CHAPTER 30

Color Phenomena

The light receptors in cameras and in the eye respond more or less strongly to different wavelengths of light. Most cameras and most eyes have several different types of receptor, whose sensitivity to different wavelengths varies. Comparing the response of several types of sensor yields information about the distribution of energy with wavelength for the incoming light; this is color information. The color of an object seen in an image depends on how the object was lit, but there are algorithms that can correct for this effect.

30.1 COLOR PHYSICS

The human visual system responds to light in a range of wavelengths from approximately 400nm to approximately 700nm, making these wavelengths of particular importance. Light may have more or less energy at different wavelengths. If the intensity is relatively uniform across the wavelengths, the light will look white, but if it isn't, the light may look colored. The distribution of energy with wavelength is sometimes called the *spectral energy density*. Light containing energy at just one wavelength looks deeply colored (these colors are known as *spectral colors*). The colors seen at different wavelengths have a set of conventional names, which originate with Isaac Newton (the sequence from 700nm to 400nm goes Red Orange Yellow Green Blue Indigo Violet, or **R**ichard **o**f **Y**ork **g**ot **b**listers **i**n **V**enice, although indigo is now frowned upon as a name because people typically cannot distinguish indigo from blue or violet).

30.1.1 The Color of Light

Different light sources have different relative spectral energy densities. Figure 30.1 shows examples from both measurements and from standard models of daylight. Clear air scatters sunlight out of its direction of travel, with a probability of scattering that depends on the fourth power of the wavelength. This means that light of a long wavelength can travel much farther before being scattered than light of a short wavelength. When the sun is high in the sky, blue light is scattered out of the ray from the sun to the earth—meaning that the sun looks yellow—and can scatter from the sky into the eye—meaning that the sky is quite bright, and looks blue.

Typical artificial light sources are commonly of a small number of types:

- An *incandescent light* contains a metal filament that is heated to a high temperature. Incandescent lights tend to produce energy over a wide range of wavelengths, but tend not to produce much light in the short wavelengths (i.e. blue – incandescent lights tend to look yellow).
- A *fluorescent light* works by generating high-speed electrons that strike gas

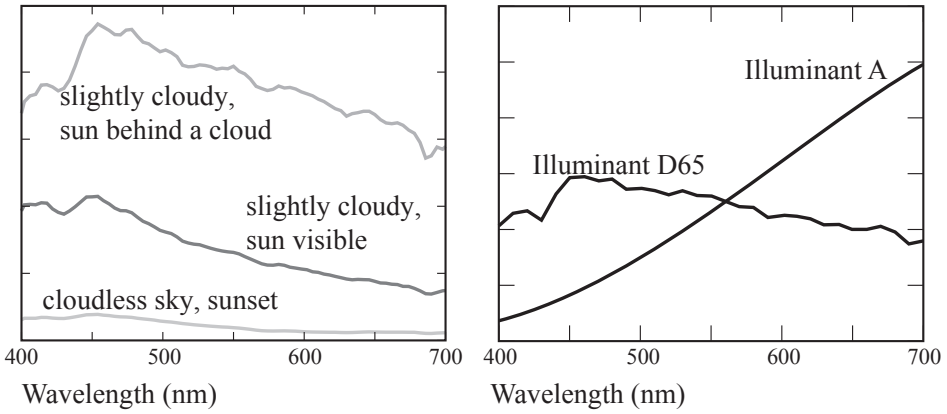


FIGURE 30.1: **Left** *shows relative spectral energy density of daylight measured under different conditions. Measurements were made by Jussi Parkkinen and Pertti Silfsten; the data is published at* `https://sites.uef.fi/spectral/databases-software/daylight-spectra/`. **Right** *shows relative spectral energy density of standard illuminant models, standardized by the CIE. The models are illuminant A—which models the light from a 100W Tungsten filament light bulb, with color temperature 2800K—and illuminant D-65—which models daylight indoors. The data is published at* `https://cie.co.at/data-tables`. *In each case, I have omitted y-labels because the point of the figure is the* relative *value of the spectral energy density.*

within the bulb. The gas releases ultraviolet radiation, which causes phosphors coating the inside of the bulb to fluoresce. The spectral energy density of fluorescent lights tends to be "spikey" with energy concentrated in a small range of wavelengths.

- An *LED*color sources (light emitting diode) produces light from physical effects in semiconductors. These effects tend to produce energy in a very narrow range of wavelengths. Lights built out of LEDs tend to use a number of different types of LED, and often use phosphors as well. LED lights are now extremely common, because they can be very bright and are extremely efficient.

The *black-body* model is often used to describe illuminant colors. This model, derived from physical arguments about an idealized heated body, describes a one-parameter family of illuminants. Write T for the temperature of the body in Kelvins, h for Planck's constant, k for Boltzmann's constant, c for the speed of light, and λ for the wavelength. Then the model gives the spectral energy density as

$$E(\lambda; T) = C\frac{\exp(-hc/k\lambda T)}{\lambda^5}$$

where C is some constant of proportionality which changes the intensity of the source. The parameter T is often referred to as *color temperature*. A black-body

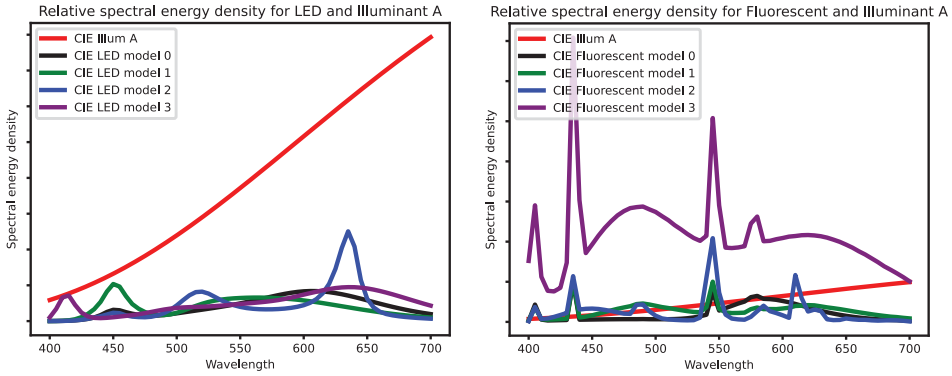


FIGURE 30.2: *The relative spectral energy distribution for some example LED lights (***left***) and fluorescent lights (***right***), compared with CIE Illuminant A, which is a model of an incandescent light. These are CIE models, so idealize the behavior of these lights. Note the bright, narrow bands that come from LED physics or flouresing phosphors. Some lights have quite uniform distribution of energy; others have quite a "spikey" distribution. The data is published at* `https://cie.co.at/data-tables`. *In each case, I have omitted y-labels because the point of the figure is the* relative *value of the spectral energy density.*

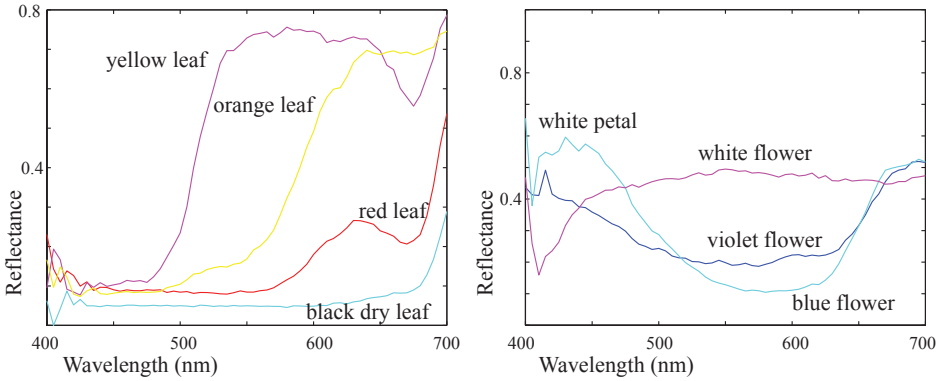


FIGURE 30.3: **Albedos** *Spectral albedoes for a variety of natural surfaces measured by Esa Koivisto, Department of Physics, University of Kuopio, Finland, plotted against wavelength in nanometers.* These figures were plotted from data available at `https://sites.uef.fi/spectral/databases-software/natural-colors/`.

source ranges from somewhat orange in appearance (for T around 1500-3000 Kelvin) through yellow (around 3000-4500 Kelvin) to neutral (around 6000 Kelvin) then drifting into the light blue (around 10000 and more Kelvin). You should not think of the color temperature as the actual temperature of an actual source (it's not easy to handle objects at 6000K!); it is just a parameter. Black body models are quite

good at describing illumination outdoors, because they can match both sunlight and skylight fairly well (exercises).

