Rendering: Monte Carlo Integration

Adam Celarek

Research Unit of Computer Graphics
Institute of Visual Computing & Human-Centered Technology
TU Wien, Austria

Hi and welcome to this rendering lecture.
My name is Adam Celarek and I will be talking about Monte Carlo integration..
Let’s begin with the roadmap

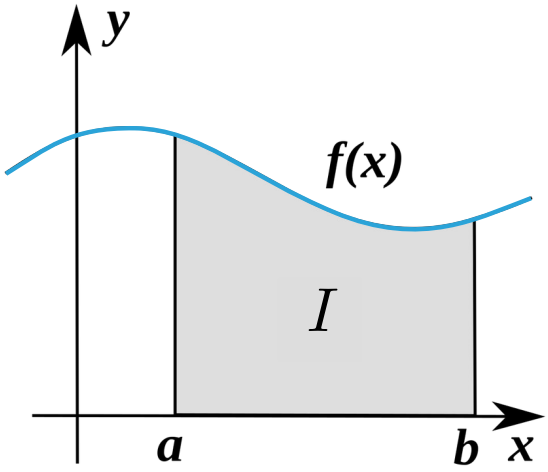
Roadmap

- Rectangle Rule for Numerical Integration
- Look at problems and why it's not suitable for us
- Monte Carlo Estimator
- Variance and Importance Sampling
- What about the Hemisphere?
- Apply

We'll first look at the rectangle rule for numerical integration. You probably already know this quadrature rule from one of the math courses.
There are some problems with that rule when we increase the number of dimensions, however, which make it unusable for rendering.
That's why we'll introduce the Monte Carlo method for Integration.
We then look at some fundamental propperties and learn how to use it in practise.

Ok, let’s start with the rectangle rule

Rectangle Rule for Numerical Integration

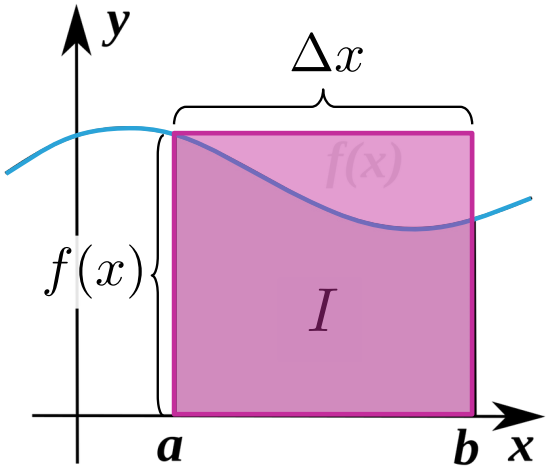


$$I = \int_a^b f(x)\,\mathrm{d}x$$

source: User 4C / Wikipedia
CC BY-SA 3.0

Here we see a function f(x), that we want to integrate numerically between a and b. In other words, we want to compute the area I under the curve.

Rectangle Rule for Numerical Integration

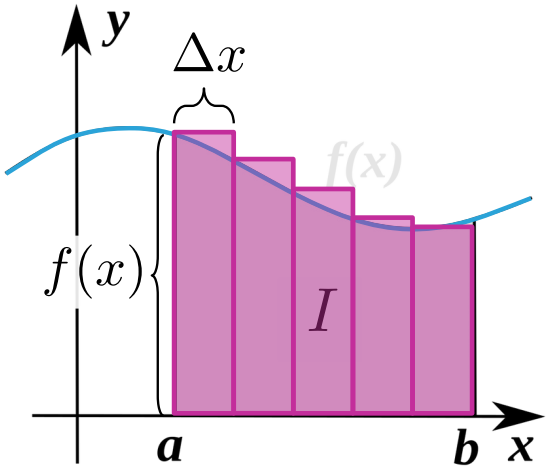


$$I \approx f(x)\Delta x$$

source: User 4C / Wikipedia
CC BY-SA 3.0

A very rough approximation would be to compute the area of a rectangle with height f(a) and width b-a

Rectangle Rule for Numerical Integration

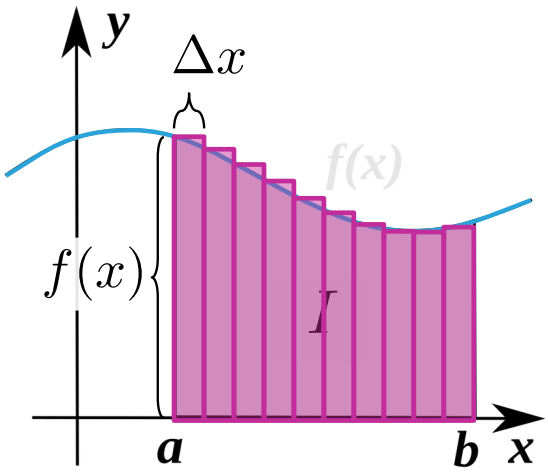


$$I \approx \sum_{i=0}^{N} f(x_i)\Delta x$$

source: User 4C / Wikipedia
CC BY-SA 3.0

Which becomes more precise if we subdivide the rectangle into N parts as shown.

Rectangle Rule for Numerical Integration



$$I \approx \sum_{i=0}^{N} f(x_i)\Delta x$$

$$\Delta x = \frac{b-a}{N}$$

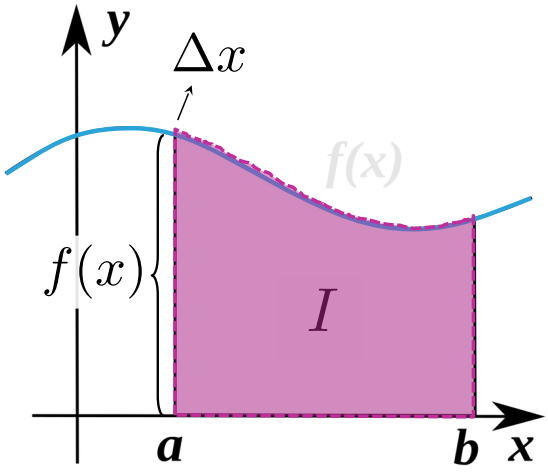
$$I \approx \frac{b-a}{N} \sum_{i=0}^{N} f(x_i)$$

source: User 4C / Wikipedia
CC BY-SA 3.0

And quite obviously, the accuracy increases while N grows.

Delta x equals (b-a) / N and x_i are equidistant samples of f in the domain from a to b. Since (b-a)/N is independent of i, we can pull it out of the sum.

Rectangle Rule for Numerical Integration



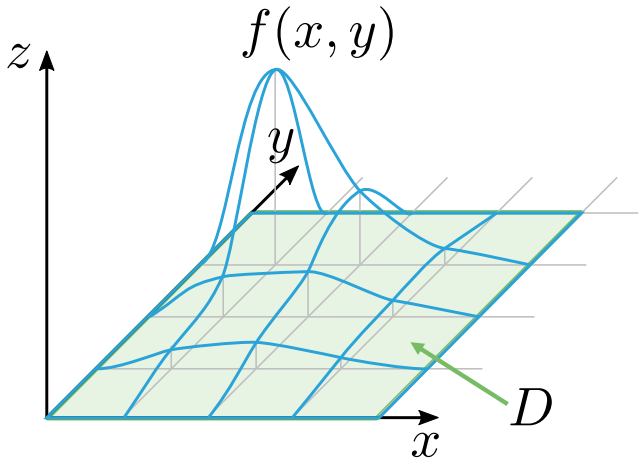
$$I = \lim_{N \to \infty} \frac{b-a}{N} \sum_{i=0}^{N} f(x_i)$$

source: User 4C / Wikipedia
CC BY-SA 3.0

Going to infinity would give us the true value of the integral.

That was a one dimensional function (f depends only on x, not on x and y), ..

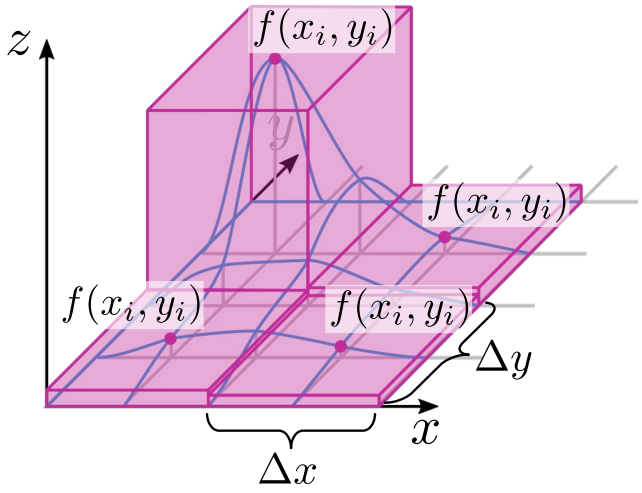
Rectangle Rule for Numerical Integration



$$I = \int_D f(x,y)\,\mathrm{d}x\,\mathrm{d}y$$

But it can be easily extended to 2 dimensions. F now depends on two variables, x and y and we have a volume instead of an area.

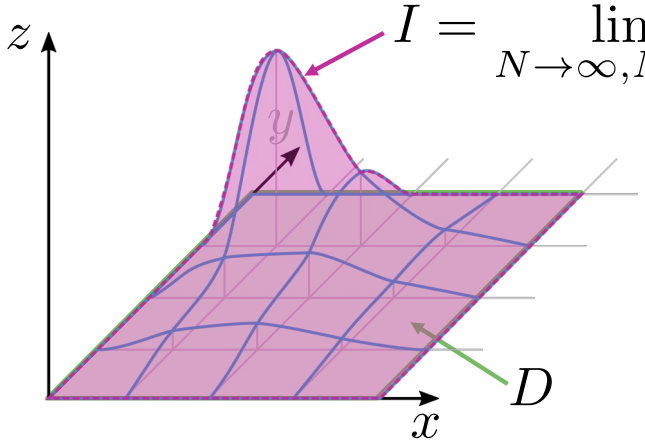
Rectangle Rule for Numerical Integration



$$I \approx \sum_{i=0}^{N} \sum_{j=0}^{M} f(x_i, y_i) \Delta x \Delta y$$

Before we had rectangles, now we have boxes with equal base rectangles (delta x times delta y) and heights f(x, y).
And we have to sum over a 2d array of boxes..

Rectangle Rule for Numerical Integration



$$I = \lim_{N\to\infty, M\to\infty} \left(\frac{D}{MN} \sum_{i=0}^{N} \sum_{j=0}^{M} f(x_i, y_i) \right)$$

Again, going with the number of boxes towards infinity would give us the correct result.

Rectangle Rule for Numerical Integration

Also works for 3 and more dimensions..

That would also work for 3 and more dimensions.
The concept stays the same but is harder to imagine..

Rectangle Rule for Numerical Integration

Also works for 3 and more dimensions..
However!

However!

There are some unavoidable problems in higher dimensions (like with this eucalyptus forest, that saw too many koalas).

And in rendering, you have many many dimensions..

Look at problems and why it's not suitable for us

Sampling regular positions in 1d

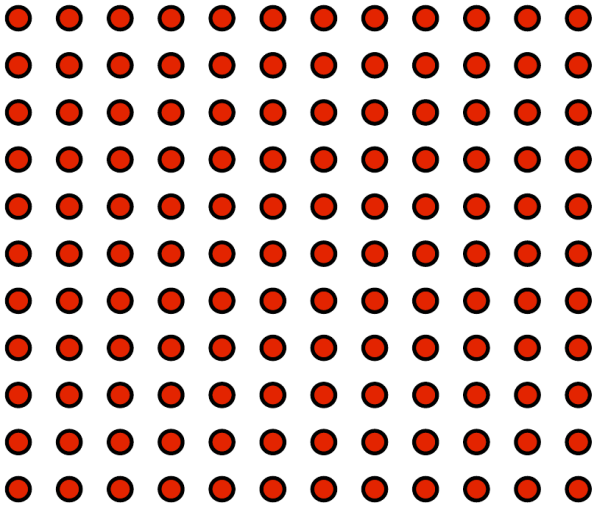
n

Slide modified from Jaakko Lehtinen, with permission

So these are sampling positions for a 1 dimensional function.

Look at problems and why it's not suitable for us

Sampling regular positions in 2d

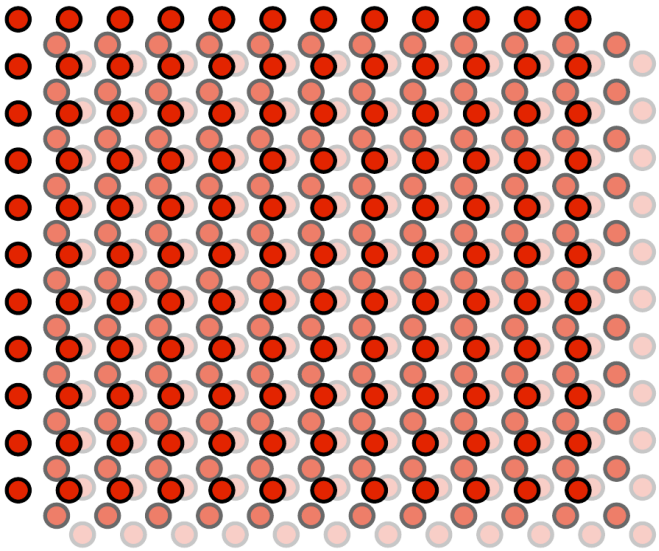


n^2

If you keep N the same and go to 2 dimension, you see that we have an O(N^2) algorithm..

Look at problems and why it's not suitable for us

Sampling regular positions in 3d



n^3

Slide modified from Jaakko Lehtinen, with permission

Going to 3d, looks already quite bad.

Look at problems and why it's not suitable for us

Sampling regular positions in many more dimensions

If have even more dimensions, let's say 20 (and that would be common in rendering), erm

Look at problems and why it's not suitable for us

Sampling regular positions in many more dimensions

The result is just not nice!

Look at problems and why it's not suitable for us

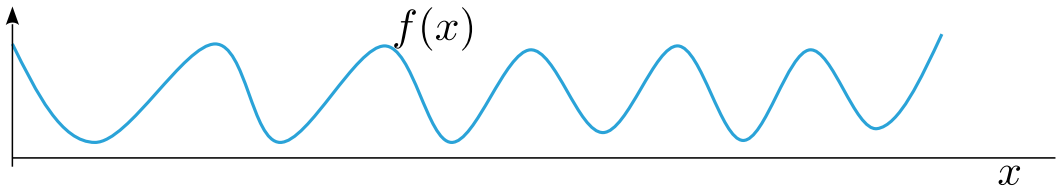
Sampling regular positions in many more dimensions

Okey, hm, but who said our **Δx** needs to be that small?

Maybe we can accept less precision and have a larger **Δx**?

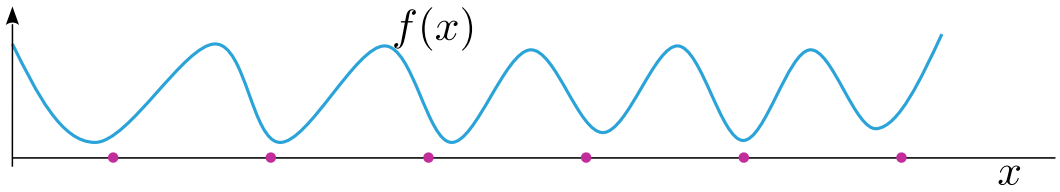
Ok cat, couldn't we just trade computation for precision?
Increasing delta x, decreasing N, maybe the accuracy will be enough?

Look at problems and why it's not suitable for us



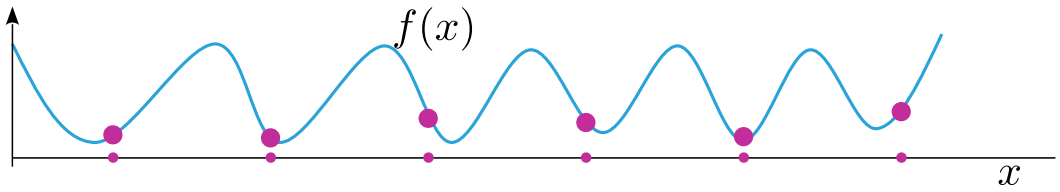
Let's check out this example. It's a bit artificial since it's just 1d, but actually it's the exact thing that would happen in higher dimensions..

Look at problems and why it's not suitable for us



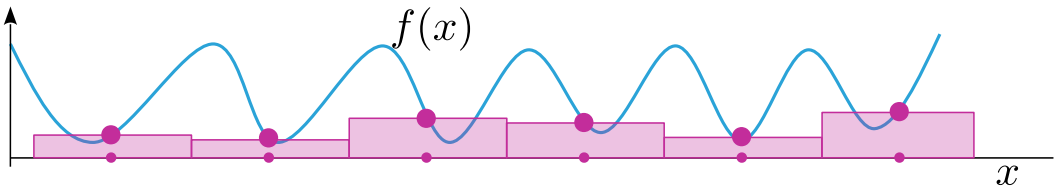
There you have our sampling positions

Look at problems and why it's not suitable for us



The corresponding function values

Look at problems and why it's not suitable for us



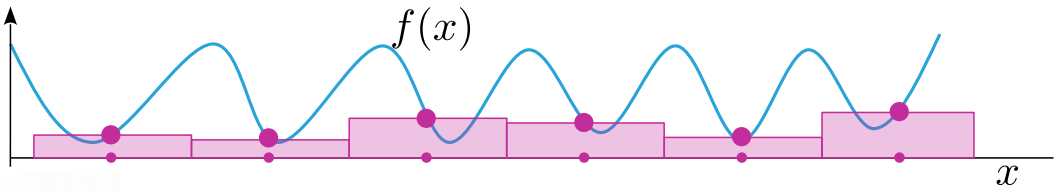
And boxes.

Dear cat, we have a problem..
Do you see it?

Look at problems and why it's not suitable for us

The computed value will be much smaller than the actual Integral.

Aliasing!



Usually some boxes would underestimate, some overestimate. Here we are quite unlucky and all boxes are underestimating. So the result would be also underestimated.

That's just by chance, if the phase of the function would be slightly different (it would be a bit further to the left or the right), it might be completely ok, or it might overestimate..

This is one form of aliasing, and it's hurting cat's eyes..

