

KEYWORDS VIRTUAL PRODUCTION // RENDERING // SPECTRAL RENDERING // FILMMAKING // SPECTRAL REDUCTION TECHNIQUES



FIGURE 1. Example of a VP shoot where there are real elements blended with virtual elements.

Spectral Rendering Techniques to Improve Color Matching in Virtual Production

By Vlad Simion, David Long, Flip Phillips, and Kris Murray

Abstract

As rendering engines become increasingly important in film and television, with their use in virtual production (VP), some underlying issues become more apparent. This paper aims to investigate how we can improve asset color matching of VP elements with real-life objects found on sets. Experiments were conducted in which objects were exposed to various types of lighting setups, and digital twins were rendered using both RGB methods and spectral methods, with data reduction techniques also employed. The renderings were then filmed, alongside their real-life counterparts. Color difference metrics were used to determine whether spectral rendering and data reduction techniques offered advantages over RGB renderings. The conclusion illustrates that spectral rendering offers advantages, including higher accuracy in rendering the colors of materials.

For many years, digitally rendered images have been utilized in a wide range of applications, including visual effects and virtual production. Virtual production (VP) is an emerging technology used in filmmaking where sets utilize digital backgrounds as a replacement for traditional green screen techniques.¹ Typically, projectors or LED panels are positioned behind the action and used to display virtual scenes. This allows actors to see and react to events in the scene and enables accurate lighting reflections. This also makes it possible to build smaller environments and extend them digitally. For example, creatives can extend bridges, walkways, or even entire worlds (**Fig. 1**). However, extending real-world assets into the virtual world introduces challenges in colorimetric consistency. Since virtual and real assets behave differently, it is often insufficient to simply match digital lighting parameters (e.g., color temperature, gains), requiring time and effort to alter them manually.

This paper aims to address these challenges by investi-

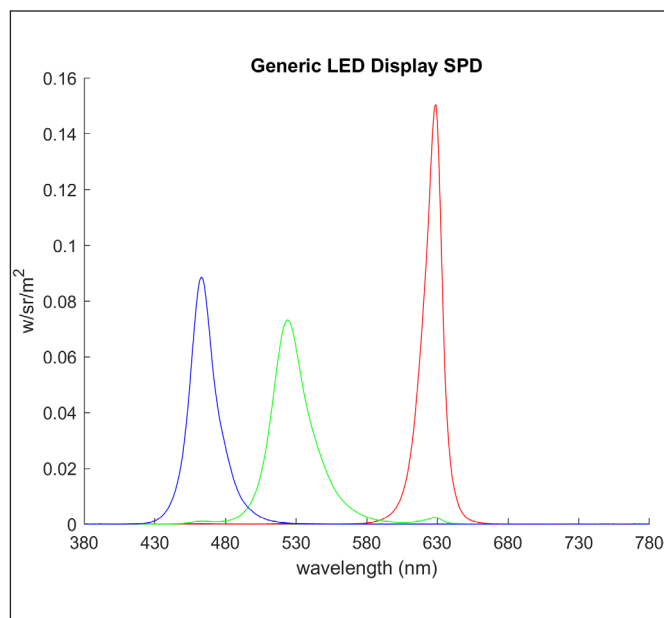


FIGURE 2. Generic LED display power distribution.

gating different color rendering methods. We will test how spectral rendering (full and abridged) compares to traditional RGB rendering within a practical environment that simulates virtual production workflows.

Most popular rendering engines used in the entertainment industry (Unreal, Unity, Blender) predominantly employ RGB rendering, where scalars are defined as Red, Green, and Blue encoding channels. Only engines that are used in more accurate or science-driven applications tend to be more spectrally accurate (e.g., Mitsuba or Manuka).^{2,3} However, even as early as the late 1980s,^{4,6} spectral rendering was being developed, but it has still not been widely adopted among popular rendering engines. This is because RGB is often sufficient to achieve results that artists like. Most modern rendering engines trace rays by computing their intersections with lights and objects in the scene and calculating RGB values that result from refraction, reflection, scattering, and transmission.² For example, rays are fired from the observer location, reflect off an object, and then hit a light source. A ray-object intersection is a “hit,” and a “shading” operation is performed to determine the color produced at that intersection. Thus, a ray has an associated RGB triplet at each step throughout the rendering pipeline, which ultimately determines the color of the pixel that represents that ray. If the objects are measured, then these RGB values are also computed from an observer’s perspective under a specific lighting condition, which makes them accurate only in certain environments. However, if the lighting in the rendering engine differs from the real-life measured lighting, or if the observer is swapped, then the RGB values should be recomputed from spectral data to ensure accuracy.

Real-world light is composed of photons within the visible spectrum, ranging from 380nm to 780nm. To quantify color vision, the CIE has adopted the XYZ color space. XYZ values contain information on how much light was detected across

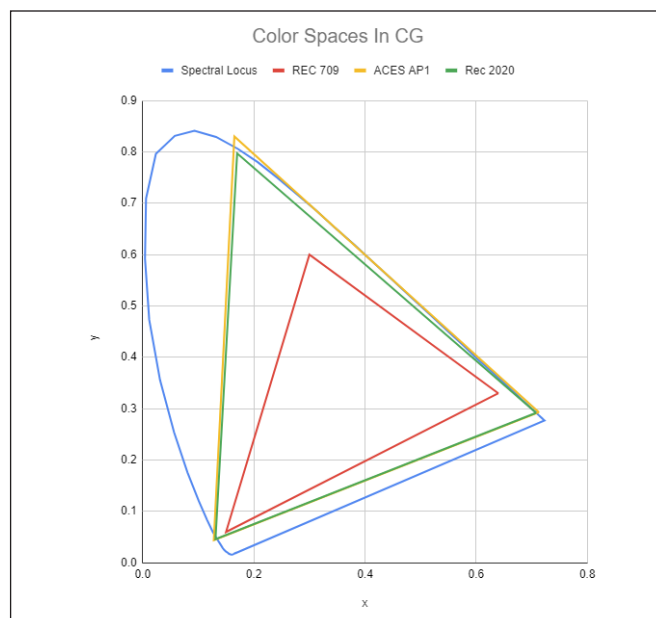


FIGURE 3. Color spaces used in computer graphics.

multiple wavelengths, thereby reducing the signal’s dimensionality. They are obtained by integrating light seen by the eye using color-matching functions,⁷ derived from the standard human observer as three integration curves. Not knowing the original spectra makes simulation challenging, as potential interactions between light and objects can be missed.

Because the eye has three types of retinal cells that integrate photons within a specific range, rather than determining their exact wavelength, stimuli with very different spectral compositions can integrate to identical XYZ values and be perceived identically by a common observer⁷ (i.e., metamerism). A common type of metamerism failure, known as illuminant metamerism, occurs when two surfaces match under the same illuminant but no longer match when the illuminant is changed. Another example is observer metamerism, which arises from differences between observers. There are almost infinite ways of reproducing the same color, as spectra can be infinitely mixed to create identical XYZ values. Displays take advantage of this, as devices with different spectra can recreate the same colors.

