Rendering Dispersion

DANIEL KOPP, Rensselaer Polytechnic Institute, USA

This report presents our work on implementing full spectrum photon mapping, which models the spectral distribution of light in a scene. Our modifications to the basic RGB photon mapping algorithm include a representation of photons that includes wavelength and energy values, and a method for calculating the spectral reflectance curve of a material from its RGB albedo value. Our approach enables the simulation of complex light behavior, such as dispersion and spectral caustics. Results from our experiments demonstrate the effectiveness of our approach.

CCS Concepts: • **Computing methodologies → Ray tracing**.

Additional Key Words and Phrases: photon mapping, full spectrum, physically based

ACM Reference Format:
Daniel Kopp. 2023. Rendering Dispersion. *ACM Trans. Graph.* 37, 4, Article 111 (April 2023), 6 pages. https://doi.org/na

1 INTRODUCTION

Rendering realistic images that accurately capture the behavior of light in the real world is a fundamental challenge in computer graphics. One aspect of light behavior is its spectral dynamics, the behavior of light dependent on the rays wavelength. To achieve realistic renderings for scenes containing dispersive elements, it is important to model the spectral distribution of light and its interactions with the materials in the scene in a physically based method.

Photon mapping is a popular algorithm for simulating caustics and indrect lighting in a scene, which has been widely used in computer graphics to generate realistic images. In our homework 3 implementation, photon mapping used RGB values to represent the energy of light, which limited its ability to accurately model spectral dynamics of light.

We begin by reviewing related work in the field of photon mapping and discussing the limitations of the basic RGB photon mapping algorithm. We then provide an overview of the modifications we made to the algorithm to enable full spectrum capabilities, including a detailed explanation of the photon representation, RGB radiance estimation, the calculation of the spectral reflectance curve and dispersion physics. We also discuss the advantages, challenges, and limitations of our implementation, and present results from our experiments that demonstrate the effectiveness of our approach.

Author's address: Daniel Kopp, Rensselaer Polytechnic Institute, Troy, USA, koppd@rpi.edu.

0730-0301/2023/4-ART111 $15.00
https://doi.org/na

2 PRIOR WORK

In [Evans and McCool 1999], the authors used stratified sampling of the wavelength, λ, and applied quasi-Monte Carlo bidirectional path tracer to achieve full spectrum rendering (focused on spectral caustics) at a convergence of $\frac{1}{N}$ rather than $\frac{1}{\sqrt{N}}$ where N is the number of samples. In their work each ray samples an independent λ to calculate the spectral radiance of each pixel.

The next work we looked at was the technical report for a industry grade rendering engine for use in movies [Fascione et al. 2018]. The implementation the authors used for the engine was to sample the full spectrum of light with each ray to get a full radiance sample, this is likely faster than the previous method but comes with the drawback of being memory intensive as each ray stores the light intensity at each wavelength .

In the final work we found, [Tandianus et al. 2015] finds optimizations for full spectrum rendering with photon mapping. Each photon was sampled with an individual wavelength and energy. They also used clustering methods to combine some wavelengths to save time computing ray reflection logic.

3 IMPLEMENTATION

We chose to use the photon mapping as Tandianus; we did not implement the clustering optimization as the implementation would take too long.

3.1 Photon Representation

To implement full spectrum photon mapping, it was necessary to adjust the representation of photons used in the RGB photon mapping algorithm. In our previous work on homework 3 (RGB Ray Tracing and RGB Photon Mapping), photons were represented as energy in the RGB color space, with each component ranging between 0 and 1. However, for full spectrum photon mapping, photons need to be represented as having both a wavelength and an energy. To achieve this, we converted the energy of each photon from RGB color space to a spectral representation. Specifically, each photon is now assigned a wavelength value λ and an energy value, enabling us to track the full spectrum of light energy as it interacts with objects in the scene. This modification to the photon representation was essential for accurately rendering dispersion and spectral caustics on objects.

