
- FLT - MulitGen Inc.'s OpenFlight format
- HDF - Hierarchical Data Format
- IGES - Initial Graphics Exchange Specification
- IV - Open Inventor File Format Info
- LWO, LWB & LWS - Lightwave 3D file formats
- MAZ - Used by Division's dVS/dVISE
- MGF - Materials and Geometry Format
- MSDL - Manchester Scene Description Language
- 3DML - by Flatland inc.
- C4D – Cinema 4D file format
- SLDPTR - SolidWork "part"
- WINGS - Wings3D object
- NFF - Used by Sense8's WorldToolKit
- SKP - Google sketch up
- KMZ - Google Earth model
- OFF - A general 3D mesh Object File Format
- OOGL - Object Oriented Graphics Library
- PLG - Used by REND386/AVRIL
- POV - "persistence of vision" ray-tracer
- QD3D - Apple's QuickDraw 3D Metafile format
- TDDD - for Imagine & Turbo Silver ray-tracers
- NFF & ENFF - (Extended) Neutral File Format
- VIZ - Used by Division's dVS/dVISE
- VRML, VRML97 - Virtual Reality Modeling Language (RIP)
- X3D – attempted successor of VRML
- PLY – introduced by Cyberware – typical of range-scanned data
- DICOM – by DICOM – typical of CAT-scan data
- Renderman – data for the homonymous renderer
- RWX - RenderWare Object
- Z3D - ZModeler File format

etc

50

Most used mesh file formats
(most used in games)

most common

- .OBJ (wavefront)
  - ☺ max diffusion
  - ☺ indexed, normals , uv-mapping
  - ☹ no colors (only material index for face)
  - ☹ no skinning or animations
- .SMD (VALVE)
  - ☺ Skeletal animation + skinning
  - ☺ normals , uv-mapping
  - ☹ no indexed!
  - ☹ no colors
- .MD3 (Quake, IDsoft)
  - ☺ vertex animations, normals
  - ☹ no colors
- .PLY (cyberware)
  - ☺ customizable
  - ☹ "academic"

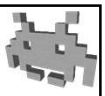
less common

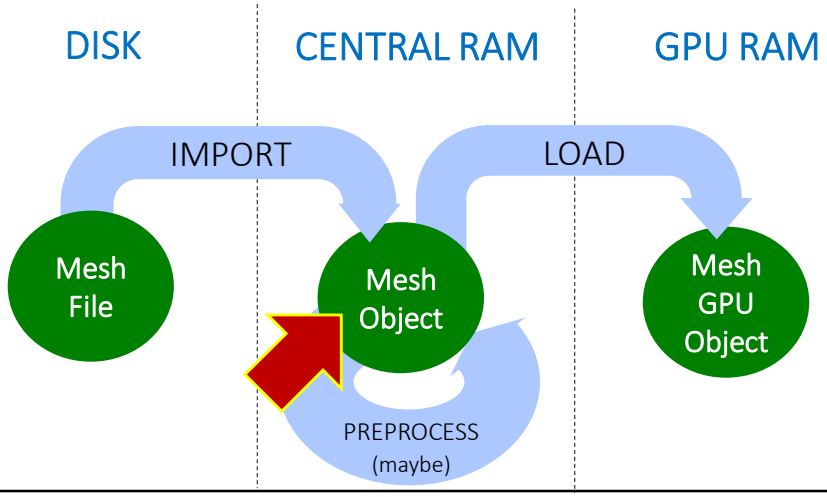
- .3DS (AUTODESK)
  - ☺ YES: colors, uv-mapping, indexed, materials, textures…
  - ☹ NO: normals
  - ☹ limited by vertex number (64K)
- .DAE (collada: SONY + KHRONOS GROUP)
  - ☺ complete
  - ☺ Born for being interchanged
  - ☺ open standard
  - ☹ Almost impossible to parsing it completely
- .FBX (AUTODESK)
  - ☺ complete, with animations
  - ☹ complex, hard to parse
- .MA / .MB (AUTODESK)
  - ☺ complete, with animations
  - ☹ complex, hard to parse

simple — complex

51

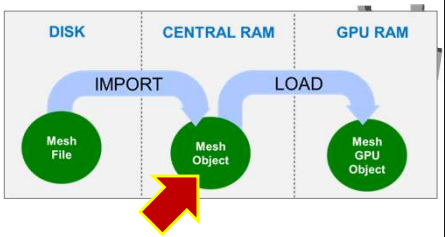
Life of a Mesh in a Game Engine





52

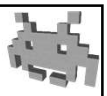
Mesh Object (in RAM)



- A (C++ / Javascript / etc) structure in main RAM
- Choices for a game engine:
  - which attribute to store?
  - storage formats… (floats, bytes, double…)
  - which preprocessing to offer (typically, at load time)

53

How to represent a mesh?
(which data structures)



- Indexed mode in C++ :

```
class Vertex {
  vec3 pos;
  rgb color;    /* attribute 1 */
  vec3 normal; /* attribute 2 */
};

class Face{
   int vertexIndex[3];
};

class Mesh{
  vector<Vertex> verts; /* geom + attr */
  vector<Face> faces;   /* connectivity */
};
```

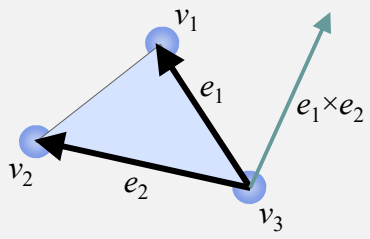
54

Computing normals from geometry

(1) compunte
normals of faces

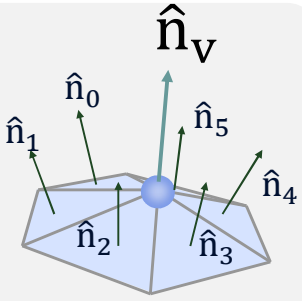
(2) compute
normals of vertices

(1)



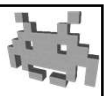
$$\hat{n}_v = \frac{\hat{n}_0 + \cdots + \hat{n}_k}{\|\hat{n}_0 + \cdots + \hat{n}_k\|}$$

(2)



57

Mesh processing:
(or, more in general, Geometry Processing)

