Perception-based Optimization of Wavelength Sampling Distributions for Spectral Rendering

Camil-Cristian Dobos

Supervisor(s): Elmar Eisemann, Christoph Peters, Michael Weinmann

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Camil-Cristian Dobos
Final project course: CSE3000 Research Project
Thesis committee: Michael Weinmann, Christoph Peters, Georgios Smaragdakis

Abstract

Spectral Monte Carlo rendering stands as the gold standard for accurately simulating complex optical phenomena, such as fluorescence and chromatic dispersion, but typically exhibits slow convergence behavior due to challenges in sampling the wavelength domain. Convergence is even slower for scenes with complex spectral distributions, and hence a robust method for sampling the wavelength domain is crucial for better performance. To address this challenge, we propose a preprocessing step consisting of optimizing a set of distributions specifically for wavelength sampling, and we investigate whether this approach can achieve a lower perceptual error than traditional sampling techniques. Our evaluation indicates that our method can significantly reduce perceptual error in single-emitter scenes featuring a complex spectral power distribution (SPD), when compared to uniform sampling, without incurring any additional overhead during rendering.

1 Introduction

In real-world scenes, light manifests as a continuous spectrum of electromagnetic radiation. For practical purposes, most physically based rendering systems represent this spectral distribution by encoding colour information into three numerical values (e.g. R, G, B) which model the way humans generally perceive colour. This representation cannot accurately simulate wavelength-dependent effects, such as chromatic dispersion, fluorescence, or metamerism, and is known to be insufficient for accurate color reproduction (Borges, 1991). Spectral rendering overcomes these limitations by representing color using full spectral power distributions across wavelengths, rather than tristimulus values.

Spectral rendering is generally implemented using advanced Monte Carlo light transport algorithms, which are extended to handle sampling of the wavelength domain. However, such algorithms come with a very high computational cost, largely attributed to the need to sample the spectral domain, leading to slow convergence and high variance. Furthermore, scenes containing complex spectral features, such as narrow-band emission from certain light sources or materials with sharp peaks and valleys in their reflectance spectra, pose significant challenges.

Recent techniques have significantly reduced noise by optimizing how wavelengths are sampled. The hero wavelength method (Wilkie et al., 2014) traces a path guided by a single primary wavelength and then evaluates that path for multiple secondary wavelengths selected via stratified sampling. In a related strategy, Evans and McCool proposed propagating a cluster of wavelengths that travel together along a unified path, splitting only at dispersive events (Evans & McCool, 1999). A different approach introduced by Van de Ruit and Eisemann consists of a multi-pass method that constructs per-pixel spectral importance distributions to guide sampling, achieving significant reductions in variance with limited overhead (van de Ruit & Eisemann, 2021).

Although such advancements managed to significantly reduce variance, spectral renderers may still exhibit slow convergence, primarily due to the difficulty in efficiently sampling the wavelength domain.

By leveraging the fact that human visual sensitivity varies across wavelengths, spectral sampling can be guided to converge more quickly toward perceptually convincing solutions. This perceptual prioritization could allow renderers to focus computational effort where it matters most for visual fidelity. This idea has been explored by Radziszewski et al., who proposed sampling wavelengths using a distribution constructed from the CIE XYZ color-matching functions (Radziszewski et al., 2009).

In this paper, we explore the idea of perceptual-based sampling further. Our contribution consists of a preprocessing step, in which we optimize a set of probability distributions for sampling multiple wavelengths from an illuminant spectrum, based on the perceived color difference (Sharma et al., 2005). We extend a spectral path tracer (Pharr et al., 2023) by loading our optimized distributions and implementing our custom sampling procedure, and show that our method lowers perceptual error in scenes featuring a complex illuminant spectrum, when compared with standard uniform sampling of the wavelength domain.

The remainder of this paper is structured as follows. We begin by reviewing related work and background information in Section 2. Section 3 then details our proposed method, followed by our implementation in Section 4. In Section 5 we evaluate our method, and discuss the results, limitations, and possible directions for future research in Section 6. After considering the ethical implications of our work in Section 7, we conclude in Section 8 with a summary of our findings.

2 Background and Related Work

2.1 Monte Carlo Integration

Physically based rendering often heavily relies on Monte Carlo integration techniques to efficiently compute high-dimensional integrals (Kajiya, 1986). In its most basic form, Monte Carlo estimates the value of an integral $I = \int_a^b f(x)dx$ by:

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

where $X_0, X_1, ..., X_{N-1}$ are points sampled uniformly in $[a, b]$. According to the law of large numbers, the estimated value approaches the true value I as $N \to \infty$.

Importance Sampling involves sampling from a proposal distribution $p(x)$ which is ideally as close as possible in shape to the integrand. This helps reduce variance by focusing samples on places of interest. The new estimator becomes:

$$I \approx \frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i)}{p(X_i)} \tag{2}$$

Note that because we are sampling from p, we need to weigh our samples by $p(X_i)$ to remove the resulting bias.

