1-1989

An investigation of the color reproduction accuracy of two halftoning algorithms for dot matrix systems

Peter A. Zuber

ROCHESTER INSTITUTE OF TECHNOLOGY

Center for Imaging Science

MASTER OF SCIENCE DEGREE THESIS

An Investigation of the Color Reproduction Accuracy of Two Halftoning Algorithms for Dot Matrix Systems

Peter A. Zuber

March 1989

Thesis Advisor

Peter G. Engeldrum

An Investigation of the Color Reproduction Accuracy

of Two Halftoning Algorithms for Dot Matrix Systems

by

Peter A. Zuber

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in the Center for Imaging Science in the College of Graphic Arts and Photography of the Rochester Institute of Technology

March 1989

Peter A. Zuber

Signature of the Author

Name Illegible

Coordinator, M.S. Degree Program

ROCHESTER INSTITUTE OF TECHNOLOGY

Center for Imaging Science

Rochester, NY

Master of Science Degree Thesis

The Master of Science Degree Thesis of Peter A. Zuber has been examined and approved by the following thesis committee as satisfactory for the thesis requirement of the Master of Science Degree

Peter G. Engeldrum

Peter G. Engeldrum, thesis advisor

Roy S. Berns

Roy S. Berns

Burt Saunders

Burt Saunders

8/1/89

Date

ABSTRACT

The recent popularity of dot matrix printing technologies has renewed interest in developing color halftoning techniques for these systems. A color reproduction scheme based on colorimetric principles would provide accurate color rendition, and can be configured to different hardware implementations. Additionally, where there are demands for multiple copies, color reproduction accuracy is assured to the nth generation.

A binary dot matrix halftoning algorithm previously used for black-and-white reproduction (error diffusion) and a new algorithm to be described (EZ method) were investigated in terms of their color reproduction capabilities, with the objective to achieve colorimetric color reproduction.

The error diffusion technique made poor system color selections when used in XYZ tristimulus space. As a result, large hue, saturation, and ΔE^*_{ab} errors were experienced. The EZ Color Algorithm provided better color accuracy, with an average color difference of less than three for a 4x4 cell size.

A uniform color space, such as CIELAB, is considered a minimum requirement in order for the error diffusion algorithm to provide colorimetric color reproduction. Hue, saturation, and ΔE^*_{ab} errors were minimized when this color space was used. The EZ Color Algorithm provides several important features including the incorporation of the black colorant explicity in the color formulation, selection of system colors prior to quantizing, and quantization of system color areas instead of reflectance values.

TABLE OF CONTENTS

TABLE OF FIGURES

NO.	FIGURE	PAGE

TABLE OF TABLES

INTRODUCTION

Several methods for converting black and white continuous tone digital images into binary form have been developed and described.[1,2] This study investigated the color reproduction capabilities of two binary halftoning algorithms, Floyd and Steinberg's error diffusion[3], and Engeldrum and Zuber's EZ Color algorithm.[4] Though Floyd and Steinberg initially developed error diffusion for black-and-white systems, potential existed for its extension into color. The EZ Color algorithm is introduced here and utilizes a recently proposed color reproduction theory called Structured Dot Theory (SDT).[5]

Analysis and Synthesis

A color reproduction system provides two functions, analysis and synthesis. Analysis is the sampling and subsequent division of an original color into three separate components which describe the original color. Synthesis is determining the amount of each colorant, the cyan, magenta, and yellow colors of the printer system, to reproduce the original color, and combining them to create the reproduction.

Color Metrics

Color has three perceptual attributes: hue (red, green, orange); saturation (pink to blood red); and lightness (light / dark quality of a color). Color order systems are devised based on these perceptual dimensions, and attempt to quantify them using the illuminant, original color, and detector. The C.I.E. 1931 recommendation[6] describes color using tristimulus values, X, Y, and Z, defined by equation 1a-c. In these expressions S represents the light source spectral power distribution, and $\bar{x}$, $\bar{y}$, and $\bar{z}$ are the C.I.E. color matching functions[7] for the 1931 observer. The product of S, $\bar{x}$, $\bar{y}$, and $\bar{z}$, and the spectral reflectance factor of the object color being considered, R, are integrated over the visible portion of the electromagnetic

spectrum (λ_a = 380nm ; λ_b = 830 nm), along with the spectral reflectance factor of the object color being considered, R.

$$X = \int_{\lambda_a}^{\lambda_b} R(\lambda)\ S(\lambda)\ \bar{x}(\lambda)\ d\lambda$$

$$Y = \int_{\lambda_a}^{\lambda_b} R(\lambda)\ S(\lambda)\ \bar{y}(\lambda)\ d\lambda \qquad \text{(1a-c)}$$

$$Z = \int_{\lambda_a}^{\lambda_b} R(\lambda)\ S(\lambda)\ \bar{z}(\lambda)\ d\lambda$$

From X, Y and Z, the chromaticity coordinates x, y are calculated using equations 2a-b. Chromaticity coordinates provide a convenient way of mapping colors on a Cartesian coordinate based chromaticity diagram.

$$x = \frac{X}{X + Y + Z} \qquad y = \frac{Y}{X + Y + Z} \qquad \text{(2a-b)}$$

X, Y and Z values can be transformed to CIE L* a* b* quantities which, when plotted in rectangular coordinates, provide an approximately uniform color space. X_n, Y_n, Z_n are the tristimulus values of the light source. L* is used as a quantitative description of color lightness and is given in equation 3.

$$L^* = 116\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 \qquad \text{(3)}$$

The a* quantity is derived from the X and Y tristimulus values and is given in equation 4.

$$a^{*} = 500\left[\left(\frac{X}{X_n}\right)^{\frac{1}{3}} - \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}}\right] \tag{4}$$

The b* quantity is derived from the Y and Z tristimulus values and is given in equation 5.

$$b^{*} = 200\left[\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - \left(\frac{Z}{Z_n}\right)^{\frac{1}{3}}\right] \tag{5}$$

Using a* and b*, hue angle (θ), a measure of hue change (red to green to blue), and metric chroma, C*, can be found.

$$\theta = arctangent\left(\frac{b*}{a*}\right)$$

- for a* = 0 and b* = 0

$$\theta = arctangent\left(\frac{b*}{a*}\right) + \pi \tag{6}$$

- for a* < 0

$$\theta = arctangent\left(\frac{b*}{a*}\right) + 2\pi$$

- for b* < 0

$$C* = \sqrt{a*^2 + b*^2} \tag{7}$$

$\Delta E\,*_{ab}$ is a measure of overall color difference.

$$\Delta E\,*_{ab} = \sqrt{(L*_{orig} - L*_{repro})^2 + (a*_{orig} - a*_{repro})^2 + (b*_{orig} - b*_{repro})^2} \tag{8}$$

Color Halftone Synthesis

In 1937 Neugebauer[8] developed mathematical expressions to quantify color reproduction with the three-color halftone process. The three colorants used were cyan, magenta, and yellow. Overprinting created red (magenta and yellow), green (yellow and cyan), blue (cyan and magenta), and black (cyan, magenta and yellow). Paper base (white) increased the total to eight possible system colors present at any one time on the reproduction.

For an original color OC, the tristimulus values of which are Xoc, Yoc, Zoc, a color match will result if the reproduced color has the same tristimulus values. The mathematical expression for this is given in equations 9a-c. The *f* coefficients represent the weightings of the 8 possible colors, the system colors.

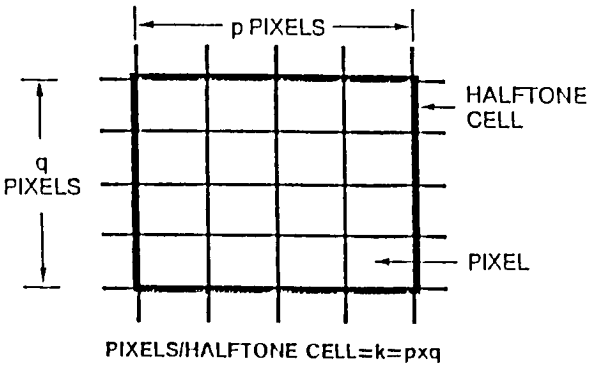
$$X_c = \sum_{i=1}^{8} X_i f_i \qquad Y_c = \sum_{i=1}^{8} Y_i f_i \qquad Z_c = \sum_{i=1}^{8} Z_i f_i \qquad \text{(9a-c)}$$

The coefficients, f_i, are described in the following way. If *c* is the fractional area of paper covered by cyan dots, *m* the area of magenta dots, and *y* the area of yellow, it follows that (1-*c*) is the area not covered by cyan, (1-*y*) the area not covered by yellow, and so on. If these expressions are identified as probabilities, then the probability of any point on the paper being covered by cyan is *c*, by magenta, *m*, and by all three colors, being black, the product *cmy*. The probability of any area being white would be the product (1-*c*)(1- *m*)(1-*y*). All eight expressions are listed as equation 10, with X_1, Y_1, Z_1,...,X_8, Y_8, Z_8 being the C.I.E. tristimulus values for each system color.

Tristimulus Value		Weighting Factors	
(X_1, Y_1, Z_1)	White	$f_1 = (1-c)(1-m)(1-y)$	
(X_2, Y_2, Z_2)	Cyan	$f_2 = c(1-m)(1-y)$	
(X_3, Y_3, Z_3)	Magenta	$f_3 = m(1-c)(1-y)$	
(X_4, Y_4, Z_4)	Yellow	$f_4 = y(1-c)(1-m)$	(10)
(X_5, Y_5, Z_5)	Red	$f_5 = my(1-c)$	
(X_6, Y_6, Z_6)	Green	$f_6 = cy(1-m)$	
(X_7, Y_7, Z_7)	Blue	$f_7 = cm(1-y)$	
(X_8, Y_8, Z_8)	Black	$f_8 = cmy$	

Since the tristimulus values of the original to be reproduced, (X_{oc}, Y_{oc}, Z_{oc}), and the eight system colors, $(X_1, Y_1, Z_1, \ldots, X_8, Y_8, Z_8)$, are known, then it remains to find the coefficients. In 1948, Hardy and Wurzburg[9] developed a system for three color halftone reproduction based on this idea. The system continuously evaluated, electronically, the amount of c, m, and y inks needed to match the tristimulus value of an original. This provided the amount of c, m, and y needed to give colorimetric color reproduction, in terms of the fractional area of a halftone.

The recent emergence of dot matrix imaging technologies has given new interest in developing color halftoning techniques for these systems. These systems are unique in placing dots of constant area in the same location, the so-called "dot-on-dot" method. The Neugebauer equations assume the reproduction will be randomly distributed colored dots of varying area. Because this assumption does not apply to dot matrix systems, new color halftoning techniques were needed.



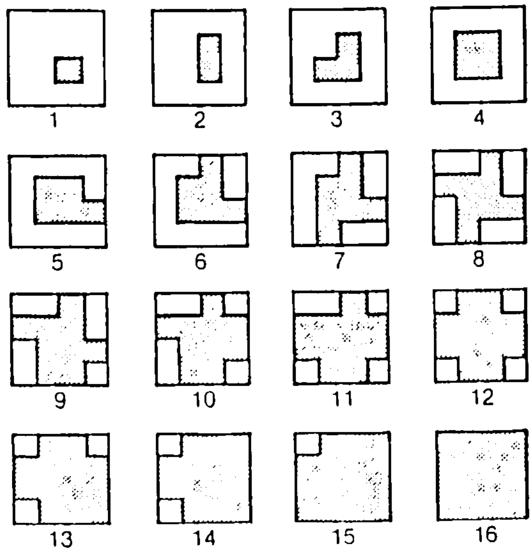


Figure 1

Halftone cell and pixels [10]

Figure 2

Possible levels with a 4 x 4 halftone cell [11]

Figure 1 shows a typical halftone cell for dot matrix systems. The basic halftone cell consists of p x q picture elements or pixels. Because each pixel maintains a constant area and position, the number of reflectance levels per pixel is limited to two--white (paper base) and a single reflectance system color. Due to this differing image structure, changes in density level are accomplished in two ways:

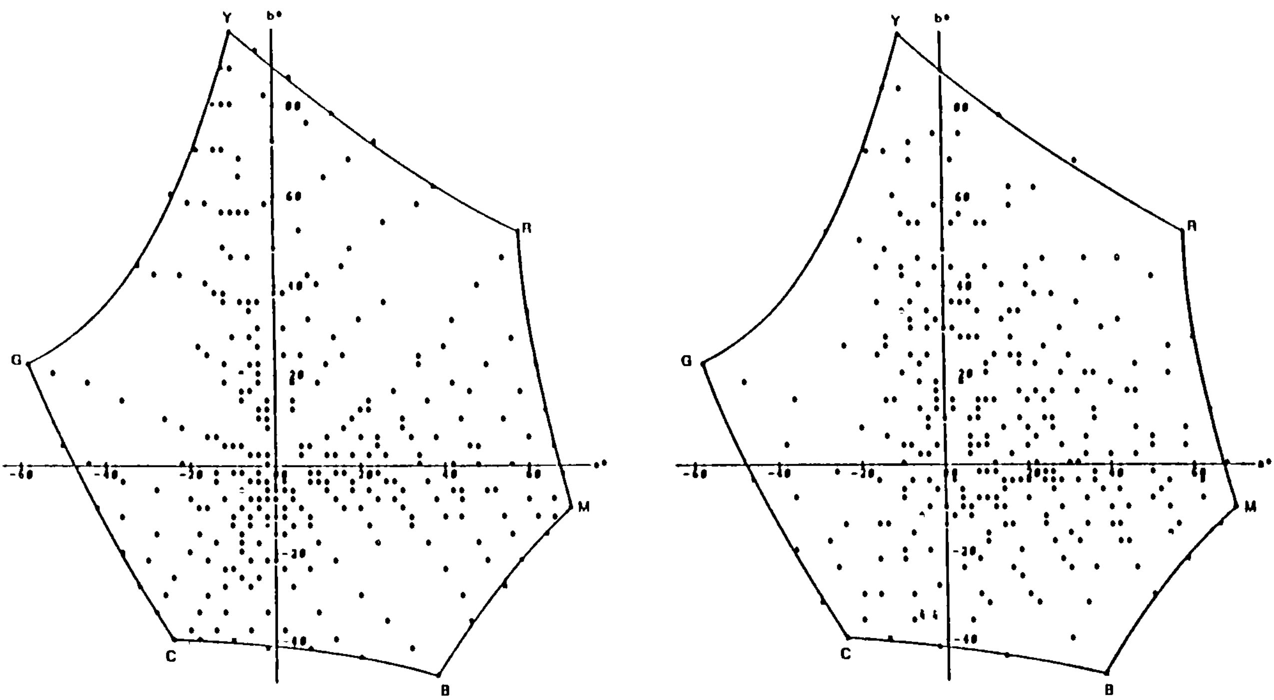
- change the system color printed at a pixel location, or
- change the number of pixels printed over the entire halftone cell.

These restrictions create a discrete and finite number of density steps available per cell. Figure 2 shows a 4x4 black-and-white cell where 17 levels are possible, each incremented in sixteenths.

Using a halftone cell of p x q = k pixels in size (figure 1), Engeldrum [5] illustrated that a simple combinatorial will provide the number of possible density levels or colors available per halftone cell. According to equation 11, n equals the total number of colors in the system, and k equals the total number of pixels per halftone cell. For example, a 4x4

$$C = \frac{(n + k - 1)!}{(n - 1)!\,k!} \tag{11}$$

halftone cell (k=16) with n=2 colors/pixel (white, black) would produce 17 different levels, as in figure 2. Using the same 4 x 4 cell size with n=8 colors/pixel (R,G,B,C,M,Y,Wht,Blk),



a) b)

Figure 3

Enumerated Colors for the following conditions [12]

6 Pixels / Halftone Cell
4 Colors / Pixel
343 Total Colors

4 Pixels / Halftone Cell
8 Colors / Pixel
330 Total Colors

245,157 colors are possible. One advantage here is the ability to enumerate the total number of possible colors. Presented in CIELAB space, figure 3 shows enumerated colors for two conditions.[12] The problem of quantizing the original image data, that is, reducing multi-bit information to binary form, remains.

Color Halftoning Algorithms

EZ Color Algorithm

A typical binary dot matrix color printer is capable of 8 distinct system colors, cyan, magenta, yellow, red, green, blue, black and white, using four colorants, cyan, magenta, yellow, and black. Red, green, and blue are created by overprinting of the appropriate colorants, and white is furnished by the paper base. These system colors create the color gamut of the printer. Each of these colors can be described as a unique triplet of tristimulus values, X, Y, Z.

If an original color "b" is represented by the tristimulus values X_b, Y_b, Z_b and the tristimulus values for each system color are known ($X_1, Y_1, Z_1, \ldots, X_8, Y_8, Z_8$), then the color is reproduced by determining the fractional area coverage of each system color. This is shown as equations 12a-c where a_i represents the fractional area coverage of the ith system color.

$$X_b = \sum_{i=1}^{8} X_i a_i \qquad Y_b = \sum_{i=1}^{8} Y_i a_i \qquad Z_b = \sum_{i=1}^{8} Z_i a_i \qquad \text{(12a-c)}$$

Though the original color is accurately reproduced when a_i is known, equation 12a-c exists as three equations with eight unknowns. The EZ color algorithm uses the following color formulation model, called the Structured Dot Theory (SDT)[5], to determine the solution for the fractional areas.

Structured Dot Theory

From the tristimulus values of the system colors, X_i, Y_i, Z_i, the EZ algorithm determines the chromaticity coordinates x_i, y_i using equation(s) 2. Using x_i, y_i as Cartesian coordinates in the 1931 C.I.E. chromaticity diagram, the system colors form a polygon representing the gamut of reproducible colors.

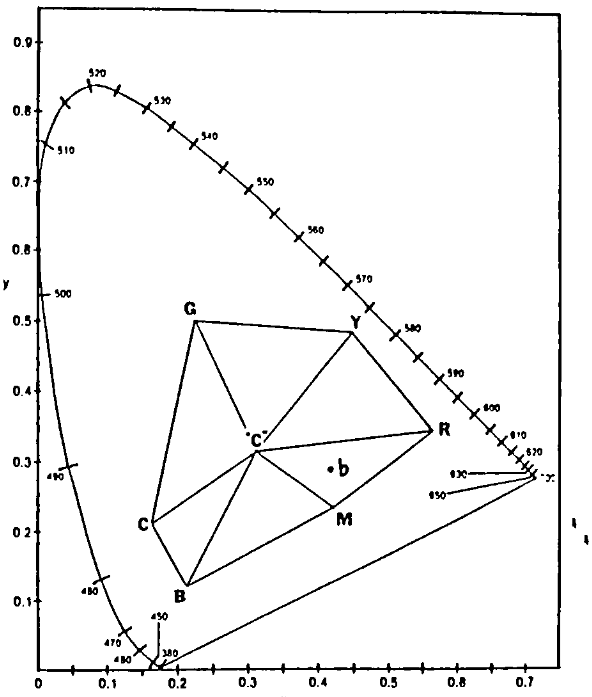


Figure 4

Ink Jet color gamut residing within the 1931 CIE chromaticity diagram. [5]
"C" is the source illuminant, "b" is the color to be reproduced.

Since additive mixtures lie along straight lines, intersecting lines between the source illuminant, "C" in figure 4, and the polygon corners provide triangular areas within which any point can be described as a combination of two colors plus black and white. Because of the binary nature of the output device (fixed colorants of single concentration) and the partitive color mixing of the EZ method, the intersecting lines can be assumed to be straight. For example, in figure 4, a mixture of red (magenta + yellow), magenta, black and white will reproduce the color labeled "b".[5] This method eliminates half of the eight possible colors needed for a match by knowing the location of the original color on the chromaticity diagram. The EZ algorithm finds the location of the original color using its hue angle. The hue angle (θ), shown in figure 4, is the angle an original color subtends from the positive x axis in the chromaticity coordinate system. It is given in equation 13.

$$\theta = arctangent\left(\frac{y}{x}\right)$$

- for x = 0 and y = 0

$$\theta = arctangent\left(\frac{y}{x}\right) + \pi \tag{13}$$

- for x < 0

$$\theta = arctangent\left(\frac{y}{x}\right) + 2\pi$$

- for y < 0

Once the hue angle of the original color is found, the algorithm compares it to the hue angles of the system colors. The correct system colors needed for reproduction of the original color are found when the following condition is met:

$$\theta_c\,(m) <= \theta\,(i) <= \theta_c\,(n)$$

where $\theta_c(m)$, $\theta_c(n)$ =hue angles of consecutive system colors, and

$\theta(i)$ = the hue angle of the color to be matched.

The system colors needed for a match are now identified.

Three equations can be written to relate the known tristimulus values of the four system colors to the known tristimulus values of the original color, as a function of fractional area coverage. Using the example of color "b", the equations are as follows:

$$
\begin{aligned}
X_b &= X_{red}a_{red} + X_{mag}a_{mag} + X_{wht}a_{wht} + X_{blk}a_{blk} \\
Y_b &= Y_{red}a_{red} + Y_{mag}a_{mag} + Y_{wht}a_{wht} + Y_{blk}a_{blk} \\
Z_b &= Z_{red}a_{red} + Z_{mag}a_{mag} + Z_{wht}a_{wht} + Z_{blk}a_{blk}
\end{aligned}
\tag{14a-c}
$$

Since the sum of the areas must equal 1, the total fractional area, the equation is rewritten, relative to black (Equation 15).

$$
\begin{aligned}
(X_b-X_{blk}) &= a_{red}(X_{red}-X_{blk}) + a_{mag}(X_{mag}-X_{blk}) + a_{wht}(X_{wht}-X_{blk}) \\
(Y_b-Y_{blk}) &= a_{red}(Y_{red}-Y_{blk}) + a_{mag}(Y_{mag}-Y_{blk}) + a_{wht}(Y_{wht}-Y_{blk}) \\
(Z_b-Z_{blk}) &= a_{red}(Z_{red}-Z_{blk}) + a_{mag}(Z_{mag}-Z_{blk}) + a_{wht}(Z_{wht}-Z_{blk})
\end{aligned}
\tag{15a-c}
$$

Once these fractional areas are determined, the black fractional area is given by equation 16.

$$a_{blk} = 1 - a_{red} - a_{mag} - a_{wht} \tag{16}$$

This method has a distinct advantage over most halftone reproduction theories (i.e., the three color Neugebauer equations) by including the black colorant explicitly in the initial formulation. In other theories, the black colorant can be formally or empirically incorporated, but a unique solution for the required fractional areas does not exist without additional assumptions.[5]

EZ dither technique

After the correct system colors have been identified and the fractional areas of each color determined, the area data must be quantized into binary form for the printer. In this example, only the magenta, yellow, and black colorants will be used by the printer, since the red system color is obtained by overprinting magenta and yellow. In fact, in every case, one of the system colors chosen for reproduction will be an overprinted color (red in this example). Therefore, the fractional area for this color is assigned by the EZ algorithm to the two printer colorants that create that color. For example, the algorithm creates four arrays to hold the fractional areas for each of the printer colorants, cyan, magenta, yellow, and black. The magenta array gets the value of a_{red}. The yellow array gets the value of a_{yellow} and a_{red}. The cyan array is empty, and the black array gets the value of a_{black}. Again, this method is unique in considering only those system colors required to create the color, and always includes the black colorant. For all arrays, the fractional areas are real numbers that range continuously from zero to one. The EZ algorithm quantizes and distributes these continuous area values into bitmaps (binary matrices) for printing using ordered dither.[13]

Ordered dither produces bitmaps by comparing continuous values (fractional areas), to pre-determined threshold values. The decision to print or not to print is based on whether the continuous values exceed the threshold values. The process is illustrated in figure 5.[14]

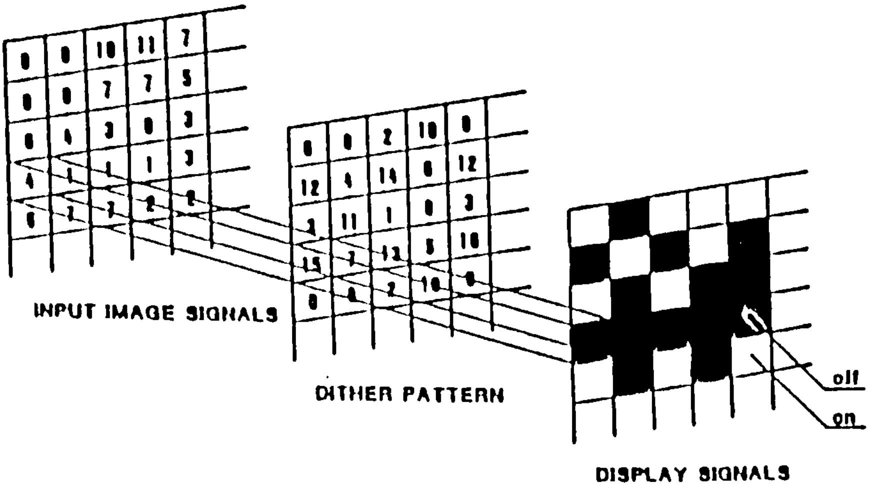


Figure 5

Schematic of conversion from continuous tone picture to bi-level image with dither technique.

An array of continuous values, representing a quantized, continuous tone image, is compared to a smaller array of threshold values, pixel by pixel. The smaller array of threshold values is the dither matrix. If a value from the quantized image is greater than the corresponding dither matrix value, a value of "1" is assigned to that same pixel position in a bit map. If the quantized image value is less than its corresponding dither matrix value, a value of "0" is assigned to the bit map. The dither matrix is repeated across the image to treat all image pixels. The quantization rule is represented by the formula shown in equation 17. At pixel position (x,y), CT(x,y) is the continuous tone image, BM(x,y) is the bi-level ("1" or "0") value of the bit map, and D(x,y) is the dither matrix threshold.

$$\begin{aligned} &BM(x, y) = 1 \quad \text{if} \quad CT(x, y) \geq D(x, y) \\ &\text{else} \\ &BM(x, y) = 0 \end{aligned} \tag{17}$$

The distribution of threshold values in the EZ dither matrices were patterned after the ordering scheme devised by Bayer,[15] using Limb's initial dither matrix.[16] Equations 18 - 20 show the recurrance scheme of Jarvis, Judice, and Ninke[17] for generating dither matrices.

Let

$$D^{2} = \begin{vmatrix} 0 & 2 \\ 3 & 1 \end{vmatrix} \tag{18}$$

and

$$U^{n} = \begin{vmatrix} 1 & 1 & \cdot & \cdot & 1 \\ 1 & 1 & \cdot & \cdot & 1 \\ \cdot & \cdot & \cdot & \cdot & 1 \\ 1 & 1 & \cdot & \cdot & 1 \end{vmatrix} \tag{19}$$

To create a dither matrix of D^n,

$$D^{n} = \begin{vmatrix} 4D^{n/2} + D^{2}_{00}\, U^{n/2} & 4D^{n/2} + D^{2}_{01}\, U^{n/2} \\ 4D^{n/2} + D^{2}_{10}\, U^{n/2} & 4D^{n/2} + D^{2}_{11}\, U^{n/2} \end{vmatrix} \tag{20}$$

For example, D^4, n=4

$$\begin{vmatrix} 4(0)+0(1) & 4(2)+0(1) & 4(0)+2(1) & 4(2)+2(1) \\ 4(3)+0(1) & 4(1)+0(1) & 4(3)+2(1) & 4(1)+2(1) \\ 4(0)+3(1) & 4(2)+3(1) & 4(0)+1(1) & 4(2)+1(1) \\ 4(3)+3(1) & 4(1)+3(1) & 4(3)+1(1) & 4(1)+1(1) \end{vmatrix}$$

$$D^{4} = \begin{vmatrix} 0 & 8 & 2 & 10 \\ 12 & 4 & 14 & 6 \\ 3 & 11 & 1 & 9 \\ 15 & 7 & 13 & 5 \end{vmatrix} \tag{21}$$

The integer values ranging from 0 to 15 in the D^4 example are the threshold values for quantizing a four bit image. Since the EZ algorithm created four arrays of real numbers

ranging from 0 to 1, the D^4 example serves only to show where to distribute the EZ threshold values (in the case of a 4x4 EZ matrix). In other words, the lowest EZ threshold value was placed in the same position as the D^4 "0" threshold, the highest EZ threshold in the position of the D^4 "15" threshold, and so on. The actual EZ threshold values were determined in the following manner.

Consider a 2x2 matrix used to halftone an image of fractional areas ranging continuously from zero to one. The matrix is divided into quarters with each quarter, or pixel, holding a discrete threshold value. This matrix quantizes fractional areas using the four discrete threshold values according to equation 17. According to Bayer, the threshold values would be distributed as the D^2 matrix, and appear as figure 6. After halftoning, this 2x2 matrix can provide five possible fractional areas, 0, .25, .5, .75, and 1. Incrementing the threshold values by eights ($\frac{1}{8}, \frac{3}{8}, \frac{5}{8}, \frac{7}{8}$), creates a "rounding" quantizer. Original area values are quantized to the next highest area only if they exceed the midrange of the five possible matrix

.125	.875
.375	.625

Figure 6

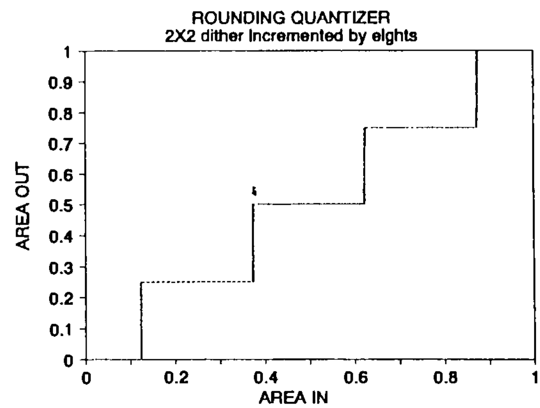


Figure 7

areas. This is analogous to the mathematical operation of "rounding". For the various cell sizes used in this study, the threshold values, T_j, and their placements were determined using the following expression.

$$T_j = \frac{1}{2p} + \frac{j}{p} \tag{22}$$

- where j = the jth sequence value from Bayer D^n matrix; j = 0,1,2,...,p-1.
- p = total number of pixels per cell to be created. i.e., n = 16 for a 4x4 cell

For example, using the Jarvis, Judice, and Ninke method, a 4x4 Bayer matrix is created. The value of p is 16. The threshold values from the D^4 example represent the "j" index in equation 22. The "j" index values are used for both computation and placement of area thresholds in the EZ matrix. In the second row, third column of the D^4 Bayer matrix, the sequence number is fourteen. Fourteen becomes the "j" index. Using equation 22, the EZ area threshold becomes

$$T_{14} = \frac{1}{2 \times 16} + \frac{14}{16} = \frac{29}{32}$$

Therefore, the area threshold value for the second row, third column of the EZ matrix is $\frac{29}{32}$. Using this technique, four matrices each with 2^n x 2^n cells were created, n = 1,2,3,4.

Next, the EZ algorithm compares the arrays of fractional areas created for each of the printer colorants to the EZ dither matrices, according to equation 17. Bitmaps are created for each of the printer colorants. If any one pixel in the bitmaps calls for C, M, and Y to be printed,

the bitmaps are set to "0" and black is printed instead. After quantizing, the average tristimulus value of the halftone cell is determined and compared to the original.

Error Diffusion

Consider figure 8, a one-dimensional "error diffusion" model. An original image reflectance (solid dot) of .20 is compared to the reflectances available from the reproduction system (0 or 1). The decision to print a pixel is based on the shortest distance, or error encountered between printing one of the two reflectance values. In this case, the distance between .20 and 0 is less than the distance between .20 and 1. Therefore, no pixel is printed on the reproduction in this position. The remaining difference, .20, represented as a solid arrow, is added to the next original pixel. The second pixel had a value of .25. However, modified by the added error of .20, this pixel now has a value of .45 (.25 + .20). It is evaluated using the same process. It also creates less error if the reproduction pixel is set to zero. The remaining difference (.45 - 0 = .45) is added to the third original pixel. The third pixel rises to .85 with the error addition. In this case, less error occurs if the reproduction pixel is printed ("1"). The ouput signal (lower curve) goes to 1 for this pixel position, and a pixel is printed on the reproduction. The remaining error (.85 - 1 = -.15) is added to the fourth original pixel, lowering it from .45 to .30, and the process continues in

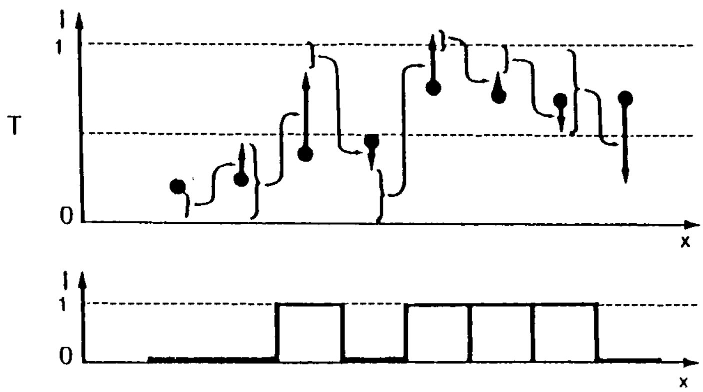


Figure 8

One-dimensional error diffusion model [18]

this way. The distribution (diffusion) of difference (error) is the "error diffusion" for which the process is named. The net effect is a binary image of identical dot areas, created from the error of quantizing multiple levels into two.

A two-dimensional error diffusion algorithm modifies the value of several neighboring image points in both the x and y positions. In figure 9, the error is distributed according to the scheme developed by Floyd and Steinberg.[19] Developed by trial and error, it produces a checkerboard pattern in a region of middle gray.

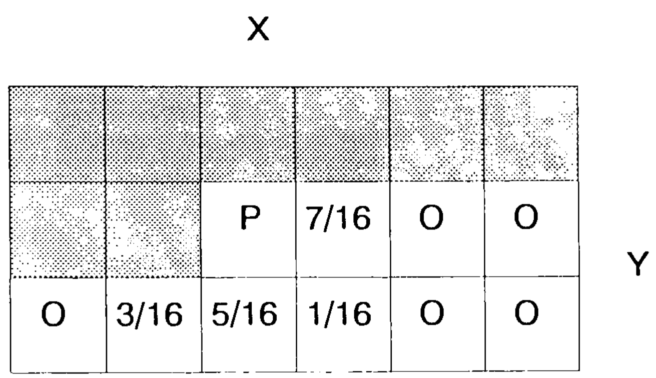
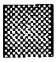


Figure 9

Two-dimensional error diffusion reflectance error allocation

- where P = pixel currently being processed.
- n/16 = amount of error from pixel P being assigned to the yet unprocessed neighboring pixels.
-  = previously processed pixels.
- O = pixels yet to be processed.

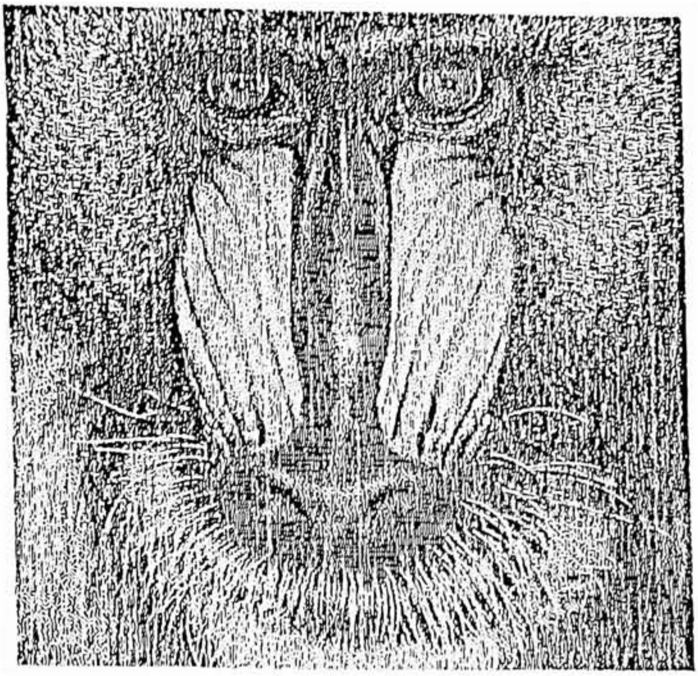


Figure 10

An image halftoned using Floyd and Steinberg's error diffusion

An example of the binary output from this technique is shown in figure 10. According to Floyd and Steinberg, this algorithm "has an obvious extension to devices that allow several brightness levels at each pixel." [20] Here, the "several brightness levels" can be analogous to the 8 colors of the output device, where the diffusion of error results from the comparison of image to system colors. The error diffusion algorithm used in this work distributed error according to the Floyd and Steinberg pattern. It compared X, Y, and Z tristimulus values between image and system colors. Error was diffused among neighboring image pixels. Error was determined in the following way.

STEP 1

Tristimulus values of an original color, X_{orig}, Y_{orig}, Z_{orig}, were compared to the respective tristimulus values of the eight system colors, ($X_{sc(i)}$, $Y_{sc(i)}$, and $Z_{sc(i)}$), of the printer device, where i represents the ith system color.

STEP 2

The difference between the original and system color tristimulus values was recorded as the tristimulus error ($X_{e(i)}$, $Y_{e(i)}$, and $Z_{e(i)}$).

$$
\begin{aligned}
X_{e(i)} &= (X_{orig} - X_{sc(i)}) \\
Y_{e(i)} &= (Y_{orig} - X_{sc(i)}) \\
Z_{e(i)} &= (Z_{orig} - X_{sc(i)})
\end{aligned}
\tag{23}
$$

STEP 3

The euclidean distance ED_i, between the original color and each system color was determined.

$$ED_i = \sqrt{X_{e(i)}^2 + Y_{e(i)}^2 + Z_{e(i)}^2} \tag{24}$$

The ith system color that gave the smallest euclidean distance error was printed as the reproduction, (X_{repro}, Y_{repro} , Z_{repro}). For example, if the fourth system color is selected,

$$
\begin{aligned}
X_{repro} &= X_{sc}(4) \\
Y_{repro} &= Y_{sc}(4) \\
Z_{repro} &= Z_{sc}(4)
\end{aligned}
$$

STEP 4

The trisimulus differences between the original color and the fourth system color are respectively added to neighboring original colors, $X_{neigh\ (x,y)}$, $Y_{neigh\ (x,y)}$, $Z_{neigh\ (x,y)}$, where (x,y) represents the x,y coordinate relative to the original color's pixel position.

$$X_{neigh\ (1,0)} = X_{neigh\ (1,0)} + (7 / 16 * X_{e\ (4)})$$

$$\text{(same for } Y_{neigh\ (1,0)} \text{ and } Z_{neigh\ (1,0)}\text{)}$$

$$X_{neigh\ (1,1)} = X_{neigh\ (1,1)} + (1 / 16 * X_{e\ (4)})$$

$$\text{(same for } Y_{neigh\ (1,1)} + Z_{neigh\ (1,1)}\text{)} \qquad (25)$$

$$X_{neigh\ (0,1)} = X_{neigh\ (0,1)} + (5 / 16 * X_{e\ (4)})$$

$$\text{(same for } Y_{neigh\ (0,1)} \text{ and } Z_{neigh\ (0,1)}\text{)}$$

$$X_{neigh\ (-1,1)} = X_{neigh\ (-1,1)} + (3 / 16 * X_{e\ (4)})$$

This four step process was repeated for all pixels of the original image.

Color Reproduction Objectives

Hunt[21] defined color reproduction objectives for various applications and operating limits. This study considered those objectives for dot matrix color applications. Which color reproduction objective to select included consideration of which color reproduction **type**[22] (duplication or copying), would be used.

Duplication is reproduction using the same colorants for both original and reproduction, typical of color slide duplication or color electrophotography (2nd to nth generation). The second type is **copying,** where originals, created using an infinite variety of colorant sets,

are reproduced using a single set of colorants. This is generally the case in most color reproduction. For an office environment, the type of reproduction will vary from duplication to copying, depending on the generation of the current "original." Therefore, the color reproduction objective must be capable of handling both types of reproduction notably, demands for nth generation copies without color degradation, and a relative insensitivity to the variety of colorants used to produce any original. A colorimetrically based reproduction objective is capable of handling both reproduction types with their inherent difficulties, being limited only by the image noise characteristics of the reproduction process. This is important when, for many situations, it is impossible for the reproduction to provide the same luminance as the original. For example, a white object in bright sunlight reproduced by a reflection print seen in artificial light. Therefore, a colorimetric reproduction objective based on the measurement of luminances relative to a well-lit reference white was preferred.

Colorimetric Color Reproduction

The Colorimetric Color Reproduction objective was chosen because it is colorimetrically based and treats luminance as a relative quantity. In Colorimetric Color Reproduction, the original and reproduction have the same tristimulus values and relative luminances, and the color match is metameric. However, the illuminant used for the reproduction process must have the same spectral power distribution as the viewing illuminant in order to provide a proper color match. Hunt [23] suggests that Colorimetric Color Reproduction is a good criterion for reflection prints if

- The surround is similar to the original, that is, reproduction and original being viewed in light of the same color.
- Differences in luminous intensity between original and reproduction are relative, measured against a reference white.

Here, the justification is the fact that the eye adapts to the available surround and recognizes objects more by their relative rather that actual luminances. The office environment provides the surround for the original and the reproduction, both in terms of spectral quality of the light source and the absolute level, for color images on paper. Under these conditions, the Colorimetric Color Reproduction objective is applicable and this reproduction objective was chosen for evaluation.

Research Objective

The purpose of this thesis was to investigate the Colorimetric Color Reproduction accuracy of two halftoning algorithms, the EZ algorithm and Floyd and Steinberg's error diffusion. Three colorant sets, representative of electrophotographic, thermal, and ink jet dot matrix printer technologies, and four halftone cell sizes, (2x2, 4x4, 8x8, and 16x16) were also investigated to determine the affects of these variables, if any, on the performance of the algorithms.

TOOLS AND TECHNIQUES

GENERAL

The original and reproduction images for this study consisted of numerical tristimulus arrays. A uniformly distributed random number generator created original images consisting of 500 X, Y, Z triplets, that is, digital representations of colors in tristimulus space. Original images were generated from within the color gamut of each colorant system. Each halftoning algorithm (EZ and error diffusion) processed the original images and produced halftone "images" which were spaced averaged to produce tristimulus values.

COLORANTS AND PRINTER SYSTEMS

Three different color dot matrix systems--electrophotography, thermal transfer and inkjet--provided colorants for this study. These systems represent the diversity of the dot matrix printer types currently available, ranging from low cost, low speed printers (inkjet), to high cost, high speed printers (electrophotography). Solid area patches, generated by each machine on the paper recommended by the manufacturer, created the colorant samples cyan, magenta, yellow and black. Black toner, ribbon, or ink created the black samples (not a process black of cyan, magenta and yellow overlays). Overprinting created red, green and blue from the two appropriate subtractive primaries. Paper base was used for white. For each colorant set, the number of system colors totaled eight.

Electrophotography

Canon's NP color copier provide the colorant set from electrophotography. This machine is an analog optical device, not a matrix technology. A full color binary dot matrix electrophotographic printer did not exist in production at the time of the study. However, the colorant set of the NP color copier is representative of what a printer of this process type would use. The Canon NP copier is a three colorant system using cyan, magenta and yellow. Sharp SF-90 toner provided the black colorant to complete a four colorant printer "system". The Colorocs Corporation prototype electrophotograhic laser printer used the Canon colorants to create sample patches of all seven system colors. Nominal printer settings for graphic output were used. Hammermill "Laserprint" paper was used for the eighth system color, white.

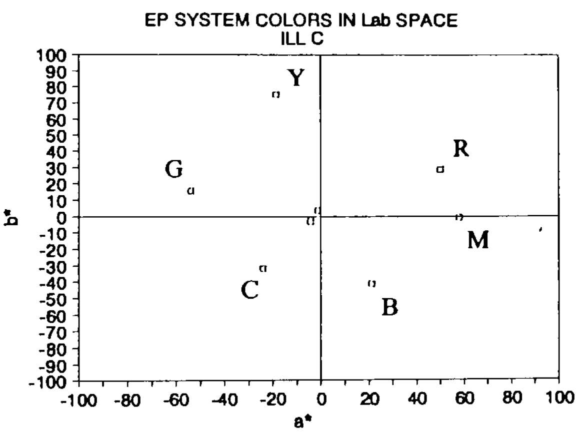


Figure 11

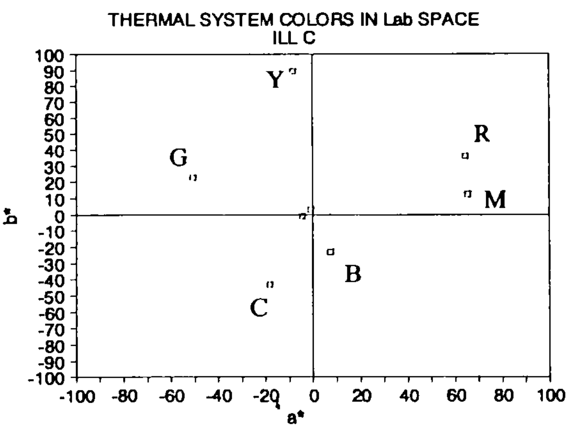


Figure 12

Thermal Transfer

Mitshubishi's G500 color printer provided the colorant set for dot matrix thermal transfer. A test pattern generated by this machine provides all the system colors including black. Mitshubishi's recommended thermal paper provided white.

