Interactive Computer Graphics

Lecture 9: Ray tracing

Object-ordering (polygon) rendering

- Object-ordering rendering
  - Each polygon or triangle is processed in turn and its visibility is determined using algorithms such as z-buffering
  - After determining its visibility, the polygon or triangle is projected onto the viewing plane
  - Polygon or triangle is shaded

$$I_{reflected} = k_a + I_i k_d \mathbf{n.s} / |\mathbf{n}||\mathbf{s}| + I_i K_s (\mathbf{r.v} / |\mathbf{r}||\mathbf{v}|)^t$$

- Advantages:
  - Fast rendering can be implemented by hardware

Problems with Object Oriented Rendering

- Disadvantages:
  - difficult to model lighting effects such as reflection or transparency
  - difficult to model lighting effects which are caused by the interaction of objects, i.e. shadows
- Assumption:
  - Light travels from visible objects towards the eye

Ray tracing - rays travel from the eye to the scene

- Question:
  - Do light rays proceed from the eye to the light source or from the light source to the eye ?
- To answer the question, it is important to understand some of the fundamental laws of physics:
  1. Light rays do travel in straight lines
  2. Light rays do not interfere if they cross
  3. Light rays travel from the light source to the eye but the physics are invariant under path reversal

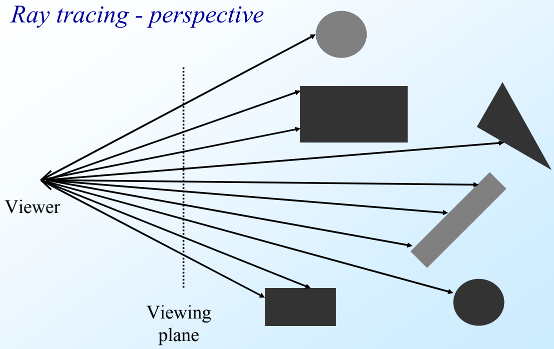
Ray tracing

- Image order rendering techniques:
  - ray tracing
  - volume rendering
- Ray tracing allows to add more realism to the rendered scene by allowing effects such as
  - shadows
  - transparency
  - reflections

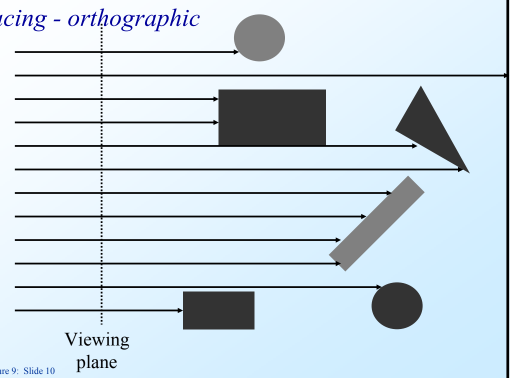
Ray tracing

- Ray tracing works as follows:
  - For each pixel on the screen, a ray is defined by the line joining the viewpoint and the pixel of the viewing plane (perspective projection) or by a line orthogonal to the viewing place (orthographic or parallel projection)
  - Each ray is cast through the viewing volume and checked for intersections with the objects inside the viewing volume

Ray tracing - perspective



Ray tracing - orthographic



Ray tracing: Primary rays

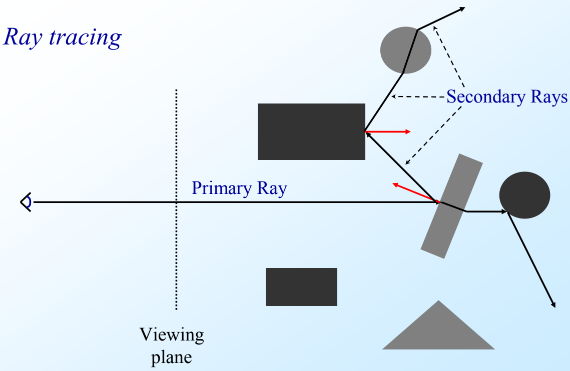
- For each ray we need to test which objects are intersecting the ray:
  - If the object has an intersection with the ray we calculate the distance between viewpoint and intersection
  - If the ray has more than one intersection, the smallest distance identifies the visible surface.
- Primary rays are rays from the view point to the nearest intersection point
- We compute the illumination as before

$$I_{reflected} = k_a + I_i k_d \, \mathbf{n.s} / |\mathbf{n}||\mathbf{s}| + I_i K_s (\mathbf{r.v} / |\mathbf{r}||\mathbf{v}|)^t$$