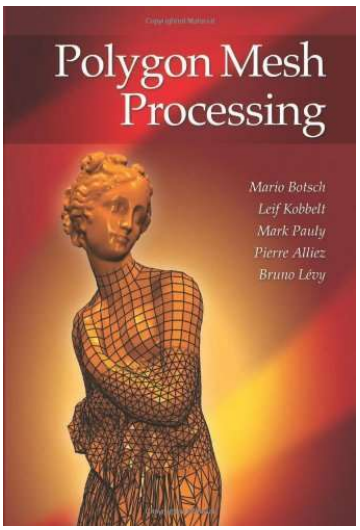


- The algorithm above
  (for the computation of per vertex normal)
  is a tiny example of processing done over a mesh
- Mesh processing: the discipline of creating, transforming, computing meshes
  - inputs and/or outputs are meshes
- Part of, geometry processing:
  - when the input and output are other data structure for 3D models
  - See CG course for a very brief overview

58

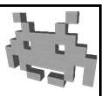
Mesh processing:
(or more in general Geometry Processing)

- A good introduction
  to
  mesh processing



59

Libraries for mesh processing



VCG-Lib

vision and
computer graphic library

CNR ( )

CGAL

computational geometry
algorithms library

INRIA ( )

OpenMesh

\+ OpenFlipper

RWTH ( )

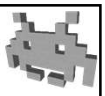
libigl

simple geometry
processing library

NYU ( )

60

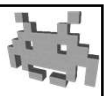
Mesh processing: typical tasks for the game industry



- Poly reduction / Retopology / Simplification
  - e.g. LOD construction
  - e.g. transition from (initial) hi-res to (final) low-poly
- Light baking LATER
  - Light precomputation
  - e.g.: Ambient Occlusion
- U-V map construction LATER
  - parametrization / unwrapping
- Texturing LATER
  - creation of different types of textures
- Rigging / Skinning / Animation LATER
  - to animate

61

Useful general tools: attribute transfer

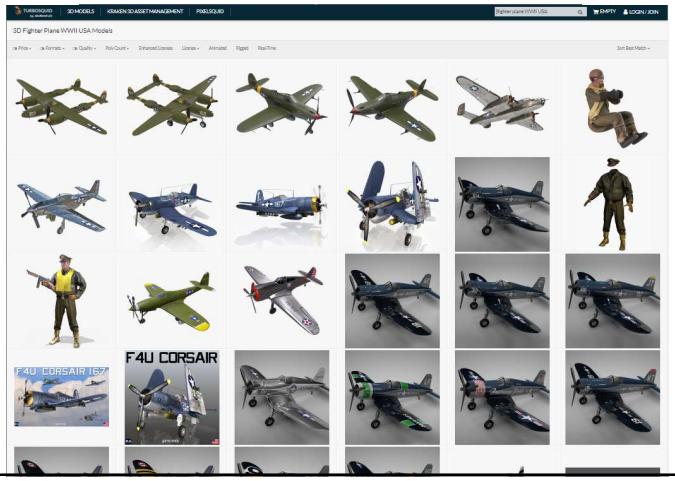


- Given
  - a source mesh M_0 with attribute A (any, see the list!)
  - a target mesh M_1 similar (but not identical) to M_0 lacking that attribute
- Define attribute A in the vertices of M_1
  - Copying the attributes from M_0
- Result: "retargeting" of…
  - Animations, UV-mapping, textures, etc
- Results aren't always perfect, but can be useful as a starting point

62

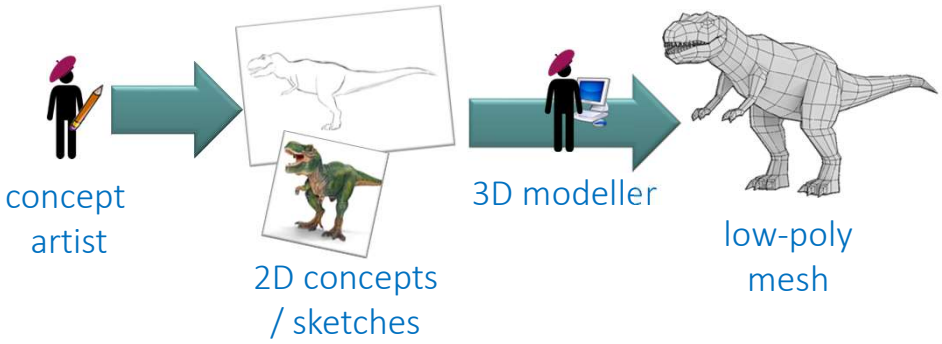
3D models: suorces

- Like any asset, often just bought / off-sourced



65

3D models: authoring pipeline



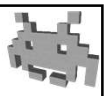
66

Mesh: authoring

- Task of the 3D modeller
  - A type of digital artist
- Popular 3D modeling approaches:
  - Manual low-poly modelling
    - e.g. with wings3D
  - Subdivision surfaces
    - e.g. with blender
  - Digital sculpting
    - e.g. with Z-brush

67

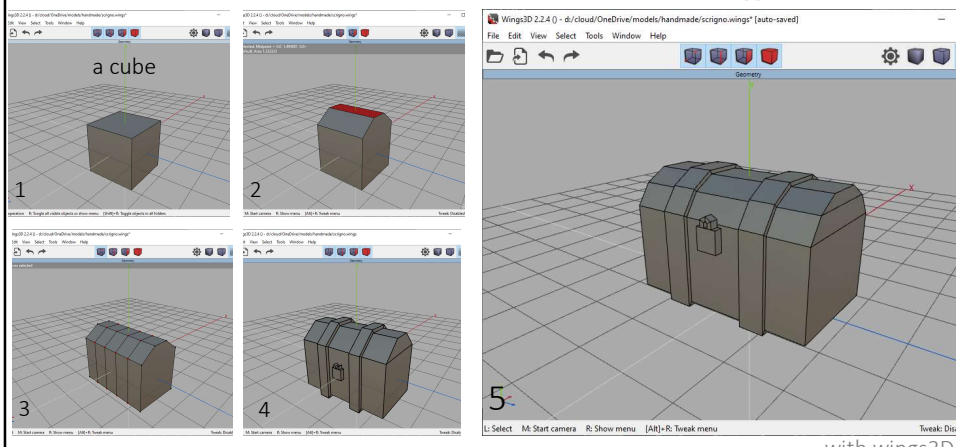
Mesh authoring (aka 3D modelling): a few applications



- 3D Studio Max (autodesk) , Maya (autodesk) , Cinema4D (maxon) Lightweight 3D (NewTek), Modo (The Foundry) , …
  - all-purpose, powerful, complete
- Blender
  - the same, plus open-source and freeware (compare: Gimp VS. Adobe Photoshop for 2D images)
- MeshLab
  - open-source, big collection of geometry processing algorithms …
- AutoCAD (autodesk), SolidWorks (SolidThinking)
  - for CAD
- ZBrush (pixologic) (+ Sculptris), Mudbox (autodesk)
  - Sculpting (inclusing texturing)
- Wings3D
  - low-poly modelling (& subdivision surfaces) open-source, small, specialized
- [Rhinoceros]
  - parametric surfaces (NURBS)
- FragMotion
  - small, specialized on animated meshes
- + a many more for specific contexts
  - editing of human models, of architectural interiors, environments, or specific editors for game-engines, etc…

