Spectral Ray Tracing

Dawson Do
daws@berkeley.edu

Joshua Fernandes
jbfern@berkeley.edu

Jeremy Chen
jcmchen@berkeley.edu

Jingyi Guo
guojingyi518@berkeley.edu

Figure 1: Renders showcasing spectral rendering of light dispersion, thin-film iridescence, and microfacet materials.

ABSTRACT

In this project, we write a ray tracer that includes spectral information about rays within the light field. A traditional ray tracer tracks RGB intensities along the ray and propagates the ray using RGB information about scene elements. In reality, light is distributed by wavelength in a continuous space, with visible light occupying the range approximately from 400 nm to 700 nm. Optical phenomena arise from wavelength-dependent interactions, including dispersion, iridescence, and reflection probability. In this project, we refactor the ray tracer we wrote in Homework 3 to incorporate wavelength dependence by adding a layer of wavelength sampling. We also implement scene elements that leverage this feature, including glass materials, mirror surfaces, microfacet surfaces, thin films, and black body emitters. With these, we can create striking photorealistic scenes accessible only through spectral information.

CS 284A: Final Project, Spring 2024, UC Berkeley, CA, USA

CCS CONCEPTS

• **Computing methodologies** → **Ray tracing**; Reflectance modeling; *Visibility*.

KEYWORDS

Spectral ray tracing, Dispersion, Iridescence, Thin-Film Interference

1 INTRODUCTION

In an RGB ray tracer, the spectrum of light is tracked as a three-vector corresponding to red, green, and blue intensity. In reality, light is an infinite-dimensional spectrum of wavelengths. A more physical rendering equation would consider the entire spectrum of light emitted and consider absorption, reflection, and refraction of the spectrum. A spectral ray tracer keeps track of the wavelength and the intensity of each. This is an important problem to capture dispersion of light, the phenomenon that the index of refraction is wavelength-dependent. This is what leads to a prism splitting light into the constitutive components. It is a challenging problem since we change a three-dimensional intensity into an infinite-dimensional one. The solution for addressing this challenge is to sample from the spectrum of wavelengths. Then, after collecting all the rays, we then can convert the distribution of wavelengths to red, green, and blue intensities for rendering on a computer screen.

Past projects implement some of the following features, such as iridescence and black body radiation [4, 5]. Our work is inspired by these previous works and seeks to extend these features to an entire spectral-based ray tracer rather than sampling wavelengths at the BSDF level. In addition, we create a mechanism for general implementation of any wavelength-dependent material or light source.

2 FROM RGB TO SPECTRAL RAY TRACER

2.1 Spectral Rays

To refactor the ray tracer for spectral information, we added an inner loop of sampling in `raytrace_pixel()`. For each ray, we also sample wavelengths and store this information in the ray object, which is passed into `BSDF::sample_f()`. The rendering equation is unchanged, but now includes a wavelength dependence, as

$$L_o(p, \lambda, \omega_o) = L_e(p, \lambda, \omega_o) + \int_{H^2} f_r(p, \lambda, \omega_i \to \omega_o) L_o(\mathrm{tr}(p, \omega_i), \lambda, -\omega_i) \cos\theta_i d\omega_i \,, \quad (1)$$

where the light intensities are now scalar intensities (rather than RGB vectors). In the implementation, each BSDF now returns a scalar quantity representing spectral intensity, which is summed to estimate the global illumination. We will discuss the changes made to the old non-spectral BSDFs in Section 2.2 and the implementation of spectral BSDFs in Section 3.

We can think of the intensity parameterized by wavelength as an infinite-dimensional vector, where each wavelength has an associated intensity. We can downsample to RGB values using the spectral response functions as

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} \int S_R(\lambda) L(\lambda)\, d\lambda \\ \int S_G(\lambda) L(\lambda)\, d\lambda \\ \int S_B(\lambda) L(\lambda)\, d\lambda \end{bmatrix} , \quad (2)$$

where S_i is the spectral response function for color i and $L(\lambda)$ is the intensity as a function of wavelength. Here, we use an analytical approximation to the CIE 1931 color matching functions, presented in Equation 4 of [10] to convert to XYZ and compose this with the CIE 1931 conversion from XYZ to RGB [8].

