Hardcopy Image Barcodes Via Block Error Diffusion

Niranjan Damera-Venkata, *Member*, *IEEE*, Jonathan Yen, Vishal Monga, *Student Member*, *IEEE*, and Brian L. Evans, *Senior Member*, *IEEE*

N. Damera-Venkata and J. Yen are with Imaging Systems Laboratory, HP Laboratories, 1501 Page Mill Road, Palo Alto, CA 94304-1126. E-mail: {damera,jyen}@exch.hpl.hp.com. N. Damera-Venkata conducted some of this research while at UT Austin. V. Monga and B. L. Evans are with the Embedded Signal Processing Laboratory, The Univ. of Texas, Austin, TX 78712. E-mail: {vishal,bevans}@ece.utexas.edu.

N. Damera-Venkata and B. L. Evans were supported by a US National Science Foundation CAREER Award under Grant MIP-9702707.

Abstract

Error diffusion halftoning is a popular method of producing frequency modulated (FM) halftones for printing and display. FM halftoning fixes the dot size (e.g. to one pixel in conventional error diffusion), and varies the dot frequency according to the intensity of the original grayscale image. We generalize error diffusion to produce FM halftones with user-controlled dot size and shape by using block quantization and block filtering. As a key application, we show how block error diffusion may be applied to embed information in hardcopy using dot shape modulation. We enable the encoding and subsequent decoding of information embedded in the hardcopy version of continuous-tone base-images. The encoding-decoding process is modeled by robust data transmission through a noisy print-scan channel that is explicitly modeled. We refer to the encoded printed version as an image barcode due to its high information capacity that differentiates it from common hardcopy watermarks. The encoding/halftoning strategy is based on a modified version of block error diffusion. Encoder stability, image quality vs. information capacity tradeoffs, and decoding issues with and without explicit knowledge of the base-image are discussed.

EDICS category— QUAN Quantization and Halftoning

Index terms— halftoning, information embedding, hardcopy security, barcodes

Contact— Prof. Brian L. Evans, 1 University Station C0803, The University of Texas, Austin, TX 78712 USA. Voice: +1-512-232-1457, Fax: +1-512-471-5907, bevans@ece.utexas.edu

I. Introduction

Digital image halftoning quantizes a grayscale image to one bit per pixel for display and printing on binary devices. The goal of digital halftoning is to produce, via a clever distribution of binary dots, the illusion of continuous tone. Digital halftoning may be classified into three categories— amplitude modulated (AM), frequency modulated (FM), and AM-FM hybrid. In AM halftoning, the dot size is varied depending on the graylevel value of the underlying grayscale image while the dot frequency is held constant, e.g. conventional clustered-dot ordered dither screening. FM halftones have a fixed dot size, but the frequency of the dots varies with the graylevel of the underlying grayscale image. Conventional digital FM halftones have a fixed dot size of one pixel, e.g. those produced by dispersed-dot ordered dither and error diffusion [1]. AM-FM halftones [2], [3], [4], [5], [6], [7] are hybrids that allow both dot size and dot frequency to be varied in order to represent the underlying grayscale image.

In this paper, we generalize conventional error diffusion halftoning to produce FM halftones

with user-controlled dot size and shape [8]. We replace a pixel in conventional error diffusion with a pixel block. In a pixel block, the quantization error at each pixel is diffused to pixels in neighboring blocks in selected proportions. Hence, an entire block of quantization error is diffused at a time. The generated FM halftones can be designed to have very low dot size/shape variation, and the dot spacing is modulated depending on the underlying grayscale image. Unlike the aforementioned FM and AM-FM halftoning methods, the proposed block error diffusion framework provides *explicit control* over the dot shape.

The idea of using block structures in error diffusion to generate clustered dot halftones is not new. Fan [9] describes a block-based error diffusion algorithm that combines traditional clustered-dot dithering and block error diffusion to reduce the contouring observed when using traditional ordered dither screens with very few levels. While we also quantize pixel-blocks, the key difference between Fan's work and ours is that Fan produces ordered dither halftones while we produce error diffused halftones with user-defined minority dot shapes. Fan's method [9] is targeted at reducing contouring in traditional ordered dither screens. In contrast, our method modulates inter-dot distances between user-defined dot shapes to achieve tone reproduction. Hence, our method is a block-FM halftoning method whose output resembles error diffusion. We also formalize the block-based mathematics in terms of linear algebra, which Fan's paper did not address. Eschbach [10] proposes an error diffusion technique to generate clustered dot halftones using block threshold modulation. However, unlike our approach, Eschbach's approach does not quantize a pixel-block at a time, nor does it diffuse a block of error. Instead, it is a scalar error diffusion system using threshold modulation that is block-periodic.

We use the dot shape control built into our proposed block error diffusion framework to modulate dot shapes within an image according to an *information* signal. Thus, information may be encoded into the hardcopy version of an original image [11]. This is a key application of the block error diffusion framework in which high information capacity is required.

Significant attention has been devoted to hardcopy watermarking by injecting watermarks into the halftoned image [12], [13], [14], [15], [16], [17], [18]. The methods in [12]–[14] are not practical for typical print-scan channels. The method in [15] employs a search over several halftone patterns, in a way similar to direct binary search [19], and is therefore very slow for real-time printing applications. While the methods in [16], [17], [18] are practical, their aim is hardcopy authentication and not high rate information embedding. Data hiding for high-quality

watermarking using scalar error diffusion has been explored in [20]. The advantage of using block-based dot shape modulation to embed information is that it provides better image quality when high information capacity is required. For example, the 2×2 block encoding method that we describe in Section V embeds data at a maximum rate of 1 bit/pixel. Scalar encoders such as [20] typically work at infomation rates < 0.5 bits/pixel. Clearly, if such an encoder operates at 1 bit/pixel and the message string is all $1's$, the output will also be all $1's$ bearing no resemblance to the original grayscale image. Block-based encoders also benefit from more robust decoder processing, since pixel-level statistics from all pixels within a block are pooled to make a decision on what codeword was used. Threshold image quality that enables the image to be recognized by a human is sufficient for our application. For this reason, we refer to the hardcopy image with information embedded in it as an image barcode (IBC).

