Author:
Dipl. Ing. (FH) Uwe Furtner
Image-Processing Products Manager
at MATRIX Vision GmbH

Status: 31.08.2001

Color processing with Bayer Mosaic sensors

Introduction

In industrial color image processing, a distinction is made between three-chip cameras and 'regular' single-chip cameras. Three-chip cameras have 3 grey sensors which view the same accurately to one pixel object through a prism using different optical filters. Each of the red, green and blue filters covers a complete sensor chip. The maximum resolution is achieved in each color area by using this method. For cost reasons, however, single-chip color cameras are still more widely used than three-chip cameras.
Single-chip color sensors are conventional CCD sensors to which a color filter is applied. These filters make individual pixels react differently to wavelengths of light. As there exists only one color component per pixel, the missing color components must be determined to obtain a single RGB pixel.
The most widely used color filters are known as 'complementary color mosaic' filters. Here, the colors cyan, yellow, magenta and green alternate pixel by pixel. Chips which convert data to RGB are available for such CCD sensors. Typical analog single-chip color cameras have CCD sensors of this type.
Much mention has been made of 'Bayer Mosaic' or 'Primary Color Mosaic' sensors in the recent past. They are single-chip color sensors but contain the primary colors red, green and blue. These sensors are mostly used in digital video technology. The camera generally supplies only the original data of the sensor rather than a converted RGB image. Against the background of ever-increasing camera speeds, this is a useful feature because it keeps data transfer rates across a PCI bus low. On the other hand, the user's desire to view a color image from his color camera is understandable.
From this starting point, this article will explain how a Bayer Mosaic Filter is designed and what methods are available for converting data to RGB.

Design of Bayer Mosaic Filter

The Bayer Mosaic Filter comprises an RGB filter mask applied to a standard grey sensor. As Fig. 1 shows, the green and blue pixels alternate in one line, and the red and green pixels in the next line. For the following model calculations, the pixels have been numbered consecutively with p(x,y).

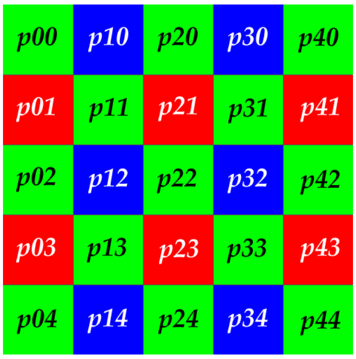

Fig. 1: Design of Bayer Mosaic Filter

Color calculation

The object of the exercise is to calculate, from the raw data generated by the Bayer Mosaic Sensor, a color image which best reproduces the actual object or a recording made with a three-chip CDD camera.

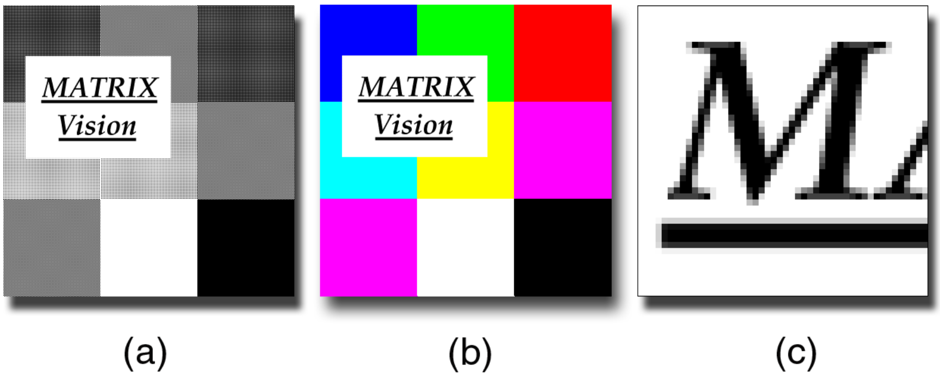

Fig. 2: a) Bayer Mosaic image, b) ideal conversion result, c) image area

Fig. 2a shows the raw data generated by the Bayer Mosaic Sensor in the form of a greyscale image. Fig. 2b shows an ideal color image such as that possible with a three-chip camera. No color artefacts are visible in image area 2c.

Because, for example, the blue value of a pixel is only known exactly in every second column and every second line of the Bayer Mosaic, the values of these color components must be calculated from the surrounding pixels.
This can be done using various methods, which will now be explained.

The simplest method of calculation

The simplest method involves using a 2x2 operator, which runs through the image pixel by pixel. The operator is set to the next row of pixels at a line break. The rule of calculation for each operator position is: take the existing color components without offsetting them against their neighbouring pixels.

