Einführung in Visual Computing

186.822

Ray Tracing

Werner Purgathofer
Surface-Rendering Methods

- polygon rendering methods
- ray tracing
- global illumination
- environment mapping
- texture mapping
- bump mapping
Ray Tracing in the Rendering Pipeline

scene objects in object space

transformed vertices in clip space

scene in normalized device coordinates

raster image in pixel coordinates

object capture/creation

modeling

viewing

projection

clipping + homogenization

viewport transformation

rasterization

shading

vertex stage ("vertex shader")

pixel stage ("fragment shader")
Ray Tracing Concepts

Visibility calculation

Shading

$s_1$ light source

$\alpha_1$

$\alpha_2$

$s_2$ 2nd light source

Visibility calculation
Ray Tracing Concepts

shading of the reflected object

shadows

reflection

transparency

$S_1$

$S_2$
Ray Tracing Concepts

for whole image: ray through every pixel

shadows

reflection

transparency

visibility calculation
Ray Tracing Concepts

for perspective projection: eye point

shadows
shading
reflection

transparency

visibility calculation
Ray Tracing Properties

- highly realistic images
- very time consuming
- multiple light sources
- visible-surface detection
- shadows
- reflections
- transparency

© W. Barth
Ray Tracing

- principles of geometric optics

projection reference point

ray-tracing coordinate reference frame

primary ray = eye point + t \cdot (pixel – eye point)
Shading: Diffuse Shading

\[ I_d = \text{xxx} \]

\( I_d \) … illumination caused by diffuse shading

\( \text{xxx} \) … any shading model

(Phong, Blinn-Phong, Cook-Torrance,...)
Ray Tracing: Shadows

ray = intersection point + t \cdot vector to light source

\[ ray = p + t \cdot (s - p) \]

- **p** ... intersection point
- **s** ... light source position

A light source influences the result only if there is no intersection with \( 0 < t < 1 \).
Ray Tracing: Shadows and Shading

- shadow ray along \( \ell \)
- ambient light \( k_a I_a \)
- diffuse reflection \( k_d (n \cdot \ell) \)
- specular reflection \( k_s (h \cdot n)^p \)

\[
I_d = k_a I_a + k_d (n \cdot \ell) + k_s (h \cdot n)^p
\]
Ray Tracing: Reflection

\[ I_r = k_r \cdot X_r \]

\( I_r \) ... illumination caused by reflection
\( k_r \) ... reflection coefficient of the material
\( X_r \) ... shading in the reflected direction

\( \alpha = \beta \)
Ray Tracing: Reflection Ray

- calculation of reflection ray

\[ r + v = (2n \cdot v)n \]
\[ r = (2n \cdot v)n - v \]
Ray Tracing: Transparency

\[ I_t = k_t \cdot X_t \]

- \( I_t \) ... illumination caused by transparency
- \( k_t \) ... transparency coefficient of the material
- \( X_t \) ... shading in the transparency direction

\[ \sin \theta_1 : \sin \theta_2 = \eta_2 : \eta_1 \]
Ray Tracing: Transparency Ray

- calculation of transparency ray

\[
\sin \theta_2 = \frac{\eta_1}{\eta_2} \sin \theta_1
\]

\[
t = -\frac{\eta_1}{\eta_2} v - (\cos \theta_2 - \frac{\eta_1}{\eta_2} \cos \theta_1)n
\]
Ray Tracing: A Complete Shading Method

\[ I = I_d + I_r + I_t \]

additional requirement: \[ k_d + k_r + k_t \leq 1 \]
Ray Tracing: Rays & Ray Tree

- primary, secondary rays

reflection and refraction
ray paths for one pixel

corresponding binary
ray tracing tree
Ray Tracing: Basic Algorithm

FOR all pixels $p_0$ DO

1. trace primary ray from eye $e$ to $p_0$
   find closest intersection $p$

2. FOR all light sources $s$ DO
   trace shadow feeler from $p$ to $s$
   IF no intersection between $p$ & $s$
   THEN shading += influence of $s$

3. IF surface of $p$ is reflective
   THEN trace secondary ray;
   shading += influence of reflection

4. IF surface of $p$ is transparent
   THEN trace secondary ray;
   shading += influence of transparency
Ray Tracing Examples
Ray Tracing Examples
True Global Illumination Example
Requirements for Object Data

(to use them for ray tracing)

- intersection calculation ray ↔ object possible
- surface normal calculation possible
  - B-Rep: simple
  - CSG: recursive evaluation
Ray-Surface Intersection

- ray equation
  \[ p(t) = p_0 + t \cdot d \]
  
  for primary rays
  \[ d = \frac{p_0 - e}{|p_0 - e|} \]

- for secondary rays
  \[ d = r \]
  \[ d = t \]

\begin{align*}
\text{describing a ray with an} \\
\text{initial-position vector } p_0 \\
\text{and unit direction vector } d
\end{align*}
Ray-Sphere Intersection

