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
- Transformations in Ray Tracing
  - Transforming the ray
    Remember: points & directions transform differently!
  - Normalizing direction & what to do with t
  - Normal transformation
    \[ n_{WS}^T = n_{OS} (M^{-T}) \]

Last Time: Local Illumination
- BRDF – Bidirectional Reflectance Distribution Function
- Phong Model- Sum of 3 components:
  - Diffuse Shading
  - Specular Highlight
  - Ambient Term
  \[ L_a = k_a + (k_d (n \cdot l) + k_r (v \cdot r))^L \]

Last Week: Transformations
- Linear, affine and projective transforms
- Homogeneous coordinates
- Matrix notation
- Transformation composition is not commutative
Phong Examples

- Shininess coefficient controls the “spread” of the specular highlight

<table>
<thead>
<tr>
<th>Shininess Coefficient</th>
<th>Specular Highlight</th>
</tr>
</thead>
<tbody>
<tr>
<td>q = 1</td>
<td>Diffuse only</td>
</tr>
<tr>
<td>q = 2</td>
<td>Diffuse + Specular</td>
</tr>
<tr>
<td>q = 4</td>
<td>Diffuse + Specular</td>
</tr>
<tr>
<td>q = 8</td>
<td>Diffuse + Specular</td>
</tr>
<tr>
<td>q = 16</td>
<td>Diffuse + Specular</td>
</tr>
<tr>
<td>q = 32</td>
<td>Diffuse + Specular</td>
</tr>
<tr>
<td>q = 64</td>
<td>Diffuse + Specular</td>
</tr>
<tr>
<td>q = 128</td>
<td>Diffuse + Specular</td>
</tr>
</tbody>
</table>

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)

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

Additional Phong Clamping Term

- Surfaces facing away from the light should not be lit (if $\mathbf{N} \cdot \mathbf{L} < 0$)

\[
\frac{L_o}{L_i} = \max(\mathbf{N} \cdot \mathbf{L}, 0)
\]

BRDFs in the Movie Industry

- 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

- For the Matrix movies

BRDFs in the Movie Industry

Gonioreflectometer  Measured BRDF

Test rendering  Measured BRDF

BRDF Models

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

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

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?
$O(n \times m)$
$n =$ number of objects, $m =$ number of pixels

Ray Casting with Phong Shading
When you've found the closest intersection:
  $\text{color} = \text{ambient} \times \text{hit->getMaterial()->getDiffuseColor()}$
  for every light
  $\text{color} += \text{hit->getMaterial()->Shade}((\text{ray}, \text{hit}, \text{directionToLight}, \text{lightColor}))$

return $\text{color}$

Complexity?
$O(n \times m \times l)$
$l =$ 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?

```cpp
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

```cpp
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

Without epsilon

With epsilon
Reflection

- Reflection angle = view angle
- \( \mathbf{R} = \mathbf{V} - 2(\mathbf{V} \cdot \mathbf{N})\mathbf{N} \)

\[ \mathbf{V} \]

\[ \mathbf{R} \]

\[ \mathbf{N} \]

\[ \theta \]

\[ R \]

\[ V \]

\[ \cdot \]

\[ N \]

\[ V \]

\[ N \]

\[ V \]

\[ N \]

Amount of Reflection

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

\[ \theta \]

\[ R \]

\[ V \]

\[ \cdot \]

\[ N \]

\[ V \]

\[ \cdot \]

\[ N \]

\[ V \]

\[ \cdot \]

\[ N \]

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

\[ I = N \cos \theta_i - M \sin \theta_i \]
\[ M = (N \cos \theta_i - I) / \sin \theta_i \]
\[ T = -N \cos \theta_i + M \sin \theta_i \]
\[ = -N \cos \theta_i + (N \cos \theta_i - I) \sin \theta_i / \sin \theta_i \]
\[ = -N \cos \theta_i + (N \cos \theta_i - I) \eta_i \]
\[ = \frac{\eta_i \cos \theta_i - \cos \theta_T}{\eta_i} \cdot \frac{N - \eta_i \eta_T}{N - \eta_i \eta_i} \]
\[ = \frac{\eta_i \cos \theta_i - \cos \theta_T}{\eta_i} \cdot \frac{N - \eta_i \eta_T}{N - \eta_i \eta_i} \]
\[ = \eta_i \frac{(N \cdot I) - \sqrt{1 - \eta_i^2 \sin^2 \theta_T}}{N - \eta_i \eta_i} \]
\[ = \eta_i \frac{(N \cdot I) - \sqrt{1 - \eta_i^2 (1 - \cos^2 \theta_i)}{N - \eta_i \eta_i} \]
\[ = \eta_i \frac{(N \cdot I) - \sqrt{1 - \eta_i^2 (1 - \cos^2 \theta_i)}}{N - \eta_i \eta_i} \]

Snell-Descartes Law:
\[ \eta_i \sin \theta_i = \eta_T \sin \theta_T \]

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

From “Color and Light in Nature” by Lynch and Livingston

Cool Refraction Demo

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

Refraction and the Lifeguard Problem

- Running is faster than swimming
How does a Rainbow Work?

- From “Color and Light in Nature” by Lynch and Livingstone

Rainbow

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

Questions?

Overview of Today

- Shadows
- Reflection
- Refraction
- Recursive Ray Tracing

Recap: Ray Tracing

- 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

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 Pre-Visualization & Phong Shading
- Assignment 4
  - Ray Tracing (Shadows, Reflections, Refractions)

Next Time

Introduction to the Graphics Pipeline

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