Cinematic Color

From Your Monitor to the Big Screen

SIGGRAPH 2012 Course Notes

Course Organizer

Jeremy Selan
Sony Pictures Imageworks

Updated: May 21, 2012
Latest Version Available Online

Course Description

This class presents an introduction to the color pipelines behind modern feature-film visual-effects and animation.

Color impacts many areas of the computer graphics pipeline. From texture painting to lighting, rendering to compositing, and from image display to the theater, handling color is a tricky problem. Tired of getting your images right on the monitor, only to have them fall apart later on? The goal of this course is to familiarize attendees with the best practices used in modern visual effects and animation color pipelines, and how to adapt these concepts for home use.

The course begins with an introduction to color processing, and its relationship to image fidelity, color reproducibility, and physical realism. We will discuss common misconceptions about linearity, gamma, and working with high-dynamic range (HDR) color spaces. Pipeline examples from recent films by Sony Pictures Imageworks will be included, with a discussion of what color transforms were utilized, and why these approaches were taken. Finally, we will present a brief introduction to the Academy's recent efforts on color standardization in computer graphics (ACES), and how the audience can experiment with all of these concepts for free using open-source software (OpenColorIO).

Level of difficulty: Intermediate

Intended Audience

This course is intended for computer graphic artists and software developers interested in visual-effects and animation.

Prerequisites

Familiarity with computer graphics fundamentals is preferred.

On the web

http://opencolorio.org

About the presenters

Jeremy Selan is an Imaging Supervisor at Sony Pictures Imageworks, specializing in color, lighting, and compositing. His work has been used on dozens of motion pictures, including *The Amazing Spider-Man*, *Alice in Wonderland*, and *The Smurfs*. He is one of the lead developers behind Katana - Imagework's lighting tool - and is also the founder of OpenColorIO. His work on color processing has been previously featured in GPU Gems 2 and Siggraph 2005's Electronic Theater.

Presentation schedule

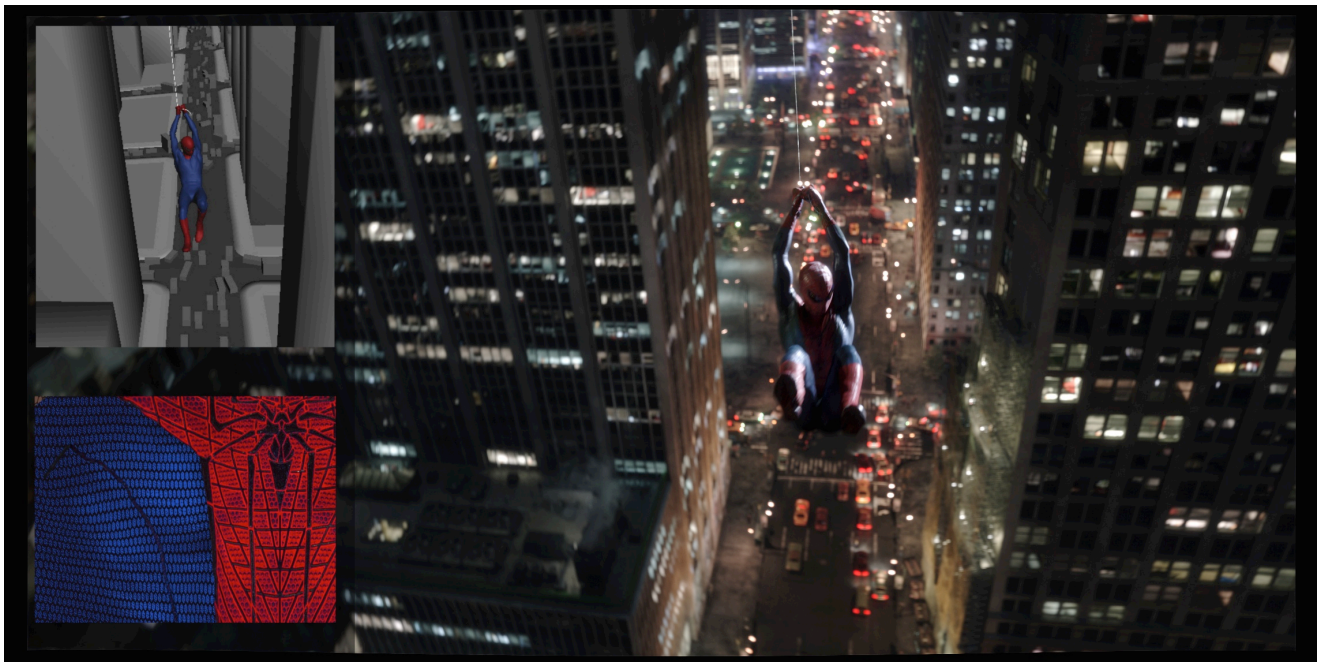
2.00 – 2.30 **Color Science**
2.30 – 3.00 **Motion-Picture Color Management**
3.00 – 3.30 **Implementation, Example, Q&A**

Table of Contents

1. Introduction

Practitioners of visual effects and animation encounter color management challenges which are not covered in either traditional color-management textbooks or online resources. This leaves digital artists and computer graphics developers to fend for themselves; best practices are unfortunately often passed along by word of mouth, user forums, or scripts copied between facilities.

This course attempts to draw attention to the color pipeline challenges in modern visual effects and animation production, and presents techniques currently in use at major production facilities. We also touch upon open-source color management solutions available for use at home (OpenColorIO) and an industry attempt to standardize a color framework based upon floating-point interchange (ACES).



This fully computer generated image touches upon many modern techniques in color management, including a scene-linear approach to rendering, shading, and illumination, in addition to on-set lighting reconstruction and texture management[1]*. Visual effects by Sony Pictures Imageworks. Images from "The Amazing Spider-Man", © 2012 Columbia Pictures Industries, Inc. All rights reserved.*

[1] This image, though dark, has good detail in the shadows. If these shadow areas appear flat black on your display, please confirm the display calibration and gamma.

What color management challenges are faced in visual effects and animation production?

Various Requirements: It is difficult to lump all of visual effects and animation into a single bucket, as each discipline has potentially differing color pipeline goals and constraints. For example, in visual effects production one of the "golden rules" is that image regions absent visual effects should not be modified in any way. This places a constraint on the color pipeline - that color conversions applied to the photography must be perfectly invertible. Animation has its own unique set of requirements, such as the elegant handling of saturated portions of the color gamut. Thus, color pipelines for motion pictures must keep track of the 'big picture' priorities, and are often tailored for specific productions.

Various Color Philosophies: There are *many* schools of thought on how to best manage color in digital motion-picture creation. (We assert there is far more variation in motion-picture color management than in desktop publishing, for example). Some facilities render in high-dynamic range (HDR) color spaces. Other facilities prefer to render in low-dynamic range (LDR). Some facilities rely on the output display characteristics (i.e., gamma) to as a primary tool in crafting the final image appearance. Others do not. It is challenging to provide standardized workflows and toolsets when current practice has so much variation.

Furthermore, the cost/benefit of adapting new color management techniques is often stacked against change. When something goes wrong in a motion-picture color pipeline, it can have potentially large financial consequences if work needs to be re-done. Furthermore, while color processing decisions are often made early on during the lifetime of a production, the consequences (both positive and negative) may not be evident until many months down the line. This decoupling of cause and effect makes experimentation and innovation difficult, and all too often leads people to assert "We've always done it this way, it's not worth trying something new".

The flip-slide is that computer graphics techniques used in motion-picture production are rapidly changing, outgrowing many classic color management techniques. For example, the recent trend toward physically-based rendering, shading, and lighting are only utilized to their fullest extents when working with dynamic ranges plausible in the real world (HDR). We thus assert that going forward, it will become increasingly beneficial for computer graphics applications, and visual-effects and animation facilities, to consider the approaches outlined in this course.

Multiple Inputs & Outputs: In live-action visual effects, imagery is often acquired using a multitude of input capture devices (digital motion picture cameras, still cameras, etc) and it is often desired to seamlessly merge sources. On the output side, the final delivery often appears under different viewing environments: digital theatrical presentation, film theatrical presentation, as well as home theater. Each of these outputs has different color considerations. Furthermore, artists often work on desktop displays with 'office' viewing conditions, and yet must have high-fidelity preview of the final outputs.

Complex Software Ecosystem: Another challenge is that the majority of visual effects and animation productions use *many* software tools: image viewers, texture / matte painting, composting apps, lighting tools, media generation, etc). Although it is imperative that artists work in a color managed pipeline across multiple applications, color support is quite varied between software vendors. Ideally, all software tools that interchange images, perform color conversions, or display images should be color managed in a consistent manner. The issue of interchange takes on an even more complex angle when you consider that multiple facilities often share image assets on a single film. Color management practices that encourage high-fidelity interchange are sorely needed.

Robust Imagery: Visual effects and animation are not the end of the line in terms of image processing. Digital Intermediate (DI) is a powerful tool for crafting the final appearance of a motion-picture (even for animated features), and may substantially impact the appearance of the final film. It is therefore a necessity to create computer graphics which are robust to such workflows, and maintain fidelity even under drastic color corrections. If digital intermediate is not considered during production, it is very likely that late stage color corrections will reveal latent problems in the computer-generated imagery. The eventual application of compression for delivery is also a consideration.

Future-Proof: Future improvements to display technology (such as wider dynamic range) are on the near horizon. For large productions, it is very prudent to take all the steps possible to future-proof the computer graphics material, such that you are only a 'remaster away' from taking advantage of the new technology.

2. Color Science

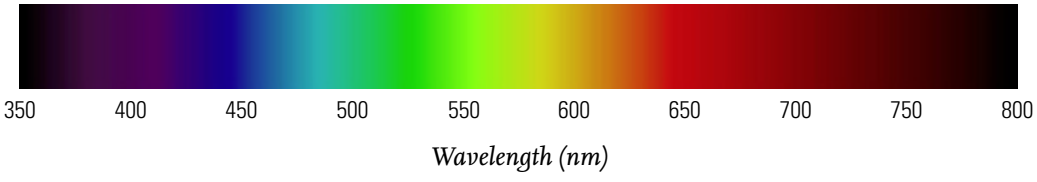
While a detailed overview of colorimetry is beyond the scope of these course notes, there are many textbooks which introduce color science in detail:

- *Color Science* [Wyszecki and Stiles, 1982] is the canonical bible for color scientists.
- *Measuring Color* [Hunt, 1998] is a compact overview of color measurement and color perception.
- *Color Imaging: Fundamentals and Application* [Reinhard, et al. 2008] presents a ground-up view of color fundamentals, and also touches upon applications such as camera and display technology.

A Brief Introduction to Color Science

Color science blends physical measurement along with characterizations of the human visual system. The fundamentals of **colorimetry** (the measurement and characterization of color) provide an important conceptual framework on which color management is built. Without color science, it would not be possible to characterize displays, characterize cameras, or have an understanding of the imaging fundamentals that permeate the rest of computer graphics. While it is possible to immediately jump into color pipeline implementations, having a rudimentary understanding of concepts such as spectral measurement, XYZ, and color appearance provide a richer understanding of why particular approaches are successful. Furthermore, being familiar with the vocabulary of color science is critical for discussing color concepts with precision.

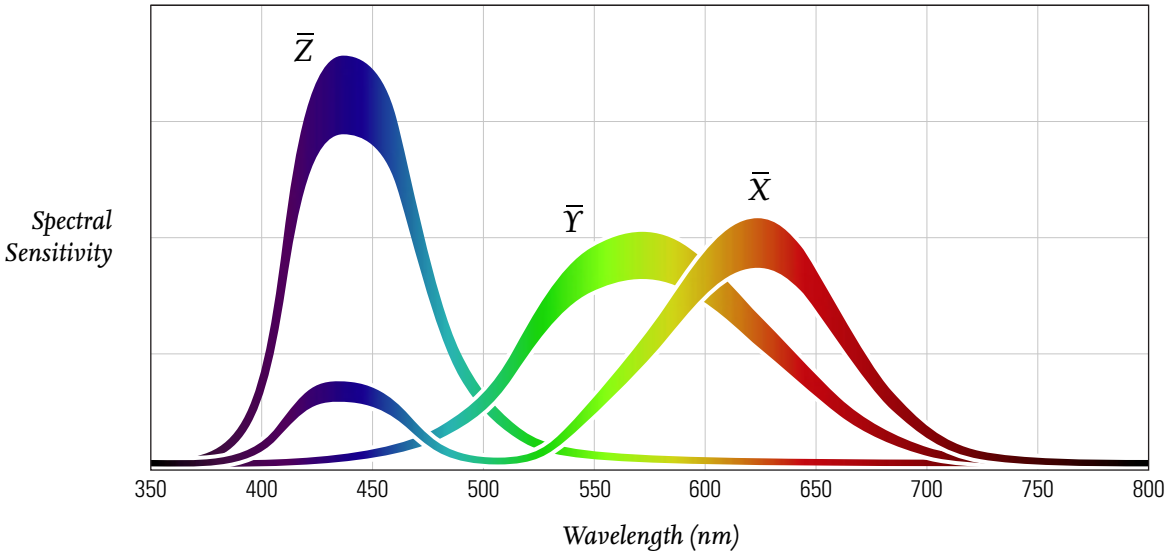
A study of color science begins with the spectrum. One measures light energy as a function of wavelength. The human visual system is most sensitive to wavelengths from 380-730 nm. Light towards the middle of this range (yellow-green) is perceived as being most luminous. At the extremes, light emitted above 730 nm (infrared) or below 380 nm (ultraviolet) appears indistinguishable from black, no matter how intense.



The electromagnetic spectrum from approximately 380-730 nm is visible to human[2] observers.

[2] Other animals can perceive light outside of this range. Bees, for example, can see into the ultraviolet.

The human visual system, under normal conditions, is trichromatic[3]. Thus, color can be fully specified as a function of three variables. Through a series of perceptual experiments, the color community has derived three curves, the **CIE 1931 color matching functions**, which allow for the conversion of spectral energy into a measure of color. Two spectra which integrate to matching XYZ values will appear identical to observers, under identical viewing conditions. Such spectra are known as *metamers*. The specific shape of these curves is constrained[4]; they are based upon the results of color matching experiments.



The CIE 1931 Color Matching Functions convert spectral energy distributions into a measure of color, XYZ. XYZ predicts if two spectral distributions appear identical to a human observer.

