Spectral Primary Decomposition for Rendering with sRGB Reflectance

Ian Mallett and Cem Yuksel, ***University of Utah***

https://graphics.geometrian.com/research/spectral-primaries.html

Most 3D models contain images called "textures", which are used to define colored data on the surfaces, such as reflectance. These images are almost always stored in the sRGB color space. When the object is rendered, these colors must be reproduced accurately.

Realistic renderers rely on spectral light transport—that is, they simulate individual wavelengths of light instead of RGB triples. This requires a *spectrum* to be somehow generated from the RGB *pixels* of the texture: an underconstrained problem called **Spectral Upsampling**. Due to mistakes in the literature, doing spectral upsampling without introducing error into the final color was long believed to be impossible.

We identify the mistakes in previous work and show that this problem can actually be solved very simply by calculating each reflectance spectrum as a linear combination of three basis spectra. The sRGB pixel value is unpacked into a linear-RGB (ℓRGB) triple. Each component is multiplied by a corresponding spectrum, and the three scaled spectra are then summed to produce the final reflectance spectrum.

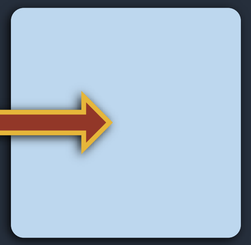
Utah Graphics

EGSR 2019 EUROGRAPHICS SYMPOSIUM ON RENDERING /////

(renderer) (paper)

Step 1

During rendering, the renderer fetches a pixel from the texture. The pixel is decompressed from sRGB (gamma-encoded, three bytes) to linear RGB (ℓRGB, three floats).



sRGB = < 189, 215, 238 >

ℓRGB ≈ < 0.51, 0.68, 0.85 >

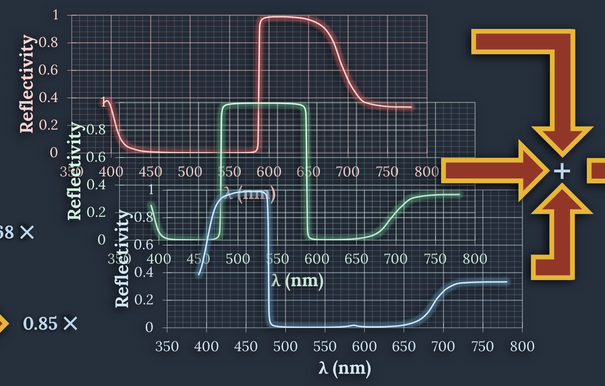
0.51 ×

0.68 ×

0.85 ×

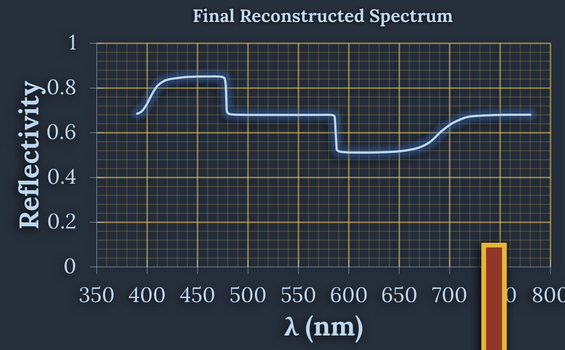
Step 2

The individual red, green, and blue components of the ℓRGB triple are multiplied by a corresponding, precomputed, constant spectrum (our **Spectral Basis**).



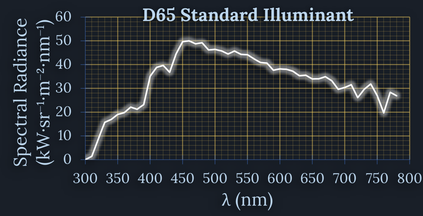
Step 3

These three intermediate spectra are summed to produce the final reflectance spectrum, which can be used immediately by the renderer. The reflectance spectrum reproduces the colors exactly ("metamerism") under D65 illumination.



Precomputing the Spectral Basis

The spectral basis is precomputed by solving a simple optimization problem. First, we observe that, by the definition of sRGB, the D65 Standard Illuminant (a particular idealized reference light source) *must* appear "white" (i.e., the sRGB value < 255, 255, 255 >).



=

sRGB = < 255, 255, 255 >

ℓRGB = < 1.0, 1.0, 1.0 >

("white")

Although sRGB is defined for emission, by applying basic optics, the above definition allows us to determine what the reflected color must be under certain lighting conditions. Specifically, for albedo described by ℓRGB triple < r, g, b >, a spectrum S that produces the correct color is given by:

$$\underbrace{\begin{bmatrix}(\mathbf{D_{65}}\odot\bar{\mathbf{x}})^T\\(\mathbf{D_{65}}\odot\bar{\mathbf{y}})^T\\(\mathbf{D_{65}}\odot\bar{\mathbf{z}})^T\end{bmatrix}}_{A\in\mathbb{R}^{3\times N}}\mathcal{S}=\underbrace{Y_{D65}\,M\begin{bmatrix}r\\g\\b\end{bmatrix}}_{\mathbf{b}\in\mathbb{R}^{3\times 1}}$$

$$\begin{aligned}\text{minimize:}\quad & \|A\mathcal{S}-\mathbf{b}\|\\ \text{subject to:}\quad & \mathbf{0}\le\mathcal{S}_{r,g,b}\le\mathbf{1}\end{aligned}$$

While this only gives a spectrum for a particular input ℓRGB triple, the optimization problem can be generalized to three spectra S_r, S_g, and S_b, which can be linearly combined to form any desired < r, g, b > triple. Solving this problem with standard techniques produces the spectral basis.

$$\underbrace{\begin{bmatrix}(\mathbf{D_{65}}\odot\bar{\mathbf{x}})^T & 0 & 0\\(\mathbf{D_{65}}\odot\bar{\mathbf{y}})^T & 0 & 0\\(\mathbf{D_{65}}\odot\bar{\mathbf{z}})^T & 0 & 0\\0 & (\mathbf{D_{65}}\odot\bar{\mathbf{x}})^T & 0\\0 & (\mathbf{D_{65}}\odot\bar{\mathbf{y}})^T & 0\\0 & (\mathbf{D_{65}}\odot\bar{\mathbf{z}})^T & 0\\0 & 0 & (\mathbf{D_{65}}\odot\bar{\mathbf{x}})^T\\0 & 0 & (\mathbf{D_{65}}\odot\bar{\mathbf{y}})^T\\0 & 0 & (\mathbf{D_{65}}\odot\bar{\mathbf{z}})^T\end{bmatrix}}_{A\in\mathbb{R}^{9\times 3N}}\begin{bmatrix}\mathcal{S}_r\\\mathcal{S}_g\\\mathcal{S}_b\end{bmatrix}=\underbrace{Y_{D65}\begin{bmatrix}M\begin{bmatrix}1\\0\\0\end{bmatrix}\\M\begin{bmatrix}0\\1\\0\end{bmatrix}\\M\begin{bmatrix}0\\0\\1\end{bmatrix}\end{bmatrix}}_{\mathbf{b}\in\mathbb{R}^{9\times 1}}$$

$$\begin{aligned}\text{minimize:}\quad & \|A\mathcal{S}_{r,g,b}-\mathbf{b}\|\\ \text{subject to:}\quad & \mathbf{0}\le\mathcal{S}_{r,g,b}\le\mathbf{1}\\ & \mathcal{S}_r+\mathcal{S}_g+\mathcal{S}_b=\mathbf{1}\end{aligned}$$

Renderer uses for surface reflectance