68

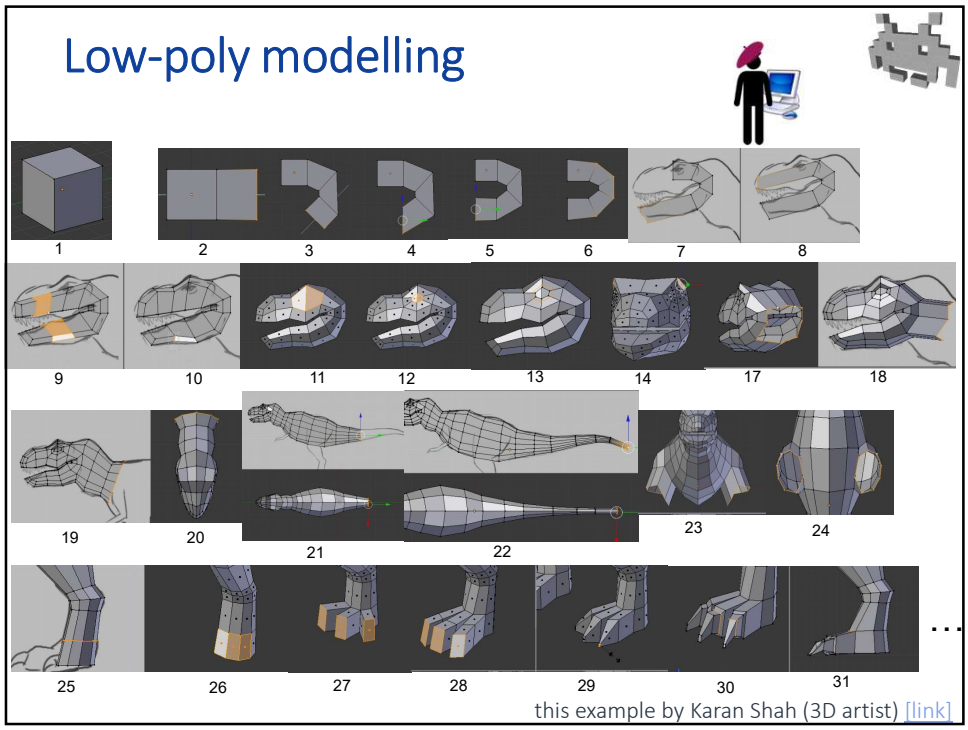
Low-poly modelling (demo)



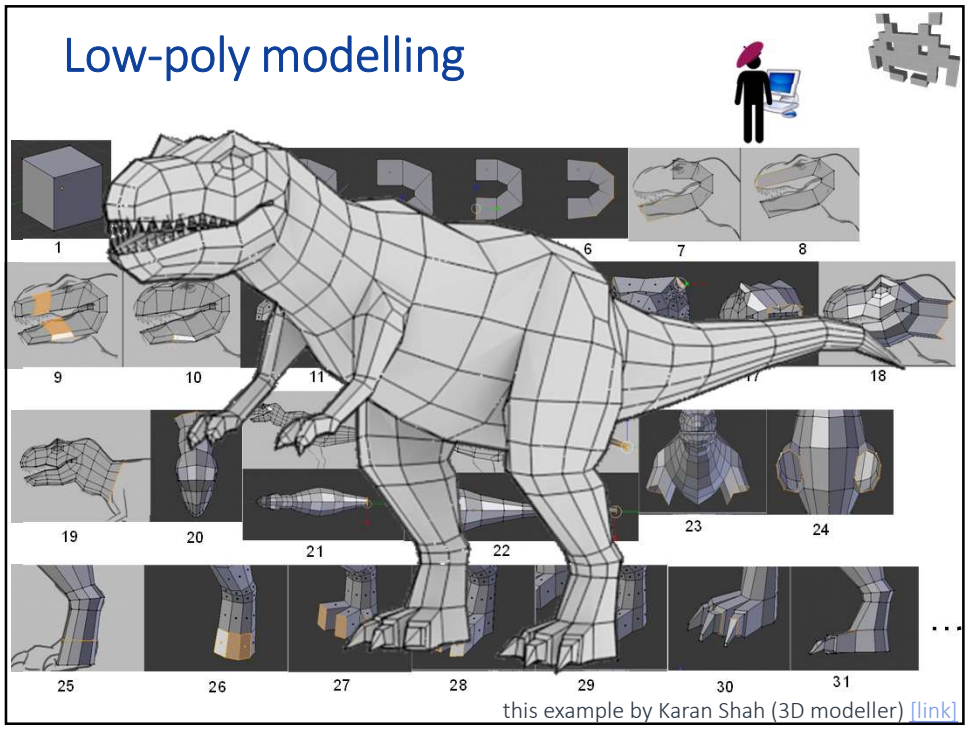
with wings3D

Note: during creation, the meshes can be polygonal instead of triangle based, but is simple to decompose any polygon into triangles
E.g. this can be done by the game engine as a simple preprocessing.

69

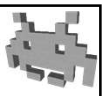

Low-poly modelling
1
2
3
4
5
6
7
8
9
10
11
12
13
14
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
this example by Karan Shah (3D artist) [link]

71

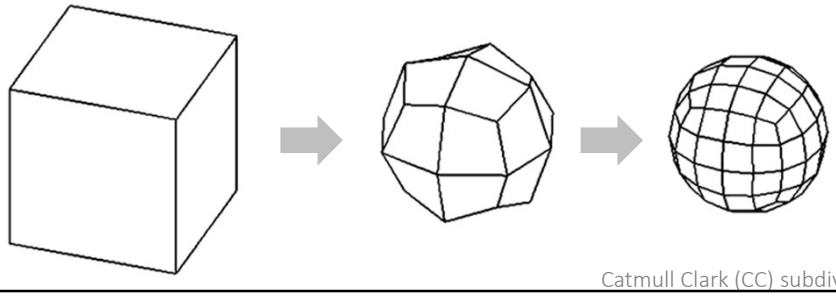

Low-poly modelling
1
6
7
8
9
10
11
18
19
20
21
22
23
24
25
26
27
28
29
30
31
this example by Karan Shah (3D modeller) [link]

