Spectral Imaging in Production

Course Notes Siggraph 2021

June 11th, 2021

ANDREA WEIDLICH (Organizer)
Weta Digital

ALEX FORSYTHE
Academy of Motion Picture Arts and Sciences

SCOTT DYER
Academy of Motion Picture Arts and Sciences

THOMAS MANSENCAL
Weta Digital

JOHANNES HANIKA
Karlsruhe Institute of Technoloy

ALEXANDER WILKIE
Charles University Prague

LUKE EMROSE
Animal Logic

ANDERS LANGLANDS
Weta Digital

Abstract

Unlike path tracing, spectral rendering is still not widely used in production although it has been around for more than thirty years. Traditionally connected to spectral effects such as dispersion and interference, spectral rendering – and, more importantly, the use of spectral data in general – is predominantly a way to guarantee colour fidelity. Additionally, with the rise of path tracing and the growing use of LED lights on-set as well as the recent shift to LED walls in virtual production, it becomes increasingly evident that the traditional way of seeing colour and light as RGB triplets is insufficient if colour accuracy is required.

The purpose of the course is two-fold. First and foremost, we want to share what we learned on our way towards a spectral image pipeline. We will talk about the unique opportunities and challenges the use of spectral data brings in a modern production pipeline and our motivation to build a spectral renderer. Since spectral data influences every step of the pipeline, the course will go beyond rendering aspects. We will discuss data acquisition and will shed some light on how to tackle the special problem of LED lights in production as well as its practical usage.

The second aim of the course is to build a community. We want to see the topic evolve over the next few years and connect people to shape the future together until spectral imaging is as ubiquitous as path tracing is in production.

Syllabus

RGB is not a Rainbow (Andrea Weidlich, 25 minutes)

Compared to path tracing, spectral rendering is something of a wallflower. Most people know that it exists, but only a few have used it in practice. Unsurprisingly, there are still many myths and misconceptions that surround spectral image synthesis, so the first part of the course will give a general introduction into the topic and discuss the opportunities it gives in regards to digital content production. New workflows are required, paradigms have to shift, but more importantly, there are still many open research questions which need to be addressed.

Spectral Characterization of Digital Cameras in Motion Picture Production (Alex Forsythe and Scott Dyer, 20 minutes)

Digital camera images are formed by the integration of scene spectral radiances which illuminate the camera's photosites. Proper characterization of the camera's response to these scene spectral radiances is critical to integrating a camera's images with images from other sources including those of other cameras, film, synthetically generated visual effects, and the final display of those images. This presentation will outline the fundamental theories, techniques and tools used to properly characterize a digital camera and build characterization transformations.

Applied Spectral Knowledge in Virtual Production (Thomas Mansencal, 30 minutes)

Measurements of the electromagnetic spectrum and their analysis are the foundation of colourimetry. This talk will present some of the colour rendering and reproduction challenges faced by LED wall based virtual productions and how spectral analysis can provide in-depth understanding and contribute to design solutions around them.

Hallucinate Colour (Johannes Hanika, 30 minutes)

For many tasks around rendering and imaging it is a useful tool to generate spectral representations of colours given tristimulus values as input. For instance path tracing can only compute correct indirect illumination when light transport is simulated per wavelength, and white balancing photography can be performed more accurately when done in the spectral domain instead of a tristimulus colour space.

Upsampling a three dimensional colour space to a continuous function space over the wavelength domain is an underspecified process, so there are many possibilities to explore. Indeed there is an incredible variety of methods which have been published to achieve this for a few different use cases. They are roughly classified into whether they allow negative energy, or unbounded energy, or none of the two. The last option is particularly interesting for surface reflectances or spectral albedos.

This talk focuses on insight rather than individual algorithms, and will give an introduction to the topic and the related constraints. It also includes a coarse overview over existing methods and will classify them with respect to properties of the resulting spectra, execution

speed, and storage considerations. This will give the audience the understanding and the necessary tools to pick the best suited algorithm for their individual needs.

Fluorescent, Spectral Workflows in ART and Spectral EXR Alexander Wilkie, 30 minutes)

We will discuss the flow of spectral information in a path tracer, and then specifically focus on Hero Wavelength Spectral Sampling (HWSS) as a production level technology that allows fast and efficient incorporation of spectra information in the actual path tracer, at almost no additional computation cost compared to colourspace rendering. We will also briefly discuss how fluorescence can be included in a spectral renderer, and how it can be made to work together with HWSS.

A Hero Beneath the Surface (Luke Emrose, 25 minutes)

Implementing a paper never quite goes as planned. Luke Emrose discusses the pathtraced road to implementing spectral subsurface scattering within Animal Logic's proprietary pathtracer Glimpse. Motivations, expectations and implementation details will be discussed along with an analysis of the results, the improvements therein and future directions. Expect to be scattered, reflected, absorbed and ultimately less noisy than you were before you started.

Spectral Rendering in Production at Weta Digital (Anders Langlands, 20 minutes)

Weta Digital has been using spectral rendering in production for the past 8 years. We discuss tools and methodologies for practical acquisition of spectral data and its use in a production rendering pipeline.

Organizer

Andrea Weidlich, Weta Digital

Andrea Weidlich is a Senior Researcher at Weta Digital where she is responsible for the material system attached to Weta's proprietary physically-based renderer, Manuka. Andrea grew up in Vienna, Austria, where she studied technical computer science, and later focused on computer graphics. Before joining Weta, she worked for the automotive industry. Her main research areas are appearance modeling and material prototyping. Andrea holds a Master of Arts in Applied Media from University of Applied Arts, Vienna and a Ph.D. in Computer Science from Vienna University of Technology.

Presenters

Alex Forsythe, Academy of Motion Picture Arts and Sciences

Alex Forsythe is Director of Imaging Technology for the Academy of Motion Picture Arts and Sciences. He has served as the Technical Project Lead for ACES since 2006 and won a 2012 Primetime Emmy Engineering Award as part of the ACES team. Prior to joining the Academy Alex worked for Intel Corporation in the New Business Initiatives Division and Eastman Kodak in the Imaging Science Division of the Research and Development Laboratories. Alex holds BS and MS degrees from Rochester Institute of Technology. He's an active member and participant various professional and standards organizations including the Society of Imaging Science and Technology, the Society of Motion Picture and Television Engineers (IS&T), the Illuminating Engineering Society (IES), the International Commission on Illumination (CIE), and the International Organization for Standardization (ISO). He's currently a member of the US delegation for the ISO TC-36 Technical Advisory Group.

Scott Dyer, Academy of Motion Picture Arts and Sciences

Scott Dyer is the Senior Imaging Engineer at the Science and Technology Council of the Academy of Motion Picture Arts and Sciences. Since 2010, Scott has been developing for the Academy Color Encoding System (ACES) – a global standard for interchanging digital image files, managing color workflows and creating masters for delivery and archiving.Scott Dyer is the Senior Imaging Engineer at the Science and Technology Council of the Academy of Motion Picture Arts and Sciences.

Thomas Mansencal, Weta Digital

Thomas Mansencal is the lead pipeline developer for the Special Projects team at Weta Digital. He is involved in porting the appearance of Manuka's PhysLight offline rendered assets into Unreal Engine, trying to maintain their physical accuracy while designing imaging and colour management workflows to support this process. As a member of the USC's Entertainment Technology Center (ETC) – Virtual Production Working Group, he contributes to providing guidelines and best practices to capture imagery on LED walls. He participates in open source software development as the lead developer for Colour Science for Python. He is also part of the Academy Color Encoding System (ACES) Architecture TAC and the maintainer of the OpenColorIO configuration for ACES.

Johannes Hanika, Karlsruhe Institute of Technology

Johannes was introduced to the secrets of computer graphics in Ulm University, working on low level ray tracing optimisation, quasi-Monte Carlo point sets, light transport simulation, and image denoising. The following ten years he worked as a researcher for Weta Digital, first in Wellington, New Zealand, later remotely from Germany. He is co-architect of Manuka, Weta Digital's physically based spectral renderer. Since 2013, Johannes works as a post-doctoral fellow at the Karlsruhe Institute of Technology. In 2009, Johannes founded the darktable open source project, a workflow tool for RAW photography. Johannes has a movie credit for "Abraham Lincoln: Vampire Hunter".

Alexander Wilkie, Charles University Prague

Alexander Wilkie is an associate professor at Charles University in Prague. His research interests are in the area of Predictive Rendering: in particular, spectral rendering, inclusion of polarisation and fluorescence effects in rendering systems, skylight models, and appearance enhancement for 3D print technologies.

Luke Emrose, Animal Logic

Luke Emrose is the Technical Lead for Rendering at Animal Logic. After a previous life of over 9 years as a shader writer, artist, lighter, modeler, on-set previs artist and TD, he has spent the past 8 years dedicated to developing core rendering technologies for the Animal Logic proprietary renderer, Glimpse. Having contributed technology to The LEGO Movie, The LEGO Batman Movie, The LEGO NINJAGO Movie, Peter Rabbit 1&2, Super Pets, The Magician's Elephant and many more, he continues to thrive on the difficult and never-ending challenge of photo-realistic rendering.

Anders Langlands, Weta Digital

Anders Langlands is an Academy Award-nominated Visual Effects Supervisor at Weta Digital, and previously MPC. He is known for his work on The Martian, War for the Planet of the Apes, and Mulan. In his abundant free time, he enjoys researching new techniques related to rendering and optics.

Contents

1

RGB is not a Rainbow

ANDREA WEIDLICH, *Weta Digital*

1.1 Motivation

While path tracing became onmi-present in modern movie production pipelines, the majority of production renderers still consider light – surface interactions as a tristimulis operation, commonly done in RGB. Spectral images, and in particular spectral rendering, is most commonly associated with dispersion or structural colours, i.e. diffraction or interference effects where certain wavelengths are either cancelled out or their intensity is increased. While these effects are beautiful in themselves, it is commonly overlooked that spectral data can be found in every single part of a production pipeline and has a much wider impact on the appearance of the final image than just on certain isolated parts.

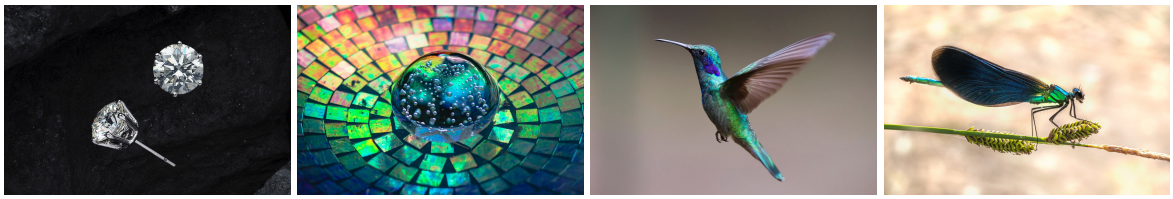


Figure 1.1: Real life examples of various effects commonly associated with spectral rendering. Strong colours can be produced by the nonlinear wavelength-dependent refraction known as dispersion. Interference and diffraction effects are caused by small structures. From left to right: Dispersion, thin film effects and diffraction on feathers and insects.

It is no coincidence that spectral rendering gained more momentum in recent years Fascione et al. [2018]. With the rise of path tracing, production renderers started to focus on physically-based materials and input data to make images ever more realistic and faithful. Arguably, spectral rendering is not the only way to generate realistic images, and many spectral effects can be simulated with a conventional tristimulus approach. However, the benefits of treating light as spectra lie elsewhere.

- **Colour fidelity of bounces.** Colour in a path tracer is an accumulation effect. Bounces with a similar saturated colour will produce an overly saturated and unnatural effect in an RGB renderer since colours are multiplied by themselves.

- **Only spectra can create metamerism.** Metamerism is a phenomenon where two spectra will result in the same perceived colour under one illumination, but in a different sensation under another illumination. An example can be seen in Figure 1.2.
- **Spectra are physical quantities.** Colour is a sensation the human brain produces when a person perceives a light of various intensities. In contrast to a spectrum, an RGB colour cannot be directly measured and its value depends on the colour space (and the white point) it is in.
- **Spectra can be measured.** In contrast to colours, spectral reflectances or emission can be measured with devices like a spectroradiometer. The results are absolute intensities.
- **Faithfullness.** Spectral data is not limited to a single part of a production pipeline, they can be found in every subsystem. A complete end-to-end spectral pipeline will allow a more faithful reproduction of reference images and plates since it simulates every subsystem correctly.
- **Natural integration of spectral effects.** Spectral effects such as dispersion, interference or fluorescence can be integrated into a spectral renderer more easily than their RGB counterparts. Since quantities are already calculated per wavelength, computations are reduced to simple operations without an extra layer that converts spectral internal calculations back to RGB or approximations which potentially reduce the accuracy of the result.

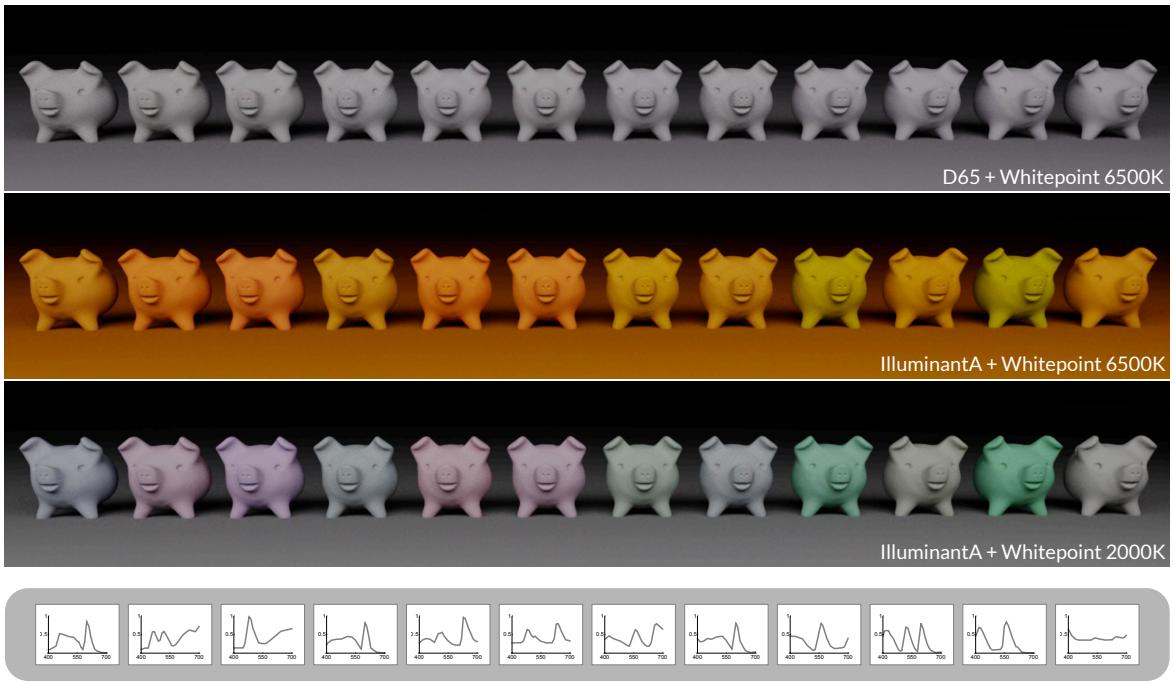


Figure 1.2: Twelve grey spectra that are metameric under D65 illumination. If the illumination is changed to IlluminantA (second row), a clear difference can be seen in the colour of the objects. The difference is even more visible if the whitepoint is changed from 6500K to 2000K (third row). The spectra were inspired by Wyszecki and Stiles [1982].

Figure 1.3: Same spectra as in figure 1.2 rendered under D65 but used as absorption coefficients. A difference can be seen even under D65 illumination.

1.2 Driving Looks with Spectra

In a production renderer, we normally have to fulfil two different conflicting objectives. On one side, we strive to reproduce reality as accurately as possible; only convincing and believable images will be able to immerse an audience into the story of a character. On the other side, we need to be able to bend reality to make an artistic vision come true. Especially the second goal has seen little attention so far in connection with spectral imaging techniques – although spectral data, and in particular spectral rendering, can be utilised to expand the appearance space of an object.

While metamerism is arguably not as important in production rendering since lights and reflectance colours can be adjusted on a per-shot basis, we think that they encompass a significant potential when it comes to designing the appearance of objects. Because metamerism does not only influence the direct appearance of the object but also its indirect look. This is what makes spectral rendering so important for a path tracer, especially for the rendering of objects where there will be multiple interactions with the same or a similar colour, such as subsurface scattering, hair or canopies.

To illustrate this point, we took the twelve metameric spectra from Figure 1.2 and used them as absorption coefficients. The subsurface scattering will lead to a subsequent multiplication of the same spectrum over various bounces. As it can be seen in Figure 1.3, the spectra react quite differently to this process, which produces the difference in the final colour.

Another aspect that has not been explored sufficiently yet is how to make use of the multi-dimensionality of colour resulting from a spectrum to our advantage Bergner et al. [2009]. In a tristimulus renderer, a colour can be defined only once, and its appearance in different illuminations cannot be changed. In contrast, by utilising the mematerial nature of a spectrum, different combinations of reflectance and illumination can be achieved.

In Figure 1.4 we define the output colour of the blue asset under two different illuminations, D65 and IlluminantA. This allows us to gain control over its appearance when we switch from D65 to IlluminantA, or use both lights in combination. Once the final output colour is determined, a solver can be used to derive a metameric spectrum which satisfies both conditions. This spectrum can then be used to drive further explorations.

These two examples are by far not all the possibilities that open up and should only illustrate the potential spectral data has when it comes to tuning the appearance of an object. In the next section we will discuss how to deal with spectral data in a production renderer.

Figure 1.4: Metamerism can be used to express colours in a multi-dimensional way. A look development can be done under different illuminations, and a target spectrum can be built that satisfies both inputs. Note that both spectra look identical under D65.

1.3 Proposal for a Spectral Production Material System

When it comes to spectral imaging in production, we know how to address most parts of the pipeline. We can for example easily measure light spectra, gel transmission or camera characteristics. And although reflectances of surfaces can be easily measured as well, appearance usually changes on a per-point basis which means that instead of a single sample, a full spectral texture needs to be measured which is a memory-intensive and time-consuming task. Since normal production assets have dozens of textures, this approach is not practical. Moreover, tools and pipelines are built on the assumption that artists will paint and think in RGB.

For the rest of this section we want to focus on this particular aspect of the pipeline. In the following sections we will explore different ways of supplying spectral data to a production renderer from a look development point of view and what they mean in terms of storage and flexibility. For the sake of demonstrating the various concepts, we will use Manuka's material system.

1.3.1 Manuka's Layered Material System

In contrast to other production renderers, Weta's proprietary renderer Manuka has a pre-shading architecture instead of evaluating shaders on a per-hit basis. Therefore we do not have to evaluate shaders on the fly and we can avoid expensive texture reads which often cause bottle-necks. Consequently, our shading networks can become much larger and more detailed than usual production shader networks and can have dozens of texture reads per BSDF. Data that comes from a shader has to be completely view-independent; view dependent calculations are done inside the material system during light transport. The shader will solely give Manuka the final input data (i.e. the BSDF parameters and the weights of BSDFs and layers) and the layout of the layer-stack.

Material layout and data are set up once during the initialisation before light transport starts. Every BSDF has a scalar weight, w_b. These weights can be between 0.0 and 1.0. Like BSDFs, layers also have scalar weights w_l which can assume values in the range of 0.0 to 1.0, weights smaller than 1.0 will result in BSDFs or layers existing only to a certain percentage. Usually weights are textured with masks. In a shader we first define once *per shape* the BSDFs we want to use, e.g.

```
AddLobe("bsdf1", SomeBsdf());
AddLobe("bsdf2", SomeBsdf());
```

and combine them later into a layer with

```
CombineLayers("layer", "mix", "bsdf1","bsdf2");
```

BSDFs can be combined with other BSDFs or layers. In theory, users can combine as many BSDFs as they want although we do not allow cycles or the use of the same BSDF in more than one place. Layering programs (i.e. the layout of the layers and BSFDs) are the same across a shape, only the input parameters vary. Since the program does not store or precompute light transport data, it is lightweight and does not cause noticeable memory overhead. Nevertheless, if the same layout is used on multiple shapes, we store it only once in the scene. The actual data of the individual BSDFs is stored *per vertex*.

```
SetLobe("bsdf1", SomeBsdf(RGB(0,1,0)), weight1);
SetLobe("bsdf2", SomeBsdf(RGB(0,0,1)), weight2);
SetLayer("layer", layerweight);
```

1.3.2 Spectral Interfaces

Since Manuka is a spectral renderer, all BSDFs compute all quantities spectrally. Traditionally, our input data used to be solely in RGB space since artist workflows require them to be; during rendering we would spectrally uplift the data. This results in the interesting problem that, since there is no unique correspondence between a single spectrum and a specific RGB value, the generated spectra set is arbitrary and not necessarily the same as you would get for a photographed reference asset. Therefore we propose to extend BSDF interfaces to be able to deal with spectral data. Figure 1.5 shows a schematic illustration of the proposed approaches.

1.3.2.1 Direct Spectral Input

While the majority of our use-cases are rather forgiving and would allow for RGB input, we do offer the possibility to pass in spectral data directly from the outside. In production, this is most often used for spectral light sources which on one hand can be measured easily and on the other hand have often unique spectra that cannot be generated from RGB data very accurately.

```
AddLobe("leaf", LeafBrdf(), AddSpectrum(myleafspectrum) );
AddLobe("paint", PaintBrdf(), AddSpectrum(mypaintspectrum) );
CombineLayers("layer1", "coat", "paint", "leaf" );

SetLobe("bsdf1", LeafBrdf(), weight1 );
SetLobe("bsdf2", PaintBrdf(), weight2 );
SetLayer("layer1", layerweight );
```

Since the spectrum is stored only once per object, we do not have to be concerned about memory. However, the usefulness of this data type is limited because the colour will be constant; apart from light sources and reference objects, we use this data type primarily for homogeneous volumes. The complete opposite is storing spectral inputs on a per vertex varying level. Currently we allow for 10 equally-spaced bins per spectrum.

```
AddLobe("leaf", LeafBrdf());
AddLobe("paint", PaintBrdf());
CombineLayers("layer1", "coat", "paint", "leaf" );

SetLobe("bsdf1", LeafBrdf(spectrum1), weight1 );
SetLobe("bsdf2", PaintBrdf(spectrum2), weight2 );
SetLayer("layer1", layerweight );
```

This comes, however, with a heavy impact on memory since we need to load and store much more data on each vertex. It should be noted that so far we only use this feature

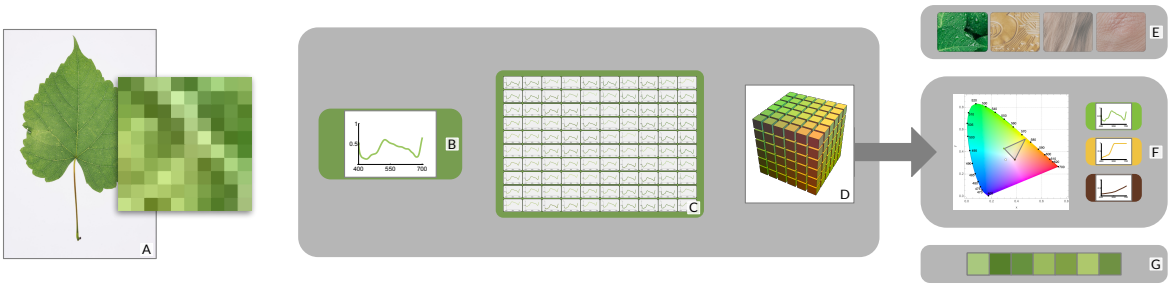


Figure 1.5: Schematic illustration of different types of spectral input. Colour information (A) can be stored directly as constant spectrum (B), an explicit spectral texture (C) or derived through a target (D). Targets can be either pre-defined (E), custom-built with one or more user-defined input spectra (F) or generic (G).

internally for emissive volumes and have never exposed this option to the artists due to the memory cost and the limitations when it comes to actually generating spectral textures. To avoid this issue, we propose to extend the material input system to be able to uplift spectra with prior information.

1.3.2.2 Indirect Spectral Input – Target-based Uplifting

Target-based uplifting describes the generation of spectral data from RGB input which is tailored to a user-defined spectrum. There are three different places where we could store a target.

- **Per object**. The target space will be constant per object. While this is the most performant and least memory intensive option, the drawback is that it is not suitable for multi-layered materials were materials (and hence spectra) of different nature are combined.
- **Per BSDF/layer**. The target space is defined on a BSDF level. Alternatively one could also choose a vertical layer operation such as coating.
- **Per parameter**. Every parameter in a BSDF gets its own target space. While this option is the most flexible, it will be the least performant because of the potential explosion in the number of target spaces.

We argue that the best trade-off between performance and storage is to define a target space per lobe or layer. It has the advantage of being more performant than the component system since there are significantly less BSDFs in a material than parameters in a BSDF. In addition, it is more flexible than the object approach. Since BSDFs in a multilayer system are usually small components that are combined together through layering, they normally describe the same or a similar type of material with similar spectral properties.

In practice, most assets which will profit from spectral input can be categorised into a few target spaces, such as chlorophyll, melanin or conductors. For these often-used types it makes sense to use pre-defined and hence pre-computed target spaces which will save time and memory.

```
AddLobe("leaf", LeafBrdf(), Target("chlorophyll"));
AddLobe("paint", PaintBrdf(), Target("dulux"));
```

```
CombineLayers("layer1", "coat", "paint", "leaf" );

SetLobe("bsdf1", LeafBrdf(RGB(0.1, 0.5, 0.2)), weight1 );
SetLobe("bsdf2", PaintBrdf(RGB(0.8, 0.75, 0.85)), weight2 );
SetLayer("layer1", layerweight );
```

For more esoteric spectra or spectra that have been designed similarly to the approach shown in Figure 1.4, one or more spectra can be passed in. The expectation is that a given RGB value will be perfectly uplifted to the given spectrum or spectrally defined gamut, whereas colours further away will use a spectrum of similar shape.

```
AddLobe("leaf", LeafBrdf(), spectrum1, spectrum2));
AddLobe("paint", PaintBrdf(), spectrum3);
CombineLayers("layer1", "coat", "paint", "leaf" );

SetLobe("bsdf1", LeafBrdf(RGB(0.1, 0.5, 0.2)), weight1 );
SetLobe("bsdf2", PaintBrdf(RGB(0.8, 0.75, 0.85)), weight2 );
SetLayer("layer1", layerweight );
```

Lastly, there will be plenty of scenarios where the composition of a material is not known (meaning it is impossible to define a valid spectrum) or the artist decides that any spectrum is good enough. For these cases it is important to supply a fast and generic uplifting algorithm such as Smits [2000], Jakob and Hanika [2019] or Peters et al. [2019]. The only requirements are that the algorithm faithfully reproduces the desired colour under a given illumination, it interpolates smoothly across neighbouring colours, and that the resulting spectrum is non-negative and within the same dimensions as the input colour.

```
AddLobe("leaf", LeafBrdf());
AddLobe("paint", PaintBrdf());
CombineLayers("layer1", "coat", "paint", "leaf" );

SetLobe("bsdf1", LeafBrdf(RGB(0.1, 0.5, 0.2)), weight1 );
SetLobe("bsdf2", PaintBrdf(RGB(0.8, 0.75, 0.85)), weight2 );
SetLayer("layer1", layerweight );
```

1.3.3 Conclusion

In this chapter we discussed how a shader system in a spectral production renderer could work with different ways of defining spectral input. We believe that using spectral data will allow for more realism as well as flexibility; its potential has not been investigated sufficiently. However, before spectral rendering in production becomes standard, many new workflows and algorithms need to be explored to make it practical. Examples are efficient ways to design spectra or how to design fluorescent responses. We do not believe that RGB will go away any time soon, but we are certain that over time the community will rise to the challenge and will develop new algorithms and build tools and pipelines to accommodate for this.

