Realistic Image Synthesis

- Path Tracing -

Pascal Grittmann

Philipp Slusallek
Karol Myszkowski
Gurprit Singh

What are we computing? – The physical phenomenon

What are we computing?
– The mathematical formulation

Given:

A point visible to the camera

Compute:

Incident light from all directions, reflected to the camera

What are we computing?
– The mathematical formulation

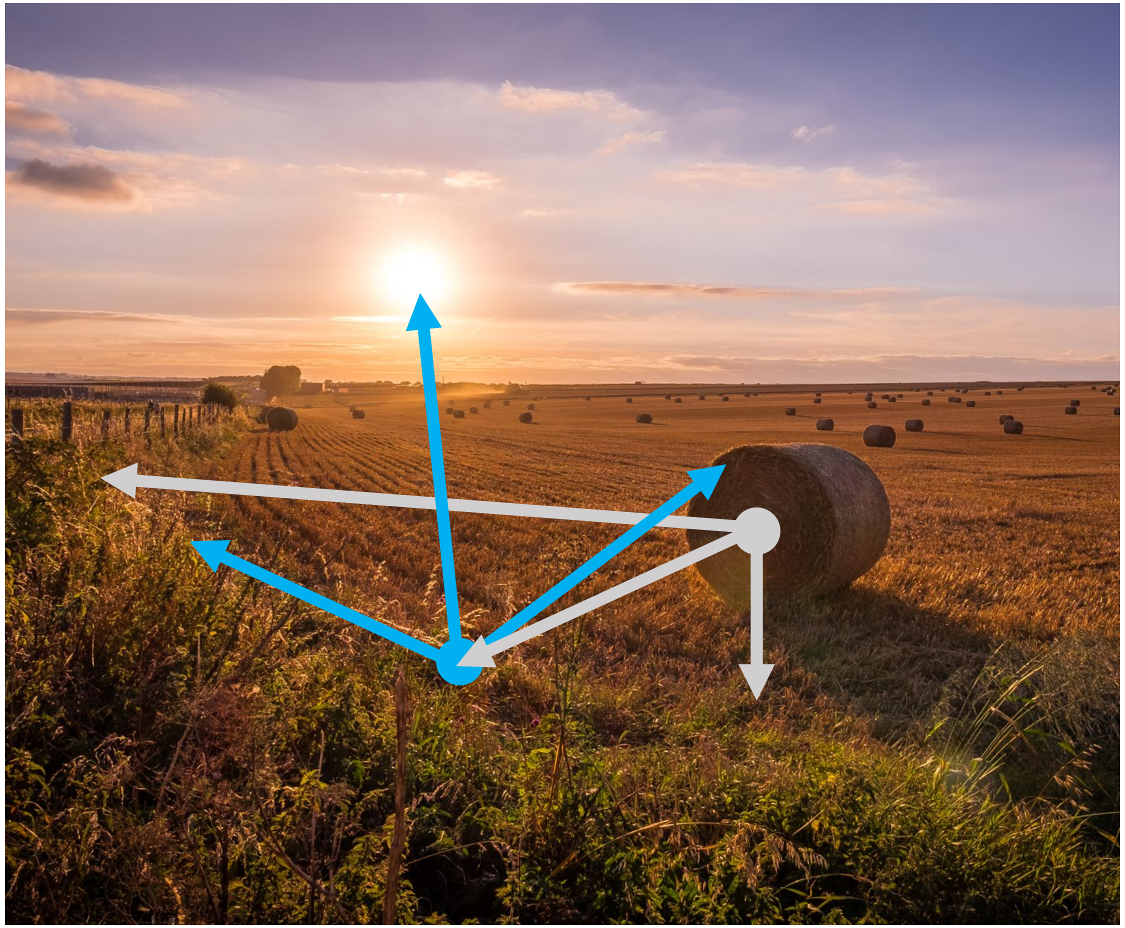
Given:

A point visible to the ~~camera~~ previous point

Compute:

Incident light from all directions, reflected to the ~~camera~~ previous point

Recursive problem!



Reminder: Rendering Equation

Integral over sphere of all directions

$$L_o(x,\omega_o) = L_e(x,\omega_o) + \int_\Omega f(x,\omega_o,\omega_i)\, L_i(x,\omega_i)\, |\cos(\theta_i)|\, d\omega_i$$

Outgoing radiance from x in direction ω_o

Emitted radiance from x in direction ω_o

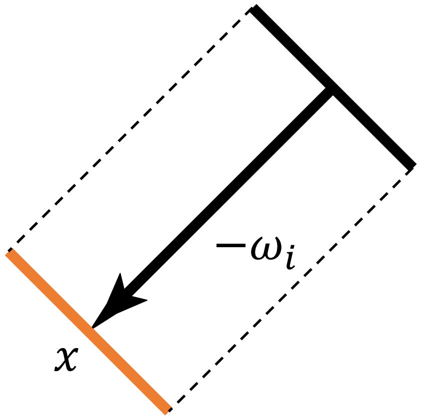
Incident radiance at x from direction ω_i

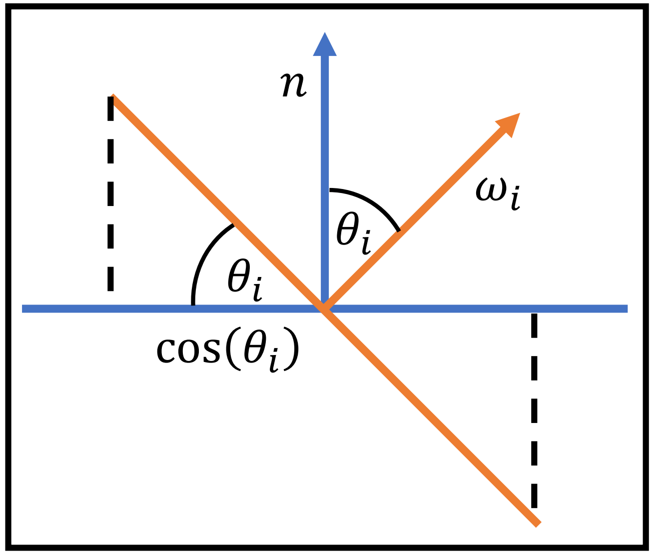
The mysterious cosine term

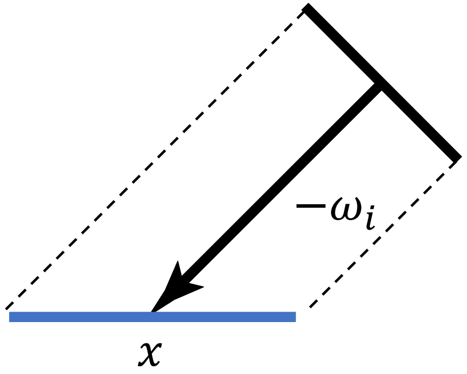
Bidirectional scattering distribution function – BSDF (how much light is reflected from ω_i towards ω_o?)

