Realistic Image Synthesis

- Density Estimation -

Roman Durikovic

Overview

- **Today**
  - Density estimation background
  - Density estimation methods
  - Global illumination algorithms based on density estimation

Reading Materials

Basic:

- **B.J. Walter, Density Estimation Techniques for Global Illumination, PhD thesis, Cornell University, 1998**
- **P. Dutre, P. Bekaert, and K. Bala, Advanced Global Illumination, AK Peters 2003**

Advanced:

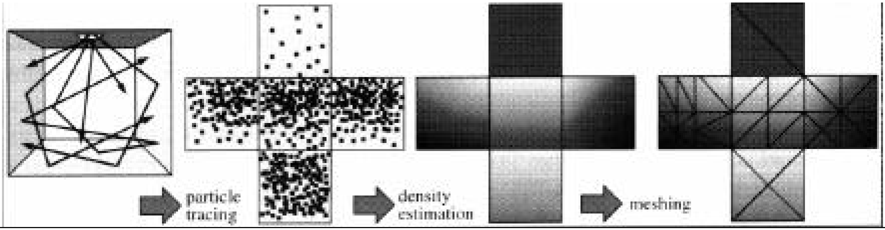
- **B.W. Silverman, Density Estimation for Statistics and Data analysis, Chapman and Hall, 1986**
- **M.P. Wand and M.C. Jones, Kernel Smoothing, Chapman and Hall, 1995**

Photon Transport Simulation

- **Instead of simulating the exact system, an *analog* system which is easier to simulate can be used**
  - must retain all the *important* characteristics of the original system.
- ***Photons* used in global illumination algorithms are simplified analogs of photons (light particles) in physics.**
- **The simplified photon characteristics**
  - emitted by light sources and carry some energy,
  - travel in space obeying geometrical optics laws,
  - traced in space until they are completely absorbed due to reflections and refractions.
- **Time factor is ignored**
  - it is assumed that photons are moving instantaneously.

Global Illumination via Density Estimation

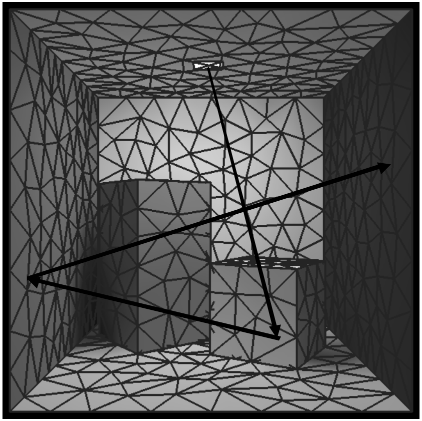
- **A typical algorithm consists of three consecutive phases:**
  1. ***photon tracing (continuous random walk)***
  2. **lighting reconstruction via *density estimation*,**
  3. **lighting storage and rendering.**
- **The lighting function is available implicitly as the density of photons hitting points**
  – Reconstructing illumination out of collected photons is a density estimation problem.
- **Various techniques are used to store/display lighting:**
  – illumination maps (textures), meshing, or a direct density estimation at chosen sample points.

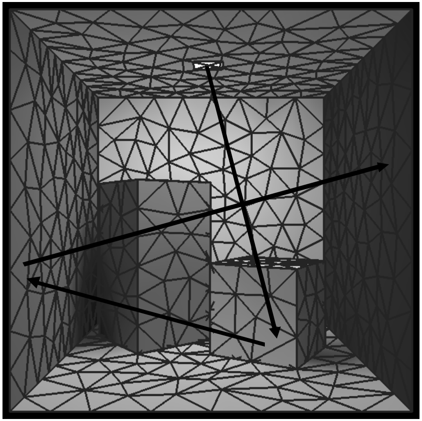


The surfaces in the room are depicted "unfolded" in the three figures on the right.

Random Walks

Continuous vs. Discrete Random Walks





Solves integral equations: radiosity or rendering equations

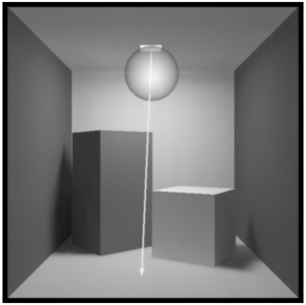
Solves linear systems: discretization error propagation

Light Source Sampling

Sample point on light source with probability proportional to self-emitted radiosity:

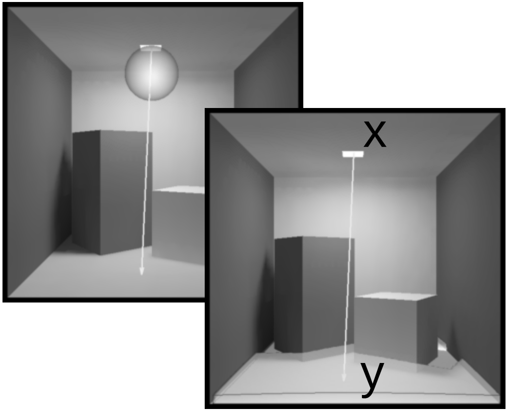
$$S(x) = E(x)/\Phi_T$$

Making the First Transition (1)



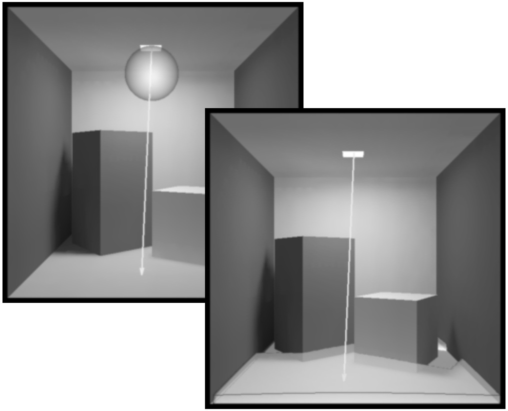
- **No absorption at the origin**
- **Sample direction according to directional distribution of self-emitted radiance.**
  - Diffuse emission: pdf is $\cos(\theta_x)/\pi$

Making the First Transition (2)



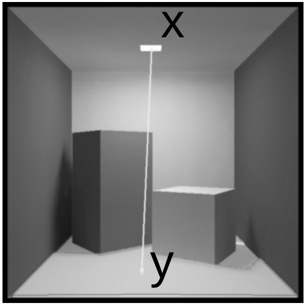
- **Shoot ray along sampled direction.**
- **Geometric density factor:**
  $\cos(\theta_y) / r^2_{xy}$

Making the First Transition (3)

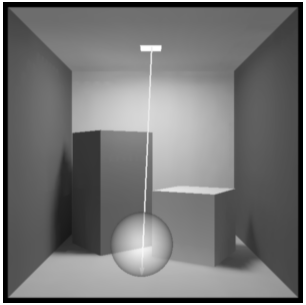


- **Full transition density T(x,y) is product:**

$\cos(\theta_x)\cos(\theta_y) / (\pi\, r^2_{xy})$



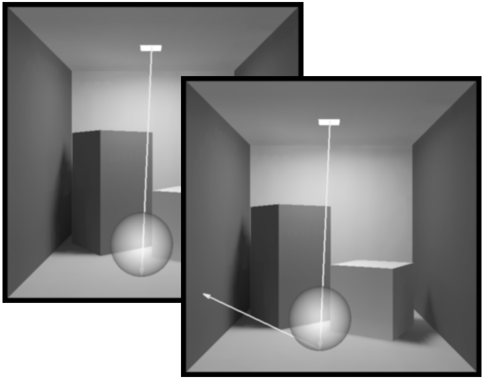
Further Transitions



1. **Absorption / survival test according to albedo**

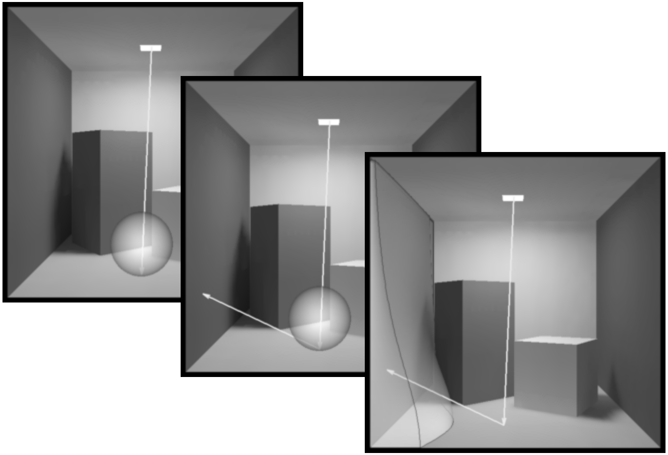
Further Transitions (2)

2. Sample direction according to brdf



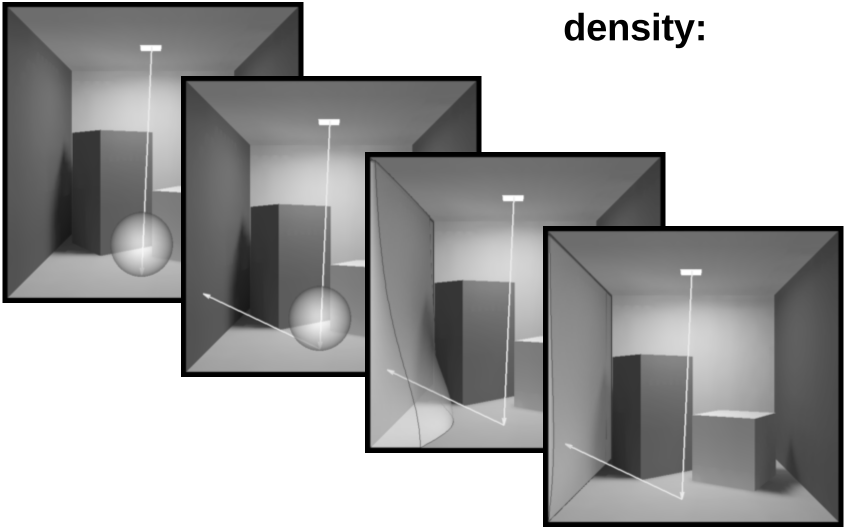
Further Transitions (3)

3. Shoot ray

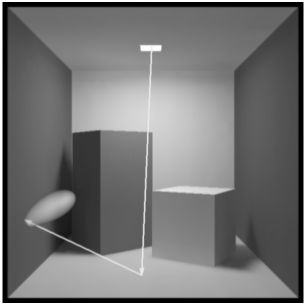


Further Transitions (4)

