The Rendering Pipeline – A Second Look

Part 1: Geometry Processing

Geometry Database

*Needs to represent models for*
- Geometric primitives
- Relations between different primitives (transformations)
- Object materials
- Light sources
- Camera

Geometric Primitives

*Different philosophies:*
- Collections of complex shapes
  - Spheres, cones, cylinders, tori, …
- One simple type of geometric primitive
  - Triangles or triangle meshes
- Small set of complex primitives with adjustable parameters
  - E.g. “all polynomials of degree 2”
  - Splines, NURBS (details in CPSC 424)
  - Fractals

Geometry Database

*Triangles and Triangle Meshes:*

Mathematical representations:
- Explicit functions
- Parametric functions
- Implicit functions
Explicit Functions

**Curves:**
- y is a function of x: \( y := \sin(x) \)
- Only works in 2D

**Surfaces:**
- z is a function of x and y: \( z := \sin(x) + \cos(y) \)
- Cannot define arbitrary shapes in 3D

Parametric Functions

**Curves:**
- 2D: x and y are functions of a parameter value \( t \)
- 3D: x, y, and z are functions of a parameter value \( t \)

\[
C(t) := \begin{pmatrix}
\cos(t) \\
\sin(t) \\
t
\end{pmatrix}
\]

Geometry Database

**Implicit Surfaces:**
- Surface is defined implicitly via the roots of a function
- E.g: \( S(x, y, z) : x^2 + y^2 + z^2 - 1 = 0 \)

The Rendering Pipeline

**Modeling and Viewing Transformation**

- **Modeling transformation:** Map points from object coordinate system to world coordinate system
- **Viewing transformation:** Map points from world coordinate system to camera (or eye) coordinate system
Modeling and Viewing Transformation

Types of transformations:
- Rotations, scaling, shearing
- Translations
- Other transformations (not handled by rendering pipeline):
  - Freeform deformation

Affine transformations
- Linear transformations + translations
- Can be expressed as a 3x3 matrix + 3 vector
- E.g. rotation + translation:

\[
\begin{bmatrix}
    x' \\
    y' \\
    z'
\end{bmatrix}
= \begin{bmatrix}
    \cos(\phi) & -\sin(\phi) & 0 \\
    \sin(\phi) & \cos(\phi) & 0 \\
    0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
    z
\end{bmatrix} +
\begin{bmatrix}
    t_x \\
    t_y \\
    t_z
\end{bmatrix}
\]

Another representation: 4x4 homogeneous matrix

The Rendering Pipeline
**Lighting**

**Complex Lighting and Shading**

**The Rendering Pipeline**

Purpose:
- Project 3D geometry onto a 2D image plane
- Simulates a camera

Camera model:
- Pinhole camera
- Other, more complex camera models also exist in computer graphics, but are less common
  - Thin lens cameras
  - Full simulation of lens geometry

**Perspective Transformation**

Pinhole Camera:
- Light shining through a tiny hole into a dark room yields upside-down image on wall
Perspective Transformation

Pinhole Camera

In computer graphics:
- Image plane is conceptually in front of the center of projection
- Perspective transformations belong to a class of operations that are called projective transformations
- Linear and affine transformations also belong to this class
- All projective transformations can be expressed as 4x4 matrix operations

The Rendering Pipeline

Part 2: Rasterization & Fragment Processing

Scan Conversion
Scan Conversion

**Problem:**
- Line is infinitely thin, but image has finite resolution
- Results in steps rather than a smooth line
  - Jaggies
  - Aliasing
- One of the fundamental problems in computer graphics

Scan Conversion

**Color interpolation**
- Linearly interpolate per-pixel color from vertex color values
- Treat every channel of RGB color separately

Scan Conversion

**Color interpolation**
- Example:

Scan Conversion

**The Rendering Pipeline**
Texturing

Issues:
- How to map pixel from texture (texels) to screen pixels
  - Texture can appear widely distorted in rendering
  - Magnification / minification of textures
- Filtering of textures
- Preventing aliasing (anti-aliasing)
**Hidden Line Removal**

**Hidden Surface Removal**

**Depth Test / Hidden Surface Removal**

*Remove invisible geometry*
- Parts that are hidden behind other geometry

**Possible Implementations:**
- Per-fragment decision
  - Depth buffer
- Object space decision
  - Clipping polygons against each other
  - Sorting polygons by distance from camera

**The Rendering Pipeline**

**Display Technology**

*Cathod Ray Tubes (CRTs)*
Display Technology

**Raster Scan Electron Beam**

![Raster Scan Electron Beam Diagram]

**Interlaced Scanning**

![Interlaced Scanning Diagram]

Display Technology

**Color CRTs**

![Color CRTs Diagram]

**Trinitron CRTs**

![Trinitron CRTs Diagram]

Display Technology

**Liquid Crystal Displays (LCD)**

![Liquid Crystal Displays (LCD) Diagram]

Coming Up...

**Friday, May 12:**
- Geometric Transformations (Affine)

**Monday, May 15:**
- Geometric Transformations (Perspective)
- Rendering Geometry in OpenGL