Ray tracing: Secondary rays

- Secondary rays are rays originating at the intersection points
- Secondary rays are caused by
  - rays reflected off the intersection point in the direction of reflection
  - rays transmitted through transparent materials in the direction of refraction

Ray tracing



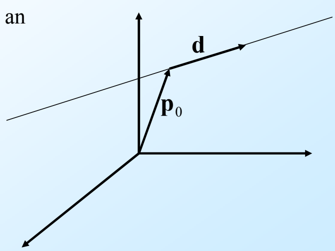
Surface calculations

- For each ray we must calculate all possible intersections with each object inside the viewing volume
- For each ray we must find the nearest intersection point
- We can define our scene using
  - Solid models
    - sphere
    - cylinder
  - Surface models
    - plane
    - triangle
    - polygon

Rays

- Rays are parametric lines
- Rays can be defined an
  - origin $\mathbf{p_0}$
  - direction $\mathbf{d}$
- Equation of ray:

$$\mathbf{p}(\mu) = \mathbf{p}_0 + \mu\mathbf{d}$$



Ray tracing: Intersection calculations

- The coordinates of any point along each primary ray are given by:

$$\mathbf{p} = \mathbf{p}_0 + \mu\mathbf{d}$$

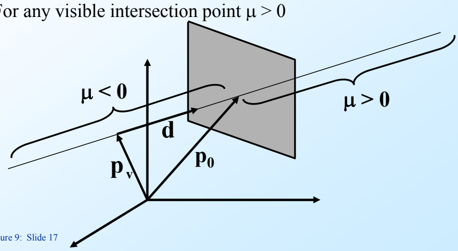
  - $\mathbf{p}_0$ is the current pixel on the viewing plane.
  - $\mathbf{d}$ is the direction vector and can be obtained from the position of the pixel on the viewing plane $\mathbf{p}_0$ and the viewpoint $\mathbf{p}_\mathbf{v}$:

$$\mathbf{d} = \frac{\mathbf{p}_0 - \mathbf{p}_\mathbf{v}}{|\mathbf{p}_0 - \mathbf{p}_\mathbf{v}|}$$

  - for a primary ray $\mathbf{p}_\mathbf{v}$ is usually the origin

Ray tracing: Intersection calculations

- The viewing ray can be parameterized by μ:
  - $\mu > 0$ denotes the part of the ray behind the viewing plane
  - $\mu < 0$ denotes the part of the ray in front of the viewing plane
  - For any visible intersection point $\mu > 0$



Intersection calculations: Spheres

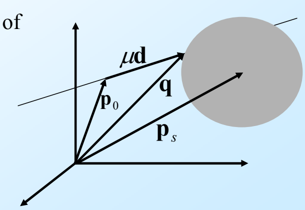
let the center be $\mathbf{p_s}$

let a point $\mathbf{q}$ be on the sphere

For any point on the surface of the sphere

$$|\mathbf{q} - \mathbf{p_s}|^2 - r^2 = 0$$

where r is the radius of the sphere



Intersection calculations: Spheres

- To test whether a ray intersects a surface we can substitute for **q** using the ray equation:

$$|\mathbf{p}_0 + \mu\mathbf{d} - \mathbf{p}_s|^2 - r^2 = 0$$

- Setting $\Delta\mathbf{p} = \mathbf{p}_0 - \mathbf{p}_s$ and expanding the dot product produces the following quadratic equation:

$$\mu^2 + 2\mu(\mathbf{d}\cdot\Delta\mathbf{p}) + |\Delta\mathbf{p}|^2 - r^2 = 0$$

Intersection calculations: Spheres

- The quadratic equation has the following solution:

$$\mu = -\mathbf{d}\cdot\Delta\mathbf{p} \pm \sqrt{(\mathbf{d}\cdot\Delta\mathbf{p})^2 - |\Delta\mathbf{p}|^2 + r^2}$$

- Solutions:
  - if the quadratic equation has no solution, the ray does not intersect the sphere
  - if the quadratic equation has two solutions ($\mu_1 < \mu_2$):
    - μ_1 corresponds to the point at which the rays enters the sphere
    - μ_2 corresponds to the point at which the rays leaves the sphere

Problem Time

- Given:
  - the viewpoint is at $\mathbf{p}_v$ = (0, 0, -10)
  - the ray passes through viewing plane at $\mathbf{p}_i$ = (0, 0, 0).
- Spheres:
  - Sphere A with center $\mathbf{p}_s$ = (0, 0, 8) and radius r = 5
  - Sphere B with center $\mathbf{p}_s$ = (0, 0, 9) and radius r = 3
  - Sphere C with center $\mathbf{p}_s$ = (0, -3, 8) and radius r = 2
- Calculate the intersections of the ray with the spheres above.