- **Full transition density:**

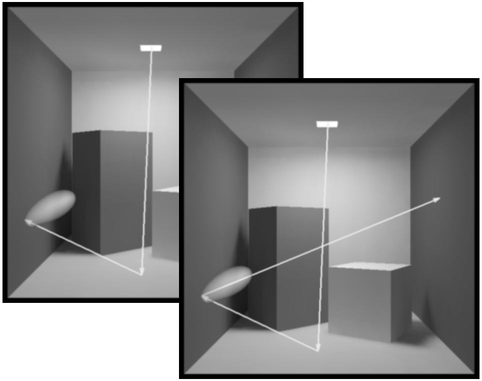


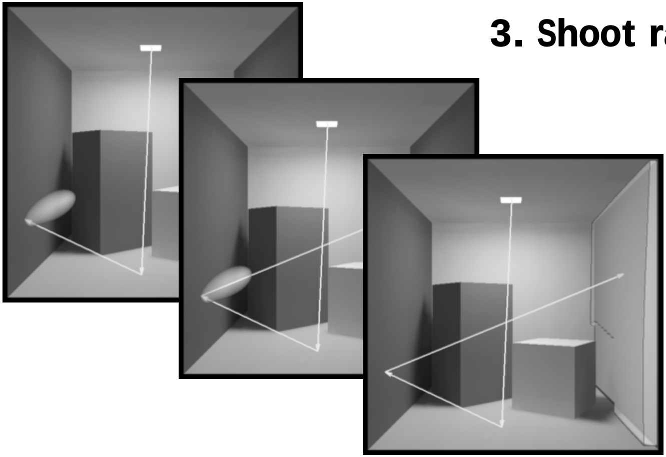
Once More …



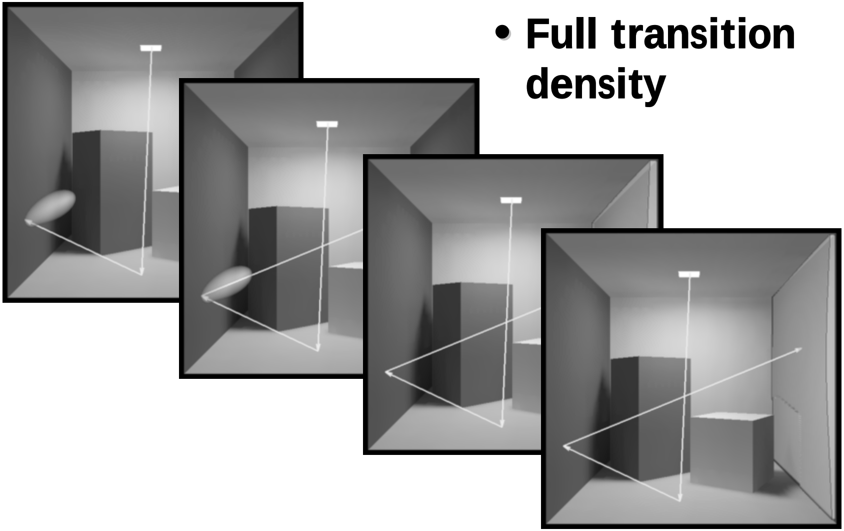
1. Absorption / survival test

2. Sample direction according to brdf





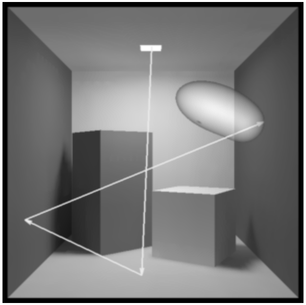
3. Shoot ray



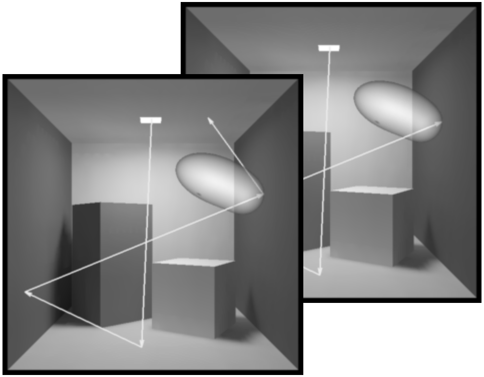
- **Full transition density**

And Yet Once More

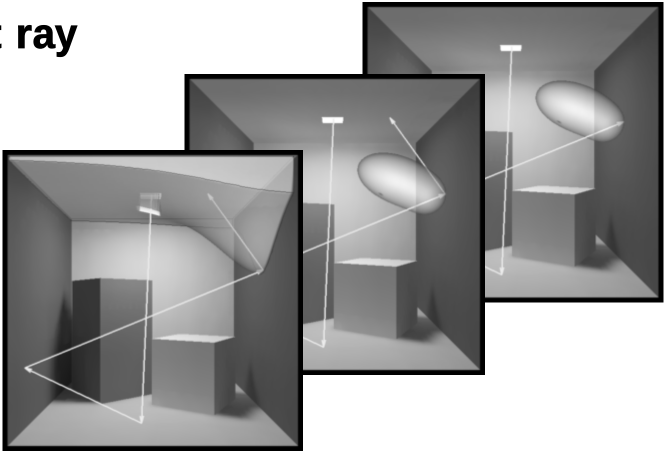
1. **Absorption / survival test**



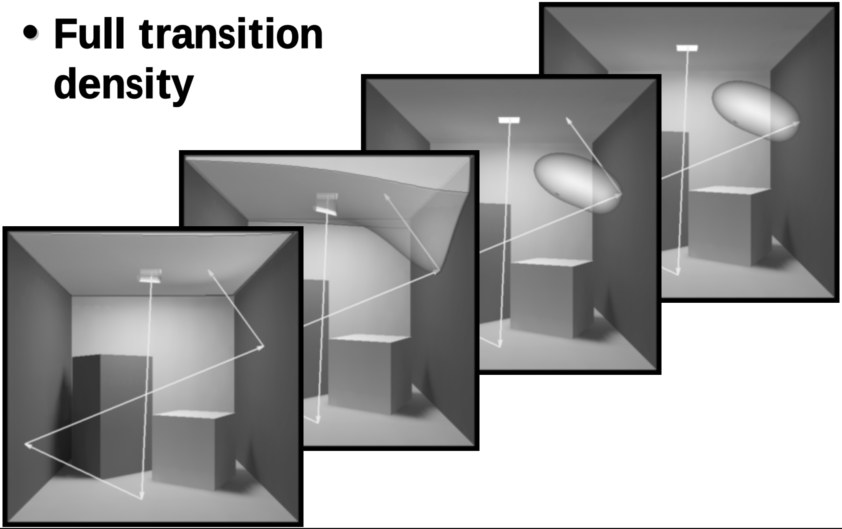
2. Sample direction according to brdf



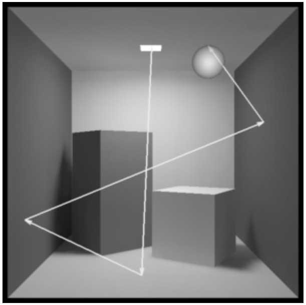
3. Shoot ray



- **Full transition density**



End of Game



1. Absorption

Collision Density

- **In general:**

$$\boxed{D(X) = \underbrace{S(X)}_{\text{Path origins at X}} + \int D(Y)\underbrace{T(Y,X)}_{\text{Visits to X from elsewhere}}dY}$$

- **Random walk simulation yields points with density which is solution of second kind Fredholm integral equation**

Collision Density for Radiosity

- **Radiosity integral equation:**

$$\boxed{B(x) = E(x) + \int_S B(y)\ \frac{\cos\theta_y}{\pi}\ \frac{\cos\theta_x}{r_{yx}^2} \mathrm{vis}(y,x)\ \rho(x)\ dA_y}$$

Source density should be normalized,
S(x) = E(x)/F_{T}, but we're almost there!

Collision Density for Radiosity

- **Divide by total self-emitted power:**

$$\boxed{\frac{B(x)}{\Phi_T} = \frac{E(x)}{\Phi_T} + \int_S \frac{B(y)}{\Phi_T} \; \frac{\cos\theta_y}{\pi} \; \frac{\cos\theta_x}{r_{yx}^2} \mathrm{vis}(y,x) \; \rho(x) \; dA_y}$$

Collision Density for Radiosity

$$\boxed{\frac{B(x)}{\Phi_T} = \underbrace{\boxed{\frac{E(x)}{\Phi_T}}}_{\text{Source density } S(x)} + \int_S \frac{B(y)}{\Phi_T}\;\frac{\cos\theta_y}{\pi}\;\frac{\cos\theta_x}{r_{yx}^2}\mathrm{vis}(y,x)\;\rho(x)\;dA_y}$$

Source density S(x)

Collision Density for Radiosity

$$\boxed{\frac{B(x)}{\Phi_T} = \boxed{\frac{E(x)}{\Phi_T}} + \int_S \frac{B(y)}{\Phi_T} \boxed{\frac{\cos\theta_y}{\pi}\ \frac{\cos\theta_x}{r_{yx}^2} \mathrm{vis}(y,x)\ \rho(x)} dA_y}$$

$\uparrow$ **Source density S(x)**

$\uparrow$ **Transition density T(y,x):**

1. **sample cosine distributed direction at y**
2. **shoot ray; ray hits x**
3. **survival test at x**

Collision Density for Radiosity

- **Collision density proportional to radiosity**

$$\boxed{\boxed{\frac{B(x)}{\Phi_T}} = \boxed{\frac{E(x)}{\Phi_T}} + \int_S \boxed{\frac{B(y)}{\Phi_T}} \boxed{\frac{\cos\theta_y}{\pi} \; \frac{\cos\theta_x}{r_{yx}^2} \mathrm{vis}(y,x) \; \rho(x)} \, dA_y}$$

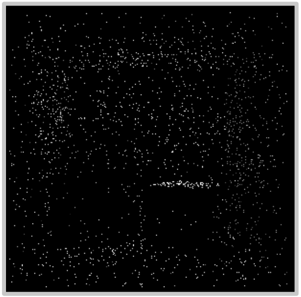
Source density S(x)

Transition density T(y,x):

1. **sample cosine distributed direction at y**
2. **shoot ray; ray hits x**
3. **survival test at x**

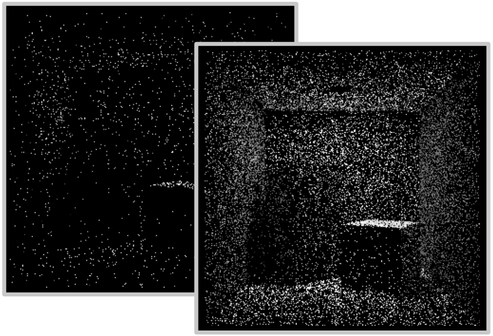
$$\boxed{\mathbf{D(x) = B(x)/}\mathrm{F}_{\mathbf{T}}}$$

Sampled Points



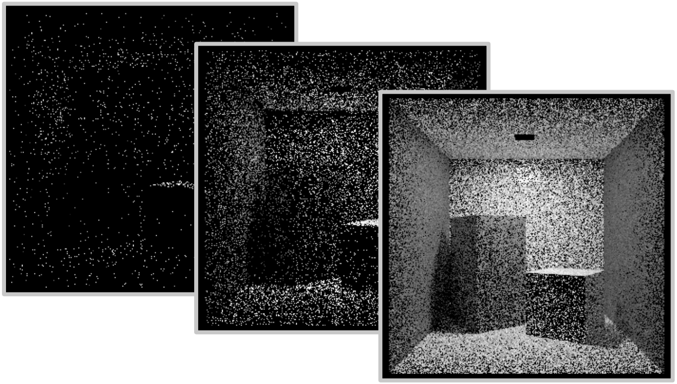
- **1,000 paths**

Sampled Points



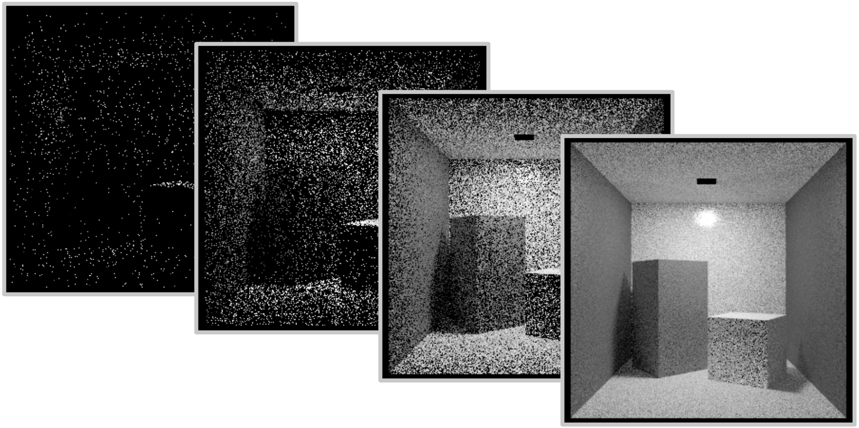
- **10,000 paths**

Sampled Points



- **100,000 paths**

Sampled Points



- **Collision density is related to radiosity!!**

Photon Tracing: Summary

For each photon repeat the following steps:

1 **Choose probabilistically:**

- the wavelength of the photon by sampling the **emission spectrum**,
- the location of the photon on the emitter surface by sampling the **positional emission power distribution**,
- the direction of propagation of the photon by sampling the **directional power distribution**.

