Debayering Demystified by Craig Stark, PhD

When choosing a camera, we often have the option of a monochrome or a color version of the camera. There are pros and cons to each. Monochrome cameras are usually touted as being more sensitive, more flexible, and produce higher resolution images than their one-shot color counterparts. One-shot color cameras are usually touted as being much simpler to use if your final output is to be a color image. All of these are true, at least in the limit. To understand how much of a performance hit one takes with one-shot color cameras, we must first understand how a color image is made.

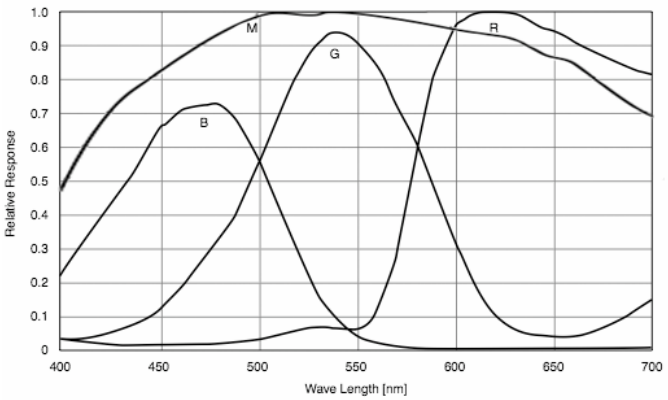

Figure 1: Relative response to light of different wavelengths from the Sony ICX285 chips. The monochrome CCD (ICX285AL) has pixels that all respond to a wide range of wavelengths (M). In contrast, the color CCD (ICX285AQ) has pixels tuned to red (R), green (G), and blue (B) portions of the spectrum. Note that the response plotted is relative to the maximum response generated by any pixel at any wavelength. The color CCD is not in truth more sensitive at 650 nm than the monochrome CCD.
(Curves reproduced from Sony datasheets)

Color Imaging on a Monochrome Camera

Monochrome cameras have pixels that respond well to light from a broad spectrum. In Figure 1, the line marked "M" shows the response of each pixel in a Sony ICX285AL CCD chip found in a number of cameras. There is no color information recorded at each pixel. Rather, each pixel records how many photons struck it with the odds of recording a photon being higher at 550 nm than at 400 nm. Thus, if we take a single image with this CCD chip, we will have a black and white picture.

To get a color picture, the standard technique is to take three separate images with this same sensor. If we put a red filter over the CCD, each pixel would have a response profile not too dissimilar from the "R" line shown in Figure 1. In so doing, we would have an image that depicts "how much red there is" at each pixel. Do this again with a green filter and a blue filter and we know how much red, green, and blue there is in each portion of our image. Combine these three images and we would have a color picture. This technique, called RGB imaging is not only how many amateurs take color pictures of astronomy-related targets, but also how very high-end color cameras work (they typically have 3 sensors, each effectively covered by a separate color filter). Thus, to generate a color image, three times as many frames are collected as would be used in a monochrome image (in a variant on this, LRGB imaging, one uses four frames, adding a "luminance" frame, typically taken with no filter.) It is worth noting strongly here that the choice of color filters used is entirely up to you. For example, one could use an H-alpha filter in place of the red filter or an O-III in place of the blue filter, etc. and synthesize a color frame in many different ways (thus, the flexibility aspect of monochrome cameras).

Color Imaging on a One-shot Color Camera: The Bayer Matrix

By far – by many orders of magnitude, this is not how most color images are made, however. Most color images are made using one-shot color cameras that make a color image in one pass using a single CCD or CMOS sensor. Point and shoot digital cameras, digital SLRs, camcorders, webcams, etc. all use one-shot color sensors that create a color picture in a single pass. They do it by using some variant of the Bayer Matrix (Figure 2) and the mathematical techniques of "debayering" or "demosaicing" the image.

