Global illumination and Radiosity

Arjan Kok
a.j.f.kok@tue.nl

Global illumination

- Accounts for all light in a scene
  - Direct illumination
  - Indirect illumination

Light paths

- Light path notation
  - L: light source
  - D: diffuse reflection
  - S: specular reflection
  - E: eye point

- Complete solution: \( L(D|S)*E \)

Missing light paths

- Local illumination model (Phong):
  - \( L(D|S)E \)
  - Ray tracing:
    - \( LDS^*E \)

- Missing:
  - Diffuse interreflection: \( D^* \)
  - Diffuse/specular interreflection: \( (DS)^* \)
  - These terms approximated with ambient light
Rendering equation

- Theoretical basis for light transport
- Describes the light being transferred from one point to another in terms of
  - Light emitted from first point to second point
  - Light being emitted from all points that reaches first point and is reflected to second point
- Rendering methods solve (part of) this equation

Radiometry

- Radiant energy
  - Energy of photons
- Radiant power (flux)
  - Radiant energy per unit time
  - Flow of energy
  - \( \Phi \) (Watt = Joule/s)

Radiometry

- Radiance
  - Radiant power leaving, passing through, or arriving at an element of the surface surrounding a point, per solid angle and per unit projected area
  - Number of photons per time at a small area from a particular direction
  - \( L = \frac{d\Phi}{dA \cos \theta d\Omega} \) (W/steradian m^2)

Radiometry

- Radiance properties
  - Radiance doesn’t change with distance
    - Therefore it’s the quantity we want to measure in e.g. a ray tracer
    - Radiance proportional to what a sensor (camera, eye) measures
    - Therefore it’s what we want to output

Radiometry

- Irradiance
  - Density of radiant power incident onto surface
  - Incoming radiant power per unit surface area
  - \( E = \frac{d\Phi}{dA} \) (Watt/m^2)
  - \( E_i = L_i \cos \theta \cos \varphi \)

Radiometry

- Radiant exitance
  - Density of radiant power leaving surface
  - Outgoing radiant power per unit surface area
  - Also know as radiosity
  - \( E = \frac{d\Phi}{dA} \) (Watt/m^2)
Reflection

- Reflectance
  - Ratio of differential radiance reflected in a given direction to the differential power density incident from a given direction
  - Fraction of the incident power that is reflected
  - Described by bidirectional reflectance distribution function (BRDF)

BRDF

- Properties
  - Non-negativity:
    - \( f_r(\omega_i, \omega_o) \geq 0 \)
  - Energy conservation:
    - \( \int_{\Omega} f_r(\omega_i, \omega_o) d\Omega_i \leq 1 \)
  - Reciprocity:
    - \( f_r(\omega_i, \omega_o) = f_r(\omega_o, \omega_i) \)

- Ideal diffuse reflectance
  - Reflects equally in all directions
  - BRDF is constant

- Ideal specular reflection
  - Reflection takes place in a plane perpendicular to surface
  - Angle of reflectance = angle of incidence
    - \( \theta_r = \theta_i \)
    - \( \phi_r = \phi_i \pm \pi \)
    - \( L_r(\theta_r, \phi_r) = L_i(\theta_i, \phi_i \pm \pi) \)

  - BRDF modeled by delta functions
    - \( f_r = \frac{\delta(\cos \theta_r - \cos \theta_i) \delta(\phi_r - (\phi_i \pm \pi))}{\cos \theta_i} \)

- Non ideal reflectance
  - Difficult to model
Reflected radiance

- Reflected radiance can now be defined as:
  \[ L_r(\omega) = \int_\Omega f(\omega, \omega_\theta) dE(\omega_\theta) = \int_\Omega f(\omega, \omega_\theta) L_r(\omega_\theta) \cos(\theta_\omega) d\omega_\theta \]
- Surface integration in place of hemispherical integration
- \( L_r(x, \omega_\theta) = \int_\Omega f(x, \omega_\theta, \omega_\theta_\omega) L_r(x, \omega_\theta_\omega) / G(x, x') V(x, x') dA' \)
- When
  \[ G(x, x') = \frac{\cos(\theta_{x'}) \cos(\theta_x)}{|x - x'|} \]
  \( V \) is visibility term