Why the cosine term?

- Definition of radiance:
  - "flux per unit solid angle per unit **projected** area"
- We are interested in:
  - "flux per unit solid angle per unit ~~projected~~ area"

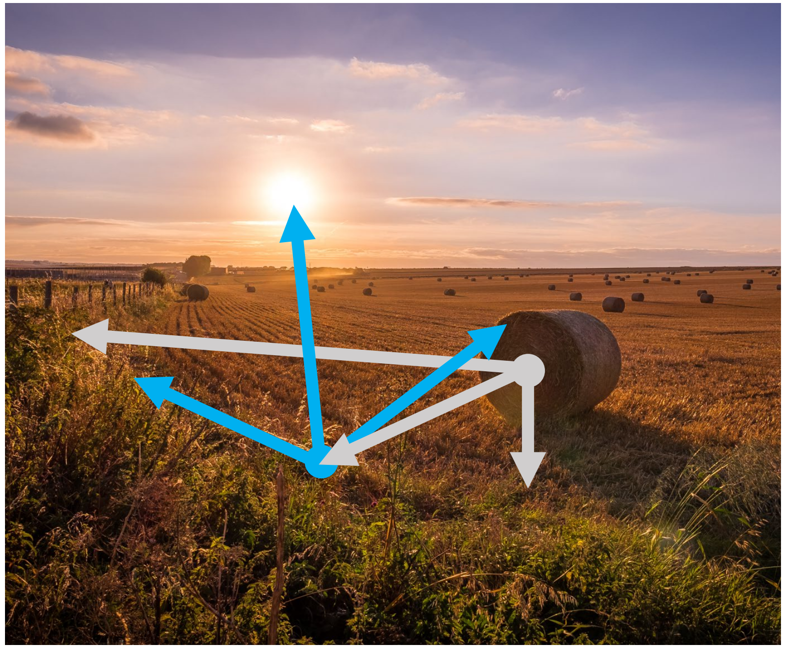






Challenges

- Recursive → high dimensional
- Discontinuities (object and shadow boundaries)



Solution: Monte Carlo integration

- In general, with a single sample (*primary estimator*)

$$\int_{\Omega} f(x)\,dx \approx \frac{f(x)}{p(x)}$$

- Applied to rendering:

$$L_o(x_1, \omega_o) \approx L_e(x_1, \omega_o) + \frac{f(x_1, \omega_o, \omega_i)\,|\cos(\theta_i)|}{p(\omega_i)} L_o(x_2, -\omega_i)$$

Estimated recursively!
→ High (infinite) dimensionality

How to sample ω_i?

- Ideally proportionally to the full integrand
  - $p(\omega_i) \propto f(x, \omega_o, \omega_i)\, L_i(x, \omega_i)\, |\cos(\theta_i)|$
- Usually not possible → Limit to one or more factors
- Simplest method: cosine
  - $p(\omega_i) \propto |\cos(\theta_i)|$
- Often also possible (approximately, at least): BSDF times cosine
  - $p(\omega_i) \propto f(x, \omega_o, \omega_i)\, |\cos(\theta_i)|$

When to terminate?

- In reality, all surfaces reflect light
- We cannot track infinitely long paths, need to terminate recursion at some point
- Options:
  - Maximum path length (biased)
  - Russian roulette (unbiased, later)

Russian Roulette and Splitting

Russian roulette

- Randomly terminate the path with some probability p
- Unbiased
- Increases variance (i.e., noise in the image)

- Given an estimator F, replace by F_{RR}:

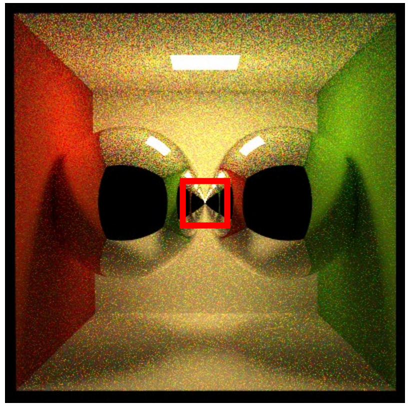
$$F_{RR} = \begin{cases} \frac{1}{p} F & \text{with probability } p \\ 0 & \text{with probability } (1-p) \end{cases}$$

What criterion to use?

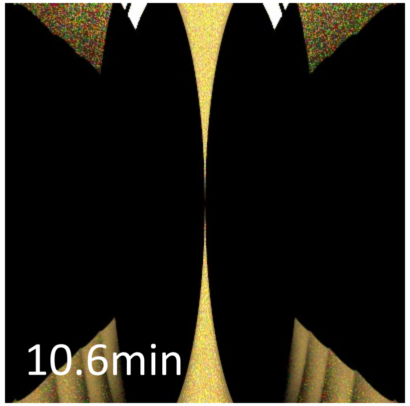
- Fixed probability
- Current path throughput
- Efficiency-optimized RR (Veach 97):
  - Based on statistics of surrounding pixels
- “Adjoint driven Russian roulette and splitting” Vorba et al 201
  - Based on statistics from a pre-pass, combined with splitting

