The rehabilitation of gamma

Charles Poynton
poynton @ poynton.com
www.poynton.com

Abstract

Gamma characterizes the reproduction of tone scale in an imaging system. Gamma summarizes, in a single numerical parameter, the nonlinear relationship between code value – in an 8-bit system, from 0 through 255 – and luminance. Nearly all image coding systems are nonlinear, and so involve values of gamma different from unity.

Owing to poor understanding of tone scale reproduction, and to misconceptions about nonlinear coding, gamma has acquired a terrible reputation in computer graphics and image processing. In addition, the world-wide web suffers from poor reproduction of grayscale and color images, due to poor handling of nonlinear image coding. This paper aims to make gamma respectable again.

Gamma's bad reputation

The left-hand column in this table summarizes the allegations that have led to gamma's bad reputation. But the reputation is ill-founded – these allegations are false! In the right column, I outline the facts:

Misconception	*Fact*
A CRT's phosphor has a nonlinear response to beam current.	The *electron gun* of a CRT is responsible for its nonlinearity, not the phosphor.
The nonlinearity of a CRT monitor is a defect that needs to be corrected.	The nonlinearity of a CRT is very nearly the inverse of the lightness sensitivity of human vision. The nonlinearity causes a CRT's response to be roughly perceptually uniform. Far from being a defect, this feature is highly desirable.
The main purpose of gamma correction is to compensate for the nonlinearity of the CRT.	The main purpose of gamma correction in video, desktop graphics, prepress, JPEG, and MPEG is to code luminance or tristimulus values (proportional to intensity) into a perceptually-uniform domain, so as optimize perceptual performance of a limited number of bits in each *RGB* (or *CMYK*) component.

Reprinted from Rogowitz, B.E., and T.N. Pappas (eds.), *Human Vision and Electronic Imaging III*, Proceedings of SPIE/IS&T Conference 3299, San Jose, Calif., Jan. 26–30, 1998 (Bellingham, Wash.: SPIE, 1998). © 2002-03-14 Charles Poynton

Misconception	*Fact*
Ideally, linear-intensity representations should be used to represent image data.	If a quantity proportional to intensity represents image data, then 11 bits or more would be necessary in each component to achieve high-quality image reproduction. With nonlinear (gamma-corrected) coding, just 8 bits are sufficient.
A CRT is characterized by a power function that relates luminance L to voltage V': $L = (V')^{\gamma}$.	A CRT is characterized by a power function, but including a black-level offset term: $L = (V' + \varepsilon)^{\gamma}$. Usually, γ has a value quite close to 2.5; if you're limited to a single-parameter model, $L = (V' + \varepsilon)^{2.5}$ is much better than $L = (V')^{\gamma}$.
The exponent γ varies anywhere from about 1.4 to 3.5.	The exponent itself varies over a rather narrow range, about 2.35 to 2.55. The alleged wide variation comes from variation in *offset* term of the equation, not the exponent: Wide variation is due to failure to correctly set the black level.
Gamma correction is accomplished by inverting this equation.	Gamma correction is roughly the inverse of this equation, but two alterations must be introduced to achieve good perceptual performance. First, a linear segment is introduced into the transfer function, to minimize the introduction of noise in very dark areas of the image. Second, the exponent at the encoder is made somewhat greater than the ideal mathematical value, in order to impose a *rendering intent* that compensates for subjective effects upon image display.
CRT variation is responsible for wide variability in tone scale reproduction when images are exchanged among computers.	Poor performance in image exchange is generally due to lack of control over transfer functions that are applied when image data is acquired, processed, stored, and displayed.
Macintosh monitors have nonstandard values of gamma.	All CRT monitors, including those used with Macintosh computers, produce essentially identical response to voltage. But the Macintosh QuickDraw graphics subsystem involves a lookup table that is loaded by default with an unusual transfer function. It is the default values loaded into the lookup table, not the monitor characteristics, that impose the nonstandard Macintosh gamma.
Gamma problems can be circumvented by loading a lookup table having a suitable gamma value.	Loading a particular lookup table, or a particular value of *gamma,* alters the relationship of data in the frame buffer to linear-light "intensity" (properly, luminance, or tristimulus value). This may have the intended effect on a particular image. However, loading a new lookup table will disturb the code-to-luminance mapping that is assumed by the graphics subsystem, by other images, or by other windows. This is liable to alter color values that are supposed to stay fixed.
Macintosh computers are shipped from the factory with gamma set to 1.8. SGI machines default to gamma of 1.7. To make an SGI machine display pictures like a Mac, set SGI gamma to 1.8.	On the Macintosh, setting a numerical gamma setting of g loads into the framebuffer's lookup table a power function with the exponent $^{g}/_{2.61}$. On an SGI, setting a numerical gamma setting of g loads into the lookup table a power function with the exponent $^{1}/_{g}$. To make an SGI machine behave like a Mac, you must set SGI gamma to 1.45.
Gamma problems can be avoided when exchanging images by tagging every image file with a suitable gamma value.	Various tag schemes have been standardized; some tags are coded into image files. However, application software today generally pays no attention to the tags, so tagging image files is not helpful today. It is obviously a good idea to avoid subjecting an image file to cascaded transfer functions during processing. However, the tag approach fails to recognize that image data should be originated and maintained in a perceptually-based code.
JPEG compresses *RGB* data, and reproduces *RGB* data upon decompression. The JPEG algorithm itself is completely independent of whatever transfer function is used.	JPEG and other lossy image compression algorithms depend on discarding information that won't be *perceived*. It is vital that the data presented to a JPEG compressor be coded in a perceptually-uniform manner, so that the information discarded has minimal perceptual impact. Also, although standardized as an image *compression* algorithm, JPEG is so popular that it is now effectively an image *interchange* standard. Standardization of the transfer function is necessary in order for JPEG to meet its users' expectations.

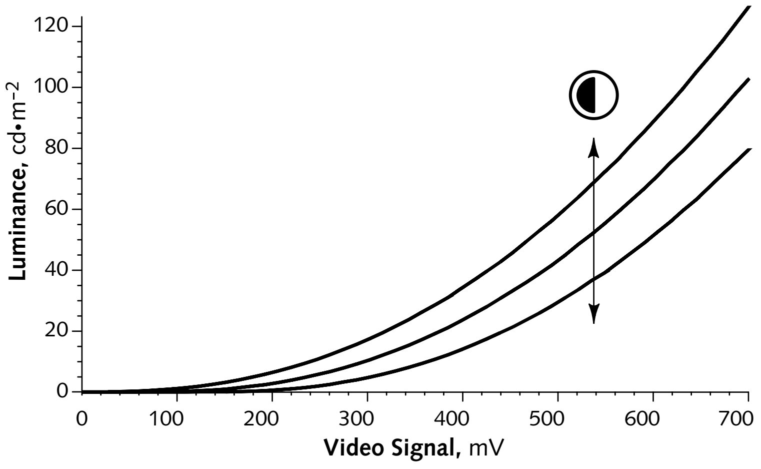


Figure 1 **CRT's transfer function** is shown at three different settings of the CONTRAST (or PICTURE) control. Here I show CONTRAST altering the *y*-axis (luminance) scaling; owing to the properties of a power function, scaling the *x*-axis (video signal) has an equivalent effect. The graph indicates a video signal having a voltage from zero to 700 mV. In a typical eight-bit digital-to-analog converter in a computer graphics subsystem, black is at code zero, and white is at code 255.

Intensity

Intensity is the rate of flow of radiant energy, per unit solid angle – that is, in a particular, specified direction. In image science, we measure power over some interval of the electromagnetic spectrum. We're usually interested in power radiating from or incident on a surface. Intensity is what I call a *linear-light* measure, expressed in units such as watts per steradian.

