Chapter

SIMULATION OF DIGITAL CAMERA IMAGES FROM HYPERSPECTRAL INPUT

Philippe Longère
longere@psych.ucsb.edu
Department of Psychology
UC Santa Barbara
Santa Barbara, CA 93106-9660

David H. Brainard
brainard@psych.ucsb.edu
Department of Psychology
UC Santa Barbara
Santa Barbara, CA 93106-9660

1. INTRODUCTION

An important goal for digital color cameras is to record enough information about a scene so that it can be reproduced accurately for a human observer. By accurately, we mean so that the human observer will perceive the reproduction as looking like the original.

We can divide factors that affect the overall performance of a digital camera into two classes. The first class is camera design parameters. These influence the information recorded by the camera. Examples of design parameters include the spatial resolution of the camera, the spectral sensitivities of its sensors, and the reliability of the sensor responses. The second class of factors is the algorithms applied to the camera data. Camera design evaluation must take both classes of factors into account, particularly if they interact.

The goal of this chapter is to describe a digital camera simulator that predicts the relation between a scene and the camera image. The input

to the simulator is a high resolution hyperspectral image of a scene. This input is used together with a working model of camera performance to predict the camera image. The simulation software uses a parametric camera description, so that the effects of varying the design may be explored. In addition, the software provides an environment for applying and evaluating image processing algorithms. Because many image processing algorithms are based on physical models of image formation and camera performance, we have emphasized data structures that make this information readily available.

The rest of this chapter is organized as follows. First we provide a description of our hyperspectral imaging system. Second, we describe the general model of camera performance that we use in our simulations. We evaluate how well we can simulate images by comparing a simulated and directly acquired image for a Kodak DCS-200 color camera. Third, we describe the image processing pipeline that is generally used to render camera data on color monitors and review standard algorithms that may be used at each step. Finally, we provide some examples that explore the effect of varying camera parameters on image quality.

2. HYPERSPECTRAL IMAGE DATA

Commonly used digital cameras provide data in several (usually three) separate image planes. If we neglect the possibility of interleaved sampling (mosaicing), each plane provides a representation of the scene as it would be seen by a single class of color sensors. Usually three sensor classes with broadband spectral sensitivities are chosen to provide red, green, and blue image planes. The principle of hyperspectral imaging is similar, except that the number of image planes is increased and the sensors are chosen to be narrowband (1, 2, 3). The narrowband sensor planes may be thought of as providing data that represent the scene at a series of wavelengths. Figure 1.1 illustrates the idea of a hyperspectral

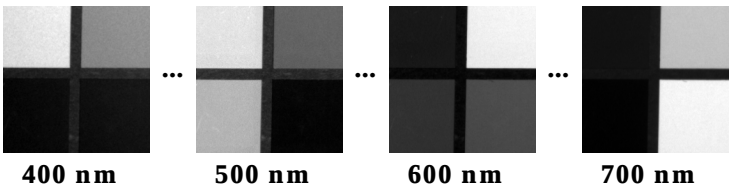


Figure 1.1 Spatial view of a hyperspectral image. The figure shows 4 image planes taken from a small region of a hyperspectral image. Each image plane provides the spatial information available at a single wavelength. Each image plane has been normalized separately for display. Our actual hyperspectral images consist of 31 image planes at 10 nm steps between 400 nm and 700 nm.

image. The figure shows a small region of 4 individual image planes, each corresponding to one wavelength. The region shown is a portion of the Macbeth Color Checker (MCC). The *spatial view* of the hyperspectral image shown in the figure emphasizes the fact that the image provides information about spatial structure at each wavelength. It is easy to see from this representation that the image contains four separate squarish regions. In the color checker documentation, these are given color names (specified clockwise from upper left) purplish-blue, moderate red, red, and green. Note that the fact that individual image planes have a similar spatial structure is not peculiar to the small image region shown – it is characteristic of all hyperspectral images we have examined. However, across the individual planes, the relative intensity between image regions differs. Larger hyperspectral image planes may be found on the world-wide-web at URL `http://color.psych.ucsb.edu/simchapter/`.

The color information contained in a hyperspectral image is not explicit in the spatial view. Figure 1.2 shows a *spectral view* of the same hyperspectral image that Figure 1.1 shows in spatial view. In this case, one location is chosen within each of the four MCC regions and its 31 band spectrum is plotted as a function of wavelength. The spectral view makes explicit the wavelength information contained in the image. From this view, it is possible to get an idea of the color that the individual regions would appear. The purplish-blue region has more power at shorter wavelengths, the green region has more power in the middle wavelengths, and the two red regions have more power in the longer wavelengths.

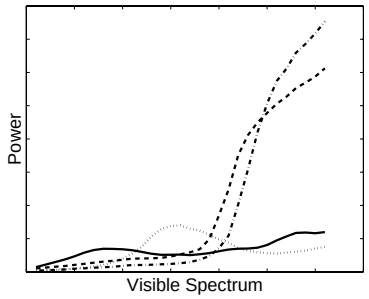
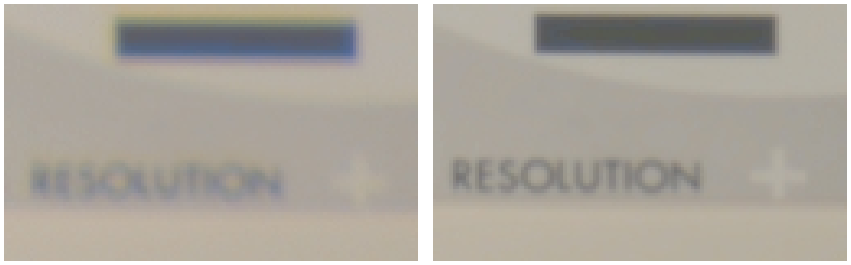


Figure 1.2 Spectral view of a hyperspectral image. The figure shows the spectra extracted from four regions of the same hyperspectral image shown in Figure 1.1. The spectral view makes explicit the wavelength information contained in the hyperspectral image. The purplish-blue region (solid line) has more power at shorter wavelengths, the green (dotted line) region has more power in the middle wavelengths, and the two red regions (dashed line for moderate, dash-dot for red) have more power in the longer wavelengths.

The hyperspectral system we use is based on a scientific grade monochrome CCD camera (Photometrics PXL, 2K by 2K spatial resolution at 12 bits/pixel direct digital interface, electronic control of temporal integration, liquid cooling for noise reduction, linear intensity-response function, 50 mm Nikkon lens) interfaced to a Macintosh host. Hyperspectral images are acquired by sequentially placing interference filters in the optical path of the camera and acquiring monochromatic images through each filter. For the work reported here, the filters were placed between the camera lens and the CCD sensor. The filters were narrowband (roughly 10 nm full-width at half-height) with peak wavelengths that evenly spanned the visible spectrum (31 filters, 400 nm to 700 nm in 10 nm steps). Aperture (f-stop) and exposure duration for each monochromatic image were adjusted so that the image data were within the response range of the camera, and focus was adjusted individually for each monochromatic image. Dark images (camera shutter not opened) were acquired for each exposure duration used, and the appropriate (same exposure duration) dark image was subtracted from each monochromatic image. We refer to the set of dark subtracted monochromatic images as a *raw hyperspectral image*. A number of processing steps must be applied to the raw hyperspectral image to produce a *calibrated hyperspectral image*.

Because the interference filters have different thicknesses and transmit light of different wavelengths, we found it necessary to refocus the camera for each narrowband image plane. This fact together with the laws of geometric optics (4) means that even in the absence of small camera movement, each image plane may have a slightly different magnification. In addition, small camera movements are unavoidable and can produce translations and rotations of the individual image planes. We registered all of the image planes to a common geometry by applying a registration algorithm. Our algorithm is a slightly modified version of one designed to handle image sequences acquired with a moving camera, as would be needed when building a panoramic composite or in remote sensing applications (5, 6). The original algorithm assumes that the only differences between individual images in the sequence are geometric. In our case, the image data also differ because the wavelength changes from plane to plane. Qualitatively, changing wavelength does not affect the basic spatial structure of the image (e.g. the location of the edges) but does change the magnitude and sign of intensity changes across edges (see Figure 1.1). We found that the effect of wavelength variation interfered with the operation of the original algorithm. We therefore applied a binary edge-finding operator to each image plane before alignment. Using this procedure, each image plane from the raw hyperspectral image

was aligned to the 550 nm image plane. Typical results of applying the registration algorithm are illustrated in Figure 1.3. Each panel shows a weighted sum of the hyperspectral image planes. The weights were chosen after radiometric calibration (see below) so that values in the resulting monochrome image are proportional to luminance. In the left panel, the summed images were not aligned, whereas in the right panel they were. The effect of alignment is quite noticeable and greatly improves the sharpness of the summed image.