References

Steven Bergner, Mark S. Drew, and Torsten Möller. 2009. A Tool to Create Illuminant and Reflectance Spectra for Light-Driven Graphics and Visualization. *ACM Trans. Graph.* 28, 1, Article 5 (Feb. 2009), 11 pages. https://doi.org/10.1145/1477926.1477931

Luca Fascione, Johannes Hanika, Mark Leone, Marc Droske, Jorge Schwarzhaupt, Tomas Davidovic, Andrea Weidlich, and Johannes Meng. 2018. Manuka: A Batch-Shading Architecture for Spectral Path Tracing in Movie Production. *ACM Transactions on Graphics* 37 (08 2018), 1–18. https://doi.org/10.1145/3182161

Wenzel Jakob and Johannes Hanika. 2019. A Low?Dimensional Function Space for Efficient Spectral Upsampling. *Computer Graphics Forum* 38 (05 2019), 147–155. https://doi.org/10.1111/cgf.13626

Christoph Peters, Sebastian Merzbach, Johannes Hanika, and Carsten Dachsbacher. 2019. Using Moments to Represent Bounded Signals for Spectral Rendering. *ACM Trans. Graph.* 38, 4, Article 136 (July 2019), 14 pages. https://doi.org/10.1145/3306346.3322964

Brian Smits. 2000. An RGB to Spectrum Conversion for Reflectances. *Journal of Graphics Tools* 4 (06 2000). https://doi.org/10.1080/10867651.1999.10487511

Guenter Wyszecki and W.S. Stiles. 1982. *Color science : concepts and methods, quantitative data and formulae* (2nd ed. ed.). Wiley, New York.

2

Spectral Characterization of Digital Cameras in Motion Picture Production

ALEX FORSYTHE, *Academy of Motion Picture Arts and Sciences*
SCOTT DYER, *Academy of Motion Picture Arts and Sciences*

2.1 Camera Characterization

2.1.1 Color Image Formation in Digital Cameras

It is critical to recognize that digital camera sensors do not inherently capture color. Sensors produce a signal from different wavelengths of light of varying levels of efficiency, but the sensor yields a single signal regardless of the color of light striking. In order to produce color images, we either break the light into broad spectral bands then capture each of them using individual sensors, or we place filters over the individual pixels such that each pixel receives one third of the total spectrum illuminating the sensor with the other two thirds being interpolated from the surrounding pixels.

It is fair to ask the question, why not to capture more bands correlated with individual wave- lengths of light. So called multispectral or hyperspectral cameras do exist. The issue in motion picture production comes down to practicality and image quality. These cameras are expensive, slow and the images may leave a lot to be desired in terms of image quality.

So, we primarily rely on trichromatic color reproduction, which has been shown that to be capable of reproducing any color through three primaries. Experiments have shown that asking a viewer to mix three primary lights allows them to match a test light of any spectrum. In some cases, it might be necessary to modify the test light with one of the primaries, but the results of these color matching experiments allow for a detailed understanding of how to match any color with from three primaries. [Stockman and Brainard 2010]

The color matching data allows us to determine the sensitivity of the three cones of the human retina [Stockman et al. 1993] and transform the data into the standardized color spaces such as CIE XYZ, which has been normalized such as that all the value are positive and the Y channel correlates with the luminous efficiency function, V-lambda. [Wyszecki and Stiles 2000]

2.1.2 Trichromatic Color Reproduction

Using the methods described earlier, we are able to mimic human color trichromatic vision with a digital camera. A digital camera is said to satisfy the Luther condition if its spectral sensitivities are a linear combination of a CIE color matching function.[Von Luther 1927],[Ohta and Robertson 2005]

Should we aim to build digital cameras with spectral sensitivities that are a linear combination of color matching functions? For example, if two similar, but measurably different color patches are captured with cameras satisfying the Luther condition, those differences will be accurately visible in the image data. However, if the camera spectral sensitivities are not a linear combination of the CIE color matching functions, there is a risk that the camera will not see the patches as different. Once the camera has captured the same RGB value for both patches, color fidelity is lost and color error exists.

In practice, however, image quality has many vectors of which color reproduction is only one. Due to a variety of factors, if cameras were to satisfy the Luther condition, they would likely have unacceptable sacrifices in other image quality attributes to achieve perfect color.[Kuniba and Berns 2009] Further, manufacturing cameras which perfectly satisfy the Luther condition is difficult. So, image sensor designers often seek a balance between color reproduction and noise attributes within the restriction of the materials to their disposal.

One can get the sense from a sample set of spectral sensitivities there is an attempt to embody aspects of the CIE color matching functions, particularly in the red channel where there's sensitivity in the short wavelengths. This was likely an effort to reduce error when converting from camera RGB to CIE XYZ.

One question that's commonly asked by those attempting to convert camera signals into standardized color spaces like CIE XYZ is "What are my camera's primaries?" That question is likely born out of running across a series of equations describing the calculation of RGB to XYZ transforms. (Equation 2.1)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = M \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

where:

$$M = \begin{bmatrix} S_r X_r & S_g X_g & S_b X_b \\ S_r Y_r & S_g Y_g & S_b Y_b \\ S_r Z_r & S_g Z_g & S_b Z_b \end{bmatrix} \tag{2.1}$$

$$X_r = x_r / y_r \qquad X_g = x_g / y_g \qquad X_b = x_b / y_b$$

$$Y_r = 1 \qquad Y_g = 1 \qquad Y_b = 1$$

$$Z_r = (1 - x_r - y_r)/y_r \qquad Z_g = (1 - x_g - y_g)/y_g \qquad Z_b = (1 - x_b - y_b)/y_b$$

$$\begin{bmatrix} S_r \\ S_g \\ S_b \end{bmatrix} = \begin{bmatrix} X_r & X_g & X_b \\ Y_r & Y_g & Y_b \\ Z_r & Z_g & Z_b \end{bmatrix}^{-1} \begin{bmatrix} X_W \\ Y_W \\ Z_W \end{bmatrix}$$

Unfortunately, the flip answer to the question is "whatever you want them to be." Cameras have spectral sensitivities, and spectral sensitivities can be expressed as a set of tristimulus values relative to any set of primaries, real or imaginary. Take for example the LMS cone fundamentals. Those can be transformed to CIE XYZ tristimulus values. Or tristimulus values corresponding to RIMM RGB.[Spaulding et al. 2000] Or even Rec.709.[ITU BT.709-6 2015] Or the ACES AP0 primaries.[SMPTE ST2065-1 2021] But the LMS cone fundamental are layer combinations of color matching functions, so there is one, and only one, easily calculable transform. But with cameras, the Luther condition is not satisfied, so the transform must be optimized.

In order to optimize the camera characterization transform, a few prerequisites need to be met. These conditions are often easily achieved with high end motion picture cameras, but lower end cameras may not produce proper data required for characterization.

- Establish training data
- Determine aim values such as CIE XYZ, ACES for training data
- Determine camera values (e.g. Linear Camera RGB) to be transformed
- Calculate the transformation(camera values to aim values)

2.1.3 Camera Characterization

There are two basic methods we can use to implement these steps

2.1.3.1 Physical Chart Measurements

Using physical charts, we can capture the aim XYZ values of a chart as it exists in from of a camera. We can do this by taking a picture of the chart. This data can be used to optimize the RGB to XYZ transform. The hardware requirements for this approach are relatively low which means it can be done in the field with turnkey software. However, the results are often subpar.

- Highly prone to measurement errors
- Charts contain limited number of samples often with similar principal components in each sample
- Limited to characterization with physically available light sources
- Linear Camera RGB values are required by may not be available from the camera

2.1.3.2 Spectral Modelling

A much better way to generate characterization transforms is with spectral modelling. With spectral modelling, hundreds, or thousands of *in situ* scene objects can be lit mathematically and can be captured with a virtual camera. The results tend to be much more robust, though the equipment is quite expensive. Advantages are

- Flexibility to change training spectra (spectral reflectances and light source SPDs) and aim values (CMFs)

- Camera measurements done in well controlled environment
- Generate transforms for may light sources quickly

Disadvantages are

- Expensive equipment required
- Linear Camera RGB values are required by may not be available from the camera

2.1.3.3 Optimization Methods

The actual process of optimization happens through minimization of differences between aim values and camera values passed though the candidate transform. When using spectral modelling, it is trivial to optimize parameter such as the exact training data used, and the weights applied to the training samples. In all cases though, the choices of the error minimization space will affect the outcome.

2.1.3.4 Types of Transform

The type of transform generated can vary as well. Typically linear transforms such as 3x3 matrices are used. If the camera spectral sensitives satisfied the Luther condition, optimization wouldn't be necessary, and the resulting transform would have no error. As the camera deviates from the Luther condition, the error increases. This often leads to the suggestion that non-linear transforms such as polynomials and 3D LUTs could be useful in camera characterization. Unfortunately, these types of transforms have some major drawbacks including unknown or undefined results for results outside the training set, and a lack of exposure invariance. 3x3 matrices maybe try to fit a square peg into a round hole, but they are well-behaved when encounter a scene spectrum not included in the training set. However, there is research going on in the improvement of the characterization [McElvain and Gish 2013], [Finlayson et al. 2015], [Finlayson and Johnson 2016]

2.2 Practice

With the theory now introduced, the practice of characterizing cameras spectrally will now be introduced. In this section, the methods of measurement and how to evaluate the accuracy of those measurements will be explained. Finally, the spectral data will be used to demonstrate their role in a color-managed system. Spectral characterization of a camera system is useful for:

- accurately simulating images as captured by a particular camera
- optimizing a transform to map camera RGB to a defined encoding space

2.2.1 Measurement

There are a few different methods to obtain spectral sensitivity data for a camera system.

The simplest method is to obtain the data directly from the camera system manufacturer. This is the ideal method whenever it is possible. Camera manufacturers should know the

constitution of their filter arrays and have the best knowledge and control over their camera's signal processing. Unfortunately, it is uncommon for manufacturers to publish spectral sensitivity data for their camera systems. And, even if they did, its usage would require trust that the data would be accurate.

Fortunately, even when camera manufacturers do not publish their data, it is still possible to measure the spectral sensitivities through careful capture of specific imagery coupled with appropriate image processing. Some enterprising companies even sell black-box products designed for off-the-shelf use to quickly derive camera spectral information (e.g. camSPECS from Image Engineering).

At the Academy, a custom-built rig consisting of a software-driven monochromator, power meter, and integrating sphere is used (Figure 2.1).

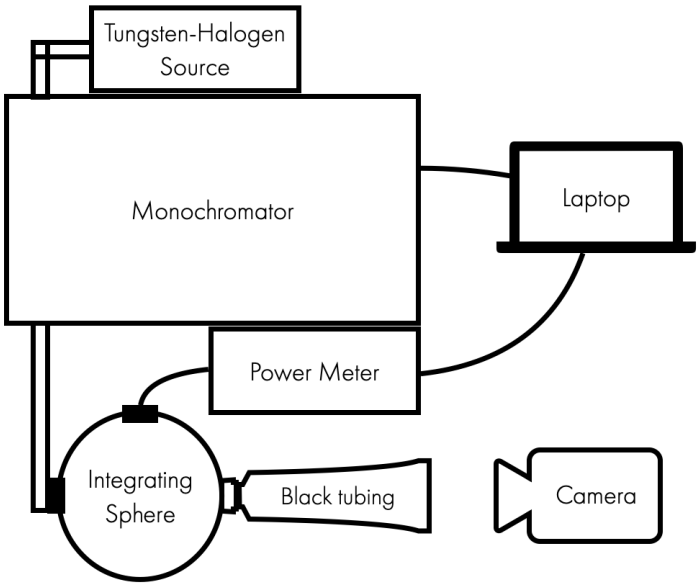


Figure 2.1: A diagram of the monochromator rig.

2.2.1.1 Using a monochromator

The monochromator filters narrow wavelengths from a tungsten-halogen light source. The monochromatic light is fed into an integrating sphere that has two exit ports. One port is connected to a power meter that feeds back to the computer controlling the system. The other port is where the camera is positioned to photograph the monochromatic light.

The monochromator is programmable at any wavelength increment. We usually use 350nm to 800nm at 5nm increments, which seems to be a "Goldilocks" range - not too much data so as to be overwhelming and not so sparse as to miss important bumps in spectral curves. The range 350nm-400nm and 700nm-800nm are beyond where camera senstivity curves usually reach, but are important to evaluate to check the camera's UV and IR response.

The computer sequences the monochromator across the selected range and at each wavelength increment pauses and captures a reading from the power meter into a text file. An

image of the monochromatic light inside the integrating sphere is captured at each wavelength, as in Figure 2.2.

The camera should be set to record in RAW or the closest equivalent.

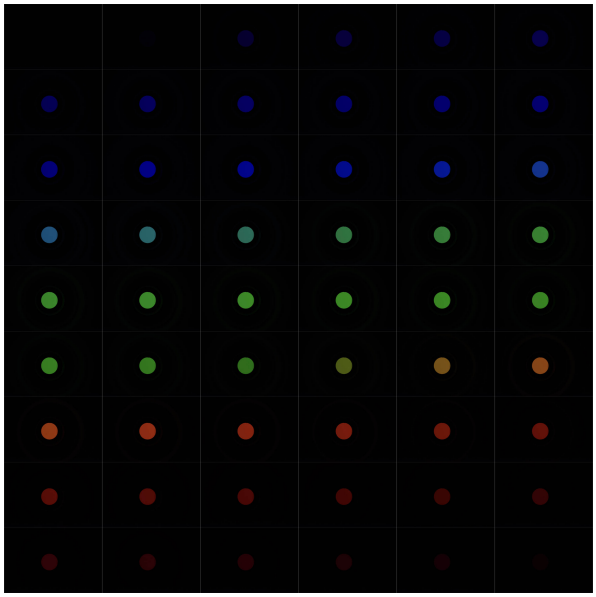


Figure 2.2: Contact sheet of monochromator “spot" images captured at each wavelength increment.

2.2.1.2 Data to capture

To derive the spectral sensitivities, the following items should be captured:

- an image at each of the selected wavelengths (e.g. Figure 2.2)
- the radiant power reading at each of the selected wavelengths
- spectral transmission of the camera lens
- a black frame, with the lens cap on and camera covered

For validation of the spectral sensitivities, the following should be captured:

- spectral reflectance of each color patch if using a reflective chart (e.g. ColorChecker 24), and/or
- spectral transmission for each color patch if using a transmissive chart (e.g. TE226)
- spectral power distribution of the light source used to illuminate the test chart(s)
- spectral transmission of the camera lens
- a black frame, with the lens cap on and camera covered

2.2.1.3 Processing the data

All images - the monochromator spot images and captured test charts - are black-frame subtracted, demosaiced, linearized and white balanced to obtain in linear camera native RGB.

Then, a region of RGB values from the center of each spot image is sampled and averaged (Fig. 2.3, resulting in an RGB response that corresponds to each wavelength increment, as depicted in Figure 2.4.

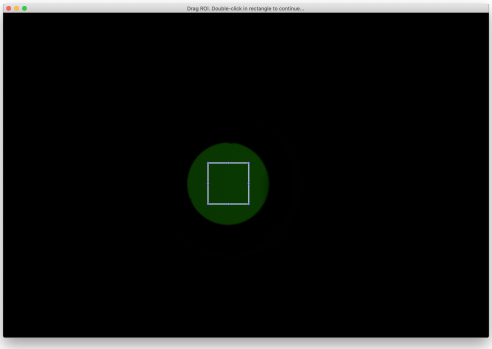


Figure 2.3: An example sampling region for a monochromator spot image.

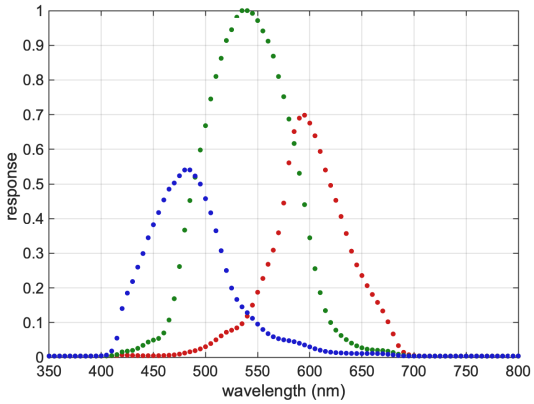


Figure 2.4: Averaged RGB response, plotted per wavelength.

The RGB response can be described by Equation 2.2, where R_i is the response for each channel i, $P(\lambda)$ is the radiant power, $SS_i(\lambda)$ is the spectral sensitivity for each channel i, and $T_{lens}(\lambda)$ is the spectral transmission of the lens.

$$R_i(\lambda) = P(\lambda)SS_i(\lambda)T_{lens}(\lambda) \tag{2.2}$$

Therefore, to get from RGB response to spectral sensitivities, the radiant power (Fig. 2.5 and lens transmission (Fig. 2.6) must be factored out. Equation 2.3 is the result of rearranging Equation 2.2 to solve for SS_i.

$$SS_i(\lambda) = \frac{R_i(\lambda)}{P(\lambda)T_{lens}(\lambda)} \tag{2.3}$$

The light source to the monochromator is tungsten-based. Therefore, radiant power is low in the blue wavelength region and much higher in the red wavelength region. The relatively low power in the blue wavelengths results in very low signal for spot images in those wavelengths. Signal-to-noise in those wavelengths could potentially be improved by filtering the tungsten light source to achieve a more uniform power spectrum. For stills cameras, one could also increase exposure time for the images captured in that region, but in general it is easiest to set the camera for a single exposure so that the most responsive wavelengths do not clip on the sensor.

Note that while factoring out the lens transmission is important to get the true spectral sensitivities of the camera sensor itself, the lens was present during photography of the test charts and so will need to be factored in during the prediction of RGB exposure in the evaluation step.

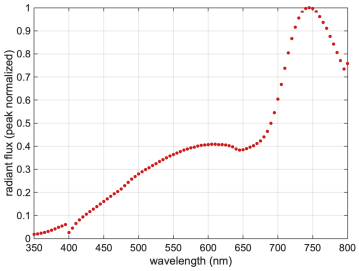


Figure 2.5: Measured power

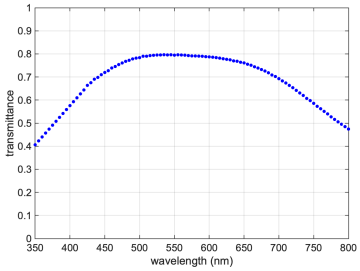


Figure 2.6: Lens transmission

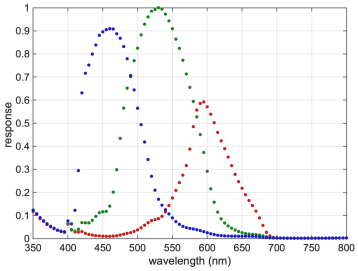


Figure 2.7: Spectral sensitivities

2.2.2 Evaluation

2.2.2.1 Linearity

The performance of the optimized matrix depends on having linear data. It is a good idea to use the bracketed exposures of the test chart(s) to confirm the RAW images were properly linearized. It is trivial to compare RGB values for a particular test patch to the same values from an image one stop different. A one stop difference should show a doubling or halving of values.

Linearity can also be confirmed visually by plotting RGB values vs. stops on a log-log plot.

2.2.2.2 Predicted vs Actual

Once linearity is confirmed, the spectral sensitivity curves can be evaluated by using them to calculate predicted RGB values. This is done according to Equation 2.4, where E_i is the predicted value for each channel i, $I(\lambda)$ is the spectral power distribution of the light source used to illuminate the test chart, $R(\lambda)$ is the spectral reflectance or transmission of each test patch on the test chart, $T_{lens}(\lambda)$ is the spectral transmission of the lens, and $SS_i(\lambda)$ is the spectral sensitivity for each channel i.

$$E_i = K_i \sum_{\lambda} I(\lambda)R(\lambda)T_{lens}(\lambda)SS_i(\lambda)d\lambda \tag{2.4}$$

Because the measurements are all relative spectra, the scale factor K_i is calculated and used to balance on one of the test patched, usually a middle gray.

The predicted RGB values can be compared to the RGB values of the test chart as actually photographed. The comparison is shown graphically in Figure 2.8 and visually in Figure 2.9. Colors in Figure 2.9 appear washed out because image has not yet been rendered for display.

2.2.3 Application

2.2.3.1 Generating an ACES Input Transform

Once the spectral sensitivity data has been demonstrated to be accurate and camera capture can be accurately modeled, the spectral sensitivity data is used to optimize a linear transformation to colorimetry.

An ACES Input Transform generation optimizes a mapping of real, physical cameraas sensitivities to a theoretical colorimetric camera known as the ACES Reference Input Capture

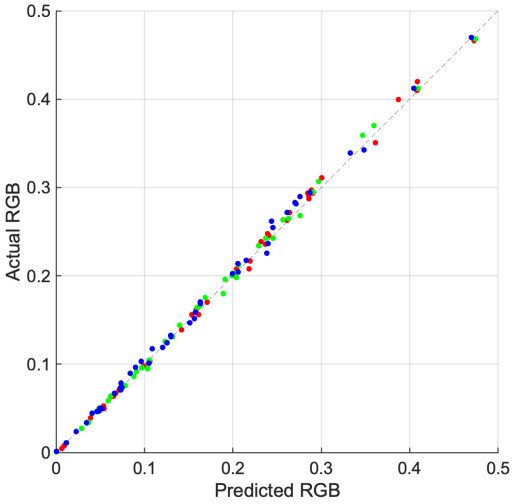


Figure 2.8: Plot of predicted RGB vs actual RGB.

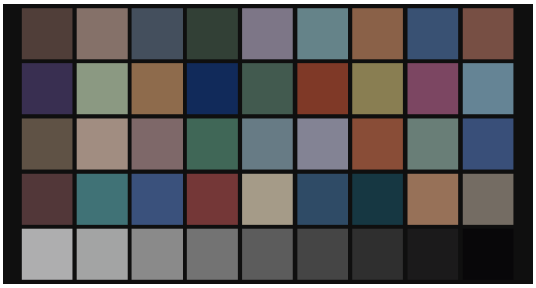


Figure 2.9: Top half of each patch shows actual RGB, the bottom half shows predicted RGB.

Device (RICD). Academy Procedure document *P-2013-001* [Academy 2013] describes how to optimize a 3×3 matrix and scale factors for a given illuminant. There are many parameters and design decisions that can be made, such as:

- training spectra
- neutral chromaticity differencne compensation strategy
- error minimization color space
- error metric (cost function) that is minimized

To demonstrate, the recommended 'defaults' from *P-2012-001* were used. The result is an optimized 3×3 matrix and scale factors for a 3200K blackbody radiator. Applying this IDT to the linearized camera RGB converts camera RGB to ACES2065-1 [SMPTE ST2065-1 2021]. That ACES2065-1 image can then be rendered to output using regular transforms that ship with the Academy Color Encoding System. See Figure 2.10.

Figure 2.10: Applying an ACES Input Transform and Output Transform.

2.2.3.2 Comparing multiple cameras

Repeating the measurement, evaluation, and IDT generation process for additional cameras allows us to compare results from camera to camera. Figure 2.11 shows four different

cameras that have been spectrally characterized, had 'tungsten' (3200K) IDTs generated and applied, and processed through an ACES Output Transform to sRGB.

Figure 2.11: Comparison of 4 cameras that have been characterized.

Using the spectral sensitivities of the ACES RICD, we can predict the colors that the RICD would capture, if it actually existed. Figures 2.12 and 2.13 add the RICD to the center of the results from the four cameras shown previously. The colors from the RICD clearly distiguishable from each the respective camera results. However, this is to be expected, because cameras are *not* colorimeters. If cameras were more colorimetric, the cameras would match the RICD results more closely.

Cameras can, however, be characterized spectrally and thus accurately modeled. They can also be transformed into a similar state so that they can be matched within a color-managed workflow. Even with their inherent limitations, 3/*times*3 matrices do a respectable job in bringing cameras with different spectral sensitivities into alignment with each other. This is crucial for establishing and maintaining color-managed workflows in motion picture production.

Figure 2.12: TE226 chart - RICD (center) and 4 cameras (perimeter)

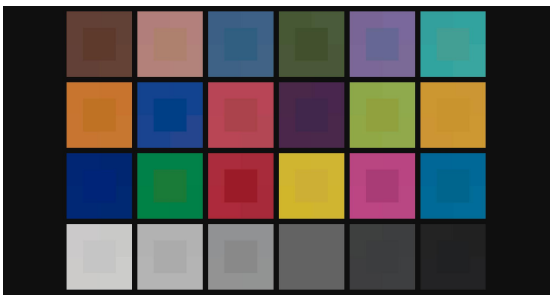


Figure 2.13: ColorChecker chart - RICD (center) and 4 cameras (perimeter)

References

Academy 2013. *Procedure P-2013-001 - Recommended Procedures for the Creation and Use of Digital Camera System Input Device Transforms (IDTs)*. Procedure. The Academy of Motion Picture Arts and Sciences, Science and Technology Council, & Academy Color Encoding System (ACES) Project Subcommittee. http://j.mp/P-2013-001

Graham D Finlayson and Garrett M Johnson. 2016. Extended Linear Color Correction. In *Color and Imaging Conference*, Vol. 2016. Society for Imaging Science and Technology, 168–173.

Graham D Finlayson, Michal Mackiewicz, and Anya Hurlbert. 2015. Color correction using root-polynomial regression. *IEEE Transactions on Image Processing* 24, 5 (2015), 1460–1470.

ITU BT.709-6 2015. *Parameter values for the HDTV standards for production and international programme exchange*. Recommendation. International Telecommunications Union.

Hideyasu Kuniba and Roy S Berns. 2009. Spectral sensitivity optimization of color image sensors considering photon shot noise. *Journal of Electronic Imaging* 18, 2 (2009), 023002.

Jon S McElvain and Walter Gish. 2013. Camera color correction using two-dimensional transforms. In *Color and Imaging Conference*, Vol. 2013. Society for Imaging Science and Technology, 250–256.

