
CS580: Monte Carlo Ray Tracing:

Sung-Eui Yoon
(윤성익)

<http://sgvr.kaist.ac.kr/~sungeui/GCG>

KAIST

The KAIST logo consists of the letters 'KAIST' in a bold, blue, sans-serif font. Below the text is a light blue, horizontal oval shape that serves as a shadow or base for the letters.

Talk Evaluations

- **Do not evaluate based on:**
 - English
 - Relative ranking among presenters

- **Evaluate based on:**
 - His/her message
 - His/her understanding, and delivery
 - Achievement of the team for the project presentation

Project Guidelines: Project Topics

- **Any topics related to the course theme are okay**
 - **You can find topics by browsing recent papers**

Expectations

- **Mid-term project presentation**
 - **Introduce problems and explain why it is important**
 - **Give an overall idea on the related work**
 - **Explain what problems those existing techniques have**
 - **(Optional) explain how you can address those problems**
 - **Explain roles of each member**

Expectations

- **Final-term project presentation**
 - **Cover all the materials that you talked for your mid-term project**
 - **Present your ideas that can address problems of those state-of-the-art techniques**
 - **Give your qualitatively (or intuitive) reasons how your ideas address them**
 - **Also, explain expected benefits and drawbacks of your approach**
 - **(Optional) backup your claims with quantitative results collected by some implementations**
 - **Explain roles of each members**

A few more comments

- **Start to implement a paper, if you don't have any clear ideas**
 - **While you implement it, you may get ideas about improving it**
- **Utilize any existing materials (codes) with proper ack.**

Project evaluation sheet

You name:

ID:

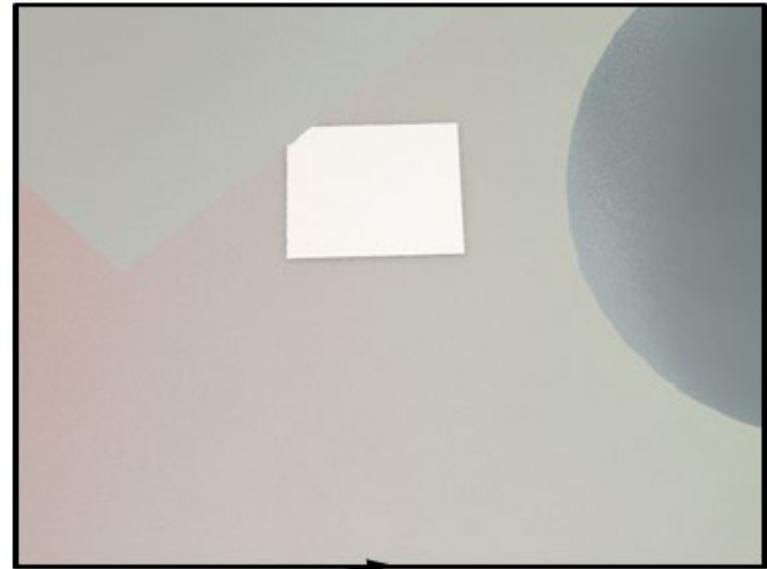
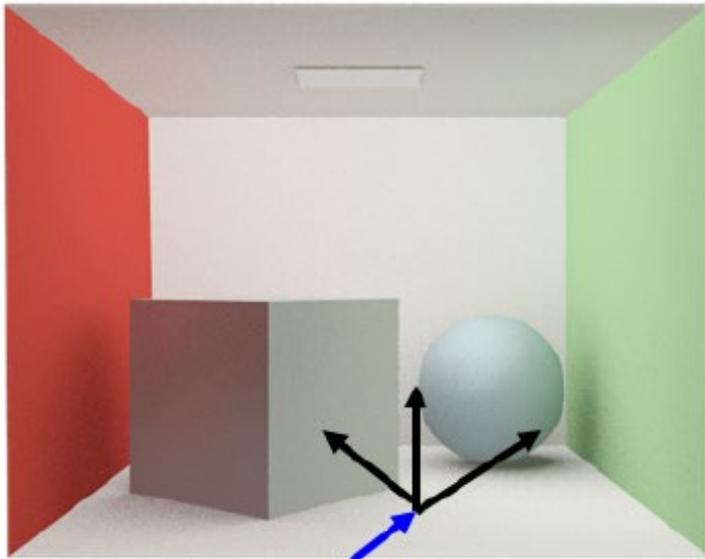
Score table: higher score is better.

Speaker	Novelty of the project and idea (1 ~ 5)	Practical benefits of the method (1 ~ 5)	Completeness level of the project (1 ~ 5)	Total score (3 ~ 15)	Role of each student is clear and well balanced? (Yes or No)
XXX					
YYY					

Class Objectives

- **Understand a basic structure of Monte Carlo ray tracing**
 - **Russian roulette for its termination**
 - **Path tracing**
 - **Biased techniques**

Rendering Equation

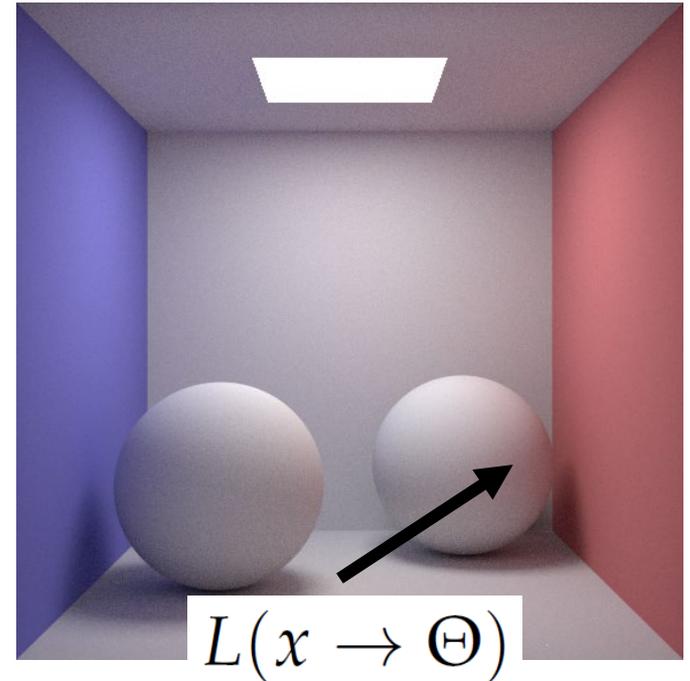


Incoming radiance on the hemisphere

$$L_r(x \rightarrow \Theta) = \int_{\Psi} L(x \leftarrow \Psi) f_r(x, \Psi \rightarrow \Theta) \cos \theta_x dw_{\Psi}$$

Evaluation

- **To compute** $L(x \rightarrow \Theta)$:
 - **Check** $L_e(x \rightarrow \Theta)$
 - **Evaluate** $L_r(x \rightarrow \Theta)$



$$L_r(x \rightarrow \Theta) = \int_{\Psi} L(x \leftarrow \Psi) f_r(x, \Psi \rightarrow \Theta) \cos \theta_x d\omega_{\Psi}$$

Evaluation

- Use Monte Carlo
- Generate random directions on hemisphere Ψ using pdf $p(\Psi)$

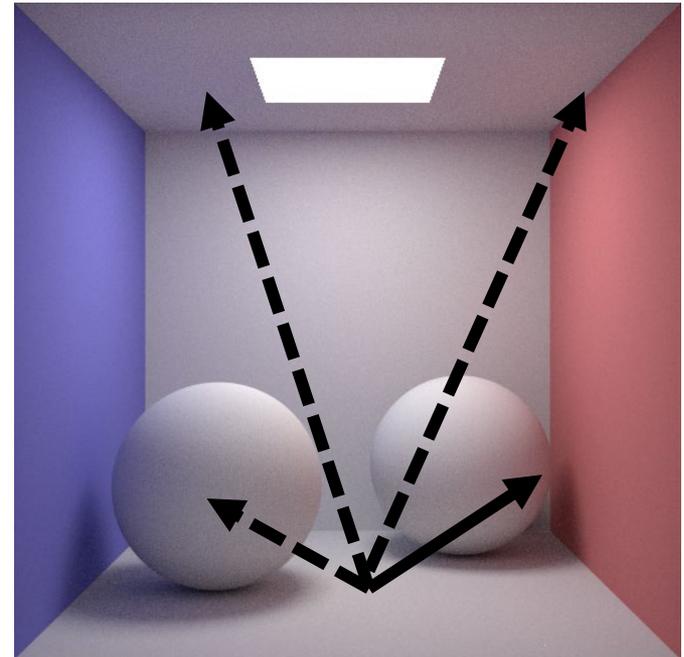
$$L_r(x \rightarrow \Theta) = \int_{\Psi} L(x \leftarrow \Psi) f_r(x, \Psi \rightarrow \Theta) \cos \theta_x d\omega_{\Psi}$$

$$\hat{L}_r(x \rightarrow \Theta) = \frac{1}{N} \sum_{i=1}^N \frac{L(x \leftarrow \Psi_i) f_r(x, \Psi_i \rightarrow \Theta) \cos \theta_x}{p(\Psi_i)}$$

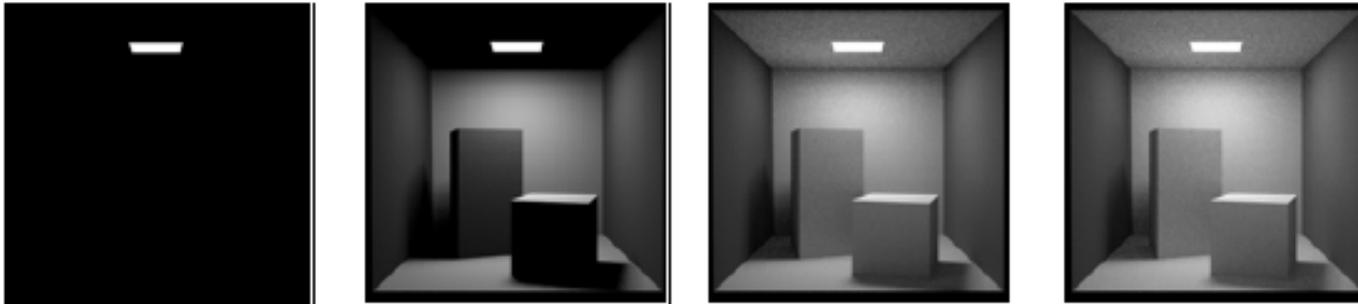
- How about $L(x \leftarrow \Psi_i)$?