Rendering equation

- \( L(x, \omega_\theta) = L_e(x, \omega_\theta) + L_r(x, \omega_\theta) \)
- \( L_e(x, \omega_\theta) \) is the emitted radiance from point \( x \) on the surface in the given direction \( \omega_\theta \)
- \( L_r(x, \omega_\theta) \) is non-zero if \( x \) is emissive (light source)

Rendering equation

- For \( x' \), compute \( L(x', \omega_\theta_\omega) \), the radiance at point \( x' \) in the direction \( \omega_\theta_\omega \) (from \( x' \) to \( x \))

Sum the contribution from all other surfaces in the scene
**Rendering equation**

\[
L(x, \omega_i) = L_e(x, \omega_i) + \frac{1}{\pi} \int f_s(x, \omega_i, \omega_o) L(x', \omega_o) G(x, x') V(x, x') dA'
\]

For each \( x' \), compute \( V(x, x') \), the visibility between \( x \) and \( x' \)

1 when surfaces are unobstructed along \( \omega_i \), 0 otherwise

---

**Radiosity**

- Surfaces are assumed to be perfectly diffuse
  - Incident light is reflected in all directions with equal intensity
  - Radiosity method computes diffuse interreflection of objects
  - Diffuse interreflection is not longer approximated by ambient light

---

**Indirect diffuse illumination**

\[
L(x, \omega_i) = L_e(x, \omega_i) + \frac{1}{\pi} \int f_s(x, \omega_i, \omega_o) L(x', \omega_o) G(x, x') V(x, x') dA'
\]

- Assume all surfaces are diffuse
- BRDF is a constant: \( f_s = \rho \)

\[
B = \int L \cos \theta d\omega \quad \text{so} \quad B = \pi L
\]

\[
B(x) = E(x) \frac{D(x)}{\pi} \int B(x') G(x, x') V(x, x') dA'
\]

- \( B(x) \) is radiosity at point \( x \) (Watts / m²)
Radiosity equation

- No analytical solution

\[ B(x) = E(x) + \rho(x) \int \frac{G(x, x')V(x, x')}{\pi} dA' \]

Form factor

- Form factor \( F_{ij} \)
- Fraction of energy leaving patch \( j \) that arrives at patch \( i \)
- Represents the \( G(x, x')V(x, x') \) factors in rendering equation

\[ F_i = \frac{1}{A_i} \int_{A_j} \cos \theta \cos \theta' V_i dA dA' \]

\( V_i = 1 \) if \( dA_j \) visible from \( dA_i \)
\( 0 \) if not visible

Radiosity equation solution

- Solve for all patches the radiosity equation:

\[ B_i = E_i + \rho \sum_j F_{ij} B_j \]

- \( n \) equations with \( n \) unknown \( B_i \) values can be written in matrix form

Radiosity matrix

\[
\begin{bmatrix}
1 - \rho F_{i1} & \cdots & \cdots & -\rho F_{in} & B_i \\
-\rho F_{1i} & 1 - \rho F_{i2} & \cdots & \cdots & B_i \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
-\rho F_{ni} & \cdots & \cdots & 1 - \rho F_{nn} & B_i \\
\end{bmatrix}
\begin{bmatrix}
E_1 \\
E_2 \\
\vdots \\
E_n \\
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
\vdots \\
0 \\
\end{bmatrix}
\]

Radiosity results

- Result of radiosity: Constant radiosity value \( B_i \) for each patch
- Radiosity is viewpoint independent
  - One solution can be used to generate several images
**Radiosity results**

- B_i radiosity values are constant over the extend of a patch
- Renderer needs radiosity values at vertices (to allow interpolation)
  - Average the radiosities of patches that contribute to the vertex
  - Vertices on the edge of a surface are assigned values by extrapolation

---

**Calculating form factors**

- Form factor $F_{ij}$
  - Fraction of energy leaving patch j that arrives at patch i
  - Accounts for
    - Geometry (size, orientation, position)
    - Visibility (occluders)

$$F_{ij} = \frac{1}{A_i A_j} \int_{A_i} \cos \theta \cos \theta' \frac{V_{dA_i} dA_i}{\pi r^2}$$

---

**Analytic solutions**

- Only feasible for VERY simple scenes
- Visibility is hard to compute analytically!