2 **Assign the energy to the photon and trace it until it is absorbed:**

- find the first object hit by the photon (use ray tracing),
- decide on photon absorption or reflection by testing a random number against surface albedo,
- if the photon is reflected:
    - assign a reflected direction to the photon by sampling BRDF,
    - update the outgoing photon flux and continue tracing the photon

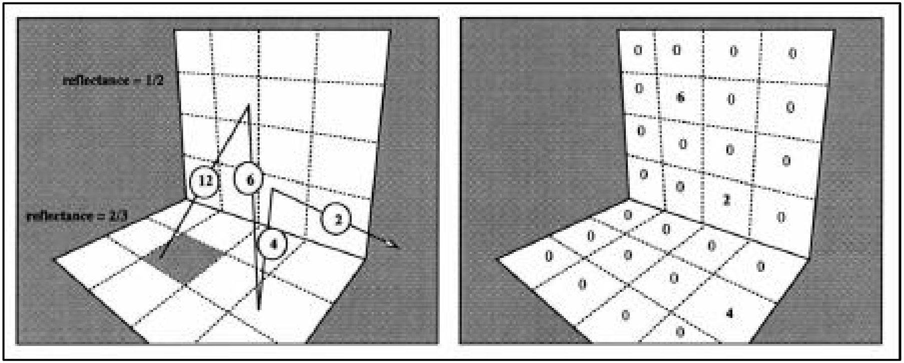
Handling Photon's Power

- **Assign *the same power* to every photon**
  - Draw a random number [0,1] from the uniform distribution and com pare with the albedo of surface hit by the photon.
  - If the photon "survives" the absorption test it is reflected and further traced carrying **the same** power.
- **Attenuate photon power for each surface hitting event**
  - Use the **Russian roulette** technique to avoid **bias (systematic error**) in the solution. Once photon energy weight ***w*** has fallen below the threshold, the photon is either absorbed with probability ***p*** or survives (with probability ***1-p***), but then its weight is increased by multiplying by ***1/(1-p).*** The expected value ***E(w)*** of the weight ***w*** after playing Russian roulette is given by*:*

$$E(w) = \Pr(\text{absorption}) \cdot 0 + \Pr(\text{survival}) \cdot \frac{w}{1-p} = p \cdot 0 + (1-p)\frac{w}{1-p} = w$$

which is the original weight of the photon, i.e. on average the photon has the right weight.

Photon with Attenuated Energy



Left: Photon with the power 12 watts is emitted by a patch on the floor

Right: the resulting power stored in the photon hit patches

Density Estimation

- **As a result of *continuous random walk* the lighting function is known *implicitly* as the density of photon collision points**
- **The lighting function in *explicit* form must be reconstructed**
  - This is a classic ***density estimation*** problem where an estimate of the probability density function is constructed from the observed data points.
- **Basic approaches to density estimation**
  - Parametric: a family of distributions is known and only predefined parameters must be found, e.g. mean μ and variance s^2 for the normal distribution.
  - Nonparametric: less rigid assumptions
    - This is the case for the global illumination problem

Density Estimation Methods

- **Histograms:**
  - A domain is subdivided into bins (buckets) in which the number of photons and/or their accumulated energy is stored.
- **Naïve estimator:**
  - Counts the number of collisions in a bin centered at point ***x*.**
- **Kernel estimators:**
  - The density is estimated as spatially spread energy distributions around each photon collision point.
- **Nearest neighbor methods:**
  - The density at a point ***x*** is estimated by dividing the number of the nearest neighbor photons ***k*** (usually fixed) by the area of a region centered at ***x***, in which these photons are collected.
- **Orthogonal series estimators:**
  - Higher order basis functions are used for lighting reconstruction in each bin (generalization of histograms)

Density Estimation

- **For simplicity let us consider 1D case.**
- **Notation:**

$f(x)$ and $\hat{f}(x)$: reconstructed pdf and its estimate at point x

X_i: photon collision location

n: the number of photon collisions

$K(x)$: the kernel function

h: the kernel radius (called also bandwidth or smoothing parameter) or the bin width for histogram estimators

Histograms

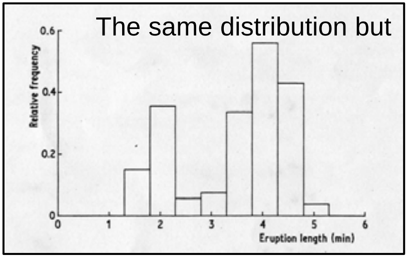
$$\hat{f}(x) = \frac{1}{nh} \times (\text{no. of } X_i \text{ in the same bin as } x)$$

or more general

$$\hat{f}(x) = \frac{1}{n} \times \frac{\text{no. of } X_i \text{ in the same bin as } x}{\text{width of bin contaning } x}$$

- $\hat{f}(x)$ **strongly depends on *h*, the choice of an origin and orientation of the grid of bins**

The same distribution but



… different histograms

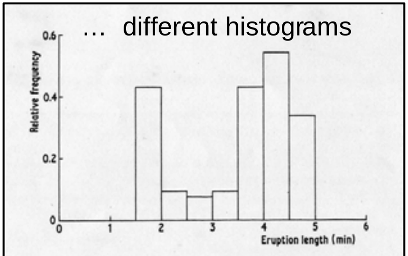


Fig. 2.1 *Histograms of eruption lengths of Old Faithful geyser.*

The Naïve Estimator

$$w(y) = \begin{cases} 0.5 & \text{if } |y| < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} w\left(\frac{x - X_i}{h}\right)$$

jumps at points X_i +/- h

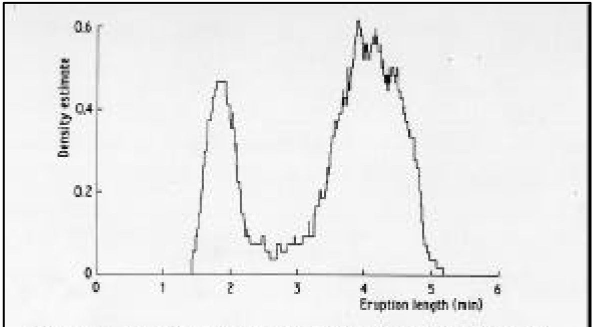


Fig. 2.3 *Naive estimate constructed from Old Faithful geyser data, h = 0.25.*

The Kernel Estimator

- **Generalization of the naïve estimator**

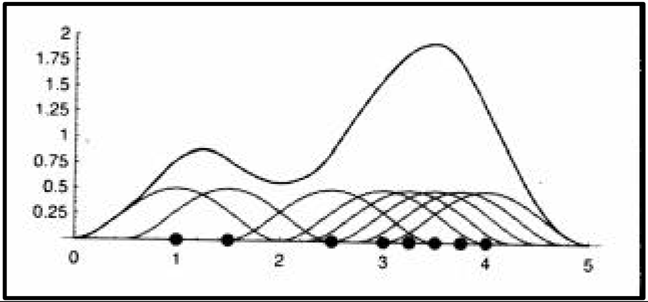
$$\int_{-\infty}^{\infty} K(x)dx = 1$$

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - X_i}{h})$$

- $\hat{f}(x)$ **inherits all the continuity and differentiability properties of the kernel** K **(usually a symmetric pdf)**
- **Well studied mathematically**
- **For a fixed** h **might have tendency to excessive smoothing** $\hat{f}(x)$ **regions with dense photon collisions** X_i **and leaving out visible noise in regions with low density of** X_i

The Kernel Estimator

At first the kernel function is chosen (usually smooth and easy to compute functions are used). Then the kernel radius ***h*** (the extent of the kernel support) is decided. It can be done globally for the whole surface or locally based on complexity of lighting distribution. Finally, the kernel function is centered at every photon location, and the photon energy is splatted (distributed) according to the kernel shape. The final lighting is estimated by summing splatted energy from all photons.



The Kernel Estimator

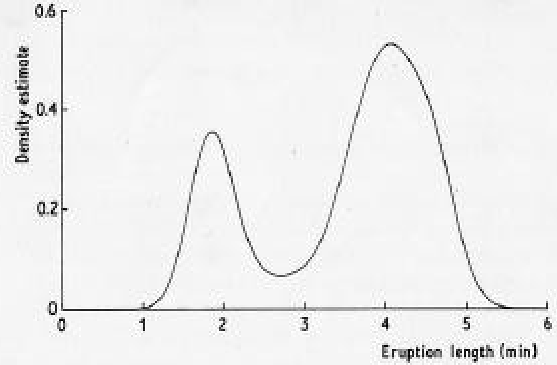
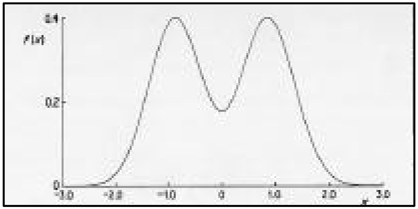


Fig. 2.8 *Kernel estimate for Old Faithful geyser data, window width 0.25.*

Bandwidth Sensitivity

- **Original bimodal density distribution**



- **Right: Kernel estimates for 200 simulated data points drawn from this bimodal density for kernel widths (a) 0.1, (b) 0.3, and (c) 0.6.**

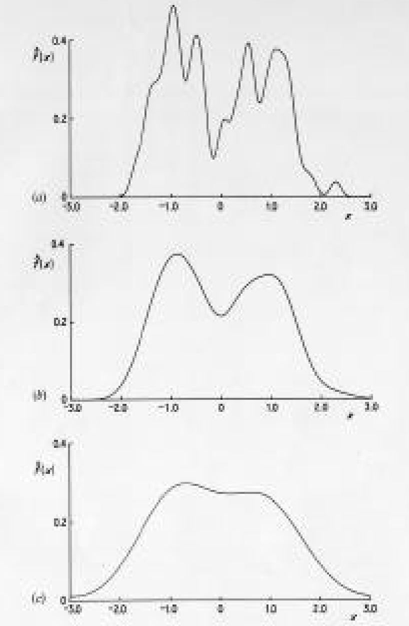


Fig. 2.6 *Kernel estimates for 200 simulated data points drawn from a bimodal density. Window widths: (a) 0.1; (b) 0.3; (c) 0.6.*