In this paper, we show how to encode information into a continuous-tone base-image, which is then printed using a conventional inkjet or laserjet printer. The printed image is scanned via a scanner and the resulting digital image is processed by a decoder to recover the transmitted message. The decoder may or may not have access to the original base-image. The information capacity of an image barcode lies between conventional 2-D barcode technologies such as PDF-117 and hardcopy watermarks. In a typical application for example, we could embed the entire biography of an individual onto the photograph on his/her business card. Further, small mp3 clips and executables may also be embedded into hardcopy images providing a rich media experience.

Section II introduces the notation used in the paper. Section III formulates the framework of block error diffusion. Section IV demonstrates the design of the block error filter from well known scalar error filter prototypes. We also discuss how FM halftones with user-controlled dot shape and size may be produced. Section V comprehensively describes a practical image communication system that encodes information into image barcodes using the block error diffusion framework. We show how the information embedded in the hardcopy images may be robustly decoded by processing the scanned image barcode. Information capacity and encoder stability are also analyzed. Section VI summarizes the contributions of this paper and presents future research directions.

II. Notation

In this paper, we process vector-valued sequences using multifilters [21]. A multifilter is a filter with matrix-valued coefficients. Vector-valued sequences arise from grouping pixels in the input

grayscale image to be halftoned into blocks of $N \times M$ pixels (rectangular blocks). We may order each block of MN samples by row, into an $MN \times 1$ vector to form an image of vectors.

Letting $\mathbf{x}$ be an image of vectors, with each vector having $MN \times 1$ elements (pixel values), the z-transform of $\mathbf{x}$ is

$$\mathbf{X}(z_1, z_2) = \sum_{m_1} \sum_{m_2} \mathbf{x}(m_1, m_2)\, {z_1}^{-m_1}\, {z_2}^{-m_2} \tag{1}$$

We may filter the vector-valued image using a multifilter. A multifilter with $K \times K$ support can be represented by a $K \times K$ sequence in which each sample is a $MN \times MN$ matrix. The z-transform of the matrix-valued filter (a.k.a. multifilter) sequence $\tilde{\mathbf{h}}$ is

$$\tilde{\mathbf{H}}(z_1, z_2) = \sum_{k_1=0}^{K-1} \sum_{k_2=0}^{K-1} \tilde{\mathbf{h}}(k_1, k_2)\, {z_1}^{-k_1}\, {z_2}^{-k_2} \tag{2}$$

An alternate representation of a multifilter with $K \times K$ support and $MN \times MN$ matrix-valued coefficients uses the $MN \times K^2MN$ matrix

$$\tilde{\mathbf{\Gamma}} = \left[\tilde{\mathbf{h}}'(0) \mid \tilde{\mathbf{h}}'(1) \mid \ldots \mid \tilde{\mathbf{h}}'(K^2 - 1)\right] \tag{3}$$

where $\tilde{\mathbf{h}}'(0), \tilde{\mathbf{h}}'(1), \ldots, \tilde{\mathbf{h}}'(K^2-1)$ are the coefficients of the matrix-valued filter $\tilde{\mathbf{h}}$ ordered by rows. The filtering operation of a multifilter $\tilde{\mathbf{h}}$ with input $\mathbf{x}$ is given by the matrix-vector convolution

$$\mathbf{y}(m_1, m_2) = \sum_{k_1=0}^{K-1} \sum_{k_2=0}^{K-1} \tilde{\mathbf{h}}(k_1, k_2)\, \mathbf{x}(m_1 - k_1, m_2 - k_2) \tag{4}$$

where $\mathbf{y}$ represents the output image of $MN \times 1$ vectors. In the z-domain, the matrix-vector convolution becomes a linear transformation by an $MN \times MN$ transformation matrix given by

$$\mathbf{Y}(z_1, z_2) = \tilde{\mathbf{H}}(z_1, z_2)\, \mathbf{X}(z_1, z_2) \tag{5}$$

For a scalar signal $x(m)$, we denote its z-transform by $X(z)$. We use $\mathbf{m}$ to denote the 2-D index (m_1, m_2), and $\mathbf{k}$ to denote the 2-D index (k_1, k_2).

III. Block Error Diffusion

Fig. 1 shows a block diagram for block error diffusion. Although the block diagram resembles conventional error diffusion halftoning, there are several key differences. The input is an $N \times M$ block of pixels (called a pixel-block) as opposed to a single pixel in conventional error diffusion. We consider each block to be ordered into an MN-element vector as discussed in Section II.

The quantizer output for each pixel in a pixel-block is exactly one element from the discrete set $\mathcal{O} = \{0, 1\}$. Here, 0 represents black and 1 represents white. We may quantize each pixel-block using a simple scalar quantizer or a vector quantizer. In the case of scalar quantization, the quantizer is defined by

$$\mathbf{Q}(\mathbf{u}) = \begin{pmatrix} Q(u_1) \\ Q(u_2) \\ \vdots \\ Q(u_{MN}) \end{pmatrix} \tag{6}$$

where

$$Q(u_i) = \begin{cases} 1 & u_i \geq \frac{1}{2} \\ 0 & u_i < \frac{1}{2} \end{cases} \tag{7}$$

