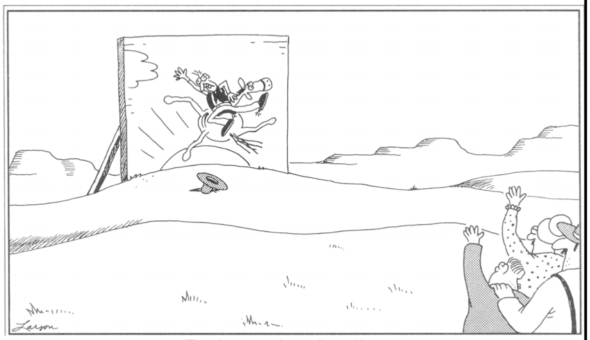
MIT 6.837 - Ray Tracing



The embarrassment of riding off into a fake sunset

Tony DeRose - *Math in the Movies*

Tony DeRose: Pixar Animation Studios- Senior Scientist
Date: **10-5-2004 (one week from today!)**
Time: 1:00 PM- 2:00 PM
Location: 32–D449 (Stata Center, Patil/Kiva)

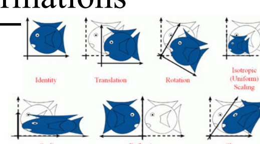
Film making is undergoing a digital revolution brought on by advances in areas such as computer technology, computational physics and computer graphics. This talk will provide a behind the scenes look at how fully digital films--- such as Pixar's "Monster's Inc" and "Finding Nemo" --- are made, with particular emphasis on the role that mathematics plays in the revolution.

Final Exam

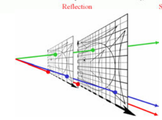
- … has been scheduled
- Thursday December 16th, 1:30-3:30pm
- DuPont
- Open Book

Last Week: Transformations

- Linear, affine and projective transforms



- Homogeneous coordinates

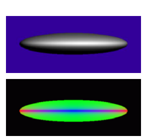


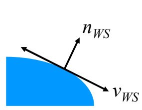
- Matrix notation
- Transformation composition is not commutative

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Last Week: Transformations

- Transformations in Ray Tracing
  - Transforming the ray
    Remember: points & directions transform differently!
  - Normalizing direction & what to do with t
  - Normal transformation



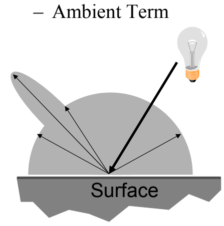


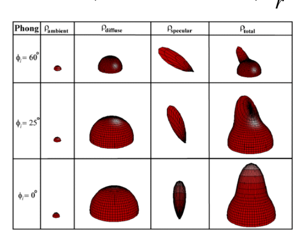
$$n_{WS}^{\mathbf{T}} = n_{OS}\ (\mathbf{M}^{-1})$$

Last Time: Local Illumination

- BRDF – Bidirectional Reflectance Distribution Function
- Phong Model- Sum of 3 components:
  - Diffuse Shading
  - Specular Highlight
  - Ambient Term

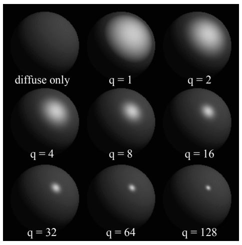
$$L_o = k_a + \left(k_d(\mathbf{n}\cdot\mathbf{l}) + k_s(\mathbf{v}\cdot\mathbf{r})^q\right)\frac{L_i}{r^2}$$



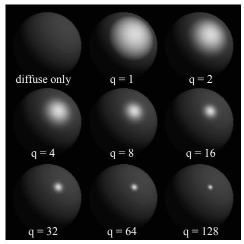


Phong Examples

- Shininess coefficient controls the "spread" of the specular highlight



Phong

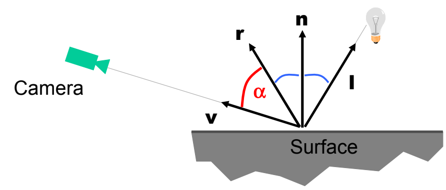


Blinn-Torrance
(scaled to approximate Phong)

The Phong Model

- Parameters
  - k_s: specular reflection coefficient
  - q : specular reflection exponent

$$L_o = k_s (\cos\alpha)^q \frac{L_i}{r^2} = k_s (\mathbf{v} \cdot \mathbf{r})^q \frac{L_i}{r^2}$$