(a) Luminance image, without registration.

(b) Luminance image, with registration.

Figure 1.3 Effect of hyperspectral image plane registration. See description in text.

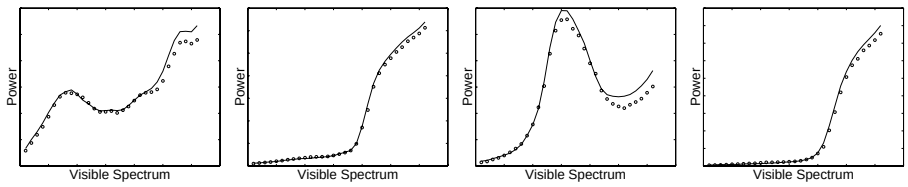
After registration, each pixel in the raw hyperspectral image provides a series of values $(c_1 ... c_{31})$. Each of these values c_i is related to the radiance of the light reflected from the corresponding scene location at a single[1] wavelength λ_i. The relation between the value c_i and the actual scene radiance depends on the spectral transmissivity of the interference filter and the spectral sensitivity of the CCD camera as well as the the lens aperture, exposure time, and camera gain at the time of acquisition. As noted above, these latter parameters were adjusted individually to keep the data for each image plane within the operating range of the camera. In addition, there was some light scatter in our optical system. To correct both for these factors we proceeded as follows.

First, we always inserted a Macbeth Color Checker Chart (MCC) into the scene. This provided 24 surfaces with known reflectance functions. At each wavelength λ_i, the camera values from each of the MCC surfaces should be proportional to the surface reflectances at that wavelength. We used linear regression to find the best fitting line between the 24 measured values and known reflectances: $c_i = mr_i + b_i$ where r_i

[1] More precisely, the light comes from a narrow wavelength band centred on λ_i.

represents known reflectance. We then subtracted the value of b_i from each location in the image plane for λ_i. This affine correction is based on the approximation that the scattered light is spatially uniform in the image plane. Although this approximation does not hold exactly, we found that applying the affine correction improved the visual quality of the individual image planes by removing an apparent veiling luminance.

After the affine correction, we converted the data for individual image planes into radiometric units. A spectroradiometer (PhotoResearch PR-650) was used to measure a reference spectrum from a small approximately uniform image region at the same time as the 31 monochromatic image planes were acquired.[2] For each wavelength, the reference spectrum provides the radiance (in $W/sr/m^2$) for a single image location. Using the reference spectrum and the affine corrected image values extracted from the same region of the hyperspectral image, we compute calibration factors for each image plane that bring the data for that plane into units of $W/sr/m^2$. Applying these factors results in the calibrated hyperspectral image. Similar radiometric calibration procedures have been used by others (1, 7).



(a) Purp. Blue (b) Mod. Red (c) Green (d) Red

Figure 1.4 Comparison of spectra obtained by direct radiometric measurement and extracted from a hyperspectral image. Each panel plots power against wavelength. Solid lines: radiometric measurements. Open circles: spectra obtained from hyperspectral image.

To evaluate the accuracy of the radiometric calibration, we can compare spectra measured directly from various image regions with spectra inferred from the same regions of the calibrated hyperspectral image. Figure 1.4 shows this comparison for the four MCC regions shown in Figure 1.1. The agreement between measured and inferred spectra is reasonable, although not perfect. Figure 1.5 shows summary compari-

[2] We generally chose a white reference patch as this provides good signal-to-noise at all wavelengths and also provides an estimate of the spectrum of the illuminant. Knowing the illuminant is useful for algorithm evaluation, see below.

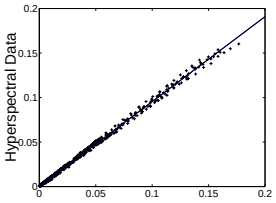


Figure 1.5 Comparison of spectra obtained by direct radiometric measurement and extracted directly from a hyperspectral image. The figure plots power as calculated from the hyperspectral image against that obtained by direct radiometric measurement. Each point shows power for one wavelength in one of the 24 MCC regions. The line shown is the best fitting line through the origin. This line has slope 0.96.

son for all 24 regions of the MCC. In this figure, each point represents spectral power at one wavelength for one image region. The x-axis shows the values obtained from direct radiometric measurements while the y-axis shows the values obtained from the hyperspectral image. These plots show that the relative spectra obtained from the hyperspectral image are essentially correct, but that the overall power in the hyperspectral measurements is slightly too low. This discrepancy may arise because we do not correct for geometric effects caused by the optics of our hyperspectral camera. Discussions of geometric effects are available elsewhere (8, 9).

3. CAMERA SIMULATION

The goal of our camera simulation is to start with a hyperspectral image and to predict the digital values that would have been obtained if an actual camera had instead imaged the same scene. To achieve this goal, we need to model camera performance. In our current work, we employ a relatively simple camera model.

Figure 1.6 illustrates the basic stages of the model. An optical system produces an image of the scene on the camera's sensor array(s). By sensor array we mean a device (often a CCD chip) which converts light intensity to digital values at each of a large number of individual pixels. Important camera parameters include the number of sensor arrays it contains, the spatial resolution of the sensor arrays, and the spectral properties of the sensors. In addition, it is necessary to model the relation between light intensity and digital values, as well as to characterize the reliability and precision of the digital representation.

A typical camera has three distinct spectral sensor classes (often labelled R, G, and B), but in general there can be any number. In many cameras, the distinct sensor classes are created by placing a mosaic of

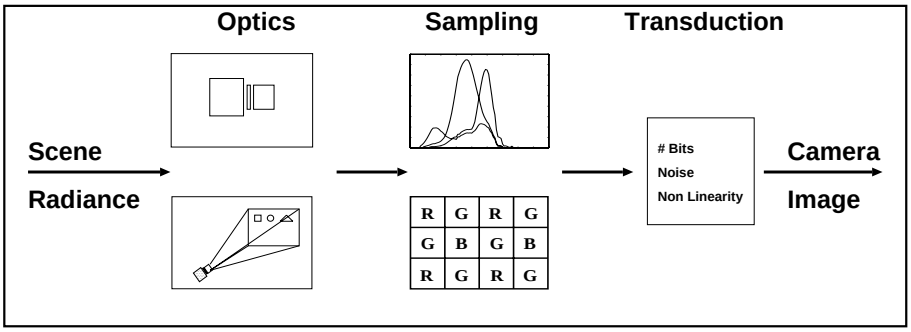


Figure 1.6 Basic camera model. As described in the text, light from the scene is imaged by the camera's optics onto one or more sensor arrays. The arrays sample the image both spatially and spectrally. Here a mosaiced sampling design is illustrated. The light intensity at the sampled locations is converted to digital values through the process of transduction. Modelling of this process takes into account the efficiency or gain of the sensor array, variability introduced by noise, quantization, and static non-linearities.

color filters over a single sensor array. For such mosaiced designs, the response of only a single sensor class is available at each pixel. In other cameras, the distinct sensor classes are produced by using optical techniques to produce multiple images of the scene, each of which is seen by a separate sensor array. For such full color designs, responses of all sensor classes are available at each pixel.

The actual digital values at each pixel are related to the light intensity imaged on that pixel, but a number of factors may make the relationship noisy (10, 11) or non-linear (12, 13). Photon fluctuations, sensor read-out noise, and quantization all produce variability. The intensity-response relation may be non-linear either because of device physics or because such a non-linearity is designed into the camera. The non-linearity may occur either before or after digitization (13).