3.2 Photon Behavior

3.2.1 Photon Gathering. The first modification we need to make to the photon behavior is to account for the fact that we have different wavelengths. In RGB photon mapping, we collect N photons and take the sum of their energies divided by the area containing the photons. For full spectrum photon mapping, we will have to modify this behavior to gather the light energy distribution across different wavelengths. Additionally, since this component will be integrated into an RGB path tracer, we will have to ensure that the final output is in RGB values.

To accomplish this, we first need to convert the spectral power distribution (SPD) of the photons into tristimulus values in the XYZ color space. These tristimulus values represent the amount of energy in each of the three color channels required to create a particular color. From there, we apply a 3x3 matrix transform to these values to convert them from the XYZ color space to the RGB color space.

An SPD is a function that describes how the energy of light is distributed across different wavelengths. We can represent this function as $E(\lambda)$, where λ is the wavelength and $E(\lambda)$ is the energy at that wavelength. In order to calculate the XYZ color values, we use the CIE Color Matching Function (CMF) [UCL 2006], which describes the sensitivity of a viewer to a color channel at a certain wavelength. We can represent this sensitivity as $S(\lambda)$, where $S : \mathbb{R} \rightarrow \mathbb{R}^3$.

To calculate the tristimulus values, we can compute the following integral:

$$C_{\text{XYZ}} = \int_{\Lambda_{\min}}^{\Lambda_{\max}} S(\lambda)E(\lambda)d\lambda. \tag{1}$$

However, since both $E(\lambda)$ and $S(\lambda)$ are discrete in our implementation, we use Monte Carlo integration to approximate the integral. Specifically, we can use the following equation to estimate the tristimulus values:

$$C_{\text{XYZ}} = \frac{1}{N}\sum_i \frac{S[\lambda_i]e_i}{P[\lambda = \lambda_i]} \tag{2}$$

Since the CIE-CMF is a discrete function we can only use values of lambda 380-7280 nanometers at a step size of 5 nanometers. We randomly sample on that distribution giving us $P[\lambda = \lambda_i] = \frac{1}{81}$ and simplifying our equation to:

$$C_{\text{XYZ}} = \frac{81}{N}\sum_i S[\lambda_i]e_i \tag{3}$$

where N is the total number of photons, $S[\lambda_i]$ is the color matching function for the wavelength of photon i, and e_i is the energy of photon i at its respective wavelength.

In order to incorporate this calculation into photon mapping, we will have to divide the tristimulus values by the area of the tightest circle containing the photons. This is because the photons should lie on approximately a two-dimensional surface.

This modified calculation will allow us to accurately represent the full range of colors visible to the human eye, including the effects of dispersion and spectral caustics, within the framework of an RGB path tracer.

3.2.2 Photon Reflectance. In order to find the reflectance of a photon, we need to have a spectral reflectance curve that describes how much light of each wavelength is reflected by a surface. We want this curve to be consistent with the RGB path tracer, meaning that it should be derived from a single RGB triplet. We can then use this curve to calculate the amount of reflected light for each wavelength, which will be used in the photon mapping algorithm.

We would like our spectral reflectance curve to bounce the same amount of energy at each color to ensure this consistency. For example the triplet (1,1,1) should bounce full energy, (0,0,0) should bounce no energy (1,0,0) should bounce only 'red' wavelengths.

We can write a formalized version of this constraint as:

$$\begin{aligned} C_{\text{RGB spec}} &= C_{\text{RGB raw}} \\ &= T_{\text{XYZ}\rightarrow\text{RGB}} \int_{\Lambda_{\min}}^{\Lambda_{\max}} S(\lambda)E(\lambda)\rho(\lambda)d\lambda \\ &= L \cdot M_{\text{RGB}} \end{aligned}$$

where $\rho(\lambda)$ is the spectral reflectance curve, $E(\lambda)$ is the SPD of the incoming light, $S(\lambda)$ is the color matching function, $T_{\text{XYZ}\rightarrow\text{RGB}}$ is the transformation matrix from XYZ to RGB color space, $C_{\text{RGB spec}}$ is the RGB values for the reflected light for the full spectrum of wavelengths, $C_{\text{RGB raw}}$ is the RGB value for the reflected light for the RGB path tracer, L is the intensity of the incoming light, and M_{RGB} is the RGB albedo value for a given material.