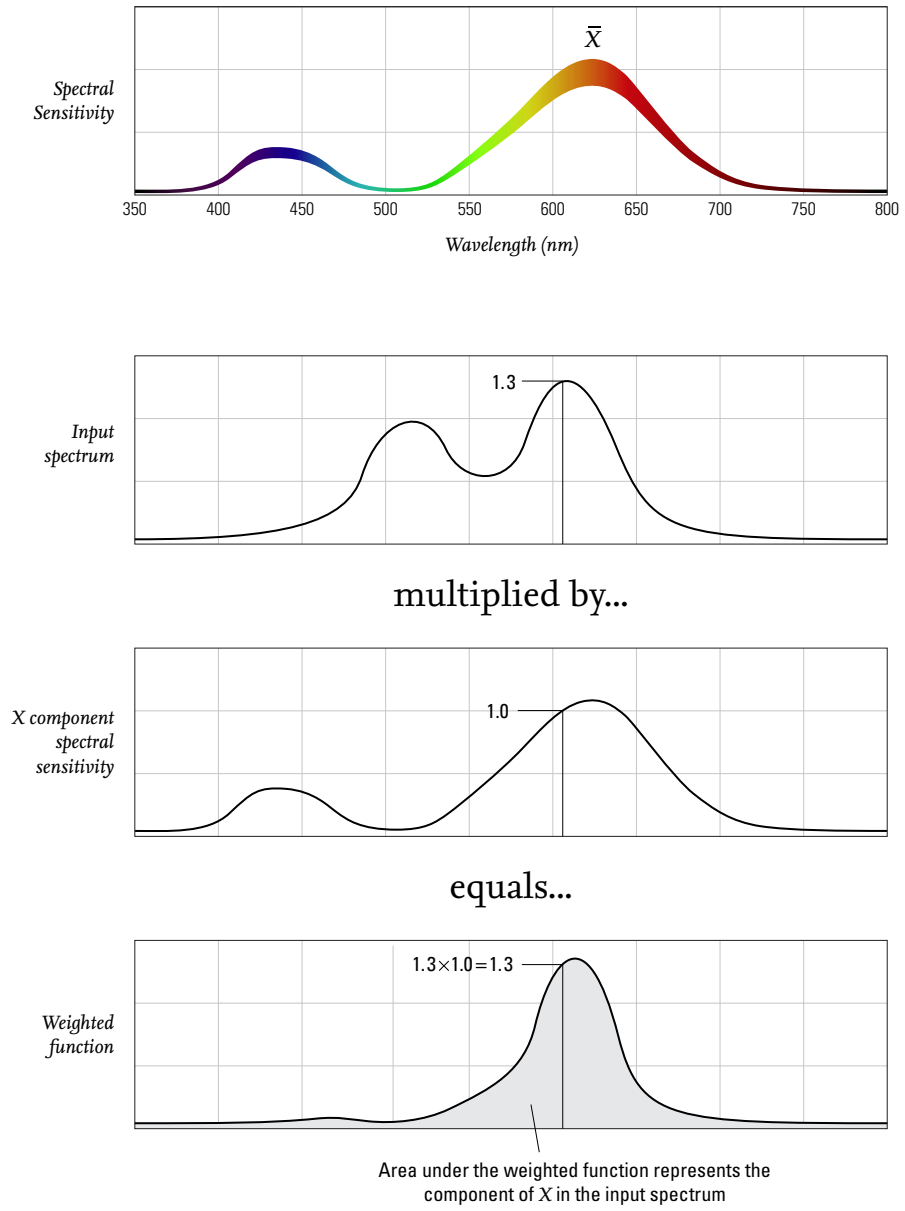
When you integrate a spectral power distribution with the CIE 1931 curves, the output is referred to as **CIE XYZ**, with the individual components being labelled **X**, **Y**, and **Z** (the capitalization is important). The **Y** component has special meaning in colorimetry, and is known as the **photopic luminance** function. Luminance is an overall scalar measure of light energy, proportionally weighted to the response of human color vision. The units for XYZ are candelas per meter squared (cd/m2), and are sometimes called "nits" in the video community. The motion-picture sometimes uses an alternative unit of luminance, "foot-lamberts", where 1 fL equals 3.426 cd/m2. An convenient trick to remember this conversion is that 14.0 fL almost exactly to 48.0 cd/m2, which coincidentally also happens to be the recommended target luminance for projected theatrical white.

---

[3] For the purposes of this document we assume color normal vision (non color-blind) and photopic light levels (cone vision).

[4] These the curves serve as the "basis functions" for characterizing a human's color vision response to light; thus all linear combinations of these color matching functions are valid measures of color. This particular basis was chosen by locking down Y to photopic luminance, and then picking an X an Z representation to place the visible locus tightly within the +X, +Z octant.

Note that XYZ does NOT model color appearance. XYZ is not appropriate for predicting a spectral energy distribution's apparent hue, determine how "colorful" a sample is, or determine how to make two color spectra appear equivalent under different viewing conditions[5]. XYZ, in the absence of additional processing, is *only* sufficient for predicting if two spectral power distributions can be distinguished.



CIE XYZ is calculated by multiplying the energy in the input spectrum (top) by the appropriate color matching function (middle), and then summing the area under the curve (bottom). As the color matching functions are based upon the sensitivities human color vision, the spectral energy during integration is zero-valued outside the visible spectrum.

---

[5] Color appearance models, far more complex than simple integration curves, model eccentricities of the human visual system and be used to creating matching color perceptions under differing conditions. One of the most popular color appearance model is iCAM. See [Fairchild, 98] for details.

Spectroradiometers measure spectral power distributions, from which CIE XYZ is computed. By measuring the spectral energy, such devices accurately measure colorimetry even on colors with widely different spectral characteristic. Spectroradiometers can also be pointed directly at real scenes, to act as high fidelity light-meters. Unlike normal cameras, spectroradiometers typically only measure the color at a single 'pixel', which has a comparatively large visual angle (multiple degrees for the color sample is common). Internally, spectroradiometers record the energy per-wavelength of light (often 2, 5, or 10 nm increments), and the integrate the spectral measurements with the color matching functions to display the XYZ or Yxy tristimulus value. The exposure times of spectroradiometers are such that color can still be accurately measured over a very wide range of luminance levels, in addition to light output with high frequency temporal flicker (such as DLP digital projector). Spectroradiometers are thus incredibly useful in high-fidelity device characterization and calibration.

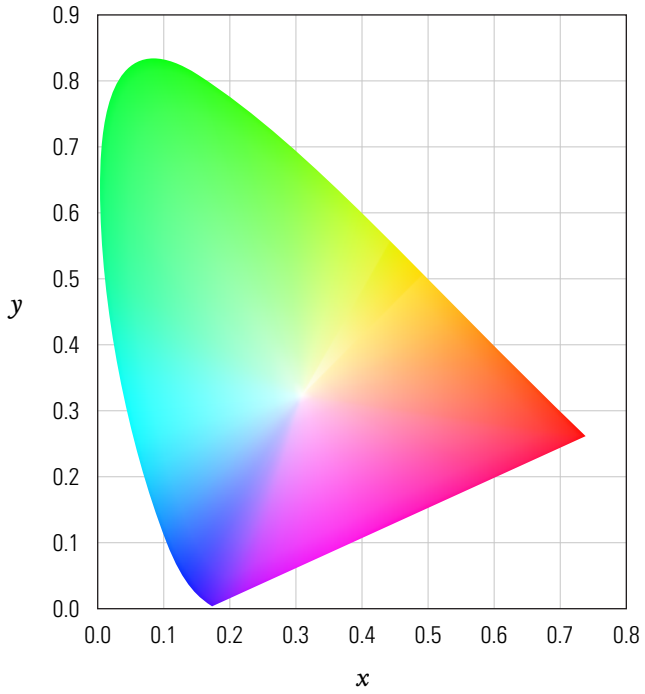
Spectroradiometers (left) accurately measure the visible spectrum and can also output an integrated CIE XYZ. Alternative approached to measuring color (such as the monitor puck on the right) are far more cost-effective but do not record the full color spectra. Such devices therefore are only color accurate when "turned" to a specific class of display technology. Images courtesy Photo Research, Inc., and EIZO.

It is often convenient to separate color representations into luminance and chroma components, such that colors can be compared and measured independent of intensity. The most common technique for doing so is to to normalize "Cap X, Y, Z" by the sum (X+Y+Z) and then to represent color as (x, y, Y). Note the capitalization.

$$x = \frac{X}{(X+Y+Z)} \qquad y = \frac{Y}{(X+Y+Z)}$$

The chromaticity coordinates (x,y) define color independent of luminance. It is very common to plot these values, particularly when referring to display device gamuts.

"Little x, little y" *(x,y)* is referred to as the **chromaticity coordinates**, and is used to plot color independent of luminance. When one converts *all possible* spectra into *x,y,Y* space and plots *x,y* they fall into a horse-shoe shaped region on the chromaticity chart. The edge of the horseshoe is called the visible locus, and corresponds to the most saturated color spectra that can be created. In this chart, luminance (Y) is plotted coming out of the page, orthogonal to *x,y*. [brucelindbloom] is a wonderful online resource for additional color conversion equations.



All possible light spectra, when plotted as xy chromaticity coordinates, fill a "horseshoe" shaped region. The region inside the horseshoe represents all possible colors; the region outside the horseshoe does not correspond to physically-possible color perceptions. (Such non-physically plausible coordinates are often useful for mathematical encoding purposes, but they are not realizable in ANY display system.)

Additive display systems such as television create colors by blending 3 colors of light. Red, green, and blue are often used as these allow for much of the visible gamut to be reproduced. The gamut of colors that can be reproduced is the triangle enclosed by the primaries. Note that because the outer boundary of horseshoe is curved, there is no possible choice of three colors which encloses the full visible gamut. (Remember, you cannot build real displays with primaries outside of the horseshoe). This is why some recent television manufactures have begun to experiment with adding a fourth color primary.

Although it is not immediately apparent, the chromaticity chart has very poor perceptual uniformity. The distances between colors in *chromaticity space* do not directly relate to their apparent perceptual differences. Two colors nearby in *xy* may be perceived as appearing very dissimilar, while colors colors far apart may be perceived as being indistinguishable. See "MacAdam ellipses" in traditional color textbooks for precise graphics representations of this non-uniformity. Roughly speaking, saturated green colors in *xy* space are over-accentuated relative to their perceptual similarity. The perceptual nonuniform of XYZ (and *xy*) is not surprising given that XYZ does not model color appearance.

Finally, as color is inherently a three dimensional quantity, any discussions which make use of two dimensional charts tends to be misleading. For a graphical exploration of XYZ using 3-D, see *Visualizing the XYZ Color Space* [Selan 2005].

2.1. Color Encoding, Color Space, and Image States

Thus far we have discussed the measurement of color, but have not tied these measurements back to seemingly familiar computer graphics concepts such as RGB. So what is RGB?

RGB is a color encoding where red, green, and blue "primaries" are additively mixed to reproduce a range (gamut) of colors. The specific color appearance of pure red, green, and blue is tied to the chosen display device; often identified using chromaticity coordinates. The code values sent to a display device often correspond non-linearly to the emitted light output, as measured in XYZ. This non-linearity was originally a consequence of display technology, but today serves a continued purpose in increasing the coding efficiency of the transmitted images.

All RGB colors have units. Sometimes an RGB pixel's units are explicit, such as measuring the emitted light from a display using a spectroradiometer and being able to reference pixel values in XYZ cd/m2. However, sometimes the units are only indirectly related to the real-world, such as providing a mathematical conversion to measurable quantities. For example, having code values represent either the logarithm or exponent of RGB is common. This definition of how measurable color quantities relate to image RGB code values is referred to as the **color encoding**, or more commonly in the motion-picture computer graphics community, **color space**[6].

In the case of display technology, common color encodings (relations of code value to measurable XYZ performance) include sRGB and DCI-P3. But considering image display only provides part of the color encoding story. In addition to relating RGB values to display measurements, one can also relate RGB values to the performance characteristics of an *input device* (i.e., a camera). Input colorimetry can be measured in real-world units as well. It is not difficult to measure an input spectra with the spectrophotometer in XYZ, and then compare this to the RGB values output from the camera. This process, called camera characterization, will be discussed further in section 2.3.

It is a meaning abstraction to categorize color spaces by the "direction" of this relationship to real-world quantities, which we refer to as **image state**. Color spaces which are defined in relation to display characteristic are called **display-referred**, while color spaces which are defined in relation to input devices (scenes) are **scene-referred**. While there are other flavors of images states (intermediate-referred, focal-plane referred) display-referred and scene-referred colorimetry are most common used in motion-picture color management, and will be the focus of the next sections.

For further information on image state and color encodings, the Ed Giorgianni publications provide significantly greater detail. [Giorgianni 98] [Giorgianni 05]

[6] The color science community looks down upon the use of *color space* to denote the RGG pixel encoding; color space is preferred to refer to the broader *class* of color encoding, examples of which are RGB, CMY, HSV, L*a*b*, etc.). However, the mis-use of the *color space* is so ubiquitous that that we will adhere to industry convention.

2.2. Display-Referred Imagery

Display-referred imagery is defined colorimetrically with regards to the appearance of the image as presented on a display. The display may be either an idealized display standard, or a physical display that exists in the real-world. When "RGB" is used casually without qualification of colorimetry (such as in web standards), it is most likely assuming display-referred imagery. The primary advantage of working with display-referred imagery is that if the user's display matches the reference display definition, one can accurately display the raw pixel values on the screen without any additional color conversions. I.e., if a user creates images by directly manipulating an image raster, they are working in display referred spaces. This simplicity in color management makes display-referred color processing engines a popular choice in desktop publishing applications.

Linearized Display-Referred Imagery

As mentioned previously, the RGB code values that are sent to the display are not directly proportional to the light being output from the display. However, there are many cases in computer graphics where working with pixels in a color space proportional to light output is preferable. For example, in anti-aliased filtering one important requirement is that pixel energy should be preserved. What this means is that the total light energy of the image emitted from the display - both before and after filtering - should be identical. If this were not true, then resizing an image would change the apparent brightness, which is not ideal. Such artifacts become even more apparent when applied to moving imagery (motion pictures) where slow transitions between light and dark regions can "crawl" when energy preservation is ignored.

To linearize display-referred imagery, one must come up with a model of the display technology which predicts how much light, as a function of code value, is being emitted. When modeled by a single number, this is referred to as **Gamma.** Gamma value of 2.2 are a very common approximation to the values seen in real-world display technologies.

$$RGB_{linear} = RGB_{device}^{\gamma}$$

The exponent value used to relate RGB code values to linear light is called gamma. Gamma is generally defined as the inverse of the exponent, but it is useful to be explicit as there the language people use to describe gamma is usually ambiguous. One easy way to remember is that middle gray display-referred values, when linearized, become smaller. So a 128 RGB code value (out of 255), when linearized with a 2.2 gamma, is approximately ~ 0.218.