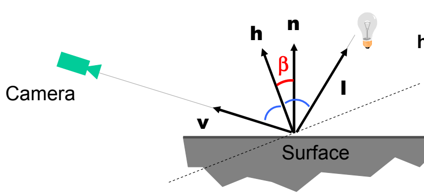


Blinn-Torrance Variation

- Parameters
  - k_s: specular reflection coefficient
  - q : specular reflection exponent

Implement this version of Phong (because it's what OpenGL uses & we want to match)

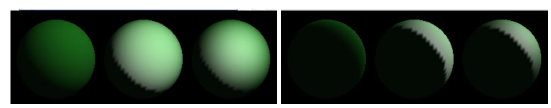
$$L_o = k_s (\cos\beta)^q \frac{L_i}{r^2} = k_s (\mathbf{n} \cdot \mathbf{h})^q \frac{L_i}{r^2}$$



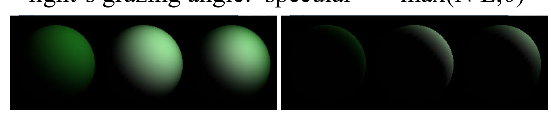
$$\mathbf{h} = \frac{\mathbf{l}+\mathbf{v}}{\|\mathbf{l}+\mathbf{v}\|}$$

Additional Phong Clamping Term

- Surfaces facing away from the light should not be lit (if N·L < 0)

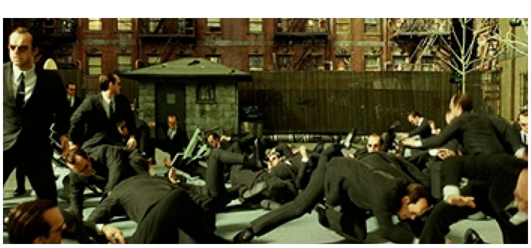


- Scale by dot product to avoid a sharp edge at the light's grazing angle: specular *= max(N·L,0)



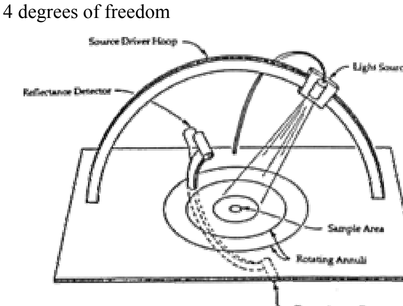
BRDFs in the Movie Industry

- http://www.virtualcinematography.org/publications/acrobat/BRDF-s2003.pdf
- For the Matrix movies
- Agent Smith clothes are CG, with measured BRDF



How Do We Obtain BRDFs?

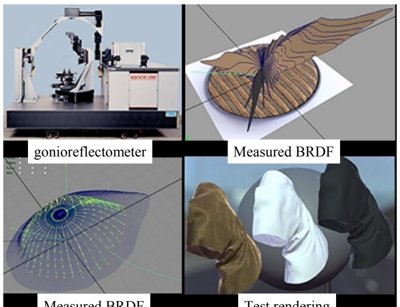
- Gonioreflectometer
  - 4 degrees of freedom



Source: Greg Ward

BRDFs in the Movie Industry

- http://www.virtualcinematography.org/publications/acrobat/BRDF-s2003.pdf
- For the Matrix movies



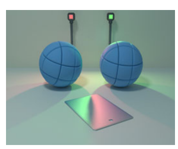
BRDFs in the Movie Industry

Photo CG Photo CG

BRDF Models

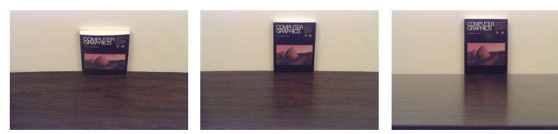
- Phenomenological
  - Phong [75]
    - Blinn [77]
  - Ward [92]
  - Lafortune et al. [97]
  - Ashikhmin et al. [00]
- Physical
  - Cook-Torrance [81]
  - He et al. [91]

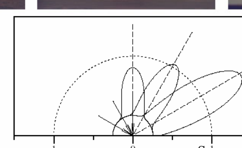
Roughly increasing computation time



Fresnel Reflection

- Increasing specularity near grazing angles.