Russian roulette experiments

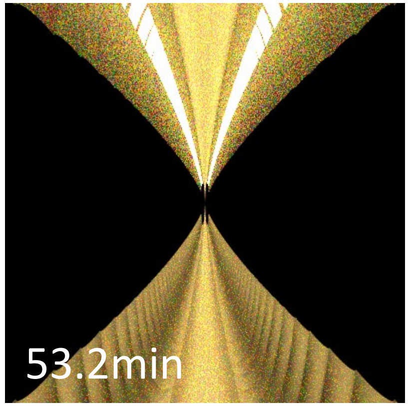
input scene



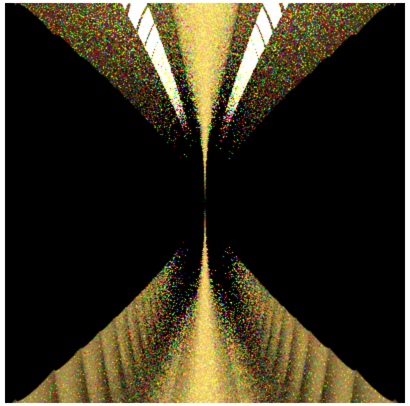
path depth < 11



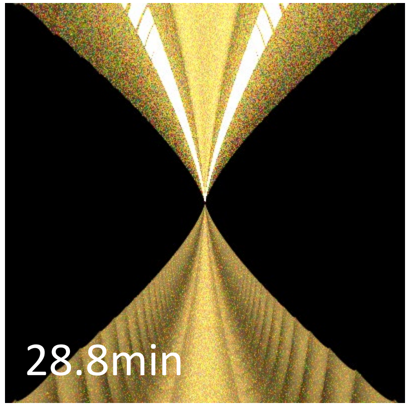
path depth < 101



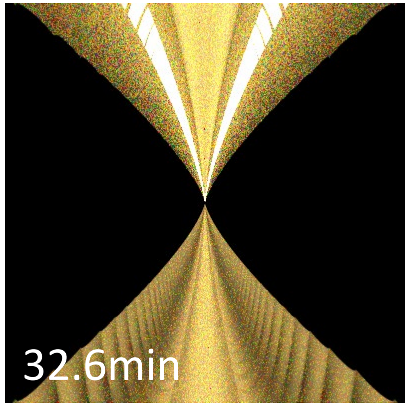
RR with p=0.3



efficiency optimized



p = throughput / 0.01

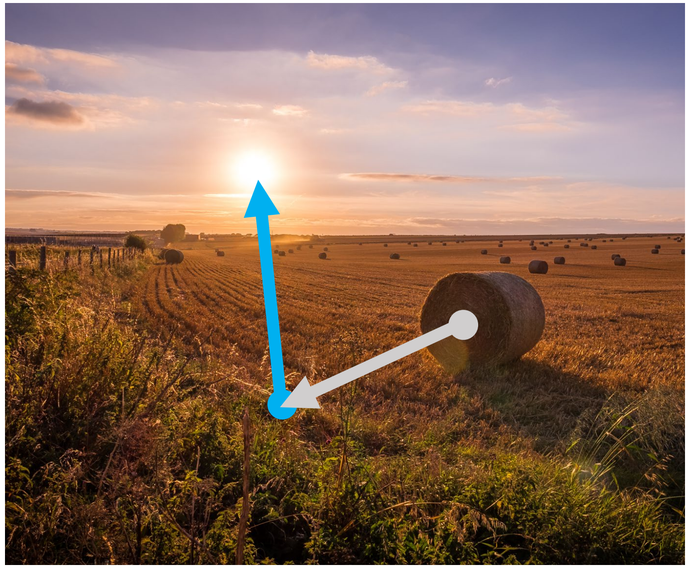


Thiago Ize (University of Utah, currently Solid Angle)
http://www.cs.utah.edu/~thiago/cs7650/hw12/

Splitting

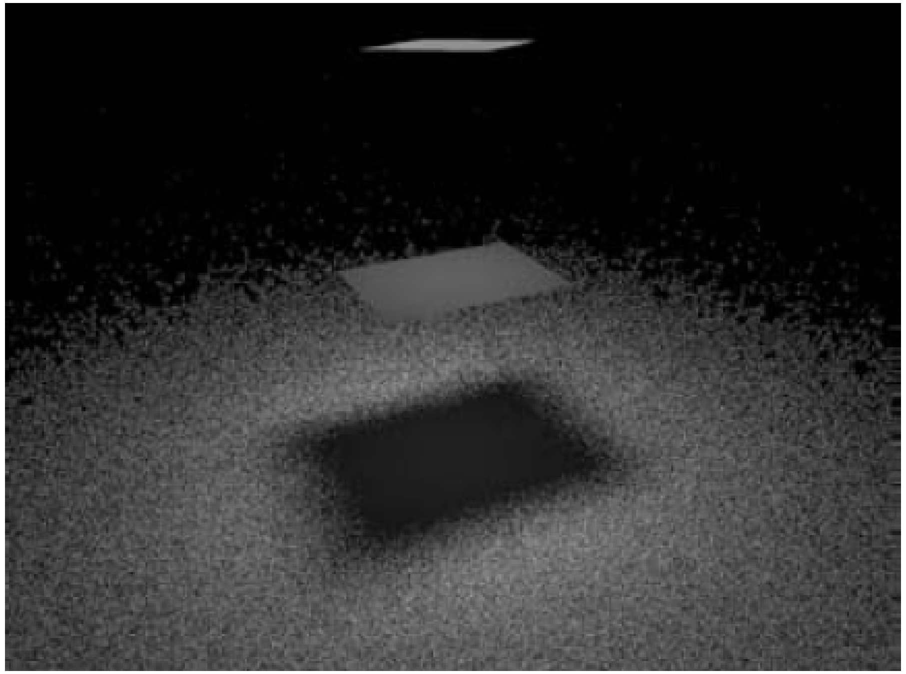
- Use more samples for the next recursion than the current one
- Often done at the first point (the one visible to the camera)
- Trade off: anti-aliasing vs variance