Solution

- The direction vector is **d** = (0, 0, 10) / 10 = (0, 0, 1)
  - Sphere A:

    Δp = (0, 0, 8), so μ = 8 ± sqrt(64 – 64 + 25) = 8 ± 5

    As the result, the ray enters A sphere at (0, 0, 3) and exits the sphere at (0, 0, 13)).
  - Sphere B:

    Δp = (0, 0, 9), so μ = 9 ± sqrt(81 – 81 + 9) = 9 ± 3

    As the result, the ray enters B sphere at (0, 0, 6) and exits the sphere at (0, 0, 12)).
  - Sphere C has no intersections with ray.

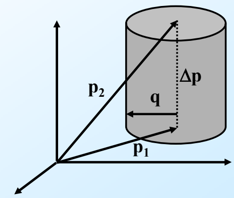
Intersection calculations: Cylinders

- A cylinder can be described by
  - a position vector $\mathbf{p}_1$ describing the first end point of the long axis of the cylinder
  - a position vector $\mathbf{p}_2$ describing the second end point of the long axis of the cylinder
  - a radius r
- The axis of the cylinder can be written as $\Delta\mathbf{p} = \mathbf{p}_1 - \mathbf{p}_2$ and can be parameterized by $0 \le \alpha \le 1$

Intersection calculations: Cylinders

- To calculate the intersection of the cylinder with the ray:

$$\mathbf{p}_1 + \alpha\Delta\mathbf{p} + \mathbf{q} = \mathbf{p}_0 + \mu\mathbf{d}$$



- Since $\mathbf{q}\cdot\Delta\mathbf{p} = 0$ we can write

$$\alpha(\Delta\mathbf{p}\cdot\Delta\mathbf{p}) = \mathbf{p}_0\cdot\Delta\mathbf{p} + \mu\mathbf{d}\cdot\Delta\mathbf{p} - \mathbf{p}_1\cdot\Delta\mathbf{p}$$

Intersection calculations: Cylinders

- Solving for α yields:

$$\alpha = \frac{\mathbf{p}_0 \cdot \Delta\mathbf{p} + \mu\mathbf{d} \cdot \Delta\mathbf{p} - \mathbf{p}_1 \cdot \Delta\mathbf{p}}{\Delta\mathbf{p} \cdot \Delta\mathbf{p}}$$

- Substituting we obtain:

$$\mathbf{q} = \mathbf{p}_0 + \mu\mathbf{d} - \mathbf{p}_1 - \left(\frac{\mathbf{p}_0 \cdot \Delta\mathbf{p} + \mu\mathbf{d} \cdot \Delta\mathbf{p} - \mathbf{p}_1 \cdot \Delta\mathbf{p}}{\Delta\mathbf{p} \cdot \Delta\mathbf{p}} \right) \Delta\mathbf{p}$$

Intersection calculations: Cylinders

- Using the fact that $\mathbf{q} \cdot \mathbf{q} = r^2$ we can use the same approach as before to the quadratic equation for μ:

$$r^2 = \left(\mathbf{p}_0 + \mu\mathbf{d} - \mathbf{p}_1 - \left(\frac{\mathbf{p}_0 \cdot \Delta\mathbf{p} + \mu\mathbf{d} \cdot \Delta\mathbf{p} - \mathbf{p}_1 \cdot \Delta\mathbf{p}}{\Delta\mathbf{p} \cdot \Delta\mathbf{p}} \right) \Delta\mathbf{p} \right)^2$$

  - If the quadratic equation has no solution: no intersection
  - If the quadratic equation has two solutions: intersection

Intersection calculations: Cylinders

- Assuming that $\mu_1 \leq \mu_2$ we can determine two solutions:

$$\alpha_1 = \frac{\mathbf{p}_0 \cdot \Delta\mathbf{p} + \mu_1\mathbf{d} \cdot \Delta\mathbf{p} - \mathbf{p}_1 \cdot \Delta\mathbf{p}}{\Delta\mathbf{p} \cdot \Delta\mathbf{p}}$$

$$\alpha_2 = \frac{\mathbf{p}_0 \cdot \Delta\mathbf{p} + \mu_2\mathbf{d} \cdot \Delta\mathbf{p} - \mathbf{p}_1 \cdot \Delta\mathbf{p}}{\Delta\mathbf{p} \cdot \Delta\mathbf{p}}$$