Evaluation

- How about $L(x \leftarrow \Psi_i)$?
- Perform ray casting backward
- Compute radiance from those visible points to x
 - Assume reciprocity
- Recursively perform the process
 - Each additional bounce supports one more indirect illumination



When to end recursion?



From kavita's slides

- **Contributions of further light bounces become less significant**
 - Max recursion
 - Some threshold for radiance value
- **If we just ignore them, estimators will be biased**

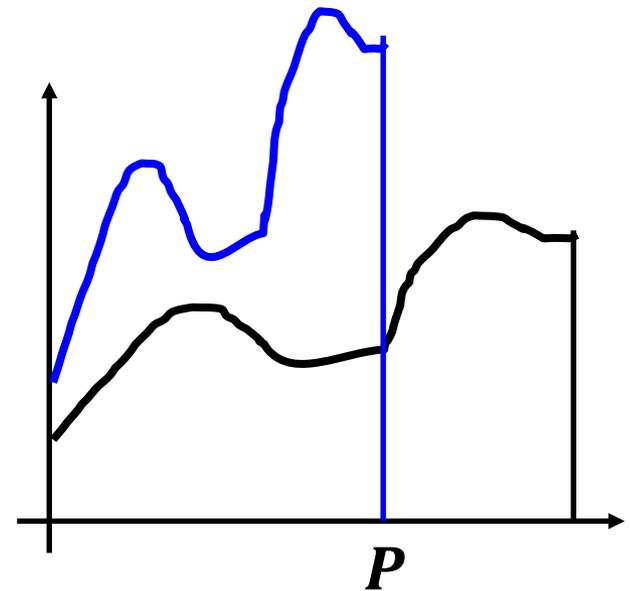
Russian Roulette

- **Integral: Substitute $y = Px$**

$$I = \int_0^1 f(x) dx = \int_0^P \frac{f(y/P)}{P} dy.$$

- **Estimator**

$$\hat{I}_{\text{roulette}} = \begin{cases} \frac{f(x_i)}{P} & \text{if } x_i \leq P, \\ 0 & \text{if } x_i > P. \end{cases}$$



- **Variance?**

Russian Roulette

- **Pick absorption probability, $\alpha = 1-P$**
 - **Recursion is terminated**

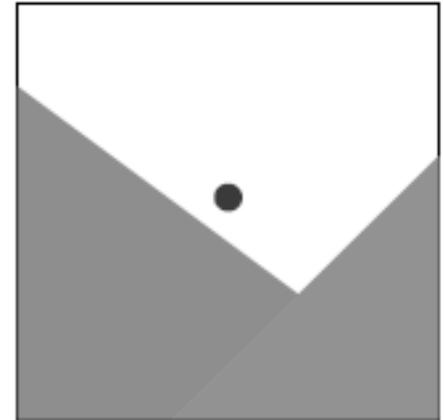
- **P is commonly to be equal to the reflectance of the material of the surface**
 - **Water: 7%**
 - **Snow: 65%**

Algorithm so far

- **Shoot primary rays through each pixel**
- **Shoot indirect rays, sampled over hemisphere**
- **Terminate recursion using Russian Roulette**

Pixel Anti-Aliasing

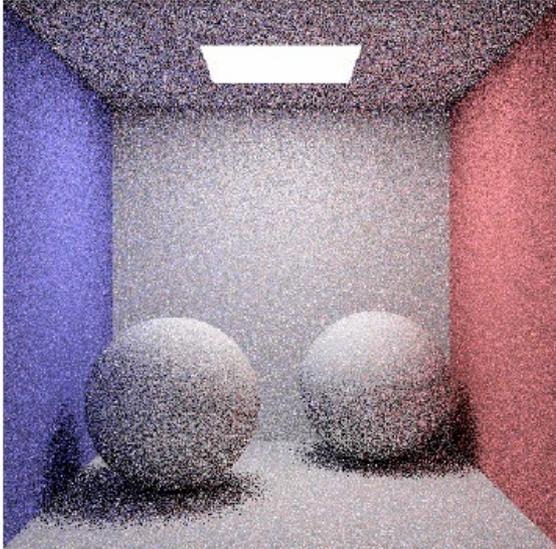
- **Compute radiance only at the center of pixel**
 - **Produce jaggies**
- **We want to evaluate using MC**
- **Simple box filter**
 - **The averaging method**



Stochastic Ray Tracing

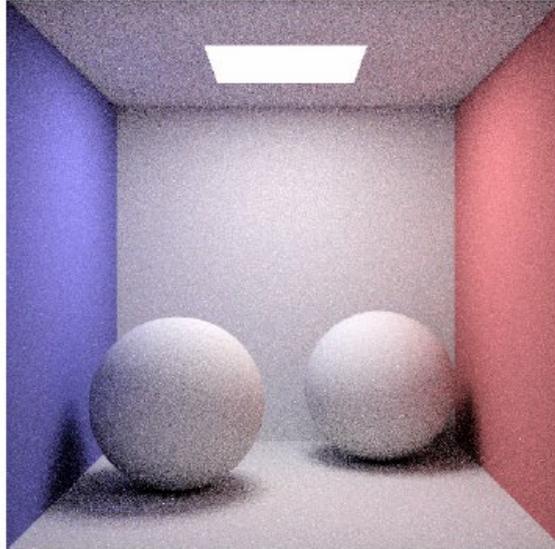
- **Parameters**
 - **Num. of starting ray per pixel**
 - **Num. of random rays for each surface point (branching factor)**
- **Path tracing**
 - **Branching factor = 1**

Path Tracing

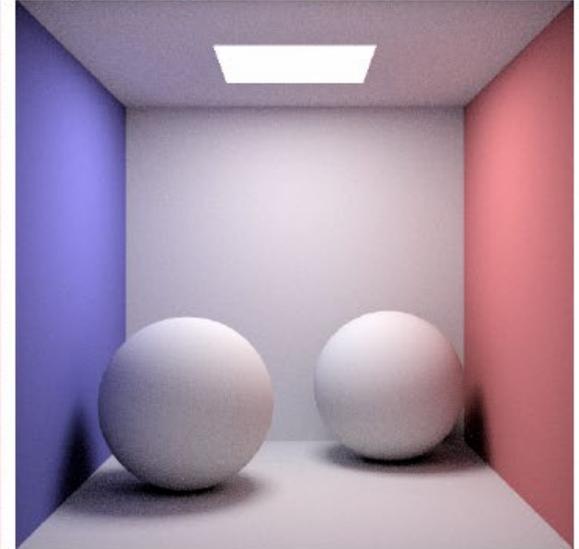


1 spp

(samples per pixel)



4 spp



16 spp

- **Pixel sampling + light source sampling folded into one method**

Algorithm so far

- **Shoot primary rays through each pixel**
- **Shoot indirect rays, sampled over hemisphere**
 - **Path tracing shoots only 1 indirect ray**
- **Terminate recursion using Russian Roulette**

Performance

- **Want better quality with smaller # of samples**
 - **Fewer samples/better performance**
 - **Quasi Monte Carlo: well-distributed samples**
 - **Adaptive sampling**
- **See my book, if you are interested**

Some Example



**Uniform sampling
(64 samples per pixel)**



Adaptive sampling



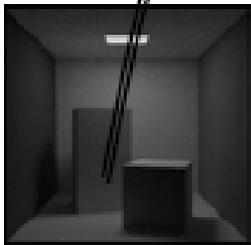
Reference

Importance Sampling

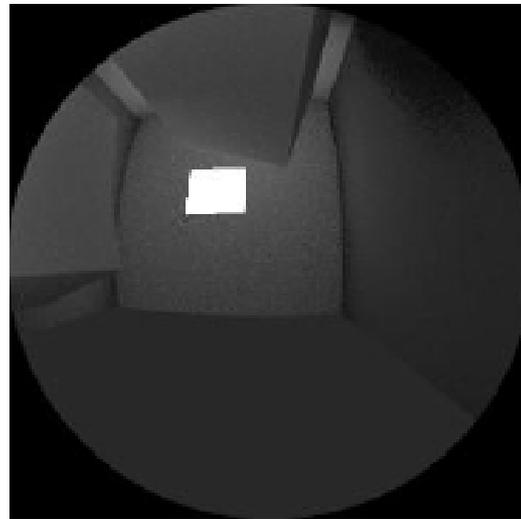
$$L(x \rightarrow \Theta) = L_e(x \rightarrow \Theta) + \int_{\Omega_x} f_r(\Psi \leftrightarrow \Theta) \cdot L(x \leftarrow \Psi) \cdot \cos(\Psi, n_x) \cdot d\omega_\Psi$$