The CIE has defined *luminance*, denoted *Y*, as intensity per unit area, weighted by a spectral sensitivity function that is characteristic of vision. The magnitude of luminance is proportional to physical power; in that sense it is like intensity. But the spectral composition of luminance is related to the brightness sensitivity of human vision.

Video equipment forms a *luma* component *Y′* as a weighted sum of *nonlinear R′G′B′* primary components. The nonlinear quantity is often incorrectly referred to as *luminance* by video engineers who are unfamiliar with color science.

Luminance can be computed as a properly-weighted sum of linear-light (tristimulus) red, green, and blue primary components. For contemporary video cameras, studio standards, and CRT phosphors, the luminance equation is this:

$$^{709}Y = 0.2126\,R + 0.7152\,G + 0.0722\,B$$

See Olson, Thor, "Behind Gamma's Disguise," in *SMPTE Journal*, v. 104, p. 452 (June 1995).

The luminance generated by a physical device is usually not proportional to the applied signal – usually, there is a nonlinear relationship. A conventional CRT has a power-law response to voltage: Luminance produced at the face of the display is approximately the applied voltage raised to the five-halves power. The numerical value of the exponent of this power function, 2.5, is colloquially known as *gamma*. This nonlinearity must be compensated in order to achieve correct reproduction of luminance. An example of the response of an actual CRT is graphed, at three settings of the CONTRAST control, in Figure 1 above.

Berns, Roy S., Ricardo J. Motta, and M.E. Gorzynski, "CRT Colorimetry: Part 1, *Theory and Practice*; Part 2, *Metrology*," in *Color Research and Application*, v. 18, 299–325 (1993).

It is alleged that the power function exponent γ of a CRT varies over a wide range; values as low as 1.4 and as high as 3.5 are cited in the literature. The graphs and captions in Figures 4 and 6 opposite show that wide variation in the apparent gamma value will result if the monitor's BLACK LEVEL (or "BRIGHTNESS") control is improperly adjusted.

Lightness

At a particular level of adaptation, human vision responds to about a hundred-to-one *contrast ratio* of luminance from white to black. Within this range, vision has a nonlinear response to luminance: Lightness perception is roughly logarithmic. A source having a luminance only 18% of a reference luminance appears about half as bright. The perceptual response to luminance is called *Lightness.* Vision researchers have modeled lightness sensitivity with various mathematical functions, as shown in Figure 2 below. The CIE has adopted a standard function L^* (pronounced "EL-star"), defined as a modified cube root:

Publication CIE No 15.2, *Colorimetry*, Second Edition. (Vienna: Central Bureau of the Commission Internationale de L'Éclairage, 1986)

$$L^* = \begin{cases} 903.3\dfrac{Y}{Y_n}; & \dfrac{Y}{Y_n} \le 0.008856 \\ 116\left(\dfrac{Y}{Y_n}\right)^{\frac{1}{3}} - 16; & 0.008856 < \dfrac{Y}{Y_n} \end{cases}$$

Y_n is the luminance of the white reference. A linear segment with a slope of 903.3 is applied near black. L^* has a range of 0 to 100. A unit change in L^* is taken to be approximately the threshold of visibility. In other words, you can detect a difference between intensities when the ratio between them is greater than about one percent.

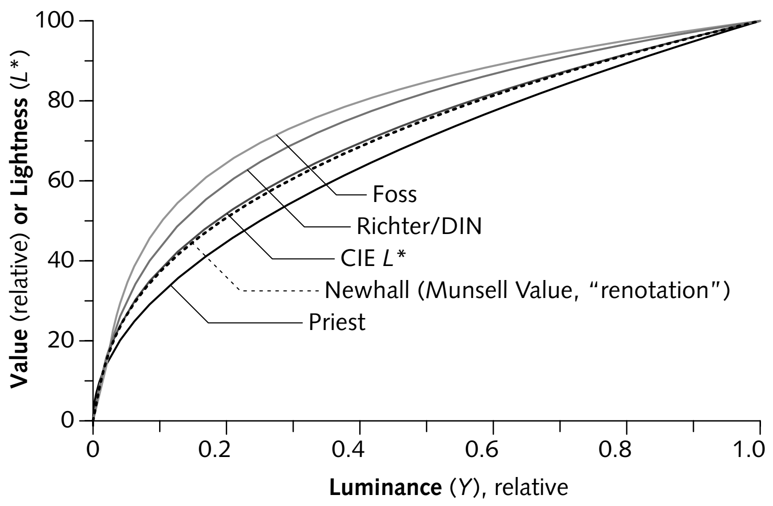


Figure 2 **Luminance and lightness.** The relationship between lightness-scale value V and luminance factor Y is plotted in accordance with different formulae. Redrawn from Fig. 2 (6.3) from Wyszecki and Stiles, *Color Science* (New York: Wiley, 1982).

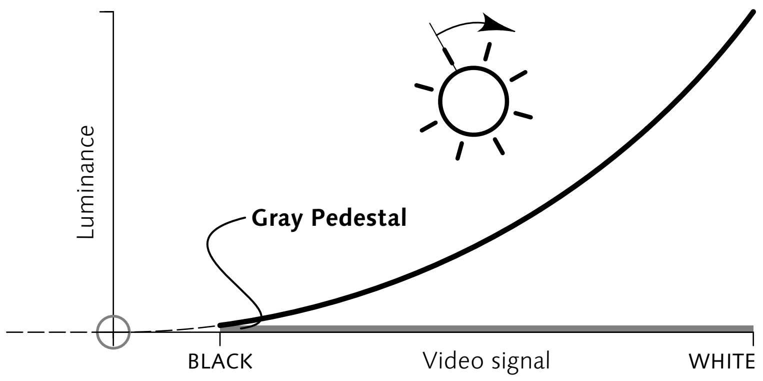


Figure 3 **Brightness control** has the effect of sliding the black-to-white video signal scale left and right along the 2.5-power function of the display. Here, BRIGHTNESS is set too high; a significant amount of luminance is produced at zero video signal level. No video signal can cause true black to be displayed, and the picture content rides on an overall pedestal of gray. Contrast ratio is degraded.

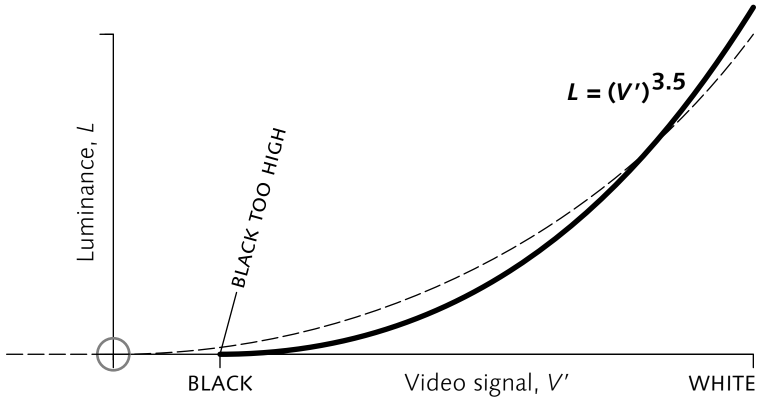


Figure 4 **Gamma 3.5** A naive approach to the measurement of CRT nonlinearity is to model the response as $L = (V')^{\gamma}$, and to find the exponent of the power function that is the best fit to the voltage-to-intensity transfer function of a particular CRT. However, if this measurement is undertaken with BRIGHTNESS set too high, an unrealistically large value of gamma results from the modelled curve being "pegged" at the origin.