Inkjet

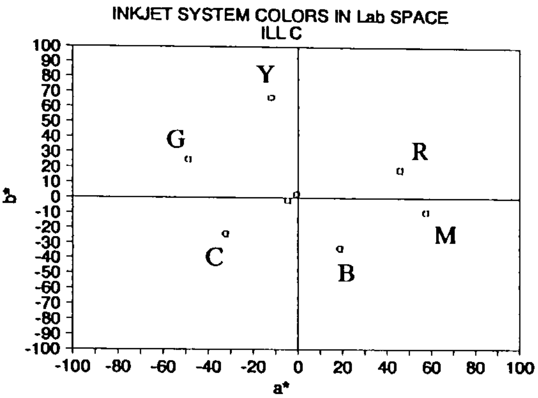


Figure 13

Quadram's Quadjet provided the colorant set for a color binary dot matrix inkjet printer. A test pattern created using PC Paintbrush output the seven system colors onto solid color patches. Quadjet's roll paper was used for white.

Measurement of colorant sets

A MacBeth 1500 Colorimeter using the C.I.E. 1931 Standard 2^{o} observer with the C.I.E. ill. "C" was used to measure the tristimulus values from each color. The colorants were measured in the colorimeter's diffuse mode, where the specular component of the source illuminant was excluded under a d/0 (diffuse/normal) geometry. The ultraviolet portion of each colorant (350-400nm) was not considered in the tristimulus equations, as the colorimeter integrated only the visible portion from 400 to 700 nm in 20 nm increments, and provided only illuminant C. However, since the objective was to evaluate the perfomance of the algorithms using a numerical comparison between randomly generated original tristimulus values and reproduced tristimulus values, any illuminant would have provided a suitable source.

COMPUTER SIMULATION

Language and Equipment

Programs were originally written in interpretive BASIC using Microsoft Inc.'s GW-BASIC and the Digital Equip. Corp. Rainbow 100B personal computer. Programs were subsequently rewritten to operate with Microsoft Inc.'s QuickBasic version 2.0, to decrease processing time. These programs were run in an IBM AT type computer. Six separate stand-alone programs were written for image creation, processing and data handling.

Random Color Generator Program (RNDGEN)

This program created three original images, one for each of the three printer systems (ink jet, thermal, electrophotography). The images consisted of numerical arrays of 500 colors each, described as X,Y,Z tristimulus values. A uniform random number generator created the tristimulus value of each color. Only original image colors that existed within the gamut of a system color set were used. A brief discussion of the program follows.

A system color set consisted of XYZ tristimulus values of the primaries red, green, blue, cyan, magenta, yellow, white, and black. The hue angle for each color was found using equation 13.

Three random numbers were generated, each ranging from 0 to 1 (r_1, r_2, r_3). These random numbers were multiplied respectively by the tristimulus values of the source illuminant, Illuminant C (X_o, Y_o, Z_o). Since relative luminances were used, $Y_o = 100$.

$$r_1 * X_o = RX$$

$$r_2 * Y_o = RY$$

$$r_3 * Z_o = RZ$$

This effectively raised the value of the random number into a range typical of tristimulus values. The resulting triplet (RX, RY, RZ) ranged in value from zero (black) to the tristimulus value of the source illuminant depending on the value of the respective random number (r_1, r_2, r_3).

Next, this triplet, or created color, was tested to see if it existed inside the gamut of the color system set used. The location of the created color in chromaticity space was found by determining its chromaticity coordinates and hue angle (equations 2,13). Since the hue angles of the system colors and the created color are known, the two system colors required for reproduction were found when the following condition was met:

$$\theta_c\,(m) <= \theta\,(i) <= \theta_c\,(n)$$

where $\theta_c\,(m)$, $\theta_c\,(n)$ =hue angles of consecutive system colors

$\theta\,(i)$ = the hue angle of the created color

Once the two system colors were found, the fractional areas required for reproduction (the areas of the two system colors plus black and white), were determined using equations 15 - 16. If any fractional area was greater than 1 or less than 0, the created color was considered out of the printer system color gamut (that is, unreproducible by the system color set), and discarded. Otherwise, the created color was saved as an original color and the program continued until 500 original colors were created. These colors were written to a file to be used by the EZ and error diffusion algorithms as an original image.

EZ Color Halftoning Algorithm Program (EZ)

This program used one of four EZ threshold matrices (2x2, 4x4, 8x8, 16x16) to halftone the 500 original colors created by the random color generator, RNDGEN. Each EZ matrix was used to process each original image color using its respective system color set.

After loading the system color, original tristimulus, and an EZ matrix, an original color was selected and loaded into all locations of an array equal in size to the EZ matrix. The location of the original color in chromaticity space and the two system colors required for reproduction were found using equations 2 and 13.

Next, the program computed the fractional areas required of the selected system colors plus black and white using equations 15 - 16. These fractional areas, ranging from 0 to 1, were written to four arrays, representing cyan, magenta, yellow and black. Because of the circular ordering of system colors in chromaticity space (red, yellow, green, cyan, blue, magenta) one color would always be a subtractive primary or colorant (cyan, magenta, yellow) and the other an additive primary (red, green, blue). The subtractive primary percentage was placed directly in its corresponding array. The additive primary percentage was recorded in **both** array positions corresponding to the subtractive primaries which create that additive primary. For example, if the two system colors selected were red and magenta, and their respective percentages were 25% and 50%, the magenta array would receive 50% from the magenta, plus another 25% representing the magenta portion of the red fractional area (magenta + yellow = red). The yellow array position would then receive 25%, representing the yellow portion of the red fractional area. The cyan array would remain empty. In every case, one of the three CMY arrays remains empty. From equation 16, the amount of black fractional area is stored to the black array. The arrays, now holding the fractional areas of the two selected system colors plus black were quantized into bitmaps using the EZ matrix according to equation 17. The bitmaps contain subtractive primary and black area data, in binary form, ready to send to the printer.

The reproduction tristimulus value was obtained by summing the system color tristimulus values according to the number of occurrences they realized in the bitmap arrays. This sum

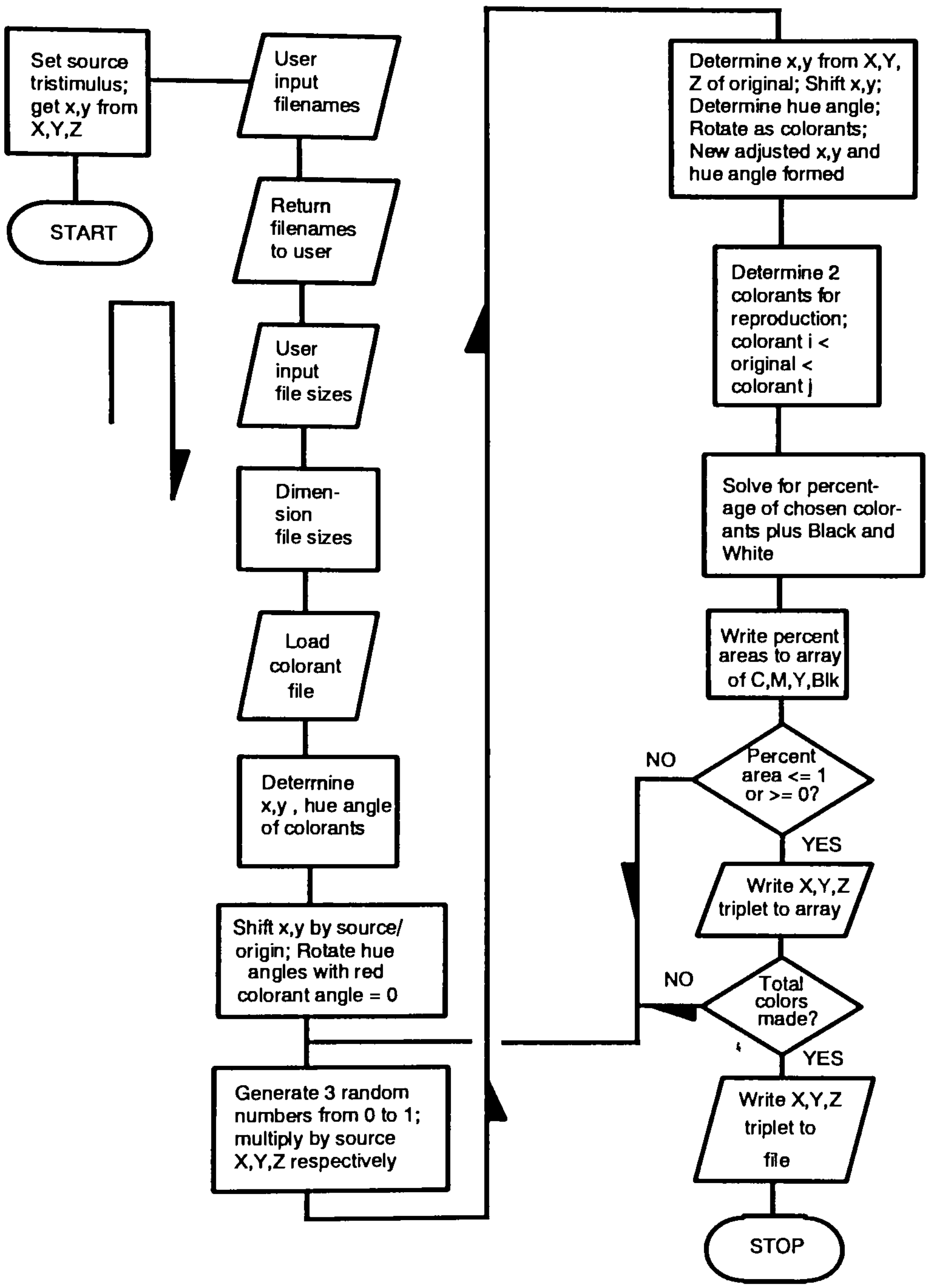


Figure 14 Flowchart of the Random Color Generator

was averaged according to cell size. As an example, in the 2x2 case, four bit maps are created containing four pixel locations each. The number of occurrences, or bits turned on (receiving a value of 1 according to equation 17) in each array is read. If the yellow array has 1 bit on, the magenta array has 3 bits on, the cyan array no bits on, and the black array no bits on, the bitmaps would appear as in figure 15.

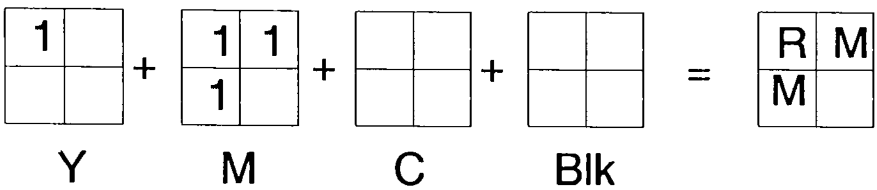


Figure 15 Bit Maps

The number of occurances of each system color is assigned to n_i, where "i" is the ith system color. From figure 15, n_i is as follows:

$n_y = 0$ $n_m = 2$ $n_c = 0$ $n_r = 1$ $n_g = 0$ $n_b = 0$ $n_w = 0$ $n_{blk} = 0$

The reproduction tristimulus (Xr, Yr, Zr), was computed using the following equations, (26a-c), where T = total number of pixels per array, i.e., 2x2 = 4, 4x4 = 16, etc.

$$X_r = \frac{\sum_{i=1}^{8} n_i X_i}{T}$$
$$Y_r = \frac{\sum_{i=1}^{8} n_i Y_i}{T} \qquad (26a\text{-}c)$$
$$Z_r = \frac{\sum_{i=1}^{8} n_i Z_i}{T}$$

This average, representing the reproduction tristimulus value, was compared to the original color tristimulus value. L*, a*, b*, Hue angle (both original and reproduction), ΔE^*_{ab}, and Metric chroma ratio was determined for each of the 500 colors.

Error Diffusion Color Halftoning Algorithm (ERROR)

This program used error diffusion to process the 500 colors in XYZ triplet form from each original image. The program generated, for each of the original 500 colors, a three dimensional 17x17 matrix where each dimension represented the X,Y, and Z respectively of the original color (essentially creating a large work space for the diffusion process of the algorithm). It compared the XYZ triplet of the first pixel in this 17x17 matrix to the eight XYZ system colors. The Euclidean distance between the original XYZ and each system color XYZ was calculated (equations 23 - 24). The system color which gave the least distance was chosen and printed. The tristimulus differences between this chosen system color and the original color were determined. The difference was allocated respectively among the remaining, and as yet untreated, pixels in the 17x17 array, according to the Floyd and Steinberg diffusion pattern (figure 9). After the entire 17x17 pixel array was processed, 2x2,

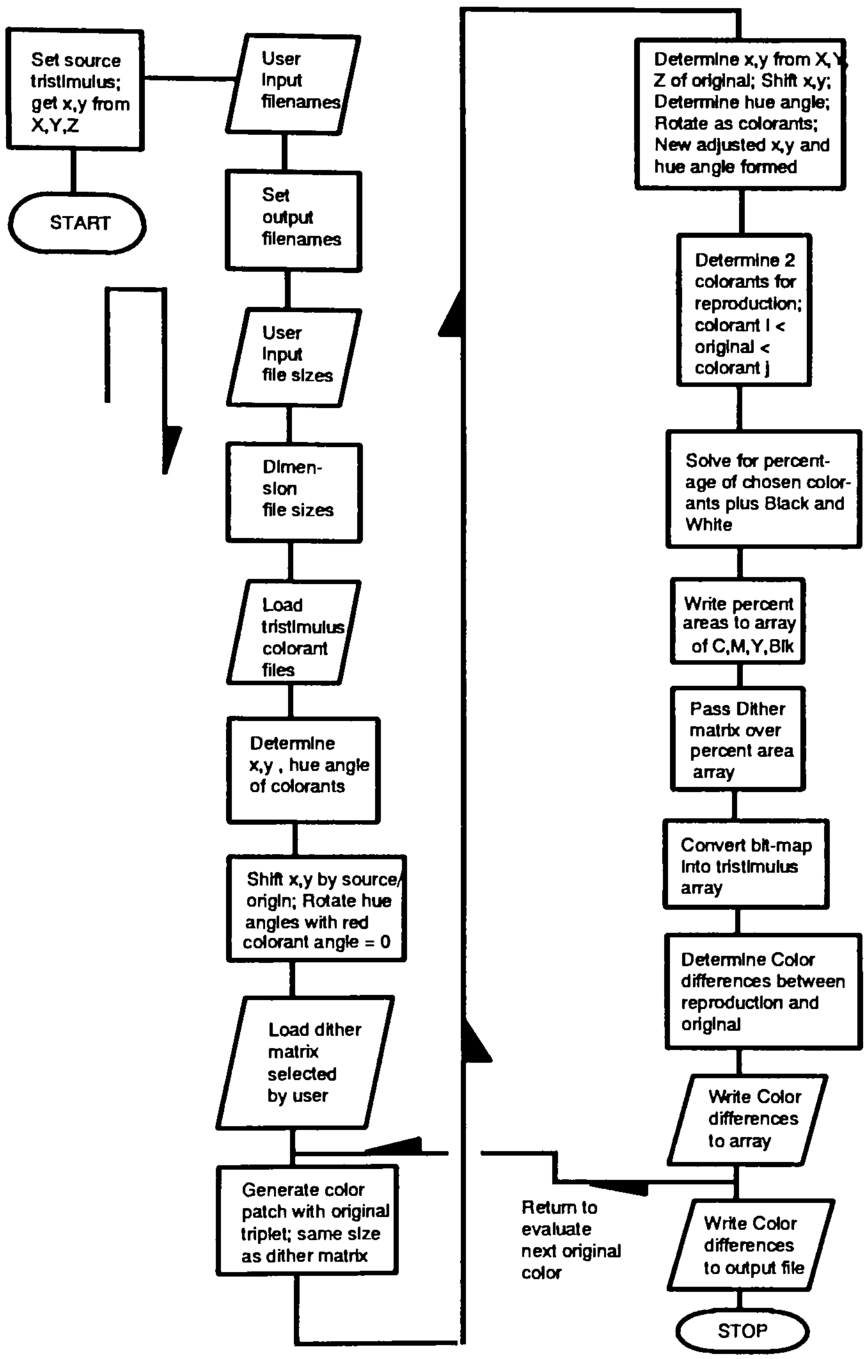


Figure 16 Flowchart of the EZ Color Halftoning Algorithm

4x4, 8x8 and 16x16 pixel areas were selected from the array. The average tristimulus values for each area was computed by summing over the respective area dimensions and dividing by the number of pixels per area. This became the reproduction tristimulus value. It was compared to the original color tristimulus value. For each colorant set and pixel area, L*, a*, b*, Hue angle (both original and reproduction), ΔE^*_{ab}, and Metric Chroma Ratio was determined. This was repeated for all 500 colors.

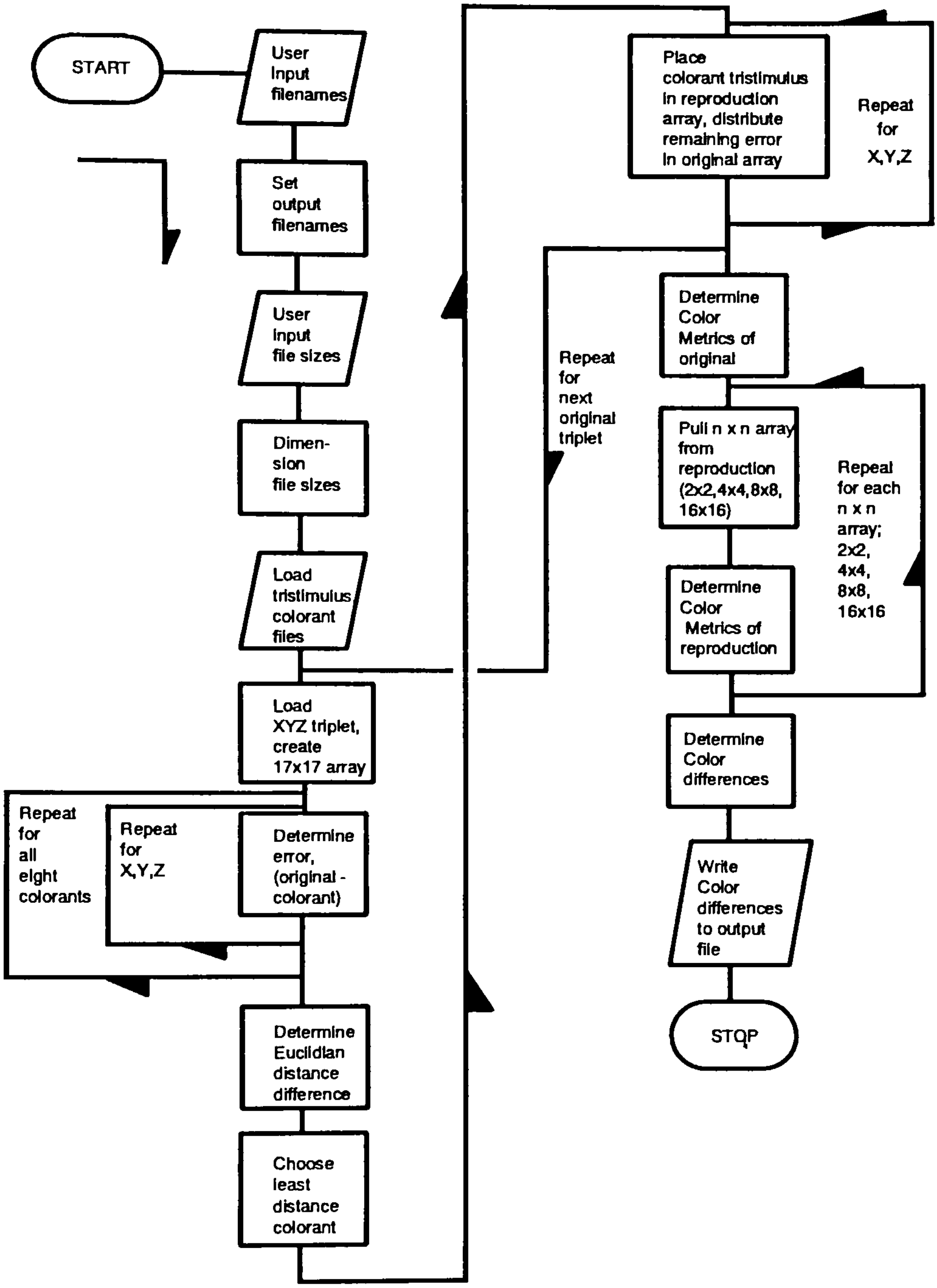


Figure 17 Flowchart of Error Diffusion Color Algorithm

COLOR METRICS

L* reproduction vs. L* original

Colorimetric Color Reproduction gives 1:1 response in L*, matching equal lightness between original and reproduction. This can be represented as a 45 degree line in a graph of L*reproduction versus L*original. Any deviation from this 1:1 line represents an inability to provide exact lightness rendition. Deviations about the 1:1 line were determined from the square root of the sum of the differences squared of the L*original and L* reproduction values according to equation 27.

$$s = \sqrt{\frac{\sum_{i=1}^{500} \left(L^*_{i_{orig}} - L^*_{i_{repro}}\right)^2}{499}} \tag{27}$$

Hue angle reproduction vs. Hue angle original

The hue circle diagram is used to reveal shifts in hue from original to reproduced color. The diagram shows if the original and reproduction hue angles are identical by displaying lines between two concentric circles, representing the original and reproduction hue angles. If the angles are identical, the lines will be straight, that is, a radius of the circle, and the system is said to provide Colorimetric Color Reproduction.

A graph of hue angle from the reproduction versus original hue angle was also made. The original hue angle was calculated using the original color a*, b* values. The reproduction hue angle was calculated from the a*, b* values of the reproduction color, obtained from averaging the colors present in the halftone cell. A 1:1 curve suggests a perfect hue match between reproduction and original. Any deviation from this curve, and a color shift has occurred. Deviations about the 1:1 line were determined from the square root of the sum of the differences squared of the original hue angle and the reproduction hue angle.

$$s = \sqrt{\frac{\sum_{i=1}^{500} (\theta_{i_{orig}} - \theta_{i_{repro}})^2}{499}} \qquad (28)$$

This approach enables a more quantitative analysis of the hue reproduction results, especially with the number of data points involved.

L* reproduction vs. Metric chroma ratio

Metric chroma in CIELAB is described as the magnitude of the two dimensional a*, b* vector from the origin of the a*, b* diagram to the color in question (equation 7). If the chroma of an original color and its reproduction are identical, the ratio of the chromas or vector magnitudes, reproduction to original, would be one. Since chroma does not account for differences in lightness, or L*, the curve of L* reproduction versus chroma ratio is useful for determining chroma differences across the lightness range. If the ratios plot a near vertical line at a chroma ratio of one, the system is said to exhibit exact metric chroma reproduction at all lightness levels. Graphs of L*reproduction vs metric chroma ratio were made for all conditions. Average metric chroma ratio and RMS deviations were also calculated.

DeltaE vs. cell size

A graph of ΔE^*_{ab} vs. cell size would show the overall color matching ability as the number of pixels in the halftone cell increase. As the number of pixels increase, the probability of matching an original color increases. Since quantization error is the primary difficulty binary systems have when treating multi-bit information, the ΔE^*_{ab} vs. cell size plot provides an overall comparison of the two algorithms. A plot of average color difference versus cell size for each algorithm was made, and average ΔE^*_{ab} and RMS deviations were calculated.

DeltaE histogram

ΔE^*_{ab} histograms were provided as an aid in visualizing the distribution of ΔE^*_{ab} values in each system/algorithm combination.

RESULTS AND DISCUSSION

L* reproduction vs. L* original

Two curves of L*reproduction vs. L*original for the 2 x 2 cell sizes are given below (figures 18 and 19). They represent the EZ and error diffusion algorithms respectively, using the electrophotographic system color set. The 2 x 2 halftone cell effectively reduces the number of lightness levels available for reproduction because of its smaller cell size. Therefore, the error incurred when quantizing the continuous image data was high, and resulted in a "stepping" effect where colors tend to cluster towards the closest available lightness level.

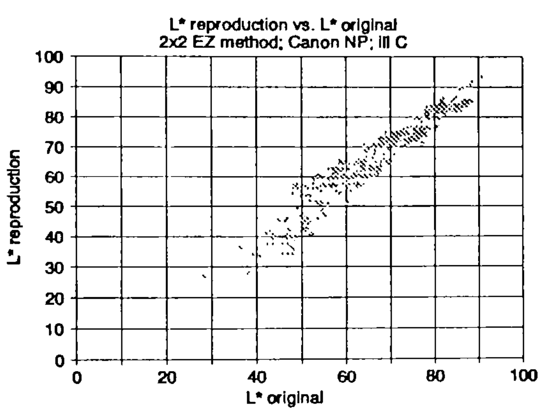


Figure 18

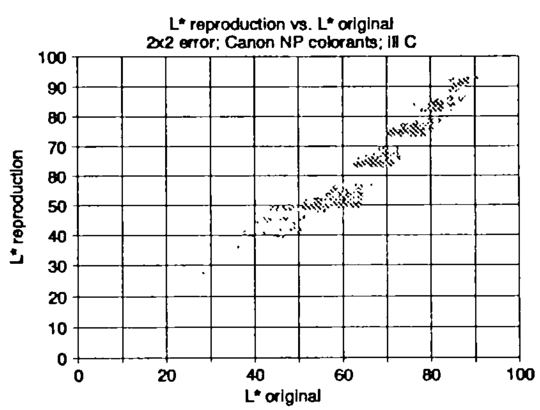


Figure 19

Further investigation revealed that this system color set contained 3 colors with L* values of 50 ± 1. This resulted in reducing the number of effective lightness levels to five for the entire system color set.

Where the stepping effect is visible in both halftoning algrotihms, the delineation of steps was more pronounced in the error diffusion algorithm. This indicates the algorithm's inability (for these conditions), to match the quantize error handling of the EZ method. As a result, the error diffusion algorithm might produce more visible contouring of images.

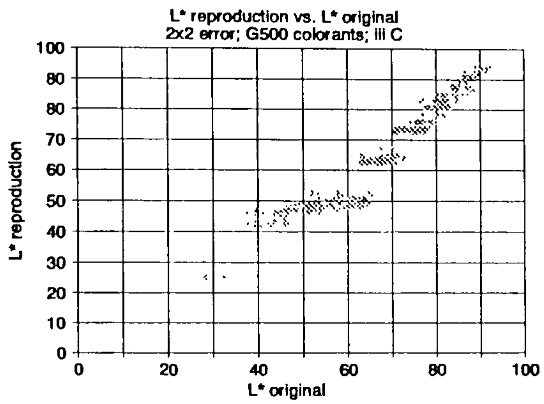


Figure 20

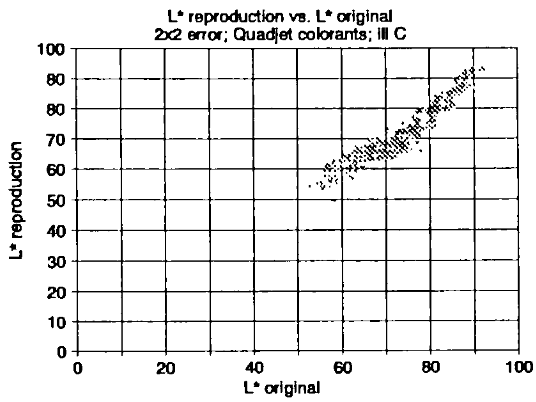


Figure 21

Lightness reproduction stepping is also more pronounced using the error diffusion algorithm with the thermal colorant set (figure 20). Because the thermal colorants were the most saturated, the gamut is larger and would require more density levels in the halftone cell to match a given color. For comparison, the inkjet colorant set, with the smallest gamut size, shows little or no stepping (figure 21). Once the cell size increased to 4 x 4, the stepping effect is virtually gone for all conditions as shown in figure 22.

Another curious result was a visible slope change in the L* curve for the error diffusion algorithm (figure 22). This occurred in the same L* region where four of the eight system

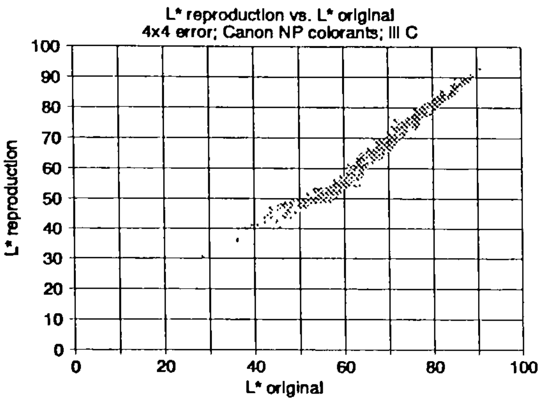


Figure 22

colors were located (Red,Green,Magenta L*=50 ± 1; Cyan L*=57). It is theorized that the algorithms selection of system colors at this lightness (euclidean distance determination) was driven more by X or Z tristimulus differences than Y tristimulus. As a result, the lightness remained relatively constant regardless of the system color chosen to minimize error. Floyd and Steinberg's comment of extending the algorithm to devices that allow multiple "brightness" levels[20] might come with a caveat when dealing with color. That is, an uneven distribution or clumping of lightness values from system colors could create non-linear responses in lightness reproduction.[24]

The deviations from a 1:1 L*reproduction versus L*original line for all conditions are given in Tables 1 and 2. As expected, L* deviations decreased with increasing cell size. However, the EZ algorithm showed better L* reproduction accuracy. For example, a 16 x 16 area is required using error diffusion to match the L* performance of a 4 x 4 EZ halftone cell.

Standard Deviations about a 1:1 of L* vs L*, EZ Algorithm

Table 1

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electro-photography	3.67	.96	.25	.11
Thermal	4.01	.91	.26	.09
Ink Jet	2.51	.60	.16	.06
AVERAGE	**3.40**	**.82**	**.22**	**.09**

Standard Deviations about a 1:1 of L* vs L*, Error diffusion

Table 2

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electrophotog-raphy	4.82	2.81	1.62	.98
Thermal	5.46	3.21	1.81	1.14
Ink Jet	3.62	1.83	.95	.55
AVERAGE	**4.63**	**2.62**	**1.46**	**.89**

Hue angle

The 2 x 2 cell / area hue circle diagrams show poor hue matching for both algorithms (figures 23 and 24). In these cases, the hue matching performance is impaired by quantization error.

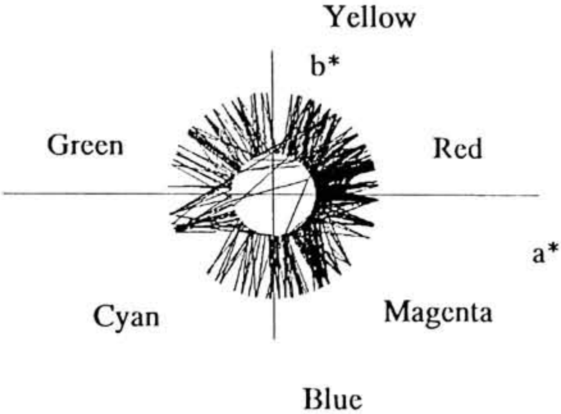


Figure 23
EZ Method; 2 x 2 cell; EP colorants

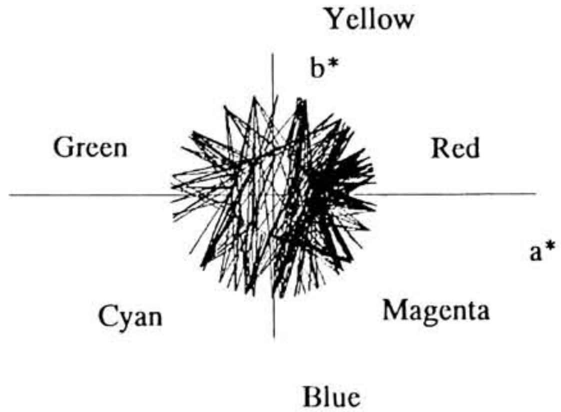


Figure 24
ED Method; 2 x 2 cell; EP colorants

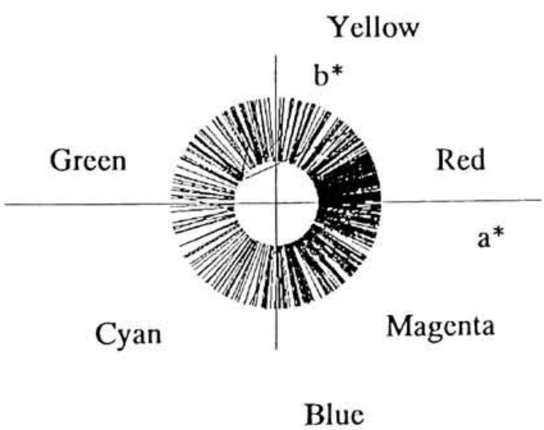


Figure 25
EZ Method; 8 x 8 cell; EP colorants

The EZ method dramatically improves hue reproduction, compared to the error diffusion algorithm, as the halftone cell is increased. Figures 25 and 26 are hue circle diagrams for both algorithms using 8 x 8 pixel areas. In figure 25, the hue circle diagram of the 8 x 8 EZ halftone cell, almost all lines are radially placed and hue matching appears excellent. Compare this to the hue circle diagram of

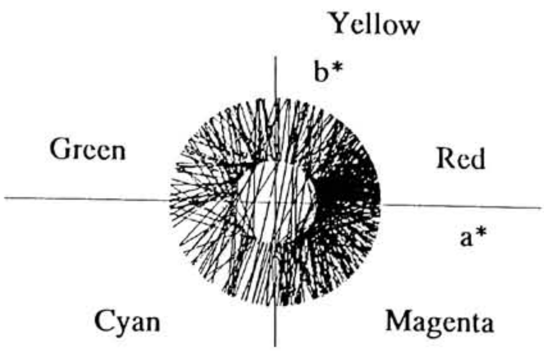


Figure 26
ED Method; 8 x 8 cell; EP colorants

figure 26, the 8 x 8 error diffusion case. Lines seem randomly distributed, with little radial tendency. Hue matching appears as a random process since a dominant color shift in the diagram is not indicated. The lack of consistent radial line placement in the hue circle diagrams for the error diffusion algorithm indicates poor colorimetric color reproduction. The hue angle curves for this same example are shown in figures 27 and 28.

Consider that approximately 1 radian separates consecutive system colors on the a*,b* diagram. The average radian deviation from the 1:1 hue angle line for an 8 x 8 error diffusion area was ± .95. This would be analogous to reproduction of a yellow color using green or red system colors. The poor hue matching ability of the error diffusion algorithm was not anticipated. As a result, an investigation into its cause was conducted.

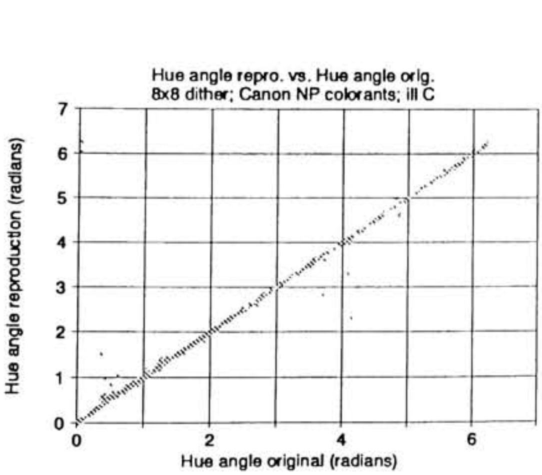


Figure 27

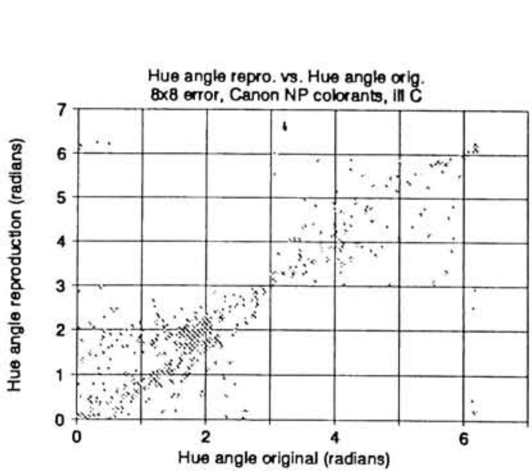


Figure 28

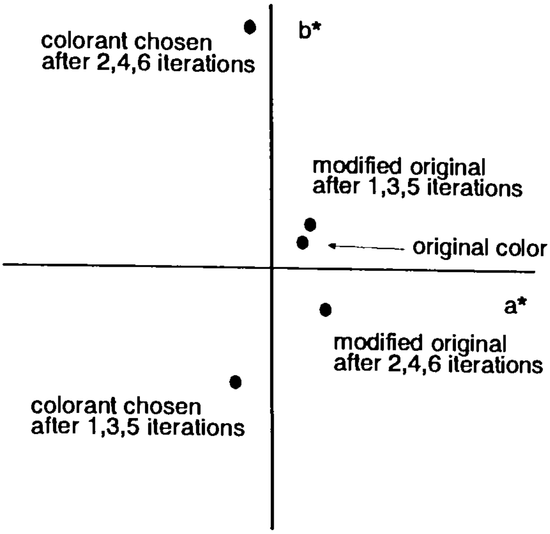


Figure 29

Figure 29 is an a*, b* diagram showing the actual progression of a color reproduction sequence from the error diffusion algorithm used in this study (since the error diffusion L* performance was reasonably well behaved, it was not investigated). An original color is shown in the diagram. It is a reddish-magenta color of low saturation. From its location in the a*, b* diagram, the likely choice of system colors for reproduction would be red and magenta. However, the first system color chosen by the error diffusion algorithm was cyan, shown in the lower left. The error experienced as a result of this choice shifted the next original color to be processed towards yellow (modified original). The shift appears slight, but it was enough to reduce the euclidean distance between the "modified" original colorant and the yellow colorant. As a result, the error diffusion algorithm selected yellow for the next reproduction pixel. As the process continued, it alternated between cyan and yellow, trying to reproduce magenta. Based on this information, the following was considered.

The EZ algorithm enjoys an advantage over the error diffusion algorithm in locating and isolating proper system colors. By using hue angle discrimination techniques, the correct system color selection is made prior to quantizing instead of being a part of the quantizing process. What this means is no considerations outweigh making the proper hue consideration. Lightness or saturation become secondary to hue rendition. This is not always true of the error diffusion process. System color selection is based on Euclidean distance between

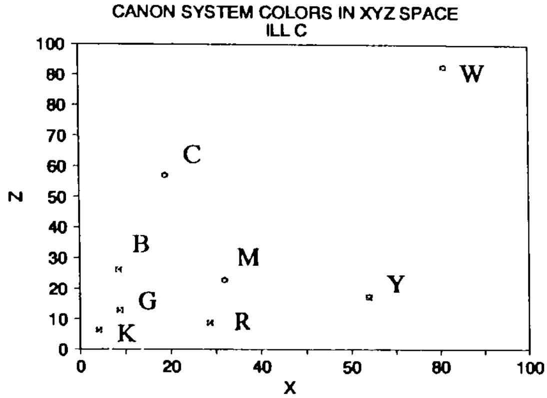


Figure 30

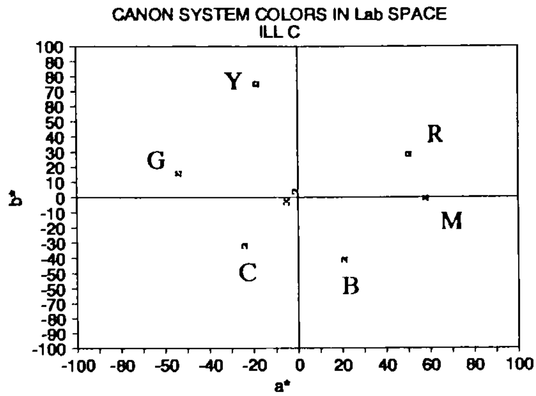


Figure 31

original and system colors. As error builds during the diffusion process, the euclidean distance may favor selection of another color from lightness or saturation considerations. Error from original / system color pairings may skew future pixels and impair correct hue matching. Consider figure 30, a plot of the XYZ tristimulus space where the error diffusion algorithm made its euclidean distance decisions. The distribution of colors in this space is non-uniform, with several system colors grouped so closely that small amounts of tristimulus error could shift a future original pixel closer to an inappropriate system color. If this system color were be chosen, hue matching errors would result. If a more uniformly distributed color space were used, such as CIELAB (figure 31), it is theorized that the color matching ability of the error diffusion algorithm would improve. To test this theory, the electrophotography system colors and cooresponding random colors were converted to L*, a*, b*. Reproduction images were generated and colorimetrically evaluated. Figures 32 through 34 show the L* reproduction vs. L* original curves, hue circle diagrams, and hue angle curves for the 8 x 8 pixel area. The use of a uniform color space is telling. As a result, the colorimetric

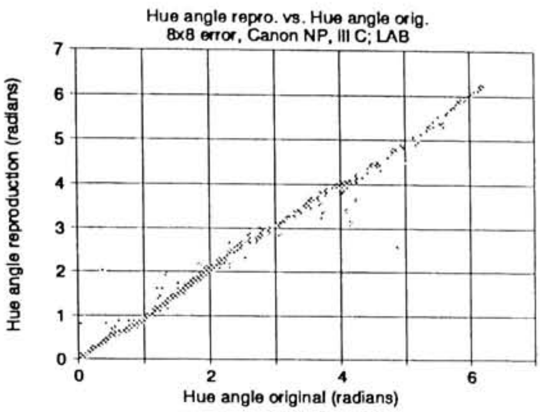


Figure 32

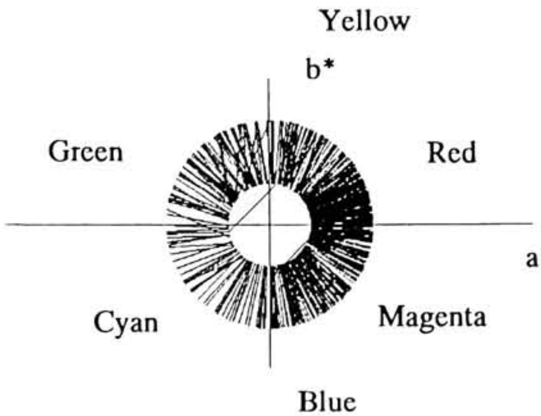


Figure 33
ED Method; Lab; 8 x 8 cell; EP colorants

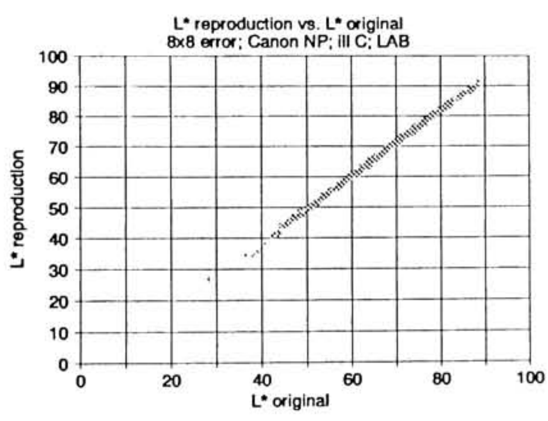


Figure 34

color reproduction potential of the error diffusion algorithm was markedly improved. Further improvement might be realized if the algorithm utilized a hue angle discrimination technique (such as the EZ method), to give priority to hue matching. Deviations from the 1:1 hue angle line for all conditions are shown in Table 3 and 4. An 8 x 8 error diffusion area is required to match the EZ method's 2 x 2 cell accuracy.

Standard Deviations about a 1:1 Hue angle line, EZ Algorithm

Table 3

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electro-photography	1.03	.52	.40	.29
Thermal	.60	.17	.15	.14
Ink Jet	1.15	.63	.04	.02
AVERAGE	**.93**	**.44**	**.20**	**.15**

Standard Deviations about a 1:1 Hue angle line, Error diffusion

Table 4

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electrophotog-raphy	1.66	1.18	.93	.67
Thermal	1.44	1.15	.90	.61
Ink Jet	1.84	1.39	1.01	.94
AVERAGE	**1.65**	**1.24**	**.95**	**.74**

L* reproduction vs. Metric chroma ratio

Two plots of L*reproduction vs. Metric chroma ratio for 2 x 2 cell sizes are given below, figures 35 and 36. They represent the EZ and the XYZ implementation of the error diffusion

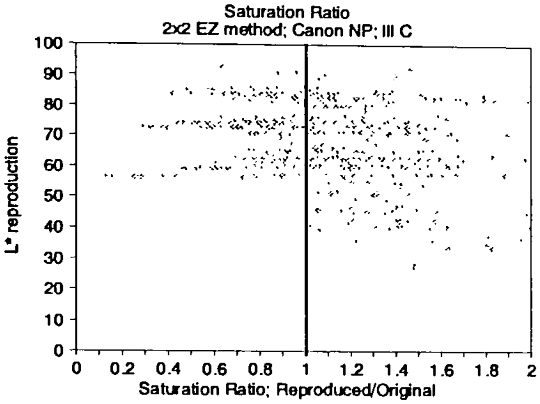


Figure 35

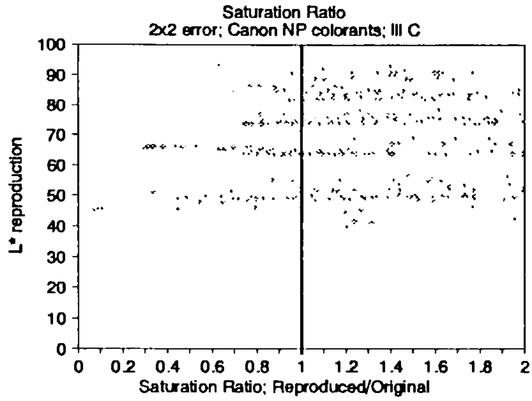


Figure 36

algorithm respectively, using the electrophotographic system color set. The "stepping" effect due to quantization error is apparent. Both plots also show significant excursions from a metric chroma ratio of one. Quantization error is considered the major cause for poor saturation reproduction in the 2 x 2 case since both algorithms show high variations in saturation ratio.

