Practical Paper

Quality Color Composite Image Display Using Low Cost Equipment

Salvador Diaz-Cayeros
IBM MEXICO Scientific Center, Rubén Darío 55 Esq. Campos Eliseos, Colonia Polanco 11560, México D.F., Mexico

ABSTRACT: A new algorithm which provides a solution for the display of color composite images is presented. It yields quality color images using a low-cost eight-bit plane refresh buffer. With this approach, it is possible to apply any pixel value mapping to the digital image. This produces an image display of similar quality to the one obtained from three eight-bit plane, or 24 bits per pixel, hardward. Eight bits per pixel hardware is limited to only 256 different colors, which are very few hues for color composite display. Hence, an approach to producing color must recognize this limitations. This technique defines a few of the most suitable colors before using them in an image display. It may prove useful for universities and small research projects where expensive hardware is frequently not available.

INTRODUCTION

FULL COLOR DISPLAY of digital images is essential in remote sensing. This application is conventionally carried out on computers with powerful peripherals and large memory capacity. The most satisfactory hardware for this task is the 24 bits per pixel refresh buffer, which uses eight bits for each basic color.

However, this variety of hardware may not be available. Taking into consideration that, at universities and in small research projects, personal computers and workstations are used for digital image processing and remote sensing applications (Macias and Díaz, 1989), a technical approach using the relatively cheaper and available eight-bit plane color displays would be useful and practical.

Graphic hardware limited to one byte for every display point is a common configuration in workstations and microcomputers. Typically, this kind of device is capable of displaying 256 colors at the same time from a palette of 2^{18} (262,144) colors.

Upon study, it is evident that the strong limitation is not the hue of the color that is attainable, but which colors are to be displayed at the same time. Selection of the most representative colors for displaying a particular digital image is crucial. The purpose of this work is to disclose a technique for producing color composite images in low cost eight bits per pixel storage buffers of similar quality to the ones produced on the more expensive hardware mentioned before. So the algorithm described in this paper will make an eight-bit refresh buffer yield a subjective image whose quality is quite comparable to a more capable eight-bit per color storage buffer.

COLOR COMPOSITE IMAGE DISPLAY

In order to treat the display of color composite images, it is necessary to take into account the software issue of how the values of pixels are modified in order to obtain a pleasant display and to mention the differences between the eight-bit plane and the 24-bit plane display devices.

An important operation done while displaying images is to use the tones which give the best aspect to a picture. A function called a "pixel value mapping" is applied to each pixel's value of intensity so that an image is not too dark, or too saturated, for instance. The function may transform a sampled value which has a range from 0 to 255 to the same range, as long as both the resolution of the stored image and the display device are 8 bits. More frequently, the pixel value mapping will produce a reduced range of intensity values. This is the result of the low spectral resolution of typical display devices. So, as an example, a pixel value mapping may receive a byte in the range (0...255) as input and produce an intensity value in the range (0...63) (Figure 1) when the display hardware is limited to 6 bits for each displayed pixel (Niblack, 1986).

Some common pixel value mappings are linear, logarithmic, or exponential adjustments and histogram techniques (i.e., histogram equalization). Pixel value mapping transformations determine contrast and brightness among other subjective characteristics of the picture, and are helpful for enhancing details. The algorithm presented here uses and preserves any pixel value mapping throughout the process. There is also a simplification in calculations and storage using only the mapped value of the pixel, and there is no loss in information content because the color ranges used from the beginning of the process are exactly the ones which were to be displayed ideally.