R11 = p21
G11 = p11 { or (p11 +p22)/2 }
B11 = p12

R21 = p21
G21 = p31 { or (p31 + p22)/2 }
B21 = p32

....

R12 = p23
G12 = p13 { or (p13 + p22)/2 }
B12 = p12

With this method, the blue or red colored pixels are simply quadrupled and the green pixels are doubled. Studying the red or blue area (Fig. 3c) of the result image shows that 4 neighbouring pixels always have the same color value.

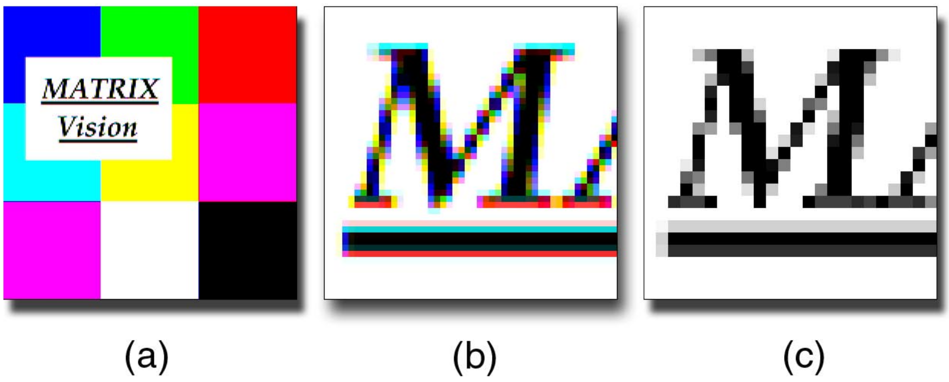

Fig. 3: Result obtained using the simple method, a) complete test image, b) section , c) blue area

This method is based on the assumption that the missing color components at a particular pixel position correpond to the color components of the neighbouring pixels - which, of course, is not the case in real color gradients.

Quick calculation for live representation

To represent a color image in 'Live' mode, a method which requires as little computation as possible must be sought to achieve a high live image refresh rate. The above method would be suitable in terms of the small amount of computation required for live representation. However, the method should be modified slightly to greatly enhance its effect. The red and

blue components are quadrupled using the 'simplest method' . Of course, it is not necessary to generate four times the volume of data to represent these color components.
If only the blue pixels are studied, and the intermediate pixels are disregarded, then an image of the blue channel is obtained. This can be done likewise with the red and green pixels. Overlaying the resulting color layers produces a color image with half the horizontal and vertical resolution of the original sensor image. In this image each of the individual color areas is plaxed offset by a half pixel in the horizontal and vertical direction compared to the real image. Scaling this image by a factor of 2 using bilinear interpolation produces a much better visual result than that obtained with the 'simplest method'. Color blocks are no longer visible.

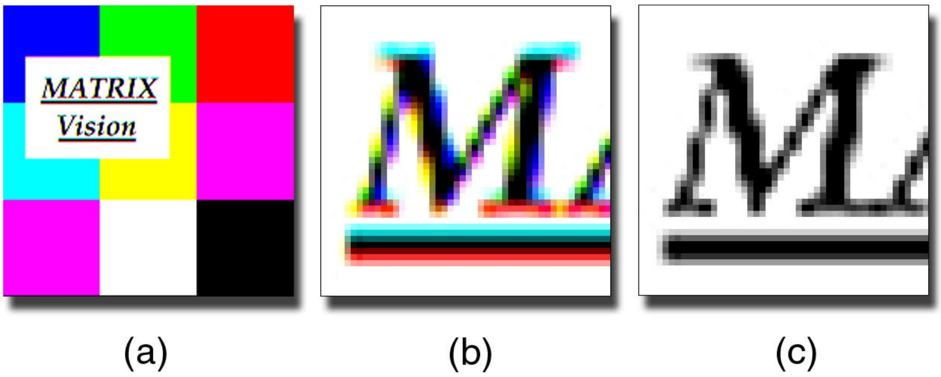

Fig. 4: Result obtained using the quick method, a) complete test image , b) section, c) blue area

Here is an example of a calculation made using this method:

R00 = p01
G00 = p00 { or (p00 +p11)/2}
B00 = p10

R10 = p21
G10 = p20 { or (p20 + p31)/2}
B10 = p30

....

R01 = p03
G01 = p02 { or (p02 + p13)/2}
B01 = p12

General interpolation

To achieve the full color image resolution, allowance must be made for the local offset of color components. It is assumed that the missing colors are obtained by averaging the value sof neighbouring pixels of identical color components. This assumption only applies approximately to seamless color gradients. The nearest color components are always averaged by applying a 3x3 operator:

R11 = (p01+p21)/2
G11 = p11*)

B11 = (p10+p12)/2

R21 = p21
G21 = (p20+p31+p22+p11)/4
B21 = (p10+p30+p12+p32)/4
...

