Last Time?

- Luxo Jr.
- Applications of Computer Graphics
- Overview of the semester
- IFS
  - Assignment 0 due tomorrow @ 11:59pm
- Questions?

Notes on Assignments

- Make sure you turn in a **linux** or **windows** executable (so we can test your program)
- Don’t use athena dialup
- In your **README.txt**
  - time spent, collaborators, known bugs, extensions
- 6.837-staff@graphics.csail.mit.edu

Administrivia: Lab & Office Hours

- Barb
  - Mondays 6-8pm in W20-575
- Fredo
  - Tuesdays 6-7pm in W20-575
- Rob
  - Wednesdays 8-11pm in W20-575
- Send email to make an appointment for some other time

Overview of Today

- Ray Casting Basics
- Camera and Ray Generation
- Ray-Plane Intersection
- Ray-Sphere Intersection

Ray Casting

*For every pixel*

Construct a ray from the eye
For every object in the scene
Find intersection with the ray
Keep if closest
**Shading**

For every pixel:
- Construct a ray from the eye
- For every object in the scene
  - Find intersection with the ray
  - Keep if closest
- Shade depending on light and normal vector

**A Note on Shading**

- Surface/Scene Characteristics:
  - surface normal
  - direction to light
  - viewpoint
- Material Properties:
  - Diffuse (matte)
  - Specular (shiny)
  - ...
- Much more next Thursday!

**Ray Tracing**

- Secondary rays (shadows, reflection, refraction)
- In a couple of weeks

**Ray Casting**

For every pixel:
- Construct a ray from the eye
- For every object in the scene
  - Find intersection with the ray
  - Keep if closest
- Shade depending on light and normal vector

**Ray Representation?**

- Two vectors:
  - Origin
  - Direction (normalized is better)
- Parametric line
  - \( P(t) = \text{origin} + t \times \text{direction} \)
Durer’s Ray Casting Machine

• Albrecht Durer, 16th century

Durer’s Ray Casting Machine

• Albrecht Durer, 16th century

Questions?

Henrik Wann Jensen & Stephen Duck

Overview of Today

• Ray Casting Basics

• Camera and Ray Generation

• Ray-Plane Intersection

• Ray-Sphere Intersection

Cameras

For every pixel

Construct a ray from the eye

For every object in the scene

Find intersection with ray

Keep if closest
Pinhole Camera

- Box with a tiny hole
- Inverted image
- Similar triangles
- Perfect image if hole infinitely small
- Pure geometric optics
- No depth of field issue

Oldest Illustration

- From R. Gemma Frisius, 1545

Camera Obscura

Camera Obscura Today

Abelardo Morell
www.abelardomorell.net

Camera Obscura in Art

Simplified Pinhole Camera

- Eye-image pyramid (frustum)
- Note that the distance/size of image are arbitrary
Camera Description?

- Eye point \( e \) (center)
- Orthobasis \( u, v, w \) (horizontal, up, -direction)
- Field of view \( \text{angle} \)
- Image rectangle \( \text{height, width} \)

Perspective vs. Orthographic

- Parallel projection
- No foreshortening
- No vanishing point

Orthographic Camera

- Ray Generation?
  - Origin = center + (x-0.5)*size*horizontal + (y-0.5)*size*up
  - Direction is constant

Other Weird Cameras

- E.g. fish eye, omnimax, panorama

Questions?

Overview of Today

- Ray Casting Basics
- Camera and Ray Generation
- Ray-Plane Intersection
- Ray-Sphere Intersection
Ray Casting
For every pixel
Construct a ray from the eye
For every object in the scene
Find intersection with the ray
Keep if closest
First we will study ray-plane intersection

Recall: Ray Representation
• Parametric line
  \( P(t) = R_0 + t \cdot R_d \)
• Explicit representation

Explicit vs. Implicit?
• Ray equation is explicit \( P(t) = R_0 + t \cdot R_d \)
  – Parametric
  – Generates points
  – Hard to verify that a point is on the ray
• Plane equation is implicit \( H(P) = n \cdot P + D = 0 \)
  – Solution of an equation
  – Does not generate points
  – Verifies that a point is on the plane
• Exercise: Explicit plane and implicit ray

Ray-Plane Intersection
• Intersection means both are satisfied
• So, insert explicit equation of ray into implicit equation of plane & solve for \( t \)
  \[
  P(t) = R_0 + t \cdot R_d \\
  H(P) = n \cdot P + D = 0 \\
  n \cdot (R_0 + t \cdot R_d) + D = 0 \\
  t = -(D + n \cdot R_0) / n \cdot R_d
  \]

Additional Housekeeping
• Verify that intersection is closer than previous
  \( P(t) < t_{\text{current}} \)
• Verify that it is not out of range (behind eye)
  \( P(t) > t_{\text{min}} \)
Normal

- For shading
  - diffuse: dot product between light and normal
- Normal is constant

Questions?