---

**The Nusselt Analog**

Form factor $F_{dA_i A_j}$
The Nusselt Analog

1. Project \( A_j \) along its normal:
   \[ A_j \cos \theta \]
2. Project result on sphere:
   \[ A_j \cos \theta \pi r^2 \]
3. Project result on unit circle:
   \[ A_j \cos \theta \cos \theta \pi r^2 \]
4. Divide by unit circle area:
5. Integrate for all points on \( A_j \):
   \[ \frac{\cos \theta \cos \theta}{\pi r^2} V_j dA_j \]
6. Divide by unit circle area:
7. Integrate for all points on \( A_j \):
   \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
8. Integrate for all points on \( A_j \):
   \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
9. Integrate for all points on \( A_j \):
   \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
10. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
11. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
12. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
13. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
14. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
15. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
16. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
17. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
18. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
19. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
20. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
21. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
22. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
23. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
24. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
25. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
26. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
27. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
28. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
29. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
30. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
31. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
32. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
33. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
34. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
35. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
36. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
37. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
38. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
39. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
40. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
41. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
42. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
43. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
44. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
45. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
46. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
47. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]
48. Integrate for all points on \( A_j \):
    \[ \frac{\cos \theta \cos \theta}{\pi r^2} \]

Hemicube method

- A hemicube is constructed around the center of each patch
- Faces of the hemicube are divided into “pixels”
- Each patch is projected (rasterized) onto the faces of the hemicube
- Each pixel stores its pre-computed form factor
- The formfactor for a particular patch is the sum of the pixel it overlaps
- Patch occlusions are handled similar to z-buffer rasterization

Ray casting

- Subdivide \( A_j \) into small pieces \( dA_j \)
- For all \( dA_j \)
  - cast ray \( dA_j \) to determine \( V \)
  - if visible
    - compute \( F_{dA_j dA_j} \)
    - \( \frac{\cos \theta \cos \theta}{\pi r^2} \)
    - sum up \( F_{dA_j dA_j} \)
    - Result now \( F_{dA_j dA_j} \)

Form factor summary

- Several ways to find form factors
- Hemicube was original method
  + Hardware acceleration
  + Gives \( F_{dA_j dA_j} \) for all \( j \) in one pass
  - Aliasing
- Area sampling methods now preferred
  - Slower than hemicube
  - As accurate as desired since adaptive

Radiosity example

Radiosity stages

Input geometry → Form factor calculation → > 90%
Reflectance properties → Solve the radiosity matrix → < 10%
Camera position and orientation → Visualization (rendering) → ~ 10%
image
**Progressive Radiosity**

- Goal: provide frequent and timely updates to the user during computation
- Key idea: update the entire image at every iteration, rather than a single patch
- Instead of summing light received by one patch, distribute the radiance of the patch with the most undistributed radiance

**Progressive Radiosity**

- Shoot from the element having the most energy
- Compute the form factors as you shoot
- Update all of the radiosities
- Display the results every iteration

**Progressive Radiosity**

- Shoot all energy from element \( i \), and compute contribution to radiosities of all elements \( j \) in the scene
- Initially all radiosities of all elements are 0, except radiosities of light sources

**Progressive Radiosity**

\[
\begin{align*}
B_i &= E_i + \rho_i \sum_j B_j F_{ij} \\
\text{Contribution } B_i \text{ to } B_j &= \rho_i B_i F_{ij} A_j / A_i \\
\text{Shoot all energy from element } i, \text{ and compute contribution to radiosities of all elements } j \text{ in the scene} \\
\text{Initially all radiosities of all elements are 0, except radiosities of light sources}
\end{align*}
\]

