Stats <3 Graphics Recitation

Stats 101 & Graphics (outline)

- random variables
- discrete distributions
- continuous distributions
- pixel sampling and distributions
- BRDFs
- monte carlo method

Random Variable

For example, let’s toss a coin. If it’s heads I get a cookie. If it’s tails you get a cookie.

P(Y = “heads”) = ½

where **Y** is a random variable that can either be “heads” (I get a cookie) or “tails” (you get a cookie). If we were rolling dice instead, Y could be 1, 2, 3, 4, 5, or 6 and P(Y = 1) = ⅙, P(Y = 2) = ⅙, etc.

Discrete Distributions

In coin tosses and dice rolls, our random variable has a *countable* number of options so we call it *discrete*.

- **probability mass function** (pmf) tells us the probability of a discrete random variable:
- **p(a) = P(X = a)**
  - where X is our random variable and a is the value of X.
- For dice, p(1) = P(X = 1) = ⅙ and p(-1) = P(X = -1) = 0 because X can never be anything other than 1 through 6

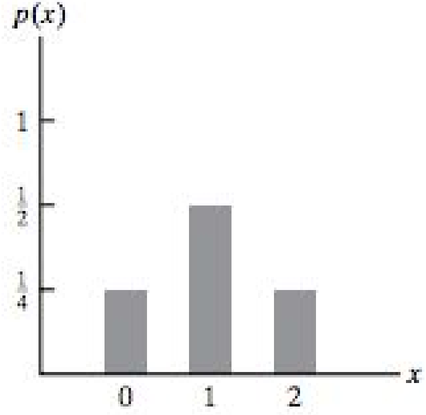
Real quick! Let’s draw the pmf of the following:

Flip a coin 3 times. Let X be the number of heads that appear.

Example:

Flip 3 fair coins. Let X = the number of heads that appear.

$$p(0) = \frac{1}{4} \qquad p(1) = \frac{1}{2} \qquad p(2) = \frac{1}{4}$$



Common Discrete Distributions

- Bernoulli
- Binomial
- Poisson
- Geometric
- Zeta

Continuous Distributions

Suppose you're playing darts and you are terrible at it in a uniform way (your chances of hitting any point on the board are equal). We can let X = the distance from the bullseye. Now X can take an *infinite* number of values between 0 and the radius of the dart board. So X is a *continuous* random variable.

- a **probability density function** (pdf) tells us the probability of a continuous random variable:
- **$P\{X \in B\} = \int_B f(x)\,dx$**
- Note that the probability that X = a is 0 because
  - $P(X = a) = \int_a^a f(x)dx = 0$

Example time!

Suppose you, blindfolded, decide to chop a 10cm stick with a large kitchen knife. The point you choose to chop the stick is uniform between 0 and 10.

- What's the probability you choose the first half of the stick?
- What's the probability you chop it into two pieces that are roughly equal?
- What's the probability you chop it exactly in the middle?

These should be easy to guess, but let's see how they're set up.

Note that uniform pdf is:

$$\begin{cases} \frac{1}{b-a} & \text{for } x \in [a, b] \\ 0 & \text{otherwise} \end{cases}$$

