Rendering: Monte Carlo Integration I

Bernhard Kerbl

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

With slides based on material by Jaakko Lehtinen, used with permission
Today’s Goal

- Integrating the cosine-weighted radiance $L_i(x, \omega)$ at a point $x$

- Integral of the light function over the hemisphere, w.r.t. direction/solid angle $\omega$

- This is easier said than done!
  - How do we integrate over the hemisphere?
  - $L_i(x, \omega)$ depends on lights, geometry... how can we integrate that?
The solution involves methods from statistics, probability and calculus that are combined to achieve *Monte Carlo Integration*

This is a lot to take in, some of the concepts are complex

We choose to explore them in an illustrative way because grasping the underlying ideas makes their application much easier

We will try to present the bare necessities you need to write a rendering routine two versions: a formal and an intuitive one
Fundamentals Recap

- Calculus
  - Derivatives
  - Integrals

- Probability and Statistics
  - Discrete/Continuous Random Variables
  - Uniform/Non-Uniform Distributions
  - Probability Density Function
  - Expected Value and Variance
Derivatives

- Derivative $f'(x)$ of $f(x)$ gives the rate of change of $f(x)$ at point $x$.

- Answers the question: how does $y = f(x)$ change within an infinitesimally small range $dx$ around $x$? 

  \[ \left( \frac{f(x+dx)-f(x)}{x+dx-x} \right) = \frac{dy}{dx} \]

- Closed-form solutions don’t always exist (discontinuous functions).

- Functions of multiple variables can be derived w.r.t. any of them, yielding a partial derivative (indicated by e.g. $\partial x$ instead of $dx$).
Basic notation: $F(x) = \int f(x) \, dx$

$F(x)$ is any function that fulfills $F(x)' = f(x)$, thus it is generally called the “anti-derivative” of $f(x)$

By this definition, solutions can include arbitrary constants $c$, e.g.:

- $\int \sqrt{x} \, dx = \frac{2^{3/2}}{3} + c$
- $\int x \, dx = \frac{x^2}{2} + c$
- $\int \cos x \, dx = \sin x + c$
Basic notation: $\int_a^b f(x) \, dx$, with
- the variable of integration $x$
- the integration interval $[a, b]$ for $x$
- the function $f(x)$ to integrate (integrand)
- the differential $dx$ for $x$

Informally: “The area under the curve”$^{[1]}$

The differential is an “infinitesimal range”, making $f(x) \cdot dx$ an infinitesimal area. The integral is the sum of these areas in $[a, b]$. 

1. Informal definition of the definite integral as the area under the curve.
Definite Integral: An interpretation

- Basic notation: \( \int_{a}^{b} f(x) \, dx \), with
  - the variable of integration \( x \)
  - the integration interval \([a, b]\) for \( x \)
  - the function \( f(x) \) to integrate (integrand)
  - the differential \( dx \) for \( x \)

- Informally: “The area under the curve”\(^{[1]}\)

- The differential is an “infinitesimal range”, making \( f(x) \cdot dx \) an infinitesimal area. The integral is the sum of these areas in \([a, b]\)
Definite Integral: An interpretation

Basic notation: $\int_a^b f(x) \, dx$, with

- the variable of integration $x$
- the integration interval $[a, b]$ for $x$
- the function $f(x)$ to integrate (integrand)
- the differential $dx$ for $x$

Informally: “The area under the curve”[1]

The differential is an “infinitesimal range”, making $f(x) \cdot dx$ an infinitesimal area. The integral is the sum of these areas in $[a, b]$. 