Radiance from light sources + radiance from other surfaces

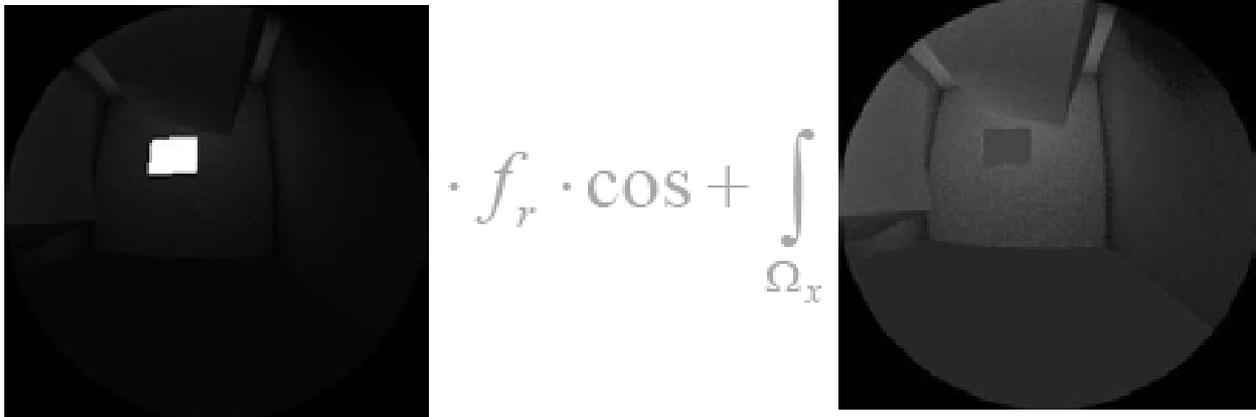


$$= L_e + \int_{\Omega_x} \text{[Image]} \cdot f_r \cdot \cos$$



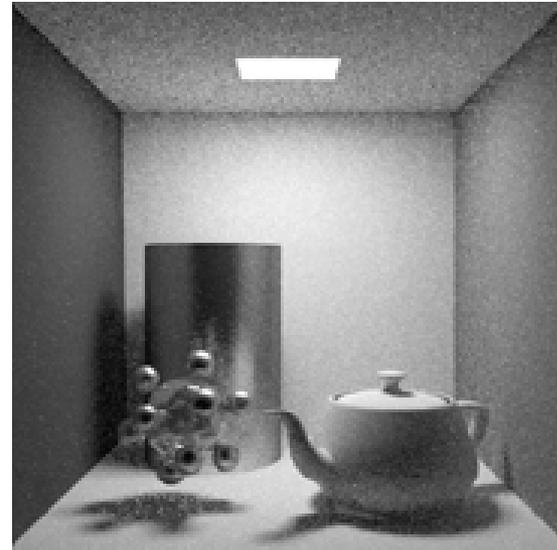
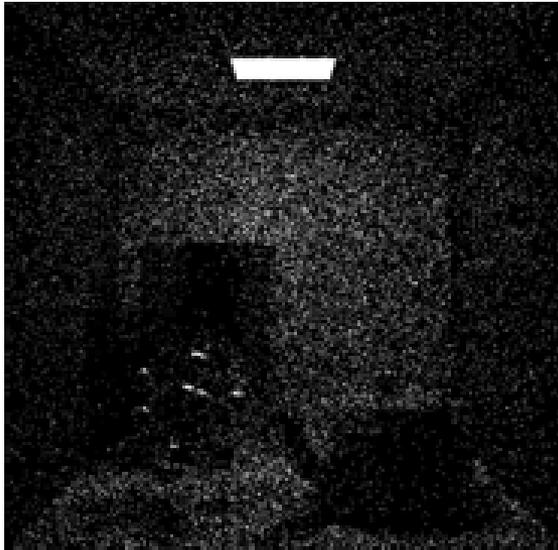
Importance Sampling

$$L(x \rightarrow \Theta) = L_e + L_{direct} + L_{indirect}$$

$$= L_e + \int_{\Omega_x} \text{img}_1 \cdot f_r \cdot \cos + \int_{\Omega_x} \text{img}_2 \cdot f_r \cdot \cos$$


- So ... sample direct and indirect with separate MC integration

Comparison

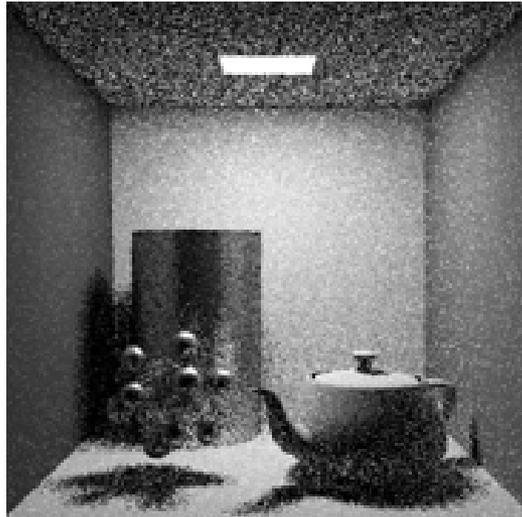


From kavita's slides

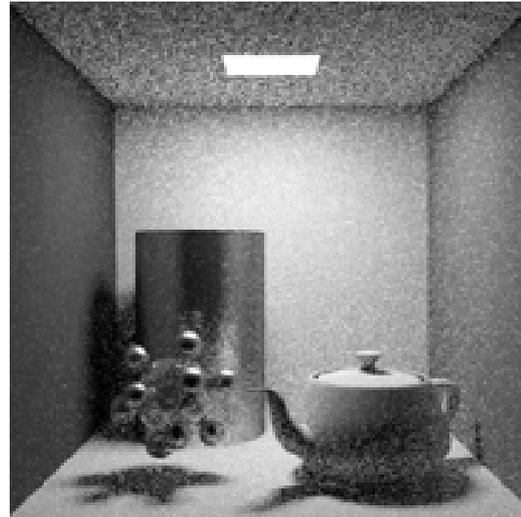
- **With and without considering direct illumination**
 - **16 samples / pixel**

Rays per pixel

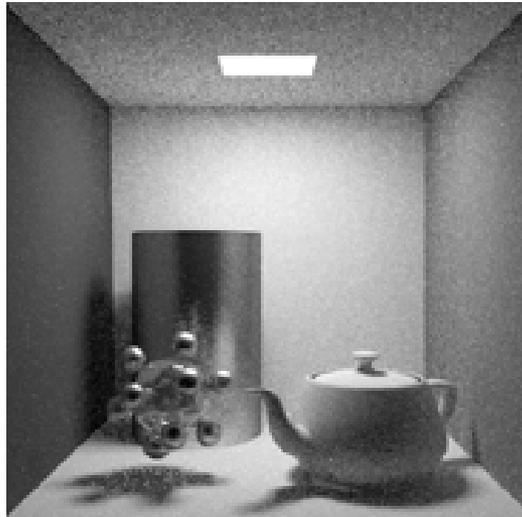
1 sample/
pixel



4 samples/
pixel



16 samples/
pixel



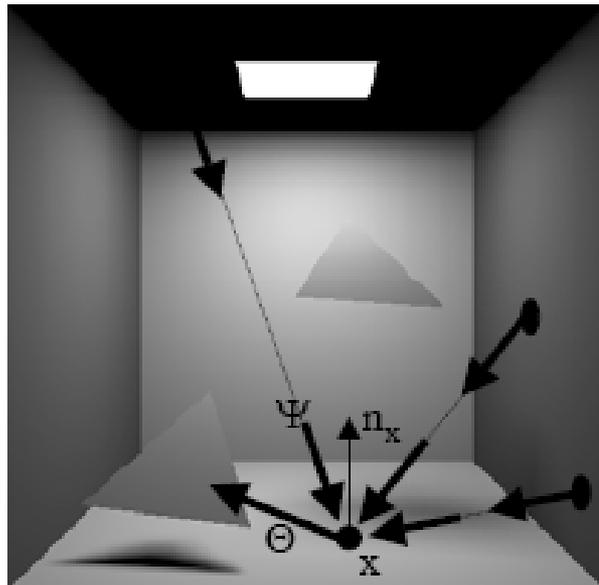
256 samples/
pixel



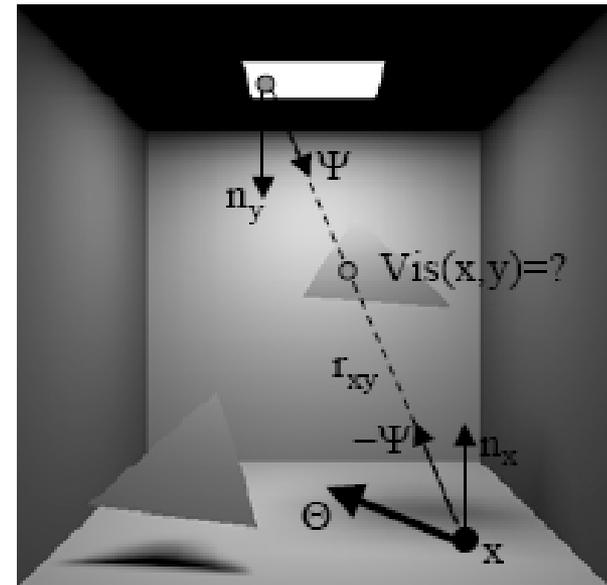
Direct Illumination

$$L(x \rightarrow \Theta) = \int_{A_{source}} f_r(x, -\Psi \leftrightarrow \Theta) \cdot L(y \rightarrow \Psi) \cdot G(x, y) \cdot dA_y$$

$$G(x, y) = \frac{\cos(n_x, \Theta) \cos(n_y, \Psi) \text{Vis}(x, y)}{r_{xy}^2}$$



hemisphere integration



area integration

Estimator for direct lighting

- Pick a point on the light's surface with pdf

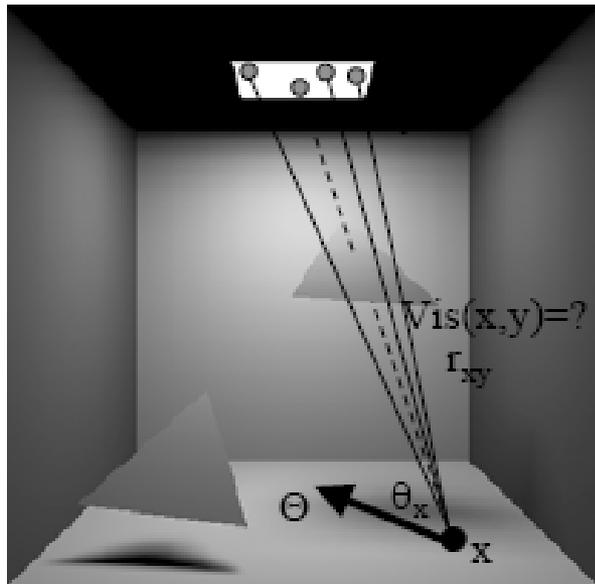
$$p(y)$$

- For N samples, direct light at point x is:

$$E(x) = \frac{1}{N} \sum_{i=1}^N \frac{f_r L_{source} \frac{\cos \theta_x \cos \theta_{\bar{y}_i}}{r_{x\bar{y}_i}^2} Vis(x, \bar{y}_i)}{p(\bar{y}_i)}$$

Generating direct paths

- Pick surface points y_i on light source
- Evaluate direct illumination integral



$$\langle L(x \rightarrow \Theta) \rangle = \frac{1}{N} \sum_{i=1}^N \frac{f_r(\dots)L(\dots)G(x, y_i)}{p(y_i)}$$