Simulating a digital camera requires implementing each stage of the camera model in software. In designing our simulation software, we felt it was important to use a modular approach. Although we have outlined the basic elements of a camera model, certain camera designs or simulation applications may require more elaboration. For example, in applications that focus on lens design, it may be useful to implement a detailed specification of each element in the optical system (14) and to use numerically intensive numerical ray tracing techniques to simulate the optical performance. In applications where the interest is in designing sensor arrays themselves, it may be useful to simulate the

transduction process in more detail. Thus one could include stages that model the conversion of light energy to analog electrical signals, followed by stages that model on-chip analog processing, followed by a stage that models analog to digital conversion (15).

Our camera simulation software specifies a camera at two levels. The first level of specification is to define a camera architecture. This is done by providing a list of functions that are applied in sequence to the image data. Each function takes as input a data structure containing camera parameters and a data structure that describes the image as simulated up to that point in the processing pipeline. It is the camera parameter structure that provides the second level of description. Within a particular architecture, many different cameras can be simulated. The camera parameter structure provides details such as lens focal length, spatial resolution, sensor spectral sensitivities and quantization precision that differentiate one particular camera from another within the context of a fixed camera architecture. In our work, we have found that we rarely vary the camera architecture but regularly vary the camera parameters (see Section 5.)

To simulate a camera image from hyperspectral data, it is necessary to have more information than the scene radiances represented by each hyperspectral image plane. For example, to correctly account for variation in the distance between the simulated camera and the scene, it is necessary to know the distance at which the hyperspectral image was taken and the angular resolution of the hyperspectral camera. Thus we always specify images using a data structure that provides information beyond that contained in the numerical arrays that represent intensities or digital values. For hyperspectral images, this structure provides a description of the hyperspectral camera used to acquire the image, the distance to the scene, radiometric calibration data and, if available, the spectrum of the actual illuminant in the scene. The illuminant specification is not required for the simulation per se, but is useful for the implementation and evaluation of color image processing methods.

As the image is passed through the simulation pipeline, the information in the image structure changes. For example, the distance and radiometric information of the hyperspectral representation are no longer necessary, but it is important for subsequent image processing to know the simulated exposure duration, lens f-stop, and camera distance. Thus each function in the simulation pipeline modifies both the image data and the image structure that is required to interpret this data. This general approach to simulation provides great flexibility.

Below we describe in more detail a basic camera architecture and the individual functions that implement it. We used this architecture to simulate a Kodak DCS-200 color camera from a hyperspectral image.

3.1 BASIC CAMERA MODEL

We describe our basic camera model in terms of the five processing modules that implement its simulation pipeline. These are a) computation of responses for each sensor class at full spatial resolution, b) resampling the image from the hyperspectral resolution to the resolution of the simulated camera, c) simulation of the camera's sensor mosaic, d) injection of noise, and e) digitization. This order of operations does not precisely mimic the propagation of light and electrical signals in the real camera. For example, the computation of separate sensor class responses occurs physically after (or as) the camera sensors sample the image. The reason for the reordering is practical. It is computationally more efficient to reduce the number of image planes early in the pipeline rather than late. This reduction means that the resampling computations need not be applied to every plane of the hyperspectral image. For our simple model, the reordering does not affect the result of the computation. If one wanted to simulate the effects of chromatic aberration, on the other hand, the stages in the pipeline would have to be reordered. Such reordering is easily accommodated within the general simulation framework described above.

Clearly, the hyperspectral image data are themselves digital. For purposes of simulation, we do not compensate for the effects of this digitization. That is, we treat the hyperspectral data as a sufficiently accurate representation of the scene.

Computation of sensor class responses. Each sensor class is characterized by its spectral sensitivity. To compute the image plane corresponding to a sensor class at the resolution of the hyperspectral image, we simply compute a weighted sum of the hyperspectral image planes. For each sensor class, the weight for a given wavelength is the sensor sensitivity at that wavelength.

This approach assumes that any non-linearities in the camera sensors occur after the effects of light at different wavelengths have been combined linearly. This assumption is reasonable for human vision (16) and for many digital cameras (12). It does not require that the overall sensor response be linearly related to light intensity, as it allows for the possibility of a static non-linearity applied after integration of light intensity over wavelength.

In our simulations, we use spectral sensitivities obtained from direct calibration of a digital camera. Our calibration procedures (17) relate the linear sensor response to the radiance of a scene for a particular exposure and camera f-stop. When the camera is calibrated in this way, the response of a sensor class is independent of the distance of the camera to the scene. Thus to simulate a particular image, we need only take into account the simulated exposure duration and f-stop. This may be accomplished by multiplying the image planes computed as described above by a single scale factor. The sensor calibration procedures we used require measurement of camera response to monochromatic lights. When such lights are not available, other methods may be used to estimate sensor spectral sensitivities (18, 19).

Geometrical factors and spatial sampling. This stage of the simulation pipeline converts the images at the resolution and size of the hyperspectral input to the resolution and size of the simulated camera image. Our calculation assumes that the simulated camera acquires the image along the same line of sight as the original hyperspectral camera and that all objects in the scene lie in a plane perpendicular to the line of sight.

The angular resolution of a camera and its optical system is defined as the number of pixels that subtend a unit of visual angle. Given the distance of the camera to the scene D, and its angular resolution Φ (in pixels per degree), we have the following relation between the size X of an object in the scene and its size in pixels N on a camera sensor array:

$$\frac{X}{2D} = \tan\left(\frac{N}{2\Phi}\right). \tag{1.1}$$

In the simulation, there are two cameras involved: the original hyperspectral camera and the simulated camera. The relation between the images acquired by these two cameras is illustrated in Figure 1.7, where uppercase letters denote parameters for the hyperspectral camera and lowercase for the simulated camera. Using the geometrical relation described above, we obtain the relation between the number of pixels an image subtends for the hyperspectral image and the number it subtends for the simulated image:

$$n = 2\phi \arctan\left[\frac{D}{d}\tan\left(\frac{N}{2\Phi}\right)\right] \tag{1.2}$$

This relationship defines the resampling that should be applied to the hyperspectral image in order to map it onto the simulated image.

This stage of the simulation could also incorporate models of optical blur and off-axis optical effects. At present we do not do so.

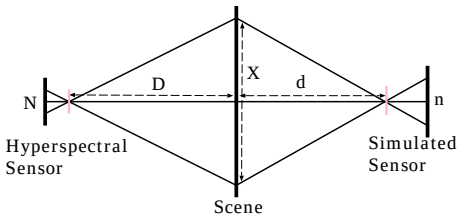


Figure 1.7 Geometrical factors. The figure shows how the same region of a scene would be imaged on two cameras with different angular resolutions placed at different distances from the scene. For illustrative purposes, the two cameras are shown on opposite sides of the scene.

Mosaicing. Many color cameras have only one type of sensor at each location, although other designs are possible. This means that a 1000x1000 RGB camera will typically have 1,000,000 total pixels rather than 3,000,000. Usually, the arrangement of the different sensor types is described by a small pattern that is repeated across the sensor array. For example one of the most popular patterns is the Bayer pattern (20), whose arrangement is shown in Figure 1.6. Our description of a simulated camera includes a specification of the periodic mosaic pattern. After we compute the image at the simulated resolution for each sensor type, we extract data according to the mosaic pattern. Our simulator also allows for simulation of non-mosaiced designs. In this case, no extraction is necessary.

Noise. To this stage in the simulation pipeline, the simulated image is noise free, in the sense that it has been computed deterministically from the scene radiance. For real cameras, the image will be variable due to noise. Photon noise, readout noise, variation in camera dark current, and noise in the analog-to-digital conversion process all distort the relation between camera data and the ideal values simulated thus far. In our current simulation, all of these noise sources are simulated by adding independent Gaussian noise to each simulated pixel value. The mean and variance of this added noise is chosen to describe the overall effect of the various individual noise sources. Obviously, it would be possible to develop more sophisticated noise models (10, 15).

Digitization. The actual output of digital cameras is digital. The last stage of our simulation pipeline simulates the analog-to-digital conversion. This quantization precision is specified by the number of bits provided by the A-D converters. For some cameras (e.g. the Kodak DCS-420), a static non-linearity is applied to the digital values before

they are available to the end-user. Such non-linearities can also be simulated at this stage.

