Barycentric Screening

Nur Arad Doron Shaked Zachi Baharav
HP Labs Israel*
and
Qian Lin
Computer Peripherals Laboratory, HP Labs .†

Keywords: Color Halftoning, Dithering, Barycentric Coordinates.

Abstract

Monochrome dither halftoning is a gray-scale image rendering procedure where gray-levels in an image are compared against a periodic threshold array, placing a *Black* dot in every location whose gray-level is smaller than the corresponding threshold value. Current generalizations to color printing result in Cartesian thresholding procedures, namely *RGB* values are compared component-wise to trivariate thresholds. In this report we develop a color dithering procedure based on a non-Cartesian coordinate system. Its main advantage over the traditional Cartesian thresholding is that it allows the rendition of solid color patches while using no more than a preselected quadruple of colors, thereby enabling a reduction of halftone noise.

*Contact: HP Labs Israel Center, Technion City, Haifa 32000, Israel. Email: [dorons,zachi]@hp.technion.ac.il

†Contact: HP Labs, 1501 Page Mill Rd., Palo Alto, CA 94304. Email: qlin@hpl.hp.com

1 Introduction

Monochrome halftone algorithms are carefully designed to reduce visible artifacts. One of the most important factors producing those artifacts is the variation in the brightness of the dots. In monochrome halftones (*Black* and *White* dots) this factor cannot be mitigated. Current color halftoning algorithms are usually a Cartesian product of three halftoned monochrome planes corresponding to the color components of the image [7]. This generalization of monochrome algorithms overlooks the fact that colored dots vary in brightness, which is one of the most important factors affecting halftone noise.

To produce a good color halftone one has to place colored dots so that the following specifications are optimally met:

1. The placement pattern is visually unnoticeable.
2. The local average color is the desired color.
3. The colors used reduce the noticeability of the pattern.

The first two design criteria are easily carried over from monochrome algorithms. However, the third cannot be satisfied by a simple Cartesian product generalization of monochrome halftoning.

In a previous technical report [6] we formalized the 3^{rd} design rule as the *Minimal Brightness Variation Criterion* (MBVC). Based on this criterion, the *Ink Relocation Postprocess* was introduced, and was shown to improve on the quality of arbitrary color halftones. Following its success it was decided that the MBVC should be directly incorporated into halftoning schemes. Color Diffusion, an error-diffusion type algorithm was introduced and investigated in [5]. In this report we propose a method for incorporating the MBVC to dithering, which requires a whole new perspective on the features and underlying processes of dithering (mainly thresholding). The different geometric setting in which thresholding is defined requires the use of a different set of coordinates.

In the next section we introduce the color brightness rational motivating the proposed formalism. Section 3 introduces dithering in barycentric coordinates. Section 4 describes a possible method to design dither screens suited for the proposed dithering scheme. In Section 5 some dithering results are presented, and finally Section 6 summarizes the report in a discussion of the pros and cons of Barycentric Screening.

2 The Bright Side of the *RGB* Cube

In this section we analyze the color-selection design criterion (design criterion number 3), see [6] for details. Consider the case of rendering a large patch of an arbitrary solid color. It is known that any color in the *RGB* cube may be rendered using the 8 basic colors located at the vertices of the cube. Actually, any color may be rendered using no more than 4 colors, different colors requiring different quadruples. Moreover the quadruple corresponding to a specific color is, in general, not unique (in a linear color space, any quadruple whose convex hull contains the desired color will do). The issue we raise in this section is: Suppose we want to print a patch of solid color; what colors should we use? Note that in previous work done on halftoning the issue was what pattern should the dots be placed in, and less often how many dots of each color should be used.

Consider the basic rationale of halftoning: When presented with high frequency patterns, the human visual system "applies" a low-pass filter and perceives only their average. Current inkjet printing resolution (up to 600 dpi) can still be resolved by the human visual system, thus still higher frequencies will have to be achieved. Relevant to the problem at hand is the fact that the visual system is more sensitive to changes in brightness than to changes in the chrominance, which average at much lower frequencies. Thus we arrive at the Minimal Brightness Variation Criterion (MBVC):

> *To reduce halftone noise, select from within all color sets by which the desired color may be rendered, the one whose brightness variation is minimal.*

To consider the brightness variation of color sets we only need to order the eight basic colors on a brightness scale. Using standard *hp* inks results in the ordering shown in Figure 1.

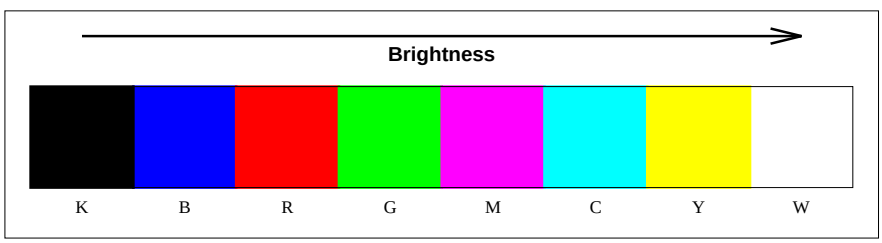

Figure 1: The brightness scale of the eight basic colors rendered using *hp* inks.

An interesting expected byproduct of the use of MBVC compliant halftones is that color patches are rendered as more saturated. This phenomenon is highly dependent on the media (paper type) and the incorporated brightness correction. Improved color saturation is expected because when applying the MBVC, neutral dots (K or W) are used less often, and saturated dots (R, G, B, C, M, or Y) are used instead. Thus rendered patches appear far from the neutral (*Gray*) axis.

In a dithering procedure where the *RGB* value is compared component-wise to a threshold (we refer to such a dither as a *Cartesian* or *rectangular* dither) all 8 basic colors are usually used in rendering almost any solid color patch. However, the use of 8 colors (where 4 would suffice) stands in blunt contradiction to the MBVC (since e.g. *Black* and *White* dots whose brightness variation is maximal are used simultaneously).

To determine the minimal brightness variation color-set for a given *RGB* triplet we partition the *RGB* cube into the six tetrahedra shown in Figure 2. For all the colors in a tetrahedron, the tetrahedron vertices are the minimal brightness variation color set, or minimal brightness variation quadruple. The detailed derivation of this partition may be found in [6]

For an arbitrary *RGB* triplet, determining the tetrahedron to which it belongs requires between 2 and 3 comparisons:

```
inline pyramid position(BYTE R, BYTE G, BYTE B)
{
  if((R+G) > 255)
    if((G+B) > 255)
      if((R+G+B) > 510)    return CMYW;
      else                 return MYGC;
    else                   return RGMY;
  else
    if((G+B) < 256)
      if((R+G+B) < 256) return KRGB;
      else                 return RGBM;
    else                   return CMGB;
}
```

Color Diffusion [5] is a novel error-diffusion type algorithm which complies with this partition. This report is devoted to the design of a compliant dithering type halftoning algorithm.