Look at problems and why it's not suitable for us

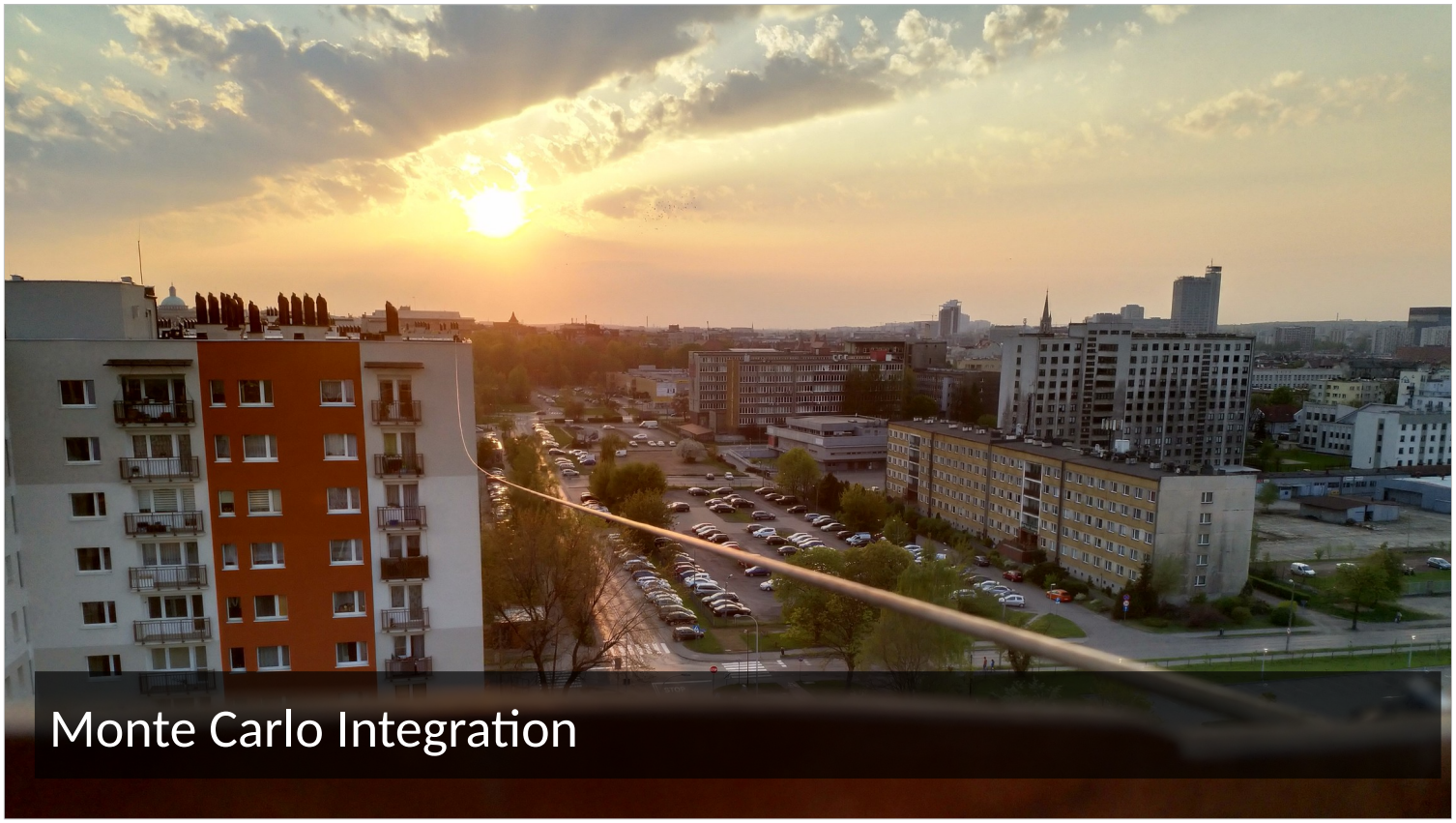
Aliasing!

- Sampling frequency too low (google Nyquist frequency)
- We could apply a low-pass filter before sampling
- Or we could find a better method – a better estimator..

To be precise, aliasing happens when the sampling frequency is too low. That is based on concepts in signal processing. Basically a sampler, in signal processing, converts a continuous signal into a discrete one. There is the sampling frequency, and the Nyquist frequency, which is half of the sampling frequency. If the signal contains components with a frequency higher than the Nyquist frequency, then there will be artifacts in the discrete signal. This is called aliasing. Now, the function that we want to integrate is the continuous signal, and we process the sampled discreet signal immediately by summing up. Nevertheless, aliasing happens and we would see it in the rendered picture.

A common method to combat aliasing is to apply a low pass filter before sampling, cutting away the higher frequencies.

But that would be non trivial actually, and we have a better method, we will describe a better estimator. better for high dimensional integrals at least.



Monte Carlo Integration.
Or, another term would be Monte Carlo estimator (for integration).

What's an Estimator?

"An estimator is a rule for calculating an estimate of a given quantity based on observed data..

For example, the sample mean is a commonly used estimator of the population mean."

Wikipedia

But what's an estimator precisely? I mean clearly it estimates something..

According to wikipedia:

...

The rectangle rule we saw before is also an estimator..

What's an Estimator?

"An estimator is a rule for calculating an estimate of a given quantity based on observed data..

For example, the sample mean is a commonly used estimator of the population mean."

Wikipedia

And, we might add, that most estimators produce a stochastic variable.

We might add, that most estimators are stochastic variables as well (the rectangle rule one is not stochastic).

Erm,

What's an Estimator?

"An estimator is a rule for calculating an estimate of a given quantity based on observed data..

For example, the sample mean is a commonly used estimator of the population mean."

Wikipedia

And, we might add, that most estimators produce a stochastic variable,

"a variable whose values depend on outcomes of a random phenomenon".

Wikipedia

What were stochastic variables again?

Wikipedia: ..

You probably guess correctly that we are going random now :)

What's an Estimator?

- In our case:
  - The random phenomenon is the process of taking samples
    Strictly speaking, equidistant sampling is not random. Therefore, the rectangle rule estimator is not a stochastic variable.
  - The rectangle rule estimator is actually a mean

In our case, we will make the sampling positions random.

And, algorithms that use random numbers are often called Monte Carlo algorithms, not only those for integration. The name stems from the casino in Monte Carlo i guess.

Wikipedia gave the example of the sample mean as an estimate for the population mean. When we look at the rectangle rule estimator we see exactly that, it's actually a mean!

What's an Estimator?

- In our case:
  - The random phenomenon is the process of taking samples (don't be nit-picky)
  - The rectangle rule estimator is actually a mean
  - I mean, look at it

$$I \approx \frac{b-a}{N} \sum_{i=0}^{N} f(x_i)$$

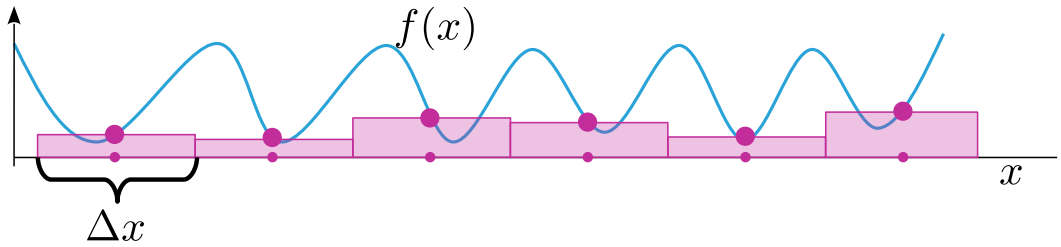
$$= \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$

Look at it!
We can push the (b-a) back in to make it even more clear..

What's an Estimator?

Estimating the integral using a mean of equidistant samples!

$$I \approx \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$



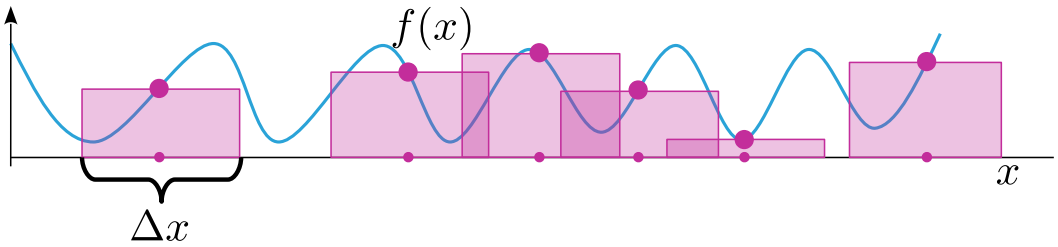
Here we have it again, the problematic example from before..

And we said that equidistant can be a problem, so maybe

What's an Estimator?

Estimating the integral using a mean of random uniform samples!

$$I \approx \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$

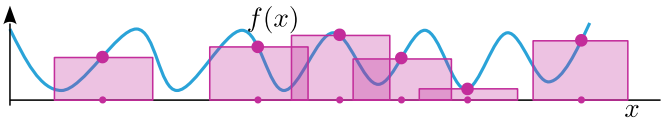


We can keep the number of samples the same, use the same delta x, but randomise the positions. It looks about right, no?

What's an Estimator?

Estimating the integral using a mean of random uniform samples!

$$I \approx \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$



Does that actually work?
Won't the result just be some random number?

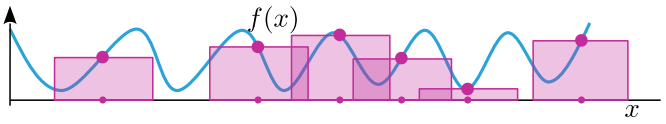
But the cat is critical. Will it actually work? The change in the algorithm is just few lines of code, but mathematically it's something totally different. No guarantees, we might just get a random number!

In short

What's an Estimator?

Estimating the integral using a mean of random uniform samples!

$$I \approx \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$



Does that actually work?
Won't the result just be some random number?

Yes!
We will prove it

Yes. It works.

But still, we have to prove it.

And unfortunetely,

Statistical Recap

- If you remember „107.254 Statistik und Wahrscheinlichkeitstheorie“ well enough, you can skip most of it.
- In daily life, we are mostly confronted with discrete random results
  - A coin flip
  - Toss of a die
  - All those combinatorial things (nightmarish for some cats)
  - Each possible outcome of a random variable is associated with a specific probability. Probabilities must sum up to 1.

$$\sum_{i=0}^{N} p_i = 1$$

We’ll need a bit of statistics for that.

If you remember statistics pretty well, at tu wien that would be the bachelor course Statistic und Wahrscheinlichkeitstheorie, you can skip most of it.

Otherwise, fear not, the cat is with you ;)

In daily life we mostly see discrete randomness, for instance a coin flip, a toss of a die, the casino etc. You probably remember that there is a set of possible outcommes, each with a certain probability and probabilities sum up to 1.

Statistical Recap

- If you remember „107.254 Statistik und Wahrscheinlichkeitstheorie“ well enough, you can skip most of it.
- In daily life, we are mostly confronted with discrete random results
- In this lesson we deal only with the continuous case
  - Probability of a cat weighing a certain amount
  - Uniform distribution between a and b (all real values between a and b are equally likely)

Probability density functions (PDFs)

Well, in monte carlo integration we deal mostly with continuous statistics. In this lesson, we'll only use continuous. A real life example would be the weight distribution of a cat, described by the probability density function (PDF). The cats' weights probably follow a normal distribution with certain parameters. Another example would be a uniform distribution between a and b.

Statistical Recap

- If you remember „107.254 Statistik und Wahrscheinlichkeitstheorie“ well enough, you can skip most of it.
- In daily life, we are mostly confronted with discrete random results
- In this lesson we deal only with the continuous case
  - Probability of a cat weighing a certain amount
  - Uniform distribution between a and b (all real values between a and b are equally likely)
  - Probability must integrate to 1

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1$$

And similar to the discrete case, the pdf, p integrates to 1.

Statistical Recap

Notation:

$X \ldots$ stochastic variable

$x \ldots$ observation of X or a normal variable (like in plots)

$f(x) \ldots$ function of x

$p(x) \ldots$ probability of x $\left(\int_D p(x)\,\mathrm{d}x = 1, p(x) \geq 0 \right)$

$\hat{I} \ldots$ estimators are denoted by a hat

We'll also have to introduce some notation:
Capital X denote stochastic variables
Small xes are either observations (often with an index variable) or normal variables (inside integrals).
Then we have functions f(x). These functions can either eat stochastic variables (producing new stochastic variables) or just normal variables (observations or inside integrals).
PDFs are also just functions, but with some special conditions (integrate to 1 and they are positive).
Finally we have estimators, which we denote with a hat.

Expectation of a stochastic variable

- Our variable is called X, but remember that a function of one or more stochastic variables is also a stochastic variable
- X is distributed according to p, a function of x

$$\mathrm{E}[X]_{X\sim p} = \int xp(x)\,\mathrm{d}x$$

Wikipedia

And then we have the expectation of a stochastic variable. This is also called expected value, or simply the mean, or the average.
The expectation itself is not a stochastic variable, it's a value.
Given a stochastic variable X, which is distributed according to the PDF p, the expectation, denoted by capital E is the integral of x times p(x). I took that definition from wikipedia :)

Statistical Recap

Important properties of the expectation $\mathrm{E}[X]_{X\sim p} = \int xp(x)\,\mathrm{d}x$

- Linearity

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

Wikipedia

- Law of the unconscious statistician

$$\mathrm{E}[f(X)]_{X\sim p} = \int f(x)p(x)\,\mathrm{d}x$$

Wikipedia

- Law of large numbers (almost surely)

$$\mathrm{E}[X] = \lim_{N\to\infty}\left(\frac{1}{N}\sum_{i=0}^{N} x_i\right)$$

x_i independent and identically distributed (i.i.d.)

Wikipedia

The expecation has some properties and laws that we will use in our proof.

It is linear, which means that the expectation of a linear combination of stochastic variables, is a linear combination of expectations of X. For example expecation of a times X plus b times X is a times expecation + b times expecation. A and b are constants here.

Then we have the law of the unconscious statistician. The expectation of a function of x is the integral of f(x) times p(x).

Finally we need the law of large numbers. It states that the average of observations converges almost surely towards the expectation, when increasing the number of samples and all X are independent and identicallty distributed (in short iid). Iid simply means that we don't do anything funny. This law is independent of the PDF of X. The pdf is implicitly used when taking observations of X.

Monte Carlo Integration

$$I \approx \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$

Probability of a random uniform value between a and b

$$p_{\text{unif}}(x) = \frac{1}{b-a}, \text{ based on } \int_a^b p_{\text{unif}}(x)dx = 1$$

Alrighty, mighty cat. Let's look back at our problem from before.
There is our estimator, 1 over N times the sum of (b-a) times f(x).
We saw the uniform distribution already once. Its pdf is 1 over b-a.

Monte Carlo Integration

$$I \approx \frac{1}{N} \sum_{i=0}^{N} (b-a) f(x_i)$$

$$p_{\text{unif}}(x) = \frac{1}{b-a}$$

$$\frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p_{\text{unif}}(x_i)}$$

x_i are samples from $X \sim p_{\text{unif}}$

We don't know whether that will approximate the integral yet!

We can plug those two friends together. 1 over p replaces the factor (b-a). We removed I here, because we want to prove that it is a good estimator, we don't know it yet. And our x_is are observations, or samples, of a uniform distribution now.

Monte Carlo Integration

Law of large numbers

$$\frac{1}{N}\sum_{i=0}^{N}\frac{f(x_i)}{p_{\text{unif}}(x_i)}$$

x_i are samples from $X \sim p_{\text{unif}}$

$$\mathrm{E}[X] = \lim_{N\to\infty}\left(\frac{1}{N}\sum_{i=0}^{N} x_i\right)$$

x_i independent and identically distributed (i.i.d.)

$$\lim_{N\to\infty}\left(\frac{1}{N}\sum_{i=0}^{N}\frac{f(x_i)}{p_{\text{unif}}(x_i)}\right) = \mathrm{E}\left[\frac{f(X)}{p_{\text{unif}}(X)}\right]_{X\sim p_{\text{unif}}}$$

$$x_i \text{ are samples from } X \sim p_{\text{unif}} \Rightarrow \frac{f(X)}{p_{\text{unif}}(X)} \text{ i.i.d.}$$

Next, we use the law of large numbers. The limes of the average is the expectation. We said that the samples of x are taken from a uniform distribution, so we note that down.
However, this is not the distribution that was used to apply the law of large numbers. f(x) divided by p(x) is not distributed uniformly. But it is enough that the samples f over p are independent and identically distributed.

Monte Carlo Integration

Law of the unconscious statistician

$$\lim_{N\to\infty}\left(\frac{1}{N}\sum_{i=0}^{N}\frac{f(x_i)}{p_{\text{unif}}(x_i)}\right) = \mathrm{E}\left[\frac{f(X)}{p_{\text{unif}}(X)}\right]_{X\sim p_{\text{unif}}}$$

$$\mathrm{E}[f(X)]_p = \int f(x)p(x)\,\mathrm{d}x$$

$$\mathrm{E}\left[\frac{f(X)}{p_{\text{unif}}(X)}\right]_{X\sim p_{\text{unif}}} = \int \frac{f(x)}{p_{\text{unif}}(x)} p_{\text{unif}}(x)\,\mathrm{d}x$$

Now we have that expectation, and we can plug it into the law of the unconscious statistician.
The f from the law is f over p, and the distribution is uniform.
The expectation is the integral of the function times the pdf.
So, our result is an integral of f over p times p.

Monte Carlo Integration

$$\mathrm{E}\left[\frac{f(X)}{p_{\mathrm{unif}}(X)}\right]_{X\sim p_{\mathrm{unif}}} = \int \frac{f(x)}{p_{\mathrm{unif}}(x)} p_{\mathrm{unif}}(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

And well, the probabiilties cancel out, and we see that this is the integral I, that we wanted to compute.

Monte Carlo Integration

$$\mathrm{E}\left[\frac{f(X)}{p_{\mathrm{unif}}(X)}\right]_{X\sim p_{\mathrm{unif}}} = \int \frac{f(x)}{p_{\mathrm{unif}}(x)} p_{\mathrm{unif}}(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

Great, the cat is happy. This is our proof.

Monte Carlo Integration

$$\mathrm{E}\left[\frac{f(X)}{p_{\mathrm{unif}}(X)}\right]_{X\sim p_{\mathrm{unif}}} = \int \frac{f(x)}{p_{\mathrm{unif}}(x)} p_{\mathrm{unif}}(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

And here we also see, that we can replace the uniform distribution by any other distribution. We'll see shortly how that can benefit us.