Noboru Ohta and Alan R Robertson. 2005. *Colorimetry: Fundamentals and Applications*. Vol. 10. John Wiley & Sons, Ltd, Chichester, UK., Chapter Measurement and calculation of colorimetric values, 153–174. https://doi.org/10.1002/0470094745.ch5

SMPTE ST2065-1 2021. *Academy Color Encoding Specification (ACES)*. Standard. Society of Motion Picture and Television Engineers.

Kevin E Spaulding, Geoffrey J Woolfe, and Edward J Giorgianni. 2000. 'Reference input/output medium metric RGB color encodings. *Proc. IS&T PICS* (2000), 155–163.

A Stockman and Brainard. 2010. *The Optical Society of America Handbook of Optics, Volume III: Vision and Vision Optics*. Vol. Volume III: Vision and Vision Optics. McGraw Hill, New York, Chapter Colorimetry.

Andrew Stockman, Donald I. A. MacLeod, and Nancy E. Johnson. 1993. Spectral sensitivities of the human cones. *J. Opt. Soc. Am. A* 10, 12 (Dec 1993), 2491–2521. https://doi.org/10.1364/JOSAA.10.002491

R Von Luther. 1927. Aus dem gebiet der farbreizmetrik. *Zeitschrift fur Technishe Physik* 12 (1927), 540–558.

G. Wyszecki and W.S. Stiles. 2000. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley.

3

Applied Spectral Knowledge in Virtual Production

THOMAS MANSENCAL, *Weta Digital*

3.1 Introduction

LED wall-based virtual productions face various colour rendering and reproduction challenges. Spectral analysis can provide an in-depth understanding while contributing to design solutions around them. The first section, skips over the colour science fundamentals, but refreshes knowledge about colour quality and colour imaging systems. Then, in the second section, we will focus on the LED wall, and show how it is an atypical colour imaging system. We will also discuss its colour quality. Finally, we will focus on the camera, the observer of the scene.

3.2 Fast Forward » Colour Science

3.2.1 Colour Quality: Not All Lights Are Created Equal

In contrast with the general smoothness of the spectral reflectances of natural surfaces, the spectral power distribution of emission sources tends to vary with high frequency, affecting colour quality (or rendering).

Colour rendering is defined by [CIE [1987]] as the "effect of an illuminant on the perceived colour of objects by conscious or subconscious comparison with their perceived colour under a reference illuminant."

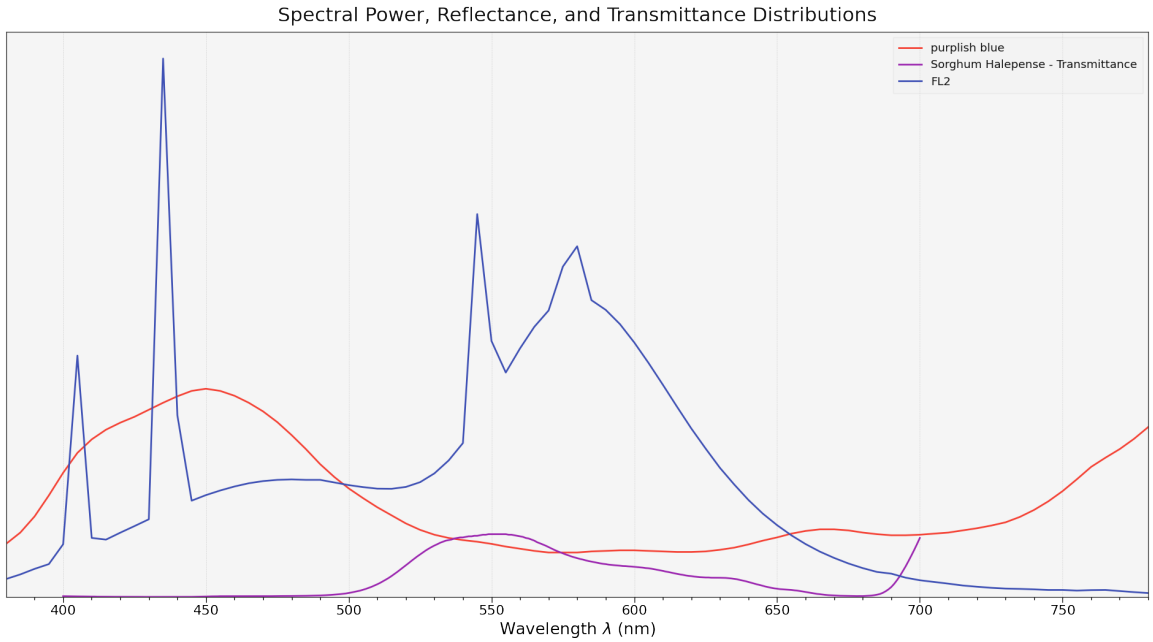


Figure 3.1: Normalised spectral power distribution of a fluorescent light source compared with the reflectance, and transmittance distributions of a colour checker swatch, and some grass. Note the emission line spikes of the fluorescent light atomic elements.

Colour quality is quantified by three main metrics:

- The CIE standardizes two primary metrics, the Colour Rendering Index (*CRI*) [CIE Division 1 [1995]] and the Colour Fidelity Index (*CFI*) [CIE TC 1-90 [2017]].
- ANSI and IES specify TM-30-18 [ANSI and IES Color Committee [2018]].

Research has also produced two other notable metrics:

- The Colour Quality Scale (*CQS*) [Davis and Ohno [2010]]
- The Academy Spectral Similarity Index (*SSI*) [The Academy of Motion Picture Arts and Sciences [2019]].

3.2.1.1 Colour Rendering Index

CRI measures the degree to which a non-incandescent light source can mimic the faithful rendering of standard tungsten and daylight illuminants. *CRI* spectral uniformity $(r')^2$, also called spectral flatness, is mediocre because of the wavelength bias present in the test sample set [Smet et al. [2016]]. The metric is computed using eight test colour samples, thus, it is possible to tune the light spectrum to yield an anomalously high colour rendering index R_a without improvement of perceived colour fidelity.

Spectral uniformity $(r')^2$ is computed as follows:

$$mean((r'_1)^2, (r'_2)^2, ..., (r'_n)^2) \tag{3.1}$$

where $(r'_i)^2$ is the first derivative, squared, of the reflectance r_i of a test sample.

3.2.1.2 Colour Fidelity Index

CFI was designed to address the shortcomings of *CRI* with narrowband emission sources, especially solid-state lighting. The *CFI* spectral uniformity $(r')^2$ is improved over that of the Colour Rendering Index. Ninety nine test colour samples were selected across thousands to exhibit better spectral uniformity. As a consequence, a high colour fidelity value R_f is strongly correlated with a high perceived colour fidelity.

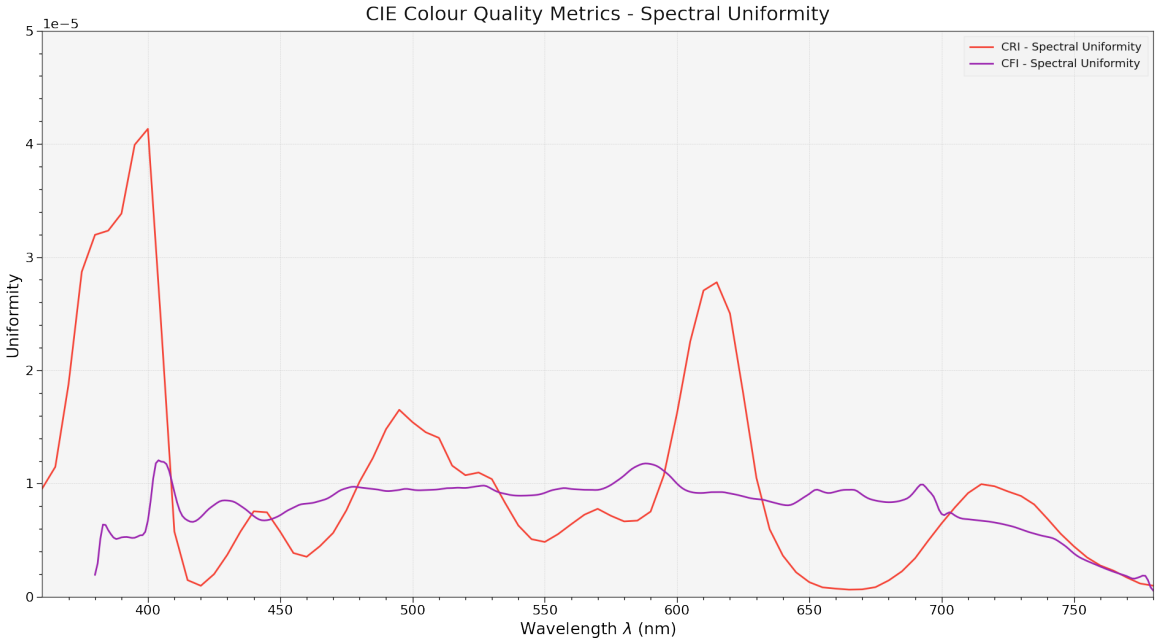


Figure 3.2: Spectral uniformity of the *CRI* and *CFI* metrics compared. *CFI* exhibits much less broad peaks and valleys.

3.2.1.3 TM-30-18

ANSI/IES *TM-30-18* extends *CFI* with a standardized visual report. The report is helpful to assess hue specific properties such as the gamut area in the middle-left section or chroma and hue shifts in the upper-right section. It also presents the colour fidelity of each test colour sample in the bottom section.

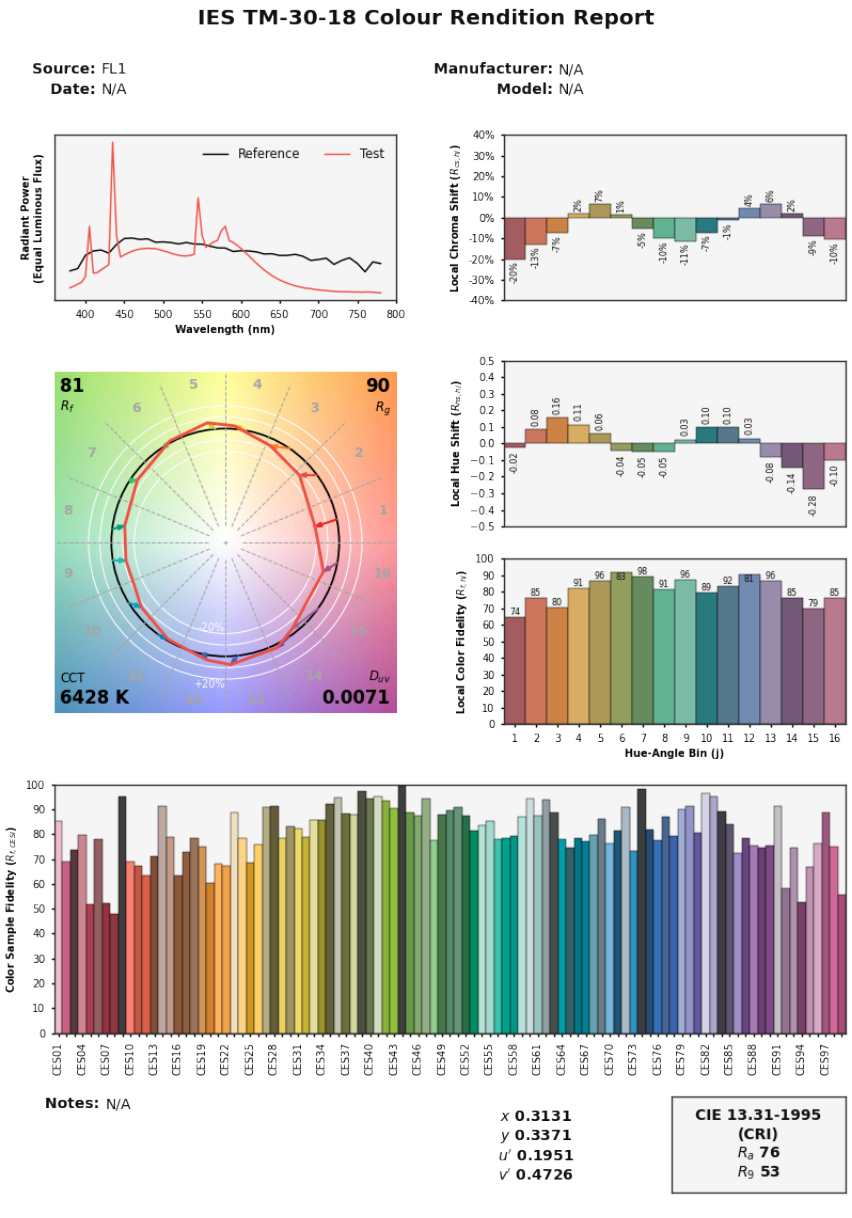


Figure 3.3: ANSI/IES TM-30-18 report for FL1 illuminant.

3.2.1.4 Spectral Similarity Index

SSI removes the human observer from the colour quality metric and is only concerned to quantify the similarity of a test light source spectral power distribution (*SPD*) to that of another.

3.2.2 Colour Imaging Systems

A colour imaging system (*CIS*) is any combination of devices and/or media able to perform:

- Image Capture
- Signal Processing
- Image Formation

3.2.2.1 Image Capture

This section will explicitly focus on cameras. Image scanners, for example, are also image capturing devices but not relevant for the course.

Consumer and movie cameras try to be colorimetric by satisfying the Maxwell-Ives criterion: Their sensitivities must be a linear combination of the *Standard Human Observer* colour matching functions to reduce metameric failures. However, no consumer or motion picture camera is truly colorimetric: The red sensitivity is offset to longer wavelengths to reduce the noise that would occur with a peak centered with that of the L cones.

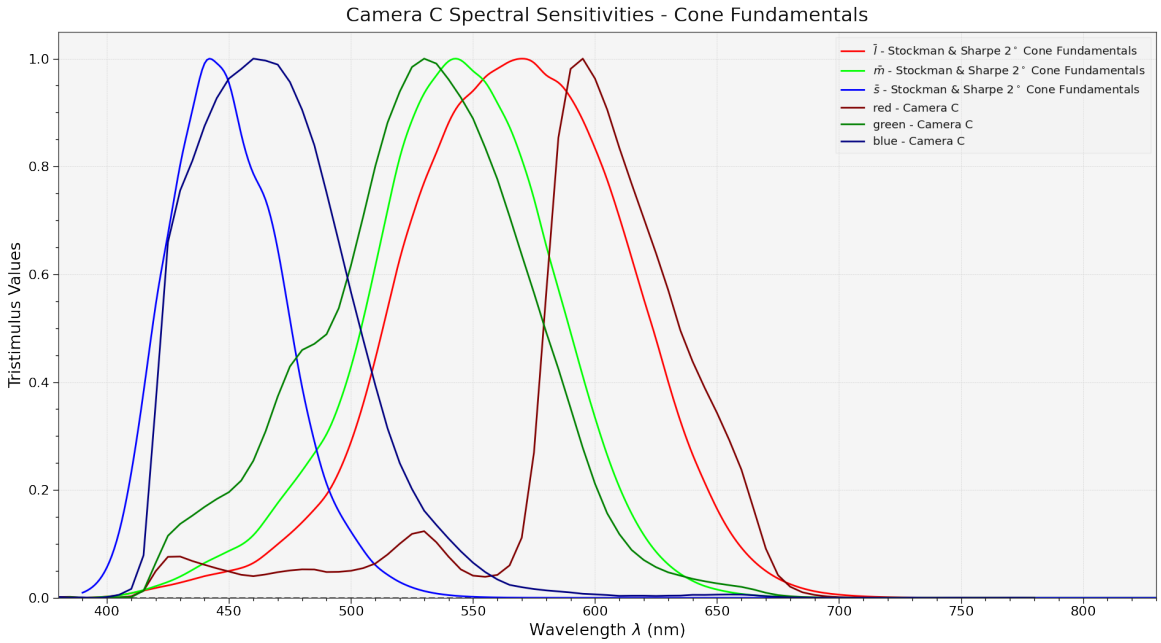


Figure 3.4: Camera C spectral sensitivities compared with the Standard Human Observer cone responses. Note how the camera red sensitivity is significantly offset to the right.

3.2.2.2 Signal Processing

The signal processing of a raw image may include, in varying order, the following main steps:

- Linearization
- Black Level Subtraction
- White Level Scaling

- Clipping
- White-Balancing
- Demosaicing
- Conversion to working RGB Colourspace

Of interest for this course are white-balancing and the conversion to working RGB colourspace. White-balancing is akin to the human visual system chromatic adaptation function. Chromatic adaptation controls the independent sensitivity of the three cone cell types of the human eye. Similarly, white-balance controls the gains of the individual camera colour channels.

After demosaicing, the white-balanced image is mapped from its native space, usually named *Camera RGB*, to the desired reference or working RGB colourspace.

Procedure *P-2013-001* describes methods for the creation and use of camera input device transforms (*IDT*) [The Academy of Motion Picture Arts and Sciences et al. [2015]]. The transformation that maps *Camera RGB* to *ACES2065-1*, i.e. the ACES reference colourspace, is computed by minimizing the error between the camera white-balanced RGB values in response to a set of test stimuli and the RGB values expected from the *Standard Human Observer* for the same set of test stimuli.

3.2.2.3 Image Formation

The majority of modern electronic display devices use an additive image formation process. The processed image signals are used to drive the display device primaries. The production of the intended colour sensation for the observer is achieved through metamerism. The calibration of a display to achieve device-independent colorimetry requires a mapping, e.g. LUT, from input code values to the desired CIE XYZ tristimulus values.

3.3 The LED Wall: A Window Into Virtual Worlds

Desiderata

It would be desirable for the LED wall to act as a window into virtual worlds:

- It would have an infinite dynamic range, for example, that of our planet, following a linear relationship with the virtual world.
- It would be equipped with a spectral engine to model any spectral power distribution faithfully.
- Finally, it would produce predictable and device-independent colorimetry to facilitate the onset crew operation of the camera and light fixtures. The production of expectable colour temperatures is deemed critical because the onset crew typically uses *Standard*

Human Observer calibrated devices, e.g. colorimeters. Their work should be as efficient as possible and not wasted in complicated colorimetric conversions when they need to change gel filters to mimick the LED wall lighting.

Reality

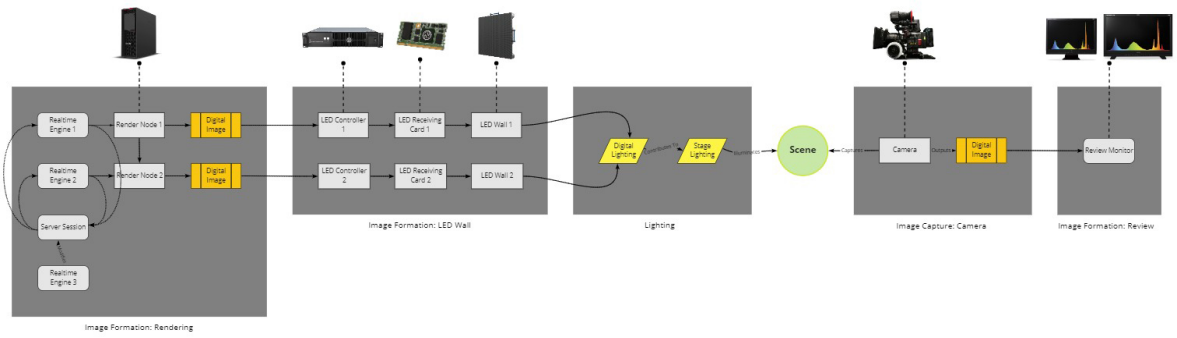
Unfortunately, the reality is much different.

- The LED wall peak luminance, flare and stray light pollution limit its dynamic range.
- The LED wall is an RGB based image formation device and has no spectral modeling capabilities.
- The camera, a non-colorimetric device, is the observer of the scene: This conflicts with the device-independent colorimetry desire if the LED wall is specifically corrected for the camera.

3.3.1 An Atypical *CIS*

This section will discuss about the LED wall in more concrete terms. The first important point is that the LED wall is an atypical colour imaging system. Discounting signal processing, and omitting non-critical components, a simplistic virtual production flow diagram requires three main phases:

- Image Formation - 1st Stage
- Image Capture
- Image Formation - 2nd Stage



3.3.1.1 Image Formation - 1st Stage

A projection mapping software or a rendering engine generates the imagery and encodes it for transmission by the render node graphics cards. The graphics cards transmit the signals to LED processors via HDMI or SDI cables. With knowledge of the input imagery state, desired output state, and LED panels capability, the LED processors handle the input signals, process them and transmit them to the LED receiving cards via distribution units. The LED receiving cards drive the LED panels according to their input signals.

3.3.1.2 Image Capture

A motion-picture camera captures the scene (or the stage):

- The real-life foreground is illuminated by the LED wall and the various onset fixtures, for example, HMI, fluorescent tubes, or LED lights.
- The virtual world background, exhibited on the LED wall, fills the entire camera frustum.

We can formulate three critical questions that we will try to answer:

1. What is the appropriate state of the imagery for the LED wall?
2. What is the colour quality of the LED wall?
3. How to capture the scene and the LED wall in a faithful way?

3.3.1.3 Image Formation - 2nd Stage

After capture of the scene by the camera and signal processing has occurred, image formation is straightforward. Nothing notable occurs during the second image formation stage, however, *Question 1* has critical implications that need answering.

Typically, in a scene-referred workflow, the scene data must be rendered to form an image on a display. The display rendering transform (*DRT*), designed for that purpose, usually adopts a S-shape. In the context of virtual production, and to avoid applying a S-shape *DRT* two times, in the first and second image formation stages, the traditional S-shaped *DRT* is only used during the second image formation stage, after image capture.

However, because the LED wall dynamic range is limited, a dedicated first image formation stage *DRT* is required, it commonly has those properties:

- Most of the picture rendering is kept linear.
- Highlights are rolled off to avoid clipping.
- A path-to-white transform might be implemented to desaturate the highlights gracefully while avoiding hue skews.
- Gamut mapping can be implemented if the RGB working colourspace is larger than the LED wall colourspace, for example ACEScg.
- Finally, it encodes the imagery for the LED wall characteristics.

3.3.2 LED Wall Colour Quality

In this section, we will be answering *Question 2*. *TM-30-18* shows that our window into the virtual worlds has a poor colour quality. It is not entirely surprising: the LED walls were originally designed for theatre, concert, stadium, and mall exhibition. It is a display technology whose primary objective is HDR and wide-gamut imagery presentation. It requires bright and narrowband primaries, e.g. laser or LEDs with narrow full width a half maximum. This conflicts directly with proper colour quality. Indeed, high colour fidelity is correlated with broadband lighting.

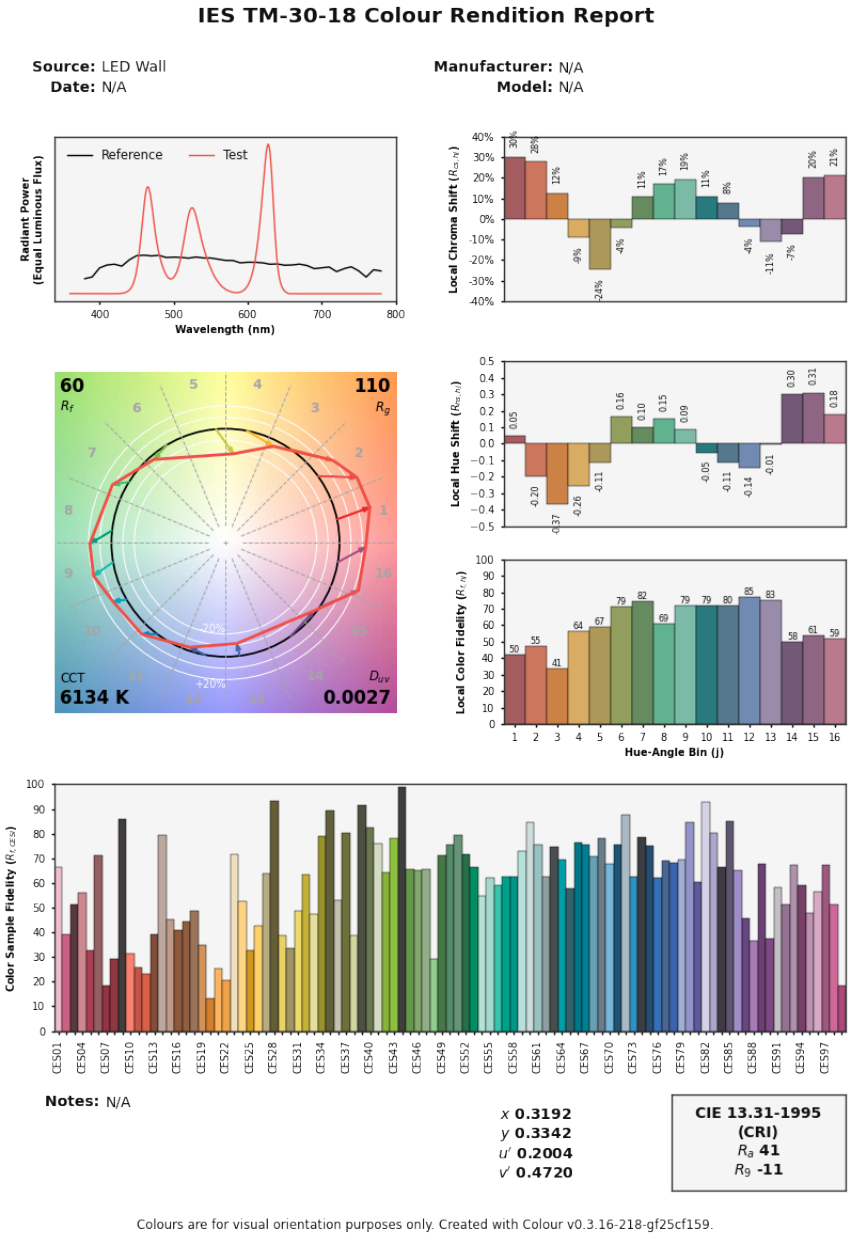


Figure 3.5: ANSI/IES TM-30-18 report for a typical LED wall.

The lack of amber spectral contribution produces a poor colour rendition of human skin and orange objects lit by the LED wall.