Cameras, however, have different sensitivities to light than the human eye because the filters in front of their pixels block different wavelengths than the human visual system, at varying ratios. This discrepancy stems from historically available materials and design practices. There is also a different amount of crosstalk between the channels of a camera and the crosstalk of the eye’s sensitivities. Paired with the absence of higher-order visual processing in the brain, cameras perceive the world differently than humans, leading to metamerism failures.

Most modern displays are LED-based and typically feature three diodes with different power distributions (**Fig. 2**). The display uses a combination of gains for each LED to create a range of colors. Because a camera has different sensitivities than the human eye, it perceives LED displays very differently. Narrowband emitters are especially troublesome, as

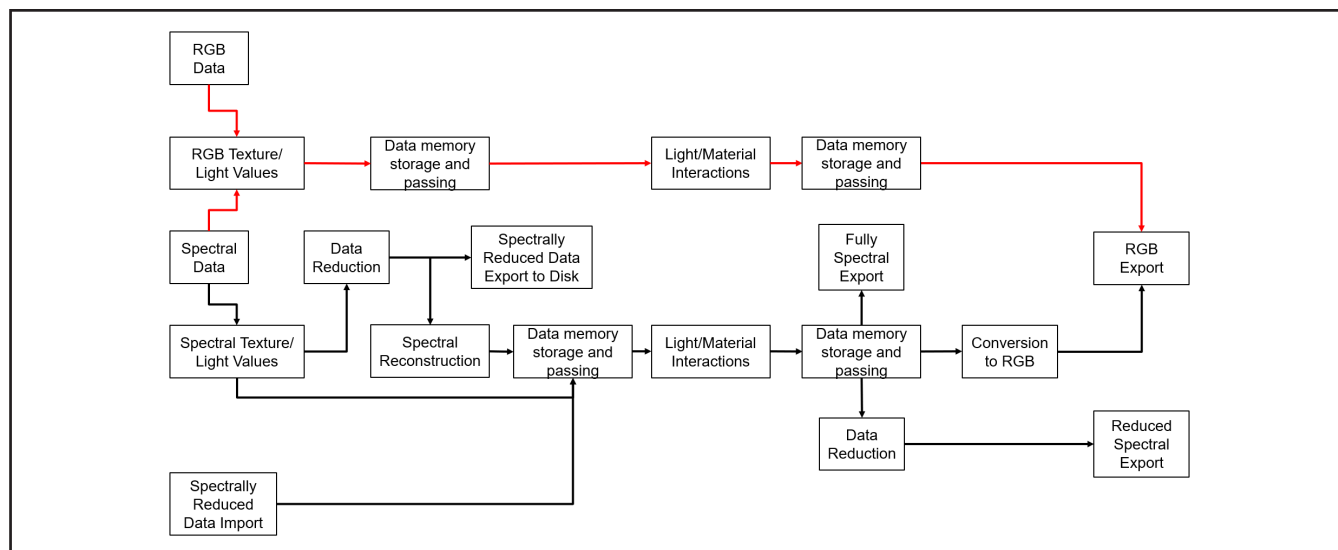


FIGURE 4. Rendering pipelines when using spectral rendering (black) versus pipelines that use traditional RGB rendering (red). The reduced spectral data can also be saved to storage for later recall, thereby skipping the analysis step and improving performance. The scene can also be exported as full-spectrum, reduced data, or converted to RGB.

even a small shift in the peak can cause large changes in the integrated color. This is a problem in current virtual production environments, where cameras are constantly pointed toward LED displays and often perceive both the real and virtual world in a way that differs significantly from the intended one. Simulating the real world using RGB values derived from XYZ values can further exacerbate this problem. This mismatch error is especially troublesome as it can break the illusion that creatives try to implement, as it makes the point where the two worlds collide visible. Currently, creatives attempt to conceal this intersection point using props (such as placing a plant that masks the blend point) or spend considerable time refining the virtual world to make the transition seamless and cohesive.

Background and Theory

Most popular rendering engines employ RGB rendering. We can convert XYZ stimulus values to estimate RGB values using a 3×3 matrix.⁸ However, this conversion introduces a fundamental limitation: RGB values can represent only colors within the triangular region formed by those three primaries in the chromaticity diagram. This is because RGB color spaces are based on additive color mixing, where any representable color must be expressible as a weighted sum of the three primaries (see **Fig. 3**). If any of the interactions of object and light would occur somewhere outside of the RGB space chosen, that interaction could not be represented.

We recognize that spectral information is crucial, and we have thus far observed how reducing it to RGB may be excessive. One goal of this research is to represent spectra using a limited number of terms, reducing both data and computational time compared to full spectral rendering while still offering benefits over traditional RGB techniques. The visible spectrum ranges from 380 to 780 nm, and most devices can measure spectra with precision every few nanometers; however, the data is reported as 400

data points. This may be taxing for each ray, but it can be condensed into fewer terms that describe the characteristics. The problem is twofold: information storage and interaction. The storage phase involves storing data about each asset's color and each ray. In contrast, the interaction phase is represented by computations that happen when a ray intersects an object. In the presented approach, light transport and storage could still occur by associating only a few values with a ray. In **Fig. 4**, we can see an example of the differences between spectral and traditional rendering pipelines. In traditional pipelines, we can only store and operate on RGB data. In spectral pipelines, several options are available, including using spectral data throughout the pipeline or applying data-reduction techniques to optimize the algorithm's memory footprint. To properly utilize the techniques, the full spectral data textures must be analyzed (or recalled from previous analyses). Ultimately, the engine can export either a fully spectral image, an image composed of reduced terms, or an RGB image converted from the spectral data.

As mentioned, carrying full spectral information is computationally expensive. However, most reflectance spectra can be represented with only five principal components with high accuracy.⁷ Principal component analysis (PCA) is a linear dimensionality reduction technique that aims to preserve important information and represent it as a set of new variables, known as components. Analyzing the principal components of a data set provides a new perspective, as we are more interested in variance between samples, given that they share common traits. This approach may represent a spectrum of light (or an object's material properties) using only a few terms.⁴ A simpler approach⁹ could separate the spectrum into "bins" (equivalent to increasing the spectral bandpass) that contain information from multiple wavelengths. Similarly, a few-term method⁵ could use each spectrum's polynomial representation in the transport

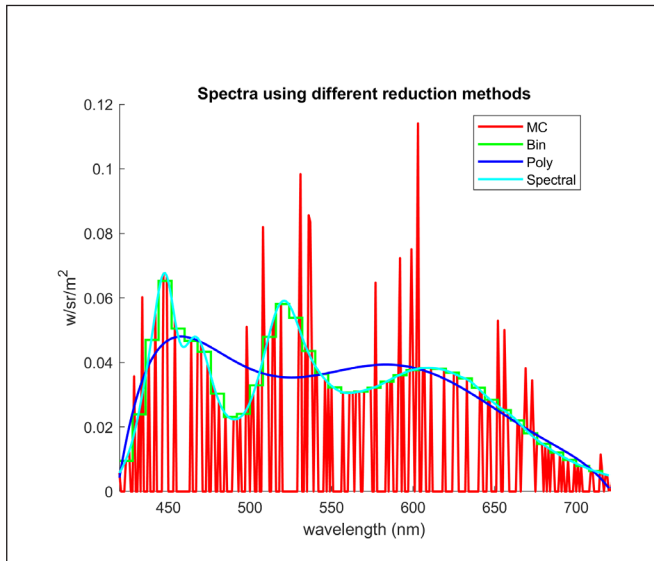


FIGURE 5. Different spectral reduction methods' effects on spectra.