We compute this integral by sampling uniformly in the wavelength space. One option is to take independent random samples, but we instead use *hero sampling*, which chooses the first sample uniformly at random and places the remaining $n - 1$ samples at equispaced positions relative to the first sample. This method is unbiased and is empirically shown to reduce the color noise [9]. In our implementation, we find using 5 samples sufficient for convergence.

2.2 Spectral Upsampling

The RGB BSDFs from HW 3-1 and 3-2 use a stored RGB albedo to return the color of the surface, but after the above modifications, they need to return the spectral intensity of the corresponding wavelength that hit the surface. To accomplish this, we implement spectral upsampling using the spectral primaries basis, $\{s_r, s_g, s_b\}$, computed by Mallet and Yuksel [6], which we store as 81 reflectivity values corresponding to equidistant wavelengths between 380 and 780 nm. Given an RGB albedo, we compute the spectral intensity at the ray's wavelength using linear interpolation of the stored basis values.

$$s(\lambda) = R \cdot s_r(\lambda) + G \cdot s_g(\lambda) + B \cdot s_b(\lambda) \quad (3)$$

For every BSDF that originally returns an RGB, we convert that to an intensity using Eq. 3.

Following these modifications, we can already see differences in our renders, even though we haven't introduced wavelength-dependent interactions. Figure 2 is an example render using the existing scene files with no wavelength-dependent light transport, shown using the RGB ray tracer and the spectral sampling method. Both should be the same in expectation, but we see that the nature of the noise changes. Colorful noise is introduced in the spectral ray tracer due to randomness in sampling wavelengths.

3 SPECTRAL BSDF

To prepare the code for the glass prism and other spectral materials, reflective, refractive, and microfacet materials were implemented according to the specifications of HW 3-2. The next steps were to make wavelength dependent versions of these functions.

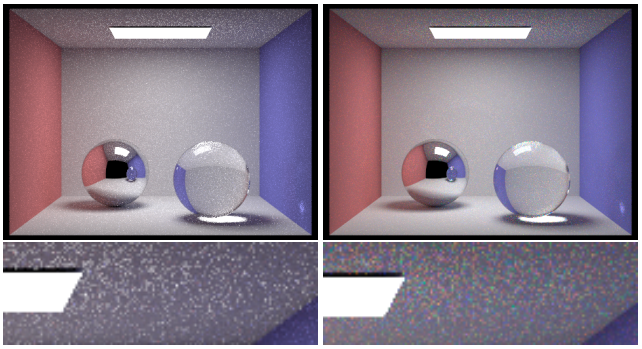

Figure 2: On the left, RGB ray tracer render. On the right, spectral sampling ray tracer render. Below shows magnifications of the observed noise (grayscale in RGB rendering, colorful in spectral sampling).

3.1 Dispersive Glass

We make a simple modification to GlassBSDF so that the index of refraction is a function of the wavelength, given by Cauchy's approximation:

$$\eta(\lambda) = A + \frac{B}{\lambda^2}, \tag{4}$$

where A and B are parameters [3]. In addition, we use the dielectric Fresnel reflectance instead of Schlick's approximation. With this BSDF, we can see chromatic aberrations on the surface and dispersion effects in the shadows, as shown in the renders of the bunny in Figure 3.