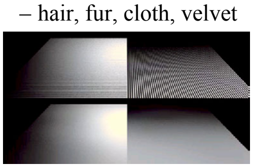




Source: Lafortune et al. 97

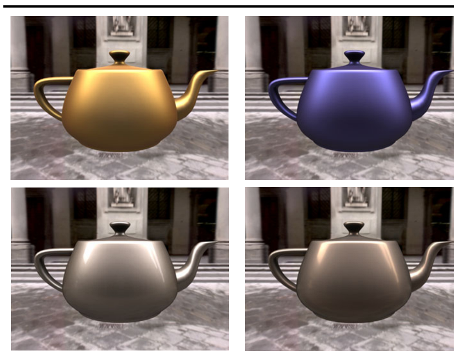
Anisotropic BRDFs

- Surfaces with strongly oriented microgeometry elements
- Examples:
  - brushed metals,
  - hair, fur, cloth, velvet



Source: Westin et.al 92

Questions?



Today: Ray Tracing

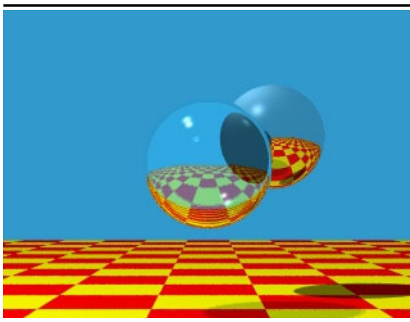
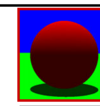


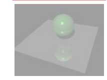
Image by Turner Whitted

Overview of Today

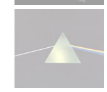
- Shadows



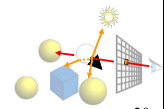
- Reflection



- Refraction



- Recursive Ray Tracing



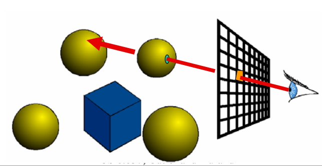
Ray Casting (a.k.a. Ray Shooting)

```
for every pixel
  construct a ray
  for every object
      intersect ray with object
```

Complexity?

O(n * m)

n = number of objects,   m = number of pixels



Ray Casting with Phong Shading

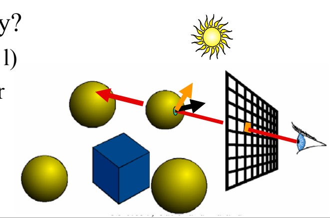
When you've found the closest intersection:

```
color = ambient*hit->getMaterial()->getDiffuseColor()
for every light
   color += hit->getMaterial()->Shade
               (ray, hit, directionToLight, lightColor)
return color
```

Complexity?

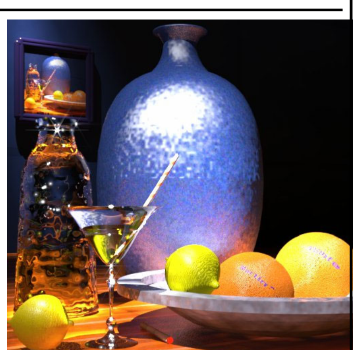
O(n * m * l)

l = number of lights



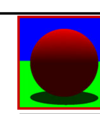
Questions?

- Image computed using the RADIANCE system by Greg Ward

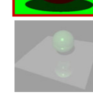


Overview of Today

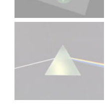
- Shadows



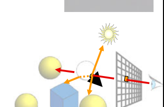
- Reflection



- Refraction

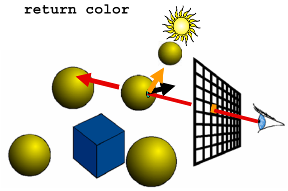


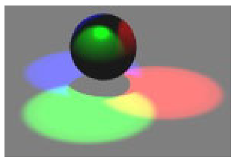
- Recursive Ray Tracing



How Can We Add Shadows?