[1] Rendering – Monte Carlo Integration I
Definite Integral: An interpretation

- Basic notation: \( \int_a^b f(x) \, dx \), with
  - the variable of integration \( x \)
  - the integration interval \([a, b]\) for \( x \)
  - the function \( f(x) \) to integrate (integrand)
  - the differential \( dx \) for \( x \)

- Informally: “The area under the curve”\[1\]

- The differential is an “infinitesimal range”, making \( f(x) \cdot dx \) an infinitesimal area. The integral is the sum of these areas in \([a, b]\)
Definite Integral: An interpretation

- Basic notation: \( \int_a^b f(x) \, dx \), with
  - the variable of integration \( x \)
  - the integration interval \([a, b]\) for \( x \)
  - the function \( f(x) \) to integrate (integrand)
  - the differential \( dx \) for \( x \)

- Informally: “The area under the curve”\(^1\)

- The differential is an “infinitesimal range”, making \( f(x) \cdot dx \) an infinitesimal area. The integral is the sum of these areas in \([a, b]\)
With a solution for the indefinite integral $F(x) = \int f(x) \, dx$, we can solve $\int_{a}^{b} f(x) \, dx = F(b) - F(a)$.

Example:

- Unit circle: $x^2 + y^2 = 1$, area is $\pi$
- $f(x) = y = \sqrt{1 - x^2}$
- $\int f(x) \, dx = \frac{1}{2} (\sqrt{1 - x^2} \cdot x + \sin^{-1} x)$
- $\int_{0}^{1} f(x) \, dx = F(1) - F(0) = \frac{\pi}{4}$
Solving Definite Integrals

- With a solution for the indefinite integral $F(x) = \int f(x) \, dx$, we can solve $\int_{a}^{b} f(x) \, dx = F(b) - F(a)$.

Example:
- Unit circle: $x^2 + y^2 = 1$, area is $\pi$.
- $f(x) = y = \sqrt{1 - x^2}$
- $\int f(x) \, dx = \frac{1}{2} (\sqrt{1 - x^2} \cdot x + \sin^{-1} x)$
- $\int_{0}^{1} f(x) \, dx = F(1) - F(0) = \frac{\pi}{4}$
Remarks on Definite Integrals

- To generalize to $n$-D, we will talk about “volume” rather than area.

- We use subscript-only symbol $\int_D$ for integral over entire domain $D$.

- Integrating 1 over range $[a, b]$ gives the length/volume of the range.

- Integrating 1 over an $n$-D domain gives the volume of the domain.

- A domain $D$ with $X \in [0, 2]$, $Y \in [2, 5]$ in and $Z \in [1, 1.5]$, we have:

$$Vol(D) = \int_D 1 \, dD = \int_{0}^{2} \int_{2}^{5} \int_{1}^{1.5} 1 \, dx \, dy \, dz = 2 \times 3 \times 0.5 = 3$$
Random Variables

- We indicate random variables with capital letters $X, Y, \ldots$ and some Greek symbols for special random variables.

- Random variables are drawn from some *domain* of possible results.

- We define an outcome, or “event” for draws from random variables. $X_i$ marks an observed outcome of a given random variable $X$.

- Random variables can be discrete or continuous. Functions of random variables can themselves be seen as random variables.
The occurrence of values drawn from a random variable usually follows a given *probability distribution*

If a random variable has a uniform distribution, all possible outcomes are equally likely to occur (e.g., a fair die or fair coin)

For non-uniform distributions, the probability of certain values is significantly higher than others (e.g., population body height)
In daily life, we are mostly confronted with *discrete* random results:

- A coin flip
- Toss of a die
- Cards in a deck

Each possible outcome of a random variable is associated with a specific probability $p$. Probabilities must sum up to 1 (100%).

E.g., a fair die: $X \in \{1, 2, 3, 4, 5, 6\}$ and $p_1 = p_2 = \cdots = p_6 = \frac{1}{6}$
A continuous random variable $X$ with a given range $[a, b)$ can assume any value $X_i$ that fulfills $a \leq X_i < b$

Working with continuous variables generalizes the methodology for many complex evaluations that depend on probability theory.

There are infinitely many possible outcomes and, consequently, the observation of any specific event has with vanishing probability.

How can we find the probabilities for continuous variables?\footnote{2}
For continuous variables, we cannot assign probabilities to values.

The *cumulative distribution function* (CDF) lets us compute the probability of a variable taking on a value *in a specified range* \([2]\)

We use notation \(P_X(x)\) for the CDF of \(X\)’s distribution, which yields the probability of \(X\) taking on any value \(\leq x\).