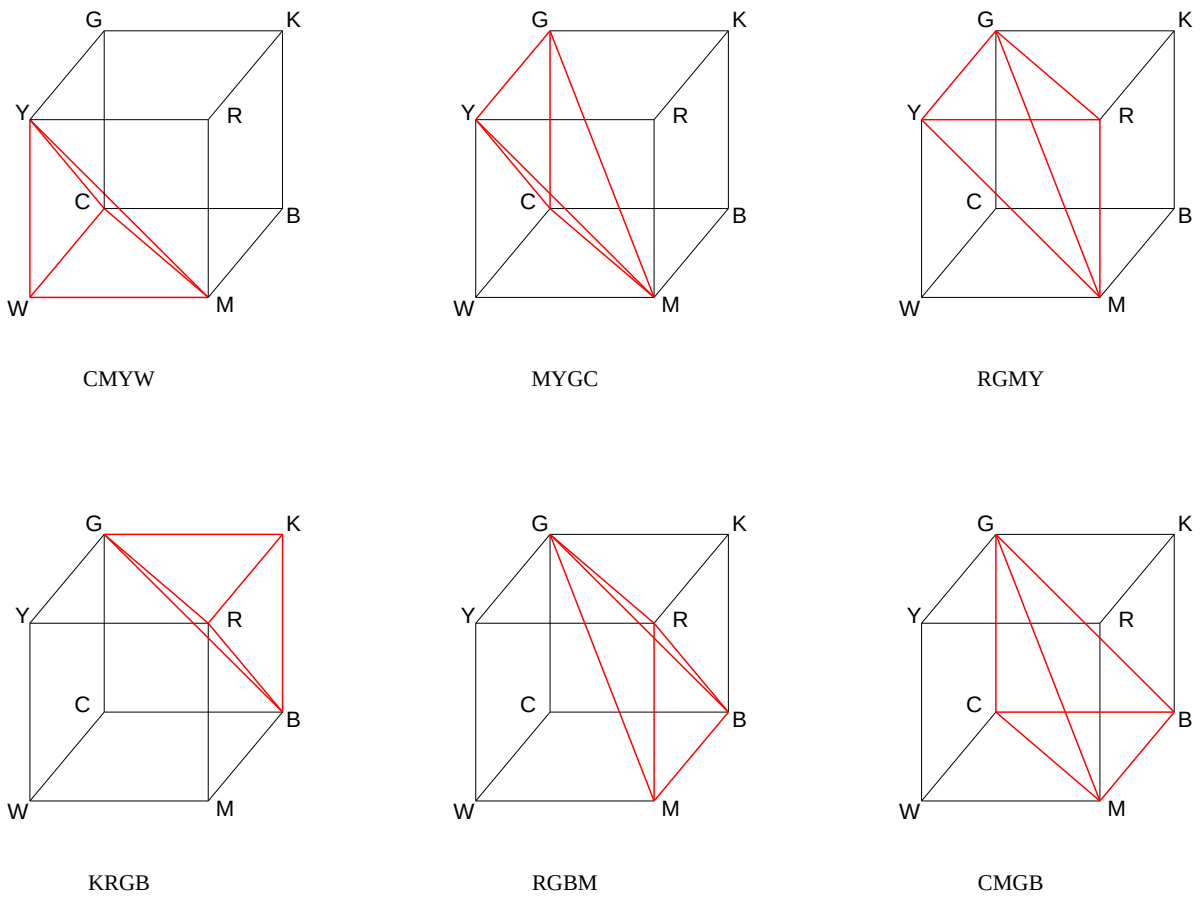

Figure 2: The partition of the RGB cube to six volumes, each of which is the convex hull of the minimal brightness variation quadruple used to render colors in that tetrahedron.

Thus in the next section we introduce an algorithm for thresholding a color value inside a tetrahedron to one of its vertices.

3 Color Dithering in a Tetrahedral Space

Consider now a subset of RGB space which is the convex hull of four colors. For example, consider the set denoted by $KRGB$ which is the set of colors for which $R+G+B < 256$, and is also the convex hull of K, R, G and B. According to the Mean Color Invariance (MCI) rule (design criterion number 2), given an RGB value C inside the tetrahedron the relative amount of K, R, G and B dots used to render a solid color patch of color C, should be α, β, γ

and δ, such that

$$\begin{cases} K\alpha + R\beta + G\gamma + B\delta = C \\ \alpha + \beta + \gamma + \delta = 1 \end{cases} \tag{1}$$

This equation should be interpreted as a vector equation in RGB space ($\mathbb{R}^3$). Assuming linear behavior of color mixing, it guarantees that the average RGB value of the dots rendering a large patch of color C is equal to C.

Equation 1 suggests the introduction of barycentric coordinates into our analysis.

3.1 Barycentric Coordinates

Given a set of $n+1$ points, $X = \{X_1, X_2, \ldots, X_{n+1}\} \subset \mathbb{R}^n$, not contained in a single hyperplane, the convex-hull of X is the n-dimensional simplex $S(x)$. For any point $P \in \mathbb{R}^n$ there exists a unique solution to the linear system

$$\begin{cases} X_1\alpha_1 + X_2\alpha_2 + \ldots + X_n\alpha_n = C \\ \alpha_1 + \alpha_2 + \ldots + \alpha_n = 1 \end{cases} \tag{2}$$

The existence and uniqueness of the solution may be shown as follows: The determinant of the linear system (2) is $|X_1' X_2' \ldots X_n'|$, where X_i' is the $n+1$ dimensional column vector, whose first n coordinates equal those of X_i, and whose last coordinate is 1. This determinant is identical to the one used to compute the volume of $S(X)$ (provided we orient the coordinate system in the same sense as that of the ordered set X):

$$Vol = \mathrm{Vol}(S(X)) = \frac{1}{n!} \left| X_1' X_2' \ldots X_{n+1}' \right| \tag{3}$$

and as such, the determinant never vanishes.

For a given ordered set $X = \{X_1, X_2, \ldots, X_{n+1}\} \subset \mathbb{R}^n$ the transformation

$$P \mapsto (\alpha_1, \alpha_2, \ldots, \alpha_{n+1}) \tag{4}$$

is called the *barycentric transformation* of $\mathbb{R}^n$ with respect to X. The coordinates of the

$n+1$ dimensional vector $(\alpha_1, \alpha_2, \ldots, \alpha_{n+1})$ are called the *barycentric coordinates* of P (with respect to X), and the barycentric coordinate corresponding to a vertex Q ($Q \in X$) is called the $Q-$barycentric coordinate of P, and is denoted by P_Q. In case the vertices comprising X are indexed, we may use the notation P_i instead of P_{X_i}. Use of a barycentric coordinate system offers a rich geometric structure on the space, and is widespread in Computer-Aided Geometric Design [4] and in finite-element methods for the solution of elliptic partial differential equations [2].