Overview of Today

- Ray Casting Basics
- Camera and Ray Generation
- Ray-Plane Intersection
- Ray-Sphere Intersection

Sphere Representation?

- Implicit sphere equation
  - Assume centered at origin (easy to translate)
  - \( H(P) = P \cdot P - r^2 = 0 \)

Ray-Sphere Intersection

- Insert explicit equation of ray into implicit equation of sphere & solve for \( t \)

\[
\begin{align*}
  P(t) &= R_o + tR_d \\
  H(P) &= P \cdot P - r^2 = 0 \\
  (R_o + tR_d) \cdot (R_o + tR_d) - r^2 &= 0 \\
  R_d \cdot Rd^2 + 2R_d \cdot R_d t + R_o \cdot R_o - r^2 &= 0
\end{align*}
\]

- Quadratic: \( at^2 + bt + c = 0 \)
  - \( a = 1 \) (remember, \( ||R_d|| = 1 \))
  - \( b = 2R_d \cdot R_o \)
  - \( c = R_o \cdot R_o - r^2 \)

- with discriminant \( d = b^2 - 4ac \)
- and solutions \( t = \frac{-b \pm d}{2a} \)
Ray-Sphere Intersection

- 3 cases, depending on the sign of $b^2 - 4ac$
- What do these cases correspond to?
- Which root ($t_+$ or $t_-$) should you choose?
  - Closest positive! (usually $t_-$)

Ray-Sphere Intersection

- It's so easy that all ray-tracing images have spheres!

Geometric Ray-Sphere Intersection

- Shortcut / easy reject
- What geometric information is important?
  - Ray origin inside/outside sphere?
  - Closest point to ray from sphere origin?
  - Ray direction: pointing away from sphere?

Geometric Ray-Sphere Intersection

- Is ray origin inside/outside/on sphere?
  - $(R_o \cdot R_o < r^2)$
  - $(R_o \cdot R_o > r^2)$
  - $(R_o \cdot R_o = r^2)$
  - If origin on sphere, be careful about degeneracies…

Geometric Ray-Sphere Intersection

- Is ray origin inside/outside/on sphere?
- Find closest point to sphere center, $t_p = -R_o \cdot R_d$
  - If origin outside & $t_p < 0 \rightarrow$ no hit
- Find squared distance, $d^2 = R_o \cdot R_o - t_p^2$
  - If $d^2 > r^2 \rightarrow$ no hit
**Geometric Ray-Sphere Intersection**

- Is ray origin inside/outside/on sphere?
- Find closest point to sphere center, \( t_P = -R_o \cdot R_d \)
- Find squared distance: \( d^2 = R_o \cdot R_o - t_P^2 \)
- Find distance \((t')\) from closest point \( (t_P)\) to correct intersection: \( t'^2 = r^2 - d^2 \)
  - If origin outside sphere \( \rightarrow t = t_P - t' \)
  - If origin inside sphere \( \rightarrow t = t_P + t' \)

**Geometric vs. Algebraic**

- Algebraic is simple & generic
- Geometric is more efficient
  - Timely tests
  - In particular for rays outside and pointing away

**Sphere Normal**

- Simply \( Q/||Q|| \)
  - \( Q = P(t) \), intersection point
  - (for spheres centered at origin)

**Questions?**

**Precision**

- What happens when
  - Origin is on an object?
  - Grazing rays?
- Problem with floating-point approximation

**The evil \( \epsilon \)**

- In ray tracing, do NOT report intersection for rays starting at the surface (no false positive)
  - Because secondary rays
  - Requires epsilons
The evil $\epsilon$: a hint of nightmare

- Edges in triangle meshes
  - Must report intersection (otherwise not watertight)
  - No false negative

Assignment 1: Ray Casting

- Write a basic ray caster
  - Orthographic camera
  - Sphere Intersection
  - Main loop rendering
  - 2 Display modes: color and distance
- We provide:
  - Ray: origin, direction
  - Hit: $t$, Material, $(normal)$
  - Scene Parsing

Object-Oriented Design

- We want to be able to add primitives easily
  - Inheritance and virtual methods
- Even the scene is derived from Object3D!

Graphics Textbooks

- Recommended for 6.837: Peter Shirley
  *Fundamentals of Computer Graphics*
  AK Peters
  - Ray Tracing

Next Time: More Ray Casting

- Other primitives
  - Boxes
  - Polygons
  - Triangles
  - IFS?