If \(X\) can take on any value with equal probability, what is the probability of \(X = 0.5\)?
Probability for a Range with CDF

- \( P_X(b) - P_X(a) = Pr\{a \leq X_i \leq b\} \)

- Read as: *the probability of* \( X \) *taking on any value from 0 to* \( b \), *minus the probability of* \( X \) *taking on any value from 0 to* \( a \)

- Example: uniform variable \( \xi \) generates values in range \([0, 1)\):
  - \( P_\xi(x) = x \)
  - \( P_\xi(0.75) - P_\xi(0.5) = 0.25 \)
Properties of the CDF

- CDF is bounded by [0, 1] and monotonic increasing
  - Probability of no outcome is 0, the probability of some outcome is 1
  - Die: Rolling a number between 1 and 6 cannot be less probable than rolling a number between 1 and 5

- CDFs can be applied for discrete and continuous random variables

- How do we compute the CDF?
Computing the CDF for Discrete Random Variables

- Determine the limits \([a, b]\) of your variable \(X\)
- For each outcome, find its probability \(p_a, \ldots, p_b\)
- The CDF of that variable is then a function \(P_X(x) = \sum_{i=a}^{x} p_i\)
Probability Density Function (PDF)

- The PDF $p(x)$ is the derivative of the CDF $P(x)$: $p(x) = \frac{dP(x)}{dx}$

- For a PDF $p(x)$, $P(x) = \int p(x) \, dx$ and $\int_{a}^{b} p(x) \, dx = P(b) - P(a)$

- $p(x)$ must be positive everywhere: a negative value would mean we can find $[a, b]$ such that $\int_{a}^{b} p(x) \, dx$ has a negative probability

- $p_{X}(x)$ can be understood as the relative probability of $X_{i} = x$. I.e., if $p_{X}(a) = 2p_{X}(b)$, then $X_{i} = a$ is twice as likely as $X_{i} = b$
Notes about the PDF

- Notation may look like probability, but PDF values can be >1!

- For both discrete and continuous variables, we can reference “p(x)” to describe their distribution

- **Summary**: for a continuous variable $X$ with a known, integrable PDF, we can find the CDF and probabilities of $X$ landing in a given *range*

- ...is this actually helpful?
Creating Variables with Custom Distributions

- By discovering the CDF, we have found a powerful new tool.

- The function is often invertible: this means, we can generate random variables with a desired distribution!

- Rationale: Since the CDF is monotonic increasing, there is a unique value of $P_X(x)$ for every $x$ with $p_X(x) > 0$.

- More informally, if we plot a 1D CDF, any $x$ value that $X$ can take on has a unique, corresponding coordinate on the $y$-axis.
Basic Sampling of Random Variables

We want to generate samples for a custom random variable from a distribution that we can easily obtain in a computer environment.

Our assumed input is the canonical random variable $\xi$:

- continuous (i.e., a real data type)
- with uniform distribution
- in the range $[0, 1)$

Goal: warp samples of $\xi$ to ones distributed according to some $p(x)$.
The Canonical Random Variable

- Our assumed default input variable

- PDF for $\xi$ is 1 in range $[0,1)$ and 0 everywhere else

- CDF for $\xi$ is linear

- Important property: we have equality $P(\xi_i) = \xi_i$
The Inversion Method

- For discrete variables: we draw $\xi$ and iterate event probabilities
- When their sum first surpasses $\xi$, we have found $X_i$
- For continuous variables: exploit $P_X$’s bijectivity and use its inverse!
- We can use canonic $\xi$ to compute $X_i = P_X^{-1}(\xi)$ according to $p_X(x)$
Example: Exponential Distribution