R12 = (p01+p21+p03+p23)/4
G12 = (p11+p22+p13+p02)/4
B12 = p12

R22 = (p21+p23)/2
G22 = p22*)
B22 = (p12+p32)/2

Fig. 5 shows the calculation result.

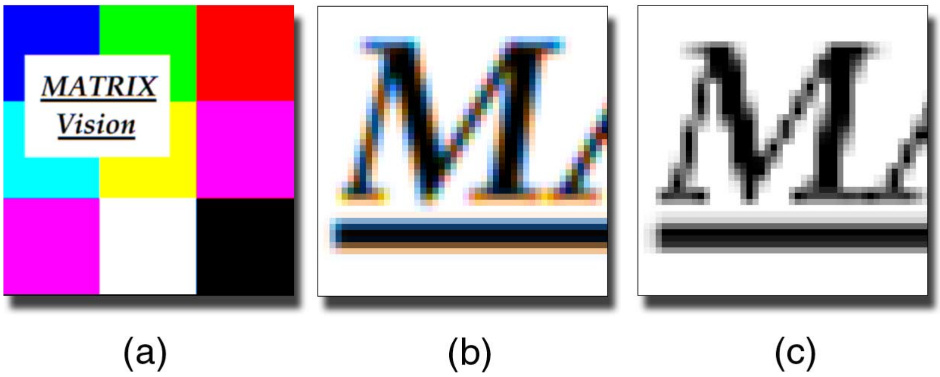

Fig. 5: Result obtained using general interpolation, a) complete test image, b) section , c) blue area

*) according to the sensor's manufacturer, a difference in brightness can occur between a green pixel within the blue lines and a green pixel from the red line. The green pixels should, therefore, not be used directly but should first be obtained by averaging the values of neighbouring green pixels or corrected by factoring. This correction factor fluctuates from sensor to sensor, however, and should be determined by white balancing.

Image focusing interpolation

During the general interpolation, it is assumed that the color changes at half the frequency compared to intensity (luminance). This may apply to a single-chip color sensor but but it brings us no closer to our goal of approaching the color resolution of a three-chip camera as closely as possible. The color artefacts which appear at object edges can interfere with automatic color image evaluation.
Moreover, the resulting image appears slightly blurred due to the interpolation.
Artefacts can be minimized and the image definition can simultaneously be increased by taking the edge information into account.
[5] shows how weighting factors are determined for neighbouring pixels by studying the gradients in various directions of a pixel.
This method naturally involves even more computation than general interpolation.

Hardware solution

Up until now, we have always assumed that the raw data of the sensor ends up in the PC image memory and that we have to convert this data to RGB using appropriate algorithms. This data can also be converted directly by a frame grabber, such as the MVtitan-DIG by MATRIX Vision. This way the PC is not burdened with the conversion work. The board also uses general interpolation for 'Live' mode, with the result that no differences in quality are discernible between live mode and snap mode.

Conclusion

The algorithm to be used depends on the application. Images can be 'enhanced' with considerable expenditure of time and effort. Of course, it must be remembered that color conversion involves estimating, i.e. inventing, the intermediate color values. The maximum resolution can really only be achieved with three-chip cameras. However, digital Bayer Mosaic cameras achieve much better image quality than standard single-chip color cameras. Therefore, there is a quality hierarchy that is also reflected in the prices of the cameras.
If visualization of the camera image is only of secondary importance, then there is no need to completely convert the image - which would reduce data volume to only one third. The alternative is to convert only the evaluation ranges, unless the image processing algorithms can handle a Bayer Mosaic image.

References

[1] Brainard, D., "Bayesian Method for Reconstructing Color Images from Trichromatic Samples", IS&T, 47th Annual Conference, pp. 375-380, 1994.

[2] Cok, D., "Reconstruction of CCD Images Using Template Matching", IS&T, 47th Annual Conference, pp.380-385, 1994.

[3] Gann, R., Reviewing and Testing Desktop Scanners. Hewlett-Packard Company, 1994.

[4] Wandell, Brian A., Foundations of Vision. Sinauer, Sunderland, dimension., ch.2-4,8, 1995.

[5] Ed Chang, Shiufun Cheung, Davis Pan, "Color Filter Array Recovery Using a Threshold-based Variable Number of Gradients", IS&T/SPIE Conference on Sensors, 1999.