Monte Carlo Integration

$$\mathrm{E}\left[\frac{f(X)}{p(X)}\right]_{X\sim p} = \int \frac{f(x)}{p(x)} p(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

This is actually quite exciting, the cat is super happy ;)

Arr, wait..

Monte Carlo Integration

$$\mathrm{E}\left[\frac{f(X)}{p(X)}\right]_{X\sim p} = \int \frac{f(x)}{p(x)} p(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

Wait, what if p(x) becomes 0?
Division by 0 is undefined, ey?

What if p(x) becomes 0?
Division by 0 is undefined, ey?

Monte Carlo Integration

Wait, what if p(x) becomes 0?
Division by 0 is undefined, ey?

$$\mathrm{E}\left[\frac{f(X)}{p(X)}\right]_{X\sim p} = \int \frac{f(x)}{p(x)} p(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

Good catch!
The result of Monte Carlo integration becomes wrong if p(x) is 0 but f(x) is not.

That's a good catch. And the result of Monte Carlo integration becomes actually wrong when p is 0 but f not!

Monte Carlo Integration

Wait, what if p(x) becomes 0?
Division by 0 is undefined, ey?

$$\mathrm{E}\left[\frac{f(X)}{p(X)}\right]_{X\sim p} = \int \frac{f(x)}{p(x)} p(x)\,\mathrm{d}x$$
$$= \int f(x)\,\mathrm{d}x = I$$

Good catch!
The result of Monte Carlo integration becomes wrong if p(x) is 0 but f(x) is not. However, our computer program will not produce NaNs, as no x_i will have $p(x_i) = 0$!

However, out computer program will not produce NaNs, as no x_i will occur, because such xes have probability of 0 and will not be sampled..

Still we have to fix that..

Monte Carlo Integration

Wait, what if p(x) becomes 0?
Division by 0 is undefined, ey?

$$p(x) > 0 \quad \forall x \in D$$

$$\int_D p(x)\,\mathrm{d}x = 1$$

$$\mathrm{E}\left[\frac{f(X)}{p(X)}\right]_{X\sim p} = \int_D \frac{f(x)}{p(x)} p(x)\,\mathrm{d}x$$

$$= \int_D f(x)\,\mathrm{d}x = I$$

Good catch!
The result of Monte Carlo integration becomes wrong if p(x) is 0 but f(x) is not. However, our computer program will not produce NaNs, as no x_i will have $p(x_i) = 0$!

And we can do that by requiring that the probability is greater than zero inside the integration domain.

In practise, it is enough that p(x) is greater than 0 whenever f(x) is not zero.

Monte Carlo Integration

$$p(x) > 0 \quad \forall x \in D$$

$$\int_D p(x)\,\mathrm{d}x = 1$$

$$\mathrm{I} = \mathrm{E}\left[\frac{f(X)}{p(X)}\right]_{X\sim p} = \mathrm{E}[\hat{I}]$$

$$\hat{I} = \frac{1}{N}\sum_{i=0}^{N}\frac{f(x_i)}{p(x_i)}$$

And here we have it again. The Monte Carlo estimator with all its glory ;)
Appreciate its beauty!

Alrighty mighty cat, but that's still a bit abstract.
Yes, let's look at an practical example!

Monte Carlo Integration

Example

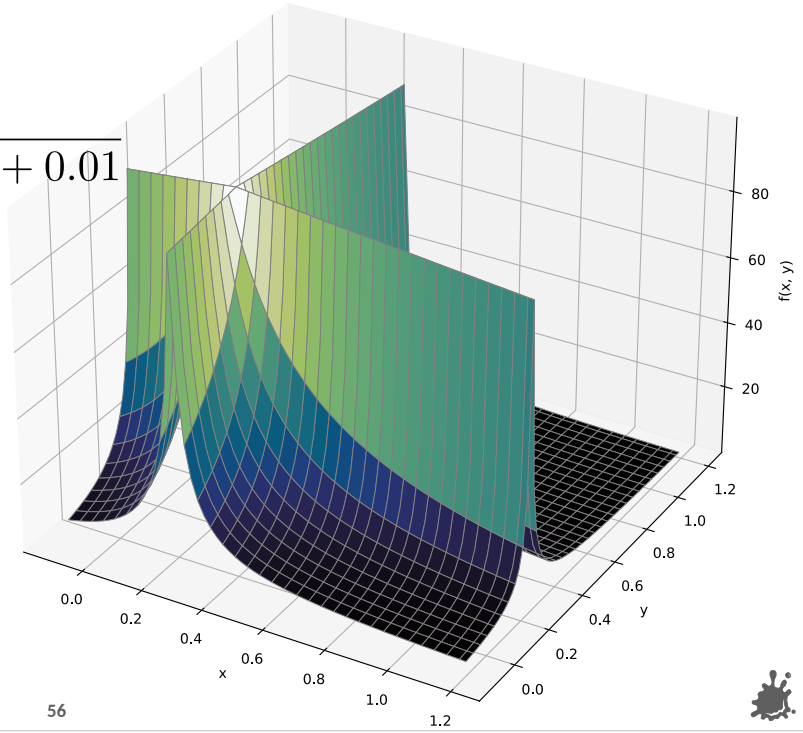
$$f(x,y) = \frac{1}{(x-0.2555)(y-0.2555)+0.01}$$

$$D = \square \text{ with}$$

$$a = (-0.1; -0.1),$$

$$b = (1.2; 1.2)$$

$$I = \int_D f(x,y)\,\mathrm{d}x\,\mathrm{d}y$$

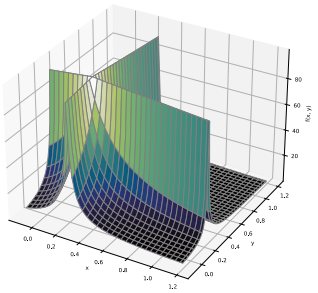


Here we go.
We are looking at a 2d function, the equation is on the top left, you see a plot of that function on the right.
The integration area is a rectangle between (-0.1/-0.1) and (1.2/1.2), just so its dimensions are not 1 by 1.
And we want to integrate it..

Monte Carlo Integration

Example

$$I = \int_D \frac{1}{(x - 0.2555)(y - 0.2555) + 0.01} \, \mathrm{d}x \, \mathrm{d}y$$



```python
def f(x, y):
    return 1/(np.abs((x - 0.2555) * (y - 0.2555)) + 0.01)


def integrate_mc(a: float, b: float, N: int, ref: float = None):
    X = np.random.rand(N) * (b - a) + a
    Y = np.random.rand(N) * (b - a) + a
    p = 1 / (b - a) ** 2
    samples = f(X, Y) / p
    estimate = samples.sum() / N
    return estimate
```


Here you see an implementation using python.

We create N uniformly distributed samples between a and b with probability p.

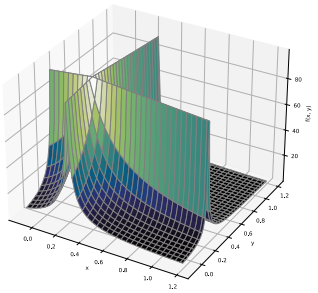
Our samples are then computed with f over p. And the average is the estimate.

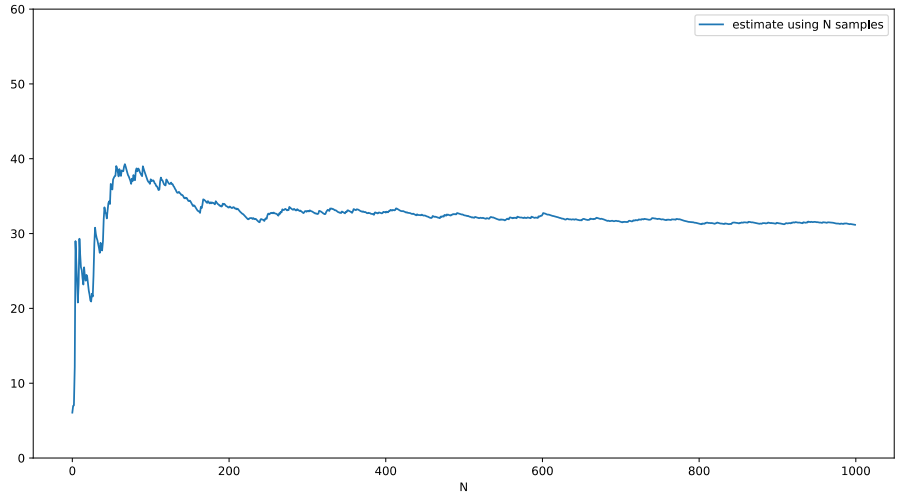
That’s all.

Monte Carlo Integration

Example

$$I = \int_D \frac{1}{(x - 0.2555)(y - 0.2555) + 0.01} \, \mathrm{d}x \, \mathrm{d}y$$





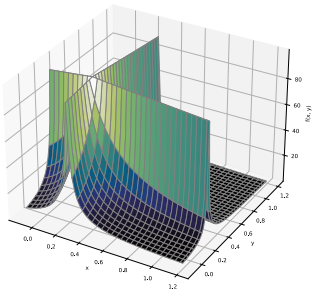
We can now look at how the estimate behave when increasing N.

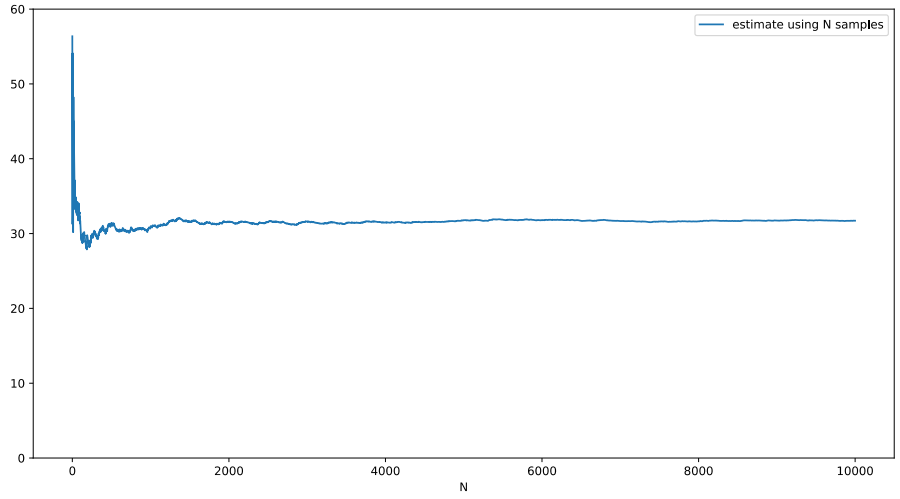
With small N, the estimate is imprecise. But when we increase it, the estimate becomes gradually better and better.

Monte Carlo Integration

Example

$$I = \int_D \frac{1}{(x - 0.2555)(y - 0.2555) + 0.01} \, \mathrm{d}x \, \mathrm{d}y$$





And it pretty much stops to wiggle around when we further increase N.

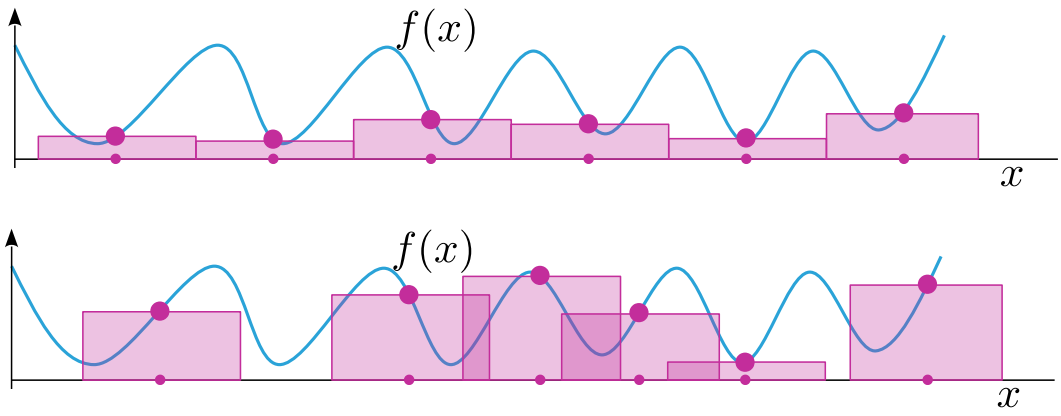
Btw, here you also see, that different runs will produce different error curves.

Let’s try to get more insight into the error

Monte Carlo Integration

Two types of error for estimators

- Bias
- Variance



Generally, there are two types of errors for estimators (not only Monte Carlo estimators).
We have bias and variance.
The rectangle rule estimator has bias while the monte carlo estimator has variance. The wiggeling in the previous graphs is the variance of the monte carlo estimator (it sometimes produces a larger and sometimes a smaller value).

Monte Carlo Integration

Two types of error for estimators

- Biased estimators
  $\mathrm{E}[\hat{I}] \neq I$
- Unbiased estimators

  $\mathrm{E}[\hat{I}] = I$

  $\mathrm{E}[\hat{I}] = \mathrm{E}\left[\frac{\hat{I} + \hat{I}}{2}\right] = I$
- It is quite common to trade reduced variance for increased bias

The expectation of biased estimators is not the value we want to estimate, here the integral. No matter how large N is. This does not mean, that such biased estimators are worse than unbiased ones. The bias can be much smaller than the variance.

Unbiased estimators on the other hand have a correct expectation. And due to the linearity of expectation, we can actually compute the average of two estimates and again get an unbiased estimate (that will be more precise than the separate estimates).

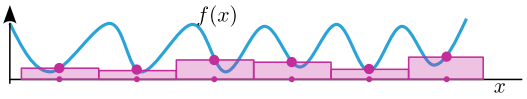
In practice it's quite common to trade variance for increased bias. But we won't go there now..

Let's look at an example instead

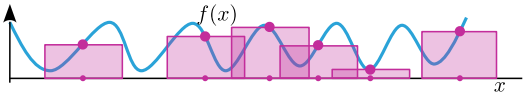
Monte Carlo Integration

Two types of error for estimators

- Bias
- Variance




```
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using MC, N = 100, estimate=26.485585056320133, error=-5.596063730066859
estimate using MC, N = 100, estimate=24.70966348202858, error=-7.371985304358411
estimate using MC, N = 100, estimate=38.99700963066361, error=6.91536084427662
estimate using MC, N = 100, estimate=33.33897223513907, error=1.257323448752075
```



We have the rectangle rule estimator on top. And you see, that no matter how often we run it, the result is always the same. Clearly the variance is 0, and the bias is -2.301..

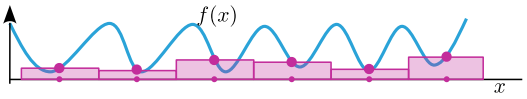
The monte carlo estimator on the other hand produces a different result every time. Variance is not 0, we could estimate it by running the estimator many times and using the quations for variance. The bias is 0 as we have proved before.

We said that we can improve the estimate by averaging several runs. Lets do that..

Monte Carlo Integration

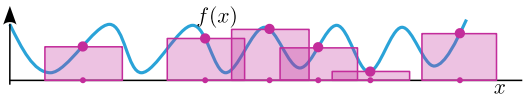
Two types of error for estimators

- Bias
- Variance




```
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using rect rule, N*M = 169, estimate=29.78041551309039, error=-2.301233273296603
estimate using MC, N = 100, estimate=26.485585056320133, error=-5.596063730066859
estimate using MC, N = 100, estimate=24.70966348202858, error=-7.371985304358411
estimate using MC, N = 100, estimate=38.99700963066361, error=6.91536084427662
estimate using MC, N = 100, estimate=33.33897223513907, error=1.257323448752075
```

Σ 123.54/4 = 30.885
Error: -1.19



Here we go, sum and divide by 4. the result 30.885 is more precise than the individual estimates with an error of approximately -1.19

Let’s look at variance in more detail. There are some important concepts. And then, we’ll look at importance sampling, a way to reduce variance.

Variance and Importance Sampling

What is the variance of that estimator?

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = ?$$

Ok, we see the Monte Carlo estimator on top. We saw that we can run it several times, and we could estimate the variance empirically..

But we're at an university and we should try to do things more principled (and even if you're not at a university that's still a good idea). We'll get to that in a second, but we need another short statistics recap before..

Statistical Recap

- Variance

$$\operatorname{Var}(X) = \operatorname{E}\left[(X - \operatorname{E}[X])^2\right] = \operatorname{E}[X^2] - \operatorname{E}[X]^2$$

Wikipedia

- Variance of a linear combination of **uncorrelated** X

$$\operatorname{Var}\left(\sum^{N} aX\right) = a^2 \sum^{N} \operatorname{Var}(X)$$

Wikipedia

Variance is the expectation of the quadratic deviation between the stochastic variable and its expectation. Wikipedia also shows us a way to reformulate it, but that's not so important for now.

Unlike expectation, the variance is not linear, but we can still say something about the variance of a linear combination of stochastic variables. Look at the formula, we can square the constant a and then pull it out. The variance of a sum of stochastic variables is the sum of the variances.

That's all we need, time to plug things together again..

Variance and Importance Sampling

- Estimator

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

- Linear combination

$$\mathrm{Var}\left(\sum^{N} aX \right) = a^2 \sum^{N} \mathrm{Var}(X)$$

- Its variance

$$\mathrm{Var}(\hat{I}) = \mathrm{Var}\left(\frac{1}{N} \sum^{N} \frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right) = \frac{1}{N^2} \sum^{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

That linear combination is quite handy when we look at the estimator.
Variance of the estimator, we expand the estimator, square the constant and pull it out, pull out the sum..

Variance and Importance Sampling

- Estimator

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

- Linear combination

$$\mathrm{Var}\left(\sum^{N} aX \right) = a^2 \sum^{N} \mathrm{Var}(X)$$

- Its variance

$$\mathrm{Var}(\hat{I}) = \mathrm{Var}\left(\frac{1}{N} \sum^{N} \frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right) = \frac{1}{N^2} \sum^{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

$$= \frac{1}{N^2} N \, \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

The variance of f over p is a constant, and the sum of N constants is N times the constant.