- Used mainly for estimation of time intervals between two events

- The probability of a value decreases exponentially

- Needs additional parameter $\lambda$, often called rate parameter

- We can find its PDF and CDF in most probability text books
  
  - $p(x, \lambda) = \lambda e^{-\lambda x}$
  
  - $P(x, \lambda) = 1 - e^{-\lambda x}$, $P^{-1}(x', \lambda) = -\frac{\log(1-x)}{\lambda}$
Warping Uniform To Exponential Distribution

```cpp
const size_t NUM_SAMPLES = 10'000;

std::array<double, NUM_SAMPLES> exponential_samples{};
std::array<double, NUM_SAMPLES> uniform_samples{};
std::array<double, NUM_SAMPLES> warped_samples{};

void inversionDemo()
{
    const double LAMBDA = 5.0;

    std::default_random_engine rand_eng_uniform(0xdecaf);
    std::default_random_engine rand_eng_exponential(0xcaffe);

    std::uniform_real_distribution<double> uniform_dist(0.0, 1.0);
    std::exponential_distribution<double> exponential_dist(LAMBDA);

    for (int i = 0; i < NUM_SAMPLES; i++)
    {
        auto R_i = exponential_dist(rand_eng_exponential);
        exponential_samples[i] = R_i;

        // uniform distribution: CDF(x) = x
        auto x_ = uniform_samples[i] = uniform_dist(rand_eng_uniform);
        auto X_i = -std::log(1.0 - x_) / LAMBDA;
        warped_samples[i] = X_i;
    }
}
```
Warping Uniform To Exponential Distribution

- Histograms of generated samples

\[ X_i = P_X^{-1}(\xi_i) \]
```cpp
const size_t NUM_SAMPLES = 10'000;

std::array<double, NUM_SAMPLES> exponential_samples{};
std::array<double, NUM_SAMPLES> uniform_samples{};
std::array<double, NUM_SAMPLES> warped_samples{};

void inversionDemo()
{
    const double LAMBDA = 5.0;

    std::default_random_engine rand_eng_uniform(0xdecaf);
    std::default_random_engine rand_eng_exponential(0xcaffe);

    std::uniform_real_distribution<double> uniform_dist(0.0, 1.0);
    std::exponential_distribution<double> exponential_dist(LAMBDA);

    for (int i = 0; i < NUM_SAMPLES; i++)
    {
        auto R_i = exponential_dist(rand_eng_exponential);
        exponential_samples[i] = R_i;

        // uniform distribution: CDF(x) = x
        auto x_ = uniform_samples[i] = uniform_dist(rand_eng_uniform);

        auto X_i = -std::log(1.0 - x_) / LAMBDA;
        warped_samples[i] = X_i;
    }
}
```

This is actually the implementation in many standard libraries anyway.
Let’s add another variable and combine them for 2D output.

In doing so, we are extending our sampling domain.

The sampling domain is defined by:
- The number of variables, and
- Their respective ranges.

Think of the domain as a space with the axes representing variables.
If multiple variables are in a domain, the joint PDF probability density of a given point in that domain depends on all of them.

In the simplest case, with independent variables $X$ and $Y$, the joint PDF of their shared domain PDF is simply $p(x, y) = p_X(x)p_Y(y)$.

We can sample independent variables in a domain by computing and sampling the inverse of their respective CDFs, separately.
Inversion Method Examples in 2D

- 2D with $Y = \xi$. For $X$, use $X \in [0, \frac{\pi}{2})$ and $p(x) = \cos x$

- $P_X(x) = \int p(x) \, dx = \int \cos x \, dx = \sin x$

- $P_X^{-1}(\xi) = \sin^{-1}(\xi)$

```cpp
void inversionDemo2D()
{
    std::default_random_engine x_rand_eng(0xdecaf);
    std::default_random_engine y_rand_eng(0xcaffe);

    std::uniform_real_distribution<double> uniform_dist;

    for (int i = 0; i < NUM_SAMPLES; i++)
    {
        auto x_ = uniform_dist(x_rand_eng);
        auto y_ = uniform_dist(y_rand_eng);

        auto X_i = x_;
        auto Y_i = asin(y_);
        samples2D[i] = std::make_pair(X_i, Y_i);
    }
}
```
**Inversion Method Examples in 2D**

- **X and Y in range [0,1)**
- For both variables, \( p(v) = 2v, P(v) = v^2, P^{-1}(\xi) = \sqrt{\xi} \)

```cpp
std::array<std::pair<double, double>, NUM_SAMPLES> samples2D{};

void inversionDemo2D()
{
    std::default_random_engine x_rand_eng(0xdecaf);
    std::default_random_engine y_rand_eng(0xcaffe);

    std::uniform_real_distribution<double> uniform_dist;

    for (int i = 0; i < NUM_SAMPLES; i++)
    {
        // uniform distribution: CDF(x) = x
        auto x_ = uniform_dist(x_rand_eng);
        auto y_ = uniform_dist(y_rand_eng);

        auto X_i = sqrt(x_);
        auto Y_i = sqrt(y_);

        samples2D[i] = std::make_pair(X_i, Y_i);
    }
}
```
Let’s pick a slow-growing portion of the distribution function.