The barycentric transformation (4) is valid for every point in $\mathbb{R}^n$. However, we are interested only in the points *inside* $S(X)$. As it turns out, these points are precisely those for which all the barycentric coordinates are non-negative. To see this consider a point $P \in S(X)$. P partitions the simplex defined by X into $n+1$ sub-simplices, each defined as the convex hull of P and n (out of $n+1$) points of X. The volume, Vol_i, of any such sub-simplex is

$$Vol_i = \frac{1}{n!} \left| X'_1 \ldots X'_{i-1} P' X'_{i+1} \ldots X'_{n+1} \right|, \tag{5}$$

thus each sub-simplex takes up

$$\frac{Vol_i}{Vol} = \frac{\left| X'_1 \ldots X'_{i-1} P' X'_{i+1} \ldots X'_{n+1} \right|}{|X'_1 X'_2 \ldots X'_{n+1}|}, \quad i = 1, 2, \ldots, n+1 \tag{6}$$

of the volume of the original simplex. By Kramer's rule these values are exactly the solution

$$P_i = Vol_i / Vol \tag{7}$$

of the linear system (2), proving that for points $P \in S(X)$ all barycentric coordinates are non-negative. An illustration of these notions (for the planar case) is shown in Figure 3. For points P not contained in $S(X)$ there exists a vertex X_i (any vertex for which the simplex defined by P and all vertices but X_i does not intersect $S(X)$, these may number up to n) such that

$$Vol_i = -\frac{1}{n!} \left| X'_1 \ldots X'_{i-1} P' X'_{i+1} \ldots X'_{n+1} \right|, \tag{8}$$

and thus the corresponding barycentric coordinate is negative.

Consider now a point $X_i \in X$, and denote by F_i the face of the $S(X)$ which does not contain X_i. All points contained in a hyper-plane parallel to F_i have the same $i-$th barycentric coor-

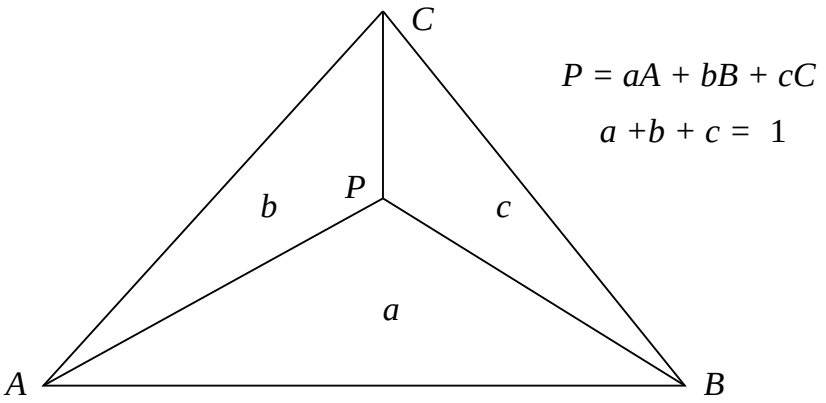

Figure 3: The barycentric coordinates of a point $P = aA + bB + cC$ inside the triangle ABC. P partitions the triangle into 3 sub-triangles. Supposing the area of triangle ABC is 1, then the area of the sub-triangle not adjacent to vertex V ($V = A, B, C$) is equal to P_V.

dinate. For example, the plane containing F_i is characterized by the fact the i-th barycentric coordinate vanishes there, and the plane parallel to F_i and which passes through X_i is characterized by its i-th barycentric coordinate equaling unity. On the other hand, all points P contained in a straight line passing through X_i may be characterized by constant ratios $P_j : P_k$ =const. for all $j, k \neq i$. For example, for points P contained in the line connecting $\frac{(X_j+X_k)}{2}$ and X_i the ratio $P_j : P_k$ equals 1. These observations are shown in Figure 4.

Another useful property of the barycentric coordinate system is that it is invariant to regular affine mappings:

If $P = (P_{X_1}, P_{X_2}, \ldots, P_{X_{n+1}})$, and $A(P) = (A(P)_{A(X_1)}, A(P)_{A(X_2)}, \ldots, A(P)_{A(X_{n+1})})$, where A is a regular affine transformation, then $P_{X_i} = A(P)_{A(X_i)}$ for $i = 1, 2, \ldots, n+1$.

3.2 Barycentric Thresholding

The connection between color production and barycentric coordinates is embodied in *additive color mixing*, where one produces a given color by mixing the correct amounts of predefined "base" colorants. We now shift our emphasis to the computational aspects of additive color mixing.

Suppose we are given a constant color patch $C \in S(X)$, such that $C = \alpha_1 X_1 + \alpha_2 X_2 + \alpha_3 X_3 + \alpha_4 X_4$, and $\sum_{i=1}^{4} \alpha_i = 1$, where $X = \{X_1, X_2, X_3, X_4\}$ is a four color set. Assuming we have a threshold array uniformly distributed in $S(X)$, we would like a threshold procedure

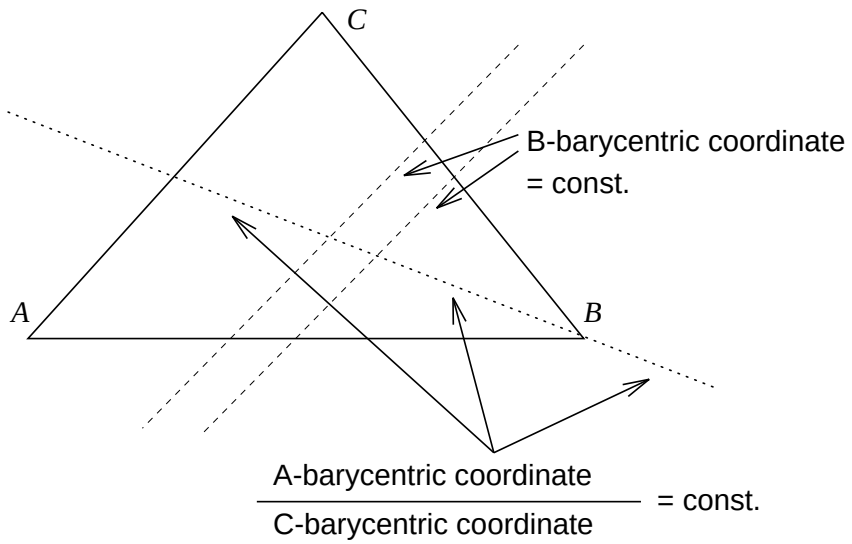

Figure 4: The iso-parametric lines of the barycentric coordinate system: The B-barycentric coordinate of points lying on a hyper-plane parallel to AC is constant. For all points P contained in a straight line passing through B the ratio $P_A : P_C$ is constant.

to threshold α_i of the pixels to color X_i for $i = 1, 2, 3, 4$. Keeping in mind the intimate connection between the barycentric coordinates of C and the volume of the sub-tetrahedra induced by C, the thresholding procedure should be carried out as follows: Partition the tetrahedron into 4 sub-tetrahedra T_i ($i = 1, 2, 3, 4$), each one referenced by the vertex it does not contain. A threshold contained in T_i will threshold color C to color (vertex) X_i. The thresholding procedure is shown (for a two-dimensional case) in Figure 5 (left). Recognizing the importance of the dual problem, we also ask the following: Given a particular threshold $Th \in S(X)$, which colors in $S(X)$ will be thresholded to which of the vertices X_i? It is quite easy to see that in such a case the colors contained in $S(X)$ will be thresholded as shown in Figure 5 (right).