30.1.2 The Color of Surfaces under White Light

Assume light with a uniform distribution of energy across wavelengths falls on a surface. For almost all applications of computer vision, energy does not arrive at one wavelength and leave at a different wavelength, and reflections are either specular or diffuse. As a result, surfaces can be described by the fraction of light reflected at each wavelength for each type of reflection. Specular reflection is relatively easily dealt with. If the surface is a conductor, the specularly reflected light may depend quite strongly on wavelength, so that white light may result in colored specularities, though this point does not usually appear in applications. If the surface is not a conductor, specularly reflected light tends to take the color of the light source.

Diffuse reflection is more interesting. There are a large number of mechanisms that cause different fractions of different wavelengths to be reflected. The result is that diffuse albedo is wavelength dependent. The wavelength-dependent diffuse albedo is sometimes referred to as the *spectral reflectance* (sometimes abbreviated to *reflectance* or, less commonly, *spectral albedo*). Figure 30.3 shows examples of spectral reflectances for a number of different natural objects. Each of these surfaces will look colored even if they are illuminated by white light.

FIGURE 30.4: *The same surface will appear to have different colors when viewed under different lights. The effects can be quite pronounced. These images, taken from the LSMI dataset disseminated at* `https://github.com/DY112/LSMI-dataset`. *On the* **far left**, *two images of the same scene under different illuminations. On the* **left**, *details cut from these images to show the color checker. Next to the top detail (on the* **right**), *the bottom detail scaled to match in intensity. This shows that the difference is not just in brightness; colors have shifted as well. On the* **far right**, *another pair of scenes viewed under different lights, showing further color shifts.*

30.1.3 The Color of Surfaces in Images

Specularities are mostly ignored in the applications of computer vision, and here I focus on diffuse surfaces. Most of the time, surfaces are lit by colored light, so the color of the light leaving the surface is affected both by the color of the light falling on it, and by the color of the surface. This is a significant nuisance in computer vision, not least because it is somewhat inconsistent with experience. The effects are pronounced, and easily observed (Figure 30.4). Humans are surprisingly good at ignoring the effects of colored light, and typically report the color a surface *would have* under white light, rather than the color of the light reflected from the surface. If we use the Lambertian plus specular model, we have

$$E(\lambda) = \rho_{dh}(\lambda)S(\lambda) \times \text{geometric terms} + \text{specular term}$$

30.2 IMAGING COLOR

Different kinds of color receptor in the human eye respond more or less strongly to light at different wavelengths, producing a signal that is interpreted as color by the human vision system. The precise interpretation of a particular light is a complex function of context; illumination, memory, object identity, and emotion can all play a part.

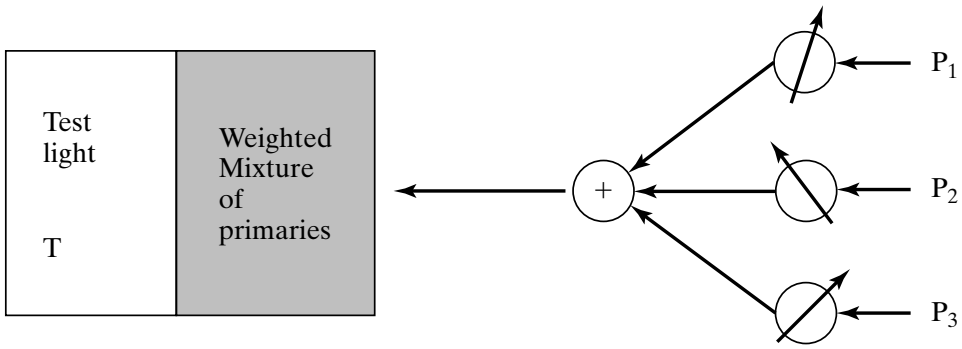


FIGURE 30.5: *Human perception of color can be studied by asking observers to mix colored lights to match a test light shown in a split field. The drawing shows the outline of such an experiment. The observer sees a test light T and can adjust the amount of each of three primaries in a mixture displayed next to the test light. The observer is asked to adjust the amounts so that the mixture looks the same as the test light. The mixture of primaries can be written as $w_1P_1 + w_2P_2 + w_3P_3$; if the mixture matches the test light, then we write $T = w_1P_1 + w_2P_2 + w_3P_3$. It is a remarkable fact that for most people three primaries are sufficient to achieve a match for many colors, and three primaries are sufficient for all colors if we allow subtractive matching (i.e., some amount of some of the primaries is mixed with the test light to achieve a match). Some people require fewer primaries. Furthermore, most people choose the same mixture weights to match a given test light.*

30.2.1 Color Matching in Humans

In a simple but informative experiment, a subject is shown a colored light—the *test light*—in one half of a split field (Figure 30.5). The subject can then adjust a mixture of lights in the other half to get it to match. The adjustments involve changing the intensity of some fixed number of *primaries* in the mixture.

Write T for the test light, an equals sign for a match, the weights—which are non-negative—as w_i, and the primaries P_i. A match can then be written in an algebraic form as

$$T = w_1 P_1 + w_2 P_2 + \ldots,$$

meaning that test light T matches the particular mixture of primaries given by $(w_1, w_2, \ldots)$. In *subtractive matching*, the subject can add some amount of some primaries to the *test light* instead of to the match. This can be written in algebraic form by allowing the weights in the expression above to be negative.

If subtractive matching is allowed and the primaries are independent (no mixture of two primaries matches a third), then most subjects require only three primaries to match a test light. This phenomenon is known as the principle of *trichromacy*. Given the same primaries and test light, most subjects select the *same* mixture of primaries to match that test light.Matching is (to an accurate approximation) linear. This yields *Grassman's laws*. First, if we mix two test lights, then mixing the matches will match the result. Second, if two test lights can be matched with the same set of weights, then they will match each other. Finally, matching is linear: a test light with doubled intensity is matched by doubling the weights. Trichromacy, the fact that different subjects select the same mixture to match a test light, and Grassman's laws are about as true as any law covering biological systems can be. The main exceptions involve very dark or very bright lights, and subjects suffering from genetic ill fortune, neural problems, or the effects of aging.