Multiple Importance Sampling (MIS)
There are situations in which we may have multiple sampling strategies $p_0(x), p_1(x), ..., p_{k-1}(x)$, each good at sampling different parts of $f(x)$. From each of these distributions p_i, *MIS* (Veach, 1998) draws N_i samples and combines them into one weighted estimator:

$$I \approx \sum_{i=0}^{k-1} \frac{1}{N_i} \sum_{j=0}^{N_i-1} w_i(x_j) \frac{f(x_j)}{p_i(x_j)} \tag{3}$$

A popular choice for the weights is the balance heuristic:

$$w_i(x) = \frac{N_i p_i(x)}{\sum_{j=0}^{k-1} N_j p_j(x)}, \tag{4}$$

which is equivalent to sampling from the mixture of the k distributions (Veach, 1998).

2.2 Spectra and Color

The XYZ color space (CIE, 1932) serves as a standard for quantifying color in various media, as it was designed to be device independent and to encompass all color perceivable by an average human observer.

The **XYZ color matching functions** (Figure 8), denoted by $\bar{x}(\lambda), \bar{y}(\lambda), \bar{z}(\lambda)$, are used to convert a spectral power distribution (SPD) $P(\lambda)$ to the XYZ color coordinates:

$$X = k \int_\Lambda \bar{x}(\lambda) P(\lambda) d\lambda \tag{5}$$

$$Y = k \int_\Lambda \bar{y}(\lambda) P(\lambda) d\lambda \tag{6}$$

$$Z = k \int_\Lambda \bar{z}(\lambda) P(\lambda) d\lambda, \tag{7}$$

where k is a scaling constant.

As Evans and McCool note, the resulting coordinates (X, Y, Z) should be interpreted as projections of the infinite-dimensional function $P(\lambda)$ onto a finite-dimensional vector space (Evans & McCool, 1999). Furthermore, they do not constitute a basis for reconstructing the original spectrum due to a phenomenon called *metamerism*, in which for every (X, Y, Z) triplet there is an infinite number of SPDs that can produce it.

2.3 The Rendering Equation

Spectral rendering involves simulating the transport of light within a scene, typically by evaluating the rendering equation (Kajiya, 1986), for which we use an extended version (van de Ruit & Eisemann, 2021) of the path-integral formulation (Veach, 1998):

$$I_j = \int_\Lambda \int_\Omega f_j(\bar{\alpha}, \lambda) \, d\mu(\bar{\alpha}) \, d\lambda \tag{8}$$

Here, I_j describes the radiance arriving at pixel j, Λ is the wavelength domain, and Ω is the path space of finite-length paths $\bar{\alpha}$. The throughput for a path $\bar{\alpha}$ at wavelength λ is denoted by $f_j(\bar{\alpha}, \lambda)$.

The integral is infeasible to solve analytically, so instead Monte Carlo integration with standard *importance sampling* is used to derive an estimate by drawing N samples from a predefined target distribution p:

$$\hat{I}_j = \frac{1}{N} \sum_{i=0}^{N-1} \frac{f_j(\bar{\alpha}_i, \lambda_i)}{p(\bar{\alpha}_i, \lambda_i)} \tag{9}$$

The term $p(\bar{\alpha}_i, \lambda_i)$ represents the *probability density function* for sampling a path-wavelength pair, which can be expressed as:

$$p(\bar{\alpha}, \lambda) = p(\lambda) p(\bar{\alpha} \mid \lambda) \tag{10}$$

Here, $p(\lambda)$ represents the wavelength sampling distribution, and can be expressed as $p(\lambda) = p_s(\lambda) p_e(\lambda)$, where p_s constitutes a sensor response, and p_e is another distribution, typically uniform or a proportional to scene emission. In their work on Continuous Multiple Importance Sampling (CMIS), West et al. use a mixture of emitter SPDs for p_e (West et al., 2020). Their method reduces color noise and shows improvements, particularly for complex illuminants.

In a different approach, Van de Ruit and Eisemann propose a pre-pass in which they estimate incident spectral radiance at each pixel (van de Ruit & Eisemann, 2021). They do this by first running a cost-effective path tracer with a lower sample count, lower image resolution, and certain restrictions regarding which paths can be traced. Then a reconstruction function upsamples this coarse estimate to create a full-resolution spectral approximation. This resulting estimate directs the wavelength sampling process by serving as a more informed distribution for p_e.

2.4 Sampling Multiple Wavelengths

The Monte Carlo estimator in Equation 8 evaluates the integrand for a single wavelength per path. While this method is straightforward to implement, it is inefficient and introduces considerable color noise. This approach is often described as wasteful because the geometric path generated is typically valid for all wavelengths, yet it only contributes information for the single one that was sampled. To address this, several methods have been proposed to evaluate multiple wavelengths for each path.

In their paper on Stratified Wavelength Clusters, Evans and McCool propose carrying several wavelength samples, grouped in a *cluster*, along each light transport path (Evans & McCool, 1999). The cluster is computed by first selecting a light source at random, and then a set of *K* stratified random numbers are generated and warped through the inverse of that light's cumulative normalized SPD.

A related technique proposed by Wilkie et al., Hero Wavelength Spectral Sampling, consists of randomly sampling a primary *hero wavelength* for each path and base all directional sampling decisions solely on it (Wilkie et al., 2014). The additional wavelengths are placed at equal distances from the hero wavelength to ensure they always cover the visible spectrum evenly. This method uses multiple importance

sampling (2.1) to correctly weight the contribution from each wavelength in the set.