3.2 SIMULATION PERFORMANCE

Figure 1.8 shows a comparison of real and simulated images for a Kodak DCS-200 color camera. This camera was used to acquire a real image of a scene. A hyperspectral image of the same scene was acquired using the system described in the previous section. The spectral sensitivities of the DCS-200 were calibrated by measuring the camera's responses to a series of monochromatic lights, as described in detail elsewhere (17). The angular resolutions of the hyperspectral and DCS-200 cameras were measured by acquiring images of objects of known size at known distances. The DCS-200 is known to have three sensor classes arranged in a Bayer mosaic pattern. With this information, we simulated the output of the DCS-200 using the pipeline outlined above. For the basic simulation, the simulated and real images were taken at the same distance from the scene. The figure shows a comparison for the G sensor class. Since the camera is mosaiced, bilinear interpolation was used to provide the missing values for both real and simulated images. Visual comparison shows good agreement between the two images.

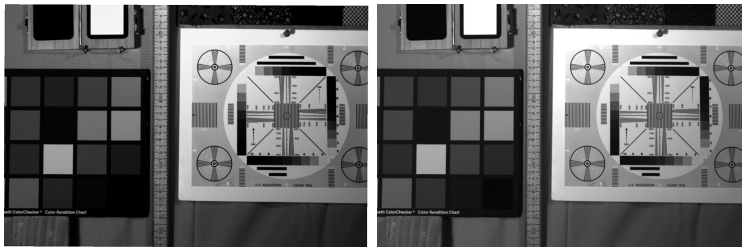


Figure 1.8 Simulator evaluation. The figure shows the G image plane of a real image acquired with a Kodak DCS-200 color camera and simulation of the same image obtained from hyperspectral input data. Bilinear interpolation was applied to create the displayed images from the mosaics of G sensor samples. RGB color images obtained by processing both the simulated and real images identically are available on the web (http://color.psych.ucsb.edu/simchapter/).

We consider more quantitative comparisons after we discuss the basic image processing operations necessary to display an image on a color monitor.

4. IMAGE PROCESSING

The raw output of a digital camera is not suited to direct display on a color monitor. First, the image is generally mosaiced, so that color

information is represented by interleaved samples in a single image plane. A demosaicing operation must be applied before the color image can be viewed. Second, because the spectral sensitivities of the camera's sensor classes do not match those of the human visual system, applying a color balancing algorithm to the camera's output will generally improve the fidelity of the displayed image. Third, even after color balancing, it is necessary to adjust the image values to take into account the calibration of the target monitor. We refer to this final step as color rendering. The division between color balancing and color rendering is not always clear, as some algorithms combine both functions into a single operation. The sequence of operations typically applied to a camera's output is shown in Figure 1.9. Other operations could also be applied. These include, for example, denoising and deblurring.

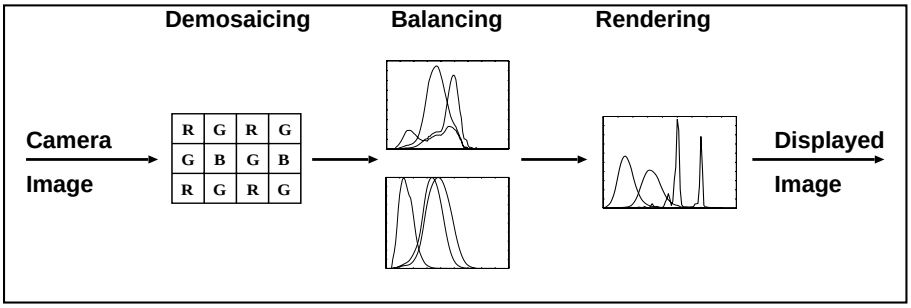


Figure 1.9 Image processing pipeline. Demosaicing. Typical camera images are mosaiced and a demosaicing algorithm is applied to estimate the missing sensor responses at each pixel. The figure shows the Bayer mosaic pattern. Balancing. Since the sensor spectral sensitivities of a typical digital camera are not generally matched to those of the human visual system, color balancing can improve the fidelity of the final displayed image. The figure shows the spectral sensitivities of the Kodak DCS-200 and those of the human cones. Both sets of sensitivities are used in color balancing. Rendering. The rendering step takes the properties of the target monitor into account. The figure shows the spectral power distributions of typical monitor phosphors.

In the following sections we review some basic approaches to each processing step.

4.1 DEMOSAICING

Since the displayed image must provide a color value for each pixel, the missing values need to be computed from the known values.[3]

[3] For non-mosaiced camera designs, this step is not necessary.

The most straightforward approach to demosaicing is to handle each sensor class separately. In this case, demosaicing becomes a matter of applying an interpolation algorithm that can handle samples that are not on a rectangular grid. A simple choice is to apply bilinear interpolation separately to the data from each sensor class. This is the baseline interpolation algorithm implemented in our simulator. Note that to apply linear interpolation, we must have access to the mosaic pattern of the camera. For this reason, our image processing pipeline takes the same data structures as the camera simulation pipeline. Indeed, there is no formal distinction between routines that simulate camera function and routines that apply processing operations to camera data. In each case both the input and output are images, and in each case the processing algorithm may modify the structure that describes the image data. For example, the demosaicing routines modify the structure to indicate that full color information is now available in the image.

Although straightforward, applying an interpolation algorithm separately to each sensor class can result in serious artifacts, especially in image regions of high spatial frequency (see Figure 1.10 for an example).

More sophisticated approaches to demosaicing are possible. One reason interpolating separately on each image plane is not optimal is that such procedures do not take advantage of the information that responses in one sensor class provide about responses in another. For a typical color camera, there is a fairly high correlation between the responses of the different sensor classes. This is illustrated in Figure 1.11 where the red sensor responses for a sample image are plotted against the green sensor responses for a Kodak DCS-200 color camera.

The fact that there is a high correlation between the responses of different sensor classes suggests that better demosaicing may be achieved by using an algorithm that treats all of the sensor classes simultaneously. A number of methods for doing this have been proposed (21, 22, 23, 24, 25). Freeman's method (21) is based on heuristics that eliminate color artifacts near luminance edges. Brainard and colleagues (24, 25) developed a Bayesian method that estimates missing pixel values based on neighboring values in both space and color. The simulator provides a tool for comparing the performance of these algorithms.

4.2 COLOR BALANCING AND COLOR RENDERING

Given a demosaiced image, we would like to display the image on a monitor so that its appearance is similar to that of the original scene. To understand how this may be done, we start with the case where the

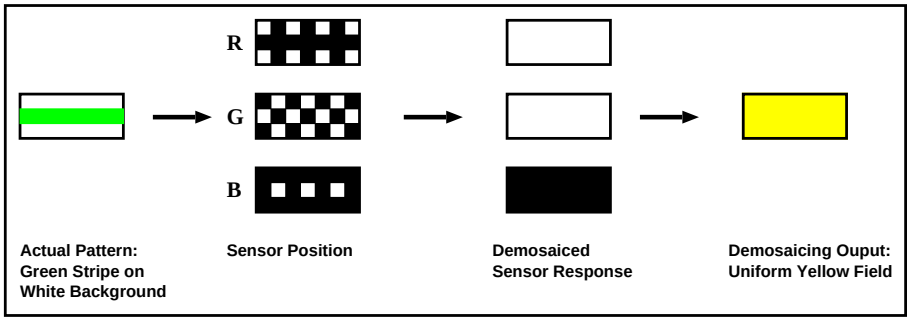


Figure 1.10 Color artifacts due to independent interpolation. Consider an image that consists of a thin green stripe on a white background. The thickness of the stripe when imaged on the camera's sensor array is equal to the size of one pixel. This is illustrated on the left of the grayscale figure by the middle gray stripe. The camera contains three interleaved sensor classes. The locations of the sensors are indicated in the second column of the figure, with sensor presence indicated by white and sensor absence indicated by black. The third column shows the demosaiced sensor responses to the scene. White image regions are seen by all the red pixels. For this reason, the output of all the red pixels is uniformly high. Similarly, green pixels see either white or green image regions and so their output is also uniformly high. Blue pixels, on the other hand, are not driven by the green stripe and their output is uniformly low. Applying interpolation to uniform samples yields a uniform output for each sensor class, with the red and green outputs being high and the blue output being low. The resulting reconstruction of the green stripe is thus a yellow uniform field, which is distorted both in terms of its spatial structure and in terms of its color. Although this example was tailored to produce an artifact, the type of effect illustrated is common for real cameras. The web site (http://color.psych.ucsb.edu/simchapter/) shows a color example from a simulated image to which bilinear demosaicing has been applied. A color version of this figure is also available on the web.