and storage phases. The drawback is that when reconstructing the spectrum for the interactions, some loss of accuracy is inevitable. However, since most spectra are very smooth, this loss can be minimal. Lastly, the more common method is used,³ whereby each ray is assigned a random wavelength through an MC simulation and traced through the scene. The drawback is that many rays are needed to ensure all characteristics influence the computation. **Figure 5** illustrates an example of a spectrum obtained through a MC simulation, an example of an original spectrum, a polynomial function that is a "best fit," and an example of how the spectrum could be binned. For this research, the studied methods are:

- Full Spectral Rendering—the ground truth, typically at 1nm intervals¹⁰

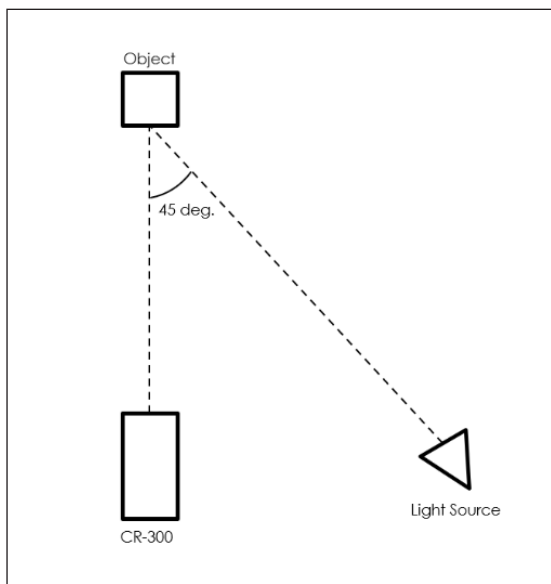


FIGURE 6. Measurement geometry

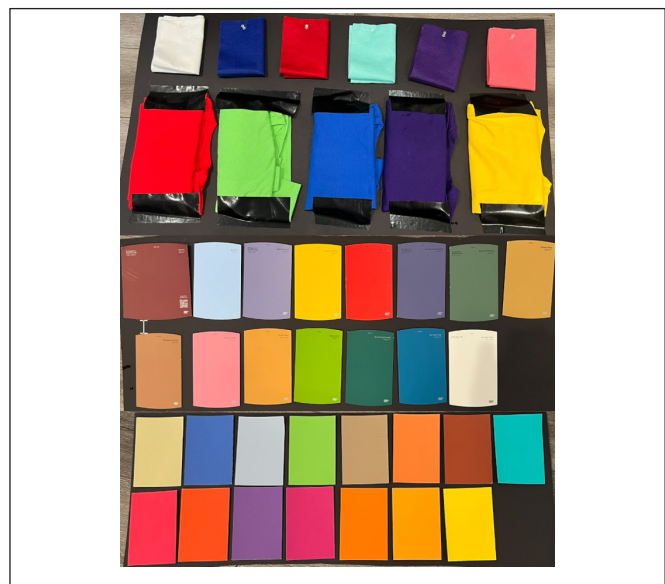


FIGURE 7. Textile samples (felts and shirts)(top), paint samples (middle), and color aid samples (bottom).

- PCA—Utilizing principal component analysis to reduce the size of the spectra^{4,7}
- Binning—Splitting each spectrum into a smaller number of bins that contain the information of multiple wavelengths.⁹
- Monte Carlo (MC)—Randomly choosing wavelengths for each ray until a good level of quality is reached³
- Polynomial—Each spectrum is represented by several polynomial terms⁵
- RGB—Traditional RGB rendering using an RGB color space

Traditional RGB rendering pipelines store information about asset color in a texture map. In a graphics pipeline, there is a finite amount of textures and pixels that a computer can have readily available. Storing full spectral information means that a significantly smaller number of textures can be saved, as their size will increase. For some of these methods, the full spectrum must be computed and then inverted to enable interaction, such as in PCA and polynomial methods, which can increase computation time. The interactions occur in a space composed of vectors with 400 terms, allowing us to sample each wavelength in the visible spectrum and match the measurement device's reporting. Relying on spectral and color accuracy alone is insufficient to determine whether spectral rendering offers benefits that outweigh its drawbacks. At a glance, here are some disadvantages and advantages of each method and how they might be implemented in a rendering engine:

- Full Spectral Rendering—all spectra need to have full spectral information saved, and each time a shader operation executes, 400-value length arrays are executed.
- PCA—All spectra in the scene are analyzed to determine the principal components, eigenvalues, and eigenvectors. Each time a shading operation is executed, the spectrum must be rebuilt. Then, a new analysis needs to be performed to determine the new principal components, and all eigenvectors are saved to memory.

- **Binning**—This allows both texture and shading operations to run on smaller arrays, increasing efficiency, though at the expense of wavelength-domain precision and masking highly dynamic spectral profiles.
- **Monte Carlo**—Textures are still stored as full spectra, but each shading operation is executed on one (or a few) wavelengths per ray.
- **Polynomial**—Spectra are analyzed, and best-fit functions are determined. The coefficients are then stored. The spectra are then undone so that the shading operations can occur. The spectra must be re-analyzed after the shading operation to store new coefficients. This adds computation overhead for constant fitting and inspection.
- **sRGB**—Traditional RGB rendering with each texture being represented as three values, and each shading operation happens on these three values.

To convey color differences more easily, the CIE CAM 16¹¹ model was selected. This model converts XYZ values into Jab values, where J represents luminance and a-b represent color difference channels. The distance between points in the CIE CAM model is the color-difference metric used to measure each method's accuracy, as shown in Equation 1. Since LCH was developed with CAM02, we apply the same logic and derive a C for use in CAM16, as shown in Equation 1. However, since we may want to consider how other observers perceive these differences, a more observer-agnostic method should also be employed. Root Mean Square Error (RMSE) quantifies the difference between the actual and computed spectra across all wavelengths.

$$C = \sqrt{a^2 + b^2}$$

EQUATION 1. C from a,b in CIE CAM16.

It is also worth mentioning how color is managed in virtual production. This experiment will attempt to replicate a typical on-set experience, and one major issue in VP is color management. The primary goal of VP is to create sets that mimic reality and appear to the camera as if the action were taking place in a real-world environment. The Virtual Art Department goes to great lengths to ensure that the environments appear natural and that lighting, texture, and feel appear real. As discussed earlier, camera metamerism failure causes the camera to perceive the LED wall differently than the human eye would. To address this, the industry predominantly uses a method described by EPIC Games.¹² The goal is to ensure the final RGB values in the camera footage match the RGB values the VAD team observed during the design phase, which are presumed to accurately represent the real-life object they are trying to emulate. They propose a simple approach to characterizing the interaction between the camera and the LED and creating a correction matrix that gets applied to the LED display to ensure proper calibration.