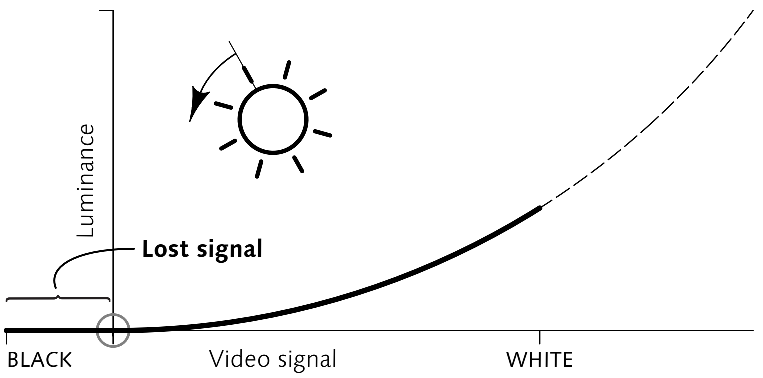


Figure 5 **Brightness control set too low** causes a range of input signal levels near black to be reproduced "crushed" or "swallowed," reproduced indistinguishably from black. A cinematographer might refer to this situation as "lack of details in the shadows," however, *all* information in the shadows is lost, not just the details.

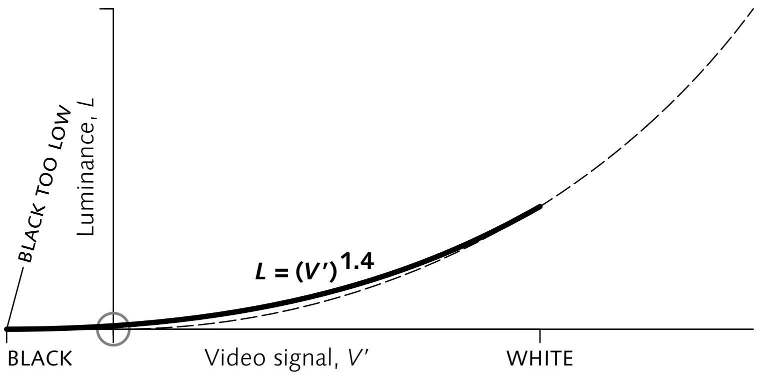


Figure 6 **Gamma 1.4** If the transfer function is modelled as $L = (V')^{\gamma}$ with *Black Level* set too low, an unrealistically small value of gamma results. However, if the transfer function is modeled with a function of the form $L = (V' + \varepsilon)^{2.5}$ that accommodates black level error, then a good fit is achieved. Misintepretations in the measurement of CRT nonlinearity have led to assertions about CRTs being highly unpredictable devices, and have led to image exchange standards employing quite unrealistic values of gamma.

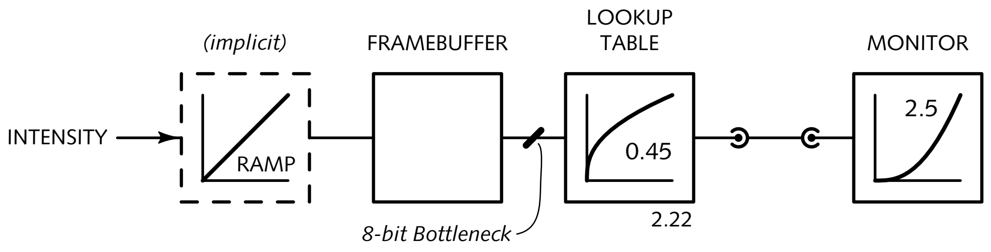


Figure 8 **Gamma in computer-generated imagery (CGI)**

Linear light coding

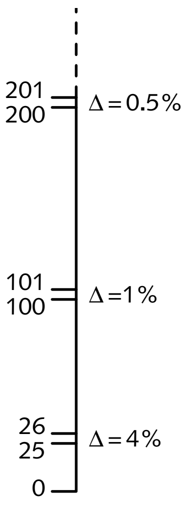


Figure 7 **Linear light coding: the code 100 problem**

Suppose that you wish to convey luminance values of an image through a channel having a few hundred or a few thousand discrete levels. Consider linear light coding, sketched in the margin, where code zero represents black. No matter what code is at the top end, code 100 represents a shade of gray that lies approximately at the perceptual threshold. For codes below 100, the ratio of intensities between adjacent code values is greater than 1 percent. At code 25, the ratio between adjacent codes is 4 percent. In smooth-shaded regions of an image, the luminance difference between adjacent code values, such as between code 25 and code 26, will cause visible *banding* or *contouring*. For codes above 100, the ratio of luminance values between adjacent codes is less than 1 percent: Code 201 is perceptually useless, and could be discarded without being noticed. In an 8-bit system, the highest code value – the brightest white – is at code 255. In an 8-bit linear-light system, the ratio between the brightest white and the darkest grey that can be reproduced without contouring is a mere 2.55:1.

To avoid perceptible steps at the black end of the scale, it is necessary to have coding that represents different luminance levels 1.00, 1.01, 1.02, and so on. If linear light coding is used, an absolute "delta" of 0.01 must be maintained all the way up the scale to white. To encompass the 100:1 luminance range vision of requires about 9900 codes, or about 14 bits for each of the *R, G,* and *B* components of the image.

Stokes, Mike, Mark D. Fairchild, and Roy S. Berns, "Precision requirements for digital color reproduction," in *ACM Transactions on Graphics*, v. 11, n. 4 (Oct. 1992), 406–422.

If you use nonlinear coding, then the 1.01 "delta" required at the black end of the scale applies as a ratio – not as an absolute increment – and progresses like compound interest up to white. This results in about 463 codes, or about nine bits per component. Eight bits, nonlinearly coded according to Rec. 709, is sufficient for broadcast-quality digital television at a contrast ratio of about 50:1.

In computer-generated imagery (CGI), linear-light coding is typically used in the frame buffer, as sketched in Figure 8 above. Often only 8 bits are provided in the framebuffer. When luminance data traverses the *8-bit Bottleneck* indicated in the sketch, serious contouring results.

Video coding

To code luminance into a small number of steps, say 256, then the codes should be assigned to intensities according to the properties of perception, in order for the most effective perceptual use to be made of

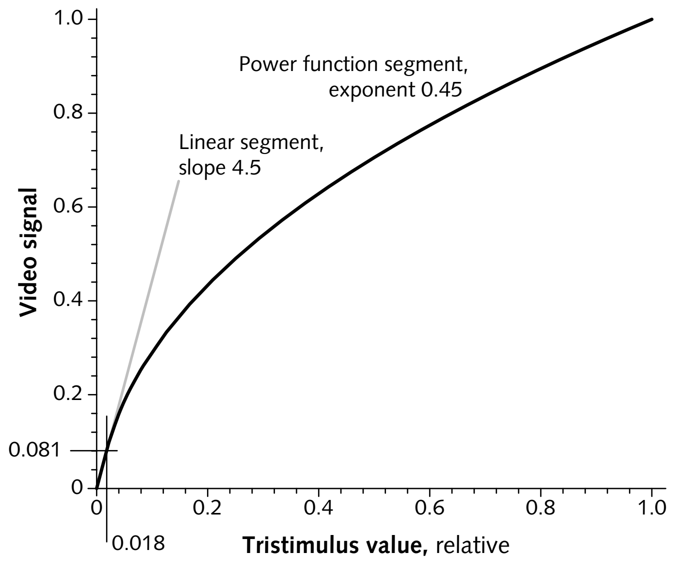


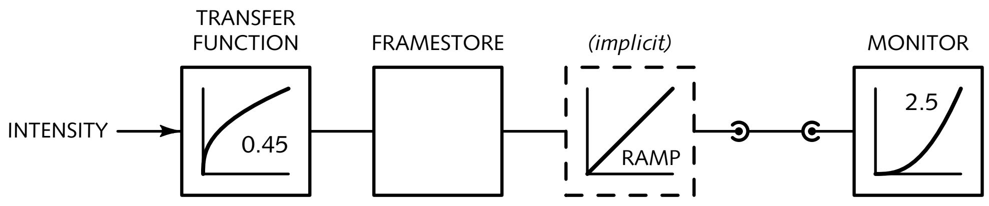
Figure 9 **The Rec. 709 transfer function of video** mimics the lightness sensitivity of vision. The standard is based on a power function with an exponent of 0.45. Theoretically, a pure power function suffices for gamma correction. In a practical system such as a television camera, the slope of the function is limited near zero in order to minimize noise in the dark regions of the picture.

the available codes. A transfer function similar to the lightness sensitivity of vision should be imposed at encoding.