At this point it is useful to adopt the following terminology: Given a color C and a threshold Th contained in $S(X_1, X_2, X_3, X_4)$, if the thresholding procedure results in the vertex X_i, then we say that "Threshold Th thresholds color C to vertex X_i", or "color C is thresholded by Th to vertex V".

Application of the barycentric threshold to a color C is carried out as follows:

1. Compute the barycentric coordinates of the threshold Th and the color C. Denote

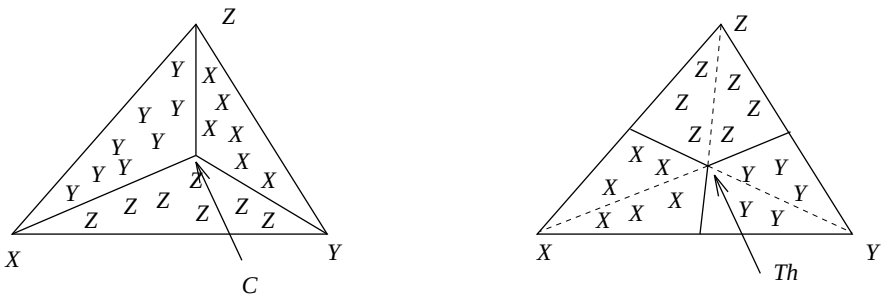

Figure 5: Barycentric thresholding. **Left**: For a given color C all thresholds in area 'X' will threshold color C to vertex X. Likewise for 'Y' and 'Z'. **Right**: For a given threshold Th all colors in area 'X' will be thresholded by Th to vertex X. Likewise for 'Y' and 'Z'.

these by (Th_1, Th_2, Th_3, Th_4) and (C_1, C_2, C_3, C_4) accordingly.

2. Compute the ratios $r_i = C_i/Th_i$ for $i = 1, 2, 3, 4$.

3. Color C is thresholded by Th to vertex X_j, where $j = \text{argmax}_i r_i$.

An example in the two-dimensional case showing that this computation results in the correct thresholding procedure is shown in Figure 6.

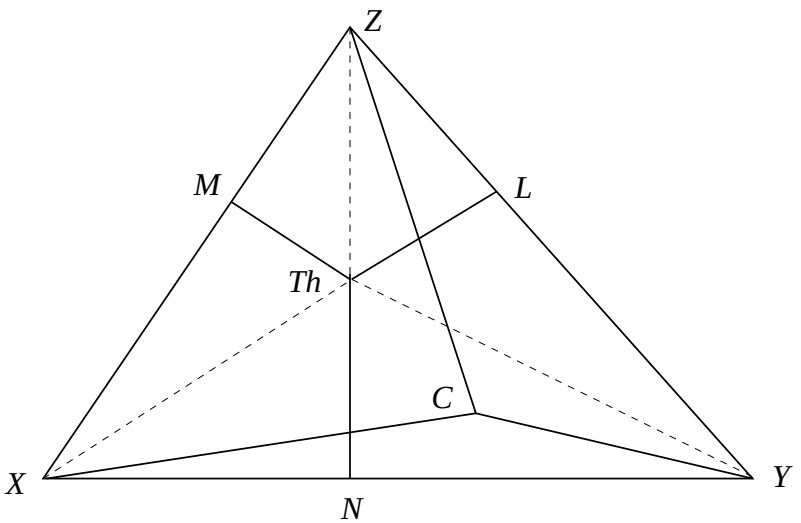

Figure 6: Applying the barycentric threshold procedure: The line ZN, $(YM,\ XL)$ contains all points P for which $P_X : P_Y = Th_X : Th_Y$ $(P_X : P_Z = Th_X : Th_Z,\ \ P_Y : P_Z = Th_Y : Th_Z)$. Since C is located right of the line ZN then $C_Y/C_X > Th_Y/Th_X$. By the same token $C_X/C_Z > Th_X/Th_Z$ and $C_Y/C_Z > Th_Y/Th_Z$. Thus vertex Y satisfies $C_Y/Th_Y > C_X/Th_X$ and $C_Y/Th_Y > C_Z/Th_Z$, and Th thresholds color C to vertex Y.

3.3 The Stacking Property

An important property of monochrome dithering procedures (as well as of Cartesian color dithering) is the stacking property. We now formulate the stacking property of dithering in the following manner:

Suppose color C is thresholded by Th to vertex V. Now regard color C as a threshold. The collection of all points (colors) which are thresholded by C to V will also be thresholded by Th to V. Figure 7 depicts the stacking property in the Cartesian and barycentric cases.

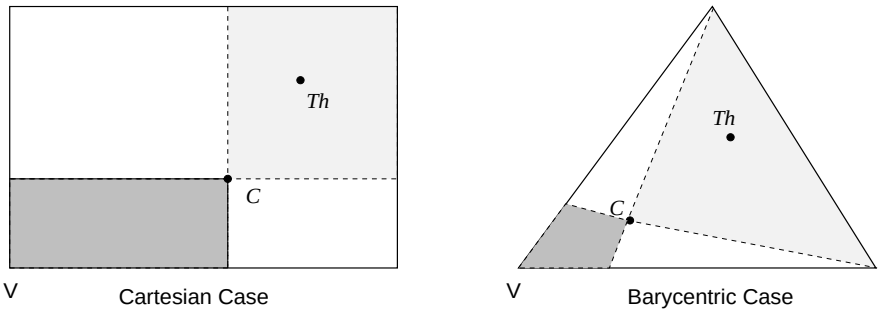

Figure 7: The stacking property of thresholding. Supposing color C is thresholded to the vertex V. Then the threshold is necessarily located in the light gray area on the right. Thus any point in the dark gray area on the left will also be thresholded to V. Stated differently, if we view C as a threshold then all colors which C thresholds to V will also be thresholded by Th to V.

In the Cartesian case, one may "decompose" the stacking property into its Cartesian components; by the same token, one may decompose the barycentric stacking property into its barycentric components.