No splitting

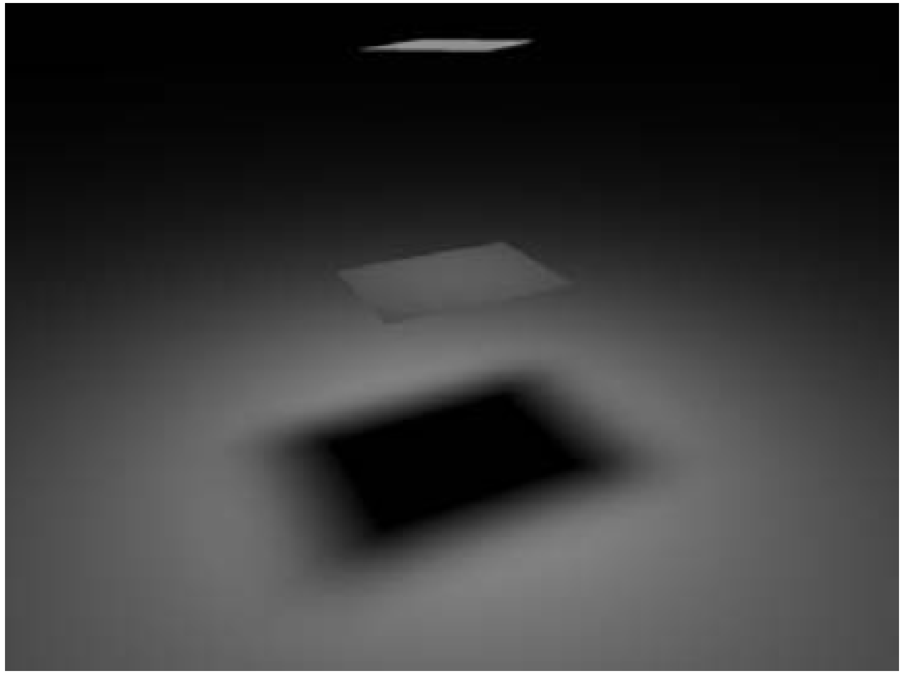


Splitting

Example: Direct illumination

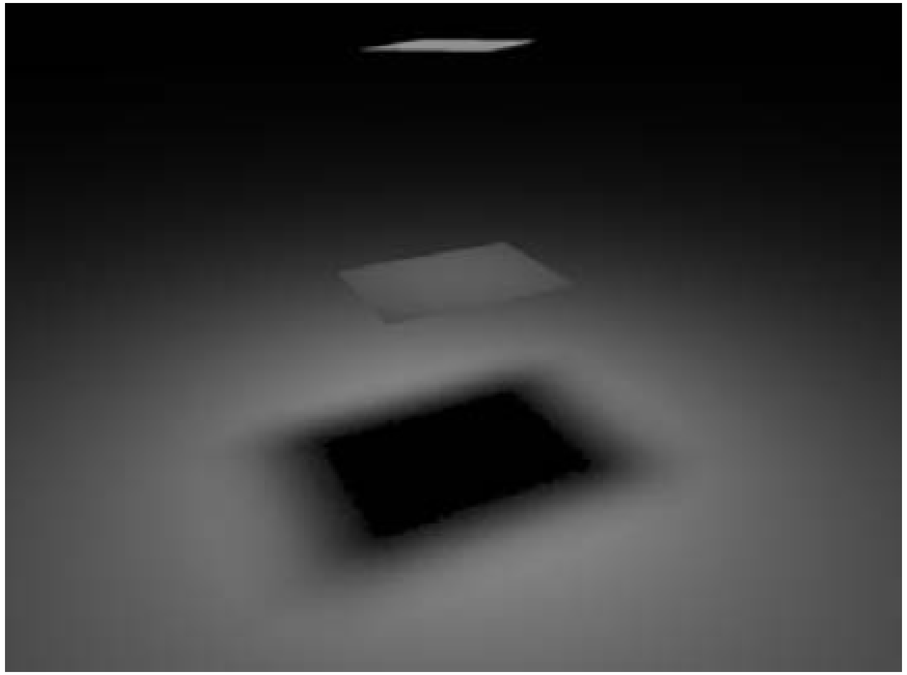


Sampling projected solid angle
4 eye rays per pixel
100 shadow rays

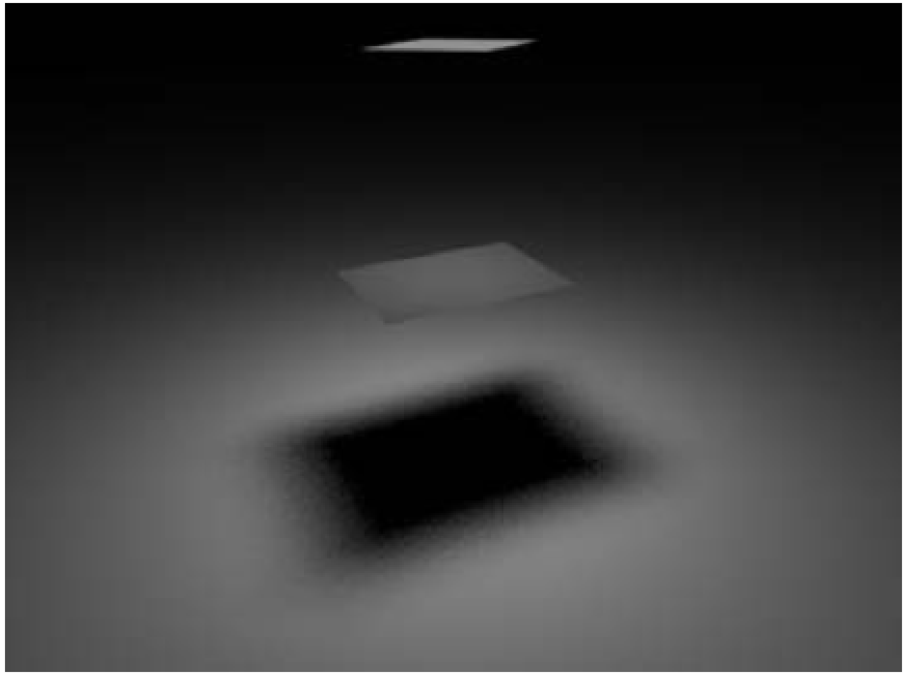


Sampling light source area
4 eye rays per pixel
100 shadow rays

Example: Splitting for direct illumination



Stratified random sample locations
4 eye rays per pixel
16 shadow ray



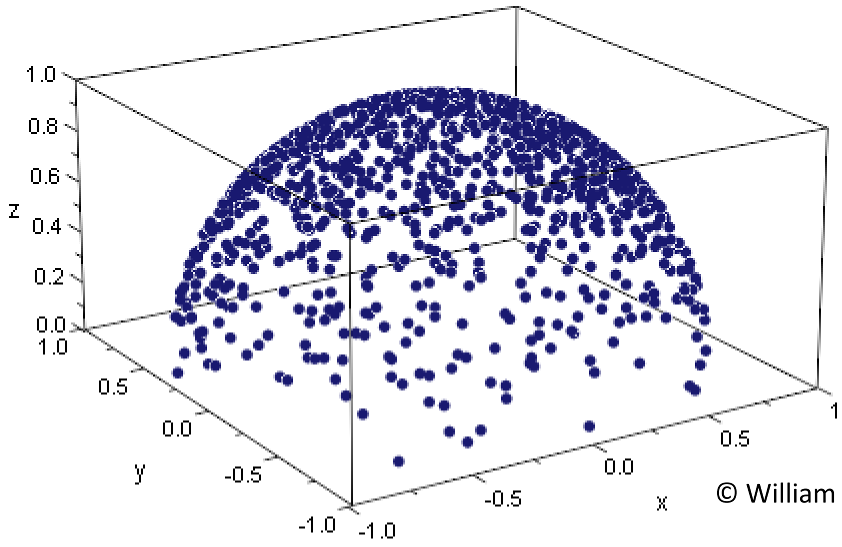
Stratified random sample locations
64 eye rays per pixel
1 shadow ray