Methodology

The following experiments were conducted to investigate the hypothesis that spectral rendering can enhance the matching of virtual production assets. The first experiment involved measuring the samples to ensure that our rendering engine is accurate. The second experiment involved film-

ing the samples with a camera while they were displayed on screen, allowing us to observe the accuracy improvements of the renderings.

For the first experiment, we used a CR-300 spectroradiometer and a light source to measure the samples (**Fig. 6**). In total, we selected 66 samples (excluding fluorescent samples): a Macbeth chart (24), 11 Textiles, 15 Paint Samples, and 15 Color Aid papers (**Fig. 7**). They were each divided into separate groups, resulting in four distinct data sets. The measurement geometry was chosen to adhere to goniometric best practices⁷ and measure only the diffuse reflectance of the samples, thereby eliminating the specular component (if any). We used four light sources:

- Creamsource Micro Bender—Broad Spectrum
- Creamsource Vortex V4—RGBWW Calibrated Mode—Broad Spectrum
- Creamsource Vortex V4—RGB Uncalibrated Mode—Narrow Band
- Quasar Rainbow RGB Only—Narrow Band

These lights were chosen because they are representative of current lighting fixtures found on movie sets. Additionally, the RGB lights (**Fig. 8**) utilize narrow-band emitters similar to those found in LED displays and are prone to metameric failures. In contrast, the Micro Bender and the RGBWW mode offer energy broadly across the spectrum. We set the lights to 6500 K, the default white point (D65) of sRGB and commonly referred to as the white point of daylight. To generate more data for our analysis, we also repeated the experiments with the lights set to 3200 K, a common temperature on sets. Because each light type has a different spectral composition, we balanced them by adjusting the settings until they produced the same x and y chromaticity values within their capabilities and within the two separate CCTs, mimicking typical set practices.

In total, each sample was measured eight times, once under each fixture and at each temperature. The data from the meter is reported in 1nm intervals. Reflectance spectra for the stimuli were determined by normalizing the full spectral

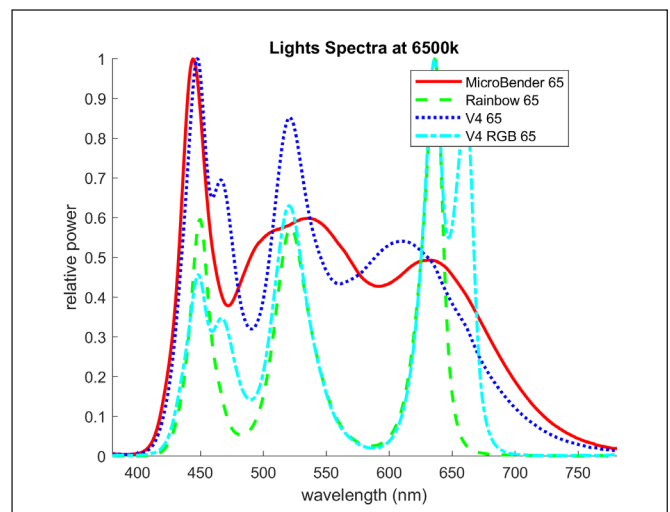


FIGURE 8. Peak normalized light spectra at 6500 K.

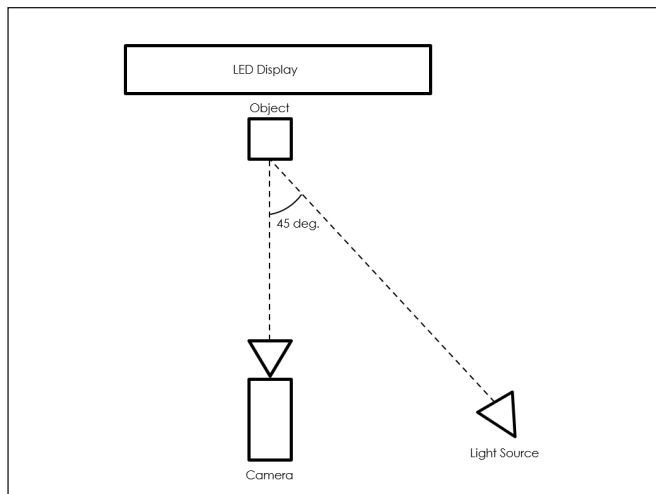


FIGURE 9. Filming geometry.

reflection from each sample under the broad-spectrum MicroBender and V4 lights to their respective sources, measured directly from a nominally perfect diffusing Spectralon reference. These measurements were performed with geometry matched as closely as possible to the photography to account for gonioscolorimetric effects.

The render engine was then fed with reflectances of each sample, and each light-image combination was rendered. The engine was coded in Python and mimics a typical ray tracer. Inside the engine, the samples were illuminated by a light also positioned at the same distance and angle as in real life. To simplify the process, we used a diffuse illumination model, and all assets were evenly lit. For the spectral methods, we used the reflectances from the previous experiment and the light spectra of each lighting device. For the RGB renderings, we converted the reflectances to RGB values, and we used the RGB values for a 6500 K light.

These light values were obtained by converting the average x,y values of the lights to ACEScg RGB values. On startup, each reflectance and light source is converted to fewer terms based on the method (or kept as spectral data). Each time an interaction occurs, we recalculate the spectra and compute the interaction, except for the RGB renderings, where we compute the interaction using RGB terms. For PCA, this became increasingly costly, as the components had to be recalculated after each interaction.

To assess the advantages in virtual production, we conducted a second experiment. A RED Raptor camera was used to film both the rendered images and the real assets in all setups. The camera was kept at the same settings during takes, using the device's native settings where possible. A volume of ROE CB3 LED panels displayed the images. To ensure consistent virtual production practices, the LED display was calibrated to accurately represent colors to the cinema camera, as recommended by EPIC Games¹² and by Murray and Simion.¹³ **Figure 9** shows the geometry used in filming. The digital images were scaled on the LED display so that the peak white patch matched

real-life values. The footage was analyzed using Nuke, Python, and MATLAB, and the values of the samples were converted to CAM16 values using default viewing environments. These CAM16 values are scene-referred and will be used later to determine how closely the camera perceived the VP-filmed renderings and the real assets.

Results and discussion

First, we evaluated the number of terms used in each method (see **Table 1**). The number of terms determines the accuracy of the methods and also the storage capacity. Ultimately, for all our spectral calculations, we saved and computed 301 values. These values fall between 420 and 720 nm, as our light sources produced light only within this interval.

Simulations were conducted in MATLAB until we determined that we had used as few terms as possible without sacrificing accuracy. We targeted a deltaE CAM16 of 2.0 on methods that did not present an apparent asymptote, as the asymptote methods seemed to level off around that value. This deltaE CAM16 was calculated by comparing spectra before and after reconstructions using the chosen methods. It was gathered from synthesized spectra-to-colorimetry under prescribed ideal illumination and viewed by a (CIE 1931 2° observer). This allowed us to determine the ideal number of terms under many conditions, not just the ones studied. Asymptotes indicate diminishing returns when additional terms are added to some methods, such as Binning and Polynomial. The results of our testing are presented in **Table 2** and **Fig. 10**.