30.2.2 Sensing Color

Color imagers – eyes or cameras – are arrays of sensors. In humans, the sensors are cells that are sensitive to light. In cameras, the sensors are now semiconductor devices that turn light into charge. These sensors do not measure wavelength directly, but instead collect different fractions of the power arriving at different wavelengths, and sum these fractions to produce their output. The fraction of power arriving at a particular wavelength that is collected by the sensor is known as the *spectral sensitivity* or *spectral response* of the sensor. Most cameras and most eyes have several different types of sensor with different spectral sensitivities. Comparing the response of several types of sensor yields color information. Write $\sigma_k(\lambda)$ for the spectral sensitivity of the k'th type of sensor. The energy collected by the sensor can then be written

$$E_k(\mathbf{x}) = \int_{\text{wavelength}} P(\mathbf{X} \to \mathbf{x}, \lambda) \sigma_k(\lambda) d\lambda \Delta_t,$$

and the model for camera intensity becomes

$$I_{\text{camera},k}(\mathbf{x}) = C_k(E_k(\mathbf{x}))$$

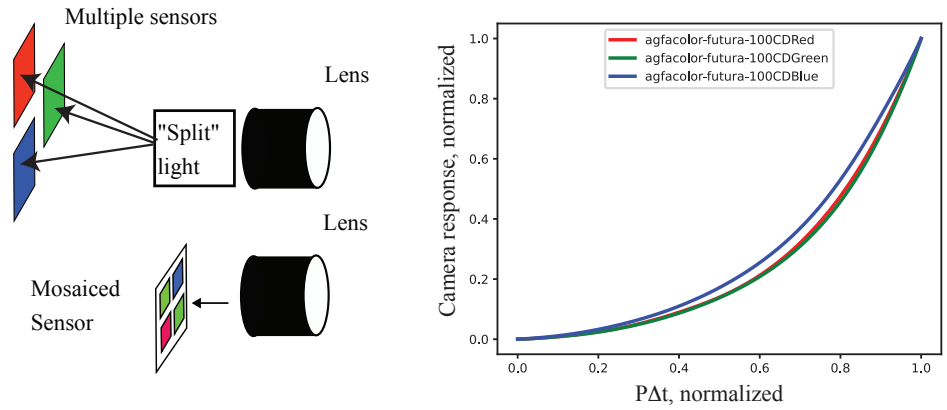


FIGURE 30.6: *There are two main ways to build color cameras. One can arrange that light leaving the lens is split into three paths, each of which leads to a different sensor (***top left***). These sensors have with different spectral sensitivities. Alternatively, one can build a sensor where each location has a spectral sensitivity chosen from a set of types (typically, three types;* **bottom left***). These types are typically arranged in one of a set of patterns (shown in 30.7), and a demosaicing algorithm reconstructs the complete set of responses for each type. The sensor measurements are then passed through camera response functions (as in Section 28.3.3). On the* **right***, example camera response functions for red, green and blue sensors. Note these are similar, but not exactly the same.*

where C_k is the camera response function for the k'th type of sensor. While camera response functions for different types of sensor tend to be quite similar, they often differ slightly.

Human retinas contain two types of cell that are sensitive to light, differentiated by their shape. The light-sensitive region of a *cone* has a roughly conical shape, whereas that in a *rod* is roughly cylindrical. Cones largely dominate color vision. Cones are somewhat less sensitive to light than rods are, meaning that in low light, color vision is poor. Trichromacy occurs because there are (usually!) three distinct types of cone in the eye that mediate color perception.

Almost every camera we deal with will have either one type of sensor (a *monochrome camera*) or three types of sensor (a *color camera*). Having more than one sensor type creates an engineering problem. Ideally, one would like to have each type of sensor at every location in the camera. This is difficult to achieve. One can place whole arrays of each type of sensor in different locations, then arrange for the incoming light to be split between the sensors. Such cameras exist, but are expensive and tend to be large and heavy. Alternatively, one can "stack" the sensor types on top of one another in an array. Such arrays can be built, but present problems because the sensor types interact by "stealing" light from one another.

For consumer cameras, it is more usual to have a single array. The three different sensor types are usually referred to as red, green and blue. The types are allocated to locations in the array in a pattern known as a *mosaic* (example

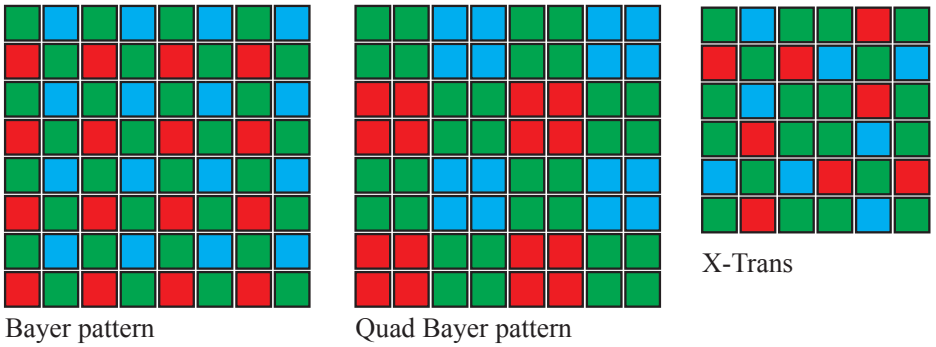


FIGURE 30.7: *Many color cameras have only one of three types of sensor at each location, where each type has a different spectral sensitivity. This is typically achieved by placing a small filter element over each location on the sensor. The filters pass more red light, more green light or more blue light respectively. The Bayer pattern (***left***) is very widely used. Alternatives include the quad Bayer pattern (***center***), which you can think of as a Bayer pattern of 2x2 super pixels and the X-trans pattern (***right***). Any pattern requires some processing to reconstruct full images of each sensor response. Different patterns offer different payoffs in this reconstruction process.*

in Figure 30.7). The camera reports a red image, a green image and a blue image even though red (and green and blue) values are measured only in a scattered set of locations. Each image is reconstructed from the scattered locations by an interpolation procedure known as *demosaicing*. Most cameras use a *Bayer pattern* mosaic. Careless interpolation procedures can result in odd color effects at boundaries and around stripes (due to aliasing; more information in Section 21.3).

30.2.3 What Colored Surfaces Look Like

When a surface is viewed under different colored lights, the color of the light reflected from that surface changes. Lighting a green surface with white light gets a green image; but so does lighting a white surface with green light. These changes have a simple and orderly form, which is easily and usefully modelled. Assume a linear camera or, equivalently, the effect of the CRF has been calibrated away. Section 21.3 gives the response $p_k(\mathbf{x})$ of the k'th type of receptor with sensitivity $\sigma_k(\lambda)$ at location $\mathbf{x}$ as

$$p_k(\mathbf{x}) = g_c E_k(\mathbf{x}) = \int_{\text{wavelength}} P(\mathbf{X} \to \mathbf{x}, \lambda)\sigma_k(\lambda) d\lambda \Delta_t$$

(where g_c is the gain constant of the linear camera). Now assume: that all power arriving at $\mathbf{x}$ comes from $\mathbf{X}$ in the scene; all reflection is diffuse; the scene is illuminated by light of one spectral energy density $E(\lambda)$; and there are no interreflections.

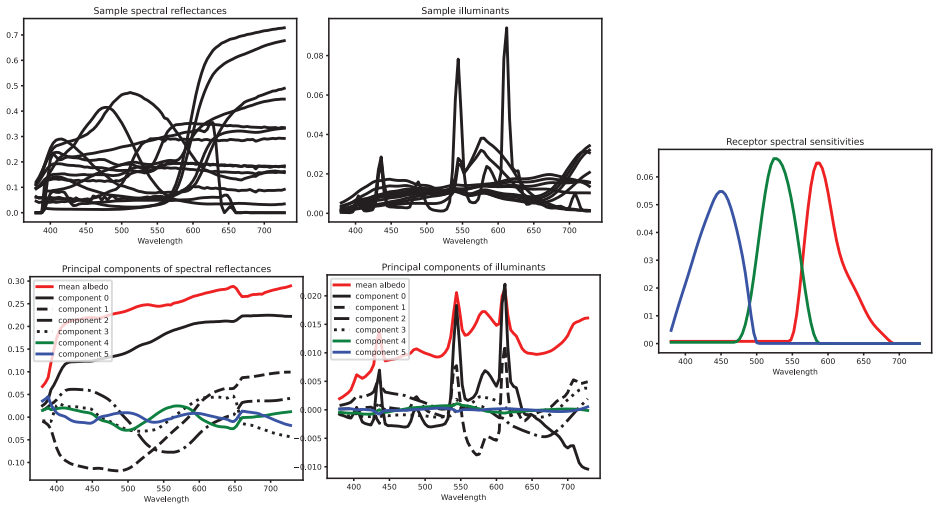


FIGURE 30.8: *On the* **top left**, *a randomly chosen subset of spectral reflectances ($\rho(\lambda)$ plotted from data published at* `https://www2.cs.sfu.ca/~colour/data/colour_constancy_synthetic_test_data/index.html`. **Top center** *shows a randomly chosen subset of illuminant spectra ($E(\lambda)$) from the same source.* **Right** *shows spectral sensitivities for receptors from the Sony DXC-930 camera, plotted using data from that source. Below the spectral reflectances and the illuminants is a set of basis functions obtained using principal components analysis of the full dataset for each. I have scaled each principal componet using the standard deviation of its coefficient when used to encode the dataset (1995 spectral reflectances, 11 illuminants in the version I used). Note that the mean and a small set of principal components will encode the spectral reflectances (respectively, illuminants) well.*