72

3D mesh authoring techniques: subdivision surfaces



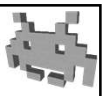
- Subdivision step:
  an algorithm that operates on a mesh
  and obtains a higher resolution, smoother mesh
- Can be iterated

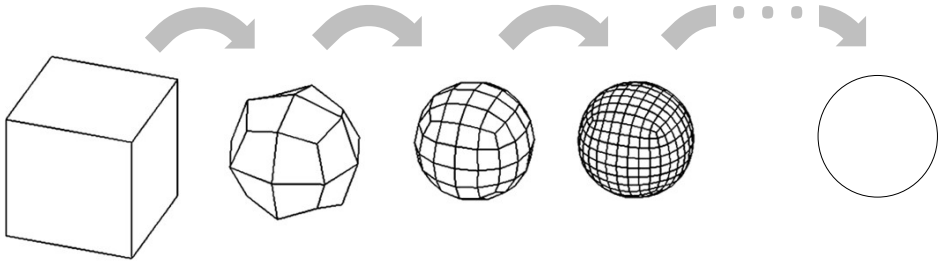


Catmull Clark (CC) subdivision

73

Example: with Catmull-Clark scheme





level 0
"control mesh"

level 1

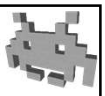
level 2

level 4

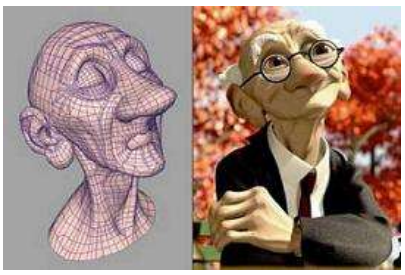
lvl ∞
"limit surface"

74

3D mesh authoring techniques: subdivision surfaces



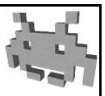
- Many subdivision algorithms (schemas) exists
  - each with its own properties
- Produces clean, regular meshes
- Excellent for smooth, curved, organic looking objects



famously pioneered by movie industry (not games):

75

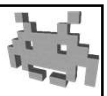
Subdivision surfaces as a tool…



- …to encode smooth surfaces
  - Idea: we encode the control mesh to represent the limit surface
  - use in games: rendering (now, rare – but popular around 2015)
    1. keep control mesh in GPU ram
    2. let 1-3 subdivision steps happen during rendering
- …to author 3D meshes
  - idea: alternate (low-poly) editing and subdivisions steps
  - at first steps: edit global shape
  - at last steps: edit minute details
  - use in games: during asset creation, by artists

76

Subdivision surfaced as way to define (curved) surfaced



- Modeler creates a low-poly mesh, the "control mesh"
  - control mesh: piecewise linear (i.e., flat) surface
- The control mesh is subdivided (in theory ∞ times) and a "limit surface" is obtained
  - limit surface: curved & smooth surface
- The control mesh is a representation of the limit surface
  - note: the subdivision steps are only performed on the fly, during rendering
  - the more step are done, the better the limit surface is approximated

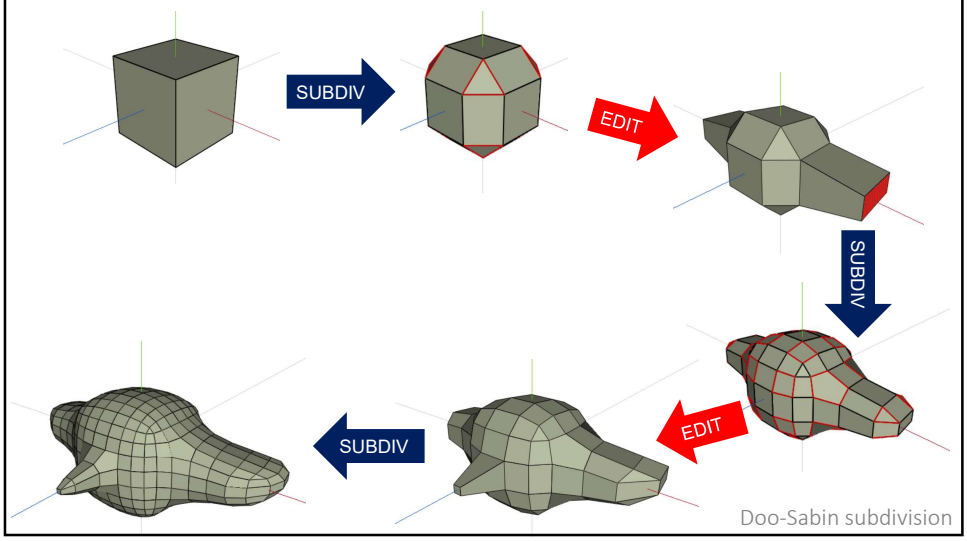
77

Subdivision surfaces as a mesh authoring tool

1. Create a coarse mesh with a very approx. shape
   - e.g., using low-poly modelling
2. Apply subdivision step
   - a higher resolution model
3. Re-edit results
   - Retouch all the smaller parts
4. Goto 2, until good final result

78

Subdivision surfaces as a mesh authoring tool (example)



80

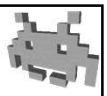
Example of subdivision schema: "Butterfly" (used for tri-meshes)

- It's a "1-to-4 schema"
  in a subdivision step, each triangle is split into 4
  by adding one vertex in each original edge

For more info, see **Computer Graphics** course

84

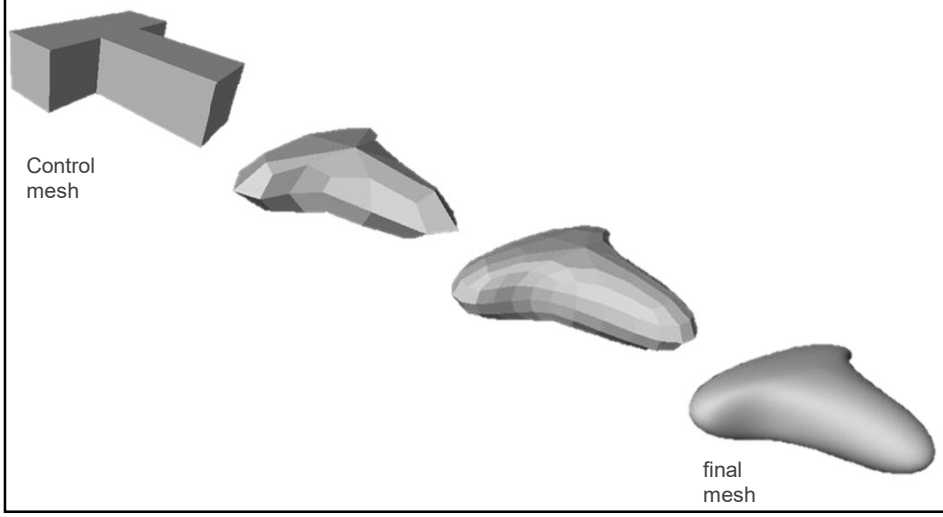
Subdivision surfaces in general



- A step typically increases resolution by a factor **x4**
- The geometry of the subidvided mesh (3D points) is computed according to a formula of the pos of their neighbors.
  - In some schemas (called interpolative), the old vertices are kept at the same positions
  - In other schemas (called approximative), old vertices are kept but moved into a new position
  - In other schemas (called dual) older vertices aren't kept
- Most created vertices are *regular*