Scott Burns developed a method for using three curves, $P_{Nx3}(\lambda)$, to create an arbitrary spectral reflectance curve given an RGB triplet [Burns 2020b] [Burns 2020a] that approximately satisfies the above condition. His method can be written as follows:

$$\rho(\lambda, M_{\text{RGB}}) = M_{\text{RGB}} \cdot P_{Nx3}(\lambda) \tag{4}$$

where $P_{Nx3}(\lambda)$ is a matrix that contains three discretized curves, one for each RGB color channel, and M_{RGB} is the RGB albedo value for a given material. The spectral reflectance curve can then be calculated as a simple weighted sum of the curves according to the RGB albedo value. This method allows us to create a spectral reflectance curve that is consistent with the RGB path tracer and can be used to accurately simulate the reflection of light in our photon mapping algorithm. P_{Nx3} is a discrete curve as it is the solution to a complex non-linear program:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} \left(z^r_{i+1} - z^r_i\right)^2 + \left(z^g_{i+1} - z^g_i\right)^2 + \left(z^b_{i+1} - z^b_i\right)^2 \\ \text{s.t. } & Te^{z^r} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \\ & Te^{z^g} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \\ & Te^{z^b} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \\ & e^{z^r} + e^{z^g} + e^{z^b} \leq 1 \end{aligned}$$

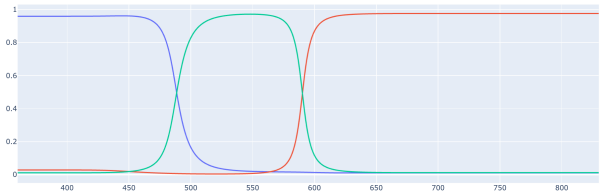

Fig. 1. P_{81x3} and solution to non-linear problem

Where $z^r = \ln P_r$, $z^g = \ln P_g$ and $z^b = \ln P_b$ and T is a Nx3 matrix that transforms a full spectrum reflectance curve to an RGB albedo triplet. Since P is a matrix that allows us to construct any reflectance

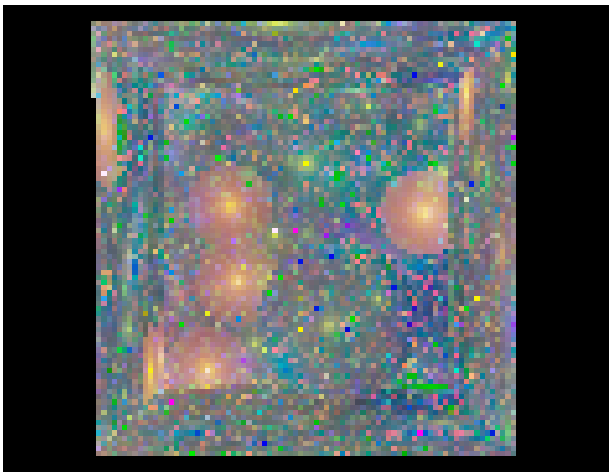

Fig. 2. Indirect lighting of Cornell Box when CIE-CMF CSV was accidentally swapped for the P function CSV, certain reflectance values with this CSV leading to certain very bright outliers

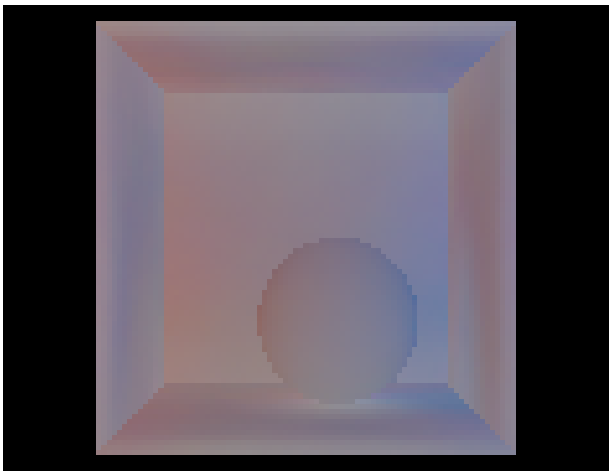

Fig. 3. Correct indirect lighting from Cornell Diffuse Box scene using full spectrum photon mapping to verify the spectral reflectance curve and gathering modification implementations work