Variance and Importance Sampling

- Estimator

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

- Linear combination

$$\mathrm{Var}\left(\sum^{N} aX \right) = a^2 \sum^{N} \mathrm{Var}(X)$$

- Its variance

$$\mathrm{Var}(\hat{I}) = \mathrm{Var}\left(\frac{1}{N} \sum^{N} \frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right) = \frac{1}{N^2} \sum^{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

$$= \frac{1}{N^2} N \, \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

Variance of the estimator is inversely proportional to N

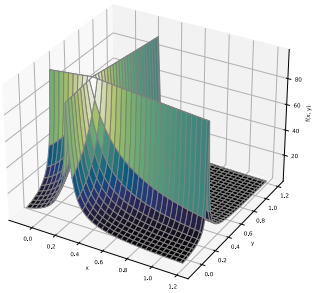
And the N cancel out. What remains is that the variance of the estimator I is one over N times the variance of a single sample.

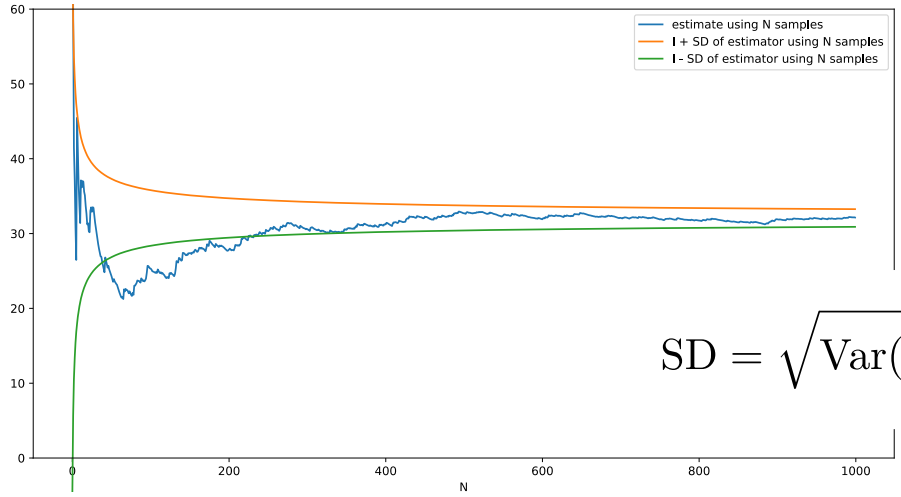
This is exactly what we saw in the plots of monte carlo error before..

Monte Carlo Integration

Example

$$I = \int_D \frac{1}{(x - 0.2555)(y - 0.2555) + 0.01} \, \mathrm{d}x \, \mathrm{d}y$$





$$\mathrm{SD} = \sqrt{\mathrm{Var}(\hat{I})} = \sqrt{\frac{\mathrm{Var}(f(x)/p(x))}{N}}$$

Here I ran the experiment again.

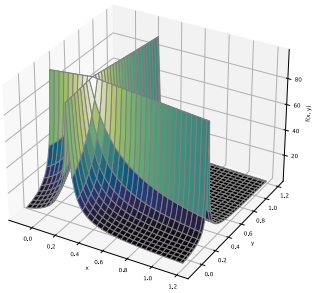
In addition to the estimate I plotted the standard deviation (which is linear scale and therefore easier to understand than variance).

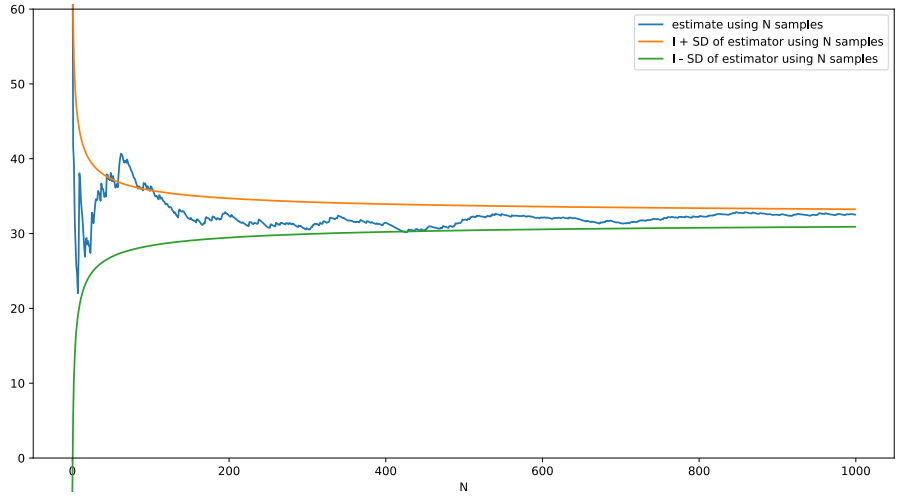
What's cool here is, that we can use all samples to estimate the variance, and then we can just scale with N.

Monte Carlo Integration

Example

$$I = \int_D \frac{1}{(x - 0.2555)(y - 0.2555) + 0.01} \, \mathrm{d}x \, \mathrm{d}y$$





Rerunning the example gives us a different plot of the estimate, but in both cases it more or less stays within the standard deviation.

Variance and Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

Here we have the result again.

What this tells us, is that we can
1. increase the number of samples and becom more and more precise
2. we can estimate the error that we are making.

Think the following: we could continue sampling until variance is below a certain threshold.

In practice there are some pitfalls, but that approach is used. Maybe something you would want to implement as a bonus task?

Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

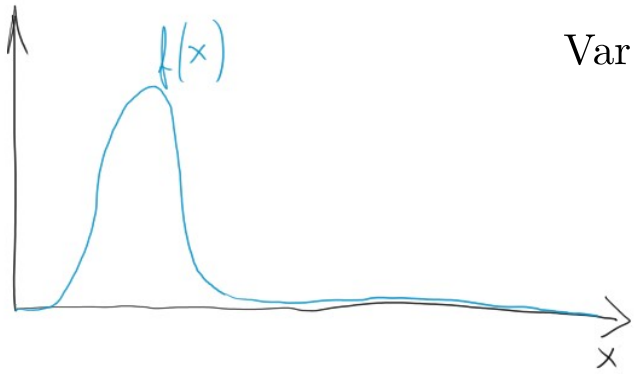
$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)}\right)$$

In the meanwhile, we have that variance. And we want to reduce it, obviously. But first, we have to understand it a bit better..

Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$



Let's look at this example of a function f(x), that we want to integrate using Monte Carlo.

We said that we can choose the PDF for our samples x freely. Different choices make life more difficult, well, well..

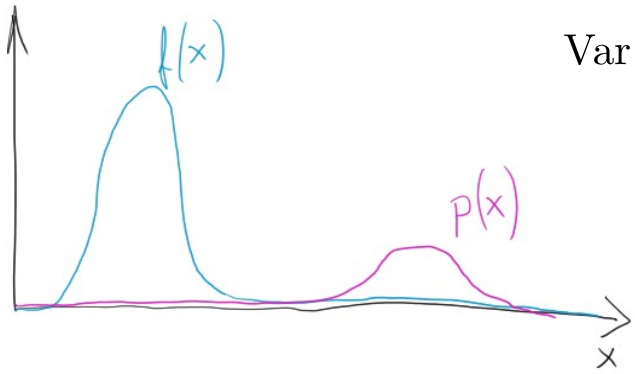
Let's look at ..

Variance and Importance Sampling

Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)}\right)$$



Such a p.

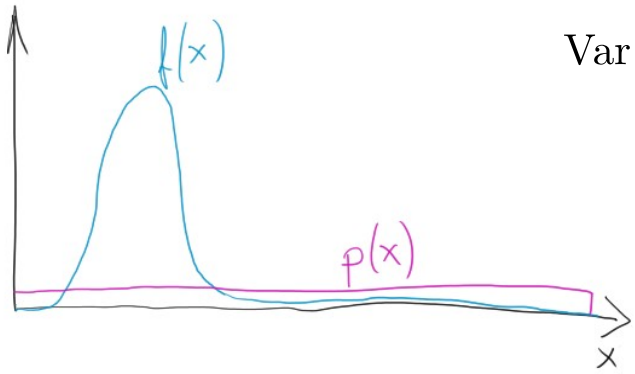
Variance of f over p, right. Here, we have a large f, but only a small p. F over p will be large. Here, on the other hand, f is small and p large. F over p will be small. That means that the variance of f over p is large..

We can do better. Even the uniform distribution would be better.. let's take a look

Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

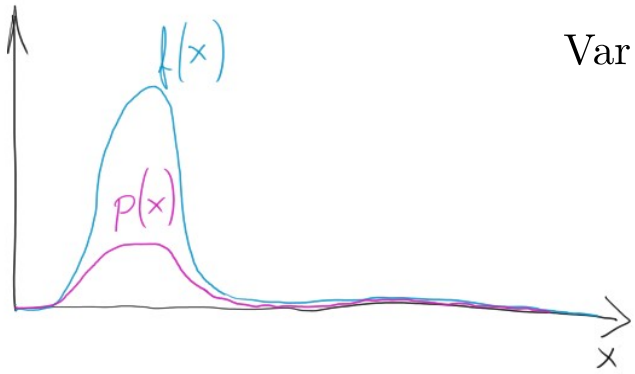


Here, we now have a larger p, the f over p will be smaller. Here p is still larger, but the difference is smaller. Overall, the variance will be smaller, but you we could find a better p..

Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$



Look at this.

P is still smaller, but it must be if f integrates to something larger than p.

This p looks quite good really.

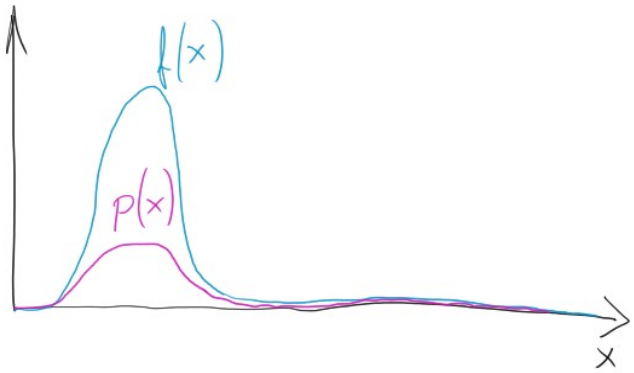
And that is, because we defined p ..

Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)}\right)$$

$$\mathrm{p}(x) = s\,\mathrm{f}(x)$$



As a f scaled with a constant s. P needs to integrate to one, so we can compute s

Variance and Importance Sampling

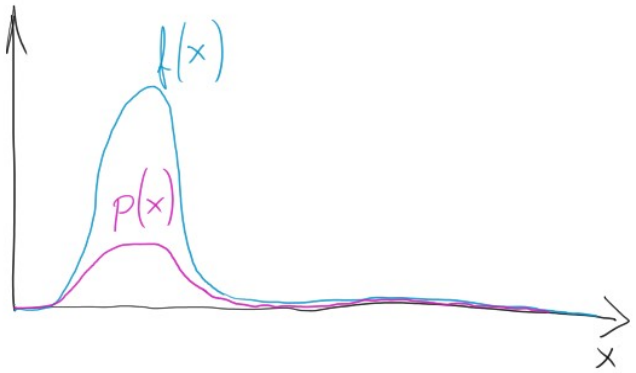
Importance Sampling

$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)}\right)$$

$$s = \frac{1}{\int \mathrm{f}(x)\,\mathrm{d}x}$$

$$\mathrm{p}(x) = s\,\mathrm{f}(x)$$



As one over the integral of f.

Let’s look at the variance of an estimator with such a p..

Importance Sampling

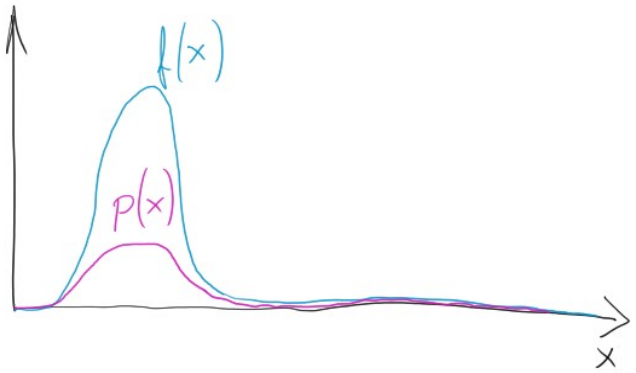
$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

$$s = \frac{1}{\int \mathrm{f}(x)\,\mathrm{d}x}$$

$$\mathrm{p}(x) = s\,\mathrm{f}(x)$$

$$\mathrm{Var}\left(\frac{\mathrm{f}(X)}{s\,\mathrm{f}(X)} \right) = \mathrm{Var}\left(\frac{1}{s} \right) = 0$$



We simply plug p into the variance equation and, after chanceling out f over f, arrive at the variance of 1 over s. S is a constant, so we have a variance of 0.

Variance and Importance Sampling

Importance Sampling

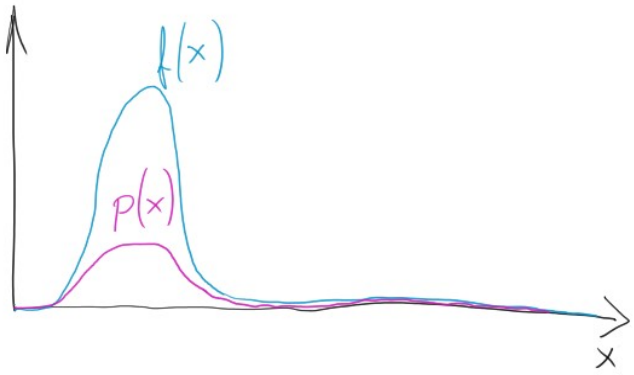
$$\hat{I} = \frac{1}{N} \sum_{i=0}^{N} \frac{f(x_i)}{p(x_i)}$$

$$\mathrm{Var}(\hat{I}) = \frac{1}{N} \mathrm{Var}\left(\frac{\mathrm{f}(X)}{\mathrm{p}(X)} \right)$$

$$s = \frac{1}{\int \mathrm{f}(x)\,\mathrm{d}x}$$

$$\mathrm{p}(x) = s\,\mathrm{f}(x)$$

$$\mathrm{Var}\left(\frac{\mathrm{f}(X)}{s\,\mathrm{f}(X)} \right) = \mathrm{Var}\left(\frac{1}{s} \right) = 0$$



Variance is 0, that's the perfect estimator.. we would need only one sample to estimate I perfectly!

That sounds wrong, doesn't it?

Well, the thing is, that in practise it's not possible to get such a probability density function for non trivial problems. Start with the fact, that computing s would involve the integral of f, which is the problem that we started with.

There is an ingenious algorithm that goes into that direction, it's called metropolis light transport and is based on correlated chains of samples. However, that algorithm has some other problems and it is too complicated for now..

For now we will try to use PDFs that are as close to f as possible. This method is called importance sampling, because we try to sample important parts of f. We'll see more about that in one of the upcoming lectures.

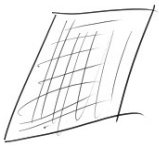
Before that, let's try to understand how to apply Monte Carlo in rendering. In the last lecture we had those hemispheres..

Cartesian 1d / 2d okey, but what about that hemisphere stuff?

And it’s not immediately clear how the 1d / 2d stuff that we saw now maps to the hemisphere. Functions in the hemisphere, that might be hard to imagine, it’s a bit weird, no?

What about the Hemisphere?

Cartesian 1d / 2d okey, but what about that hemisphere stuff?



surface

hemisphere

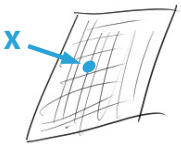
Both are 2d manifolds in 3d space!

Well, a 2d cartesian coordinate plane can be seen as a surface in 3d. We call such a surface a 2d manifold in 3d space. Locally you can move in 2 independent directions.

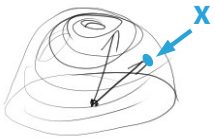
You can do the same on a hemisphere, locally you can move in to independent directions. It's just another manifold in space.

What about the Hemisphere?

Cartesian 1d / 2d okey, but what about that hemisphere stuff?



surface



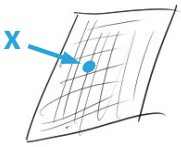
hemisphere

Both are 2d manifolds in 3d space!
x is a point on that manifold.
f(**x**) assigns a value to that point.
p(**x**) is also just a function.

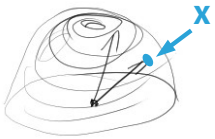
Let's say we have a point x on that surface. We can have a function f, that assigns a certain value to the point. The pdf is also just a function, so that also works..

What about the Hemisphere?

Cartesian 1d / 2d okey, but what about that hemisphere stuff?



surface



hemisphere

There can be several equivalent ways to describe **x**

u/v coordinates
Global x/y/z
..

θ/φ angles
x/y/z local direction with $x^2 + y^2 + z^2 = 1$
Origin and destination global positions
..

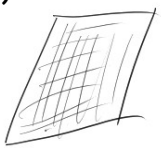
And we can have different systems to describe the point, or its coordinates.
For a flat surface, we could have UV coordinates, or global XYZ coordinates or other options..
For the hemisphere, we could have two angles, theta and phi, or we could have a local vector x/y/z with a length of one (so that we stay on the manifold, or we could have the origin of the hemisphere and another point in space and define the point by intersection (vector of length 1, pretty similar to the second description method actually).

This is just to paint a picture. In practise we will use theta and phi..

Now for choices of the pdf..

What about the Hemisphere?

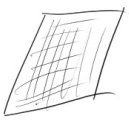
Cartesian 1d / 2d okey, but what about that hemisphere stuff?



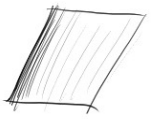
surface

hemisphere

Options for p(**x**)



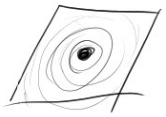
uniform



linear

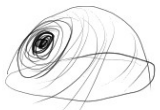
uniform

Surface mapped to hemisphere



gaussian

textured



BRDF

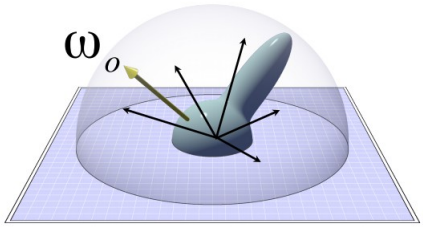
cosine

For the surface, uniform distribution is often a good choice. But we could also have linear, or a gaussian, or even a pdf based on a texture. The choice of importance sampling strategy depends on the function that we want to integrate.