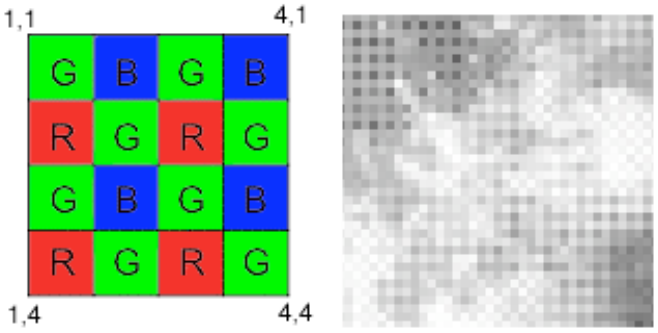

Figure 2: (left) A typical Bayer Matrix for a small 4x4 area of pixels. Each pixel codes for red, green, or blue components of the image. (right) A portion of the M1 image (see below) encoded as a Bayer matrix (i.e., a "RAW" image). Prior to debayering, data from one-shot color cameras are fundamentally grayscale images.

continued ⇨

Debayering Demystified *continued*

In the Bayer matrix (named after Dr. Bryce Bayer of Eastman Kodak), a single color filter is placed over each pixel in the mosaic pattern shown in Figure 2. The filters are similar those used in RGB imaging on a monochrome camera, but the difference lies in the patchwork nature of the matrix. In RGB imaging with a monochrome camera, a red filter is placed over all pixels for one image, a green filter for the next, and a blue for a third. Thus, red, green, and blue signals are measured at each point in the image.

Here, each pixel gets only one color – red, green, or blue. (We should note at this point that technically, one should refer to this as a Color Filter Array, or CFA, as there are many possible variants on this basic theme and only this one is properly called a Bayer matrix.) Green pixels outnumber blue and red pixels by a factor of two. This bias towards sampling the green pixels comes from the fact that our eyes have their best sensitivity to green.

Many would at this point assume that there must be three or four times as many pixels in on the CCD to make this work. For example, if we were to take a small 2x2 area from Figure 2, we would have one red, two green, and one blue samples. Combine these into a single pixel that is twice as big in each direction and you have a single pixel that has full-color data – measures of red, green, and blue for this single, larger pixel. While quite simple, this is not what happens. In fact, if the monochrome version of a CCD chip has 1024 x 768 pixels, the color version has 1024 x 768 pixels as well and the color image you see has 1024 x 768 pixels. Each pixel has red, green, and blue values assigned to it even though each native pixel had filters that only measured red, green, or blue data.

It is important to realize that there is nothing different about a color imaging chip than a monochrome chip except for the presence of this filter array. The data that come off the chip are monochrome in nature. Each pixel simply has a filter that tunes what wavelength of photons it is sensitive to. In fact, just as we demonstrated above that there is no color information per se on a monochrome chip, there is no color information per se in each pixel of a color chip. There is only a measurement of how much signal was recorded at each pixel. For example, on a green pixel with the response profile shown in Figure 1, the same intensity would be recorded if 1000 photons with a 540 nm wavelength hit the pixel as if 2000 photons with a 590 nm hit the pixel. When data are read off the chip, they give purely grayscale information.

Debayering: The Basics

The process by which this image gets decoded from a color matrix (or "mosaic") into a full-resolution color image is called "debayering" or "demosaicing". There are many ways of reconstructing the color image, ranging from the simple to the exceedingly complex. Some work very poorly, but some work exceptionally well. What follows is a description of several of the techniques. It is good to keep in mind that in general there is a trade-off between speed and accuracy. If you need an image quickly, the reconstruction of the image will likely be less accurate than if you allow yourself the time and computational resources to perform a more sophisticated reconstruction of the image. This is why webcams that must stream images at 30 frames per second or even digital still cameras (whose users do not want to wait many seconds between snapshots) often produce images that suffer from color artifacts.

That said, all demosaic techniques involve interpolation – or the estimation of missing values based on surrounding, known values. Interpolation is an educated guess as to what a missing value should be and it is just that – a guess. But, some guesses can be quite accurate. For example, if you were shown the sequence 1, 2, 3, 4, X, 6, 7, 8, 9 and asked guess what value should replace X, most would choose 5. This is an interpolation (and odds are a 100% accurate one). As we will see below, however, in situations a bit more complex than the numbers 1-9, some guesses are better than others.

Methods and Evaluation

The demonstrations shown in Figures 3 and 4 will help evaluate the algorithms and tradeoffs. In each, high-resolution images were taken from http://www.hubblesite.org, one from the Hubble Telescope (M1) and the other from Kitt Peak (M4), shrunk by adaptive binning (in Nebulosity) to reduce noise and imported into Matlab® where they were converted into a Bayer matrix representation of the image (each pixel got red, green, or blue values from the original image). Small 150x150 segments were cropped off so that they could be magnified to easily reveal the errors in the reconstruction. Code for each of the techniques was taken from Ting Chen's website (http://www-ise.stanford.edu/~tingchen/main.htm) which also describes the algorithms in greater detail.