TABLE 1. Terms per pixel in each method used.

Method	RGB	Bin	MC	PCA	Poly	Spectral
Terms Per Pixel	3	38	150	6	6	301

TABLE 2. Mean and Max deltaE CAM16 for method testing showing the relationship between the number of terms and deltaE CAM16 between original spectrums and reconstructed spectrums using the chosen methods.

Method	Terms	Mean deltaE CAM16	Max deltaE CAM16
Bin	25	5.89	9.21
	30	3.56	5.39
	35	2.40	3.88
	40	1.75	2.55
	45	1.75	2.60
MC	100	7.12	10.55
	150	6.00	8.90
PCA	4	2.65	5.00
	5	2.14	3.58
	6	1.59	2.66
	7	1.08	2.38
	8	0.63	1.28
Poly	4	4.06	8.51
	5	2.44	5.77
	6	2.08	5.34
	7	1.26	2.52
	8	1.19	2.34

TABLE 3. Average and Max RMSE between methods within sample sets and collapsed across all sets. The comparison was made between the radiance measurement and the predicted radiance, with all data peak-normalized using a perfect reflective diffuser.

Method	Macbeth	Textiles	Paints	ColorAid	Collapsed
Average					
PCA	3.43E-03	3.42E-03	4.33E-03	3.90E-03	3.77E-03
Poly	4.08E-03	4.23E-03	3.93E-03	4.61E-03	4.21E-03
Bin	1.17E-05	5.37E-05	1.17E-05	1.30E-05	2.25E-05
Max					
PCA	2.65E-02	2.16E-02	2.57E-02	2.16E-02	2.65E-02
Poly	1.17E-02	3.12E-02	2.95E-02	2.91E-02	3.12E-02
Bin	3.45E-05	6.11E-04	9.20E-05	1.06E-04	6.11E-04

TABLE 4. Average and Max RMSE between methods within lighting devices and collapsed across all devices.

Method	MicroBen der32	MicroBen der65	Rainbo w32	Rainbo w65	V432	V465	V4RGB3 2	V4RGB6 5	Collapsed
Average									
PCA	1.84E-03	2.30E-03	5.57E-04	5.31E-04	2.75E-03	2.89E-03	1.02E-02	9.12E-03	3.77E-03
Poly	1.51E-03	2.20E-03	6.16E-04	5.84E-04	2.19E-03	2.50E-03	1.35E-02	1.06E-02	4.21E-03
Bin	6.01E-06	7.68E-06	1.69E-06	7.01E-05	8.37E-06	1.15E-05	3.88E-05	3.62E-05	2.25E-05
Max									
PCA	3.79E-03	8.12E-03	1.60E-03	1.53E-03	6.48E-03	1.05E-02	2.65E-02	2.37E-02	2.65E-02
Poly	3.93E-03	7.32E-03	1.68E-03	1.61E-03	5.92E-03	8.54E-03	3.53E-02	2.90E-02	3.53E-02
Bin	1.27E-05	1.82E-05	6.02E-06	4.95E-06	1.85E-05	3.64E-05	1.06E-04	9.12E-05	1.06E-04
LightType	Wide-band	Wide-band	Narrow-band	Narrow-band	Wide-band	Wide-band	Narrow-band	Narrow-band	

TABLE 5. deltaE CAM16 statistics for all color samples and data reduction methods.

Samples	RGB	Bin	MC	PCA	Poly	Spectral
Average						
Macbeth	5.610	5.632	6.559	5.712	6.424	5.507
Textiles	9.624	7.585	8.326	8.562	7.856	7.827
Paints	7.252	6.423	7.404	6.861	7.003	6.102
Color Aid	9.937	7.058	7.865	7.544	7.445	7.179
Max						
Macbeth	10.132	11.852	14.669	10.210	12.609	10.589
Textiles	17.935	16.950	17.026	16.647	16.928	16.908
Paints	15.401	16.608	17.490	12.988	14.036	14.158
Color Aid	18.644	15.953	16.256	15.816	16.563	16.213

The spectral distributions resulting from the renderings were also analyzed. The Monte Carlo (MC) method was excluded because it sometimes failed to choose a wavelength, rendering the RMSE values inaccurate. The Spectral method was excluded because it consistently yielded results within a very small error, likely due to engine precision, similar to the measured data. By inspecting the max and average RMSE values (**Tables 3–5**), we can see that the binning method consistently outperformed PCA, which in turn outperformed the Polynomial method. With narrow-band emitters, PCA outperformed the Polynomial method.

Conversely, with broad-spectrum lights, the Polynomial method performed slightly better. This could be because polynomial methods struggle with spectra that have sharp peaks, as they require more terms to represent the values accurately. **Figure 11** illustrates various spectral reconstructions. One notable issue arises when using narrow-band emitters.

Some methods fail to preserve the peaks properly, most likely because they use fewer terms.

For the virtual production test, we reviewed the footage to determine if there were any improvements in accuracy. The RGB values of each patch were extracted and converted from camera-native values to linear ACEScg RGB values. The values were then converted to CIE CAM 16 values, and then the delta E between the real and VP values were calculated. The mean and standard deviation are presented in **Figs. 12 & 13**.

We can see that RGB consistently underperforms and has a higher deltaE CAM16 than any spectral method. As expected, the full spectral rendering is the most accurate, with the Binning method closely following. The polynomial method consistently underperforms. We should note that, when comparing the difference between the V4 set in narrow-band (**Figs. 14 & 15**) and broad-band modes

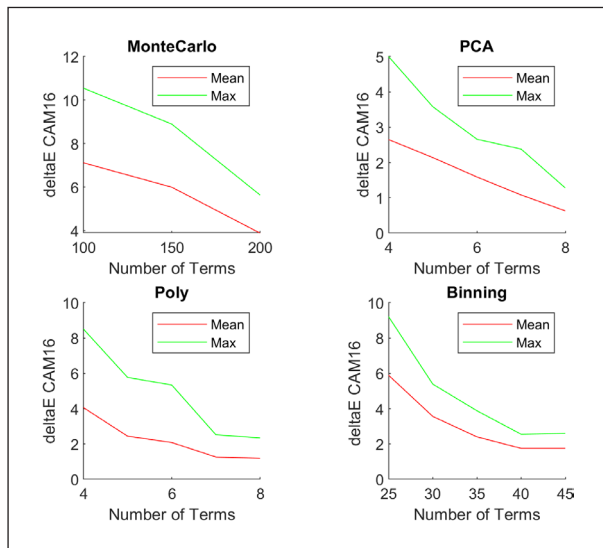


FIGURE 10. Number of terms affecting deltaE CAM16 plots, showing asymptotes that indicate diminishing returns when terms are added in certain methods, which influenced term decision.