Importance Sampling the BSDF

Simplest method: cosine hemisphere sampling

- PDF proportional to the cosine:
  - $p(\omega_i) = \frac{\cos\theta_i}{\pi}$
- Ideal for diffuse surfaces

- Given two uniform samples u_1, u_2:
  - $\phi_i = 2\pi u_1$
  - $\theta_i = \cos^{-1}\sqrt{u_2}$
  - (Derivation in Assignment 3)



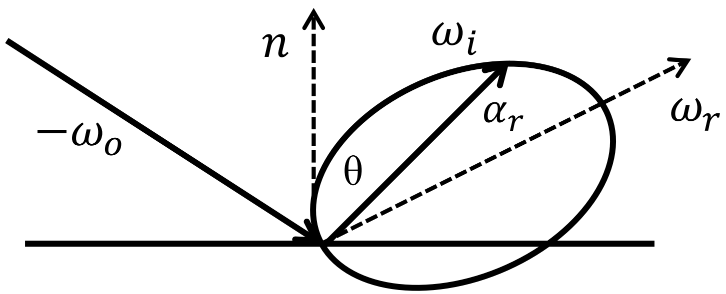
© William Sherif

Example: Phong BRDF

- Simple glossy BRDF
- Multiplies by a cosine lobe around the perfect reflection

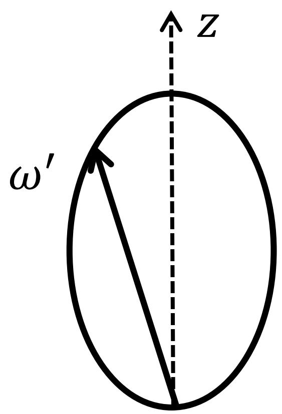
$$f(x, \omega_o, \omega_i) = \frac{n+2}{2\pi} \cos^n \alpha_r$$

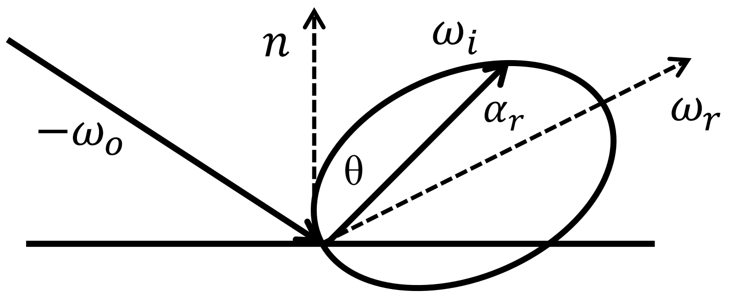
- α_r is the angle formed by ω_i and ω_r
- ω_r is the prefect reflection of ω_o



Sampling the cosine lobe

- Step 1: sample the cosine lobe about z axis
  - $p(\omega') = \cos^n \theta' \frac{n+1}{2\pi}$
- Step 2: compute ω_r
  - Reflect ω_o about n
- Step 3: transform ω' to coordinate system where ω_r is the z axis
  - Matrix multiplication