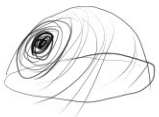
For the hemisphere, we could have a uniform distribution, we could map a surface to the hemisphere and use that, we could use importance sampling of the BRDF (the material, we'll see an example on the next slide), or the cosine (which can be used for diffuse materials to account for the cosine rule).

Importance Sampling

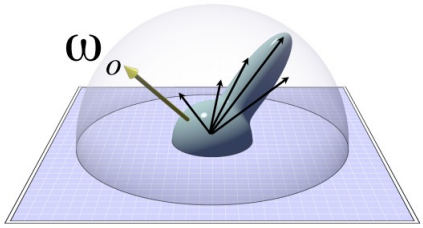
uniform

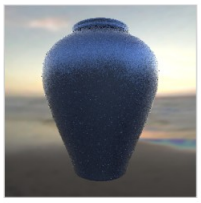


5 Samples/Pixel



BRDF



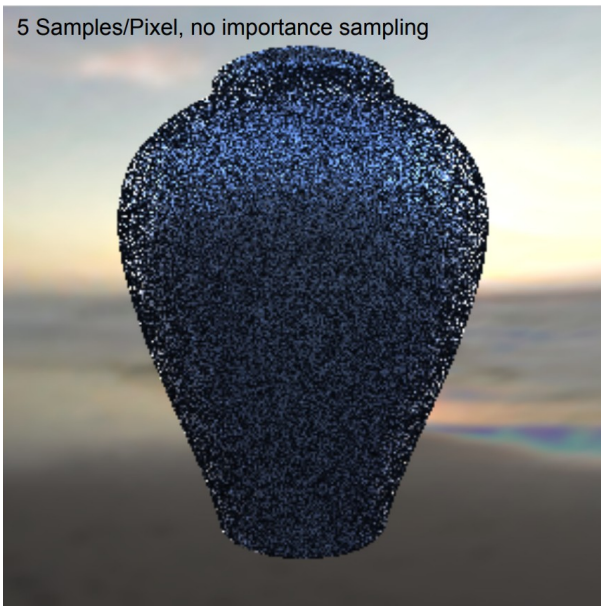


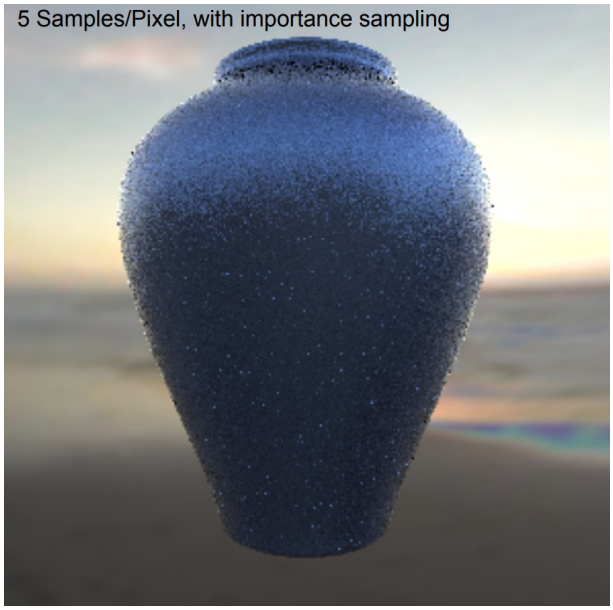
Slide modified from Jaakko Lehtinen, with permission

Here we see different sampling strategies for a BRDF, on top uniform sampling and on the bottom importance sampling. The BRDF function along with few samples is visualised in the middle and the result of rendering on the right.

We we look at the results

Importance Sampling



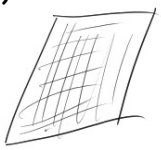


Slide modified from Jaakko Lehtinen, with permission

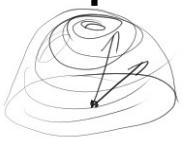
We clearly see, that importance sampling gives us less noise in the result. That's what we saw before. The PDF is closer to the function that we want to integrate, and therefore we have less noise.

What about the Hemisphere?

Cartesian 1d / 2d okey, but what about that hemisphere stuff?

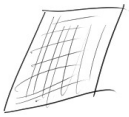


surface

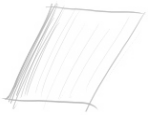


hemisphere

Options for p(**x**)



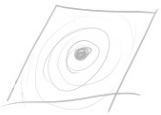
uniform



linear

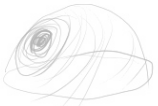
uniform

Surface mapped to hemisphere

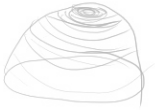


gaussian

textured



BRDF

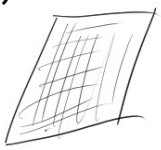


cosine

Right now we will not look further into these importance sampling techniques. We have a separate lecture for that, stay tuned :)

What about the Hemisphere?

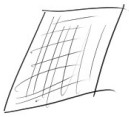
Cartesian 1d / 2d okey, but what about that hemisphere stuff?



surface

hemisphere

Options for p(**x**)



uniform

Easy for rectangles, it's just a 2d integral

uniform

Used in first assignment

Surface mapped to hemisphere

Implicit, when performing change of variables. See first lecture!

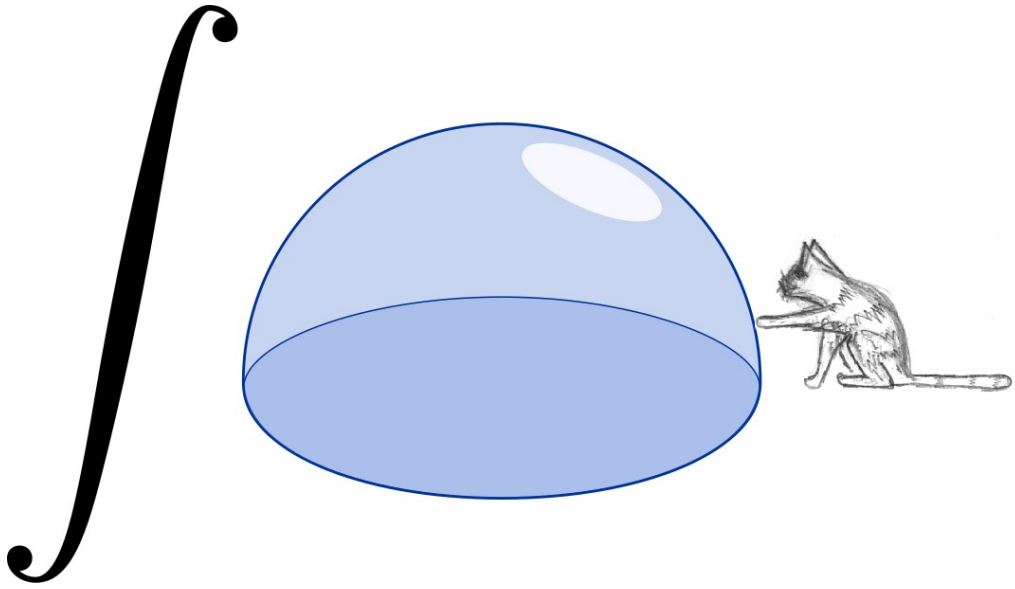
Uniform sampling of a rectangle is pretty easy, we did it in an example before already.

We will try to understand sampling of the hemisphere next.

And surface mapping kinda happens implicitly, when performing a change of variables.

What about the Hemisphere?

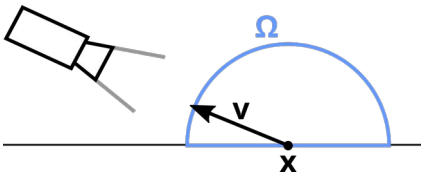
Understand the hemisphere..



Ok, finally, let’s try to understand that hemisphere..

What about the Hemisphere?

Remember the equation for reflected light?



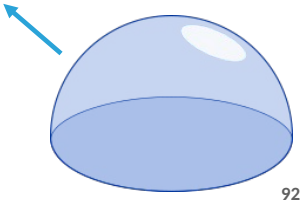
Material, modelled by the BRDF

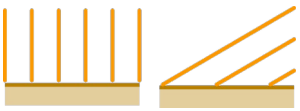
Light from direction ω

Solid angle

$$L_e(x, v) = \int_\Omega f_r(x, \omega \rightarrow v) L_i(x, \omega) \cos(\theta_x) \, \mathrm{d}\omega$$

Light going in direction v





Adam Celarek

92

We have seen that equation before.
The result is the light going to the camera, we are integrating over the hemisphere, and the function consists of the material BRDF times the incoming light times the cosine.

Now we will try to get some intuition on how to perform Monte Carlo integration on it..

What about the Hemisphere?

Remember the equation for reflected light?

This is just a function that eats the position, scene, integration variable (ω) and returns a value!

$$L_e(x, v) = \int_\Omega \boxed{f_r(x, \omega \rightarrow v) L_i(x, \omega) \cos(\theta_x)} \, \mathrm{d}\omega$$

As said, this function just eats certain values, in whatever representation, and returns another value.

What about the Hemisphere?

Remember the equation for reflected light?

$$L_e(x,v) = \int_\Omega f_r(x, \omega \rightarrow v) L_i(x,\omega) \cos(\theta_x) \, \mathrm{d}\omega$$

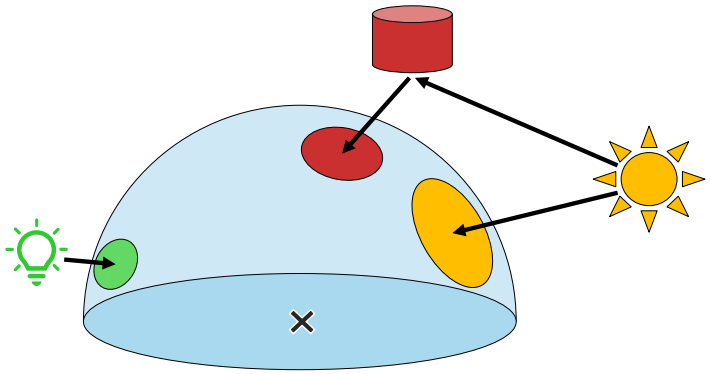
Let's assume white diffuse for now, so this becomes constant 1/π.

We assume a white diffuse material for now, so the brdf becomes 1 over pi.

What about the Hemisphere?

Remember the equation for reflected light?

$$L_e(x, v) = \int_\Omega \frac{1}{\pi} \boxed{L_i(x, \omega)} \cos(\theta_x)\, \mathrm{d}\omega$$

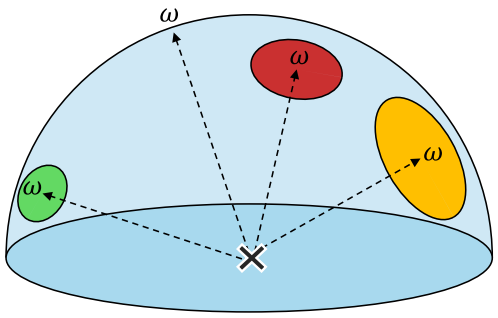


Incoming light is computing by tracing rays. These lights are further away, somewhere in the scene, but for the integral that part doesn't matter.

What about the Hemisphere?

Remember the equation for reflected light?

$$L_e(x, v) = \int_{\Omega} \frac{1}{\pi} L_i(x, \omega) \cos(\theta_x) \, \mathrm{d}\omega$$

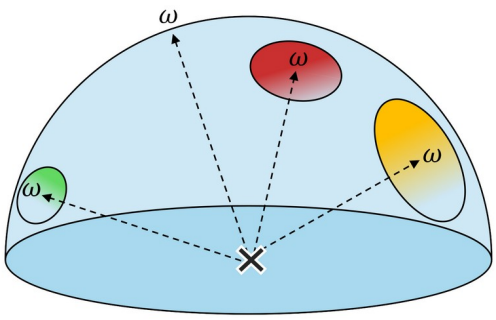


So we can assume that they sit right on the surface..

What about the Hemisphere?

Remember the equation for reflected light?

$$L_e(x, v) = \int_{\Omega} \frac{1}{\pi} L_i(x, \omega) \cos(\theta_x) \, \mathrm{d}\omega$$



And multiplying it with the cosine basically weights the lights depending on theta..

What about the Hemisphere?

Remember the equation for reflected light?

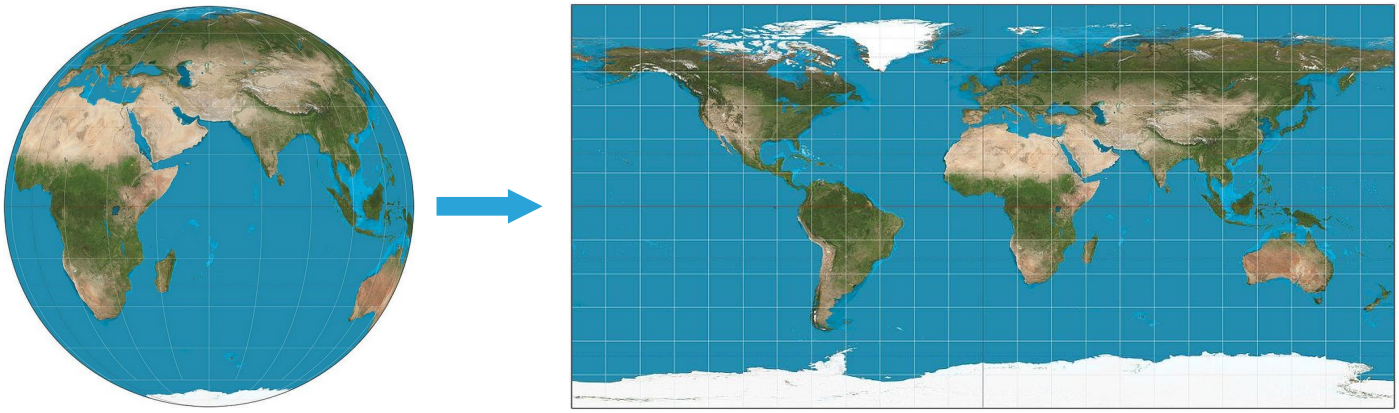
$$L_e(x, v) = \int_{\Omega} \frac{1}{\pi} L_i(x, \omega) \cos(\theta_x) \, \mathrm{d}\omega$$

?

Finally, the hemisphere..

Well, we can use an analogy of

What about the Hemisphere?



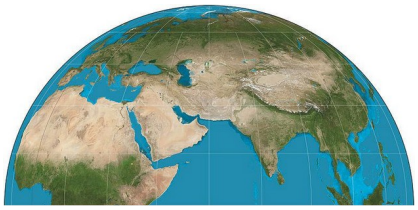
source: User Strebe / Wikipedia
CC BY-SA 3.0

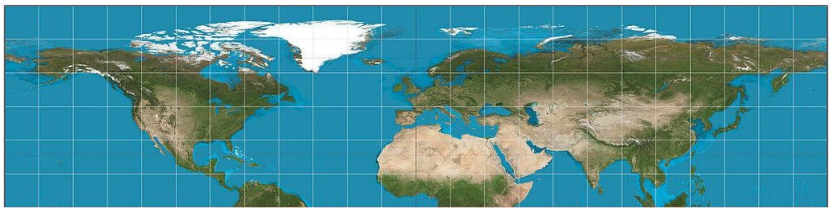
source: User Strebe / Wikipedia
CC BY-SA 3.0

Earth and a 2d map of the continents..

Since we only need a hemisphere and not a sphere, we cut it in half..

What about the Hemisphere?



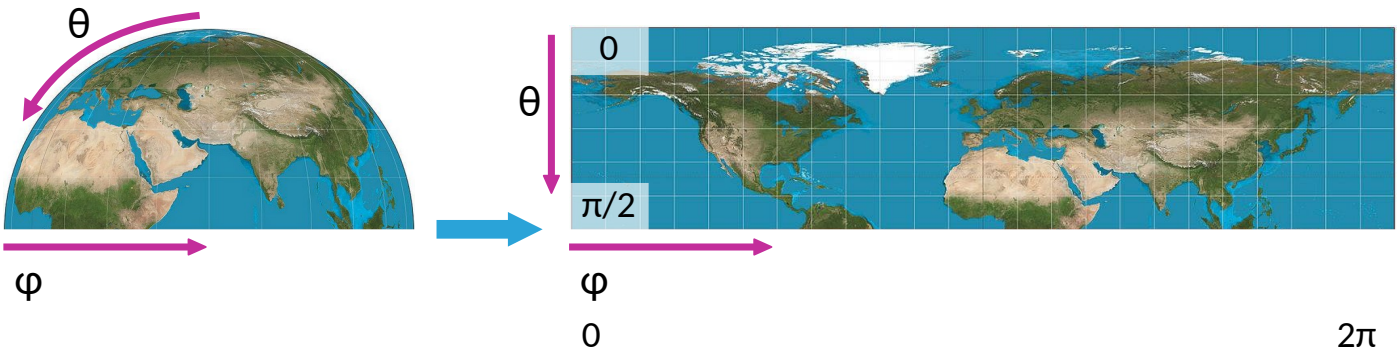


source: User Strebe / Wikipedia
CC BY-SA 3.0

source: User Strebe / Wikipedia
CC BY-SA 3.0

Sorry Australia, there are no kangaroos in Austria ;)

What about the Hemisphere?



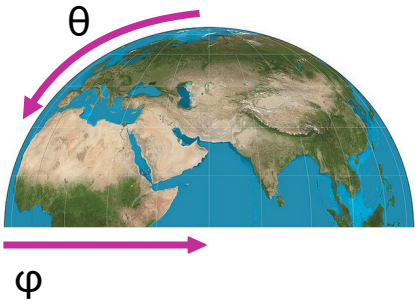
source: User Strebe / Wikipedia
CC BY-SA 3.0

source: User Strebe / Wikipedia
CC BY-SA 3.0

We can describe any point on the upper hemisphere by phi and theta.