An excellent hardware for color composite pictures is the 24 bits per pixel frame buffer at which each componet of a color image (Red, Green, and Blue) is stored in an eight-bit plane. Figure 2 presents this sort of device. The schematics of this and the next figure (Figure 3) draw heavily on Di and Rundquist (1988). Each group of bit planes drives an eight-bit Digital to Analog Converter (DAC). Any eight-bit group can generate 256

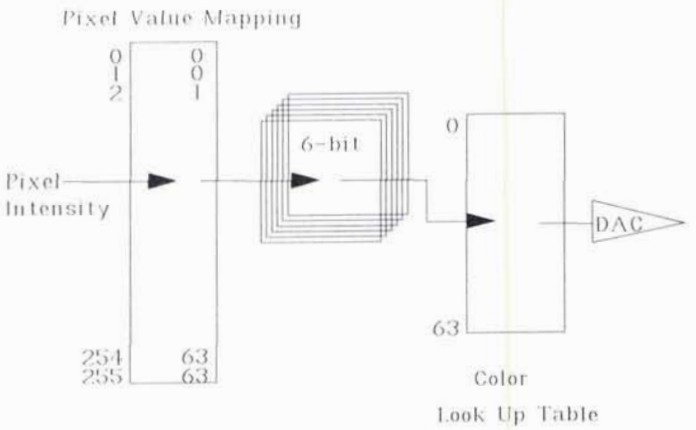

FIG. 1. Pixel value mapping for a 6-bit display device.

PHOTOGRAMMETRIC ENGINEERING & REMOTE SENSING,
Vol. 56, No. 12, December 1990, pp. 1625-1629.

0099-1112/90/5612–1625$03.00/0

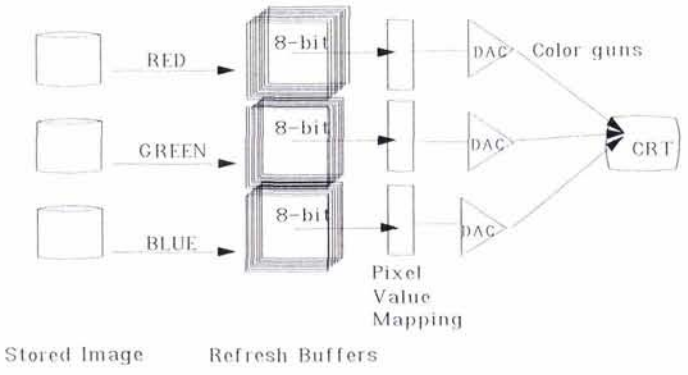

FIG. 2. Full color display schematic drawing.

(2^8) shades of intensity. Each of the three DACs drive one color gun, either red, green, or blue. All may be combined into 16,777,216 possible colors (2^{24}). This type of device is called a "full" color frame buffer (Rogers, 1985).

Today the full color refresh buffer is hard to find. A more popular display device is the eight bits per pixel kind, mainly due to its relatively low cost. The scheme below (Figure 3) is a description of the IBM PS/2 Video Graphic Array (VGA) which has eight bit planes and six bits for driving each of the color DACs of a 256 entry color look up table (CLUT) (IBM, 1987). The algorithm and the implementation which follow are considered for these machines and ones of similar capabilities.

COLOR DEFINITION BY MULTI-BAND EQUALIZATION

Some techniques have been proposed for solving the problem of color composite images in eight-bit displays. Outstanding among them is Liping Di and Donald C. Rundquist's (Diand Rundquist, 1988) method. The method defines a CLUT with the combinations of six levels of red, six of green, and six of blue. Then the color compostie is read and a formula which finds the color in the table and determines which of the predefined colors is used. The best display is obtained using histogram equalization as the pixel value mapping.

In this paper a different method is illustrated with several advantages, including being able to use any pixel value mapping, and its capability of taking the most out of the few available colors in order to achieve a high quality display.

This algorithm uses a three-dimensional histogram or density matrix which records the occurrence of every vector of RGB after applying the pixel value mapping to every data point in the image. In this way, any pixel value mapping is preserved throughout all the process. Normally, mappings are done in the hardware CLUT. In this case, though, the hardware CLUT is reserved for a better use, to accurately define each of the needed colors for the display.