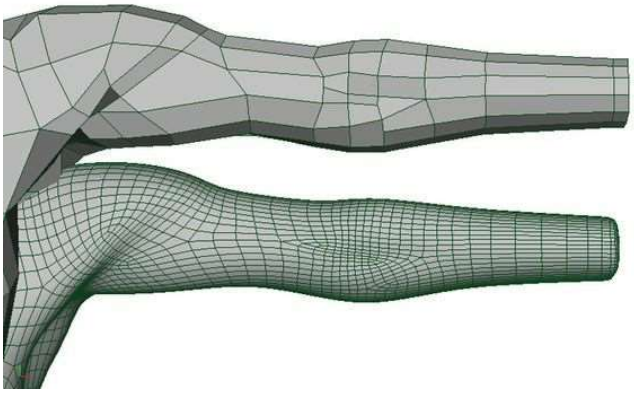
85

An example with Catmull Clark



86

An example with Catmull Clark



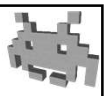
87

Some common subdivision schemas

- Doo-Sabin
  - operates on any polygonal mesh
  - produces polygonal meshes
- Loop
  - 1-to-4 scheme for triangle meshes (only)
- Butterfly
  - 1-to-4 scheme for triangle meshes (only)
- Catmull-Clark
  - operates on any polygonal mesh
  - produces quad-meshes
  - traditionally, movie-industry favorite
  - a recent trend in games: use during mesh rendering

91

3D Mesh authoring: approaches



- Popular 3D modeling approaches:
  - Direct low-poly modelling
    - e.g. with wings3D
  - Subdivision surfaces
    - e.g. with blender
  - Digital sculpting
    - e.g. with Z-brush,
      (or Sculptris Alpha)

93

Digital Sculpting

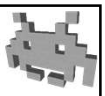
mouse
(or stylus)

=

chisel

94

Digital Sculpting

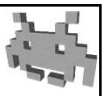


- demo

with wings3D

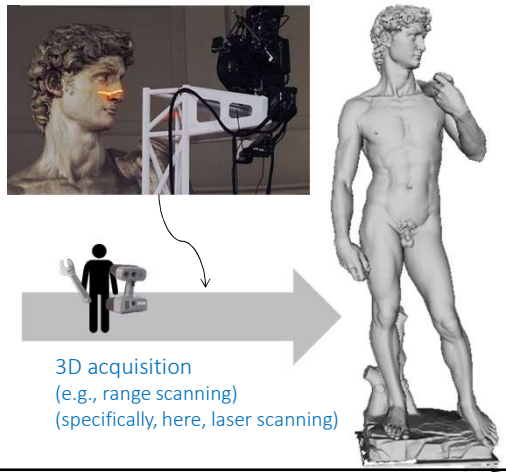
95

Sources for 3D models: 3D acquisition



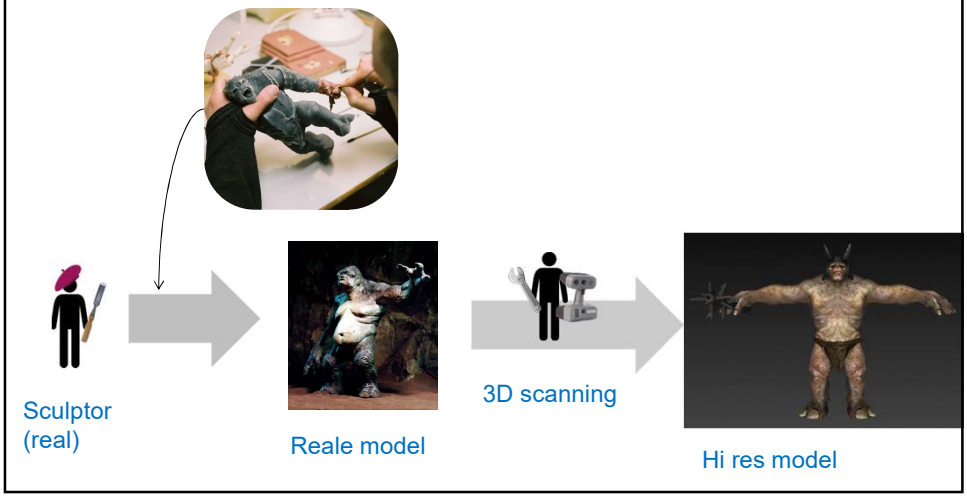
For more info, see **Computer Graphics** course

- 3D acquisition / 3D scanning
  - Technologies for obtaining 3D digital models from real-world objects