While the strategy of stratifying wavelengths, such as placing them at equal distances to evenly cover the visible spectrum, provides robust spectral coverage that effectively reduces color noise, this may prove inefficient when dealing with complex spectra. For illuminants with sharp peaks and valleys, such as fluorescent lights, the rigid, uniform placement of samples may fail to capture these narrow, high-energy peaks.

3 Methodology

A key strategy for reducing color noise in spectral rendering is to importance sample wavelengths based on an illuminant spectrum, which is usually sampled uniformly or based on relative emission. This focuses computation on the most prominent wavelengths where the light source emits more energy.

Prior methods have often implemented this using stratified sampling (Wilkie et al., 2014; Evans & McCool, 1999), but as we have seen, this can be inefficient for complex spectra. To address these limitations, we propose generating wavelength sampling functions through a pre-optimization process. We optimize a set of correlated sampling functions with the goal of minimizing the final perceived color error, measured using the CIE Delta E standard (Sharma et al., 2005).

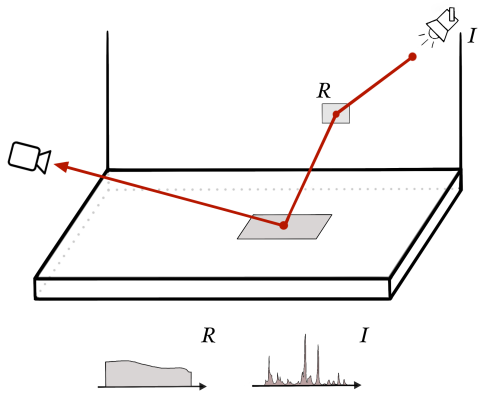

Figure 1: Simplified illustration of a light path. R is a reflectance spectrum, and I is the illuminant spectrum.

Concretely, our goal is to sample N wavelengths based on a single uniform random number u in $[0, 1]$ using some monotonic functions $\lambda_0(u), \lambda_1(u), ..., \lambda_{N-1}(u)$ that have not yet been determined. These functions act as inverse cumulative distribution functions (CDFs) for probability density functions (PDFs) $p_0(\lambda), p_1(\lambda), ..., p_{N-1}(\lambda)$, which we want to optimize for sampling from a chosen illuminant spectrum I.

If we evaluate the integral in Equation 8 for a single path $\bar{\alpha}$, apply the color matching functions $\bar{x}, \bar{y}, \bar{z}$, and expand the term $f_j(\bar{\alpha}, \lambda)$, the integral in the wavelength domain that we wish to estimate is:

$$c_l = \int_\Lambda h_l(\lambda) R(\lambda) I(\lambda) \quad \text{for } l \in \{0, 1, 2\}, \tag{11}$$

where $h_0 = \bar{x}, h_1 = \bar{y}, h_2 = \bar{z}$, R is the product of the reflectance spectra encountered along the path, and I is the illuminant spectrum for which we wish to adapt our sampling strategy.

Using MIS (2.1) with our importance sampling distributions $p_0, p_1, ..., p_N$, an unbiased Monte Carlo estimate can be derived:

$$c_l \approx \sum_{j=0}^{N-1} \frac{h_l(\lambda_j(u)) R(\lambda_j(u)) I(\lambda_j(u))}{\sum_{k=0}^{N-1} p_k(\lambda_j(u))} \tag{12}$$

Since the sampling strategy should be independent of the reflectance spectra, we will make a simplifying assumption and set the reflectance spectra equal to a constant, in our case $R(\lambda) = 1$.

Given this simplification, (c_1, c_2, c_3) is simply the color of the illuminant spectrum in the XYZ color space. The new Monte Carlo estimate we need to compute is:

$$c_l \approx \sum_{j=0}^{N-1} \frac{h_l(\lambda_j(u)) I(\lambda_j(u))}{\sum_{k=0}^{N-1} p_k(\lambda_j(u))} \tag{13}$$

Ideally, the estimate dependence on u is perceptually as small as possible, so that the perceived variance is also minimal. Therefore, we define our objective function:

$$\int_0^1 \Delta_E \left(\left(\sum_{j=0}^{N-1} \frac{h_l(\lambda_j(u)) I(\lambda_j(u))}{\sum_{k=0}^{N-1} p_k(\lambda_j(u))} \right)^2_{l=0}, c_l \right) du \tag{14}$$

where Δ_E represents the perceived error, which we measure using the CIE Delta E standard (Sharma et al., 2005).

Starting with a set of initial sampling densities, we perform non-linear optimization to minimize the objective function. During rendering, these optimized PDFs are used to sample multiple wavelengths for a single path.

For our optimization step, it would be ideal if the initial sampling densities are as close to the true values as possible, so that the optimizer can converge faster. We decided to use the PDF:

$$q(\lambda) = \frac{I(\lambda)}{\int_\Lambda I(\lambda') d\lambda'} \tag{15}$$

If we set all densities equal to $q(\lambda)$, then their inverse CDFs are also identical. Thus, for a single random number u, all N wavelength samples will be the same, and our estimator evaluates a single point in the spectrum, making our final color monochromatic.