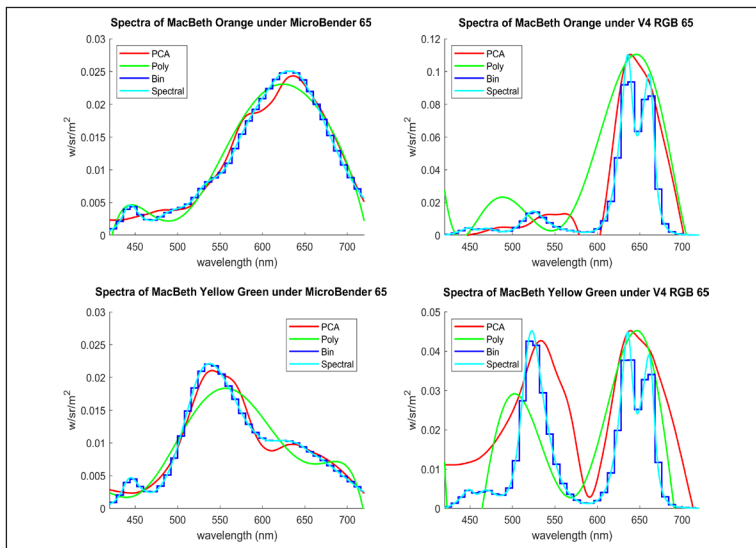


FIGURE 11. Different methods showing the reconstructed reflectance spectrum of Macbeth patches under different lighting conditions.

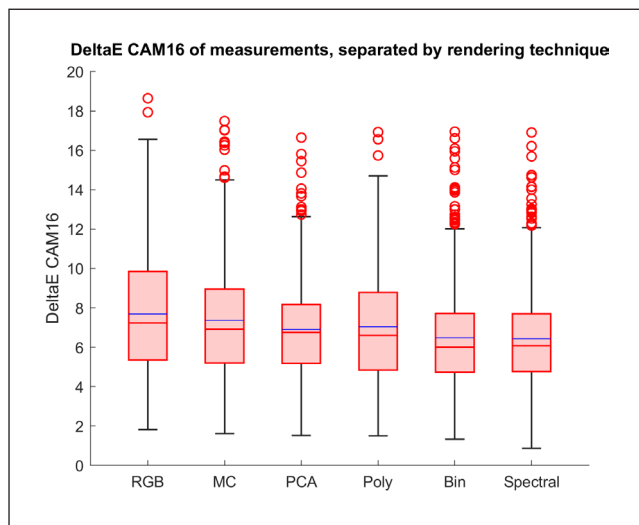


FIGURE 12. deltaE CAM16 data sorted by rendering methods with mean (blue), median (red), and lower and upper quartiles (red box), minimum and maximum data points not considered outliers (whiskers), and outliers (red circles).

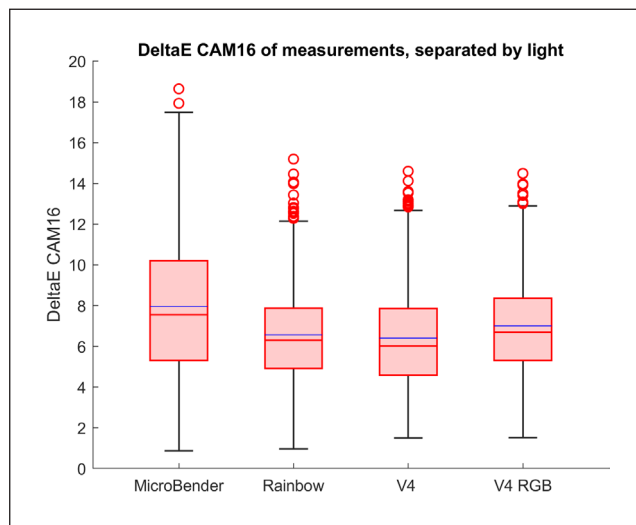


FIGURE 13. deltaE CAM16 data sorted by light device, with MicroBender and V4 being wide-band emitters and Rainbow and V4RGB being narrow-band emitters.

(Figs. 16 & 17), and when using PCA, deltaE CAM16 is higher in the narrow-band emitting mode. This could be because the narrow-band spectrum is much different from the smoother reflectance spectra, thus increasing the error, as we would need more eigenvectors to describe the behavior properly. The MC method also has a large deltaE CAM16 average, potentially due to noise, as the image seemed much less uniform.

When converting to CAM16 JAB, we used a Spectralon measurement for each light scenario as the scene illuminant value so data was not lost to clipping. These trends were observed across all datasets; therefore, we plotted only the Macbeth data at 6500 K for easier visualization.

For the visualization shown in Figs. 14 & 16, the dynamic range was matched during footage processing, as we

noticed that in some cases, the grayscale of the charts was not evenly distributed across the camera's dynamic range. The CAM16 plots also confirm the dynamic range issue identified during image visualization, which can also be attributed to lighting uniformity. While the charts in real life are not evenly lit, as some of the lights are brighter toward the middle of the chart, even with diffusion, the LED wall does not suffer from this issue, and it remains evenly bright across the entire surface.

The yellow patch in Fig. 14 shows this dynamic-range difference: it consistently looks darker than the surrounding patches, which could be due to luminance mapping from screen to camera. Interestingly, the ab color difference plots (Figs. 15 & 17) show no directional trends. This makes fixing the differences post-recording extremely difficult, as one

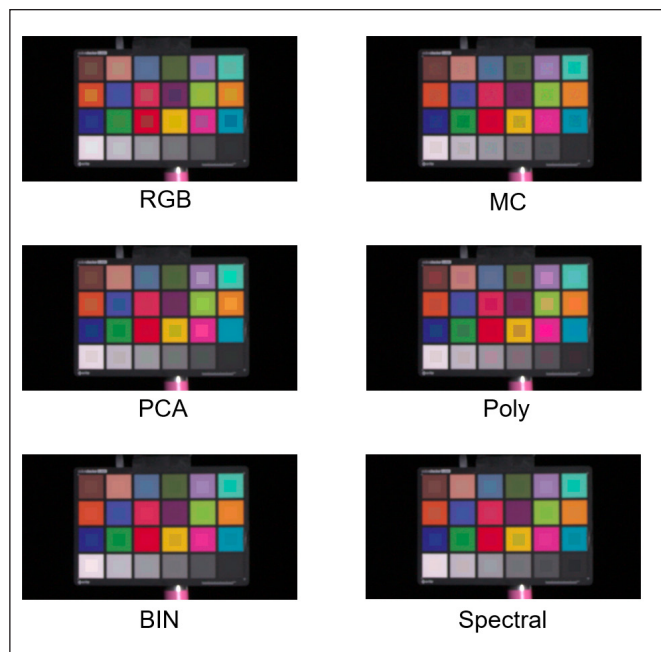


FIGURE 14. Example of the Macbeth chart illuminated by the narrow-band emitting V4RGB65 light source, after dynamic range matching. Outer patches are the real filmed samples, while the inner samples are the rendered, filmed versions.

would need to model the behavior and then create a proper fix. This can become very time-consuming, making it crucial and beneficial to address the behavior in the rendering.