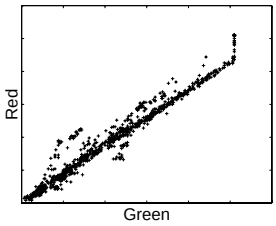


Figure 1.11 Correlation of red and green sensor responses. The figure plots the red response at each pixel against the green response at the same pixel. Data were obtained by simulating a non-mosaiced version of the Kodak DCS-200 from hyperspectral input. It is clear the responses of the two sensor classes are highly correlated.

camera's sensors have the same spectral sensitivities as the cone photoreceptors in the human eye. In this case, the camera data tell us how strongly the light from the scene would have stimulated the cones had an observer been viewing the same scene. To produce a colorimetrically accurate rendition of the camera data, a reasonable principle is to set the monitor inputs so that the emitted light produces the same effect on the camera's sensors as did the original scene. This principle is illustrated in Figure 1.12 and is fairly straightforward to achieve. Note that for this principle to work perfectly, the original scene and rendered image must be viewed in the same context. This ensures that the visual system is in the same state of adaptation when the two images are viewed.

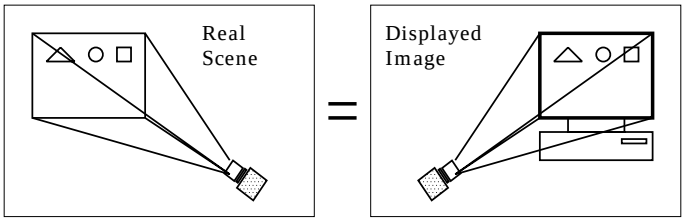


Figure 1.12 Basic rendering principle. Set the monitor image so that it has the same effect on the camera as did the original scene. When the camera's spectral sensitivities are within a linear transformation of those of the human cones, this principle produces an image that is colorimetrically correct for a human observer.

The spectrum emitted at a particular monitor pixel is obtained from the monitor settings at that pixel and the characteristic spectrum of each monitor phosphor. Thus, the linear response (before digitization) produced by in each camera sensor class is given by the following matrix equation:

$$r_C = T_C B_M w_M \tag{1.3}$$

In this equation, r_C is a column vector containing the responses of each sensor class, w_M is a column vector containing the monitor settings, T_C is a matrix whose rows specify the camera sensor spectral sensitivities at a set of discrete sample wavelengths, and B_M is a matrix whose columns specify the spectral power distribution of each monitor phosphor at the same sample wavelengths. Thus the column dimension of T_C and the row dimension of B_M are equal to the number of wavelength samples N_λ used to discretize spectral functions. The use of a vector or matrix representation for this sort of calculation is becoming standard in computational colorimetry (26). In Equation 1.3 we assume that the spacing between wavelength samples has been folded into the units that specify light power.

To produce stimuli on the monitor that will look the same to the camera as did the original scene, we invert the matrix equation given above. Thus, given a camera response, the corresponding pixel settings on the monitor are given by the following:

$$w_M = (T_C B_M)^{-1} r_C \tag{1.4}$$

Note that the w_M represent linearized monitor settings, rather than the actual digital values stored in the display hardware. CRT monitors produce output for each phosphor whose intensity is not linearly related to the input voltage, thus giving a non-linear relation between output light intensity and digital setting. Correcting for this non-linearity is referred to as gamma correction and is easily accomplished by table lookup (27).

The approach described above provides a baseline rendering algorithm. It can be shown that this algorithm will produce a colorimetrically correct rendering when the camera's sensor spectral sensitivities are within a linear transformation of the spectral sensitivities of the human cones (8, 28, 26). Although the algorithm may still be applied when this condition does not hold, it will not in general produce a colorimetrically correct rendering (8, 28, 26). It is useful, therefore, to explore more sophisticated algorithms that take the camera sensitivities, the sensitivities of the human cones, and statistical properties of the scene into account. One general approach is to use the camera data to estimate how the human cones would have responded had they viewed the original scene. Once this estimate is in hand, the basic rendering algorithm described above may be applied using the estimates rather than the raw camera data as input.

A complete review of balancing and rendering is beyond the scope of this chapter. It is instructive however, to review one line of thinking that has emerged from consideration of the computational problem of color constancy (29, 26, 30, 18, 31). In this approach, the camera data are first used to estimate the illuminant spectral power distribution and surface reflectance functions of the scene. These in turn are used to predict how the human cones would have responded to the same scene. The key idea that makes this approach possible is that in natural scenes, illuminant spectral power distributions and surface reflectance functions may be represented using small-dimensional linear models.

For fixed viewing geometry, the spectral power distribution of the light reaching the camera is obtained by the wavelength-by-wavelength product of the illuminant power e and surface reflectance spectra s. Here again we use column vectors to represent discretized versions of the underlying spectral functions. Thus if c is the spectral power distri-

bution of the light reaching the camera, $c = diag(e)s$, where the function $diag(x)$ produces a diagonal matrix with the entries of x as the diagonal elements.

Consider a collection of surface reflectances s_i. If we can define a N_λ by N_s matrix B_s such that each s_i may be written as:

$$s_i = B_s w_i \tag{1.5}$$

where w_i is an N_s-dimensional column vector, then we say that the surfaces s_i lie within an N_s-dimensional linear model. The columns of B_s may be regarded as functions of wavelength and are called the basis functions of the model. Analyses of measured surface reflectance spectra indicate that naturally occuring surfaces lie within small-dimensional linear models and that a reasonable approximation may be obtained when the dimension is as small as three (32, 33, 34, 35). It also appears that naturally occuring illuminant spectra lie within small-dimensional linear models (36, 37).

Suppose that we know the illuminant spectral power distribution e and also that the surface reflectances lie within a three-dimensional linear model. Then the light reaching the camera may be obtained as

$$c = diag(e)B_s w. \tag{1.6}$$

If we take the camera sensors T_C into account, we have

$$r_C = T_C diag(e)B_s w = M w, \tag{1.7}$$

where M is a three-by-three matrix. We can invert this equation to obtain w from r_C. Given the surface weights, we can then synthesize how the human cones would have responded to the same scene through Eq. 1.3 with T_C replaced by T_H, the spectral sensitivities of the human cones. This synthesized cone image may then be passed to the basic rendering method described above.

Of course, in many practical applications, the illuminant is not known. A number of algorithms have been developed for estimating the illuminant from camera data (38, 39, 40, 41, 29, 30, 18, 31). By applying one of these algorithms to obtain an illuminant estimate, we substitute this estimate in the procedure described above.

Also note that when computing the human cone responses, it is not necessary to re-render the surfaces with respect to the same illuminant that was present in the scene. If the illuminant under which the image was acquired is biased away from daylight, a more pleasing result may sometimes be obtained by re-rendering with an illuminant closer to daylight. Such re-rendering may be particularly effective if there are differences in the viewer's state of adaptation as the original scene and the rendered image are viewed. These might be expected to occur, for

example, if the monitor is viewed in a room where the illuminant differs markedly from that of the original scene. Methods for taking adaptation into account as part of color balancing and rendering are a topic of active research interest. One approach to this problem is to incorporate a color appearance model (42, 43) into the balancing algorithm (13).

4.3 EVALUATION OF IMAGE QUALITY

Camera simulation enables comparison of different camera designs and processing techniques. But to make the evaluation quantitative we need an image quality metric. One approach is to compare images in the camera sensor representation. Such an approach has the advantage that it may be used without the necessity for color balancing and rendering. On the other hand, this approach will not work when comparing the performance of cameras with different sensor spectral sensitivities, nor is it likely to capture image quality as judged by a human observer. An alternative, which we favor, is to compare images as they are rendered for display. Rendered images have the feature that their effect on the human visual system may be computed, as they are specified in terms of the light reaching the eye from a monitor at each image location. Working with rendered images allows us to compare the output of different cameras and processing algorithms in a common representation. It also allows us to take advantage of image difference metrics that have been developed for human vision.