One of the additional benefits of using a gamma function is that it offers a more perceptually uniform encoding space, which better utilizes the limited number of bits available in the display link. Thus, even on devices which were based upon inherently linear technology (such as the DLP digital projectors), it remains useful to artificially emulate a gamma value. See Charles Poynton's Gamma FAQ [Poynton 12] for a thorough discussion of gamma.

sRGB

Due to differences in inherent display technologies, there is substantial variation in the appearance of RGB when the same code values are sent to multiple displays, making the unambiguous distribution of RGB imagery difficult. As a solution, a standard "idealized" display has been defined, called **sRGB**, which real displays attempt to reproduce. The intent of **sRGB** ("Standard RGB") is to define the color characteristics of a standardized "average" RGB monitor, such that imagery on one monitor matches the appearance of a different monitor. When a monitor is properly calibrated to sRGB, the output is reproducible and well defined. Older display technology such as cathode-ray tube (CRT) technology naturally approach the sRGB specification (which is how it was defined), However, even modern technologies such as LCD and OLED, which have very different inherent image responses, typically provide an option to emulate the sRGB response to maintain compatibility with existing imagery.

Linearizing RGB_{device} (sRGB)

$$\text{If } RGB_{device} \leq 0.03928 \text{ then } RGB_{linear} = \frac{RGB_{device}}{12.92}$$

$$\text{else } RGB_{linear} = \left(\frac{0.055 + RGB_{device}}{1.055} \right)^{2.4}$$

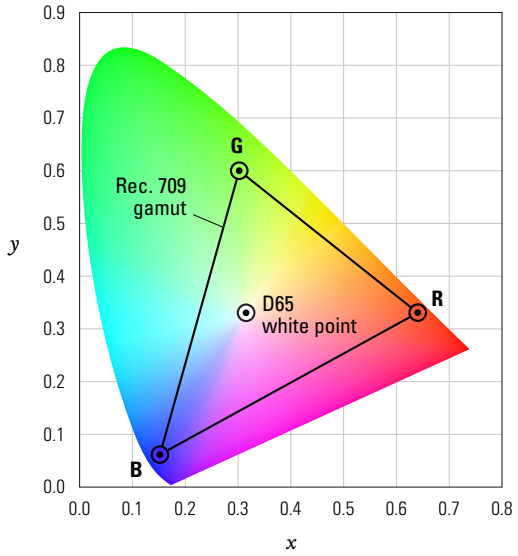
RGB_{linear} to XYZ conversion (D65 reference white point)

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124564 & 0.3575761 & 0.1804375 \\ 0.2126729 & 0.7151522 & 0.0721750 \\ 0.0193339 & 0.1191920 & 0.9503041 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

These steps allow one to predict emitted light in CIE XYZ, as emitted from a calibrated sRGB. First, device RGB is converted to linearized RGB. Next, the linear RGB is converted to XYZ using the conversion matrix. Note that even though the sRGB transfer function uses a 2.4 exponent, due to the inclusion of the scaling and offer factor this transfer function approximates a 2.2 gamma over the range of [0,1].

Because display-referred imagery is referenced to the light emitted from a display, it's possible to convert RGB values to output scene CIE XYZs. For example, it is sensical the specify a display's white point and black point. The white point would be the real-world XYZ for the maximum RGB value (on a 8-bit display, 255, 255, 255). The black point is the XYZ for the minimum RGB value (0,0,0).

As the dynamic rage of display-referred image is well defined. (with a min code value and max code value), it is common to use integer codings to represent display RGB. 8 bits is common, and lets you represent the range of [0, 255]. Note that on high quality displays, under ideal conditions, 8 bits is not sufficient to prevent the appearance of banding (this artifact is particularly noticeable on grayscale imagery with smooth gradient). For this reason professional displays (such as medical displays, or those used in professional color applications) often display images with greater than 8 bits of precision (10 or 12 bits are not uncommon).



	x	*y*
White point	0.3127	0.3290
R	0.64	0.33
G	0.30	0.60
B	0.15	0.06

sRGB relies on the "Rec709" primaries and white point, and thus can re-create any of the color in the above triangle (gamut).

Display referred imagery is also the realm of ICC profiles and traditional appearance modeling techniques. If you have two different displays, with different color reproductions, you can use ICC to convert between them while preserving image appearance. You can also use ICC for display calibration, where libraries will compute the color transform necessary to have your display emulate an 'ideal calibration'.

Another display-referred image specification is **DCI-P3**. This color space is common in digital cinema production, and is well suited to theatrical presentation. Where as the sRGB specification uses a color encoding suited for the desktop environment, the DCI-P3 specification uses a color encoding suited for theatrical luminance levels. Another example of display-referred imagery in motion-picture production **X'Y'Z'** (called "x-prime, y-prime, z-prime") This is the color space for the final imagery actually sent to the theater in the digital cinema package (DCP), and is a gamma encoded version of XYZ output colorimetry. See appendix 4.4 for further details on DCI-P3 and X'Y'Z.

Limitations of Display-Referred Imagery

Display referred imagery has dynamic ranges which are inherently tied to displays. Thus, even though the real world can have enormously luminous intensities, when working in a display-referred space values above the display-white are essentially meaningless. This mismatch between the dynamic ranges of the real world and the dynamic range of display-technology makes working in display-referred color spaces (even linear ones) ill suited for physically-based rendering, shading, or compositing. Yet even though working with display referred imagery has limitations, it forms a critical component of pipelines. Even in full motion-picture color pipelines which work in higher dynamic range color spaces, there is always remains a portion of the pipe where display referred, and even linearized display referred imagery, is critical.

2.3. Scene-Referred Imagery

The second major image state in motion pictures imaging pipelines is that of scene-referred imagery, where the code values are proportional to measurements of real-world scenes. The typical way to create scene-referred image is either through the characterization of a camera system, or though synthetic means. As there is no absolute maximum white point, pixel values can be arbitrarily large, within the constraints of the capture device. Scene-referred image pipelines are inherently linear, as pixel values are proportional to photons in the real-world by definition.

As the real world has a very high-dynamic range, it's often useful to talk about light in terms of 'stops', or doublings of light. You can compute the number of 'stop' and exposure is by taking the logarithm, base-2.

$$\text{relative exposure} = log_2\left(\frac{\text{luminance}}{100\ cd/m^2}\right)$$

Relative exposure in stops, is the log, base-2, relative to some reference exposure level. In this example, the choice of 100 cd/m2 has been selected arbitrarily. Any normalization factor would suffice.

Stops	Multiplication Factor
0	1
+0.5	1.414
+1	2
+2	4
+3	8
+8	256

Stops	Multiplication Factor
0	1
-0.5	0.707
-1	0.5
-2	0.25
-3	0.125
-8	0.0039

Talking about scene-referred exposure values is most often done in units of stops, as the range between values is so large. For example, it's difficult to get an intuition for what it means to change the emitted light by a factor of 0.0039. However, it's very sensible when that same ratio is expressed as "-8 stops".

In a real scene, if you measure luminance values with a tool such as a spectroradiometer, one can observe a very wide range of values in a single scene. Pointed directly at emissive light sources, very large values such as 10,000 cd/m2 are possible (and if the sun is directly visible, specular reflections may be another +6 stops over that)! Even in very brightly lit scenes, dark values are created either through material properties, through scene-occlusions, or a combination of the two. Dark materials reflect a small fraction of incoming light, often in the 3-5% range. As a single number, this overall reflectivity is called 'albedo'. Considering illumination and scene-geometry, other objects can cast shadows and otherwise occlude illumination being transported around the scene. Thus in real scenarios, It's often possible with complex occlusions and a wide variety of material properties to have dark values 1,000-10,000 times darker then the brightest emissive light sources.

Luminance (cd/m^2)	Relative exposure	Object
1,600,000,000	23.9	Sun
23,000,000	17.8	Incandescent lamp (filament)
10,000	6.6	White paper in the sun
8,500	6.4	HDR monitor
5,000	5.6	Blue sky
100	0	White paper in typical office lighting (500 lux)
50 to 500	-1.0 to 2.3	Preferred values for indoor lighting
80	-0.3	Office desktop sRGB display
48	-1.1	Digital Cinema Projector
1	-6.6	White paper in candle light (5 lux)
0.01	-13.3	Night vision (rods in retina)

All values are in the case of direct observation.

Luminance values we encounter on a daily basis span an enormous dynamic range (greater than a million to one).

It's important to observe that in the real-world, there is not any particular "maximum luminance" where light can't get any brighter. This differs form display-referred imagery, where it's easy to define the maximum light that a display system can possibly emit. On the dark side (black point), where one could always remove a bit more light, getting closer and closer to zero photons. In real scenes its very hard to create a situation where there is NO light - more typical is that you just have very small positive value.

When we bring HDR imagery into the computer its useful to normalize the scene exposure. Even though an outdoor HDR scene may be at an absolute level 1000 times more luminous than the equivalent indoor scene, it is very useful to have them at equivalent overall intensities, yet still preserving the relative intra-frame luminance levels. As the absolute maximum luminance is quite variable (even frame to frame), scene-referred imagery tends to be normalized with respect to an average *gray level*. Convention in the industry is to pin middle gray at 0.18 (representing the reflectivity of an 18% gray card). Observe that even when we "normalize" the exposure of scene-linear imagery, it is expected that many portions of the scene will continue to have luminance values >> 1.0. Furthermore, nothing "magical" happens with bright code values > 1.0. There is generally a continuum of luminance values in the scene, and it generally is not feasible to assert that above a particular value denotes specularity.

Integer representation are not appropriate for storing scene-referred imagery due to the distribution of luminance-levels seen in real-world values (even when gray normalized). If you look at the actual distribution of luminance levels in the scene, one sees that greater precision is required around dark values, and that less precision is required in highlights. Consider doing a test to find the smallest 'just noticeable difference' that is suitable for recording shadow detail in linear light, without introducing visible banding. Thus if one uses integers to get adequate precision in the shadows, when this same light "JND" is used on very bright pixels, there will be far too many of them and bits will be wasted.

Conversely, if one tailors a code value step size to provide reasonable luminance resolution on the highlights, then the shadows will not have sufficient detail.

Floating-point representations gracefully address the precision issues associated with encoding scene-linear pixels. Float representations are built from two components: the individual storage of an exponent, and a linear scaling (the mantissa). This hybrid log / linear coding scheme allows for an almost ideal representation of scene-referred imagery, providing both adequate precision in the shadows along with the highlights. In modern visual effects an color pipelines, OpenEXR is most commonly used to store floating-point imagery and helped to popularize the 16-bit half float format. which now has native support in a wide variety of systems including GPUs. See Appendix 4.3 for more information on OpenEXR. It's important to note that while EXR popularized high dynamic range file representations in the motion-picture industry, Greg Ward made major contributions to HDR storage many years earlier with the RGBE HDR image format.

Characterizing Cameras

Creating scene referred imagery is usually tackled by setting up a camera under known test conditions, and then determining how to relate the output camera RGB code values to linear light in the original scene. When going this route, its usually best to start with a camera RGB image as close to 'camera raw' as possible, as these encodings preserve the greatest fidelity.

Even better than characterizing a camera yourself is when the manufacturer does this for you by providing a set of curves or lookup tables to "unbake" the input transformation. In the situation where you *do* need to characterize a new camera (or validate that the linearization being used is appropriate) the general approach is to is to setup a scene with a stable light-source, and then to do an 'exposure sweep' with the camera in known increments. By changing the exposure in known "stop" increments (a stop is a doubling of light in the scene) you can relate exposure values in scene-linear to code values in the output imagery.

Camera characterizations are often approached as channel independent mappings, using 1-D transforms. However, sometimes the camera's response is different per-channel. The two common approaches to handling this are to either perform a weighted average of the channels and then use that as the basis for converting to scene-linear, or to do a different 1D conversion for each channel. Common practice is that for systems where the channels have approximately equal response curves (most digital cameras fall into this category) to use a single mapping for all three channels. Only when capture systems have very different responses (such as film negatives) are separate curves per channel appropriate.

Channel independent mappings to scene-linear are simple, but not always sufficient. Consider two cameras from different manufacturers imaging the same scene. Even when we do the proper 1D conversion to scene-linear, there are still likely to be differences in the residual color appearance due to to differing color filter technologies being utilized. These differences are often accounted for by imaging a series of color patches, and then coming up with a 3x3 matrix transform that minimizes the differences between devices. One common reference used for this purpose is the macbeth chart, which has standardized patches with known reflectances. The chart reflects incident illumination, so its appearance is a function of the wide-spectrum performance of the lights. I.e., the Macbeth chart appears slightly different under tungsten vs daylight illumination.

The Macbeth color checker is commonly used to validate camera characterizations; the patches have known reflectance values. Image courtesy of X-Rite.

While a full discussion of input characterization is out of the scope of these notes, camera linearizations have a bit of variation when when used at very darkest portions of the camera capture range. One major challenge is to determine if the lowest code values out of the camera represent 'no light', in which case the average black level could be a true 0.0 in scene linear, or if instead the lowest code values from the camera correspond to a small but positive quantity of light. This issue becomes more complex in the context of preserving sensor noise and grain in the very darkest regions. If you consider imaging true black in a camera system, which when mapped to linear must averages 0.0, it implies that some of the code values of noise will be small, yet positive, and other parts of the noise will be small, and negative. While not physically plausible, preserving these negative values is critical to maintaining an accurate black level, on average. There are similar considerations even when black is mapped to small, positive quantities.