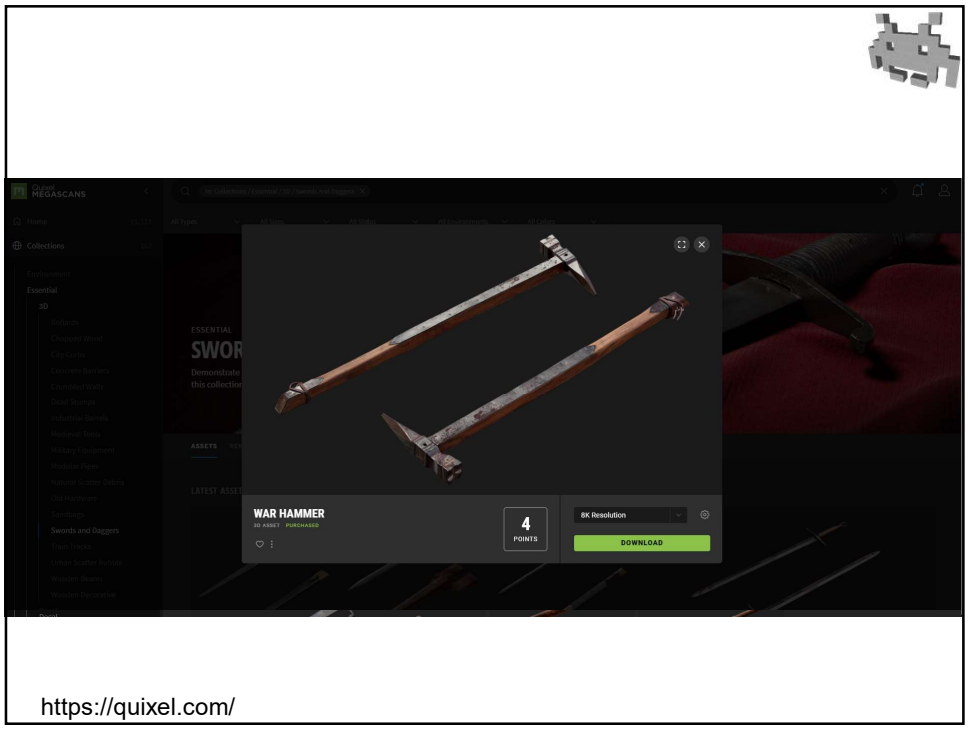


96

Sources for 3D models: 3D acquisition



97

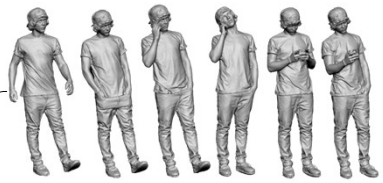


https://quixel.com/

98

Sources for 3D models: 3D acquisition

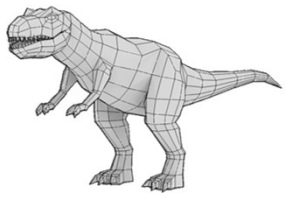
- 3D scanning
  - A.k.a. *automatic 3D model acquisition*
  - Lot of different technologies
    - Laser scanners
    - Time of flight
    - Structured light (kinect)
    - …
  - Different characteristics
    - Results quality
      - Noise / resolution
    - Automatism
    - Invasiveness
      - Markers? Powder?
    - Real time? (kinect)
    - Price
    - Max object dimension
      - (full body scanner?)



99

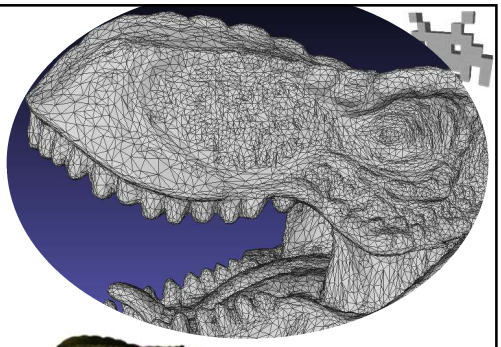
3D models sources: comparison

PERFECT for games!
(much easier to: animate, re-edit, uvmap, …)



manually edited
low-poly mesh
(2K triangles)

VS



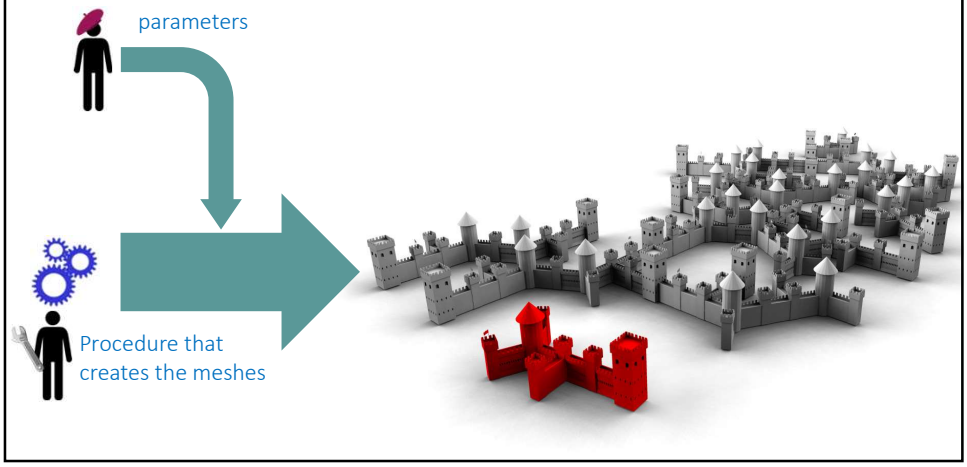
Dino, scanned by artec3d

scanned & cleaned
hi res mes
(30K triangles)

(sculpted meshes are similar)

100

Sources for 3D models: procedural modelling



101

Procedural modelling – see also…

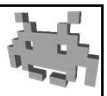
EPC2022
EVERYTHING PROCEDURAL
CONFERENCE ON PROCEDURAL CONTENT GENERATION FOR GAMES
22-04-2022
MASTERCLASS 21-04-2022

http://everythingprocedural.com/

this week
Game-of-the-Week

102

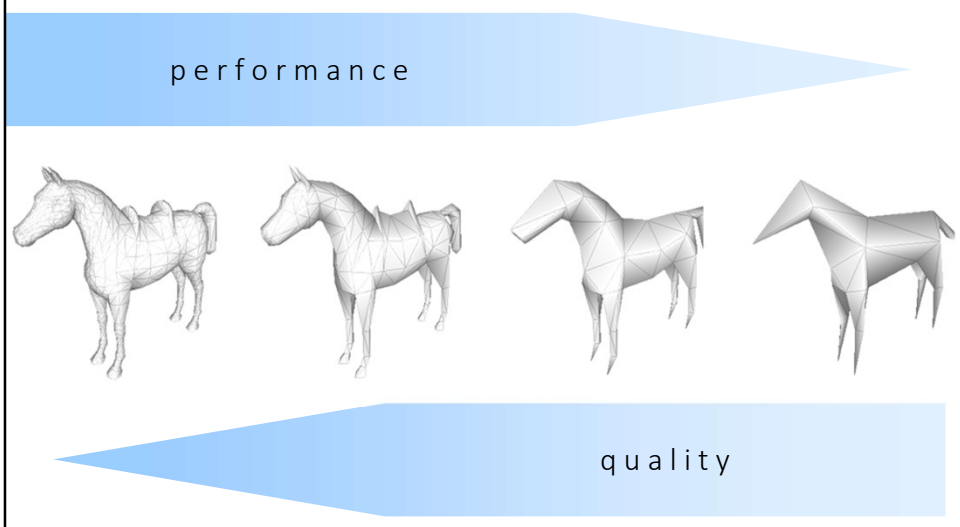
Notes about mesh resolution



- all costs: linear on the triangles number
  - in memory (disk, CPU RAM, GPU RAM)
  - in time (rendering, loading, etc)
- (and, linear with # of vert. with # triangles)
  - (*rule of thumb*: K verts → 2K tris)
- reminder: possible adaptive resolution
  - higher-res in some parts
  - lower-res in others