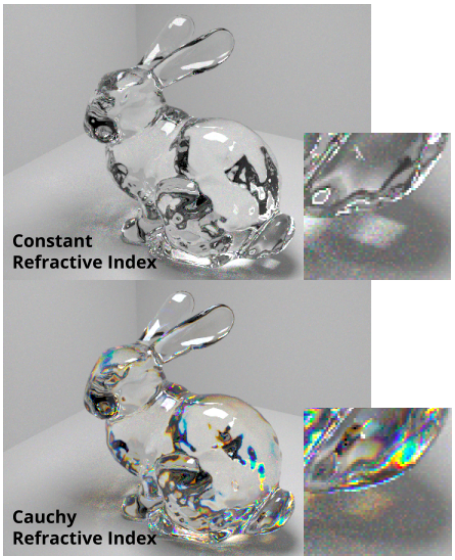

Figure 3: Top, bunny rendered with non-dispersive glass ($A = 1.45, B = 0$). Bottom, bunny rendered with dispersive glass ($A = 1.45, B = 50,000$).

3.2 Spectral Microfacet

The modification to microfacet materials is similar, and solely inside the computation of the Fresnel reflectance, F, in the following:

$$f(\lambda, \omega_i \to \omega_o) = \frac{F(\lambda, \omega_i \cdot h) \cdot G(\omega_i, \omega_o) \cdot D(h)}{4 \cdot (n \cdot \omega_i) \cdot (n \cdot \omega_o)}. \tag{5}$$

In the RGB version, the BSDF stores the refractive index η and extinction coefficient κ at 614 nm (red), 549 nm (green) and 466 nm (blue) and returns the corresponding reflective intensity for these particular wavelengths. The spectral version stores n arbitrarily spaced wavelength samples ($\lambda_i \forall i = 1 \ldots n$) between 380 nm and 780 nm (inclusive) and the corresponding η_i and κ_i at those wavelengths. Given the incoming ray's wavelength, we use a linear interpolation to compute $\eta(\lambda)$ and $\kappa(\lambda)$, which is used to compute the Fresnel reflectance. f now returns the reflective intensity corresponding to the individually sampled wavelength. In Figure 4, we can see that this subtly affects the color of the material, as expected since we sample more of the visible spectrum. The RGB version may appear more "saturated" because it only samples the three primary wavelengths.

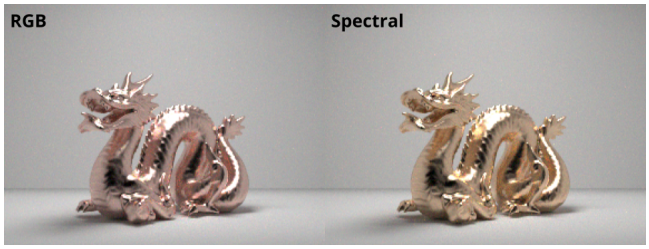

Figure 4: CBdragon with copper surface, $\alpha = 0.15$. On the left, RGB up-sampled microfacet BSDF. On the right, spectral microfacet BSDF.

3.3 Dielectric Thin-film Interference

We extend the spectral microfacet BSDF to additionally model a thin dielectric film on top of a dielectric or conductor base. In this BSDF, we again modify the Fresnel reflectance to account for the bounces between the dielectric film and base (Figure 5). For clarity, we will call the thin-film reflectance the Airy reflectance, R, which we use to replace F in Eq. 5. The following formula are based from those reproduced in Belcour and Barla [1], which were originally given in Born and Wolf (Section 14.4) [2]. Assuming equal distribution of polarized to non-polarized light, the reflectance R is defined as:

$$R = \frac{1}{2}\left(|\mathbf{r}^{\perp}|^2 + |\mathbf{r}^{\parallel}|^2\right), \tag{6}$$

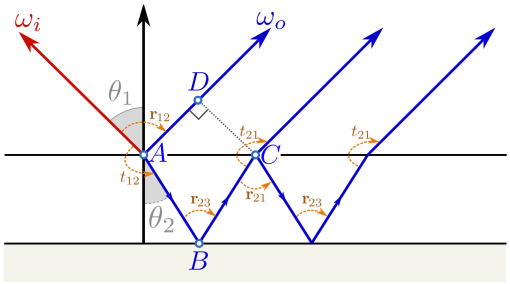

Figure 5: Airy reflectance (Eq. 7) is the summation of the reflectance of all paths that bounce inside the thin film. Figure originally from [1].