Here, u_i refers to the i^{th} pixel value in a $N \times M$ pixel block and hence i varies from 1 to MN. The output (quantization) levels are chosen to be 0 and 1 for the convenience of having midgray at $\frac{1}{2}$. In Fig. 1, the filter in the feedback loop has matrix-valued coefficients. The filter operates on the quantization error sequence $\mathbf{e}(\mathbf{m})$ to produce the feedback signal

$$\mathbf{f}(\mathbf{m}) = \sum_{\mathbf{k} \in \mathcal{S}} \tilde{\mathbf{h}}(\mathbf{k})\, \mathbf{e}(\mathbf{m} - \mathbf{k}) \tag{8}$$

where $\mathbf{m}$ and $\mathbf{k}$ are two-dimensional index vectors, $\tilde{\mathbf{h}}(\cdot)$ is an $MN \times MN$ matrix-valued sequence, and $\mathcal{S}$ is the filter support. In this paper, we assume four-tap filter support defined by (horizontal, vertical) offsets to the current pixel being processed as $\mathcal{S} = \{(0, 1), (1, 0), (1, 1), (1, -1)\}$ unless otherwise specified. The input to the quantizer is given by

$$\mathbf{u}(\mathbf{m}) = \mathbf{x}(\mathbf{m}) - \mathbf{f}(\mathbf{m}) \tag{9}$$

In terms of block filtering the operation defined by (8) can be described with the help of Fig. 2, which illustrates a block error filter operating on pixel-blocks of 2×2 pixels. The output pixel-block is computed by forming four different linear combinations of all 16 pixels in the pixel-block mask. Each linear combination produces a single output pixel of the output pixel-block. The next section shows how the matrix-valued coefficients of an error filter may be designed to promote user-defined minority pixel clustering in the generated halftone image.

IV. FM halftoning via block error diffusion

The block error filter in the feedback loop governs how quantization error is diffused to the neighboring pixel-blocks. For conventional error diffusion, one only needs to decide how much

of the quantization error is to be diffused to each neighboring pixel under the constraint that all the quantization error be diffused. The same applies in block error diffusion, for which the constraints become

$$\tilde{\mathbf{\Gamma}}\, \mathbf{1}_{MNK^2 \times 1} = \mathbf{1}_{MN \times 1} \tag{10}$$

$$\tilde{\mathbf{\Gamma}} \geq 0 \tag{11}$$

where $\mathbf{1}_{r \times c}$ represents an $r \times c$ matrix of all one entries. These conditions correspond to the assertion that the elements of the matrix-valued error filter coefficients be non-negative and that their rows sum to unity. Section IV-A designs error filters for block error diffusion. Section IV-C generates FM halftones using block error diffusion.

A. Error filter design

In designing the error filter coefficients $\tilde{\mathbf{\Gamma}}$ in (3), we map the coefficients of a conventional error filter into the corresponding block filters. If we start with a scalar filter with the same support as the multifilter or block filter, and represent its coefficients by the row vector $\tilde{\gamma}$, where

$$\tilde{\gamma} = \Big[g'(0) \mid g'(1) \mid \ldots \mid g'(K^2 - 1) \Big] \tag{12}$$

then a multifilter $\tilde{\mathbf{\Gamma}}$ may be derived from it as

$$\tilde{\mathbf{\Gamma}} = \tilde{\gamma} \bigotimes \tilde{\mathbf{D}} \tag{13}$$

where $\bigotimes$ denotes the Kronecker product operation and $\tilde{\mathbf{D}}$ is an $MN \times MN$ *diffusion matrix*. Since the elements of $\tilde{\gamma}$ are the coefficients of a conventional error filter, they are non-negative and sum to one. Thus, to satisfy the constraints imposed by (10), the diffusion matrix must satisfy the constraints

$$\tilde{\mathbf{D}}\, \mathbf{1}_{MN \times 1} = \mathbf{1}_{MN \times 1} \tag{14}$$

$$\tilde{\mathbf{D}} \geq 0 \tag{15}$$

Thus, by imposing structure on $\tilde{\mathbf{\Gamma}}$, we only need to design the $MN \times MN$ diffusion matrix $\tilde{\mathbf{D}}$.

The decomposition of (13) is a natural and intuitive way of designing suitable error filters to generate FM halftones via block error diffusion. The physical meaning of deriving the block filter from a given conventional error filter via (13) is that the quantization error incurred at the current pixel block is diffused to the neighboring pixel-blocks in the same proportions that a

conventional error filter diffuses error to its neighboring pixels. The diffusion matrix $\tilde{\mathbf{D}}$ governs the proportions to which errors are distributed within the pixels of a block. According to our proposed structure, these proportions are constant, and are independent of the relative position of the pixel-blocks to which errors are diffused. This enforces a local isotropy constraint, by which we mean that no pixel within a pixel block is given preference over other pixels within the same block. The constraints on the diffusion matrix simply indicate that *all* of the quantization error that is diffused to a pixel block must be diffused among pixels that compose the block. Thus, the pixel-blocks in the block-error diffusion framework are made to behave like pixels in conventional error diffusion and the block errors are diffused in much the same way as pixel errors in conventional error diffusion.

B. FM halftoning with rectangular dots

By using block error diffusion, we show how to produce FM-halftones with dot clusters that are greater than one pixel in size. One method of achieving dot-clustering would be to halftone a downsampled version of the grayscale image and then replicate pixels to obtain a halftone of the same size as the grayscale image. For example, if we want 2×2 minority pixel dot clusters, then we could filter the original grayscale image with a halfband filter, downsample the original grayscale image by retaining every other sample in the horizontal and vertical directions, halftone the downsampled grayscale image, and interpolate the halftone to the resolution of the original grayscale image by pixel replication. This process is identical to halftoning the filtered image after replicating the upperleftmost sample in each block to all samples and using the identity diffusion matrix $\tilde{\mathbf{D}} = \tilde{\mathbf{I}}_{4\times 4}$. Fig. 3 shows an example halftone generated by this method. The method first filters the original image (to prevent aliasing) and then performs downsampling. The downsampled image is then halftoned using conventional error diffusion. Pixel clustering is then induced by replicating each pixel to form a pixel-block. The spatial resolution of the example halftone suffers due to the pixel replication and pre-filtering.