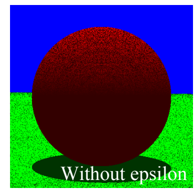
```
color = ambient*hit->getMaterial()->getDiffuseColor()
for every light
   Ray ray2(hitPoint, directionToLight)
   Hit hit2(distanceToLight, NULL, NULL)
   For every object
      object->intersect(ray2, hit2, 0)
   if (hit2->getT() = distanceToLight)
      color += hit->getMaterial()->Shade
               (ray, hit, directionToLight, lightColor)
return color
```

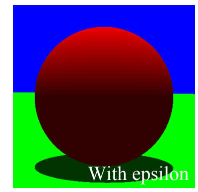




Problem: Self-Shadowing

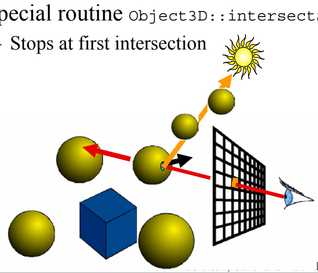
```
color = ambient*hit->getMaterial()->getDiffuseColor()
for every light
   Ray ray2(hitPoint, directionToLight)
   Hit hit2(distanceToLight, NULL, NULL)
   For every object
      object->intersect(ray2, hit2, epsilon)
   if (hit2->getT() = distanceToLight)
      color += hit->getMaterial()->Shade
               (ray, hit, directionToLight, lightColor)
return color
```





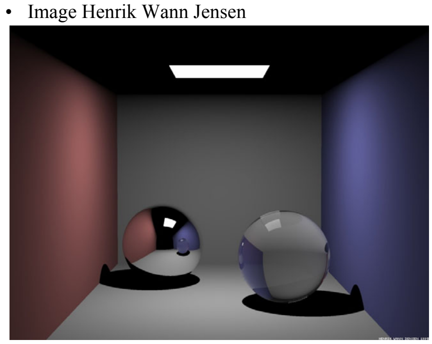
Shadow Optimization

- Shadow rays are special: Can we accelerate our code?
- We only want to know whether there is an intersection, *not* which one is closest
- Special routine `Object3D::intersectShadowRay()`
  - Stops at first intersection



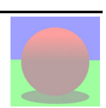
Questions?

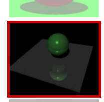
- Image Henrik Wann Jensen

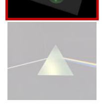


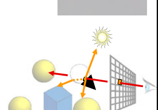
Overview of Today

- Shadows
- Reflection
- Refraction
- Recursive Ray Tracing



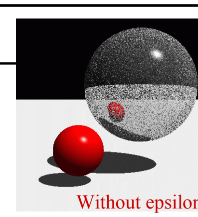


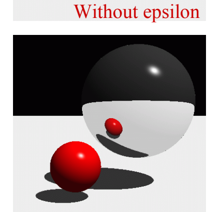


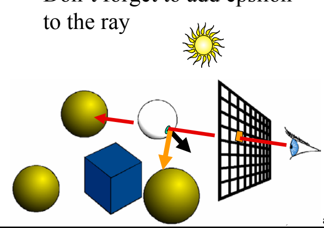


Mirror Reflection

- Cast ray symmetric with respect to the normal
- Multiply by reflection coefficient (color)
- Don't forget to add epsilon to the ray

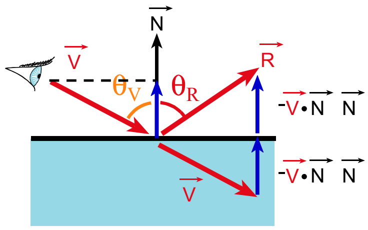






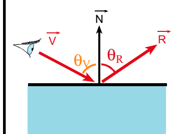
Reflection

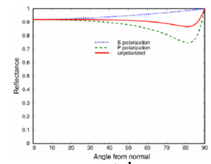
- Reflection angle = view angle
- **R** = **V** – 2 (**V** · **N**) **N**



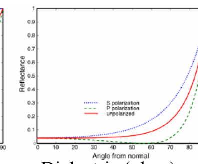
Amount of Reflection

- Traditional ray tracing (hack)
  – Constant `reflectionColor`
- More realistic:
  – Fresnel reflection term (more reflection at grazing angle)
  – Schlick's approximation: $R(\theta)=R_0+(1-R_0)(1-\cos\theta)^5$





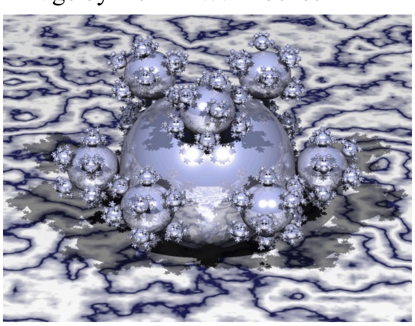
metal



Dielectric (glass)

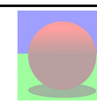
Questions?