Displaying Scene-Referred Imagery

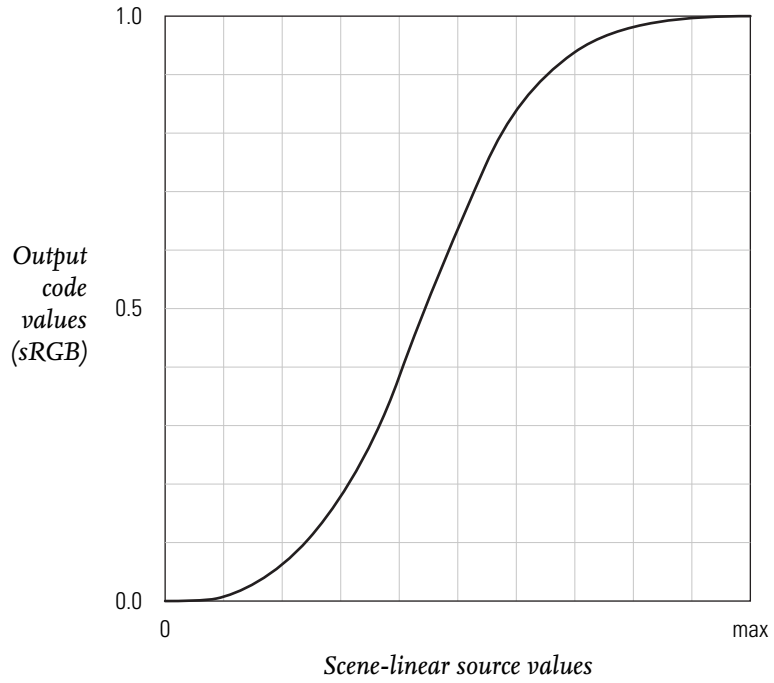
The primary complexity of working with scene-referred imagery is solving the issue of reproduction on displays. While HDR scene-referred imagery is natural for processing, displays can only reproduce relatively low dynamic ranges (LDR) portion of the range. While it seems like a reasonable approach to directly map scene-referred linear directly to the display (at least for overlapping portions of the contrast range), this has terrible results in practice. The inability to directly map scene-linear to display-linear is actually quite a large conceptual hurdle to overcome, and it's not immediately obvious to most people this is so. See [Giorgianni 05], Color Management for Digital Cinema, for further justifications.

So how can you pleasingly reproduce high-dynamic range pixels on a low dynamic range reproduction? This process in the color community, is called tone rendering (not to be confused with what graphics people call rendering), and is an active area of research.

Many pleasing tonal renditions of high-dynamic range data use similarly shaped transforms. On first glance this may be surprising, but when one sits down and actually designs a tone rendering transform there are convergent processes at work corresponding to what looks "good". The general solution to this tone mapping problem is to intelligently compress the HDR into an smaller tonal range to fit the display's possible output space. Unfortunately, simply compressing it linearly results in a low-contrast appearance, so the compromise is to compress it more in the high and low ends.

To begin with, most tone rendering map a traditional scene gray exposure (0.18 by convention) to an appropriately 'middle' value on the output display. Under theatrical viewing conditions, mapping middle gray to ~10% of the maximum output luminance (in linearized display-referred space) yields

pleasing results. If one directly maps the remaining scene-linear image to the display (and clips), the resulting image will appear low contrast. Thus, a reconstruction slope which is greater in contrast than 1 to 1 bumps the contrast around the middle gray and creates a pleasing appearance. But, with this increase in contrast the tops an bottoms are still clipped, so a rolloff is applied on both the high and low ends to allow for highlight and shadow detail to "rolled off" The final curve resembles an 'S' shape, with the center of the "S" centered on the midtones. The final transfer curve from scene-linear to output display is shockingly consistent between technologies, roughly matching both the end to end transform of both digital and film imaging pipelines.



An 'S' shaped curve is traditionally used to tone render scene referred HDR colorimetry into an image suitable for output on a low dynamic range display. The input axis is log, base 2, of scene-linear data, The output axis directly corresponds to code-values on a calibrated sRGB display.

If we characterize the film color process, from negative to print, we see that it almost exactly produces this 's-shaped' transfer curve, which is not surprising given the beautiful tonal reproductions film offers. In the traditional film imaging process, the negative stock captures wide dynamic range (well beyond the range of even modern digital camera), and the print stock imparts a very pleasing S tone mapping for reproduction on limited dynamic range devices. Broadly-speaking, film negatives encode an HDR scene-referred image, and the print embodies a display referred tone-mapping. For those interested in further details on the high-dynamic range imaging processes implicit in the film development process, see the Ansel Adams Photography Series [Adams 84].

It's worth noting that current tone mapping research often utilize spatially varying color correction operators. While that research is very promising and may impact motion picture production at some point in the future, most motion-picture color management approaches currently assume that each pixel is treated independently. However, much of the current 'look' of spatially varying tone-mapping

operators are currently achieved in alternate ways. For example, tone-mapping operators often accentuate high-frequency detail in shadow areas which would have otherwise been masked. In the cinematic world, this is addressed by directly crafting on-set lighting by the cinematographer, or as an explicit artistic correction during the digital intermediate (DI) process.

To summarize, do not directly map high-dynamic range scene-referred data to the display. A tone-rendering is required, and there is great historical precedence for using an 'S-Shaped' curve. HDR scene-acquisition accurate records a wide range of scene-luminance value, and pleasing tonal renditions preserve much of this range during reproduction even on low-dynamic range devices.

Please refer to Chapter 3.2 for a comparison of the visual differences between our recommended s-shaped tone mapping operator, versus a naive scene-linear to display-linear mapping.

Consequences of Scene-Referred Imagery

Working with the dynamic ranges typical of scene-referred imagery positively impacts almost every area of the computer graphics pipeline, particularly when the goal is physical realism. In rendering and shading, scene-referred imagery naturally allows for the use of physically-based shading models and global illumination. In compositing, physically-plausible dynamic ranges allow for realistic synthetic camera effects, particularly as related to filtering operations, defocus, motion-blur, and antialiasing. See Chapter 3 for further details of how a scene-linear workflow impacts specific portions of the color pipeline.

Unfortunately, scene-referred imagery also presents some interesting challenges as well. Filtering, antialiasing, and reconstruction kernels which make use of "negative lobed" filters (sinc, lanczos3, simon, etc.) are much more susceptible to ringing when applied to HDR imagery. While high-dynamic range isn't the root cause of the ringing, it certainly exacerbates the issue. Another pitfall of working with HDR data is that storage requirements are increased. In a low dynamic range workflow, 8-bit textures are sensible, while in common HDR workflows 16-bits of HDR is usually a minimum. There are further issues with "classic" common compositing tricks that rely on assumptions which do not hold in floating point. Chapter 3 will also discuss potential solutions to many of these issues.

A Plea for Precise Terminology

The computer graphics community - as a whole - is far too casual about using the word "linear" to reference both scene-referred and display-referred linear imagery. We *highly* encourage our attendees to set a positive example, and to always distinguish between these two image states even in casual conversation. To clarify...

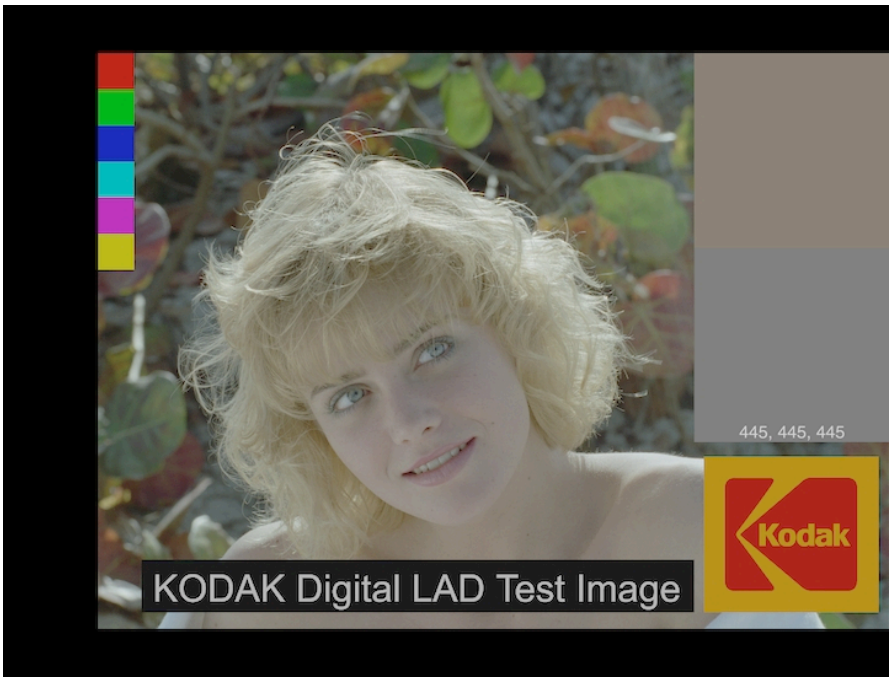
Are you referencing the linear light, as emitted by the display? Did you use the "g-word", "gamma"? Do odd things happen to your renders when values go above 1.0f? If so, please use the term **display-linear**.

Alternatively, are you referencing high-dynamic range or physically-plausible lighting models? Is your middle gray at 0.18? Are you talking about light in terms of "stops"? Does 1.0 have no particular consequence in your renders? If so, please use the term **scene-linear**, and by all means make sure you are using a view transform that goes beyond a simple gamma model. *Friends don't let friends view scene-linear imagery without an 's-shaped' view transform.*

2.4. Color Correction, Color Space, and "Log"

It is sometimes necessary to encode high-dynamic range, scene-referred color spaces with integer representations. For example, digital motion picture cameras often record live to 10-bit integer tape media (such as HDCAM SR). As noted earlier, if a camera manufacturer were to store a linear integer encoding of scene-referred imagery, the process would introduce a substantial amount of quantization. It turns out that using a logarithmic "log" integer encoding allows for most of the benefits of floating point representations, without actually requiring float-point storage media. In logarithmic encodings, successive code values in the log space map to multiplicative values in linear space. Put more intuitively, this is analogous to converting each pixel to a representation of the number of stops above or below middle gray, and then storing an integer representation of this quantity.

As "log" images represent a very large dynamic range in their coding space, most midtone pixels reside in the middle portion of the coding space. Thus, if you directly display a 'log' image on an sRGB monitor it appears low contrast.

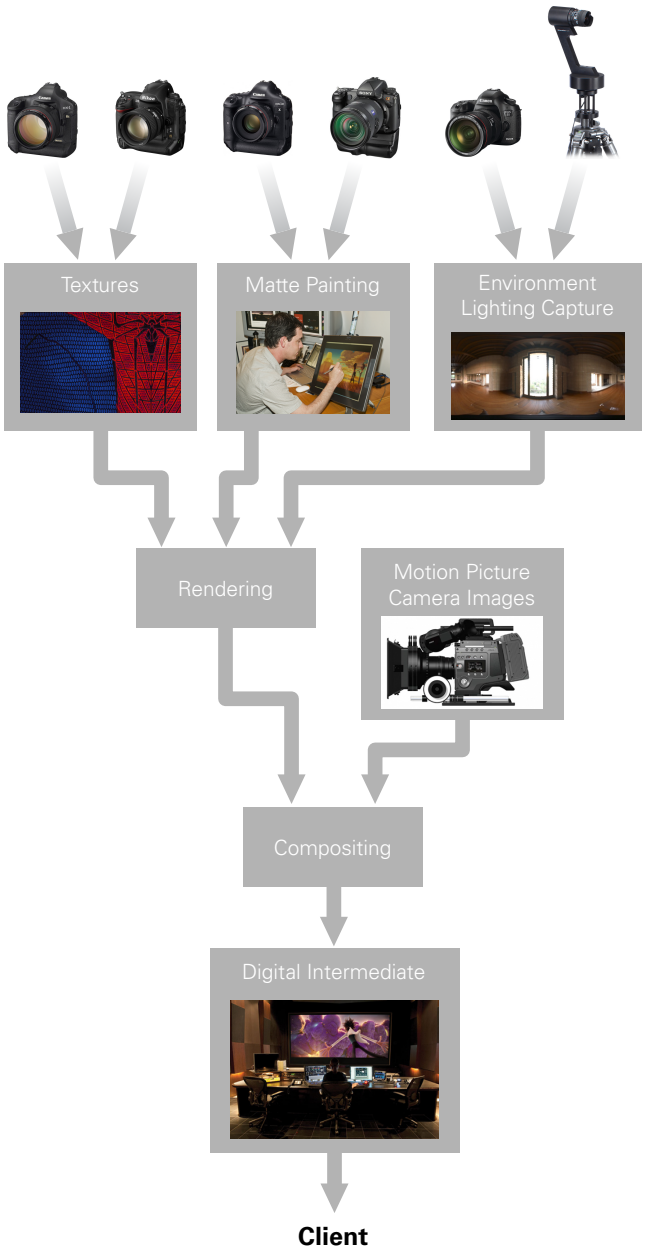


This image appears low contrast because it is using an integer logarithmic encoding to represent a very-high dynamic range of scene intensities. This image is colloquially known as "Marci", and is a common reference image in theatrical exhibition. This image originated from a scan of a film negative, courtesy Kodak Corporation. It can be downloaded from their website in DPX and CIN formats.

Not all 'log' spaces are equal. Most camera manufacturers customize a log encoding equation to optimize the usage of code values based on the camera's dynamic range, noise characteristics, and clipping behavior. Examples of different log spaces are Sony S-Log, REDLog (from the RED Cameras), Arri LogC, and the classic Cineon (used in film negative scanning). Care must be taken during image handling to determine which log space is associated with the imagery, and to use the proper linearization.

3. Motion-Picture Color Management

A **Color Pipeline** is the set of all color transformations used during a motion-picture production. In a simplified computer graphics pipeline (below), one must consider the processes such as texture painting, matte painting, on-set lighting capture, traditional motion-picture camera inputs, lighting, rendering, compositing, and digital intermediate (DI). In a properly color-managed pipeline, the color encodings are well defined for all images, at all stages of processing.



A well-defined computer-graphics color pipeline rigorously manages the color transformations for each stage of the process, in addition to the transforms required for accurate image preview.

We define the color pipeline by carefully tracking the color encodings (color space) for all image inputs, image outputs, and intermediate representations. We also rigorously define the transforms used to convert between color spaces.

The Visual Effects Color Pipeline

Traditional visual effects color pipelines are based around the core principle of doing no harm to the plate. Plate photography is brought in whatever color space is delivered. (camera log, commonly), and then a converted to scene-linear using an invertible transform. Often, a pure 1d camera to linear transform is used, though sometimes the linearization is augmented with a matrix transformation. For visualization, a 3D-LUT is used which will emulates the output processing, which may either be a print film emulation or a similar s-shaped transformation. But during the visual effects process, the display transform is never baked into the imagery.

3.1. Digital Intermediate, Mastering, and Delivery

Digital Intermediate (DI) is the process where the entire motion-picture is loaded into a dedicated hardware device, for the purpose of color-correcting in an environment that exactly mirrors the final exhibition (i.e., in a theater). Viewed in this final environment, DI is where per-shot color corrections are added, and the visual look of the film is finalized. DI is sometime also referred to as "color timing", or often simply, "timing". The final step of baking in view transforms (specific to an output device) and final color corrections is known as "mastering".

Getting the viewing transforms locked down in the DI process is the most important part of a cinematic color pipelines, as it pins down every other part of the process. The remainder of the pipeline essentially relies on visualizations that are created by working backwards from this step.