As the cell area increases, the EZ algorithm shows significant improvement in saturation ratio uniformity compared to the error diffusion algorithm, producing a smaller variance about a metric chroma ratio of one. The 8 x 8 cell examples are shown in figures 37 and 38.

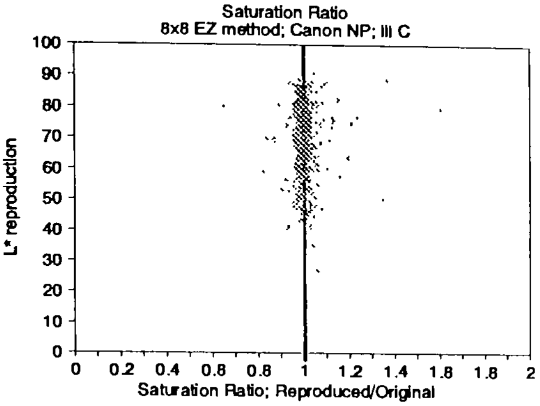


Figure 37

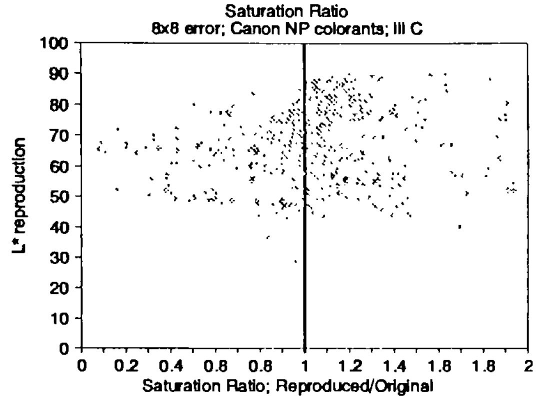


Figure 38

Poor hue matching ability may not necessarily indicate poor metric chroma reproduction. The magnitude of the two-dimensional a*, b* vector can be identical for many colors. However, for the error diffusion algorithm, consider the following.

In three-dimensional space, the euclidean distances between original and system colors are measured. The algorithm chooses the system color with the smallest euclidean distance to the original color. That is, the system color with the smallest distance differences over all X, Y, Z dimensions is chosen. If one or two tristimulus dimensions experienced large distance differences, the remaining dimension(s) would have to have small distance differences in order for that color to be chosen. With this in mind, review of the data presented so far indicates that the error diffusion algorithm experienced saturation difficulties due in part to hue matching errors. The L* curves from the XYZ implementation of this algorithm showed reasonably good perfomance. This means that one dimension of the three (Y tristimulus) produced reproduction values comparable to the original. However, the hue

matching ability of the algorithm was quite poor. So, at this point, the following was considered.

- The error diffusion equations attempt to find the minimal distance over all dimensions.
- One of the three dimensions is reasonably well minimized, resulting in good L* performance.
- At least one of the remaining two dimensions is not well minimized giving poor hue reproduction.

Since one dimension is reasonably well behaved, the other two can be considered in light of a two-dimensional space. As an example, this can be visualized in the a*, b* system (figure 39). Consider two points, p1, an original color, and p2, the selected system color each with different angles, θ_1 and θ_2 with respect to the a* axis. The angles represent the poor hue matching ability of the error diffusion process. The euclidean distance between these two points is the vector connecting them, v1. Since the intent of the error diffusion equations was to minimize distances, p2 would be brought (if possible) towards v2, the shortest vector to p1 along the constant angle, θ_2. This movement creates an error in saturation ratio, and that error tends to appear as a value less than one.

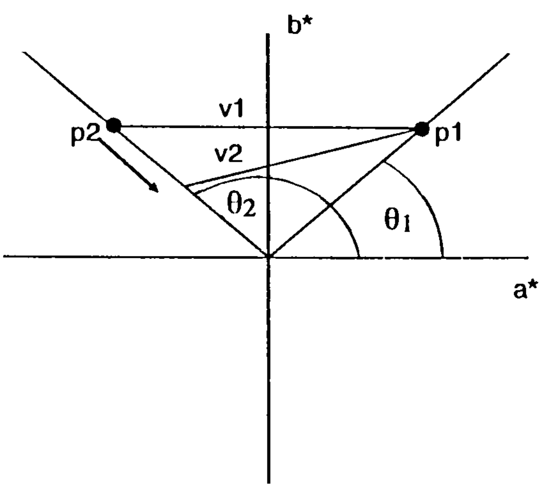


Figure 39

Figures 40 and 41 are the 2 x 2 and 8 x 8 Metric chroma ratio plots for the L* a* b* implementation of the error diffusion algorithm. The plots show this algorithm has better

saturation performance than the XYZ implementation. It also shows that most errors tend to be less than one.

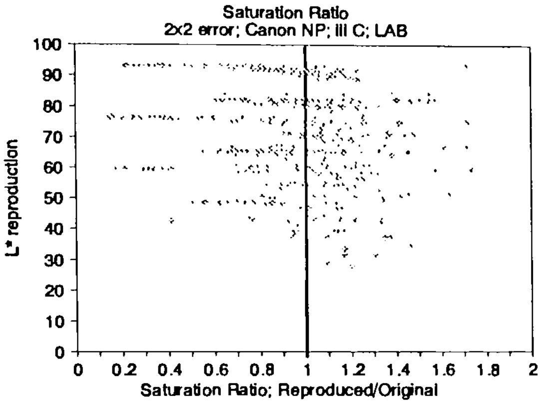


Figure 40

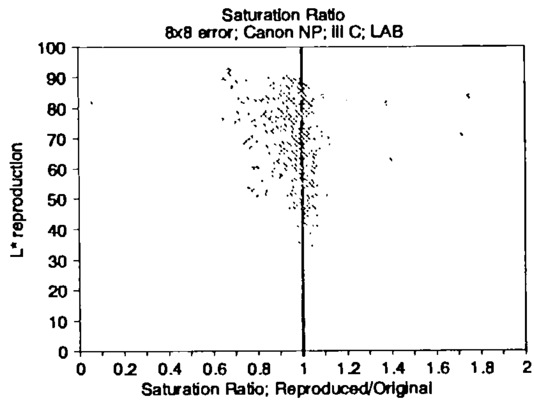


Figure 41

Average Metric chroma ratios with standard deviations for all conditions are shown in Table 5 and 6. Again, an 8 x 8 XYZ error diffusion area closely matches the EZ method's 2 x 2 halftone cell accuracy.

Average and Standard Deviation for Metric Chroma Ratio, EZ algorithm

Table 5

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electro-photography	1.16 ± .58	1.02 ± .19	1.01 ± .09	1.01 ± .08
Thermal	1.18 ± .57	1.01 ± .16	1.01 ± .11	1.00 ± .01
Ink Jet	1.14 ± .92	1.02 ± .13	1.00 ± .05	1.00 ± .03
AVERAGE	**1.16 ± .69**	**1.02 ± .16**	**1.01 ± .08**	**1.00 ± .04**

Average and Standard Deviation for Metric Chroma Ratio, Error diffusion

Table 6

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electrophotog-raphy	1.97 ± 2.44	1.53 ± 1.85	1.26 ± 1.19	1.12 ± .74
Thermal	1.88 ± 2.06	1.41 ± 1.31	1.19 ± .82	1.08 ± .53
Ink Jet	1.71 ± 3.29	1.25 ± 1.69	1.06 ± .68	1.02 ± .34
AVERAGE	**1.85 ± 2.60**	**1.40 ± 1.62**	**1.17 ± .90**	**1.07 ± .54**

CIELAB color difference (ΔE^*_{ab})

Figures 42 and 43 show the average color difference versus the number of pixels per cell for both algorithms. In all cases, average color difference decreases as the number of pixels increase. Because the error diffusion algorithm experienced saturation and hue matching

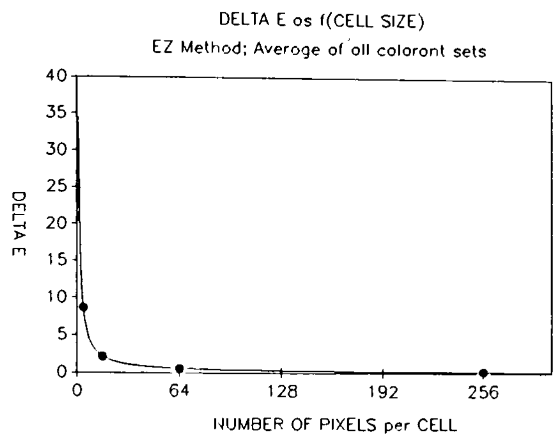


Figure 42

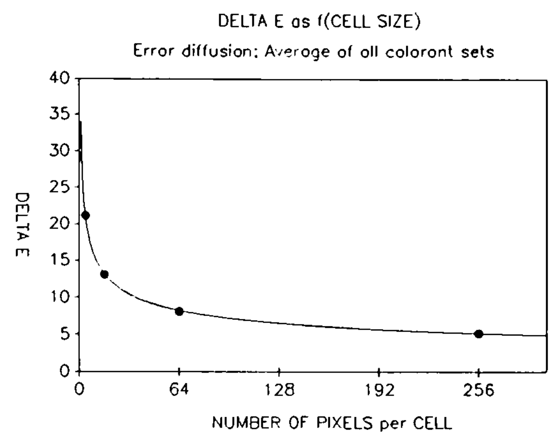


Figure 43

problems, the ΔE^*_{ab} performance was also expected to be poor when compared to the EZ algorithm. This is verified from the given curves, where an 8 x 8 area from error diffusion is required to match the 2 x 2 halftone cell of the EZ method. In no instance could error diffusion match the accuracy of the 4 x 4 EZ halftone cell.

Using the same data in curves 42 and 43, a linear relationship was produced by plotting $\log\Delta E^*_{ab}$ versus the log of the number of pixels per cell. For both algorithms, an equation describing this linear relationship was derived using the average of all system color sets. They are presented in figures 44 and 45, with the corresponding regression equation. Both methods show good fit to this functional form. Using both linear regression equations, an

estimate of the number of pixels required for the XYZ implementation of the error diffusion algorithm to match the EZ algorithm's ΔE^*_{ab} performance was made.

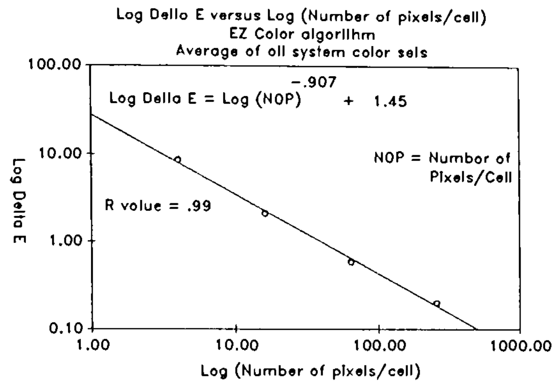


Figure 44

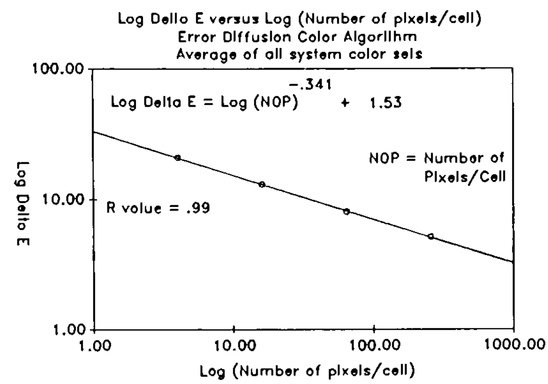


Figure 45

$\text{Log}\Delta E^*_{ab}$ as a function of log pixels/cell for the EZ and XYZ error diffusion methods is given in equations 29a and 29b, where a_0 , a_1 , b_0 , b_1 represent the intercept and slope terms of the EZ and error diffusion equations respectively. By equating 29a to 29b, a relationship between the number of error diffused pixels required to equal the color reproduction performance of a given number of EZ pixels is created.

$$\text{Log}\,\overline{\Delta E^*}_{abEZ} = a_1 \left(Log\,{}^{\text{pixels}}/_{\text{cell}_{EZ}}\right) + a_0 \qquad 29(\text{a-b})$$

$$\text{Log}\,\overline{\Delta E^*}_{abED} = b_1 \left(Log\,{}^{\text{pixels}}/_{\text{cell}_{ED}}\right) + b_0$$

$$\overline{\Delta E^*}_{abEZ} = 10^{\,a_1\,Log\,{}^{\text{pixels}}/_{\text{cell}_{EZ}} + a_0}$$

$$\overline{\Delta E^*}_{abED} = 10^{\,b_1\,Log\,{}^{\text{pixels}}/_{\text{cell}_{ED}} + b_0}$$

$$\overline{\Delta E^*}_{ab_{EZ}} = \left(\text{pixels}/_{\text{cell}}\right)^{a_1} 10^{a_0}$$

$$\overline{\Delta E^*}_{ab_{ED}} = \left(\text{pixels}/_{\text{cell}}\right)^{b_1} 10^{b_0}$$

Let $C_{EZ} = 10^{a_0}$ and let $C_{ED} = 10^{b_0}$. Equating error diffusion to the EZ algorithm,

$$C_{ED} \left(\text{pixels}/_{\text{cell}_{\text{ED}}}\right)^{b_1} = C_{EZ} \left(\text{pixels}/_{\text{cell}_{\text{EZ}}}\right)^{a_1}$$

$$\text{pixels}/_{\text{cell}_{\text{ED}}}{}^{b_1} = \frac{C_{EZ} \ \text{pixels}/_{\text{cell}_{\text{EZ}}}{}^{a_1}}{C_{ED}}$$

$$\text{pixels}/_{\text{cell}_{\text{ED}}} = \left(\frac{C_{EZ} \ \text{pixels}/_{\text{cell}_{\text{EZ}}}{}^{a_1}}{C_{ED}}\right)^{\frac{1}{b_1}} \qquad 30$$

Using the regression values given in figures 44 and 45, where a0=1.45, a1=-.907, b0=1.53, and b1=-.341, a plot of $\text{pixels}/_{\text{cell}_{\text{ED}}}$ versus $\text{pixels}/_{\text{cell}_{\text{EZ}}}$ is given in figure 46.

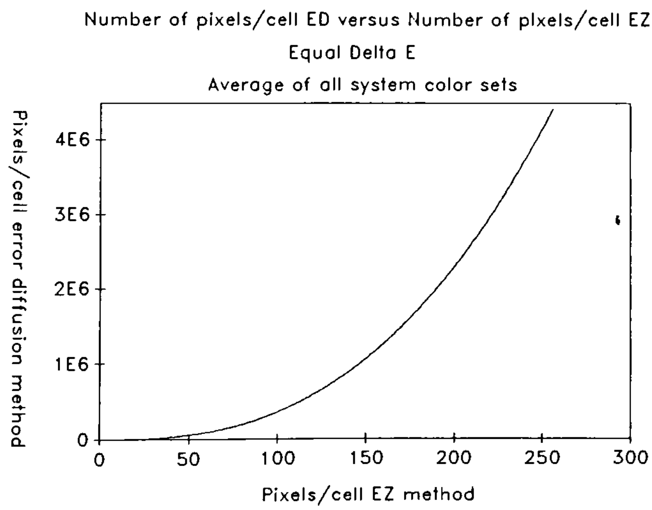


Figure 46

It is obvious from figure 46 that the EZ Color algorithm showed a marked improvement in color reproduction over the XYZ error diffusion. The EZ algorithm is able to provide ΔE^*_{ab} values less than 3 using a 4 x 4 cell. In contrast, a 16 x 16 error diffused area can just meet this criteria but only using the smaller gamut of the ink jet system color set (table 8). This poor performance happens in spite of the fact that a 16 x 16 cell is capable of providing more than one hundred million colors. Since the number of colors capable does not change with algorithm, the difference clearly lies in the manner the two methods work. Results for both algorithms are given in tables 7-8.

The L* a* b* error diffusion algorithm created lower ΔE^*_{ab} values than its XYZ counterpart. A curve of ΔE^*_{ab} versus the number of pixels per cell is given in figure 47, and overall results are given in table 9, with L* and Hue angle standard deviations, and Metric chroma ratio averages and deviations.

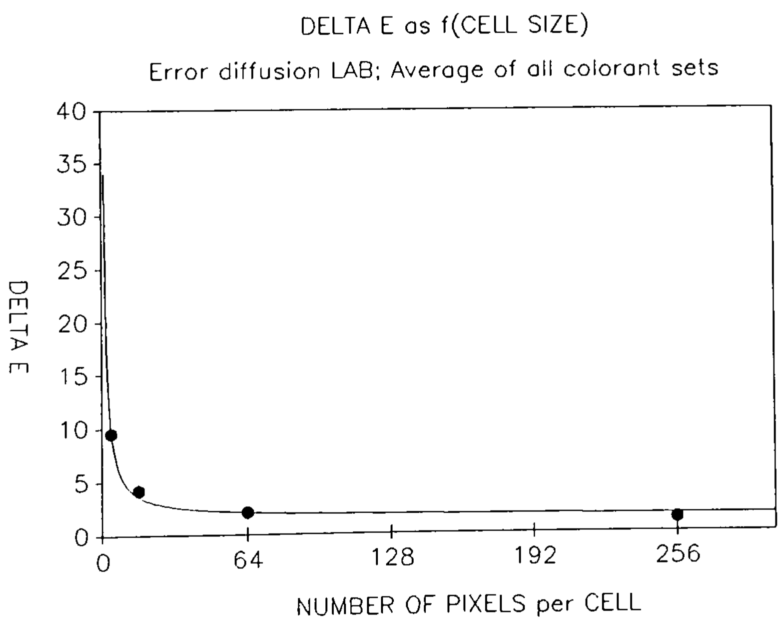


Figure 47

Average Color Difference with Standard Deviation, EZ algorithm

Table 7

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electro-photography	8.89 ± 5.64	2.15 ± 1.23	.62 ± .42	.22 ± .34
Thermal	10.10 ± 6.60	2.52 ± 1.63	.70 ± .45	.24 ± .34
Ink Jet	6.96 ±3 .52	1.68 ± .83	.44 ± .23	.13 ± .15
AVERAGE	**8.65 ± 5.25**	**2.12± 1.23**	**.59 ± .37**	**.20 ± .28**

Average Color Difference with Standard Deviation, Error diffusion

Table 8

Cell size / Colorants	2 x 2	4 x 4	8 x 8	16 x 16
Electro-photography	22.24 ± 14.30	15.09 ± 9.45	9.66 ± 5.35	6.08 ± 2.98
Thermal	24.17 ± 16.45	15.32 ± 11.64	10.09 ± 6.91	6.70 ± 4.06
Ink Jet	17.08 ± 10.62	8.84 ± 5.29	4.47 ± 2.54	2.66 ± 1.57
AVERAGE	**21.16 ± 13.79**	**13.08 ± 8.79**	**8.07 ± 4.93**	**5.15 ± 2.87**

Color Performance of Error diffusion LAB

Table 9

Colorants \ Cell size	2 x 2	4 x 4	8 x 8	16 x 16
Standard Deviation from 1:1 L* line	6.90	2.94	1.44	.87
Standard Deviation from 1:1 Hue angle line	1.11	.67	.17	.13
Average Metric chroma ratio and deviation	.99 ± .51	.97 ± .44	.97 ± .24	.98 ± .10
ΔE^*_{ab} average and deviation	9.51 ± 3.76	4.16 ± 1.68	2.05 ± .78	1.23 ± .42

CONCLUSION

The purpose of this thesis was to investigate the Colorimetric Color Reproduction accuracy of two halftoning algorithms, the EZ Color algorithm and Floyd and Steinberg's error diffusion. Computer models of each algorithm, original image, and system color set were created. Varying halftone cell sizes (2x2, 4x4, 8x8, 16x16) were used. The reproductions were evaluated for colorimetric color reproduction accuracy using CIELAB color metrics.

Initial assumptions about each algorithm tended to prefer error diffusion since it is an adaptive algorithm, meaning the quantization process is image dependent. Although Structured Dot Theory allows the selection of proper system colors for quantization, the array of threshold values in the EZ matrix is insensitive to changes in image structure, color, etc. With error diffusion, tristimulus errors are not discarded but added to neighboring pixels. This suggested that the method would provide inherently better color reproduction capability. However, the propagation of error in tristimulus space was seen to create poor system color selections. Large hue, saturation, and ΔE errors make this algorithm unsuitable for colorimetric color reproduction if XYZ tristimulus space is used. A more uniform color space, such as CIELAB, is considered a minimum requirement in order for this algorithm to perform well, especially when several system colors have similiar tristimulus values. Hue, saturation, and ΔE^*_{ab} errors were minimized when this color space was used. However, an additional improvement could be made to this algorithm if the proper system colors were selected before the error diffusion process began.

The EZ Color Algorithm provides several important features including the incorporation of the black colorant explicity in the color formulation, selection of the proper system colors before quantizing, and quantization of system color areas instead of reflectance values. The

algorithm produced the highest color accuracy with all cell sizes considered. Using the 4 x 4 halftone cell, overall color difference did not exceed 3 ΔE^*_{ab} units. Average color difference decreased to less than .6 for the 8 x 8 and higher halftone cells. It is considered the better choice for colorimcric color reproduction accuracy over the XYZ or LAB error diffusion implementations.

