Computer Graphics 1

Ludwig-Maximilians-Universität München
Summer semester 2020

Prof. Dr.-Ing. Andreas Butz
lecture additions by Dr. Michael Krone, Univ. Stuttgart

http://www.wikwand.com/
Chapter 3 - 3D Modeling

- Polygon Meshes
- Geometric Primitives
- Constructive Solid Geometry (CSG)
- Extrusion & Rotation
- Interpolation Curves
- Levels Of Detail (LOD)
- Volume- and Point-based Graphics
The 3D rendering pipeline (our version for this class)

3D models in model coordinates → 3D models in world coordinates → 2D Polygons in camera coordinates → Pixels in image coordinates

- Scene graph
- Camera
- Animation, Interaction
- Rasterization
- Lights
Representations of (Solid) 3D Objects

• Complex 3D objects need to be constructed from a set of primitives
  • Representation schema is a mapping of 3D objects \( \rightarrow \) primitives
  • Primitives should be efficiently supported by graphics hardware

• Desirable properties of representation schemata:
  • Representative power: Can represent many (or all) possible 3D objects
  • Representation is a mapping: Unique representation for any 3D object
  • Representation mapping is injective: Represented 3D object is unique
  • Representation mapping is surjective: Each possible representation value is valid
  • Representation is precise, does not make use of approximations
  • Representation is compact in terms of storage space
  • Representation enables simple algorithms for manipulation and rendering

• Most popular on modern graphics hardware:
  • Boundary representations (B-Reps) using vertices, edges and faces.
Polygon Meshes

- Describe the surface of an object as a set of polygons
- Mostly use triangles, since they are trivially convex and flat
- Current graphics hardware is optimized for triangle meshes
3D Polygons and Planes

• A polygon in 3D space should be flat, i.e. all vertices in one 2D plane
  • Trivially fulfilled for triangles

• Mathematical descriptions of a 2D plane in 3D space (hyperplane)
  • Method 1: Point $p$ and two non-parallel vectors $v$ and $w$
    \[ x = p + s \vec{v} + t \vec{w} \]
  • Method 2: Three non-collinear points
    (take one point and the difference vectors to the other two)
  • Method 3: Point $P$ and normal vector $n$ for the plane
    \[ \vec{n} \cdot (x - p) = 0 \]
  • Method 4: Single plane equation
    \[ ax_1 + bx_2 + cx_3 + d = 0 \quad a, b, c, d \in \mathbb{R} \]
    \[(a, b, c)\) is the normal vector of the plane

• All description methods easily convertible from one to the other
  (e.g. using cross product to compute normal vector)
Right Hand Rule for Polygons

• A “rule of thumb” to determine the front side (= direction of the normal vector) for a polygon
• Please note: The relationship between vertex order and normal vector is just a convention!
  • Can be defined in OpenGL (clockwise/counter-clockwise)
  • Q: How can we see this from the previous slides?
Face-Vertex Meshes

→ Left-handed (clockwise)

Möbius Strip: Non-Orientable Surface

Complete object: Does not have a front and back side!

M. C. Escher: Moebius Strip II
Polygon Meshes: Optional Data

- Color per vertex or per face: produces colored models
- Normal per face:
  - Easy access to front/back information (for visibility tests)
- Normal per vertex:
  - Standard computation accelerated (average of face normals)
  - Allows free control over the normals
    - use weighted averages of normals
    - mix smooth and sharp edges (VRML/X3D: crease angles)
  - Wait for shading chapter...
- Texture coordinates per vertex
  - Wait for texture chapter...

Polygon Meshes: Other Descriptions

- Other representations for polygon meshes exist
  - Optimized for analyzing and modifying topology
  - Optimized for accessing large models
  - Optimized for fast rendering algorithms
  - Optimized for graphics hardware

- Example: triangle strip
  - Needs N+2 points for N polygons
  - Implicit definition of the triangles
  - Optimized on graphics hardware

Approximating Primitives by Polygon Meshes

• Trivial for non-curved primitives...
• The curved surface of a cylinder, sphere etc. must be represented by polygons somehow (Tessellation).
• Not trivial, only an approximation and certainly not unique!
  • GLU (Graphics Library Utilities) utility functions for tessellation exist
• Goal: small polygons for strong curvature, larger ones for areas of weak curvature
  • This means ideally constant polygon size for a sphere
  • Q: Where do we know this problem from? Something playful...

http://www.evilbastard.org/slight/tesselation.gif
Chapter 3 - 3D Modeling

- Polygon Meshes
- Geometric Primitives
- Constructive Solid Geometry (CSG)
- Extrusion & Rotation
- Interpolation Curves
- Levels Of Detail (LOD)
- Volume- and Point-based Graphics
Geometric Primitives