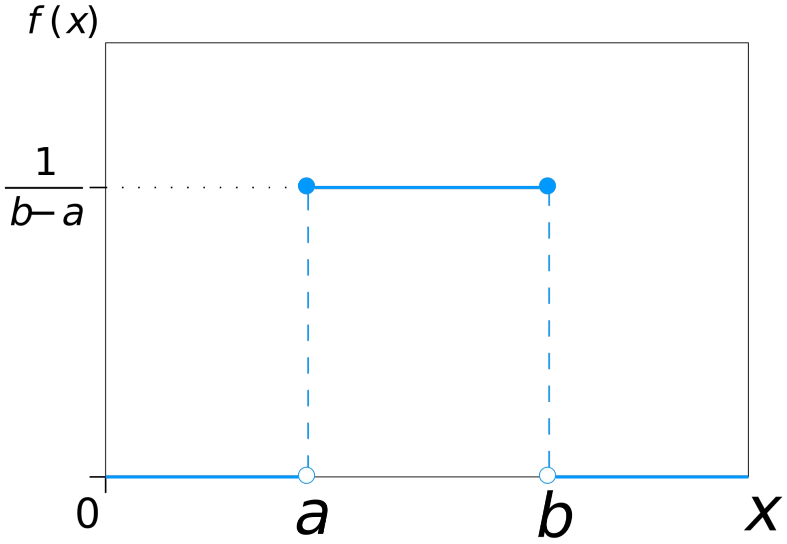
Example Continued

- What's the probability you choose the first half of the stick? *Guess is ½*
    - The uniform [0, 10] pdf looks like **$\int 1/b\text{-}a\,dx$** = $\int .1\,dx$
    - $f(0 < x < 5\text{cm}) = \int_0^5 .1\,dx =$ **.5**
- What's the probability you chop them roughly equally? What is roughly? Let's say within **1cm**:
    - $f(4.5 < x < 5.5) = \int_{4.5}^{5.5} .1\,dx =$ **.1**
- What's the probability you chop it exactly in the middle?
    - $f(x = 5\text{cm}) = \int_5^5 .1\,dx =$ **0**

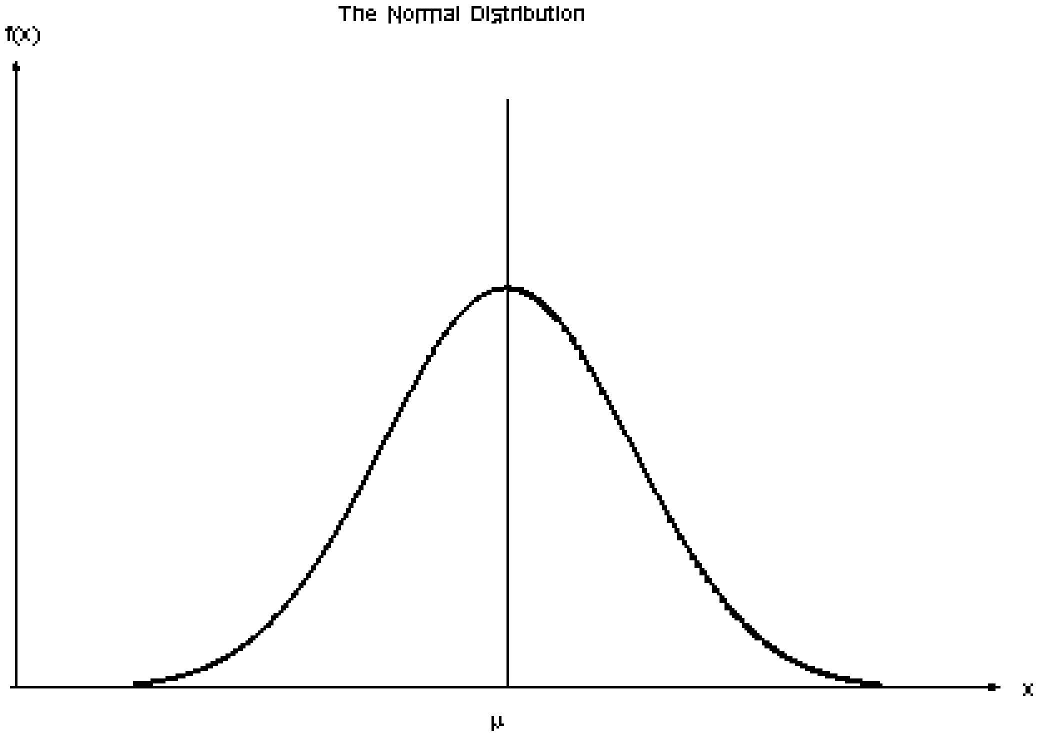
Continuous Distributions

Maybe you’re not blindfolded anymore. You might have a tendency to chop the stick closer to the middle, closer to the end, or maybe your tendencies correlate to a sin wave? If we know the distribution, we can still get an accurate probability.