Write $\rho(\mathbf{X}, \lambda)$ for the spectral albedo at $\mathbf{X}$. Then

$$p_k(\mathbf{x}) = g_c E_k(\mathbf{x}) = \int_{\text{wavelength}} g(\mathbf{X})\rho(\mathbf{X}, \lambda)\sigma_k(\lambda)E(\lambda)d\lambda\Delta_t$$

where $g(\mathbf{X})$ is a term due to geometric effects (as in Section 21.3). Assume the camera gain and the time interval are known and so can be ignored. Finally, assume that both spectral albedo and spectral energy density can be written as a weighted sum of basis functions, so that

$$\rho(\mathbf{X}, \lambda) = \sum_i r_i(\mathbf{X})\phi_i(\lambda) \text{ and } E(\lambda) = \sum_j e_j\psi_j(\lambda).$$

This assumption is sound for quite a small basis (Figure 30.8). As the figure suggests, the basis functions for albedo and for spectral energy density may be different.

Because the lens system ensures that all power arriving at $\mathbf{x}$ comes from $\mathbf{X}$,

we can substitute $\mathbf{x}$ for $\mathbf{X}$. Now write

$$g_{ijk} = \int \phi_i(\lambda)\psi_j(\lambda)\sigma_k(\lambda)d\lambda,$$

and obtain

$$p_k(\mathbf{x}) = g(\mathbf{x}) \sum_{ijk} e_i r_j(\mathbf{x}) g_{ijk}$$

or

$$\mathbf{p}(\mathbf{x}) = g(\mathbf{x})\mathcal{A}_\mathbf{e}\mathbf{r}(\mathbf{x}).$$

(where the responses of different types of receptor have been stacked into a vector $\mathbf{p}$, the vector $\mathbf{r}$ represents the reflectance coefficients, the $\mathbf{e}$ represents the color of the illuminant, and $\mathcal{A}_\mathbf{e}$ is the matrix whose u, v'th element is $\sum_i e_i g_{ivu}$. A model of this form is known as a *finite dimensional linear model*. Usually, g_{ijk} are known or can be calibrated.

Some useful information can be extracted from finite dimensional linear models without going into details of which basis function are used. For any given illuminant, there is a linear map (given by $\mathcal{A}_\mathbf{e}$) from $\mathbf{r}$ (the coefficients of the spectral reflectances) to $\mathbf{p}$ (the pixel values). If the basis for spectral reflectances has more than three dimensions, then, for each illuminant, there will be surface reflectances that are different, but yield exactly the same pixel values (because a linear map from $d > 3$ dimensions to 3 dimensions must have a kernel).

The map $\mathcal{A}_\mathbf{e}$ is quite strongly affected by $\mathbf{e}$. This experimental fact is most easily observed by looking at the collection of all pixel values in the image – its *gamut*. As Figure 30.14 shows, if the illuminant is orange in color, then the colors in the image shift toward orange; if it is blue, they shift to blue. Because $\mathcal{A}_\mathbf{e}$ changes when $\mathbf{e}$ changes its kernel may change. In turn, if the basis for spectral reflectances has more than three dimensions, there may be pairs of surfaces that have different colors under one light and the same color under another light (exercises). This occurs in practice (Figure 30.14) and such pairs of surfaces are known as *metamers*.

30.3 REPRESENTING COLOR

Describing colors accurately is a matter of great commercial importance. For example, some brands are associated with specific colors, meaning there are people who are willing to go to a great deal of trouble to control colors exactly. Doing so requires a standard system for talking about color. Simple color names are insufficient because relatively few people know many color names, and most people are willing to associate a large variety of colors with a given name. This section barely scratches the surface of a very well-developed subject: there are many linear and non-linear color spaces (**?** is a good reference).

Useful color terms include: *hue*, the property of a color that varies in passing from red to green; *saturation*, the property of a color that varies in passing from red to pink; and *brightness* (sometimes called *lightness* or *value*, the property that varies in passing from black to white.

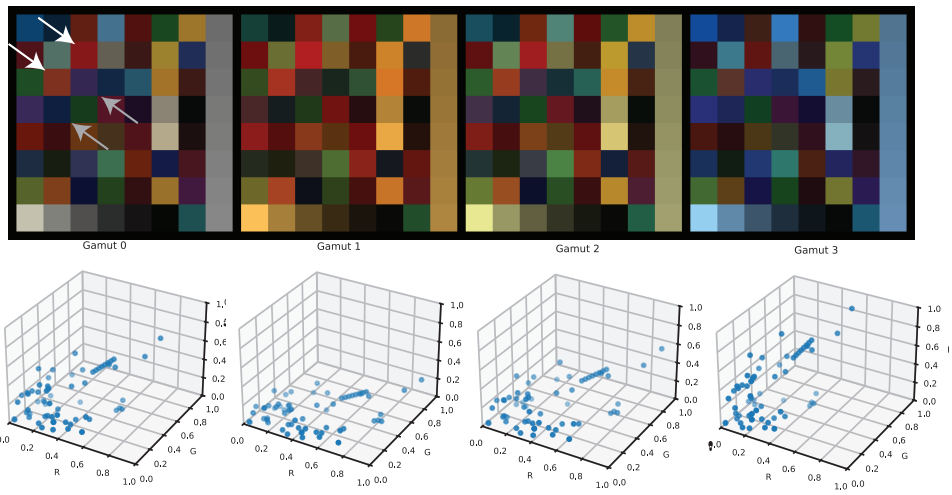


FIGURE 30.9: *The* **top row** *shows a set of images of color patches under different illuminants, simulated from data published at* `https://www2.cs.sfu.ca/~colour/data/colour_constancy_synthetic_test_data/index.html`. *Notice how the two color patches indicated by the white arrows look very similar under illuminant 1, and quite different under illuminant 3. Similarly, the two patches indicated by gray arrows look very similar under illuminant 3, but rather different under illuminant 1. Each pair is a pair of metamers. The* **bottom row** *shows the gamut of each image. Note how the gamut changes from illuminant color to illuminant color.*

30.3.1 Additive Linear Color Spaces

The natural mechanism for representing color is to agree on a standard set of primaries, and then describe any colored light by the three values of weights that people would use to match the light using those primaries. One can represent surface colors as well by using a standard light for illuminating the surface. This is a *linear color space*. The three primaries P_1, P_2, and P_3 need not be be physically realizable.