A more effective initialization strategy for the optimization is to define a unique density for each of the N wavelength samples, which avoids the problem of generating monochromatic results. To this end, we use the CDF of $q(\lambda)$ to partition the wavelength domain into N non-overlapping intervals,

$[a_j, b_j)$, which have equal probability mass. The endpoints for each interval are determined using the inverse CDF of the importance function:

$$a_j = F_q^{-1}\left(\frac{j}{N}\right) \quad b_j = F_q^{-1}\left(\frac{j+1}{N}\right) \tag{16}$$

With the domain partitioned, each of the N probability densities $p_j(\lambda)$ is constructed to be non-zero only within its assigned interval. Within that interval, its shape follows the original importance function $q(\lambda)$ but is re-normalized to ensure it is a valid PDF that integrates to one:

$$p_j(\lambda) = \begin{cases} \frac{q(\lambda)}{F_q(b_j) - F_q(a_j)} & \text{if } \lambda \in [a_j, b_j) \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

4 Implementation

To test the feasibility of our proposed methodology, we developed a practical implementation that integrates a pre-optimization step with a physically based spectral renderer. This section details the components and processes involved, covering rendering, optimization, and the strategies employed to manage computational complexity.

4.1 Rendering

For the rendering component of our work, we used PBRT-v4, a state-of-the-art, open-source spectral rendering system designed for physically based rendering research and education (Pharr et al., 2023).

Our implementation required adding additional functionality to the PBRT-v4 source code. A custom module was developed to load the optimized probability density functions (PDFs) from an external CSV file during the renderer's initialization phase, and a custom wavelength sampling logic was added to implement our proposed method. Furthermore, because rendering on the GPU is much more efficient, we decided to make our modified pipeline GPU-compatible.

Loading PDFs

In `spectrum.h`, PBRT defines `NSpectrumSamples`, the number of wavelengths sampled for each path, which is also equal to the number of distributions that we optimized. We decided to use four wavelength samples per path as this is the standard, and it is also what PBRT uses by default. We also define:

```
static constexpr int NumDistributionBins = 81;
static constexpr Float lambdaStep = 5.0;
```

which represent the total number of discrete samples we store for our distributions, and the step used to bin the wavelength domain into `NumDistributionBins` samples, respectively.

The initialization of the optimized sampling distributions is done in `sampling_distributions.cpp` and `sampling_distributions.h`. Here we define our sampling table:

```
struct WavelengthSamplingPDFs {
    Float *wavelengths;
    Float *pdfs_flat;
    Float *cdfs_flat;
    Float *mix_pdf;
};
extern WavelengthSamplingPDFs *globalSamplingTable;
```

We store the optimized PDFs in a flat array, their corresponding CDFs, and the mixture PDF, which we later use for MIS. We also store the discrete wavelengths at which we have our PDFs sampled, for which we used the range of 380 to 780 nanometers, sampled every 5 nanometer steps. We declare the function:

```
void InitializeWavelengthSamplingTable(const std::
    string &filename);
```

which contains all the logic for initializing the table. First, the optimized PDFs are read from the CSV file and normalized. Then the corresponding CDFs are computed, as well as the normalized mixture PDF. Finally, the table is constructed and loaded into the GPU memory.

Note that MIS (2.1) does not use the actual normalized mixture, but the sum of the individual PDFs. We do this because PBRT *averages* the samples when computing the Monte Carlo estimate, which is not correct when using standard MIS. Therefore, to counteract the effect of this averaging, we normalize the mixture, such that the final weights we provide to the estimator are correct and yield unbiased results.

To load these distributions at startup, we add an option `--wavelength-pdf` to the main PBRT script, which allows passing a csv file path which stores a set of optimized PDFs.

Wavelength Sampling

PBRT uses the class `SampledWavelengths`, defined in `spectrum.h`, to keep track of a set of sampled wavelengths along with their corresponding probabilities. Here we declare:

```
PBRT_CPU_GPU
static SampledWavelengths SampleFromCustom(const
    WavelengthSamplingPDFs *wavelengthPDFs, Float u
    );
```

which takes a pointer to the loaded sampling table, and a uniform random number u, and samples a set of `NSpectrumSamples` wavelengths, one from each of the optimized distributions, using the *inverse transform method*. Then it records the value of the mixture PDF at these wavelengths for later use in the Monte Carlo estimator.

The PBRT `WavefrontPathIntegrator` constructor was modified to accept a `WavelengthSamplingPDFs` object. If successfully loaded, this is then used during rendering in the `GenerateCameraRays` method in `camera.cpp` for sampling wavelengths:

```
SampledWavelengths lambda;
if (wavelengthPDFs != nullptr) {
    lambda = SampledWavelengths::SampleFromCustom(
        wavelengthPDFs, lu);
} else {
    lambda = film.SampleWavelengths(lu);
}
```

If the sampling distributions are not provided, PBRT falls back to using its standard uniform sampling method.

4.2 Optimization

The initial probability distributions are sampled every 5 nanometers within the wavelength range of 380 to 780 nanometers. These are provided to a non-linear optimizer as a flattened one-dimensional, which then iteratively adjusts these values to minimize the objective function defined in Equation 14.

Handling Complexity

The high dimensionality of the optimization problem, combined with the complexity of the objective function, presents significant computational challenges. To ensure efficiency and promote convergence, we adopted several key strategies.