Digital intermediate is done in a viewing environment that exactly mirrors the final exhibition. If you are mastering for digital cinema, the DI must be done in an equivalent theatrical setting with matched digital projection. If you are mastering for home theater, you should be mastering in an equivalent 'idealized' home environment. (There are well-defined mastering specifications for both situations). If you are mastering for film release, you typically use a theatrical environment with a digital projector, with a film emulation 3DLUT made in conjunction with the print house to confirm a visual match. Digital intermediate (and mastering) typically have very tight tolerances on calibration, as the decisions made in this process are the last time image color appearance is tweaked.

Examples of popular commercial DI systems include Lustre, DaVinci, Baselight, NuCoda, etc. Companies that historically have historically done DI include Deluxe and Technicolor.

Color Correction and Working Space

There are two main approaches to handing color in digital intermediate. The first "video-centric" approach is where display-referred imagery is loaded into the DI; no viewing transforms is necessary. The display referred imagery is directly manipulated (sort of like a motion-picture version of Photoshop). In the second, "filmic" approach, scene-referred imagery is loaded into the machine and a viewing transform (3DLUT) is required to create the final the color appearance. The color correction manipulates the underlying scene-referred representation, but all color judgements are made previewing through the display transform.

The advantage of the "video" approach is one of simplicity. When color correcting pre-rendered imagery, a relative modest amount of color correction can be applied without introducing artifacts in the imagery. The downside to this approach is that much of the detail in the original imagery is lost when pre-baking in the view transform (this baking in some cases may have even happened inside the camera). For example, if a shot was originally overexposed, a sensible color correction is to darken the image. However, it is likely that in the overexposed image that large portions of the image was clipped to a constant maximum value, and no possible correction can bring back this lost detail.

In the alternative mode, working with a scene-referred imagery, a high-fidelity representation of the full dynamic range from the original camera is loaded. Sometimes this is the equivalent of camera raw, but most commonly it is a camera-specific integer 'log' encoding. In the future, DI systems that load floating-point scene-referred imagery will become increasingly common, but for now log DPX files are the norm. For viewing, an S-Shaped tone curve is used for preview, and is crafted to precisely emulate the appearance of the target output. For example, on a show that is going to be printed to film, a 3D LUT which emulates the specific print stock + development process is appropriate. If this film will also be distributed digitally, this specific print film emulation will eventually be baked into the imagery prior to distribution to the theaters. Of course, if the production will only be distributed digitally there is far more latitude in selecting a viewing transform. In both cases, the appearance of the imagery is modified by color correcting the image in the native log (or scene-linear) color space, and making visual decisions based on the appearance post viewing transform.

Going back to our "over-exposed" example, remember that more of the dynamic range from the original camera capture is preserved when using a scene-referred approach. Thus, when we change the exposure on our log data we may expose new details which had not previously been visible (no clipping has occurred). In terms of classic film production processes, as more data was captured on the negative, we have a wide latitude of possible print exposures over which a high print fidelity is maintained. This allows color correction in "log" DI to be very high fidelity; most modest corrections do not drive the image to flat black or flat white images.

DPX is commonly used as a log delivery format to Digital Intermediate, and both 10-bit and 16-bit versions are common-place. For grainy material, 10-bits is often sufficient, but 16-bit is becoming increasingly common for computer generated imagery with low noise. In the case of passing true scene-linear renders directly to DI, 16-bit floating point (EXR) is preferable. 16-bit integer linear representations are not ever recommended.

The original log film plate (top-left) is loaded into the color corrector. When viewed with a film emulation table (top-right), the appearance is predictive of the final output with a default exposure. If we were to lower the exposure using an additive offset in log-space (lower-left), new details in the flame highlights are revealed. When a similar correction is applied to a color-corrector working in a display-referred space (lower-right), this detail is lost. Imagery from Spider-Man, © 2002 Columbia Pictures. All rights reserved.

In scene-linear, a gain operation is typically used to change the color balance and scene exposure. In log space, this roughly corresponds to offsets. If a mathematically exact log is used, they are in fact identical, though most manufacturers tweak the log encodings as previously mentioned. Log offset color corrections are most common in motion-picture industry color correction, and are often referred to as the "primary grade" or "one-lights". Theatrical "fades to black" have a very particular appearance to them, which is a direct consequence of the fade applying in log space, as viewed through the traditional 's-shaped' film emulation.

Artistically, the DI process can be segmented into a per-shot correction that neutralizes shot to shot color variation, and then a secondary process that craft the overall look of the film. It is common for the DI house, if an initial grade happens early enough, to communicate to the VFX houses the looks being used. This often is send as a CDL or a 3DLUT, per shot, and does not have the viewing transform baked into it. It is also important to communicate the color space that the color corrections should be applied in. Most often, the color correction color space is the same as the visual effects delivery format. The advantage of working with this data downstream is that for computer generated imagery, you have have a better preview of the eventual appearance of the theatrical release.

Trim Passes and Mastering

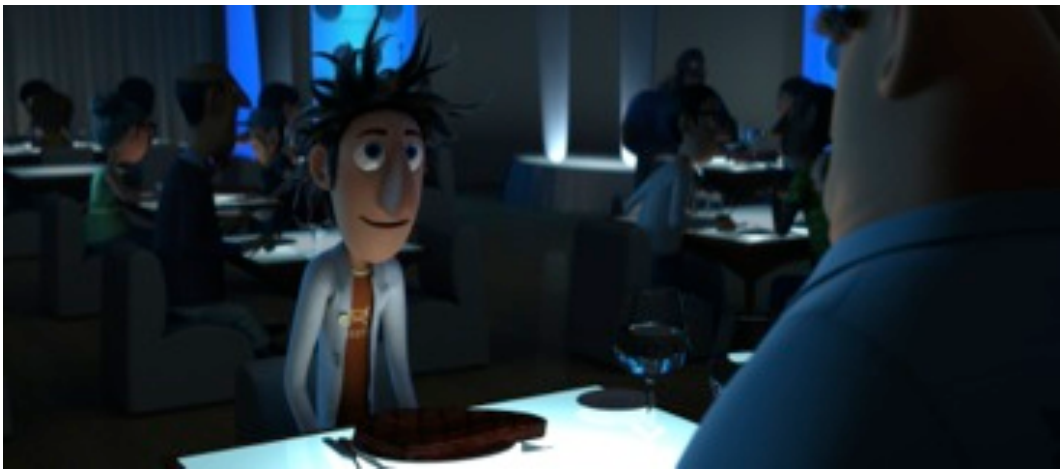
It is common in DI to create masters for a multitude of output devices. For example, on major motion pictures one would expect to have eventual outputs on digital projectional, film releases, home theater releases in both HD and standard def. The general approach is to identify one output process as the "gold standard", and to spend the majority of the artistic time correcting the images to look perfect on that device (in our experience, the theatrical digital projection is usually most appropriate to do first). The director, producers, etc will all be present at this process. Once the main color grade is complete, additional outputs are handled as **trim passes** upon the main output. Trim passes are most often implemented as additional color correction layers added atop primary corrections, and only utilize minor corrections such as contrast, brightness, and saturation. The display transform is device-specific; for trim passes a different view transform tailored to the particular output device is a necessity.

Viewing environment greatly impacts the appearance of colors, most often contrast and colorfulness. Theatrical view environments typically have dark surround, combined with lower screen luminance (48 cd/m2 is common in theatrical projection). In the home theater, a dim surround is common and a screen luminance of 80 cd/m2 is typical. In a desktop office setting, a bright surround is assumed. Thus, if you take the same image, display it using the exact same color reproduction in each environment the appearance will be very different. Color appearance models can be used to attempt to correct for these settings, but trim passes with a human in the loop typically yield higher fidelity results.

If mastering for 3D (stereo), an additional trim pass is required. Due to the extra optics necessary to create the 3D image (additional filters in the projector, in addition to glasses) the 3d image is usually far lower luminance than 2D projection. As lower luminance images tend to appear less colorful, the trim pass for the 3D master typically bump both saturation and contrast.

3.2. Lighting, Rendering, Shading

The stages of rendering, lighting, and shading most closely approximate physical realism when performed in a high-dynamic range, scene-linear color space. Ideally, no color space conversions should be required during the execution of the render, as all input assets such as textures, plate reprojections, and skydomes, can be linearized beforehand. Image viewing of scene-linear data is typically handled by converting to the color space being delivered to digital intermediate (commonly a log color space), and then applying the view transform suitable for the specified display. For convenience, these transforms are typically baked into a single 3D-LUT, though care must be taken to assure the LUT has suitable fidelity over an appropriate HDR domain. As mentioned earlier, you *must* use a viewing transform when viewing HDR scene-linear data.



This image is a direct visualization of a high dynamic range, scene-linear render. Values greater than 1.0 (specular highlights) are clipping in this raw visualization (though preserved in the data).

Using a naive gamma visualization (directly mapping scene-linear to display-linear), results in an image with low apparent contrast and poor highlight rendition (observe clipping on the table).

Using an s-shaped tone curve to visualize the scene-linear render results in a pleasing appearance of contrast, with well balanced highlight and shadow details. Renders by Sony Pictures Imageworks, available for download at opencolorio.org. Images from "Cloudy With a Chance Of Meatballs", © 2009 Sony Pictures Animation Inc. All rights reserved.

Why is scene-linear preferred for lighting? First, the render itself really benefits. Physically plausible light transport renderer mechanisms (aka global illumination) yield natural results when given scenes with high dynamic ranges. Physically-based specular models, combined with area lights, produce physically plausible results with high-dynamic range data. Rendering in scene-linear also allows lights and material components to be re-balanced post-render, with results that track identically to if the original render had been tweaked. The output of renderers are typically floating-point imagery, and are often stored in the OpenEXR format. See appendix 4.1 for additional details on OpenEXR.

One issue to watch out for with high dynamic range renders is scene noise. When scene textures (such as skydomes) contain what would be considered "emissive specular" areas that substantially contribute to the scene illumination (RGB >> 1.0), care must be taken in terms of sampling or noise is likely. Modern rendering optimizations such as multi-importance sampling (MIS) are useful to mitigate such issues. Even still, it is common to paint out very compact and/or bright light sources (such as the sun) from skydomes, and then to add them back into the scene as native renderer lights to allow for lower-noise sampling.

Light shaders also benefit from working with scene-referred linear, specifically in the area of light falloff. In the past, the default mode of working with lights in computer-graphics was to not use falloff. However, when combined with physically-based shading models, using an r^2 light falloff behaves naturally. If one tries to shoehorn in realistic lighting falloff models into lower dynamic range spaces (such as display-referred linear), it's very difficult to avoid clipping. On the downside, one consequence of working with natural light falloff is that it's often required to have very high light intensity values. It is therefore common to express light intensity in user interfaces in terms of "stops", as it's much friendlier to the artist to present an interface value of +"20 stops", compared to a RGB value of "1.048e6".

Antialiasing operations also benefit from scene-linear, though one must be more careful with renderer reconstruction filters. Negatively lobed filters such as Catmull-Rom or Lanczos3 have an increased tendency to exhibit ringing due to the extra dynamic range. This is a particular problem on elements lit with very bright "rim lights", as this creates bright specular highlights directly in contact with edges. There are two common approaches to working around such filtering issues. First, very bright specular samples can be rolled-off such that these samples do not contribute such large amounts of energy. However, this throws out much of the visually significant appearance which adds so much to the realism. Another approach is that the extra energy can be spread amongst neighboring pixels such that the specular hits show an effect analogous to camera flare. Both of these effects can be implemented either during compositing or internal to the renderer; the advantage of the latter is the processing can work on the sub-pixel level for higher quality anti-aliasing.

HDR Environment Captures

HDR captures are increasingly being used to capture onset lighting. The classic capture technique to was photograph a chrome ball, but now productions tend to directly capture the scene either with a multi-exposure fisheye camera setup or with dedicated hardware. One recent extension to this methodology is to capture the scene at high-dynamic ranges from multiple heights and/or locations, which through triangulation allows for energy estimations of the scene-lighting.

Care must be taken in calibrating the HDR lighting acquisition to account for colorimetry, linearity, and white balance, or the resulting lighting data may not integrate into the computer-generated environments. Adding a diffuse sphere during capture is useful in validating the lighting reconstruction later on.

3.3. Compositing

Compositing is the process where the live action plates are merged with the computer generated imagery. Image processing in both scene-linear and logarithmic encoding spaces are both useful, though scene-linear is increasingly the default. As in lighting, the image display must using viewing transforms that emulate the eventual look in DI. Examples of commercially available compositing applications include Nuke, Fusion, After Effects, and Flame.

In feature film visual effects, plates (live action photography) are typically represented on disk as a log encoding of the linear camera data, often as DPX files. Frames when brought into the compositing package are typically converted to scene-linear on input, and then converted back to the original color space on output. End to end, the compositing process represents a 'no-op', and typically has perfect invertibility.

The benefits of compositing in scene-linear are numerous, and similar to the benefits found in rendering, shading, and lighting. All operations which blend energy with spatially neighboring pixels (motion blur, defocus, image distortion, resizing, etc) have more physically plausible (aka realistic) results by default. Antialiasing works better, light mixing preserves the appearance of the original renders and most importantly, even simple compositing operations such as "over" produce more realistic results, particularly on semi-transparent elements such as hair and volumetrics.

In this original scene, we use a defocus operation visualize the energy inherent in different linearizations. Pay particular attention to the string of lights on the top of the bridge span.