Our approach to FM halftoning relies on forming minority pixel dot clusters by diffusing the quantization error from each pixel block equally to all samples within the neighboring pixel blocks. The error diffused to each block within the block error filter mask will, however, be unequal since it is governed by the corresponding conventional error filter coefficients $\tilde{\gamma}$. Thus

for 2×2 pixel clusters, we use the diffusion matrix

$$\tilde{\mathbf{D}} = \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \tag{16}$$

In general, for an $N \times M$ pixel block, the diffusion matrix will take the form $\frac{1}{MN}\tilde{\mathbf{1}}_{MN \times MN}$ where $\tilde{\mathbf{1}}_{MN \times MN}$ is an $MN \times MN$ matrix with all its elements equal to 1. The motivation for using this diffusion matrix is that the error at any sample within the current pixel-block that is diffused to an adjacent pixel-block will be spread to all of the samples within the pixel-block equally. The quantization decisions of all pixels within the modified pixel-block will be biased in the same direction. Intuitively, this should result in the halftoned samples of that pixel-block organizing themselves into a pixel-block cluster.

Fig. 4 shows halftones obtained by using block error diffusion with the diffusion matrix given by (16). Fig. 4(a) uses $\tilde{\gamma} = \frac{1}{16}[1\ 5\ 3\ 7]$, and Fig. 4(b) uses $\tilde{\gamma} = \frac{1}{48}[1\ 3\ 5\ 3\ 1\ 3\ 5\ 7\ 5\ 3\ 5\ 7]$, which correspond to the well-known Floyd-Steinberg [1] and Jarvis [22] error filters, respectively. The support for the Floyd-Steinberg and Jarvis error filters are shown in Fig. 5(a) and 5(b), respectively. For the rest of the paper, we fix $\tilde{\gamma} = \frac{1}{16}[1\ 5\ 3\ 7]$. There is no need to use a pre-filter to prevent spatial aliasing. From visual inspection, the spatial resolution of the grayscale image is not compromised, and the dots are clustered in 2×2 blocks. The halftones even exhibit sharpening, which is characteristic of conventional Floyd-Steinberg and Jarvis error diffusion [23].

Using the method described above, it is possible to cluster the halftone dots into rectangular dots of desired size. Halftones with dot clusters of 3×2 and 2×3 are produced using 6×6 diffusion matrices having all their elements equal to $\frac{1}{6}$. Figs. 6(a) and 6(b) show halftones with rectangular dot clusters of size 3×1 and 1×3 respectively.

C. FM halftoning with user defined shapes

The simple block error diffusion framework described in Section IV may be modified, to produce FM halftones with user defined shapes. This is accomplished by extending the scalar quantization equations (6) and (7) to a vector form.

In this case the quantizer is defined by:

$$Q(\mathbf{u}) = \begin{cases} \mathbf{S} \otimes \mathbf{1}_{MN\times 1} & if \frac{1}{MN}\sum_{i=0}^{MN-1} u_i(\mathbf{m}) \geq \frac{1}{2} \\ \mathbf{S} \otimes \mathbf{0}_{MN\times 1} & if \frac{1}{MN}\sum_{i=0}^{MN-1} u_i(\mathbf{m}) < \frac{1}{2} \end{cases} \tag{17}$$

The vector function $\mathbf{S}$ defines the desired dot shape and $\otimes$ is the bitwise XOR operator. For 'L' shapes using 2×2 pixel-blocks we have $\mathbf{S} = [1\ 0\ 1\ 1]^T$. For 'T' shaped dots using 3×3 pixel-blocks we have $\mathbf{S} = [1\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 0]^T$.

Since we constrain the normal thresholding process in block error diffusion based on the dot shape, it is possible that the error might become become large resulting in unstable behavior and degradation in image quality. Fig. 7 illustrates instability in block error diffusion when a 'multiply' shape is used in the quantization step. This effect occurs since the algorithm is not able to 'catch up' with the error due to quantizing a pixel-block with a given shape.

To mitigate unstable behavior, dot shapes should not be enforced at every location. For example, it is possible to enforce the dot shape quantization given by (17) only at minority pixel-block locations, while (6) and (7) are used to quantize the majority pixel-block locations. Minority pixel-block locations $\mathcal{C}$ are determined by the following conditions.

$$\left.\begin{array}{lll} \frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \geq \frac{1}{2} & AND & \frac{1}{MN}\sum_{i=0}^{MN-1} u_i(\mathbf{m}) < \frac{1}{2} \\ \frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) < \frac{1}{2} & AND & \frac{1}{MN}\sum_{i=0}^{MN-1} u_i(\mathbf{m}) \geq \frac{1}{2} \end{array}\right\} \Longrightarrow \mathbf{m} \in \mathcal{C} \tag{18}$$

Figs. 9(a) to 9(d) show the pixels within a pixel-block (shaded) that are part of the dot shapes corresponding to the halftones of Figs. 8(a) to 8(d). The dot shape constraint has been enforced only on minority pixel-blocks as determined by equation (18). Clearly unstable behavior is not seen in these halftones.

For $M \times N$ blocks, using a dot shape $\mathbf{S}$ with an average intensity of $\alpha < 0.5$ the following stability result holds.