S-CIELAB image metric. The development of metrics suitable for comparison of color images is an area of active research. For our work here we use the CIELAB color metric when we consider uniform image regions and the S-CIELAB metric when we consider comparisons of images.

The CIELAB metric was introduced by the *Commission Internationale de l'Éclairage* (44) in order to standardize specification of color tolerances or color differences. Although its correlation with perceptual judgements is not perfect, it is a reasonable choice when the influence of image spatial structure on perceived color is small (43).

The S-CIELAB metric is an extension of CIELAB that takes image structure into account (45, 46). The S-CIELAB calculation takes as input two images and produces as output a difference image whose magnitude indicates the perceptual difference at each pixel. This difference image may be summarized by the mean difference taken over pixels. Note that the S-CIELAB metric is designed to take spatial structure into account when evaluating the perceptual magnitude of color diffe-

rences, not to evaluate the perceptual magnitude of spatial differences between images.

Both CIELAB and S-CIELAB metrics are designed to work on inputs where color information is provided in terms of 1931 CIE XYZ tristimulus coordinates. We obtain these values by converting images rendered for a monitor to XYZ coordinates. For the basic rendering method this means that the XYZ values depend on the choice of monitor primaries if the original camera sensors were not within a linear transformation of the color matching functions. This dependence is a fundamental feature of the basic rendering algorithm, in the sense that the output of this algorithm itself depends on the primaries of the target monitor. Here we use primaries obtained from measurements of an Apple 20" monitor in our lab.

In evaluations of image quality, we have not taken potential differences in image size into account. That is, we are implicitly assuming that any differences in image size between the original scene and the rendered image may be neglected.

As an example where an image metric is useful, we can evaluate the difference between the real and simulated images shown in Figure 1.8. That figure shows that the spatial structure of the real and simulated images is quite similar. Recall that the S-CIELAB metric, however, is not designed to evaluate the perceptual effect of the small differences in spatial structure that are inevitable between the real and simulated images. To compare the color differences between these two images, we computed a block average of the real and simulated images and then applied the CIELAB color metric independently to each block. Figure 1.13 shows a histogram of the CIELAB Δ_{E*} errors obtained in this way. The median Δ_{E*} error is 6.0. For comparison, just noticeable differences under optimal comparison conditions correspond to a range of Δ_{E*} values of between 0.4 and 5 with a mean value of about 2 (43). Thus our simulation errors are likely to be perceptible, but not highly so. We regard the agreement between real and simulated images as quite good. A number of factors may contribute to the error. First, the hyperspectral data itself is not perfectly accurate, particularly in the luminance dimension (see Figures 1.4 and 1.5). Second, our simulation pipeline does not include terms for off-axis effects in the simulated camera. Finally, the real camera's spectral sensitivities may diverge from the estimates we obtained.

Note that for purposes of evaluating camera and algorithm performance, it is not crucial that the agreement between real and simulated images be perfect. Rather what is important is that the simulation ac-

curately model the effect of camera design and algorithm variation on the output image.

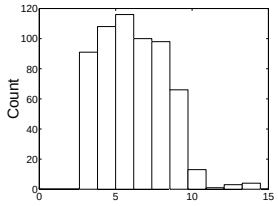


Figure 1.13 Numerical comparison of real and simulated images. The histogram shows the CIELAB Δ_{E*} errors obtained between the real and simulated images shown in Figure 1.8. The CIELAB differences were obtained from block averaged versions of the two images. The block size was 50 pixels by 50 pixels. Before block averaging, the small spatial differences between the two images were further minimized by applying our image registration algorithm (see Section 2.) The median Δ_{E*} error is 6.0.

Generation of a comparison image. For some applications, it is useful to establish an ideal image against which rendered images may be compared. For this purpose, we simulate a non-mosaiced camera that has sensor spectral sensitivities matched to those of the human visual system. This produces an image that has the same spatial resolution as the camera image but that does not contain mosaic or color correction artifacts. In the context of the camera simulator, producing such an image is straightforward. We simply specify a camera with the appropriate sensors and without mosaicing. For the examples reported in this chapter, the ideal image is computed without added noise but with the same 8-bit quantization as used in the simulated cameras. It is possible to consider ideal images with finer quantization or higher spatial resolution, but we do not do so here. Note that it is possible to compute the ideal image because we start with hyperspectral rather than RGB image data.

There are a number of standard color spaces that may be used to represent the response of the human visual system. For example, we can compute the CIE 1931 tristimulus coordinates at each pixel of the image. This system of colorimetry is based on a large set of trichromatic color matches. Alternatively, we can represent the image in terms of the responses of the L, M, and S cone photoreceptors. This system of colorimetry is based on our understanding of the physiological basis for trichromacy. For our purposes, either system is acceptable, since the L, M, and S cone spectral sensitivities are well-approximated by a

linear transformation of the CIE 1931 color matching functions. In the examples below we use the LMS system to produce the ideal images.

5. EXAMPLES

In this section, we provide two simple examples to illustrate how the simulator may be used. In the first example, we consider the effect of varying the camera sensor spectral sensitivities on image quality. In the second example, we compare the efficacy of two mosaicing architectures. One uses three sensor types arranged in a Bayer mosaic pattern. The other uses four sensor types arranged in a rectangular grid. The examples are intended to be illustrative rather than definitive.

Effect of sensor choice. In the previous section, we discussed two methods of rendering a camera image on a color monitor. The first method (baseline) produced an image on the monitor that has the same effect on the camera as the original scene (see Figure 1.12). Given a particular target monitor, how the human visual system would have responded to the rendered image may be computed from the monitor's output. The second method (illuminant-based) takes advantage of additional information about the illuminant to estimate the spectral properties of the surfaces in the scene and then uses the estimate to predict how the human visual system would have responded to the original scene. The monitor image is then computed to produce this same response. To implement the illuminant-based method, we used a direct measurement of the scene illuminant. For many applications such a measurement would not be available and it would be necessary to estimate the illuminant from the image data.

Here we compare the performance of these two methods as a function of the camera's spectral sensitivities. When the camera's sensors are within a linear transformation of the human cone spectral sensitivities and there is no noise, it can be shown that the two rendering methods both yield a perfect match to the comparison image (8, 28, 26). As the camera's sensor spectral sensitivities diverge from those of the human cones, the two methods give different results from each other and, in general, do not match the comparison image.

We use a simple one parameter method to vary the camera spectral sensitivities. Let $R_h(\lambda)$, $G_h(\lambda)$, and $B_h(\lambda)$ be the spectral sensitivities of the human cones. Let $R_c(\lambda)$, $G_c(\lambda)$, and $B_c(\lambda)$ be the spectral sensitivities of the Kodak DCS-200 color camera, as measured by Vora et al. (17). We obtain a one parameter family of spectral sensitivities $R(\lambda, \alpha)$, $G(\lambda, \alpha)$, and $B(\lambda, \alpha)$ by mixing: $R(\lambda, \alpha) = (1 - \alpha)R_h(\lambda) + \alpha R_c(\lambda)$ and similarly for $G(\lambda, \alpha)$ and $B(\lambda, \alpha)$, where $0 \leq \alpha \leq 1$.

We chose a series of values for α in the range $[0, 1]$ and simulated the output of a camera with the corresponding spectral sensitivities. The simulated camera spatial parameters were those of a Kodak DCS-200, so that simulated image had the spatial structure shown in Figure 1.8. In each case, we rendered the image with each of the two algorithms discussed above. For the illuminant-based method, we used a three-dimensional linear model for surfaces computed from measurements of the reflectance spectra of Munsell papers (47, 48, 33, 34). For rendering purposes, we used calibration data for an Apple 20" color monitor. Both the camera and the monitor were taken to have a precision of 8-bits per pixel. We examined the no noise case and two levels of simulated noise. The standard deviations of the Gaussian noise were 2 and 5 units in a representation where the camera range was 0-255. Monitor output was converted back to an image with respect to the human visual system. We computed the mean S-CIELAB error between the rendered image and a noiseless comparison image generated as described above.