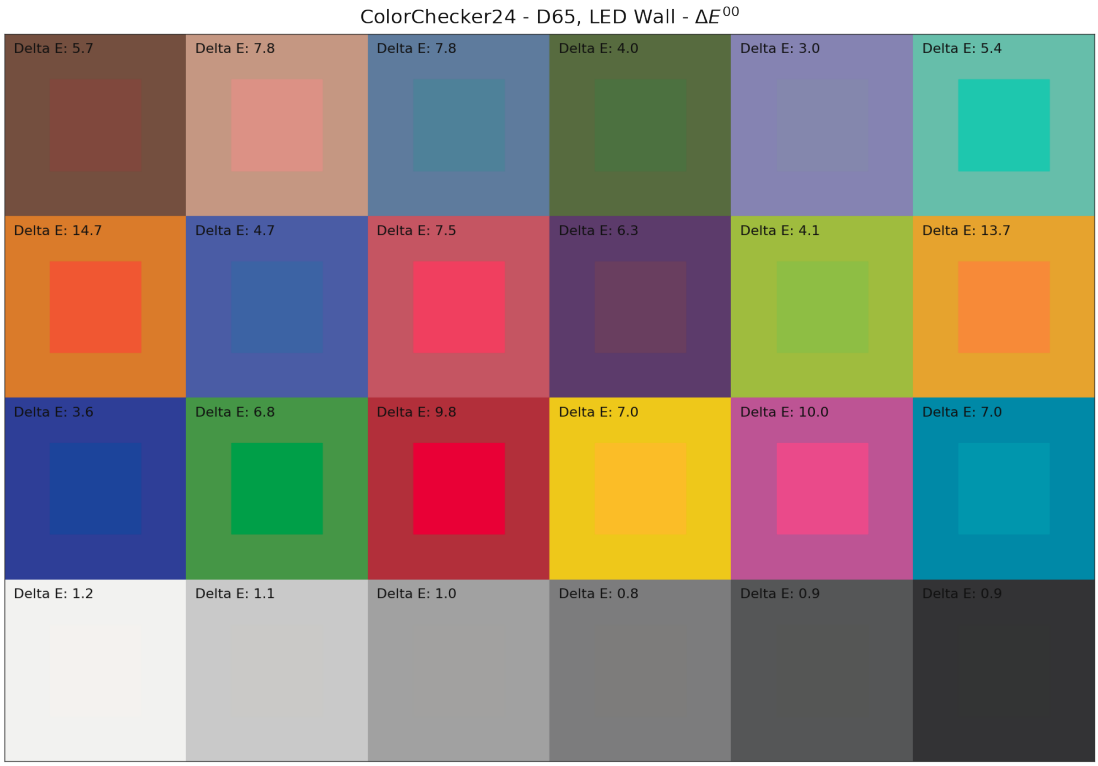


Figure 3.6: A simulated ColorChecker24 lit by an LED wall whose whitepoint is set to D65, stacked onto a simulated ColorChecker24 lit by D65 illuminant.

3.3.2.1 Spectral Gap Filling with a Normal Distribution

Can the colour quality be improved? Thankfully, working in the spectral domain puts a different perspective on the data. The LED wall spectral power distribution valleys suggest filling the spectral gap between the green and red primaries. It seems logical to use an amber normal distribution centred between the green and red peaks to test the idea.

Naively adding an amber primary skews the lighting output, thus the green and red primaries need to be adjusted to generate a metamer matching the original combination of green and red primaries CIE XYZ tristimulus values such as:

$$P_{red} + P_{green} = \alpha P_{red} + \beta P_{green} + \gamma P_{amber} \tag{3.2}$$

where α, β, and γ are scalar factors.

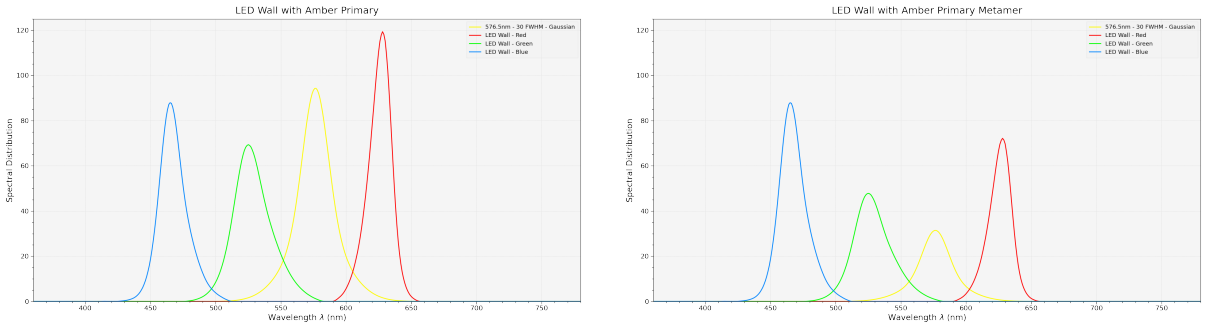


Figure 3.7: A virtual LED wall with an artificial non-scaled amber primary and the scaled metamer variant.

As seen in Figure 3.8, the introduction of an amber primary significantly improves the colour quality. In this case, the additional amber primary divided the mean CIE ΔE^{00} by 2.6 and R_f increased from 60 to 80.

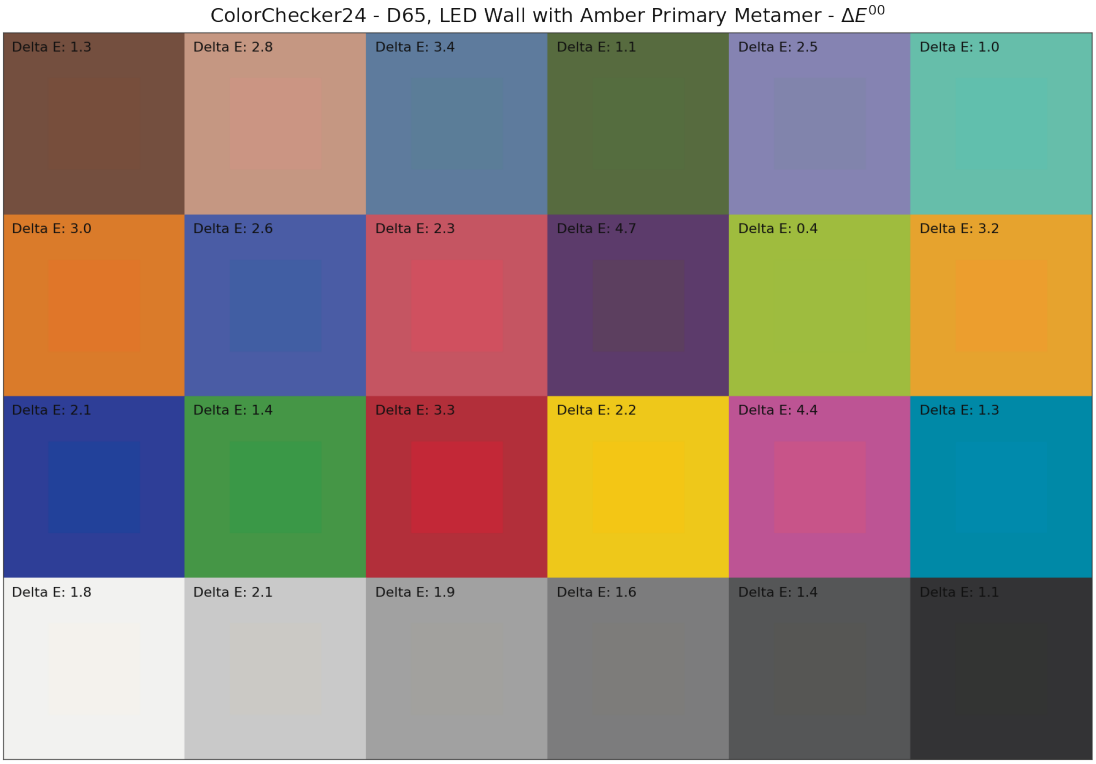


Figure 3.8: A simulated ColorChecker24 lit by an amber-boosted LED wall whose whitepoint is set to D65, stacked onto a simulated ColorChecker24 lit by D65 illuminant.

However, it is not trivial for LED panel vendors to add an extra primary as it raises the manufacturing complexity and cost. Nonetheless, the motion picture industry should highlight the importance of colour quality and fidelity in the context of virtual production. It could be a differentiating factor between LED panel vendors in the future.

Are there potential solutions to address the problem today?

- Various cinema LED lights, for example, the ARRI Orbiter, have amber LED primaries.
- LEE and Rosco also have extensive collections of gel filters.

3.3.2.2 Spectral Gap Filling with Gel Filters

A gel filter layered onto a broadband light source might produce a spectral power distribution with a profile similar to the amber primary normal distribution. We can use *SSI* to measure the spectral similarity across a filter database, for example the "LEE" filters database.

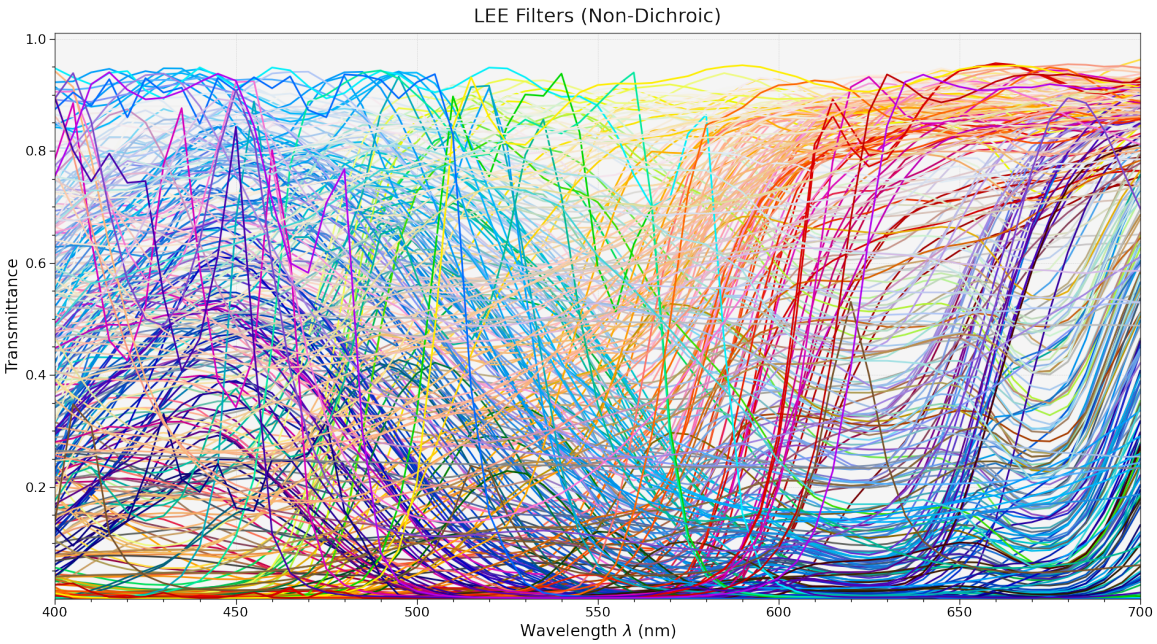


Figure 3.9: The LEE gel filter (non-dichroic) transmittance distributions.

For a given tungsten light source, the spectral similarity indices for the LEE spectral database with the amber primary are computed and ranked by highest order:

Gel Filters	SSI[Amber]
741	30.0
642	-9.0
740	-13.0
LD202	-23.0
244	-23.0

Table 3.1: The five first best SSI ranked LEE filters for the amber primary.

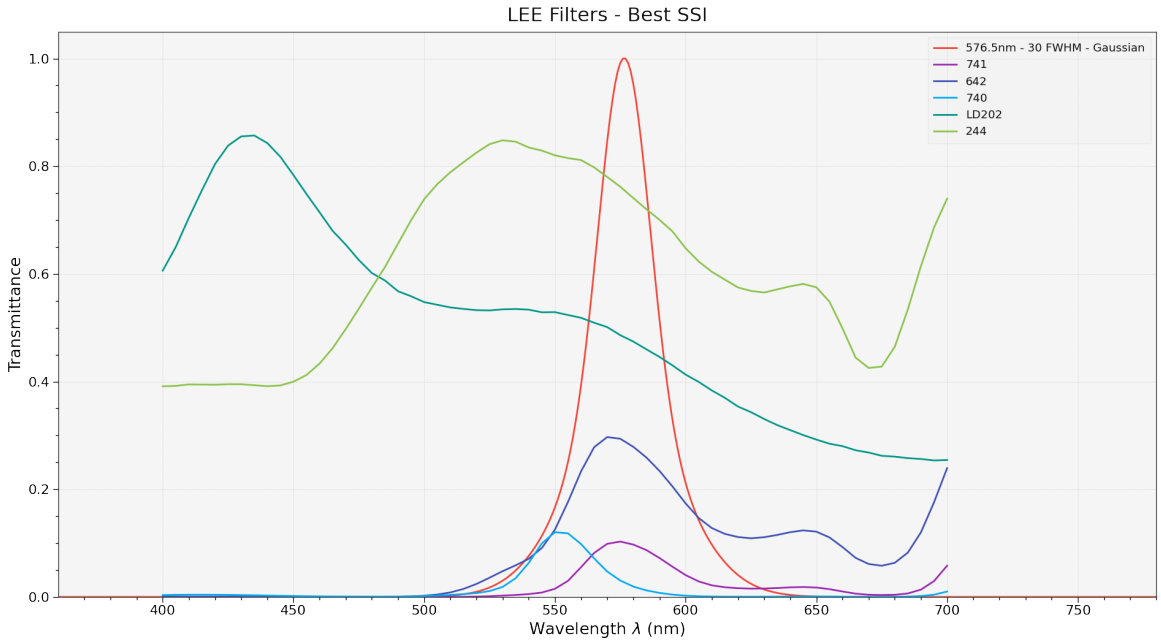


Figure 3.10: The transmittance distributions of the five first best SSI ranked LEE filters for the amber primary.

Unfortunately, the spectral similarity indices drop rapidly. However, nothing enforces that a single gel filter must be used. Given the wide transmittance variety and some constraints, a stack of gel filters might produce better results. We can then define search constraints as follows:

- x2 Stacked Gel Filters
- $SSI[Amber] > 25$
- Peak Transmittance > 15%

We find that $642 * 738$ is the best-ranked combination with a SSI of 45 and a peak transmittance greater than 15%.

Stacked Gel Filters	SSI[Amber]	Peak Transmittance
642 * 738	45.0	15.04
121 * 642	39.0	18.0

Table 3.2: The best SSI and peak transmittance ranked stacked LEE filters for the amber primary.

A stack of gel filters also significantly improves the colour quality, in this context, better than the amber primary as the mean CIE ΔE^{00} is divided by 3.1.

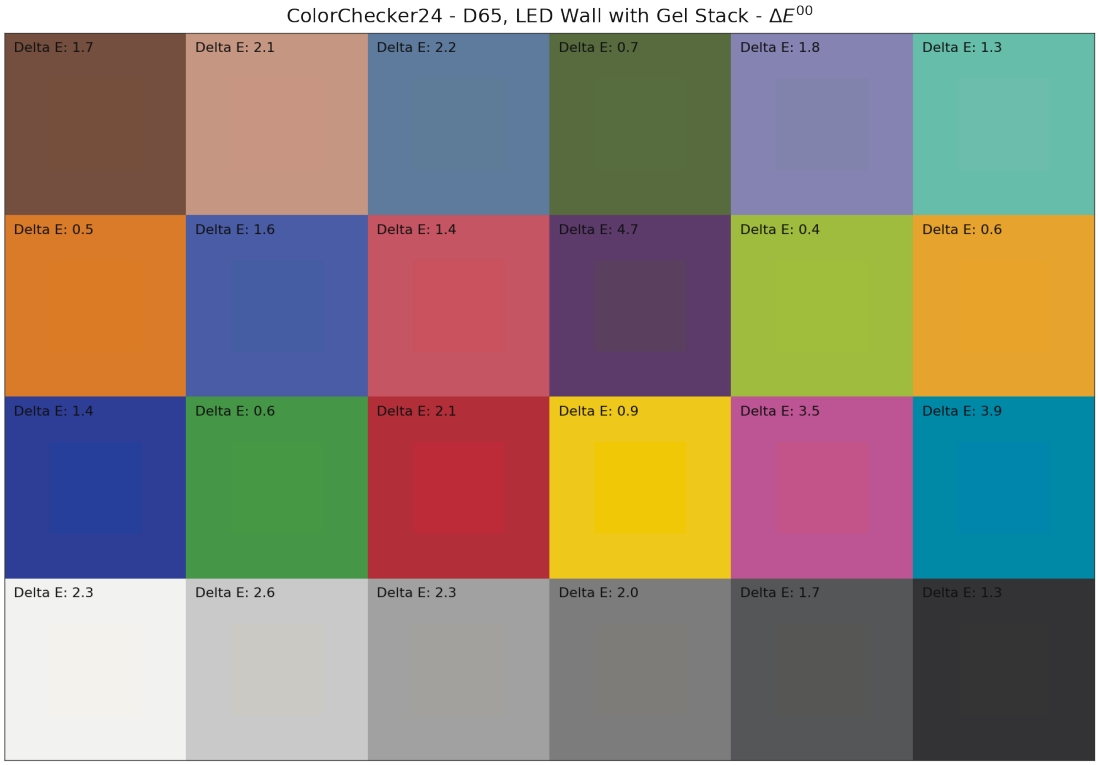


Figure 3.11: A simulated ColorChecker24 lit by a filterered tungsten light and the LED wall whose whitepoint is set to D65, stacked onto a simulated ColorChecker24 lit by D65 illuminant.

3.3.3 The Camera: Observer of the Scene

Answering the *Question 3* about capturing the scene and the LED wall in a faithful way calls for another question: For which observer should the colour transformation from the RGB working colourspace to the LED wall colourspace be computed?

- The *Standard Human Observer*?
- The camera, a non-colorimetric device, but, ultimately, the scene observer?

A motion-picture camera is commonly optimized for daylight and tungsten illuminants. The *IDT* attempts to make it colorimetric. White-balancing selects the best *IDT* for the desired image processing colour temperature. A thought experiment can then be expressed: There is typically no choice but to use a non-optimal *IDT* when processing imagery of a scene lit by fluorescent or LED lights. Nevertheless, the imagery is deemed acceptable. As we saw, the virtual production stage is lit by non-optimal lights with poor colour quality. Interestingly enough, the virtual world exhibited on the LED wall has the spectral composition of a scene lit by narrowband LEDs. Then, can the camera, without the LED wall corrected with a camera-specific transformation, produce “acceptable” imagery?

To answer this question, we need to design a process to compare how the camera reproduces a reference image exhibited on the LED wall.

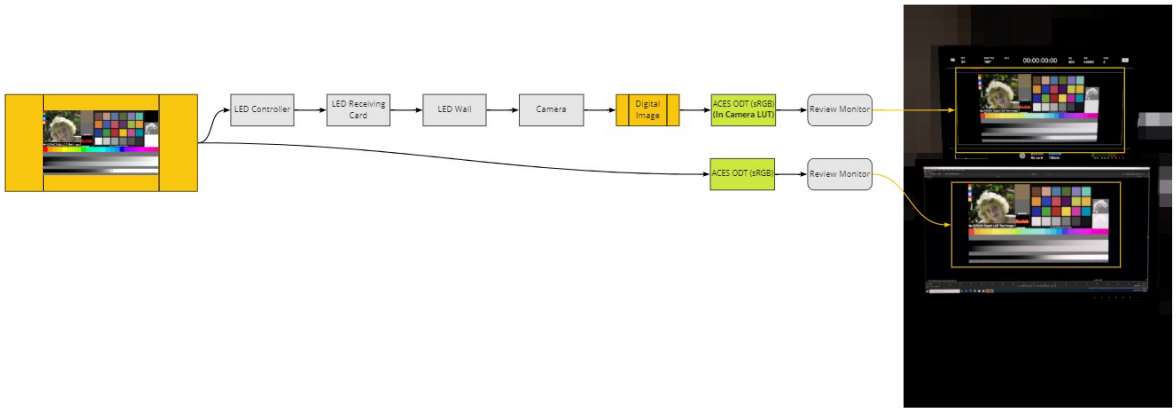


Figure 3.12: The top portion of the diagram represents the LED wall CIS, the bottom portion is dedicated to the reference image exhibition.

With the LED wall calibrated to produce device-independent colorimetry, i.e. calibrated for the *Standard Human Observer*, we can assess the reproduction of different cameras, e.g. *Camera A* and *Camera B*.

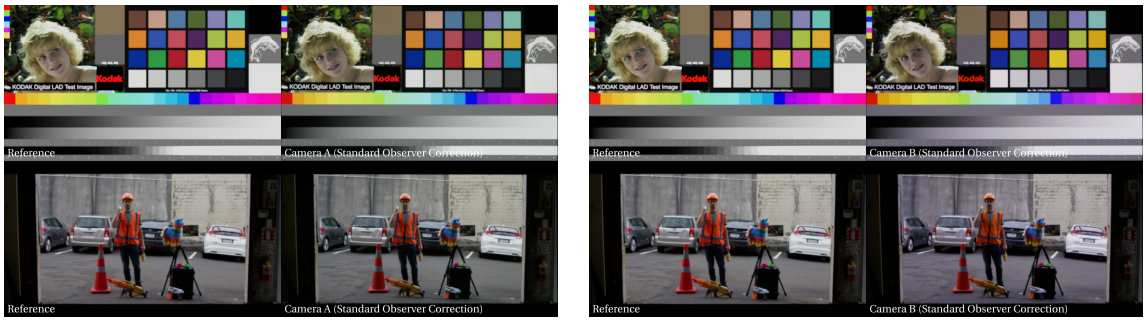


Figure 3.13: The first and third columns contain the reference images. The second and fourth columns contain their reproduction by Camera A and B, respectively.

Both *Camera A* and *B* were white-balanced to the LED wall white, which adopts the D65 whitepoint. *Camera A* measured $[6000K, -3]$, its colour reproduction is a bit greenish and slightly desaturated. *Camera B* measured $[5400K, +15]$, its colour reproduction is noticeably bluish and slightly desaturated. Neither camera capturing the LED wall footage reproduced the reference imagery correctly.

Is the imagery produced acceptable? Contextually and subjectively, it might very well be. Each camera has a *Look* imposed by its optical path, the hardware and the software processing the captured data. As seen in Figure 3.14, both cameras capturing an outdoor scene at equal white-balance produce imagery with a very different *Look*.

Figure 3.14: Camera A and B reproduction of an outdoor scene with their white-balance set to 6500K.

Often, the director (or director of photography) selects a camera body and lens combination over another for its *Look*. At *Weta Digital*, we use the camera sensitivities and the colour science of camera vendors in Manuka to be faithful to the camera colour reproduction.

Acknowledging and recognizing the camera *Look*, whether and how to match a specific reference, becomes a blend of artistic and technical decision-making. A few typical but possibly conflicting matching expectations are as follows:

- It might be required to match existing camera footage while preserving the camera *Look*.
- It is also common having to match computer graphics content.
- Sometimes, multiple cameras capturing the LED wall together must produce similar colour reproduction, in which case the camera *Look* cannot be preserved.
- Finally, it can be required to match different LED panels, which is only achievable by using the camera as the scene observer because of metameric failures.

3.3.3.1 Making the LED Wall "Wrong" So That It Looks "Right" in the Camera

On the paper, the method to make the LED wall "wrong" so that it looks "right" in the camera is simple. As a pre-requisite, we recommend having the LED wall colorimetry under control to produce device-independent colours as a baseline. The sampling process is typically performed as follows:

- The LED wall is sampled by presenting a series of codes, for example, an RGB cube, to the camera.
- After processing the samples, the $F_{Camera(LED\ wall)}$ function is solved by regression or optimization, for example, Linear Least Squares (*LLS*).
- Then the $F^{-1}_{Camera(LED\ wall)}$ function is applied to the imagery before the exhibition on the LED wall.
- Using a larger samples count, higher precision transformations such as high-order matrices or LUTs can be defined, but with the risk of eroding even more of the camera *Look*.

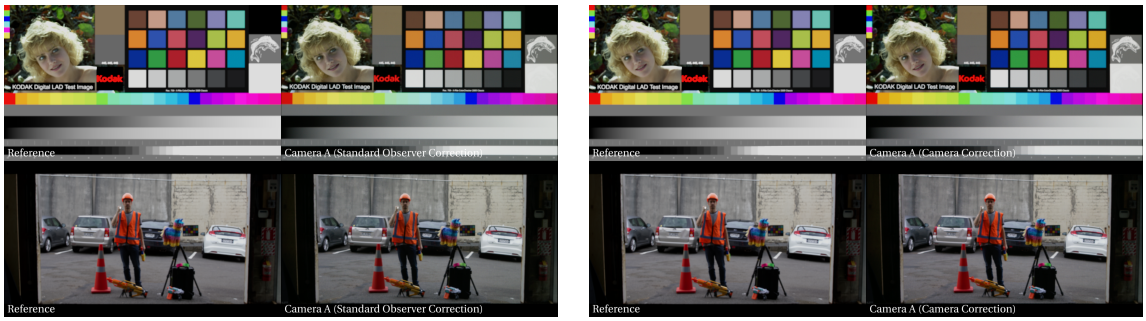


Figure 3.15: The first and third columns contain the reference images. The second and fourth columns contain their reproduction by Camera A, without and with the LED wall using $F^{-1}_{Camera(LED\ wall)}$, respectively.

A naive colour-correction removes the camera *Look* but not only: upon correction, a strong camera *Look* biases the LED wall colour reproduction, impacting colorimetry and the onset crew work negatively. For example, an inconsiderate $F^{-1}_{Camera(LED\ wall)}$ function would correct *Camera B*'s warmth by emitting bluish tristimulus values at the display. Unfortunately, this is not the only problem because there is an elephant in the room: the *white-balance*.

An *IDT* produced according to *P-2013-001* is optimized for a given white-balancing colour temperature. The resulting $nx3$ matrix defines the basis of an implicit RGB colourspace that changes as a function of colour temperature. This can be observed in Figure 3.16, where the implicit RGB colourspaces for *Camera C* are plotted for various colour temperatures.

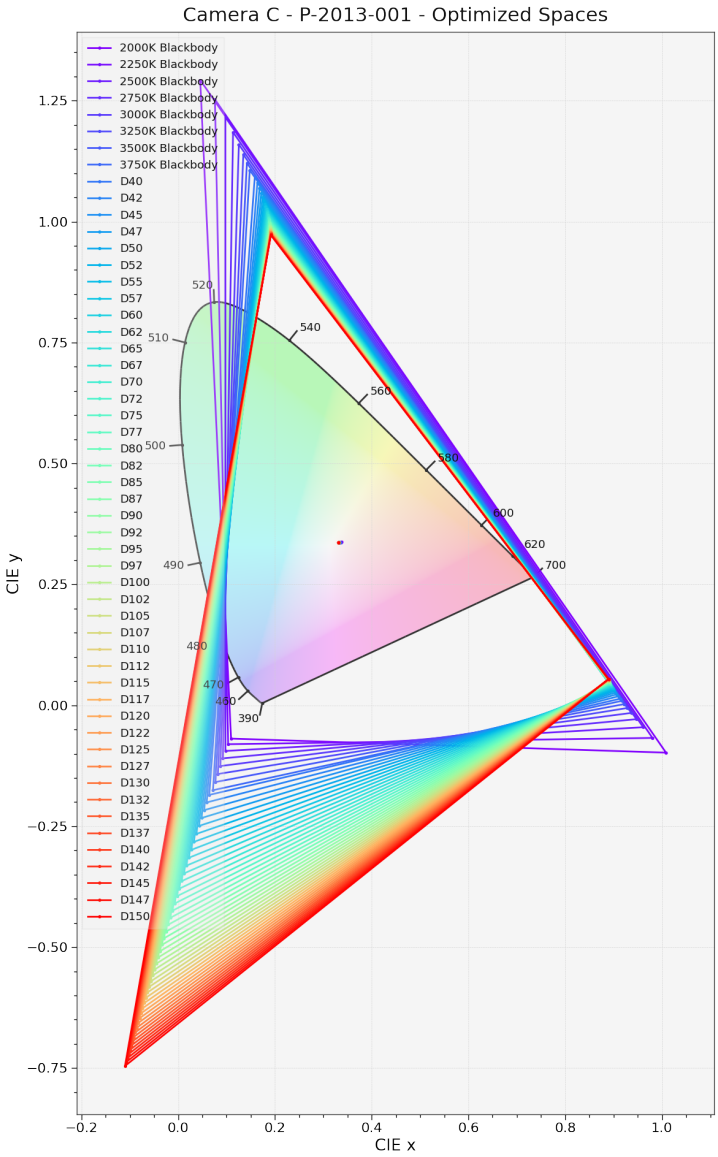


Figure 3.16: Implicit RGB colourspaces for Camera C.