These two images were chosen because they stress different aspects of debayering. The region from the Crab Nebula is much more like the kind of image that debayer algorithms have been designed for. By far the greatest use is in still cameras or video cam-

continued ⇨

Debayering Demystified *continued*

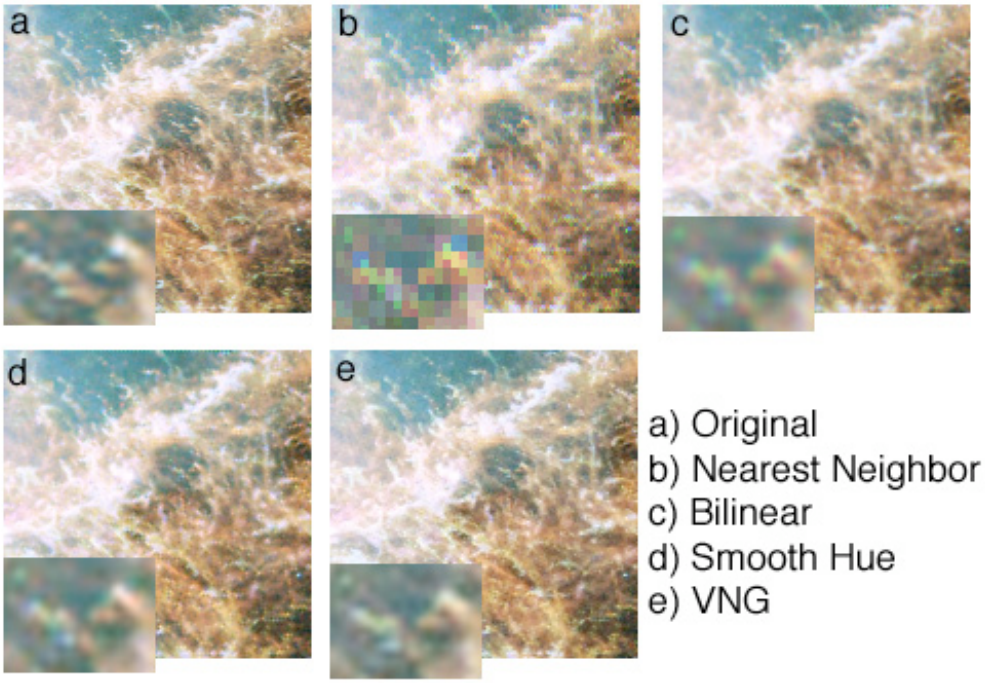

Figure 3: Detail section of a region of the Crab Nebula comparing several debayer techniques. The original image (a) was converted into a Bayer matrix and then reconstructed using: (b) Nearest Neighbor, (c) Bilinear, (d) Smooth Hue, and (e) Variable Number of Gradients. Insets show details of performance, zoomed to 400%.