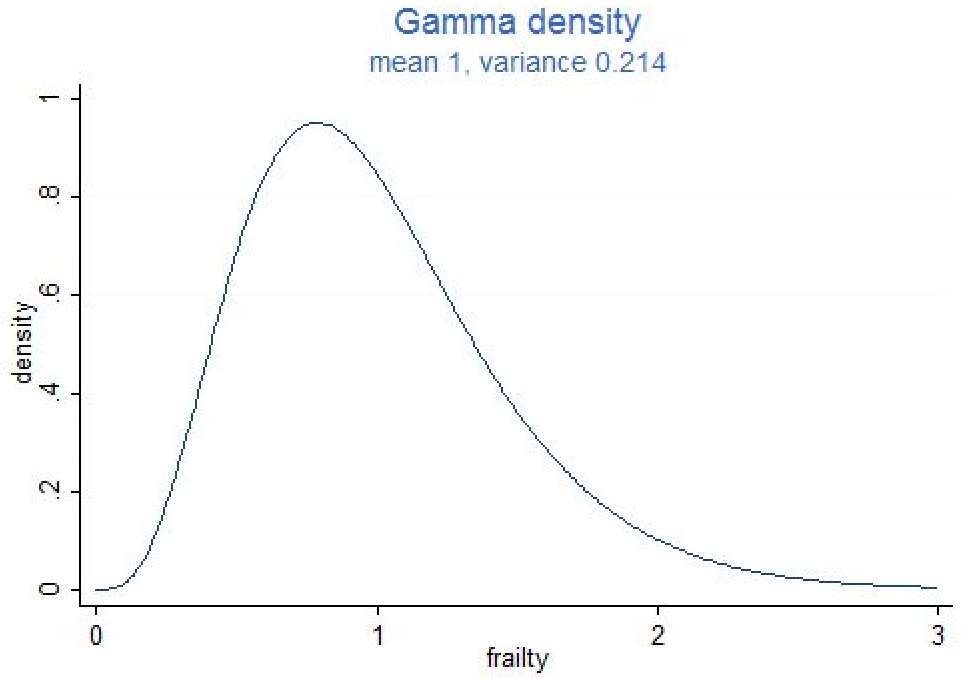
Uniform



Normal (or Gaussian)



Gamma


Gamma density
mean 1, variance 0.214
density
0
.2
.4
.6
.8
1
0
1
2
3
frailty

Exponential

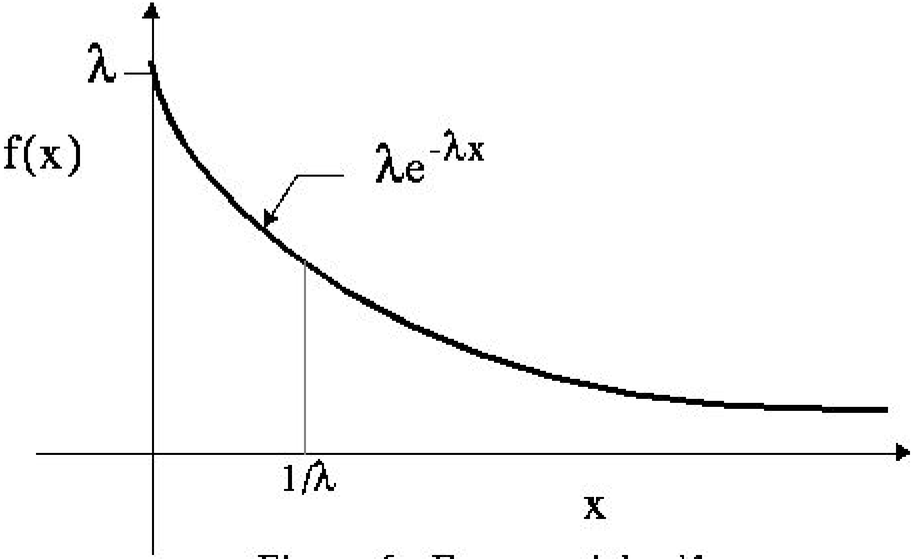


Figure 6. Exponential *pdf*

Any many more!

- Another useful one for raytracing is a **cosine-weighted distribution**, which we'll get to a little later.

Variance and Standard Deviation

Standard deviation:

$$\sigma = \sqrt{\int_{\mathbf{X}} (x-\mu)^2 \, p(x) \, dx}, \quad \text{where} \quad \mu = \int_{\mathbf{X}} x \, p(x) \, dx,$$

Variance:

$$\operatorname{Var}(X) = \sigma^2 = \int (x-\mu)^2 \, f(x) \, dx \; = \int x^2 \, f(x) \, dx \; - \mu^2$$

Expected Value

- is the probability weighted average of all possible values
- For discrete: $\operatorname{E}[X] = x_1 p_1 + x_2 p_2 + \cdots + x_k p_k \, .$
  - Example: for a dice roll E[X = number on the die] = 1*(⅙) + 2 * (⅙) + 3 * (⅙) + 4* (⅙) + 5*(⅙) + 6 * (⅙) = 3.5
- Similarly, for continuous: $\operatorname{E}[X] = \int_{-\infty}^{\infty} x f(x) \, \mathrm{d}x.$
  - For our twig chopping example: E[X = where we chop the twig] = (1/10)∫x dx = (1/20) $*x^2|_0^{10} = 5$
- There are lots of other fun properties of expected value too. Expected value is useful for when the actual distribution is hard to compute.