Table 5 contains statistics gathered from the other samples. Further analyzing the data from the other samples reveals that they are similar trends of what methods perform better than others, and the lack of directionality of deltaE CAM16. Correlating this with previous RMSE data, the RMSE from the spectra accurately indicates which method will perform better in this experiment. Methods that do not preserve the spectrum well tend to underperform.

A few limitations influenced the data. First, the camera does not fully adhere to the Maxwell-Ives-Luther criterion, which means that the camera perceives stimuli differently than the human eye would. However, since this error applies to everything the camera perceives, and we are interested in differences between stimuli, it does not affect the conclusions. Our minimum observed average deltaE CAM16 is around 6. It might present itself as a large number; however, it originates from a multitude of sources that compound. Another factor is the accuracy of the calibration between the LED wall and the camera. The average deltaE CAM16 of the calibration is 1.63, with a maximum of 2.22. There may also be slight errors in our measurements due to variations in angles and light fall-off, as well as lens effects such as bloom or vignetting. To alleviate this, each sample was averaged over the entire area rather than selecting a single pixel. Other sources of error could also come from the LED panel drifting after being powered on, lights reacting to temperature differences after being powered on, or even the camera noise floor changing over time. All data from this paper is available upon request.

This research aims to inspire current industry trends to

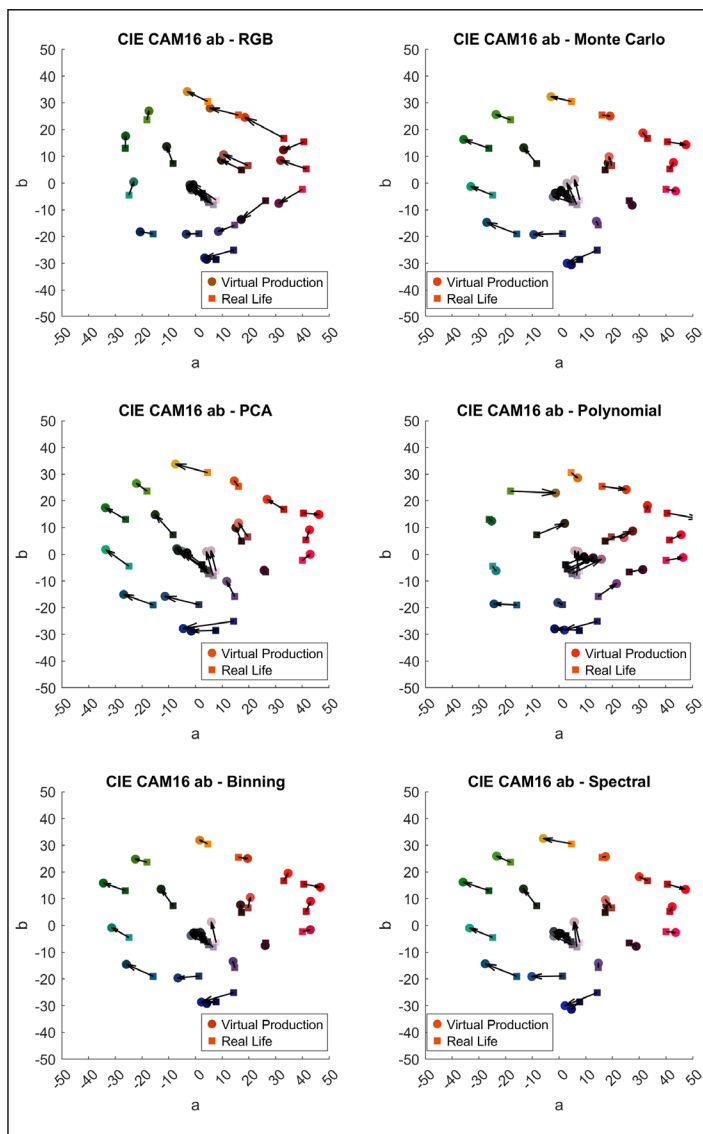


FIGURE 15. Differences in ab between real data and rendering data of the Macbeth from CAM16, as seen by the camera looking at the narrow band emitting V4 RGB 65. Each data point has the color of the patch it represents.

move toward a more spectrally accurate solution, which can alleviate color-matching issues between real and virtual assets. The paper demonstrated how pure spectral rendering increases accuracy; however, it is costly in both time and memory usage. Using methods that can compress spectral data helps reduce complexity, albeit at the cost of accuracy. Some of these methods performed as expected, with MC being largely a hit or miss, and PCA outperforming most methods but requiring more time. The polynomial method appears to be time-consuming (as it requires extensive fitting) and offers little benefit, as it sometimes creates impossible spectra that significantly increase the error. The binning method performed the best, and perhaps we can draw some inspiration from it. Further optimization of the number of terms could also yield more accurate results. There could even be a solution developed by combining different methods (for example,

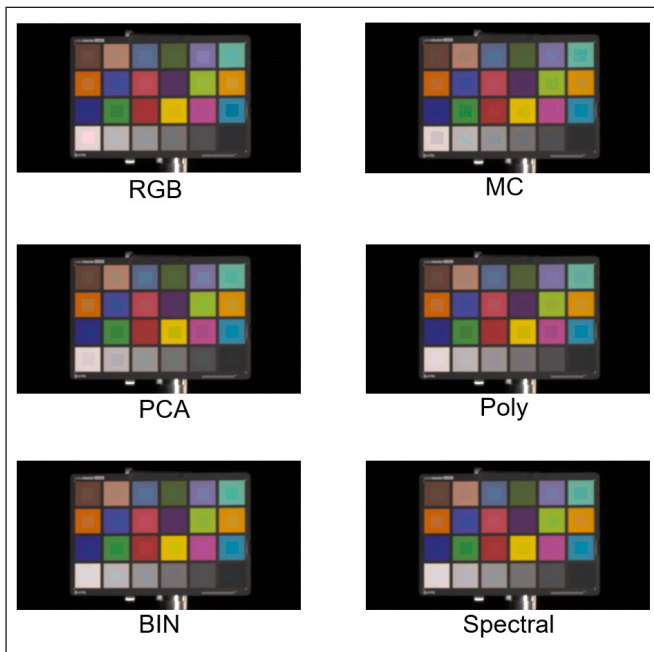


FIGURE 16. Example of the Macbeth chart illuminated by the wide band emitting V4 65 light source, after dynamic range matching. Outer patches are the real filmed samples, with inner samples being the rendered, filmed versions.

using a more optimized PCA but performing the interactions at larger intervals, more similar to binning). In a future iteration of research on these topics, one could also include the spectra of the LED and the camera to create RGB values that can be used in VFX compositing and “fix” the camera-to-wall calibration spectrally. Lastly, optimizing computing methods could tip the scales in favor of one method over the other in terms of time and memory usage. We recommend that the industry investigate implementing these methods in order to improve light simulation in rendering engines. The industry could also explore more spectral reconstitution methods that may create synthetic spectra from current RGB assets, thereby aiding in color matching under various illuminants.