Stability result 1: *If every continuous-tone image block* $\mathbf{x}(\mathbf{m})$ *satisfies the condition* $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \in [\alpha, 1-\alpha]$, *then block error diffusion is stable.*

Proof: The proof of this result follows by induction. If $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \in [\alpha, 1-\alpha]$, where $\alpha \in [0, 0.5]$. Then the average quantization error for the first image block is $\frac{1}{MN}\sum_{i=0}^{MN-1} e_i(\mathbf{m}) \in [-\alpha, \alpha]$. This is true as the average of the output image block lies in the set $\{\alpha, 1-\alpha\}$. Since the

error filter coefficients sum to unity the feedback signal $f(\mathbf{m}+\delta)$ at the next scan location lies in the range $[-\alpha, \alpha]$. Since $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}+\delta) \in [\alpha, 1-\alpha]$, the modified input block at the next scan location $\mathbf{u}(\mathbf{m})$ satisfies $\frac{1}{MN}\sum_{i=0}^{MN-1} u_i(\mathbf{m}) \in [0,1]$ which results in $\frac{1}{MN}\sum_{i=0}^{MN-1} e_i(\mathbf{m}+\delta) \in [-\alpha, \alpha]$. Thus, the average error is bounded and hence the block error diffusion is stable. □

The above result is a sufficient condition to obtaining stable results from block error diffusion with user defined dot shapes. Equations (6) and (7) could be used to quantize the pixel-block locations when $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \notin [\alpha, 1-\alpha]$.

We may use block error diffusion with different user-defined dot shapes to encode information into a printed image. Here the user defined dot shapes function as codewords. An illustration of this application is presented in the next section.

V. Image Barcodes

Section V-A introduces the image barcode (IBC) encoder. We show that the proposed algorithm is able to represent any continuous tone variation in a stable manner and is able to trade image quality for increased capacity. Section V-B discusses how a scanned image barcode may be decoded when the original grayscale base image is either explicitly known at the decoder or is unknown at the decoder. We refer to the former case as guided decoding and the latter case as blind decoding.

A. Encoding information into an Image Barcode

A.1 *Image Barcode Encoder*

Fig. 12 shows the system block diagram for the encoding of information into an image barcode. The message is encoded into the output halftone $\mathbf{w}(\mathbf{m})$ using codewords corresponding to different dot shapes.

1. The original image is divided into blocks. The blocks need not be rectangular, but must tile to cover the entire image.
2. The processing of the blocks usually proceeds in raster or serpentine scan order. At each block, a decision is made to represent that block with either a block with all pixels equal to 1 (white) or another with all pixels equal to 0 (black). These blocks make up the intermediate halftone image B'. The decision to allow a black block or a white block in the intermediate halftone is made by simply thresholding the average modified input block value (modified

by past errors).

3. The current message code word is modulated onto the binary block using XOR modulation. The modulated blocks make up the encoded image $\mathbf{w}(\mathbf{m})$. Fig. 11 shows codeword alphabet $C = \{[0\ 0\ 0\ 1]^T, [0\ 0\ 1\ 0]^T, [0\ 1\ 0\ 0]^T, [1\ 0\ 0\ 0]^T\}$ used to encode 2×2 intermediate halftone blocks. The last two steps are equivalent to equation (17).
4. The quantization error between the modulated binary result and the current graylevel block is diffused to neighboring unprocessed graylevel blocks using a block error filter with a diffusion matrix $D = \frac{1}{4}\mathbf{1}_{4\times 4}$, which corresponds to diffusing the average block error to neighboring unprocessed pixel blocks.
5. The next graylevel block in the scan path is considered and steps 2–5 are repeated until all image pixel blocks have been processed.

The resulting encoded block obtained by modulating the message onto an intermediate halftone pixel-block, is always equal to the given codeword or its complement. Fig. 11 shows the possible encoded image blocks in this case.

The following equations describe the image barcode encoding process using 2×2 block encoding. The quantizer $\mathbf{Q}(\cdot)$ converts the modified input block $\mathbf{u}(\mathbf{m})$ into the intermediate halftone block $\mathbf{b}'(\mathbf{m})$.

$$\mathbf{b}'(\mathbf{m}) = \mathbf{Q}(\mathbf{u}(\mathbf{m})) = \begin{cases} [1\ 1\ 1\ 1\ ]^T & \frac{1}{4}\sum_{i=0}^{3} u_i(\mathbf{m}) > \frac{1}{2} \\ [0\ 0\ 0\ 0\ ]^T & else \end{cases} \tag{19}$$

The codeword $\mathbf{c}(\mathbf{m})$ is modulated onto the intermediate halftone using XOR modulation.

$$\mathbf{e}(\mathbf{m}) = \mathbf{c}(\mathbf{m}) \otimes \mathbf{b}'(\mathbf{m}) \tag{20}$$

The resulting quantization error block $\mathbf{e}(\mathbf{m})$ is diffused using a block error filter $\tilde{\mathbf{h}}$ with matrix valued coefficients. We use the block error filter with coefficients

$$\begin{aligned} \tilde{\mathbf{h}}(0,1) &= \frac{7}{64}\mathbf{1}_{4\times 4} \\ \tilde{\mathbf{h}}(1,1) &= \frac{1}{64}\mathbf{1}_{4\times 4} \\ \tilde{\mathbf{h}}(1,0) &= \frac{5}{64}\mathbf{1}_{4\times 4} \\ \tilde{\mathbf{h}}(1,-1) &= \frac{3}{64}\mathbf{1}_{4\times 4} \end{aligned}$$

This corresponds to diffusing the average error using the Floyd-Steinberg weights and distributing the error diffused to a pixel-block equally to all elements within the pixel-block. To ensure

robust transmission the codeword bitstream is mapped to a more redundant bitstream using error correcting codes. For example, if a $16 \rightarrow 31$ BCH code were used, the actual information capacity per block is halved. Fiducial marks are inserted at the corners to keep the decoder in alignment. If the target output resolution is 100 dpi and if the original image is 100x100 pixels it will be rendered in print in a 1 inch $\times$ 1 inch square. Since the native resolution of the printer is 600 dpi, each encoded halftone pixel is replaced by a 6×6 block by pixel replication before printing. This ensures that the effective resolution of the print is 100 dpi.