PDF for sampling light

- Uniform

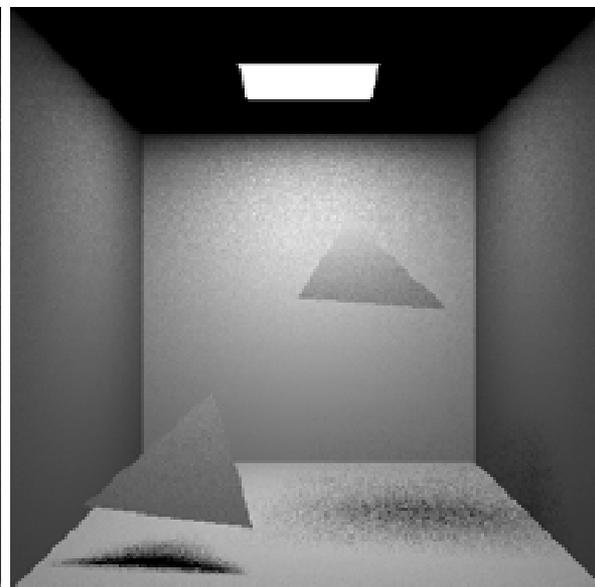
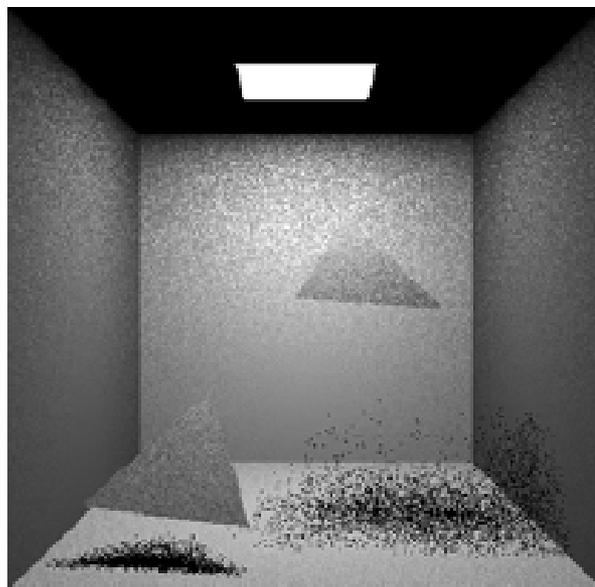
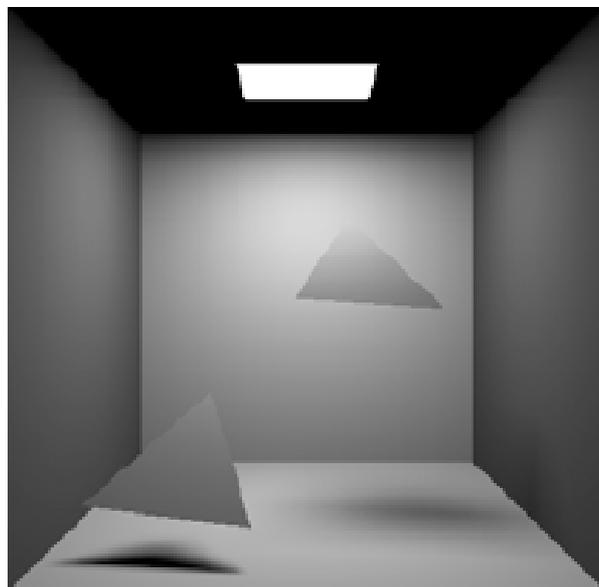
$$p(y) = \frac{1}{Area_{source}}$$

- Pick a point uniformly over light's area
 - Can stratify samples

- Estimator:

$$E(x) = \frac{Area_{source}}{N} \sum_{i=1}^N f_r L_{source} \frac{\cos \theta_x \cos \theta_{\bar{y}_i}}{r_{x\bar{y}_i}^2} Vis(x, \bar{y}_i)$$

More points ...

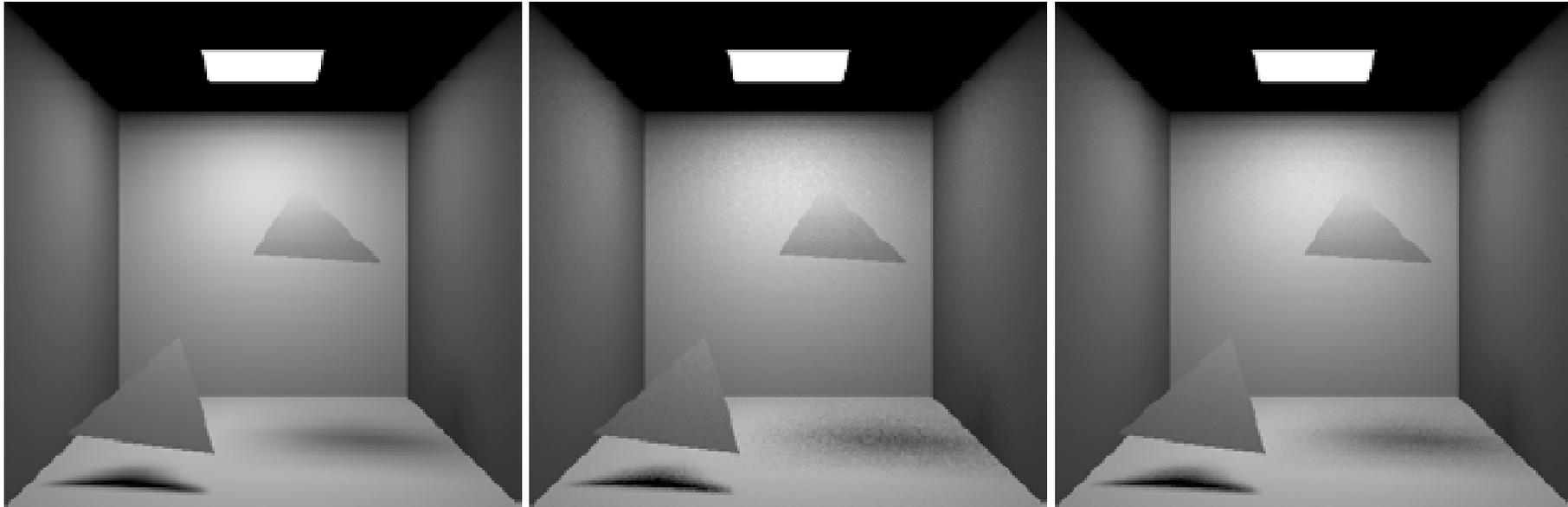


1 shadow ray

9 shadow rays

$$E(x) = \frac{Area_{source}}{N} \sum_{i=1}^N f_r L_{source} \frac{\cos \theta_x \cos \theta_{\bar{y}_i}}{r_{x\bar{y}_i}^2} Vis(x, \bar{y}_i)$$

Even more points ...



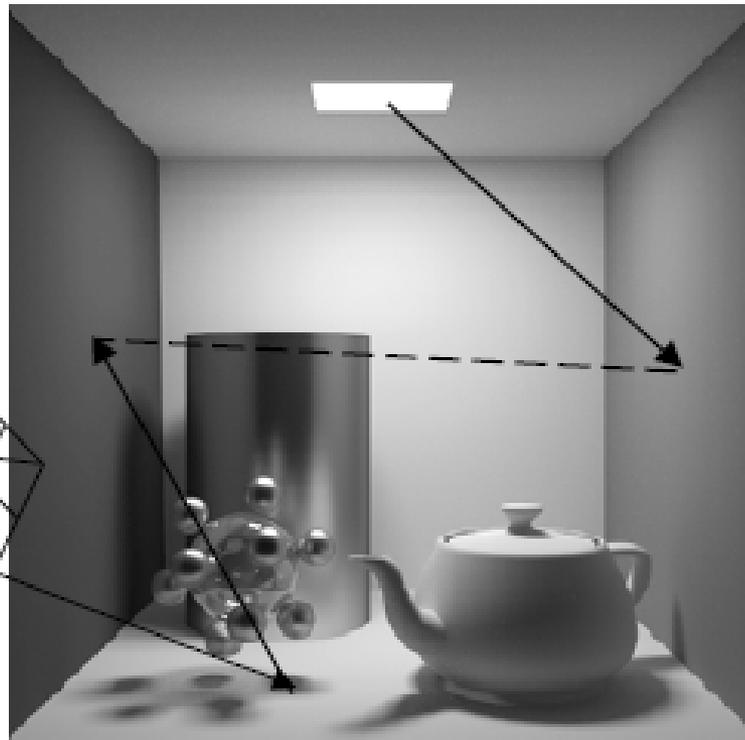
36 shadow rays

100 shadow rays

$$E(x) = \frac{Area_{source}}{N} \sum_{i=1}^N f_r L_{source} \frac{\cos \theta_x \cos \theta_{\bar{y}_i}}{r_{x\bar{y}_i}^2} Vis(x, \bar{y}_i)$$