APPENDICES

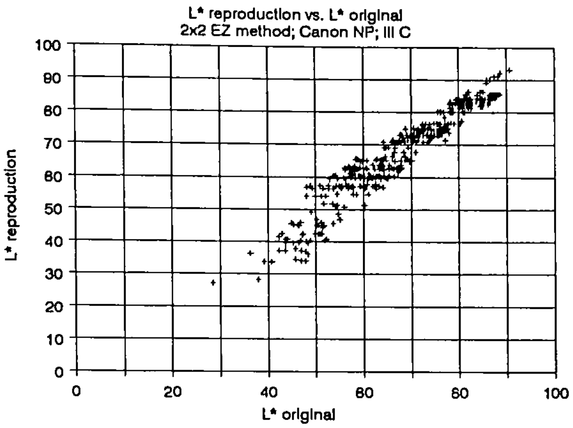


Figure 48

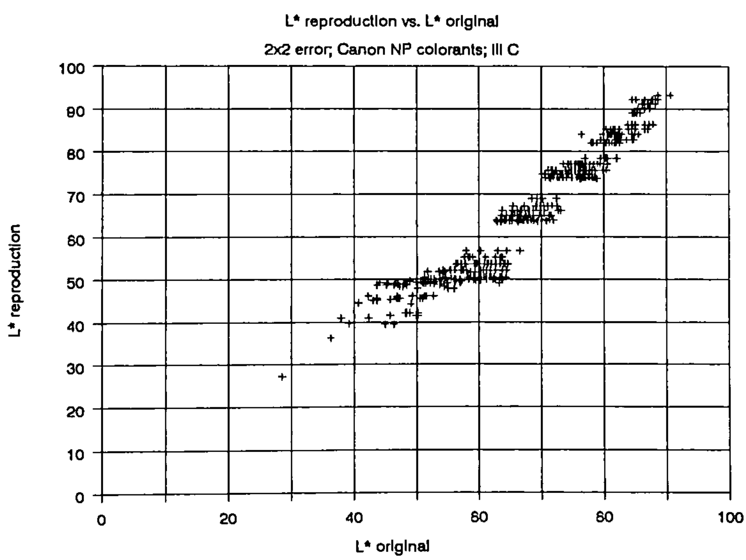


Figure 49

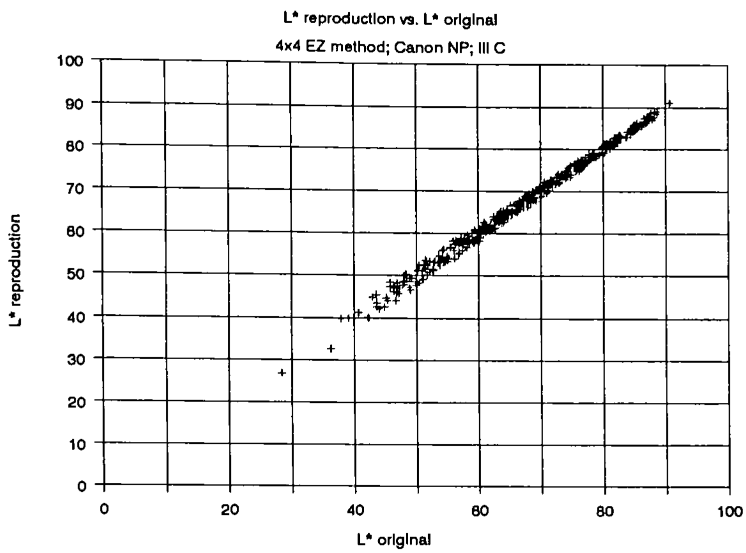


Figure 50

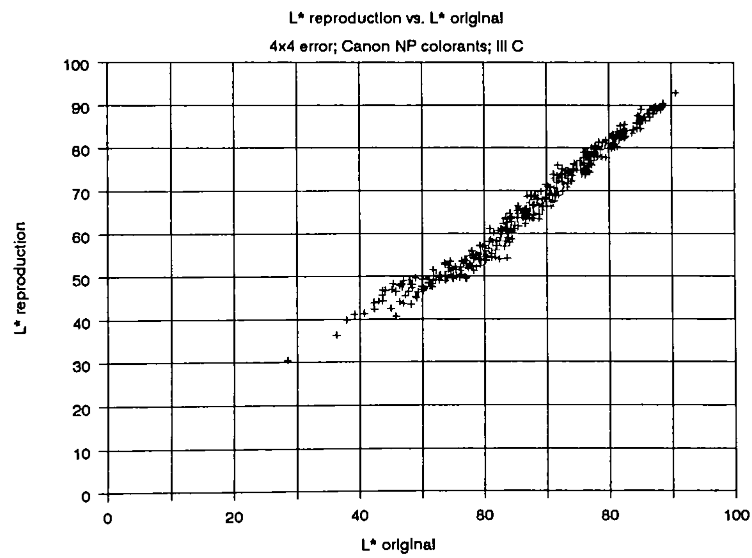


Figure 51

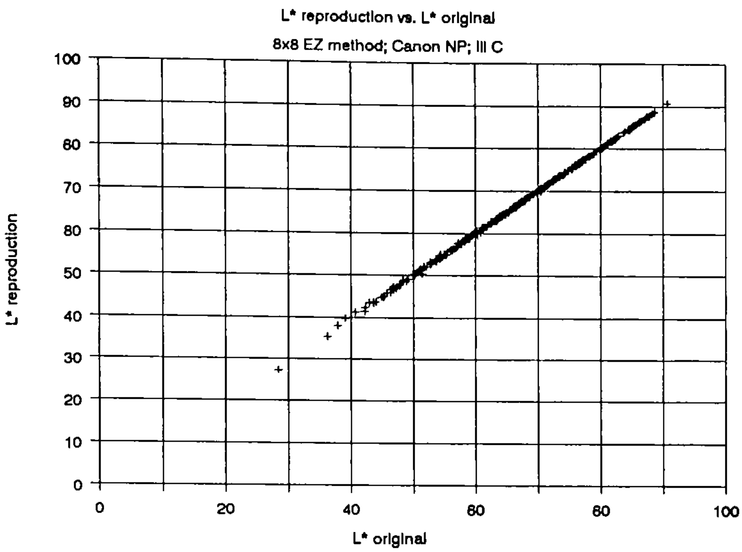


Figure 52

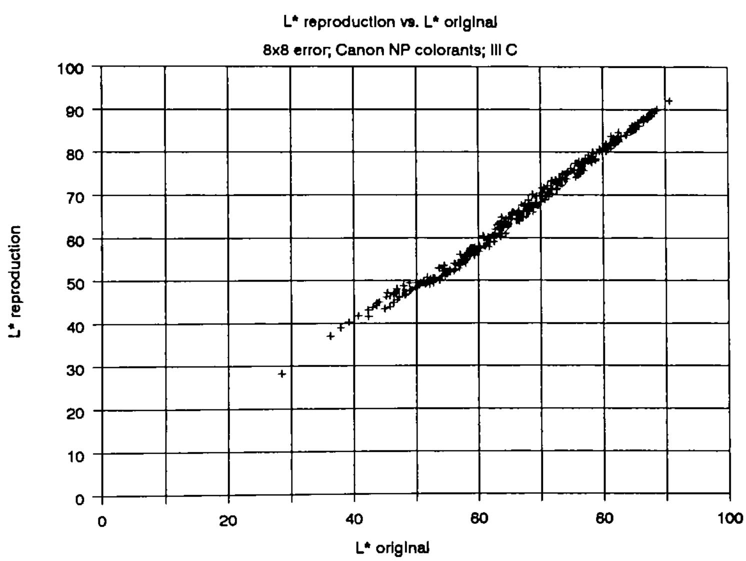


Figure 53

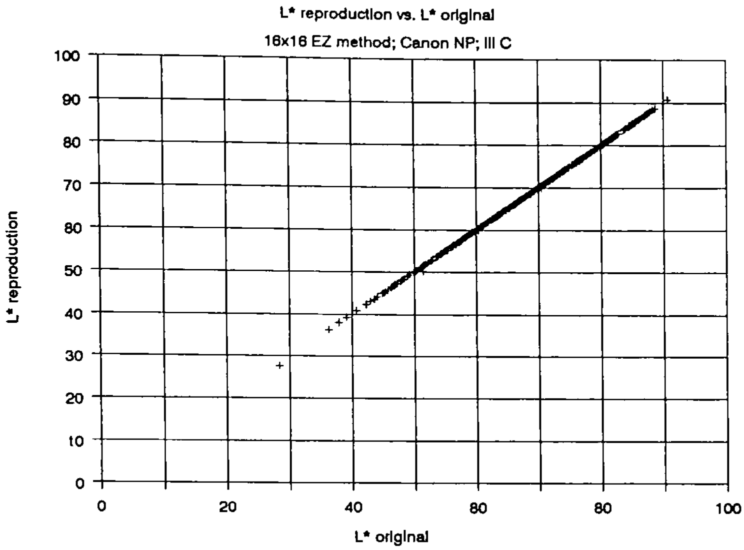


Figure 54

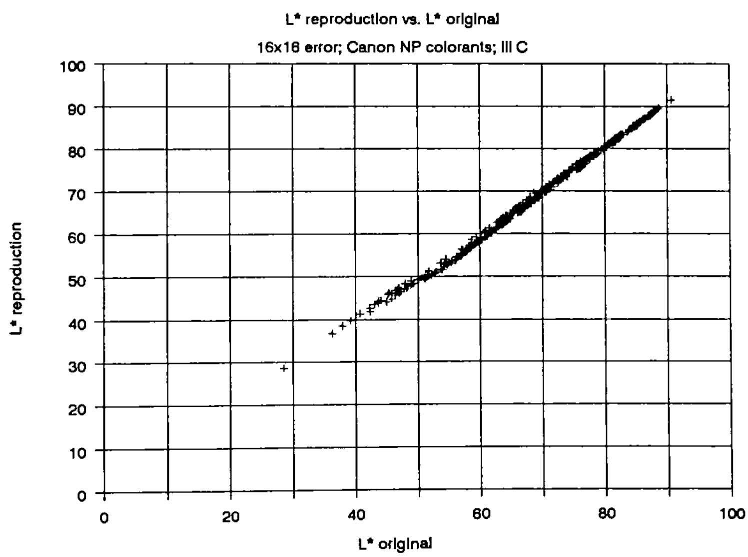


Figure 55

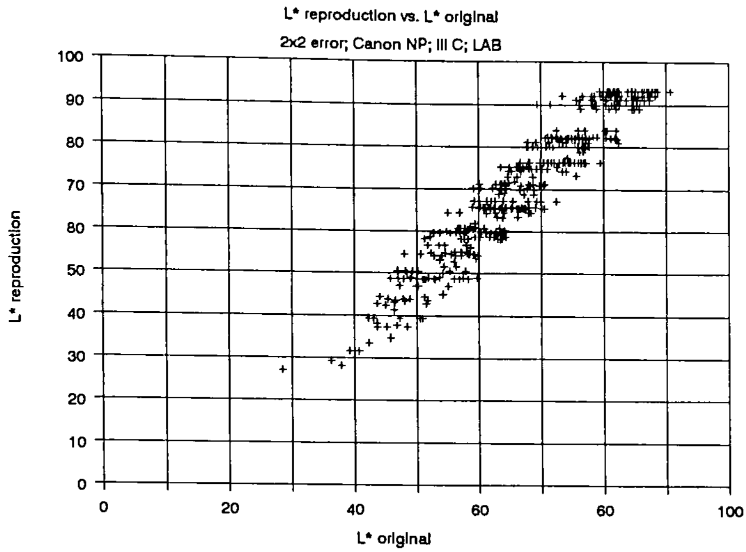


Figure 56

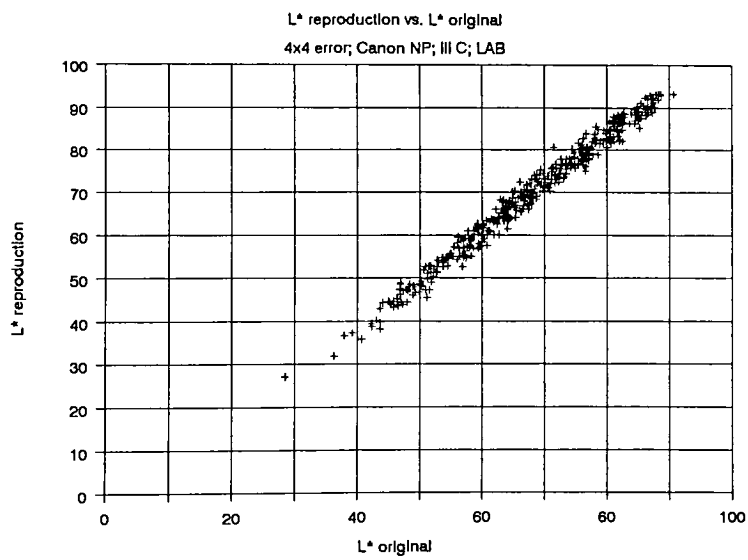


Figure 57

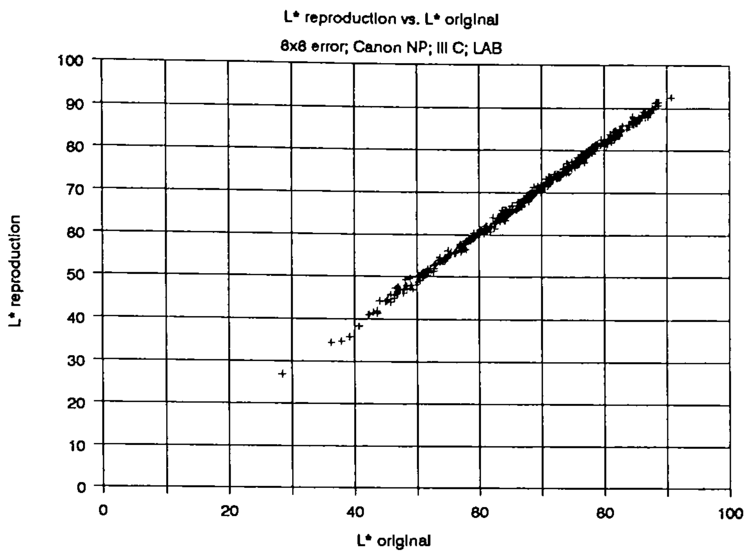


Figure 58

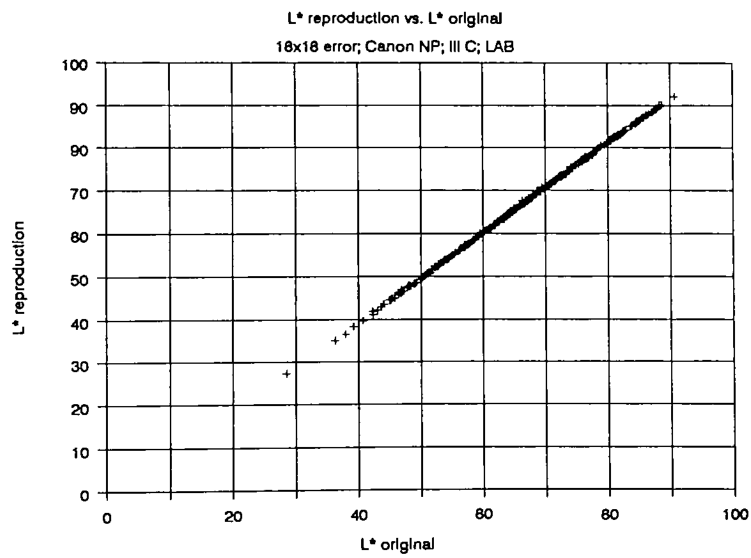


Figure 59

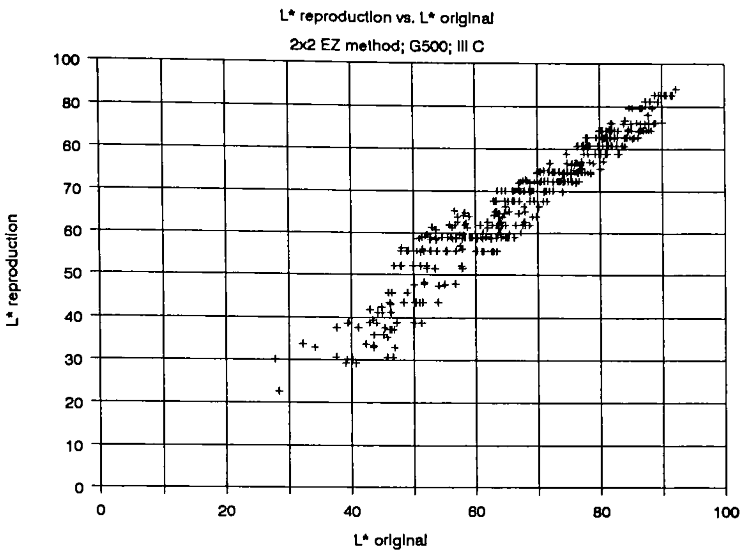


Figure 60

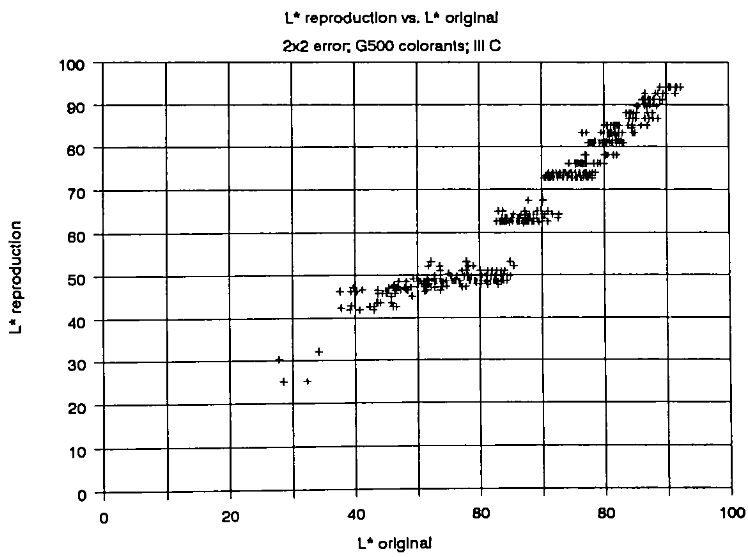


Figure 61

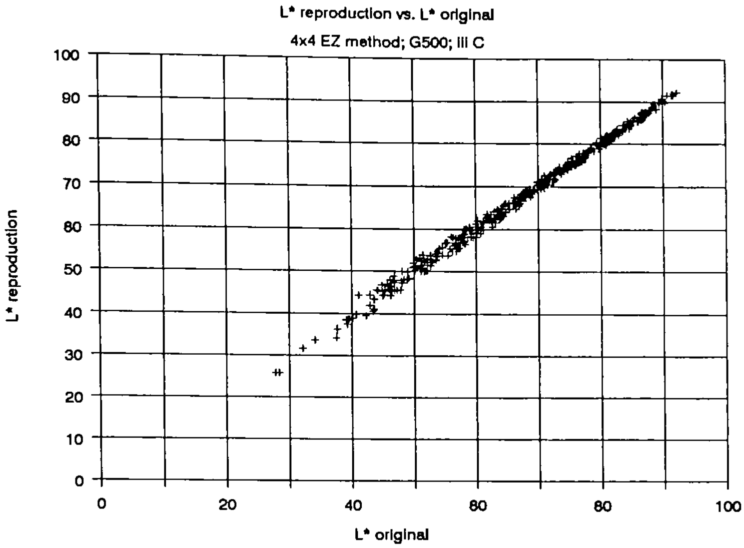


Figure 62

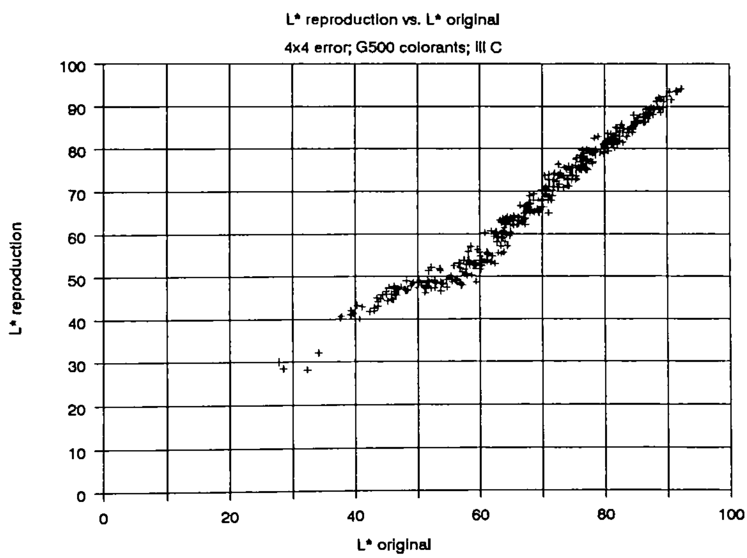


Figure 63

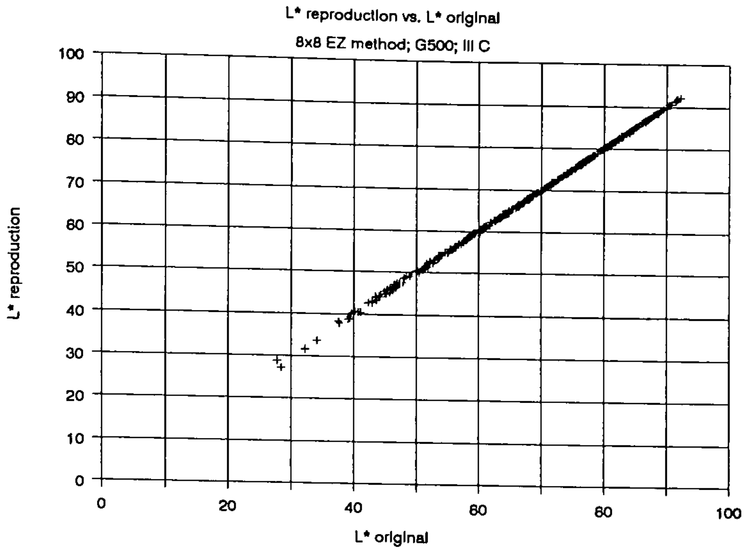


Figure 64

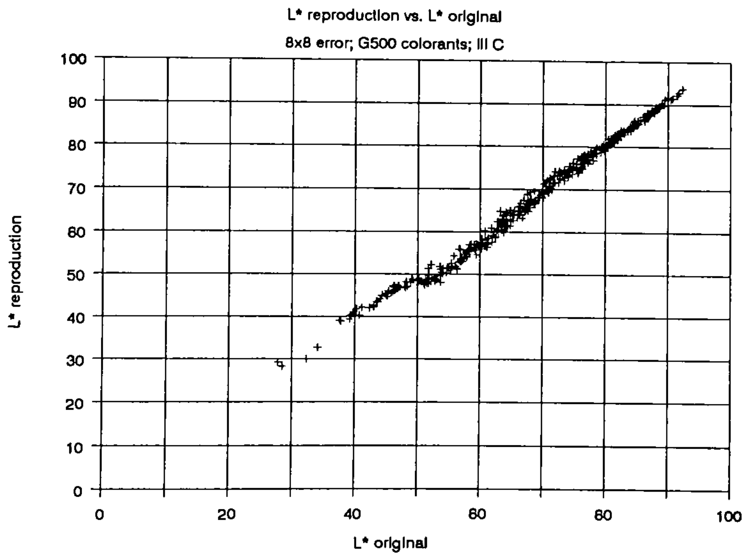


Figure 65

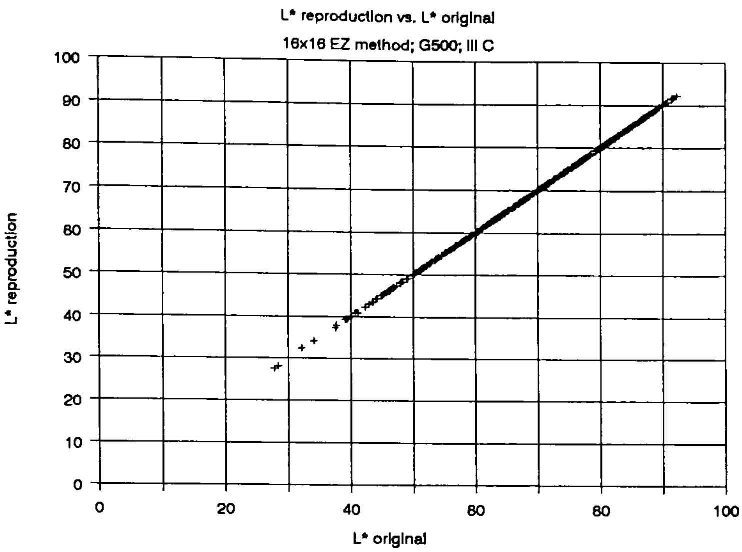


Figure 66

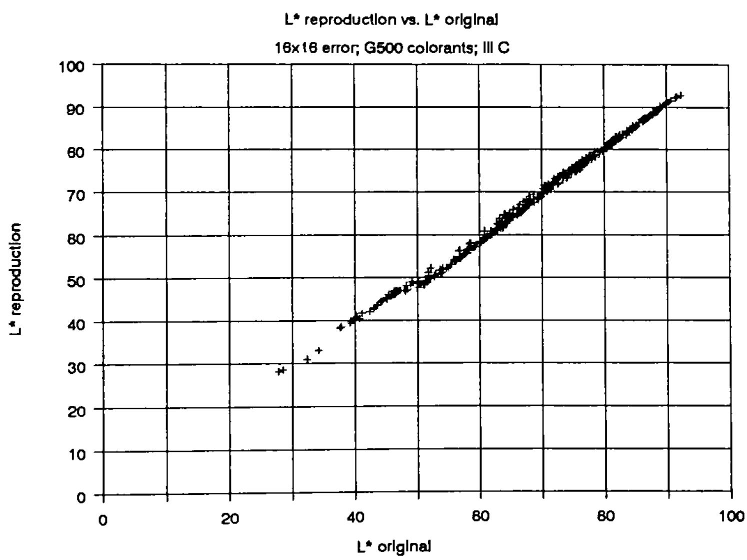


Figure 67

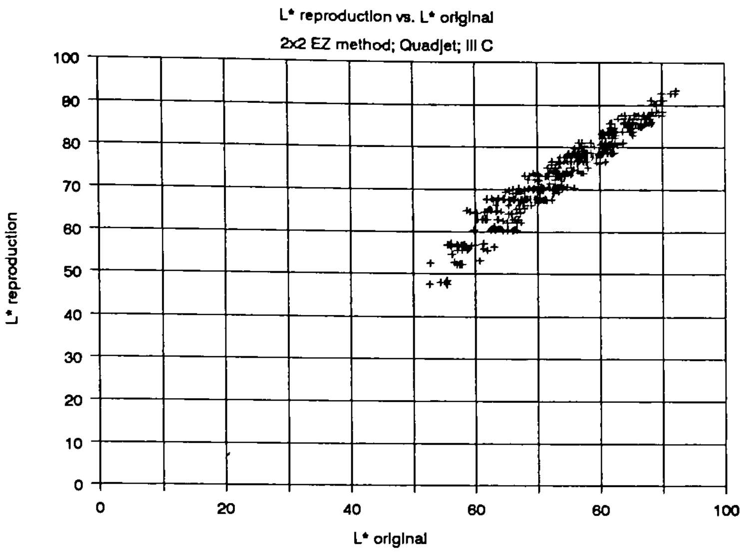


Figure 68

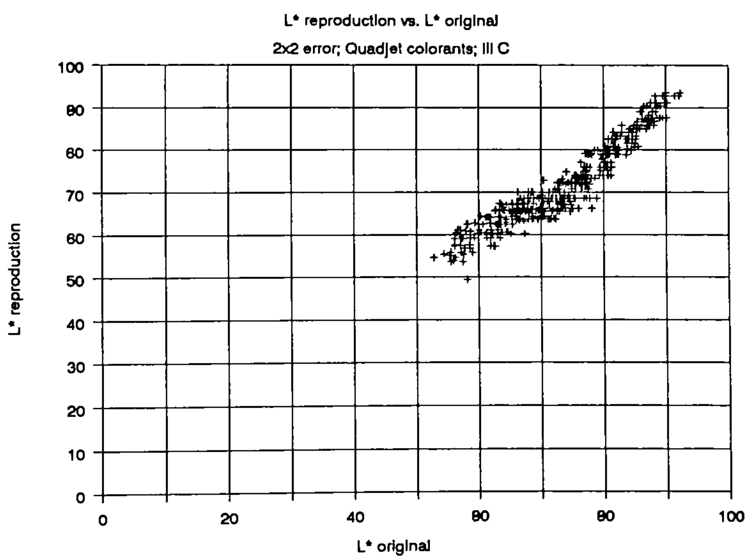


Figure 69

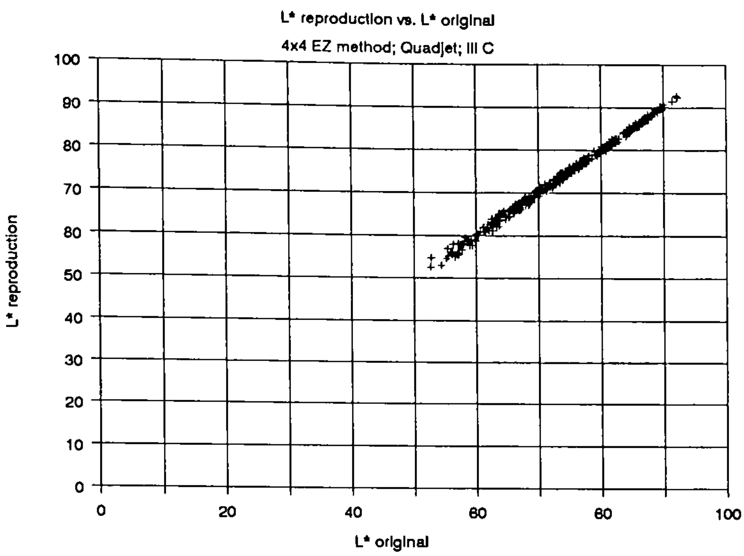


Figure 70

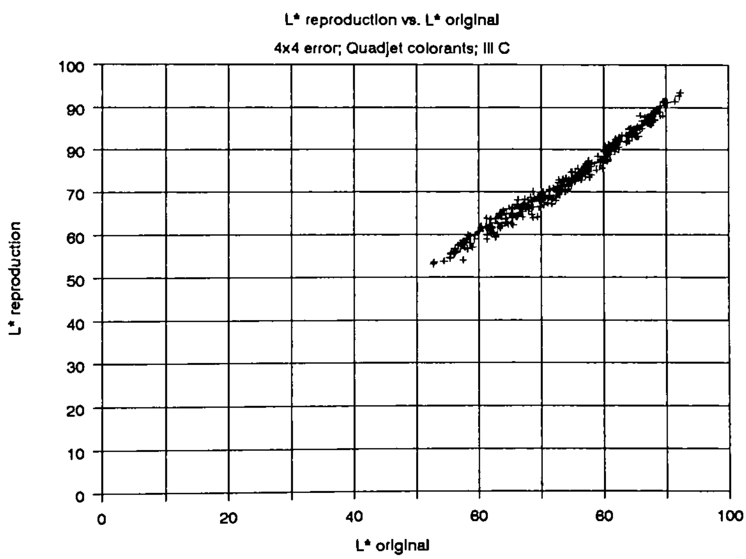


Figure 71

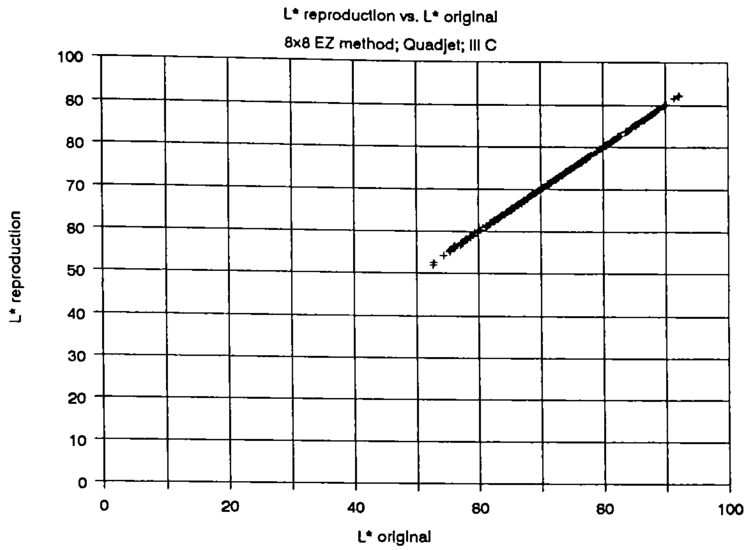


Figure 72

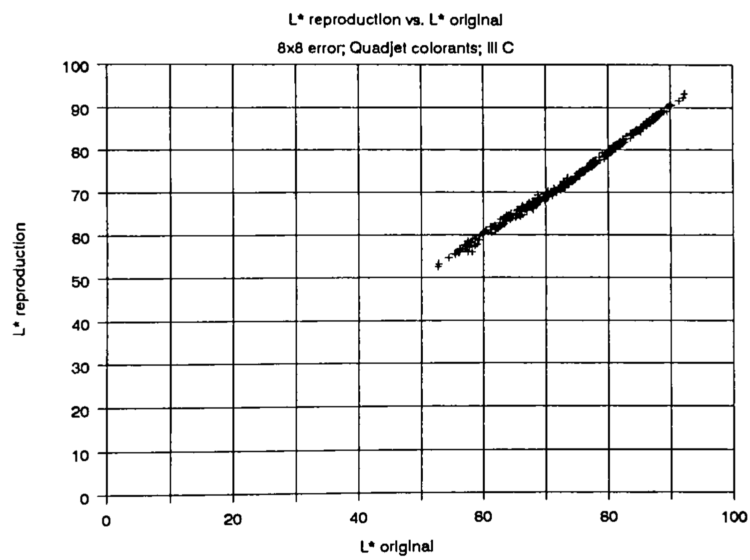


Figure 73

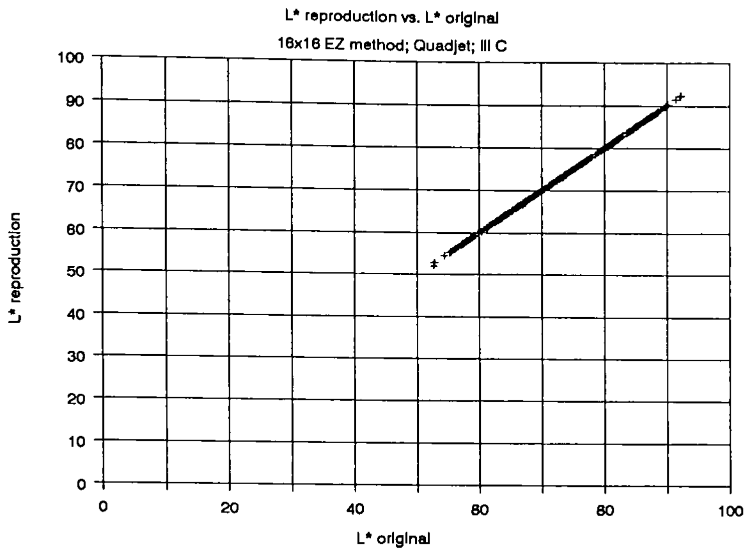


Figure 74

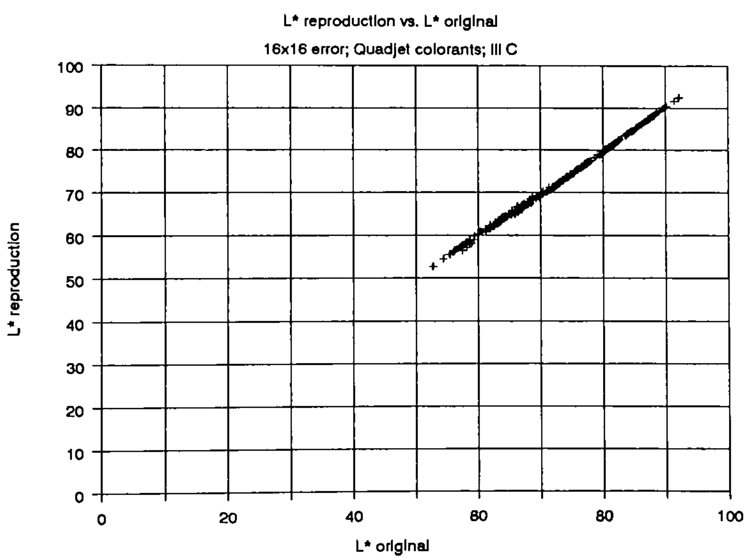


Figure 75

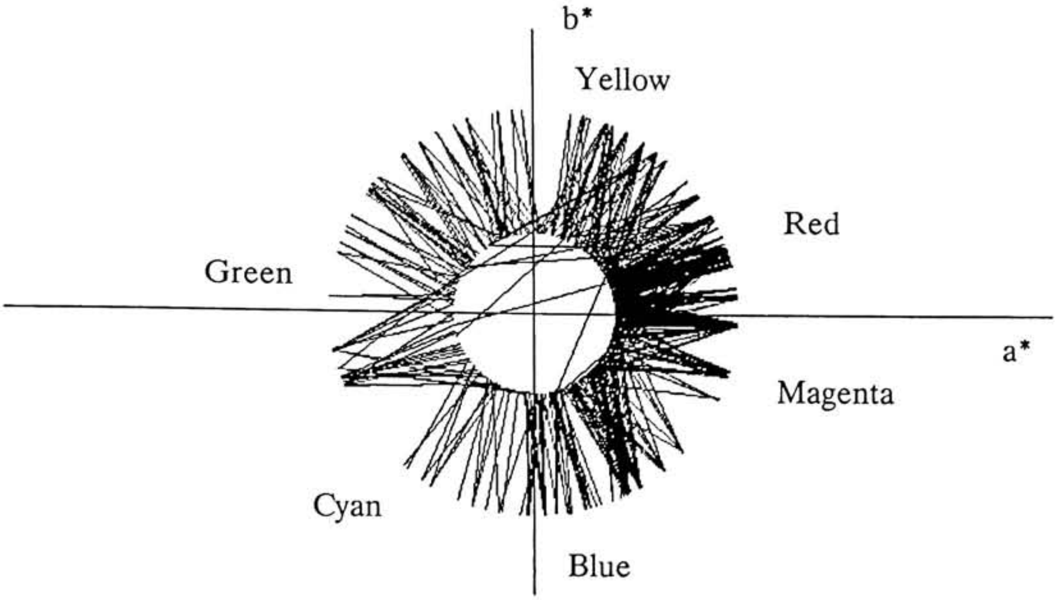


Figure 76

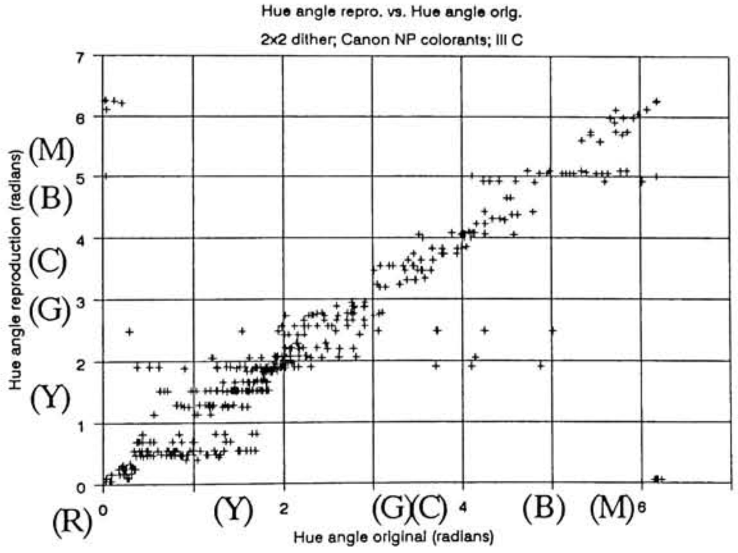


Figure 77

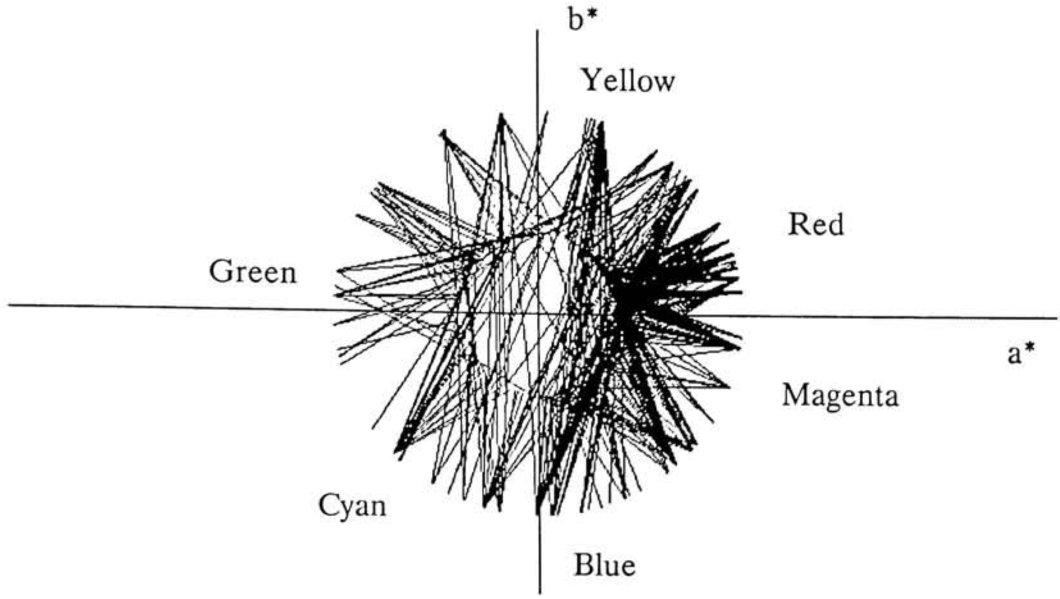


Figure 78

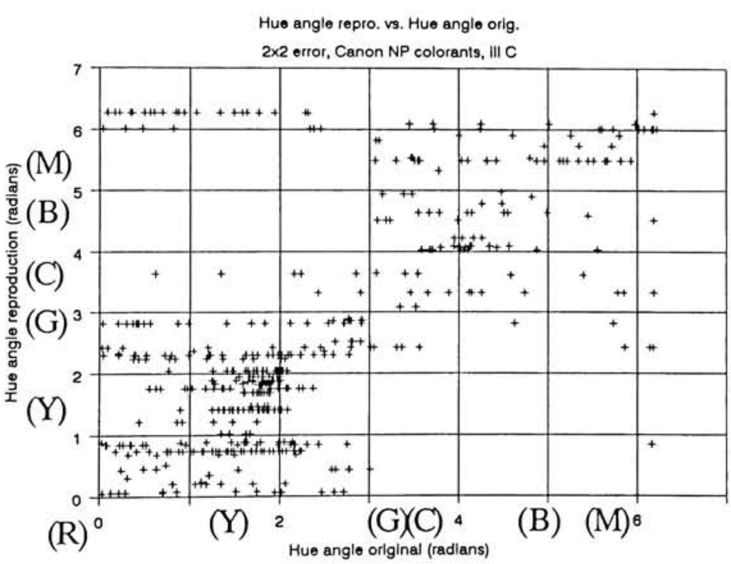


Figure 79

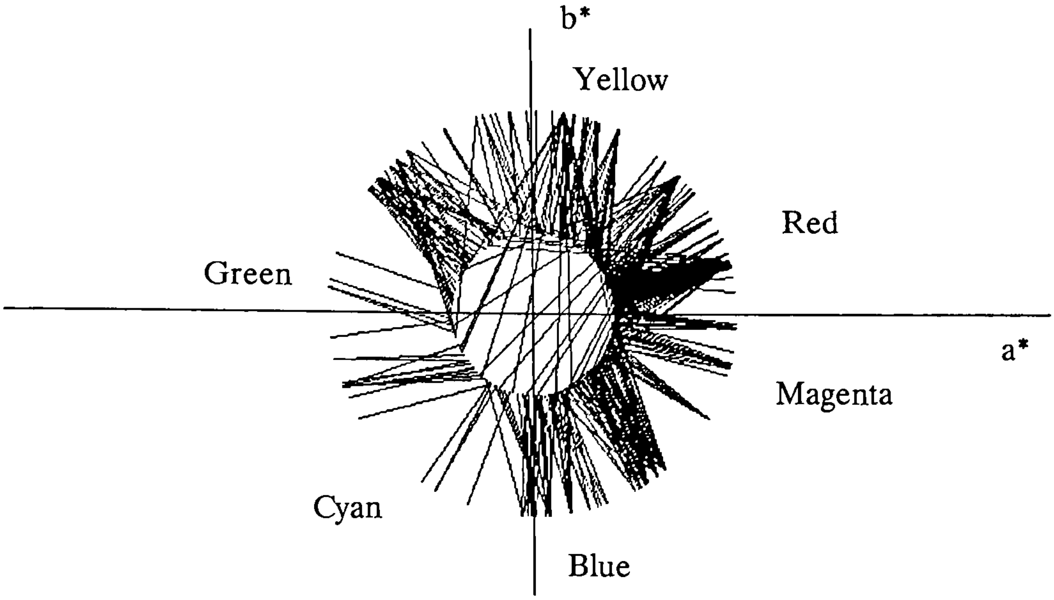


Figure 80

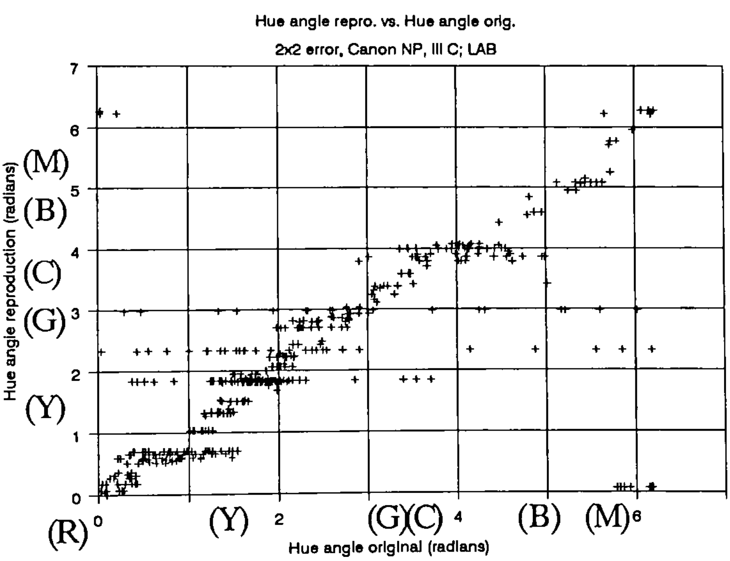


Figure 81

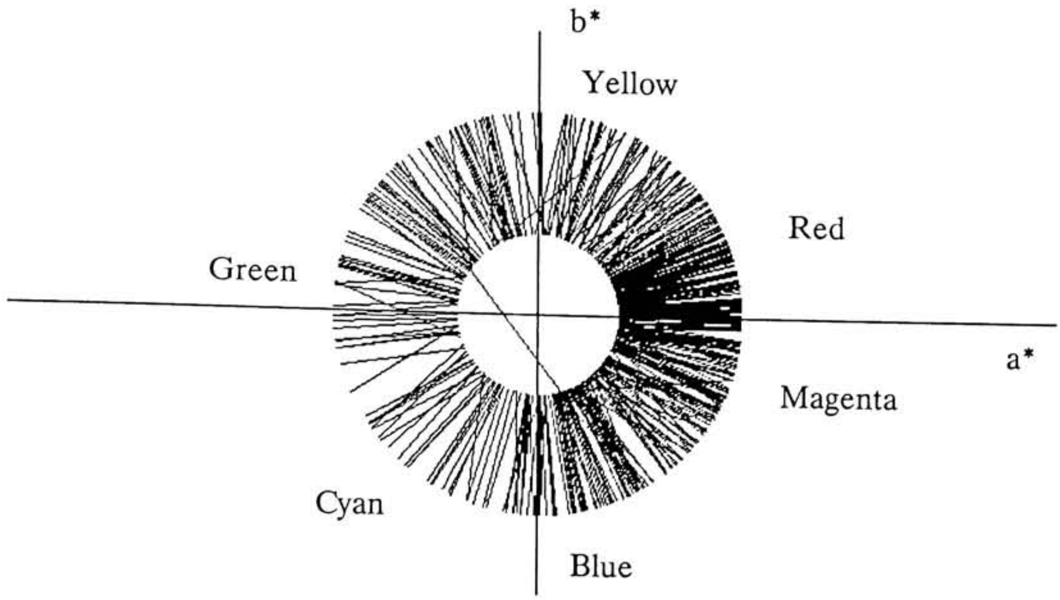


Figure 82

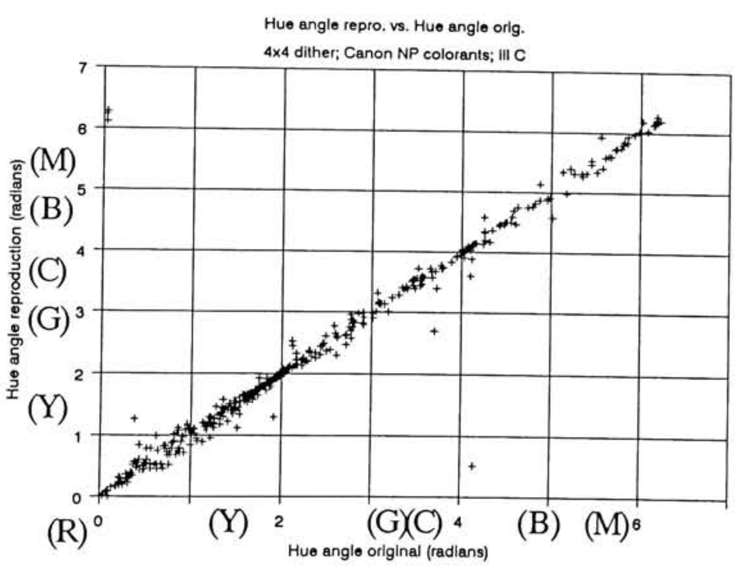


Figure 83

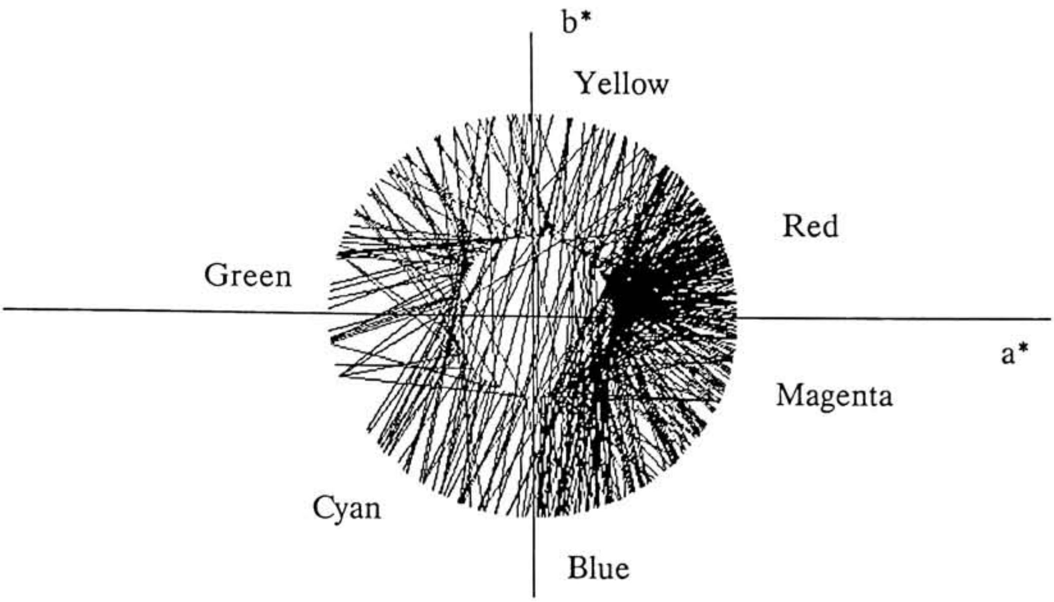


Figure 84

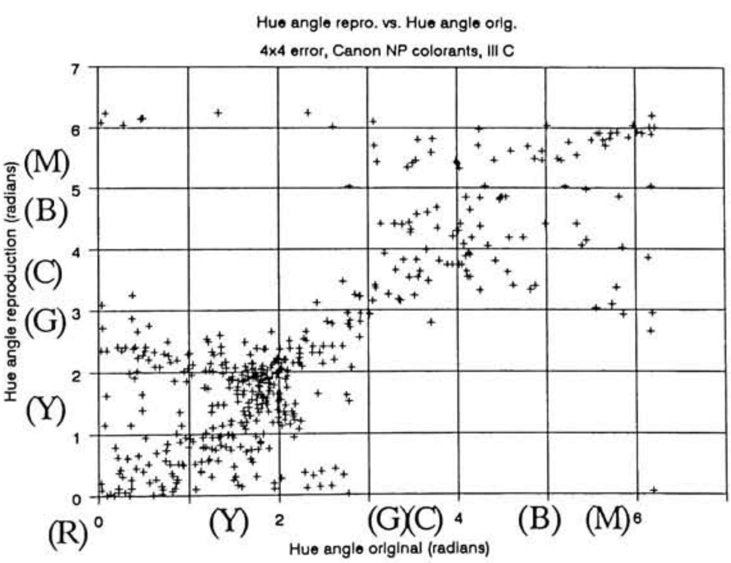


Figure 85

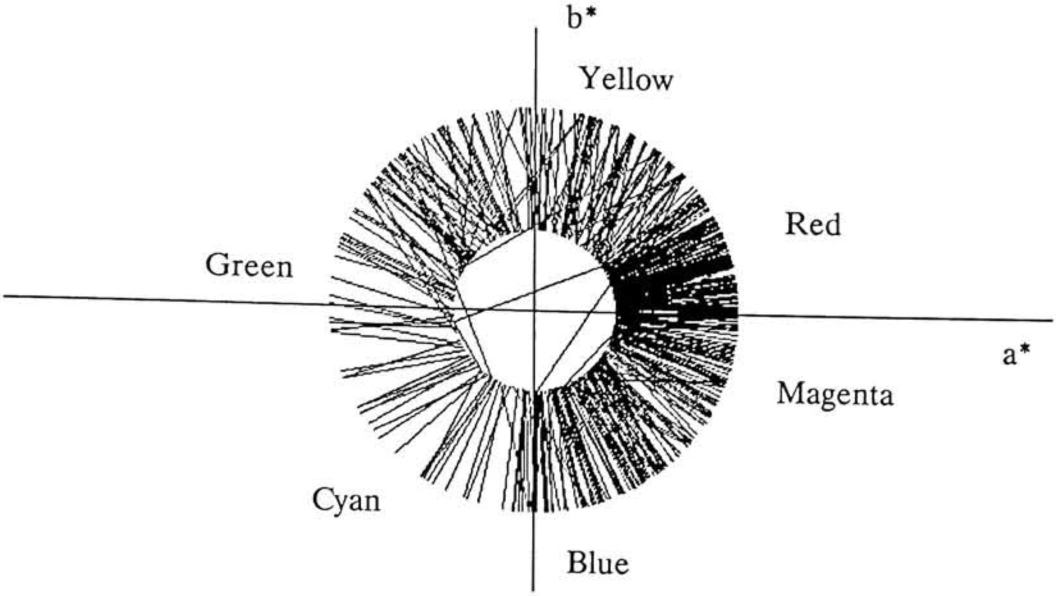


Figure 86

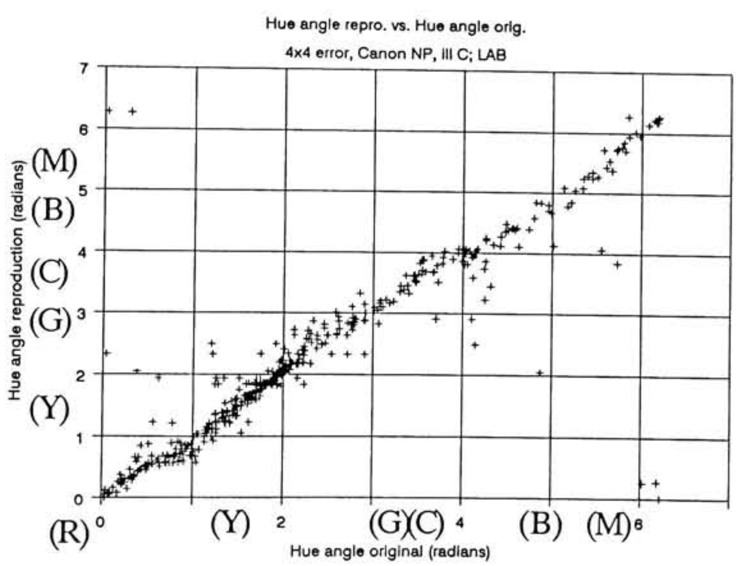


Figure 87

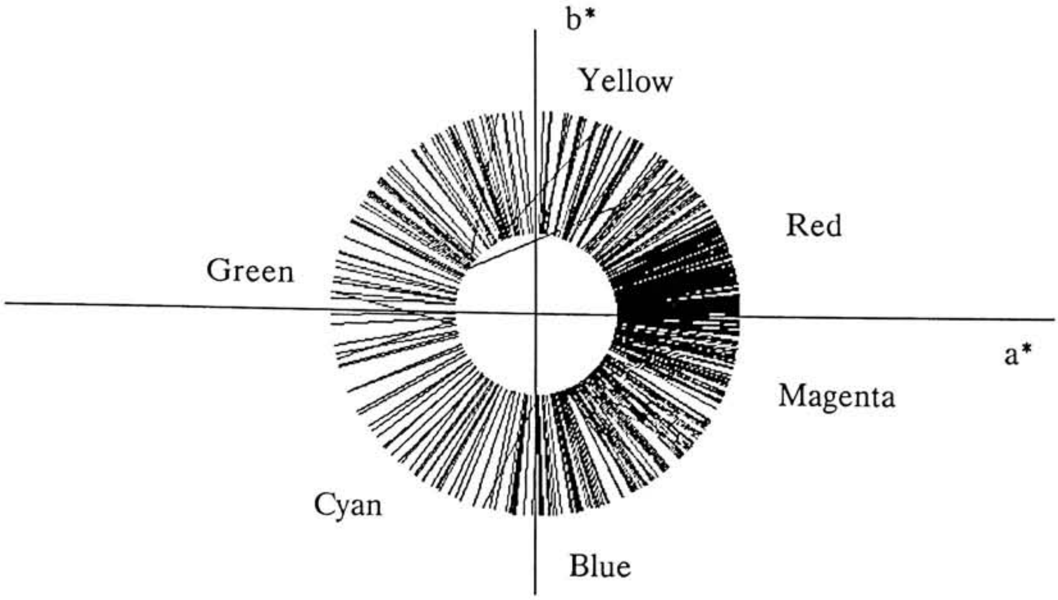


Figure 88

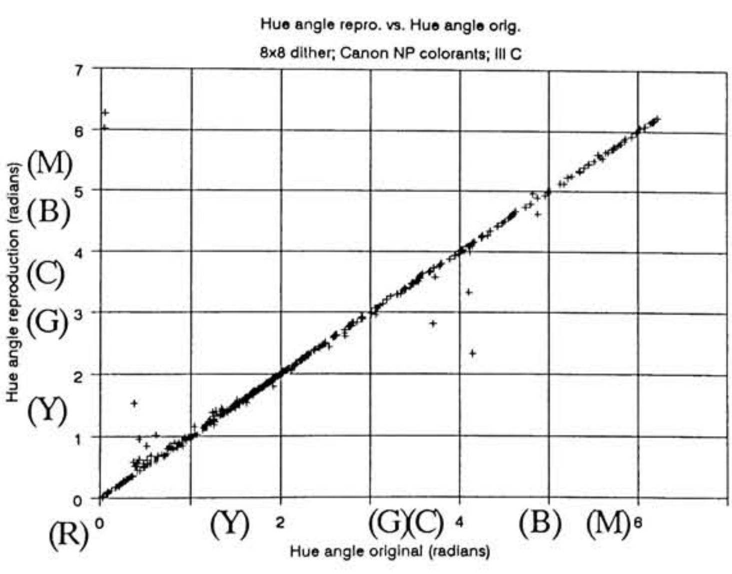


Figure 89

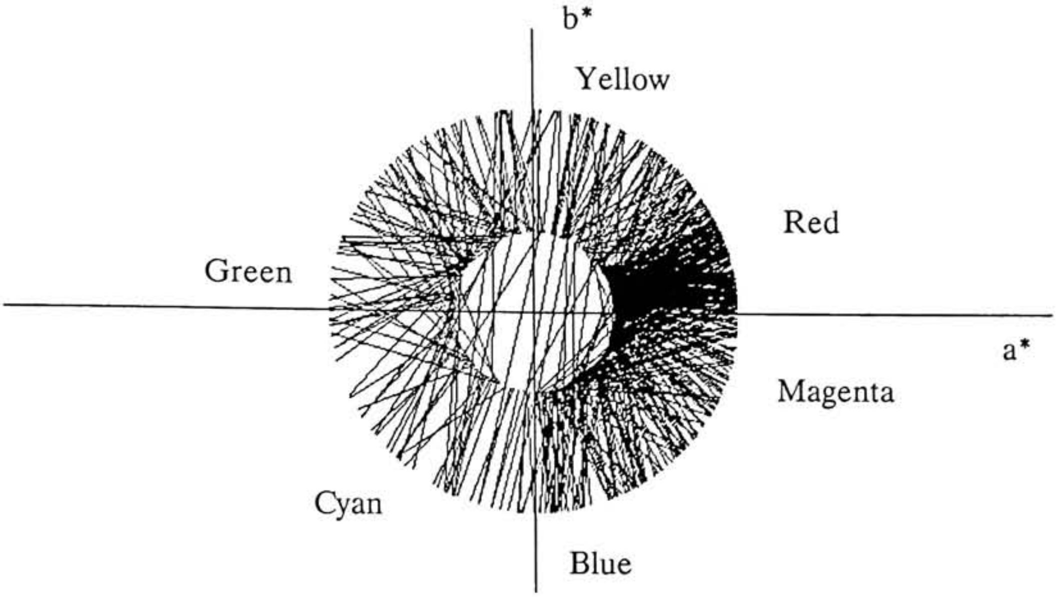


Figure 90

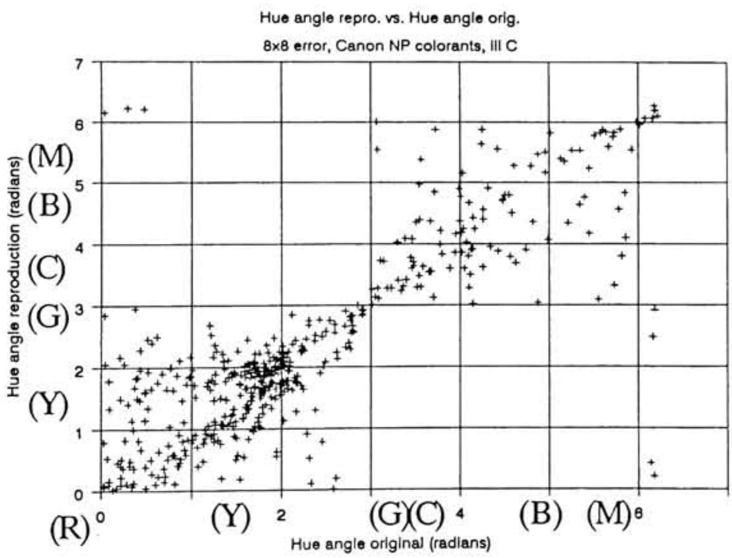


Figure 91

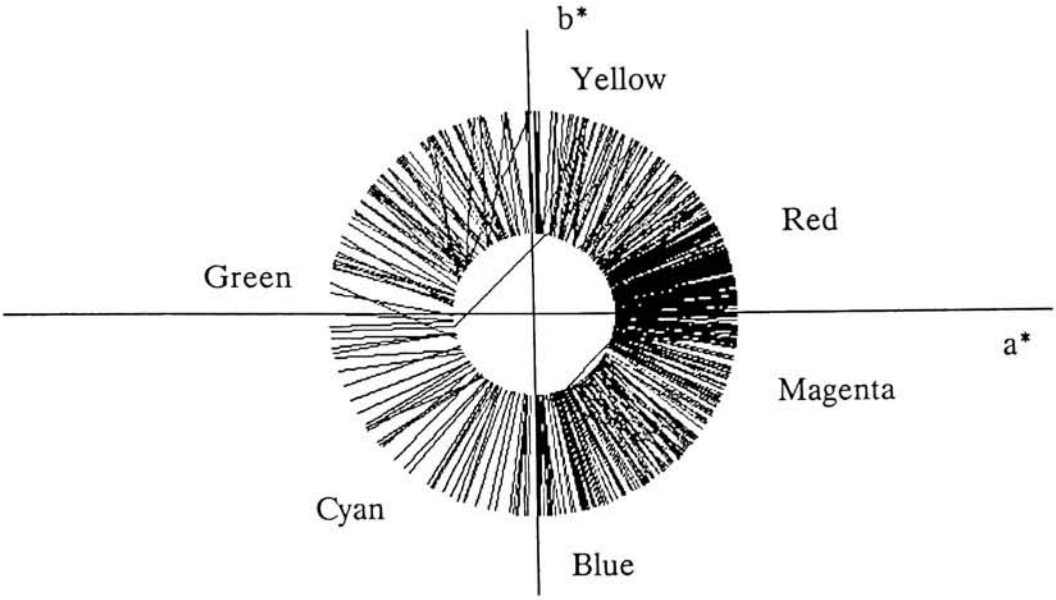


Figure 92

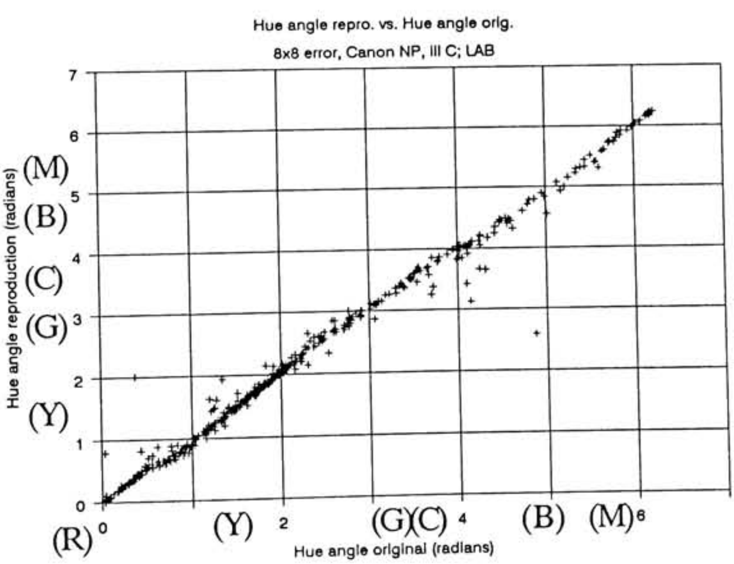


Figure 93

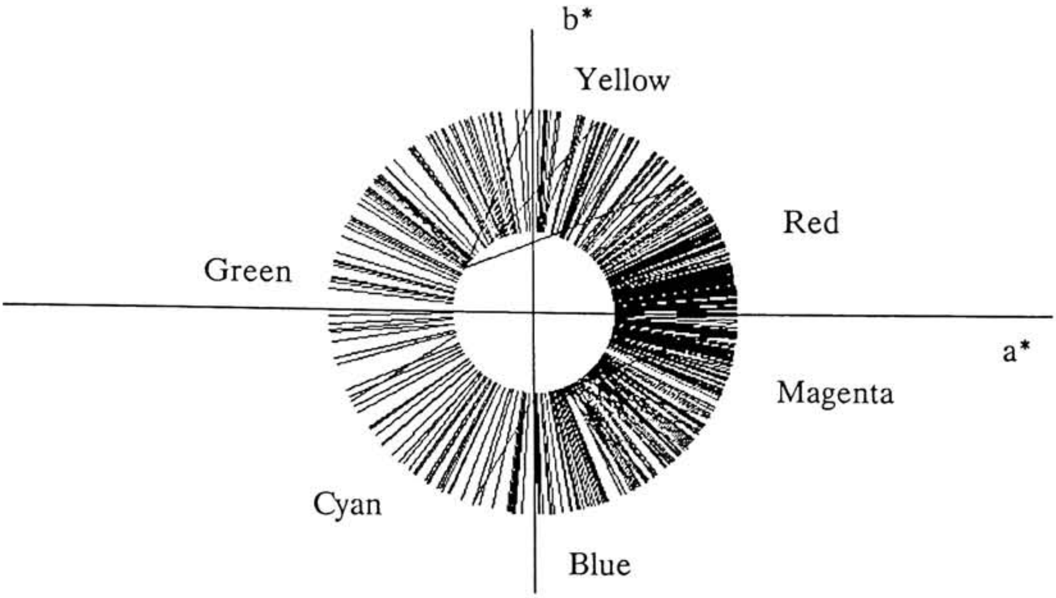


Figure 94

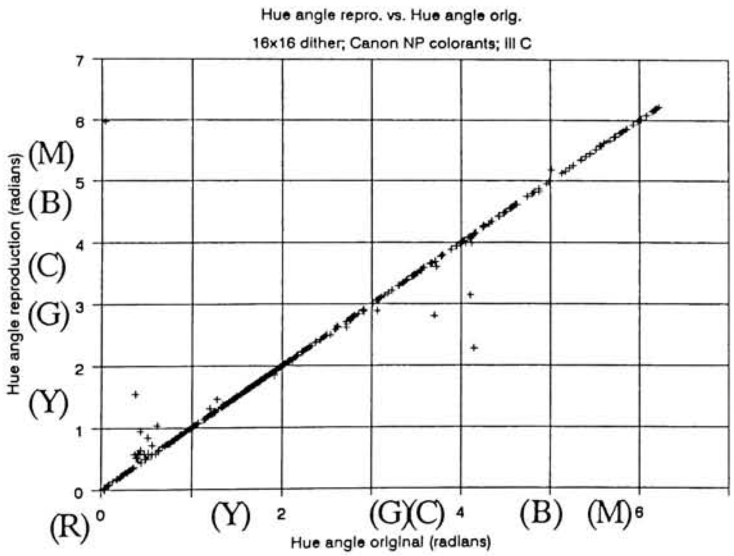


Figure 95

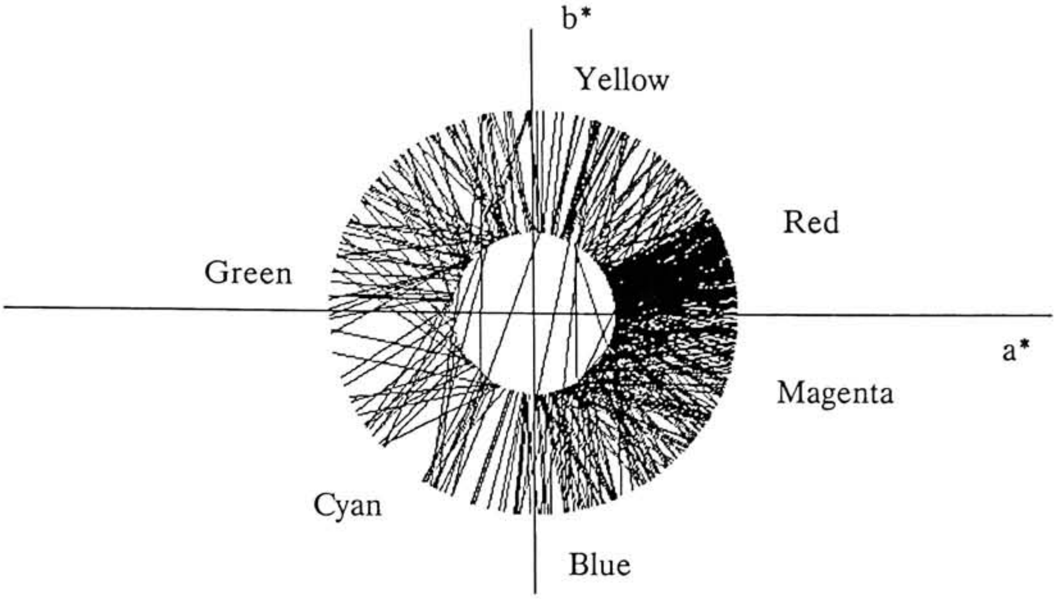


Figure 96

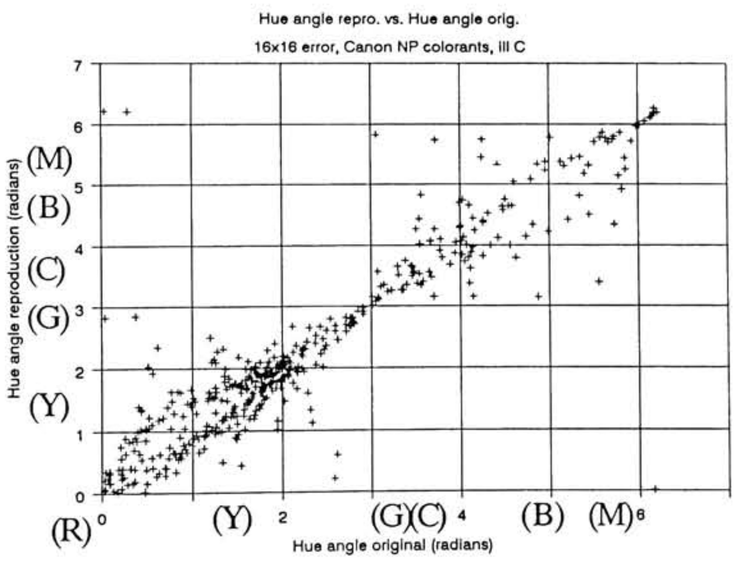


Figure 97

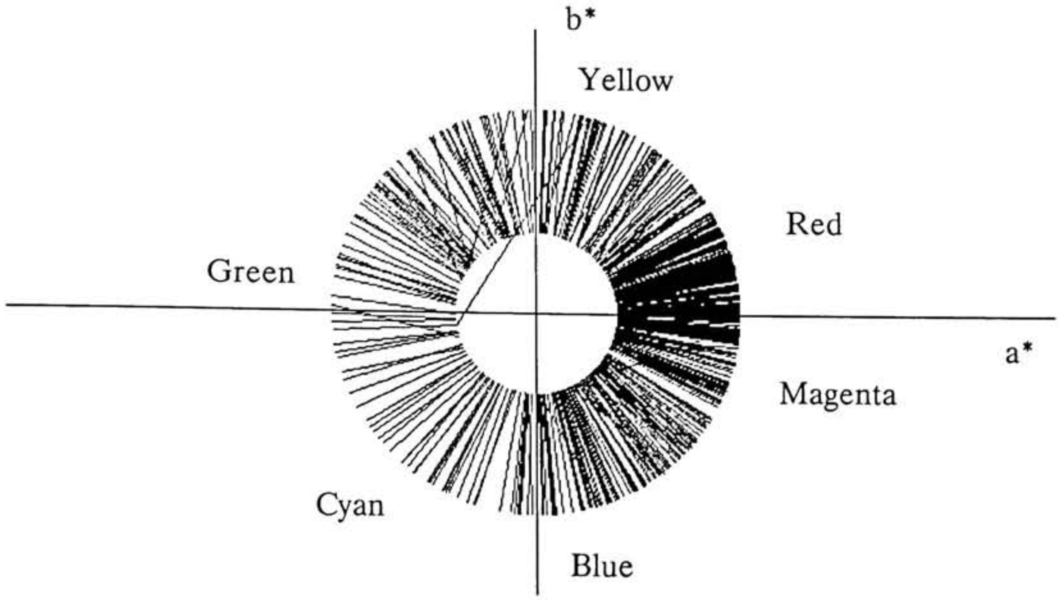


Figure 98

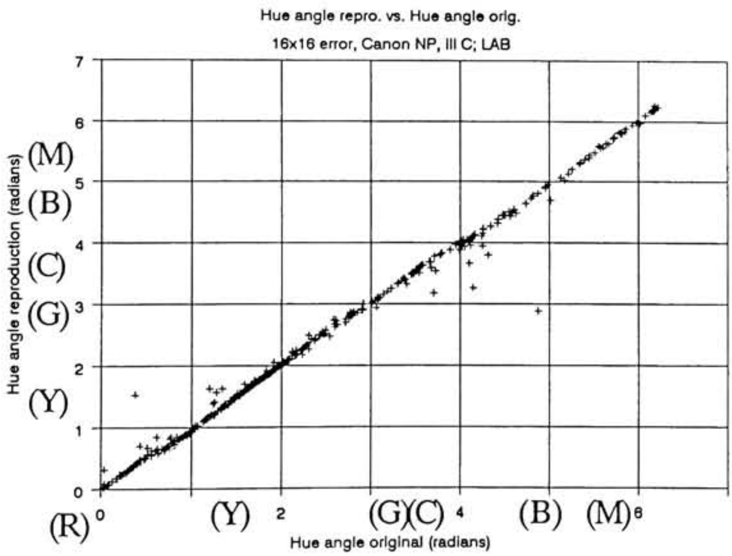


Figure 99

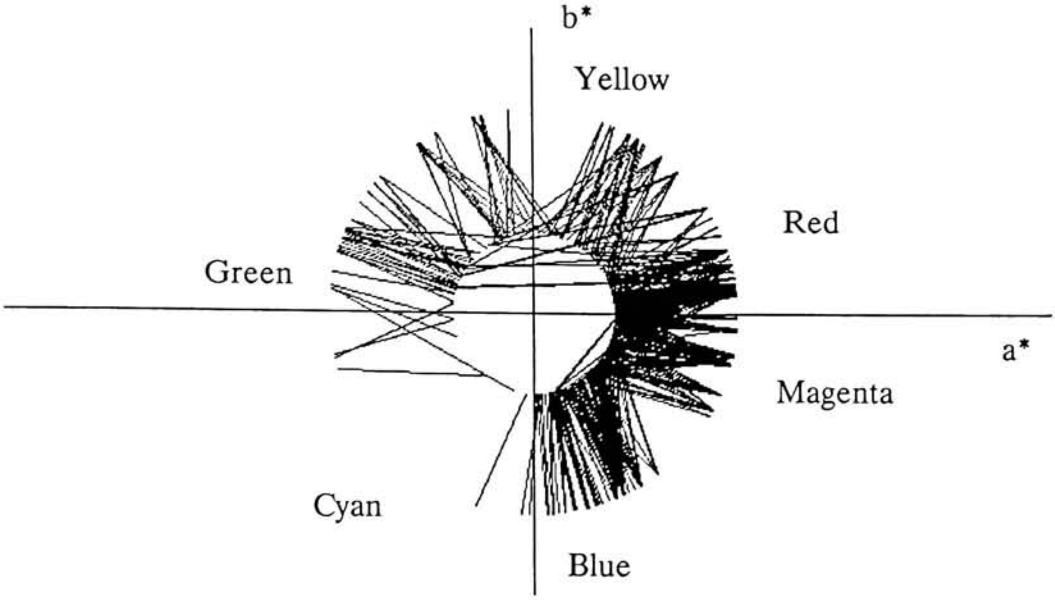


Figure 100

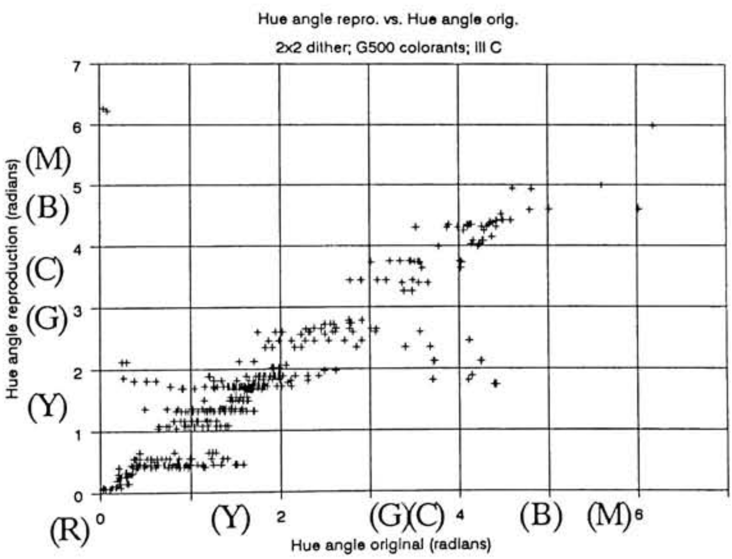


Figure 101

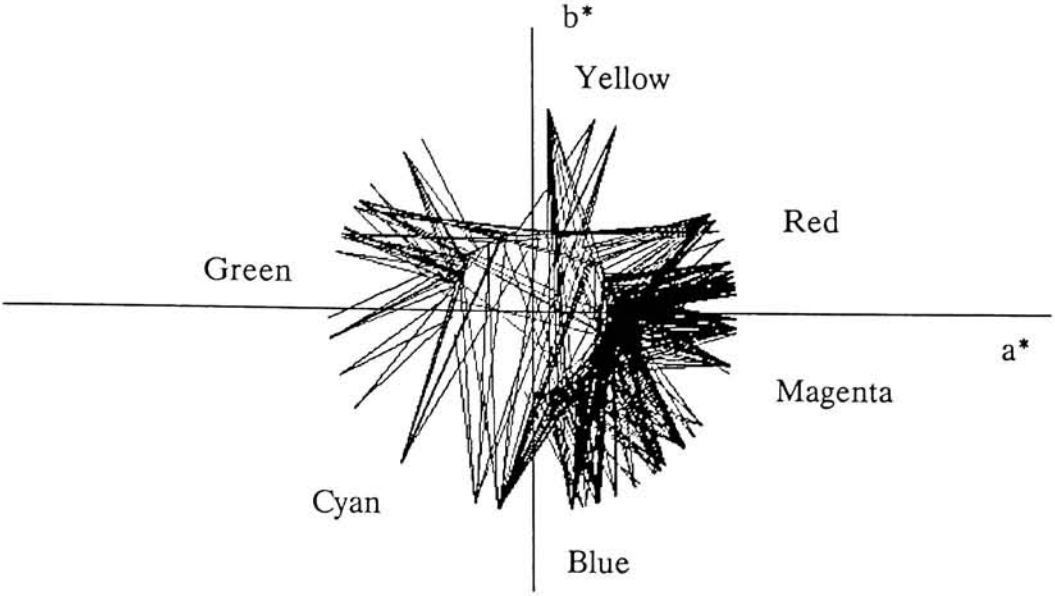


Figure 102

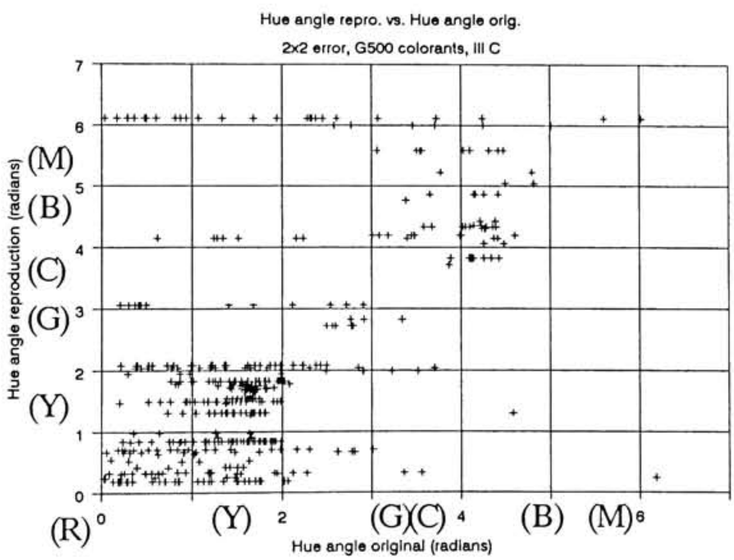


Figure 103

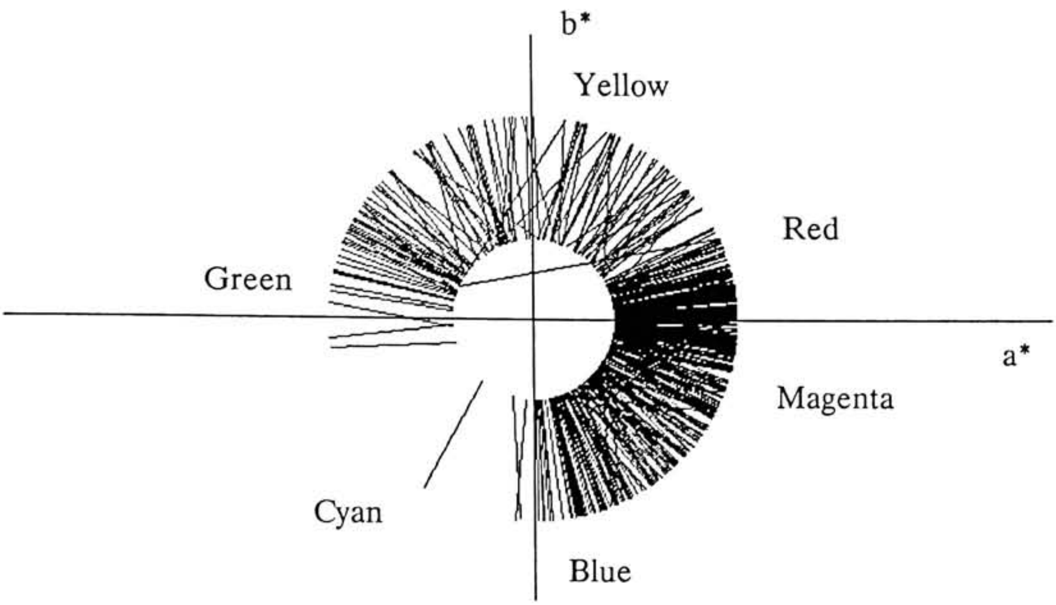


Figure 104

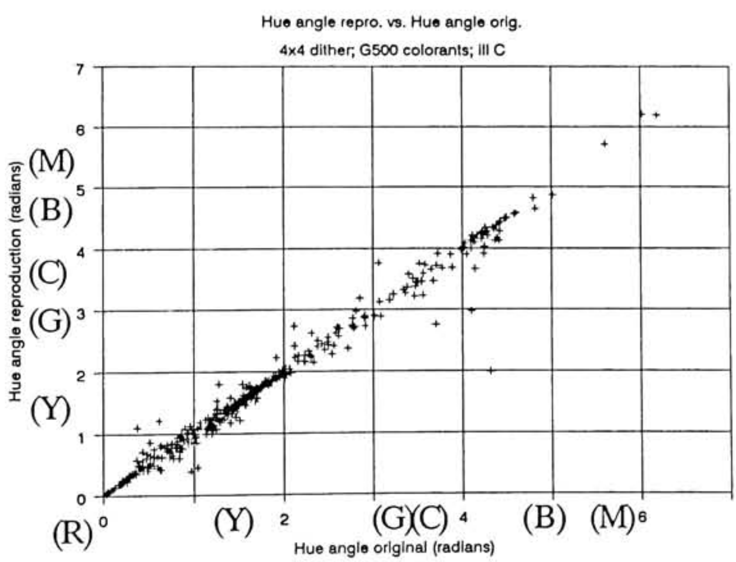


Figure 105

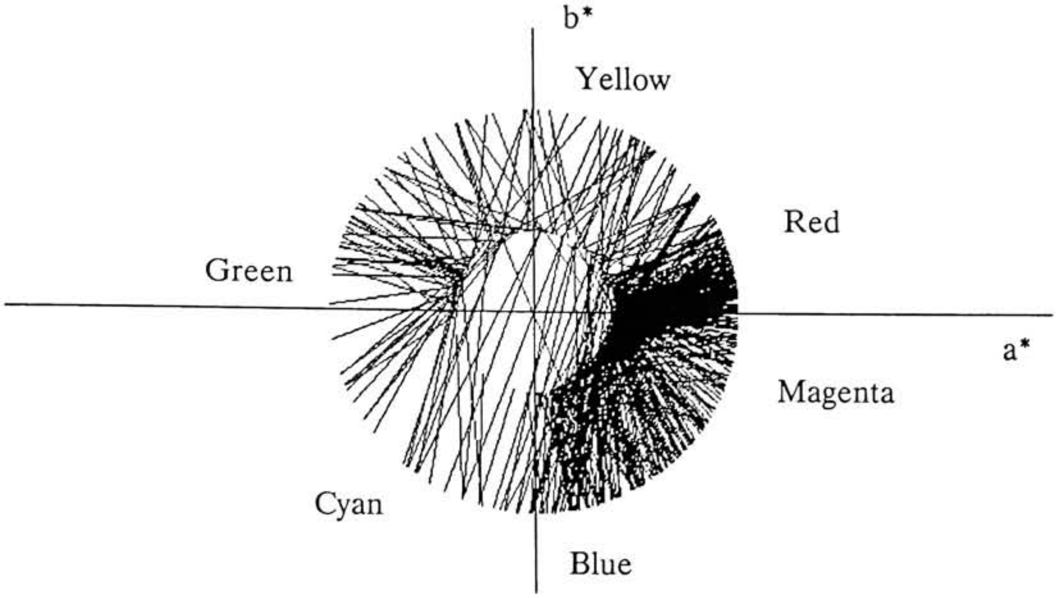


Figure 106

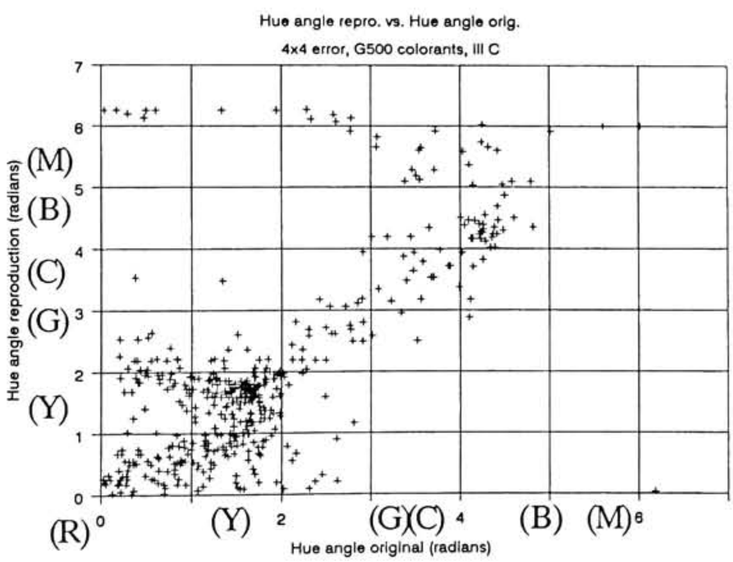


Figure 107

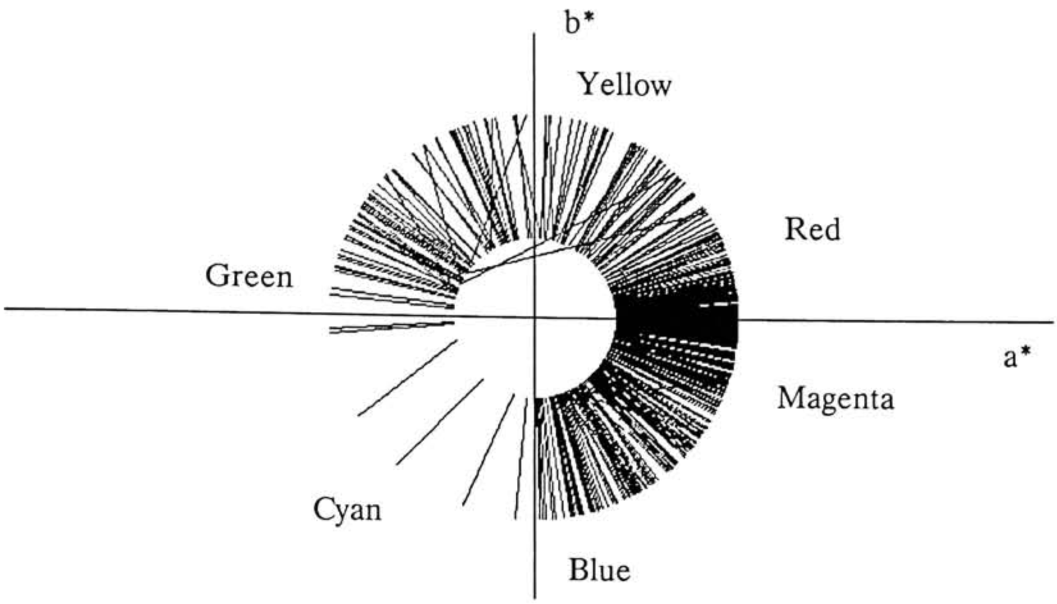


Figure 108

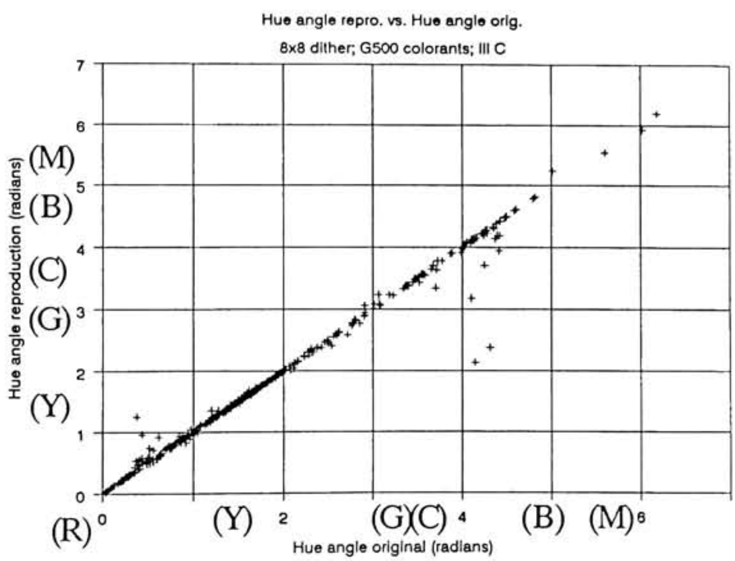


Figure 109

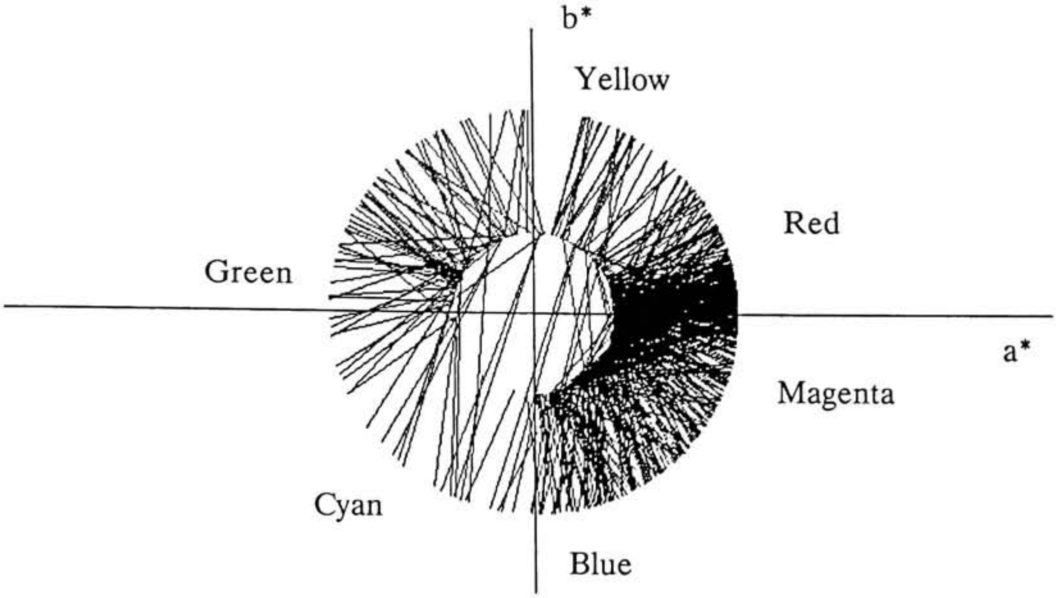


Figure 110

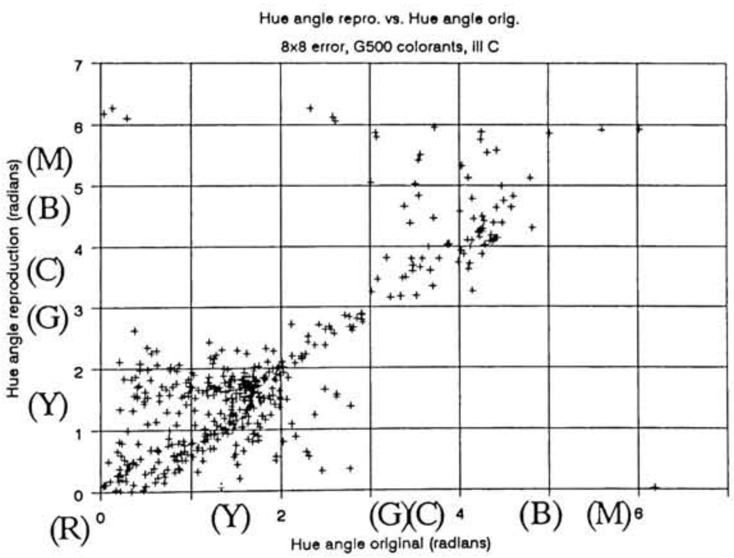


Figure 111

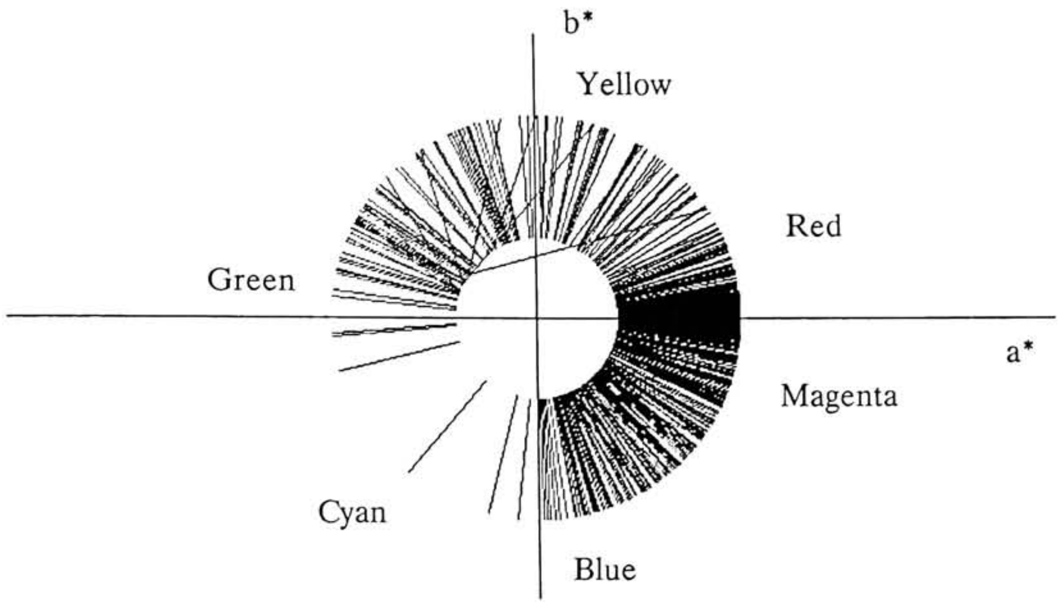


Figure 112

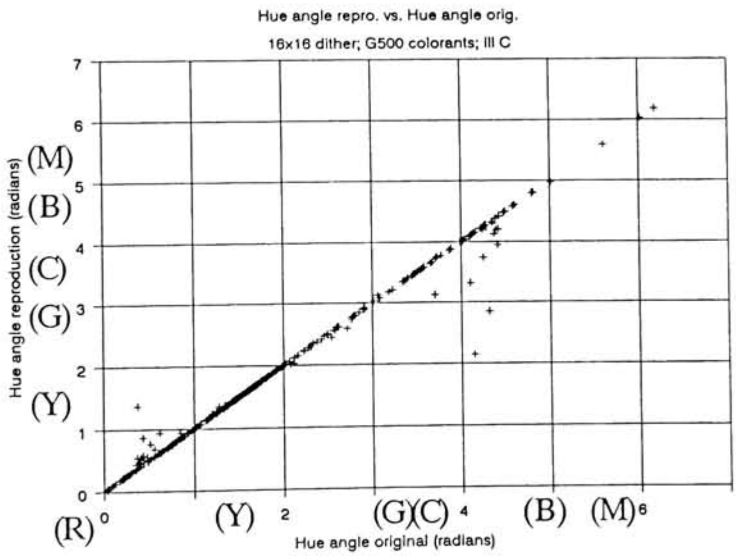


Figure 113

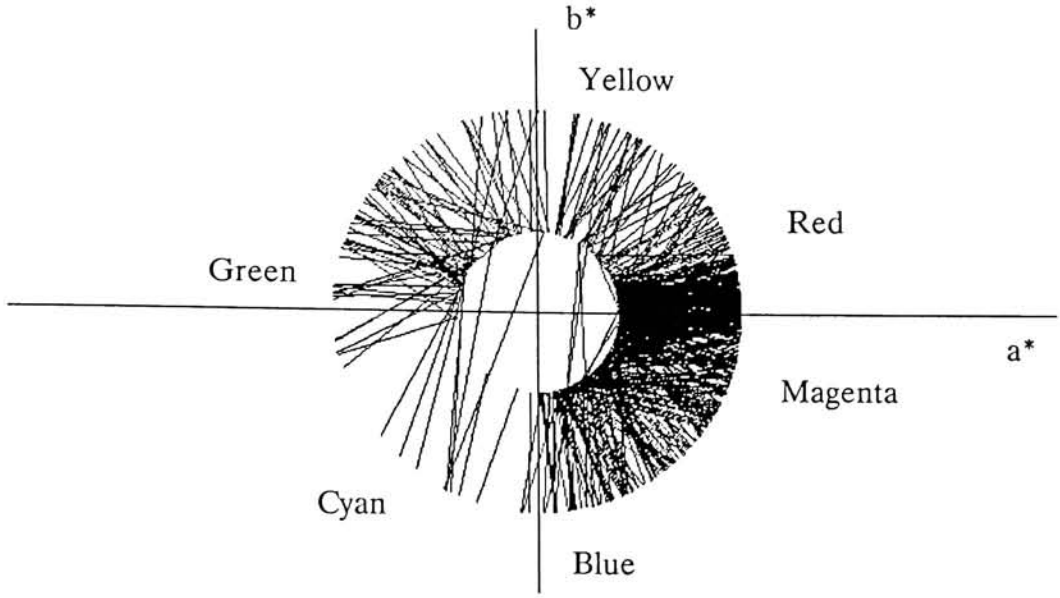


Figure 114

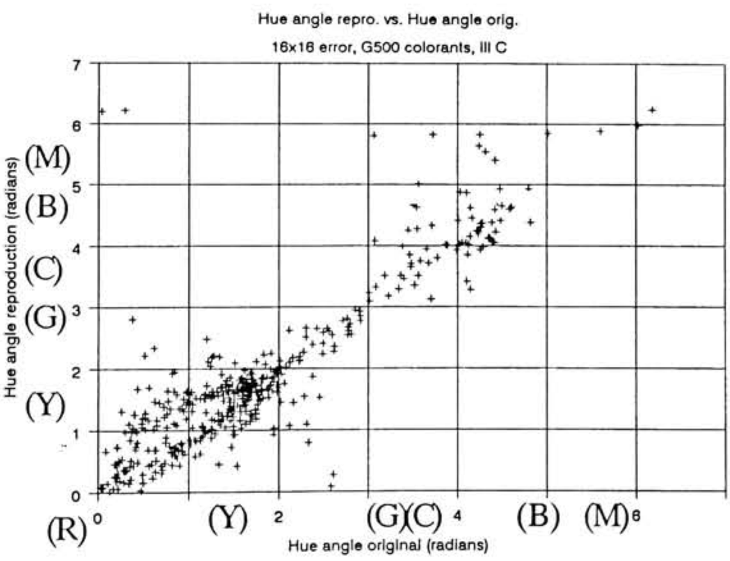


Figure 115

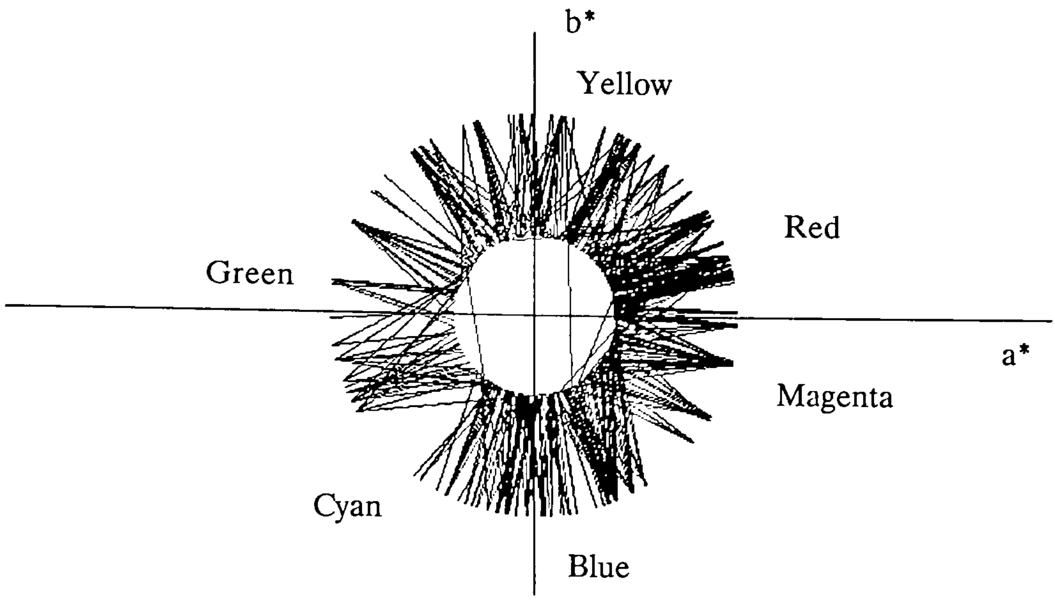


Figure 116

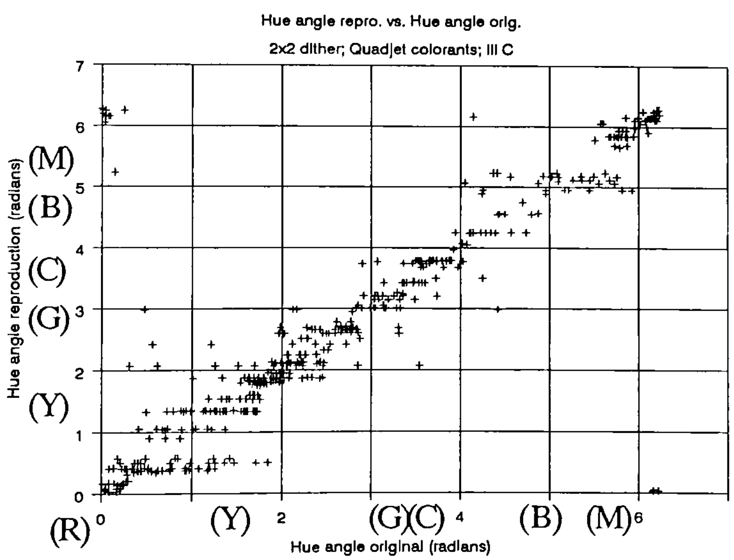


Figure 117

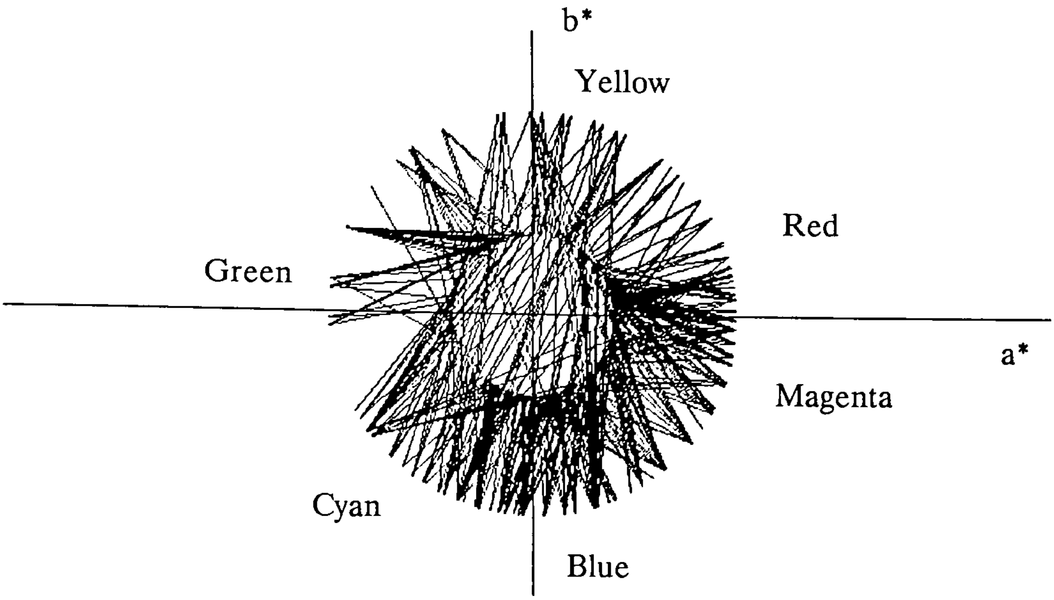


Figure 118

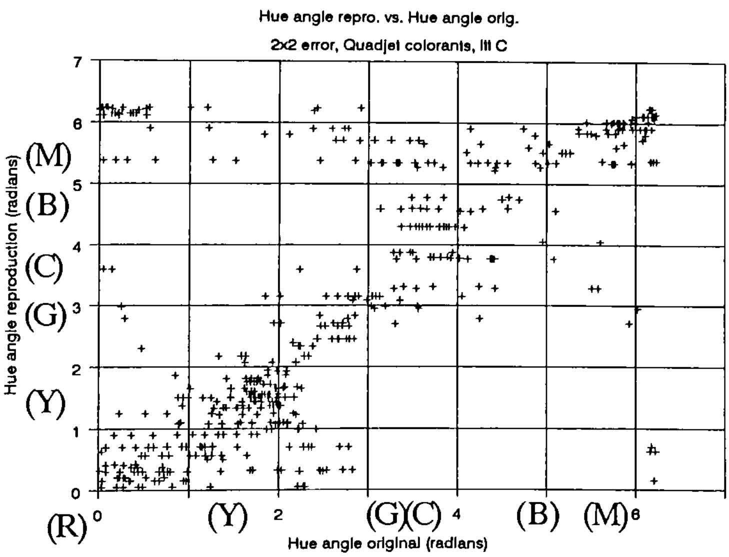


Figure 119

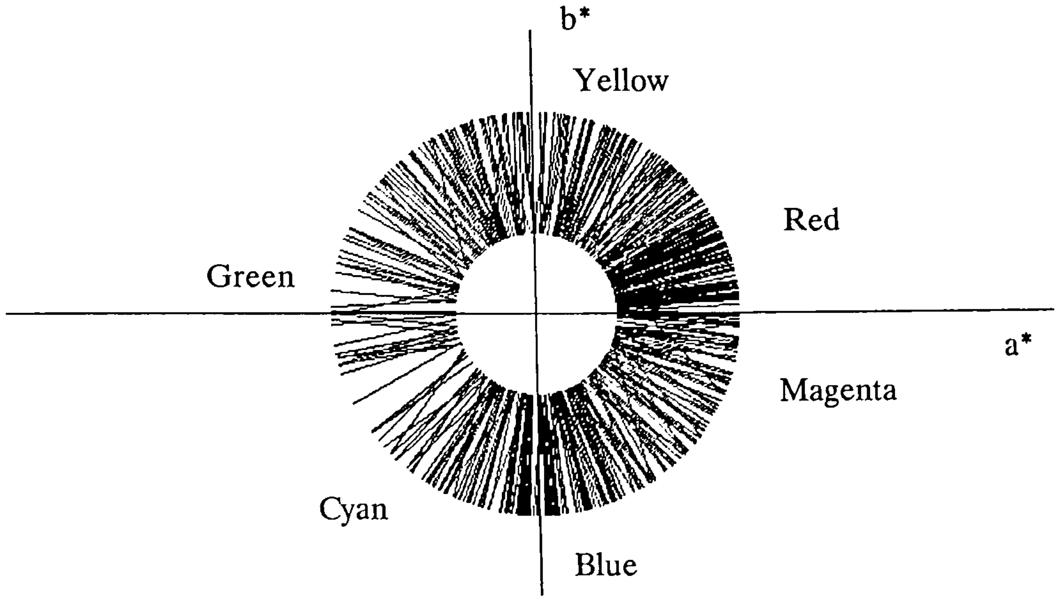


Figure 120

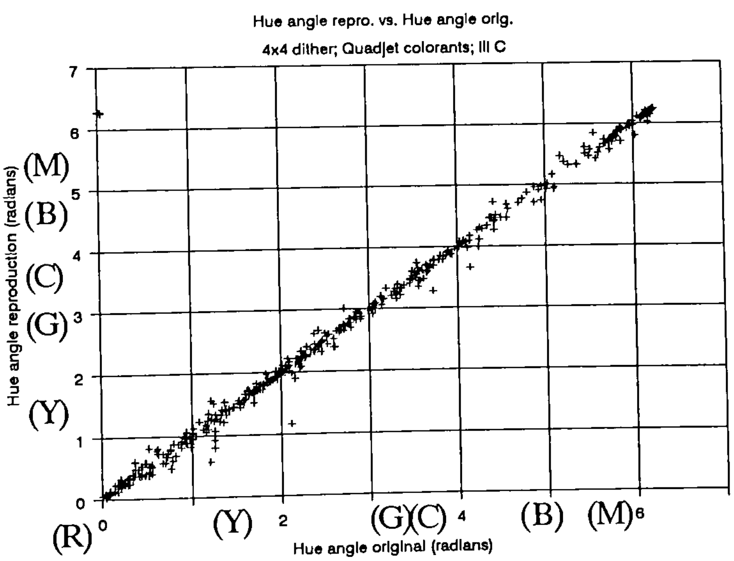


Figure 121

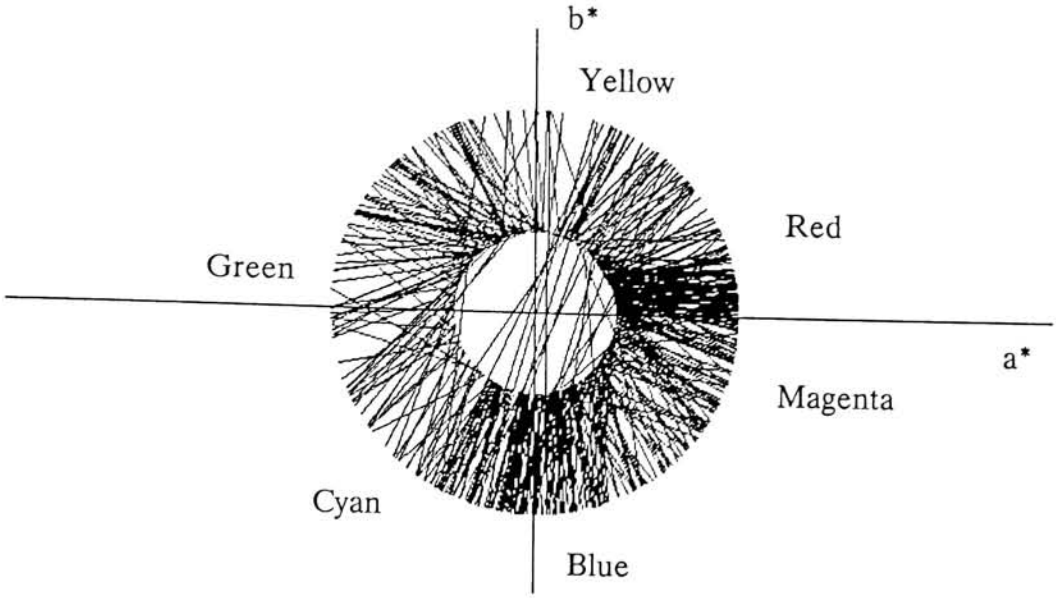


Figure 122

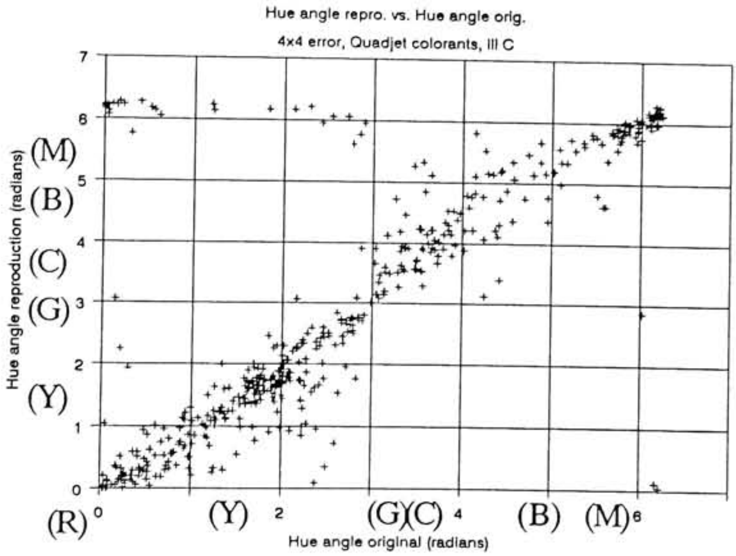


Figure 123

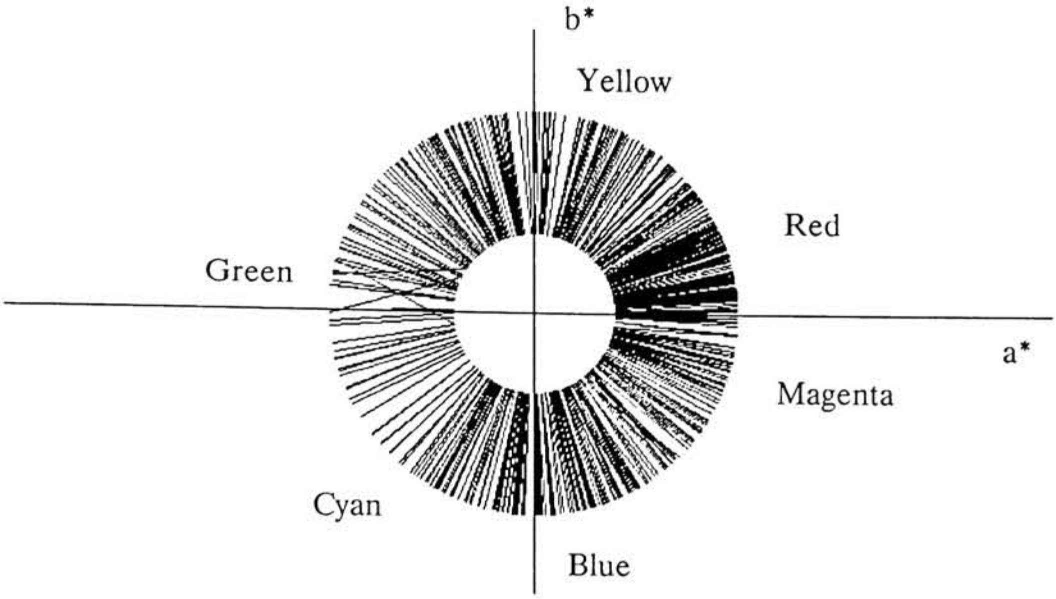


Figure 124

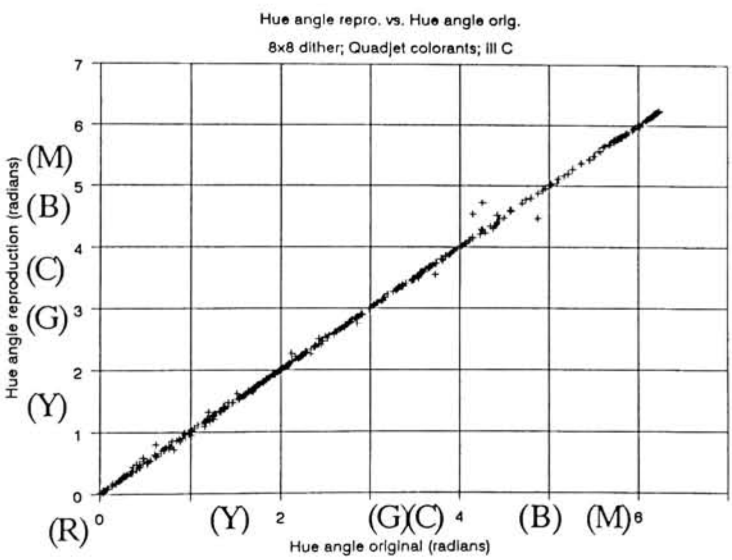


Figure 125

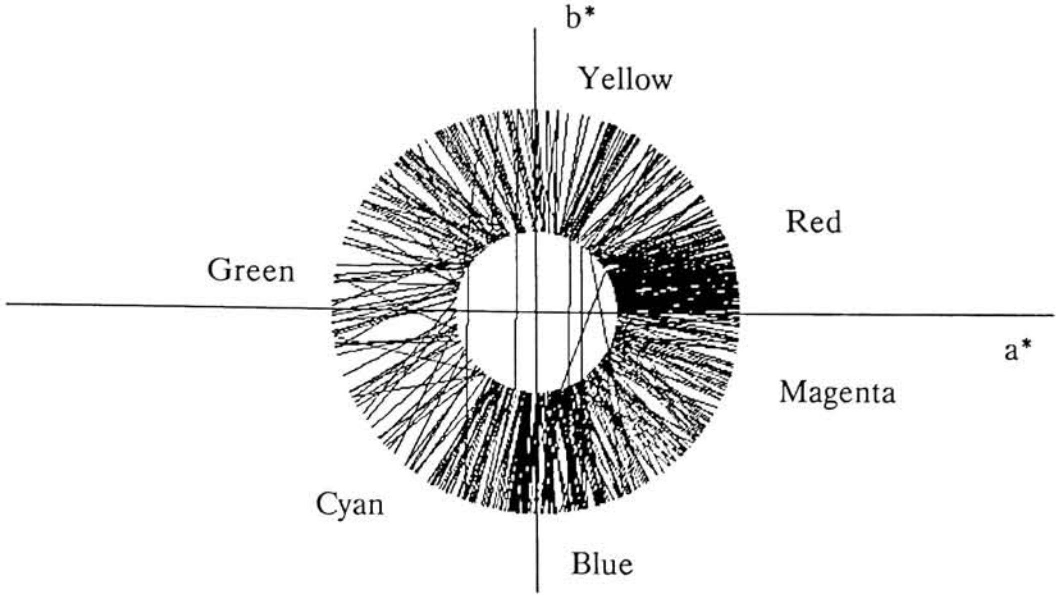


Figure 126

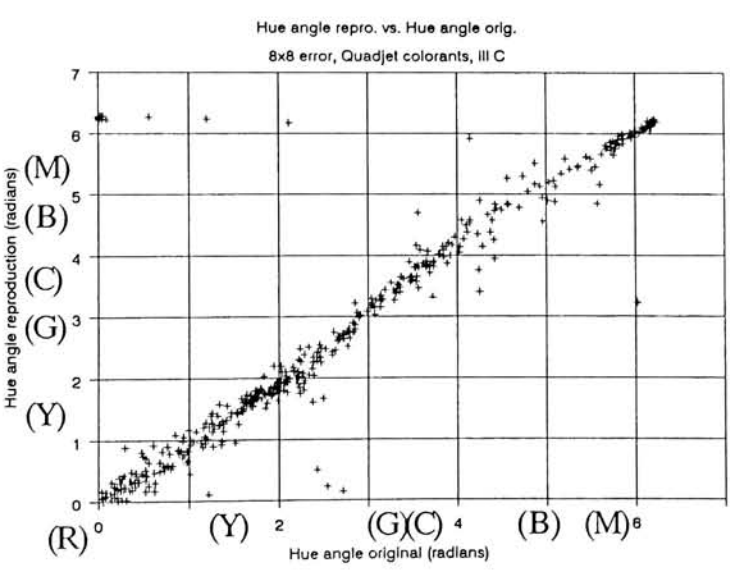


Figure 127

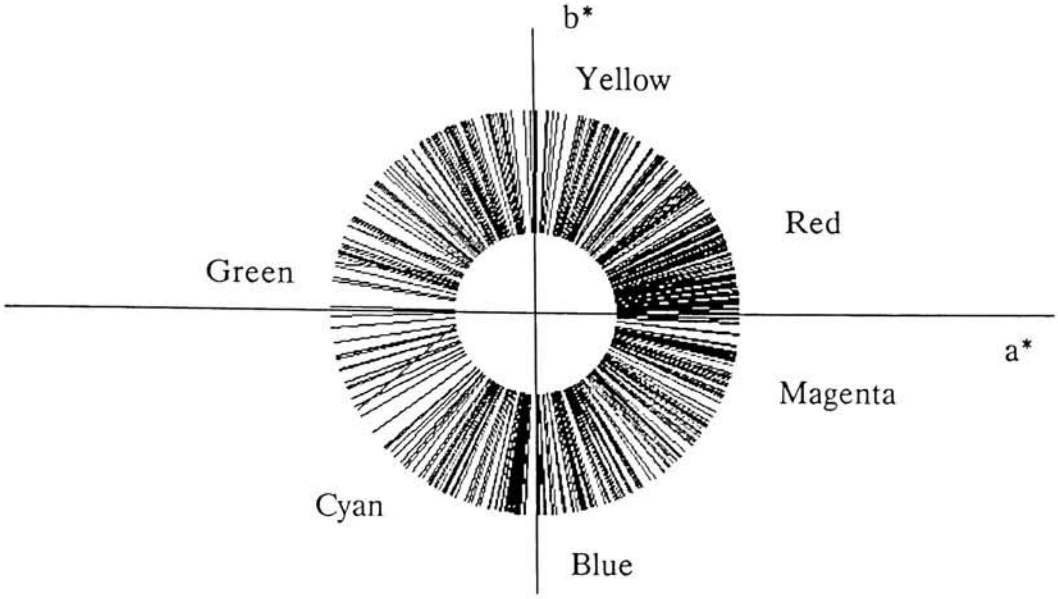


Figure 128

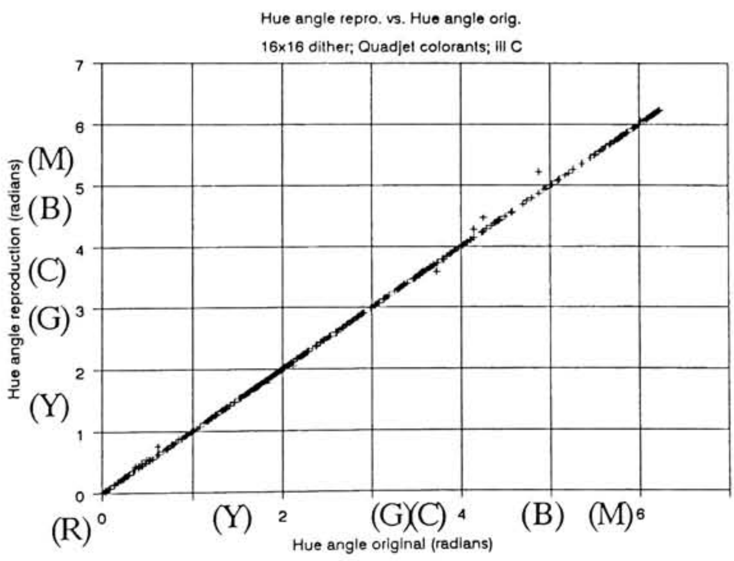


Figure 129

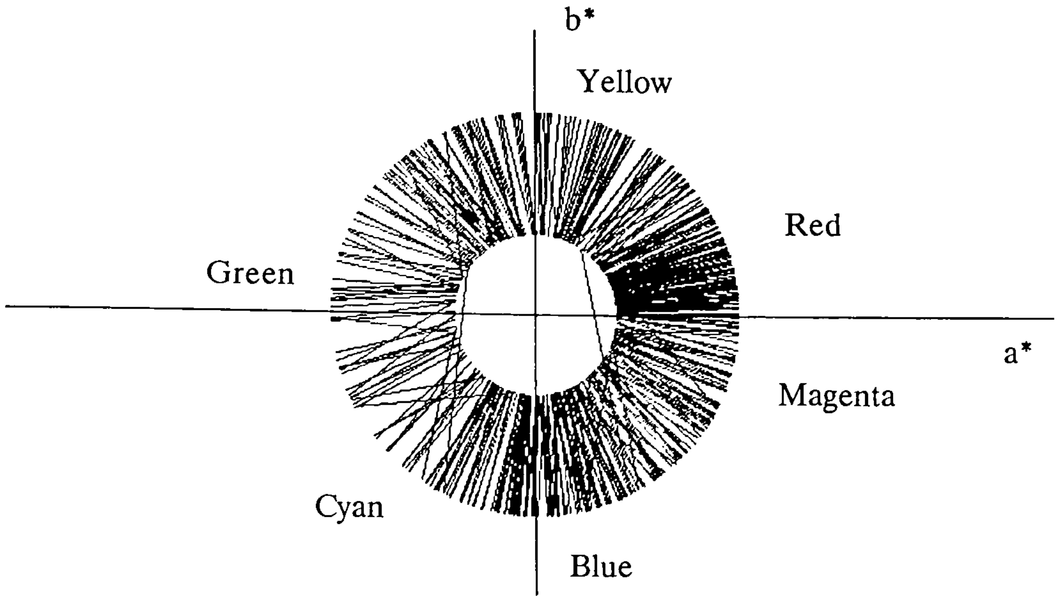


Figure 130

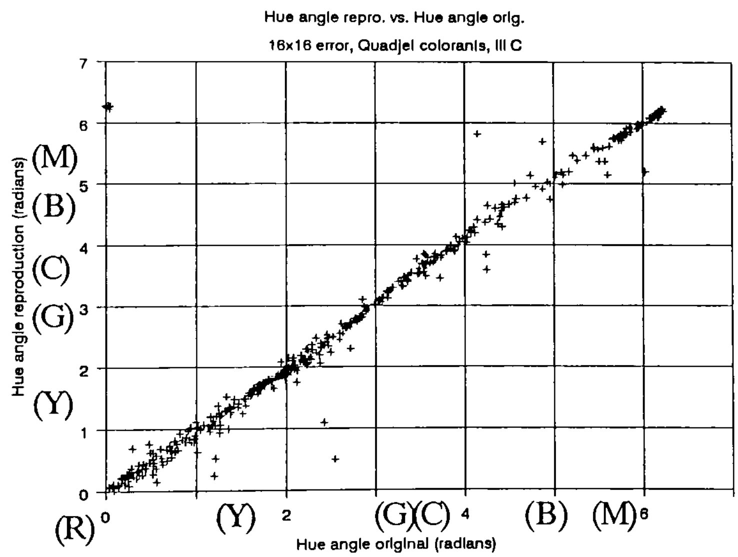


Figure 131

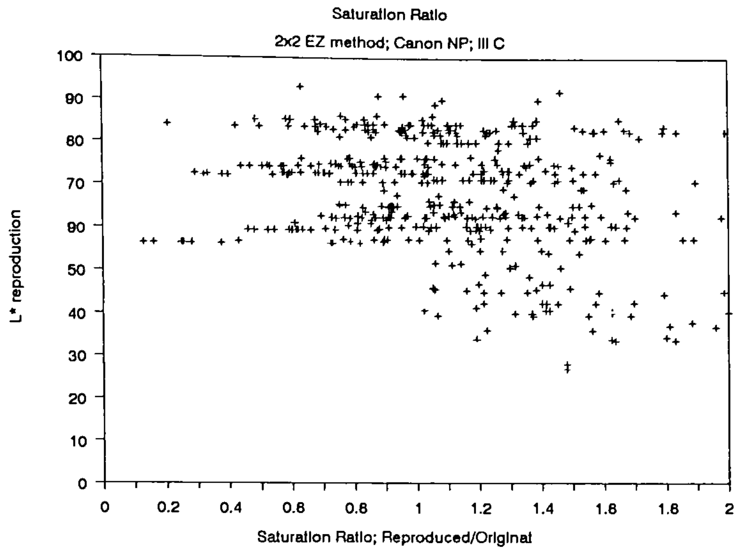


Figure 132

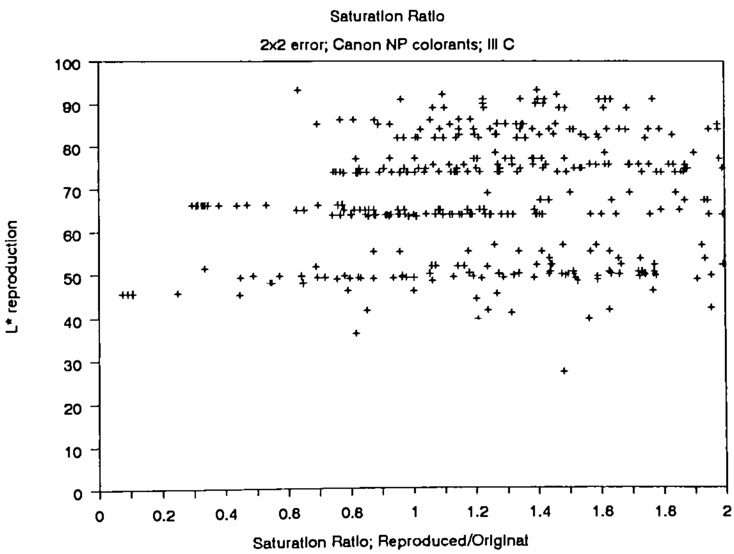


Figure 133

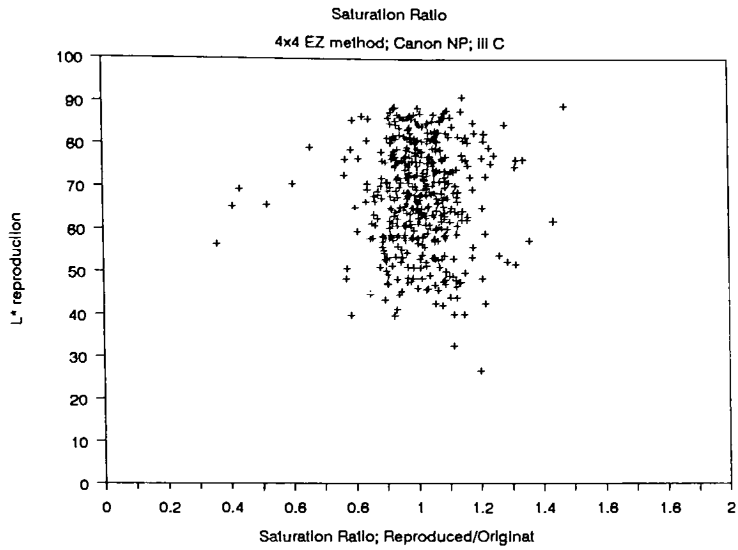


Figure 134

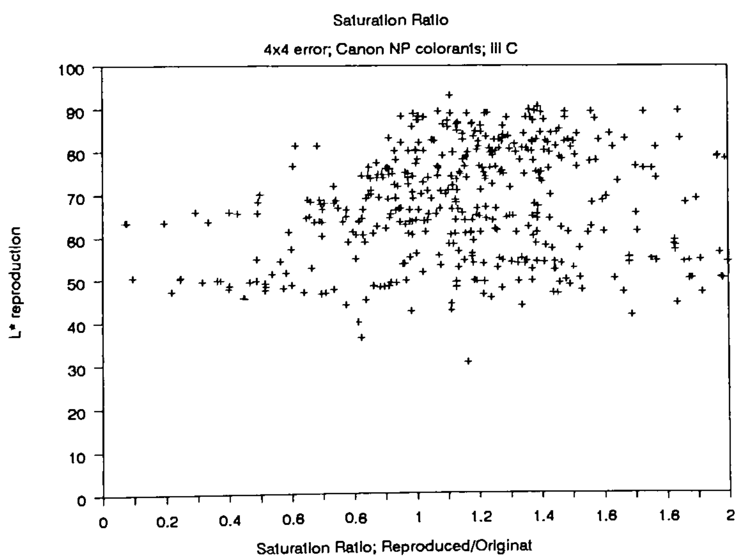


Figure 135

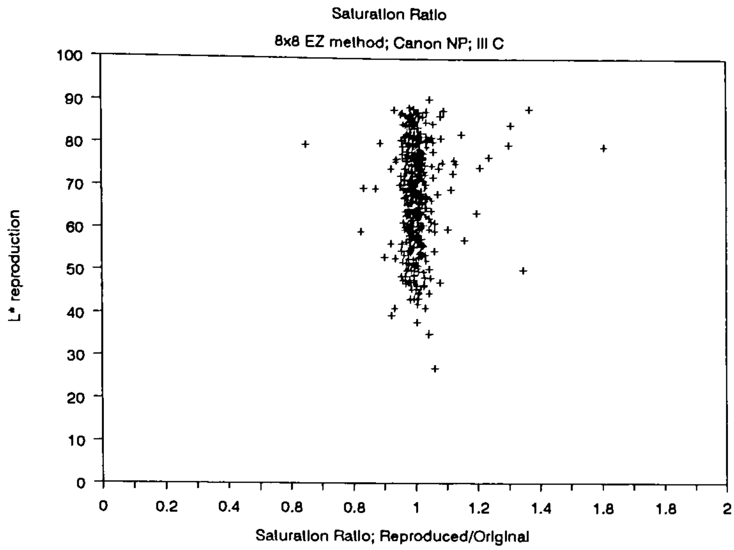


Figure 136

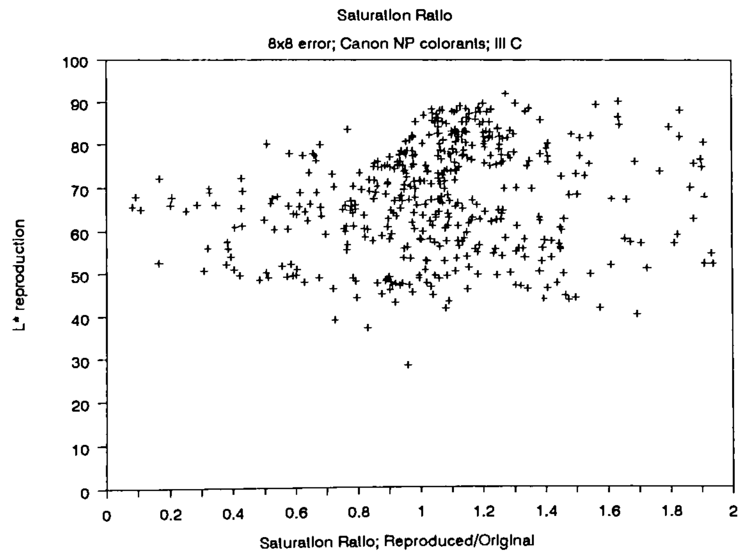


Figure 137

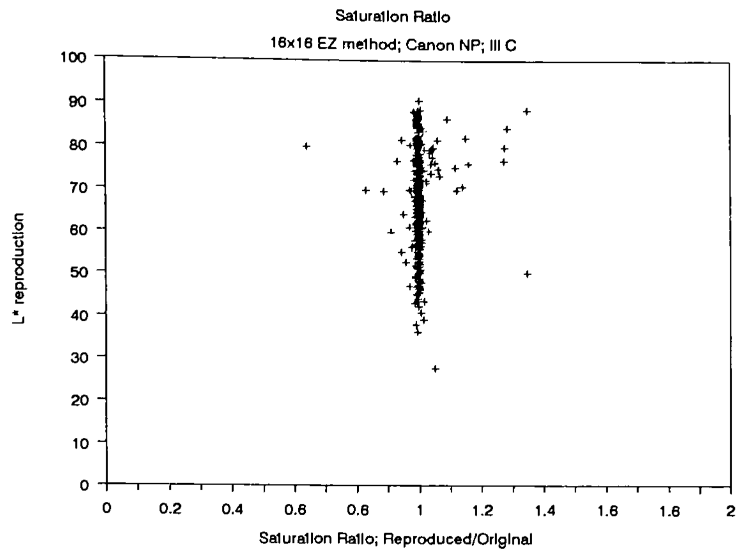


Figure 138

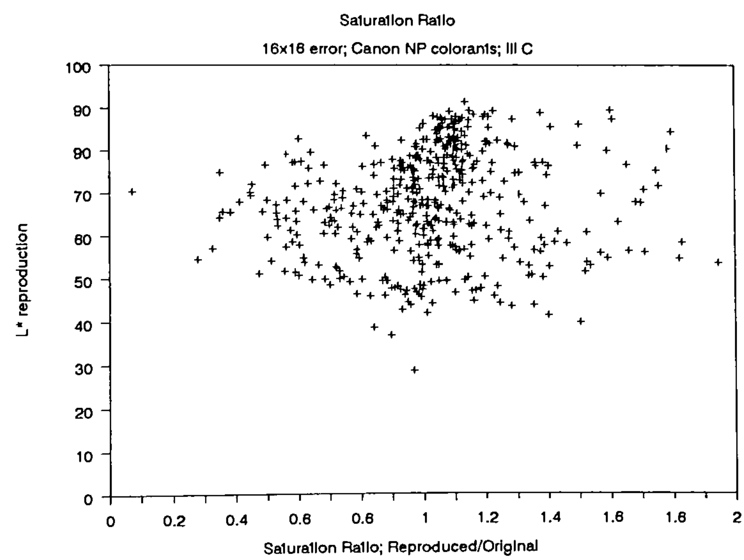


Figure 139

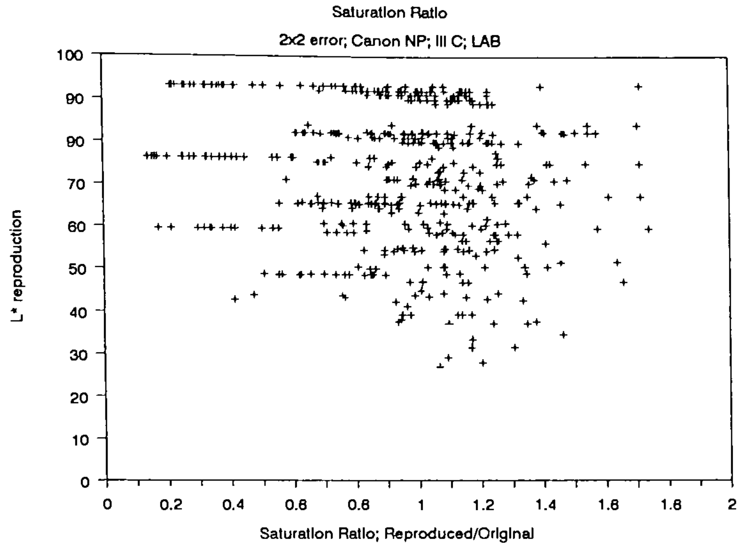


Figure 140

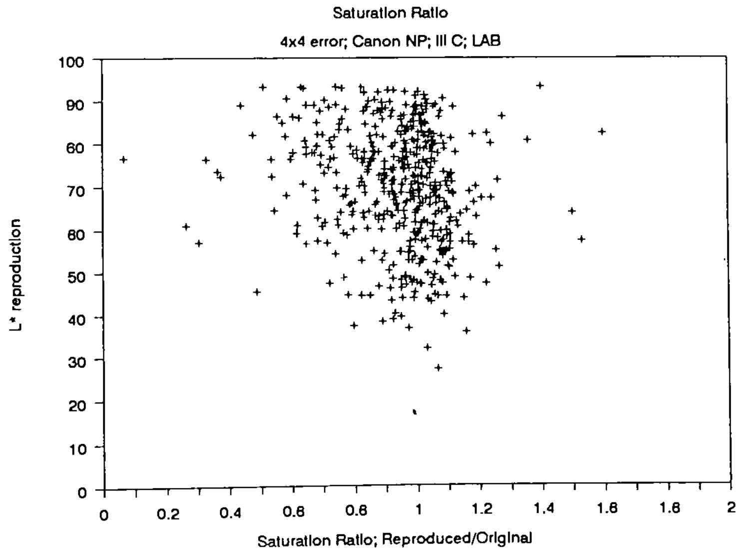


Figure 141

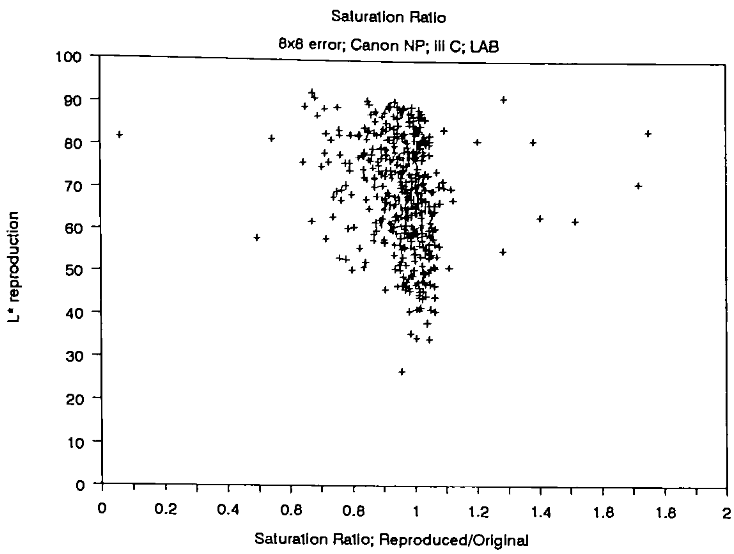


Figure 142

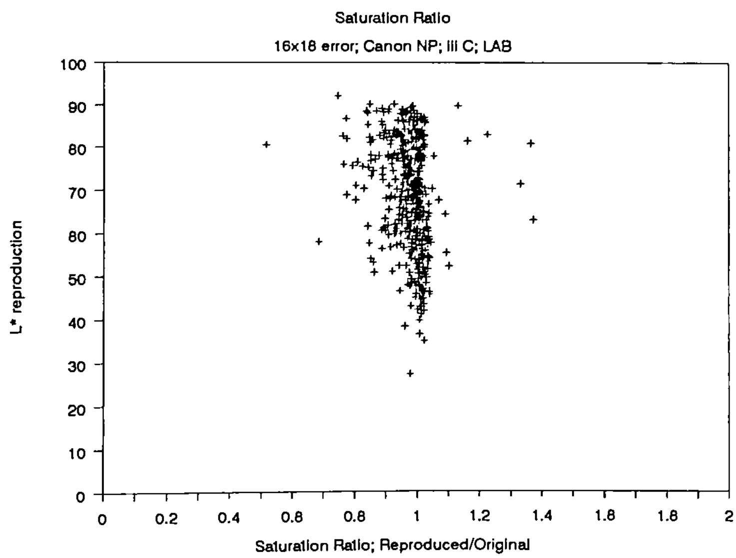


Figure 143

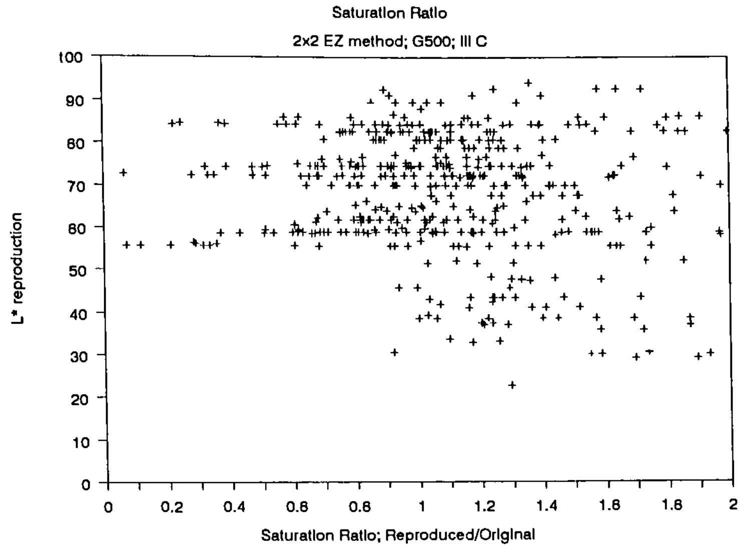


Figure 144

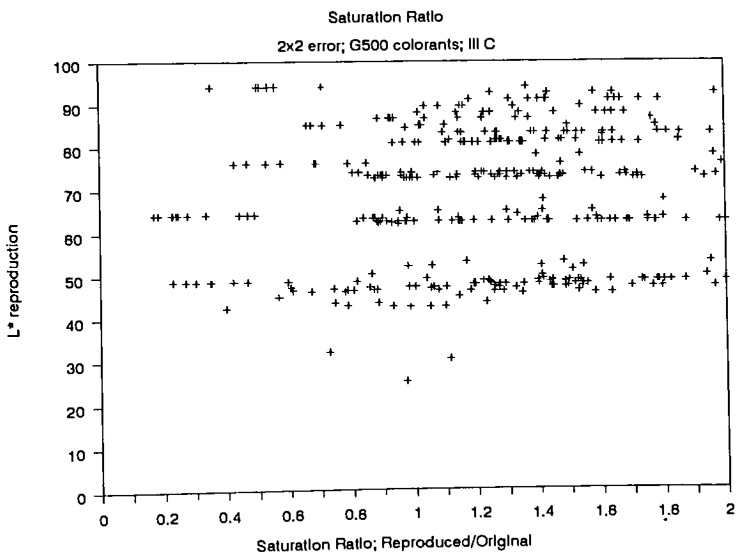


Figure 145

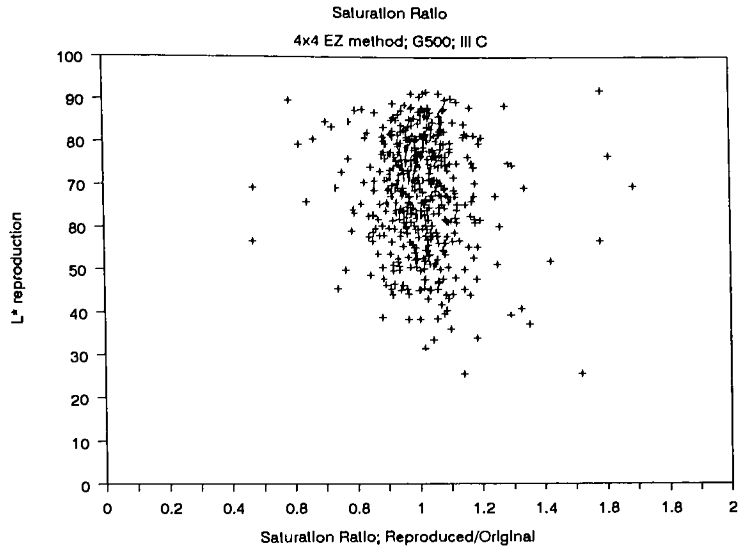


Figure 146

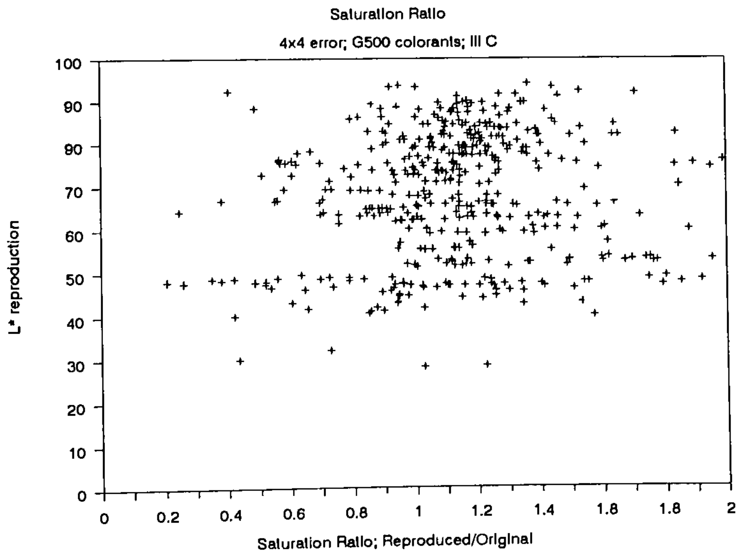


Figure 147

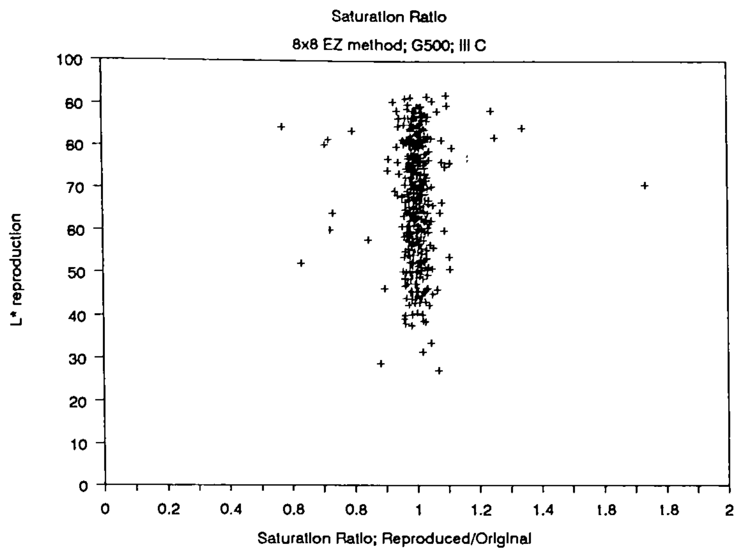


Figure 148

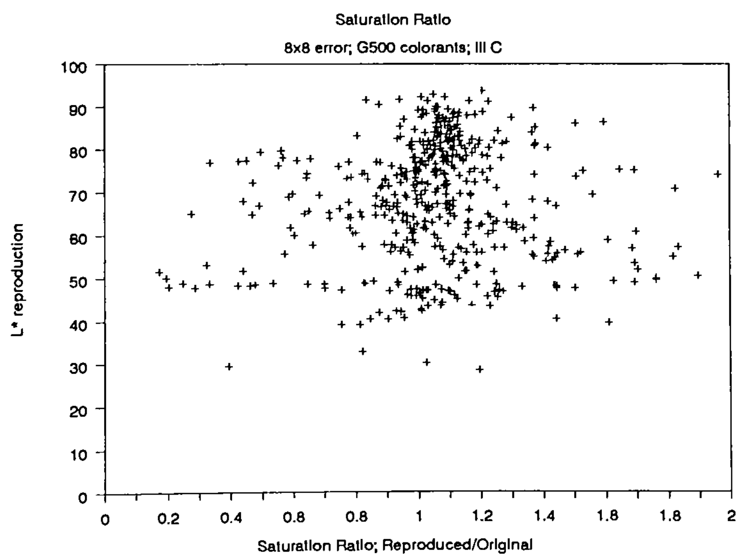


Figure 149

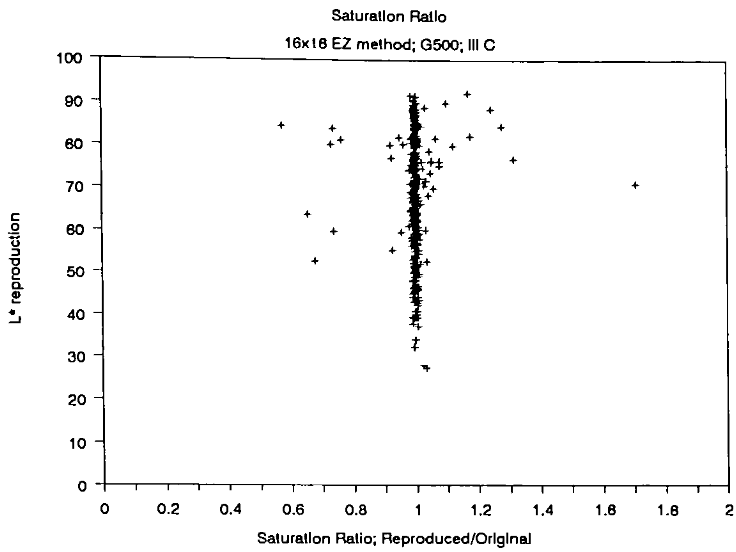


Figure 150

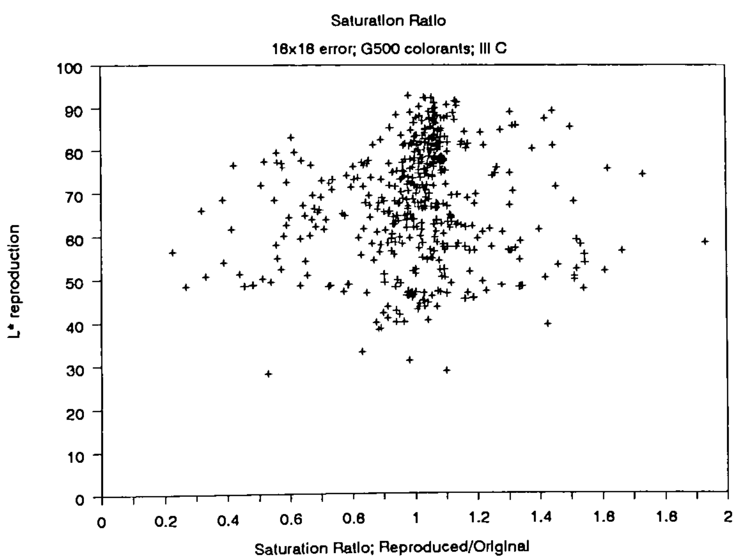


Figure 151

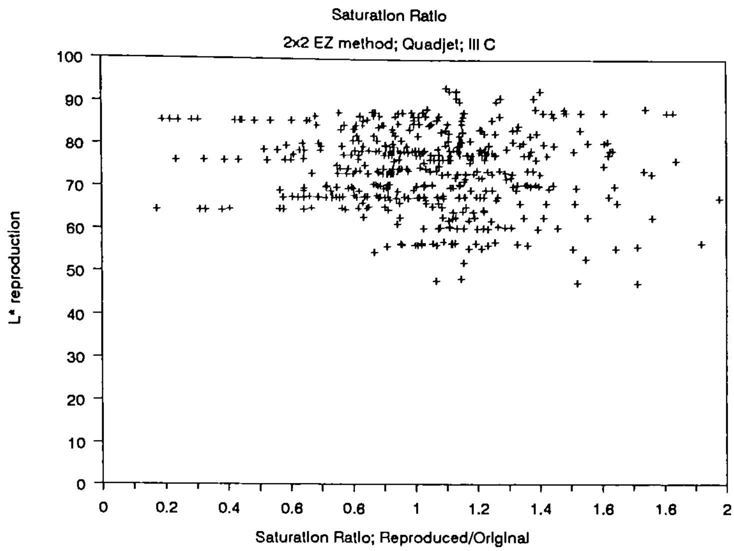


Figure 152

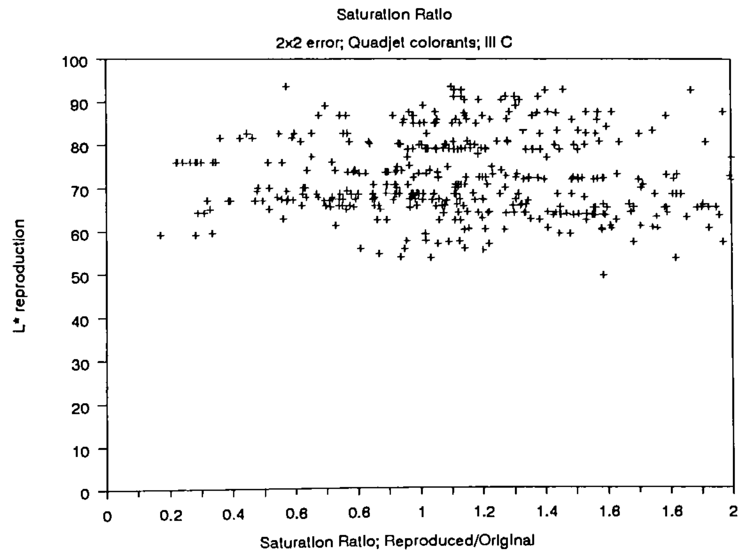


Figure 153

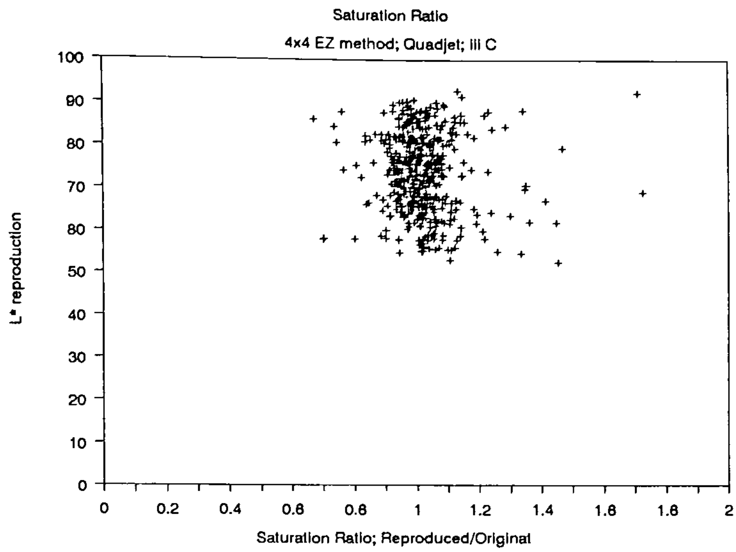


Figure 154

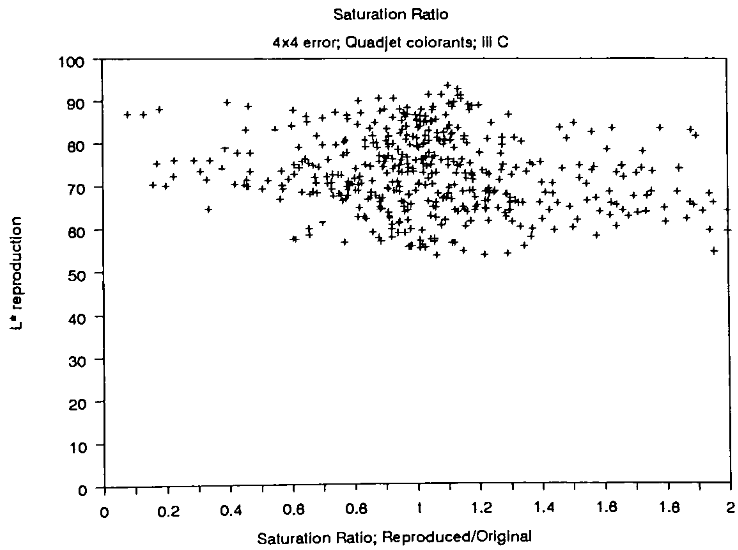


Figure 155

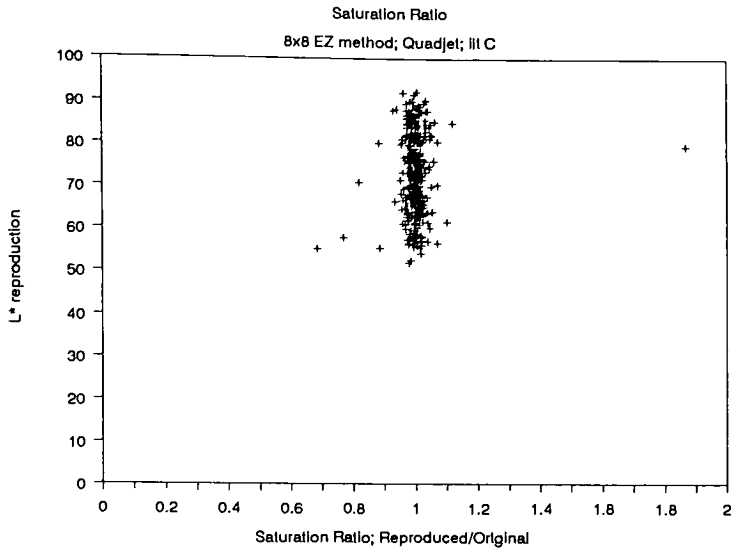


Figure 156

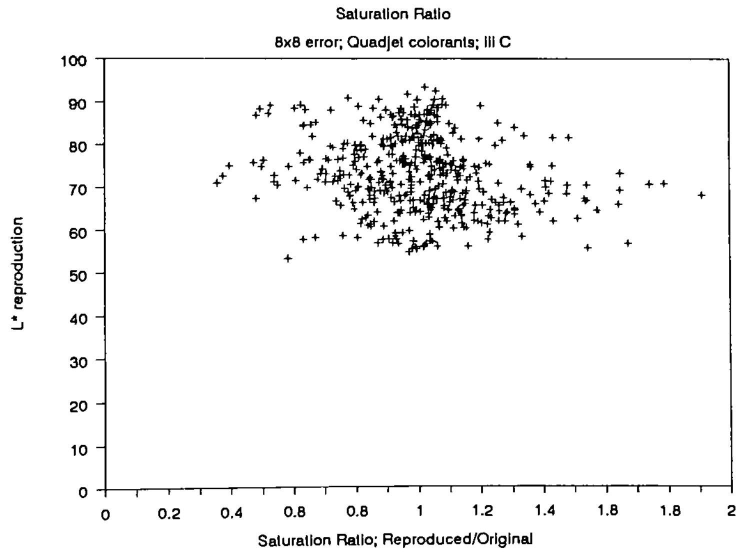


Figure 157

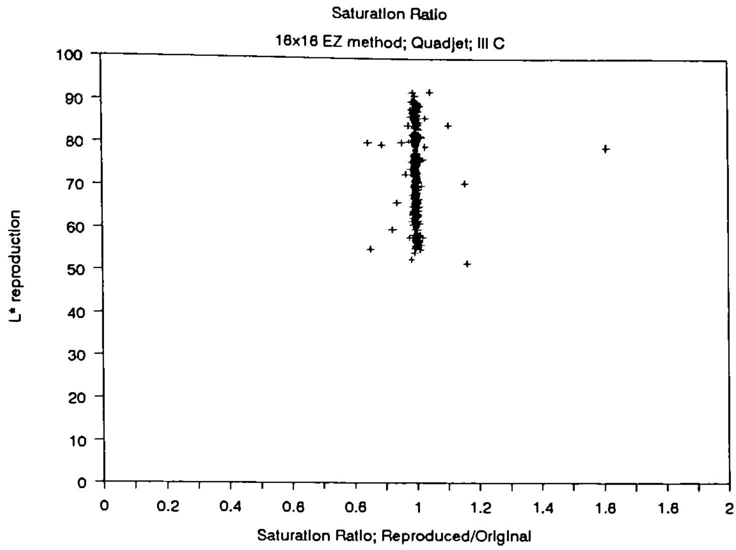


Figure 158

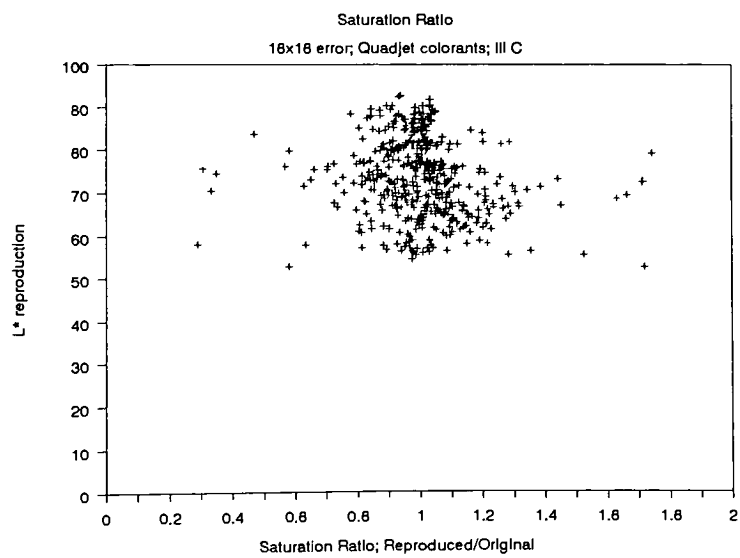


Figure 159

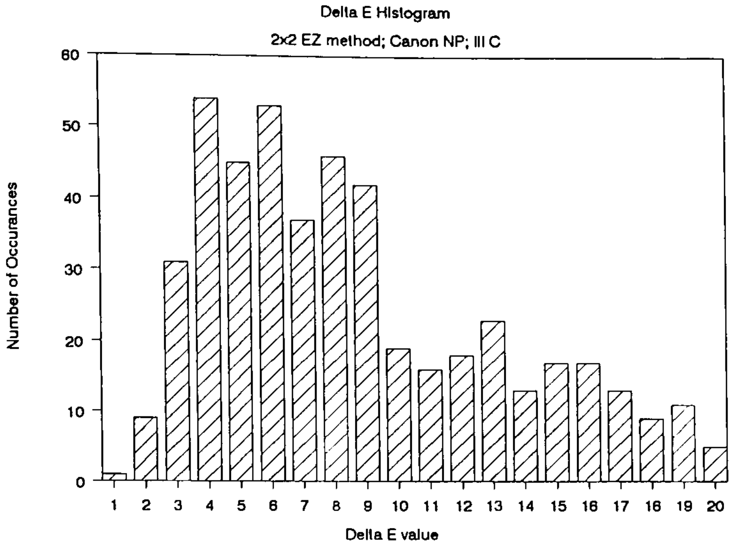


Figure 160

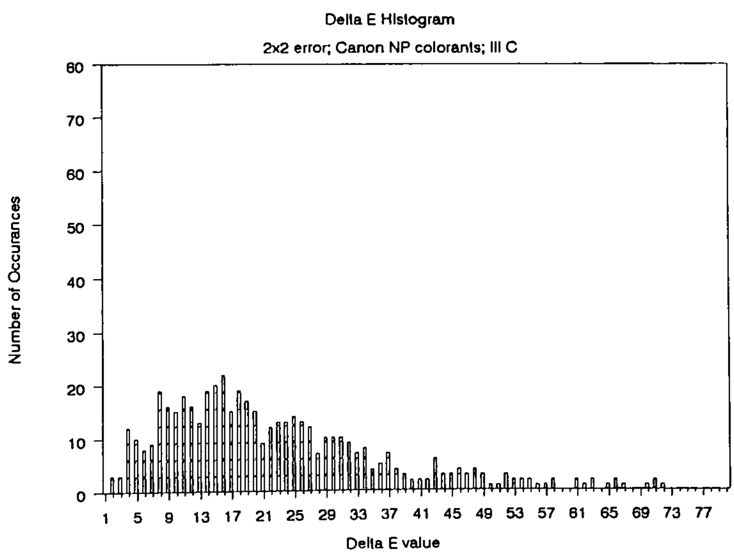


Figure 161

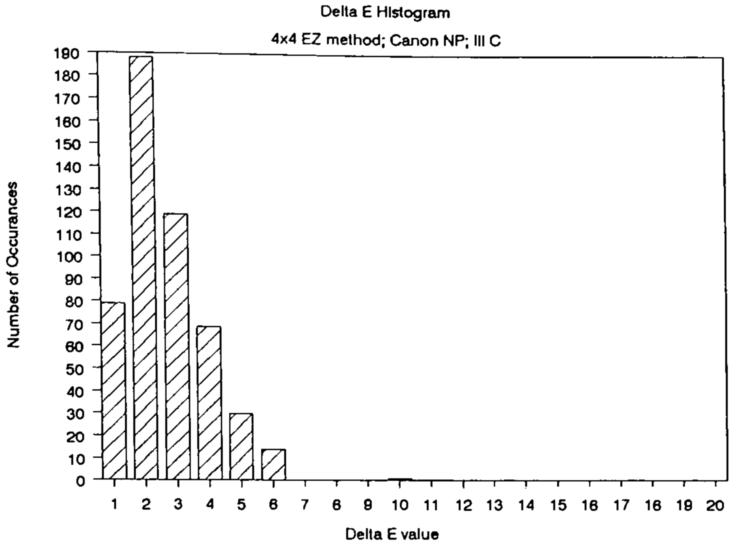


Figure 162

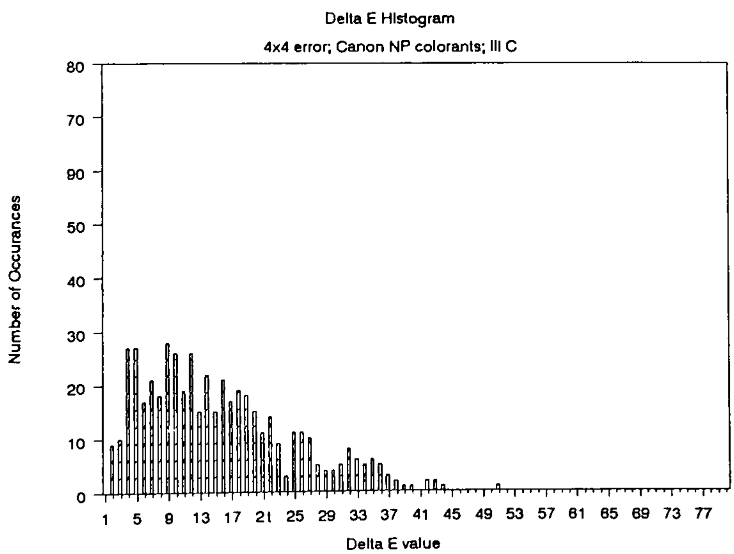


Figure 163

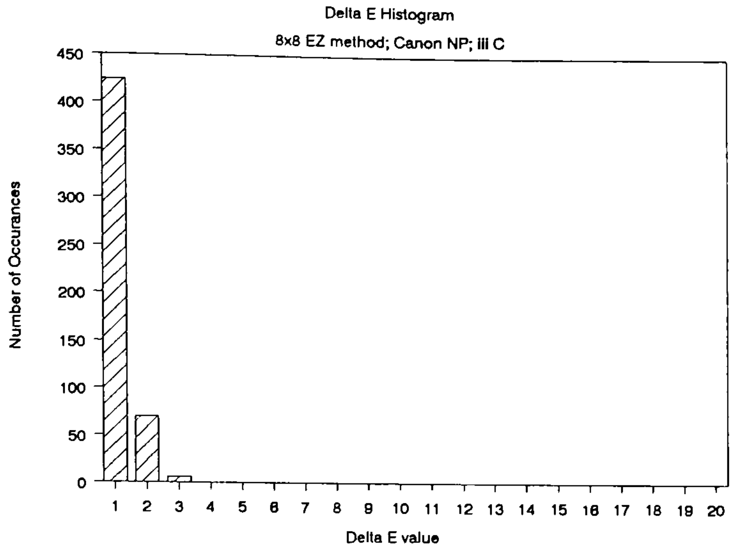


Figure 164

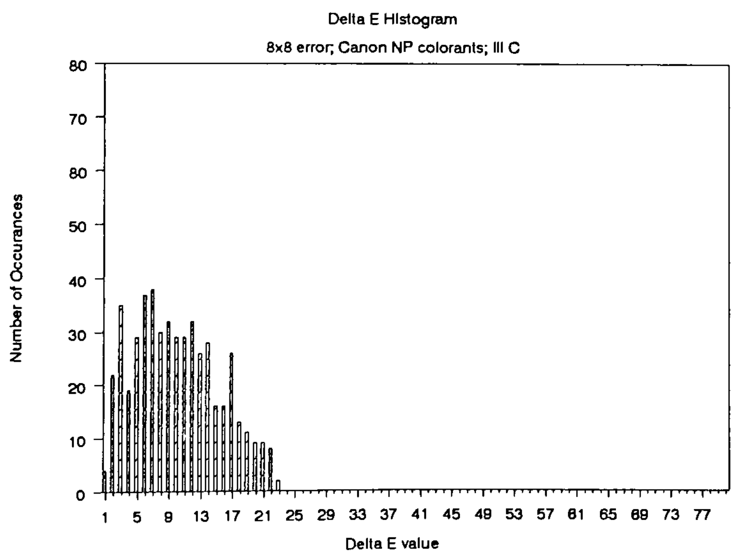


Figure 165

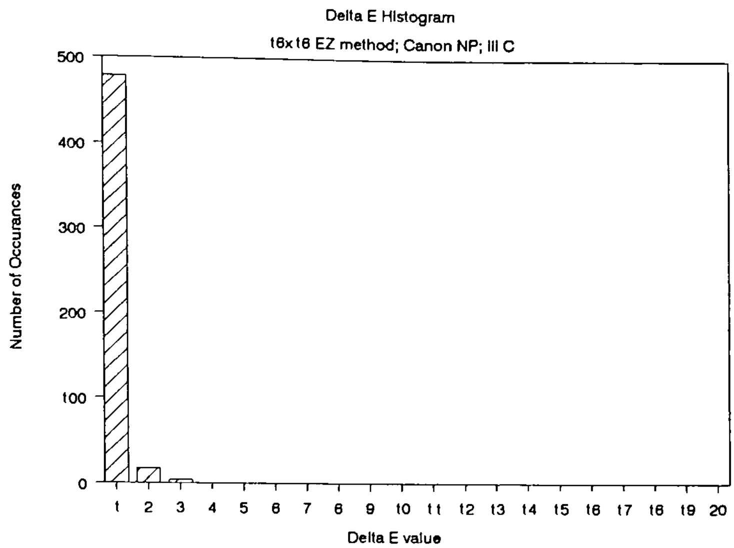


Figure 166

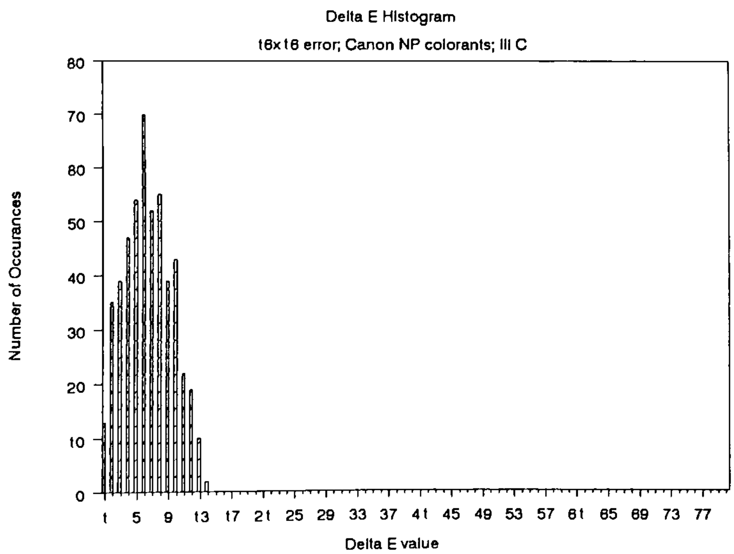


Figure 167

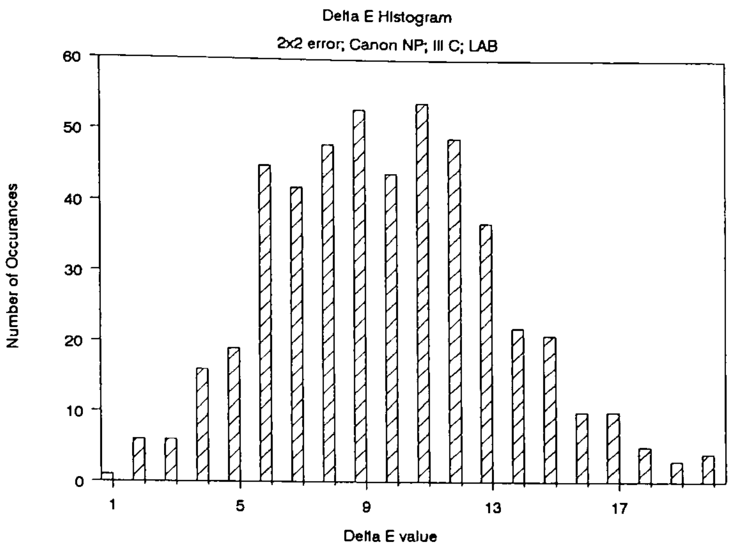


Figure 168

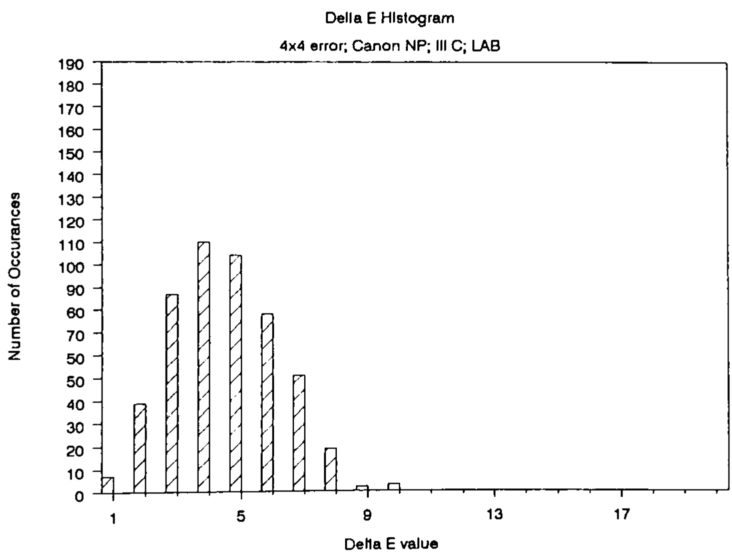


Figure 169

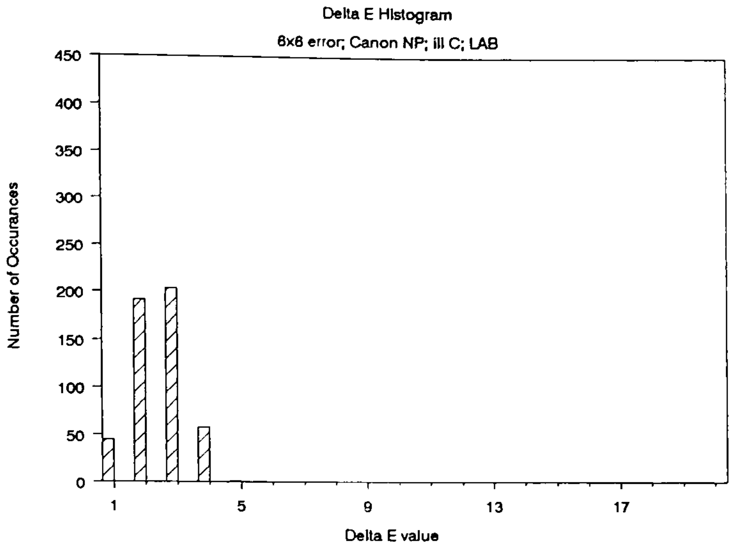


Figure 170

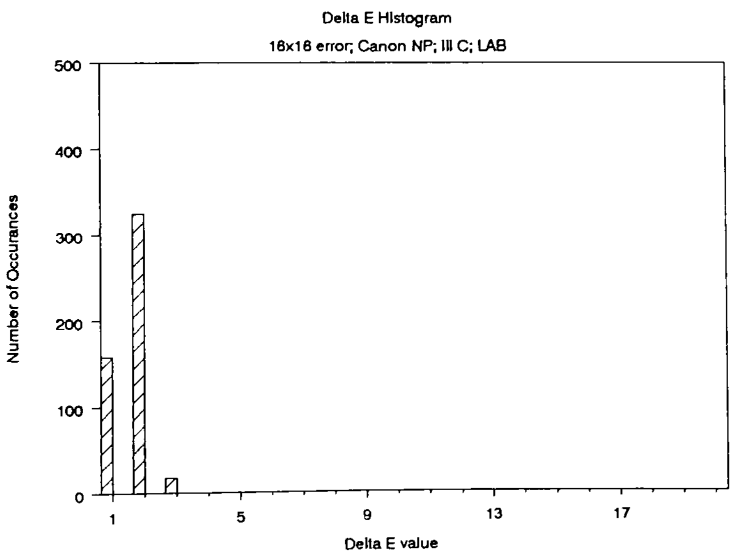


Figure 171

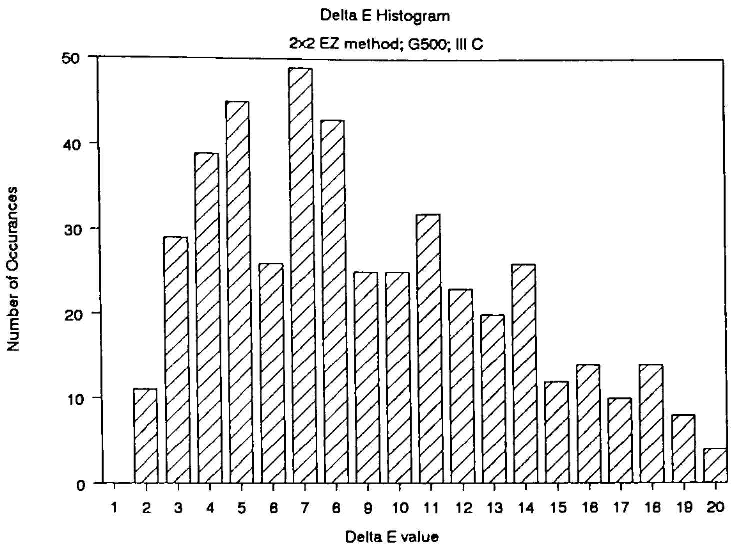


Figure 172

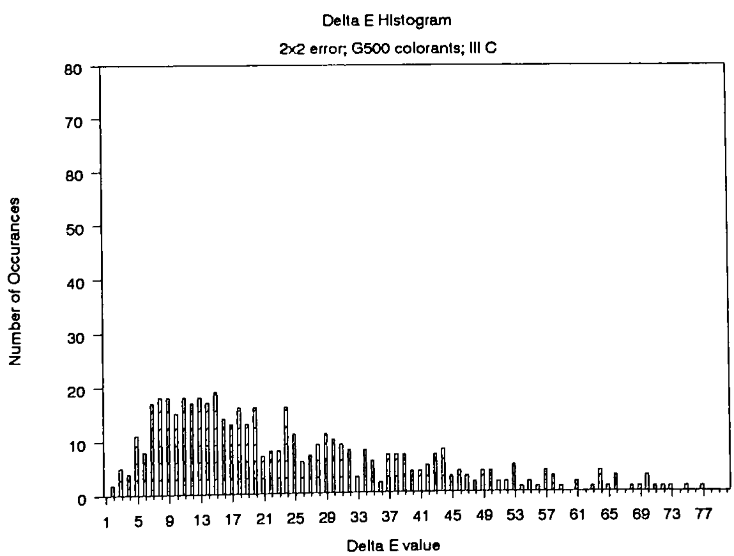


Figure 173

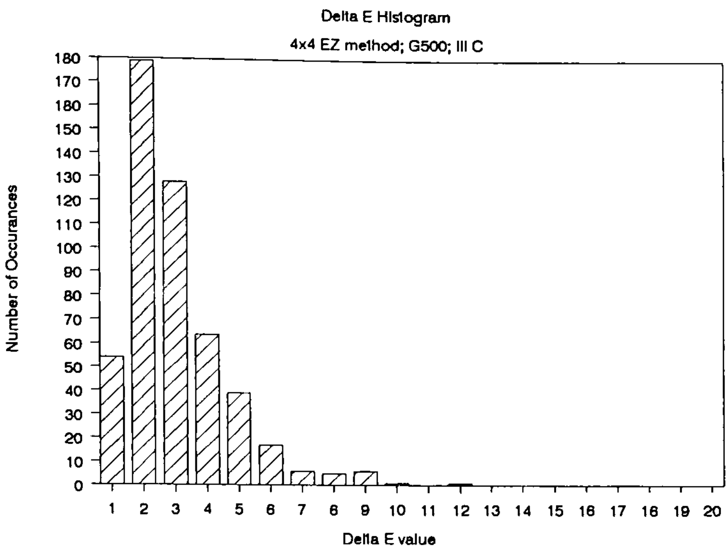


Figure 174

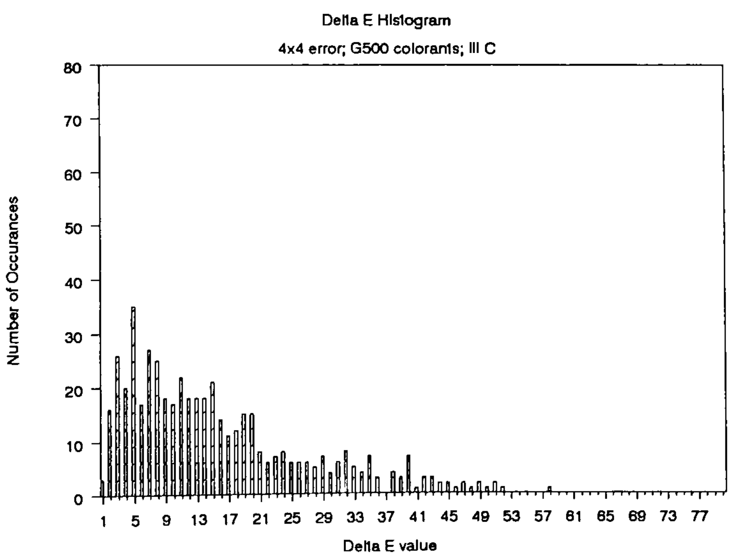


Figure 175

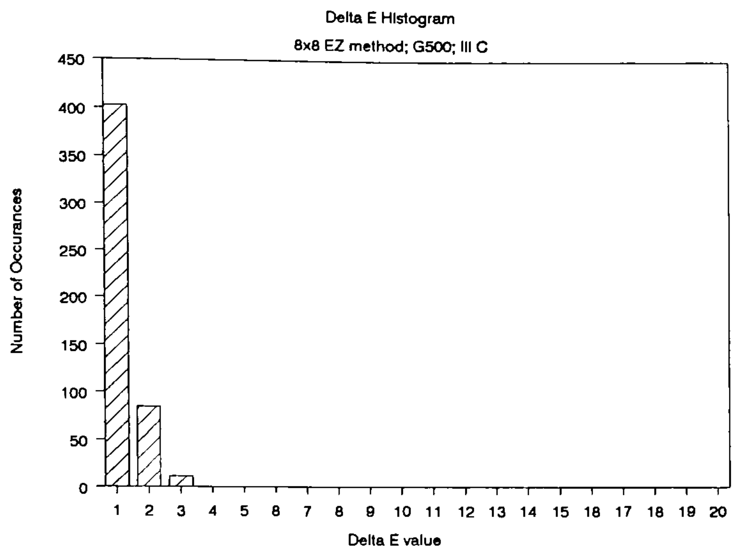


Figure 176

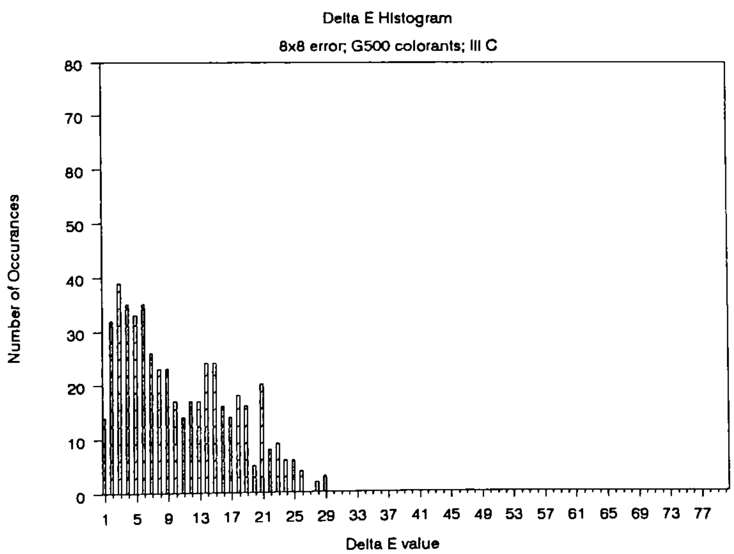


Figure 177

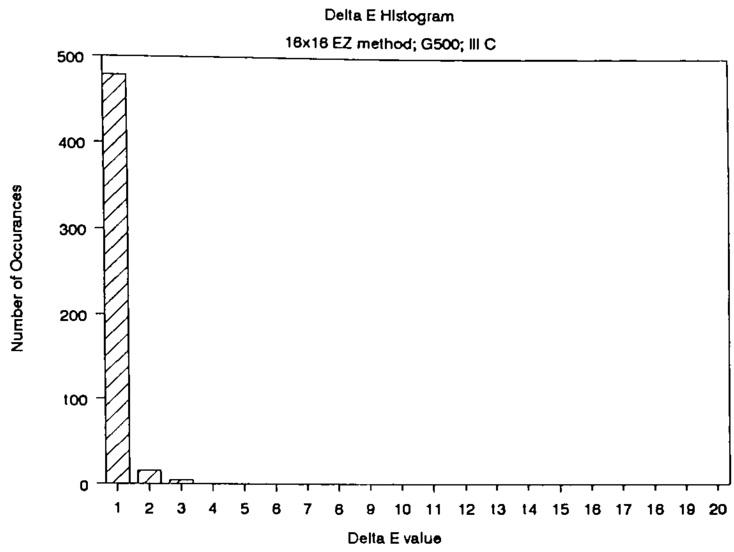


Figure 178

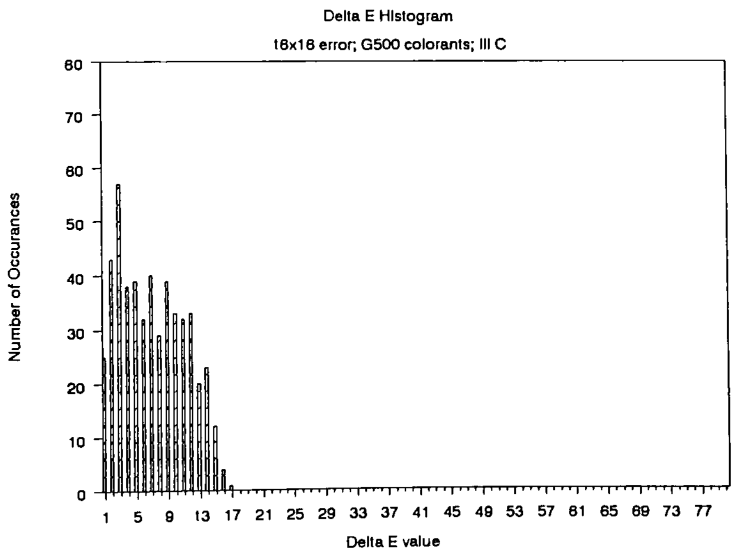


Figure 179

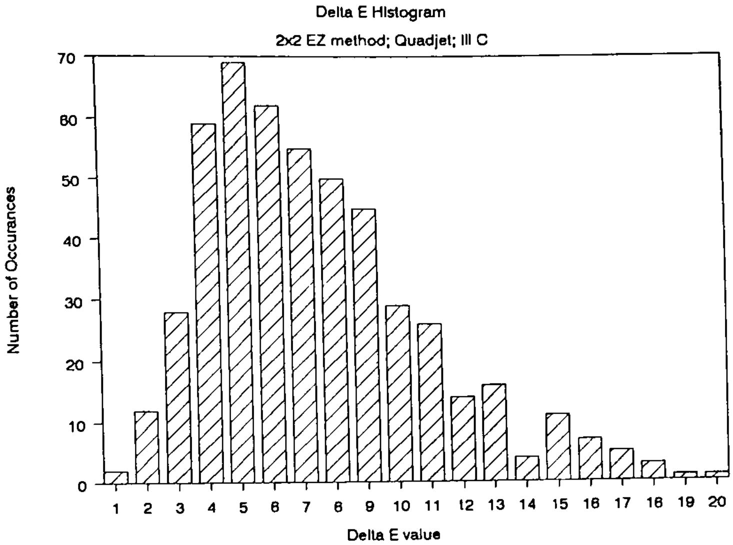


Figure 180

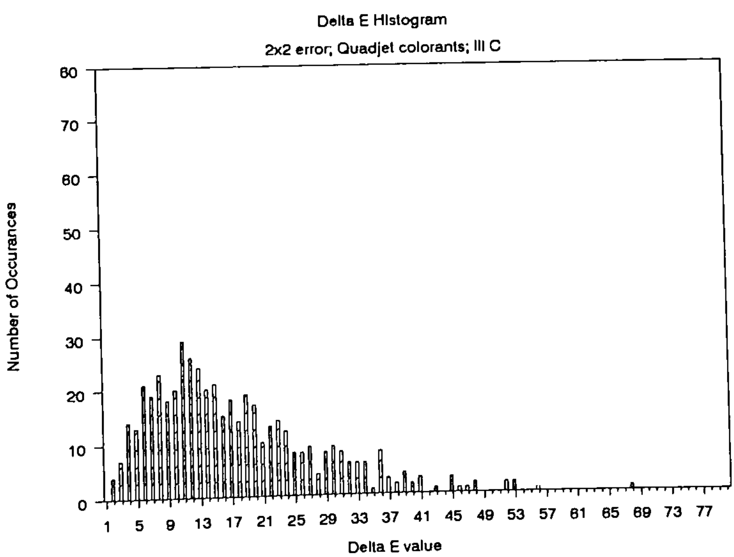


Figure 181

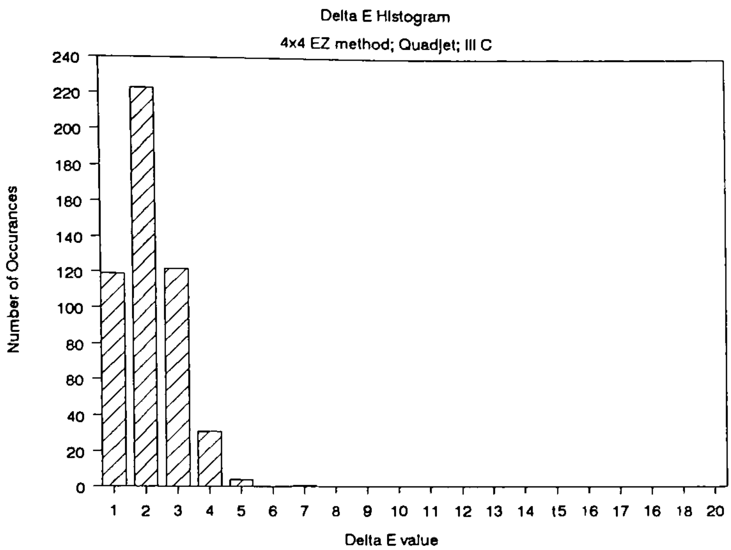


Figure 182

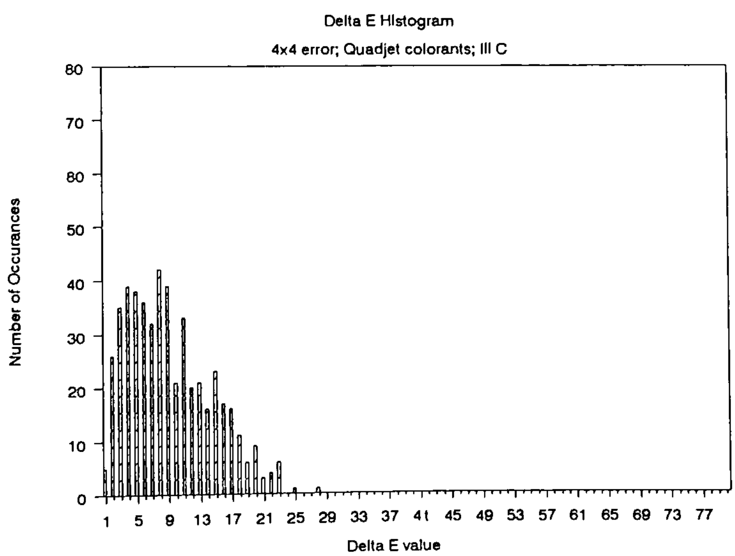


Figure 183

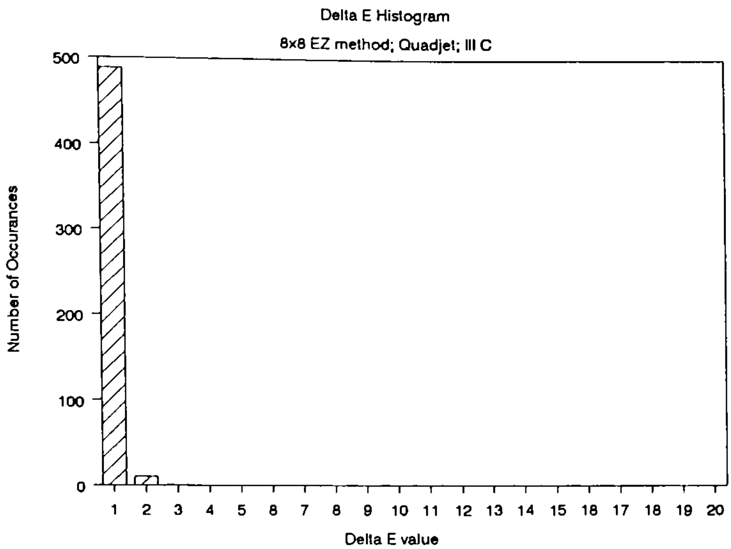


Figure 184

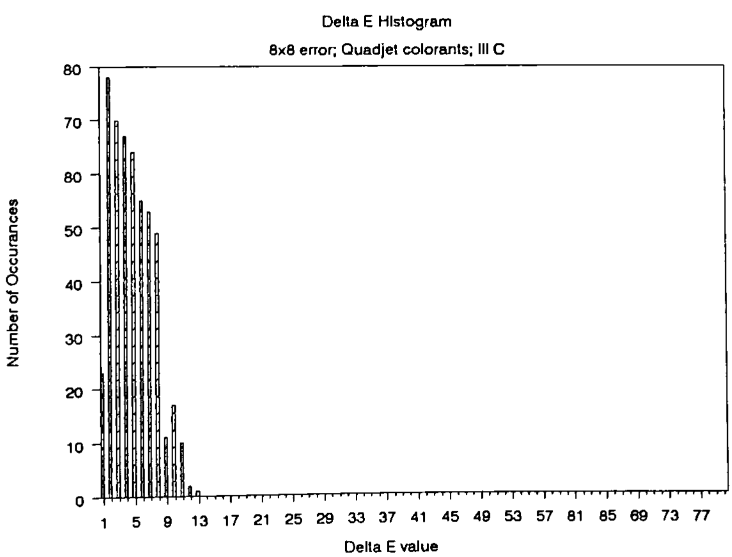


Figure 185

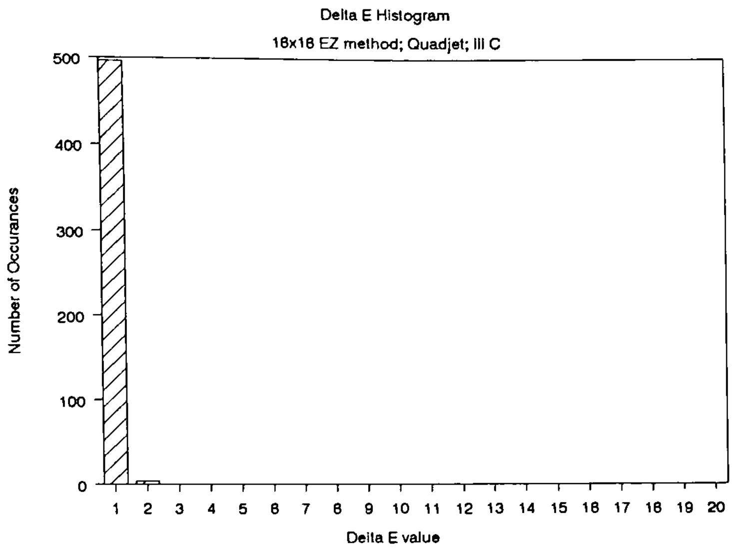


Figure 186

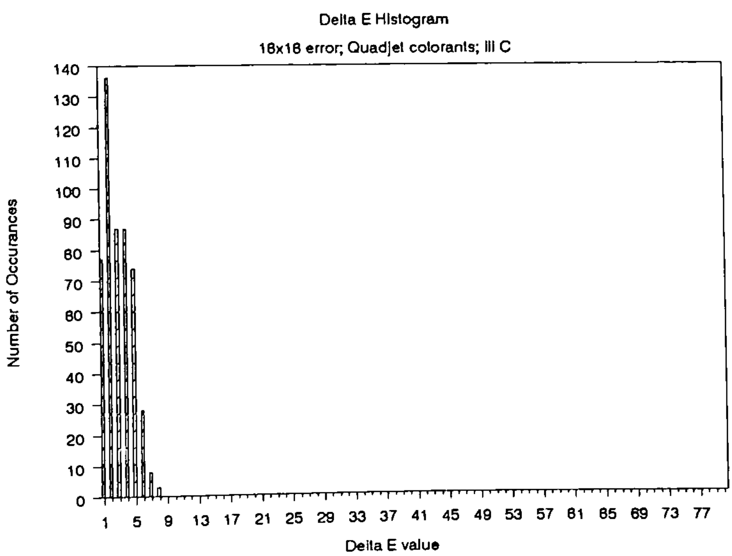


Figure 187

SOURCE CODE LISTINGS

```
'       ***************************************************
'       Random number generator of tristimulus values for color
'       patches   Peter A. Zuber  December 9,1987
'       ***************************************************
'       Source tristimulus values (Ill C)
'
xl=98.071 : yl=100 : zl=118.225 : sum=xl+yl+zl
'
'       Chromaticity coordinates of illuminant
'
xlc=xl/sum : ylc=yl/sum
flagclrnt=1
'
'       Opening page and remarks
'
cls
print "                                                                "