The Nearest Neighbor Method

$$\hat{f}(x) = \frac{k}{2nd_k(x)}$$

where $d_k(x)$ is such a distance that in the interval

$[x - d_k, x + d_k]$ k neareast collisions X_i is located

$$d_1(x) \le d_2(x) \le \ldots \le d_k(x)$$

- **The amount of smoothing locally adapts to the density of photon collisions** X_i
- **Reconstructed** $\hat{f}(x)$ **is not a pdf since it does not integrate to unity (problems with energy conservation)**
- **The generalized** k**-th nearest neighbor estimate**

$$\hat{f}(x) = \frac{1}{nd_k(x)} \sum_{i=1}^{n} K\left(\frac{x - X_i}{d_k(x)}\right)$$

The Nearest Neighbor Method

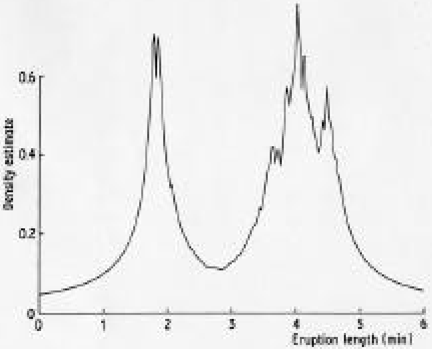


Fig. 2.10 *Nearest neighbour estimate for Old Faithful geyser data, k = 20.*

Generalization to Higher Dimensions

- **All discussed methods have similar properties when higher dimensional density estimation is considered**
- **2D is considered in the global illumination computation**
  - Histograms

$$\hat{f}(x) = \frac{1}{n} \times \frac{\text{no. of } X_i \text{ in the same bin as } x}{\textbf{area} \text{ of bin contaning } x}$$

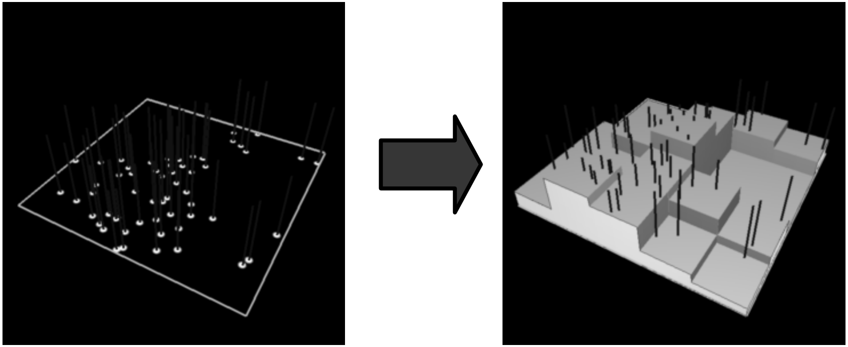
  - Kernel methods

$$\int_{-\infty}^{\infty} K(\mathbf{x})\,d\mathbf{x} = 1$$

$$\hat{f}(\mathbf{x}) = \frac{1}{nh^2} \sum_{i=1}^{n} K\{\frac{1}{h}(\mathbf{x} - \mathbf{X}_i)\}$$

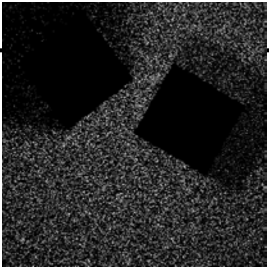
Histogram Method

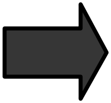
- **Break surfaces in small elements. Count photons hitting each element:**

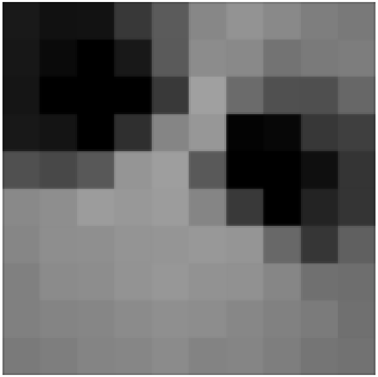


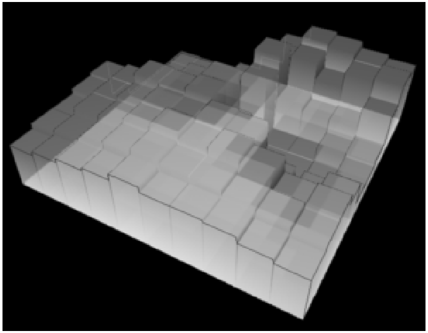
$$\hat{f}(x) = \frac{1}{n} \times \frac{\text{no. of } X_i \text{ in the same bin as } x}{\textbf{area} \text{ of bin contaning } x}$$

Histogram Method









Orthogonal Series Density Estimation

- **Linear, bi-linear, quadratic, cubic, … approximations**

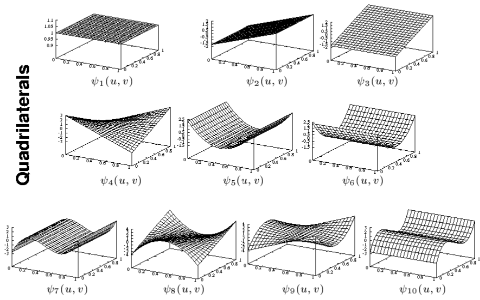
$\hat{f}(x)$ can be reconstructed using K orthonormal basis functions $y_n(x)$:

$$\hat{f}(x) = \sum_{n=1}^{K} \hat{f}_n y_n(x)$$

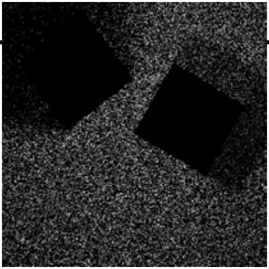
with the projection coefficients $\hat{f}_n$:

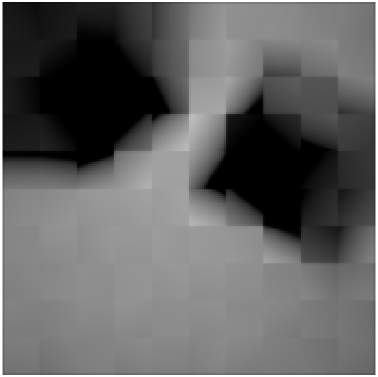
$$\hat{f}_n = \frac{1}{n} \sum_{i=1}^{n} y_n(X_i) a(X_i)$$

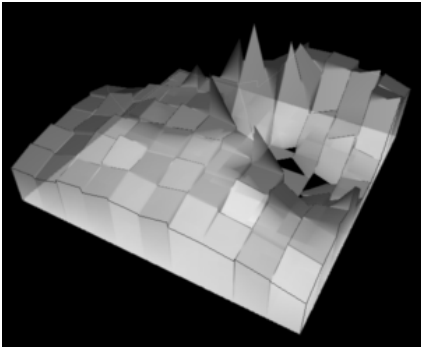
$$\int_{-\infty}^{+\infty} y_m(x) y_n(x) a(x) dx = d_{mn} = \begin{cases} 1 & m = n \\ 0 & \text{otherwise} \end{cases}$$



Orthogonal Series

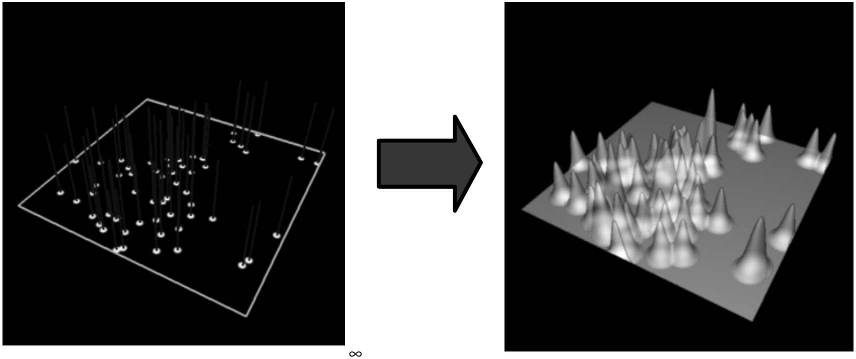






Kernel Density Estimation

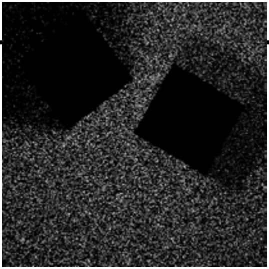
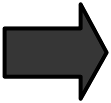
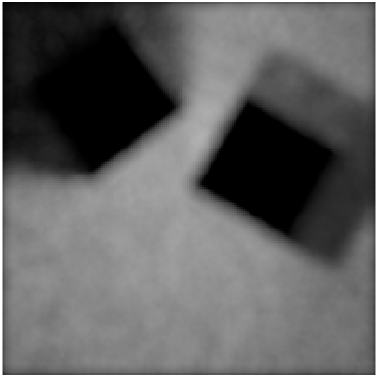
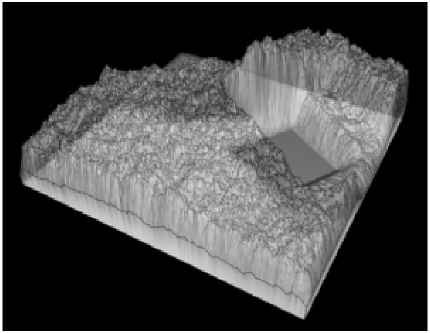
- **Place finite-width density kernel at each sample point**



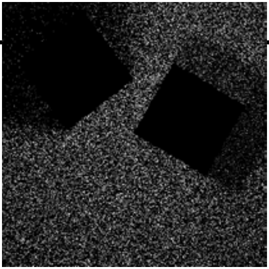
$$\int_{-\infty}^{\infty} K(\mathbf{x})\,d\mathbf{x} = 1$$

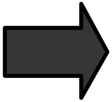
$$\hat{f}(\mathbf{x}) = \frac{1}{nh^2} \sum_{i=1}^{n} K\{\frac{1}{h}(\mathbf{x} - \mathbf{X}_i)\}$$

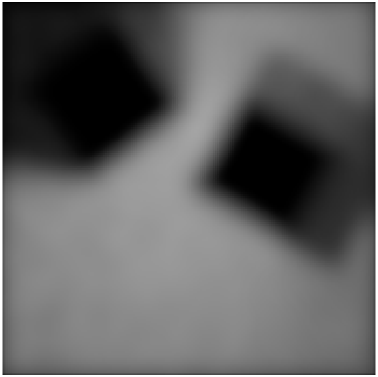
Cylindrical Kernel

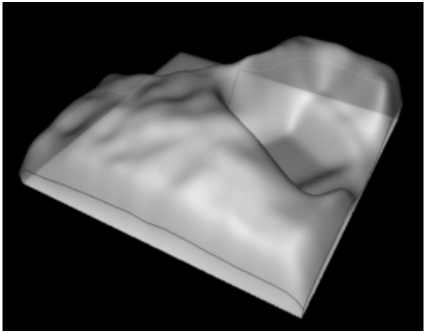





Gaussian Kernel









Comparison of Density Estimation Methods

- **Problems with energy conservation**
  - Nearest neighbor method
- **Discontinuities in reconstructed density**
  - Histogram, naïve, and nearest neighbor methods
- **Complexity as a function of the photon number** n
  - $O(n)$
    - **Histogram - fast**
    - **Naïve and kernel estimators – average**
  - $O(n \log n)$
    - **Nearest neighbor method – slow**
- **Adaptability to local density fluctuations**
  - Histogram, naïve and kernel estimators – poor
  - Nearest neighbor method – good

Bias and Random Error

- **The error between the illumination function** $f(x)$ **and its estimate** $\hat{f}(x)$ **can be expressed as:**

$$f(x) = f(x) - \hat{f}(x) = \underbrace{f(x) - E(\hat{f}(x))}_{f_{bias}(x)} + \underbrace{E(\hat{f}(x)) - \hat{f}(x)}_{f_{random}(x)}$$

$$E\hat{f}(x) = E\left[\frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)\right] = \frac{1}{nh}\sum_{i=1}^{n} E\left[K\left(\frac{x - X_i}{h}\right)\right] = \frac{1}{h}\int K\left(\frac{x-y}{h}\right) f(y)\,dy$$

- **The bias is a smoothed version of the true density** f
  - ➢ Bias = convolution of f with the kernel K scaled by the kernel size h

$$\hat{f}(x) = \textbf{smoothed version of true density} + \textbf{random error}$$

- **The bias does not depend directly on the number of photons**
  - ➢ Bias ***cannot*** be eliminated just by shooting more photons!