- parametric ray equation inserted into sphere equation

\[ |\mathbf{p} - \mathbf{c}|^2 - R^2 = 0 \]

\[ |(\mathbf{e} + t\mathbf{d}) - \mathbf{c}|^2 - R^2 = 0 \]

\[ \Delta \mathbf{p} = \mathbf{c} - \mathbf{e} \]

\[ t^2 - 2(\mathbf{d} \cdot \Delta \mathbf{p}) t + (|\Delta \mathbf{p}|^2 - R^2) = 0 \]

\[ t = \mathbf{d} \cdot \Delta \mathbf{p} \pm \sqrt{(\mathbf{d} \cdot \Delta \mathbf{p})^2 - |\Delta \mathbf{p}|^2 + R^2} \]
- discriminant negative $\Rightarrow$ no intersections

$$t = d \cdot \Delta p \pm \sqrt{(d \cdot \Delta p)^2 - |\Delta p|^2 + R^2}$$

$\rightarrow$ roundoff errors
when $R^2 << |\Delta p|^2$

“sphereflake”
Ray-Sphere Intersection

- discriminant negative $\Rightarrow$ no intersections
  $$t = \frac{d \cdot \Delta p \pm \sqrt{(d \cdot \Delta p)^2 - |\Delta p|^2 + R^2}}{R^2}$$
  $$\Rightarrow t = d \cdot \Delta p \pm \sqrt{R^2 - |\Delta p - (d \cdot \Delta p)d|^2}$$

(to avoid roundoff errors when $R^2 \ll |\Delta p|^2$)

$|\Delta p - (d \cdot \Delta p)d|^2 = |\Delta p|^2 - 2d^2|\Delta p|^2 + (d \cdot \Delta p)^2d^2$

because $d^2 = 1$
Ray-Polyhedron Intersection

- use **bounding sphere** to eliminate easy cases

  - ray does not hit bounding sphere
    - no intersection with object
  - ray hits bounding sphere
    - further investigation necessary
      - ray hits bounding sphere but no intersection with object
      - ray hits bounding sphere and intersection with object
Ray-Polyhedron Intersection

- use bounding sphere to eliminate easy cases
- locate front faces \( \mathbf{d} \cdot \mathbf{n} < 0 \)
- solving plane equation
  \[
  Ax + By + Cz + D = 0 \\
  \mathbf{n} = (A, B, C) \\
  \mathbf{n} \cdot \mathbf{p} = -D \\
  \mathbf{n} \cdot (\mathbf{e} + t\mathbf{d}) = -D \\
  t = -\frac{D + \mathbf{n} \cdot \mathbf{e}}{\mathbf{n} \cdot \mathbf{d}}
  \]
Ray-Polyhedron Intersection

- intersection point inside polygon boundaries?
- inside-outside test
- smallest $t$ to inside point is first intersection point of polyhedron

![Diagram showing a ray intersecting a polyhedron and testing for inside or outside.]
Ray-Surface Intersection

- quadric, spline surfaces:
  - parametric ray equation inserted into surface definition
  - methods like numerical root-finding, incremental calculations
Reducing Object-Intersection Calculations

- bounding volumes
- bounding volume hierarchies

2nd hierarchy bounding spheres

3rd hierarchy bounding spheres

bounding sphere
Reducing Object-Intersection Calculations

- space-subdivision methods
  - regular grid
  - octree

- preprocess: find object data in each cube
Reducing Object-Intersection Calculations

- space-subdivision methods
  - incremental grid traversal
    - 3D Bresenham
    - processing of potential exit faces

Ray traversal through a subregion of a cube enclosing a scene
Incremental Grid Traversal

- ray direction \( \mathbf{d} \) / ray entry position \( \mathbf{p}_{\text{in}} \)
- potential exit faces \( \mathbf{d} \cdot \mathbf{n}_k > 0 \)
- normal vectors

\[
\mathbf{n}_k = \begin{cases} 
(\pm 1, 0, 0) \\
(0, \pm 1, 0) \\
(0, 0, \pm 1) 
\end{cases}
\]

- check signs of components of \( \mathbf{d} \)
Incremental Grid Traversal

- calculation of exit positions, select smallest $t_k$

$$p_{\text{out},k} = p_{\text{in}} + t_k d$$

$$n_k \cdot p_{\text{out},k} = -D_k$$

$$t_k = \frac{-D_k - n_k \cdot p_{\text{in}}}{n_k \cdot d}$$

- example: $n_k = (1,0,0)$

$$x_k = -D_k \quad \Rightarrow \quad t_k = \frac{x_k - x_0}{x_d}$$
Incremental Grid Traversal

- variation: trial exit plane
  - perpendicular to largest component of $\mathbf{d}$
  - exit point in 0  ⇒  done
  - $\{1, 2, 3, 4\}$  ⇒  side clear
  - $\{5, 6, 7, 8\}$  ⇒  extra calc.

sectors of the trial exit plane