print "This routine  generates an  original  tristimulus image  file from"
print "a  random number  generator,  verifies that the  created  color is"
print "reproducible within a 6 colorant set if specified, and stores that"
print "color.  Any number of colors can be created and stored to one file"
print "Currently set up for illuminant C
print "                                                                "
print
'
'       Prompt user for filenames to be used
'
1000 print "Do you want the color file checked against a colorant set (Y/N)? "
input " ---> ",answer$
if answer$ = "Y" then 1050 else if answer$ = "y" then 1050
if answer$ = "N" then 1100 else if answer$ = "n" then 1100
goto 1000
1050 print "Enter the colorant set filename (full name)" : flagclrnt=0
input " ---> ",colrntfn$
1100 print "Enter the filename for the original tristimulus values (full name)"
input " ---> ",origfn$
'
'       Returning defined filenames to user
'
print " Your file will be called "origfn$ : print
'
'       Prompt user for file sizes
'
print " Enter the number of original tristimulus colors to create"
input " ---> ",tota
'
'       Begin program execution; dimension file arrays
'
dim triorig(tota,3),colrnt(9,1,3),a(4),area(4)
dim ang(9),trial(3)
'
'       Loading reproduction colorants;
'       Determine chromaticity hue angle of colorants
'
if flagclrnt=1 then 2050
cls : locate 12,30,0 : print "Loading colorants..."
open "i",#1,colrntfn$
for k=1 to 9
    j=1
    sum0=0
```

```
          for i=1 to 3
              input#1,colrnt(k,j,i)
              sum0=sumo+colrnt(k,j,i)
          next i
          if k=9 then goto 2020
          tx=(colrnt(k,j,1)/sum0)-xlc
          ty=(colrnt(k,j,2)/sum0)-ylc
          if k=1 then goto 2010
          if tx < 0 then goto 1990
          if ty < 0 then goto 2000
          ang(k)=atn(ty/tx) : goto 2020
1990      ang(k)=(atn(ty/tx)+3.1415927#) : goto 2020
2000      ang(k)=(atn(ty/tx)+6.2831853#) : goto 2020
2010      ang(1)=atn(ty/tx)
2020 next k
     '
     '     Rotate angles so red-illuminant angle = 0
     '
     rot=-ang(1)
     for k=1 to 8
         ang(k)=ang(k)+rot
     next k
     close#1
     '
     '      Generating original color patch from original tristimulus file
     '
2050 colorgood = 0 : rancnt = 1
     cls : locate 12,26,0 : print "Processing original colors..."
     locate 18,28,0 : print "Created color number "
     while colorgood < tota
         randomize (rancnt)
             for i=1 to 3
                 trial(i) = rnd
             next i
             trial(1)=trial(1)*xl
             trial(2)=trial(2)*yl
             trial(3)=trial(3)*zl
             if flagclrnt=1 then 3090
             sum1=trial(1)+trial(2)+trial(3)
     '
     '      Begin processing; determine x,y from X,Y,Z of each pixel
     '
             capx=trial(1) : capy=trial(2) : capz=trial(3)
             smallx=capx/sum1 : smally=capy/sum1
     '
     '      Shift chromaticity coordinates to place original colors around
     '      illuminant centered at 0,0 for hue angle determination
     '
             xcor=smallx-xlc : ycor=smally-ylc
     '
     '      Determine original chromaticity hue angle
     '
             if xcor < 0 then goto 2510
             if ycor < 0 then goto 2520
             theta=atn(ycor/xcor) : goto 2530
2510         theta=(atn(ycor/xcor)+3.1415927#) : goto 2530
2520         theta=(atn(ycor/xcor)+6.2831853#)
     '
     '      Rotate angle with respect to red
     '
```

```
2530      theta=theta+rot
     '
     '      Determine 2 primaries used for reproduction by comparing hue
     '      angle values from original and colorant set
     '
     if theta <= ang(1) then goto 2670 else if theta <= ang(6) then goto 2620
     if theta <= ang(2) then goto 2630 else if theta <= ang(4) then goto 2640
     if theta <= ang(3) then goto 2650 else if theta <= ang(5) then goto 2660
          goto 2670
2620 q=6 : r=1 : goto 2710 : '  (toggle yellow  and red   colorants)
2630 q=6 : r=2 : goto 2710 : '  (toggle yellow  and green colorants)
2640 q=4 : r=2 : goto 2710 : '  (toggle cyan    and green colorants)
2650 q=4 : r=3 : goto 2710 : '  (toggle cyan    and blue  colorants)
2660 q=5 : r=3 : goto 2710 : '  (toggle magenta and blue  colorants)
2670 q=5 : r=1 : goto 2710 : '  (toggle magenta and red   colorants)
     '
     '      Tristimulus values of colorant primaries,(minus black tristimulus)
     '
2710 x1=colrnt(Q,1,1):y1=colrnt(Q,1,2):z1=colrnt(Q,1,3):'Subtractive colorant
     x2=colrnt(R,1,1):y2=colrnt(R,1,2):z2=colrnt(R,1,3):'Additive    colorant
     x3=colrnt(7,1,1):y3=colrnt(7,1,2):z3=colrnt(7,1,3):'White       colorant
     x4=colrnt(8,1,1):y4=colrnt(8,1,2):z4=colrnt(8,1,3):'Black       colorant
     x1b=x1-x4 : y1b=y1-y4 : z1b=z1-z4 : x2b=x2-x4 : y2b=y2-y4 : z2b=z2-z4
     x3b=x3-x4 : y3b=y3-y4 : z3b=z3-z4
     capxm=capx-x4 : capym=capy-y4 : capzm=capz-z4
     '
     '      Solving for percent area of each colorant
     '       area(1) = fraction of subtractive colorant, C,M or Y
     '       area(2) = fraction of additive colorant, R, G, or B
     '       area(3) = fraction of white
     '       area(4) = fraction of black
     '
     det1=x1b*y2b*z3b+x2b*y3b*z1b+x3b*y1b*z2b
     det2=(x3b*y2b*z1b+x2b*y1b*z3b+x1b*y3b*z2b)
     det=det1-det2
     area(1)=capxM*y2b*z3b+x2b*y3b*capzM+x3b*capyM*z2b
     area(1)=(area(1)-(x2b*capyM*z3b+capxM*y3b*z2b+x3b*y2b*capzM))/det
     area(2)=x1b*capyM*z3b+capxM*y3b*z1b+x3b*y1b*capzM
     area(2)=(area(2)-(z1b*capyM*x3b+capzM*y3b*x1b+z3b*y1b*capxM))/det
     area(3)=x1b*y2b*capzM+x2b*capyM*z1b+capxM*y1b*z2b
     area(3)=(area(3)-(z1b*y2b*capxM+z2b*capyM*x1b+capzM*y1b*x2b))/det
     area(4)=1-area(1)-area(2)-area(3)
          a(4)=0
 if q=6 and r=1 then a(3)=area(1)+area(2)+area(4)
                     a(1)=area(4):a(2)=area(2)+area(4)
 if q=6 and r=2 then a(3)=area(1)+area(2)+area(4)
                     a(1)=area(2)+area(4):a(2)=area(4)
 if q=4 and r=2 then a(1)=area(1)+area(2)+area(4)
                     a(3)=area(2)+area(4):a(2)=area(4)
 if q=4 and r=3 then a(1)=area(1)+area(2)+area(4)
                     a(3)=area(4):a(2)=area(4)+area(2)
 if q=5 and r=3 then a(2)=area(1)+area(2)+area(4)
                     a(1)=area(2)+area(4):a(3)=area(4)
 if q=5 and r=1 then a(3)=area(2)+area(4)
                     a(2)=area(1)+area(2)+area(4):a(1)=area(4)
     '
     '      Writing percent area coverages to array
     '      of C,M,Y,Blk percentages respectively
     '      Check for rounding errors, Then set area to 0 or 1
     '
```

```
              for i=1 to 3
                  if a(I)>-.0001 and a(I)<.0001 then a(I)=0
                  if a(I)<1.0001 and a(I)>.9999 then a(I)=1
                  if a(I) > 1 or a(I) < 0 then 3100 else 3080
3080          next i
3090          colorgood = colorgood + 1
              for i=1 to 3 : triorig(colorgood,i)=trial(i) : next i
              locate 18,49,0 : print colorgood
3100 rndcnt = rndcnt + 1
      wend
      '
      '  Write data to file.
      '
      open "O",#1,origfn$
           for m=1 to tota
               for l=1 to 3
                   print#1,triorig(m,l)
           next l : next m
      close#1
```

```
1000 ' **********************************************************
1010 ' ROUTINE TO GENERATE COLOR REPRODUCTION ARRAY ACCORDING TO
1020 ' OPTIMUM DITHER (EZ METHOD). JULY 11, 1985  PETER A. ZUBER
1030 ' MODIFIED May 8, 1987 PETER G. ENGELDRUM - IMCOTEK, INC.
1040 ' PROGRAM NAME:EZ.BAS
1050 '
1060 ' **********************************************************
1070 '
1080 CLS
1090 '              Source tristimulus values
1100 XL=98.071:ZL=118.225:SUM=100+XL+ZL
1110 '               Chromaticity coordinates of illuminant.
1120 XLC=XL/SUM:YLC=100/SUM
1130 '
1140 '        Opening page and remarks
1150 '
1160 PRINT "                                                               "
1170 PRINT " This routine generates an original image from file input, then "
1180 PRINT " produces a tristimulus valued reproduction and bit-maps using  "
1190 PRINT " a color dither scheme                                          "
1200 PRINT "                                                               "
1210 PRINT
1220 '
1230 '        Prompt user for filenames to be used
1240 '