Because color matching is linear, predicting the weights that would be needed to match a particular spectral energy density is straightforward. Obtain a *color matching function* for each primary by experiment. The i'th color matching function ($f_1(\lambda)$, $f_2(\lambda)$, and $f_3(\lambda)$) records the weight for the i'th primary matching a unit energy source at wavelength λ. Because the human color system is linear, if a source $S(\lambda)$ is matched by $w_1P_1 + w_2P_2 + w_3P_3$, then

$$w_i = \int f_i(\lambda)S(\lambda)d\lambda.$$

One can obtain a linear color space by constructing the color matching functions and then looking for primaries that produce these color matching functions. A variety of different systems have been standardized by the CIE (the *commission*

international d'éclairage, which exists to create standards for such things).

The *CIE XYZ color space* is one quite popular standard. The color matching functions were chosen to be everywhere positive, so that the coordinates of any real light are always positive. It is not possible to obtain CIE X, Y, or Z primaries because for some wavelengths the value of their pectral energy density is negative. However, given color matching functions alone, one can specify the XYZ coordinates of a color and hence describe it. It is common to intersect the XYZ space with the plane $X + Y + Z = 1$ and draw the resulting figure using coordinates

$$(x, y) = \left(\frac{X}{X + Y + Z}, \frac{Y}{X + Y + Z} \right).$$

This space, which is often referred to as the *CIE xy color space* is shown in Figure **??**. CIE xy is widely used in vision and graphics textbooks and in some applications, but is usually regarded by professional colorimetrists as out of date.

The *RGB color space* is a linear color space that formally uses single wavelength primaries (645.16 nm for R, 526.32 nm for G, and 444.44 nm for B). Available colors are usually represented as a unit cube—usually called the *RGB cube*—whose edges represent the R, G, and B weights. In practice, RGB refers to the values of Red, Green and Blue components found in images. These are not particularly reliable or accurate measures of color. An easy experiment shows that images of the same scene under the same lighting with different cameras can have somewhat different RGB values. The main possible causes are that the spectral sensitivity of the camera sensors (Section 21.3) can differ from camera to camera, and the camera response function (Section 21.3) can differ from camera to camera. Another easy experiment shows that the same image can look rather different on different monitors, looks different in printed form and on a monitor, and usually looks different when printed in different ways (exercises). Another difficulty with RGB is that the R, G and B layers in an RGB image are typically very similar, but a linear transformation can decorrelate these layers quite well (exercises).

There are three constructions worth remembering.

- In both RGB and XYZ space, the sum of the color coefficients is a good representation of intensity.
- Because the color spaces are linear, and color matching is linear, all colors that can be obtained by mixing two primaries A and B lie on the line segment joining them plotted on the color space.
- Because the color spaces are linear, and color matching is linear, all colors that can be obtained by mixing three primaries A, B, and C lie in the triangle formed by the three primaries plotted on the color space. This construction determines the set of colors (or *gamut*) that a monitor can display. A glance at Figure 30.10 confirms that no triangle of points inside the color space will encode everything, meaning that no monitor will display all colors.

30.3.2 Uniform Color Spaces

In many computer vision applications it is important to know whether a color change is big enough to be noticed by a human. A straightforward experiment

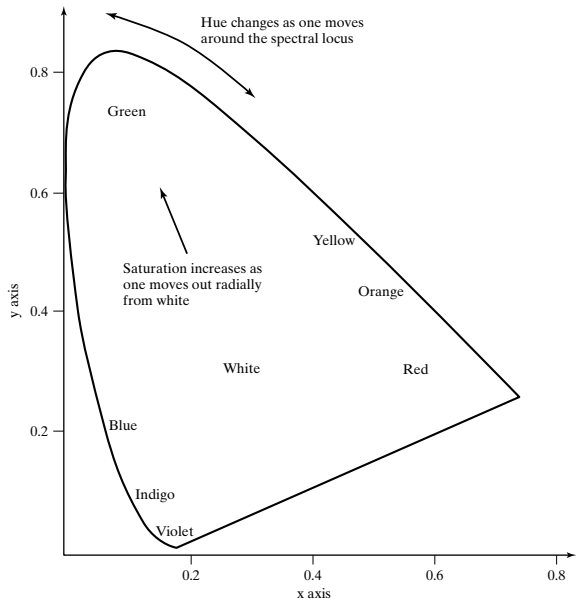


FIGURE 30.10: *The standard 1931 standard CIE xy color space, with color names marked on the diagram. The curved boundary of the figure is often known as the* spectral locus*; it represents the colors experienced when lights of a single wavelength are viewed. Generally, colors that lie farther away from the neutral point are more saturated—the difference between deep red and pale pink—and hue—the difference between green and red—as one moves around the neutral point. Near the center of the diagram is the neutral point, the color whose weights are equal for all three primaries. CIE selected the primaries so that this light appears achromatic. I have deliberately not shown this space in color, because the colors that would print on the figure are misleading. Color printing processes cannot reproduce the full set of colors that people can perceive. For example, colors on the spectral locus cannot be reproduced in print.*

determines for each color how big a change in color is required for the change to be noticeable (this is a *just noticeable difference*). Show a split screen with two colors in it to an observer. Start with both colors the same base color, then change one until the observer can only just tell it has changed by comparing it to the other. The result is a set of small blobs of equivalent colors around each base color. These can be represented as ellipses (Figure 30.11). In linear color spaces, these ellipses vary quite strongly with the base color. In turn, this means that, to get an accurate estimate of the significance of a color difference (say $(\Delta R, \Delta G, \Delta B)$), one needs to apply some transformation to $(\Delta R, \Delta G, \Delta B)$ *that depends on the particular* (R, G, B) of the base color. This is wildly inconvenient in practice.

With an appropriate choice of non-linear transformation applied to linear color coordinates, one can find a *uniform color space*. In such a space, if the distance in coordinate space is below some threshold, a human observer would not be able

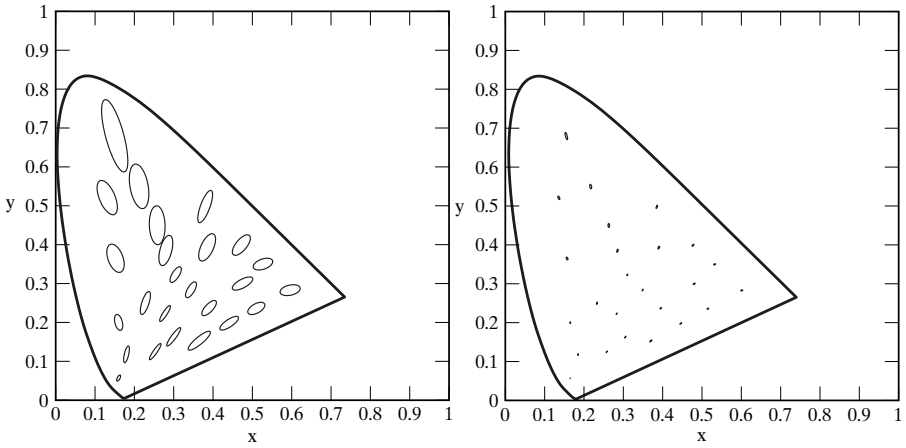


FIGURE 30.11: *This figure shows variations in color matches on a CIE xy space. At the center of the ellipse is the color of a test light; the size of the ellipse represents the scatter of lights that the human observers tested would match to the test color; the boundary shows where the just noticeable difference is. The ellipses in the figure on the* **left** *have been magnified 10x for clarity; on the* **right** *they are plotted to scale, with color names on the CIE diagram as a reference. The ellipses are known as MacAdam ellipses after their inventor. Notice that the ellipses at the top are larger than those at the bottom of the figure, and that they rotate as they move up. This means that the magnitude of the difference in x, y coordinates is a poor guide to the difference in color. Ellipses are plotted using data from* **?**.

to tell the colors apart. A uniform space can be obtained from CIE XYZ using a projective transformation to obtain the *CIE u′v′ space CIE u'v' space*. The coordinates are:

$$(u', v') = \left(\frac{4X}{X + 15Y + 3Z}, \frac{9Y}{X + 15Y + 3Z} \right).$$

Generally, the distance between coordinates in $u\prime$, $v\prime$ space is a fair indicator of the significance of the difference between two colors. Of course, this omits differences in brightness.