- Image by Henrik Wann Jensen



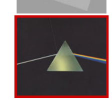
Overview of Today

- Shadows

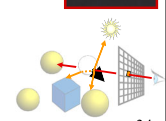


- Reflection

- Refraction

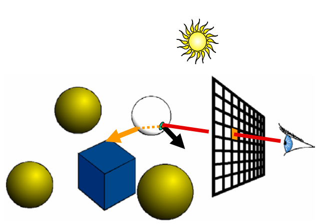


- Recursive Ray Tracing

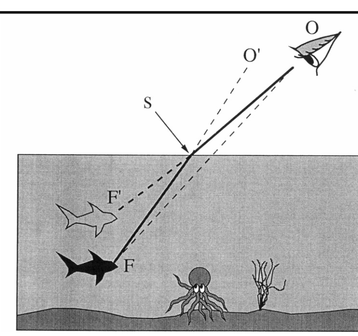


Transparency

- Cast ray in refracted direction
- Multiply by transparency coefficient (color)

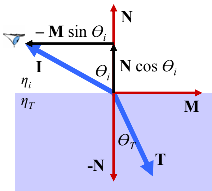


Qualitative Refraction



From "Color and Light in Nature" by Lynch and Livingston

Refraction



$\mathbf{I} = \mathbf{N} \cos \Theta_i - \mathbf{M} \sin \Theta_i$

$\mathbf{M} = (\mathbf{N} \cos \Theta_i - \mathbf{I}) / \sin \Theta_i$

$\mathbf{T} = -\mathbf{N} \cos \Theta_T + \mathbf{M} \sin \Theta_T$

$= -\mathbf{N} \cos \Theta_T + (\mathbf{N} \cos \Theta_i - \mathbf{I}) \sin \Theta_T / \sin \Theta_i$

$= -\mathbf{N} \cos \Theta_T + (\mathbf{N} \cos \Theta_i - \mathbf{I}) \eta_r$

$= [\, \eta_r \cos \Theta_i - \cos \Theta_T \,] \mathbf{N} - \eta_r \mathbf{I}$

$= [\, \eta_r \cos \Theta_i - \sqrt{1 - \sin^2 \Theta_T} \,] \mathbf{N} - \eta_r \mathbf{I}$

$= [\, \eta_r \cos \Theta_i - \sqrt{1 - \eta_r^2 \sin^2 \Theta_i} \,] \mathbf{N} - \eta_r \mathbf{I}$

$= [\, \eta_r \cos \Theta_i - \sqrt{1 - \eta_r^2 (1 - \cos^2 \Theta_i)} \,] \mathbf{N} - \eta_r \mathbf{I}$

$= [\, \eta_r (\mathbf{N} \cdot \mathbf{I}) - \sqrt{1 - \eta_r^2 (1 - (\mathbf{N} \cdot \mathbf{I})^2)} \,] \mathbf{N} - \eta_r \mathbf{I}$

Snell-Descartes Law:

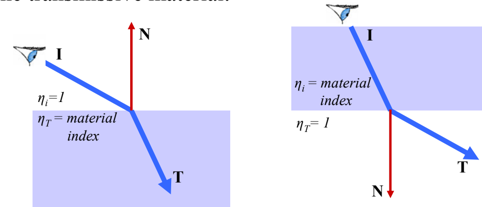
$\eta_i \sin \Theta_i = \eta_T \sin \Theta_T$

$$\frac{\sin \Theta_T}{\sin \Theta_i} = \frac{\eta_i}{\eta_T} = \eta_r$$

- **Total internal reflection when the square root is imaginary**
- **Don't forget to normalize!**

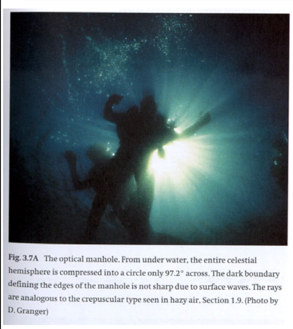
Refraction & the Sidedness of Objects

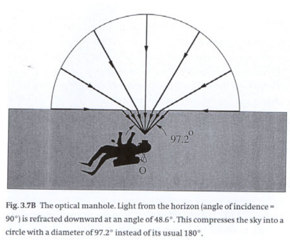
- Make sure you know whether you're entering or leaving the transmissive material:



- Note: We won't ask you to trace rays through intersecting transparent objects

Total Internal Reflection

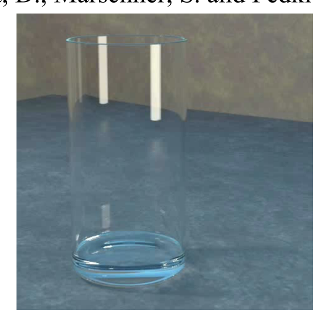




From "Color and Light in Nature" by Lynch and Livingston

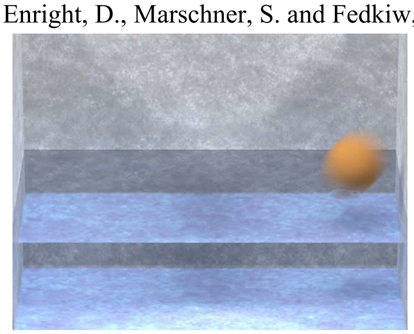
Cool Refraction Demo

- Enright, D., Marschner, S. and Fedkiw, R.,



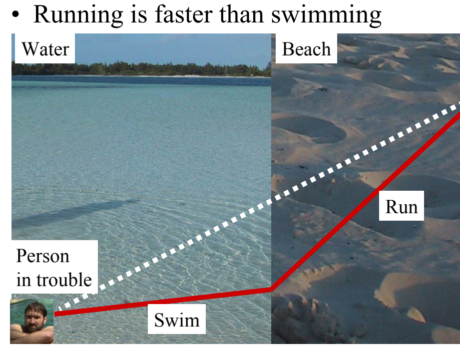
Cool Refraction Demo

- Enright, D., Marschner, S. and Fedkiw, R.,



Refraction and the Lifeguard Problem

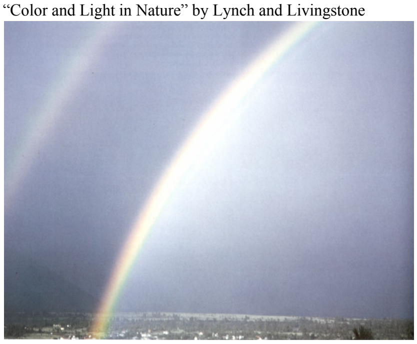
- Running is faster than swimming



Lifeguard

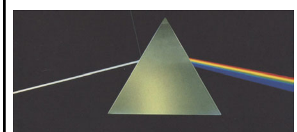
How does a Rainbow Work?

- From "Color and Light in Nature" by Lynch and Livingstone



Wavelength

- Refraction is wavelength-dependent
  - Refraction increases as the wavelength of light decreases
  - violet and blue experience more bending than orange and red
- Newton's experiment
- Usually ignored in graphics

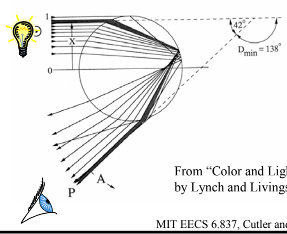


Pink Floyd, *The Dark Side of the Moon*

Pittoni, 1725, Allegory to Newton

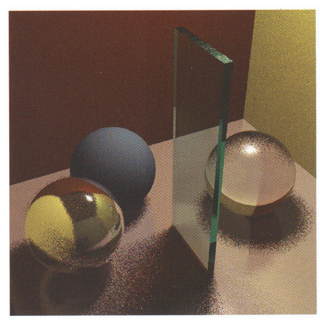
Rainbow

- Refraction depends on wavelength
- Rainbow is caused by refraction + internal reflection + refraction
- Maximum for angle around 42 degrees



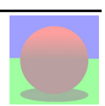
From "Color and Light in Nature" by Lynch and Livingstone

Questions?

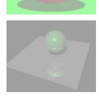


Overview of Today

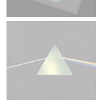
- Shadows



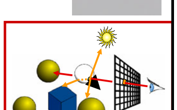
- Reflection



- Refraction



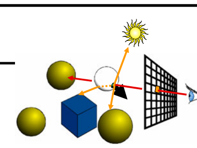
- Recursive Ray Tracing



Recap: Ray Tracing

```
trace ray
  Intersect all objects
  color = ambient term
  For every light
    cast shadow ray
    color += local shading term
  If mirror
    color += color_refl *
          trace reflected ray
  If transparent
    color += color_trans *
          trace transmitted ray
```