1250 PRINT " Enter the colorant set filename (no extensions)"
1260 INPUT " ---> ",C$
1270 PRINT " Enter the original tri. filename (no extensions)"
1280 INPUT " ---> ",T$
1290 PRINT " Enter the dither matrix filename (no extensions)"
1300 INPUT " ---> ",D$
1310 PRINT " Enter the filename for the reproduction (no extensions)"
1320 INPUT " ---> ",R$
1330 '
1340 '        Defining filenames
1350 '
1360 EL$=".LSR":EH$=".HUE":ES$=".SAT"
1365 ESTR$=".STR":EE$=".DEE":EC$=".COL":ET$=".TRI":ED$=".MAT"
1370 FL$=R$+EL$:FH$=R$+EH$:FS$=R$+ES$:FSTR$=R$+ESTR$:FE$=R$+EE$
1375 FC$=C$+EC$:FT$=T$+ET$:FD$=D$+ED$
1380 '
1390 '        Returning defined filenames to user
1400 '
1410 PRINT " One reproduction will have filename "FL$:PRINT
1420 '
1430 '        Prompt user for file sizes
1440 '
1450 PRINT " Enter the X,Y dimensions of the tristimulus file"
1460 INPUT " ---> ",X,Y
1470 PRINT " Enter the X,Y dimensions of the dither matrix"
1480 INPUT " ---> ",XD,YD:CLS
1490 PRINT " Do you wish the color patch displayed to the screen? (Y/N)"
1500 PRINT " (Displaying the patch requires user input before the next "
1510 PRINT "  color is evaluated.  In either case, all data is stored
1520 PRINT "  to "F$")":COLOR 4,0
1530 PRINT " REQUIRES A COLOR MONITOR (of course) ":COLOR 7,0,1
1540 INPUT " ---> ",ANS$
1550 IF ANS$ = "Y" THEN TEST=1 ELSE IF ANS$ = "y" THEN TEST=1 ELSE TEST=0
1560 '
1570 '        Begin program execution; dimension file arrays
```

```
1580 '
1590 TOTA=X*Y
1600 CLS:LOCATE 12,25,0:PRINT "Loading tristimulus values..."
1610 DIM TORIG(X,Y,3),COLRNT(9,1,3),IMAG(XD,YD,3),A(4),AREA(4)
1620 DIM PERCEN(XD,YD,4),DITHER(XD,YD),FIL(X,Y,23),LABEL$(4)
1630 DIM REPRO(XD,YD,4),RETRI(XD,YD,3),REP(3),ANG(9),SUM(8)
1640 '    Offsets to index colorants from cmyk bit plane number
1650 DIM JDEXF(8)
1660 JDEXF(0)=7:JDEXF(1)=3:JDEXF(2)=3:JDEXF(3)=0:JDEXF(4)=2
1665 JDEXF(5)=-3:JDEXF(6)=-5:JDEXF(8)=0
1670 '
1680 '       Loading original tristimulus values
1690 '
1700 OPEN "I",#1,FT$
1720 FOR J=1 TO Y
1730     FOR K=1 TO X
1750             FOR I=1 TO 3
1760             INPUT#1,TORIG(K,J,I)
1770 NEXT I:NEXT K:NEXT J
1780 CLOSE#1
1790 '
1800 '       Loading reproduction colorants;
1810 '       Determine CHROMATICITY hue angle of colorants
1820 '
1830 CLS:LOCATE 12,30,0:PRINT "Loading colorants..."
1840 OPEN "I",#1,FC$
1850 FOR K=1 TO 9
1860     J=1
1870       SUM0=0
1880         FOR I=1 TO 3
1890             INPUT#1,COLRNT(K,J,I)
1900             SUM0=SUM0+COLRNT(K,J,I)
1910         NEXT I
1920 IF K=9 THEN GOTO 2020
1930         TX=(COLRNT(K,J,1)/SUM0)-XLC
1940         TY=(COLRNT(K,J,2)/SUM0)-YLC
1950 IF K=1 THEN GOTO 2010
1960         IF TX < 0 THEN GOTO 1990
1970         IF TY < 0 THEN GOTO 2000
1980         ANG(K)=ATN(TY/TX)  : GOTO 2020
1990         ANG(K)=(ATN(TY/TX)+3.1415927#)  : GOTO 2020
2000         ANG(K)=(ATN(TY/TX)+6.2831853#)  : GOTO 2020
2010         ANG(1)=ATN(TY/TX)
2020 NEXT K
2030 '   Rotate angles so red-illuminant angle = 0.
2040 ROT=-ANG(1)
2050 FOR K=1 TO 8:ANG(K)=ANG(K)+ROT:NEXT K
2060 CLOSE#1
2070 '
2080 '       Loading dither matrix
2090 '
2100 CLS:LOCATE 12,28,0:PRINT "Loading dither matrix..."
2110 OPEN "I",#1,FD$
2120         FOR J=1 TO YD
2130             FOR K=1 TO XD
2140                 INPUT#1,DITHER(K,J)
2150             NEXT K
2160         NEXT J
2170 CLOSE#1
2180 '
```

```
2190 '        Generating original color patch from original tristimulus file
2200 '
2210 FOR M=1 TO Y
2220     FOR L=1 TO X
2230         SUM1=0
2240         FOR I=1 TO 3
2250             TRI=TORIG(L,M,I)
2260             SUM1=SUM1+TRI
2270             FOR J=1 TO YD
2280                 FOR K=1 TO XD
2290                     IMAG(K,J,I)=TRI
2300         NEXT K:NEXT J:NEXT I
2310 '
2320 '        Begin processing; determine x,y from X,Y,Z of each pixel
2330 '
2340 CLS:LOCATE 12,26,0:PRINT "Processing original image..."
2350 '
2360         FOR J=1 TO YD
2370             FOR K=1 TO XD
2380                 CAPX=IMAG(K,J,1) : CAPY=IMAG(K,J,2) : CAPZ=IMAG(K,J,3)
2390                 SMALLX=CAPX/SUM1 : SMALLY=CAPY/SUM1
2400 '
2410 '        Shift chromaticity coordinates to place original colors around
2420 '        illuminant centered at 0,0 for hue angle determination
2430 '
2440                 XCOR=SMALLX-XLC : YCOR=SMALLY-YLC
2450 '
2460 '        Determine original CHROMATICITY hue angle
2470 '
2480                 IF XCOR < 0 THEN GOTO 2510
2490                 IF YCOR < 0 THEN GOTO 2520
2500                 THETA=ATN(YCOR/XCOR) : GOTO 2530
2510                 THETA=(ATN(YCOR/XCOR)+3.1415927#) : GOTO 2530
2520                 THETA=(ATN(YCOR/XCOR)+6.2831853#)
2530                 THETA=THETA+ROT:'  Rotate angle WRT red
2540 '
2550 '        Determine 2 primaries used for reproduction by comparing hue
2560 '        angle values from original and colorant set
2570 '
2580 IF THETA <= ANG(1) THEN GOTO 2670 ELSE IF THETA <= ANG(6) THEN GOTO 2620
2590 IF THETA <= ANG(2) THEN GOTO 2630 ELSE IF THETA <= ANG(4) THEN GOTO 2640
2600 IF THETA <= ANG(3) THEN GOTO 2650 ELSE IF THETA <= ANG(5) THEN GOTO 2660
2610 GOTO 2670
2620 Q=6:R=1:GOTO 2710:'  (toggle yellow  and red   colorants)
2630 Q=6:R=2:GOTO 2710:'  (toggle yellow  and green colorants)
2640 Q=4:R=2:GOTO 2710:'  (toggle cyan    and green colorants)
2650 Q=4:R=3:GOTO 2710:'  (toggle cyan    and blue  colorants)
2660 Q=5:R=3:GOTO 2710:'  (toggle magenta and blue  colorants)
2670 Q=5:R=1:GOTO 2710:'  (toggle magenta and red   colorants)
2680 '
2690 '        Tristimulus values of colorant primaries,(minus black tristimulus)
2700 '
2710 X1=COLRNT(Q,1,1):Y1=COLRNT(Q,1,2):Z1=COLRNT(Q,1,3):'Subtractive colorant
2720 X2=COLRNT(R,1,1):Y2=COLRNT(R,1,2):Z2=COLRNT(R,1,3):'Additive    colorant
2730 X3=COLRNT(7,1,1):Y3=COLRNT(7,1,2):Z3=COLRNT(7,1,3):'White       colorant
2740 X4=COLRNT(8,1,1):Y4=COLRNT(8,1,2):Z4=COLRNT(8,1,3):'Black       colorant
2750 X1B=X1-X4:Y1B=Y1-Y4:Z1B=Z1-Z4:X2B=X2-X4:Y2B=Y2-Y4:Z2B=Z2-Z4
2760 X3B=X3-X4:Y3B=Y3-Y4:Z3B=Z3-Z4:CAPXM=CAPX-X4:CAPYM=CAPY-Y4:CAPZM=CAPZ-Z4
2770 '
2780 '        Solving for percent area of each colorant
```

```
2790 '    AREA(1) = fraction of subtractive colorant, C,M or Y
2800 '    AREA(2) = fraction of additive colorant, R, G, or B
2810 '    AREA(3) = fraction of white
2820 '    AREA(4) = fraction of black
2830 '
2840 DET1=X1B*Y2B*Z3B+X2B*Y3B*Z1B+X3B*Y1B*Z2B
2845 DET2=(X3B*Y2B*Z1B+X2B*Y1B*Z3B+X1B*Y3B*Z2B)
2847 DET=DET1=DET2
2850 AREA(1)=CAPXM*Y2B*Z3B+X2B*Y3B*CAPZM+X3B*CAPYM*Z2B
2860 AREA(1)=(AREA(1)-(X2B*CAPYM*Z3B+CAPXM*Y3B*Z2B+X3B*Y2B*CAPZM))/DET
2870 AREA(2)=X1B*CAPYM*Z3B+CAPXM*Y3B*Z1B+X3B*Y1B*CAPZM
2880 AREA(2)=(AREA(2)-(Z1B*CAPYM*X3B+CAPZM*Y3B*X1B+Z3B*Y1B*CAPXM))/DET
2890 AREA(3)=X1B*Y2B*CAPZM+X2B*CAPYM*Z1B+CAPXM*Y1B*Z2B
2900 AREA(3)=(AREA(3)-(Z1B*Y2B*CAPXM+Z2B*CAPYM*X1B+CAPZM*Y1B*X2B))/DET
2910 AREA(4)=1-AREA(1)-AREA(2)-AREA(3)
2920 A(4)=0
2930 IF Q=6 AND R=1 THEN A(3)=AREA(1)+AREA(2)+AREA(4)
                         A(1)=AREA(4):A(2)=AREA(2)+AREA(4)
2940 IF Q=6 AND R=2 THEN A(3)=AREA(1)+AREA(2)+AREA(4)
                         A(1)=AREA(2)+AREA(4):A(2)=AREA(4)
2950 IF Q=4 AND R=2 THEN A(1)=AREA(1)+AREA(2)+AREA(4)
                         A(3)=AREA(2)+AREA(4):A(2)=AREA(4)
2960 IF Q=4 AND R=3 THEN A(1)=AREA(1)+AREA(2)+AREA(4)
                         A(3)=AREA(4):A(2)=AREA(4)+AREA(2)
2970 IF Q=5 AND R=3 THEN A(2)=AREA(1)+AREA(2)+AREA(4)
                         A(1)=AREA(2)+AREA(4):A(3)=AREA(4)
2980 IF Q=5 AND R=1 THEN A(3)=AREA(2)+AREA(4)
                         A(2)=AREA(1)+AREA(2)+AREA(4):A(1)=AREA(4)
2990 '
3000 '       Writing percent area coverages to array
3010 '       of C,M,Y,Blk percentages respectively
3020 '
3030                    FOR I=1 TO 3
3040 '  Check to see if rounding errors.  Then set area to 0 or 1.
3050                IF A(I)>-.0001 AND A(I)<.0001 THEN A(I)=0
3060                IF A(I)<1.0001 AND A(I)>.9999 THEN A(I)=1
3070    IF A(I) > 1 OR A(I) < 0 THEN PRINT "Color not reproducable":GOTO 4100
3080                        PERCEN(K,J,I)=A(I)
3090                    NEXT I
3100 '
3110 '       Return to consider next row of image
3120 '
3130                NEXT K
3140            NEXT J
3150 '
3160 '       Pass Dither matrix over percent area coverage array;
3170 '       Filter out process black pixels and convert to real black