A CRT's response is very nearly the inverse of the lightness sensitivity of vision: When image data is coded for perception at the encoder – for example, by the Rec. 709 transfer function graphed in Figure 9 above – the coding is inverted by the CRT, without the necessity to dedicate any circuitry to the task. The fact that a CRT's transfer function is very nearly the inverse of the lightness sensitivity of vision is an amazing, and fortunate, coincidence! The Rec. 709 transfer function standardized for 525/59.94 studio video, 625/50 studio video, and HDTV.

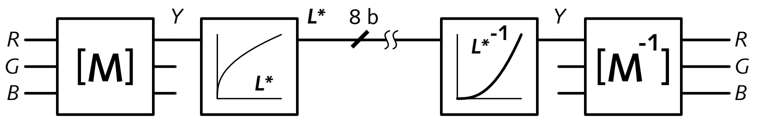
A summary sketch of gamma in video is shown in Figure 10 below. At the camera, luminance (or, in a color system, a set of three tristimulus values) is subjected to the Rec. 709 transfer function – or loosely, *gamma correction* – whose graph resembles the lightness sensitivity of vision. Video data is stored, processed, recorded, and transmitted in the perceptual domain. The monitor inverts the transform. The main purpose of gamma correction is to code luminance into a perceptually uniform domain, so as to obtain the best perceptual performance from a limited number of bits in each of *R′*, *G′*, and *B′*. (The prime symbols denote the nonlinearity.)

Figure 10 **Gamma in video**



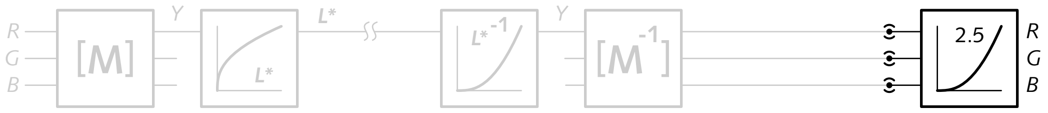
Rendering intent

Nonlinear encoding involves applying a transfer function similar to the lightness sensitivity of human vision. Ideally, luminance would first be *matrixed*, that is, formed as a weighted sum of linear-light (tristimulus) *RGB* signals. Then, the CIE *L** transfer function would be applied, to code the signal into a perceptually uniform domain. At the decoder, the inverse of the *L** function would restore luminance, then the inverse matrix would reconstruct *RGB*. The *L** signal would be accompanied by two other signals, to enable the representation of color:

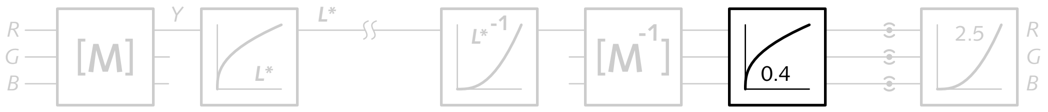


Coding *L** with eight bits achieves good image quality. Coding *Y* directly would require 11 bits or more to achieve similar quality.

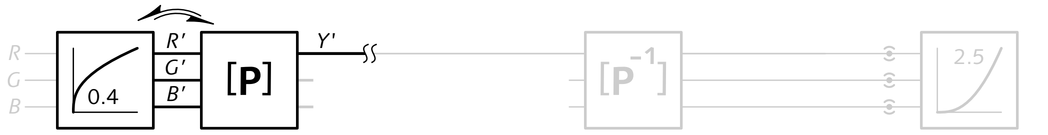
As I have outlined, the electron gun of a CRT monitor introduces a power function having an exponent of about 2.5:



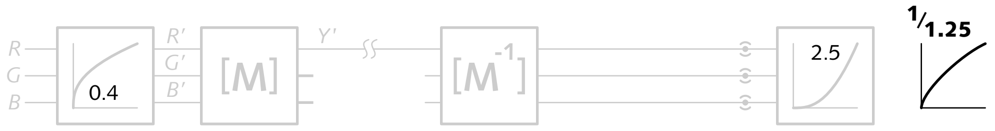
If we were to encode according to the *L** function, the decoder would have to invert that function, then impose the inverse of the 2.5-power function of the CRT:



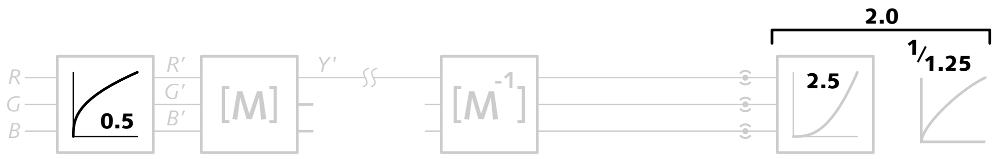
The CRT's power function is so similar to the inverse of the *L** function that we make an engineering compromise: Instead of encoding *Y* using the *L** transfer function, we encode *RGB* intensities to the inverse of the CRT's function. This allows us to dispense completely with transfer function circuitry at the display. We must then interchange the order of the matrix and the transfer function at the encoder. Changing the order of operations causes a departure from the *Principle of constant luminance*. In theory, the encoder would require a 0.4-power function:



This arrangement reproduces physical luminance correctly. However, it has a serious problem: The pictures do not look very good! When viewing a reproduced image, human viewers prefer a reproduction whose contrast ratio has been stretched slightly to a reproduction that is physically correct. The subjective preference depends somewhat upon the viewing environment. In effect, the visual system of the viewer imposes a power function with an exponent of about $^{1}/_{1.25}$:



For television, a power function with an exponent of about 1.25 must be applied to overcome this effect, in order to produce images that are subjectively pleasing. Rather than introducing circuitry at the display to apply this function, we modify the transfer function at encoder. We use an exponent of about 0.5, instead of the physically-correct 0.4:



If you think of encoding in physical terms, you could consider a video image to be encoded such that a 2.0-power function would reproduce physically-correct luminance at the display. (The NTSC video standard is often said to have gamma of 2.2, because the FCC standards described gamma in this way.) However, I think it is more evocative consider the application of the power function at the encoder to impose a *rendering intent* upon the image data. The encoding assumes the image is to be reproduced in a subjectively-acceptable manner through a physical 2.5-power function at the display.

Though ubiquitous in video, this subjective correction is rarely considered explicitly in computer graphics; belief in the "bits are bits" philosophy suggests to programmers that luminance should be reproduced in the physically-correct manner. However, subjective correction is as necessary in computer graphics as it is in video, and the Rec. 709 transfer function is appropriate for computer graphics. In traditional computer-generated imagery (CGI), as in Figure 8, the subjective correction is typically accomplished by "gamma correction" using a $^{1}/_{2.2}$-power function (instead of $^{1}/_{2.5}$). This is called "gamma of 2.2".

Poynton, Charles, *A Technical Introduction to Digital Video* (New York: Wiley, 1996).

The Rec. 709 transfer function is standard for 525/59.94 and 625/60 conventional video, and for HDTV. The Rec. 709 function is based on a power function exponent of 0.45, but the pure power function is modified by the insertion of a linear segment near black. The overall function is very similar to a square root. For details, consult the *Gamma* chapter in my book. In the diagrams in this section, I use the notation *0.5* as shorthand for the Rec. 709 function.