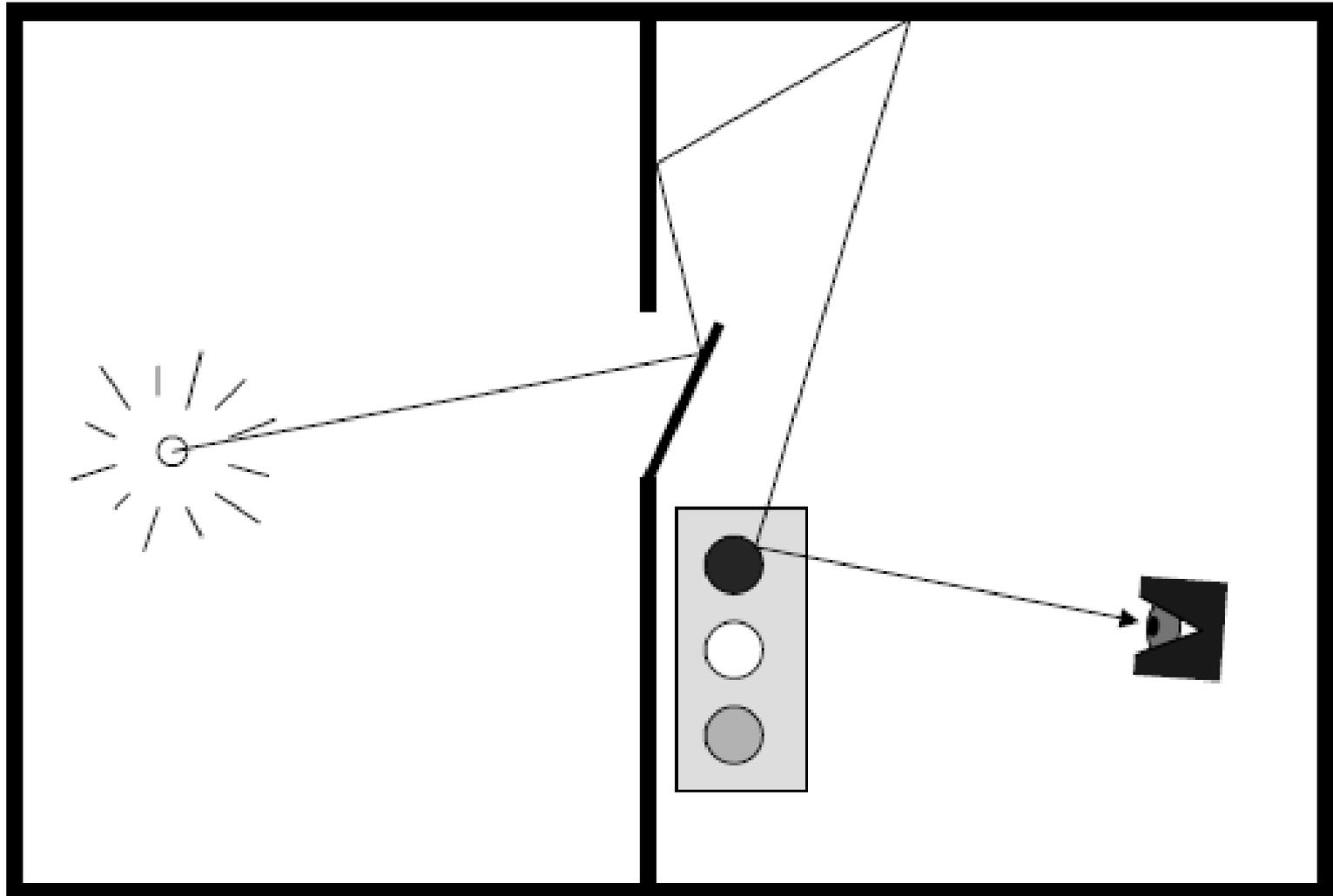
Bidirectional Path Tracing

- Or paths generated from both camera and source at the same time ...!