3180 '
3190 CLS:LOCATE 12,22,0:PRINT "Passing dither matrix over image..."
3200 '
3210      FOR J=1 TO YD
3220          FOR K=1 TO XD
3230          SUM2=0
3240              FOR I=1 TO 3
3250 PERCEN(K,J,I) > DITHER(K,J) THEN REPRO(K,J,I) = 1 ELSE REPRO(K,J,I) = 0
3260              SUM2=REPRO(K,J,I)+SUM2
3270              NEXT I
3280              IF SUM2=3 THEN GOTO 3290 ELSE REPRO(K,J,4)=0:GOTO 3300
3290              REPRO(K,J,1)=0:REPRO(K,J,2)=0:REPRO(K,J,3)=0:REPRO(K,J,4)=1
3300          NEXT K
```

```
3310     NEXT J
3320 '
3330 '       Print bit maps to display.
3340 '
3350 IF TEST=0 THEN GOTO 3470
3360 GOSUB 4470
3370 '
3380 '       Print color image to color display.
3390 '
3400 GOSUB 4750
3410 '
3420 '       Convert bit-map file to tristimulus valued file;
3430 '       Write tristimulus valued file to disk
3440 '
3450 '       Calculate index from bit map with cyan the LSB; 4 bits
3460 '
3470      FOR I=1 TO 8:SUM(I)=0:NEXT I
3480      FOR J=1 TO YD
3490          FOR K=1 TO XD
3500          INDEX=REPRO(K,J,1)+2*REPRO(K,J,2)+4*REPRO(K,J,3)+8*REPRO(K,J,4)
3510 '     Use INDEX as index to offsets for colorants. NEWDX = index into
3520 '          colorant array stored R,G,B,C,M,Y,W,K.
3530          NEWDX=INDEX+JDEXF(INDEX)
3540 '     Sum stores the number to times a colorant is used in the HT cell.
3550          SUM(NEWDX)=SUM(NEWDX)+1
3560          NEXT K
3570      NEXT J
3580 '
3590 '       Determining Color Differences
3600 '
3610 CLS:LOCATE 12,24,0:PRINT "Determining Color Differences..."
3620 '
3630 XOD=TORIG(L,M,1)/XL:YOD=TORIG(L,M,2)/100
3640 ZOD=TORIG(L,M,3)/ZL
3650 OLSTAR=116*(YOD^.33333)-16:OASTAR=500*(XOD^.33333-YOD^.33333)
3660 OBSTAR=200*(YOD^.33333-ZOD^.33333):OCSTAR=SQR((OASTAR^2)+(OBSTAR^2))
3680 FIL(L,M,1)=OLSTAR:FIL(L,M,2)=OASTAR:FIL(L,M,3)=OBSTAR
3700 '
3710 '       Determine original a*, b* hue angle
3720 '
3730          IF OASTAR < 0 THEN GOTO 3760
3740          IF OBSTAR < 0 THEN GOTO 3770
3750          THETA=ATN(OBSTAR/OASTAR) : GOTO 3780
3760          THETA=(ATN(OBSTAR/OASTAR)+3.1415927#) : GOTO 3780
3770          THETA=(ATN(OBSTAR/OASTAR)+6.2831853#)
3780 FIL(L,M,4)=THETA
3790 REP(1)=0:REP(2)=0:REP(3)=0
3800 '  REP() is X,Y,Z of the reproduction, respectively.
3810 '  Determine tristimulus values of reproduction.
3820      FOR I=1 TO 8
3830      REP(1)=REP(1)+SUM(I)*COLRNT(I,1,1)
3840      REP(2)=REP(2)+SUM(I)*COLRNT(I,1,2)
3850      REP(3)=REP(3)+SUM(I)*COLRNT(I,1,3)
3860      NEXT I
3870 '  Scale tristimulus values to cell size.
3880      FOR I=1 TO 3:REP(I)=REP(I)/(XD*YD):NEXT I
3890 '  compute CIEL*a*b* values of reproduction.
3900 XRD=REP(1)/XL:YRD=REP(2)/100:ZRD=REP(3)/ZL
3910 RLSTAR=116*(YRD^.33333)-16:RASTAR=500*(XRD^.33333-YRD^.33333)
3920 RBSTAR=200*(YRD^.33333-ZRD^.33333):RCSTAR=SQR((RASTAR^2)+(RBSTAR^2))
```

```
3930 DELTAL=OLSTAR-RLSTAR:DELTAA=OASTAR-RASTAR:DELTAB=OBSTAR-RBSTAR
3940 DELTAC=OCSTAR-RCSTAR
3950 DELTAE=SQR((DELTAL^2)+(DELTAA^2)+(DELTAB^2))
3960 RATIO=RCSTAR/OCSTAR
3970 '
3980 '       Determine reproduction a*, b* hue angle
3990 '
4000          IF RASTAR < 0 THEN GOTO 4030
4010          IF RBSTAR < 0 THEN GOTO 4040
4020          RTHETA=ATN(RBSTAR/RASTAR) : GOTO 4060
4030          RTHETA=(ATN(RBSTAR/RASTAR)+3.1415927#) : GOTO 4060
4040          RTHETA=(ATN(RBSTAR/RASTAR)+6.2831853#)
4060 FIL(L,M,5)=RLSTAR:FIL(L,M,6)=RASTAR:FIL(L,M,7)=RBSTAR
4070 FIL(L,M,8)=RTHETA:FIL(L,M,9)=DELTAE:FIL(L,M,10)=RATIO
4100     NEXT L
4110 NEXT M
4120 '
4130 '  Fill string array with names of the data created;
4140 '  Load into output file for columnizing in Lotus
4150 '
4240 '
4250 '  Write data to file.
4260 '
4270 OPEN "O",#1,FL$
     FOR M=1 TO Y
     FOR L=1 TO X
     PRINT#1,FIL(L,M,1),FIL(L,M,5)
     NEXT L:NEXT M
     CLOSE#1
     OPEN "O",#1,FH$
     FOR M=1 TO Y
     FOR L=1 TO X
     PRINT#1,FIL(L,M,4),FIL(L,M,8)
     NEXT L:NEXT M
     CLOSE#1
     OPEN "O",#1,FS$
     FOR M=1 TO Y
     FOR L=1 TO X
     PRINT#1,FIL(L,M,10)
     NEXT L:NEXT M
     CLOSE#1
     OPEN "O",#1,FSTR$
     FOR M=1 TO Y
     FOR L=1 TO X
     PRINT#1,FIL(L,M,2),FIL(L,M,3),FIL(L,M,6)FIL(L,M,7)
     NEXT L:NEXT M
     CLOSE#1
     OPEN "O",#1,FE$
     FOR M=1 TO Y
     FOR L=1 TO X
     PRINT#1,FIL(L,M,9)
     NEXT L:NEXT M
     CLOSE#1
     BEEP:BEEP:BEEP
4370 CLS:PRINT "PROGRAM IS OVER ..."
4380 '
4390 '       Close file and return to main menu
4400 '
4410 END
4420 '-----------------------------------------
```

```
4430 ' SUBROUTINE TO PRINT BIT MAPS TO SCREEN
4440 ' PETER G. ENGELDRUM - IMCOTEK, INC
4450 ' APRIL 29, 1987
4460 '-------------------------------------------
4470 ONE$=STR$(1):ZER$=STR$(0)
4480 LABEL$(1)="CYAN":LABEL$(2)="MAGENTA":LABEL$(3)="YELLOW":LABEL$(4)="BLACK"
4490 CLS
4500 FOR KK=1 TO 4
4510 JI=(KK-1)*(2*XD+2)+1
4520 LOCATE 1,JI+3,0
4530 PRINT LABEL$(KK)
4540 FOR J1=1 TO XD
4550 FOR J2=1 TO YD
4560 JX=J1+1:JY=JI+2*J2:'PRINT "JX=";JX,"JY=";JY
4570 IF REPRO(J1,J2,KK)=0 THEN GOTO 4610
4580 LOCATE JX,JY,0
4590 PRINT ONE$
4600 GOTO 4630
4610 LOCATE JX,JY,0
4620 PRINT ZER$
4630 NEXT J2
4640 NEXT J1
4650 NEXT KK
4660 RETURN
4670 '-------------------------------------------------------
4680 ' SUBROUTINE TO PRINT COLOR BIT PATTERN TO SCREEN.  A CGA OR
4690 ' AND EGA IS REQUIRED.  THE ROUTINE USES THE TEXT MODE TO PRINT
4700 ' THE HALFTONE CELL.
4710 ' PETER G. ENGELDRUM - IMCOTEK, INC.
4720 ' 5/8/87 - VERSION 1.0
4730 '-------------------------------------------------------
4740 '  Set a flag the first time the subr is called
4750 IF IFLAG =1 THEN 4830
4760 '  Data array are offsets to determine the IBM color display color
4770 '  from the bit map colors.
4780 DIM JDX(8)
4790 DATA 7,2,3,-2,10,-3,-2,0,-8
4800 FOR IS=0 TO 8:READ JDX(IS):NEXT IS
4810 IFLAG=1
4820 '  Space two lines
4830 PRINT:PRINT
4840 FOR JS=1 TO YD
4850    PRINT TAB(13);
4860    FOR KS=1 TO XD
4870    INDEX=REPRO(JS,KS,1)+2*REPRO(JS,KS,2)+4*REPRO(JS,KS,3)+8*REPRO(JS,KS,4)
4880    NDX=INDEX+JDX(INDEX)
4890 '           Set color according to NDX
4900 COLOR NDX,0,0
4910            FOR PX=1 TO 2
4920            PRINT CHR$(219);:NEXT PX
4930    NEXT KS:PRINT
4940 NEXT JS
4950 '  Set color back to gray on black background
4960 COLOR 7,0,1
4970 '  Get a carriage return to continue with program.
4980 LOCATE 23,1,1:INPUT"   <CR> to continue ";IA$
4990 RETURN
```

```
'
' *****************************************************
' Routine to generate a reproduction array according to a
' color error diffusion method.
' July 11, 1985
' Peter A. Zuber
' Revised for euclidean distance check January 2, 1988
' *****************************************************
'
cls
'
'       Opening page and remarks
'
print "                                                                    "
print " This  routine  retrieves a  file of  known  tristimulus values, "
print " creates  a  17x17  color  patch for  each  tristimulus  set, and "
print " using  a  selected  error  diffusion scheme and eight colorants, "
print " attempts to produce  a  colorimetrically accurate  reproduction. "
print " (Uses euclidean distance original to colorant to choose colorant "
print "                                                                    "
'
'       Dimension storage arrays for original, colorant and
'       reproduction; create original and load colorant values
'
print
print "Enter the complete filename of your colorant set"
input "---> ", c$
print
print "Enter the complete filename of your original tri. values"
input "---> ", t$
print
print "Enter output filename for this trial (no extension)"
input "---> ", d$
el$ = ".lsr": eh$ = ".hue": es$ = ".sat" : estr$ = ".str": ee$ = ".dee"
fl$ =  d$ + el$: fh$ = d$ + eh$
fs$ =  d$ + es$: fstr$ = d$ + estr$: fe$ = d$ + ee$
print " One reproduction will have filename "; fl$
print
print "Enter the X,Y dimensions of the tristimulus file"
print
print : input "---> ", x, y
cls
locate 12, 28, 0
print "Loading original colors..."
'
dim torig(x, y, 3), orig(17, 17, 3), repro(16, 16, 3)
dim colrnt(8, 1, 3), trierr(8, 1, 3), rep(3), fil(x, y, 28)
'
'       Loading original tristimulus values
'
open "i", #1, t$
for k = 1 to x
    for j = 1 to y
            for i = 1 to 3
            input #1, torig(k, j, i)
next i: next j: next k
close #1
'
'       Loading reproduction colorants
'
```