The consequence is that the $F^{-1}_{Camera(LED\ wall)}$ function is white-balance dependent.

3.3.3.2 Insight Through Spectral Simulation

We can get some visual insight into this behaviour through spectral simulation. With knowledge of the LED wall spectral power distribution, camera sensitivities and lens transmission, it is possible to simulate the response of the camera to incident light of the LED wall.

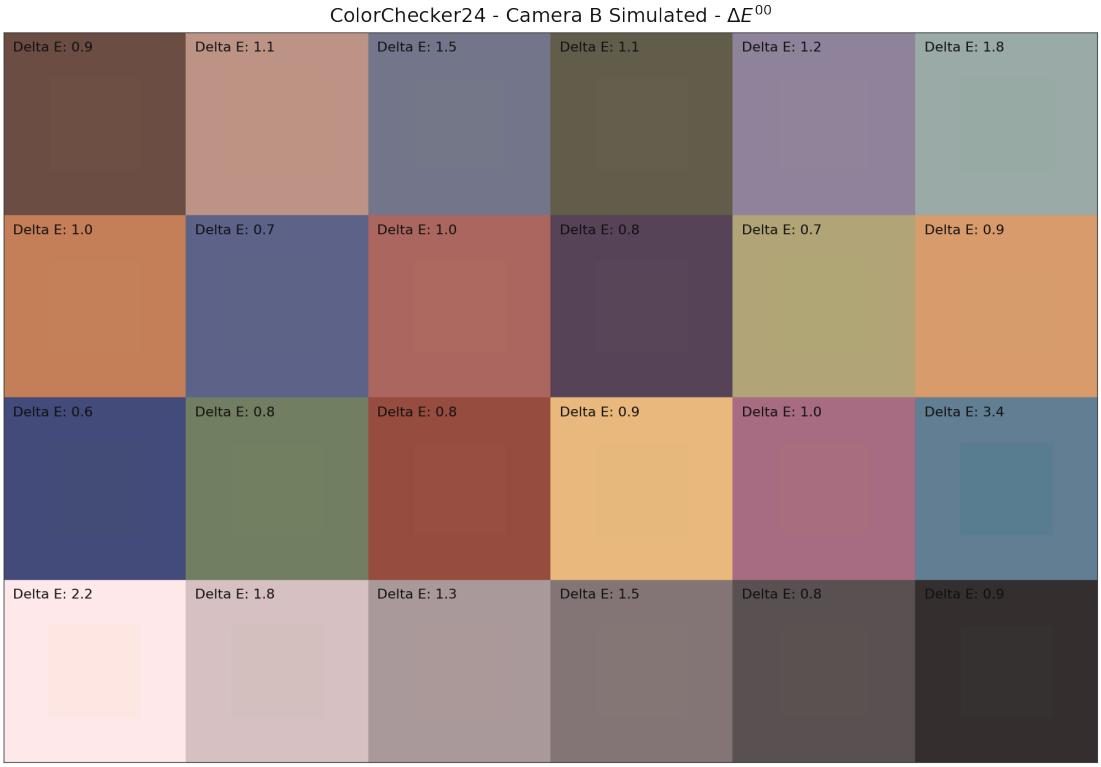


Figure 3.17: After sampling the LED wall, processed the imagery and corrected the optical lens defects, the reference ColorChecker24 is compared to the simulated ColorChecker24 to validate the spectral simulation. The slight differences are not explained, e.g. they are not induced by flare, but the LED wall and camera spectral sensitivities were measured at a different time that the reference imagery was captured.

We will use two simulated ColorChecker24 under daylight and blackbody illumination (6500K and 3000K, respectively), as seen by the *Standard Human Observer* chromatically adapted accordingly, and exhibited on the *Standard Human Observer* calibrated LED wall. The two simulated ColorChecker24 are then captured by the virtual *Camera B* white-balanced at 6500K and 3000K, respectively.

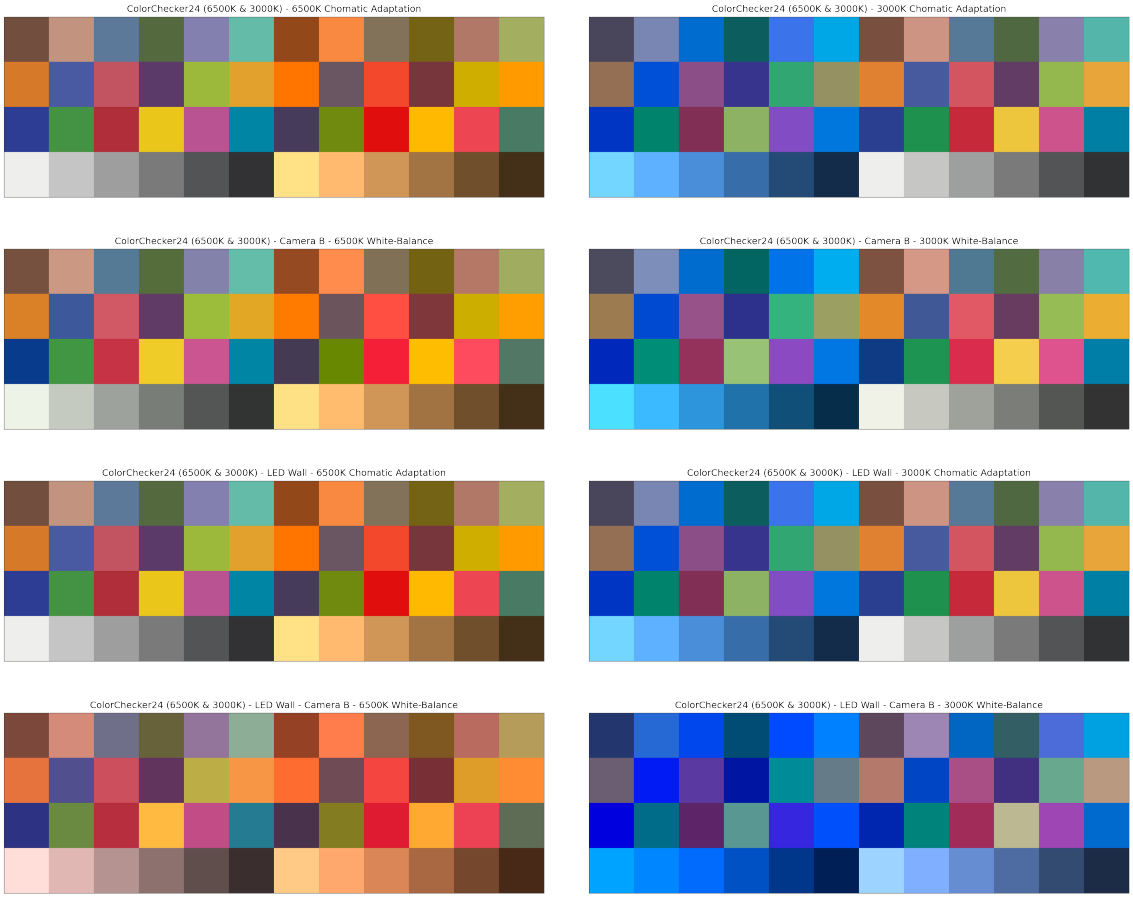


Figure 3.18: A series of simulated ColorChecker24: The first row represents the ColorChecker24 as seen by the Standard Human Observer and chromatically adapted to 6500K, i.e. daylight, and 3000K, i.e. blackbody, respectively. The second row represents the ColorChecker24 captured by Camera B using the 6500K and 3000K IDT, respectively. Note how the second row matches reasonably well with the first row. The third row represents the ColorChecker24 exhibited on the LED wall, and chromatically adapted to 6500K and 3000K, respectively. Note how the third row matches exactly with the first row, i.e. metamerism, but has the spectral distributions of the LED wall. The last row represents the reproduction of the third row by Camera B, white-balanced at 6500K and 3000K, respectively

If we focus our attention on the outside columns of the last row in Figure 3.18, it is unquestionable that the $F^{-1}_{Camera(LED\ wall)}$ function for 6500K cannot be the same as for 3000K. While in this case, the $F^{-1}_{Camera[6500K](LED\ wall)}$ function divides mean CIE CIE ΔE^{00} by 6.8, it multiplies it by 1.13 once used with the *Camera B* white-balanced at 3000K.

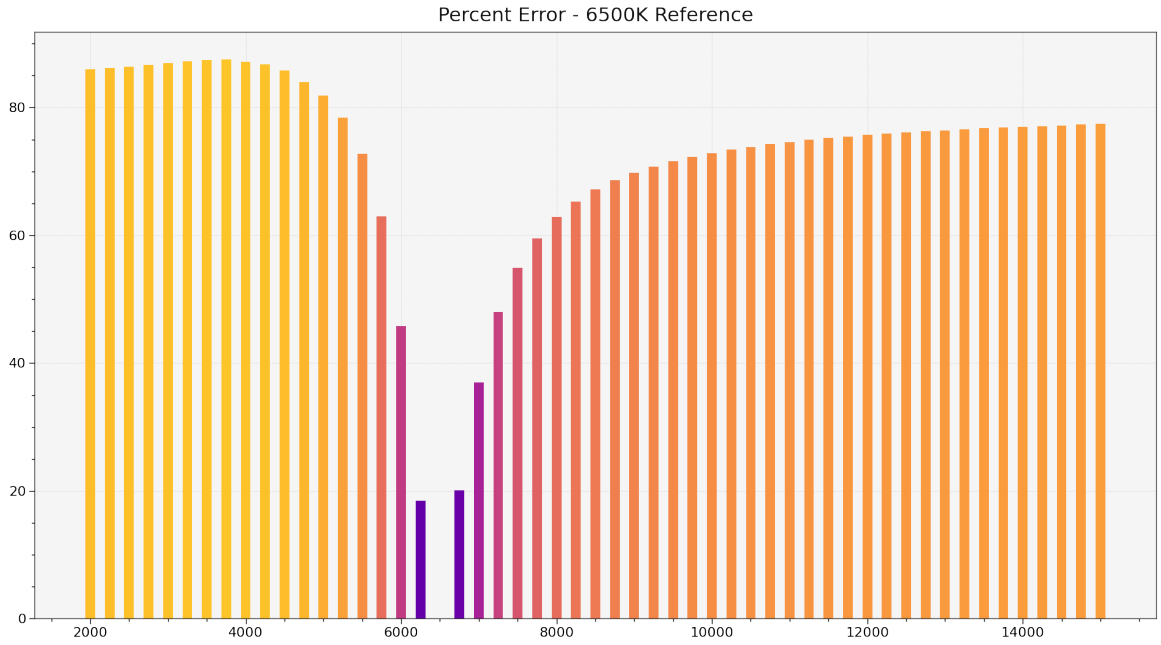


Figure 3.19: Percent errors when $F^{-1}_{Camera[6500K](LED\ wall)}$ is used with other colour temperatures.

As shown in Figure 3.19 $F^{-1}_{Camera[6500K](LED\ wall)}$ error increases dramatically as the white-balance colour temperature diverges from the white-balance value used for the LED wall sampling.

3.3.3.3 White-Balance Dependence Consequences

What are the consequences of the $F^{-1}_{Camera(LED\ wall)}$ function dependence to white-balance? We certainly cannot require a director to use a fixed white-balance, and we cannot really adjust the rendering engine white-balance to compensate:

- It defeats the purpose of the $F^{-1}_{Camera(LED\ wall)}$ function and the effort to create it.
- It is not reciprocal to in-camera white-balance as the working RGB colourspace and the *Camera RGB* space basis are different.
- We are reaching a point where one might as well grade until the output is deemed acceptable.

Ideally, the LED wall sampling process should be done for every colour temperature that the camera might be white balanced at, and the proper $F^{-1}_{Camera(LED\ wall)}$ must be used accordingly. While the LED wall sampling process complexity is significantly increased, access to spectral and *Camera RGB* data shortcuts the need to sample the LED wall. The transformations can be computed directly using simulated data.

References

ANSI and IES Color Committee. 2018. *ANSI/IES TM-30-18 - IES Method for Evaluating Light Source Color Rendition*. ANSI/IES.

CIE. 1987. 17-22-107 colour rendering, <of a light source>. http://cie.co.at/eilvterm/17-22-107

CIE Division 1. 1995. CIE 013.3-1995 Method of Measuring and Specifying Colour Rendering of Light Sources.

CIE TC 1-90. 2017. *CIE 2017 colour fidelity index for accurate scientific use*. Number 224 in Technical report / CIE. CIE Central Bureau, Vienna.

Wendy Davis and Yoshiro Ohno. 2010. Color quality scale. *Optical Engineering* 49, 3 (March 2010), 033602. https://doi.org/10.1117/1.3360335

Kevin A. G. Smet, Aurelien David, and Lorne Whitehead. 2016. Why Color Space Uniformity and Sample Set Spectral Uniformity Are Essential for Color Rendering Measures. *LEUKOS* 12, 1-2 (April 2016), 39–50. https://doi.org/10.1080/15502724.2015.1091356

The Academy of Motion Picture Arts and Sciences. 2019. Academy Spectral Similarity Index (SSI): Overview. , 7 pages.

The Academy of Motion Picture Arts and Sciences, Science and Technology Council, and Academy Color Encoding System (ACES) Project Subcommittee. 2015. Procedure P-2013-001 - Recommended Procedures for the Creation and Use of Digital Camera System Input Device Transforms (IDTs). , 29 pages. http://j.mp/P-2013-001

4

Hallucinating Colour

JOHANNES HANIKA, *Karlsruhe Institute of Technology*

Processing colour can be challenging. In computer graphics, there is a long established tradition of computing directly on the tristimulus values. This is motivated by the point of view that the RGB colour primaries could be monochromatic (as they are in the CIE RGB colour space). This means we can process the (r, g, b) coefficients individually as spectral power for their respective wavelengths.

In practice this is not how rendering systems work, they often have very different RGB working spaces. Unfortunately using anything but monochromatic primaries will result in wrong indirect lighting, as the multiplication of illumination and reflectance spectra is not identical to multiplying RGB coefficients.

When post-processing images, this problem is somewhat aggravated further. For instance simple white balancing is usually done by multiplying three white balance coefficients in some 3D colour space. Doing so for extreme input values (say for instance very blue LED lighting) can easily result in output values that are outside the spectral locus and would thus require negative radiance to be reached [Burns 2019].

These issues can be mitigated by performing compute in the spectral domain, sticking to the physical definition of light. Attempting this gives rise to the need for tristimulus to spectrum upsampling techniques. In particular in graphics, we are concerned with

- storage size: ideally this does not exceed the size of an RGB texture,
- evaluation speed: this should not be much more than a filtered texture lookup,
- generality: we need to work on the full gamut of all colours,
- plausibility: the spectral shape should have realistic features (such as for instance $\in [0, 1]$ or smoothness).

In the following we want to introduce the problem space and classify a few existing approaches.

4.1 Properties of Upsampling Techniques and their Spectra

Common to all methods summarised in here is the round-trip constraint: They are designed to guarantee exact conversion from (r, g, b) coordinates to a spectrum $f(\lambda)$ defined over the

wavelength $\lambda \in \Lambda = [360, 830]$nm and back via integration with the 1931 colour matching functions, $\bar{x}(\lambda), \bar{y}(\lambda), \bar{z}(\lambda)$. More generally speaking, if we convert (r, g, b) from an arbitrary RGB colour space to the standardised CIE XYZ colour space, generate a spectrum $f(\lambda)$ for these coordinates, we can exactly compute XYZ again by

$$(X, Y, Z) = \int_\Lambda (\bar{x}, \bar{y}, \bar{z})(\lambda) \cdot f(\lambda) \, \mathrm{d}\lambda. \tag{4.1}$$

Note that this assumes we are directly looking at an object emitting spectral radiance proportional to $f(\lambda)$. To describe the reflectance of an object, $f(\lambda)$ in the above equation is replaced by a reflectance spectrum (multiplied by the equal energy white point illuminant E, which does not change the equation). This is done because the definition of XYZ uses illuminant E as white point. When going directly to another RGB colour space, for instance a D65 white point can be multiplied inside the integration instead.

One crucial difference between reflectances and emission spectra is the bound. Emission spectra are positive, but unbounded. Reflectance spectra are bounded by $[0, 1]$ because of energy conservation (you can't reflect more than what came in). We will see that these bounds pose problems to some of the available techniques for upsampling.

Another observation is that, again due to energy conservation laws, the shape of reflectances varies greatly with brightness of the surfaces. The brighter a saturated colour, the more boxy is the shape of the spectrum [MacAdam 1935]. On the other hand, illumination spectra don't have to obey this bound (light sources *are* producing energy!). This additional degree of freedom, and the fact that arbitrarily scaling a spectrum by $c > 0$ results in

$$\int_\Lambda (\bar{x}, \bar{y}, \bar{z})(\lambda) \cdot c \cdot f(\lambda) \, \mathrm{d}\lambda = c \cdot (X, Y, Z), \tag{4.2}$$

$$\frac{(c \cdot X, c \cdot Y)}{c \cdot (X + Y + Z)} = \frac{(X, Y)}{X + Y + Z} \tag{4.3}$$

$$= (x, y), \tag{4.4}$$

i.e. unchanged chromaticity coordinate (x, y), means that emission spectra can be derived from a 2D lookup table over (x, y) and scaled up to the appropriate brightness. This is exploited by Meng et al. [2015] who precompute smooth emission spectra on a 2D grid, see Fig. 4.1. In general, this is not the case for reflectances. The boxiness of the spectra will have to change with brightness to yield optimal results, i.e. the lookup tables will have to be 3D. This can be seen in Fig. 4.2, where the same colour is depicted, once using a bright, precomputed 2D lookup which is scaled to fit the brightness, and once using a 3D table which can generate much smoother spectra for dim brightness.

It is worth pointing out that often times a reconstructed spectral distribution is neither an emission nor a reflectance spectrum. Inside a rendering system, often one needs to define spectrally varying values for microfacet roughness, mean cosines for phase functions, indices of refraction, or collision coefficients for participating media. Such parameters are inputs to a non-linear light transport model and are thus not subject to the round trip expressed in eq. 4.1. Since in these cases it is also borderline meaningless to define the values in RGB colour space, it is best to do look development with a certain upsampling method and stick to it. Changing the upsampling technique here will result in an equally meaningless but different spectrum and require artists to re-tweak the input RGB textures. This illustrates, however, the need for upsampling techniques that operate outside the bounded domains. For instance mean cosines can be $[-1, 1]$, i.e. even negative.

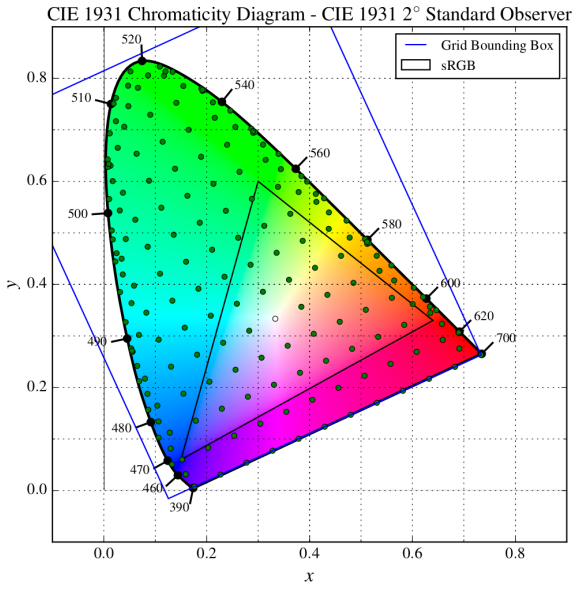


Figure 4.1: Emission spectra can be precomputed on a 2D grid, as illustrated in this figure by Meng et al. [2015].

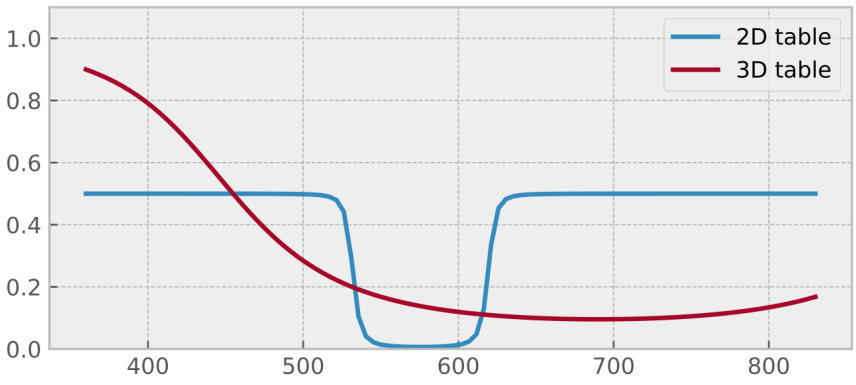


Figure 4.2: Reflectance spectra profit from a 3D LUT: dim colours can be assigned much smoother spectra. Figure reproduced from Jakob and Hanika [2019].

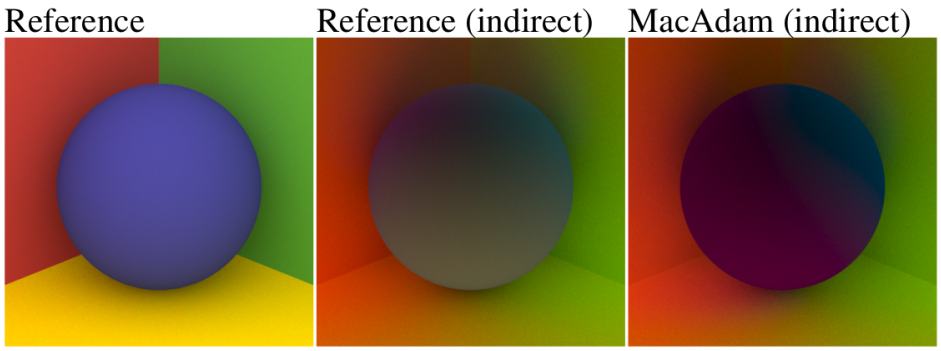


Figure 4.3: Using boxy MacAdam spectra for rendering results in clearly visible changes to indirect illumination as compared to using the ground truth spectra from a colour chart. Figure by Meng et al. [2015].

One more difference between types of spectra is the smoothness. Reflectance spectra are generally smooth (until they become boxy). Some emission spectra exhibit very spiky shape, for instance fluorescent light bulbs. To generate spectra closer in shape to the intended materials, it is possible to build priors from datasets into the upsampling methods, by using clustered principal component analysis [Otsu et al. 2018].

4.2 The Methods

4.2.1 The theoretical limit: MacAdam 1935

These spectra are interesting for their theoretical value [MacAdam 1935]. They represent reflectances at maximum possible brightness. Their shape is boxy: there is exactly one rising and one falling edge in the spectral shape (though not necessarily in that order), forming a peak or a dip. Storage is thus only a set of two wavelengths, their order can indicate the peak or dip shape. Note that this does not include a brightness value, the resulting colour will always have the maximum possible brightness. This can be evaluated extremely quickly, by just comparing whether λ is between the rising and the falling edge.

These spectra are unusual for natural materials and thus result in implausible indirect illumination, see Fig. 4.3.

4.2.2 Interpolating Precomputed Spectra

Glassner 1989 This method [Glassner 1989] starts out with a spectrum that is zero everywhere but at three fixed wavelengths λ_u, λ_v, and λ_w. Such spectra matching a desired (r, g, b) coordinate can be obtained by simple matrix math. Since convex combinations of any such spectra for any three such wavelengths are again expressing the same (r, g, b) coordinate, the method can be iterated to obtain dense spectra. Going the iterative route is expensive, but until then the spectral shape remains sub-optimal.

A slight twist to this approach is using the monitor analog: Expressing spectra as the linear combination of three primary response spectra exactly as a computer monitor does it. The three emission profiles are mixed by (r, g, b) coefficients. This is described by Mallett and Yuksel [2019] and Burns [2020a]. The latter also gives some in-depth evaluation

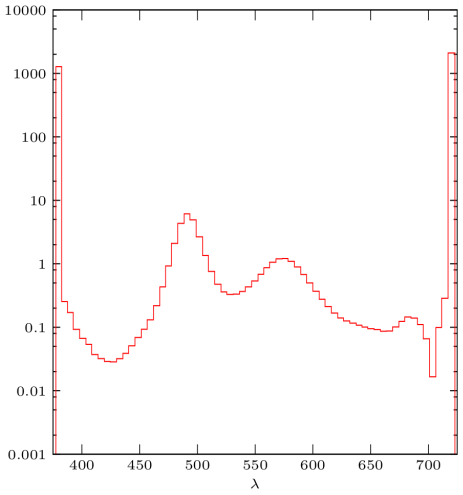


Figure 4.4: Reproduced from Fig. 4.2 by Borné [2014]. Naive mean value weighting of monochromatic delta peaks yields strongly peaked features at the minimum and maximum wavelength, even for D65 white as shown here. This problem can be alleviated by different weighting schemes but remains present on the purple line.

of spectral shapes resulting from this method. These are smooth, but since the three basis spectra assigned to (r, g, b) resemble a smoothed disjoint box basis, the resulting spectra are also prone to (smoothed) staircasing. Another fundamental limitation of the three-primary approach is that outside the gamut formed by the primaries, it requires negative coefficients and will thus produce unbounded spectra with negative radiance.

Borné 2014 To break free from these gamut restrictions, Borné [2014] extended the barycentric interpolation across a triangle in xy chromaticity coordinates to mean value coordinates on polygons with many more vertices: every monochromatic spectrum for each wavelength λ would be represented as such a vertex on the spectral locus, and every spectrum would be represented as a convex combination of these. Unfortunately such an approach yields very spiky and awkward spectral shapes, especially on the purple line: essentially just interpolating between a peak at the minimum and one at the maximum wavelength.