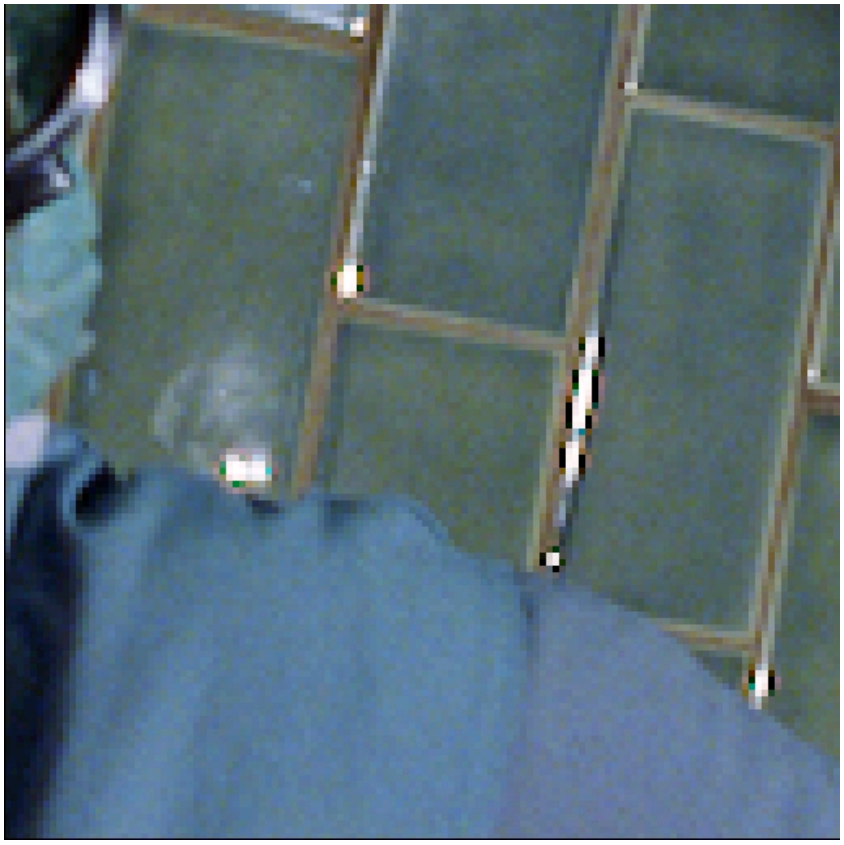
Applying a 50 pixel defocus in scene-linear causes a beautiful bokeh effect on each of the bridge span lights, mimicking the visual look of having performed this defocus during image capture. This is because the pixel values are proportional to light in the original scene, and thus specular pixels have sufficient energy to remain visible when blended with their neighbors. Other energy effects such as motion-blur achieve similar improvements in realism from working in a scene-linear space.

Applying a 50 pixel defocus in display-linear tends to de-emphasize the specular highlights. This is because even though the operation is being applied in "linear", post-display transform, the highlights have already been tone-rendered for a lower dynamic range display and thus do not have physically-plausible amounts of energy.

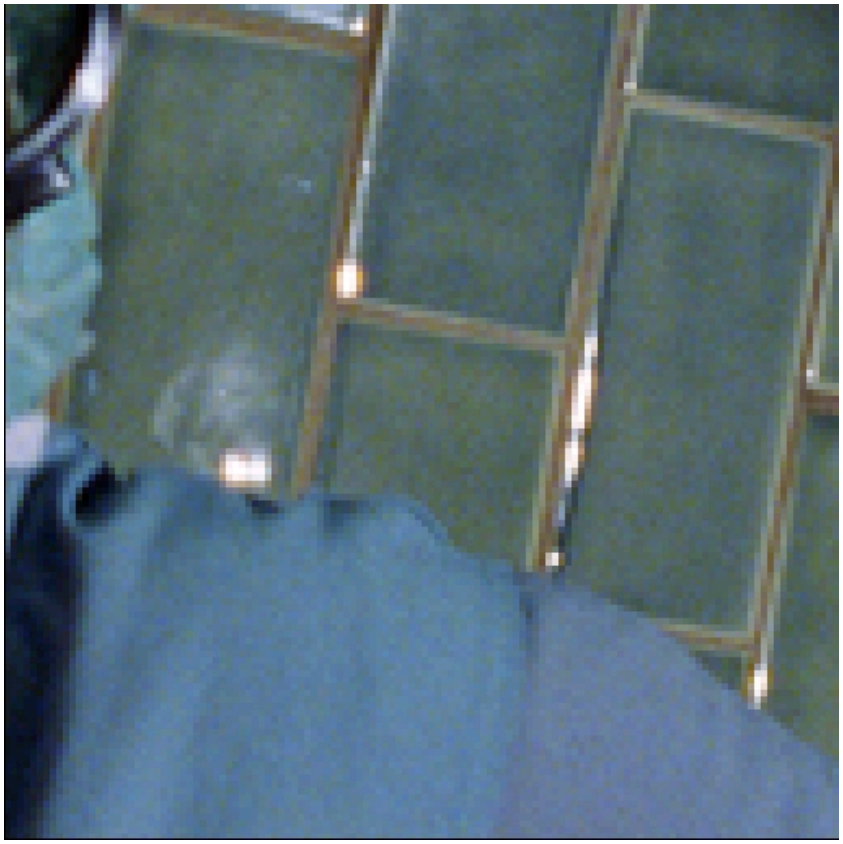
Scene-Linear Compositing Challenges

There are some challenges with working with high-dynamic range in compositing. First, filtering operators that use sharp, negative lobed kernels such as lanczos, keys, sinc, are very susceptible to ringing (negative values around highlights) . While interpolatory filters such as box, gaussian, bilinear, bicubic cannot cause this artifact, they do introduce softening. For cases where sharp filtering is required (lens distortions, resizing, and other spatial warping techniques) a variety of approaches are useful to mitigate such HDR artifacts.

The simplest approach to mitigating overshoot/undershoot artifacts is to rolloff the highlights, process the image, and then unroll the highlights back. While this does not not preserve highlight energy (it has the net effect of reducing specular intensity), the results are visually pleasing and is suitable for processing images with alpha. Another approach to HDR filtering is to apply a simple camera flare model, where very bright highlights share their energy with neighboring pixels. Finally, for images without alpha, converting to log, filtering, and converting back will greatly reduce the visual impact of overshoot and undershoot. Though log-space processing results in a gross distortion of energy, for conversions which are approximately 1:1 (such as lens distortion effects for plates) this works well.



Applying negative-lobed filter kernels to high-dynamic range, scene-linear images (in this case a lanczos3 resize) may cause visuall-significant ringing artifacts. Observe the black silhouettes around the sharp specular highlights.



Ringing artifacts may be avoided when using sharp filters, on scene-referred imagery, by processing in alternate color representations (such as log), using energy-rolloff approaches, or by flaring highlight regions. This image demonstrates the visual appearance of the roll-off technique.

Another challenge in working with scene-linear data in compositing is that tricks often relied upon in integer compositing may not be effective when applied to floating-point imagery. For example, the screen operator (which remains effective when applied to matte passes), is sometimes used on RGB data to emulate a 'partial add'. As the screen operator is ill-defined above 1.0, when applied to HDR data unexpected results will be common. Some compositing packages swap out screen for 'max' when either input is above 1.0, but this is primarily to prevent artifacts and is not artistically helpful. Alternative "partial add" maths such as hypotenuse are useful, but do not exactly replicate the original intent of screen. Another related issue to beware of when compositing is that when combining HDR image, alphas must not go outside the range of [0,1]. While it is entirely reasonable for the RGB channels to have any values (even negative in certain cases!) the compositing operators, such as over, produce totally invalid results on non [0,1] alphas.

One subtlety of working with floating point imagery is that artists must become familiar with some of the corner cases in floating-point representations: NaNs and Infs. For example, if one divides by very small values (such as during unpremultiplication) it's possible to drive a color value up high enough to generate infinity. NaNs are less frequent, but may be introduced during shading (inside the renderer), or during divide by zero operations. Both nans and inf can cause issues in compositing if not detected and cleaned up early, as most image processing algorithms are not robust to their presence and generally fail in unexpected ways.

Working with Log Imagery

Despite HDR having benefits in many compositing operators, there are just some cases even in modern compositing workflows where log-like spaces are still useful. For example, manipulating contrast of HDR images is non-trivial. Contrast is typically describes as a color correction which makes the shadows darker, and the highlights brighter, centered around a pivot. In unconstrained floating-point spaces, if you use a linear + offset contrast operator (mx+b), it is difficult to choose offsets that do not drive pixels negative. And the alternative, a gamma function (exponentiation), has the potential to drive highlights to unreasonably large values. This is one of the cases where a conversion to a log-like space is very helpful. In log space, both the linear and gamma contrast operators have reasonable performance (this is one of the reasons that DI color work is often preferred to work in log-like spaces). Other operations which are useful to perform in log are grain matching, pulling keys, and many spatial distortion tricks. But be careful when using log color spaces in the graph, as not all log transforms preserve the full range of scene-linear code values. Many log conversions clip values above certain highlight colors, below certain shadow colors, and typically all negative values. Finally, compositing must also generate plates for DI that hold up under a variety of color corrections. See the section on "Critical inspection of imagery" for further details.

Plate Timing

When working with plate photography it is often useful to come up with simple color corrections for each plate to neutralize the lighting across a sequence. During the shoot, there will inevitably be color drift across the shots (such as the sun moving behind a cloud, the lights moved to different locations, etc). Using neutralized plates in lighting and compositing leads to nice gains in efficiency as light rigs and sequence standards track better. In most workflows the plate neutralization is done at the top of the compositing graph, the rendered 'neutral' elements are composited in, and then before the file is written out the neutralization is undone. This approach is often called "reverse out", and its advantage is that the output is consistent with the input, independent of which color correction was used. The color corrections used for plate neutralization must be reversible, and as such are typically handled as either simple offsets in camera log space, or as simple gains in linear. More complex operations such as contrast, saturation, etc are usually avoided as these type of corrections break the linearity of the plate and will likely make compositing more difficult (presuming the linearization was properly done to begin with).

It is important to draw a distinction between timing number necessary for plate neutralizations, and the potentially more sophisticated "looks" that are crafted in the DI. At DI, the full suite of color correction operators can be used (contrast, keys, primaries, secondaries, etc), as there is no need to ever invert this process. However, the color neutralization numbers needed for lighting and compositing must be invertible, and as such are typically limited to a simple color balance change. Beware that if you bake in the neutralized color correction, to make sure area very near the top and bottom of the colorspace are preserved. I.e., if your compositing packages uses a log-convert on input, and then you apply timing number, you must confirm that downstream log-converts on the neutral graded plate do not clamp either the high or low end.

Premultiplication

Premultiplication gets a unwarranted bad wrap, as the name implies that one starts with an RGB color, adds the concept of transparency, and then 'premultiplies' by it. This isn't actually true. **Premultiplied RGB is the natural representation for color and transparency** and is output of renderers by default. A nice mental model for premultiplied RGBA is that the RGB channels represent how much light is emitted from within the region of the pixel, and "A" represents how much a pixel blocks light behind it. As alpha represents the fraction of occlusion, to maintain physical correctness alpha values must be between 0 and 1. RGB channels have no such constraints, and can range from -infinity to infinity. Furthermore, there is absolutely no issue with RGBs greater than alpha (a common misconception). Even when alpha is zero, a positive RGBs is completely reasonable, and corresponds to light being emitted from a pixel without occluding anything behind it. Indeed, this situation is common in FX renders for emissive elements such as fire.

So how should we interpret unpremultiplied RGBA? Unpremultiplied RGBA is most easily understood as, "If this pixel were fully opaque, how much light would have been emitted?" This representation is certainly useful in special compositing contexts (such as when pulling keys as a function of luminance), but as converting to unpremultiplied RGBA is a lossy approximation it's prudent to only do the conversion when absolutely necessary.

For additional information on premultiplied vs. unpremultiplied RGBA representations, see [Blinn 98].

3.4. Texture and Matte Painting

Texture painting and matte painting presents a color pipeline challenge. Ideally, one could directly paint in a scene-linear space and use appropriate visualizations to get realistic previews of the eventual render appearance. However, the majority of texture painting applications work most intuitively in display-referred color spaces, where the painted image is directly displayed on the screen. Furthermore, texture reference is usually acquired as display-referred imagery (such as the JPG/TIFF output from a digital camera); scene-linear texture reference is only available in rare situations.

This color map represents the color modulation of the diffuse component of a surface shader, and is painted in a display-referred space. When converted to linear (prior to mipmapping) the resulting color values should range between [0,1] and are representative of physically-plausible surface albedos.

Thus, a common solution is to utilize an "inverse tone rendering", to back convert display-referred imagery to a hypothetical scene-referred space. Conceptually, the texture artist is painting the tone-rendered *appearance* of the texture, not the texture itself. There are a few difficulties with this conversion from display-referred to scene-referred space. First, the tone rendering may not be invertible, particularly when filmic 3D-LUTs are used for image preview. Second, traditional "s-shaped" tone renderings have very horizontal portions of the curve, which when inverted result in very steeply sloped transfer functions. This steep contrast has the effect of amplifying very small changes in input code values. I.e, a steep inverse could results in the situation where a single code value change in a painted texture could correspond to a delta of a stop or more of scene-linear light. This sensitivity is very difficult for artists to work with.

A common solution to both of these issues is to not use a perfect inverse display transform, but instead to create an approximate 1-D inverse that is well behaved in terms of color performance and dynamic range. Of course, as a consequence the texture process is not truly WYSIWYG. Thus, a residual "difference visualization" 3D-LUT is crafted for purposes of color preview in the texture painting tool. This residual 3D preview is never baked into the imagery, merely used as a visualization.

It is often useful to consider matte painting and texture painting as two different color transforms. In matte painting, it is very common for the artist to want to paint values which will eventually utilize the full scene-linear dynamic range, even when working in a display-referred space. Thus, an inversion which allows for the creation of bright specular values is preferable. In such cases, it is often convenient to provide a visual preview both at a normal exposure, as well as a few stops darker, to let the artist have a good feel for the dynamic range in their plate.

In texture painting, the artist often paints color maps which when linearized will control material color reflectivity, as an input to physically-based shading. In these situations, linear data from [0,1] is required and necessitates a color transform that gracefully handles these limits. In both matte painting and texture painting, one must also distinguish painted color maps from data maps (bump maps, control maps, etc), which should not be processed colorimetrically.

Matte painters typically prefer to paint in a display-referred color space for convenience. But as renders are natively in scene-linear, an "inverse display transform" is used to synthesize plausible scene-linear colors prior to rendering.

When texture acquisition is done in a controlled environment, it's possible to create scene-linear texture conversions leveraging camera raw workflows. Command-line utilities, such as dcraw, allow color pipelines which capture a reasonable linear representation surface color. The use of polarizing filters and softbox lighting during acquisition further allows for increased reference texture fidelity.

Another axis of variation in texturing is when to perform the conversion to scene-linear. The approach we advocate is to linearize *prior* to mipmap texture generation. The primary advantage of this approach is that scene-linear energy is preserved through all the mipmap levels, allowing for the highest fidelity sampling and fewest color space related artifacts. Further, disallowing color space conversions at shading time prevents shaders from doing evil (non-linear) color math. The disadvantage is that the storage requirements for linearized data are potentially increased. I.e., even if a texture is painted at 8-bits of precision in a display-referred space, after linearization to scene-linear increased bit-depths are required. When dealing with 16-bit painted textures, this is less of a concern.

3.5. Critical Inspection of Imagery

One unfortunate circumstance in visual effects and animation production is that the typical artist desktop display is not as high fidelity as the eventual theatrical viewing environment. I.e., the majority of the time the people in the theater will see more detail than the artist. Furthermore, while artists typically craft the scene at a particular exposure, during DI is possible that the imagery will be stylistic taken in a direction the artists did not anticipate. It is therefore critical to inspect the imagery under a variety of "worst-case" color corrections, to make sure the computer generate imagery is as robust as possible.