First, to accelerate the optimization process, we utilized the JAX library (Bradbury et al., 2018). JAX is a high-performance numerical computing library that has automatic differentiation capabilities. Using JAX, we can automatically compute the gradient of our complex objective function with respect to the input distributions. Supplying these derivatives to a gradient-based optimizer significantly aids in navigating the high-dimensional parameter space, leading to more stability. Another significant advantage of using JAX is the GPU support it provides, enabling the computationally intensive objective function and gradient calculations to be offloaded to the GPU, which results in a substantial acceleration of the entire optimization process.

Furthermore, the optimization is subject to several constraints: each sampling function must represent a valid probability density function, meaning it must be non-negative and integrate to one over the wavelength domain. Directly enforcing these constraints, particularly the integral constraint, substantially increases the complexity of the optimization problem. Hence, we also experimented with relaxing the constraints of the problem. We tried enforcing only non-negativity and transforming the functions into valid probability distributions at the beginning of the objective function, by dividing each of the functions by its integral value:

$$p_i(\lambda) = \frac{f_i(\lambda)}{\int_\Lambda f(\bar{\lambda}) d\bar{\lambda}}, \tag{18}$$

where $f_i(\lambda)$ are the nonnegative functions that the optimizer passes to the objective function.

We also experimented with making the problem completely unconstrained, removing also the non-negativity bounds. To turn these into valid probability distributions, we applied a Softmax-like function, which transforms a vector $X = x_0, x_1, x_2, \ldots, x_{K-1}$ of real-valued inputs into a valid probability distribution over the wavelength domain:

$$p_i(\lambda_i) = \frac{e^{x_i}}{Z}, \quad Z = \sum_{i=0}^{K-1} e^{x_i} dx \tag{19}$$

Here Z is the integral approximation, calculated using a simple Riemann sum, and dx is the size of the sample step.

Optimization Strategies

We considered different optimization strategies, both local and global. For local, gradient-based optimization, we used the Python libraries SciPy and Ipopt. In SciPy we used the `scipy.optimize.minimize` function, which features multiple solvers suitable for high-dimensional optimization problems.

Ipopt, which stands for interior point optimizer, is another powerful open-source library for solving large-scale nonlinear optimization problems (Wächter & Biegler, 2005). We compiled Ipopt with the Coin-HSL package under the HSL Academic License, which provides a number of advanced solvers, available with an academic license (Science and Technology Facilities Council, 2024).

The success of gradient-based optimization depends greatly on the quality of the initial solution provided. Attempting to find a better initial solution, we used global optimization techniques as an initial phase in the optimization process to explore the solution space. The result from this phase was used as an initial solution for local optimization. We experimented with a basin hopping algorithm, available in SciPy, as well as a number of genetic algorithms available in the Python library evosax (Lange, 2022).

Unfortunately, none of the approaches using global optimization yielded satisfactory results, which is mostly attributed to the high dimensionality of the problem. The best results were obtained using local gradient-based optimization alone.

5 Evaluation

We evaluated the performance of our proposed method on a number of pre-made scenes obtained from Benedikt Bitterli's website (Bitterli, 2016). We used scenes either with a single emitter, or with a small number of emitters, all of them set to use the same emission spectrum.

Name	LSPDD idx.	Category
A Standard Illuminant	2659	Standard Illuminant
D65	2661	Standard Illuminant
Globe Twister	2482	CFL
Sun	2629	–
Scotopic	2639	Biological Sensitivity
Philips High Bay	2641	Metal Halide

Table 1: Emission spectra.

We selected a variety of illuminants with different SPDs from the *Lamp Spectral Power Distribution Database* (LSPDD) (Roby & Aubé, 2019) under the CC BY-NC-ND 4.0 license. The illuminants used are shown in Table 1 and Figure 2.

5.1 Evaluation Methods

We paired every scene with individual illuminant spectra and rendered ground truths, as well as lower-sample versions for each. Ground truth images were rendered at a very high sample count (e.g. 16k, 32k samples per pixel), using standard uniform sampling for the wavelengths. We rendered lower sample count versions (e.g. 512 samples per pixel) using the uniform sampling baseline method, as well as our custom method (Figure 3).

