Lighting and Reflectance

COS 426, Spring 2018
Princeton University
R2Image *RayCast(R3Scene *scene, int width, int height)
{
    R2Image *image = new R2Image(width, height);
    for (int i = 0; i < width; i++) {
        for (int j = 0; j < height; j++) {
            R3Ray ray = ConstructRayThroughPixel(scene->camera, i, j);
            R3Rgb radiance = ComputeRadiance(scene, &ray);
            image->SetPixel(i, j, radiance);
        }
    }
    return image;
}
Ray Casting

R3Rgb ComputeRadiance(R3Scene *scene, R3Ray *ray)
{
    R3Intersection intersection = ComputeIntersection(scene, ray);
    return ComputeRadiance(scene, ray, intersection);
}
Illumination

- How do we compute radiance for a sample ray once we know what it hits?

ComputeRadiance(scene, ray, intersection)

Angel Figure 6.2
Goal

- Must derive computer models for ...
  - Emission at light sources
  - Scattering at surfaces
  - Reception at the camera

- Desirable features ...
  - Concise
  - Efficient to compute
  - “Accurate”
Overview

• Direct Illumination
  • Emission at light sources
  • Scattering at surfaces

• Global illumination
  • Shadows
  • Refractions
  • Inter-object reflections
Emission at Light Sources

- $I_L(x,y,z,\theta,\phi,\lambda)$ ...
  - describes the intensity of energy,
  - leaving a light source, ...
  - arriving at location $(x,y,z)$, ...
  - from direction $(\theta,\phi)$, ...
  - with wavelength $\lambda$
Empirical Models

• Ideally measure irradiant energy for “all” situations
  • Too much storage
  • Difficult in practice

\[ x, y, z, \theta, \phi, \lambda \]
OpenGL Light Source Models

• Simple mathematical models:
  • Point light
  • Directional light
  • Spot light
Point Light Source

- Models omni-directional point source
  - intensity \( (I_0) \),
  - position \( (p_x, p_y, p_z) \),
  - coefficients \( (c_a, l_a, q_a) \) for attenuation with distance \( (d) \)

\[
I_L = \frac{I_0}{c_a + l_a d + q_a d^2}
\]
Point Light Source

\[ I_L = \frac{I_0}{c_a + l_a d + q_a d^2} \]

- Physically-based: “inverse square law”
  - \( c_a = l_a = 0 \)
- Use \( c_a \) and \( l_a \neq 0 \) for non-physical effects
  - Better control of the look (artistic)
Directional Light Source

- Models point light source at infinity
  - intensity \( (I_0) \),
  - direction \( (d_x, d_y, d_z) \)

\[
I_L = I_0
\]

No attenuation with distance
Spot Light Source

- Models point light source with direction
  - intensity \( (I_0) \),
  - position \( (p_x, p_y, p_z) \),
  - direction \( (d_x, d_y, d_z) \)
  - attenuation with distance
  - falloff (sd), and cutoff (sc)

\[ \Theta = \cos^{-1}(L \cdot D) \]

\[ I_L = \begin{cases} 
\frac{I_0 (\cos \Theta)^{sd}}{c_a + l_a d + q_a d^2} & \text{if } \Theta \leq sc, \\
0 & \text{otherwise}
\end{cases} \]
Overview

• Direct Illumination
  • Emission at light sources
  • Scattering at surfaces

• Global illumination
  • Shadows
  • Refractions
  • Inter-object reflections
**Scattering at Surfaces**

**Bidirectional Reflectance Distribution Function**

\[ f_r(\theta_i, \phi_i, \theta_o, \phi_o, \lambda) \] ...

- describes the aggregate fraction of incident energy,
- arriving from direction \((\theta_i, \phi_i)\), ...
- leaving in direction \((\theta_o, \phi_o)\), ...
- with wavelength \(\lambda\)
Empirical Models

Ideally measure BRDF for “all” combinations of angles: $\theta_i, \phi_i, \theta_o, \phi_o$

- Difficult in practice
- Too much storage
Parametric Models

Approximate BRDF with simple parametric function that is fast to compute.

- Phong [75]
- Blinn-Phong [77]
- Cook-Torrance [81]
- He et al. [91]
- Ward [92]
- Lafortune et al. [97]
- Ashikhmin et al. [00]
- etc.
OpenGL Reflectance Model

• Simple analytic model:
  • diffuse reflection +
  • specular reflection +
  • emission +
  • “ambient”

Based on model proposed by Phong
OpenGL Reflectance Model

- Simple analytic model:
  - diffuse reflection +
  - specular reflection +
  - emission +
  - “ambient”

Based on model proposed by Phong
Diffuse Reflection

• Assume surface reflects equally in all directions
  • Examples: chalk, clay
Diffuse Reflection

• What is brightness of surface?
  • Depends on angle of incident light
Diffuse Reflection

- What is brightness of surface?
  - Depends on angle of incident light

\[ dL = dA \cos \Theta \]
Diffuse Reflection

• Lambertian model
  • cosine law (dot product)

\[ I_D = K_D (N \cdot L) I_L \]
OpenGL Reflectance Model

- Simple analytic model:
  - diffuse reflection +
  - specular reflection +
  - emission +
  - “ambient”
Specular Reflection

- Reflection is strongest near mirror angle
  - Examples: mirrors, metals
Specular Reflection

How much light is seen?