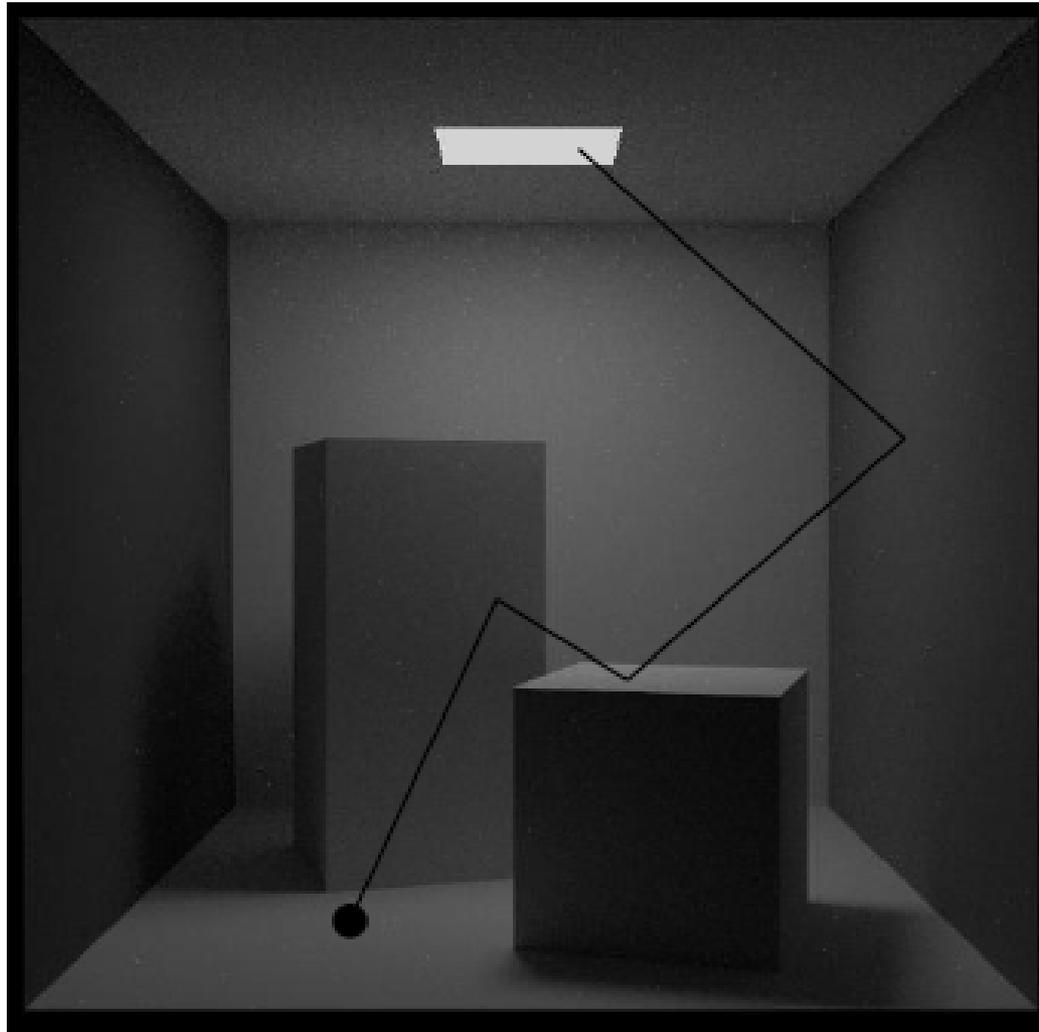


- Connect endpoints to compute final contribution

Metropolis

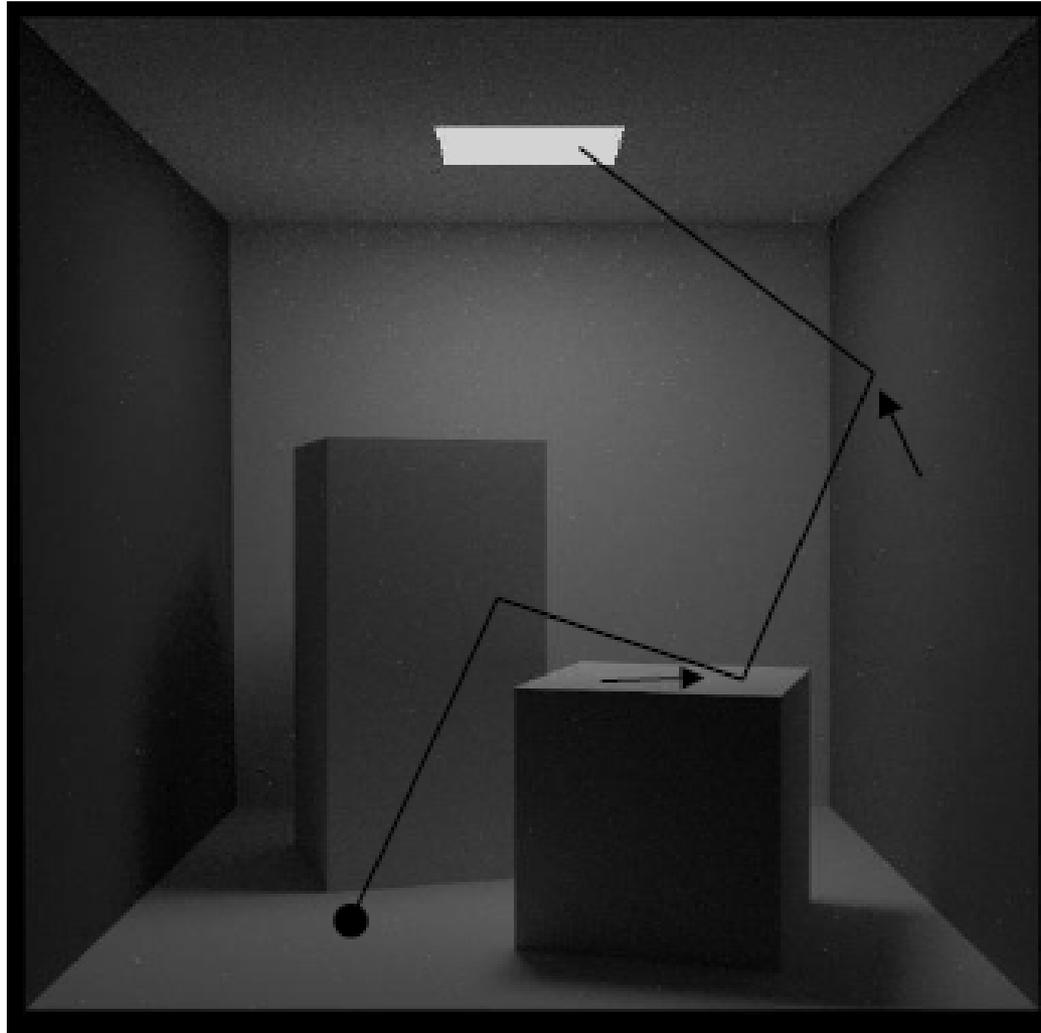


Metropolis



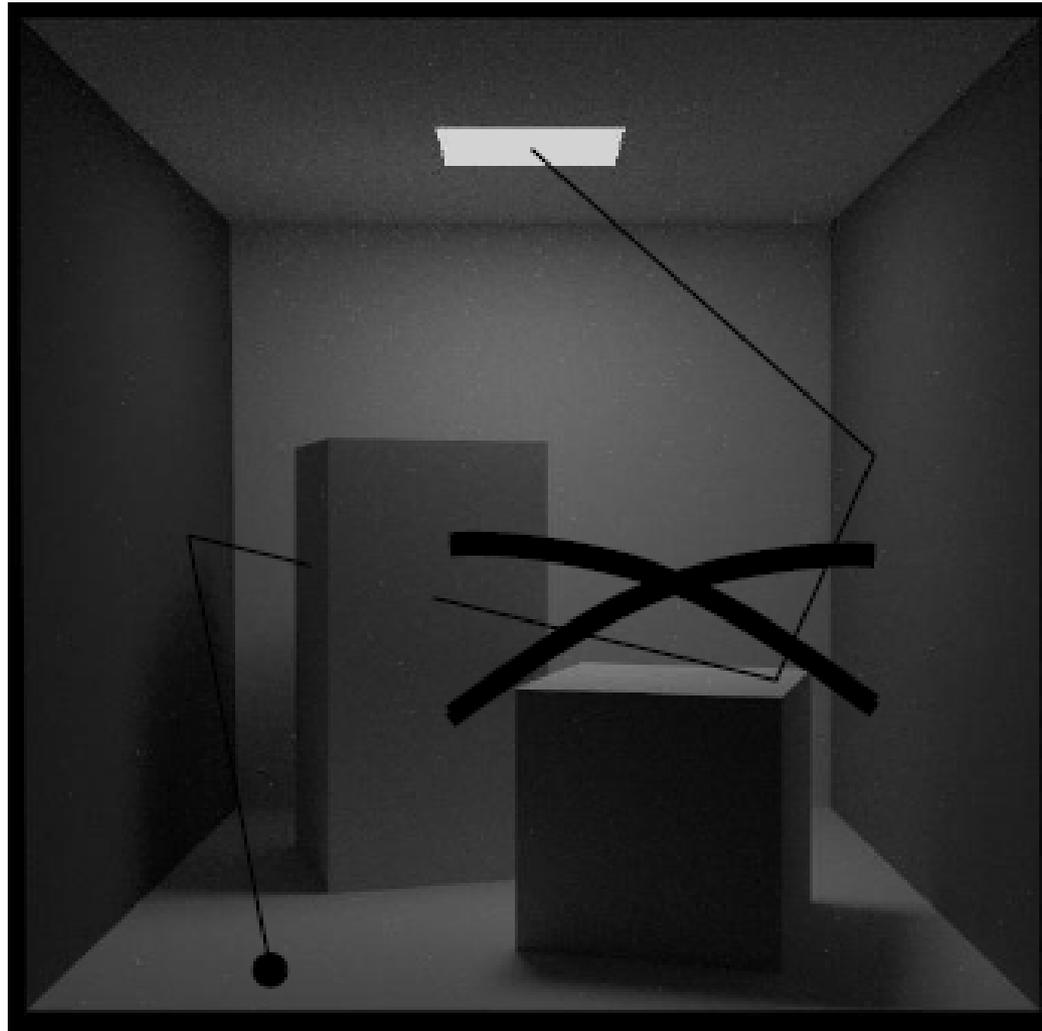
valid path

Metropolis



small
perturbations

Metropolis



Accept
mutations
based on
energy
transport

Unbiased vs. Consistent

- **Unbiased**

- **No systematic error**
- **$E[\mathbf{I}_{\text{estimator}}] = \mathbf{I}$**
- **Better results with larger N**

- **Consistent**

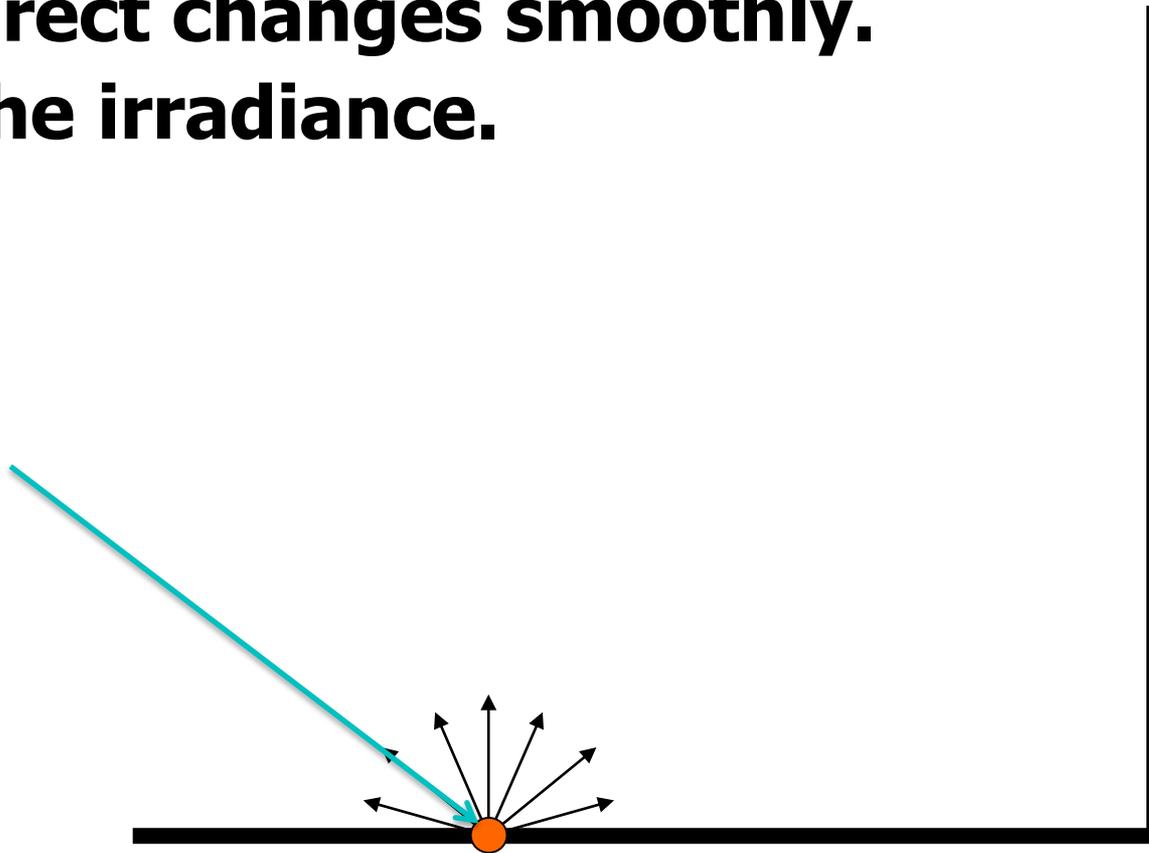
- **Converges to correct results with more samples**
- **$E[\mathbf{I}_{\text{estimator}}] = \mathbf{I} + \boldsymbol{\varepsilon}$, where $\lim_{n \rightarrow \infty} \boldsymbol{\varepsilon} = \mathbf{0}$**

Biased Methods

- **MC methods**
 - **Too noisy and slow**
 - **Noise is objectionable**
- **Biased methods: store information (caching)**
 - **Irradiance caching**
 - **Photon mapping**

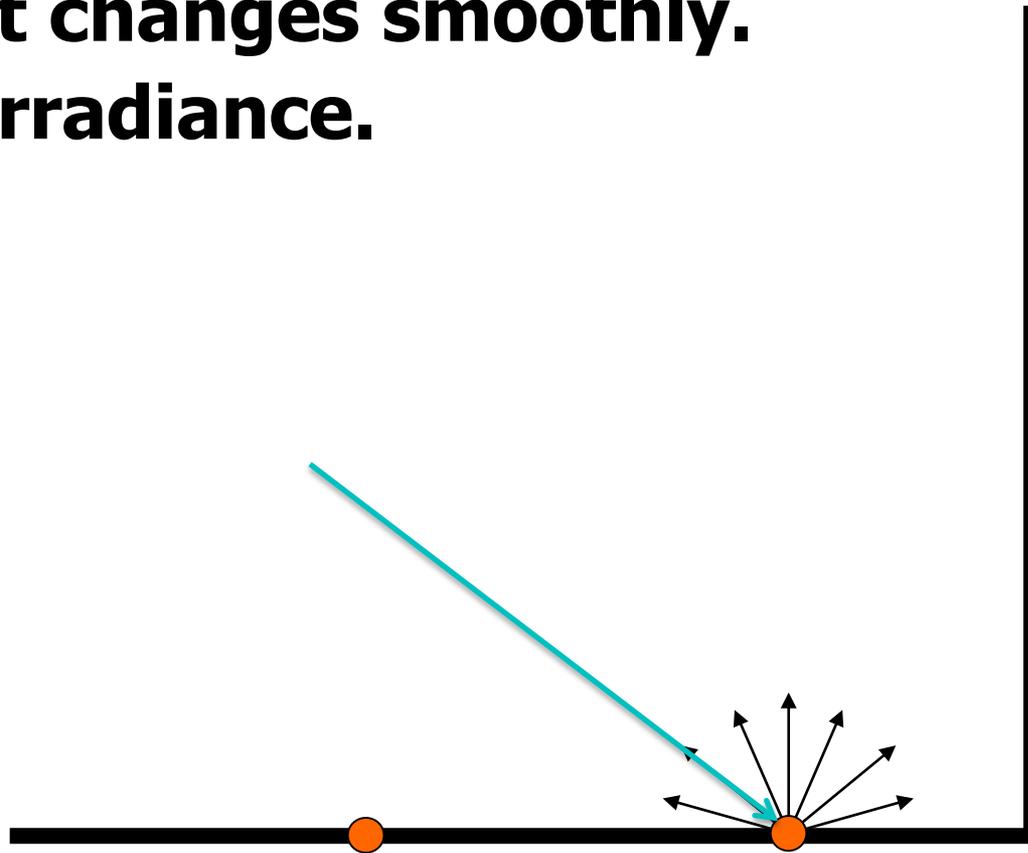
Biased Methods: Irradiance Caching

- Indirect changes smoothly.
- Cache irradiance.