where $\mathbf{r}$ is a *complex* reflection coefficient. $\mathbf{r}$ is *complex* because the conductor base has a complex index of refraction $\eta + i\kappa$. $\mathbf{r}$ for either polarization sums the contribution of all rays reflected inside the film-base interface:

$$\begin{aligned}\mathbf{r} &= \mathbf{r}_{12} + t_{12}\mathbf{r}_{23}t_{21}e^{i\phi} + t_{12}\mathbf{r}_{23}\mathbf{r}_{21}\mathbf{r}_{23}t_{21}e^{2i\phi} + \dots \\ &= \mathbf{r}_{12} + \sum_{k=1}^{\infty} t_{12}\mathbf{r}_{23}\left[\mathbf{r}_{21}\mathbf{r}_{23}\right]^{k-1} t_{21}e^{ik\phi} \\ &= \mathbf{r}_{12} + \frac{t_{12}\mathbf{r}_{23}t_{21}e^{i\phi}}{1 - \mathbf{r}_{21}\mathbf{r}_{23}e^{i\phi}} \end{aligned} \tag{7}$$

where $\mathbf{r}_{ab} = r_{ab}e^{i\phi_{ab}}$ is a complex reflection coefficient going from medium a to medium b. Similarly t_{ab} is a real transmission coefficient from a to b. ϕ is the phase shift due to the optical path difference, defined as $\phi = \frac{4\pi}{\lambda}\eta_2 d\cos\theta_2$, where d is the film thickness in nanometers and $\cos\theta_b = \sqrt{1 - \frac{\eta_1^2}{\eta_b^2}(1 - \cos^2\theta_1)}$ according to Snell's law. The formula for r_{ab}, t_{ab}, and ϕ_{ab} for each polarization are included in Appendix A.

Figure 6: Soap ($\eta_2 \approx 1.33$) films on PVC ($\eta_3 \approx 1.54$) and Copper with film thicknesses given on the left.

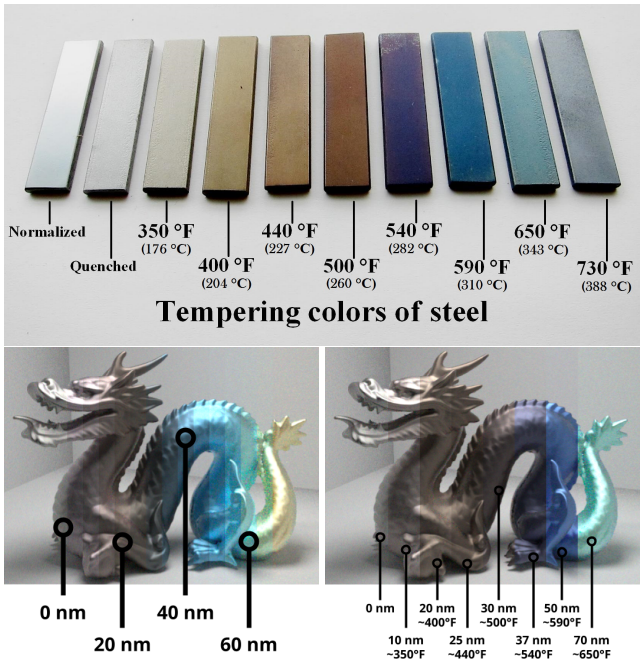

Figure 7: Iron oxide film on iron ($\eta_3 \approx 2.8, \kappa_3 \approx 3$) dragons, compared to a real reference of tempered steel [Wikipedia]. The left dragon uses varying refraction index from [RefractionIndex.info], the right dragon uses constant $\eta_2 = 2.91$, similar to [1].