CIE LAB is now almost universally the most popular uniform color space. Coordinates of a color in LAB are obtained as a non-linear mapping of the XYZ coordinates:

$$\begin{aligned} L^* &= 116 \left(\frac{Y}{Y_n} \right)^{\frac{1}{3}} - 16 \\ a^* &= 500 \left[\left(\frac{X}{X_n} \right)^{\frac{1}{3}} - \left(\frac{Y}{Y_n} \right)^{\frac{1}{3}} \right] \\ b^* &= 200 \left[\left(\frac{Y}{Y_n} \right)^{\frac{1}{3}} - \left(\frac{Z}{Z_n} \right)^{\frac{1}{3}} \right] \end{aligned}$$

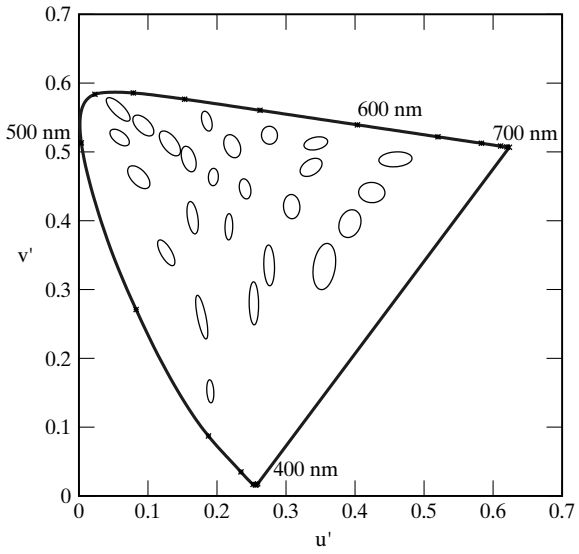


FIGURE 30.12: *This figure shows the CIE 1976 u′, v′ space, which is obtained by a projective transformation of CIE x, y space. The intention is to make the MacAdam ellipses (from Figure 30.11) uniformly circles. This would yield a uniform color space. A variety of non-linear transforms can be used to make the space more uniform (see* **?** *for details).*

Here X_n, Y_n, and Z_n are the X, Y, and Z coordinates of a reference white patch. The reason to care about the LAB space is that it is substantially uniform. In some problems, it is important to understand how different two colors will look *to a human observer*, and differences in LAB coordinates give a good guide.

30.3.3 HSV Color Space

It is a common intuition that hue changes from red through orange to yellow, and then green, and from there to cyan, blue, purple, and then red again. Another way to think of this is to picture local hue relations: red is next to purple and orange; orange is next to red and yellow; yellow is next to orange and green; green is next to yellow and cyan; cyan is next to green and blue; blue is next to cyan and purple; and purple is next to blue and red. Each of these local relations works, and globally they can be modeled by laying hues out in a circle. This means that no single coordinate of a linear color space can model hue, because that coordinate has a maximum value that is far away from the minimum value. Applying a non-linear transformation to the RGB space can produce a color space that respects these relations. The *HSV space* (for hue, saturation, and value), is obtained by looking down the center axis of the RGB cube. Because RGB is a linear space, brightness—called *value* in HSV—varies with scale out from the origin. We can flatten the RGB cube to get a 2D space of constant value and for neatness deform it to be a hexagon. This gets the structure shown in Figure 30.13, where hue is

given by an angle that changes as one goes round the neutral point and saturation changes as one moves away from the neutral point.

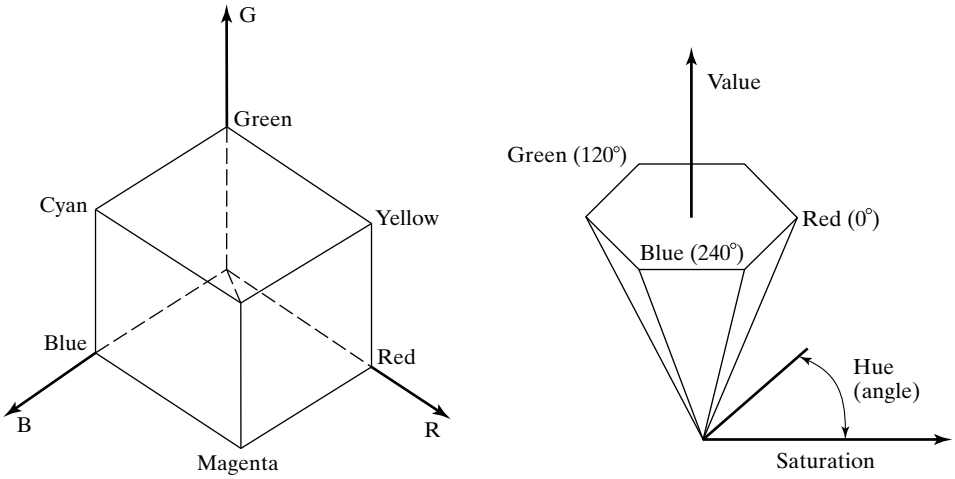


FIGURE 30.13: *On the* **left**, *we see the RGB cube; this is the space of all colors that can be obtained by combining three primaries (R, G, and B—usually defined by the color response of a monitor) with weights between zero and one. It is common to view this cube along its neutral axis—the axis from the origin to the point (1, 1, 1)—to see a hexagon. This hexagon codes hue (the property that changes as a color is changed from green to red) as an angle, which is intuitively satisfying. On the* **right**, *we see a cone obtained from this cross-section, where the distance along a generator of the cone gives the value (or brightness) of the color, the angle around the cone gives the hue, and the distance out gives the saturation of the color.*

30.3.4 Subtractive Mixing and Inks

Intuition from one's finger-painting days suggests that the primary colors should be red, yellow, and blue, and that yellow and blue mix to make green. The reason this intuition doesn't apply to monitors is that paints involve pigments—which mix subtractively—rather than lights. Pigments can behave in quite complex ways, but the simplest model is that pigments remove color from incident light, which is reflected from paper. Thus, red ink is really a dye that absorbs green and blue light—incident red light passes through this dye and is reflected from the paper. This is *subtractive color mixing*.

Color spaces for this kind of mixing can be quite complicated. In the simplest case, mixing is linear (or reasonably close to linear), and the *CMY space* applies. In this space, there are three primaries: *cyan* (a blue-green color), *magenta* (a purplish color), and *yellow*. These primaries should be thought of as subtracting a light primary from white light; cyan is $W - R$ (white − red); magenta is $W - G$ (white − green), and yellow is $W - B$ (white − blue). Now the appearance of mixtures can be evaluated by reference to the RGB color space. For example, cyan

and magenta mixed give

$$(W - R) + (W - G) = R + G + B - R - G = B,$$

that is, blue. Notice that $W + W = W$ because we assume that ink cannot cause paper to reflect more light than it does when uninked. Practical printing devices use at least four inks (cyan, magenta, yellow, and black) because mixing color inks leads to a poor black, it is difficult to ensure good enough registration between the three color inks to avoid colored haloes around text, and color inks tend to be more expensive than black inks. One reason that fingerpainting is hard is that the color resulting from mixing paints can be quite hard to predict. This is because the outcome depends very strongly on details such as the specific pigment in the paint, the size of pigment particles, the medium in which the pigment is suspended, the care put into stirring the mixture, and similar parameters; usually, we do not have enough detailed information to use a full physical model of these effects. A useful study of this difficult topic is [**?**].