Correct interpretation of the stacking property proves useful when designing dither screens. Take for example the monochrome case and a solid grayscale patch. The rendering of a patch in this case may be viewed as a process where *Black* dots are progressively placed on a *White* page, until the desired gray-level is reached. The stacking property enforces one limitation: Once a dot pattern has been agreed upon for gray-level k, one may only *add* dots to the pattern to obtain the dot pattern for grey-level l for $l > k$. With color printing and Cartesian thresholding the same holds, but for each of the color components.

We may, however, interpret the process of printing a color C as that of placing at each position on the page one of the four colors defining the tetrahedron containing C. The notion of adding a K dot on a *White* media may thus be reformulated as *replacing* a W dot

by a K one. The stacking property now enforces the following limitation: Suppose color C was rendered with a certain pattern. Then for all colors stacked on the V-side of C (the shaded area on the lower left side of Figure 7) the set of pixel positions for which halftone color V is placed is a subset of the set corresponding to color C.

An alternative interpretation of the stacking property, an interpretation we will be using, is: On any straight line passing through a vertex and for every pixel position there is a single *switch color* in which the halftone placed at that position changes. This change is always from the vertex color (used for colors close to the vertex) to some other halftone color (used for colors far from the vertex).

3.4 Threshold Distribution

The issue of the distribution of the thresholds constituting a dither screen in the color space offers further insight into the geometry of dithering.

We first define a dither screen as *MCI* if it fulfills the Mean Color Invariance rule (design criterion number 2), namely, for any solid patch of constant color C, the average color of the halftone is also C. With Cartesian dither screens the situation is quite simple: A Cartesian dither screen is MCI iff the projection of the thresholds on any one of the major axes results in a uniform distribution. Such a constraint does not necessarily imply that the distribution of thresholds throughout the *RGB* cube is uniform. Indeed, one may obtain an MCI dither screen by taking a Cartesian product of identical copies of one MCI monochrome screen. In this case, all thresholds will be located on the gray axis (a distribution which is far from a uniform distribution in the *RGB* cube).

When performing barycentric dithering, a much stronger constraint on the distribution of the thresholds must be satisfied.

Theorem: A barycentric dither screen is MCI iff the distribution of thresholds in the simplex is uniform.

Proof: The fact that a barycentric screen with uniformly distributed thresholds is MCI, is a direct consequence of the procedure itself. We prove the converse for a two dimensional case. The generalization to a three dimensional (or n-dimensional) case is straightforward.

For the sake of simplicity we assume that a dither screen is a function S defined over the

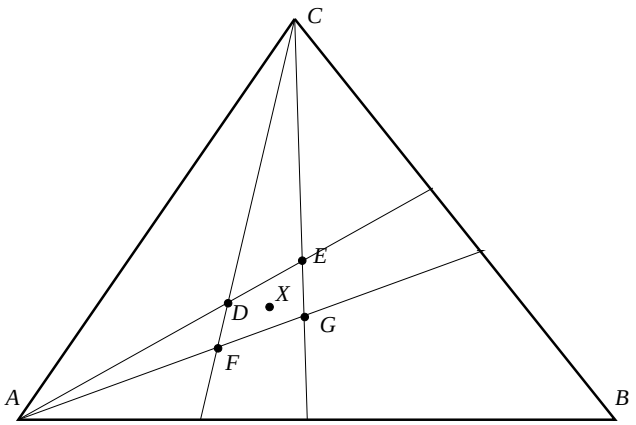

Figure 8: An illustration of the proof that MCI barycentric screens have uniform threshold distributions.

square $[0,1]\times[0,1]$, attaining values in the triangular color space, $\triangle(A,B,C)$, whose vertices are the colors A, B and C. In a continuous setting we may define $f(x)$, the threshold density for $x \in \triangle(A,B,C)$. We must therefore prove that $f(x) = \text{const}$.

From (7) the B-barycentric coordinate of any point $X \in \triangle(A,B,C)$ is the ratio

$$X_B = \frac{\int\limits_{\triangle(A,C,X)} dx}{\int\limits_{\triangle(A,B,C)} dx} = \frac{|\triangle(A,C,X)|}{|\triangle(A,B,C)|}, \tag{9}$$

where $|\triangle(\cdot,\cdot,\cdot)|$ denotes the triangle area. However, in an MCI barycentric screen X_B is the ratio of thresholds sending color X to vertex B which is in our case

$$X_B = \int\limits_{\triangle(A,C,X)} f(x)\,dx. \tag{10}$$

Without loss of generality assume that area of of the entire color space is unity $|\triangle(A,B,C)| = 1$, resulting in

$$|\triangle(A,C,X)| = \int\limits_{\triangle(A,C,X)} f(x)\,dx \tag{11}$$

for all $X \in \triangle(A,B,C)$.

Using the notation of Figure (8) we have

$$|\square(D,E,G,F)| = |\triangle(A,C,G)| - |\triangle(A,C,E)| - |\triangle(A,C,F)| + |\triangle(A,C,D)| \tag{12}$$

substituting (11) in the above we obtain

$$\int_{\square(D,E,G,F)} f(t)\,dt = |\square(D,E,G,F)| \tag{13}$$

Finally, for any point $X \in \triangle(A,B,C)$ we may construct a sequence of neighborhoods of X of the type $\{D,E,G,F\}$ (as in Figure 8) whose radius approaches zero. Assuming f is continuous, we conclude by the mean-value theorem for integrals that $f(X) = 1$ for any $X \in \triangle(A,B,C)$. □

It is instructive to make a futile attempt at proving a similar theorem for the Cartesian case. Such an attempt is doomed to fail since Equation (11) has no counterpart in the Cartesian case. The validity of the equation in the barycentric case may be traced back to the fact that the barycentric transformation is well-defined, while no such well-defined transformation exists in the Cartesian case. The positive result of the theorem in the barycentric case further supports the claim that barycentric screening better captures the true multi-dimensional character of the color space.

4 Dither Screen Design

A common way of measuring the quality of a monochrome dither screen is its performance on uniform gray-level patches [3]. Supposing one has a method of optimizing a *Black* & *White* dot pattern, one may design a monochrome screen using the following steps, where for the sake of simplicity we assume throughout this section that the screen size is always 16×16 (up to 256 different thresholds);

1. Begin with a 16×16 matrix of *Black* pixels. This is obviously the best quality dot pattern for gray-level 0.

2. Suppose a dot pattern has been generated for gray-level k. Among all pixel positions that are still *Black* (their number is $256 - k$) switch to *White* that position which

results in the best quality dot pattern gray-level $k+1$. Note that the transition from pattern k to pattern $k+1$ is severely limited by the stacking property.