Some more Expected Value

Linearity:

$$\mathrm{E}[X+c] = \mathrm{E}[X] + c$$

$$\mathrm{E}[X+Y] = \mathrm{E}[X] + \mathrm{E}[Y]$$

(holds even if X is not independent of Y)

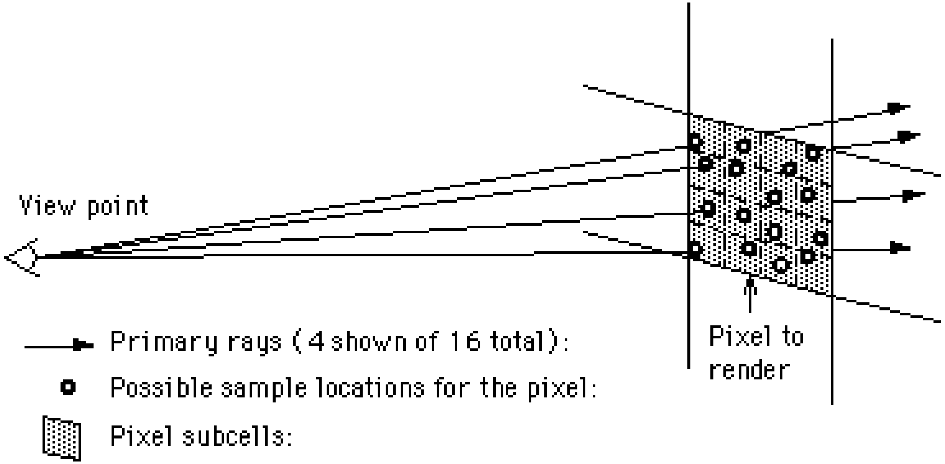
$$\mathrm{E}[aX] = a\,\mathrm{E}[X]$$

So, if you have two functions, f(x) and g(x), we can evaluate them individually.
E[f(x)] + E[g(x)] = E[f(x) + g(x)]

There is an example of an application of this in Ch. 15.1 of Physically Based Rendering (pg. 746)

Distributions and Pixel Sampling

Spaces to render are not discrete so sampling one point per pixel will not give us smooth results. As you’ve done in your pixel samplers, you can use random or uniform distributions to get multiple samples in a pixel.

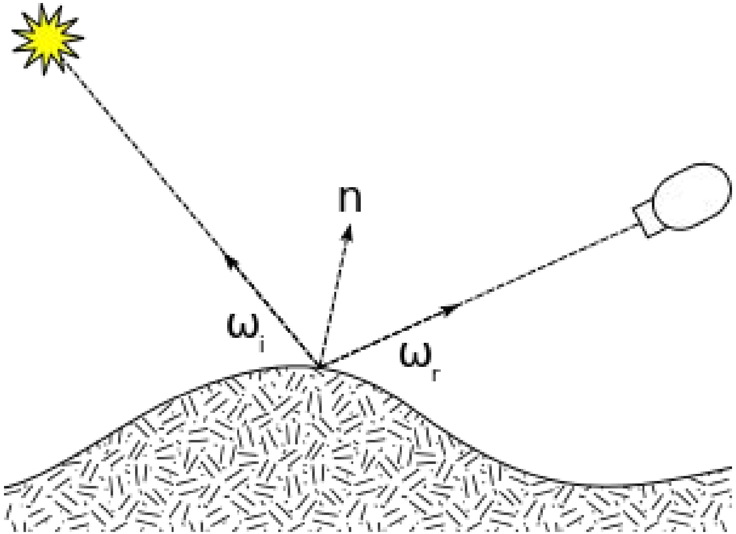

View point
Pixel to render
Primary rays (4 shown of 16 total):
Possible sample locations for the pixel:
Pixel subcells:

Distributions and BRDFS

The reflectance of rays are not uniform, so we need a better distribution. We use BRDFs, defined as follows:

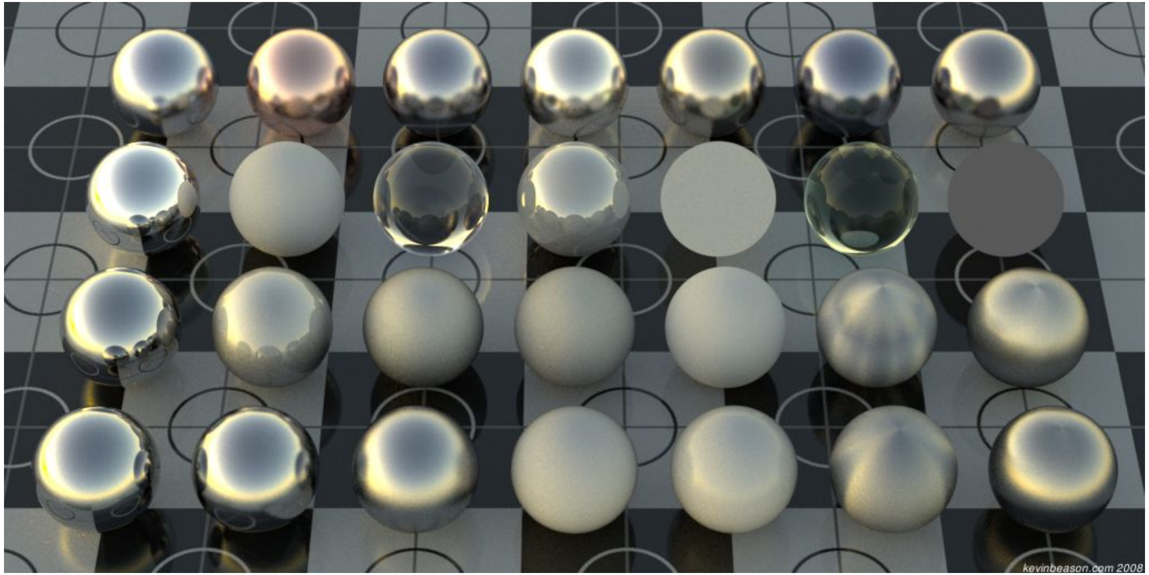
$$f_{\mathrm{r}}(\omega_{\mathrm{i}},\,\omega_{\mathrm{r}}) \;=\; \frac{\mathrm{d}\,L_{\mathrm{r}}(\omega_{\mathrm{r}})}{\mathrm{d}\,E_{\mathrm{i}}(\omega_{\mathrm{i}})} \;=\; \frac{\mathrm{d}\,L_{\mathrm{r}}(\omega_{\mathrm{r}})}{L_{\mathrm{i}}(\omega_{\mathrm{i}})\cos\theta_{\mathrm{i}}\;\mathrm{d}\,\omega_{\mathrm{i}}}$$

L: radiance (light leaving a point)
E: Irradiance (light arriving at a point)
$\boldsymbol{\theta}_{\mathbf{i}}$: angle between ω_{i} and n
r means reflected light and i means incident light



Result of BRDFs

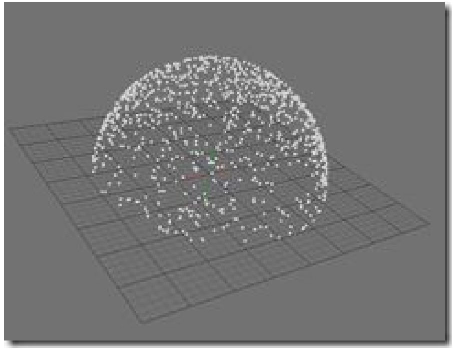
Changing the distribution of light reflecting off a material gives it completely different looks:



Sampling Emitted Rays - Uniform

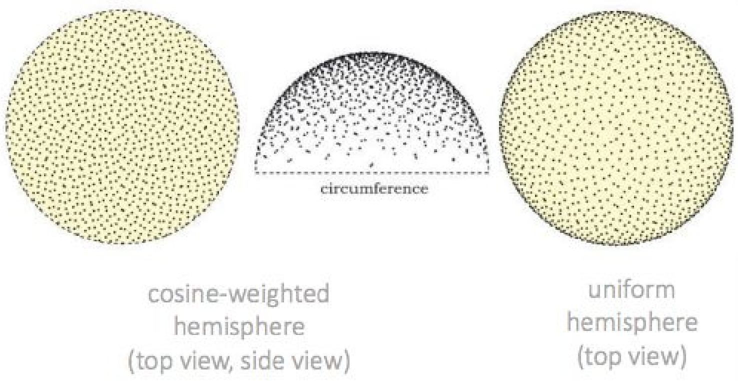
In path tracing, when a light ray hits an object, it can either be emitted or absorbed. This gives us global illumination!