Using this BSDF, we can reproduce iridescent films on top of both dielectric and conductive bases (Figure 6). In addition, we can also model the effects of tempering on the color of steel alloys. Tempering steel creates a thin surface film of oxide on the base during the heating process. As this film thickens, the appearance changes due to thin-film interference. In Figure 7, we use a base layer of iron with progressively thicker layers of iron oxide, Fe_2O_3, to recreate the phenomenon. On the left, we use a varying refractive index and test thickness between 0 nm and 70 nm at 5 nm increments. On the right, we reproduce the results from the supplemental materials of [1], with a constant refractive index at the thicknesses they used. This approximation has a few potential sources of error. For one, the surface may not just be iron oxide, but a combination which includes chromium oxide. Additionally, the extinction coefficient of oxides are not truly negligible, and thus it cannot be treated as a dielectric. A conductive film has a different equation for reflectance (See Section 14.4 of Born and Wolf [2]).

3.4 Films on Transparent Objects

We can directly use the Airy reflectance (Eq. 6, 7) as our probability of reflection in transmissive materials. With this,

we can model thin-films on transparent materials such as glass and air. The latter is how we create soap bubbles. However, we are unable to sufficiently model translucency as the soap bubbles still cast dark shadows (Figure 8). In addition, films in the real world will have varying thickness due to the effects of air movement and gravity.

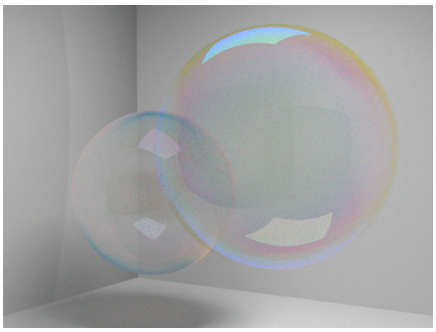

Figure 8: Soap bubbles modeled using a film ($\eta_2 = 1.33$) on a vacuum ($\eta_3 = 1.0, \kappa_3 = 0$). The foreground and background bubbles have a thickness of 500 nm and 350 nm, respectively.

4 BLACK BODY RADIATION

The final elements of the scene to be converted to a spectral version are lights. A common type of light source is a *black body*, which emits a spectrum of light based on its temperature. The black body model can be used to approximate the spectral power distributions for incandescent lighting, the sun, and even the infrared radiation of living things! The SPD of a black body at temperature T is given by

$$s(\lambda) = \frac{2hc^2}{\lambda^5}\left[\exp\left(\frac{hc}{\lambda k_B T}\right) - 1\right]^{-1}, \tag{8}$$

where h is Planck's constant, c is the speed of light, k_B is Boltzmann's constant, and λ is the wavelength [7]. We allow for black body light emitters (point and area) that we parameterize by two variables, a temperature and brightness. Brightness serves as a normalization factor in front of the SPD that can account for rescaling of the black body. When we sample the black body BSDF, we simply return the appropriate intensity at that wavelength and similarly when we sample the light in direct lighting sampling.

Colder objects have black body spectra that appear orange-reddish, while hotter objects have black body spectra that appear blueish. At $T \sim 6500$ K, we get white light, which is the effective temperature of the sun and the basis of the D65 illuminant standard. Figure 9 shows the CBspheres scene for two more extreme temperatures to highlight this difference. Colder lights will appear darker, so we also have a parameter for brightness used to scale Eq. 8.

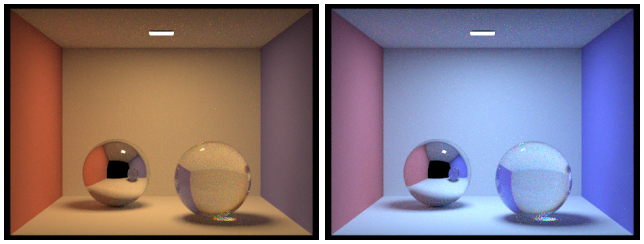

Figure 9: CBspheres with a black body area light temperature at 3000 K (left) and 12000 K (right).