Rec. 709 appears to strictly define the transfer function at the camera. However, real video cameras have controls that can alter the transfer function. These controls are routinely used by cinematographers and videographers to achieve their artistic intents. Obviously the artistic intention of the cinematographer must be imposed at the camera, not at the display – it ought to be the *displays* that are standardized, not the cameras! But there is no mechanism to impose standards on displays, so we standardize the reference transfer function at the camera instead. In effect, Rec. 709 is standardized so as to produce acceptable reproduction on a conventional display. Despite the lack of standards, CRT displays are tacitly considered to have similar response.

The engineering of video systems – and, by extension, of desktop computer systems – involves an implicit assumption about the 2.5-power function of the monitor. Alternate display devices, such as LCDs, plasma panels, DMDs, and so on, do not have the 2.5-power function of the CRT. But the most important aspect of image coding is the establishment of a nearly perceptually-uniform image code. In a closed system employing an alternate display technology, you might be tempted to use of a transfer function at encoding that is the inverse of the transfer function at the display. However, if the transfer function of the display was very different than a 2.5-power function, more than 8 bits would be required to code luminance.

More significantly, there is a huge installed base of encoding and decoding equipment that assumes image coding similar or identical to that of video. The installed base includes roughly 1,300,000,000 television receivers, 400,000,000 VCRs, 250,000,000 camcorders, and 300,000,000 desktop computers. These devices are all, in effect, wired to directly reproduce *R'G'B'* signals encoded according to Rec. 709. Any proposal for a new encoder transfer function would compromise the interchange of images among these systems.

Poynton, Charles, "Luminance, luma, and the migration to DTV," presented at *32nd SMPTE Advanced Motion Imaging Conference*, Toronto (Feb. 6, 1998).

I have discussed the reproduction of black-and-white images. These concepts extend into the domain of luma and color difference coding, used in video, JPEG, and MPEG. At a SMPTE conference, I discussed the effect of transfer functions in luma and color difference coding.

Pseudocolor, hicolor, and truecolor

The block diagrams of pseudocolor, hicolor, and truecolor systems used in desktop computing are sketched in Figures , , and opposite. These sketches show the hardware pipeline from the framebuffer to the monitor. The interface from application software to the graphics subsystem (and window system) assumes the same processing. Comparable processing is implicit in file formats for pseudocolor and truecolor images. (File formats for hicolor are rare.)

A pseudocolor image is always accompanied by its color lookup table (CLUT). The CLUT may be optimized for the particular image, or it may contain a *system palette*. Upon display of a pseudocolor image, the graphics subsystem may directly load the colormap that accompanies the image. Alternatively, the graphics subsystem may recode the image

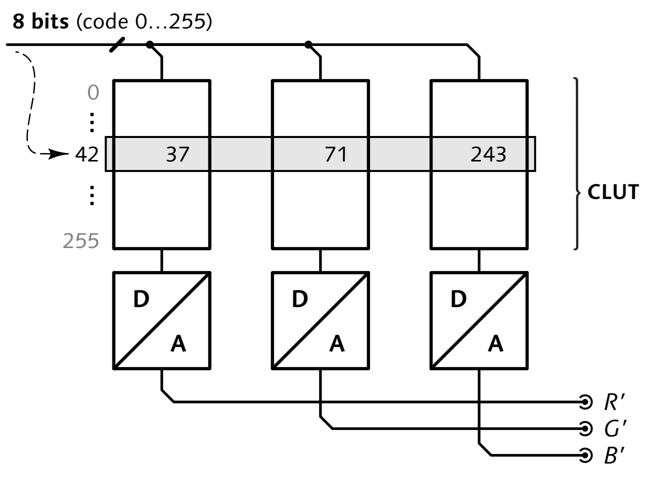


Figure 11 **Pseudocolor (8-bit)** graphics systems are common in low-end PCs. For each pixel, the framebuffer stores a color index value (typically 8 bits). Each index value is mapped, through a *color lookup table* (CLUT) that is part of the display hardware, to a triplet of *R′G′B′* codes. When a pixel is accessed from the framebuffer, the corresponding triplet is accessed from the CLUT; those values are applied to the digital-to-analog converter (DAC). *R′G′B′* codes from the CLUT translate linearly into voltage applied to the monitor, so code values are comparable to video *R′G′B′* values – the *R′G′B′* values are proportional to displayed intensity raised to the 0.4 power, comparable to video *R′G′B′* codes.

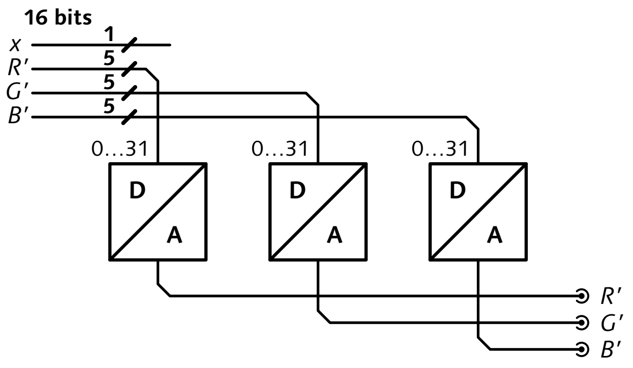


Figure 12 **Hicolor (16-bit)** graphics systems store, in the framebuffer, *R′G′B′* codes partitioned into three components of five bits each (5-5-5), or partitioned five bits for red, six bits for green, and five bits for blue (5-6-5). In low-end systems, these codes are applied directly to the DACs with no intervening colormap. Because the *R′G′B′* codes are translated linearly into monitor voltage, the code values are implicitly proportional to displayed intensity raised to the 0.4 power.

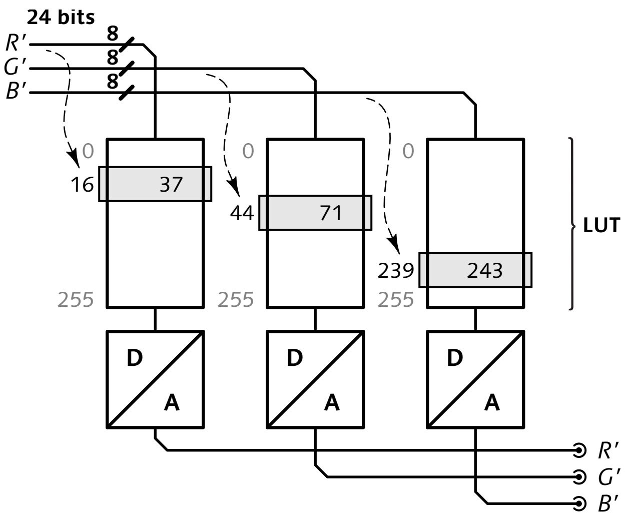


Figure 13 **Truecolor (24-bit)** graphics systems store 8 bits for each of the red, green, and blue components. Truecolor systems usually implement a set of *lookup tables* (LUTs) between the framebuffer memory and the DACs. The LUTs allow a transfer function to be imposed between the *R′G′B′* codes and the DACs: The *R′G′B′* values in the framebuffer need not be related to displayed intensity raised to the 0.4 power. Application software or system software can impose arbitrary functions. In order for the application software to provide the same default behavior as low-end pseudocolor and hicolor graphics systems, each LUT is set by default to a *ramp* corresponding to the identity (or *unity*) function.

according to some other map that is already loaded into the hardware, or according to the system palette native to the application. Recoding of pseudocolor image data may introduce color errors. Pseudocolor image data is always coded in terms of monitor *R'G'B'* – that is, pseudocolor image colors are implicitly perceptually coded.

A hicolor system has no lookup tables. Image data is coded in terms of monitor *R'G'B'*: Image data is implicitly perceptually coded (though coarsely quantized).