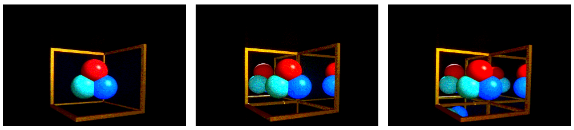
- ***Does it ever end?***



Stopping criteria:

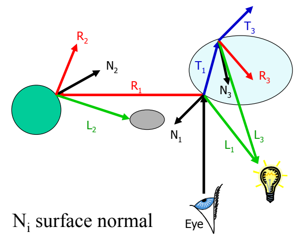
- Recursion depth
  - Stop after a number of bounces
- Ray contribution
  - Stop if reflected / transmitted contribution becomes too small

Recursion For Reflection

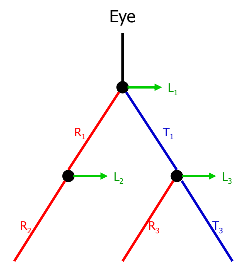


0 recursion　　1 recursion　　2 recursions

The Ray Tree



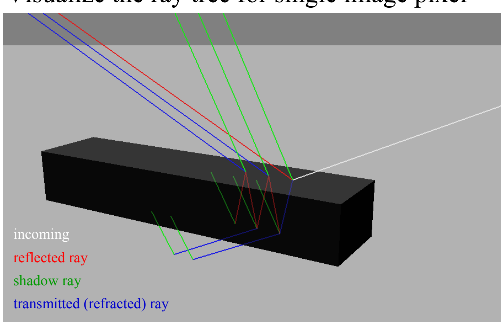
N_i surface normal
R_i reflected ray
L_i shadow ray
T_i transmitted (refracted) ray



Complexity?

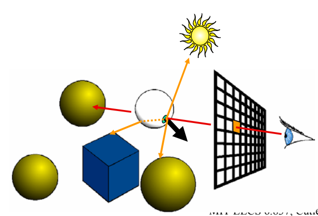
Ray Debugging (Assignment 4)

- Visualize the ray tree for single image pixel



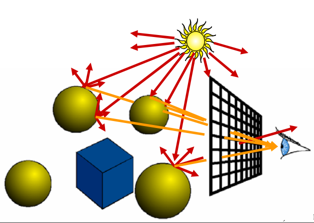
Does Ray Tracing Simulate Physics?

- Photons go from the light to the eye, not the other way
- What we do is *backward ray tracing*



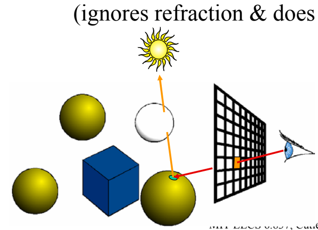
Forward Ray Tracing

- Start from the light source
  - But low probability to reach the eye
- What can we do about it?
  - Always send a ray to the eye…. still not efficient



Does Ray Tracing Simulate Physics?

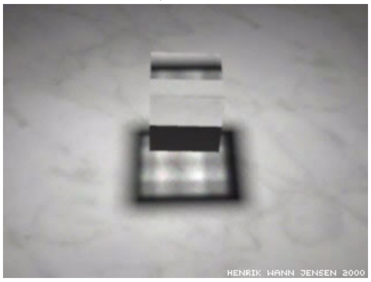
- Ray Tracing is full of dirty tricks
- For example, shadows of transparent objects:
  - opaque?
  - multiply by transparency color? (ignores refraction & does not produce caustics)



Correct Transparent Shadow

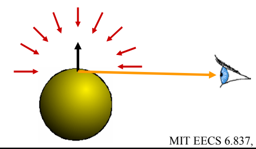
Animation by Henrik Wann Jensen

Using advanced refraction technique
(refraction for illumination is usually not handled that well)



The Rendering Equation

- Clean mathematical framework for light-transport simulation
- We'll see this later
- At each point, outgoing light in one direction is the integral of incoming light in all directions multiplied by reflectance property



A Look Ahead

- Assignment 2
  – Transformations & More Primitives
- Assignment 3
  – OpenGL Previsualization & Phong Shading
- Assignment 4
  – Ray Tracing (Shadows, Reflections, Refractions)

Next Time

- Modeling Transformations
- Illumination (Shading)
- Viewing Transformation (Perspective / Orthographic)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Visibility / Display

Introduction to the Graphics Pipeline