The three-dimensional density matrix defines an RGB color cube. Along each axis a basic color intensity is represented. Each cell is the density of a given color combination (Figure 4). The RGB color cube was divided into 252 zones, each containing approximately the same number of color combinations which occurred after applying a pixel value mapping to a particular image. The equalization was done on a band-per-band basis. First, in the red band: six divisions. Afterwards, seven divisions were made in each of the six previous red divisions. Finally, the red and green divided prisms (42 in total at this point) were divided into six regions each, resulting in a total of 252 partitions.

The reason for using 252 partitions is that the maximum num-

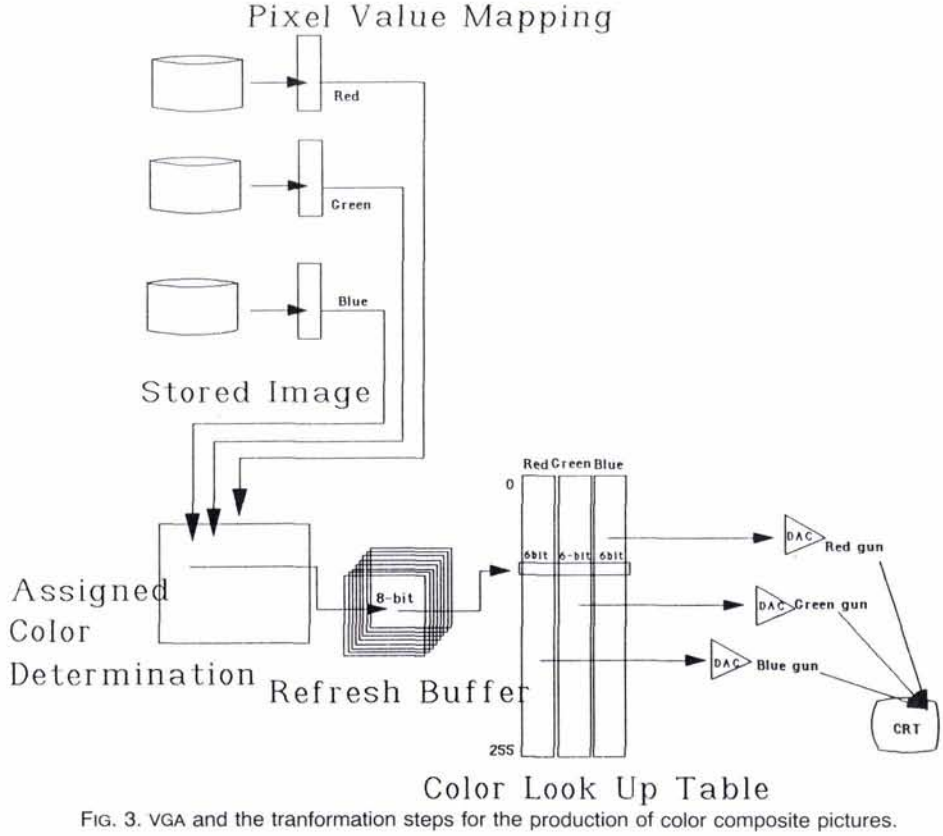

FIG. 3. VGA and the tranformation steps for the production of color composite pictures.

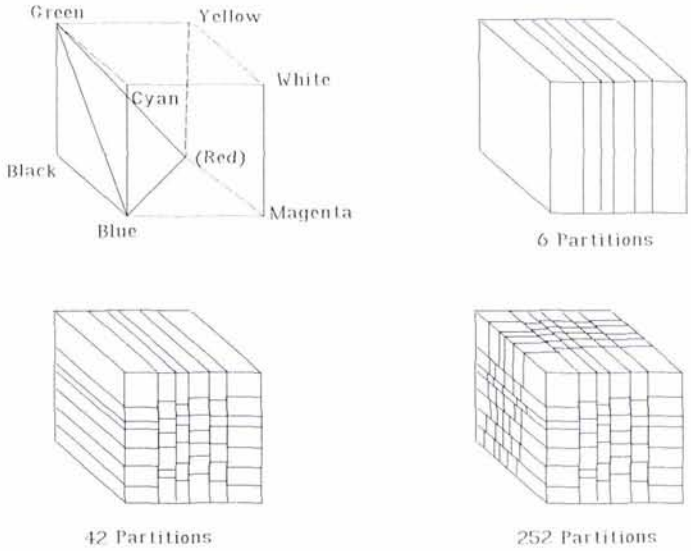

FIG. 4. RGB color cube and partitions.