In truecolor, each of the *RGB* channels is associated with a lookup table (LUT) that applies a transfer function. (Ordinarily, the three tables have identical contents.) Different default lookup tables are in use, for different platforms. Truecolor image files are ordinarily stored without any lookup tables; most truecolor file formats make little or no provision for conveying the transfer function that is expected at display.

Hicolor and truecolor display hardware can typically be operated in pseudocolor mode. But this mode switch applies to the whole display. If a pseudocolor image is to be displayed in a window of a display that is operating in hicolor or truecolor mode, the graphics subsystem must perform the pseudocolor color lookup operation in software. If the truecolor system is operating with a LUT that is not a ramp, then the *RGB* codes from the pseudocolor CLUT must be mapped through the *inverse* of the truecolor LUT prior to being stored in the framebuffer.

If a hicolor or truecolor image is to be presented on a display that is operating in pseudocolor mode, the graphics subsystem must find, for each hicolor or truecolor pixel (*RGB* triplet), the index of the closest color that is available in the CLUT currently in use. If the CLUT is organized systematically, then this operation can be fairly rapid; if the CLUT is unstructured, then the conversion proceeds slowly. The translation to pseudocolor causes coarse quantization of the image colors. Dithering may be applied, to spread the quantization error over a small area of the image. In any event, colormap quantization generally causes serious degradation of color fidelity.

A generic application program on a PC must assume the lowest common denominator of display capability: It must be prepared to operate without a lookup table. (Even some graphics cards with 24-bit capability have no lookup tables.) Even if a hardware lookup table is present, PC software generally operates as if there is no table. If a LUT is present, it is ordinarily loaded with a ramp so that it has no effect. Image data exchanged among PCs is therefore coded as monitor *R'G'B'*. Though the situation arose by accident, this is quite comparable to video coding, and is nearly optimal for perception! Image data that originates on (or is intended for display on) a PC carries the implicit assumption that the lookup table contains a ramp, that is, that the image data is represented in gamma-corrected monitor *R'G'B'*. So Figure 10 applies to video *and* to PCs: the coding is comparable.

Macintosh gamma

Apple Computer, Inc., *Inside Macintosh* (Reading, Mass.: Addison-Wesley-Longman, 1992). 27 volumes.

Contrary to popular belief, Macintosh computers use monitors that have the same physics as monitors used in video systems and other brands of computers. Though it is nowhere documented in the 27 volumes of the *Inside Macintosh* series of books, the QuickDraw graphics subsystem loads an unusual transfer function into the lookup tables of a Mac. The default lookup table (in hexadecimal code) is this:

Figure 14 **Default Macintosh LUT**

```
00: 00, 05, 09, 0B, 0E, 10, 13, 15, 17, 19, 1B, 1D, 1E, 20, 22, 24
10: 25, 27, 28, 2A, 2C, 2D, 2F, 30, 31, 33, 34, 36, 37, 38, 3A, 3B
20: 3C, 3E, 3F, 40, 42, 43, 44, 45, 47, 48, 49, 4A, 4B, 4D, 4E, 4F
30: 50, 51, 52, 54, 55, 56, 57, 58, 59, 5A, 5B, 5C, 5E, 5F, 60, 61
40: 62, 63, 64, 65, 66, 67, 68, 69, 6A, 6B, 6C, 6D, 6E, 6F, 70, 71
50: 72, 73, 74, 75, 76, 77, 78, 79, 7A, 7B, 7C, 7D, 7E, 7F, 80, 81
60: 81, 82, 83, 84, 85, 86, 87, 88, 89, 8A, 8B, 8C, 8C, 8D, 8E, 8F
70: 90, 91, 92, 93, 94, 95, 95, 96, 97, 98, 99, 9A, 9B, 9B, 9C, 9D
80: 9E, 9F, A0, A1, A1, A2, A3, A4, A5, A6, A6, A7, A8, A9, AA, AB
90: AB, AC, AD, AE, AF, B0, B0, B1, B2, B3, B4, B4, B5, B6, B7, B8
A0: B8, B9, BA, BB, BC, BC, BD, BE, BF, C0, C0, C1, C2, C3, C3, C4
B0: C5, C6, C7, C7, C8, C9, CA, CA, CB, CC, CD, CD, CE, CF, D0, D0
C0: D1, D2, D3, D3, D4, D5, D6, D6, D7, D8, D9, D9, DA, DB, DC, DC
D0: DD, DE, DF, DF, E0, E1, E1, E2, E3, E4, E4, E5, E6, E7, E7, E8
E0: E9, E9, EA, EB, EC, EC, ED, EE, EE, EF, F0, F1, F1, F2, F3, F3
F0: F4, F5, F5, F6, F7, F8, F8, F9, FA, FA, FB, FC, FC, FD, FE, FF
```

This table contains a pure power function with an exponent of $^1/_{1.45}$. Image data that originates on – or is intended for display on – a Macintosh computer carries the implicit assumption that the lookup table contains this function. This default lookup table, in combination with a conventional monitor, causes the *R*, *G*, and *B* values presented to QuickDraw to represent the $^1/_{1.8}$-power of luminance.

Although Apple has historically failed to publish any meaningful documentation of gamma, a Macintosh is widely considered to have a default gamma of 1.8. This *de facto* nomenclature was established by the *Gamma* control panel, by Knoll Software, which was distributed with Adobe Photoshop up to and including version 4.

Figure 15 below summarizes the gamma situation for Macintosh.

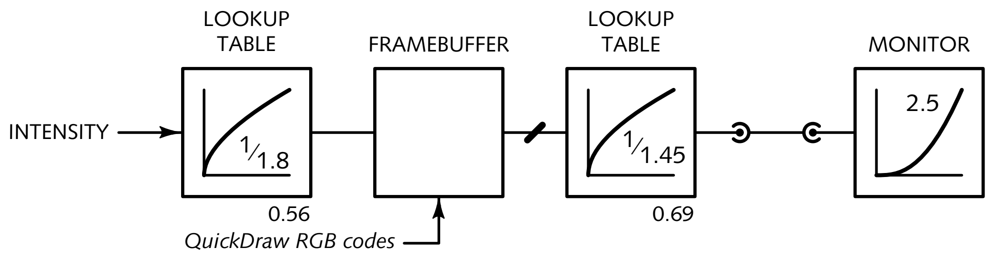


Figure 15 **Gamma in Macintosh**

Poynton, Charles, *A Technical Introduction to Digital Video* (New York: Wiley,1996).

In the *Gamma* chapter of my book, I explain the *dot gain* phenomenon of offset printing. Offset printing uses code values proportional to the 1.8-power of reflectance. But QuickDraw *RGB* codes are related to