- If the value of α_1 is between 0 and 1 the intersection is on the outside surface of the cylinder
- If the value of α_2 is between 0 and 1 the intersection is on the inside surface of the cylinder

Intersection calculations: Plane

- Objects are often described by geometric primitives such as
  - triangles
  - planar quads
  - planar polygons
- To test intersections of the ray with these primitives we must whether the ray will intersect the plane defined by the primitive

Intersection calculations: Plane

- The intersection of a ray with a plane is given by

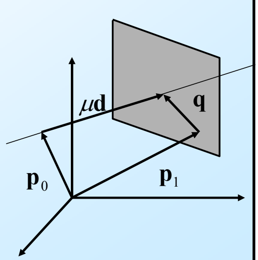
$$\mathbf{p}_1 + \mathbf{q} = \mathbf{p}_0 + \mu\mathbf{d}$$

where $\mathbf{p_1}$ is a point in the plane. Subtracting $\mathbf{p_1}$ and multiplying with the normal of the plane **n** yields:

$$\mathbf{q} \cdot \mathbf{n} = \mathbf{0} = (\mathbf{p}_0 - \mathbf{p}_1) \cdot \mathbf{n} + \mu\mathbf{d} \cdot \mathbf{n}$$

- Solving for μ yields:

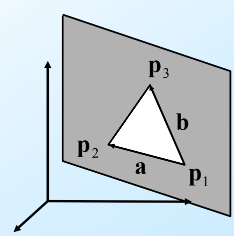
$$\mu = -\frac{(\mathbf{p}_0 - \mathbf{p}_1) \cdot \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}$$



Intersection calculations: Triangles

- To calculate intersections:
  - test whether triangle is front facing
  - test whether plane of triangle intersects ray
  - test whether intersection point is inside triangle
- If the triangle is front facing:

$$\mathbf{d} \cdot \mathbf{n} < 0$$



Intersection calculations: Triangles

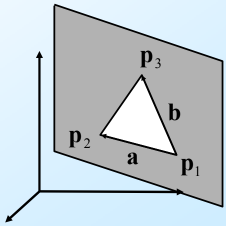
- To test whether plane of triangle intersects ray
  - calculate equation of the plane using

$$\mathbf{p}_2 - \mathbf{p}_1 = \mathbf{a}$$
$$\mathbf{p}_3 - \mathbf{p}_1 = \mathbf{b}$$

  - calculate intersections with plane as before

$$\mathbf{n} = \mathbf{a} \times \mathbf{b}$$

- To test whether intersection point is inside triangle: $\mathbf{q} = \alpha\mathbf{a} + \beta\mathbf{b}$



Intersection calculations: Triangles

- A point is inside the triangle if

$$0 \le \alpha \le 1$$
$$0 \le \beta \le 1$$
$$\alpha + \beta \le 1$$

- Calculate α and β by taking the dot product with **a** and **b:**

$$\alpha = \frac{(\mathbf{b}\cdot\mathbf{b})(\mathbf{q}\cdot\mathbf{a}) - (\mathbf{a}\cdot\mathbf{b})(\mathbf{q}\cdot\mathbf{b})}{(\mathbf{a}\cdot\mathbf{a})(\mathbf{b}\cdot\mathbf{b}) - (\mathbf{a}\cdot\mathbf{b})^2}$$

$$\beta = \frac{\mathbf{q}\cdot\mathbf{b} - \alpha(\mathbf{a}\cdot\mathbf{b})}{\mathbf{b}\cdot\mathbf{b}}$$

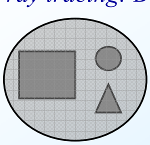
Accelerating ray tracing

- Ray tracing is slow:
  - 10^3 x 10^3 = 10^6 rays x 100 calculations per intersection test
- Ray tracing spends most of the time calculating intersections with objects in the scene
- Ray tracing can be accelerated by reducing the number of intersection tests:
  - enclose groups of adjacent objects in a bounding volume
  - test whether the ray intersects the bounding volume

Accelerating ray tracing

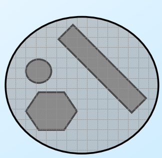
- Intersection calculation:
  - if the ray does not intersect the bounding volume, the ray will not intersect any objects within the bounding volume
  - if the ray does intersect the bounding volume, each of the objects within the bounding volume must be checked for intersections with the ray
- Bounding volumes:
  - bounding spheres
  - bounding boxes