3. Repeat the previous step for all gray-levels from 1 to 255.

4. For each pixel position (i,j) count the number $N(i,j)$ of appearances of *Black* throughout the dot patterns.

5. Define the threshold at position (i,j) as $N(i,j)$.

The dot patterns for the individual gray-levels will be called *control patterns*. It is easy to see that the resulting screen reproduces the original control patterns from corresponding constant gray-level images.

When generalizing the above method to Cartesian color dithering, the stacking property imposes a much stronger limitation. Suppose an optimal dot pattern has been generated for the *RGB* value (i,j,k). The Cartesian stacking property now forces the dot pattern for $RGB = (l,j,k)$ $(0 \leq l \leq 255)$ to be identical to that of (i,j,k) in the second and third coordinates. Thus in order to be compatible with the stacking property one has to design the screen by generating control patterns complying to the stacking property in each coordinate. One such method is by generating control patterns for the colors on the gray axis, i.e., $(r,g,b) = (i,i,i)$, $i = 0,1,\ldots,255$, and then generating the dither screen componentwise [1]. The collection of all colors whose rendering is used in the design process is called the *control path*, and the individual colors are called *control colors*.

We now show that a similar design principle may be used to generate a barycentric dither screen. Assume for the sake of simplicity that we are designing a screen for the *KRGB* tetrahedron and all *RGB* components are normalized to the interval $[0,1]$. In such a case the barycentric transformation is

$$(r,g,b) \mapsto (1-(r+g+b),r,g,b) \tag{14}$$

Note that our interpretation of the notion "ink dot" is different from that implied in [1], where an ink dot may attain one of the values $\{K,C,M,Y\}$, dots may overlap, and not every pixel position is covered. In our interpretation, an ink dot attains any one of eight values (actually four, since we restrict ourselves to one of the six tetrahedra), and each position

is covered by precisely one dot. Using this interpretation it is convenient to multiply the barycentric coordinates (14) by 256, thus obtaining the *number* of dots (of each color) used to render a 16×16 patch of RGB value (r, g, b). This representation will be referred to as the *dot-normalized barycentric representation*, where for a particular color the coordinates are identical to the number of corresponding dots used to render a 16×16 patch of that color. For example, if one wishes to produce a dot pattern for $RGB = (64, 64, 64)$, transforming to dot-normalized barycentric coordinates one obtains

$$(64, 64, 64) \mapsto (64, 64, 64, 64), \tag{15}$$

meaning that one should use 64 dots of K, R, G and B to render such a patch. This particular color is located in the centroid of the $KRGB$ tetrahedron, and the dot pattern corresponding to it will be referred to as the *centroid pattern*.

We now outline the steps in the design of the barycentric screen.

1. Design the optimal centroid pattern. This dot pattern contains 64 dots of K, R, G and B.

2. For $k = 1, 2, \ldots 64$ design a dot pattern for color $(64 + 3k, 64 - k, 64 - k, 64 - k)$. Dot pattern $k + 1$ is based on pattern k: Optimally select one dot of R, one dot of G, and one dot of B, to be transformed to K.

3. For $r = 1, 2, \ldots 64$ design a dot pattern for color $(64 - r, 64 + 3r, 64 - r, 64 - r)$. Dot pattern $r + 1$ is based on pattern r: Optimally select one dot of K, one dot of G, and one dot of B, to be transformed to R.

4. For $g = 1, 2, \ldots 64$ design a dot pattern for color $(64 - g, 64 - g, 64 + 3g, 64 - g)$. Dot pattern $g + 1$ is based on pattern g: Optimally select one dot of K, one dot of R, and one dot of B, to be transformed to G.

5. For $b = 1, 2, \ldots 64$ design a dot pattern for color $(64 - b, 64 - b, 64 - b, 64 + 3b)$. Dot pattern $b + 1$ is based on pattern b: Optimally select one dot of K, one dot of R, and one dot of G, and transform them to B.

This increment design method is meant to comply with the barycentric stacking property. Note also that in steps (2)–(5) the 0 index control pattern is the centroid pattern.

This procedure (on a two-dimensional case – mixtures of K, R, and G only) is illustrated in Figure 9. An example of four control patterns belonging to the 3D sub-path from the centroid pattern of $KRGB$ to the pure *Black* pattern is shown in Figure 10.

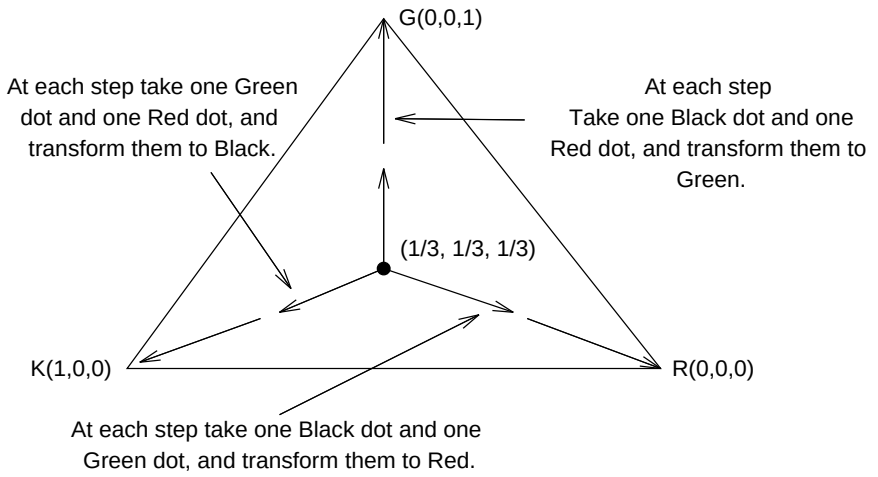

Figure 9: Designing the control patterns (for a two dimensional color space). The control path is decomposed into $n+1$ sub-paths (where n is the dimension of the space, $n = 2$ in this example). The dot patterns along each sub-path are constructed sequentially while complying to the barycentric stacking property. Barycentric coordinates of the endpoints of the sub-paths are shown.

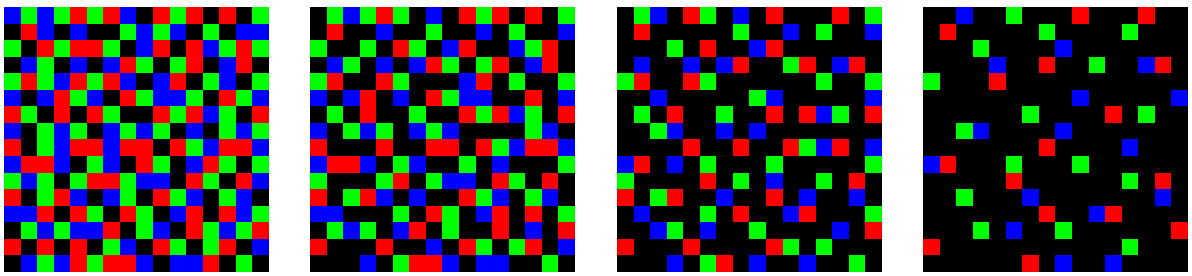

Figure 10: Examples of control patterns along the control path from the centroid pattern to the pure *Black* one. Left to Right: Centroid pattern followed by dot patterns of dot-normalized coordinates $(112, 48, 48, 48)$, $(160, 32, 32, 32)$ and finally $(208, 16, 16, 16)$. Each pattern is obtained from the one to its left by taking 16 R, G, and B dots (each) and replacing them with K dots.

