Indirect illumination

- Path tracing: slow
- Final gathering using photon maps: slow (though not quite as slow as path tracing)
- Radiance estimation using photon maps: inaccurate for reasonable number of photons
- Other ideas?

Observation

- Indirect irradiance (irradiance due to indirect illumination) changes very smoothly
  - Except for caustics
- Remember
  - Irradiance is incident power per area, before being reflected
  - Hence textures (spatially varying BRDFs) don’t play a role

Irradiance caching

- „A ray tracing solution for diffuse interreflection“, Ward, Rubinstein, Clear, SIGGRAPH 88
- One of the most popular strategies to accelerate global illumination
- Works well in practice, though artifacts may appear because of heuristics involved
Irradiance caching

Idea
• Use only on diffuse surfaces (or diffuse component of BRDF)
• Cache irradiance samples in image plane
  - Compute and store irradiance samples at sparse locations in image plane (using photon mapping or path tracing)
  - Interpolate cached samples to each pixel and use for diffuse shading
  - Greedily compute new samples only if interpolation fails (use heuristics)

Comparison to photon maps
• Photon mapping has two passes
  - Tracing photons from lights
  - Using photon map for shading
• Irradiance caching is performed in main rendering pass
  - Photon mapping and irradiance caching can be combined
• Note
  - Photons store "incident radiance" instead of irradiance
  - Photons are stored everywhere in scene, irradiance samples only at hit points of primary (eye) rays
• Limitation of irradiance caching
  - Works only with diffuse surfaces (or diffuse component of BRDF) since irradiance is stored
• Irradiance caching is more of a hack...

Irradiance caching algorithm
Three components
• Irradiance computation for a sample
• Irradiance caching
  - Sample point distribution
  - Determine where to place samples
• Irradiance extrapolation
  - Using samples to shade pixels

Irradiance computation
\[ E(x) = \int_{\mathbb{D}} L_i(x, \omega_i) \cos \theta_i d\omega_i \]

- Monte Carlo estimate
- Stratified sampling of the hemisphere using spherical coordinates \(\theta, \phi\)
- Probability density (pdf) of stratified samples cancels out \(\sin\) and \(\cos\) factors
Irradiance computation

\[ E(\mathbf{x}) = \int_{\mathbf{sr}} L(\mathbf{x}, \omega_i) \cos \theta_i d\omega_i \]
\[ = \int_0^\pi \int_0^{2\pi} \frac{r_i^2}{2} L(\mathbf{x}, \theta_i, \phi_i) \cos \theta_i \sin \theta_i d\theta_i d\phi_i \]
\[ = \pi \int_0^1 \int_0^1 L(\theta_i, \phi_i) \theta_i \sin \theta_i \theta_i d\theta_i d\phi_i \]

- Subdivision of azimuth, elevation angles into \( T \cdot P \) steps, \( r=1..P, p=1..P \)
- Uniform random random variables \( \xi, \psi \) for jittering
  \[ \theta_i = \sin^{-1} \left( \sqrt{\frac{t - \xi}{T}} \right), \quad \phi_i = 2\pi \frac{p - \psi}{P} \]

Irradiance sample distribution

- For each sample, determine a region in the image within which it can be used for interpolation
- When computing irradiance at hit point of primary (eye) ray
  - Look up if there is a cached sample within valid region, if so use it to extrapolate
  - If not, compute new sample

Indirect irradiance

Irradiance sample distribution

- Where irradiance changes quickly, valid range for cached samples should be small
- Where irradiance changes slowly, range should be large
- Rate of change of irradiance depends on distance to closest visible surfaces as seen from sample point

Irradiance caching

- Irradiance sample
  ```c
  struct irradiance_sample {
    vector3 k // \text{iRradiance}
    vector3 n // normal
    vector3 p // position
    float r // range
  }
  ```
Irradiance extrapolation

- Need fast access to nearby samples similar as in photon mapping (radiance estimation)
- Store samples in spatial data structure
  - Octree
- Add sample to each cell that it overlaps
- Adaptively subdivide octree such that each cell has limited number of samples

Irradiance extrapolation

- Extrapolation: compute weighted average of nearby samples
- Need to determine
  - Which samples should be used for extrapolation
  - Not all samples within valid range should be used
  - Weight for used samples

Error estimate

- Given a point on the surface and its normal and a cached location, estimate the difference of their irradiances
- Use error estimate to compute weight

Irradiance extrapolation

- Interpolation weights
  \[ w_j(x) = \frac{1}{\|x-x_j\| + \sqrt{1 - n(x) \cdot n(x_j)}} \approx e_j(x) \]

- Interpolated irradiance
  \[ E(x) = \frac{\sum_i w_i(x) E(x_i)}{\sum_i w_i(x)} \]
  \[ E(x_i) \] is irradiance stored at sample \(i\)

Error estimate

- Ad-hoc estimate for difference \(e_j(x)\) of true irradiance at point \(x\) with normal \(n(x)\) and sample \(j\) with position \(x_j\), normal \(n(x_j)\)
- "Split sphere" model [Ward et al. 88]

\[ e_j(x) \leq E_j \frac{\|x-x_j\|}{r_j} + \sqrt{2 - 2n(x) \cdot n(x_j)} \]

Algorithm

Finds irradiance at hit point \(x\) of eye ray
\(W = 0\)

for( all irradiance samples \(j\) in octree cell overlapping with \(x\) ) {
  compute weight \(w_j\)
  if( <sample is valid> ) {
    \(W += w_j; \text{wE} += w_j*E[j]\)
  }
}

if( \(W > 0\) ) return \(\text{wE}/W\)
else {
  compute & store new irradiance sample
  return value of new sample
}
Algorithm

\[ \text{<sample is valid> } = \]
\[ \text{dist}(x - x[j]) < r[j] \quad \text{// within range} \]
\[ a_s > 1/a \quad \text{// sufficient weight} \]
\[ \text{dot}(x[j] - x, n(x)) < 0 \quad \text{// } x[j] \text{ is behind } x \]

Sample at \( x_j \) is invalid

Non-diffuse surfaces/BRDFs

Approximation
\[ I_{sh}(x, \omega_i) = \int_{\Omega} f(x, \omega_i, \omega_o) I_o(x, \omega_o) \cos \theta_i d\omega_o \]
\[ \approx \int_{\Omega} f(x, \omega_i, \omega_o) d\omega_o \left( \int_{\Omega} I_o(x, \omega_o) \cos \theta_i d\omega_o \right) \]
\[ = \frac{1}{2} \rho_{sh}(\omega_i) E(x) \]

Hemispherical directional reflectance \( \rho_{sh}(\omega_i) \)

Photon mapping and irradiance caching

- Final gathering often used to compute irradiance samples
  - Advanced: use photon map for importance sampling when tracing gather rays
- Caustics break assumptions of irradiance caching
  - Exclude caustic paths from irradiance sample computation
  - Use photon map to render caustics

Examples

- 1000 sample rays

Examples

- 5000 sample rays

Examples

- Same computation time
Conclusion

- **Pros**
  - Irradiance caching can speed up computation of indirect illumination a lot!

- **Cons**
  - Shading using irradiance samples works only for diffuse surfaces/BRDFs
  - Non-diffuse requires approximation
  - Irradiance sample distribution and interpolation is hacky

- **Debatable**
  - Images look plausible, but are not "physically plausible" because of approximations