3D Rendering

Adam Finkelstein
Princeton University
COS 426, Spring 2005

Course Syllabus

I. Image processing
II. Rendering
III. Modeling
IV. Animation

Where Are We Now?

I. Image processing
II. Rendering
III. Modeling
IV. Animation

Rendering

• Generate an image from geometric primitives

3D Rendering Example

3D scene representation
3D viewer representation
Visible surface determination
Lighting simulation

Overview

What issues must be addressed by a 3D rendering system?
Overview

» 3D scene representation
  • 3D viewer representation
  • Visible surface determination
  • Lighting simulation

How is the 3D scene described in a computer?

3D Scene Representation

• Scene is usually approximated by 3D primitives
  o Point
  o Line segment
  o Polygon
  o Polyhedron
  o Curved surface
  o Solid object
  o etc.

3D Scene Representation

• Scene is usually approximated by 3D primitives
  o Point
  o Line segment
  o Polygon
  o Polyhedron
  o Curved surface
  o Solid object
  o etc.

3D Point

• Specifies a location

3D Point

• Specifies a location
  o Represented by three coordinates
  o Infinitely small

typedef struct {
  Coordinate x;
  Coordinate y;
  Coordinate z;
} Point;

3D Vector

• Specifies a direction and a magnitude

3D Vector

• Specifies a direction and a magnitude
  o Represented by three coordinates
  o Magnitude \|V\| = \sqrt(dx^2 + dy^2 + dz^2)
  o Has no location

typedef struct {
  Coordinate dx;
  Coordinate dy;
  Coordinate dz;
} Vector;
3D Vector

• Dot product of two 3D vectors
  - \( \mathbf{V}_1 \cdot \mathbf{V}_2 = ||\mathbf{V}_1|| ||\mathbf{V}_2|| \cos(\theta) \)

• Cross product of two 3D vectors
  - \( \mathbf{V}_1 \times \mathbf{V}_2 = (dy_1dx_2 - dz_1dy_2, dz_1dx_2 - dx_1dz_2, dx_1dy_2 - dy_1dx_2) \)
  - \( ||\mathbf{V}_1 \times \mathbf{V}_2|| = ||\mathbf{V}_1|| ||\mathbf{V}_2|| \sin(\theta) \)

3D Line Segment

• Linear path between two points
  - Parametric representation:
    - \( \mathbf{P} = \mathbf{P}_1 + t (\mathbf{P}_2 - \mathbf{P}_1), \quad (0 \leq t \leq 1) \)

3D Ray

• Line segment with one endpoint at infinity
  - Parametric representation:
    - \( \mathbf{P} = \mathbf{P}_1 + t \mathbf{V}, \quad (0 \leq t < \infty) \)

3D Line

• Line segment with both endpoints at infinity
  - Parametric representation:
    - \( \mathbf{P} = \mathbf{P}_1 + t \mathbf{V}, \quad (-\infty < t < \infty) \)
### 3D Plane
- A linear combination of three points
  \[ \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3 \]

  \[ \mathbf{N} = (a, b, c) \]

  \[ d \]

  \[ \mathbf{N} \text{ is the plane “normal”} \]

  \[ \text{Unit-length vector} \]

  \[ \text{Perpendicular to plane} \]

  \[ \text{Implicit representation:} \]

  \[ \mathbf{P} \cdot \mathbf{N} + d = 0, \text{ or} \]

  \[ ax + by + cz + d = 0 \]

### 3D Polygon
- Area “inside” a sequence of coplanar points
  - Triangle
  - Quadrilateral
  - Convex
  - Star-shaped
  - Concave
  - Self-intersecting

  Points are in counter-clockwise order

  - Holes (use > 1 polygon struct)

### 3D Sphere
- All points at distance “r” from point “(c_x, c_y, c_z)”
  - Implicit representation:
    \[ (x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 = r^2 \]
  - Parametric representation:
    \[ x = r \cos(\theta) \cos(\phi) + c_x \]
    \[ y = r \cos(\theta) \sin(\phi) + c_y \]
    \[ z = r \sin(\theta) + c_z \]

### 3D Scenes
- Comprise set of geometric primitives

### Other Geometric Primitives
- More detail on 3D modeling later in course
  - Point
  - Line segment
  - Polygon
  - Polyhedron
  - Curved surface
  - Solid object
  - etc.
Overview

- 3D scene representation
  » 3D viewer representation
- Visible surface determination
- Lighting simulation

How is the viewing device described in a computer?

Camera Models

- The most common model is pin-hole camera
  - All captured light rays arrive along paths toward focal point without lens distortion (everything is in focus)
  - Sensor response proportional to radiance

Other models consider ...
- Depth of field
- Motion blur
- Lens distortion

View plane
Eye position (focal point)

Camera Parameters

- What are the parameters of a camera?
- Position
  - Eye position (px, py, pz)
- Orientation
  - View direction (dx, dy, dz)
  - Up direction (ux, uy, uz)
- Aperture
  - Field of view (xfov, yfov)
- Film plane
  - “Look at” point
  - View plane normal

View Frustum

View Frustum

Overview

- 3D scene representation
- 3D viewer representation
  » Visible surface determination
- Lighting simulation

How can the front-most surface be found with an algorithm?
Visible Surface Determination

- The color of each pixel on the view plane depends on the radiance emanating from visible surfaces.

Ray Casting

- For each sample ...
  - Construct ray from eye position through view plane
  - Find first surface intersected by ray through pixel
  - Compute color of sample based on surface radiance

Ray Casting

- For each sample ...
  - Construct ray from eye position through view plane
  - Find first surface intersected by ray through pixel
  - Compute color of sample based on surface radiance

Construct Ray

Ray: \( P = P_0 + tv \)

Find First Surface Intersection

\( P_0 V & T_1 \ T_2 \ T_3 \)
Visible Surface Determination

- For each sample ...
  - Construct ray from eye position through view plane
  - Find first surface intersected by ray through pixel
    - Compute color of sample based on surface radiance

More efficient algorithms utilize spatial coherence!

Rendering Algorithms

- Any samples can be used!
  - Rendering is a problem in sampling and reconstruction

Overview

- 3D scene representation
- 3D viewer representation
- Visible surface determination
  - Lighting simulation

How do we compute the radiance for each sample ray?

Lighting Simulation

- Lighting parameters
  - Light source emission
  - Surface reflectance
  - Atmospheric attenuation
  - Camera response

More on these methods later!

Lighting Simulation

- Direct illumination
  - Ray casting
  - Polygon shading

- Global illumination
  - Ray tracing
  - Monte Carlo methods
  - Radiosity methods

More on these methods later!
Summary

• Major issues in 3D rendering
  - 3D scene representation
  - 3D viewer representation
  - Visible surface determination
  - Lighting simulation

• Concluding note
  - Accurate physical simulation is complex and intractable
    » Rendering algorithms apply many approximations to simplify representations and computations

Next Lecture

• Ray intersections
• Light and reflectance models
• Indirect illumination

For assignment #2, you will write a ray tracer!