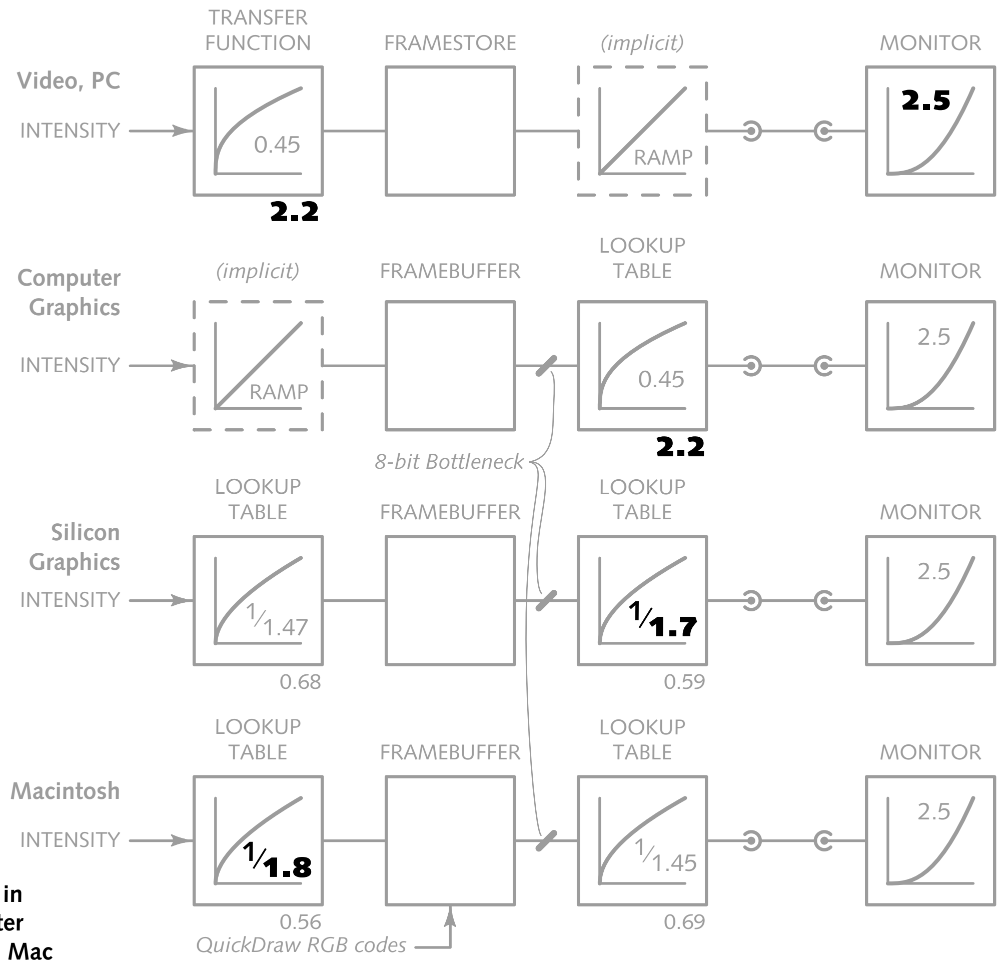


Figure 16 **Gamma in video, PC, computer graphics, SGI, and Mac**

luminance by the 1.8 power! Though the situation arose by accident, QuickDraw *RGB* coding is well suited to offset printing, and it is ubiquitous in desktop publishing and prepress. (QuickDraw coding is also widely used in multimedia, and on the World-wide Web, though in these applications it is not as suitable as coding according to Rec. 709.)

System issues

Figure 16 above collects the three gamma sketches already presented (video, computer-generated imagery, and Macintosh), and adds a fourth sketch, for Silicon Graphics (SGI). Given the diverse transfer functions, it is no surprise that it is difficult to exchange image data. I have indicated in bold type the numerical quantity that is referred to as *gamma* in each of these four domains: You can see that the *gamma* number is applied in four different places! So even if you know the gamma value, it is difficult to determine where it is applied! As I have mentioned, a Macintosh is considered to have a default gamma of 1.8. An SGI computer has a default gamma of 1.7. But Figure 16 shows that these two numbers are not comparable!

The graphics subsystems of most computers allow the lookup table to be changed. On a Mac, the *Gamma* control panel accomplishes this. When a *gamma* value of *g* is specified, the control panel loads the Mac lookup table with a power function whose exponent is $^{g}/_{2.61}$. When a *gamma* value of *g* is specified to an SGI computer, system software loads the SGI lookup table with a power function whose exponent is $1/g$. The convention differs from that on a Mac. To program an SGI computer to behave like a Mac, you must set SGI gamma to 1.45.

Image data in the framebuffer is not usually changed upon a change of the lookup table. Any time you jam a particular value of *gamma* into the back end of the graphics subsystem, you override assumptions that may have been made about the color interpretation of image data. When you change gamma, the colors of displayed objects (icons, menus, and windows) and the colors of displayed images, will change!

Computer graphics standards

To exchange images using computer graphics standards requires knowledge of the transfer function. Standards such as PHIGS and CGM stem from computer-generated imagery (CGI), where linear-light (tristimulus) coding is the norm, and in PHIGS and CGM it is implicit that *RGB* data is coded in linear-light (tristimulus).

JPEG and other lossy image compression algorithms depend on discarding information that won't be *perceived*. It is vital that the data presented to a JPEG compressor be coded in a perceptually-uniform manner, so that the discarded information has minimal perceptual impact. In practice, JPEG works well only on nonlinearly-coded (gamma-corrected) image data.

But nowhere in the PHIGS, CGM, or JPEG standards is gamma or transfer function explicitly mentioned, and nowhere in the data streams or image file formats for PHIGS, CGM, or JPEG, is the transfer function conveyed! The user must handle the transfer function or face poor image quality. If image data is transferred between these systems without regard for the transfer function, then the pictures will have terrible quality. Figure 17 below summarizes the situation: *RGB* codes [128, 128, 128] produce completely different intensities at the face of the screen in PHIGS (or CGM) and JPEG. But the standards themselves provide absolutely no information concerning this issue.

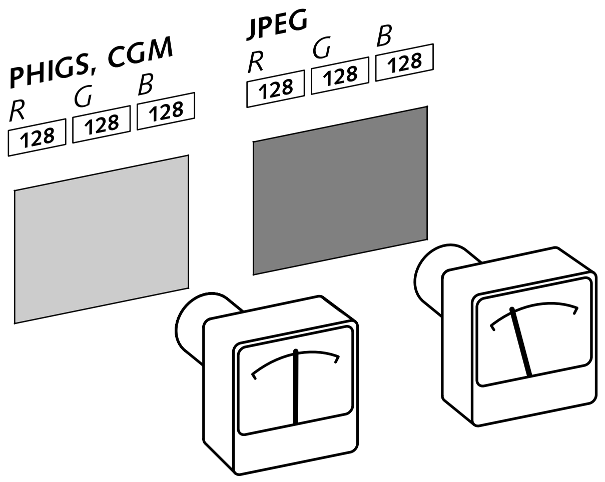
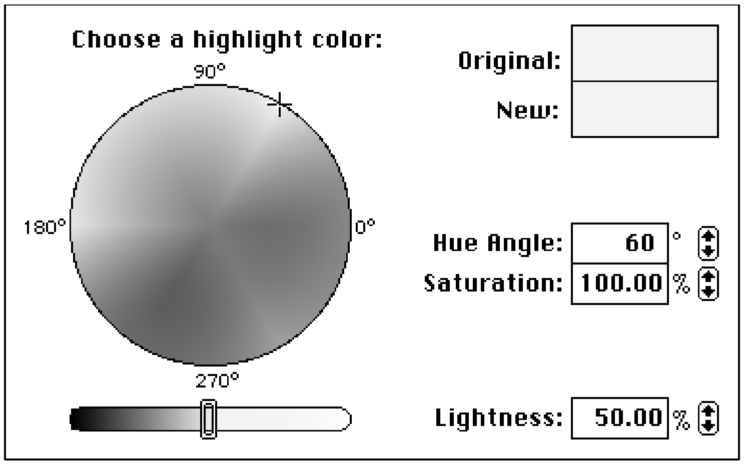


Figure 17 **Gamma in PHIGS, CGM, and JPEG.** In the PHIGS and CGM standards there is no mention of gamma or transfer function, but it is implicit that image data is coded as linear-light tristimulus values. In the JPEG standard there is no mention of gamma or transfer function, but it is implicit that image data is gamma-corrected.



Hue Angle: 60 °
Saturation: 100.00 %
Lightness: 50.00 %

Figure 18 **Lightness in HSL.** *Lightness*, in its CIE definition, is a perceptual quantity. However, in the textbook *HSL* color representation used in computer graphics – here exemplified by Apple's Macintosh Color Picker – no account is taken of transfer function. Apple implies that all of the shades in the disk have the same lightness of 50%. Does the disk appear uniformly shaded to you?