At a minimum, all artists working in scene-linear (namely the lighter and compositor) need to view their imagery stopped up and down in scene-linear on a regular basis. Only then can the artists get a true sense of the dynamic range in their imagery, and to match computer-generated elements in a similar manner. Then in compositing, it's very handy to convert the imagery to the colorspace (and quantization / clamping) that will be delivered to DI, to test how this imagery holds up to drastic grades. We recommend playing with substantial log space offsets, as well as contrast and saturation boost, to make sure all portions of the imagery track in a consistent manner. This is particularly important to match across shots and sequences, for it's all too easy to make individual shots that are self consistent, yet fall apart when viewed side to side. When stopping down, it's important to make sure that highlight intensity, color, and sharpness are consistent. And most critical is to stop up the imagery to make sure the shadows have a consistent color range, that there are no "flat-black" portions of the image, and that grain and black noise track between the live action and the computer generated imagery.

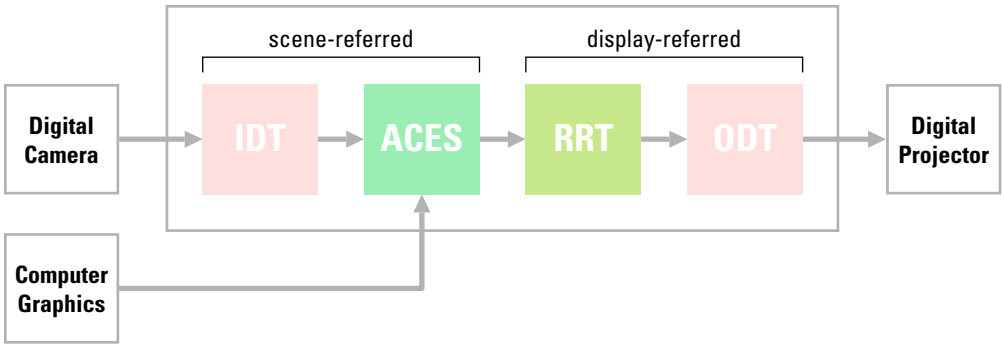
Histograms are a very useful tool when analyzing image fidelity, and often help forensically track down what processing has been applied to an image. For example, log frames from certain image sources often are hard-clipped in the shadow region. This can be detected in the histogram, as well as sharpening filters that applied overly aggressive undershoot and overshoot. It's also possible to detect from the histogram if any image quantization occurred (this appears as comb-like peaks and valleys). We have also found that its a valuable tool to look at histograms in HSV-like color spaces, and to quality-check composites in these alternate spaces.

There are other image artifacts, such as aliasing, that are best inspected looking at moving imagery. For example, the use of low-tech interpolatory filters, such as "nearest neighbor interpolation", is sometime unobjectionable on still imagery, yet highly objectionable on slowly animating transforms (particularly near horizontal lines). Always be on the lookout for moire patterns and other high-frequency artifacts, particularly when spatial transformations have been applied to digital motion-picture imagery with high frequency details (such as cloth patterns). If all digital imagery were ideally sampled, this would not be an issue. But many camera manufacturers generate images which are "overly sharp" (aka aliased) and certain motion transformations accentuate the artifacts).

For stereo deliveries, it's critical to quality check both eyes individually, as well as together in a proper stereo viewing environment. There are many stereo artifacts which are only visible on real 3D displays, and catching these early on is the easiest way to handle artifacts.

3.6. ACES

The Academy of Motion Picture Arts & Science has proposed a unification of scene-linear floating-point workflows, which adheres to many of the approaches taught in this class. If successful, the ACES project will allow for the unambiguous interchange of floating point imagery. At its core is the **Academy Color Encoding Space**, called **ACES** for short. This color space is a high-dynamic range, scene-linear space, with middle gray pegged to 0.18, and a very wide color gamut. When stored on disk, aces files are be stored in a constrained version of the OpenEXR format, with the *.aces* file extension.



The ACES workflow defines scene-referred working space, and a reference viewing transform in an attempt to standardize float-linear interchange in the motion-picture industry.

The Academy also defines the viewing transform necessary for viewing ACES files. The view transform is conceptually segmented into two portions. First, the **Reference Rendering Transform (RRT)** applies a local contrast boost and renders scene-linear data to display linear. Then, a second portion of the view transform called the Output Device Transform (ODT) is used to provide gamut mapping and further tone mapping to the target output device. The RRT portion is constant across all displays, but the ODT varies from output to output. ODTs are provided for common display specifications such as sRGB, Rec709, DCI-P3, X'Y'Z', etc.

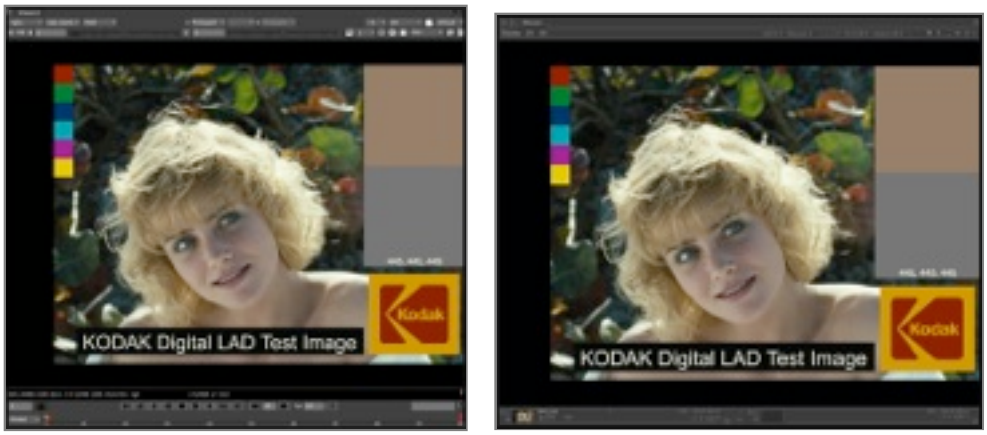
On the input side, there are also a series of published input device transforms (IDTs) which convert input colorimetry to scene-linear (ACES). Film scans are a special case, where the academy defines a new density encoding standard, the Academy Density Encoding (ADX), which is typically stored in a DPX file. Both 10 and 16 bit flavors of ADX are provided, which encode for different negative density ranges. The academy input transform for film negatives is not stock specific, but relies on a generic film input linearization.

In the ACES workflow, the intent is for artistic changes to be made by manipulating the aces data, as viewed through the RRT + ODT. All of the transforms (both input and output) for ACES are defined using the Color Transformation Language (CTL), which was contributed by Industrial Light & Magic. CTL is an interpreted language similar in spirit to shader languages commonly used in renderers, but with a focus on color transforms. CTL provides a rich color correction API, including scattered data interpolation. CTL is run independently per pixel, and is thus suitable for baking into 1D/3D LUTs - allowing for real-time performance.

ACES is also fully supported using OpenColorIO, as detailed in the next section.

3.7. OpenColorIO

OpenColorIO (OCIO) is an open source color pipeline created by Sony Pictures Imageworks (and the author), which implements most of the concepts concepts in this course. It has two major goals: **consistent color transforms**, and **consistent image display**, across multi-application / multi-OS color pipelines.



Two applications, configured with OpenColorIO, provide matched image display. The image on the left is Nuke, which has loaded a log film plate in the DPX image format. On the right is the display in Katana, which has loaded a scene-linear OpenEXR version of this image. Both applications utilize OpenColorIO for color management.

The design goal behind OCIO is to decouple the color pipeline API from the specific color transforms, which allows supervisors setting up color processes to tightly manage (and experiment) with color transforms from a single location. "Setup your color once, and all application obey." Unlike other color management solutions (such as ICC) OCIO is natively designed to handle both scene-referred (float HDR) and display-referred imagery. All color transforms are loaded at runtime, from a color configuration (.ocio file), optionally crafted by the user. OCIO does not make *any* assumptions about your imagery; all color transformations are "opt in". Sony Pictures Imageworks has also open-sourced some of the real color configurations for productions, such as those used in *Cloudy with a Chance of Meatballs* and *Spider-Man*. This lets users at home (or other studios) experiment with validated color pipelines. OpenColorIO also ships with a configuration compatible with the Acedemy's ACES effort, allowing for graceful experimentation with this "next-gen" color pipeline in existing applications.

OCIO color configuration files define all of the conversions that may be used. For example, if you are using a particular camera color space, one would define the conversion from the camera's color encoding to scene-linear. You can also specify the display transforms (for multiple displays) in a similar manner. OCIO transforms can rely on a variety of built-in building-blocks, including all common math operations and the majority of common lookup table formats. OCIO also has full support for both CPU and GPU pathways, in addition to full support for CDLs and per-shot looks.

OpenColorIO is in use at many of the major visual effects and animation studios, and is also supported out of the box in commercial software. See opencolorio.org for up-to-date information on supported software, and to download the source code for use at home.

4. Appendix

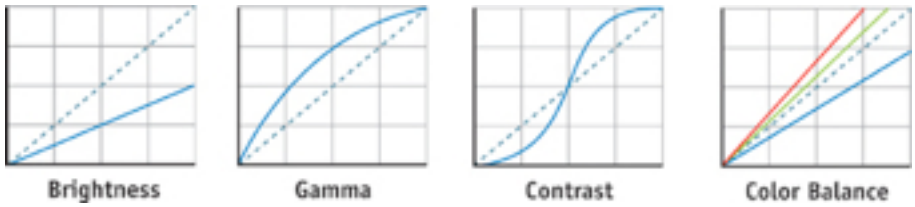
This page intentionally left blank.

4.1. Lookup Tables

Lookup tables (LUTs) are an technique for optimizing the evaluation of functions that are expensive to compute and inexpensive to cache. By precomputing the evaluation of a function over a domain of common inputs, expensive runtime operations can be replaced with inexpensive table lookups. If the table lookups can be performed faster than computing the results from scratch, then the use of a lookup table will yield significant performance gains. For data requests that fall between the table's samples, an interpolation algorithm can generate reasonable approximations by averaging nearby samples. LUTs are also useful when wanting to separate the calculation of a transform from its application. For example, in color pipelines it is often useful to 'bake' a series of color transforms into a single lookup table, which is then suitable for distribution and re-use, even in situations where the original data sets are not appropriate for distribution.

1-D LUTs

A lookup table is characterized by its dimensionality, that is, the number of indices necessary to index an output value. The simplest LUTs are indexed by a single variable and thus referred to as one-dimensional (or 1D) LUTs. One-dimensional LUTs have long been utilized in image processing, most commonly in the form of the monitor gamma table. Typically leveraging three LUTs—one for each color channel—these tables enable the computationally efficient modification of pixel intensity just before an image reaches the monitor.



Consider an analytical color operator, f(x), applied to an 8-bit grayscale image. The naive implementation would be to step through the image and for each pixel to evaluate the function. However, one may observe that no matter how complex the function, it can evaluate to only one of 255 output values (corresponding to each unique input). Thus, an alternate implementation would be to tabulate the function's result for each possible input value, then to transform each pixel at runtime by looking up the stored solution. Assuming that integer table lookups are efficient (they are), and that the rasterized image has more than 255 total pixels (it likely does), using a LUT will lead to a significant speedup.

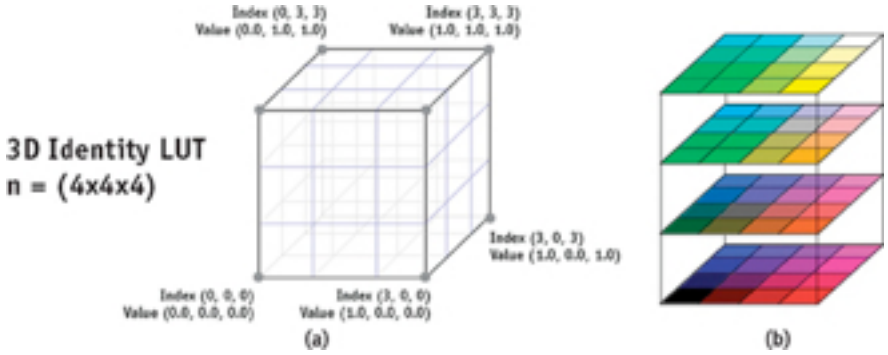
All color operators that can be parameterized on a single input variable can be accelerated using 1D LUTs, including the brightness, gamma, and contrast operators. By assigning a 1D LUT to each color channel individually, we can implement more sophisticated operations, such as color balancing. For those familiar with the Photoshop image-processing software, all "Curves" and "Levels" operations can be accelerated with 1D LUTs.

Unfortunately, many useful color operators cannot be parameterized on a single variable, and are thus impossible to implement using a single-dimensional LUT. For example, consider the "luminance operator" that converts colored pixels into their grayscale equivalent. Because each output value is derived as a weighted average of three input channels, one would be hard-pressed to express such an operator using a 1D LUT. All other operators that rely on such channel "cross talk" are equally inexpressible.

3-D LUTs

Three-dimensional lookup tables offer the obvious solution to the inherent limitation of single-dimensional LUTs, allowing tabular data indexed on three independent parameters.

Whereas a 1D LUT requires only 4 elements to sample 4 locations per axis, the corresponding 3D LUT requires 43 = 64 elements. Beware of this added dimensionality; 3D LUTs grow very quickly as a function of their linear sampling rate. As a direct implication of smaller LUT sizes, high-quality interpolation takes on a greater significance for 3D LUTs.



Complex color operators can be expressed using 3D LUTs, as completely arbitrary input-output mappings are allowed. For this reason, 3D LUTs have long been embraced by the colorimetry community and are one of the preferred tools in gamut mapping (Kang 1997). In fact, 3D LUTs are used within ICC profiles to model the complex device behaviors necessary for accurate color image reproduction (ICC 2004).

The majority of color operators are expressible using 3D LUTs. Simple operators (such as gamma, brightness, and contrast) are trivial to encode. More complex transforms, such as hue and saturation modifications, are also possible. Most important, the color operations typical of professional color-grading systems are expressible (such as the independent warping of user-specified sections of the color gamut).