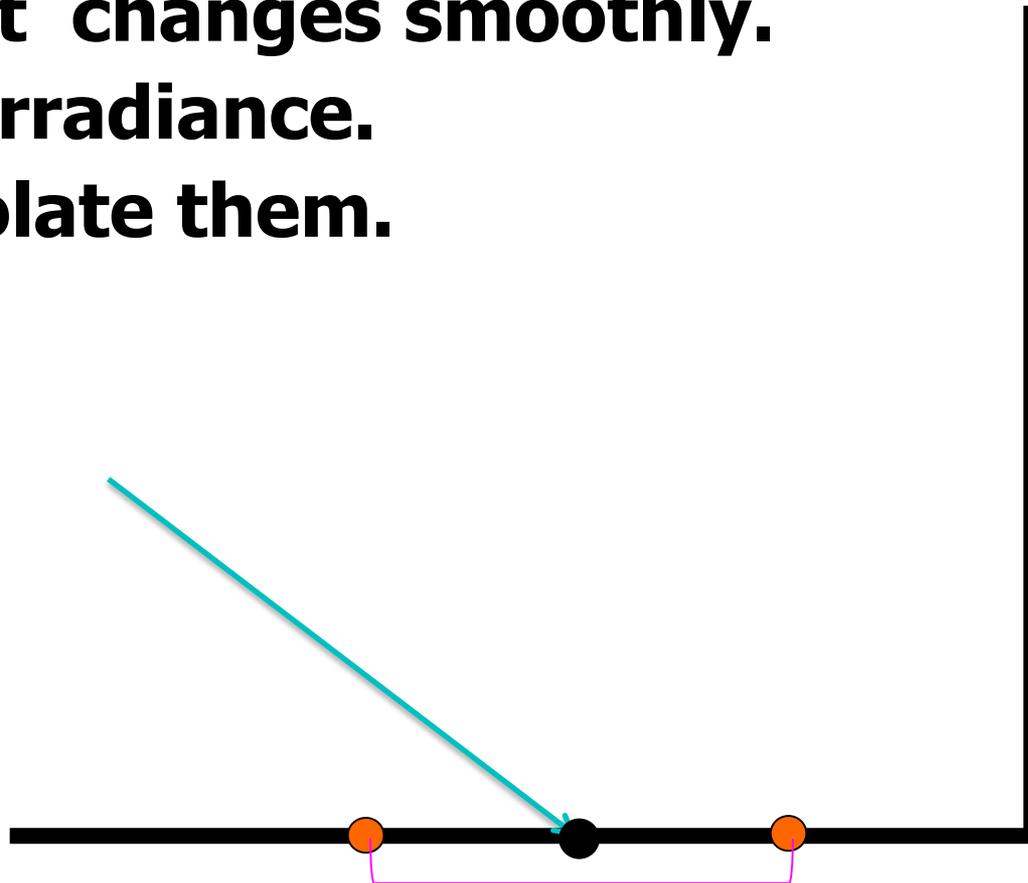
Irradiance Caching

- **Indirect changes smoothly.**
- **Cache irradiance.**



Irradiance Caching

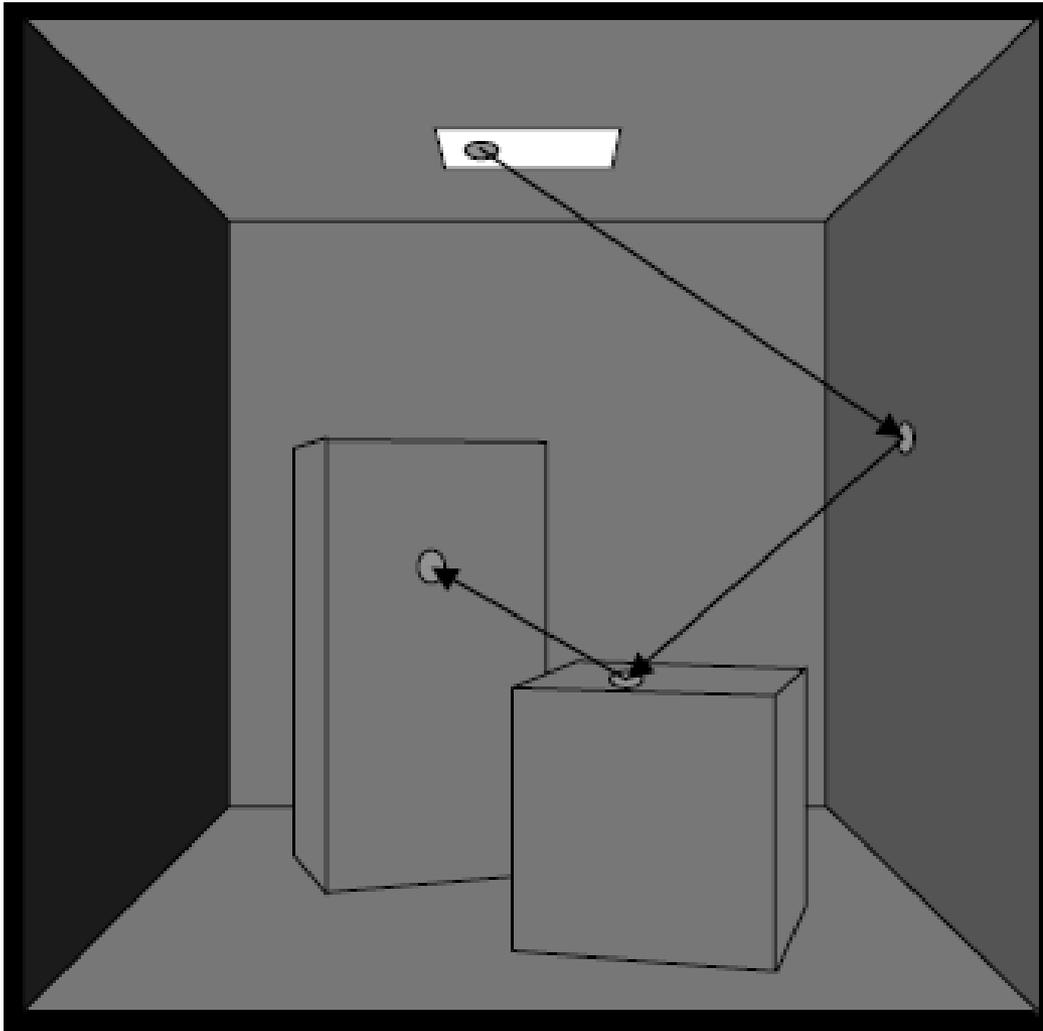
- **Indirect changes smoothly.**
- **Cache irradiance.**
- **Interpolate them.**



Biased Method: Photon Mapping

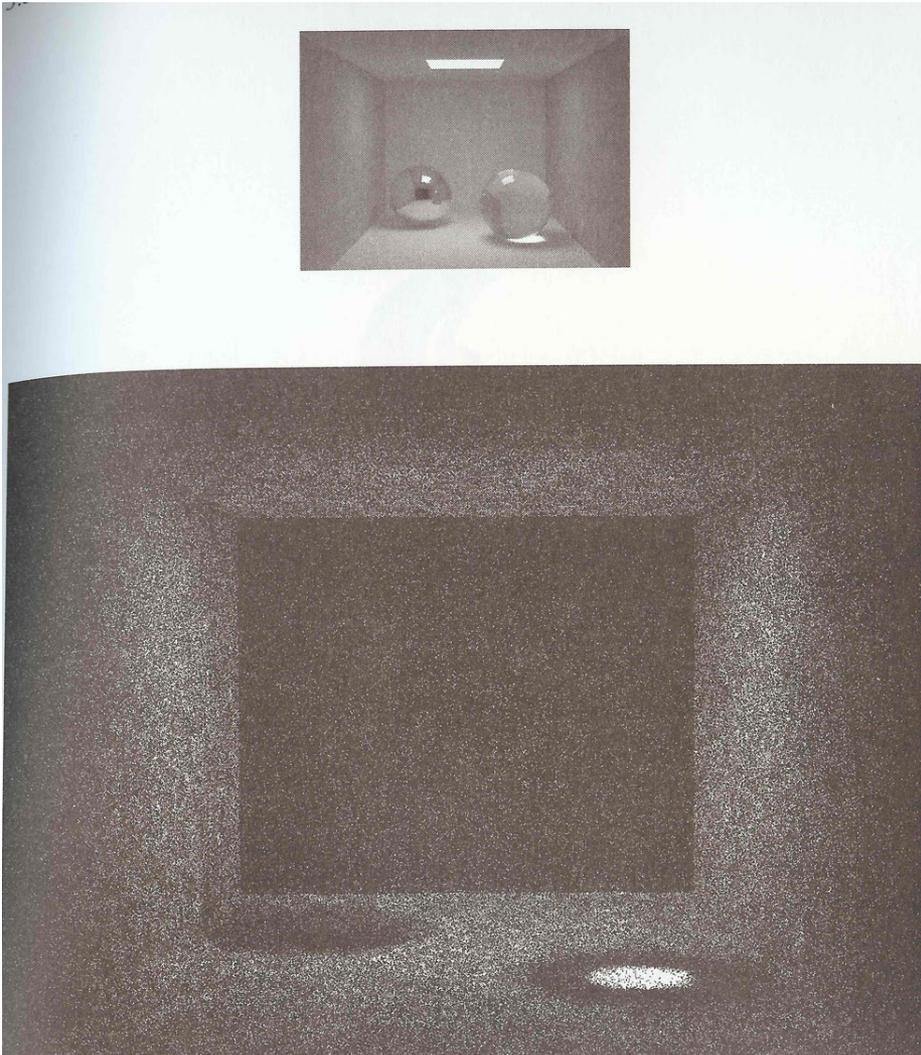
- **2 passes:**
 - **Shoot “photons” (light-rays) and record any hit-points**
 - **Shoot viewing rays and collect information from stored photons**

Pass 1: shoot photons



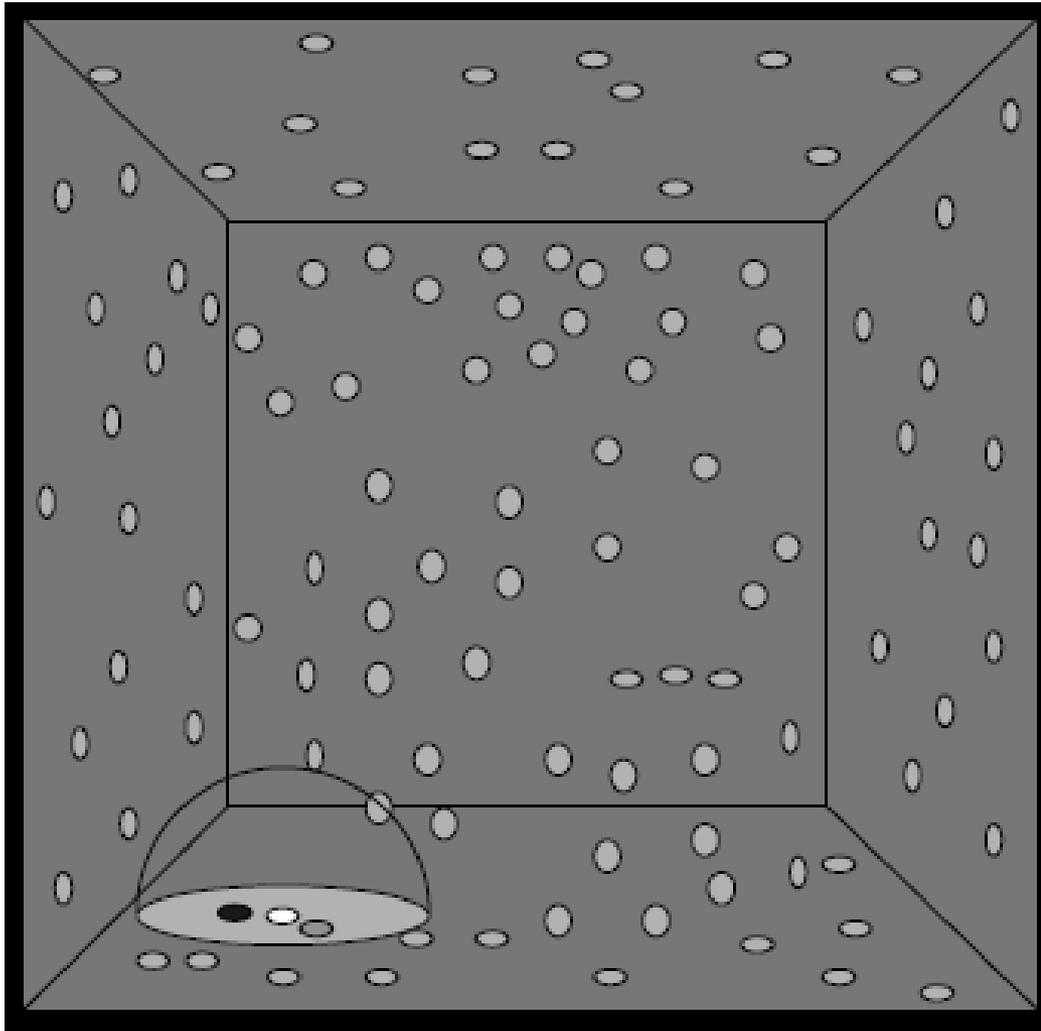
- Light path generated using MC techniques and Russian Roulette
- Store:
 - position
 - incoming direction
 - color
 - ...