curve with only a dot product operation we only have to solve the non-linear program once. We solved this with `scipy.optimize` in python then saved the matrix as a CSV which was loaded in C++ as an `std::unordered_map<float, Vec3f>` for constant time reflections (with respect to wavelength samples).

3.2.3 Dispersion Physics. Dispersion occurs when light passes through a medium with varying refractive indices, causing the different wavelengths of light to bend at different angles according to Snell's Law. This results in a separation of the colors of the light, creating a rainbow effect.

First we will recall Snell's Law:

$$n_i \sin\theta_i = n_o \sin\theta_o \rightarrow \sin\theta_o = \frac{n_i}{n_o} \sin\theta_i \tag{5}$$

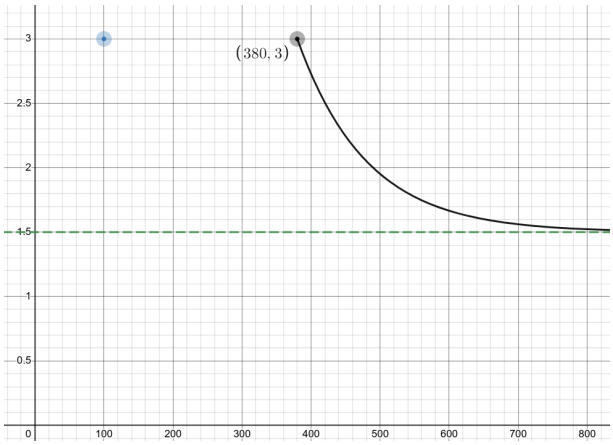

Fig. 4. Approximation of general IOR function: click for interactive graph

Where n is the index of refraction, θ is the angle between a ray and the normal, i is the incoming ray, o is the outgoing ray. For the purpose of easy vector algebra we can represent as:

$$R \cdot N = \sqrt{1 - (\frac{n_i}{n_o}\sqrt{1 - (N \cdot V)^2})^2} \tag{6}$$

Where R is the reflected ray, N is the normal, V is the incoming ray. We can use this to simulate dispersion as dispersion occurs when the IOR is wavelength dependent, or $\texttt{IOR}(\lambda) \neq c$. To find a general purpose $\texttt{IOR}(\lambda)$, we looked at physical dispersion curves and found that many are similar to an exponential decay function, which means we can use the following approximation (shown in figure 4):

$$\texttt{IOR}(\lambda) = (\texttt{ior\_1} - \texttt{ior\_2})\, e^{-\frac{1}{\texttt{spread}}(\lambda - \Lambda_{\min})} + \texttt{ior\_2} \tag{7}$$

Where `ior_1` is the IOR at the minimum sampled λ `ior_2` is the IOR of the asymptote or the IOR as $\lambda \rightarrow \infty$ and `spread` is an arbitrary constant between 0 and ∞ that controls the rate of decay.