ber of divisions which can be made are 256, given the 256 entry CLUT and eight-bit per pixel refresh buffer. One of the primitive approaches for color definition with three color guns was to use three binary bits for red ($2^3=8$), three for green ($2^3=8$), and the remaining two bits for blue ($2^2=4$), so one axis is divided in 8, another in 8, and the third in 4. Such a disparity among the axis partitions is not fuctional; to choose the appropriate number of partitions, some of the available colors are not used for the display of an image in this algorithm. When one color is not used, and 255 divisions are made, it is not possible to make a three-dimensional distribution because the prime factors of 255 are 5 and 51. When two colors are left for future use (i.e., lettering), the prime factors of 254 are 127 and 2 and, for 253 colors used in the algorithm, the factors are 23 and 11, so the best choice left for having a balanced number of partitions at every band and using as many colors as possible is 252 which is decomposed in three factors: 6, 7, and 6. Two axes are assigned six partitions and the axis with most of the information content has seven partitions assigned.

Once 252 cubic volumes are obtained, the best choice of color for representing the pixels within each partition is not the center of the cube, but the center of mass of all the occurrences of pixel vectors within that cube, so a three-dimensional mean vector is obtained from the population of a given partition. This vector is scaled to the units of the hardware CLUT which drive the color DACs, defining the color which will represent all color combinations of pixels which fall within a given partition.

STEPS OF THE ALGORITHM

The process is defined here in a more detailed manner. The algorithmic steps include

(1) Calculate the distribution matrix:

```
FOR every pixel of the image in disk storage
{
  READ (red, green, blue); /*Elements of the pixel vector*/
  i = TT_red[red];     /*apply a pixel value mapping*/
  j = TT_green[green];
  k = TT_blue[blue];
  dm[i, j, k] = dm[i, j, k,] + 1; /* record this event */
}
```

Where $TT_x[\ ]$ is any pixel value mapping.

(2) Next, histogram equalization is done in each dimension with the following algorithm. This algorithm is a modification of one presented by Boyle and Thomas (1988). The image has M levels of gray represented by intensities (0,1,...,M-1). A transformation table which can be fast and easily accessed is produced; i.e.,

$$s = T[r] \text{ for } 0 \le r \le M-1,\ 0 \le T[r] \le N-1$$

where N is smaller than M in order that the transformation simplifies further calculations. T[r] is monotonic and is a transfrom which will make sure that each gray level is occupied by approximately Total/N pixels, where Total is the total number of pixels at the image. The floor function $\lfloor\ \rfloor$ is used for truncation; i.e.,

$$p[r] = \text{number of pixels at gray level } r,\ 0 \le r \le M-1$$

$$P[r] = \sum_{j=0}^{r} p[j] = \text{number of pixels at gray level } r \text{ or less}$$

$$s = T[r] = \left\lfloor \frac{N\ P[r]}{\text{Total} - 1} \right\rfloor$$

The density for making the histogram equalization will be calculated from $dm[i, j, k]$. This is the value of the density for a particular combination of red, green, and blue after pixel value mapping. The partition is defined between $il_{\text{red}}[i]$ and $sl_{\text{red}}[i]$. The equalization is obtained from the densities $p_{\text{red}}[i]$ defined as

$$p_{\text{red}}[i] = \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} dm[i, j, k]$$

where, $il_{\text{red}}[i]$ is defined as the smaller value of $T_{\text{red}}[i]$, and $sl_{\text{red}}[i]$ is defined as the larger value of T_{red} in partition i.