Local Accuracy

- **Mean Square Error (MSE)**

$$MSE_x(\hat{f}) = E\{f(x) - \hat{f}(x)\}^2 =$$

$$\underbrace{\{E\hat{f}(x) - f(x)\}^2}_{\text{bias}^2} + \underbrace{\operatorname{var}\hat{f}(x)}_{\text{variance}}$$

$$\text{bias} = E\hat{f}(x) - f(x) = \frac{1}{h}\int K\left(\frac{x-y}{h}\right) f(y)dy - f(x)$$

$$\operatorname{var}\hat{f}(x) = \frac{1}{nh^2}\left\{\int K^2\left(\frac{x-y}{h}\right) f(y)dy - \left[\int K\left(\frac{x-y}{h}\right) f(y)dy\right]^2\right\}$$

- **The bias and variance equations depend on the unknown** $f(x)$ **and are not very intuitive**

Global Accuracy

- **Mean Integrated Square Error (MISE)**

$$MISE(\hat{f}) = E\int\{f(x) - \hat{f}(x)\}^2 dx = \int E\{f(x) - \hat{f}(x)\}^2 dx =$$

$$\int MSE_x(\hat{f})dx = \int\{E\hat{f}(x) - f(x)\}^2 dx + \int \mathrm{var}\ \hat{f}(x)dx$$

- **For a symmetric kernel function K such that:**

$$\int K(t)dt = 1 \quad \int tK(t)dt = 0 \quad \int t^2 K(t)dt = k_2$$

$$\int bias_h^2(x)dx = \int\left[\frac{1}{h}\int K\left(\frac{x-y}{h}\right)f(y)dy - f(x)\right]^2 dx \approx \frac{1}{4}h^4 k_2^2 \int [f''(x)]^2 dx$$

$$\int \mathrm{var}\ \hat{f}(x)dx \approx \frac{1}{nh}\int [K(t)]^2 dt$$

- **The bias and random error depend strongly on each other. The bias can be reduced when the region h in which photons are processed to evaluate $\hat{f}(x)$ is decreased. This results in the increasing random error.**

Optimal Bandwidth

- **Through minimizing MISE in respect to** h

$$\int bias_h^2(x)dx + \int \operatorname{var} \hat{f}(x)dx \approx \frac{1}{4}h^4 k_2^2 \int [f''(x)]^2 dx + \frac{1}{nh}\int [K(t)]^2 dt$$

$$h_{opt} = \left(\frac{\int K^2(t)dt}{k_2^2 n \int [f''(x)]^2 dx} \right)^{\frac{1}{5}}$$

- h_{opt} **converges to zero as the number of photons** n **increases**
- h_{opt} **depends directly on the density function** $f(x)$
  - **The second derivative** $f''(x)$ **is a measure of the rapidity of fluctuations in** $f(x)$ **and smaller** h **should be chosen for more rapidly fluctuating densities.**

Optimal Kernel

- **By substituting the value of h_{opt} in**

$$\int bias_h^2(x)dx + \int \operatorname{var} \hat{f}(x)dx \approx \frac{1}{4} h_{opt}^4 k_2^2 \int [f''(x)]^2 dx + \frac{1}{nh_{opt}} \int [K(t)]^2 dt \approx$$

$$\frac{5}{4} C(K) \left(\frac{1}{n^4} \int [f''(x)]^2 dx \right)^{\frac{1}{5}} \quad \text{where} \quad C(K) = k_2^{2/5} \left(\int K^2(t) dt \right)^{\frac{4}{5}}$$

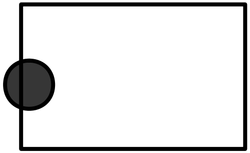
- **The optimal kernel can be found by minimizing** $C(K)$
  - **Epanechnikov**

$$\frac{3}{4\sqrt{5}}(1 - \frac{1}{5}t^2) \quad \text{for} \quad |t| < \sqrt{5}$$

- **Even simple kernels such as gaussian or cylindrical lead to only small increase of density estimation error.**

Boundary Bias

- **Problem for kernel methods**
  - Darkening near the surface boundary
- **Solutions**
  - Replicate photon collision points by virtually reflecting them across the boundary
  - Extending surface by some virtual margin at which photon collisions are still registered but it does not occlude the original photon path
  - Normalizing kernels
  - More advanced: local linear density estimation based on locally-weighted linear least-squares regression



Density Estimation in Global Illumination

Main problems:

- **The parameters of density estimation are usually decided *globally* for the whole scene or surfaces**
  - **This may result in uncontrolled smoothing/noise for complex illumination patterns.**
- **Local estimate the lighting reconstruction error would be useful to find h_{opt} for a given scene region.**

Local Error Estimate

- **The lighting reconstruction error e in some region *H* can be measured by the standard norm:** $e = L_2(f(x) - \hat{f}(x))$
- **Rough estimate of this error:**

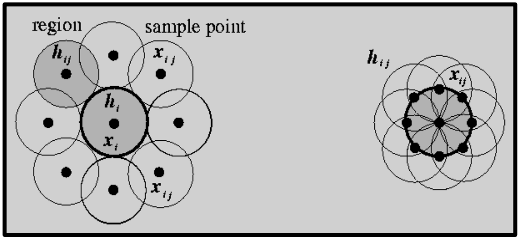
$$\hat{e}^2 \le M \frac{\sum_i \sum_j^J \left(\hat{f}(x_i) - \hat{f}(x_{ij})\right)^2 \Delta h_i}{JH}$$

Δh_i : elementary surface area

$H = \sum_i \Delta h_i$

$\hat{f}(x_i)$: estimate for region h_i centered around x_i

$h_i = h_{ij}$: random error kept on the same level



Adaptive Density Estimation

Two stage approach:

- **Use histograms to find crude approximation of lighting distribution (photon density) function**
- **Based on this function find local h_{opt}**
- **Use more precise density estimation methods such as**
  - **kernel methods with adaptively changing h_{opt}**
  - **nearest neighbor methods with adaptively changing k_{opt}**

Variable Kernel Method (1)

- **Illumination textures (*IT*) used as histogram bins Δh_i for collecting photons**
  - Each texel stores the number of colliding photons
  - Bias error: Photon collision points are discretized to the nearest texel location.
- **The texel grid of IT is used to build summed area table (*SAT*) in which photons are summed up**
  - Density estimation for any rectangular region is very fast
- **For every texel i of *IT* photon density is estimated L times for the increasing size of the region h_i^l in which photons are counted using *SAT*.**
- **For every texel i of *IT*, and for every h_i^l, $\hat{e}^2$ is estimated. We search for the minimal $\hat{e}^2$ to decide h_{opti}.**

Variable Kernel Method (2)

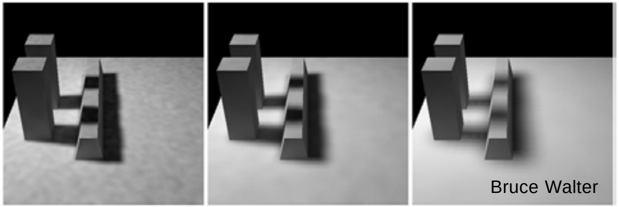
- **Based on the size of** h_{opti} **the optimal radius kernel** R_{opti} **is computed for each texel** i
- **To remove outliers the median filter is applied to** R_{opti}**.**
- **Since** R_{opti} **is stored only for texel locations of *IT*, bilinear interpolation is used to find the radius at each photon location.**
- **To avoid energy leaks for kernels crossing *IT* boundaries, the kernel can be re-scaled so that its *IT* volume sums to unity.**

Rendering Illumination Textures

- **The final step of *IT* processing does not depend on the density estimation method.**
- **If the direct lighting is computed as the illumination texture using the deterministic approach, then total illumination must be summed up**
  - This may require re-sampling one texture to the resolution of the other.
- **Then illumination is converted to luminance taking into account the surface reflectance function, and transformed into displayable RGB using a tone reproduction function.**

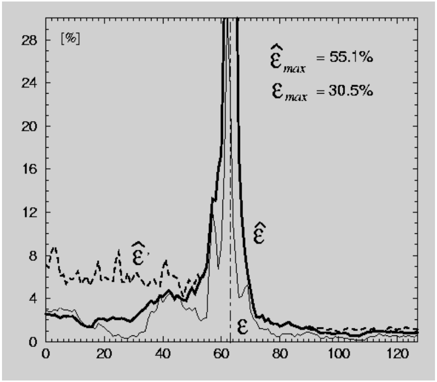
Lighting Textures: Color Treatment

- **The most straightforward solution is to process independently each color channel**
  - some "uncorrelated color oscillations" for small h_i are possible.



- **Another solution:**
  - Store for every photon a record of its normalized RGB components.
  - Modify this record during photon tracing as a result of light reflection.
  - Process *SAT* with total photons energy to compute R_{opt}.
  - For the final density estimation scale kernel for each RGB component. The kernel size R_{opt} remains the same for all color channels.