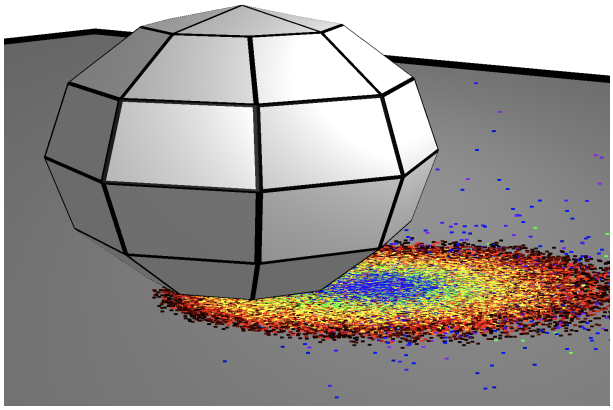

Fig. 5. Dispersion with exaggerated IOR function

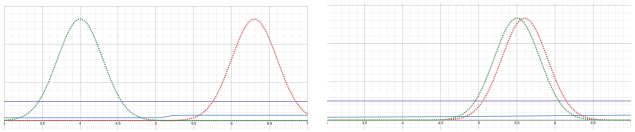

Fig. 6. Initial spectral reflectance curve test: purple is the source SPD, blue shows the reflected SPD, sensitivity to red, green shown in respective colors; Left: working approximation, Right: failing approximation (ratio of predicted to real red reflectance is 1.5); for better resolution / interactivity click here

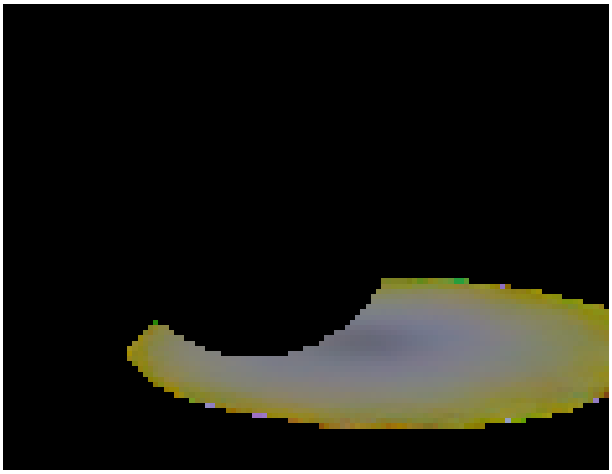

Fig. 7. Indirect radiance estimate for dispersion without cone filtering, same scene and photons as figure 5

4 DISCUSSION

4.1 Challenges

Over the course of this work we encountered many issues that were critical to the project but also pushed our understanding of the methods required to achieve the desired result.

Once such challenge was our initial method of spectral reflectance curve construction. The initial attempt was to approximate a solution to the condition outlined in section 3.2.2, by taking the weighed sum of the sensitivities and taking the norm at that wavelength such that:

$$\rho(\lambda) = \frac{S(\lambda)^T C_{\text{RGB}}}{\sqrt{S(\lambda)^T S(\lambda)}} \tag{8}$$

We tested this first in Desmos (figure 6, an online graphing calculator), the initial results looked correct and verified the stated condition. We then moved to a simulation in Python where we had an array of photons and applied $e' = \rho(\lambda)e$ to each but got odd results, the SPD of the reflected photons was not aligning with the RGB albedo value. After looking back at the Desmos graph, we realized that the simulated sensitivities distribution were mutually exclusive. If we moved the sensitivities to overlap each other, the method we created failed; the calculated reflectance values did not match the predicted.

The other large issue we encountered was fuzzy edges on the caustic leading to unclear dispersion as can be seen in figure 7. This was caused by photons far the sampled point having a large contribution to the radiance estimate. This was made clear when we implemented cone filtering as documented [Jensen 2010] which applies a weight $w_p = 1 - \frac{d_p}{kr}$, where k is a constant, d_p is the distance of the photon to the sampled point and r is the distance of the farthest photon in the grouping. We multiply each photon's flux by this weight and adjust the final sum by dividing it by $1 - \frac{2}{3k}$ to normalize the sum. This removed the issue of the fuzzy caustics.

4.2 Advantages

This method is faster than using full spectrum photon mapping in addition to full spectrum ray / path tracing. We can use the traditional RGB rendering techniques while estimating the full spectrum radiance dynamics using photon mapping cutting down the cost of adding an additional dimension to the rendering equation's integral.

Our method also allows for physically accurate / physically inspired phenomena. With the exception of the spectral reflectance curve estimation, each step was inspired by the natural world and the physical laws that governs it.