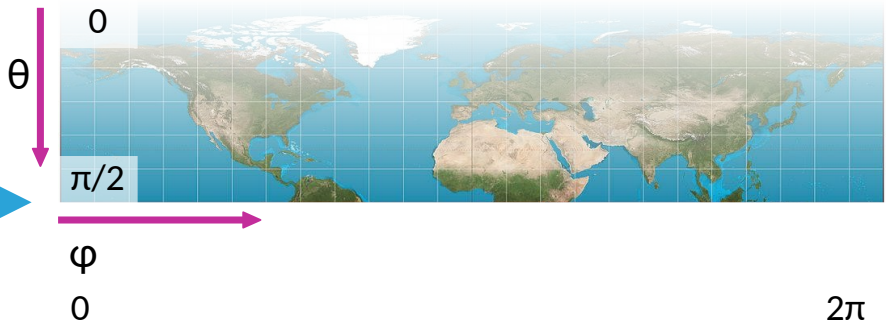
But beware, Greenland and Svalbard are way too big on the flat map. They are overrepresented.

What about the Hemisphere?



source: User Strebe / Wikipedia
CC BY-SA 3.0

Give less weight to the pole => sin(θ)

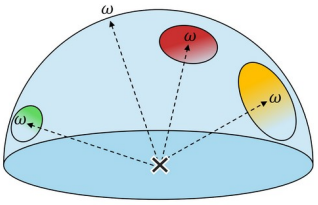


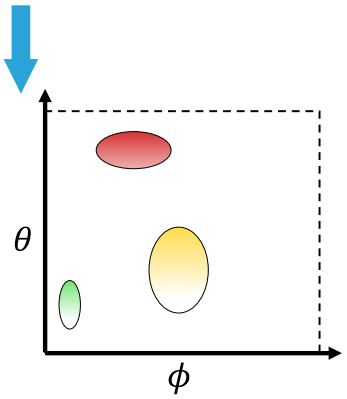
source: User Strebe / Wikipedia
CC BY-SA 3.0

We can multiply with sin of theta to account for that

What about the Hemisphere?

$$L_e(x,v) = \int_\Omega \frac{1}{\pi} \; \mathrm{d}\omega$$





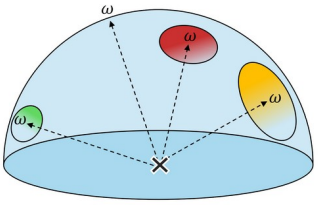
$$L_e(x,v) = \int_0^{\pi/2} \int_0^{2\pi} \frac{1}{\pi} \; \sin(\theta)\, \mathrm{d}\theta\, \mathrm{d}\phi$$

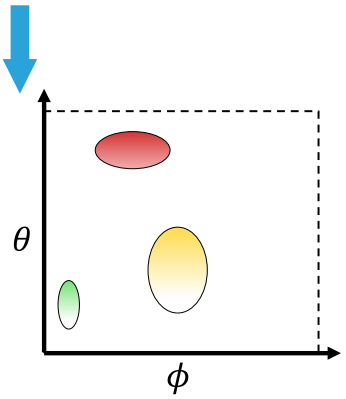
And so we arrive at this mapping:

Instead of integrating over the hemisphere, have 2 integrals now, that use a rectangular domain..

What about the Hemisphere?

$$L_e(x,v) = \int_\Omega \frac{1}{\pi} \; \mathrm{d}\omega$$





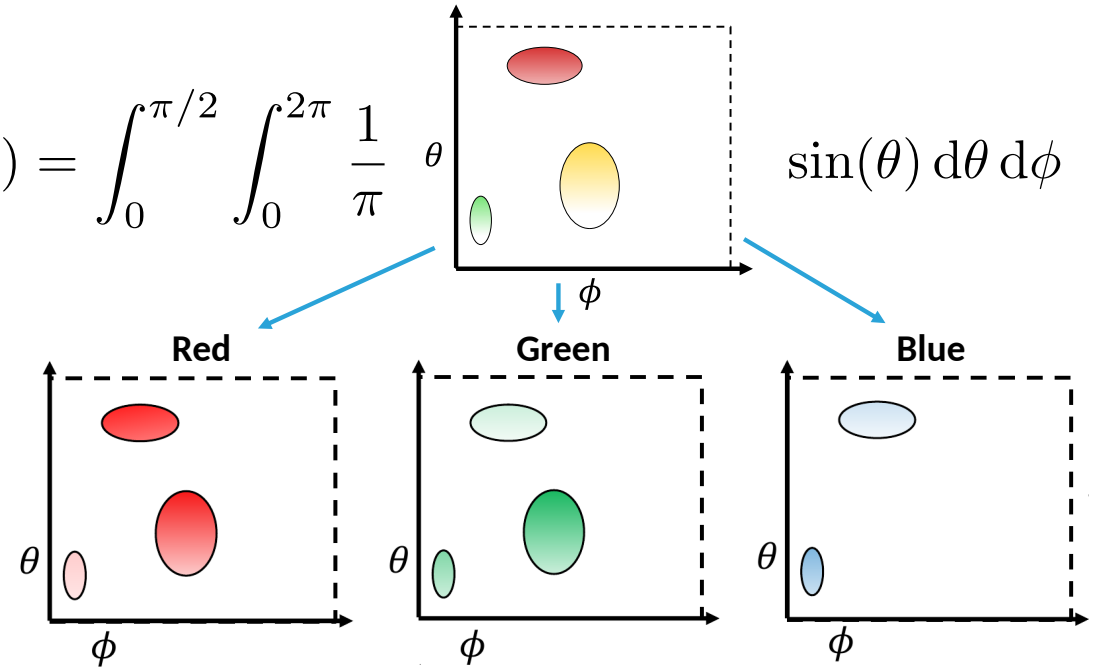
$$L_e(x,v) = \int_0^{\pi/2} \int_0^{2\pi} \frac{1}{\pi} \; \sin(\theta)\, \mathrm{d}\theta\, \mathrm{d}\phi$$

We could directly use uniform sampling here!

And we could use uniform sampling of theta and phi for integration (just like before in the example).

What about the Hemisphere?

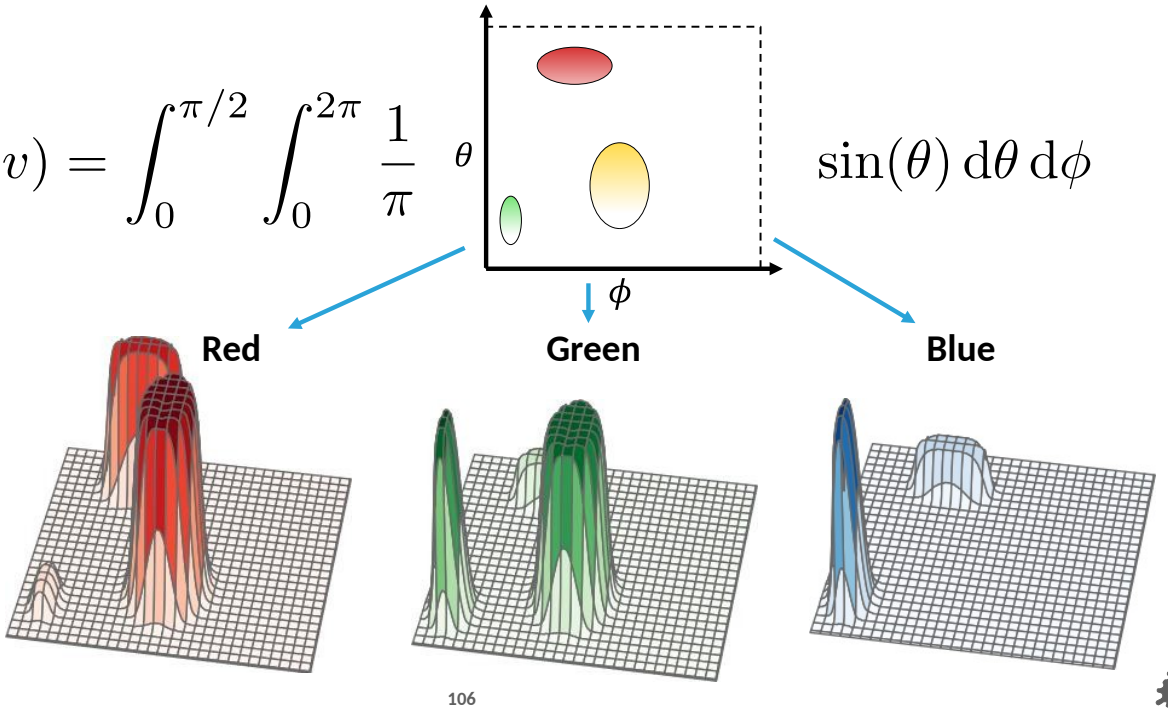
$$L_e(x,v) = \int_0^{\pi/2} \int_0^{2\pi} \frac{1}{\pi} \sin(\theta)\, \mathrm{d}\theta\, \mathrm{d}\phi$$



We compute the integral separately for 3 colour channels..

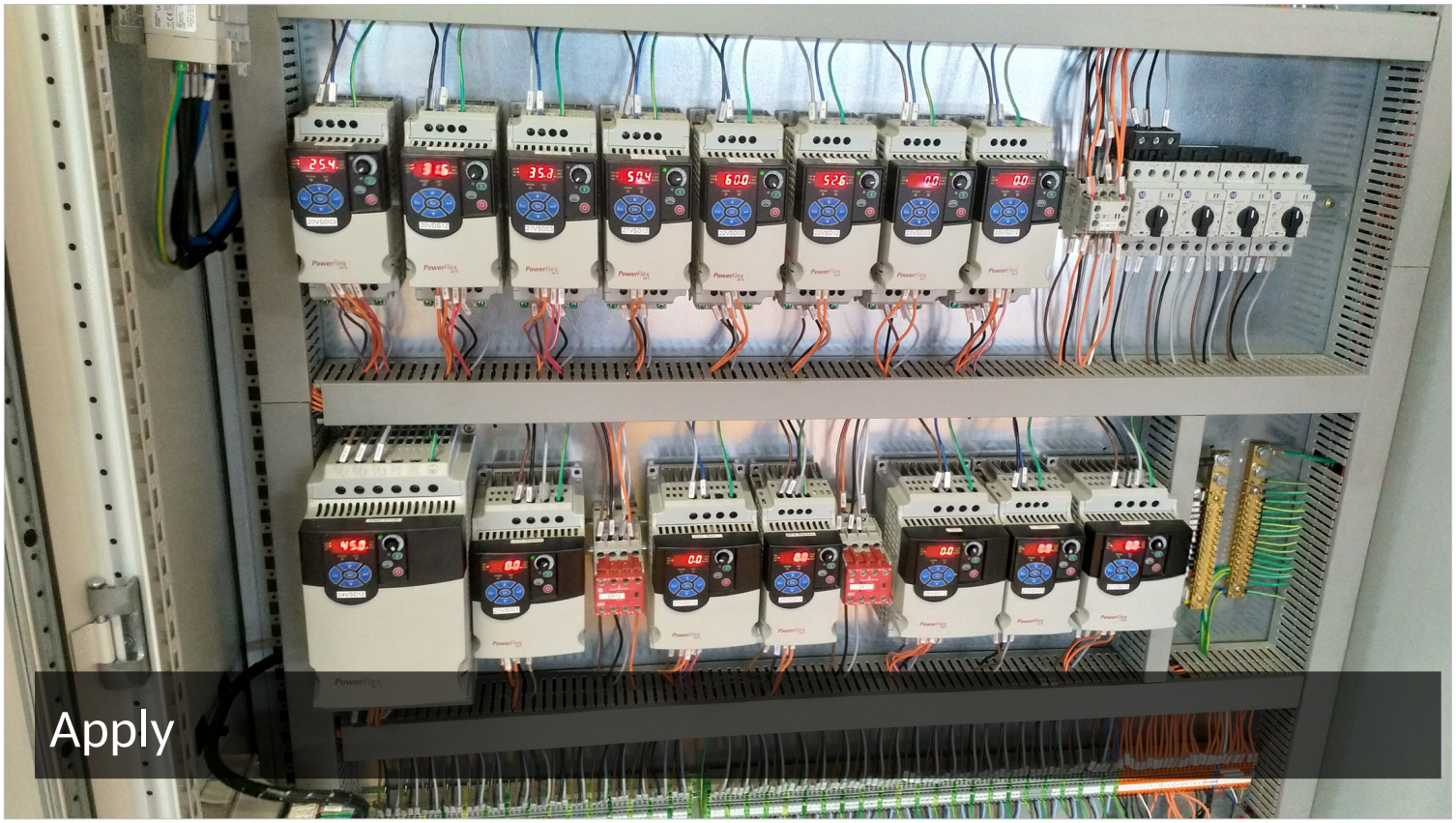
What about the Hemisphere?

$$L_e(x, v) = \int_0^{\pi/2} \int_0^{2\pi} \frac{1}{\pi} \sin(\theta)\, \mathrm{d}\theta\, \mathrm{d}\phi$$



And this is just to visualise, that these are really just 2d functions.

In practise, we will not split the integral into theta and phi, but sample the hemisphere directly. That saves us computing sine, and some of the importance sampling methods would be difficult in this domain..



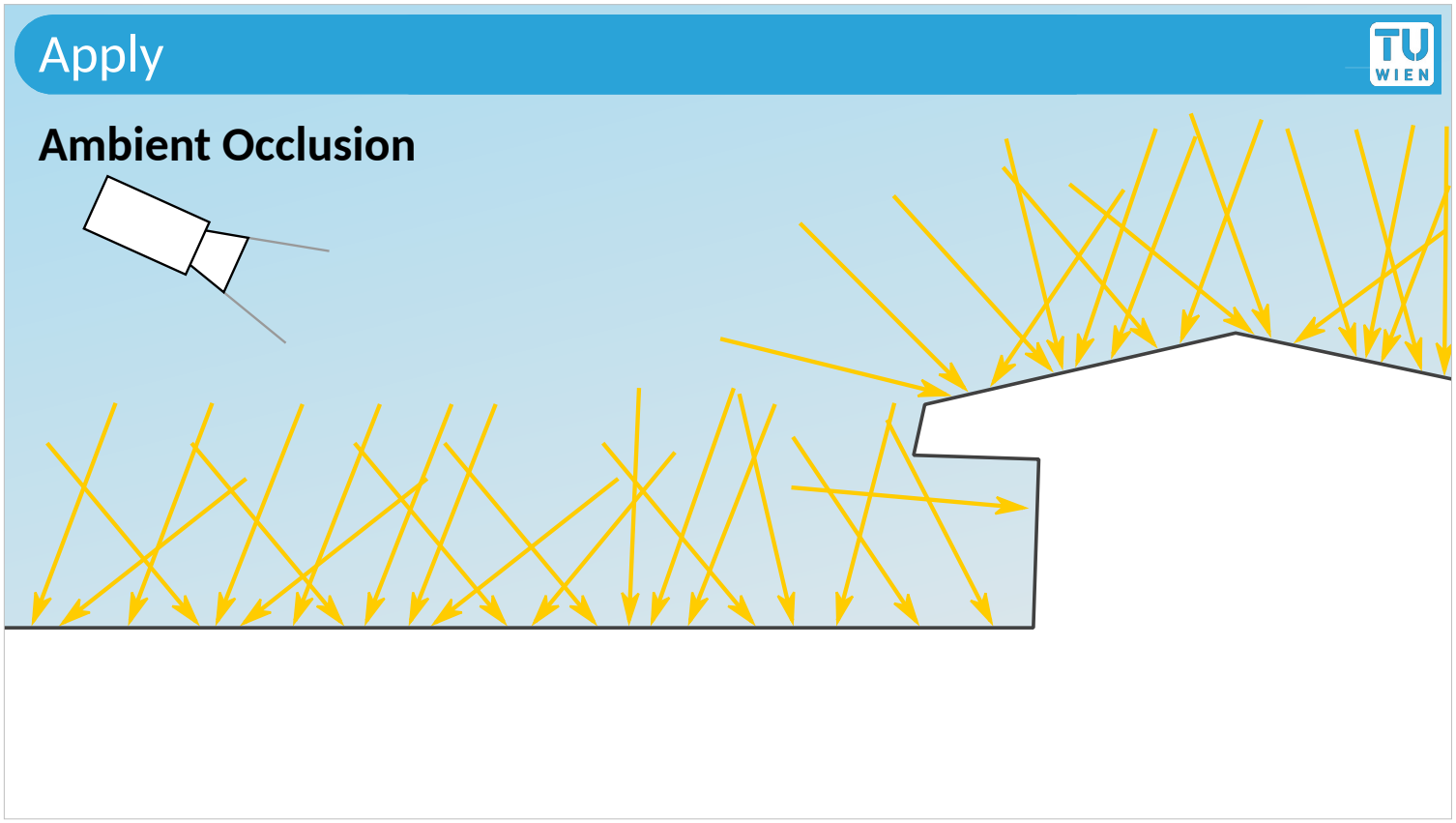
Now that you understand Monte Carlo Integration and the hemisphere, we can finally apply it to compute some nice renderings..