5 DISCUSSION

There were several technical challenges we encountered. First, we had to refactor significantly more code than we originally expected. As an example, since the underlying path tracer algorithm calls the BSDF of the primitive associated with each intersection, we had to rewrite the BSDF class to take wavelength as an argument throughout the code. This required modifying all BSDF definitions, even if the BSDF for several materials were not wavelength-dependent.

Similarly, we had to parse and understand the COLLADA file format to add our own features to incorporate our new materials, such as dispersive glasses and thin films. We also had to understand how COLLADA files were read in to the project to correctly include parsing of our new features.

Another challenge we faced was implementing faceted objects to highlight dispersion (such as the prism). The polymesh class uses vertex normal interpolation to smooth the appearance of scene objects, but we specifically sought angular surfaces. To get around this without writing a new class, we repeatedly refined our meshes such that the effects of normal interpolation were limited to small boundary regions.

We also learned much more about physics and optics than we expected to working on this project. To correctly implement the Fresnel equations and the thin film interference equations, we had to learn how light interacts with surfaces and understand the wave-like nature of light. Upsampling also proved a challenge, as it is highly underdetermined to go from an RGB color to a spectral power function, so it took a significant amount of reading and thinking to determine the most appropriate way to implement upsampling.

6 CONCLUSIONS

This work demonstrates one approach for converting an existing RGB ray tracer into a spectral ray tracer and showcases several wavelength-dependent effects. There are several ways this project can be extended. For example, the utility of the BSDFs can be improved. A next step would be to properly model a conductive film, such as oxide layers. Additionally, since currently no global information about

the object or primitive is passed into each BSDF, we were unable to model nonuniform surface properties such as varying roughness and thin-film thickness. Lastly, in Eq. 5, the shadowing-masking term, G, and normal distribution function, D, can be modified to produce a wider range of effects.

Computationally, this implementation is fairly slow to both run and converge, which is inconvenient when trying to iterate on compositions. Parallelism and optimization would strongly improve the code base. The light dispersion effects are especially poorly sampled—the prism scene in Figure 1 uses 8192 pixel samples (30 billion rays in about 4 hours to render)—because the probability a camera ray refracts into the light source is extremely low when the area light is small (and the light beam must be thin in order to showcase a clear rainbow!). The convergence of this effect could be improved with photon mapping or targeted sampling.

REFERENCES

[1] Laurent Belcour and Pascal Barla. 2017. A practical extension to microfacet theory for the modeling of varying iridescence. *ACM Trans. Graph.* 36, 4, Article 65 (jul 2017), 14 pages. https://doi.org/10.1145/3072959.3073620

[2] Max Born and Emil Wolf. 2019. *Principles of Optics: 60th Anniversary Edition* (7 ed.). Cambridge University Press.

[3] A L Cauchy. 1830. Sur la réfraction et la réflexion de la lumière. *Bulletin de Férussac* XIV (1830), 151–157. https://gallica.bnf.fr/ark:/12148/bpt6k901948/f164.item

[4] Sydnie Shea Cohen, Ada Hu, Eugene Lee, and Daniel Tseng. 2021. Iridescence. *CS 184/284A: Computer Graphics and Imaging* (2021). https://sydnie-shea.github.io/iridescentFinal/

[5] Jason Li, Katherine Song, and Alex Yang. 2020. Rendering Iridescence. *CS 184/284A: Computer Graphics and Imaging* (2020). https://kwsong.github.io/cs184_final/final.html

[6] Ian Mallett and Cem Yuksel. 2019. Spectral Primary Decomposition for Rendering with sRGB Reflectance.. In *EGSR (DL/I)*. 9–15.