4.3 Disadvantages

While this method is faster than full spectrum photon mapping and full spectrum ray / path tracing, it still requires a large amount of photons to render visible dispersive effects. Most scenes we rendered, shot at least 1,000,000 photons with 500 collected photons to get clear results. It is clear that we need to implement more optimizations in order to render more complex scenes.

Additionally, the drawbacks of only using photon mapping for spectral dynamics is that we lose the ability to render chromatic aberration, the phenomenon where light received though a dispersive medium is scattered, as well as the ability to render rainbows in participating media.

5 RESULTS

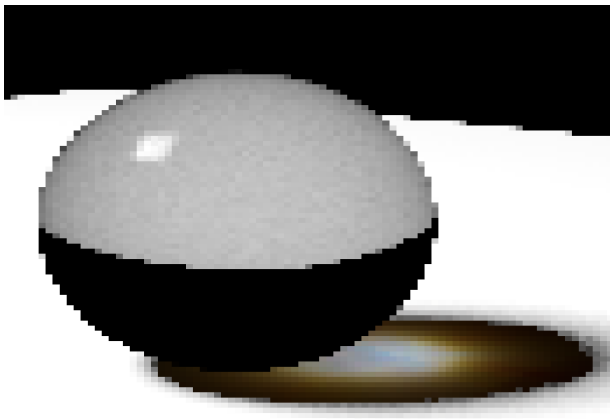

Fig. 8. Glass sphere on plane (1,000,000 photons; 100 gathered; 1 hour)

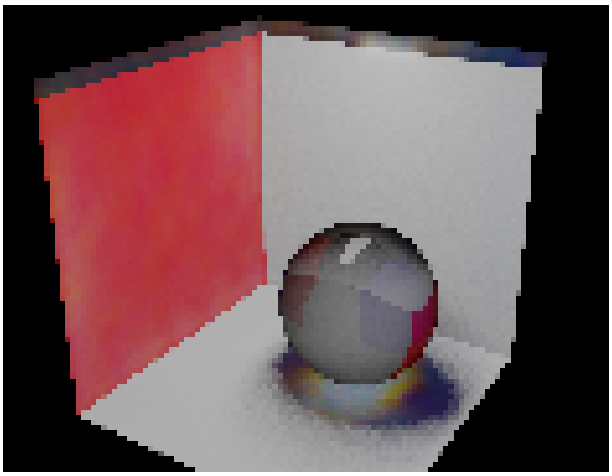

Fig. 9. Cornell Box with glass sphere inside (10,000,000 photons; 500 gathered; 5 hours)

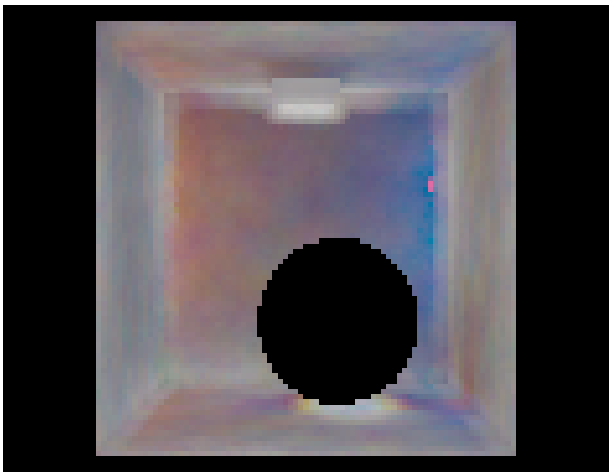

Fig. 10. Indirect radiance estimate from figure 9 (light source was moved to a small area above the sphere instead of in front)

