Photon Mapping
Photon Maps

• The photon map stores the lighting information on points or “photons” in 3D space (“on”/near 2D surfaces)
  – As opposed to the radiosity method that stores information on surface triangles
Monte Carlo Integration
Monte Carlo Method

• The finite element method is a Newton-Cotes quadrature approach, and as such it does not scale well to higher dimensions
  – 3D space + 2D angles = 5D (or 4D ignoring participating media)

• Thus we previously made the approximation that the lighting was purely diffuse in order to reduce the dimensionality of the problem and make it tractable
  – Integrated over angles to drop two dimensions; results was 2D (or 3D)

• Although Monte Carlo integration is worse than Newton-Cotes quadrature for lower dimensional problems, it scales well on higher dimensional problems
  – No curse of dimensionality

• The Monte Carlo method allows us to tackle the full higher dimensional lighting equation, without assuming that the lighting is purely diffuse

• Monte Carlo methods rely on repeated random sampling
Example

Consider calculating $\pi \approx 3.14$ using both a finite element method and a Monte Carlo method....
Finite Element Approach

- Inscribe $T$ triangles within a circle of radius 1 and calculate the total area covered by the triangles
- As the number of triangles increases, the triangles more accurately approximate the area of the circle
- Total area covered by the triangles approaches $\pi$ since $A = \pi r^2$ and $r = 1$

$T = 4, \pi \approx 2$

$T = 8, \pi \approx 2.8284$
Monte Carlo Approach

• Randomly choose $N$ points in a 2 by 2 square enclosing the circle
• Let $N_{\text{inside}}$ denote points whose distance to the center of the square is less than 1
• The approximation for $\pi$ is given by $4 \frac{N_{\text{inside}}}{N}$, since $\frac{A_{\text{circle}}}{A_{\text{box}}} = \frac{\pi}{4}$ and each point in the box is equally likely to be chosen

$N = 1000, \pi \approx 3.136$

$N = 2000, \pi \approx 3.1440$
Random Numbers

- **Random variables** are expressions whose value is the outcome of a random experiment.
- The **sample space** is the set of all outcomes.
- A **probability distribution** $p(x)$ describes the probability of selecting each outcome in the sample space.
- We **sample** the value of a random variable by choosing a random element in the sample space with a probability determined by $p(x)$.
- **Pseudorandom number generators** (PRNGs) are deterministic algorithms that allow us to generate sequences of quasi-“random” numbers based on an initial seed (a starting point in the predetermined sequence).
- PRNGs allow us to generate a (pseudo) random real number between 0 and 1 with an equal, or **uniform**, probability.
- Uniformly sampling [0,1] allows us to sample any other sample space with its non-uniform probability distribution.
Monte Carlo Integration

• Consider the integral

\[ \int_a^b f(x) \, dx \]

• The Monte Carlo estimator is given by

\[ F_N = \frac{b - a}{N} \sum_{i=1}^{N} f(X_i) \]

– where \( N \) is the number of samples and the \( X_i \) are uniformly randomly generated in the interval \([a,b]\)

• The \textbf{expected value} of the estimator, is the value of the integral
Importance Sampling
Importance Sampling

• Suppose $f(x)$ is only nonzero in $[a_1, b_1]$ which is a subset of $[a, b]$, i.e.,
\[ \int_a^b f(x) \, dx = \int_{a_1}^{b_1} f(x) \, dx \]

• Only need to sample $X_i$ uniformly over $[a_1, b_1]$, since all samples outside $[a_1, b_1]$ do not contribute to the integral

• Change the probability distribution $p(x)$ from which we sample $X_i$ from the uniform distribution over $[a, b]$ to the uniform distribution over $[a_1, b_1]$

• More generally, the probability distribution $p(x)$ should prefer samples in areas with higher contributions, or importance, to the integral

• For a general $p(x)$ with $\int_a^b p(x) \, dx = 1$ (and $p(x) = 0$ outside $[a, b]$), the Monte Carlo estimator is
\[ F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \]

  – reduces to the usual case if we uniformly sample with $p(x) = \frac{1}{b-a}$
Importance Sampling

• Monte Carlo estimates for $\int_0^1 x^2 \, dx$ with $N = 100$ samples:

<table>
<thead>
<tr>
<th>$p(x)$</th>
<th>Estimate using $F_N$</th>
<th>Relative Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.33671</td>
<td>1.01%</td>
</tr>
<tr>
<td>2x</td>
<td>0.33368</td>
<td>0.105%</td>
</tr>
<tr>
<td>3x^2</td>
<td>0.33333</td>
<td>0.000%</td>
</tr>
</tbody>
</table>

• Importance sampling does not necessarily reduce the error
  – it does not always work and can give worse results
• The more $p(x)$ “resembles” $f(x)$, the lower the error for a fixed number of samples
• If $p(x) = \frac{f(x)}{\int_a^b f(x) \, dx}$ then we obtain the analytic answer with a single sample
• However, this $p(x)$ depends on the integral we want to compute
• Instead, choose $p(x)$ based on information that does not require knowing $f(x)$ such as physical principles or an approximate solution
Gummy Bears
by
Kirk Shimano
and
Mike Sego