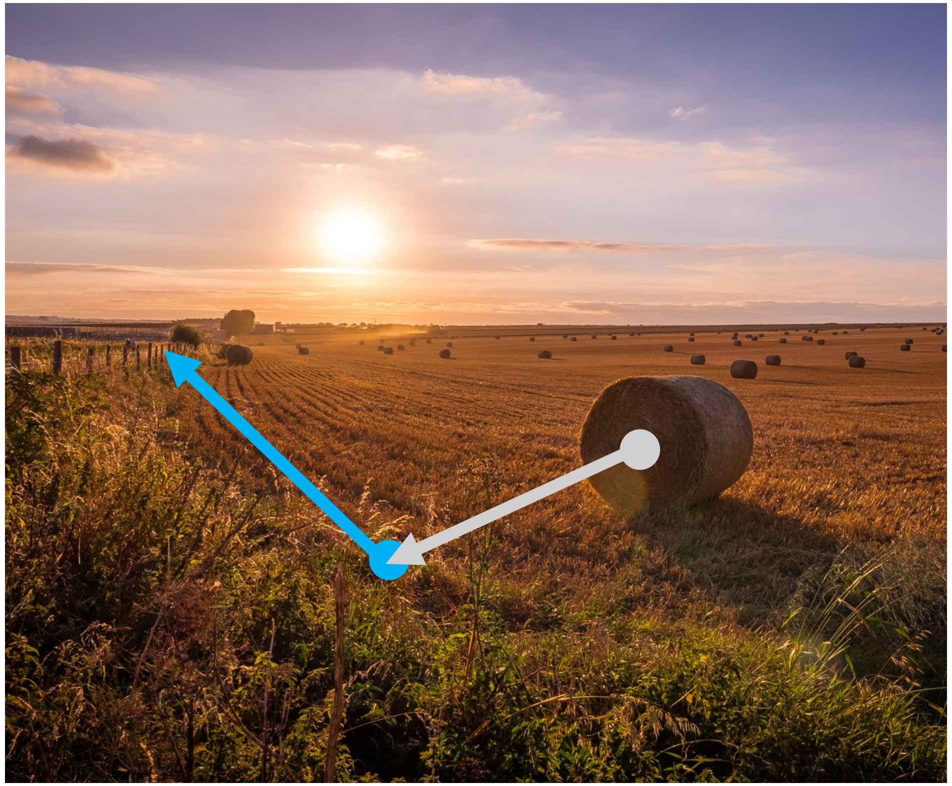




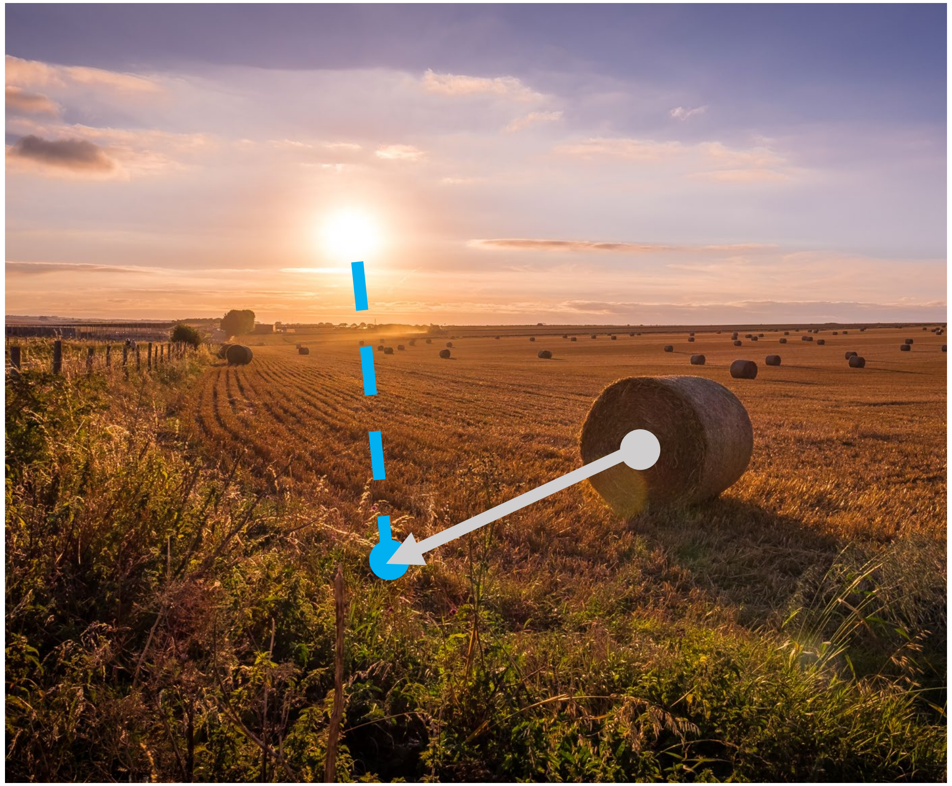
Next Event Estimation

Reduce variance by connecting to points on the light source

Sampling ω_i from the hemisphere



Sampling ω_i as a direction to a point on the light

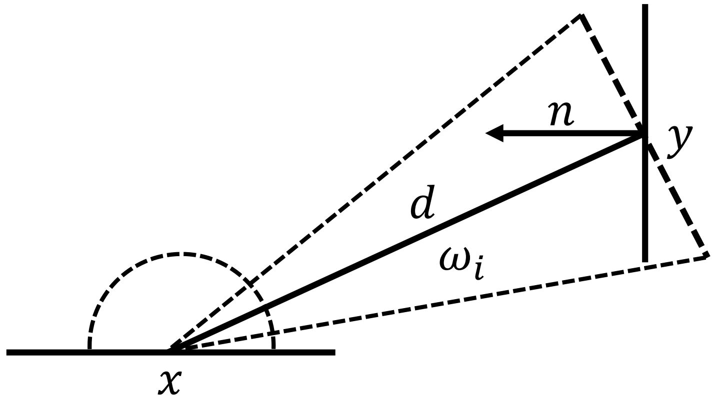


But it is a point, not a direction…

- We sample a direction ω_i by sampling a point y on the light
- With some pdf $p(y)$ defined on the **surface area** of the light
- But our integral is over the **(hemi-) sphere**
- We need to perform a **change of variables**

From surface area to hemisphere

- Projected unit surface area at the light source
  - $\cos\theta_y$
- Compute surface area on unit hemisphere
  - Similar triangles → $\frac{1}{d}$
  - 2D → $\frac{1}{d^2}$
- We must also account for the visibility!
  - $V(x, y)$
- Result: $\frac{\cos\theta_y}{d^2} V(x, y)$

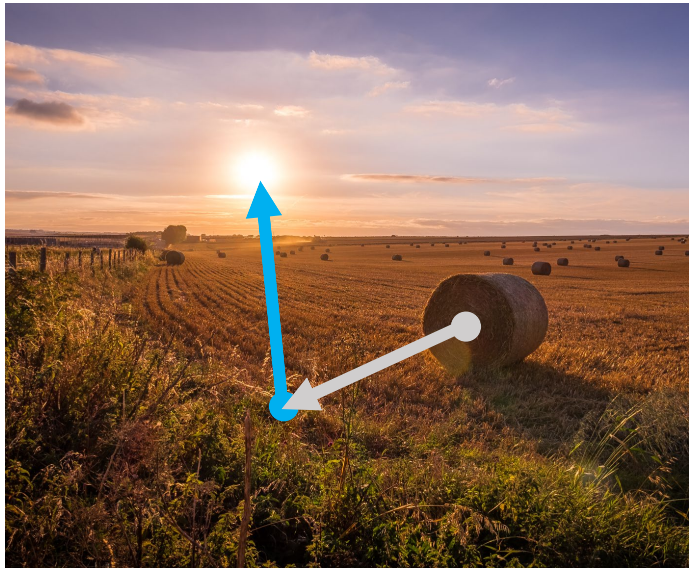


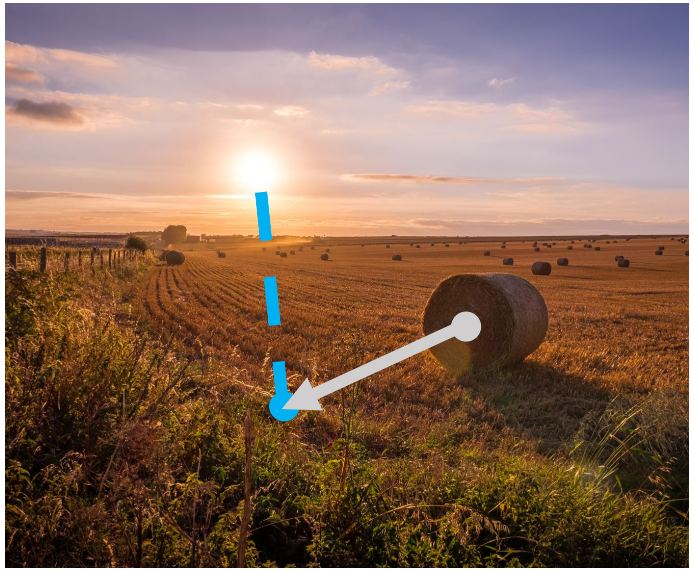
How to select the light source

- Total power
- Uniformly
- Estimated unoccluded contribution
- Estimated occluded contribution (prepass)
- Clustering to support many lights

What if a BSDF sample hits the light

- Cannot count both: would yield twice the actual value!
- Could average both and divide by two → high variance
- Or: use MIS