A.2 *Stabilizing the Image Barcode Encoder*

For $M \times N$ blocks, with the IBC encoder using codewords with average intensity of $\alpha_i < 0.5$ the following stability results hold.

Stability result 2: *If every continuous-tone image block* $\mathbf{x}(\mathbf{m})$ *satisfies the condition* $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \in [\alpha_{max}, 1-\alpha_{max}]$*, then the encoder is stable.*

Proof: The proof of this result is a direct application of Stability result 1 of Section IV-C applied to the worst case scenario when $\alpha = \alpha_{max}$ and $\alpha_m ax \in [0, 0.5]$. $\square$

We select the codewords to all have the same average intensity but represent different shapes or different orientations of a canonical shape. To deal with pixel-blocks where $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \notin [\alpha, 1-\alpha]$, we may code them with appropriately chosen fictitious codewords, assuming that these blocks need to be processed by the decoder. If we had prior knowledge of where these blocks occur, we could use any function to quantize these blocks since they would not be processed by the decoder. Without prior knowledge of the original image, the decoder must process all pixel-blocks.

Stability result 2: *If the fictitious codeword* $\mathbf{c}_f = [0\,0\,0\,0]^T$ *is used whenever* $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \notin [\alpha, 1-\alpha]$ *then the encoder is stable for all continuous-tone inputs in the range* $[0, 1]$*.*

Proof:

The proof of this result follows from the fact that the fictitious code word $\mathbf{c}_f$ simply reproduces the intermediate halftone block as the output encoded halftone block. It is a do-nothing code that does not embed any information. The proof is similar to the one presented above, and follows by induction.

If $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \in [0, 1]$, then the average quantization error for the first image block is $\frac{1}{MN}\sum_{i=0}^{MN-1} e_i(\mathbf{m}) \in [-\frac{1}{2}, \frac{1}{2}]$. This is true as the average of the output encoded image block lies

in the set $\{0, \alpha, \alpha, 1\}$ irrespective of the particular codeword that is used. Since the error filter coefficients sum to unity, the feedback signal $f(\mathbf{m}+\delta)$ at the next scan location lies in the range $[-\frac{1}{2}, \frac{1}{2}]$. Further, since $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}+\delta) \in [0,1]$, the modified input block at the next scan location $\mathbf{u}(\mathbf{m})$ satisfies $\frac{1}{MN}\sum_{i=0}^{MN-1} u_i(\mathbf{m}) \in [-\frac{1}{2}, \frac{3}{2}]$, which results in $\frac{1}{MN}\sum_{i=0}^{MN-1} e_i(\mathbf{m}+\delta) \in [-\frac{1}{2}, \frac{1}{2}]$. Hence, the average error due to the encoding is bounded and the encoder is stable for all continuous tone input images when fictitious codewords are allowed. □

In the next section, we show how fictitious codewords may also be used to trade information capacity for image quality.

A.3 *Trading information capacity for image quality*

Ignoring fictitious codewords the information capacity of an image barcode using the codeword set in Fig. 11 is 2 bits per block. Typically half of this capacity is not actually used to embed message codewords due to error correction coding. This reduces the effective capacity to 1 bit per pixel block. If fictitious codewords are used, then the information capacity is image dependent and is given by

$$\text{Capacity (in bits)} = (T - F) \times BPP \times ECC \tag{21}$$

where T is the total number of image blocks, F is the total number of image blocks coded with fictitious codes, BPP is the bits encoded by the encoding alphabet per block, and ECC is the loss fraction due to error control coding. At each block location one has the choice of encoding information or not encoding information by using fictitious codewords. If a distortion metric is used to determine which blocks use fictitious codewords, then the image quality is enhanced at these locations since no information is embedded. Note that it is not necessary that we encode every input block with $\frac{1}{MN}\sum_{i=0}^{MN-1} x_i(\mathbf{m}) \notin [\alpha, 1-\alpha]$ with a fictitious codeword, although this would be sufficient to guarantee encoder stability. We could encode information in these regions at a low embedding rate using for example, a stochastic pattern that encodes information in $10\% - 30\%$ of such pixels. This gives the encoder sufficient opportunity to catch up with the accumulated error in encoding such a pixel-block with a codeword. Fig. 10(d) shows an example 216×198 image barcode with 1.2 kB of embedded biography information. The actual rendering resolution is 150 dpi.

If 2×2 blocks are used to encode 4 different codewords, then 2 bits are encoded per base image block. Note that in theory, we can use more than one distinct codeword block to encode

a single unique codeword to improve image quality in the encoding process. For example if we were using 2×2 blocks and encoding 2 bits/block, using 4 codeword templates with all black pixels except one, we can improve quality by adding 4 vertical/horizontal edge templates.

In the following sub-section we discuss how the information embedded within an image barcode may be recovered from a scanned version.

B. Decoding Image Barcodes

Our image barcode decoder uses several components of the visually significant barcode (VSB) technology developed by Shaked *et al.* [24]. The VSB technology is a sophisticated image processing pipeline to enable efficient hardcopy information encoding on bi-level base images, such as company logo images. We make use of the print-scan channel model developed by Shaked *et al.* for visually significant barcodes.

The IBC decoder operates on a scanned version of the encoded hardcopy image. Let us assume that the scanning resolution is 600 dpi. Then an observed 6×6 block corresponds to one encoded halftone pixel. The decoder first identifies fiducial marks to enable the decoder to compensate for global geometric distortions such as global translation, rotation and skew due the the print scan process. This step involves corner detection, estimation and application of a global geometric transform followed by bilinear interpolation. The result of this step is a rectangular image. Before the image barcode may be decoded using maximum likelihood detection theory, the halftone dots in the encoded image must be matched up with corresponding grayscale observations in the scanned image. The IBC accounts for local space deformations arising due to the fact that dots corresponding to certain co-ordinates in the original image are located at different points in the copy. Most deformations were observed to be approximately separable, i.e., a dot expected at $(x_0, \cdot)$ is located at $(x_0 + \Delta_x, \cdot)$, and similarly a dot expected at $(\cdot, y_0)$ is located at $(\cdot, y_0 + \Delta_y)$. Row and column interfaces are aligned with pixel rows and columns, and the deformation is expressed only in their uneven distribution. The sum of absolute values of horizontal gradients in columns is used to determine the interface at columns (from peaks in the gradient sum). A similar procedure is used to find row interfaces. Dots are then virtually aligned by augmenting the scanned image with a list (describing a a potentially non-uniform grid) of dot centers.