- If the light is absorbed, we stop there.
- But if a light is emitted, how do we know which direction? *Sampling!*
  - We can use **uniform samples** around a hemisphere to get a new direction.
  - This is noisy.



Sampling Emitted Rays - Cosine Weighted

- Get better, less noisy results by uniformly sampling a cosine-weighted hemisphere.
- Why?
  - Look at the integral for diffuse (lambertian):
  - $$L_o = \int \frac{c}{\pi} L_i \cos\theta, d\omega$$
  - Many samples will end up getting canceled out by the cosine term, so weight the samples so you include less of those.


circumference
cosine-weighted hemisphere (top view, side view)
uniform hemisphere (top view)

Cosine Weighted Distribution

- Cosine weighting will give you more sample in the direction of the normal.
    - Weighted by the cosine of the angle between the reflected ray and the normal
- This also means you don’t multiply by cos(angle) because it’s already cosine weighted

Monte Carlo Method

- In general, the Monte Carlo method just says that a random variable can be approximated by the empirical mean.
  - Silly example: TODO!!
  - Graphics example: Similarly, light rays bouncing around a scene do not always result in the same light color result (what you will color your pixel). Sometimes a ray might get absorbed by an object right away, sometimes it bounces off objects a few times before getting absorbed. The color can be estimated by taking many samples and averaging them out.

Monte Carlo Method

- To define it more math-y
- let $f_X(x)$ be the distribution (pdf). Then the expected value of g(X) is:

$$\mathbb{E}(g(X)) = \int_{x\in\mathcal{X}} g(x) f_X(x)\,dx$$

We take n samples of x (x_1,x_2 … x_n) and find the mean of g(x) over the sample:

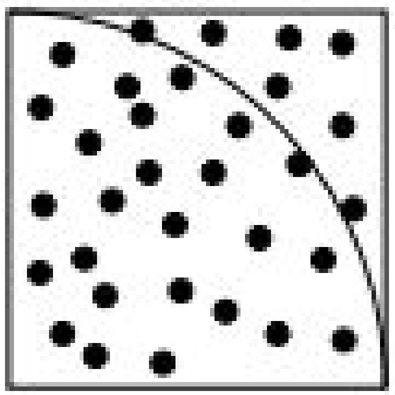
$$\tilde{g_n}(x) = \frac{1}{n}\sum_{i=1}^{n} g(x_i)$$

This is the “monte carlo estimate”

Example: Computing

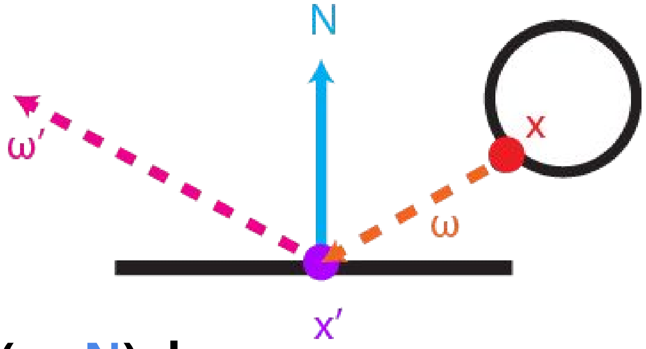
Suppose you have a ¼ unit disk inside a unit square.