Since a threshold in the barycentric threshold matrix influences only the corresponding position in all the control patterns, we locate the barycentric threshold of the (i, j) matrix position by inspecting the (i, j) position in all the control patterns. Thus we do the following for all the positions (i, j) of the barycentric threshold matrix.

Without loss of generality, suppose that the centroid pattern renders K at position (i, j). The stacking property dictates that all the dot patterns along the path to $(256, 0, 0, 0)$ (pure

K) also render K at the (i, j) position. The stacking property also dictates that at some control color along the path to $(0, 256, 0, 0)$ (pure R) the K dot should switch to R. In the same fashion at some control color along the path from the centroid pattern to $(0, 0, 256, 0)$ (pure G) the dot switches to G, and at some control color along the path to $(0, 0, 0, 256)$ (pure B) it switches to B.

The barycentric threshold for the (i, j) position may be calculated from the three switch color (as shown for a two dimensional example in Figure 11): Assuming again that the centroid color in the (i, j) position was K, the G-switch-color dictates that the threshold be on the plane through R, B, and the G-switch-color. Thus, the threshold is the intersection of three planes: The plane mentioned above, the plane through G, B, and the R-switch-color, the plane through R, G, and the B-switch-color.

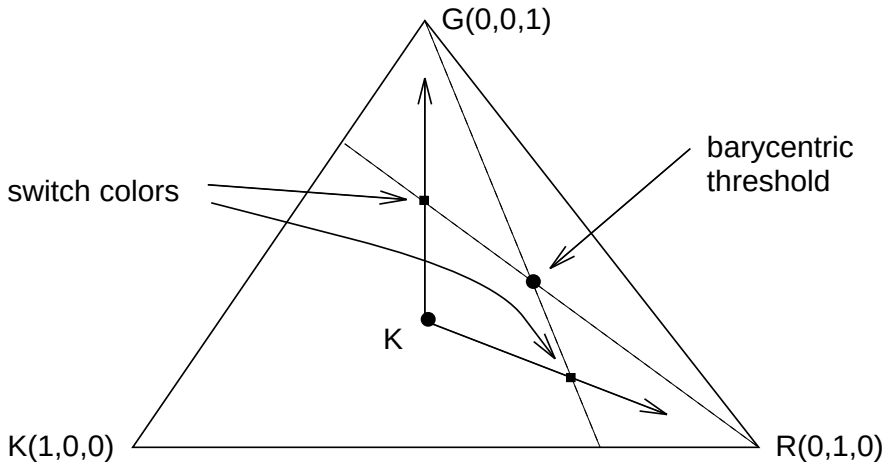

Figure 11: Computing the barycentric threshold from the collection of control patterns generated (for a two dimensional color space). Supposing a particular position was K at the centroid pattern, and it switches to R and G along the corresponding sub-paths. The simple geometric computation shown is used to determine the location of the barycentric threshold. This is the unique barycentric threshold inducing the given switch colors.

Note that this procedure ensures that the resulting screen reproduces all the control patterns, and that the control patterns are designed with the correct mixtures of halftone dots, i.e. *at the control colors* the screen is MCI. Since the control path expands over the whole simplex there is reason to believe that this property extends to all colors, and the resulting screen is close to complying with the MCI rule. Unfortunately this extension does not occur, and small though noticeable color changes may be detected in colors that are far from the control path (mainly those near the middle of the simplex edges).

The fact that barycentric dither screens designed by the abovementioned routine are not MCI is obvious from the fact that the resulting threshold distribution is not uniform in the simplex.

Similar color-space deformation problems occur for every halftoning method because of several illinear effects (e.g. dot overlap, ink mixing, paper and ink interaction, etc.). Traditionally such problems are corrected for by color correction routines, producing an inverse color deformation prior to the rendering stage. In barycentric dither screens designed by the abovementioned routine the illinear effects are more noticeable than in standard halftoning methods, however, not significantly so.

4.1 Tetrahedral Dither Stitching

The same type of procedure used to generate the $KRGB$ screen can be evoked to produce screens for the other five tetrahedra. However a simpler and more efficient option exists: Instead of storing the barycentric screen in RGB coordinates it is more efficient to store it in barycentric coordinates. Since barycentric coordinates are affine invariant, and all the tetrahedra are affine equivalent (there exists a regular affine transformation mapping any one of the tetrahedra to any other) we may use the same barycentric screen for all the tetrahedra, provided we establish a correspondence between the vertices of $KRGB$ and the vertices of all the other tetrahedra.

The correspondence we have chosen is the following:

$$\begin{aligned} K,R,G,B &\mapsto M,R,G,B \\ K,R,G,B &\mapsto M,C,G,B \\ K,R,G,B &\mapsto M,R,G,Y \\ K,R,G,B &\mapsto M,C,G,Y \\ K,R,G,B &\mapsto M,C,W,Y \end{aligned}$$

The reason for choosing this correspondence may be understood by looking at the following situation: Suppose when rendering an image there is a continuous color gradation from one tetrahedron to its neighbor. At the boundary between two tetrahedra only three out of four dot colors are used for rendering. The same three colors are used regardless of which of the two bounding tetrahedra the boundary color is classified to. We would like the resulting

pattern to be invariant to the classification at the boundary. This is possible iff for every couple of tetrahedra with an adjacent face the vertices are arranged so that the common ones match. An examination of Figure 2 shows that the correspondence presented is the only one possible.

5 Experimental Results

We first show an example of the advantage of barycentric screening by producing a halftone of a constant color patch by comparable rudimentary methods. The first is a *random Cartesian dither*, where each *RGB* value is thresholded in a Cartesian fashion by a threshold chosen randomly from a uniform distribution in the *RGB* cube. In the second method, thresholds uniformly distributed in the appropriate tetrahedron are randomly chosen, and are applied in a barycentric fashion to the same *RGB* values. The results, scaled $\times 4$, are shown in Figure 12. Although the dither pattern of both results is obviously of poor quality, the barycentric-rendered patch appears much smoother, due to reduced brightness fluctuations relative to Cartesian dithering. The difference in the perceived average color of the two patches is not relevant at this point; it may be dealt with by standard (printer and halftone-method dependent) color correction procedures.