Photon Maps
Photon Maps

• Monte Carlo integration methods can be used to create light maps

• The photon map stores the lighting information on points or “photons” in 3D space (“on”/near 2D surfaces)
  – As opposed to the radiosity method that stores information on surface triangles

• For efficiency and ease of implementation, every photon is given the same strength
  – Thus, for example, brighter lights emit more photons instead of higher energy photons
Photon Maps - Emission

- Given an overall desired number of photons, divide them up amongst the various lights in the scene proportionally based on the relative power of each light.
- Using random numbers, emit all of a light’s photons into the scene.
- For point lights, all the photons are emitted from a single point.
- For area lights, use random numbers to select a point to emit each photon.
- Sometimes semi-random emission is used.
  - For example: first divide a rectangular light into a uniform 2D grid of some resolution, and then emit a certain number of photons per 2D grid cell – the position within a 2D grid cell can still be randomized.
- Every photon needs to be emitted in some direction, so use random numbers to choose a direction on the sphere (or hemisphere) for emission.
  - Or a subset of the sphere for a spotlight, etc.
- For some scenes, such as those outdoors (consider the sun), many or most photons will miss the scene entirely.
  - As an optimization, one can simply ignore those photons.
  - Typically, it is efficient to only consider the fraction of the light that interacts with your scene, and to only generate and emit photons for this sub-light (don’t forget to scale down the energy of a sub-light when dividing up photons).
Photon Map - Storage

• For each photon, use the standard ray tracer to find the first piece of geometry that it intersects.
• Every time a photon intersects a surface, its data is added to the photon map representing incoming light.
• Simply create a new particle to store in the photon map.
• Store the photon’s current location in 3D space along with the incoming ray direction that brought it to that intersection location:
  – don’t need to record the energy, since all photons have the same energy.
• Don’t delete the photon, or move it into the photon map, since we may still need it to bounce around more:
  – Simply create a new storage photon in the photon map.
Photon Map - Absorption

- After storing the photon’s incoming light direction in the photon map, use random numbers to determine what the photon does next.
- There is some chance that the photon is absorbed by the surface.
  - Objects absorb some of the incoming light, which is what gives them their color.
- Absorbing some fraction of the photon’s energy results in photons with unequal energy - so we do not do that.
- Instead, we use the fraction of light energy that would be absorbed in order to create a probability that a photon is absorbed.
- Then we uniformly generate a random number and compare it to that probability in order to see if the photon is absorbed or not.
- When a photon is absorbed, the process stops and we move on to the next photon.
- Otherwise we bounce the current photon and continue following it to the next surface.
Photon Map - Bouncing

- Reflected photons need a new direction
- This is computed by first mapping the directions of the BRDF into probabilities
  - E.g. a purely diffuse BRDF has an equal probability for all directions on the hemisphere
- Then a random number is generated and compared to the BRDF table in order to determine the bounce direction
- The photon travels off in this new direction until it intersects another surface
- The new surface intersection location and the current incoming light direction are then stored in the photon map
- The photon is once again checked for absorption, and if it’s not absorbed, it is again bounced according to the BRDF
- A maximum number of bounces before termination is typically set
  - It can be set rather high as photons typically have a diminishing chance of avoiding absorption as the number of bounces increases
Photon Map

Physically Based Rendering by Pharr and Humphreys
Rendered Image

*Physically Based Rendering* by Pharr and Humphreys
Using Photon Maps
Photon Map – Direct Lighting

• It’s more accurate to evaluate direct lighting at a point using the typical shadow rays from that point to the lights
  – rather then interpolating lighting information from nearby photons in the photon map
• Thus, the first time a photon emitted from a light hits an object, it is not stored in the photon map
  – (you will store it for your HW assignment though)
• This makes the photon map a lot more efficient since we don’t need to store any photons for direct illumination
Photon Map – Specular

• It is often more convenient to create separate BRDFs for the diffuse and specular lighting
• Then when bouncing a photon, first use a random number to determine if the photon is absorbed, or undergoes a diffuse bounce, or undergoes a specular bounce
• Afterwards, a second random number is used to pick the direction for the diffuse or specular bounce
• We make two photon maps:
  – A caustic photon map stores samples from photons that have undergone only specular bounces up to the point at which they are stored in the map
  – An indirect lighting map stores any photon that has undergone at least one diffuse bounce
Photon Maps

Direct → Specular → Caustic → Any
Emission

Indirect ← Nonspecular ← Specular/Non-specular
Reminder: Caustics
Gathering Radiance

• Trace rays from the camera and at each intersection point:
  – Use traditional shadow rays to account for direct lighting
  – Estimate the radiance contribution to the ray from caustics and indirect lighting using the respective photon maps
• Find the $N$ closest photons to a point on the surface of an object
  – Use an acceleration structure to store the photons: K-D tree, uniform grid, etc.
Photon Map – Color

• If all photons are white light, then the entire photon map will be white light
• Create three photon maps, one for each color channel: red, green, blue
• Then objects of a certain color will tend to absorb photons of different colors creating differences in the photon maps
• This gives color bleeding and related effects...
And...
Question 1: Name?
Don’t forget make-up...