Apply

Ambient Occlusion

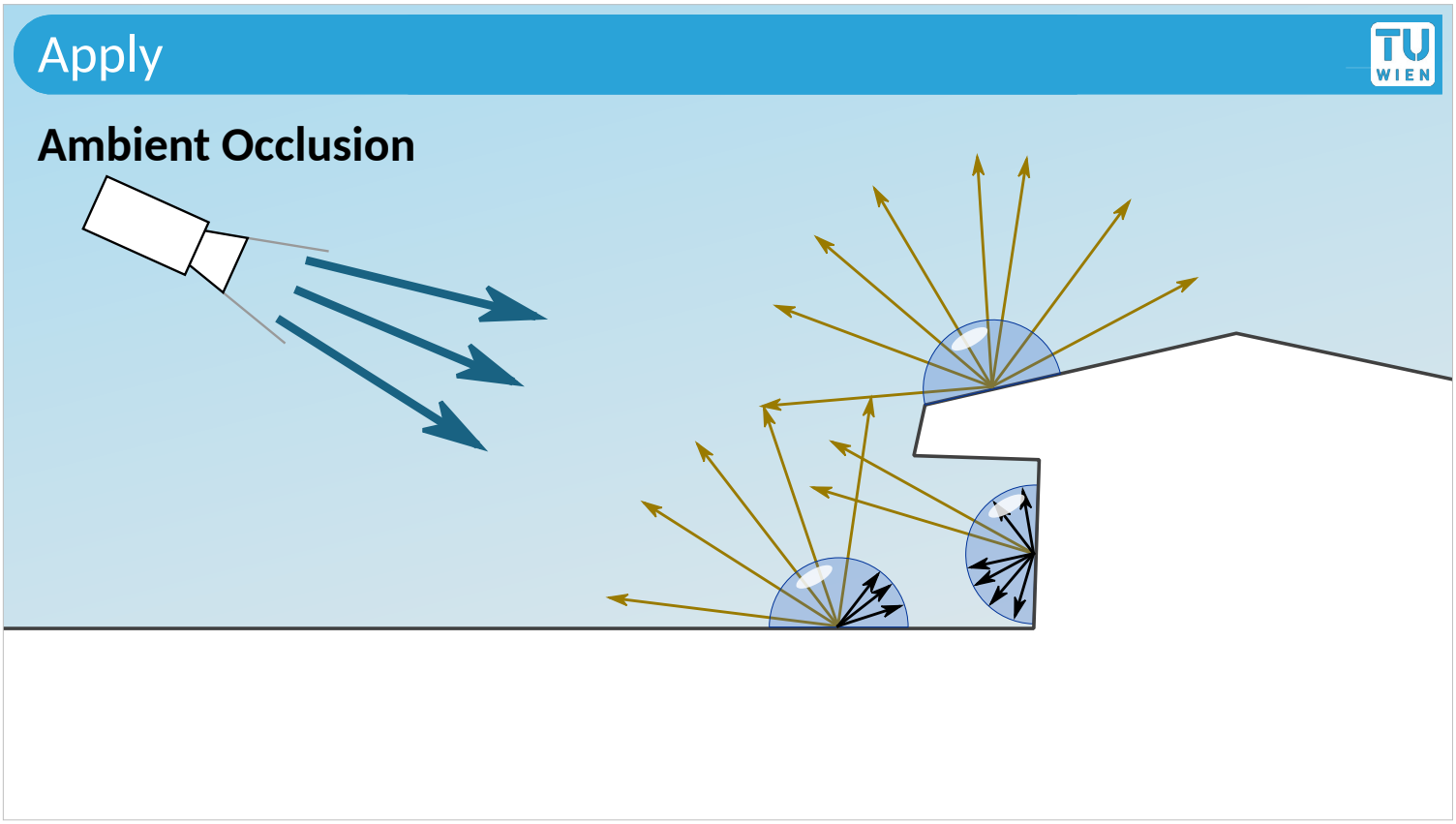
We'll start with ambient occlusion, same as in the assignments.

You might know this technique from one of the real time courses. There you probably computed it in screen space, which is a quicker approximation to the real deal that you'll see here. I'll repeat the principle, so that everybody is on the same page..



Ambient occlusion assumes that light can reach exposed surfaces more easily, and therefore cavities are darker.

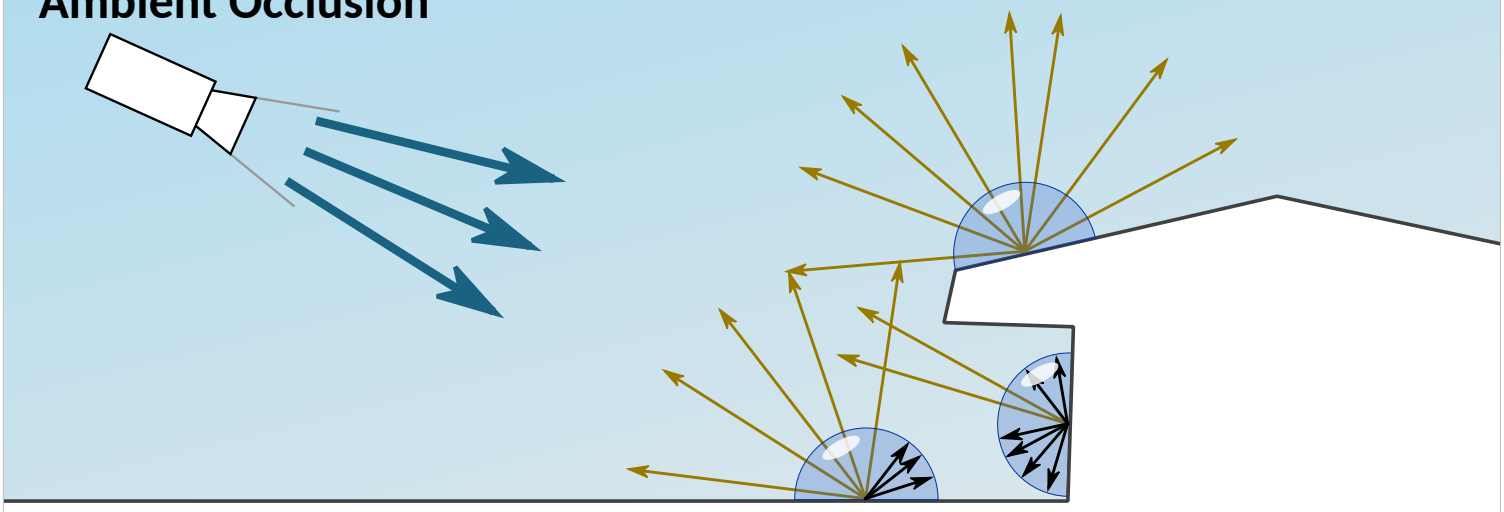
Think of computing the amount of open sky above a point. It'll be less in corners like here and more in points like here and here..



We can compute it by integrating the hemisphere and checking outgoing rays..
Very often, the length of that ray will be caped.

Apply

Ambient Occlusion

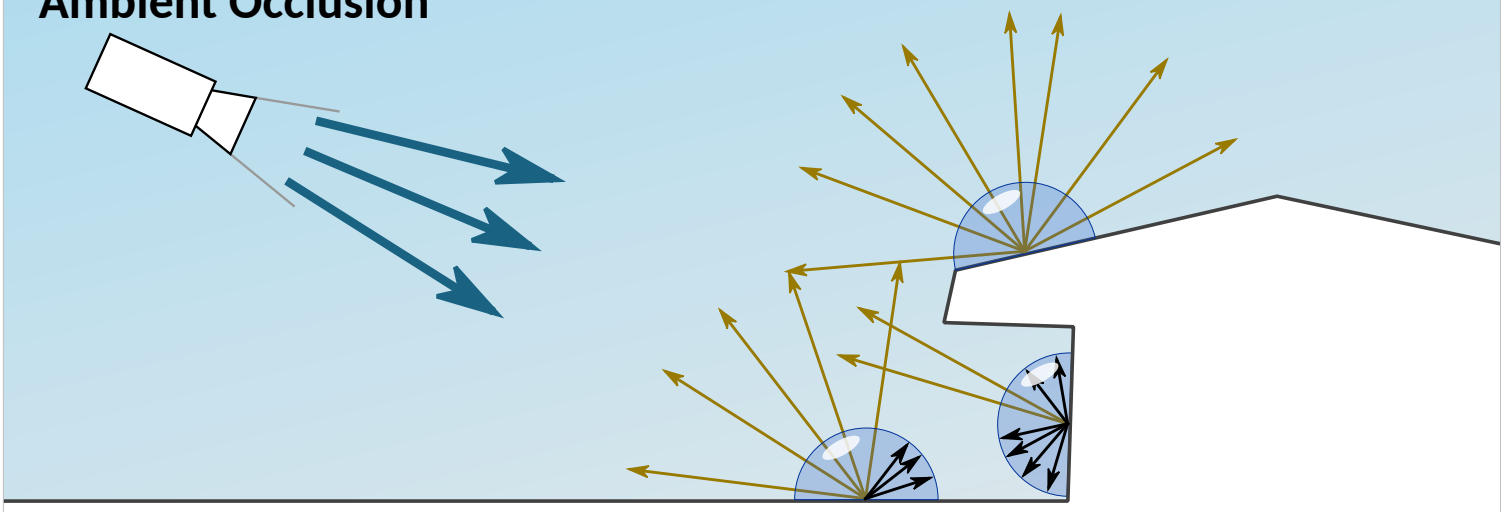


$$L_e(x, v) = \int_\Omega f_r(x, \omega \rightarrow v) L_i(x, \omega) cos(\theta_x) \, \mathrm{d}\omega$$

Here is the integral for reflected light. This is not the integral for ambient occlusion yet, but we'll arrive there shortly..

Apply

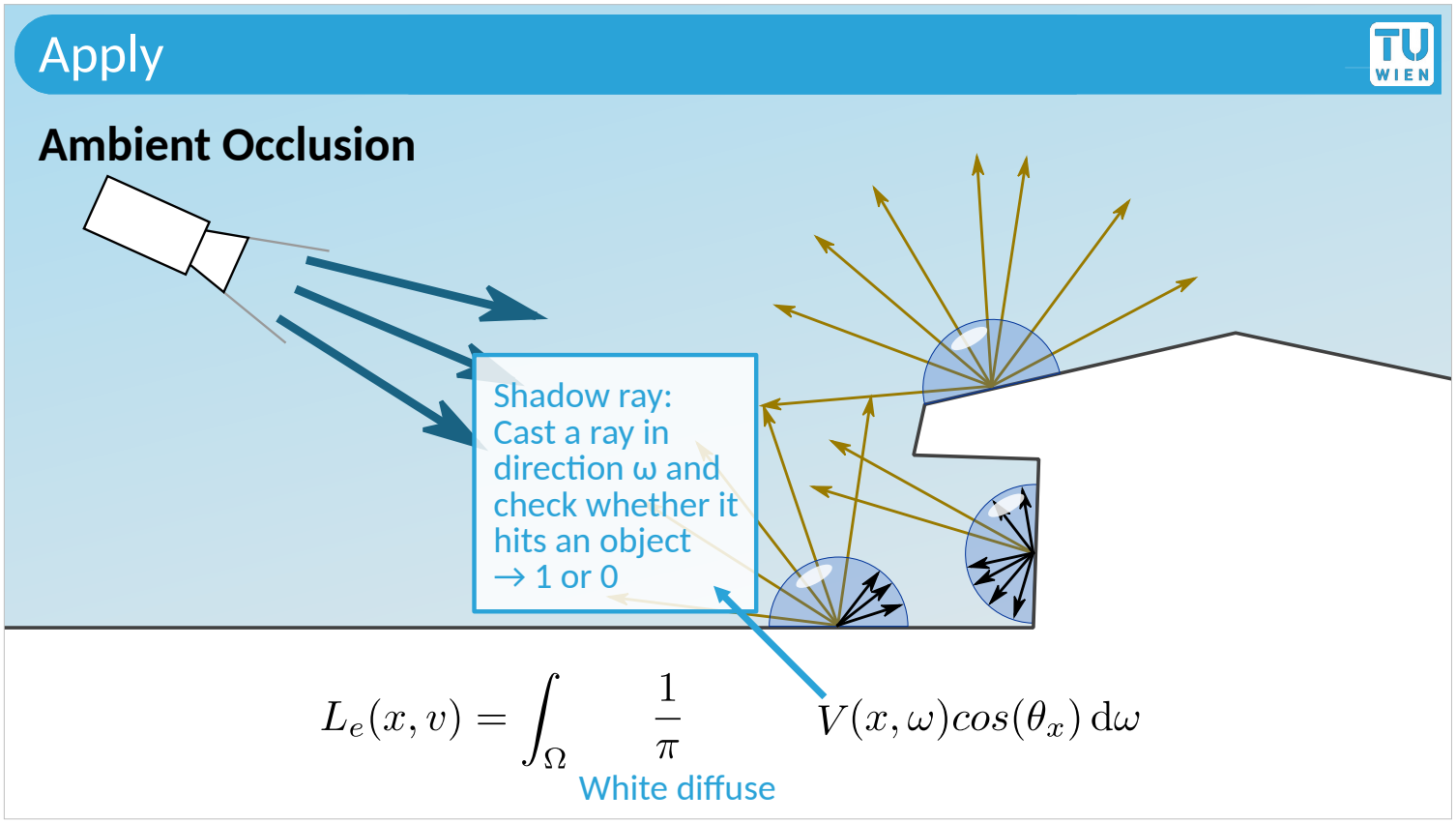
Ambient Occlusion



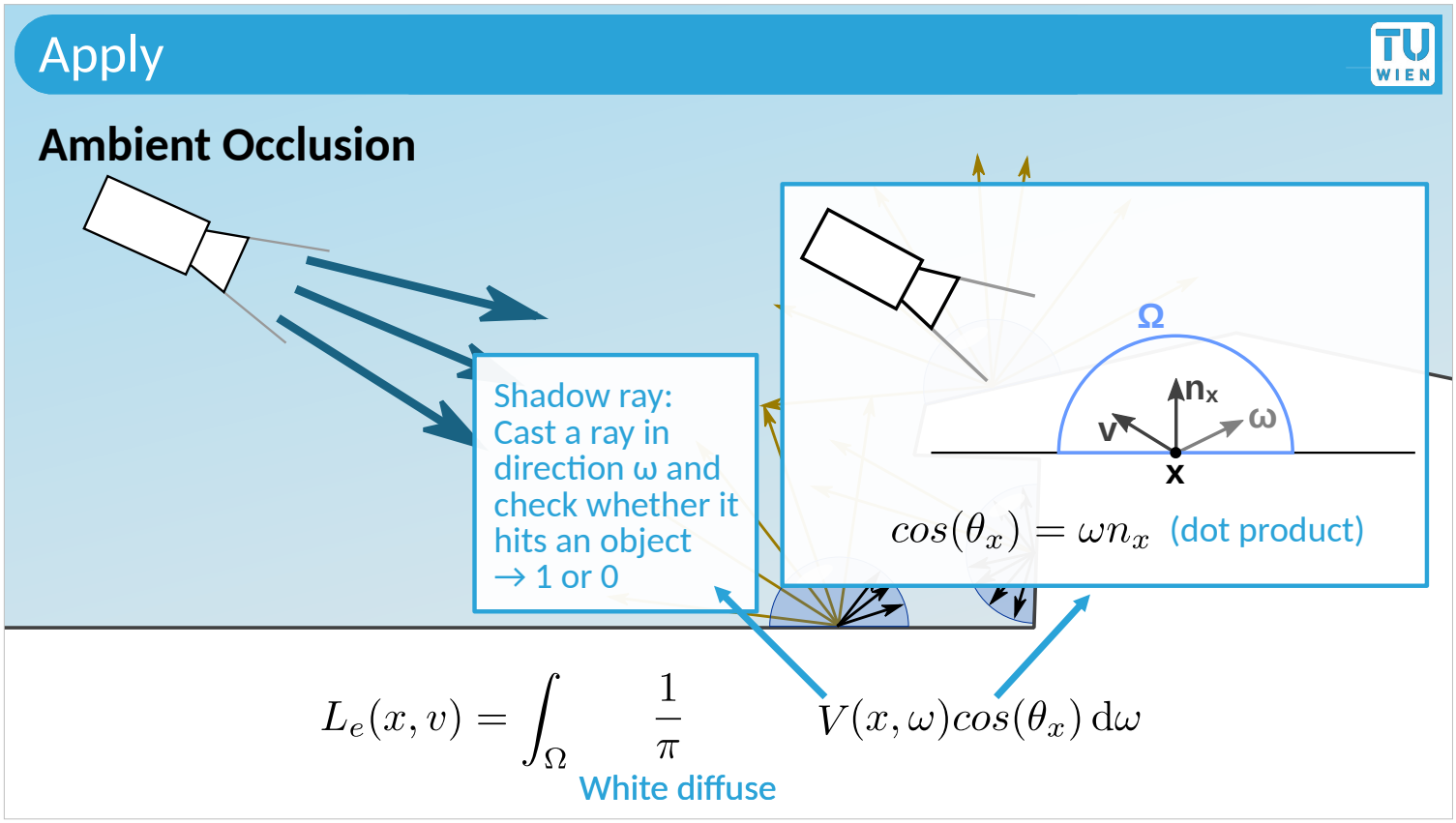
$$L_e(x,v) = \int_\Omega \frac{1}{\pi} \qquad L_i(x,\omega) cos(\theta_x)\,\mathrm{d}\omega$$

White diffuse

We assume white diffuse material again



And replace the incoming light by a shadow ray that gives either 0 or one, depending on whether the ray hit something or not.

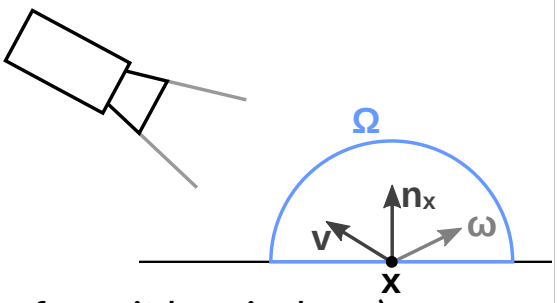


We still need the cosine, which is easy to compute using a dot product beween omega (the sample direction) and the normal.

Apply

Ambient Occlusion

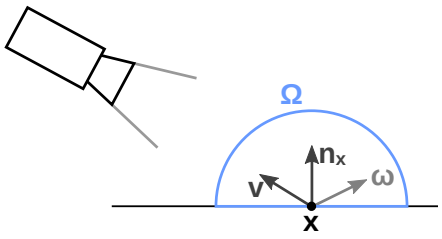
- Cast a ray from camera into the scene and find the first hit-point
- Locally, evaluate $L_e(x, v) = \int_\Omega \frac{1}{\pi} V(x, \omega) \omega n_x \, \mathrm{d}\omega$
- Okey, Monte Carlo:
  - Generate samples for ω -> uniform sampling
  - Evaluate the function -> 1/π V(x, ω) n_xω (needs a ray cast into the scene)
  - Divide by the probability p(ω) = 1/2π (surface of a unit hemisphere)
  - Calculate the average of the above.



That's the algorithm:
We start at the camera, and cast rays according to the pixel position, finding our first hit-point.
On that hit-point we evaluate that integral using Monte Carlo.
To that end, we generate a sample omega using uniform sampling (search the internet or check our assignment sheet).
We then evaluate the function 1/pi times visibility times dot between the normal and omega, which needs a ray cast into the scene
Next, we divide by the probability, which is one over the surface, since we do uniform sampling, and the surface of the hemisphere is 2pi (wikipedia).
And finally we compute the average of these samples..