- Choose a random point in the square. What is the probability it lands in the disk? $A_{1/4\ \text{circle}}/A_{\text{square}}$ = **$\pi/4$**
- Take a large sample of random points on the square, and solve for **π**.
  - **π ≈ 4*(points in disk/total points)**
- **We can do the same with the rendering equation.**



http://www.cs.rpi.edu/~cutler/classes/advancedgraphics/S11/lectures/13_monte_carlo.pdf

Monte Carlo Path Tracing



Rendering equation:

$$L(x',\omega') = E(x',\omega') + \int_{\Omega} \rho x' (\omega,\omega')\, L(x,\omega)\, G(x,x')\, \mathrm{dot}(\omega, N)\, d\omega$$

- L: radiance from a point **x'** and direction **ω'**.
- E: emitted radiance from a point (nonzero only if **x'** is a light source; all other light is *reflected*)
- ρ: BRDF of surface at **x'**
- L: radiance at some point x in direction **ω**.
- V: visibility between **x** and **x'** (1 when unobstructed, 0 otherwise)
- We incorporate the dot product of **ω** and **N** due to Lambert's law
- Ω is the hemisphere of all possible **ω** that can reach **x'**

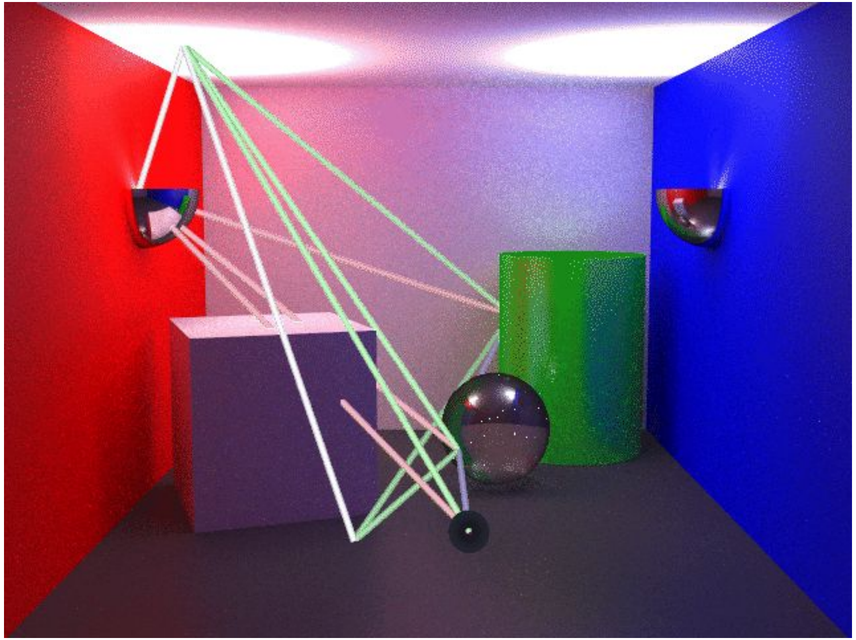
We approximate the output of this equation by sampling many times and taking the average.

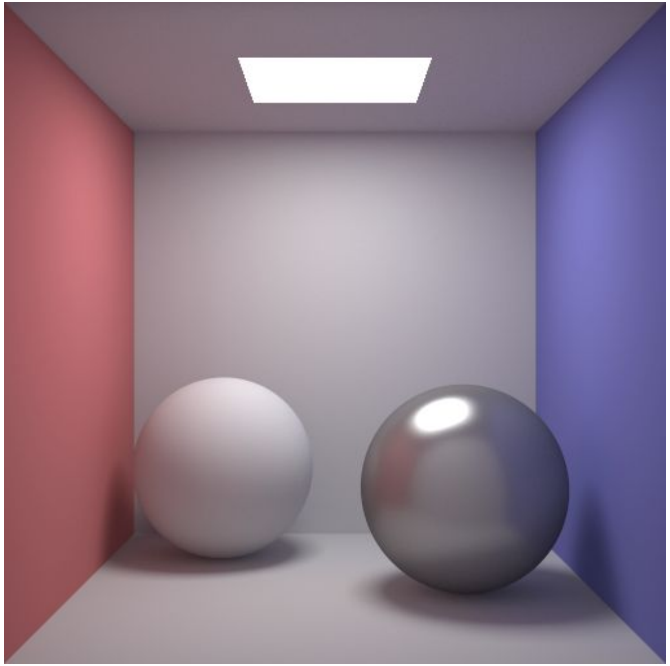
Monte Carlo Path Tracing - Basic Algorithm