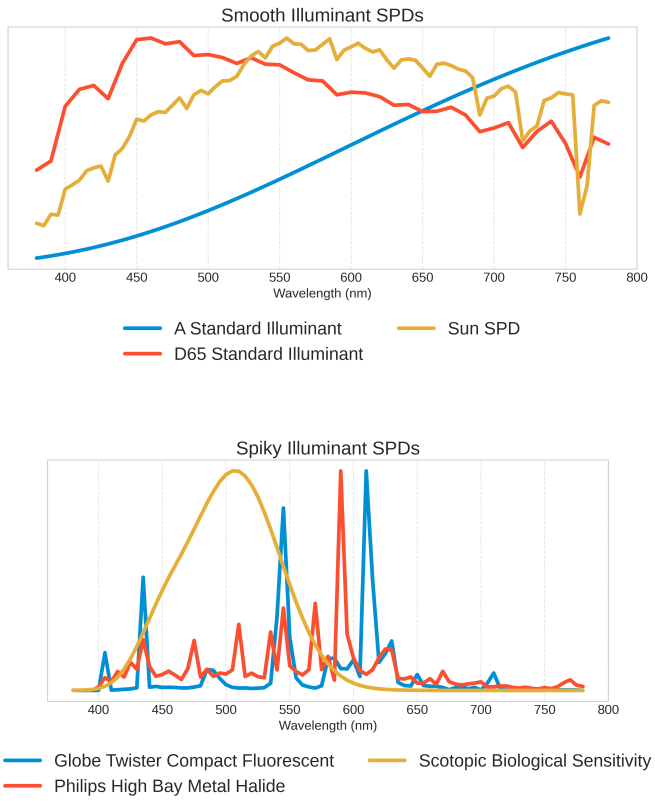

Figure 2: Spectral Power Distributions (SPDs) of selected light sources.

We then measured the average perceptual difference per pixel between the baseline method and the ground truth, and between our method and the ground truth, using the CIE Delta E standard (Sharma et al., 2005). The results of the evaluation are listed in Table 2.

Optimization

The optimized sampling distributions obtained are the result of a tailored optimization process. We applied various strategies to each illuminant spectrum, as explained in Section 4.2, and selected the one that yielded the best outcome. Although no single strategy was optimal for all cases, we determined that local gradient-based methods from SciPy and Ipopt were the most successful.

In addition to the raw value of the objective function, a useful way of visualizing the quality of the optimization is with a *color bar* (Figure 4), which plots the color the estimator computes for each densely sampled $u \in [0, 1]$. As mentioned in Section 3, the goal of the optimization is that the estimator's dependence on u is minimal. Therefore, the resulting color bar should be relatively constant across the domain, and its color should be the same as the target illuminant (Figure 7).

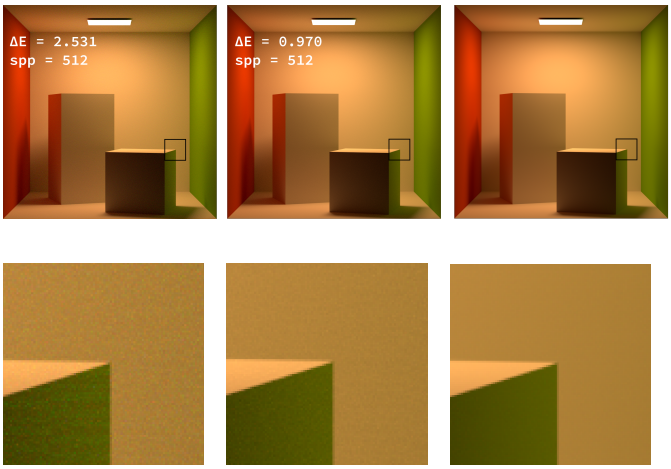

Figure 3: Comparison of baseline uniform sampling and our custom method, sampled at 512 spp, against the ground truth for the Cornell Box scene (Bitterli, 2016)., using a Globe Twister illuminant (LSPDD index 2723)

(a) Initial densities **(b)** Optimized densities

Figure 4: Comparison between (a) color bar of initial densities supplied to the optimizer, and (b) the color bar resulting from the optimized densities, for a Globe Twister illuminant (LSPDD Index 2723). The result was obtained using the IPOPT optimizer.

The optimization process did not yield perfectly smooth color bars for all the tested spectra, however. For example, for the D65 Illuminant, our optimized distributions result in high variance around the edges of the domain (Figure 6).

Figure 5: Comparison of our method against the baseline for the wooden staircase scene (Bitterli, 2016) using two different illuminants: D65 (left) and Globe Twister (right). Left image is the ground truth, middle image is rendered using our method, and the right image is rendered using the baseline method.

5.2 Results

Our evaluation, summarized in Table 2, indicates that our sampling strategy significantly outperforms the baseline strategy in scenes featuring complex emission spectra, such as the Philips High Bay or the Globe Twister emitters (Figure 2). For instance, in the Cornell Box scene lit by a Globe Twister fluorescent lamp, our method reduces the average perceptual error (ΔE) to 0.970, a substantial improvement over the baseline's 2.531. This is visually confirmed in Figure 3, where our result shows less color noise, as well as in the Staircase scene with the Philips High Bay illuminant (Figure 5), where the error at 1024 spp was nearly halved from 0.278 to 0.146.

In scenes with spectrally smooth illuminants, such as the sun or the D65 standard illuminant (Figure 2), our method achieves similar performance to the baseline. For example, in the Staircase scene with D65 illumination, the ΔE values are nearly identical for both methods at all sample rates (Table 2), such as 0.259 for our method versus 0.256 for the baseline at 1024 spp (Figure 5).

In terms of performance, our method introduces no significant overhead during the rendering phase. As shown in Table 2, render times were nearly identical to the baseline across all test cases. The only additional computational cost of our approach is the optimization step, which might take longer. However, this is a one-time, offline process performed once for each illuminant spectrum, and therefore does not impact final rendering performance. The memory overhead consists only of the sampling table that needs to be loaded into memory. In our case with four optimized PDFs sampled every five nanometers between 380 and 780 nanometers, this amount is negligible.

6 Discussion

In this section, we interpret our findings, discuss the limitations of our current approach, and outline potential directions for future work.