30.4 INFERENCE FROM COLOR

30.4.1 White Balancing

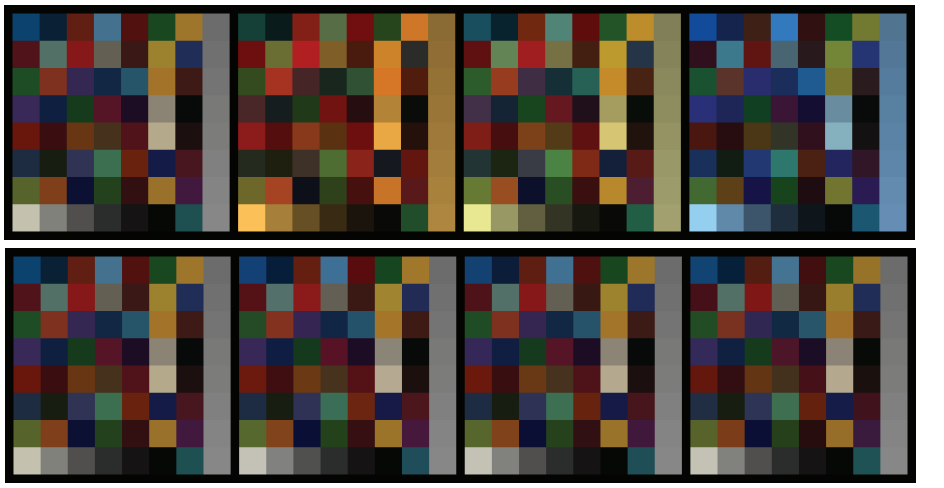
Color changes caused by illuminant color are an established nuisance in photography. The mechanism that allows humans to discount the effect of the illuminant color seems not to work on photographs. In turn, a scene viewed near (say) sunset that looks natural to the photographer may look very strongly colored in the photograph. *White balancing* is the process of changing the color cast of the image to make it look as if it had been taken under white light.

Recall the map $\mathcal{A}_{\mathbf{e}}$ from Section 30.2.3. Write $\mathbf{t}$ for the vector of coefficients of the target light in the basis for the illuminant. The pixel at $\mathbf{x}$ in the photograph sees a spectral reflectance represented by $\mathbf{r}(\mathbf{x})$, and has color $\mathbf{p}(\mathbf{x}) = \mathcal{A}_{\mathbf{e}}\mathbf{r}(\mathbf{x})$. In white light, this surface would take the color $\mathcal{A}_{\mathbf{w}}\mathbf{r}(\mathbf{x})$. Assume for the moment that $\mathbf{e}$ is known. It is tempting to think that one might recover $\mathbf{r}(\mathbf{x})$ using an inverse. Resist this temptation, because $\mathbf{A}_{\mathbf{e}}$ will not have an inverse in most circumstances (recall metamers from Section 30.2.3; these are pairs surfaces that have the same color under one light, and different colors under another).

Instead, assume there are one or more references in the image. In practical photography, the reference is usually a surface that the photographer knows to be neutral and light in color (perhaps a wall painted white indoors, or the body of a seagull). Write $\mathbf{p}_{i,\mathbf{t}}$ for the color of the i'th reference surface under the target light and $\mathbf{p}_{i,\mathbf{e}}$ for the color of that surface under the illuminant of the photograph. Then the image can be corrected by finding

$$\underset{\mathcal{M}}{\operatorname{argmin}} \sum_i \left(\mathbf{p}_{i,\mathbf{t}} - \mathcal{M}\mathbf{p}_{i,\mathbf{e}}\right)^T \left(\mathbf{p}_{i,\mathbf{t}} - \mathcal{M}\mathbf{p}_{i,\mathbf{e}}\right)$$

which is a simple least-squares problem (exercises). The material of Section 30.2.3 implies that $\mathcal{M}$ is a general 3×3 matrix (exercises). It is quite usual to assume that $\mathcal{M}$ is a diagonal matrix instead, and compute one scale for each channel. In the simplest case, one simply scales by the pixel values for a white surface (exercises).



dw: 8.6 dl: 6.1 ls: 5.1 dw: 5.5 dl: 3.1 ls: 2.7 dw: 14.8 dl: 13.1 ls: 8.3

FIGURE 30.14: *White balancing using a linear map, illustrated for various regimes. The* **top row** *shows a set of images of color patches under different illuminants, simulated from data published at* `https://www2.cs.sfu.ca/~colour/data/colour_constancy_synthetic_test_data/index.html`. *The* **bottom row** *shows these images corrected to the illuminant of the first image by computing the $\mathcal{M}$ of the text using every pixel value as a reference. Details in text.*

If you scale each channel separately, the color space in which you work makes some difference (exercises). You could reasonably expect all these effects to make a large difference, but Figure **??** suggests the differences are quite small. This figure compares an extreme case of white balancing using a general $\mathcal{M}$ and all pixels as references to a range of other strategies. This exposes the extent to which the kernel of $\mathcal{A}_\mathbf{e}$ can create problems. The first image has no error. Under each of the other corrected images is the root mean square error of correcting using each of three regimes. The error is on a scale of 0-255, very common for 8 bit images. The cases are: `dw`, where the image is corrected by scaling R, G, and B independently so that the patch in the bottom right corner matches the original, so $\mathcal{M}$ is diagonal and determined by one reference; `dl`, where the image is corrected by scaling R, G, and B independently so that all pixels match, so $\mathcal{M}$ is diagonal and determined by all pixels; and `ls`, where $\mathcal{M}$ is general and determined by all pixels. Notice `ls` gives the lowest error, but the error is not zero (metamers again!); `dl` has larger error than `ls`, but not much larger; and `dw` is surprisingly good given there is only one reference and the map is diagonal.

30.4.2 Color Constancy: Surface Color from Image Color

Color constancy refers to algorithms that can take an image, discount the effect of the light, and report the actual color of the surface being viewed. This report could be in the form of the color of the scene under white light, so there is a strong link

to white balancing. One strategy involves identifying some form of reference, then using that reference to white balance the image.

Humans have some form of color constancy algorithm. People are often unaware of this, and inexperienced photographers are sometimes surprised that a scene photographed indoors under fluorescent lights has a blue cast, whereas the same scene photographed outdoors may have a warm orange cast. Human color constancy is not perfectly accurate, and people can choose to disregard information from their color constancy system. As a result, people can often report:

- the color a surface would have in white light (often called *surface color*);
- the color of the light arriving at the eye (a useful skill that allows artists to paint surfaces illuminated by colored lighting); and
- the color of the light falling on the surface.

Color constancy is easiest when the camera is linear or can be photometrically calibrated to behave like a linear camera. In the first instance, assume the scene is flat and frontal, and has piecewise constant spectral albedo. This is known as a *Mondrian world* assumption, because such a scene looks like a collage of colored papers apparently reminiscent of the works of the artist Piet Mondrian. Because the scene is frontal and the spectral albedos are constant, $\mathbf{x}$ can be replaced with an index u (one index for each colored patch), yielding

$$\mathbf{p}_u = \mathbf{c}(\mathbf{r}_u, \mathbf{e})$$

The receptor responses are observed, and the problem is to recover each $\mathbf{r}_u$ (the spectral albedo of each patch) and $\mathbf{e}$ (the color of the illuminant). Straightforward strategies are:

- Assume there is a white patch, and it can be identified. For that patch, $\mathbf{r}$ is known. It is an exercise to show that, assuming appropriate dimensions for the bases, $\mathbf{e}$ can then be recovered, and from that the other $\mathbf{r}$.
- Assume that the average spectral albedo is known. It is again an exercise to show that $\mathbf{e}$ can be recovered from this information, and from that the $\mathbf{r}$.

These assumptions yield quite workable algorithms in practice. More sophisticated procedures rely on the *gamut* of observed colors. Illuminating a colored surface with a colored light tends to shift the color of the surface somewhat toward the color of the light, and so some colors aren't observed. So, for example, in greenish light one does not see strongly red surfaces; in reddish light one does not see very blue surfaces; and so on. This effect is sufficiently pronounced that it can be used to estimate the illuminant (exercises).