Once the scanned barcode has been aligned with the encoded halftone, there is a correspondence between a measurement patch and an encoded halftone pixel. A linear discriminant y is formed by multiplying the elements of the patch pointwise by a truncated Gaussian and summing.

This reduces the dimensionality of the observations per encoded halftone pixel to unity. From an empirical analysis of the shape of the discriminant histogram for several inkjet and laserjet printers, Shaked *et al.* [24] propose the following asymmetric Laplacian probability model for the observations

$$P(y|0) = \begin{cases} \alpha_{0L} e^{-\alpha_{0L}(\mu_0 - y)} & y < \mu_0 \\ \alpha_{0R} e^{-\alpha_{0R}(\mu_0 - y)} & y > \mu_0 \end{cases} \tag{22}$$

$$P(y|1) = \begin{cases} \alpha_{1L} e^{-\alpha_{1L}(\mu_1 - y)} & y < \mu_0 \\ \alpha_{1R} e^{-\alpha_{1R}(\mu_1 - y)} & y > \mu_0 \end{cases} \tag{23}$$

For each individual image the parameters $\{\alpha_{0L}, \alpha_{0R}, \alpha_{1L}, \alpha_{1R}, \mu_0, \mu_1\}$ are different and are estimated using an expectation maximization approach [24]. Once the parameters are estimated, a likelihood score is computed for each possible message symbol. Thus the decoded codeword at location $\mathbf{m}$ is given by

$$\hat{\mathbf{c}}(\mathbf{m}) = \underset{\{\mathbf{c} \in \mathcal{C}: \mathbf{w} = \mathbf{b}(\mathbf{m}) \otimes \mathbf{c}\}}{\arg\max} \prod_{i=0}^{N^2-1} P(y_i|w_i) \tag{24}$$

where $\mathbf{b}(\mathbf{m})$ represents the known bi-level base image block at location $\mathbf{m}$ and $\mathbf{c}$ represents a possible codeword block. Here, $w_i \in \{0, 1\}$ represents the encoded halftone value expected at location i if codeword $\mathbf{c}$ was used to encode block $\mathbf{b}(\mathbf{m})$.

First the corners are determined using the detected fiducial marks. Then a global geometric transform is applied and local shape deformations are compensated. The barcode is now ready for probablistic decoding. Two different scenarios for decoding exist depending on whether (or not) the decoder has knowledge of the continuous-tone base image. Section V-B.1 deals with the case when the continuous-tone base image is known at the decoder. Section V-B.2 describes the situation when the decoder has no information regarding the continuous-tone base image and must decode the information by estimating the bi-level intermediate halftone image directly from the scanned data.

B.1 *Image barcode decoding when base image is known*

Fig. 14 outlines the pipeline used to implement base-image guided decoding of image barcodes. To use the maximum likelihood decoding strategy of equation (24) as the probablistic decoding scheme we need to estimate the bi-level intermediate halftone image on which the encoding codewords were modulated. From Fig. 12 we see that in general the intermediate halftone B' depends not only on the continuous-tone base image but on the message codewords as well, due

to the feedback mechanism in the block error diffusion. However by proper choice of possible codewords we can ensure that the intermediate halftone in-fact depends only on the continuous-tone base image and not on a particular message codeword used. To use knowledge of the base image, the intermediate halftone we enforce the property that the result of an XOR operation with any of the codewords has the same average intensity. Thus an arbitrary message could be used to generate the intermediate halftone B' from the known continuous-tone base image B. Note that the locations of the fictitious codewords are known *a priori* from the continuous-tone base image, and they can be incorporated in the exact determination of the intermediate bi-level intermediate halftone image. Once the bi-level intermediate halftone image is determined, equation (24) is used after the Laplacian probability model given by (22) and (23) is determined from the scanned data to decode the message codewords.

B.2 *Blind decoding of image barcodes*

Fig. 14 outlines the pipeline used to implement blind decoding of image barcodes. The intermediate halftone must be estimated from the scanned image barcode without knowledge of the continuous tone base-image. The decoder must process every pixel block, in effect 'decoding' the fictitious codewords inserted by the encoder.

In practice the estimation of the bi-level intermediate halftone image must be performed from imperfect observations due to print-scan channel degradations and decoder preprocessing. After alignment there is a correspondence between a measurement patch and an encoded halftone pixel. A linear discriminant $y_i(\mathbf{m})$ is formed by multiplying the elements of the patch pointwise by a truncated Gaussian and summing. If 2×2 blocks were used in the encoding process, a new discriminant $Y(\mathbf{m})$ is formed by averaging the pixel-level discriminants $y_i(\mathbf{m})$. Thus

$$Y(\mathbf{m}) = \frac{\sum_{i=0}^{3} y_i(\mathbf{m})}{4} \tag{25}$$

Fig. 15 shows the histogram of a block-level discriminant Y. From the observation $Y(\mathbf{m})$ we need to determine the intermediate the intermediate halftone image block $\hat{\mathbf{b}'}(\mathbf{m})$. This may be done by deriving optimal thresholds to classify the codewords from the observations. This can be framed as a maximum-likelihood estimation problem. First a Gaussian mixture model is fit to the observations using the expectation maximization paradigm. The estimation of the