Smits 1999 This work is *the* classic way of upsampling (r, g, b) values to a full spectrum Smits [1999]. The original publication defined a set of seven spectra, sampled with 10 spectral samples, one for white, yellow, magenta, cyan, red, green, and blue.

This provides a clever way of working around the limitation mentioned in the previous sections: The spectrum is constructed as close as possible to white, by iteratively subtracting portions of the (r, g, b) coordinate that can only be expressed by the more peaky spectra (red, green, blue), and then subtracting what can be expressed by the wider ones (equal contribution in red and blue can be represented by a magenta spectrum). This keeps some of the simplicity of the "3-primary spectra only approach" as pioneered by Glassner [1989], but results in smoother spectra. It is, however, unclear how to extend this to include more saturated spectra such as the delta peaks used by Borné [2014].

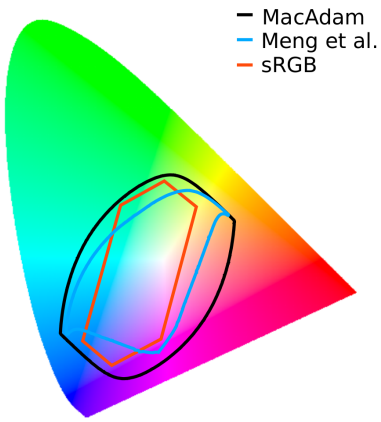


Figure 4.5: Figure reproduced after Jung et al. [2019]: the gamut of smooth spectra by Meng et al. [2015] cannot represent all valid reflectances (shown in black), even while some of them are still inside sRGB (shown in red). The plotted gamuts show a slice through a certain brightness, hence the hexagonal shape of sRGB (it is a slice through a cube).

A particular Gem is Hachisuka's fast GPU implementation using Gaussian fits for the spectra, as can be found in the supplemental code to the SPPM publication [Hachisuka and Jensen 2009].

The main limitation of this method is that it is conceptually constrained to sRGB. Larger colour spaces make the optimisation step generating the LUT spectra unstable, since then not all values inside $[0,1]^3$ represent valid reflectances any more.

Meng et al. 2015 Another method based on precomputed spectra from an optimisation is the one by Meng et al. [2015]. They precompute smooth emission spectra and place them in a 2D LUT over the xy chromaticity chart. Intermediate colours can be represented precisely by linearly interpolating the neighbouring spectra. This LUT is relatively compact since it is 2D, but also requires clamping or gamut mapping to create valid reflectances for extremely saturated colours. Since they use a smoothness constraint, their spectra cannot represent all reflectances inside the sRGB gamut, see Fig. 4.5.

Otsu et al. 2018 Otsu et al. [2018] propose to cluster the chromaticity plane (x, y) and for every 2D voxel in a kd-tree in this domain use principal component analysis (PCA) to recover spectra that are similar in shape to an input training set.

This approach is very suitable to faithfully reproduce indirect lighting, as can be seen in the green patch in [Jakob and Hanika 2019, Fig. 9], reproduced here in Fig. 4.6.

Unfortunately a simple linear combination of precomputed spectra inherits the same problems as Glassner [1989] had: the gamut is limited to the convex combination of the exemplars and in general this can yield negative and values greater than one. Another issue is that the kd-tree voxel boundaries cause discontinuous changes for colour gradients.

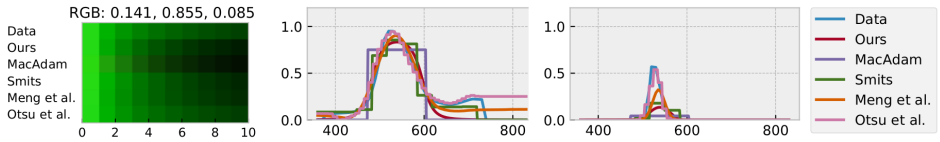


Figure 4.6: Clustered PCA can very faithfully represent the shapes of spectra, leading to better match for indirect lighting. Figure from Jakob and Hanika [2019]. Left: an RGB colour chart with simulated indirect bounces from 0 to 10, using the indicated methods for spectral upsampling. Note how Otsu et al. [2018] matches the ground truth much better in the right plot (the last patch of the indirect bounces chart).

4.2.3 Constrained Optimisation

There is a class of algorithms which runs their solvers directly on the (r, g, b) input to generate a high quality spectrum, regardless of computation speed. Notable representatives are introduced by Heikkinen et al. [2008] and Burns [2020b]. Both also present a variant that uses a hyperbolic tangent transform to explicitly limit the shapes of their resulting spectra to within $[0, 1]$. This is a trick that is also used in the remaining two methods described in the next section. One key difference, however, is the inline optimisation without any precomputation. This approach is very slow and not suitable for inline evaluation inside a rendering or compositing system. Another is that the function space inside the hyperbolic tangent is the plain discretised wavelength domain, for instance 39 spectral bins.

4.2.4 Function Spaces

All previously mentioned methods either use a set of precomputed curves or directly operate on discretised wavelengths to produce their output. Another class of approaches uses parameterised function spaces. There have been two different proposals in computer graphics recently.

Sigmoid Polynomials This approach by Jakob and Hanika [2019] is based on the composition of a polynomial and a sigmoid function: The spectral power distribution $f(\lambda) = s(p(\lambda))$ is composed of a polynomial part $p(\lambda)$ and a sigmoid function $s(x)$, with

$$p(\lambda) = \sum_{i=0}^{d-1} c_i \cdot \lambda^{d-i-1} \tag{4.5}$$

$$s(x) = \frac{1}{2} + \frac{x}{2\sqrt{1+x^2}}. \tag{4.6}$$

This model is very fast to evaluate, and for $d = 3$ can be easily parameterised with intuitive parameters such as the dominant wavelength for hue [König et al. 2020]. It has also been extended to include fluorescent contributions to extend the gamut of valid reflectances [Jung et al. 2019].

The biggest drawback of this model is that the translation from (r, g, b) coordinates to c_i coefficients requires a Gauss/Newton optimisation step or a precomputed LUT. The resulting coefficients can be stored as a texture ($d = 3$ is enough to represent all colours). However, the range and precision of the stored values have a big impact on the accuracy of the result, so using at least 32-bit floating point numbers for c_i in a 3D LUT is mandatory. This is true to

a lesser extent to the reparameterisation of König et al. [2020]. However this representation is even less suited for direct linear interpolation of the coefficients than using the plain c_i, as is discussed in König et al. [2020] and Jung et al. [2019].

Fourier Moments Another function space has been proposed by Peters et al. [2019b]. We repeat their eq. (11):

$$f(\lambda) = \frac{1}{\pi} \arctan\left(\Re\lambda_0 + 2\Re\sum_{l=1}^{m} \lambda_l \exp(-il\varphi)\right) + \frac{1}{2} \tag{4.7}$$

Here, φ is the phase that is a warped version of the wavelength λ, and the λ_l in this equation are Lagrange multipliers as constructed by their eq. (10). Note that these λ_l have the function of Fourier coefficients in the inner expression, but are constructed to match the Fourier moments of the complete $f(\lambda)$ including the arctan transform.

We don't attempt a fully self-contained description here and instead refer to their supplemental code. We want to point out here, however, how similar the structure is to the sigmoid. It is essentially a truncated Fourier series bounded by the application of arctan (as opposed to a truncated polynomial bounded by the application of a sigmoidal function based on a square root). The λ_l could be stored directly instead of the Fourier moments, making the formulations resemble each other even more closely. This would, however, inherit all the drawbacks of non-linear interpolation and non-uniform quantisation.

This formulation reconstructs functions that are strictly within $[0, 1]$ from Fourier moments. Such moments could have been used directly in a truncated Fourier series, but such a simple expansion would not enforce bounds and result in the usual ringing also below zero and above one.

While the reconstruction (using the *bounded Maximum Entropy Spectral Estimate, MESE*) itself is non-linear, a linear interpolation of the input Fourier moments will result in a very well behaved and meaningful outcome. This is good news in this context, because we can thus store the Fourier moments in textures and use filtered lookups as usual.

Furthermore, it is possible to store these moments quantised at low bit rates, since their bounds are known Peters et al. [2019a]. Down to 10 bits this is no problem at all, if going to 8 bits special care has to be taken to bring the moment space closer to RGB space. Dithering can help hide banding artifacts, as is often the case when storing colours.

Another remarkable property is that conceptually we do not have to search for the Fourier moments corresponding to a colour. They are an intrinsic property of the spectrum $f(\lambda)$ and can be computed by integration. However, if the spectrum is unknown and we want to build a table mapping from XYZ to Fourier moments, we can perform the same Gauss/Newton search as has been done for sigmoids Jakob and Hanika [2019]. Peters et al. [2019b] perform an exhaustive inverse lookup and construct forward XYZ values for all 3D Fourier moments quantised to 10 bits instead.

The Fourier moments can also be used with the unbounded version of the MESE, to reconstruct spiky emission spectra. For complicated shapes, this requires more moments, see Fig. 4.7.

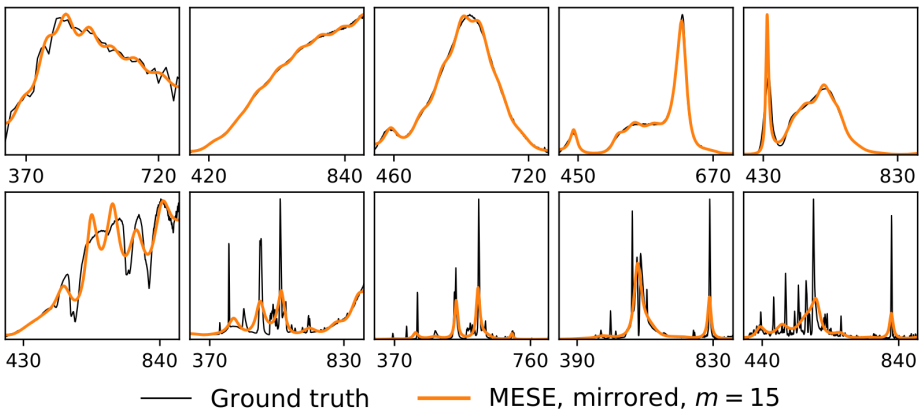


Figure 4.7: The unbounded MESE can be used to reconstruct emission spectra $\in [0, \infty)$ from a set of Fourier moments. Figure reproduced from Peters et al. [2019b].

4.2.5 Performance

Peters et al. [2019b] give explicit performance numbers for many methods mentioned in here. Unsurprisingly the AVX variant of Jakob and Hanika [2019]'s method is a clear winner, but most methods are well within practical run times. For the others the main difference comes from the size of the employed lookup tables. This is probably the major concern when implementing such techniques for certain hardware architectures. For instance a GPU implementation might depend even more on memory accesses.

4.3 A Classification Attempt

To guide practitioners, we classify the above mentioned upsampling techniques with respect to a few key properties:

- compute: This evaluates how much computation is required run-time to evaluate $f(\lambda)$ from (r, g, b) or any intermediate representation that may be stored instead.
- mem: This evaluates the required memory for a potentially employed lookup table, such as tables of reference spectra or colour cubes with coefficients.
- texture: This is the type of quantisation that can be employed to a texture storing the intermediate coefficients of the model, or (r, g, b) if not applicable.
- filter: This indicates whether the texture representation can be linearly filtered.
- gamut: This indicates whether the method is applicable on the full spectral locus as input gamut or whether it has certain limitations.
- R/E: This property indicates whether the method is suitable for reflectance or emission spectra.
- shape: This soft skill describes the features of the shape of the spectral curves that come out of a particular method.

The results can be seen in Table 4.1.

Table 4.1: Comparison of some key properties of various upsampling approaches. Please see the main text for an explanation. Note for the methods of Jakob and Hanika [2019] and Peters et al. [2019b] either the LUT or the texture is required run-time.

	compute	mem	texture	filter	gamut	R/E	shape
MacAdam 1935	—	++	2x ui8	no	full	R	box
Glassner 1989	–	–	3x ui8	yes	RGB	E	:(
Smits 1999	-	-	3x ui8	yes	sRGB	R	about smooth
Borné 2014	+	none	3x ui8	yes	full	E	spiky purple
Meng et al. 2015	+	++	3x ui8	yes	almost full	E	smooth
Otsu et al. 2018	-	++	3x ui8	yes	input	(R)	input
Jakob 2019	-	+++	3x f32	no	full	R	smooth+box
Peters et al. 2019	+	+++	3x f10	yes	full	R(+E)	smooth+box
Burns 2020	+++	none	3x ui8	yes	full	R	smooth+box

4.4 Conclusion

In this chapter, we gave a short overview over the problem space and explained the approach of a few related techniques on a high level. We proposed a classification scheme based on key properties of the upsampling method as required by the inner loops of rendering and compositing systems. There are very many different upsampling techniques to choose from, and this survey is by no means complete. In the future hopefully there will be one unified method to cater for all use cases. In particular an unsolved problem seems to be a fast method that achieves the high quality of spectral similarity as clustered PCA does, but strictly enforces the bounds and smoothness properties of the function space-based methods.

References

Joscha Borné. 2014. Konvertierung von RGB nach Spektrum mittels Mean-Value-Koordinaten. Master's thesis, Karlsruhe Institute of Technology.

Scott Burns. 2019. Chromatic Adaptation Transform by Spectral Reconstruction. *Color Research & Applications* 44, 5 (2019), 682–693.

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

Scott Burns. 2020b. Numerical methods for smoothest reflectance reconstruction. *Color Research & Applications* 45, 1 (2020). https://doi.org/10.1002/col.22437

Andrew S. Glassner. 1989. How to derive a spectrum from an RGB triplet. *IEEE Computer Graphics and Applications* 9, 4 (1989), 95–99. https://doi.org/10.1109/38.31468

Toshiya Hachisuka and Henrik Wann Jensen. 2009. Stochastic Progressive Photon Mapping. In *ACM SIGGRAPH Asia 2009 Papers* (Yokohama, Japan). Article 141, 8 pages. https://doi.org/10.1145/1661412.1618487

Ville Heikkinen, Reiner Lenz, Tuija Jetsu, Jussi Parkkinen, Markku Hauta-Kasari, and Timo Jääskeläinen. 2008. Evaluation and unification of some methods for estimating reflectance spectra from RGB images. *J. Opt. Soc. Am. A* 25, 10 (2008), 2444–2458. https://doi.org/10.1364/JOSAA.25.002444

Wenzel Jakob and Johannes Hanika. 2019. A Low-Dimensional Function Space for Efficient Spectral Upsampling. *Computer Graphics Forum* 38, 2 (2019). https://doi.org/10.1111/cgf.13626

Alisa Jung, Alexander Wilkie, Johannes Hanika, Wenzel Jakob, and Carsten Dachsbacher. 2019. Wide Gamut Spectral Upsampling with Fluorescence. *Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering)* 38, 4 (2019). https://doi.org/10.1111/cgf.13773

Lars König, Alisa Jung, and Carsten Dachsbacher. 2020. Improving Spectral Upsampling with Fluorescence. In *Workshop on Material Appearance Modeling*.

David L. MacAdam. 1935. Maximum Visual Efficiency of Colored Materials. *J. Opt. Soc. Am.* 25, 11 (1935). https://doi.org/10.1364/JOSA.25.000361

Ian Mallett and Cem Yuksel. 2019. Spectral Primary Decomposition for Rendering with sRGB Reflectance. In *EGSR '19 Proceedings of the 30th Eurographics Symposium on Rendering*.

Johannes Meng, Florian Simon, Johannes Hanika, and Carsten Dachsbacher. 2015. Physically Meaningful Rendering using Tristimulus Colours. *Computer Graphics Forum* 34, 4 (2015). https://doi.org/10.1111/cgf.12676

Hisanari Otsu, Masafumi Yamamoto, and Toshiya Hachisuka. 2018. Reproducing Spectral Reflectances From Tristimulus Colours. *Computer Graphics Forum* 37, 6 (2018), 370–381. https://doi.org/10.1111/cgf.13332

Christoph Peters, Sebastian Merzbach, Johannes Hanika, and Carsten Dachsbacher. 2019a. Spectral Rendering with the Bounded MESE and sRGB Data. In *MAM2019: Eurographics Workshop on Material Appearance Modeling*.

Christoph Peters, Sebastian Merzbach, Johannes Hanika, and Carsten Dachsbacher. 2019b. Using Moments to Represent Bounded Signals for Spectral Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)* 38, 4 (2019), 136:1–136:14. https://doi.org/10.1145/3306346.3322964

Brian Smits. 1999. An RGB-to-spectrum Conversion for Reflectances. *Journal of Graphics Tools* 4, 4 (1999), 11–22. https://doi.org/10.1080/10867651.1999.10487511

5

Fluorescence, Spectral Workflows in ART and Spectral OpenEXR

ALEXANDER WILKIE, *Charles University*

5.1 Fluorescence

Fluorescence is the effect where a molecule re-emits a previously absorbed photon in a lower energy state than the one it was absorbed in: this means the re-emitted light from a collision event with such a molecule has a longer wavelength than the original incoming light. This stands in stark contrast to the behaviour of "normal" object reflectance, where no such shifting activity takes place. The shift from the absorbed frequency to the re-emitted one is spread around a main frequency as shown in Figure 5.1: the phenomenon is referred to as *Stokes shift*.

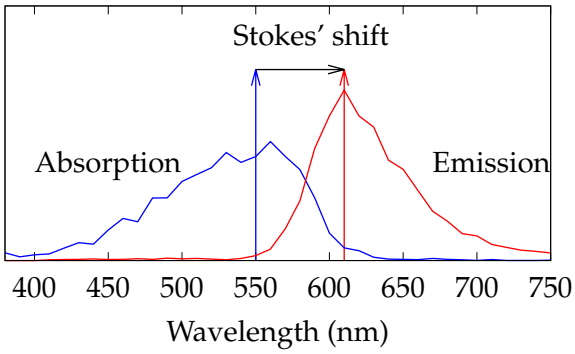


Figure 5.1: An example of the so-called *Stokes shift*, here shown for the material found in pink fluorescent 3M pink Post-it notes. Practically all fluorophores exhibit similar characteristics: namely, that there is one main band where energy is absorbed, and one where it is re-emitted. Note that one still needs the full re-radiation matrix of the fluorophore in question to generate such a plot, which only shows the cumulative shifting effect of the material. Figure 5.2 shows the entire re-radiation matrix of this material.

Such phenomena can be described by a fluorescence response function $\Phi(\lambda_i, \lambda_o)$, which for an excitation wavelength λ_i returns the amount of energy relaxed to another re-emission wavelength λ_o. For $\lambda_i = \lambda_o$, this corresponds to the non-fluorescent reflectance.

The fluorescence response $\Phi(\lambda_i, \lambda_o)$ can be represented in a discretised form as a so-called re-radiation matrix. An example dataset is shown in Figure 5.2. The incoming wavelength λ_i lies on its vertical axis, and the outgoing wavelength λ_o on its horizontal. The diagonal of such a re-radiation matrix describes the amount of light that does not undergo any wavelength shift.

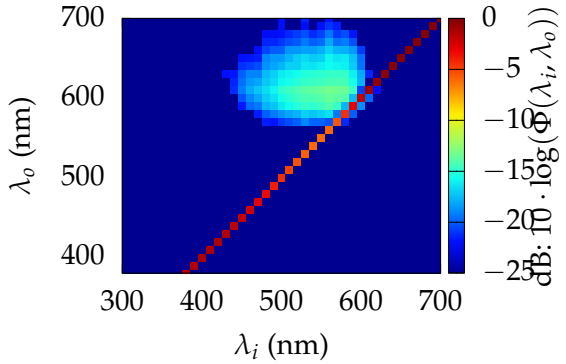


Figure 5.2: The full re-radiation matrix $\Phi(\lambda_i, \lambda_o)$ for the material shown in figure 5.1. Values lower than 0.006 are ignored in this plot, as they are considered measurement noise.

From a rendering technology viewpoint, it is important to note that such wavelength-shifting materials raise issues in all stages of the rendering pipeline, namely

1. how one can describe the phenomenon, purely from a technical viewpoint,
2. how artists can work with it,
3. how one can integrate it into a spectral renderer, in particular if Hero Wavelength Sampling is being used,
4. how it affects volume rendering, and
5. the effect it has on the gamut of output images.

We will briefly discuss all of these in turn, most of them with a link to further literature.

5.2 Relevance of Fluorescence for Production Rendering

Before one embarks on a technical discussion of a phenomenon which is not yet to be found in current production systems, it is worthwhile to discuss why it is actually potentially important. Typically, spectral rendering is being used in production if high levels of realism are desired in the resulting output images: which is why any effect that has a perceptible impact on light-matter interactions has to be taken into account. And as fluorescence has a marked appearance impact for some asset classes, it definitely needs some attention.

Fluorescence is commonly only associated with very bright warning colours, such as day-glo orange and yellow. Objects with these colours are not extremely rare, but not very common, either: due to their garish appearance, such colours are mostly restricted to specialist usage cases such as the warning paint on emergency service vehicles and such. There

are existing techniques to jerry-rig such appearance in existing workflows (in the rare cases when they are needed), so one could assume that fluorescence is actually a corner case that only needs to be covered when doing lookdev on such special objects.

But contrary to what one might expect, fluorescence is actually a fairly common feature in a lot of other modern man-made materials, in addition to a sizeable number of natural substances. An example of the latter are all green plants: chlorophyll shifts purple and blue light to red, to the extent that one can judge the health of a plant by the amount of fluorescence one can observe on it. The effect is not extremely strong: but strong enough to throw off any attempt to accurately predict plant appearance under UV-rich outdoor lighting with plain spectral measurements of reflectance and absorption.

The same goes for a sizeable number of modern fabrics, paints and plastics, especially in the white, orange and red colour range. In a lot of cases, such colours contain *optical brighteners* or *chroma boosters*. The former aim at increasing the whiteness of textiles and paper, while the latter increase the colourfulness of the object in question. A good way to detect the presence of such brighteners and chroma boosters is to obtain a purple laser pointer: $406[\text{nm}]$ devices are commonly available as novelty items. Due to the characteristics of the human eye, they are not much use as laser pointers for presentations: we are not very good at actually seeing this wavelength. But what they can be used for is as a "fluorescence detector": if one points such a device at a surface, and the laser dot changes colour, then wavelength shifting of some sort occurs.

Important: *when doing anything of the sort, please always observe basic laser safety rules, and only work with the weakest such lasers you can find! Optimally* $< 5[\text{mW}]$ *power, less is better for this! And never look directly into the laser beam! With a purple laser, this is even more dangerous than with a green or red one, because as humans we do not see this wavelength well anymore (so the reflex to shut one's eye is weaker): but the capability to damage the retina is the same as with the other colours – radiative power is radiative power, even if we cannot see it well.*

If this "laser pointer test" detects any wavelength shifting activity (that is, if the dot changes colour), one has to realise that the material in question will not properly colour match between renderings and plate footage in any "plain" spectral workflow that does not take fluorescence into account. How big the error is varies considerably: but some error there always will be.

TL;DR: the main thing to keep in mind for spectral production work is that if an exact match between plate footage of a real object and renderings of it under measured illumination and with measured spectral reflectance fails, fluorescence might be playing a role. In the medium term future, spectral rendering systems will provide some capabilities to incorporate this feature: but for now, all one can do is to at least be aware that there might be issues in this regard. And possibly avoid such materials in real scene assets for the time being, if an exact match with rendered spectral assets is needed later.

5.2.1 Describing Fluorescence

As one can see from figure 5.2, fluorescence data is inherently two dimensional, so the natural way to store it is in a two dimensional array of floating point numbers. However, unlike plain spectra, where such an approach might just be a bit inefficient (but depending on circumstances, possibly still acceptably so), re-radiation matrices require fairly high resolutions – and due to being two dimensional, square the memory cost of using a regular sampling approach. So while an implementation that uses naive arrays to store this data is

of course useful for an initial integration of such functionality in a rendering system, it is clearly impractical for use in a production system due to memory issues. If multiple fluorescent datasets are present in a scene (and if textures are being used, there potentially are millions of them), any reasonably dense 2D sampling of the re-radiation properties will require unsustainable amounts of storage space. And this is in addition to the problem of efficiently sampling the re-radiation process described by such arrays: for this, one needs to pre-compute probability tables (again 2D matrices) that can then be importance sampled.

Recent work has shown that Gaussian Mixture Models Qingqin et al. [2021] can be used effectively to significantly reduce the memory demands of re-radiation matrices: in addition to providing very high compression rates of the underlying data, they also enable efficient sampling of the fluorescence process. This technique allows integration of re-radiation effects into a rendering workflow with orders of magnitude smaller memory footprint, so that it is now feasible to have fluorescence as a feature in actual production systems (as opposed to research renderers) in the medium term future.

5.2.2 Working with Fluorescence as a 3D Artist

The current state of the art in this area is still highly experimental, but it is reasonable to assume that production workflows will offer two orthogonal options for the inclusion of such effects in the future.

The first will be the direct assignment of particular re-radiation data to assets, so that specific appearance for a given object can be achieved. This will likely be similar to normal, non-wavelength shifting spectral data being specifically assigned to a given asset: this will not become the norm (lookdev work will always mainly be done in colour space), but will definitely start to appear as an option in workflows in the medium term future.

The second option is to automatically include simulated optical brighteners during spectral uplifting, as demonstrated by Jung et al. [2019]. This technique is intended for the spectral uplifting of wide gamut RGB input textures, and deals with the issue that RGB colours beyond sRGB increasingly cannot be created by normal reflectance anyway. The technique proposed in that paper automatically adds suitable but arbitrary optical brighteners to saturated colours that can no longer be represented by plain reflectance: this is a first step, and a basic demonstration of capability. For production work, one would also want an ability to narrow down the types of virtual fluorophores that are being added to the uplifted reflectance spectra, to achieve specific behaviour found in real scene assets that one might be attempting to match.

5.2.3 Integrating Fluorescence into a Path Tracer

The main thing to note here is that since the work of Hullin et al. [2010] and Mojzík et al. [2018], this is mostly a solved problem. With the notable exception of reducing the storage requirements of re-radiation matrices Qingqin et al. [2021], all the components for integration of fluorescence effects in a modern path tracer that uses Hero Wavelength sampling have been derived and tested in practice. Such an integration comes at a performance cost, both in terms of memory and execution time (and of course also code complexity). But there are no issues left that would require actual *research* on the part of those who wish to add this to a renderer: all that is needed is a careful engineering assessment regarding the best way to adapt an existing system to the inclusion of re-radiation data.

The ART Open Source rendering research toolkit Wilkie [2018] contains a reference implementation of the technique proposed by Mojzík et al. [2018], so that there is also a freely available example of how this sort of thing can be integrated into a path tracer in practice.