Compared to $[0,1)$, densities only double inside range $[2,4)$. 
Try $X$ and $Y$ in range $[2,4)$

For both variables, $p(v) = 2v$, $P(v) = v^2$, $P^{-1}(\xi) = \sqrt{\xi}$

Nothing happens.

How can we adapt variable ranges?

Something is missing!
Consider a given range from $a$ to $b$ for a variable and a candidate PDF $f(x)$ with the desired distribution shape.

If $\int_{a}^{b} f(x) \, dx \neq 1$, $f(x)$ is not a valid PDF for this variable.

The probability that the result is one of all possible results $\neq 100\%$.

To fix this, we compute the proportionality constant $c = \int_{a}^{b} f(x) \, dx$ and compute a valid $P(x) = \frac{F(x)}{c}$ while ensuring $p(x) \propto f(x)$. 

Restricting the PDF / CDF
Restricting the PDF / CDF

- For range \([a, b]\) where \(a \neq 0\), we add a constant offset \(k = -P(a)\)

- Try \(X, Y \in [2, 4)\) and \(f(v) = 2v\) again

- We compute \(c_Y = c_X = \int_2^4 2v \, dv = 12\) and add \(k = -\frac{4}{12}\) to get:

\[
P(v) = \frac{v^2 - 4}{12}, \quad P^{-1}(\xi) = 2\sqrt{3} \cdot \xi + 1
\]
The Inversion Method, Completed

- Find a candidate function \( f(x) \) with the desired distribution shape
- Choose the range \([a, b]\) in \( f(x) \) you want your variable to imitate
- Determine the indefinite integral \( F(x) = \int f(x) \, dx \)
- Compute the proportionality constant \( c = F(b) - F(a) \)
- The CDF for the new variable \( X \) is \( P_X(x) = \frac{F(x) - F(a)}{c} \)
- Compute the inverse of the CDF \( P_X^{-1}(\xi) \)
- Use \( P_X^{-1}(\xi) \) to warp the samples of a canonic random variable so that they are distributed with \( p(x) \propto f(x) \) in the range \([a, b]\)
We saw samples being “warped”: we can interpret the inversion method as a spatial transformation of uniform samples.

Let’s treat regular intervals in the input domain as infinitesimal hypercubes: a segment in 1D, a square in 2D and a cube in 3D.

If we warp a space where each variable is $\xi$ to one with joint PDF $p_D$, then $\frac{1}{p_D}$ is the change in volume of the hypercubes after warping.
Visualizing the PDF in 1D

- Let’s look at an example with a custom 1D random variable

- If the target defines the variable $X$, $p_X(x) = 2x$ means the volume of transformed hypercubes at $x = 1$ is half of those at $x = 0.5$

- We check for tiny 1D hypercubes (0.01-long segments)
  - $p_X(x) = 2x, P_X(x) = x^2, x = P_X^{-1}(\xi) = \sqrt{\xi} \iff x = 0.5$ at $\xi = 0.25$
  - $\sqrt{1.00} - \sqrt{0.99} \approx 0.005$
  - $\sqrt{0.25} - \sqrt{0.24} \approx 0.010$
Visualizing the PDF in 2D

- The left represents our inputs and the right our target distribution.
- This time, we warp grid coordinates with the inversion method.

\[ Y = \xi_2 \text{ and } X \in [0,1), p_X(x) = 2x \]
The areas of all 2D hypercubes (squares) are scaled by $\frac{1}{p_X(x)}$.

On the right, rectangles at $(1, y)$ are half the width of the original.