Conclusion

This research showed that rendering assets spectrally can provide significant improvements over traditional RGB rendering in terms of matching real and virtual objects in virtual production. Through a series of experiments involving various lighting conditions and data reduction techniques, along with full spectral rendering, we demonstrated measurable improvements over traditional RGB rendering. These findings suggest that adopting spectral rendering approaches significantly improves the realism and consistency of digitally created assets in VP. With technology improvements over the last years, it may be possible to implement these techniques effectively and bring their benefits into production environments.

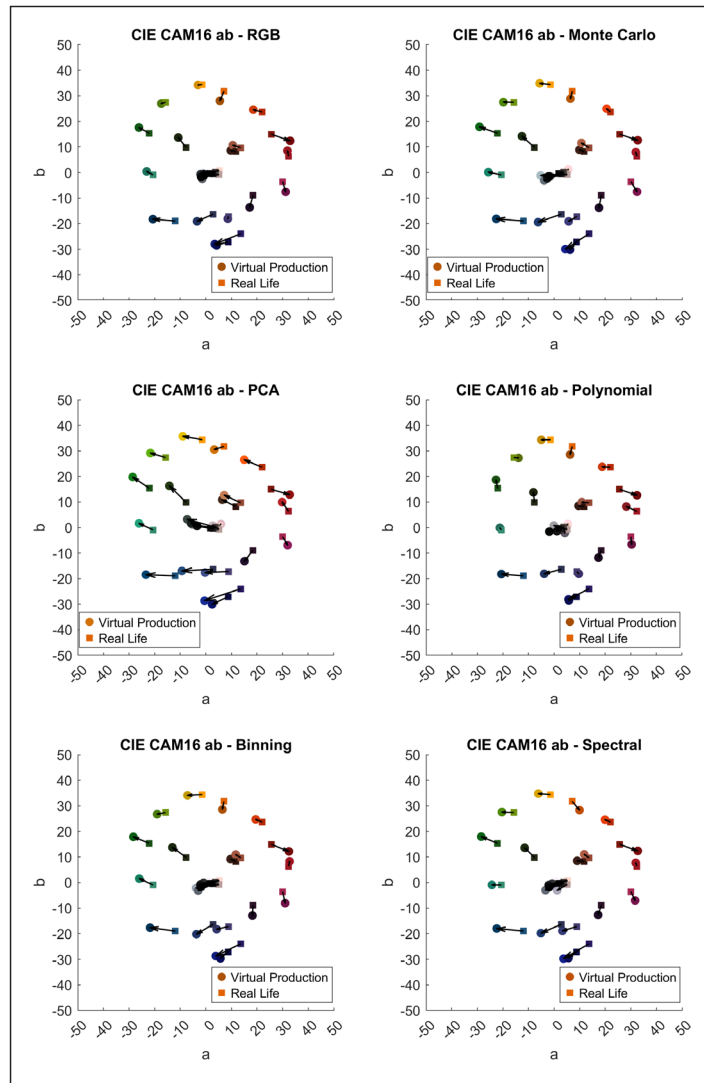


FIGURE 17. Differences in ab between real data and rendering data of the Macbeth from CAM16, as seen by the camera looking at the wide band emitting V4 65. Each data point has the color of the patch it represents.

Acknowledgments

The authors would like to acknowledge the RIT Munsell Color Science Laboratory, ROE, Megapixel VR, Creamsource, and Quasar for their support throughout this research. Flip Phillips supported, in part, by the National Science Foundation under Grant Award Number 2238180.

References

1. S. Zwerman and J. A. Okun, *The VES Handbook of Virtual Production*. New York, NY, USA: Routledge, 2024.
2. M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*. Cambridge, MA: MIT Press, 2023.
3. L. Fascione, J. Hanika, M. Leone, M. Droske, J. Schwarzhaupt, T. Davidović, A. Weidlich, and J. Meng, “Manuka: A Batch-shading Architecture for Spectral Path Tracing in Movie Production,” *ACM Trans. Graph.*, 37 (3): 1–18, Aug. 2018. doi: 10.1145/3182161.
4. M. S. Peercy, “Linear Color Representations for Full-speed Spectral Rendering,” *Proc. SIGGRAPH '93*, 1993, pp. 191–198, doi: 10.1145/166117.166142.
5. M. G. Raso and A. Fournier, “A Piecewise Polynomial Approach to Shading Using Spectral Distributions,” *Proc. Graphics Interface '91*, Calgary, AB, Canada, pp. 40–46, June 3–7, 1991. doi: 10.20380/GI1991.06.

6. G. W. Meyer, "Wavelength Selection for Synthetic Image Generation," *Comput. Vis. Graph. Image Process.*, 41 (1):57–79, Jan. 1988. doi: 10.1016/0734-189X(88)90117-X.
7. R. S. Berns, F. W. Billmeyer, and M. Saltzman, *Billmeyer and Saltzman's Principles of Color Technology*. Hoboken, NJ: Wiley, 2019.
8. G. Wyszecki, *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Hoboken, NJ: Wiley, 2001.
9. C. C. Borel and R. F. Tuttle, "Multi- and Hyperspectral Scene Modeling," in *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVII*, 2011. doi: 10.1117/12.884585.
10. G. M. Johnson and M. D. Fairchild, "Full-Spectral Color Calculations in Realistic Image Synthesis," *IEEE Comput. Graph. Appl.*, 19 (4): 47–53, Jul./Aug. 1999. doi: 10.1109/38.773963.
11. C. Li, Z. Li, Z. Wang, Y. Xu, M. R. Luo, G. Cui, M. Melgosa, M. H. Brill, and M. Pointer, "Comprehensive Color Solutions: CAM16, CAT16, and CAM16-UCS," *Color Res. Appl.*, 42 (6): 703–718, Nov./Dec. 2017. doi: 10.1002/col.22131.
12. Epic Games, "Camera color calibration for in-camera VFX in Unreal Engine," *Unreal Engine 5.5 Documentation*, Epic Developer Community. [Online]. Available: <https://dev.epicgames.com/documentation/en-us/unreal-engine/camera-color-calibration-for-in-camera-vfx-in-unreal-engine>
13. K. Murray and V. Simion, "Color management in virtual production," *ACM SIGGRAPH 2023 Talks*, 2023, pp. 1–2, doi: 10.1145/3587421.3595445.

About the Authors



Vlad Simion is a color scientist who works at Seisimiq, building color pipelines, imaging workflows, and color management software for virtual production, immersive, and emerging technologies.



David Long, PhD, is Director of MAGIC Spell Studios and Professor of Motion Picture Science at Rochester Institute of Technology.



Flip Phillips, PhD, is Professor of Motion Picture Science at Rochester Institute of Technology.



Kris Murray is CTO of 180 Studios and co-founder of Seisimiq, building virtual production workflows and technology tools for filmmaking and creative teams.

Paper submitted for Journal first publication and presented at the SMPTE 2025 Media Technology Summit, Pasadena, CA, 16–19 Oct. 2025. Copyright © 2026 by SMPTE.

DOI: 10.5594/JMI.2026/JGAM2497

Date of publication: 1 October 2026

Join the Board of Editors

Volunteer to help shape and maintain the Journal's high editorial quality.

email the editor



MI
MOTION IMAGING JOURNAL