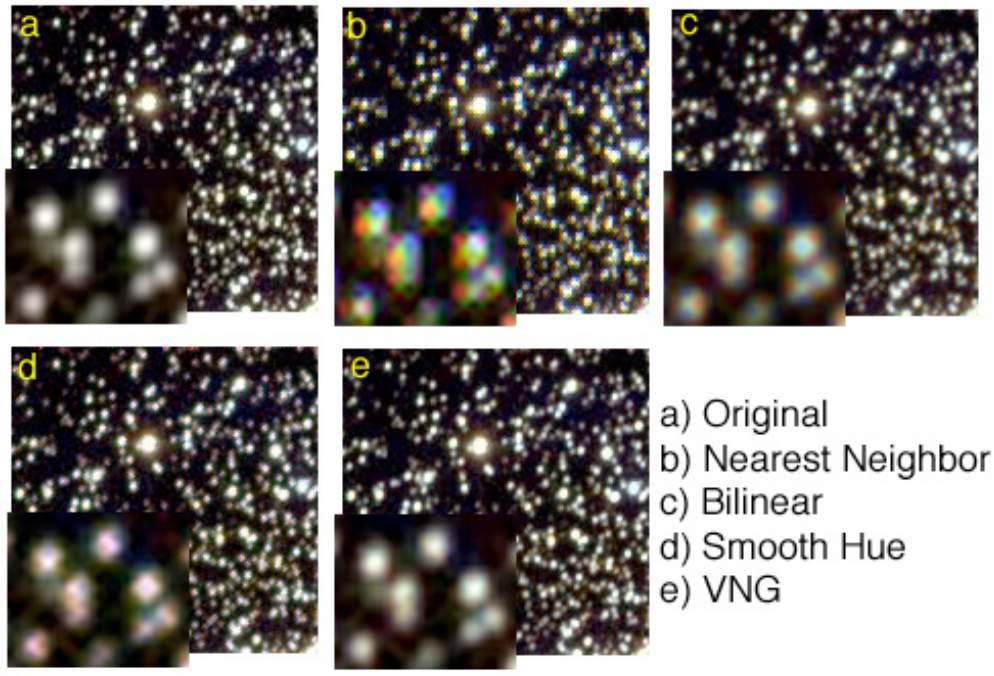

Figure 4: Detail section of a region from M4 comparing several debayer techniques. The original image (a) was converted into a Bayer matrix and then reconstructed using: (b) Nearest Neighbor, (c) Bilinear, (d) Smooth Hue, and (e) Variable Number of Gradients. Insets show details of performance, zoomed to 400%.

continued ⇨

Debayering Demystified *continued*

eras taking pictures of natural scenes here on earth, which tend to have smooth transitions (at least of hue). In contrast, the severe contrast change associated with a star sitting on a dark background is not typical of what the algorithms have been designed to do or previously evaluated on. Yet, both kinds of images are typical of what astrophotographers deal with.

Nearest Neighbor Interpolation

By far the simplest means to fill in missing values is called nearest neighbor interpolation. If you don't know the value for a particular pixel, find the value of the pixel nearest to you that has the right kind of information (i.e., is of the same color) and use its value as your best guess. So, if we are trying to estimate the red value for the upper-left pixel in Figure 1 (where we only really know the green value), we would take the value from the red pixel below this and copy it over as the red value. Thus, we could say that Red(1,1) = Red(1,2). Green(1,1) = Green (1,1) and Blue(1,1) = Blue(2,1). To estimate the full set of color values at pixel 1,1 we used the green from this pixel, the red from pixel 1,2 and the blue from pixel 2,1 – the nearest neighboring pixels that had the right kind of information.

Nearest neighbor interpolation is blisteringly fast but suffers from both a loss of resolution and from color error at the edges of objects (Figure 3b and 4b). Stars are particularly brutal for nearest neighbor interpolation as we often go from fairly white stars to a fairly black sky. Thus, on the last bright part of the edge of a star (which might fall on a green pixel) we use the bright signal recorded from that green along with the dim signal recorded from the neighboring sky for the blue and the red, leading to a very green looking edge of the star. This kind of "color fringing" error is clearly evident in Figure 4b.

While fast and easy to implement, nearest neighbor interpolation is very low quality. It is this kind of performance that has given one-shot cameras a poor reputation and to the notion that the resolution is only one half or even one quarter what one can get with a monochrome camera.

Bilinear Interpolation

Only slightly more complex than nearest neighbor interpolation is bilinear interpolation. Here, you take your best guess by using the surrounding pixels. For example, to estimate the green value at pixel 3,2 (where we know the red) we would use the green pixels above, below, to the left, and to the right of this pixel. If one were to look at the green intensity at 3,1 and at 3,3 and fit a line between the intensity values at these two points, we could interpolate the value that is missing at 3,2. Likewise, we could draw a line on a plot from the intensity at Green(2,2) to Green(4,2) we could also interpolate the missing intensity. By combining these two estimates, we have bilinearly interpolated the missing value. In this simple case, bilinear interpolation becomes averaging the value at these four pixels.

Bilinear interpolation is also quite fast and produces clearly superior results to nearest neighbor (Figures 3c and 4c). Color error is reduced and resolution is increased. Given the improvements and the modest computational cost, many devices like webcams will employ bilinear interpolation to reconstruct the image. When directly compared to the original, however, few would say the performance is excellent, however.