Stored Photons



**Generate a few
hundreds of
thousands of
photons**

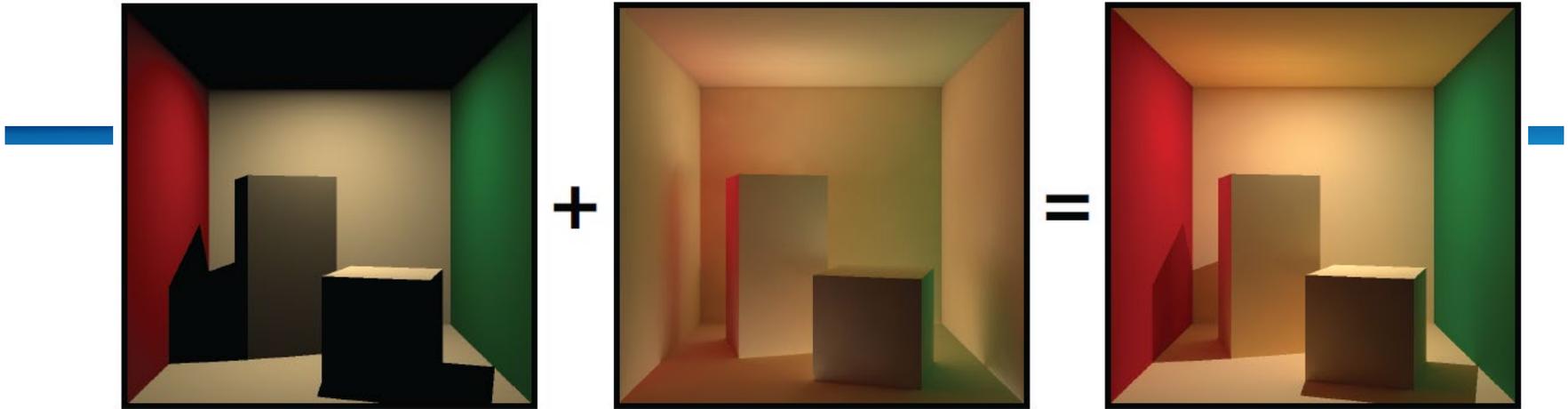
Pass 2: viewing ray



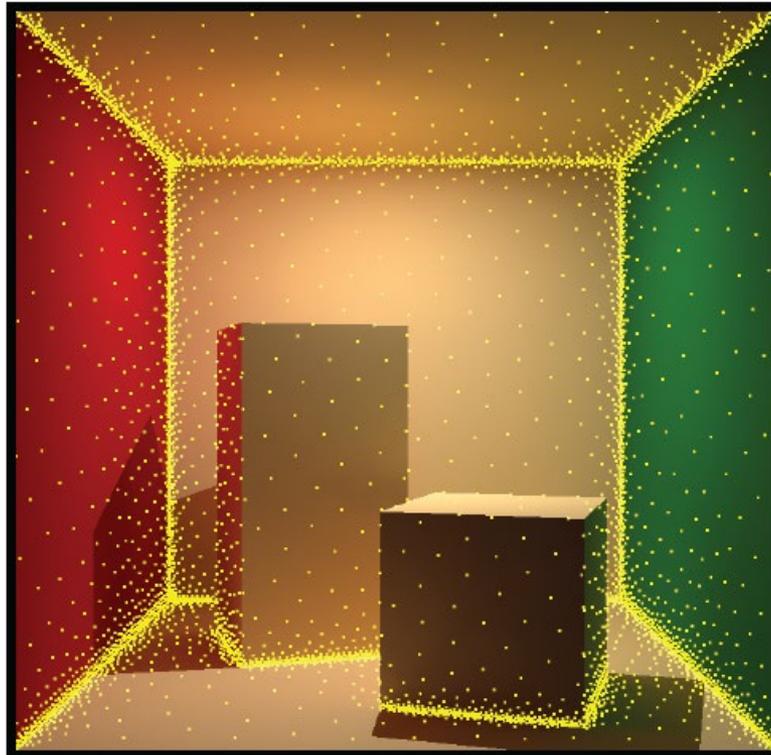
- Search for N closest photons (+check normal)
- Assume these photons hit the point we're interested in
- Compute average radiance

Direct

Indirect



Indirect \approx



Result



**350K photons
for the caustic
map**

Result



**350K photons
for the caustic
map**

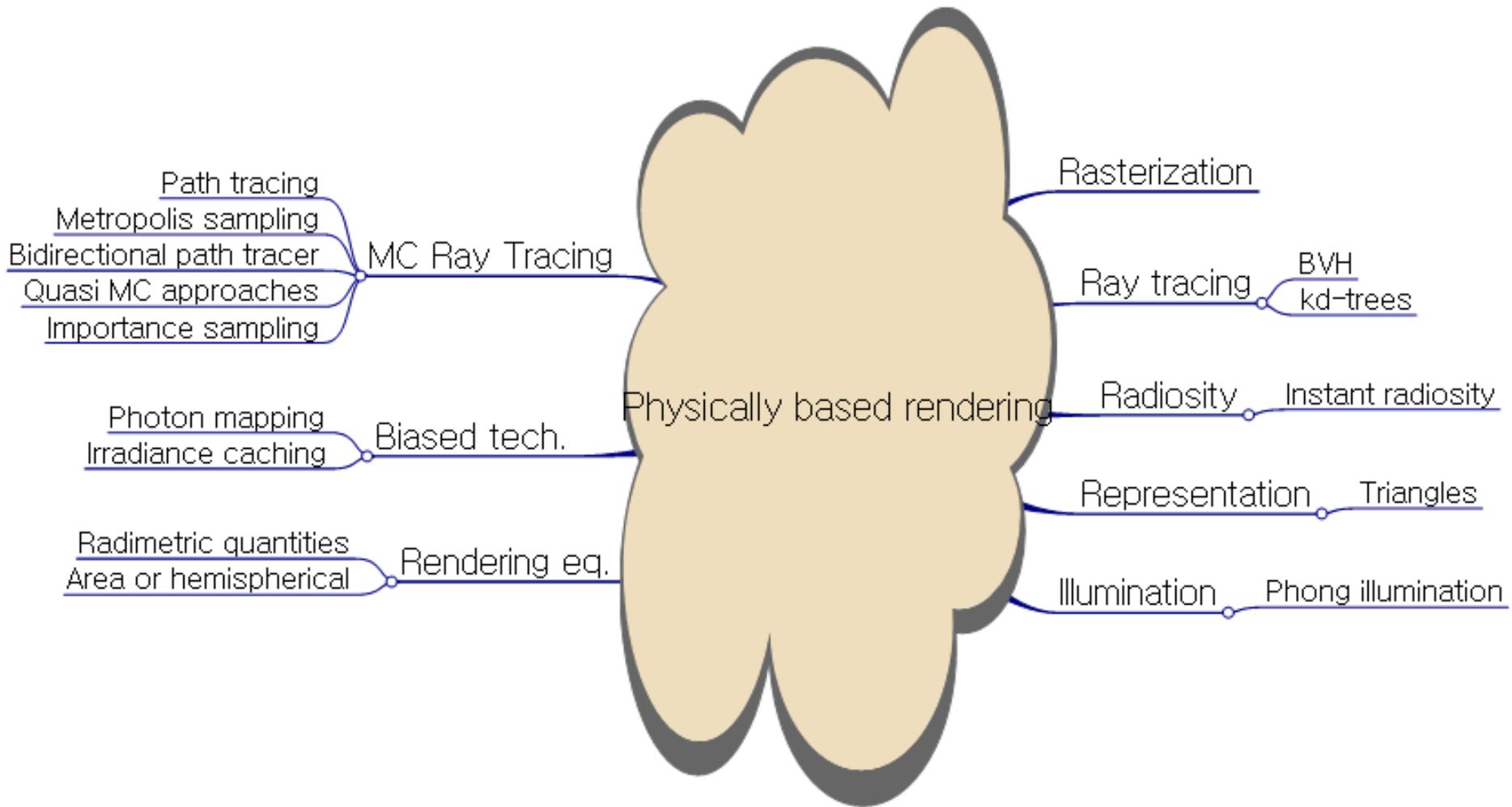
Class Objectives were:

- **Understand a basic structure of Monte Carlo ray tracing**
 - **Russian roulette for its termination**
 - **Path tracing**

Summary

- **Two basic building blocks**
- **Radiometry**
- **Rendering equation**
- **MC integration**
- **MC ray tracing**
 - **Unbiased methods**
 - **Biased methods**

Summary



Next Time...

- **Denoising techniques**

Homework

- **Go over the next lecture slides before the class**
- **Watch 2 SIG/CVPR/ISMAR videos and submit your summaries every Mon. class**
 - **Just one paragraph for each summary**
 - **Any top-tier conf (e.g., ICRA) is okay**

Example:

Title: XXX XXXX XXXX

Abstract: this video is about accelerating the performance of ray tracing. To achieve its goal, they design a new technique for reordering rays, since by doing so, they can improve the ray coherence and thus improve the overall performance.

Any Questions?

- **Submit three times before the mid-term exam**
- **Come up with one question on what we have discussed in the class and submit:**
 - **1 for typical or already answered questions**
 - **2 for questions that have some thoughts or surprise me**