6.1 Interpretation of Results

The significant reduction in perceptual error for complex, "spiky" illuminants shows that our optimization process successfully identifies and prioritizes the narrow peaks in the illuminant's SPD, as opposed to uniform sampling, which might miss certain spectral features.

For smooth illuminants, the performance of our method was almost identical to the baseline. This is an expected and favorable outcome, as it shows that our optimization does not negatively impact performance in such scenarios. When the spectrum is already smooth, the optimal sampling distribution is inherently uniform-like, and our optimizer correctly identifies such a solution.

6.2 Limitations

Our approach optimizes a set of sampling distributions for a single, chosen illuminant spectrum. This is effective for scenes with a unified lighting environment but does not address scenarios with multiple light sources that have different SPDs. An approach to this can be optimizing distributions for all the different illuminant spectra in the scene. During rendering, wavelengths are sampled by selecting an illuminant at random, using its associated sampling distributions. However, such an approach does not scale well with the number of different illuminant spectra in the scene.

Furthermore, the optimization process is independent of scene geometry, assuming a constant reflectance spectrum of $R(\lambda) = 1$. This means that the sampling strategy does not account for the interaction between an illuminant and specific material reflectances. The final perceived color is a product of

both, and by ignoring reflectance, our optimization may miss important features.

Our implementation restricts wavelength sampling to 5-nanometer bins to maintain consistency with the optimized PDFs, but this approach may fail to capture narrow spectral features, either in emission or reflectance. An approach to resolve this would be to use interpolation in our custom sampling method.

Moreover, the quality of the initial distribution can have a great impact on the success of the optimization step. Our initial distributions (Equation 17) are a reasonably good baseline, but as can be seen in Figure 4, there is room for improvement.

6.3 Future Work

Our method can be extended by changing the optimization process to account for the reflectance spectrum, as currently our method treats it as constant. Future work could also explore more advanced initialization strategies for the initial distributions, to help the optimizer find better solutions more consistently.

Moreover, we did not find a single optimization solver that was universally optimal. While gradient-based methods from SciPy and Ipopt performed best, achieving a high-quality distribution for every spectrum sometimes required extensive experimentation with different solvers. This suggests that this part of our pipeline could be a focus of future work to improve robustness.

7 Responsible Research

To ensure our work can be reproduced and validated by other researchers, we explained our approach and its implementation in detail in Sections 3 and 4. In Section 6, we interpret our findings, discuss limitations, and demonstrate how our evaluation supports our conclusions. Our implementation will also be made available, along with the paper (Dobos, 2025).

Furthermore, the software used during the research process, such as PBRT-v4, SciPy, and Ipopt are fully open-source and publicly available. The only exception to this are the advanced Coin-HSL solvers that we used for our optimization in Ipopt, which are under an academic license that must be requested from the STFC (Science and Technology Facilities Council, 2024).

The data used constitutes of the illuminant spectra we selected from the Lamp Spectral Power Distribution Database (Roby & Aubé, 2019), under the CC BY-NC-ND 4.0 license. Additionally, the scenes used are made freely available by Benedikt Bitterli on his personal website (Bitterli, 2016).

8 Conclusion

In this paper, we introduced a preprocessing step for reducing perceived variance in single-emitter scenes, that optimizes wavelength sampling distributions based on perceptual error. Our evaluation demonstrates that this method can significantly reduce perceptual error compared to uniform sampling, particularly for illuminants with complex spectral power distributions, while incurring no computational overhead during the final render.

References

Borges, C. F. (1991). Trichromatic approximation for computer graphics illumination models. *SIGGRAPH Comput. Graph.*, *25*(4), 101–104. https://doi.org/10.1145/127719.122729

Wilkie, A., Nawaz, S., Droske, M., Weidlich, A., & Hanika, J. Hero wavelength spectral sampling. In: In *Proceedings of the 25th eurographics symposium on rendering*. EGSR '14. Lyon, France: Eurographics Association, 2014, 123–131. https://doi.org/10.1111/cgf.12419

Evans, G. F., & McCool, M. D. Stratified wavelength clusters for efficient spectral monte carlo rendering. In: In *Proceedings of the graphics interface 1999 conference, june 2-4, 1999, kingston, ontario, canada*. 1999, 42–49. http://graphicsinterface.org/wp-content/uploads/gi1999-7.pdf

van de Ruit, M., & Eisemann, E. (2021). A multi-pass method for accelerated spectral sampling. *Computer Graphics Forum*, *40*, 141–148. https://doi.org/10.1111/cgf.14408

Radziszewski, M., Boryczko, K., & Alda, W. (2009). An improved technique for full spectral rendering. *Journal of WSCG*, *17*.

Sharma, G., Wu, W., & Dalal, E. N. (2005). The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, *30*(1), 21–30. https://doi.org/10.1002/col.20070

Pharr, M., Jakob, W., & Humphreys, G. (2023). *Physically based rendering: From theory to implementation* (4th). MIT Press. https://pbrt.org

Kajiya, J. T. The rendering equation. In: In *Proceedings of the 13th annual conference on computer graphics and interactive techniques*. SIGGRAPH '86. New York, NY, USA: Association for Computing Machinery, 1986, 143–150. ISBN: 0897911962. https://doi.org/10.1145/15922.15902

Veach, E. (1998). *Robust monte carlo methods for light transport simulation* [Doctoral dissertation] [AAI9837162]. Stanford University.