Depends on:

- angle of incident light
- angle to viewer
Specular Reflection

- Phong Model
  - \((\cos \alpha)^n\) This is a (vaguely physically-motivated) hack!

\[
I_S = K_S (V \cdot R)^n I_L
\]
OpenGL Reflectance Model

- Simple analytic model:
  - diffuse reflection +
  - specular reflection +
  - emission +
  - “ambient”
Emission

Represents light emanating directly from surface

• Note: does not automatically act as light source!
  Does not affect other surfaces in scene!
OpenGL Reflectance Model

- Simple analytic model:
  - diffuse reflection +
  - specular reflection +
  - emission +
  - “ambient”
Ambient Term

Represents reflection of all indirect illumination

This is a hack (avoids complexity of global illumination)!
OpenGL Reflectance Model

• Simple analytic model:
  • diffuse reflection +
  • specular reflection +
  • emission +
  • “ambient”
OpenGL Reflectance Model

• Simple analytic model:
  • diffuse reflection +
  • specular reflection +
  • emission +
  • “ambient”
OpenGL Reflectance Model

Sum diffuse, specular, emission, and ambient

<table>
<thead>
<tr>
<th>Phong</th>
<th>$\rho_{ambient}$</th>
<th>$\rho_{diffuse}$</th>
<th>$\rho_{specular}$</th>
<th>$\rho_{total}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_i = 60^\circ$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi_i = 25^\circ$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi_i = 0^\circ$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Leonard McMillan, MIT
OpenGL Reflectance Model

Good model for plastic surfaces, …
Direct Illumination Calculation

Single light source:

\[ I = I_E + K_A I_{AL} + K_D (N \cdot L) I_L + K_S (V \cdot R)^n I_L \]
Direct Illumination Calculation

Multiple light sources:

\[ I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) I_L \]

\[ \sum_i \]  

Note: all of the \( K \) and \( I \) are RGB colors.
Example from production

This scene had 400 virtual lights (~100 params)
Overview

- Direct Illumination
  - Emission at light sources
  - Scattering at surfaces

- Global illumination
  - Shadows
  - Transmissions
  - Inter-object reflections

Global Illumination
Global Illumination

Greg Ward
Ray Casting (last lecture)

Trace primary rays from camera

- Direct illumination from unblocked lights only

\[ I = I_E + K_A I_{AL} + \sum_{L} \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) I_L \]
Shadow term tells if light sources are blocked

- Cast ray towards each light source
- $S_L = 0$ if ray is blocked, $S_L = 1$ otherwise

\[
I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L
\]
Recursive Ray Tracing

Also trace secondary rays from hit surfaces

- Mirror reflection and transparency

\[
I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L + K_S I_R + K_T I_T
\]
Mirror reflections

Trace secondary ray in mirror direction

- Evaluate radiance along secondary ray and include it into illumination model

\[ I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L + K_S I_R + K_T I_T \]
Transparency

Trace secondary ray in direction of refraction

- Evaluate radiance along secondary ray and include it into illumination model

\[ I = I_E + K_A I_{AL} + \sum_{L} \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L + K_S I_R + K_T I_T \]
Transparency

Transparency coefficient is fraction transmitted

- $K_T = 1$ for translucent object, $K_T = 0$ for opaque
- $0 < K_T < 1$ for object that is semi-translucent

\[ I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L + K_S I_R + K_T I_T \]
Refractive Transparency

For thin surfaces, can ignore change in direction

- Assume light travels straight through surface

\[ T \approx -L \]
Refractive Transparency

For solid objects, apply Snell’s law:

\[ \eta_r \sin \Theta_r = \eta_i \sin \Theta_i \]
Recursive Ray Tracing

Ray tree represents illumination computation

\[ I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L + K_S I_R + K_T I_T \]
Recursive Ray Tracing

Ray tree represents illumination computation

\[ I = I_E + K_A I_{AL} + \sum_L \left( K_D (N \cdot L_i) + K_S (V \cdot R_i)^n \right) S_L I_L + K_S I_R + K_T I_T \]
Recursive Ray Tracing

ComputeRadiance is called recursively

```c
R3Rgb ComputeRadiance(R3Scene *scene, R3Ray *ray, R3Intersection& hit)
{
    R3Ray specular_ray = SpecularRay(ray, hit);
    R3Ray refractive_ray = RefractiveRay(ray, hit);
    R3Rgb radiance = Phong(scene, ray, hit) +
        Ks * ComputeRadiance(scene, specular_ray) +
        Kt * ComputeRadiance(scene, refractive_ray);
    return radiance;
}
```
Example

Turner Whitted, 1980 (74 minutes)
Summary

• Ray casting (direct Illumination)
  • Usually use simple analytic approximations for light source emission and surface reflectance

• Recursive ray tracing (global illumination)
  • Incorporate shadows, mirror reflections, and pure refractions

All of this is an approximation so that it is practical to compute

More on global illumination after next week!
Illumination Terminology

• Radiant power [flux] (Φ)
  • Rate at which light energy is transmitted (in Watts).

• Radiant Intensity (I)
  • Power radiated onto a unit solid angle in direction (in Watts/sr)
    » e.g.: energy distribution of a light source (inverse square law)

• Radiance (L)
  • Radiant intensity per unit projected surface area (in Watts/m²sr)
    » e.g.: light carried by a single ray (no inverse square law)

• Irradiance (E)
  • Incident flux density on a locally planar area (in Watts/m²)
    » e.g.: light hitting a surface at a point

• Radiosity (B)
  • Exitant flux density from a locally planar area (in Watts/m²)