**Progressive Radiosity**

\[
\begin{align*}
\text{Initialization} \\
\{ & B_i = E_i; \\
\} & \Delta B_i = E_i
\end{align*}
\]

**Progressive Radiosity**

\[
\begin{align*}
do & \text{ find element } i \text{ with most "unshot" energy } \Delta B A_i; \\
& \text{ for all other elements } j \text{ do} \\
& \text{ compute form-factor } F_{ji} \\
& \Delta \text{Rad} = \rho_i \Delta B_i F_{ji} (A_j / A_i) \\
& \Delta B_j = \Delta B_j + \Delta \text{Rad} \\
& B_j = B_j + \Delta \text{Rad}; \\
& \Delta B_i = 0; \\
& \text{ until convergence reached}
\end{align*}
\]
Advantages

- You see progress
- Ability to stop before process completely converged
- You don’t store an $O(n^2)$ matrix of form factors.
- When the process starts out, all of the unshot energy is at lights.
- As the process unfolds, the energy is spread around and the residuals become more even.

Ambient term

- To make the images look better (less dark) at the beginning, use an ambient term.
- It’s related to the reflected illumination not yet accounted for (or in other words the energy yet unshot)

Ambient term

- An estimate of the average form factors can be made from their areas.
  \[
  F_{ij} = \frac{A_i}{\sum_{j=1}^{m} A_j} 
  \]
- The area-weighted average of reflectivities
  \[
  \rho = \frac{\sum \rho_i A_i}{\sum A_i}
  \]

Ambient term

- The energy will be reflected over and over, so the total reflection can be expressed as
  \[
  R_{\text{total}} = 1 + \rho + \rho^2 + \rho^3 + \ldots = \frac{1}{1 - \rho}
  \]

Ambient term

- Ambient term is total of the area-weighted unshot energy times the total reflectivity
  \[
  B_{\text{ambient}} = R_{\text{total}} \sum (\Delta B_i F_{ij})
  \]
- Each element displays its own fraction
  \[
  B_{\text{display}} = B_i + \rho_i B_{\text{ambient}}
  \]

Progressive radiosity + ambient
Accuracy of solution

• Result: 1 radiosity value per element so 1 color per element

• Meshing:
  • Partition surfaces in scene into small patches
  • Meshing conditions
    • Good representation of intensity changes
    • As less elements as possible

Meshing

• Uniform meshing
• Adaptive meshing
  • Make (uniform) start mesh and modify mesh (more elements) where large intensity differences found
  • Discontinuity meshing
    • Determine before radiosity computations where large intensity changes will occur. Mesh finer along intensity transitions

Meshing

Uniform mesh

Reference picture & uniform mesh
**Meshing problems**

**Increase resolution**

**Adaptive meshing**

**Discontinuity Meshing**

- Create mesh depending on expected shadow boundaries
- Captures nice shadow box
- Complex geometric comp
- Resulting mesh complex
Radiosity steps

- Create scene
- Take care of accurate material and light source definitions
- During modeling, keep in mind the problems that can occur during application of radiosity
- Make a coarse meshing, compute and inspect results
- Adapt materials and light sources
- Make a "good" meshing
- Compute radiosity results
- Use radiosity results to make one or more images

Radiosity summarized

- Computation of diffuse interreflection
- Light paths: LD^x
- Extension to L(DIS)^x possible
  - Very expensive
  - Viewpoint independent
  - Accuracy of results depends on meshing
  - High memory and time costs

Rendering after radiosity step - 1

- Direct use of radiosity results:
  - Radiosity results replaces diffuse computation in local illumination function
  - Gouraud shading
  - Fine meshing to get nice shadows

Rendering after radiosity step - 2

- Use radiosity results for indirect illumination only
  - Re-compute direct light during rendering
  - Less fine meshing required
Rendering after radiosity step - 2

Rendering after radiosity step - 3

- Use radiosity results to re-compute direct and indirect illumination
- Use radiosity results as emission values
- Regard all patches will as light sources
- Coarse meshing suffices

Radiosity today

- Used in architectural simulation
- Used for game lighting preprocessing