It is convenient to ignore the geometric term $g(\mathbf{x})$ for simplicity, but it isn't necessary. Recall the lightness model of Section 21.3. Assume that $g(\mathbf{x})$ varies slowly like the shading term in that model, and that $\mathbf{r}(\mathbf{x})$ is piecewise constant like the lightness term in that algorithm. Apply that algorithm to each type of receptor response separately, to recover the components of $\mathbf{r}(\mathbf{x})$ separately up to a constant (exercises). Now recover these constants using one of the procedures above (exercises)

30.4.3 Shadow Removal Using Color

Lightness methods make the assumption that "fast" edges in images are due to changes in albedo (Section 29.4.1). This assumption is usable, but fails badly at shadows, particularly shadows in sunlight outdoors (Figure 30.16), where there can be a large and fast change of image brightness. People usually are not fooled into believing that a shadow is a patch of dark surface, so must have some method to identify shadow edges. Home users often like editing and improving photographs, and programs that could remove shadows from images would be valuable. A shadow removal program would work something like a lightness method: find all edges, identify the shadow edges, remove those, and then integrate to get the picture back.

There are some cues for finding shadow edges that seem natural, but don't work well. One might assume that shadow edges have very large dynamic range (which albedo edges can't have; see Section **??**), but this is not always the case. One might assume that, at a shadow edge, there was a change in brightness but not in color. It turns out that this is not the case for outdoor shadows, because the lit region is illuminated by yellowish sunlight, and the shadowed region is illuminated by bluish light from the sky, or sometimes by interreflected light from buildings, and so on. However, a really useful cue can be obtained by modelling the different light sources.

Assume that light sources are black bodies (Section **??**), so that their spectral energy density is a function of temperature, that surfaces are diffuse, and that the color receptors each respond only at one wavelength. Write λ_k for the wavelength at which the k'th receptor responds, so that $\sigma_k(\lambda) = \delta(\lambda - \lambda_k)$. View a surface with spectral albedo $\rho(\lambda)$ illuminated by one of these sources at temperature T. The response of the j'th receptor will be

$$r_j = \int \sigma_j(\lambda)\rho(\lambda)K\frac{\exp(-hc/k\lambda T)}{\lambda^5}d\lambda = K\rho(\lambda_j)\frac{\exp(-hc/k\lambda_j T)}{\lambda_j^5}.$$

A color space that is very well behaved can be formed by taking $c_1 = \log(r_1/r_3)$, $c_2 = \log(r_2/r_3)$, because

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} + \frac{1}{T}\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

where $a_1 = \log\rho(\lambda_1) - \log\rho(\lambda_3) + 5\log\lambda_3 - 5\log\lambda_1$ and $b_1 = (hc/k)(1/\lambda_3 - 1/\lambda_1)$ (and a_2, b_2 follow). Notice that, when one changes the color temperature of the source, the (c_1, c_2) coordinates move along a straight line. The direction of the line depends on the sensor, *but not on the surface.* Call this direction the **color temperature direction**. The intercept of the line depends on the surface.

Now consider a world of colored surfaces, and map the image colors to this space. There is a family of parallel lines in this space, whose direction is the color temperature direction. Different surfaces may map to different lines. Changing the color temperature of the illuminant will cause each color in this space to move along the color temperature direction, but colors will not move from line to line. Represent a surface color by a coordinate describing its line. For example, construct

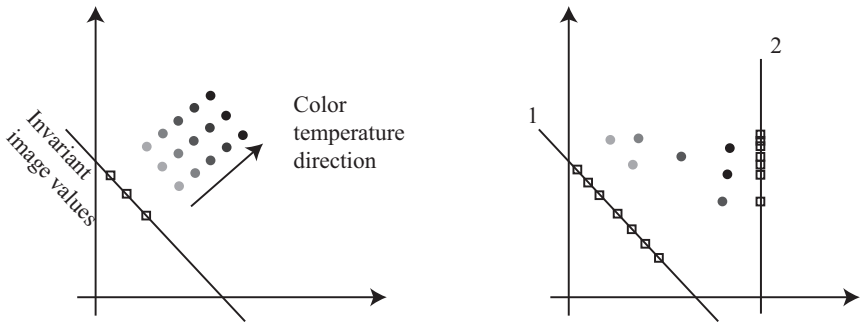


FIGURE 30.15: *Changing the color temperature of the light under which a surface is viewed moves the* (c_1, c_2) *coordinates of that surface along the color temperature direction (***left***; the different gray patches represent the same surface under different lights). Project the coordinates along the* (c_1, c_2) *direction onto some line to obtain a value that doesn't change when the illuminant color temperature changes. This is the invariant value for that pixel. Generally, we do not know enough about the imaging system to estimate the color temperature direction. However, we expect to see many different surfaces in each scene; this suggests that the right choice of color temperature direction on the* **right** *is 1 (where there are many different types of surface) rather than 2 (where the range of invariant values is small).*

a line through the origin that is perpendicular to color temperature direction, then represent a surface color by distance along this line (Figure 30.15). Represent each pixel in the image in this space. In this representation the color image becomes a gray-level image, *where the gray level does not change inside shadows* (because a shadow region just has a different color temperature to the non-shadowed region). **?** calls this the *invariant image*. Any edge that appears in the image but not in the invariant image is a shadow edge, so now apply the original formula: find all edges, identify the shadow edges, remove those, and then integrate to get the picture back.

There are some practical difficulties. Usually, it is hard to know enough about the sensors to evaluate the as and bs that define this family of lines, so it is hard to get the invariant image directly. However, as Figure 30.15 suggests, a good estimate of the color temperature direction in (c_1, c_2) follows from a form of entropy reasoning. This means the invariant image can be constructed without knowing anything about the sensor. Search directions in (c_1, c_2) space, projecting all the image colors along that direction; the estimate of the color temperature direction is the one where this projection yields the largest entropy. In practice, the method works well, though great care is required with the integration procedure to get the best results (Figure 30.16).

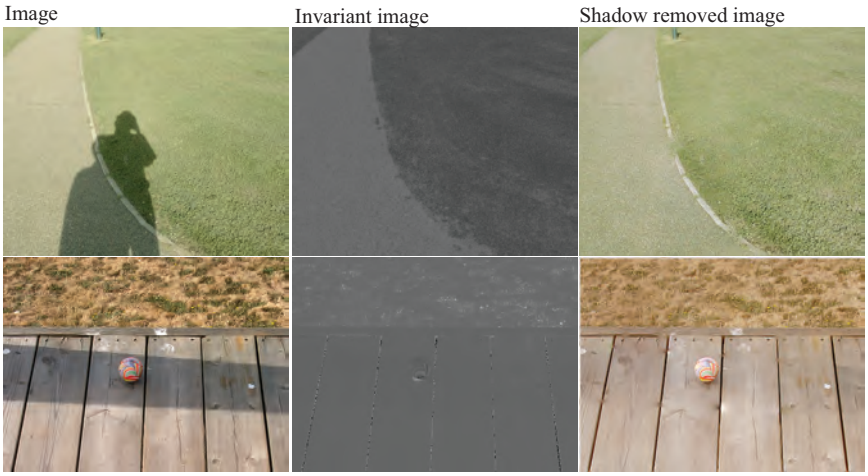


FIGURE 30.16: *The invariant of the text and of Figure 30.15 does not change value when a surface is shadowed. Finlayson* et al. *use this to build a shadow removal system that works by (a) taking image edges; (b) forming an invariant image; then (c) using that invariant image to identify shadow edges; and finally (d) integrating only non-shadow edges to form the result. The results are quite convincing.*