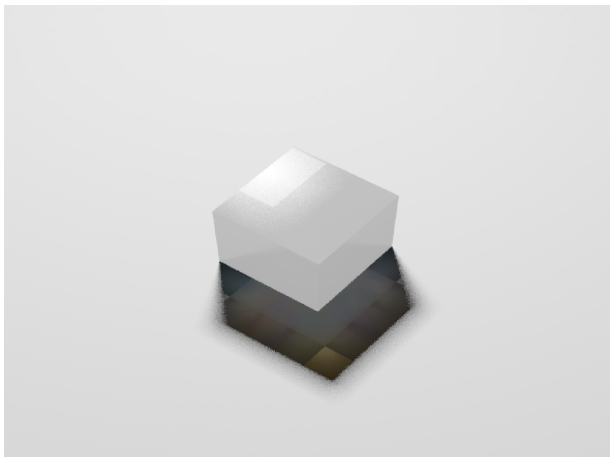

Fig. 11. Glass Box on plane (1,000,000 photons; 500 gathered; 2 hours). This render did not come out well as in order to render in a reasonable amount of time I had to turn up the Russian Roulette continue probability to 0.8 meaning that photon / ray depth is limited to a small depth

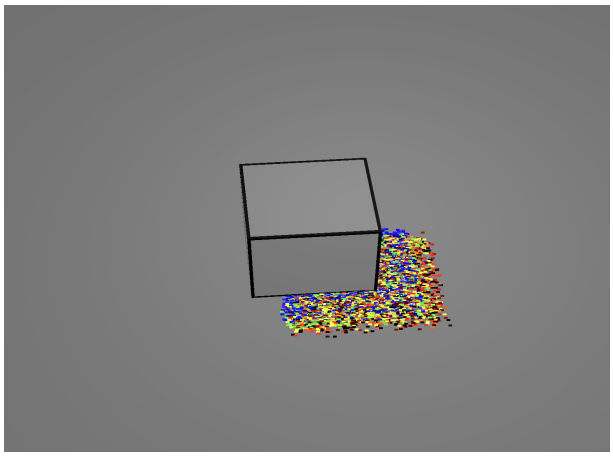

Fig. 12. Photons shot in figure 11

6 CONCLUSION

In this report, we presented our work on implementing full spectrum photon mapping, which allows for accurate modeling of the spectral distribution of light in a scene. Our modifications to the basic RGB photon mapping algorithm included introducing a photon representation that includes wavelength and energy values, the ability to get RGB values from full spectrum photon mapping and a method for calculating the spectral reflectance curve of a material from its RGB albedo value. Our approach enables the simulation of complex light behavior, including the effects of dispersion and spectral caustics.

In conclusion, full spectrum photon mapping is an important extension of the basic RGB photon mapping algorithm for spectral caustics, and our work demonstrates its effectiveness in accurately modeling the spectral distribution of light in a scene.

REFERENCES

Scott A. Burns. 2020a. *Fast RGB to Spectrum Conversion for Reflectances.* http://scottburns.us/fast-rgb-to-spectrum-conversion-for-reflectances/

Scott A. Burns. 2020b. Numerical methods for smoothest reflectance reconstruction. *Color Research & Application* 45, 1 (2020), 8–21. https://doi.org/10.1002/col.22437 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/col.22437

Glenn F. Evans and Micheal D. McCool. 1999. Stratified Wavelength Clusters for Efficient Spectral Monte Carlo Rendering. In *Proceedings of the 1999 Conference on Graphics Interface '99* (Kingston, Ontario, Canada). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 42–49.

Luca Fascione, Johannes Hanika, Mark Leone, Marc Droske, Jorge Schwarzhaupt, Tomáš Davidovič, Andrea Weidlich, and Johannes Meng. 2018. Manuka: A Batch-Shading Architecture for Spectral Path Tracing in Movie Production. *ACM Trans. Graph.* 37, 3, Article 31 (aug 2018), 18 pages. https://doi.org/10.1145/3182161

Herik W. Jensen. 2010. *CSE272: Advanced Image Synthesis.* www.cs.princeton.edu/courses/archive/fall16/cos526/lectures/03-photonmapping.pdf

Budianto Tandianus, Henry Johan, Hock Soon Seah, and Feng Lin. 2015. Spectral caustic rendering of a homogeneous caustic object based on wavelength clustering and eye sensitivity. *Vis. Comput.* 31, 12 (Dec. 2015), 1601–1614.

UCL. 2006. *CIE Colour Matching Functions.* http://cvrl.ucl.ac.uk/cmfs.htm

Received 24 April 2023