```
open "i", #1, c$
for k = 1 to 8
    j = 1
        for i = 1 to 3
            input #1, colrnt(k, j, i)
next i: next k
close #1
'
'     Generating 17x17 Master color patch
'
cls : locate 12, 26, 0: print "Creating original image..."
'
for m = 1 to y
    for l = 1 to x
        for i = 1 to 3
            tri = torig(l, m, i)
                for j = 1 to 17
                    for k = 1 to 17
                        orig(k, j, i) = tri
        next k: next j: next i
'
'       Begin processing; determine respective error from X,Y,Z
'       (X orig - X colorant, Y orig - Y colorant, etc.)
'
cls : locate 12, 26, 0: print "Processing original image..."
        for j = 1 to 16
            for k = 1 to 16
                for i = 1 to 3
                    for n = 1 to 8
      trierr(n, 1, i) = abs(orig(k, j, i) - colrnt(n, 1, i))
                    next n
                next i
'
'       Sum total X,Y,Z error from original pixel and
'       colorants, determine least error colorant
'
'
                litsum = 1.7d+38: sum = 0
                for o = 1 to 8
                    for n = 1 to 3
                        sum = (trierr(O, 1, n) ^ 2) + sum
                    next n
                    sum = sqr(sum)
                    if litsum >= sum then p = o
                    if litsum >= sum then litsum = sum
                    sum = 0
                next o
'
'       Write least error colorant to reproduction array,
'       return to consider next pixel
'
                for i = 1 to 3
                    repro(k, j, i) = colrnt(p, 1, i)
                    remain = (orig(k, j, i) - colrnt(P, 1, i))
 orig(k + 1, j, i) = orig(k + 1, j, i) + ((7 / 16) * remain)
 orig(k + 1, j + 1, i) = orig(k + 1, j + 1, i) + ((1 / 16) * remain)
 orig(k, j + 1, i) = orig(k, j + 1, i) + ((5 / 16) * remain)
                    if k = 1 then goto 2120
 orig(k - 1, j + 1, i) = orig(k - 1, j + 1, i) + ((3 / 16) * remain)
2120            next i
```

```
               next k
          next j
'
cls : locate 12, 24, 0: print "Determining Color Differences..."
'
xod = torig(l, m, 1) / 98.071: yod = torig(l, m, 2) / 100
zod = torig(l, m, 3) / 118.225
olstar = 116 * (yod ^ .333) - 16
oastar = 500 * (xod ^ .333 - yod ^ .333)
obstar = 200 * (yod ^ .333 - zod ^ .333)
ocstar = ((oastar ^ 2) + (obstar ^ 2)) ^ .5
'
'  Determine original a*, b* hue angle
'
          if oastar < 0 then goto 2280
          if obstar < 0 then goto 2290
          otheta = atn(obstar / oastar): goto 2310
2280      otheta = (atn(obstar / oastar) + 3.1415927#): goto 2310
2290      otheta = (atn(obstar / oastar) + 6.2831853#)
2310 fil(l, m, 1) = olstar: fil(l, m, 2) = oastar: fil(l, m, 3) = obstar
fil(l, m, 4) = otheta
cnt = 5: p = 2: cnt2 = 1
          while p < 17
               xl = p: yl = xl
                    for i = 1 to 3
                         sumit = 0
                              for j = 1 to yl
                                   for k = 1 to xl
                                        sumit = repro(k, j, i) + sumit
                              next k: next j
                         rep(i) = sumit / (xl * yl)
                    next i
xrd = rep(1) / 98.071: yrd = rep(2) / 100: zrd = rep(3) / 118.225
rlstar = 116 * (yrd ^ .333) - 16
rastar = 500 * (xrd ^ .333 - yrd ^ .333)
rbstar = 200 * (yrd ^ .333 - zrd ^ .333)
rcstar = ((rastar ^ 2) + (rbstar ^ 2)) ^ .5
deltal = olstar - rlstar
deltaa = oastar - rastar
deltab = obstar - rbstar
deltac = ocstar - rcstar
deltaE = ((deltal ^ 2) + (deltaa ^ 2) + (deltab ^ 2)) ^ .5
ratio = rcstar / ocstar
'
'  Determine reproduction a*, b* hue angle
'
          if rastar < 0 then goto 2570
          if rbstar < 0 then goto 2580
          rtheta = atn(rbstar / rastar): goto 2590
2570      rtheta = (atn(rbstar / rastar) + 3.1415927#): goto 2590
2580      rtheta = (atn(rbstar / rastar) + 6.2831853#)
2590 fil(l, m, cnt) = rlstar: fil(l, m, cnt + 1) = rastar
fil(l, m, cnt + 2) = rbstar: fil(l, m, cnt + 3) = rtheta
fil(l, m, cnt + 4) = deltaE: fil(l, m, cnt + 5) = ratio
cnt = cnt + 6: cnt2 = cnt2 + 1: P = 2 ^ cnt2
wend
     next l
next m
'
'  Write data to file.
```

```
'
open "o", #1, fl$
for m = 1 to y
for l = 1 to x
print #1, fil(l, m, 1), fil(l, m, 5), fil(l, m, 11);
print #1, fil(l, m, 17),fil(l, m, 23)
next l: next m
close #1
open "o", #1, fh$
for m = 1 to y
for l = 1 to x
print #1, fil(l, m, 4), fil(l, m, 8), fil(l, m, 14);
print #1, fil(l, m, 20), fil(l, m, 26)
next l: next m
close #1
open "o", #1, fs$
for m = 1 to y
for l = 1 to x
print #1, fil(l, m, 10), fil(l, m, 16);
print #1, fil(l, m, 22), fil(l, m, 28)
next l: next m
close #1
open "o", #1, fstr$
for m = 1 to y
for l = 1 to x
print #1, fil(l,m,2), fil(l,m,3),fil(l,m,6); fil(l,m,7), fil(l,m,12);
print #1, fil(l,m,13), fil(l,m,18), fil(l,m,19), fil(l,m,24), fil(l,m,25)
next l: next m
close #1
open "o", #1, fe$
for m = 1 to y
for l = 1 to x
print #1, fil(l, m, 9), fil(l, m, 15), fil(l, m, 21), fil(l, m, 27)
next l: next m
close #1
cls
print "Program is over ..."
beep : beep : beep
'
'       Close file and return to main menu
'
end
```

TRISTIMULUS VALUES OF SYSTEM COLOR SET

	ELECTROPHOTOGRAPHY (CANON NP)	THERMAL (MITSUBISHI)	INKJET (QUADJET)
RED (X / Y / Z)	28.7 / 17.9 / 8.8	30.2 / 16.3 / 5.8	47.2 / 33.2 / 25.6
GREEN	8.9 / 17.3 / 13	7.8 / 15 / 8.6	26.2 / 40.8 / 27.7
BLUE	8.5 / 6.4 / 26.3	5.0 / 4.5 / 12.3	21.3 / 17.9 / 46.4
CYAN	18.9 / 24.5 / 57.1	17.2 / 21.2 / 63.4	33.7 / 44.8 / 82.3
MAGENTA	32.1 / 18.8 / 22.7	32.0 / 17.3 / 14.3	50.5 / 32.4 / 47.2
YELLOW	64.1 / 74.1 / 17.5	66.8 / 72.0 / 10.7	69.8 / 77.0 / 23.7
WHITE	81.1 / 83.3 / 92.7	83.4 / 85.4 / 96.0	82.0 / 84.0 / 95.4
BLACK	4.3 / 4.7 / 6.3	2.7 / 3.0 / 3.7	15.3 / 16.4 / 20.4

REFERENCES

1) Jarvis, J. F., C. N. Judice, and W. H. Ninke: "A Survey of Techniques for the Display of Continuous Tone Pictures on Bi-Level Displays," *Computer Graphics and Image Processing*, **5**: 13-40 (1976).

2)Stoffel, J. C. and J. F. Moreland: "A Survey of Electronic Techniques for Pictorial Image Reproduction," *IEEE Trans. Commun. COM-29*, pages 1898-1924, December 1981.

3) Floyd, R. W. and L. Steinberg: "An Adaptive Algorithm for Spatial Greyscale", *Proc. Soc. Inf. Disp.*, **17**: 75- 77 (1976).

4) Engeldrum P. G. and P. A. Zuber: "A Colorimetric Halftoning Algorithm for Four Color Printers," *ISCC / SID Joint Tech. Conf. Meeting Digest,* pages 19-20, May 1988.

5) Engeldrum, P. G.: "Four Color Reproduction Theory for Dot Formed Imaging Systems," *J. Imag. Tech.*, **12**: 126-130 (1986)

6) Wyszecki, G. and W. S. Stiles: *Color Science*, 2nd Ed., Wiley, New York, 1982, chapter 3.

7) Wyszecki, G. and W. S. Stiles, Ref. 6, page 162-163.

8) Neugebauer, H. E. J.: "Die Theoretischen Grundlagen des Mehrfarbenbuchdrucks," *Z. Technol. Phys.*, **36**: 75-89 (1937).

9) Hardy, A. C. and F. L. Wurzburg: "Color Correction in Color Printing," *J. Opt. Soc. Am.*, **38**: 300-307 (1948).

10) Engeldrum, P. G.: "Color Reproduction Principles for Hard Copy Systems," Image Technology, Walworth, NY, 1985, page PGE 100.

11) Engeldrum, P. G., Ref. 10, page 129.

12) Engeldrum, P. G., Ref. 5, page 129.

13) Bayer, B. E.: "An Optimum Method for Two-Level Rendition of Continuous-Tone Pictures," *IEEE Inter. Conf on Comm., Conference Record*, 26-11 to 26-15 (1973)

14) Kinoshita, H., Y. Miyake, Y. Tamoto, and S. Kubo: "A New Evaluation of Image Quality of Digital Halftone Images Obtained by Ordered Dither Method," *J. Imag. Tech.*, **10:** 181-185 (1984)

15) Bayer, B.E., Ref. 13, pages 26-11 to 26-15

16) Limb, J.O.: "Design of Dither Waveforms for QuantizedVisual Signals," *Bell System Tech. Jour.*, **48:** 2555-2582 (1969).

17) Jarvis, J. F., C. N. Judice, and W. H. Ninke, Ref. 1, 23-26.

18) Engeldrum, P. G., Ref. 10, page PGE 135.

19) Floyd, R. W., Steinberg, L., Ref. 3, page 75-77.

20) Floyd, R. W., Steinberg, L., Ref. 3, page 76.

21) Hunt, R. W. G.: *The Reproduction of Colour*, 3rd Edition, Fountain Press, Hertfordshire, England, 1975, pages 165-184.

22) Engeldrum, P. G., Ref. 10, page 16.

23) R. W. G. Hunt, Ref. 21, page 176.

24) Allision, R., Y. Gur, and F. X. D. O'Donnell: "Color Reproduction Using Error Diffusion," *SPSE 40th Annual Conf. Adv. Printing of Conf. Summaries*, pages 82-85, May 1987.