(3) Partitions are done for the green axis. For each of the six red partitions using index "a":

$$p_{\text{green, red\_part}[a]}[j] = \sum_{i=il_{\text{red}}[a]}^{sl_{\text{red}}[a]} \sum_{k=0}^{N-1} dm[i, j, k].$$

(4) Taking as summation limits the partitions of red and green axis, the blue partitions are then determined; i.e.,

$$p_{\text{blue, red\_part}[a], \text{green\_part}[b]}[k] = \sum_{i=il_{\text{red}}[a]}^{sl_{\text{red}}[a]} \sum_{j=il_{\text{green}}[b]}^{sl_{\text{green}}[b]} dm[i, j, k]$$

Centers of mass for each partition are calculated by averaging the population (Figure 5):

$$\tau = \sum_{i=il_{\text{red}}[a]}^{sl_{\text{red}}[a]} \sum_{j=il_{\text{green}}[b]}^{sl_{\text{green}}[b]} \sum_{k=kl_{\text{blue}}[c]}^{sl_{\text{blue}}[c]} dm[i, j, k]$$

Taking the same limits:

$$\mu_{\text{red}} = \frac{\Sigma\Sigma\Sigma i dm[i, j, k]}{\tau}$$

$$\mu_{\text{green}} = \frac{\Sigma\Sigma\Sigma j dm[i, j, k]}{\tau}$$

$$\mu_{\text{blue}} = \frac{\Sigma\Sigma\Sigma k dm[i, j, k]}{\tau}$$

The color definition in the CLUT, once all the elements in the partition are taken into account, is the following:

$$x = 42a + 6b + c$$
$$\text{color}_{\text{red}}[x] = \text{Scale\_factor}\,\mu_{\text{red}}$$
$$\text{color}_{\text{green}}[x] = \text{Scale\_factor}\,\mu_{\text{green}}$$
$$\text{color}_{\text{blue}}[x] = \text{Scale\_factor}\,\mu_{\text{blue}}$$

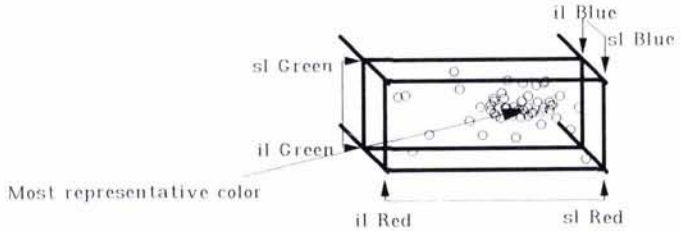

FIG. 5. Center of a partition.

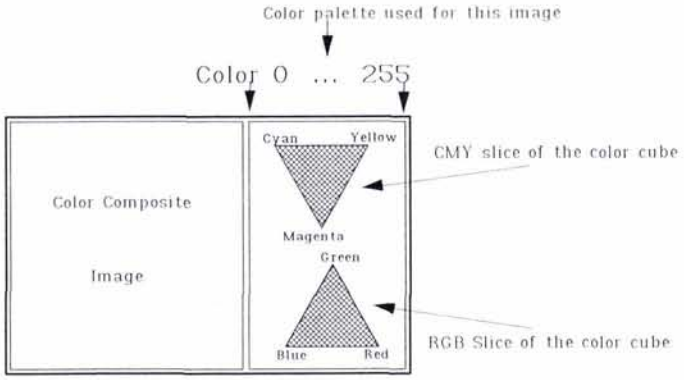

FIG. 6. Elements of a color plate: image and palette.

The scale factor depends on the range of red output pixels and the allowable range for the red component in the definition of the color (i.e., for an output pixels range (0 . . . 39) and allowable range in the CLUT (0 . . . 63) for a six bits per color DAC, the scale factor may be 1.61 or less). A table $TM[i,j,k]$ from which each value is the color for representing a certain specific pixel vector $[i,j,k]$ is calculated during the process.