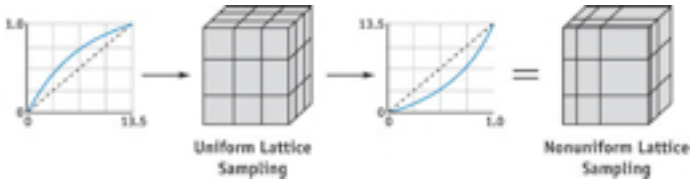
Unfortunately, in real-world scenarios, not all color transforms are definable as direct input-output mappings. In the general case, 3D LUTs can express only those transforms that obey the following characteristics:

A pixel's computation must be independent of the spatial image position. Color operators that are influenced by neighboring values, such as Bayesian-matting (Chuang et al. 2001) or garbage masks (Brinkman 1999), are not expressible in lookup-table form.

The color transform must be reasonably continuous, as sparsely sampled data sets are ill suited to represent discontinuous transformations. If smoothly interpolating over the sampled transform grid yields unacceptable results, lookup tables are not the appropriate acceleration technique.

The input color space must lie within a well-defined domain. An "analytically" defined brightness operator can generate valid results over the entire domain of real numbers. However, that same operator

baked into a lookup table will be valid over only a limited domain (for example, perhaps only in the range [0,1]).



3D LUTs can be extended for use on HDR color spaces by wrapping the 3d LUT lookup in a matched set of 1D ‘shaper’ LUTs. Say we have set our ceiling at a pixel value of 100.0. Dividing this into equally sampled regions for a 32x32x32 LUT yields a cell size of about 3. Assuming a reasonable exposure transform, almost all perceptually significant results are going to be compressed into the few lowest cells, wasting the majority of our data set. We thus want to place our samples in the most visually significant locations, which typically occur closer to the dark end of the gamut. We achieve this effect by wrapping our 3D lookup-table transform with a matched pair of 1D "shaper" LUTs. The ideal shaper LUT maps the input HDR color space to a normalized, perceptually uniform color space. The 3D LUT is then applied normally (though during its computation, this 1D transform must be accounted for). Finally, the image is mapped through the inverse of the original 1D LUT, "unwrapping" the pixel data back into their original dynamic range.

This section was adapted from GPU Gems 2, Chapter 24 (available online). See original for further implementation details.

4.2. ASC-CDL

The world of color correction, particularly as it is handled onset, has a huge amount of variation. Even though it is common to apply a "primary grade" (consisting of a scale operation, some offsets, and maybe a gamma and saturation adjustment), every manufacturer tends to apply these corrections in a different order which means that the settings are not portable. The American Society of Cinematographers (ASC) thus has created a color correction specification to try and bring a bit of order to the chaos. Similar to the EDL (edit decision list) in use by editorial systems, the ASC came up with the CDL (Color Decision List) format. This specification also defines the math for what is expected on a "primary" correction.

The CDL defines a fixed transform color correction:

1. Scaling (3 channels independent)
2. Offset (3 channels independent)
3. Power (exponent) (3 channels independent)
4. Saturation (with a fixed 709 saturation ratio)

Having a fixed order is not always ideal. For example, in a CDL you cannot desaturate the image to grayscale, and *then* tint the image using scales + offsets. (The opposite is allowed, of course). But having an unambiguous way to interchange simple grade data is a huge improvement in interoperability. The ASC has also defined an XML format for the grade data, called the ColorCorrectionCollection. The Scaling, Offset and Power are sent as a set of 9 numbers (the SOP) and the saturation is sent as a single number (SAT).

```
<ColorCorrectionCollection xmlns="urn:ASC:CDL:v1.01">
    <ColorCorrection id="example_correction_01">
        <SOPNode>
            <Slope> 1.1 1.1 1.1 </Slope>
            <Offset> -0.07 -0.07 -0.07 </Offset>
            <Power> 1.0 1.0 1.0 </Power>
        </SOPNode>
        <SatNode>
            <Saturation> 1.0 </Saturation>
        </SatNode>
    </ColorCorrection>
</ColorCorrectionCollection>
```

Example .ccc (Color Correction Collection) xml file demonstrate the SOP and Sat elements.

So what does the ASC CDL *not* define? Color space. If one applied an additive offset to logarithmic encoded data, the result is very different than if the same offset is applied to scene-linear imagery. The CDL also does not specify if any viewing LUTs were used. This ambiguity is both CDLs biggest strength and its biggest weakness. It is a weakness, because if you get a CDL in isolation the color correction is still not well defined. However, this is also CDLs biggest strength as it enables CDLs to be highly versatile, serving as building block across many color pipelines. For example, the author is familiar with CDLs being used to send plate neutralization (log offsets) between facilities, and also used to send display-referred color corrections from onset to the VFX vendors.

4.3. File Formats

OpenEXR

OpenEXR is an open-source image format creating by ILM [7] in 1999, which has near universal adoption in the VFX and animation industries . EXR is primarily intended for storing floating point, scene referred imagery. Although EXR was not the first image format suitable for storing HDR float data, is it the most popular in feature film production due to its efficient losses compression codecs, support for 16-bit (half) float pixel type, in addition to other professional features such as data window / display window tracking, multiple layers, rich metadata encoding, and the lack of legacy format baggage.

In our experience, the half data format is sufficient for color images. (We consider color renders to be RGB(a) channels that one would view using a tone-rendered view transform, such as beauty renders, specular color, diffuse color, per-light AOVs, etc). For "data" images which represent numerical quantities (such as depth, normals, control maps, etc) the full precision of 32-bit float is usually required.

The maximum value for float16 is 65504.0f. The minimum value which can be represented, without a decrease in numerical precision, is 6.1035e-05f. It is important to remember when dealing with float16 data, that the bits are allocated very differently from an integer encodings. Whereas integer encodings have uniform allocation of bits over the entire coding space, the half-float format (like all floating-point formats) has increased precision in the low end, and decreased precision in the high end. Relatedly, if one has an uint16 image, converts to an f16 representation using the code values from [0,1], and then converts back, 16-bits of precision will not be maintained. Specifically, the round trip will preserve 7168 unique code values (of the original 65535), which corresponds to approximately 12.8 bits of round-trip precision.

OpenEXR supports a variety of compression options. In our experience, piz (wavelet) offers the highest lossless compression ratio on grainy material, but is relatively expensive computationally. The zip options offer a reasonable compression ratio on computer-generated elements, and is less computationally intense. The b44 codec is lossy, and intended for real-time playback. OpenEXR also can store mipmapped representations, and is thus suitable for use in the renderer as input textures.

DPX

DPX is a SMPTE standardized image format that is commonly used in the motion-picture industry. While the DPX format supports a variety of pixel types - even float32 - DPX is most commonly associated with storing uncompressed, 10-bit unsigned integer RGB imagery (though 16-bit DPX is catching up in popularity). Unlike EXR - which is synonymous with scene-linear colorimetry - DPX does not have a canonical color space. It is common for DPX to store any number of integer camera log encodings, in addition to broadcast-ready Rec709. Generally, if you've got a DPX file the only way to know for sure what you have is through communication with the person who sent it (or detailed forensic analysis). DPX has limited metadata support, including a few settings related to colorimetry. But beware these flags, even under the best of intentions their fidelity is rarely preserved for long.

---

[7] EXR stands for "**EX**tended **R**ange". Internally, the format was original known as "**IM**age **F**ormat"; a name that lives on in the codebase as the C++ namespace.

4.4. DCI P3 and X'Y'Z'

The Digital Cinema Initiative (DCI) specification defines the standards for digital cinema mastering and theatrical exhibition, including colorimetry.

A new color encoding, X'Y'Z', is specified for image encoding, which is an output-referred, gamma 2.6 encoded version of CIE XYZ with a reference white point at 48cd/m2. As the X'Y'Z' coding space spans an area larger than the visual gamut, a minimum display gamut, P3, is defined. The P3 primaries are very saturated relative to television standards; sRGB red is a relatively desaturated orange color, while P3 red is "blood red"; almost on the spectral locus.

The intent of the X'Y'Z' coding space is that the full color appearance of the theatrical imagery (such as any film emulation 3D-LUTs) is already baked into the X'Y'Z' image. So an image in this color space is completely unambiguous; there should be essentially no image variation between properly calibrated theatrical digital projectors.

The DCI specification chose a gamma 2.6 encoding after a series of perceptual experiments using "golden-eye observers", to maximize the bit-depth fidelity under theatrical viewing conditions. DCI specifies 12 bits per channel, which is intended to prevent banding artifacts under even the most trying conditions. DCI also specifies a series of color patches which are useful in calibration. (Please refer to the specification for additional details.) X'Y'Z' files are encoded using JPEG-2000 compression, which is a lossy, wavelet compression codec. No inter-frame compression is used, which allows for both forwards and backwards scrubbing of the format. The DCI specification also specifies two resolutions known as "2K" and "4K". The 2K raster is 2048x1080 and the 4K raster is 4096x2160. DCI compliant films must fill at least one of the the axes. Material with an aspect ration of 1:85 is delivered for a "2K" release at 1998x1080, and 2:35 material is delivered at 2048x858. Most X'Y'Z' encoders accept 16-bit tiffs, so it's convention to use the full 16-bit code range of 0-65535, and then to let the compressor quantize to 12-bits.

See [Kennel 07] for additional details on mastering for digital cinema. The full DCI specification is also available online.

DCI-P3 Color Encoding (Gamma 2.6)

- Red: 0.680, 0.320
- Green: 0.265, 0.690
- Blue: 0.150, 0.060
- White: 0.3140, 0.3510, 48cd/m2

X'Y'Z' Example Code Values (12-bit representation):

P3 Red-1: 2901, 2171, 100
P3 Green-1: 2417, 3493, 1222
P3 Blue-1: 2014, 1416, 3816
P3 White: 3794, 3960, 3890

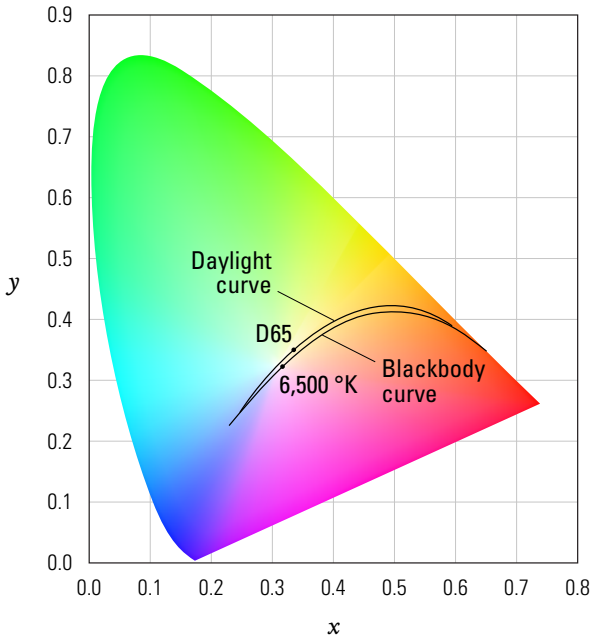
4.5. Daylight Curve vs. Blackbody Curve

Colors near whites are often referenced with respect to their **color temperature**. This single number denotes the relative red vs. blueness of the color. Low color temperatures appear redder (warmer), and high temperatures appear bluer (cooler). The projection of a color onto the blackbody curve is sometimes known as the CCT (correlated color temperature). As CCT is only a single measure of color, a very large number of colors with different appearances can all have matching CCTs. (Roughly speaking, these matching CCT colors would vary in the direction of how much green they contained).

The blackbody curve (and the corresponding measure of color temperature) is based on material physics. Specifically, all materials emit light referred to as "blackbody radiation" as a function of temperature. Hotter objects emit more light, and as the temperature increases so does the frequency. Near room temperature, objects radiate mostly infrared light, which we feel as heat. When an object is heated to high enough temperatures, the emitted radiation increases enough in frequency to spill over into the visible spectrum starting with the red wavelengths; the object appears to glow "red hot". As an object is heater even further, the frequency of the emitted object will increase until reds and blues are in balance (aka, "white hot"), and then further increases will appear blue when hottest. Plotting these colors forms the blackbody curve.

The daylight curve is a series of standard CIE illuminants, which have the intent to simulate an "average color" of daylight, with different CCTs. Colors on the daylight curve are often preferred in display calibrations due to their their "neutral white" appearance. For example, rec709 specifies the use of a D65 illuminant. Other defined daylight illuminants include D50 and D55.

While the daylight illuminants roughly have a CCT matching their names, it's important to note that the daylight curve is parallel, but different, from the blackbody curve. So you must distinguish between 6500K CCT, vs D65.



The daylight curve and the blackbody emission curve, while roughly parallel, are distinct. As visualized above, D65 is a distinct color from an idealized 6500K blackbody emitter.

5. Acknowledgements

Course notes by Jeremy Selan.
Illustrations by Kazunori Tanaka.

A very special thanks goes to:

- Sony Pictures Imageworks and Rob Bredow in particular, for the opportunity to contribute publicly to the color community. Addition thanks to Erik Strauss, Bob Peitzman, the Katana team, and all the artists at Imageworks.
- Contributors to OpenColorIO, including Malcolm Humphreys, Ben Dickson, Mark Fickett, and Joseph Slomka.
- The Foundry
- The Academy's Scientific and Technical Council including Alex Forsythe, Ray Feeney, and Andy Maltz.

6. References & Further reading

Books

Ansel Adams – The Camera, Book 1. Little, Brown, and Company.
Ansel Adams – The Negative, Book 2. Little, Brown, and Company.
Ansel Adams – The Print, Book 3. Little, Brown, and Company.
Jim Blinn – Jim Blinn's Corner: Dirty Pixels. Morgan Kaufmann.
Mark Fairchild – Color Appearance Models. Addison-Wesley.
Ed Giorgianni – Digital Color Management: Encoding Solutions. Prentice Hall.
Glenn Kennel – Color and Mastering for Digital Cinema. Focal Press.
R.W.G. Hunt – Measuring Colour. Fountain Press.
Charles Poynton – A Technical Introduction to Digital Video. Wiley.
Erik Reinhard – Color Imaging, Fundamentals and Applications. AK Peters, Ltd.
Gunter Wyszecki & W.S. Stiles– Color Science: Concepts and Methods. Wiley & Sons, Inc.

Papers

Giorgianni, Ed. Color Management for Digital Cinema: A Proposed Architecture and Methodology for Creating, Encoding, Storing and Displaying Color Images in Digital Cinema Systems. Submitted to the Science and Technology Council, Academy of Motion Picture Arts and Sciences. 2005.

Online

Bruce Lindbloom - Online resource for color conversion math
Charles Poynton - Information for video standards and gamma, available for download.
DCI Specification - Standards for theatrical digital distribution and colorimetry.
DCraw - Free, high quality, camera raw convertor
OpenColorIO - Open Source Color Management Framework for Visual Effects and Animation
Visualizing the XYZ Color Space (YouTube) - Visual exploration of CIE XYZ