Estimated vs. Exact Reconstruction Errors

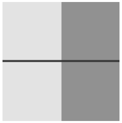


Distribution of the locally measured e and $\hat{e}$ **for the *adaptive nearest neighbor* method. The errors are measured along the scanline (128 pixels long) marked in the figure below. The average number of photons per texel is ~6.1.**

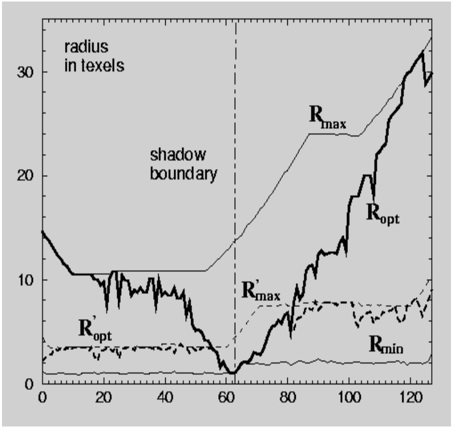
e : actual error

$\hat{e}$: error estimate

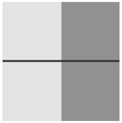
$\hat{e}'$: error estimate for $k_{\max} = 500$ photons



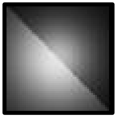
Distribution of R_{opt}

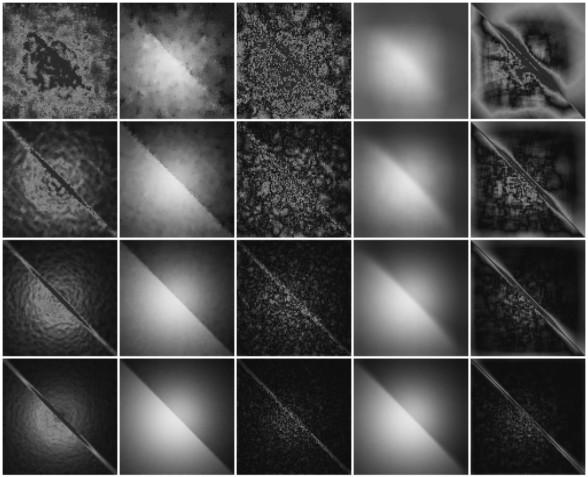


Distribution of the optimal region size R_{opti} **along the scanline (128 pixels long) marked in the figure below. The average number of photons per pixel is ~6.1.**

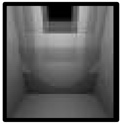


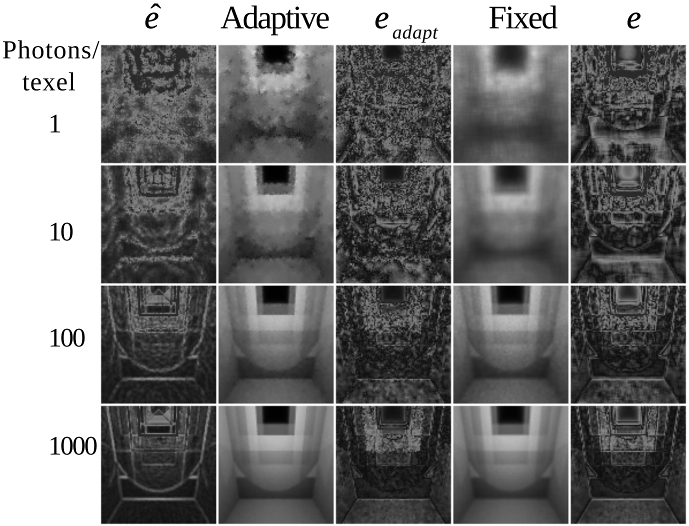
Adaptive vs. Fixed #Neighbor Photons





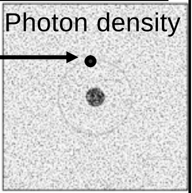
Adaptive vs. Fixed #Neighbor Photons

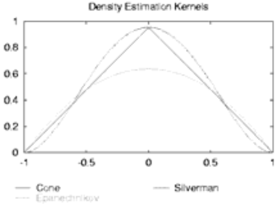


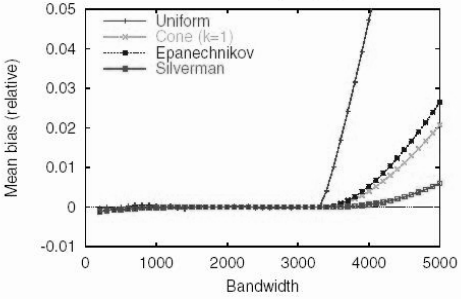


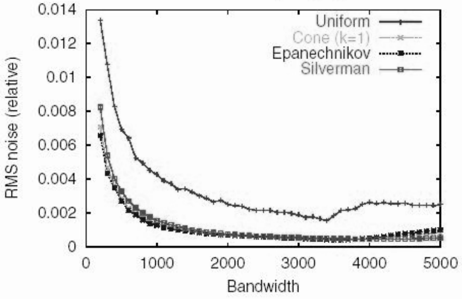
Bias Compensation for the Nearest Neighbor Method

- **Highlight case study**
  - Illumination estimate at the red point
  - Analysis of bias and noise as a function of increased number of nearest neighbor photons









Bias Compensation for the Nearest Neighbor Method

- **Algorithm**
  - For each point *x* perform estimate of illumination $\hat{f}(x,1),\ldots,\hat{f}(x,N_{\min})$ for the number of nearest photons ranging from 1,…, N_{min}
  - Compute the expected value and variance of *f(x,j)* using some weighting function *w(j)*

$$m[\hat{f}(x,N_P)] = \sum_{j=1}^{N_{\min}} w(j)\cdot\hat{f}(x,j)$$

$$\hat{s}^2[\hat{f}(x,N_P)] = m[\hat{f}^2(x,N_P)] - m^2[\hat{f}(x,N_P)]$$

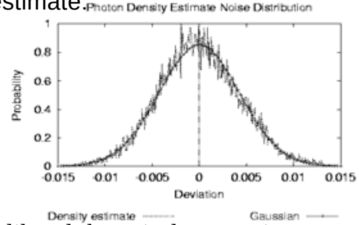
  - Recursively split the interval *[N_{min}, N_{max}]* at N_{mid}*=(N_{min}- N_{min})/2* and decide which interval to choose based on a density estimate:

$\hat{f}(x,N_{mid})$ using N_{mid} photons

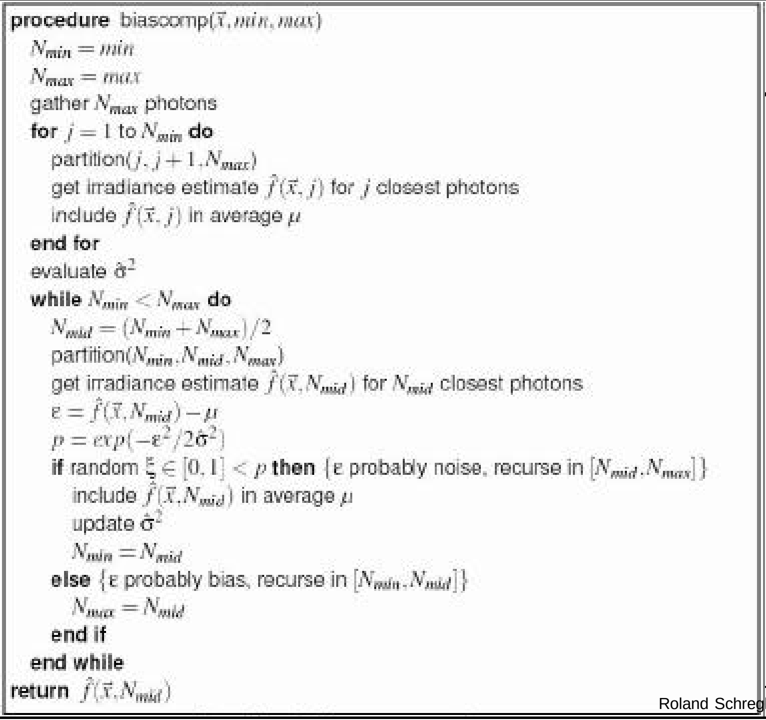
$e[\hat{f}(x,N_{mid})] = \hat{f}(x,N_{mid}) - m[\hat{f}(x,N_P)]$

with the probability p that m is attributed to noise:

$p = e^{-e^2[\hat{f}(x,N_{mid})]/2\hat{s}^2[\hat{f}(x,N_P)]}$

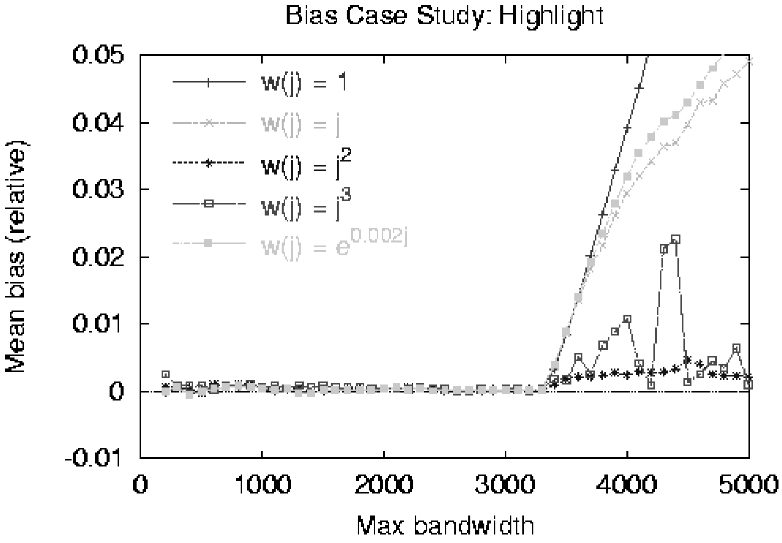


where based on the central limit theorem p is the likelihood that e is due to noise

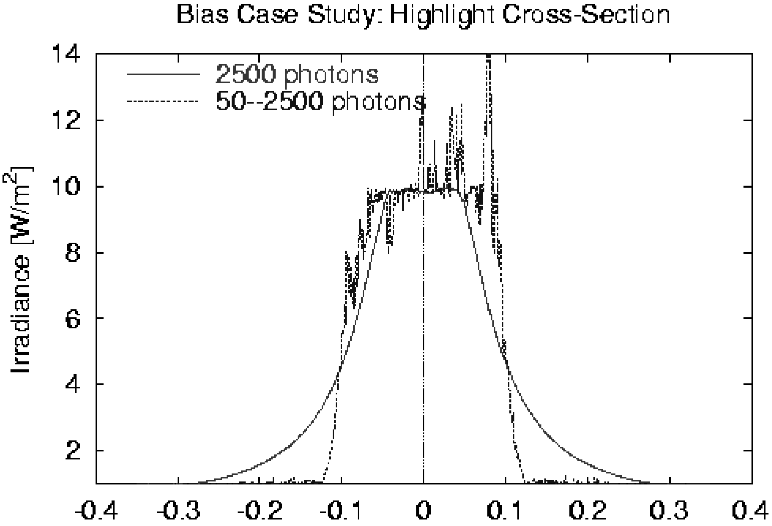


```
procedure biascomp($\vec{x}, min, max$)
  $N_{min} = min$
  $N_{max} = max$
  gather $N_{max}$ photons
  for $j = 1$ to $N_{min}$ do
    partition($j, j+1, N_{max}$)
    get irradiance estimate $\hat{f}(\vec{x}, j)$ for $j$ closest photons
    include $\hat{f}(\vec{x}, j)$ in average $\mu$
  end for
  evaluate $\hat{\sigma}^2$
  while $N_{min} < N_{max}$ do
    $N_{mid} = (N_{min} + N_{max})/2$
    partition($N_{min}, N_{mid}, N_{max}$)
    get irradiance estimate $\hat{f}(\vec{x}, N_{mid})$ for $N_{mid}$ closest photons
    $\varepsilon = \hat{f}(\vec{x}, N_{mid}) - \mu$
    $p = exp(-\varepsilon^2 / 2\hat{\sigma}^2)$
    if random $\xi \in [0,1] < p$ then $\{\varepsilon$ probably noise, recurse in $[N_{mid}, N_{max}]\}$
      include $\hat{f}(\vec{x}, N_{mid})$ in average $\mu$
      update $\hat{\sigma}^2$
      $N_{min} = N_{mid}$
    else $\{\varepsilon$ probably bias, recurse in $[N_{min}, N_{mid}]\}$
      $N_{max} = N_{mid}$
    end if
  end while
return $\hat{f}(\vec{x}, N_{mid})$
```