CIE. (1932). *Proceedings of the commission internationale de l'éclairage, 8th session* [Standard adopted in 1931]. Cambridge University Press.

West, R., Georgiev, I., Gruson, A., & Hachisuka, T. (2020). Continuous multiple importance sampling. *ACM Trans. Graph.*, *39*(4). https://doi.org/10.1145/3386569.3392436

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/jax-ml/jax

Wächter, A., & Biegler, L. T. (2005). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, *106*(1), 25–57. https://doi.org/10.1007/s10107-004-0559-y

Science and Technology Facilities Council. (2024). *"HSL, a collection of fortran codes for large-scale scientific computation"* (Version 2024.05.15). https://www.hsl.rl.ac.uk

Lange, R. T. (2022). Evosax: Jax-based evolution strategies. *arXiv preprint arXiv:2212.04180*.

Bitterli, B. (2016). Rendering resources [https://benedikt-bitterli.me/resources/].

Roby, J., & Aubé, M. (2019). *Lamp spectral power distribution database (lspdd)* [Accessed 2025-06-21]. https://lspdd.org/

Dobos, C.-C. (2025). Perception-based Optimization of Wavelength Sampling Distributions for Spectral Rendering. https://github.com/cristidobos/Perception-based-Spectral-Rendering

Appendix

Figure 6: Color bars obtained for the D65 Standard Illuminant (LSPDD index 2661) and the Globe Twister (LSPDD index 2482)

Figure 7: Color bars obtained for the A Standard Illuminant (LSPDD index 2659) and the Sun (LSPDD index 2629)

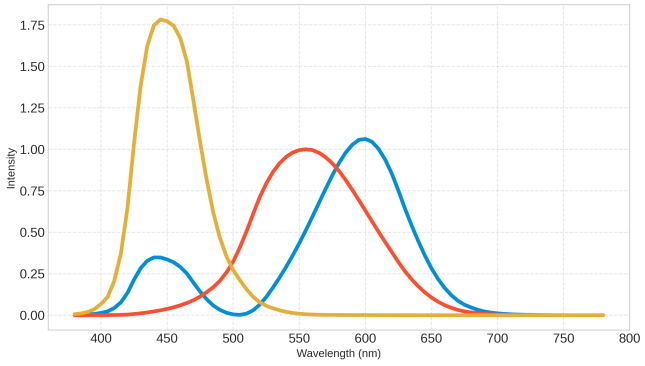

Figure 8: $\bar{x}, \bar{y}, \bar{z}$ color matching functions

Scene	Illuminant	SPP	Theirs		Ours	
			Avg. ΔE	Time (s)	Avg. ΔE	Time (s)
Cornell Box	Sun	512	0.6250	17.94	**0.5765**	17.73
	Compact Fluorescent	512	2.5311	17.75	**0.9703**	17.14
	Scotopic Vision	512	0.6190	17.93	**0.4529**	17.34
Glass of Water	CIE Illuminant A	256	2.7038	9.95	**2.4082**	9.76
	CIE Illuminant A	512	1.9520	19.18	**1.7280**	19.37
	CIE Illuminant D65	256	2.3054	9.96	2.3918	9.97
	CIE Illuminant D65	512	1.6659	19.56	1.7854	19.40
	Compact Fluorescent	256	7.2244	9.78	**2.8654**	9.57
	Compact Fluorescent	512	5.6761	19.57	**2.2336**	19.17
Staircase	CIE Illuminant D65	256	0.4807	13.36	0.4809	12.98
	CIE Illuminant D65	512	0.3525	26.22	0.3546	26.19
	CIE Illuminant D65	1024	0.2565	53.21	0.2590	51.81
	Philips High Bay	256	0.5126	13.36	**0.2524**	12.77
	Philips High Bay	512	0.3791	26.58	**0.1908**	25.57
	Philips High Bay	1024	0.2781	52.58	**0.1462**	50.67
Teapot	Compact Fluorescent	256	3.0135	7.78	**0.9685**	7.76
	Compact Fluorescent	512	1.9999	15.39	**0.6967**	15.36
	Sun	256	1.0464	7.78	**0.8340**	7.78
	Sun	512	0.7526	15.37	**0.6121**	15.39
Veach Bidir	Compact Fluorescent	512	2.8588	10.76	**0.8241**	10.75
	Compact Fluorescent	2048	1.5811	42.38	**0.4925**	42.16
Veach MIS	CIE Illuminant A	256	0.7454	4.26	**0.7141**	4.34
	CIE Illuminant A	512	0.4429	8.56	**0.4127**	8.55
	Philips High Bay	256	1.1687	4.26	**0.6046**	4.25
	Philips High Bay	512	0.8455	8.55	**0.3580**	8.56
	Sun	256	0.6128	4.34	**0.5620**	4.44
	Sun	512	0.3537	8.55	**0.3207**	8.56

Table 2: Comparison of Perceptual Difference (ΔE) and Render Times Across Various Scenes and Illuminants between stratified uniform sampling and our custom method.