- Simplest way to describe geometric objects
- Can be used directly by some renderers (e.g., ray tracing)
- Can be transformed into polygons easily (tessellation)
- Can be transformed into volumetric description (solid objects) easily
- Useful for creating simple block world models

- Supported in many frameworks of different levels
  - VRML/X3D, Java 3D, Three.js
  - OpenGL, WebGL, JOGL
Box

- Described by \((\text{width}, \text{length}, \text{height})\)
- Origin usually in the center
- 8 points, 12 edges, 6 rectangles, 12 triangles
Pyramid, Tetrahedron (Tetraeder)

• Basis of pyramid = rectangle
  • given by \((width, length, height)\)
  • 5 points, 8 edges, 6 triangles

• Basis of tetrahedron = triangle
  • given by \((width, length, height)\)
  • 4 points, 6 edges, 4 triangles,
Generalization: Polyhedra

• Polyhedron (Polyeder):
  • Graphical object where a set of surface polygons separates the interior from the exterior
  • Most frequently used and best supported by hardware: surface triangles
  • Representation: Table of
    • Vertex coordinates
    • Additional information, like surface normal vector for polygons

• Regular polyhedra: Five Platonic regular polyhedra exist
  • Tetrahedron (Tetraeder)
  • Hexahedron, Cube (Hexaeder, Würfel)
  • Oktahedron (Oktaeder)
  • Dodekahedron (Dodekaeder)
  • Icosahedron (Ikosaeder)

http://www.aleakybos.ch/
Cylinder, cone, truncated cone

- Cylinder given by (radius, height)
- Number of polygons depends on tessellation

- Cone given by (radius, height)
- Number of polygons depends on tessellation

- Truncated cone given by (r1, r2, height)
- Number of polygons depends on tessellation

Q: Which of these would you rather have if you only had one available?
Sphere, Torus

• Sphere is described by \((\text{radius})\)
• Torus is defined by \((\text{radius}_1, \text{radius}_2)\)
• Number of polygons dep. on tessellation

Geometric Primitives: Summary

• Not all of these exist in all graphics packages
• Some packages define additional primitives (dodecahedron, teapot...)

• Practically the only way to model in a text editor
• Can give quite accurate models
• Extremely lean! Very little data!

• Think of application areas even in times of powerful PC graphics cards!
  •
  •
  •
Chapter 3 - 3D Modeling

- Polygon Meshes
- Geometric Primitives
- Constructive Solid Geometry (CSG)
- Extrusion & Rotation
- Interpolation Curves
- Levels Of Detail (LOD)
- Volume- and Point-based Graphics
Constructive Solid Geometry

• Basic idea: allow geometric primitives and all sorts of boolean operations for combining them
  • Can build surprisingly complex objects
  • Good for objects with holes (often the simplest way)

• Basic operations:
  • OR: combine the volume of 2 objects
  • AND: intersect the volume of 2 objects
  • NOT: all but the volume of an object
  • XOR: all space where 1 object is, but not both

• Think about:
  • Wheels of this car
  • Tea mug
  • Coke bottle (Problems??)

• CSG not supported by OpenGL!
CSG: A Complex Example

- rounded_cube = cube And sphere

CSG: A Complex Example

- \texttt{rounded\_cube} = \texttt{cube And sphere}
- \texttt{cross} = \texttt{cyl1 Or cyl2 Or cyl3}

CSG: A Complex Example

- rounded_cube = cube And sphere
- cross = cyl1 Or cyl2 Or cyl3
- result = rounded_cube And (Not cross)

CSG: A Complex Example

- \texttt{rounded\_cube} = \texttt{cube And sphere}
- \texttt{cross} = \texttt{cyl1 Or cyl2 Or cyl3}
- \texttt{result} = \texttt{rounded\_cube And (Not cross)}

**Q:** Are CSG operations associative?
- ...

**...commutative?**
- ...
Chapter 3 - 3D Modeling

- Polygon Meshes
- Geometric Primitives
- Constructive Solid Geometry (CSG)
- Extrusion & Rotation
- Interpolation Curves
- Levels Of Detail (LOD)
- Volume- and Point-based Graphics
Extrusion (sweep object)

- Move a 2D shape along an arbitrary path
- Possibly also scale in each step

http://www.cadimage.net/cadtutor/lisp/helix-02.gif
Rotation

- Rotate a 2D shape around an arbitrary axis
- Can be expressed by extrusion along a

- How can we model a vase?
- •
- •
- •

- How a Coke bottle?
- •
- •
- •
Chapter 3 - 3D Modeling

• Polygon Meshes
• Geometric Primitives
• Constructive Solid Geometry (CSG)
• Extrusion & Rotation
• Interpolation Curves
• Levels Of Detail (LOD)
• Volume- and Point-based Graphics
Interpolation Curves, Splines