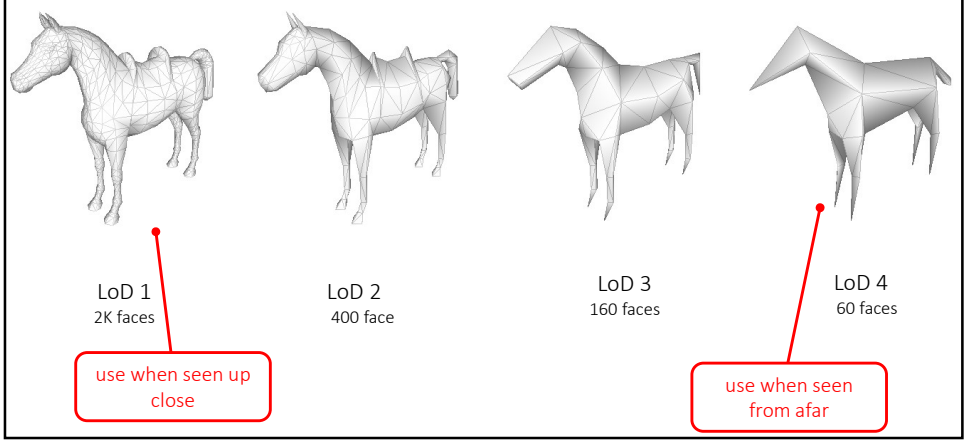
103

Rendering quality and resolution



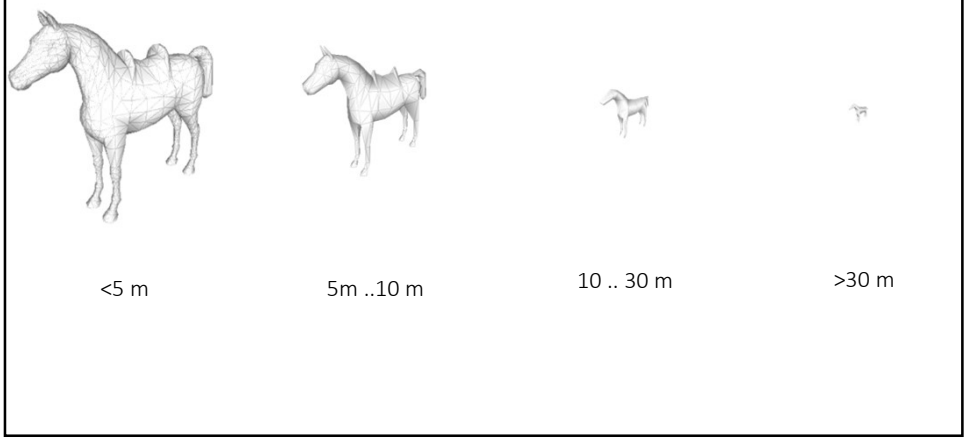
104

LoD pyramids
(Level Of Detail)



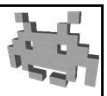
105

LoD pyramids
(Level Of Detail)



106

LoD pyramids (Level Of Detail)



- Goal:
  - decrease the geometry budget (total number of vertices)
  - ideal: size of triangles in screen space (in pixel): constant
    - (if importance / complexity is the same)
- Task: determining the level to use (dynamically, at runtime)
  - depending on observer distance
  - and/or, depending on rendering workload
    - e.g.: rendering is lagging ⇒ decrease LoD
  - this is task of the rendering engine)

computed from scene graph (how?)

- Task: LOD creation or "LOD-ding" (during asset creation)
  - starting from LOD-0 (higher-res)
  - manual, or automatic *(see later on),* or assisted (mixed)
    - often manual
  - note: sometimes "LoD 0" is used only in special cases
    - e.g. during a cut-scenes

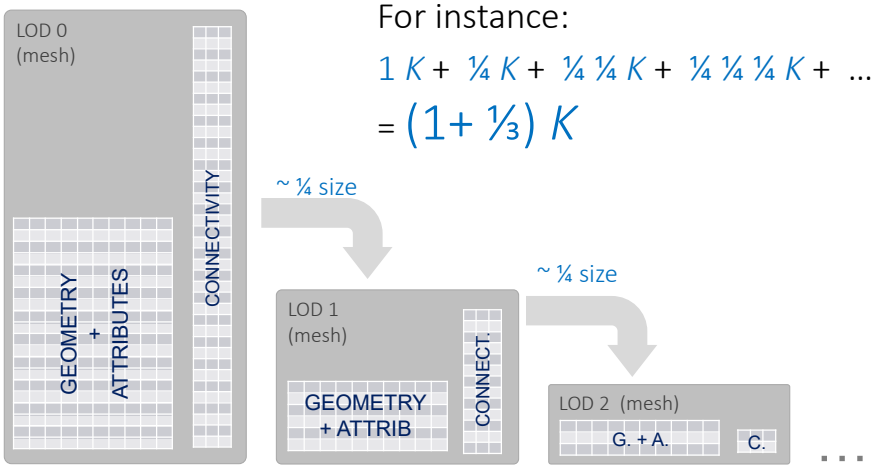
107

LoD pyramids (Level Of Detail)

Total memory usage: limited
For instance:

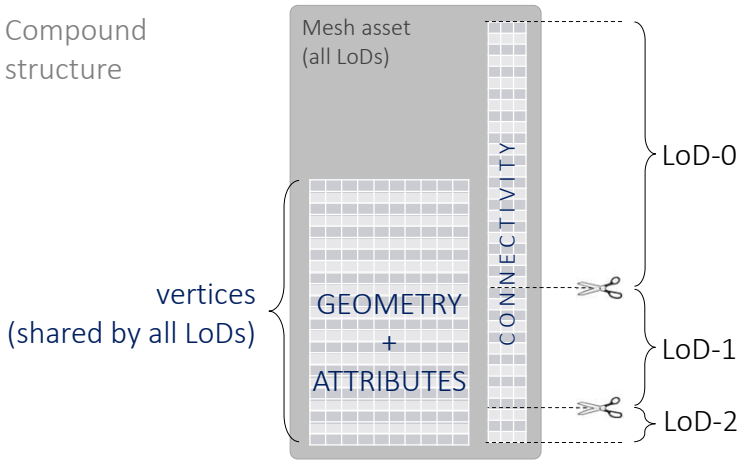
$1\ K + \frac{1}{4}\ K + \frac{1}{4}\frac{1}{4}\ K + \frac{1}{4}\frac{1}{4}\frac{1}{4}\ K + \ldots$