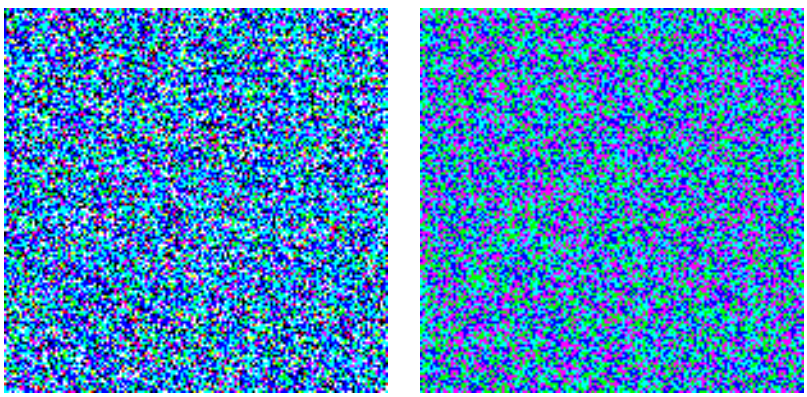

Figure 12: Random dithering in Cartesian and barycentric coordinates. **Left:** Patch of $RGB = (64, 128, 192)$ rendered with a random Cartesian screen. **Right:** The same patch rendered with a random barycentric screen. Barycentric screening uses only four colors, whose brightness variation is minimal (B, G, C and M in this case) as opposed to all eight possible colors used by Cartesian dithering.

Figure 13 is a barycentric screened, using a barycentric dither matrix designed as described in section 4. Its overall quality establishes that Barycentric Screening is feasible. It is

important to mention that Barycentric Screening does not yet compete with state of the art Cartesian dither matrices, mainly because the barycentric matrix design process has not yet been properly optimized. In that respect it should be noted that the first design rule (optimal dot placement) is more important then the MBVC, and until a comparably well designed dither matrix is not available, the quality of Barycentric Screening will be worse than the state of the art Cartesian dither. Efforts are being made to optimize the barycentric dither matrix design process.

6 Discussion

Color halftoning by barycentric screening fully complies to the Minimal Brightness Variation Criterion. We now analyze in detail the pros and cons of the proposed algorithm.

Time and Memory Requirements: Although barycentric screening runs considerably slower than Cartesian screening, error diffusion, and color diffusion on existing machines, it is still essentially a point-wise thresholding operation, and as such admits to highly efficient implementations. The reason for the relative sluggishness of current implementations of barycentric screening relative to existing methods is in the need for transforming the *RGB* representation of a color to its barycentric representation. However, pixel color values may be represented from the start in barycentric form, thereby greatly speeding up the thresholding procedure. Memory requirements for the proposed method are similar to that of Cartesian screening: One pixel at a time is processed, and the screen itself may be stored in read-only memory. For computational efficiency it is recommended that the screen be stored in barycentric form – 4 coefficients per location (although 3 would suffice, since they sum to unity, however storing only 3 out of 4 coefficients would require an additional computation) with 16-bit precision for each coordinate. The reason for this relatively high precision relates to subtle implementation details which are beyond the scope of this report.

An Intrinsic Coordinate System: A major advantage of the proposed screening method over traditional Cartesian screening is the fact that the coordinate system used is intrinsic, and is independent of the particular initial representation used for color. Moreover, the whole procedure is invariant to regular affine mappings, a claim which cannot be made with respect to separable screening, regardless of the external coordinate system. Apart from important analytic properties of the barycentric coordinate system, its use is, to some extent, much more natural to the problem: The barycentric coordinates of a color (with respect to a set

Figure 13: A Barycentricaly Screened image.

of 4 basic colors which may be rendered exactly) are actually the relative number of dots of each basic color which should be used in rendering. Note that we refrain from using the term *RGB*, since we view colors as belonging to some abstract 3-dimensional space (which is the convex hull of the basic colors) and thus we do not commit ourselves to any predefined Cartesian coordinate system (such as the *RGB* one).

A Different View of the Printing Process: In traditional separable screening the printing process is seen as an incremental procedure in which color dots are added to a *White* surface until the desired *RGB* value is reached [3]. With barycentric screening the interpretation is somewhat different. Here, every surface (even a completely *White* one) is occupied by exactly one basic color dot at each position, and the printing process is viewed as that of replacing dots of one basic color by those of another. The beauty of such a view is that the whole process is acknowledged as being totally symmetric with respect to the basic colors used to render a particular color. This change in view-point is also reflected in the manner in which the barycentric dither screen is constructed, i.e. the geometry of the control path relative to the space.

Multi-State Color Printing: The research presented in this report was focused on reducing the halftone noise present in images rendered with 8-color printers. Recent advances in inkjet technology have resulted in printers that can place dots having colors located inside the *RGB* cube, in addition to the eight colors at its vertices. Barycentric screening may be modified so as to give a solution to the control of such printers: Instead of partitioning the color space into six tetrahedra as we have done, one must first partition the gamut of the printer into (a probably larger number of) tetrahedra, each of which is a convex hull of 4 dot colors. Once this is done, the application of barycentric screening is quite straightforward. A similar partition of a multi-state printer gamut into cubes (in order to facilitate Cartesian dithering) is in general impossible, and is certainly not natural. An open problem that needs to be solved for such applications, relates to the possibility of color space simplex-partition which renders itself to stitching.

7 References

[1] Q. Lin, and J. Allebach, "Color FM Screen Design Using DBS Algorithm", in *Proc. SPIE Electronic Imaging*, Vol. 3300, 1998.

[2] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, volume 4 of *Studies in Mathematics and its Applications*, North-Holland, Amsterdam, 1978.

[3] C. Gotsman and J. P. Allebach, "Bounds and algorithms for dither screens", Technical Report HPL-96-28, Hewlett-Packard Laboratories, February 1996.

[4] J. Hoschek and D. Lasser, *Fundamentals of Computer Aided Geometric Design*, A. K. Peters, Wellesley, Massachusetts, 1993. Translated by L. L. Schumaker.

[5] D. Shaked, N. Arad, A. Fitzhugh, and I. Sobel, "Color-diffusion: Tailoring error-diffusion to the world of color", Technical Report HPL 96-129, Hewlett-Packard Laboratories, August 1996. Proceedings version in *Proc. of SPIE Electronic Imaging*, Vol. 3648, January 1999.

[6] D. Shaked, N. Arad, A. Fitzhugh, and I. Sobel, "Ink relocation for color halftones", Technical Report HPL 96-128, Hewlett-Packard Laboratories, August 1996. Proceedings version in *Proc. of IS&T's Image Processing Image Quality and Image Capture Conf.*, May 1998.

[7] A. Zakhor, S. Lin, and F. Eskafi, "A new class of B/W and color halftoning algorithms", in *Int. Conference on Acoustics, Speech and Signal Processing*, 1991.