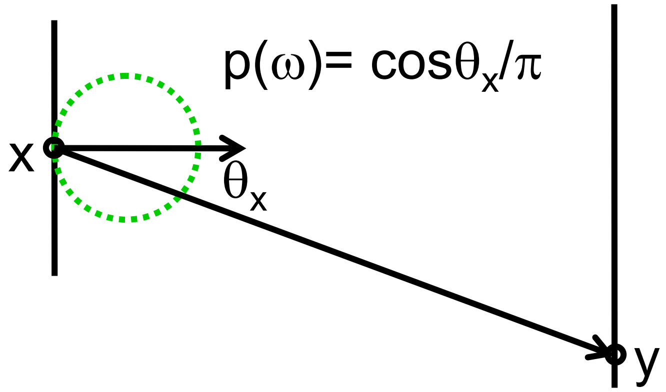


Multiple Importance Sampling (MIS)

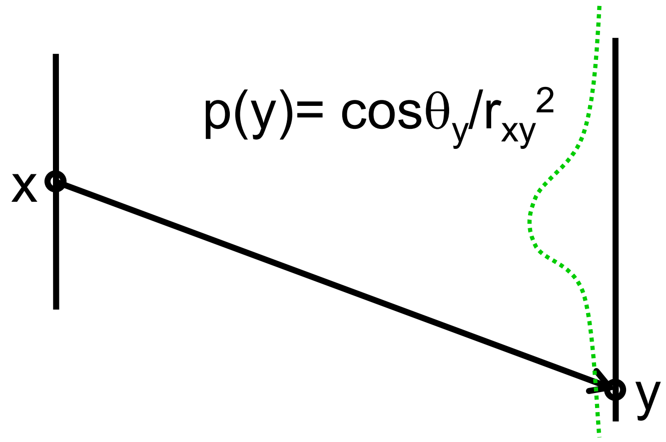
Veach & Guibas 1995

Combining multiple sampling techniques

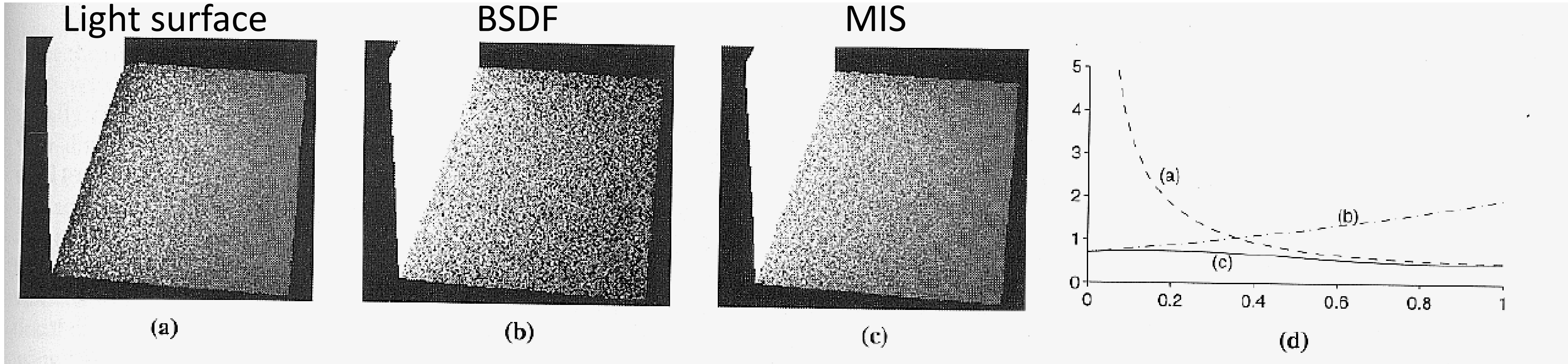
- Sampling the BSDF

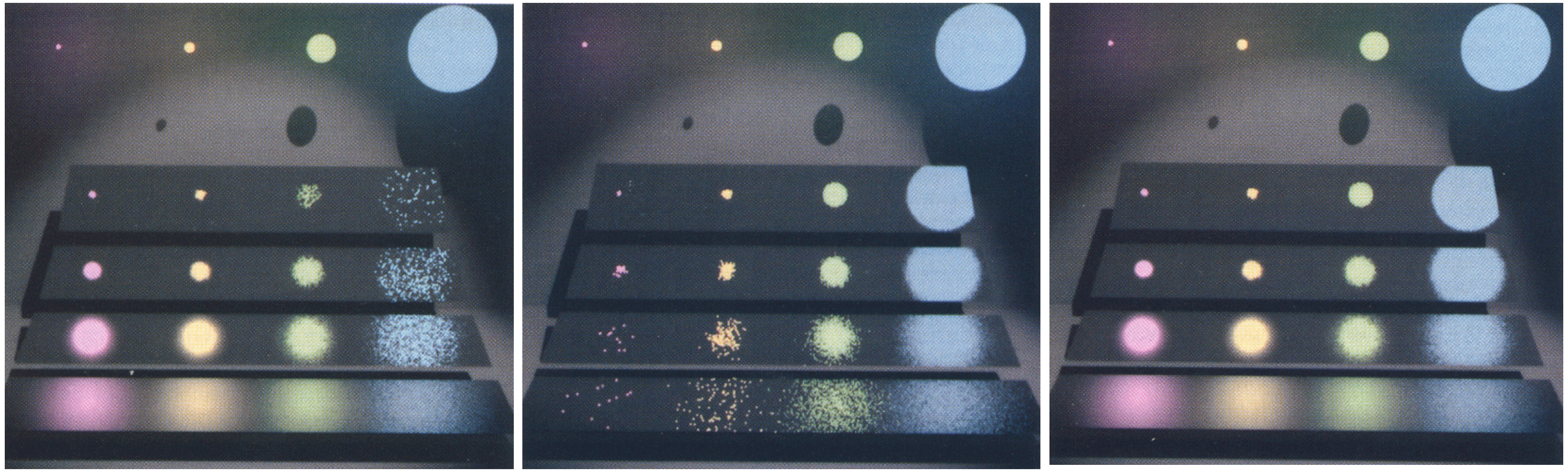


- Sampling the light surface



Both techniques perform well for different effects





Light surface BSDF MIS

How to combine them?