5.2.4 Gamut Issues of the End Result

Most VFX workflows use large target RGB spaces anyway, so the following brief caveat does not apply to them. But if sRGB is the intended target, one should be aware that the sometimes very saturated and/or bright colours one can obtain via the addition of wavelength shifters to asset appearance will with a fairly high probability push the end result out of the sRGB gamut. This should be taken into consideration before incurring the still considerable effort to add such effects to a rendering pipeline: even if a case for their addition can be made otherwise, a small target RGB gamut might make the whole effort pointless.

5.3 The Spectral Rendering Workflow in ART

Something that requires careful consideration in any spectral renderer is how the spectral data one works with is managed on its path from input scene description to output image. As spectral rendering systems are not very common yet, descriptions of the entire workflow are also still rare. To give an example of what has to be done, we briefly discuss the spectral pipeline of ART Wilkie [2018], which is an Open Source rendering research toolkit. While this system is not a production renderer, the issues that one faces are the same as in such systems: all that is different are the engineering choices that were made.

As can be seen in figure 5.3, the two key issues are what to do with the variety of input data that one has to work with (a marked difference from an RGB system, where there is essentially only a single data type coming into the system), and what the output format looks like.

To raise rendering speed, ART uses an approach where the Hero Wavelength Sampling process only has to grab the nearest $1[\mathrm{nM}]$ spectral bucket from a high-resolution spectral representation that is pre-computed for all spectral data in the system. For a production system, this is clearly impractical, which is why such renderers need to focus on very fast and on the fly spectral uplifting instead. Similarly, integration of genuine spectral data needs to be done with fast on the fly resampling, and possibly feature-preserving compression of the input data Peters et al. [2019] (paired with fast on the fly unpacking of this compressed representation).

Also, integration of fluorescence into such a workflow compounds the problems of finding a size-preserving internal representation that the Hero Sampling process can work with. Recent work has focused on GMM models for this Qingqin et al. [2021], but other approaches might reduce the size needed for this data even further. Crucial to any integration of fluorescence into a production workflow is to minimise the size of the re-radiation information, far more so than for the plain spectral data the system needs to work with.

Finally, it should be noted that the output step of any spectral rendering process will, in practically all normal usage cases, only require storage of the final result in a colourspace image. That is, the spectral information computed during the rendering pass will not actually be stored for further processing. The reasons for this are fivefold:

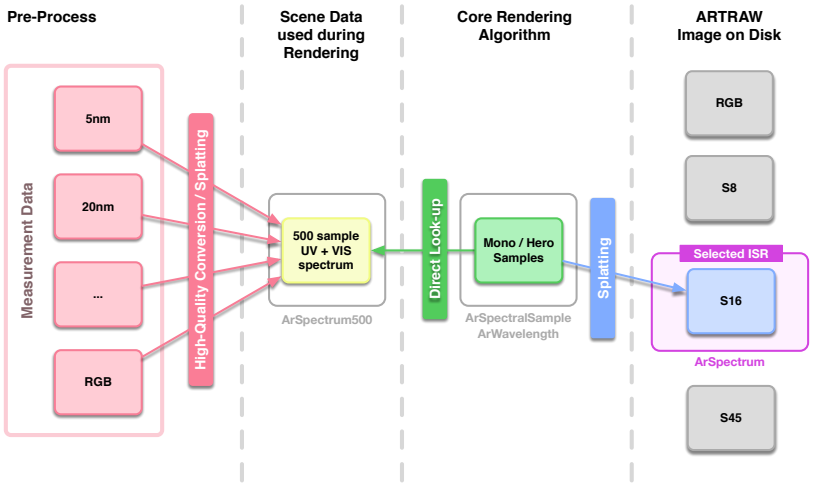


Figure 5.3: The spectral workflow in the renderer of ART 2.x. The user selects a particular spectral resolution (underlaid in purple: here, 16 spectral bands are chosen) for the output image. For path tracing, all other data is upsampled or uplifted to an internal form with 1[nM] spectral resolution. The advantage of this is that random wavelength sampling can directly grab any wavelength bin, and use it without any interpolation or further processing. The downside of this technique is memory usage, which is typically not an issue unless the scene contains image maps. Results of the core rendering loop are splatted into the result spectral image. Note that this result file could also just contain colour space data, if the user selects this. Also note that issues surrounding fluorescence are omitted from this diagram: while 1[nM] spectra are still doable for plain reflection and emission spectra, they are completely impractical for re-radiation matrices.

1. There is usually no need to retain spectral information, as standard VFX work only requires colour data from the rendering step onwards.

2. If one splats Hero Samples into a spectral image, one faces aliasing issues, and has to take care to splat the samples using a kernel that does not just use the nearest spectral bucket.

3. Which spectral resolution is sufficient to store a reasonably accurate version of the computed image is actually a non-trivial question in itself: but as each additional spectral channel has a direct image size penalty, there is a strong motivation to keep their number low.

4. Only lossless compression can be used on such images, as the impact of lossy compression on spectral data is still poorly understood.

5. Until very recently (see next section), there were no file formats for spectral image data that were suited for Computer Graphics work.

5.4 Spectral OpenEXR

In this section, we want to briefly acquaint readers with a recently published proposal for storing spectral reflectance and emission data within the framework of OpenEXR im-

ages Fichet et al. [2021]. The proposal is mainly about defining an unambiguous way to store spectral data as OpenEXR layers, in a fashion that even non-spectral image viewers can still gain some information from it. The proposal includes a clear delineation between reflective images (in other words, textures), and emissive images (the result of rendering computations). It even includes provisions for storage of bi-spectral reflectance data, and as such is the first standardised way to store and retrieve full re-radiation matrices for fluorescence work. As such, it can be used both for input to the rendering process, as well as an output format for the rendering step.

A noteworthy feature of the proposal is also that it retains a colour space version of the image in the OpenEXR file, so that standard, non spectrally aware image viewers can display a version of the picture.

We refer the reader to the publication for detailed information on how exactly the format is organised, and which software initially supports it. As it requires no modifications to the OpenEXR standard (all it does is provide an unambiguous way to organise such data), one can reasonably expect it to see some use within the rendering community.

References

Alban Fichet, Romain Pacanowski, and Alexander Wilkie. 2021. An OpenEXR Layout for Spectral Images. *Journal of Computer Graphics Techniques (JCGT)* (2021).

Matthias B. Hullin, Johannes Hanika, Boris Ajdin, Hans-Peter Seidel, Jan Kautz, and Hendrik P. A. Lensch. 2010. Acquisition and Analysis of Bispectral Bidirectional Reflectance and Reradiation Distribution Functions. *ACM Trans. Graph.* 29, 4, Article 97 (July 2010), 7 pages.

Alisa Jung, Alexander Wilkie, Johannes Hanika, Wenzel Jakob, and Carsten Dachsbacher. 2019. Wide gamut spectral upsampling with fluorescence. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 87–96.

Michal Mojzík, Alban Fichet, and Alexander Wilkie. 2018. Handling Fluorescence in a Uni-directional Spectral Path Tracer. *Comput. Graph. Forum* 37, 4 (2018), 77–94.

Christoph Peters, Sebastian Merzbach, Johannes Hanika, and Carsten Dachsbacher. 2019. Using moments to represent bounded signals for spectral rendering. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–14.

Hua Qingqin, Alban Fichet, and Alexander Wilkie. 2021. A Compact Representation for Fluorescent Spectral Data. In *Eurographics Symposium on Rendering (EGSR)*. Eurographics Association.

Alexander Wilkie. 2018. The Advanced Rendering Toolkit. http://cgg.mff.cuni.cz/ART.

6

A Hero Beneath the Surface

LUKE EMROSE, *Animal Logic*

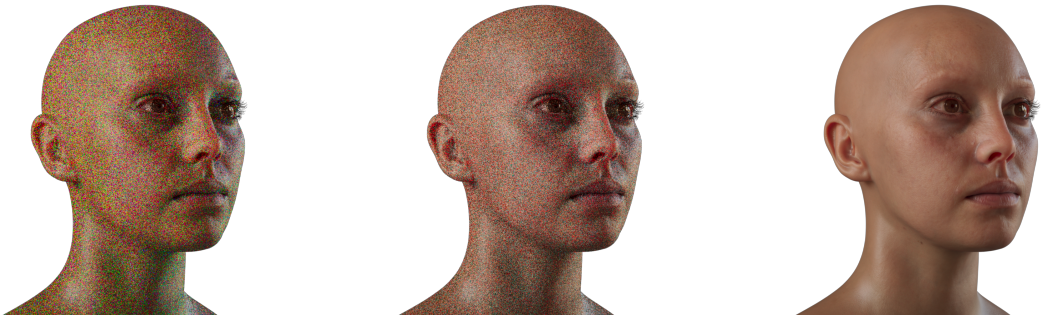


Figure 6.1: Left: 4spp single wavelength sampling noise, Middle: 4spp hero wavelength sampling noise, Right: 4096spp converged reference. Animal Logic added Hero Wavelength Sampling to *Glimpse*, its production path tracer, to improve the visual quality of Pathtraced Subsurface Scattering without increasing render times. Model: "DigitalEmily2" from USC Institute for Creative Technologies. Shading setup and texture map modifications by Jean Pascal leBlanc.

6.1 Introduction

Tristimulus RGB rendering offers some practical advantages which offset the qualities of spectral rendering presented elsewhere in this course. The majority of artists have a deep familiarity with RGB images which are the primary representation used for image acquisition, distribution, authoring and editing. This is unlikely to change any time soon.

Spectral rendering is simulating a continuum of spectral values over a defined range of wavelengths. Hence it consumes more memory and computation time than tristimulus rendering to an extent dependent on the accuracy of spectral detail. It generates higher variance due to the additional spectral noise it creates. For tools where high-performance interactivity and constrained memory is a concern, tristimulus rendering will be with us for some time to come. As a result, retro-fitting some of the advantages of spectral techniques into a tristumulus renderer is a reasonable approach, and one that Animal Logic has recently undertaken.

Animal Logic developed its in-house production pathtracer *Glimpse* in 2014 for *The Lego Movie* [Heckenberg et al. [2017]]. Having been used on every production since, it has continued to evolve and expand in its capabilities. *Glimpse* is primarily an RGB tristimulus renderer, but does utilize spectral additions for specific features such as thin film interference [Belcour and Barla [2017]] and lens chromatic aberration [Hullin et al. [2012]].

6.2 Not All Heroes Wear Capes

For an upcoming slate of projects, Animal Logic decided to overhaul *Glimpse* with more advanced and accurate subsurface scattering. Subsurface scattering captures the way light repeatedly bounces within a translucent object. Prominent real-world examples are skin, candles, fruit and jade. Traditionally this effect has been achieved using the dipole approximation: [Jensen et al. [2001], Frisvad et al. [2015] and Christensen [2015]]. In short, subsurface scattering is approximated [Fig. 6.2] as a radial falloff (r) from the entry point of light incident on a geometric surface by use of "real" (p) and "virtual" (v) point sources. This radial falloff can be importance sampled using efficient techniques such as: King et al. [2013]. It is fast to compute (since we are importance sampling a relatively simple, well-behaved smooth function), and for simple geometry with a lack of surface micro-detail or deep concave features, can look quite realistic. For more complex geometry with micro-details (like human skin), wavelength-dependent anisotropic scattering (dipole approximations are almost always isotropic approximations) and heavily concave features (like a human nose), pathtraced subsurface scattering [PTSSS: Chiang et al. [2016]] is a superior technique [Fig. 6.2]. PTSSS works by simulating the full random walk that a ray performs within a geometric object as it bounces and scatters within its internal structure. As a result, it takes a lot longer to render (since the random walk length is both not known in advance, could be extremely long, and the computational effort for each bounce is non-negligible), but is capable of accurately resolving light scattering for any object topology.

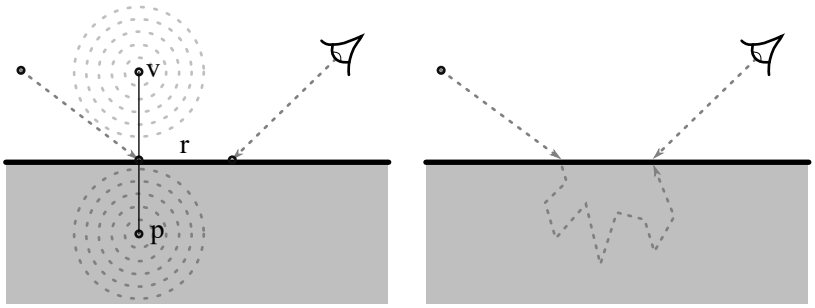


Figure 6.2: Left: the dipole approximation. Right: pathtraced subsurface scattering. Figure based on [Jensen et al. [2001]].

6.2.1 Motivation

When artists began to experiment with an early implementation of PTSSS in *Glimpse*, they reported what they termed "rainbow noise" [Fig. 6.1]. This noise is a by-product of performing wavelength-dependent sampling by choosing a single wavelength for the entire path,

and setting the contribution for other wavelengths to zero. Since a unique random stochastic sample is chosen per wavelength, this results in spectral noise. This noise is particularly noticeable at small sample counts (the contribution of each wavelength resolves variance independently) and can make interactive tuning of lighting quite difficult due to the central limit theorem [Cook et al. [2007]]. This distorts the perception of colour and contrast until a sufficient number of samples have been accumulated. To address this issue we used hero wavelength spectral sampling: [HWSS: Wilkie et al. [2014]].

6.2.2 Hero Wavelength Spectral Sampling (HWSS)

HWSS works by choosing a "hero" wavelength for a path. This hero wavelength is used for all sampling choices, but other wavelengths are "carried along" for the ride. The final estimator is then computed to allow all wavelengths to contribute to the final path in an unbiased way. This significantly mitigates the rainbow noise, since all the wavelengths are able to contribute some energy to the final path contribution [Fig. 6.3]. An arbitrary number of spectral samples can be used, with the wavelengths spaced equidistantly around the stochastically chosen hero wavelength. In *Glimpse* we stochastically choose between R, G and B and use the fixed tristimulus wavelengths for the spectra.

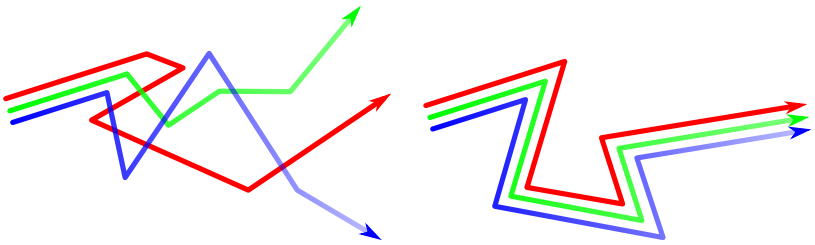


Figure 6.3: Left: single wavelength spectral sampling causes spectral noise due to diverging paths decoupling the spectral sampling noise. Right: hero wavelength spectral sampling reduces spectral noise by combining different spectral contributions along a shared path. Inspired by a similar diagram in [Novák et al. [2018]]

As per [Wilkie et al. [2014], Sec. 3.2 Eq. 11] HWSS gives us the following estimator, which combines the spectral samples along a path:

$$\langle I \rangle = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} \frac{f(X_i, \lambda_i^j)}{\sum_{k=1}^{C} p(\lambda_i^k) p(X_i | \lambda_i^k)} \tag{6.1}$$

By inspection, the connection to the balance heuristic [Veach [1998], Sec. 9.2.2.1] can be intuited. Other research in the spectral field suggests a similar formulation [Radziszewski et al. [2009]]. As you can see in [Fig. 6.4] the technique provides a significant improvement in spectral noise, especially for low sample counts, and even when used in a tristimulus renderer.

6.3 With Great Paper Comes Great Responsibility

Given the problem (spectral noise in PTSSS), a potential solution (HWSS) and the machinery to calculate the mathematics to be turned into code [Eq. 6.1], one could be forgiven for thinking that the renderer implementation should be easy. Not all is as it seems, however....

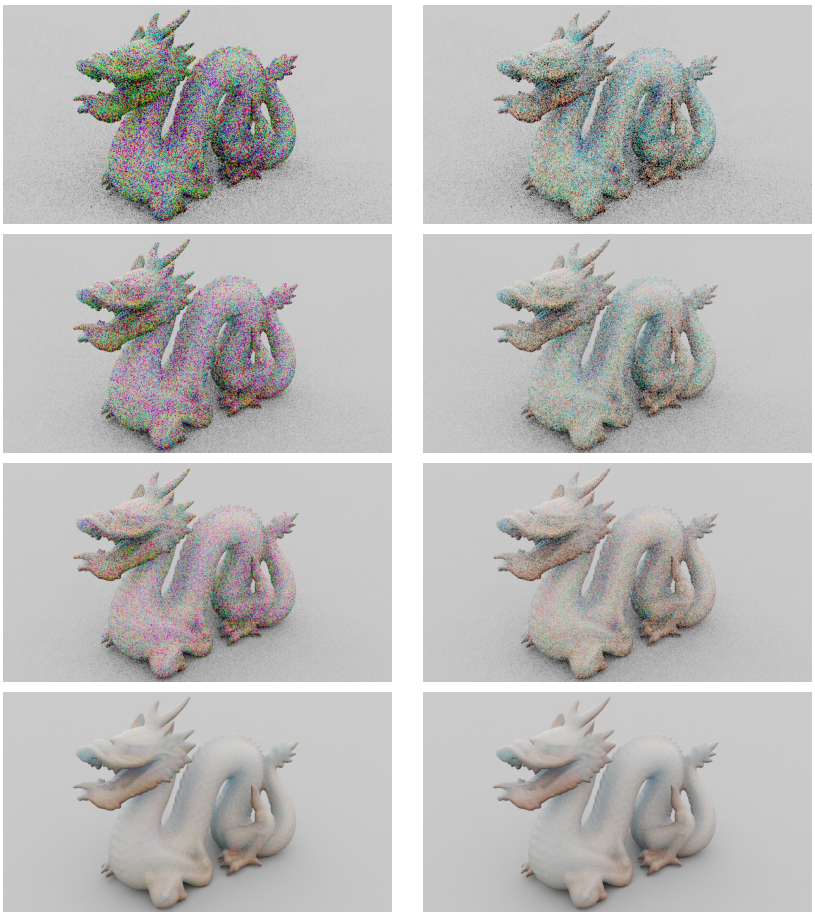


Figure 6.4: Left: single wavelength spectral sampling for 1, 2, 4, 64 samples per pixel. Right: hero wavelength spectral sampling for 1, 2, 4, 64 samples per pixel. Model: "Dragon" from Stanford University Computer Graphics Laboratory.

6.3.1 The Precision of Imprecision

An efficient single wavelength sampling implementation is able to make use of mathematical cancellation to turn the full path formulation into a trivial chain of albedo multiplications. Consider a scenario where subsurface scattering uses the Beer-Lambert law for distance sampling (exponential falloff), and an arbitrary but perfectly sampled directional function (where $f(\omega) = p(\omega)$) [all of which is covered in: Novák et al. [2018]]:

$$\langle I \rangle = \frac{f}{p} = \frac{f(t)}{p(t)} . \frac{f(\omega)}{p(\omega)} = \frac{\mu_s T(t)}{p(t)} . \frac{p(\omega)}{p(\omega)} = \frac{\mu_s e^{-\mu_t t}}{\mu_t e^{-\mu_t t}} = \frac{\mu_s}{\mu_t} = \alpha \tag{6.2}$$

This implies that a simple estimator simply needs to multiply at each path interaction with the single scattering albedo α. Computationally and numerically, this is well-behaved (no possible division by zero, infinities, or NaNs given a similarly well-bounded single scattering albedo).

Alternatively with HWSS we need the components of the estimator to compute [Eq. 6.1], and hence similar simplifications cannot be used. As a result, we are multiplying the function and its probability by an exponential function at each scattering interaction

[Radziszewski et al. [2009] Eq. 12]. If we store these quantities in 32bit floating-point registers, for a diffuse mean free path (DMFP) matching human skin [Jensen et al. [2001] Fig 5(b)] (0.68mm-4.82mm, let's choose 1mm), with an albedo of 1, a cumulative path length of only ≈8.73365cm takes us all the way to the smallest 32bit floating-point value of $1.175\,494 \times 10^{-38}$ [see Eq. 6.6]. To deal with this representation issue, we can inspect how f and p are used.

Let us assume a tristimulus HWSS implementation. The function values will be: (f_r, f_g, f_b), with associated probabilites: (p_r, p_g, p_b) and wavelength probabilities: $(p_{\lambda_r}, p_{\lambda_g}, p_{\lambda_b})$. Using [Eq. 6.1] we see that:

$$\langle I \rangle = \frac{(f_r, f_g, f_b)}{p_{\lambda_r} p_r + p_{\lambda_g} p_g + p_{\lambda_b} p_b} \tag{6.3}$$

Multiplying the numerator and denominator by a constant β will not change the value of the resulting estimators:

$$\langle I \rangle = \frac{(\beta f_r, \beta f_g, \beta f_b)}{\beta p_{\lambda_r} p_r + \beta p_{\lambda_g} p_g + \beta p_{\lambda_b} p_b} \tag{6.4}$$

Giving us a new set of working values: $(\beta f_r, \beta f_g, \beta f_b)$, and $(\beta p_r, \beta p_g, \beta p_b)$. The entire point of this observation is to avoid f or p getting too small (or large) before the computation of the estimator (to avoid single-precision floating-point underflow or overflow), hence we choose β to ensure that all of our intermediate quantities are as close as possible to a floating-point value of 1. To do this, at each bounce along our ray we compute:

$$\beta = \frac{1}{\max(f_r, f_g, f_b, p_r, p_g, p_b)} \tag{6.5}$$

The difference this makes to our result is remarkable. Without β it was found that double precision floating point was required to obtain accurate results. With the use of our per-step multiplier β, single-precision floating point is more than adequate [Fig. 6.5].

Figure 6.5: Left: 64bit double-precision reference HWSS. Middle: 32bit single-precision HWSS (note the incorrect colour tinting due to numerical precision issues; with a DMFP of (1,2,3) red and green underflow faster to zero than blue does, tinting the result blue). Right: 32bit single-precision HWSS with β factor used in computations provides identical results to double-precision at lower cost. [Eq. 6.5]. Model: "Dragon" from Stanford University Computer Graphics Laboratory.

6.3.2 Which Way Is Out?

After a ray begins a subsurface walk by hitting the input surface, the implementation needs to determine when the random walk exits to be able to correctly terminate. For closed

non-overlapping surfaces, this is trivial to determine, but for real-world production meshes, things get more complicated. Holes in meshes create problems [Fig 6.6], since random walks are then able to "leave" an object and walk around in the never ending ether outside of it. These issues were initially observed when artists described "coloured tinting" near self-intersections of character hand geometry [Fig. 6.4]. To solve this, we drew inspiration from the well established technique of nested dielectrics [Schmidt and Budge [2002]]. In our case we simply check the dot product between the subsurface ray direction R and the intersection with any geometry using the geometric normal N. Any time we hit geometry front-on: $R.N < 0$, we add one to a counter. Every time we leave a section of geometry by hitting it from the back: $R.N > 0$, we subtract one from a counter. When that counter hits zero, we know we are exiting the geometry. This works even in the presence of overlapping geo (like in the case of overlapping finger geometry). The result is that we are able to ignore all the intersections where our counter is > 0 and continue our random walk as if this internal geometry never existed at all [Fig 6.6]. In our experience, this gives the visual result that artists expect to see, and avoids potentially difficult and costly animation post-processing. This is obviously not robust for geometry with holes in it, which should still be handled with care.

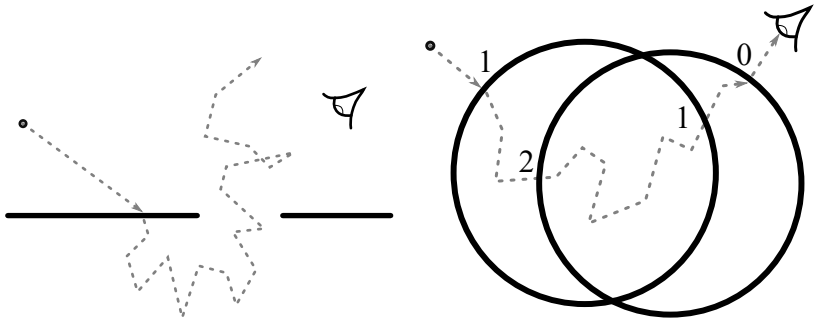


Figure 6.6: Left: holes in geometry can lead to random walks that never terminate. Right: overlapping geometry can be handled via intersection counting as long as the surfaces are closed.

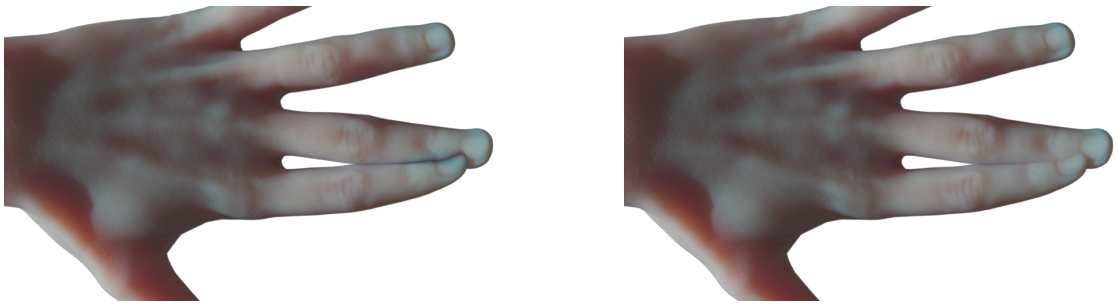


Figure 6.7: Left: incorrect handling of overlapping geometry can cause issues. Right: counting interfaces to skip over internal geometry gives a more artistically desirable result for non-ideal scenarios often encountered in real-world projects. Model: "Hand" from Artec Group inc.

6.4 Conclusion/Hero Advice

We introduced the Hero Wavelength Spectral Sampling (HWSS) technique as a useful method for reducing spectral noise even within a tristimulus renderer such as Animal Logic's *Glimpse*. This highlights the applicability and value of spectral techniques and methods in specific situations within a rendering process. Along the journey we saw different ways to address practical issues that arise when implementing algorithms within a production renderer. Suggestions for dealing with numerical precision and less-than-ideal meshes were presented. There are many further improvements to reduce spectral and overall noise that could be investigated and implemented, such as zero-variance Dwivedi sampling methods: [Křivánek and d'Eon [2014], Meng et al. [2016]] and beyond [Keller et al. [2020]], and other as-yet undiscovered techniques that you, the reader, can tell us about in the future.