Accelerating ray tracing: Bounding volumes

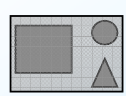


Bounding sphere

Bounding sphere

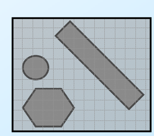


Accelerating ray tracing: Bounding volumes

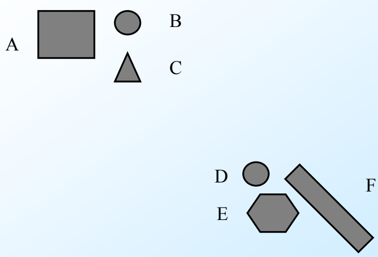


Bounding box

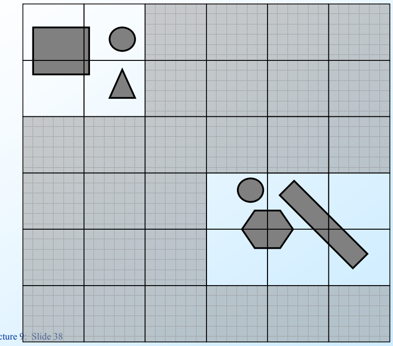
Bounding box



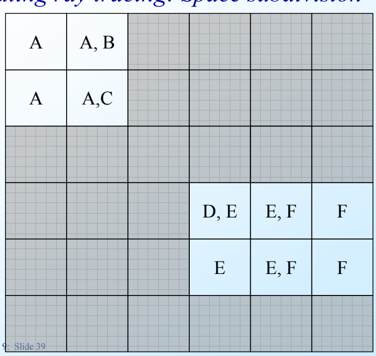
Accelerating ray tracing



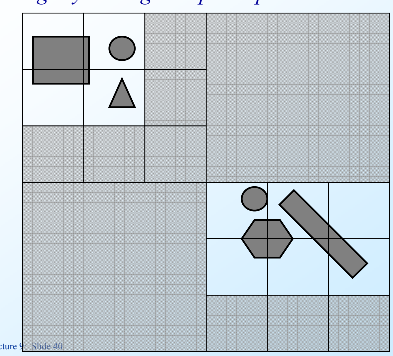
Accelerating ray tracing: Space subdivision



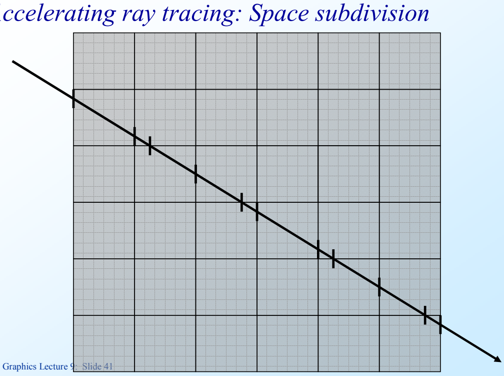
Accelerating ray tracing: Space subdivision



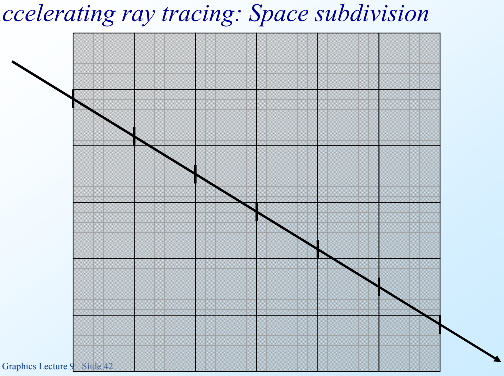
Accelerating ray tracing: Adaptive space subdivision



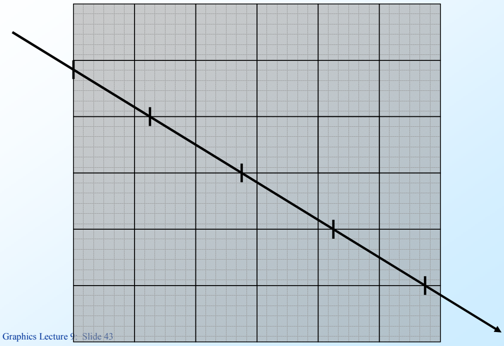
Accelerating ray tracing: Space subdivision



Accelerating ray tracing: Space subdivision



Accelerating ray tracing: Space subdivision



More information and downloads

- Persistence of Vision Raytracer
  - http://www.povray.org/
- Some examples are shown on the following slides