Bias Compensation for the Nearest Neighbor Method

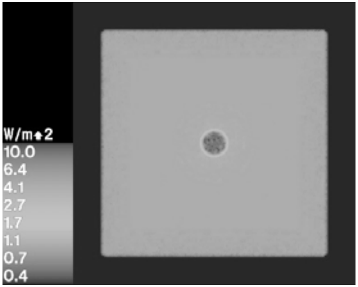


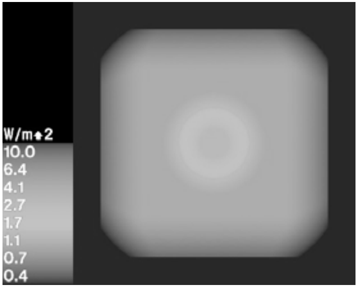
Bias Compensation for the Nearest Neighbor Method



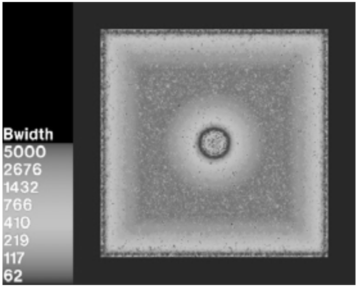
Bias Compensation for the Nearest Neighbor Method

Bias compensation with adaptive bandwidth for 50-5,000 nearest photons

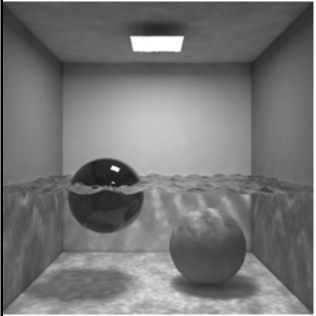


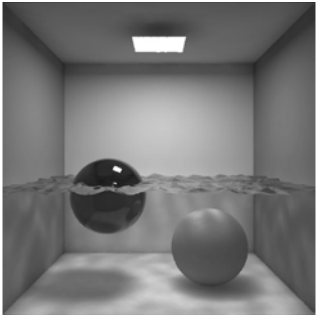


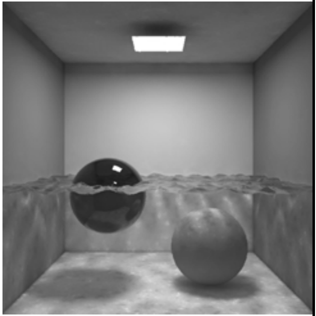
Fixed bandwidth for 5,000 nearest photons



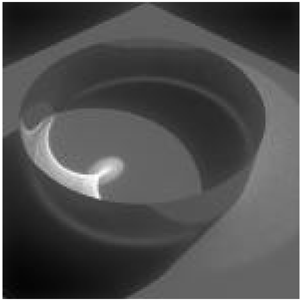
Bias Compensation for the Nearest Neighbor Method


Fixed bandwidth with 50 nearest photons

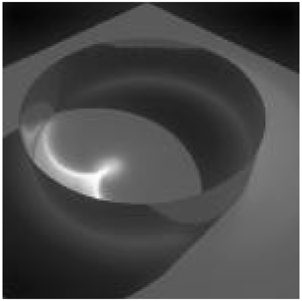

Fixed bandwidth with 500 nearest photons


Adaptive bandwidth with 50-500 nearest photons

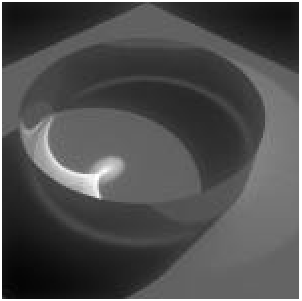
Bias Compensation for the Nearest Neighbor Method



Fixed bandwidth with 50 nearest photons



Fixed bandwidth with 2,000 nearest photons



Adaptive bandwidth with 50-2,000 nearest photons

Simple Algorithm Example

- **Histogram approach:**
  - Use fine mesh as bins and count the number of collisions
  - Keep separate counters for photons arriving directly and indirectly from light sources
    - At later stages of computation replace density based direct lighting reconstruction by deterministically computed lighting
      - adaptive mesh subdivision – view-independent approach
      - for each pixel – view-dependent approach (better quality)
  - Continuous Random Walks guarantees the rendering equation solution, but the final lighting stored in the mesh is Lambertian only
  - Graphics hardware can be used to display mesh-reconstructed lighting, and walkthrough animation is possible at any stage of computation
  - The final gather step is not required to obtain images of good quality
    - Problem: mesh elements with small number of colliding photons may look noisy

Illumination Maps Filtering

- **Remove noise at expense of increasing bias**
- **Average photon collision number for neighboring mesh elements**
- **Use static, balanced kd-tree with mesh vertices to search neighboring mesh elements**
  - Mesh topology does not matter
  - Mesh normal vectors should be roughly aligned
- **Adaptive selection of density estimation filter support based on mathematically-sound local statistic measures of illumination variation:**

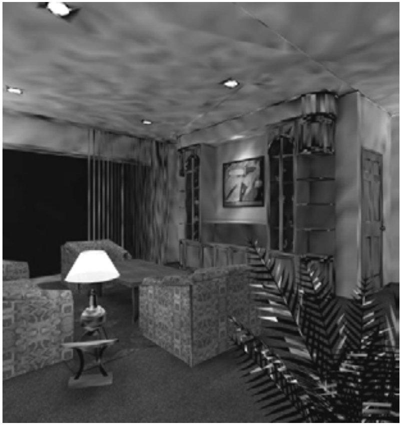
$$\text{relative standard deviation } s_{\%} = \frac{100\%}{\sqrt{n_i}}$$

$$\text{where } n_i : \text{local number of collisions}$$

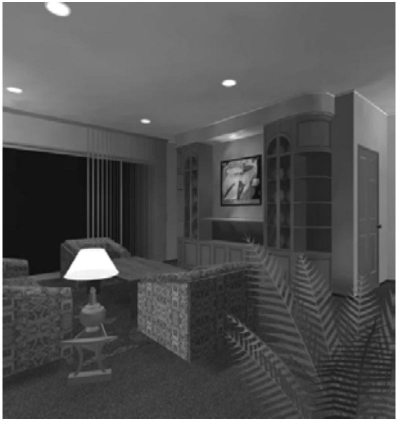
  - As solution converges the local filter support shrinks reducing bias.

Filtering Example (1)

Solution after 10 sec. of computation
R10,000 195 MHz processor



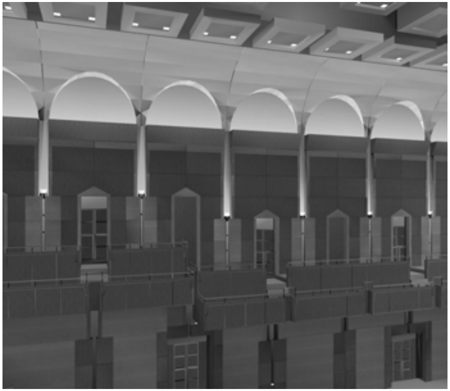
Without filtering

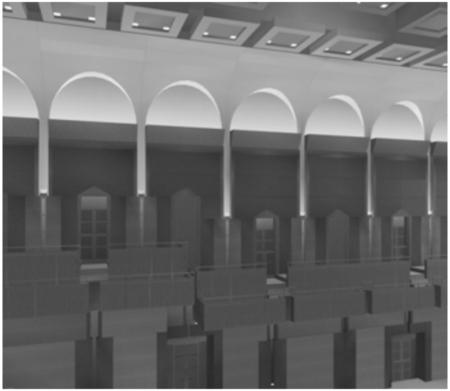


With filtering

Filtering Example (2)

Solution after 30 sec. of computation
R10,000 195 MHz processor


Without filtering


With filtering

Scene complexity: 22,314 polygons, and 581 light sources

Random Error Estimate

- **Split photons into two halves A and B**
  - For example A: even photons and B:odd photons

$$\hat{f}_A(x) = \frac{2}{n}\sum_{i\in A} v(X_i) \quad \text{and} \quad \hat{f}_B(x) = \frac{2}{n}\sum_{i\in B} v(X_i)$$

$$\hat{f}(x) = \frac{1}{2}(\hat{f}_A(x) + \hat{f}_B(x))$$

$$\hat{e}(x) = \frac{1}{2}(\hat{f}_A(x) - \hat{f}_B(x))$$

$$E\hat{e}(x) = 0$$

$$\operatorname{var}\hat{e}(x) = \frac{1}{2}(\operatorname{var}\hat{f}_A(x) + \operatorname{var}\hat{f}_B(x)) = \operatorname{var}\hat{f}(x)$$

$$\text{Unbiased estimate}: E\hat{e}(x)^2 = \operatorname{var}\hat{e}(x) = \operatorname{var}\hat{f}(x)$$

$$MISE = E\int \hat{e}(x)^2 dx$$

Case Study Scene - Various Views

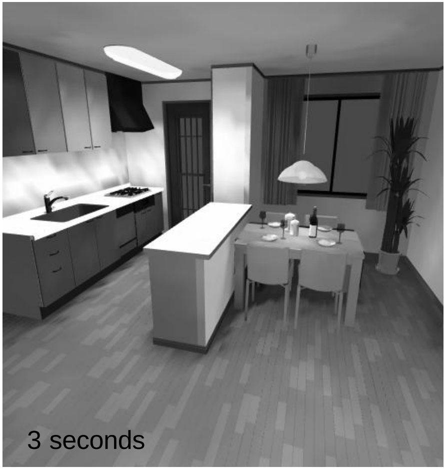
Geometry complexity: 131,700 polygons that are tessellated into 350,600 mesh elements