(5) For optimum speed in the actual plotting of the image, the translation matrix $TM[i,j,k,]$ is used as follows:

```
RESET file (s) of three color components
FOR each pixel in the image
  READ(red,green,blue);
  I = TM[TT_red[red],TT_green[green],TT_blue[blue]];
  SET (I); /* for this pixel */
```

Note that pixel value mappings are applied before using the translation matrix so that TM is a smaller array.

IMPLEMENTATION

The implementation was done in the C programming language and it was run in two machines: an IBM RISC Technology RT PC workstation and an IBM PS/2 Model 80 Microcomputer.

The RT PC was equipped with the Megapel adapter with the capability of displaying 1024 by 1024 pixels; each pixel was eight bits wide (IBM, 1986). After programming and testing the algorithm, the display of a color image of 512 pixels by 400 lines size took one minute fifteen seconds. Aided by the great spatial resolution of the Megapel adapter, the display compares favorably to the 24-bit frame buffers.

Plate 1 is a Landsat MSS color composite image with a linear pixel value mapping. The size of the image is 512 by 512 pixels. At the right side, the selected colors are displayed (Figure 6). The two triangles are cuts made on the RGB color cube illustrated in Figure 4, so that different colors with the same intensity are compared. In the background are the 252 chosen colors. Where black represents colors not used. Plate 2 is a widely known test picture, and this display may be compared to the one produced by other authors (Niblack, 1986).

Another version of the program was done for microcomputers. The dispersion matrix is one of the largest structures in the program. Most microcomputers are limited to data structures less than 64 kilobytes in size, so several variables of unsigned character type were declared for holding the dispersion matrix. Each of these variables takes one byte of storage space. The pixel mapping was from a range (0...255) to the range (0...39) so that the table kept a reasonable size of less than 64 kilobytes ($40^3 = 64000$). The IBM PS/2 has a color display adapter denominated 8514 with a resolution of 1024 by 768 pixels. Each pixel is 8 bits wide, which allows the display of 256 simultaneous colors. The display of 512 by 512 pixels color image in a PS/2 Model 80 with 80387 math coprocessor and 8514 display takes approximately two minutes.

CONCLUSIONS

The algorithm described in this paper provides a cost effective solution for the display of color composite images.

The main goal of this work is to have remote sensing systems based on very small machines. These systems could accomplish specific analysis of remote sensed data. In this way, it is possible to have a good representation of color composite images with the current generation of eight bits per pixel refresh buffers. This will be useful in small remote sensing systems based on microcomputers and for the economical display of color digital imagery in general.

REFERENCES

Boyle, R.D., and R.C. Thomas, 1988. *Computer Vision*. Blackwell Scientific Publications, Osney Mead, Oxford, Great Britain, 210 p.

Di, Liping, and D.C. Rundquist, 1988. Color-Composite Image Generation on an Eight-Bit Graphics Workstation. *Photogrammetric Engineering & Remote Sensing* 54:1745-1748.

International Business Machines Corporation, 1986. *IBM RT PC HARDWARE TECHNICAL REFERENCE*, Volume II, Armonk, New York.

———, 1987. *Personal System/2™ Model 50 and 60 Technical Reference*, Armonk, New York.

Macias, J.M, and S.C. Díaz, 1989. *A Microcomputer Based Remote Sensing Application*. IBM Scientific Center, México, 10 p.

Niblack, W., 1986. *An Introduction to Digital Image Processing*. Prentice-hall International (UK) Ltd, London, 215 p.

Rogers, D.F., 1985. *Procedural Elements for Computer Graphics*. McGraw-Hill International Student Edition, Singapore, 433 p.

(Received 5 July 1989; revised and accepted 13 March 1990)

PLATE 1. Landsat MSS image with a linear pixel value mapping displayed in the IBM Personal System/2 eight-bit plane display adapter 8514/A.

PLATE 2. The baboon: a test picture for color composite display.