• Original idea: „Spline“ used in ship construction shapes:
  • Elastic wooden band
  • Fixed in certain positions and directions
  • Mathematically simulated by interpolation curve
  • Piecewise described by polynomials

• Different types exist
  • Natural splines
  • Bézier curves
  • B-Splines

• Control points may be on the line or outside of it.
  • All on the line for a natural spline
Bézier Curves (and de-Casteljau Algorithm)

- Bézier curves first used in automobile construction (1960s, Pierre Bézier - Renault, Paul de Casteljau - Citroën)
- Degree 1: straight line interpolated between 2 points
- Degree 2: quadratic polynomial
- Degree 3: cubic Bézier curve, described by cubic polynomial
- Curve is always contained in convex hull of points
- Algorithm (defines line recursively):
  - Choose t between 0 and 1
  - I1: Divide line between P1 and P2 as t : (1-t)
  - I2, I3: Repeat for all Ps (one segment less!)
  - J1, J2: Repeat for I1, I2, I3 (same t)
  - K: Repeat for J1, J2 (single point!)
  - Bézier curve: all points K for t between 0 and 1
- see [http://goo.gl/m7Z1Y](http://goo.gl/m7Z1Y) (Dominik Menke)
Bézier Patches

• Combine 4 Bézier curves along 2 axes
• Share 16 control points
• Results in a smooth surface
• Entire surface is always contained within the convex hull of all control points
• Border line is fully determined by border control points
• Several patches can be combined
  • connect perfectly if border control points are the same.
• Advantage: move just one control point to deform a larger surface…
• Other interpolation surfaces based on other curves
  • Generalization of Bézier idea: B-splines
  • Further generalization: Non-uniform B-splines
  • Non-uniform rational B-splines (NURBS) (supported by OpenGL GLU)
Interpolation in OpenGL (Bézier Example)

- Utah teapot
  - Martin Newell, 1975
  - 306 vertices
  - 32 bicubic Bézier surface patches

→ Only outer surface, no interior walls!

http://www.realtimerendering.com/teapot/
Chapter 3 - 3D Modeling

- Polygon Meshes
- Geometric Primitives
- Constructive Solid Geometry (CSG)
- Extrusion & Rotation
- Interpolation Curves
- Levels Of Detail (LOD)
- Volume- and Point-based Graphics
Levels of Detail

• Assume you have a very detailed model
  • from close distance, you need all polygons
  • from a far distance, it only fills a few pixels

• How can we avoid drawing all polygons?
  •
  •
  •
  •
Mesh Reduction

- Original: ~5,000 polygons
- Reduced model: ~1,000 polygons
  about 80% reduction

- Very strong reductions possible
  - depending on initial mesh

- Loss of shape if overdone

http://www.okino.com/conv/polygon_reduction/geoman2/
polygon_reduction_tutorial1.htm
A Method for Polygon Reduction

- Rossignac and Borell, 1992, „Vertex clustering“
- Subdivide space into a regular 3D grid
- For each grid cell, melt all vertices into one
  - Choose center of gravity of all vertices as new one
  - Triangles within one cell disappear
  - Triangles across 2 cells become edges (i.e. disappear)
  - Triangles across 3 cells remain
- Good guess for the minimum size of a triangle
  - Edge length roughly equals cell size
- Yields constant vertex density in space
- Does not pay attention to curvature

- more: http://mkrus.free.fr/CG/LODS/xrds/
Billboard

• A flat object that is always facing you
• Very cheap in terms of polygons (2 triangles)
• Needs a meaningful texture
• Example (from SketchUp): guy in the initial empty world rotates about his vertical axis to always face you
Chapter 3 - 3D Modeling

• Polygon Meshes
• Geometric Primitives
• Constructive Solid Geometry (CSG)
• Extrusion & Rotation
• Interpolation Curves
• Levels Of Detail (LOD)

• Volume- and Point-based Graphics
Volumes and Voxel Data

- Volume rendering = own field of research
  - e.g. surface reconstruction from voxels
  → Basics will be covered in a later chapter

- "Voxel" = "Volume" + "Pixel", i.e., voxel = smallest unit of volume
- Regular 3D grid in space
  - Each cell is either filled or not
  - Memory increases (cubic) with precision
- Solid object instead of boundary representation
- Rendering: "Minecraft"-like appearance possible
- Also the result of medical scanning devices
  - MRI, CT, 3D ultrasonic
Point-based Graphics

• Objects represented by point samples of their surface („Surfels“)
• Each point has a position and a color
• Surface can be visually reconstructed from these points
  • Purely image-based rendering
  • No mesh structure
  • Very simple source data (x, y, z, color)

• Point-data is acquired e.g., by 3D cameras
• Own rendering techniques
• Own pipeline
  → Own lecture

http://www.crs4.it/vic/data/images/img-exported/stmatthew_4px_full_shaded2.png