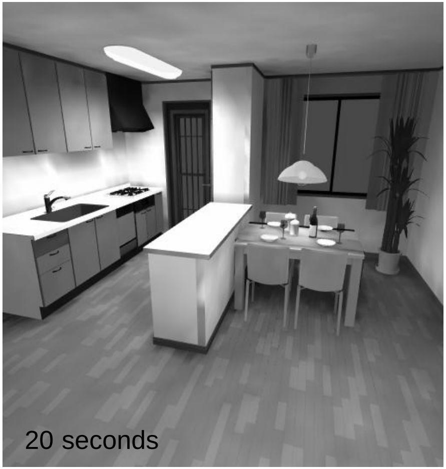
Lighting complexity: 8 luminaires

Timings measured for a R10,000 195Mhz processor

Progressive Rendering Example

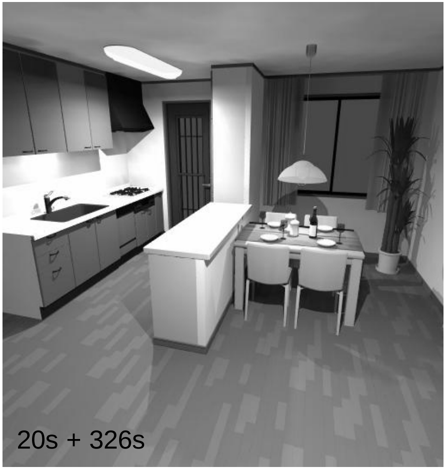
Density estimation only

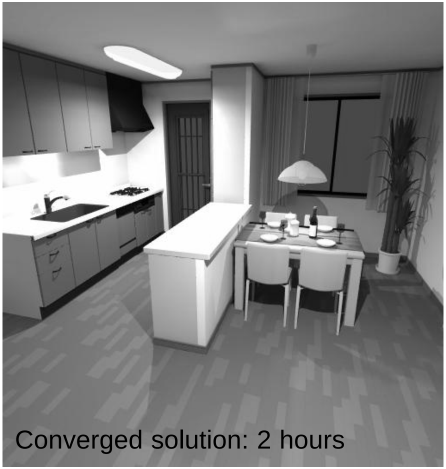




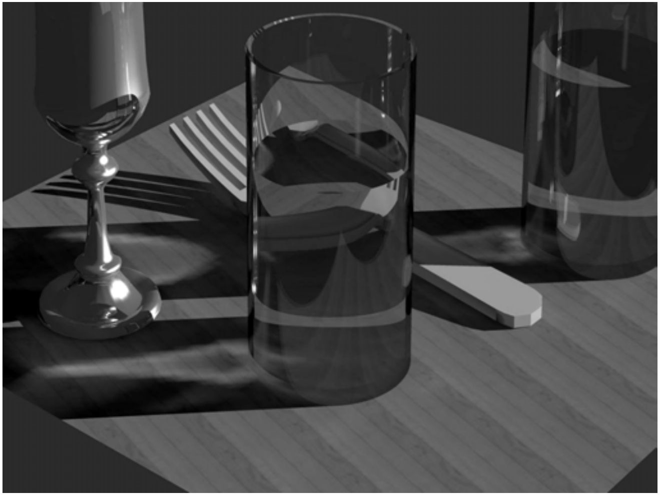
Progressive Rendering Example

$$\text{density estimation} + \begin{cases} \text{deterministic direct lighting (left)} \\ \text{ray tracing (right)} \end{cases}$$


20s + 326s

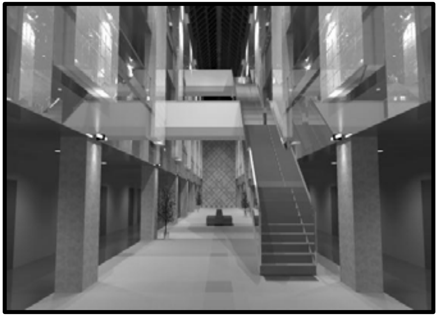

Converged solution: 2 hours

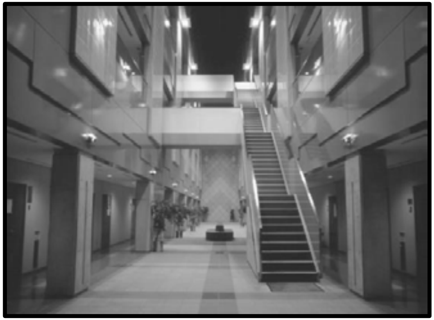
General BRDF and Caustics



Validation Experiment

- **An atrium of the University of Aizu:**
  - LEFT: ray traced image with indirect lighting computed via photon bucketing
  - RIGHT: photograph



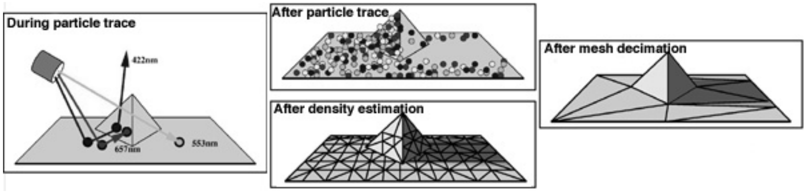


More Advance Algorithm Example

- **Spectral color representation**
- **Kernel based density estimation**
  - Adaptive kernel support – fixed kernel size h_i for each surface A_i

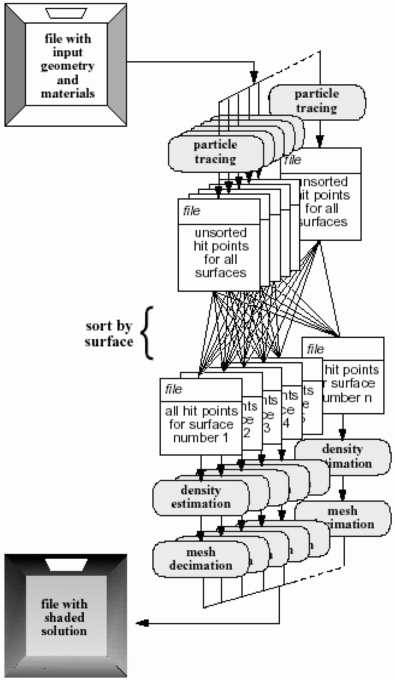
$$h_i = \sqrt{\frac{CA_i}{n_i p}}$$

  - Reconstruction performed at vertices of dense mesh
- **Mesh decimation**
  - Perception-driven (the Weber law is used to check visibility of luminance changes across surface)



Processing Flow

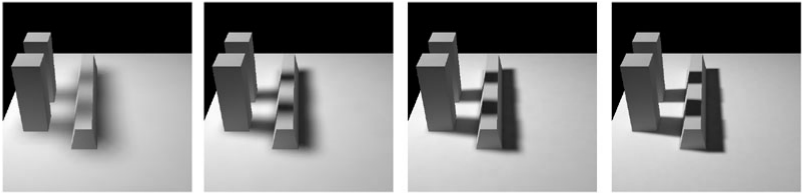
- **Software naturally divides into independent modules**
- **Computation easy to perform in parallel**
  - Communication required only at the photon sorting stage
    - well studied problem


file with input geometry and materials
particle tracing
particle tracing
file
unsorted hit points for all surfaces
file
unsorted hit points for all surfaces
sort by surface
file
all hit points for surface number 1
file
hit points for surface number n
density estimation
density estimation
mesh decimation
mesh decimation
file with shaded solution

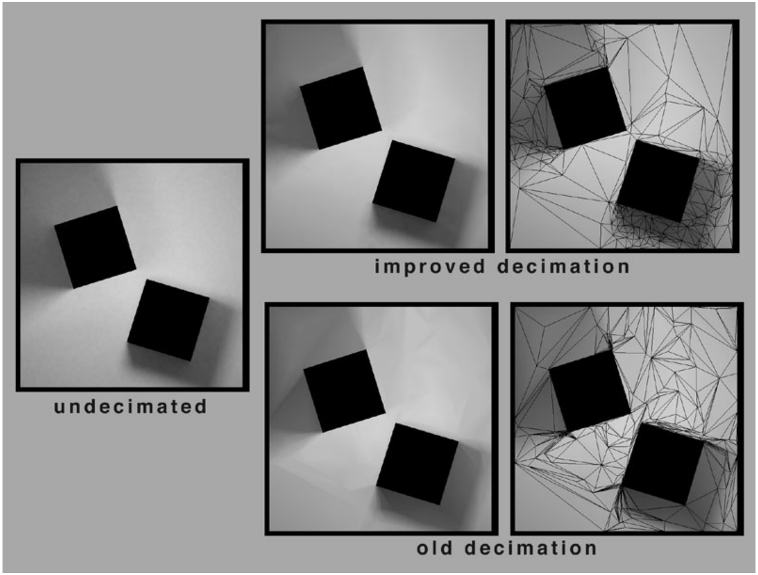
Solution Convergence

- **From left to right: increasing number of particles and corresponding reduction of kernel bandwidth**

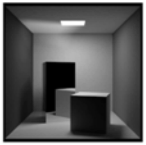
$$h_i = \sqrt{\frac{CA_i}{n_i p}}$$

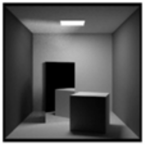


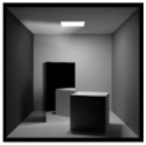
Decimation

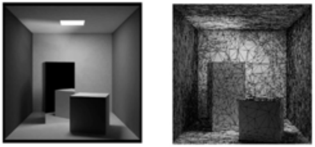


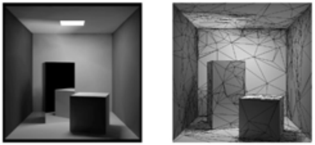
Comparison with other Rendering Techniques


density estimation

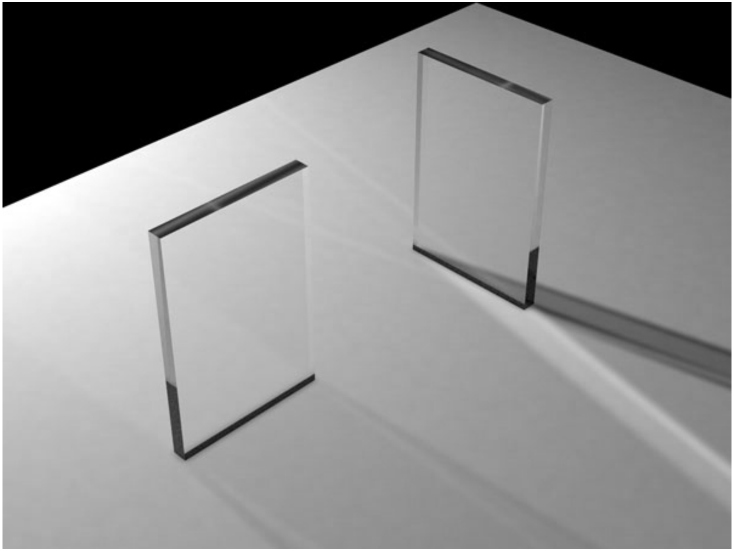

path tracing


Radiance


conservative decimation

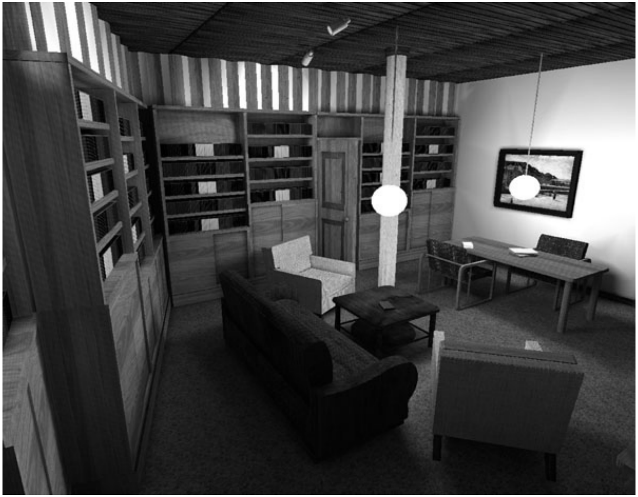

liberal decimation

Realism



Performance: Pentium II, 400MHz

- **#polygons 13,729**
- **Particle tracing 18.6hrs**
- **300,000,000 collisions**
- **Sorting 0.7hrs**
- **Density estimation 2,6hrs**
- **#triangles 3,441,944**



Acknowledgements

- **I would like to thank Philippe Bekaert, Roland Schregle, Bruce Walter, and Karol Myszkwski for sharing with me some slides and images that I used during this lecture.**