Smooth Hue Transition

Taking about twice as much time to compute as bilinear interpolation, we have an algorithm that tries to build a bit more intelligence into the debayering process. Called smooth hue transition, it is based on the idea that in natural scenes, the luminance value may change rapidly, but the hue – or color tone – does not change as rapidly. If we enforce a smooth, gradual change in the color as we move around the image, we can avoid color error. We should note at this point, that this basic principle is employed by a number of very successful imagers. Many imagers using monochrome cameras will shoot a luminance frame at full resolution and shoot R, G, and B frames with their camera in a binned mode (and thus at a lower resolution). They do this to save time, boost signal to noise in the color, and because the color (or hue) changes much more slowly than the brightness (or luminance).

To do this, we first run through the array and interpolate the green values, typically using bilinear interpolation. Green is interpolated first, since there are twice as many green pixels as red or blue pixels, thus giving us our clearest estimate of green across the image. At this point, two "hue maps" are made, one representing the blue-green ratio in the image and one representing the red-green ratio in the image. When interpolating here, what is estimated are these ratios, rather than the blue values or red values themselves. Thus, the green pixels are serving as a baseline, helping to enforce a constant hue. Once these hue maps are created, the green component is removed by simple multiplication and we are left with red, green, and blue values at each pixel.

The results from smooth hue transition are show in Figures 3d and 4d. Resolution has again gone up (as

Debayering Demystified *continued*

the stars are tighter and the tendrils of the nebula are sharper) and color error has gone down. Some artifacts still exist, as the stars are not quite round and some color error is still present. This is certainly not a bad image, however.

Variable Number of Gradients

When discussing bilinear interpolation, two estimates for a given pixel's intensity were calculated (two linear interpolations) and these were simply averaged. What if there is a hard vertical or horizontal edge going through this pixel, however? We might be better off using one or the other of these estimates rather than using both. This is notion is employed by edge-sensing algorithms (and this sometimes replaces the bilinear interpolations performed in other algorithms like smooth hue). Edges are gradients – regions where the intensity is changing rapidly as you move from pixel to pixel. Pick one of the two interpolations based on this edge, or gradient, and you'll likely be more accurate.

We can expand this notion further by calculating more than two gradients and by intelligently selecting which subset of gradients to use in our estimates. This is the basic premise behind the variable number of gradients (VNG) approach. VNG is quite computationally intensive and is not remotely suitable for real-time debayering of video images, but is very accurate as can be seen in Figures 3e and 4e. While the image is not a perfect recreation, few would argue that it is one half or one quarter as good as the original. In fact, if you use a tool like PhotoShop to layer the original and the VNG reconstruction and show the difference, it takes a good bit of stretching to see any difference show through.

Conclusions

One-shot color cameras operate using a Bayer matrix (or a variant thereof) and sample only one color at each pixel. Thus, your 8 megapixel DSLR has 8 million pixels, but only 4 million green pixels and only 2 million red and 2 million blue pixels. This has led to the belief that the resolution of such imagers is far below that of a monochrome camera and that the quality is necessarily much lower as a result of this Bayer matrix and the debayering process. While encoding the image into a Bayer matrix and reconstructing it does involve interpolation and can be done in ways that are horribly inaccurate, the loss need not be severe.

Recall that to generate Figures 3e and 4e, the original (Figure 3a and 4a) was converted into a Bayer matrix so that only one color existed at each pixel. This stripped-down image was then reconstructed and interpolated using VNG to create the panels shown in Figures 3e and 4e. While nearest neighbor interpolation may give one-shot color imaging a bad reputation, more sophisticated algorithms like VNG can go a long way to restoring that reputation.

I do not mean to imply that the debayer process has reached its pinnacle with the VNG algorithm or that there is no reason to use a monochrome imager. Quite the contrary – there are other algorithms out there that can be even more accurate than VNG and this is an active field of research. Further, monochrome imagers do give you added flexibility over color imagers. For example, if you were to place a Hydrogen-alpha filter over a 1 megapixel monochrome chip, you would have one million pixels forming the image. Do this on an RGB Bayer matrix and you would only have the 250,000 red pixels with any response at all. No fancy math will get you all of that back. But even here, all is not lost for one-shot color imagers. Stay tuned for "*Line Filter Image Reconstruction on One-Shot Color Cameras*". ♦

By day, Craig Stark, Ph.D. is a professor of Cognitive Neuroscience at Johns Hopkins University. By night, he is an amateur astrophotographer and developer of software for astrophotography
http://www.stark-labs.com