Many other computer graphics standards ignore or discount transfer functions. Figure 18 above shows a screenshot of the Apple Macintosh Color Picker, which implements the textbook *HSL* representation of color. This presentation implies that all of the colors shown share the same lightness value, but clearly the disk is not uniformly shaded. The *HSL* representation has no objective basis in color science.

World-wide web

The World-wide web uses GIF and JPEG file formats to convey images. (Other file formats are in use, but none of these are widely deployed.)

A GIF file represents an image in pseudocolor form. A web browser operating on a pseudocolor display does not attempt to reconcile the potentially conflicting CLUTs found among the several (or several dozen) GIF images that might share a window on the user's monitor. Instead, a browser typically recodes every pseudocolor image into a *browser palette* comprising a 6×6×6 *colorcube* of monitor *R'G'B'* codes. The browser palette comprises 216 colors. (To display GIF images on a hicolor or truecolor system, the browser's graphics subsystem uses each file's CLUT to translate each image to *R'G'B'*.)

JPEG image coding is based on truecolor. The JFIF specification is the *de facto* standard for JPEG file interchange. JFIF is unclear concerning the handling of transfer function. In practice, an image is encoded into JPEG using the encoding transfer function that is in effect for the platform that it is encoded on. A decoded JPEG image is displayed using the transfer function in effect on the platform upon which it is decoded. Figure 19 overleaf sketches the gamma situation for JPEG on the web. It's chaos! Image data is exchanged among platforms without regard for the transfer function that will be applied upon display. The same file displays differently on different platforms!

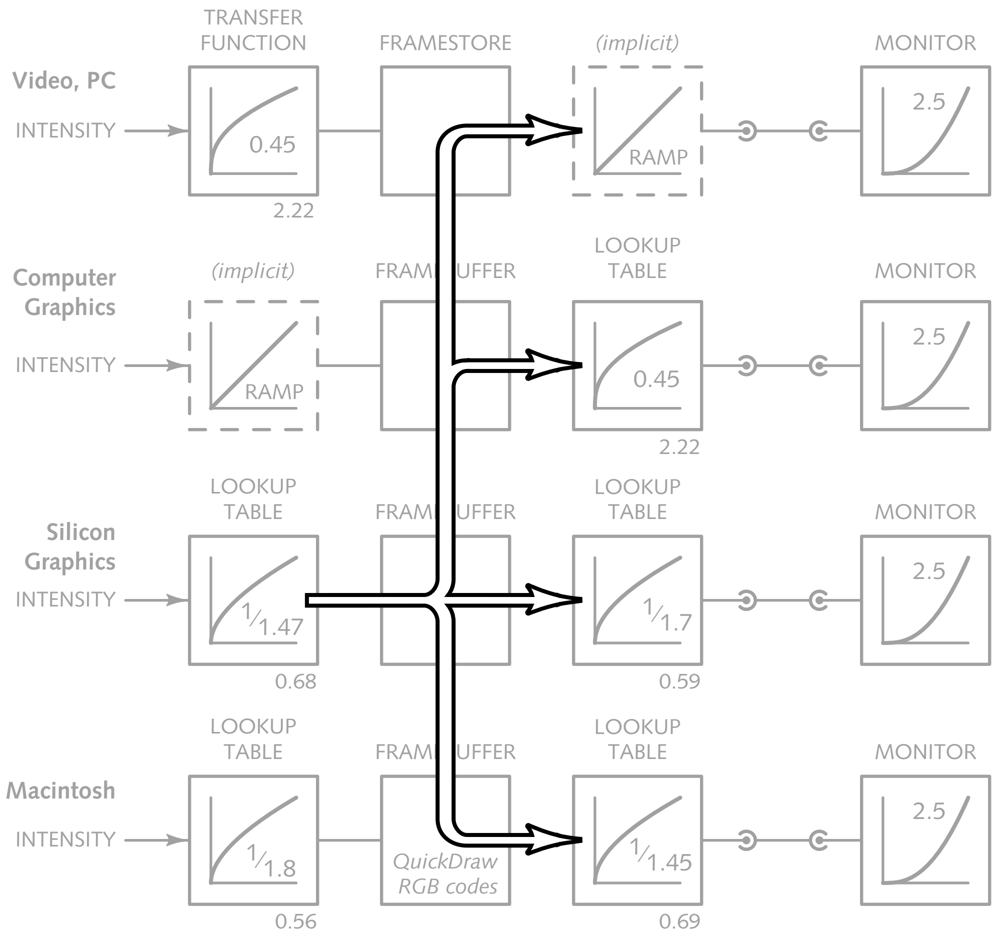


Figure 19 **Gamma on the web.** Image data is exchanged on the web without regard for the transfer function that will be applied at display. Consequently, the same image is displayed differently on different platforms. Here, the image originates on an SGI computer. In practice, the dominant platforms are PC and Mac.

Gille, J.L., J.O. Larimer, and J. Luszcz, "Error diffusion using the 'web-safe' colors: how good is it across platforms?", in Rogowitz, B.E. and T.N. Pappas (eds.), *Human Vision and Electronic Imaging III*, Proceedings of SPIE, Volume 3299, 368–375 (Bellingham, Wash.: SPIE, 1998).

In Figure 19 above, I show an image originating on an SGI computer. You can see that it is decoded in four different ways; in particular, it is reproduced incorrectly on PC and Mac, the dominant viewing platforms. If a JPEG image originates on a PC, it displays incorrectly on a Mac; if a JPEG image originates on a Mac, it displays incorrectly on a PC. The relatively low penetration of Macintosh computers in the marketplace might suggest that image origination should be optimized for PCs: let the Mac browsers do what they will. But the tools for image preparation and web page creation are much more capable on Macs than on PCs. A large fraction of web images are prepared on Macs, so the implicit Mac transfer function is important.

The second row of the sketch shows the CGI situation. In practice, images are rarely displayed through a LUT configured for CGI. But the VRML language stems from CGI, and shading in synthetic computer graphics has historically been performed in the linear-light domain. (Shading in nonlinear domains is poorly understood.) VRML is best considered to originate in the computer graphics row of the sketch.

Color management

Work is underway to implement facilities in graphics systems to allow device-independent specification of color. Users and applications will be able to specify colors, based on the CIE standards, without concern for gamma correction. Color image files will be tagged with their transfer functions (along with other color parameters). When this transition is complete, it will be much easier to obtain color matching across different graphics libraries and different hardware.

However, these developments will not render gamma irrelevant. Proper use of transfer functions will remain necessary in order to code images in a perceptual manner, so as to achieve maximum performance from a reasonable number of bits per component – 8, say! Also, it will take a long time for this technology to be deployed.

In the meantime, if you are a programmer, you can take the following steps:

- Use gamma-corrected *R'G'B'* representations whenever you can. An image coded with gamma correction has good perceptual uniformity, resulting in an image with much higher quality than one coded as 8-bit luminance (or tristimulus) values.
- When you exchange images either in truecolor or pseudocolor form, code *R'G'B'* color values using the Rec. 709 transfer function.
- In the absence of reliable information about your monitor, display pictures assuming a monitor gamma value of 2.5.

If you are a user, you can take these steps:

- Establish good viewing conditions. If you are using a CRT display, you will get better image quality if your overall ambient illumination is reduced.
- Ensure that your monitor's BLACK LEVEL (or BRIGHTNESS) control is set to correctly reproduce black elements on the screen. Consult my note *"BRIGHTNESS" and "CONTRAST" controls*, available on the Internet.
- Demand, from your hardware and software developers and vendors, that they document how they handle transfer functions.