$Y = \xi_2$ and $X \in [0,1), p_X(x) = 2x$
We just saw samples of $X, Y \in [0,1)$ with $p_X(x) = 2x$, $p_Y(y) = 2y$.
In this 2D setup, we have joint PDF $p(x, y) = p_X(x)p_Y(y) = 4xy$.

The areas near point (1,1) are squished to $\frac{1}{4}$ of the original squares.

Visualizing the PDF in 2D

Rendering – Monte Carlo Integration I
This PDF condenses areas at higher values of $x, y$, expands at lower.

If the area changes, the points in it distribute accordingly!
Expected Value

- Expected value of a continuous variable $X$, its domain $D$ and distribution defined by PDF $p_X(x)$, is defined as:

$$E[X]_{p_X} = \int_D x \cdot p_X(x) \, dx$$

- Computes a weighted average over domain, basic average if $X = \xi$

- Answers the question:

“What is the **average** value that we can expect to draw from $X$?”
Variance

- Average (expected), squared deviation from the mean \( \mu = E[X]_{p_X} \)

\[ \sigma_X^2 = Var(X) = E[(X - \mu)^2]_{p_X} \]

- Taking its root \( \sqrt{\sigma_X^2} \) yields the standard deviation \( \sigma_X \)

- Answers the question: “How strongly do values drawn from \( X \) fluctuate about its expected value?”

- Note that, as for expected value, PDF \( p_X \) is included in the definition.
Monte Carlo Integration

- With refreshed knowledge of calculus, random variables, CDFs and PDFs, we have all the tools to approach Monte Carlo integration.

- Simply put, integration approximates the area under a curve with increasing accuracy by splitting it into ever smaller, basic shapes.

- Let us consider this approach to find a way for computing the integral of given functions by sampling.
Why Monte Carlo Integration?

- We cannot always find a closed-form solution for the integral.

- The light function in rendering is one such case.

- We might have a decent idea what the function of incoming light looks like, but its exact shape is not known.
  - Computing the total incoming light at a point means evaluating entire scene geometry for every point we hit.
  - Hard shadows make the light function discontinuous.
  - The rendering equation is an infinite-dimensional (!) integral.
Approximating the Integral

- We can sample an integrand $f(x)$ evenly at regular intervals $h$.

- Find areas of trapezoids under the curve and compute their sum.

- Can simplify to rectangles instead of trapezoids.

- Needs more samples for same precision, but simpler.
Multidimensional Problems

- Regular sampling causes noticeable *patterns* and *aliasing*

![Graph showing regular sampling issues]

The integral computed from these red samples will vastly underestimate the true value!

- Need $N^n$ samples to evaluate an $n$-D function at $\frac{1}{N}$ intervals
  - If we want to sample the grid in 2D, we must change the total number of samples in increments of $2N + 1$, e.g.: 1, 4, 9, 16, etc.
  - This only gets worse with more dimensions (*curse of dimensionality*)
The Curse of Dimensionality

$n$
The Curse of Dimensionality

\[ n^2 \]
The Curse of Dimensionality

$n^3$
Monte Carlo Integration

- Two observations for the integration of a function via sampling
  - The order of the samples doesn’t matter, only their sum
  - We can switch the fixed interval $\frac{1}{N}$ with something expected to be $\frac{1}{N}$

- Replace fixed-order regular samples with uniform random variable
  - Doesn’t matter that generated values are not in any defined order
  - With $N$ uniform samples, the expected interval between them is $\frac{1}{N}$
  - Randomness also reduces aliasing problems!
Monte Carlo Integration for Uniform Variables

- We take $N$ uniform, random samples and treat the results as if we obtained them by subdividing the domain into $N$ regular intervals.

- Sum samples of $f(x)$, multiply with domain volume and average.

$$\int_D f(x) \, dx \approx \frac{\text{Vol}(D)}{N} \sum_{i=1}^{N} f(X_i)$$

- If this seems coarse, remember: we want an approximation of the **total area** under the curve that improves with increasing $N$. 
Monte Carlo Integration for Non-Uniform Variables

- We can generalize the Monte Carlo integration to work with variables that have arbitrary PDFs. The final MC formula:

\[ \int_D f(x) \, dx \approx F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \]

- \( p(X_i) \) tells us how likely it is that samples land in that portion of the domain: values that are sampled frequently receive a smaller weight.