$= (1 + \frac{1}{3})\ K$



108

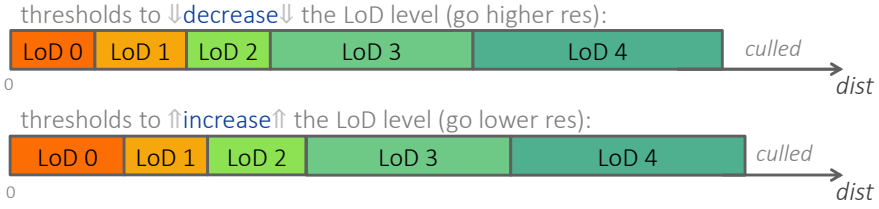
LoD pyramids (Level Of Detail)



109

LoD pyramids: which level to use

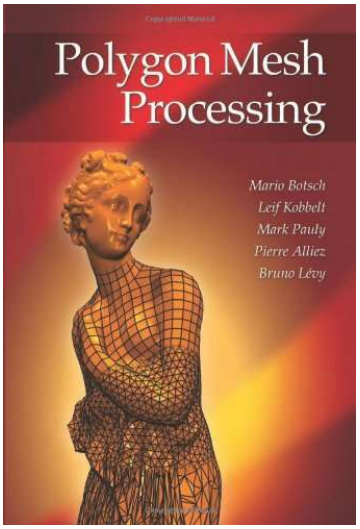
- Basic strategy: use a fixed LoD for each interval of distance (from camera)
- ⚠ popping artefacts!
  - to mitigate it: used different thresholds to increase and to decrease the LoD



110

Mesh processing:
(or more in general Geometry Processing)

- A good introduction to mesh processing



111

Libraries for mesh processing

VCG-Lib

vision and computer graphic library

CNR ( )

CGAL

computational geometry algorithms library

INRIA ( )

OpenMesh + OpenFlipper

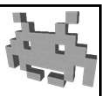
RWTH ( )

libigl

simple geometry processing library

NYU ( )

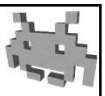
112



Mesh processing: typical tasks for the game industry

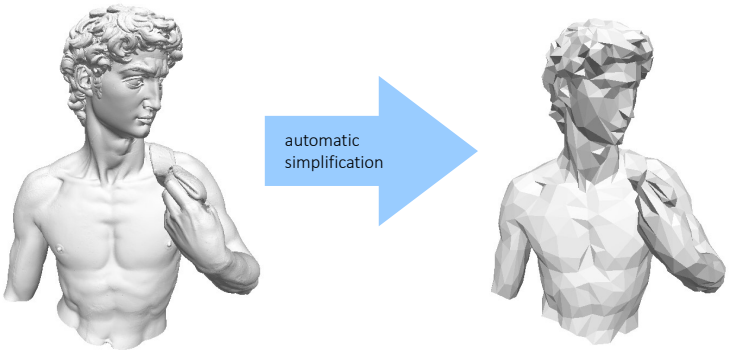
- Poly reduction / Simplification
  - e.g. LOD construction
  - e.g. transition from (initial) hi-res to (final) low-poly
  - Variant: “Retopology” : change the meshing, keep the shape
- Light baking SEE LATER
  - Light precomputation
  - e.g.: Ambient Occlusion
- U-V map construction SEE LATER
  - parametrization / unwrapping
- Texturing SEE LATER
  - creation of different types of textures
- Rigging / Skinning / Animation SEE LATER
  - to animate

113



Poly-reduction (aka Mesh “simplification” / “decimation”)

- parameters:
  - a maximum error
  - or number of faces objective

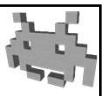


Original mesh
500K triangles

Simplified mesh
2K triangles

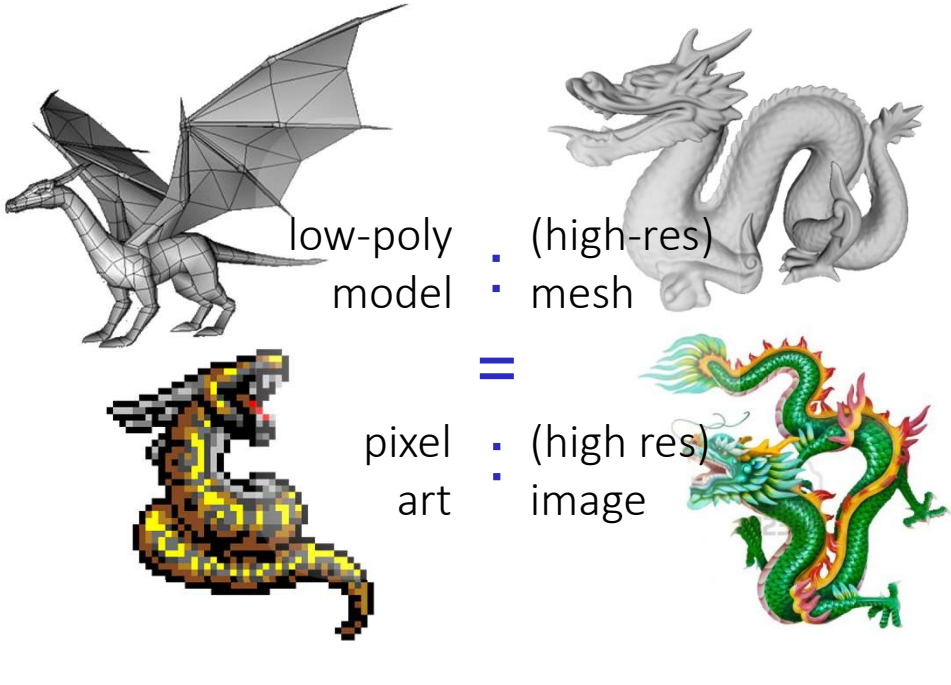
114

Poly-reduction
(aka mesh simplification, mesh coarsening)



- Different approaches are studied in Geometry Processing.
  - Adaptive or not
    - use more triangles where needed (ex. not in flat parts)
    - or not
  - Maximum error introduced:
    - measured and/or limited
    - or not
  - Topology:
    - kept
    - or not
  - Streamable
    - Possible
    - or not

115



116