Apply

Ambient Occlusion



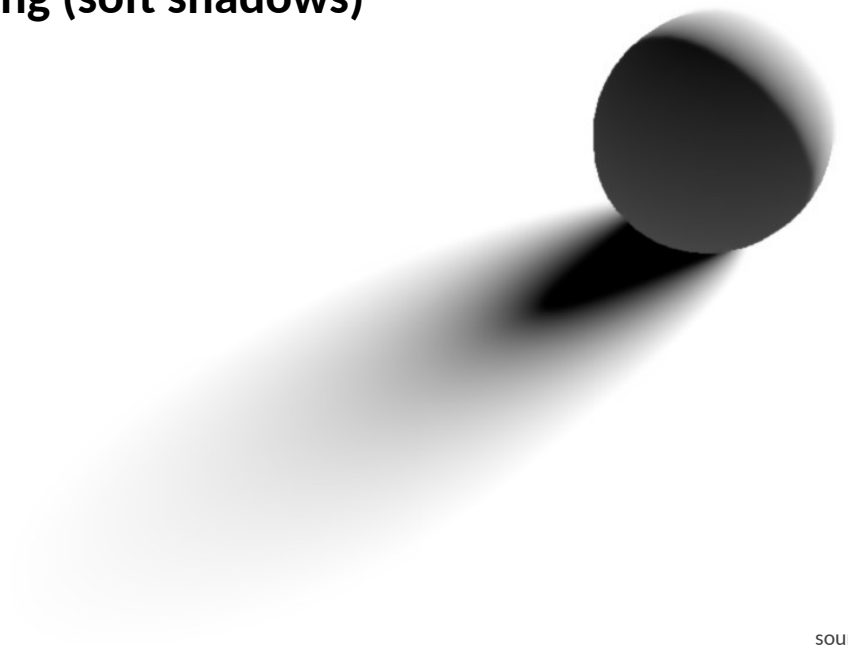

```
for (py = 0; py < height; py++)
    for (px = 0; px < width; px++)
        camera_ray = camera.gen_ray(px, py)
        x = scene.trace(camera_ray)
        if (!x.valid)
            rendering[y][x] = 0;
            continue;
        for (i = 0; i < N; i++)
            omega, prob = uniform_hemisphere(x.normal)
            shadow_ray = ray(x, omega)
            y = scene.trace(shadow_ray)
            f = y.valid * dot(x.normal, omega) / pi
            rendering[y][x]+= f / prob
        rendering[y][x] /= N
```

$$L_e(x,v) = \int_\Omega \frac{1}{\pi} V(x,\omega)\omega n_x \, \mathrm{d}\omega$$

Here is some pseudo code..

When implementing, you can pause, or read the slides on the course homepage. Note that for our students, which use nori, that code is spread over several files and classes. In particular, the loop over N is done globally, while the integrator code will be in separate plugins. The assignment sheets should give enough hints on where to find the relevant parts.. The good thing is, that these can then be reused for other integrators..

Direct lighting (soft shadows)

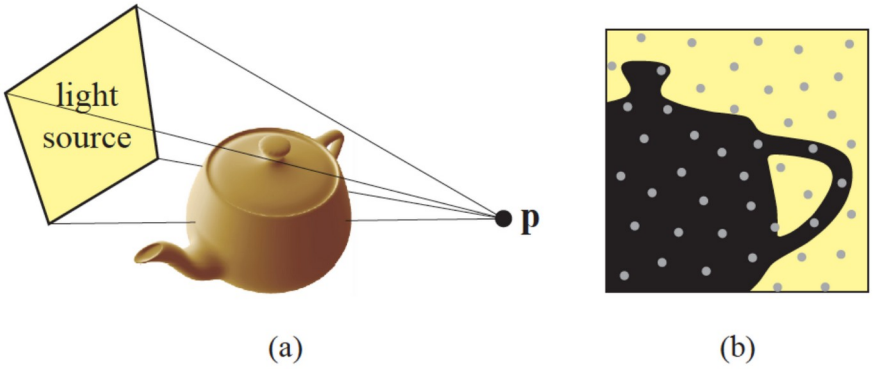


source: Martin Kraus, Wikipedia
(no changes, CC BY-SA 3.0)

Let’s now look at direct lighting, or in other words how to compute soft shadows..

Apply

Intuitive Picture



Laine et al. SIGGRAPH 2005
Slide modified from Jaakko Lehtinen, with permission

Basically, what that does is computing the percentage of visible light for every surface point visible in the camera.

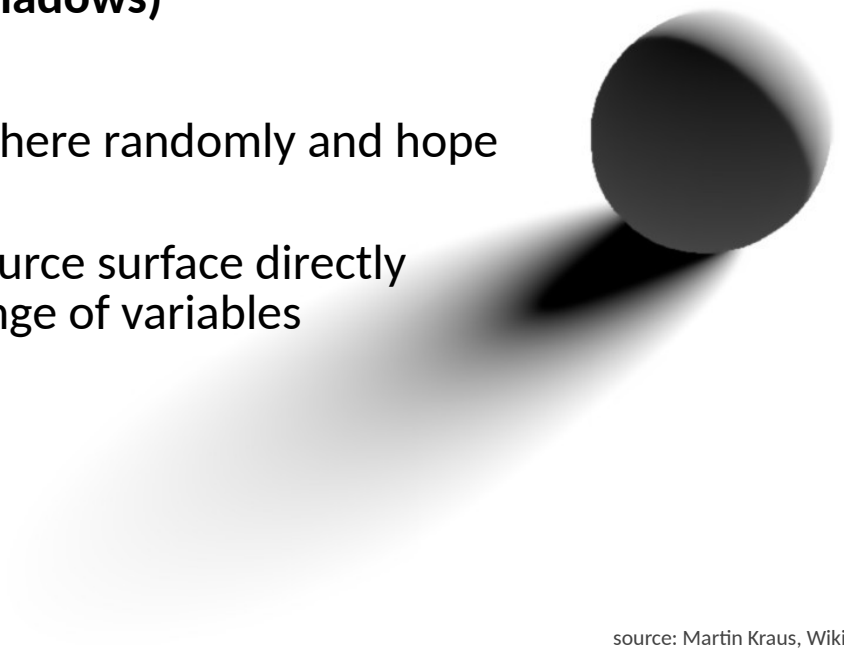
It’s not that different from the ambient occlusion case, just that we aim for light source surfaces instead of the whole world..

Apply

Direct lighting (soft shadows)

We can do it 2 ways

- Sample the hemisphere randomly and hope to hit a light source
- Sample the light source surface directly and perform a change of variables



source: Martin Kraus, Wikipedia
(no changes, CC BY-SA 3.0)

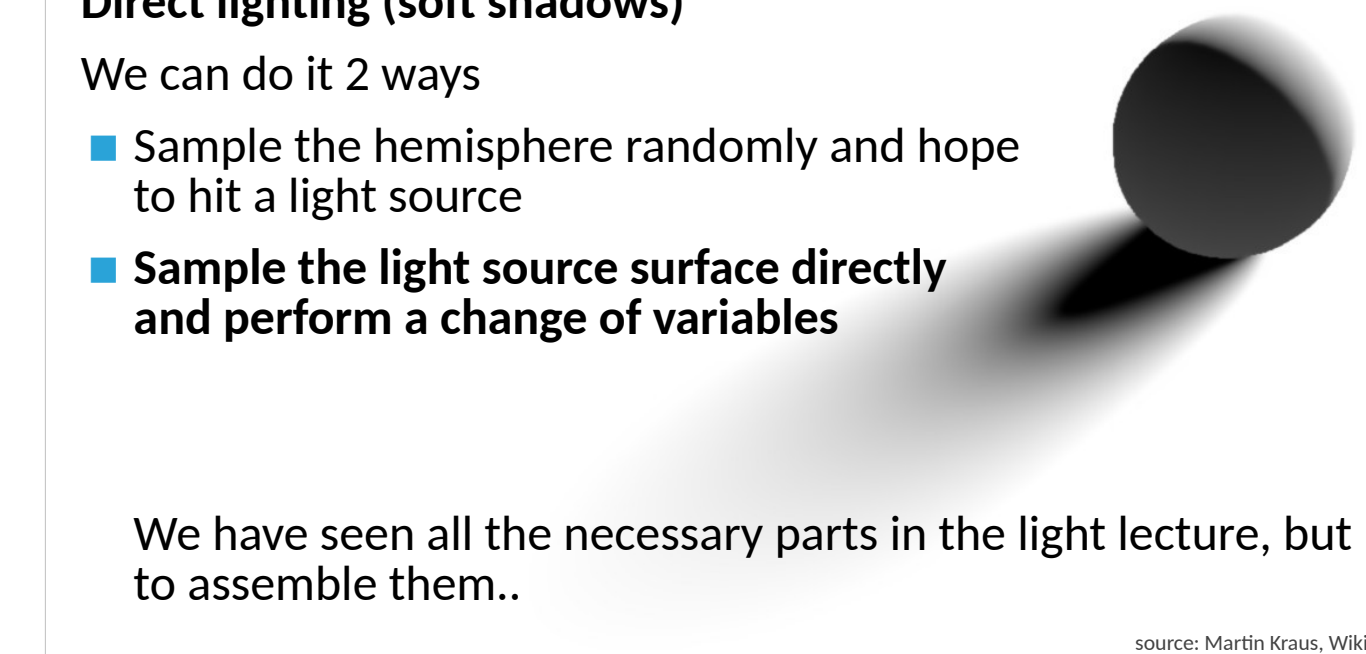
We can do that in 2 ways, either sampling the whole hemisphere, or perform a change of variables and sample the light source surface.

Apply

Direct lighting (soft shadows)

We can do it 2 ways

- Sample the hemisphere randomly and hope to hit a light source
- **Sample the light source surface directly and perform a change of variables**



We have seen all the necessary parts in the light lecture, but need to assemble them..

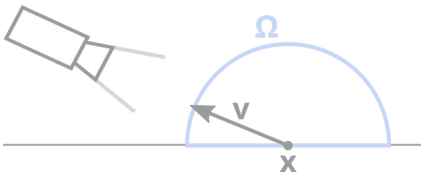
source: Martin Kraus, Wikipedia
(no changes, CC BY-SA 3.0)

In the assignments you can implement both ways. The first way is quite similar to ambient occlusion code wise, so we'll skip it.

For the second way, we have seen all the necessary parts in the lecture about light, but we still need to assemble them..

Apply

Integral for outgoing light



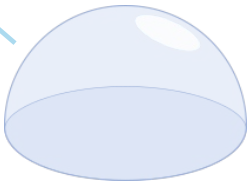
Material, modelled by the BRDF

Light from direction ω

Solid angle

$$L_e(x, v) = \int_\Omega f_r(x, \omega \to v) L_i(x, \omega) \cos(\theta_x) \, \mathrm{d}\omega$$

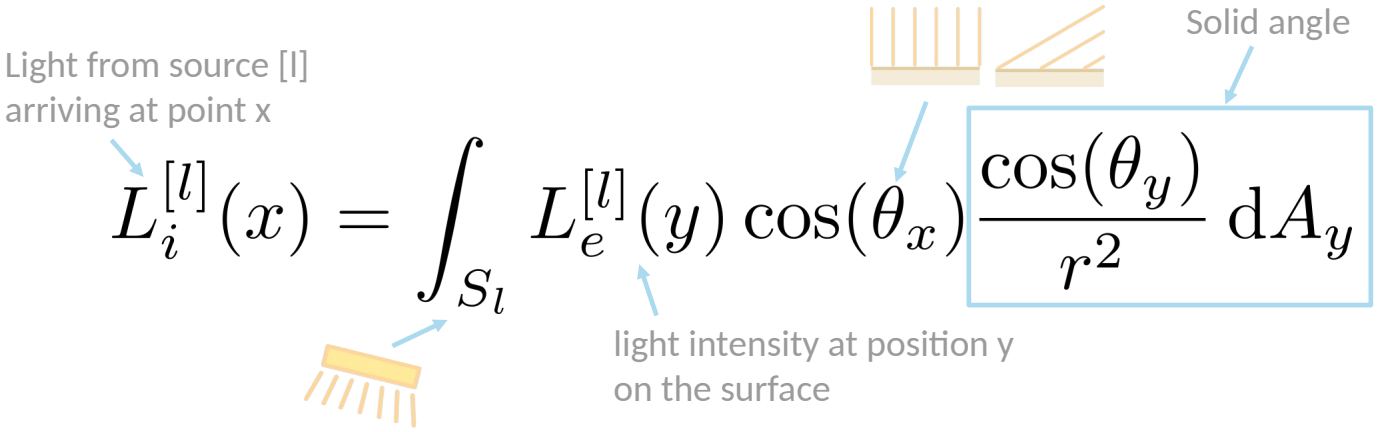
Light going in direction v



Again, the integral for outgoing light. But there is no surface sampling..

Apply

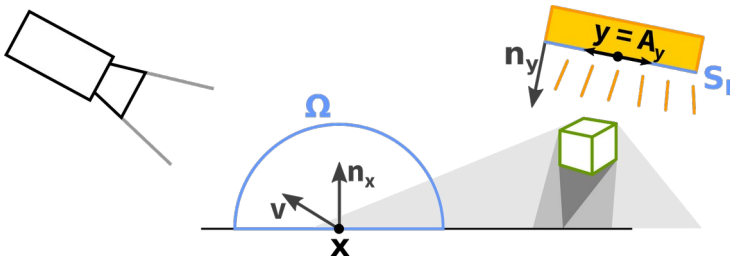
Change of variables for integrating over a light source surface



Here we have the integral for incoming light, with surface sampling and the change of variables, but without the BRDF.

Apply

Direct lighting (soft shadows) (something is missing)

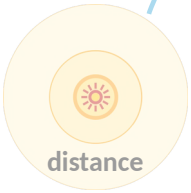


light intensity at position y on the surface

$$L_e^{[l]}(x) = \int_{S_l} f_r(x, y \to v) L_e^{[l]}(y) \cos(\theta_x) \frac{\cos(\theta_y)}{r^2} \mathrm{d}A_y$$

Light going in direction v

Material, modelled by the BRDF

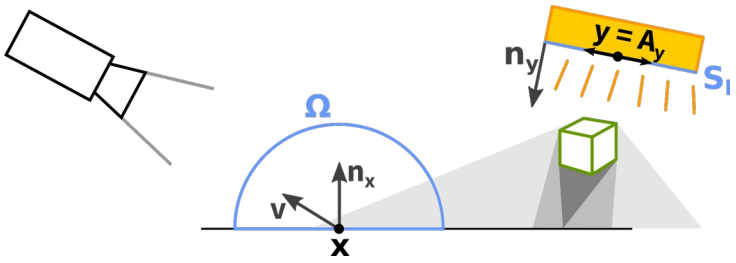


Easy enough to plug those two friends together, but something is missing. Can you spot it?

I'll give you a few seconds..

Apply

Direct lighting (soft shadows) (usable for rendering)



light intensity at position y on the surface

visibility (new, ray tracing)

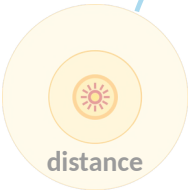
emitter cos(θ)

$$L_e^{[l]}(x) = \int_{S_l} f_r(x, y \to v) L_e^{[l]}(y) V(x, y) \cos(\theta_x) \frac{\cos(\theta_y)}{r^2} \, \mathrm{d}A_y$$

Light going in direction v

Material, modelled by the BRDF

receiver cos(θ)



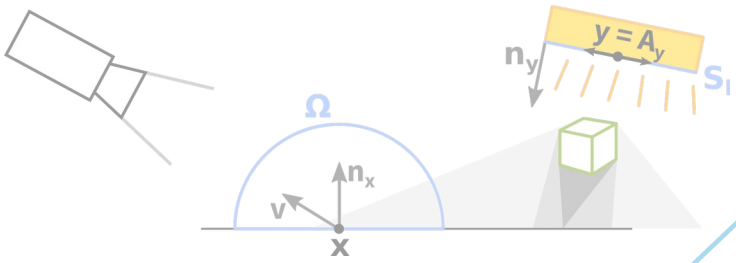
distance

The visibility term!

We need ray tracing to compute it..

Apply

Direct lighting (soft shadows) (the same, but more explicit)



light intensity at position y on the surface

visibility (new, ray tracing)

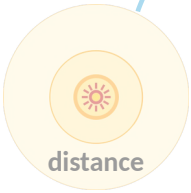
emitter cos(θ)

$$L_e^{[l]}(x) = \int_{S_l} f_r(x, y \to v) L_e^{[l]}(y) V(x, y) \left(\frac{y - x}{|y - x|} \cdot n_x\right) \frac{\left(\frac{x-y}{|x-y|} \cdot n_y\right)}{|x - y|^2} \, \mathrm{d}A_y$$

Light going in direction v

Material, modelled by the BRDF

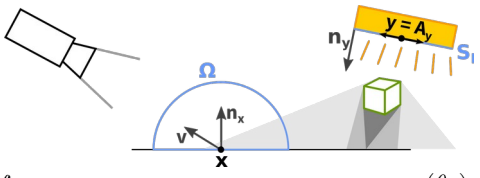
receiver cos(θ)



distance

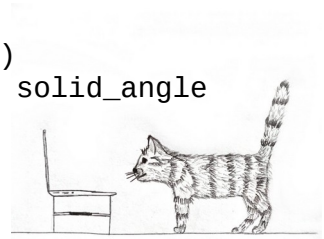
That's the same, but cosines and distance squared is expanded..

Direct lighting (soft shadows)



$$L_e^{[l]}(x) = \int_{S_l} f_r(x, y \to v) L_e^{[l]}(y) V(x, y) \cos(\theta_x) \frac{\cos(\theta_y)}{r^2} \, \mathrm{d}A_y$$

```
for (py = 0; py < height; py++)
    for (px = 0; px < width; px++)
        camera_ray = camera.gen_ray(px, py)
        x = scene.trace(camera_ray)
        if (!x.valid)
            rendering[y][x] = 0;
            continue;
        for (i = 0; i < N; i++)
            y, prob = sample_light_source() // uniform sampling; prob = 1/A
            v = scene.are_visible(x, y)
            omega = (y-x).normalised()
            f_r = scene.brdf(x, omega, v)
            solid_angle = dot(y.normal, -omega) / dot(y-x, y-x)
            f = f_r * y.emittance * v * dot(x.normal, omega) * solid_angle
            rendering[y][x]+= f / prob
        rendering[y][x] /= N
```



And here we have some pseudo code again..

So..

That is it for today. Next lecture we will take a look at the rendering equation, which describes the scattering of light over several bounces, and after that we'll look at path tracing the most important rendering algorithm..

Take care and see you..