6.5 Derivations

Calculation for the distance at which a diffuse mean free path of 1mm causes a transmission value to hit the smallest 32bit floating point number. This is greatly simplified from what you might see in a production renderer, since it does not perform the full reflectance inversion, but for illustrative purposes, it is sufficient:

$$\begin{aligned}
\mu_t = \frac{1}{\text{dmfp}} = \frac{1}{1\text{mm}} = \frac{1}{0.001\text{m}} &= 1000 \\
T(s) = e^{-\mu_t s} &= 1.175\,494 \times 10^{-38} \\
-1000s &= \ln(1.175\,494 \times 10^{-38}) \\
s = -\frac{\ln(1.175\,494 \times 10^{-38})}{1000} &\approx 0.0873365m = 8.73365cm
\end{aligned} \tag{6.6}$$

Acknowledgements

- Daniel Heckenberg
  - Feedback, assistance and guidance
- Curtis Black, Jakub Jeziorski, Linas Beresna and Emanuel Schrade
  - Our amazing Glimpse team!
- Jean Pascal leBlanc
  - Assistance with shading and texturing
- Johannes Hanika
  - For inviting and encouraging me to write these course notes
- Andrea Weidlich
  - For infinite patience and assistance with this course

References

Laurent Belcour and Pascal Barla. 2017. A Practical Extension to Microfacet Theory for the Modeling of Varying Iridescence. *ACM Transactions on Graphics* 36, 4 (July 2017), 65. https://doi.org/10.1145/3072959.3073620

Matt Jen-Yuan Chiang, Peter Kutz, and Brent Burley. 2016. Practical and Controllable Subsurface Scattering for Production Path Tracing. In *ACM SIGGRAPH 2016 Talks* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 49, 2 pages. https://doi.org/10.1145/2897839.2927433

Per H. Christensen. 2015. An Approximate Reflectance Profile for Efficient Subsurface Scattering. In *ACM SIGGRAPH 2015 Talks* (Los Angeles, California) *(SIGGRAPH '15)*. Association for Computing Machinery, New York, NY, USA, Article 25, 1 pages. https://doi.org/10.1145/2775280.2792555

Robert L. Cook, John Halstead, Maxwell Planck, and David Ryu. 2007. Stochastic Simplification of Aggregate Detail. *ACM Trans. Graph.* 26, 3 (July 2007), 79–es. https://doi.org/10.1145/1276377.1276476

Jeppe Revall Frisvad, Toshiya Hachisuka, and Thomas Kim Kjeldsen. 2015. Directional Dipole Model for Subsurface Scattering. *ACM Trans. Graph.* 34, 1, Article 5 (Dec. 2015), 12 pages. https://doi.org/10.1145/2682629

Daniel Heckenberg, Luke Emrose, Matthew Reid, Michael Balzer, Antoine Roille, and Max Liani. 2017. Rendering the Darkness: Glimpse on the LEGO Batman Movie. In *ACM SIGGRAPH 2017 Talks* (Los Angeles, California) *(SIGGRAPH '17)*. Association for Computing Machinery, New York, NY, USA, Article 8, 2 pages. https://doi.org/10.1145/3084363.3085090

Matthias B. Hullin, Johannes Hanika, and Wolfgang Heidrich. 2012. Polynomial Optics: A Construction Kit for Efficient Ray-Tracing of Lens Systems. *Computer Graphics Forum (Proceedings of EGSR 2012)* 31, 4 (July 2012).

Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. 2001. A Practical Model for Subsurface Light Transport. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. Association for Computing Machinery, New York, NY, USA, 511–518. https://doi.org/10.1145/383259.383319

Alexander Keller, Pascal Grittmann, Jiří Vorba, Iliyan Georgiev, Martin Šik, Eugene d'Eon, Pascal Gautron, Petr Vévoda, and Ivo Kondapaneni. 2020. Advances in Monte Carlo Rendering: The Legacy of Jaroslav KřIváNek. In *ACM SIGGRAPH 2020 Courses* (Virtual Event,

USA) *(SIGGRAPH '20)*. Association for Computing Machinery, New York, NY, USA, Article 3, 366 pages. https://doi.org/10.1145/3388769.3407458

Alan King, Christopher Kulla, Alejandro Conty, and Marcos Fajardo. 2013. BSSRDF Importance Sampling. In *ACM SIGGRAPH 2013 Talks* (Anaheim, California) *(SIGGRAPH '13)*. Association for Computing Machinery, New York, NY, USA, Article 48, 1 pages. https://doi.org/10.1145/2504459.2504520

Jaroslav Křivánek and Eugene d'Eon. 2014. A Zero-variance-based Sampling Scheme for Monte Carlo Subsurface Scattering. In *ACM SIGGRAPH 2014 Talks* (Vancouver, Canada) *(SIGGRAPH '14)*. ACM, New York, NY, USA, Article 66, 1 pages. https://doi.org/10.1145/2614106.2614138

Johannes Meng, Johannes Hanika, and Carsten Dachsbacher. 2016. Improving the Dwivedi Sampling Scheme. *Computer Graphics Forum (Proceedings of Eurographics Symposium on Rendering)* 35, 4 (2016), 37–44.

Jan Novák, Iliyan Georgiev, Johannes Hanika, Jaroslav Křivánek, and Wojciech Jarosz. 2018. Monte Carlo Methods for Physically Based Volume Rendering. In *SIGGRAPH 2018 Courses* (Vancouver, British Columbia, Canada). Article 14, 1 pages. https://doi.org/10.1145/3214834.3214880

Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. 2018. Monte Carlo methods for volumetric light transport simulation. *Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports)* 37, 2 (May 2018). https://doi.org/10/gd2jqq

Michal Radziszewski, Krzysztof Boryczko, and Witold Alda. 2009. An Improved Technique for Full Spectral Rendering. *J. WSCG* 17, 1-3 (2009), 9–16. http://wscg.zcu.cz/WSCG2009/Papers_2009/!_2009_J_WSCG_No_1-3.zip

Charles Schmidt and Brian Budge. 2002. Simple Nested Dielectrics in Ray Traced Images. *Journal of Graphics Tools* 7 (01 2002). https://doi.org/10.1080/10867651.2002.10487555

Eric Veach. 1998. *Robust Monte Carlo Methods for Light Transport Simulation*. Ph.D. Dissertation. Stanford, CA, USA. Advisor(s) Guibas, Leonidas J. AAI9837162.

A. Wilkie, S. Nawaz, M. Droske, A. Weidlich, and J. Hanika. 2014. Hero Wavelength Spectral Sampling. In *Proceedings of the 25th Eurographics Symposium on Rendering* (Lyon, France) *(EGSR '14)*. Eurographics Association, Goslar, DEU, 123–131. https://doi.org/10.1111/cgf.12419

7

Spectral Rendering in Production at Weta

ANDERS LANGLANDS, *Weta Digital*

Manuka is Weta's production spectral pathtracer, that has been used to render all projects since around 2015 Fascione et al. [2018].

Manuka is a batch-shading architecture, which means that all shading is computed exactly once and the resulting material description cached on micropolygon vertices as closures, which are evaluated during the light-transport phase.

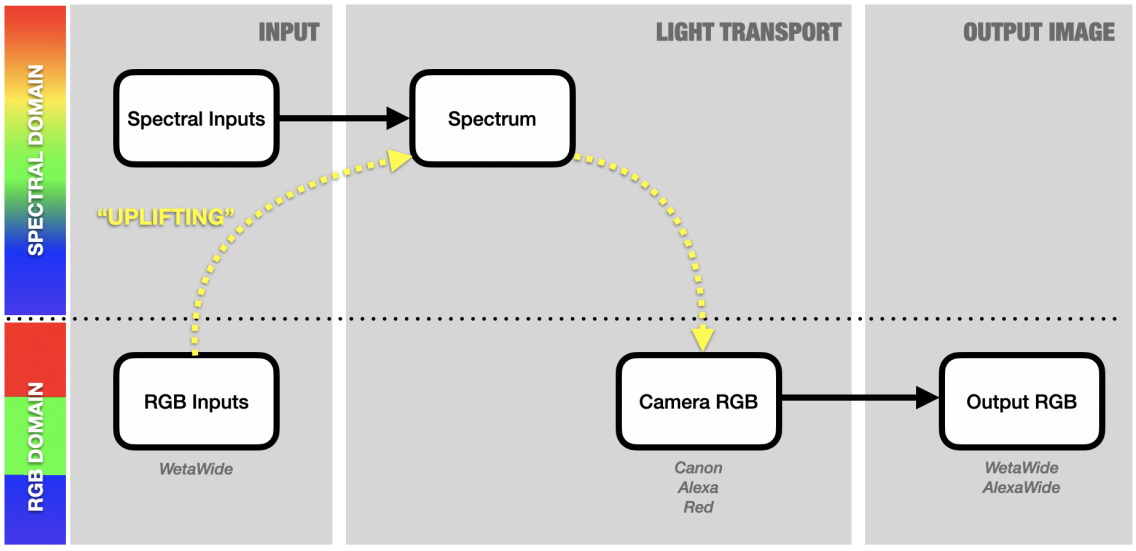


Figure 7.1: A high-level view of Manuka's colour pipeline.

Figure 7.1 shows a broad overview of Manuka's colour pipeline. Light transport is carried out entirely in the spectral domain. The resulting samples are converted to RGB using the process described in Langlands and Fascione [2020] before being written to image files on disk.

Inputs to the light transport phase such as shader and light parameters may be stored either as RGB or as piecewise spectral data. Inputs stored as RGB are "uplifted" to the spectral domain using the methods described in Section 7.2.

In order to sample the entire visual spectrum efficiently during light transport, we use hero wavelength sampling (Wilkie et al. [2014]), which samples four randomly offset, uniformly spaced wavelengths per path. This dramatically increases efficiency compared to single wavelength sampling, while avoiding the overhead of a densely binned approach.

For the majority of materials in production scenes, hero wavelength sampling is efficient enough that residual colour noise is mostly gone after eight or sixteen samples per pixel. Since our renders typically use hundreds or thousands of paths per pixel, colour noise is essentially a non-issue.

7.1 Input Parameters

In our shading and lighting tools, users normally interact with colour as RGB triplets, stored in a wide-gamut colour space we call WetaWide. Textures are also stored in the same space.

Calculations internal to the shader network are performed in RGB, with the final result being stored as a stack of BSDF closures (Fascione et al. [2018]) on each micropolygon vertex. With micropolygon counts in a typical scene being measured in the hundreds of millions, having a compact representation for colours in the BSDF stack is essential.

During light transport, when a ray intersects a micropolygon, the BSDF stacks from its vertices are interpolated, and any colours stored as RGB are uplifted to spectra before the stack is traversed to evalute the scattering event for the ray hit.

Some shader parameters may be stored either as piecewise spectra, or functions that evaluate to spectra, such as the blackbody function or Cauchy's transmission equation. These are simply evaluated directly at the four wavelengths being tracked by the current path.

Even when we have measured spectral data available, such as for melanin pigmentation in hair, or complex fresnel coefficients for metals, we typically choose to convert these to RGB before exposing them as parameters to the user. We find that in general (but not always!) it is more important to give artists control to tweak everything in an intuitive way than to adhere slavishly to the correct spectral curve.

Finally, emissive materials such as lights are represented by an emission spectrum—either D65, blackbody or a measured spectrum—multipled by an uplifted RGB tint value. This is conceptually similar to applying a gel to a practical light source.

7.1.1 Why All the RGB?

There are several reasons for working in RGB and uplifting at the last possible moment:

- Pretty much the entire computer graphics universe assumes that colour is represented by a tristimulus value, which includes all displays, artist-facing tools and common texture formats.
- Even a coarse 10-nanometer sampling of the visible spectrum requires an order of magnitude more storage for a single colour than the equivalent RGB representation. Multiplying that by several colours on each of hundreds of millions of micropolygon vertices would quickly become intractable.

- Artists think in RGB (in no small part due to all the tools operating that way). It is usually more important to give artists an intuitive way of working to tweak a colour how they desire rather than representing a given spectrum exactly.

7.1.2 Why Bother With Spectral Then?

Given that most of our inputs are RGB, why do we bother with spectral rendering at all?

- Where we're deriving RGB parameters from spectral data, the first thing we want to do is validate them against the original spectral data, so we obviously need a spectral renderer to do that.
- We need to be able to jump back to spectral again if we decide the loss of visual fidelity from an RGB representation is not worth the extra artist control from RGB. Where this line lies is different for each show, so we need to preserve maximum flexibility.
- Wavelength-dependent effects like dispersion and interference can be handled naturally within the spectral framework, with no hacks.
- Finally, given that the overhead of rendering spectrally is essentially a rounding error in the sort of scenes we render, why not use the most accurate colour representation available?

7.2 Uplifting

Until very recently we used Brian Smit's method (Smits [2000]—from hereon refererred to just as "Smits") for all our uplifting.

Rather than using the original Matlab curves from the paper, we solve the curves directly at render startup (Eq. 7.1) using a C++ FEM solver written by Marc Droske. This is mainly used to solve for specific emission curves for uplifting colours from HDR environment maps.

Smits's original formulation used CIE Illuminant E and the CIE 1931 observer functions. For our environment maps we solve curves using Illuminant D65 and the spectral response function of the camera that was used to capture the HDRI (Langlands and Fascione [2020]). This gives us much better round-trip accuracy. For emission we can also relax the $[0, 1]$ bound restriction on the curves to just constrain them to being positive and allowing them to go above one, improving round-trip accuracy further (see Figure 7.3).

$$C = M_{sRGB} \int f(\lambda) \cdot I \cdot W_{cam} \tag{7.1}$$

where:

$$\begin{aligned} \mathbf{C} &= \text{Primary to solve for, e.g. } red = [1,0,0], cyan = [0,1,1] \\ \mathbf{M_{sRGB}} &= \text{Camera space to sRGB matrix} \\ \mathbf{I} &= \text{Illuminant, typically D65 for IBL round trips} \\ \mathbf{W_{cam}} &= \text{Sensor response function, typically a Canon 5D} \end{aligned} \tag{7.2}$$

For reflectances we use the classic Smits Illuminant E/CIE XYZ combination. In theory, we could also use the sensor response of the capture device to generate improved uplifting curves for reflectance textures, but since those textures are always edited manually in some form by an artist, we have not explored this.

7.2.1 Gamut Clipping?

The inquisitive reader may now be wondering how, if we store our input textures in a wide-gamut colour space, we preserve that wide gamut when pushing all the colours through sRGB on their way to the spectral domain.

The answer is that we don't. All colours are transformed to sRGB, uplifted to a spectrum, then clipped to $[0, 1]$. The fact that this works in the general case is interesting in and of itself. An intuitive way to understand why this works is to compare the coverage of sRGB with the Pointer's gamut, which is a representative sample of approximately 4000 "real-world" colours, measured from photographic printing inks and television displays.

As can be seen from figure 7.2, sRGB covers the majority of our samples of real-world colours, with the missing areas being the more saturated colours which are less commonly found in typical sets and costumes. Note that transforming from WetaWide to sRGB generates out of gamut colours, but we clamp *after* uplifting to the spectral domain. This preserves a little more saturation than clamping to sRGB before uplifting.

Of course, that's not to say we *never* need those colours, and so we've been searching constantly for a method that meets our criteria of wide-gamut support, low computational overhead, and efficient storage.

After comparing the sigmoid method of Jakob and Hanika (Jakob and Hanika [2019]) and the Fourier moments method of Peters et al. (Peters et al. [2019]) we've chosen Fourier moments to replace our custom Smits solver for reflectances in recent versions of Manuka (Figure 7.4).

Fourier moments is able to round-trip reflectances for the full wide gamut (where it intersects the spectral locus) with only three coefficients, which can be effectively compressed for efficient storage on micropolygon grids. It is more expensive to compute than the Smits and sigmoid methods in micro-benchmarks, but this difference is not measurable in a real render.

7.3 Using measured spectral data on *Gemini Man*

In the pursuit of our highest-fidelity digital human yet, for *Gemini Man* we decided to use measured spectra for lighting as much as possible. This was primarily motivated by the need to get the most accurate match to the reference.

To create Junior, the young clone of Will Smith's character, Henry, we started with scans in the ICT and OTOY light stages that would be used for model, textures and materials reference.

The process of creating materials for a digital asset relies on removing as many extraneous variables as possible, so that one can be sure that when one sets a particular colour or roughness property of the surface, it's because the surface is that way and not because one is inadvertently matching some artefact of the lighting.

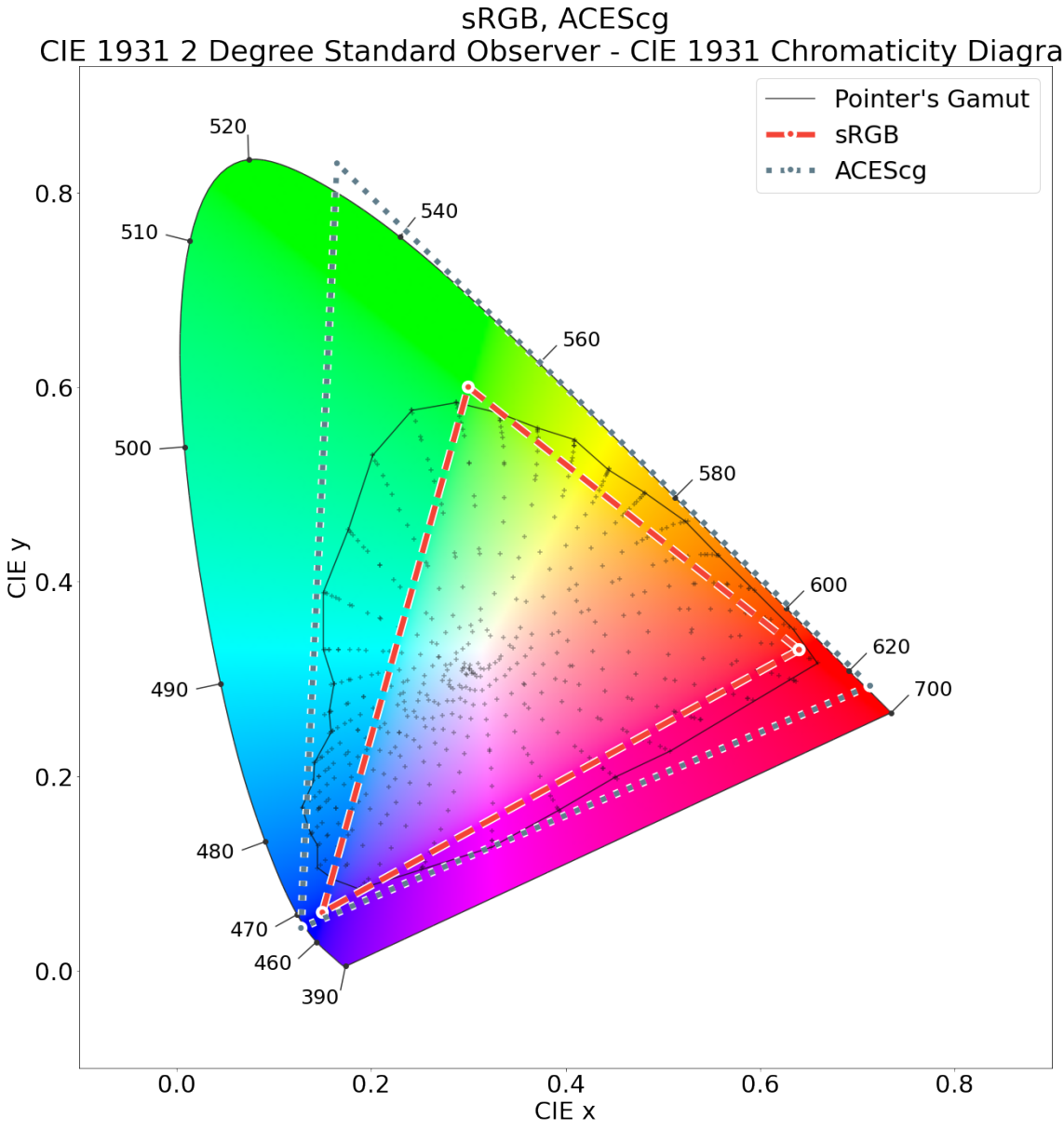
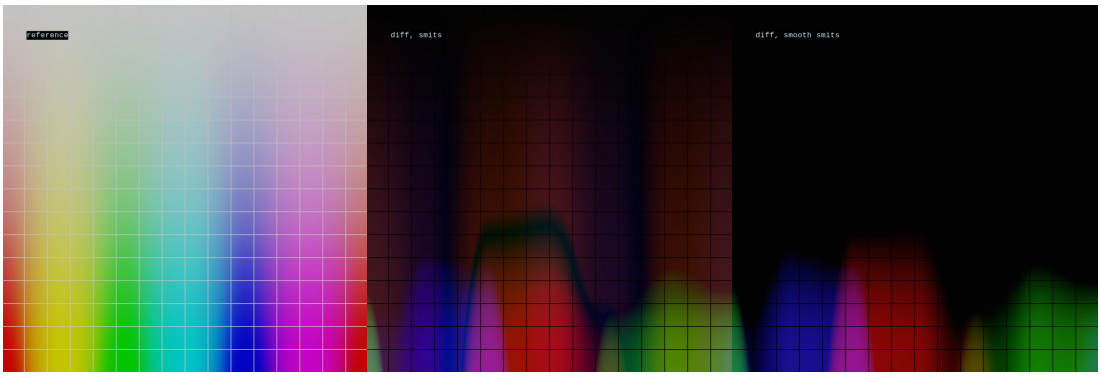


Figure 7.2: Pointer's gamut compared to sRGB

So to match Will's scan, we rebuilt the ICT and OTOY scanning rig in 3D, correctly positioning each LED light source, representing it as an area light with emission intensity mapped by HDR photographs of the source to capture the LED pattern.

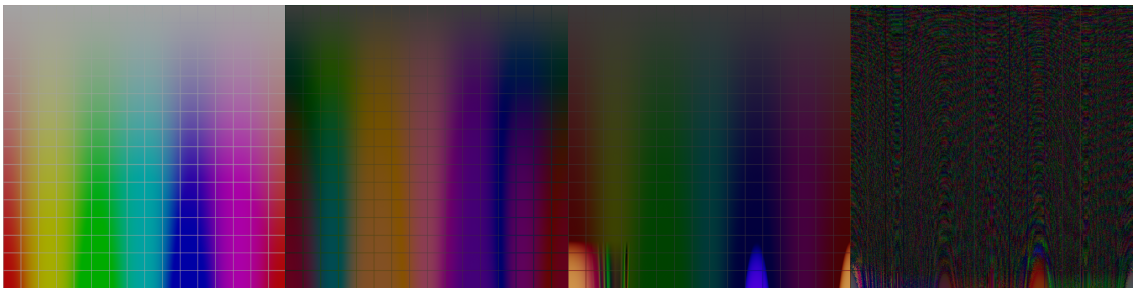
Finally, to ensure the colour matched as closely as possible, we measured the emission spectra of the lights with a spectrometer, and used that as the emission spectra for the lights in Manuka.

By representing the light stage as accurately as possible using measured spectra, we could have complete confidence that our material setup was accurate. This then led us to use measured light sources in the shots as well.



(a) (b) (c)

Figure 7.3: Comparison of round-trip spectral uplift for emission curves showing the reference (a) against 8x difference images of classic Smits (b) and our custom solver (c).



(a) (b) (c) (d)

Figure 7.4: Comparison of round-trip spectral uplift for reflectance curves showing the reference (a) against 16x difference images of Smits (b) sigmoid (c) and Fourier moments (d).

We measured as many sources from the lighting kit as we could on a dedicated VFX day at the start of photography. This consisted of taking measurements with the spectrometer and capturing matching HDRI images of the lightsource faces to be used for intensity mapping.

From these we created a library of sources that lighters could use in shots. Our typical shot lighting process is to start by verifying our IBL capture matches the reference balls, then extracting sources from the IBL into area lights and positioning these correctly to match the set. We would identify the sources being used from the IBL or from witness photography, and replace the extracted lights with correct ones from our measured library.

The rest of the IBL was still represented as a tint on a D65 illuminant since we don't currently have a way of refitting the IBL colour onto a given spectrum, and we assume that in most shots there is a mix of sources so the spectra would be smoothed out after a bounce or two.

7.4 Future Work

7.4.1 Spectral properties of RGB LEDs

LED lights with programmable colour output are now firmly part of the DoP's toolkit, for their flexibility in terms of choosing a precise colour, as well as their low thermal output and power requirements.

From the limited experiments we've done so far, it seems as though using an emission spectrum measured from the source on the "white" setting and then multiplying in a tint value gives us a reasonable approximation to the real emission spectrum of the light, but this needs further experimentation.

It would be helpful if manufacturers could provide spectra for their lights at a range of different settings.

7.4.2 Taxonomy of reflectance spectra

Different types of light sources such as incandescents, HMIs, LEDs and fluoroescents have wildly different and distinctively shaped emission spectra. As far as we know, there has been no investigation into whether reflectance spectra can be categorized by their shape, and whether those shapes relate to the material in question.

In other words, do organic materials, dyes, plastics or paint suspensions have characteristics in their spectra that are unique to those families of materials, and if so, how do those characteristics affect the perception of their colour, if at all?

7.4.3 Editing spectra

Once we have a taxonomy, could we use a library of spectral shapes and edit them in some way to define colours–maybe some sum of gaussians or a moments-based representation? Is there any practical advantage to doing this over uplifting a colour value?

7.4.4 Handling emissive textures

Currently, for texturing measured spectral sources we just multiply in the uplifted HDRI texture of the lightsource. This means we either need to desaturate it to avoid doubling up the colour of the light (because the measured spectrum itself defines a colour), or dividing out some "average colour" to try and preserve colour variation in the texture.

In theory, our custom uplifting solver could target the desired spectrum given the colours from the HDRI, but we need to investigate this further—it is unclear if the resulting spectra would be smooth enough for the Smits curves to represent effectively.

7.5 Conclusion

The combination of spectral uplifting and hero wavelength sampling in Manuka makes the fact that light transport is calculated in the spectral domain transparent to the user unless they decide to specify spectral data.

This allows us great flexibility in choosing how we trade colour accuracy for ease of use depending on the requirements of a particular show, and makes it easy to compare simplified solutions against a ground truth.

The overhead of uplifting from RGB to spectral is negligible compared to shading and light transport in a typical production scene, so we firmly believe the right question isn't "why should we go spectral?", but "why not?"

References

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

Wenzel Jakob and Johannes Hanika. 2019. A Low-Dimensional Function Space for Efficient Spectral Upsampling. *Computer Graphics Forum (Proceedings of Eurographics)* 38, 2 (March 2019).

Anders Langlands and Luca Fascione. 2020. PhysLight: An End-to-End Pipeline for Scene-Referred Lighting. In *ACM SIGGRAPH 2020 Talks* (Virtual Event, USA) *(SIGGRAPH '20)*. Association for Computing Machinery, New York, NY, USA, Article 19, 2 pages. https://doi.org/10.1145/3388767.3407368

Christoph Peters, Sebastian Merzbach, Johannes Hanika, and Carsten Dachsbacher. 2019. Using Moments to Represent Bounded Signals for Spectral Rendering. *ACM Trans. Graph.* 38, 4, Article 136 (July 2019), 14 pages. https://doi.org/10.1145/3306346.3322964

Brian Smits. 2000. An RGB to Spectrum Conversion for Reflectances. *Journal of Graphics Tools* 4 (06 2000). https://doi.org/10.1080/10867651.1999.10487511

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