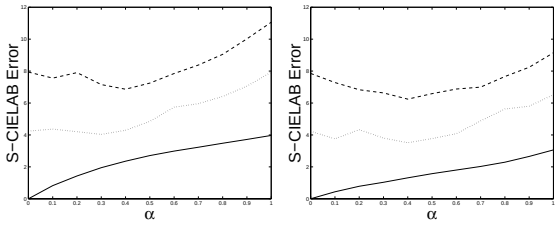


Figure 1.14 S-CIELAB error for two rendering methods. The left panel shows the S-CIELAB error for the baseline rendering method. The right panel shows the S-CIELAB error for the illuminant-based method. Each panel plots error as a function of α. For $\alpha = 0$ the camera's sensors match those of the human visual system. For $\alpha = 1$ the camera's sensors match those of the Kodak DCS-200. Intermediate values of α correspond to mixtures of the two sets of sensors. Individual curves correspond to different levels of simulated noise (standard deviations: 0, 2, and 5).

Figure 1.14 plots the mean S-CIELAB error as a function of α for the two methods and different noise levels. For $\alpha = 0$, both methods provide perfect performance in the no noise case. This is because for $\alpha = 0$ the camera sensors have the same spectral sensitivities as the human cones and both methods are guaranteed to produce the correct result. As α increases, the deviation between the camera's sensor spectral sensitivities and the human cones increases. We see that in both cases the reproduction error goes up. The illuminant based method is slightly less sensitive to divergence in sensor spectral sensitivities from those of the human visual system. The same basic relation between the two rendering methods is observed when the images are noisy. Interest-

ingly, the function relating image quality to α is not monotonic when noise is added. We have not explored this effect further.

Although the illuminant-based rendering method provides performance superior to that of the basic method, note that the illuminant-based method requires additional information, namely the spectral power distribution of the illuminant. In the present example, that information was available. For more general application, it is necessary to estimate the illuminant from the image data. This estimation step will presumably introduce additional rendering error. Exploring the efficacy of illuminant estimation algorithms is one of our goals for future work.

Effect of varying the mosaic. Typical color cameras have three classes of color sensor. This number is often justified on the grounds that human vision is itself trichromatic. As the example above illustrates, however, if the camera sensors are not within a linear transformation of the human cones, color reproduction error is introduced. Intuitively, this error can be reduced by using more than three sensor classes, as this increases the spectral information available in the camera image. For a mosaiced design, however, the increase in spectral information is purchased at a cost of reduced spatial information for each channel, as four sensors must be packed into the same area as are three for the RGB design.

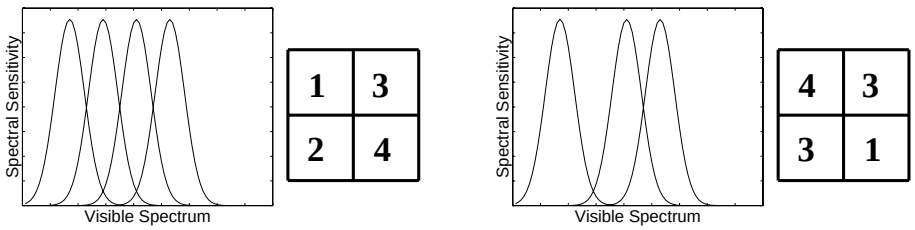


Figure 1.15 Spatial and spectral patterns used for comparison. In one case (left panels) four sensor classes, whose spectral sensitivities span the visible spectrum with gaussians sensitivities of 27 nm standard deviation, are packed on a regular square grid. From left to right we refer to these as sensor classes 1, 2, 3, and 4. In the other case (right panels), three sensor classes are packed in a Bayer pattern, the second sensor class (2), whose spectral sensitivity was centered on 520 nm, is not used.

To better understand this tradeoff, we compare the performance of a three sensor and four sensor design. In each case, we chose sensors with Gaussian spectral sensitivities with standard deviation of 27 nm. The three and four sensor spectral sensitivities are shown in the right and left panels of Figure 1.15. For the three sensor design, we packed these sensor types using the Bayer pattern. For the four sensor design we used

a regular square grid (see Figure 1.15). We then simulated each camera and demosaiced the output using a Bayesian algorithm developed by Brainard and colleagues (24, 25). The Bayesian demosaicing algorithm takes advantage of correlations between the responses of different sensor classes. For images whose statistics are Gaussian and known, it produces an optimal (in the expected squared error sense) reconstruction from an arbitrary mosaic pattern. Although real images are not described by Gaussian statistics (49) the algorithm still provides reasonable performance. In applying the algorithm, the image statistics are obtained by analyzing the distribution of sensor responses in a bilinearly demosaiced version of the image. After demosaicing, we rendered the images using the illuminant based method described above. For the four sensor design, a four rather than three dimensional linear model for surfaces may be used with this algorithm.

Table 1.16 shows the mean S-CIELAB error for the two designs as a function of the simulated noise level. We see that for this image, the four sensor design provides lower error than the three sensor design for low noise, but that when the noise level increases the three sensor design provides better performance. At low noise levels, the extra color information provided by the fourth sensor presumably dominates the loss of spatial information that results from the addition of a fourth sensor to the mosaic pattern. At higher noise levels, mosaic artifacts become more important and the three sensor design does relatively better. We note that this is a preliminary evaluation. Tailoring demosaic and rendering algorithms precisely to the image statistics and noise level is a subtle matter and it is possible we have not yet optimized all of the demosaicing parameters. Still, the example illustrates the types of questions we can ask and the sorts of answers we can obtain through simulation.

# Sensors \ Noise σ	0	2	5
3 sensors	2.37	2.67	4.04
4 sensors	1.09	1.99	4.16

Figure 1.16 S-CIELAB error for two different mosaic patterns and sensor sets. As described in the text, the table provides the S-CIELAB error for two camera designs at several noise levels.

6. DISCUSSION

In this chapter, we described our hyperspectral imaging system and showed how hyperspectral data may be used to predict how a real digital camera would have responded to the same scene. To do so, we developed a basic camera model. Although the model is simple, it is sufficiently accurate to provide satisfactory agreement between real and simulated images. We also discussed the image processing steps required to render camera data on a display device.

Simulation of digital cameras is useful for evaluating camera designs and image processing algorithms. To make this point, we compared the peformance of two color rendering algorithms as a function of camera sensor spectral sensitivities and noise level. We also compared performance of three and four sensor camera designs. The examples serve to illustrate the type of questions that may be addressed using the camera simulator.

A number of interesting directions remain for future research. First is the development of more sophisticated camera models. In our view, the most important extension is to model the individual components (lens, sensor chip) of the camera at a more microscopic level. This should lead to more accurate simulation of geometric effects and better models of camera noise. See (15, 14) for descriptions of how such modelling might proceed. Extensions of this sort will be necessary before we can address questions such as how, for fixed sensor array size, the improvement in signal-to-noise ratio obtained by increasing pixel size trades off against the loss of resolution caused by the same change.

We would also like to improve our hyperspectral imaging system. The main limitation of the system described here is that it is very slow. This is both because filters must be inserted manually in the optical path and because the particular CCD camera we use requires fairly long exposure times. With our current system, it takes about one hour to acquire a hyperspectral image. Thus we can only take images of indoor scenes containing inanimate objects. We are working to develop a system that can acquire similar data in one minute or less. With the new system we hope to be able to acquire images of natural outdoor scenes.

Acknowledgments

This work was supported by a research grant from the Hewlett-Packard corporation and a Eurodoc fellowship from the Région Rhone-Alpes (France).

We thank E. Adelson, P. Catrysse, J. Farrell, W. Freeman, E. Harding, R. Iimura, J. Kraft, J. Tietz, B. Wandell, and X. Zhang for useful discussions and technical

help. D. Heeger provided the implementation of the original registration algorithm. X. Zhang provided the implementation of the S-CIELAB image metric.

References

[1] G. Brelstaff, A. Parraga, T. Troscianko, and D. Carr. "Hyperspectral camera system: acquisition and analysis". In *Proc. of SPIE Conference on Geographic Information Systems, Photogrammetry, and Geological/Geophysical Remote Sensing*, pp. 150–159, 1995.