[7] M Planck. 1900. Entropie und Temperatur strahlender Wärme. *Bulletin de Férussac* 1, 4 (1900), 719–737. https://gallica.bnf.fr/ark:/12148/bpt6k15311v/f736.tableDesMatieres

[8] T Smith and J Guild. 1931. The C.I.E. colorimetric standards and their use. *Transactions of the Optical Society* 33, 3 (jan 1931), 73. https://doi.org/10.1088/1475-4878/33/3/301

[9] A. Wilkie, S. Nawaz, M. Droske, A. Weidlich, and J. Hanika. 2014. Hero Wavelength Spectral Sampling. *Computer Graphics Forum* 33, 4 (2014), 123–131. https://doi.org/10.1111/cgf.12419 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12419

[10] Chris Wyman, Peter-Pike Sloan, and Peter Shirley. 2013. Simple Analytic Approximations to the CIE XYZ Color Matching Functions. *Journal of Computer Graphics Techniques (JCGT)* 2, 2 (12 July 2013), 1–11. http://jcgt.org/published/0002/02/01/

A THIN-FILM FORMULA

We reproduce these formula here as we discovered a typo in one of our references. We keep similar notation, with minor changes for clarity. These formula are used for the interference between a dielectric outer medium (air/vacuum), dielectric film, and either dielectric or conductor base (i.e. $\kappa_1 = \kappa_2 = 0$ and $\kappa_3 \geq 0$). Each medium a is defined to have index $\eta_a + i\kappa_a$, and the complex reflection coefficient is $\mathbf{r}_{ab} = r_{ab}e^{i\phi_{ab}}$, with:

$$|r_{ab}^{\perp}|^2 = \frac{(\eta_a \cos\theta_a - u_{ab})^2 + v_{ab}}{(\eta_a \cos\theta_a + u_{ab})^2 + v_{ab}}$$

$$\tan\phi_{ab}^{\perp} = \frac{2v_{ab}\eta_a \cos\theta_a}{u_{ab}^2 + v_{ab}^2 - \eta_a^2 \cos^2\theta_a}$$

for perpendicular polarization; and

$$|r_{ab}^{\parallel}|^2 = \frac{\left[(\eta_b^2 - \kappa_b^2)\cos\theta_a - \eta_a u_{ab}\right]^2 + \left[2\eta_b\kappa_b\cos\theta_a - \eta_a v_{ab}\right]^2}{\left[(\eta_b^2 - \kappa_b^2)\cos\theta_a + \eta_a u_{ab}\right]^2 + \left[2\eta_b\kappa_b\cos\theta_a + \eta_a v_{ab}\right]^2}$$

$$\tan\phi_{ab}^{\parallel} = 2\eta_a\cos\theta_a \frac{2\eta_b\kappa_b u_{ab} - (\eta_b^2 - \kappa_b^2)v_{ab}}{(\eta_b^2 + \kappa_b^2)^2\cos^2\theta_a - \eta_a^2(u_{ab}^2 + v_{ab}^2)}$$

for parallel polarization, where

$$u_{ab} = \frac{\sqrt{U_{ab} + V_{ab}}}{2}, \quad v_{ab} = \frac{\sqrt{V_{ab} - U_{ab}}}{2}$$

with

$$U_{ab} = \eta_b^2 - \kappa_b^2 - \eta_a^2\sin^2\theta_a, \quad V_{ab} = \sqrt{U_{ab}^2 + 4\kappa_b^2}.$$

Lastly, the Fresnel transmission coefficients are:

$$t_{ab}^{\perp} = \frac{2\eta_a\cos\theta_a}{\eta_a\cos\theta_a + \eta_b\cos\theta_b}$$

$$t_{ab}^{\parallel} = \frac{2\eta_a\cos\theta_a}{\eta_a\cos\theta_b + \eta_b\cos\theta_a}.$$

B SUPPLEMENTAL MATERIALS

Visit https://www.dawsondo.net/projects/spectral for video, slides and additional renders.

C CONTRIBUTIONS

- DD: Implemented spectral upsampling, RGB downsampling, spectral microfacet & thin-film BSDFs, composed scenes for final render and figures.
- JF: Refactored the code for wavelength dependence, implemented the dispersive glass BSDF, implemented the black body BSDF, and handled parsing of modified COLLADA files for new inputs.
- JC: Modeled custom objects and modified COLLADA files for final scene render and figures.
- JG: Report formatting.

Received 30 April 2024