- We can see \( \frac{1}{p(X_i)} \) as the volume of a hypercube \( V_{X_i} \) at sample location \( X_i \) and see that \( \frac{V_{X_i}}{N} f(X_i) \) is quite close to \( \frac{Vol(D)}{N} f(X_i) \).
The Rationale Behind $1/p(x)$

- Using a non-uniform $p(x)$ to sample a constant function $f(x)$
- Sample arrows indicate the value of $\frac{1}{p(x)}$: blue = low, red = high
- Red samples are rare, they represent a larger area under the curve
The Rationale Behind $1/p(x)$

- Using a non-uniform $p(x)$ to sample a non-uniform function $f(x)$
- Same weight for each sample: overestimates area under the curve
- Using $\frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)}$ instead of $\frac{\text{Vol}(D)}{N} \sum_{i=1}^{N} f(X_i)$ is the right choice
The Rationale Behind $1/p(x)$

**Final word:** During Monte Carlo integration, we use $\frac{1}{p(x)N}$ from the start as the $\Delta x$, so that $\Delta x \cdot f(x)$ gives us an area under the curve. The more samples $N$ we take, the closer the distance between the two closest samples near a point $x$ gets to $\frac{1}{p(x)N}$ and the better the approximation of the true integral, i.e., the sum of infinitesimal areas under the curve.
Verifying the Monte Carlo Integral

- Formal verification that expected value of $F_N$ is the integral of $f(x)$

$$E[F_N] = E \left[ \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \right] \quad \text{with} \quad X \in D$$

- Constants and sums can be moved out of the expected value operator

$$= \frac{1}{N} \sum_{i=1}^{N} E \left[ \frac{f(X_i)}{p(X_i)} \right]$$

$$= \frac{1}{N} \sum_{i=1}^{N} \int_{D} \frac{f(x)}{p(x)} p(x) \, dx$$

- Expected value for any event $X_i$ drawn from $X$ is equal to $E[X]$

$$= \frac{1}{N} \sum_{i=1}^{N} \int_{D} f(x) \, dx = \int_{D} f(x) \, dx$$

- Probability of $\frac{f(x)}{p(x)}$ depends only on $x$
Importance Sampling

- Importance sampling = picking a good PDF that adapts to $f(x)$

- Intuitive justification: Sample more in places where we have larger contributions to the integral to capture high-frequency details there
Choosing the Right PDF

\[ \int_D f(x) \, dx \approx F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \]

- \( F_N \) is itself a random variable, variance shows up as random noise

\[ Var(F_N) = \frac{1}{N} Var \left( \frac{f(x)}{p(x)} \right) = \frac{1}{N} E \left[ \left( \frac{f(x)}{p(x)} - E \left[ \frac{f(x)}{p(x)} \right] \right)^2 \right] \]

- No noise if \( \frac{f(x)}{p(x)} \) is a constant \( \Rightarrow \) what is a good PDF to choose?
Choose a PDF that mimics the shape of $f(x)$, but is easy to sample.

- Note: $\int_D p(x) \, dx$ must integrate to 1, so can’t just take $p(x) = f(x)$.
- To normalize $\int_D f(x) \, dx$, we would have to know the integral.
The Importance of Importance Sampling

5 Samples/ Pixel
The Importance of Importance Sampling

Rendering – Monte Carlo Integration I
The Importance of Importance Sampling

5 Samples/Pixel, with importance sampling
A minimal sampling and integration procedure could look like this:

Given: function $f(x)$, PDF $p(x)$ and CDF $P(x)$

```python
value = 0
for i in [0, N) do
    u = uniform_random_sample()
    x = P_inverse(u)
    value += f(x)/p(x)
end for
value /= N
```
References and Further Reading

- Slide set based mostly on chapter 13 of *Physically Based Rendering: From Theory to Implementation*