[2] C.A. Parraga, G. Brelstaff, T. Troscianko, and I.R. Moorehead. "Color and luminance information in natural scenes". *J. of the Optical Society of America A*, Vol. 15, pp. 563–569, 1998.

[3] D.L. Ruderman, T.W. Cronin, and C. Chiao. "Statistics of cone responses to natural images: implications for visual coding". *J. of the Optical Society of America A*, Vol. 15, pp. 2036–2045, 1998.

[4] E. Hecht. *Optics*. Addison-Wesley, 1997.

[5] J.R. Bergen, P. Anandan, K. Hanna, and R. Hingorani. "Hierarchical model-based motion estimation". In *Proc. of the Second European Conference on Computer Vision*, pp. 237–252, 1992.

[6] D.J. Heeger. *Notes on Motion Estimation*. Stanford University, 1996. http://white.stanford.edu:80/~heeger/registration.html.

[7] M.A. Webster and J.D. Mollon. "Adaptation and the color statistics of natural images". *Vision Research*, Vol. 37, pp. 3283–3298, 1997.

[8] B.K.P. Horn. "Exact reproduction of colored images". *Computer Vision Graphics and Image Processing*, Vol. 26, pp. 135–167, 1984.

[9] R. Kingslake. *Lens Design Fundamentals*. Academic Press, 1978.

[10] G.E. Healey and R. Kondepudy. "Radiometric CCD camera calibration and noise estimation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 3, pp. 267–276, 1994.

[11] G.C. Holst. *Sampling, Aliasing, and Data Fidelity for Electronic Imaging Systems, Communications, and Data Acquisition*. SPIE, 1998.

[12] P.L. Vora, J.E. Farrell, J.D. Tietz, and D.H. Brainard. "Digital color cameras - 1 - response models". Technical Report HPL-97-53/54, Hewlett Packard, 1997. http://www.hpl.hp.com/techreports/97/HPL-97-53.ps.

[13] B.A. Wandell and L.D. Silverstein. "Color technology and color reproduction", In *The Science of Color, 2nd Edition*, (S. Shevell, ed.), Optical Society of America, 2000.

[14] R. Short, D. Williams, and A.E.W. Jones. "Image capture simulation using an accurate and realistic lens model". In *SPIE Proc.: Sensors, cameras, and applications for digital photography*, Vol. 3650-20, pp. 138–148, January 1999.

[15] P.B. Catrysse, B.A. Wandell, and A. El Gamal. "Comparative analysis of color architectures for image sensors". In *SPIE Proc.; Sensors, cameras, and applications for digital photography*, Vol. 3650-04, pp. 26–35, 1999.

[16] B.A. Wandell. *Foundations of Vision*. Sinauer Associates, 1995.

[17] P.L. Vora, J.E. Farrell, J.D. Tietz, and D.H. Brainard. "Digital color cameras - 2 - spectral response". Technical Report HPL-97-54, Hewlett Packard, 1997.

[18] K. Barnard. *Pratical Color Constancy*. PhD Thesis, Simon Fraser University, 1999.

[19] G. Sharma and H.J. Trussel. "Characterization of scanner sensitivity". In *Proc. of IS&T/SID Color Imaging Conference*, pp. 103–107, 1993.

[20] B.E. Bayer. "Color imaging array. U.S. Patent # 3,971,065, 1976.

[21] W.T. Freeman. "Method and apparatus for reconstructing missing color samples. U.S. Patent # 4,663,655; Polaroid Corporation, 1987.

[22] D.R. Cok. "Reconstruction of ccd images using template matching". In *Proc. of IS&T 47th Annual Meeting*, pp. 380–385, 1994.

[23] M.A. Wober and R. Soini. "Method and apparatus for recovering image data through the use of a color test pattern. U.S. Patent #5,475,769; Polaroid Corporation, 1995.

[24] D.H. Brainard. "Bayesian method for reconstructing color images from trichromatic samples". In *Proc. of IS&T 47th Annual Meeting*, pp. 375–380, 1994.

[25] D.H. Brainard and D. Sherman. "Reconstructing image from trichromatic samples : from basic research to practical applications". In *Proc. of IS&T/SID Color Imaging Conference*, pp. 4–10, 1995.

[26] D.H. Brainard. "Colorimetry", In *Handbook of Optics: Volume 1. Fundamentals, Techniques, and Design.*, (M. Bass, ed.), McGraw-Hill, 1995.

[27] D.H. Brainard. "Calibration of a computer controlled color monitor". *Color Research and Application*, Vol. 14, pp. 23–34, 1989.

[28] B.A. Wandell. "Color rendering of camera data". *Color Research and Application*, Vol. Sup. 11, pp. S30–S33, 1986.

[29] L.T. Maloney. "Color constancy and color perception: The linear models framework.", In *Attention and Performance XIV: Synergies in Experimental Psychology, Artificial Intelligence, and Cognitive Neuroscience*, MIT Press, 1992.

[30] A. Hurlbert. "Computational models of color constancy", In *Perceptual Constancies*, (V. Walsh and J. Kulikowski, ed.), Optical Society of America, 1998.

[31] L.T. Maloney. "Physics-based approaches to modeling surface color perception.", In *Color Vision, From Genes to Perception*, Cambridge University Press, 2000.

[32] E.L. Krinov. "Spectral reflectance properties of natural formations". In *Technical Translation: TT-439*. National Research Council of Canada, 1947.

[33] J. Cohen. "Dependency of the spectral curves of the Munsell color chips". *Psychonomic Science*, Vol. 1, pp. 369–370, 1964.

[34] J.P.S. Parkkinen, J. Hallikainen, and T. Jaaskelainen. "Characteristics spectra of munsell chips". *J. of the Optical Society of America A*, Vol. 6, pp. 318–322, 1989.

[35] L.T. Maloney. "Evaluation of linear models of surface spectral reflectance with small number of parameters". *J. of the Optical Society of America*, Vol. A 3, pp. 1673–1683, 1986.

[36] D.B. Judd, D.L. MacAdam, and G. Wyszecki. "Spectral distribution of typical daylight as a function of correlated color temperature". *J. of the Optical Society of America*, Vol. 54, pp. 1031–1040, 1964.

[37] CIE. "Colorimetry". Technical report, Bureau Central de la CIE, 1986.

[38] G. Buchsbaum. "A spatial processor model for object colour perception". *J. Franklin Inst.*, Vol. 310, pp. 1–26, 1980.

[39] L.T. Maloney and B.A. Wandell. "Color constancy : a method for recovering surface spectral reflectance". *J. of the Optical Society of America A*, Vol. 3, pp. 29–33, 1986.

[40] D.H. Brainard and W.T. Freeman. "Bayesian color constancy". *J. of the Optical Society of America*, Vol. A 14, pp. 1393–1411, 1997.

[41] G.D. Finlayson, P.M. Hubel, and S.D. Hordley. "Colour by correlation". In *Proc. of IS&T/SID Color Imaging Conference*, pp. 6–11, 1997.

[42] M.D. Fairchild. *Color Appearance Models.* Addison-Wesley, 1997.

[43] D.H. Brainard. "Color specification", In *The Science of Color, 2nd Edition*, (S. Shevell, ed.), Optical Society of America, 2000.

[44] CIE. "Recommendations on uniform color spaces, color difference equations,and psychometric terms". Technical Report Supplement No.2 to CIE publication No.15, Bureau Central de la CIE, 1978.

[45] X. Zhang. *Spatial color fidelity metric S-CIELAB*. PhD Thesis, Stanford University, 1997.

[46] X. Zhang and B.A. Wandell. "A spatial extension of CIELAB for digital color image reproduction". *Society for Information Display Journal*, Vol. 5, pp. 61–64, 1997.

[47] K.L. Kelly, K.S. Gibson, and D. Nickerson. "Tristimulus specification of the munsell book of color from spectrophotometric measurements". *J. of the Optical Society of America*, Vol. 33, pp. 355–376, 1943.

[48] D. Nickerson. "Spectrophotometric data for a collection of munsell sample". Technical report, U.S. Department of Agriculture, 1957.

[49] E.P. Simoncelli. "Bayesian denoising of visual images in the wavelet domain", In *Bayesian Inference in Wavelet Based Models*, Springer-Verlag, 1999.