- Form an affine combination of the estimators:

$$\sum_{t \in T} \sum_{j}^{n_t} w_t(x_{t,j}) \frac{f(x_{t,j})}{n_t \, p_t(x_{t,j})}$$

- Theoretically, any weighting functions $w_t(x)$ work, provided:
- $w_t(x) = 0 \quad \text{if} \quad p_t(x) = 0$
- $\sum_t w_t(x) = 1 \qquad \forall x$

First, all densities need to be in the same measure

- $p(\omega) = \frac{\cos(\theta)}{\pi}$ has unit sr^{-1} (density on the hemisphere)
- $p(y)$ has unit m^{-2} (density on surface area)

- Applying the same logic as before, we can transform them:
- $p(y|\omega) = \frac{\cos(\theta)}{\pi} \frac{\cos(\theta_y)}{d^2}$

Balance heuristic

- Provably good choice: minimizes upper bound of the variance

$$w_t(x) = \frac{n_t p_t(x)}{\sum_k n_k p_k(x)}$$

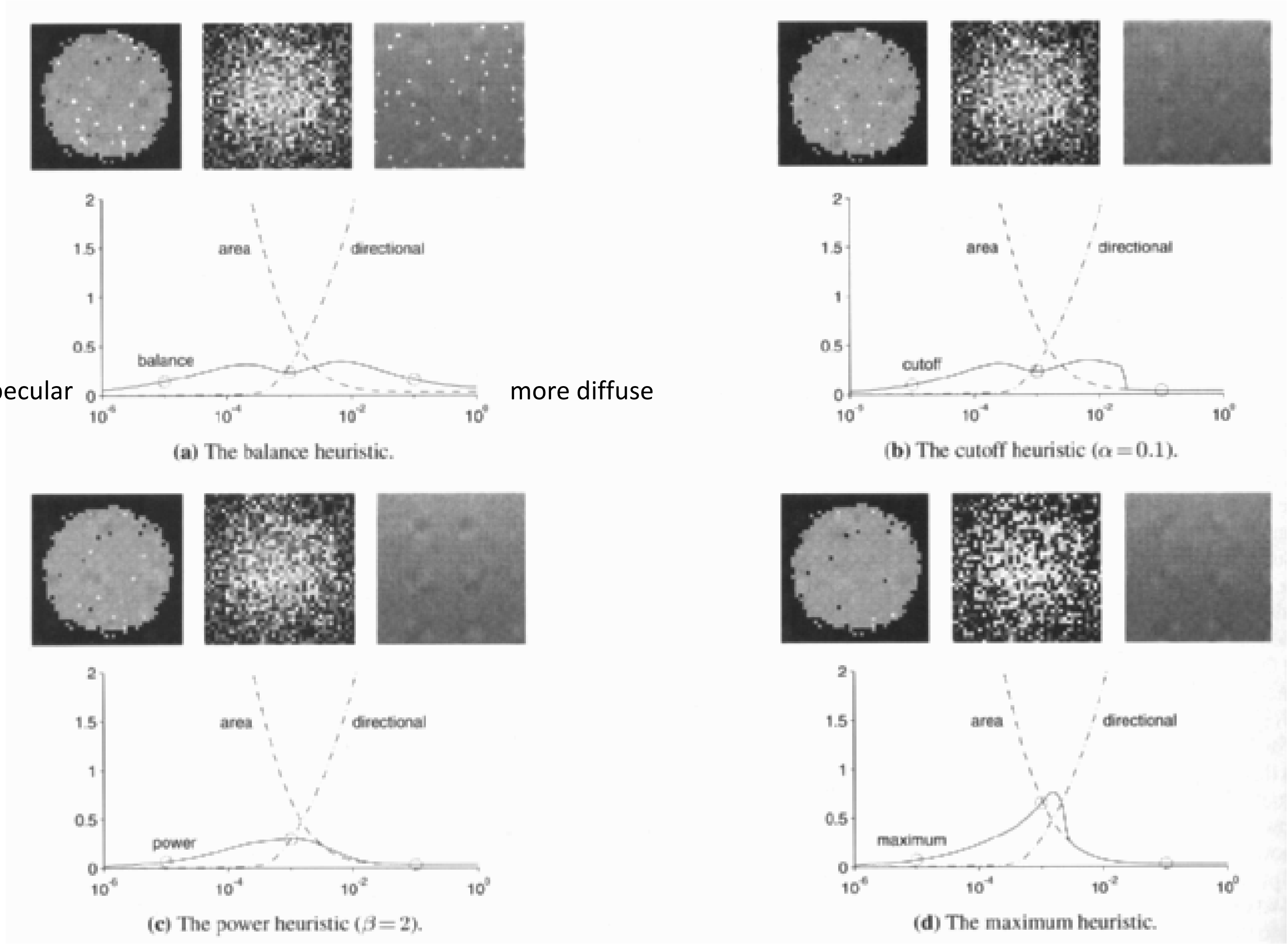
Power, maximum, and cuttoff heuristics

- Can sometimes perform better for low-variance techniques
- Amplify the weights to remove residual noise from “bad” techniques

- Maximum: only consider technique with largest $n_t p_t(x)$
- Cutoff: Balance, but set $n_t p_t(x)$ to zero if below some threshold
- Power heuristic: $w_t(x) = \frac{\left(n_t p_t(x)\right)^2}{\sum_k \left(n_k p_k(x)\right)^2}$

Comparison

– Image of a light source on surfaces with different roughness



Optimal weights

- It is possible to derive the optimal set of functions $w_t(x)$
- Result: linear system involving many integrals
- Kondapaneni et al 2019
- (Our recent SIGGRAPH paper)

Putting it all together

A less basic path tracer

- Given a point visible to the camera
- (1) Compute direct illumination
  - Select light source and point on the light
  - Trace a shadow ray for $V(x, y)$
  - Sample BSDF and check if a light was intersected
  - Compute MIS weights and add estimate

A less basic path tracer

- Given a point visible to the camera
- (1) Compute direct illumination
  - Select light source and point on the light
  - Trace a shadow ray for $V(x, y)$
  - Sample BSDF and check if a light was intersected
  - Compute MIS weights and add estimate
- (2) Terminate with Russian roulette

A less basic path tracer

- Given a point visible to the camera
- (1) Compute direct illumination
  - Select light source and point on the light
  - Trace a shadow ray for $V(x, y)$
  - Sample BSDF and check if a light was intersected
  - Compute MIS weights and add estimate
- (2) Terminate with Russian roulette
- (3) Continue the path
  - Could sample a new direction from the BSDF, or re-use the one from (1)
- Go back to (1), repeat until RR terminates

Many improvements possible

- Splitting (e.g., at first intersection)
- Combine multiple light sampling strategies (via MIS)
- Store statistics to approximate L_i for importance sampling ("guiding", later)
- And so on…