- Choose a ray, give it a “weight” (starting with 1)
- Trace the ray to find intersection with surface
- If the surface is emissive
    - return the light color * weight
- If surface is not emissive, then reflect
    - weight = weight * reflectance (e.g. mirrors can reflect 100% without getting duller, but diffuse materials cannot)
    - Randomly scatter the ray according to the surface’s BRDF
    - Go back to the trace ray *step.

http://cs.brown.edu/courses/cs224/papers/mc_pathtracing.pdf

Monte Carlo Path Tracing





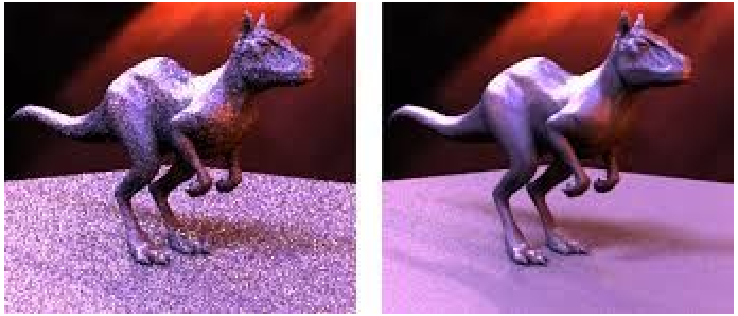
Variance Reduction & Reducing Noise

There are various *variance reduction* techniques, that give more precise samples. In path tracing, this results in less noisy pictures. (these are all outlined in PBRT Ch. 14 by the way)

- **Russian Roulette** -throw out samples too close to 0
- **Splitting** - split samples into image samples and light samples per image sample
- **Bias** - use an estimator
- **Importance Sampling** - Take a distribution more similar to f(x) in the integrand to converge faster
- **Multiple Importance Sampling** - Like importance sampling, but for integrals of the form $\int f(x)g(x)dx$

Importance Sampling

- The Monte Carlo estimator converges more quickly if samples are taken from a distribution similar to f(x)
  - $F_N = (1/N) * \sum f(X_i)/p(X_i)$ = monte carlo estimator
- The perfect pdf requires us to know the value of the integral, which is what we’re trying to compute in the first place. What the heck!?
  - We have to find a pdf similar to f(x) without knowing f(x) ahead of time - there are various methods to do this for sampling BSDFs, light souces, and other functions. (PBRT pg. 733 if you’re interested).

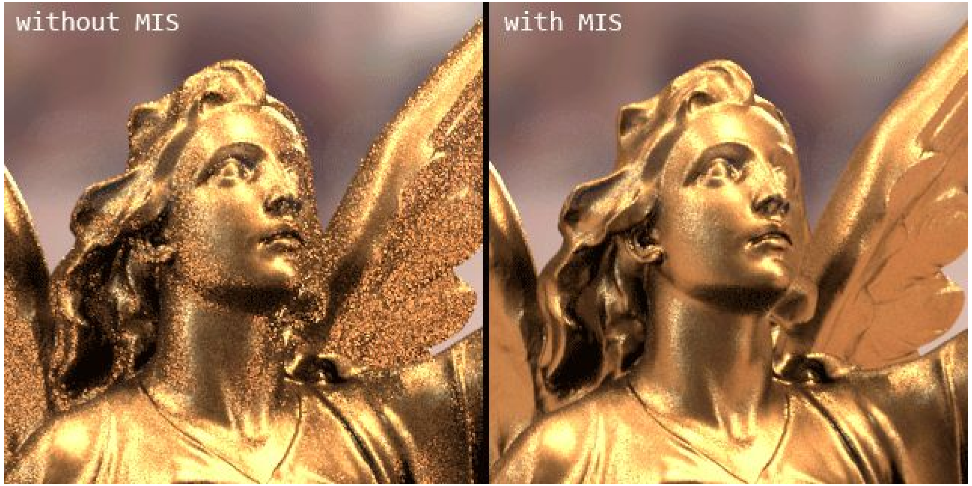


Multiple Importance Sampling

What if you have an importance sampling strategy for f(x) and one for g(x), but you need one for f(x)*g(x)?

- Getting a new pdf for f(x)*g(x) isn’t always possible or easy
- Idea: when estimating an integral, draw samples from multiple distributions, with the hope that at least one will match the integrand well.
- Weight the samples from each distribution eliminates variance spikes.
- See PBRT pg. 690

Multiple Importance Sampling (MIS)


without MIS
with MIS