intermediate halftone blocks is then reduced to

$$\hat{\mathbf{b}'}(\mathbf{m}) = \arg\max_{\mathbf{b}' \in \mathcal{B}} P(Y|\mathbf{b}') \tag{26}$$

where $\mathcal{B}$ represents the set of allowed intermediate halftone patterns. The individual class conditional probability distributions $P(Y|\mathbf{b}')$ are Gaussians obtained from the Gaussian mixture model. After the intermediate halftone has been estimated, decoding proceeds by using equations (22) and (23) to determine the pixel-level Laplacian probability model and then finding the codeword used to encode a given intermediate halftone image block as:

$$\hat{\mathbf{c}}(\mathbf{m}) = \arg\max_{\{\mathbf{c} \in \mathcal{C} \bigcup \mathcal{C}_f : \mathbf{w} = \hat{\mathbf{b}'}(\mathbf{m}) \otimes \mathbf{c}\}} \prod_{i=0}^{3} P(y_i|w_i) \tag{27}$$

where $\mathcal{C}$ and $\mathcal{C}_f$ denote the set of information encoding and fictitious codewords respectively. Note that the fictitious codewords must be explicitly estimated in the blind decoding case while this was not required when the decoder had knowledge of the continuous-tone base image. Fig. 16(a) shows the actual intermediate halftone image. Fig. 16(b) shows the estimated intermediate halftone image from a scanned version of the image barcode.

B.3 *Decoder Performance*

We tested the performance of the image barcode decoder using around 25 test images printed on 3 different InkJet and LaserJet printers. The print resolution was 100 dpi (We replicated encoded halftone pixels prior to printing based on the native printer resolution to achieve this output resolution). A $16 \rightarrow 31$ BCH code was used for error correction. We scanned the output at 600 dpi. In practice we were able to obtain robust decoding $(90\% - 100\%)$ at resolutions upto 150 dpi when the base image is known. This corresponds to a scan to print resolution ratio of $4:1$. Robust blind decoding (without knowledge of the original) was achieved at resolutions up to 100 dpi. The decoding algorithm is robust to minor image rotations while scanning and a variety of image degradations. For example, a single line drawn with a pen across the image or minor dust did not affect the decoding. This is attributed to decoder pre-processing and the error correction codes inserted at the encoding stage.

Note that while it is easier to decode with the knowledge of the original image other tradeoffs exist in general.

VI. Conclusion

In this paper, we have introduced a general framework for producing FM and AM-FM halftones with user-controlled shape and size. Standard enhancement techniques used in conjunction with scalar error diffusion may be extended in a straightforward manner for use in the context of block error diffusion [6], [25], [26], [27], [28]. Within the block error diffusion framework, we have shown how information may be embedded into hardcopy images using dot shape modulation. We have also shown how this information may be recovered from a scan of an image barcode with and without decoder knowledge of the base image. Future research topics include the application of the block error diffusion framework for high addressability systems (this possibility was suggested by an anonymous reviewer), e.g. 600×300 addressable spots with a minimum spot size of 300×300, and the design and use of more general block error filters.

References

[1] R. Floyd and L. Steinberg, "An adaptive algorithm for spatial grayscale," *Proc. Soc. Image Display*, vol. 17, no. 2, pp. 75–77, 1976.

[2] L. Velho and J. M. Gomez, "Digital halftoning with space filling curves," *Computer Graphics*, vol. 25, pp. 81–90, July 1991.

[3] T. Scheermesser and O. Bryngdahl, "Control of texture in image halftoning," *J. Opt. Soc. Am. A*, vol. 13, pp. 81–90, Aug. 1996.

[4] D. L. Lau, G. R. Arce, and N. C. Gallagher, "Green-noise digital halftoning," *Proceedings of the IEEE*, vol. 86, pp. 2424–2442, Dec. 1998.

[5] D. L. Lau, G. R. Arce, and N. C. Gallagher, "Digital color halftoning with generalized error-diffusion and green-noise masks," *IEEE Trans. Image Processing*, vol. 9, pp. 923–935, May 2000.

[6] R. Levien, "Output dependent feedback in error diffusion halftoning," *IS&T Imaging Science and Technology*, vol. 1, pp. 115–118, May 1993.

[7] Z. He and C. A. Bouman, "AM-FM halftoning: A method for digital halftoning through simultaneous modulation of dot size and dot density," *Proc. SPIE Color Imaging: Device-Independent Color, Color Hardcopy, and Applications VII*, vol. 4663, pp. 322–334, Dec. 2001.

[8] N. Damera-Venkata and B. L. Evans, "FM halftoning via block error diffusion," *Proc. IEEE Conf. Image Processing*, pp. 1081–1084, Oct. 2001.

[9] Z. Fan, "Dot-to-dot error diffusion," *J. Electronic Imaging*, vol. 2, pp. 62–66, Jan. 1993.

[10] R. Eschbach, "Pixel-based error-diffusion algorithm for producing clustered halftone dots," *J. Electronic Imaging*, vol. 3, pp. 198–202, Apr. 1994.

[11] N. Damera-Venkata and J. Yen, "Image barcodes," *Proc. SPIE Color Imaging: Processing, Hardcopy, and Applications VIII*, vol. 5008, pp. 493–503, Jan. 2003.

[12] K. Tanaka, Y. Nakamura, and K. Matsui, "Embedding secret information into a dithered multi-level image," *Proc. IEEE Military Communications Conf.*, pp. 216–220, Sept. 1990.

[13] Z. Baharav and D. Shaked, "Watermarking of dither halftoned images," *HP Labs Technical Report*, 1998.

[14] Z. Baharav and D. Shaked, "Watermarking of dithered halftone images," *Proc. SPIE Sym. Electronic Imaging*, 1999.

[15] D. Kacker and J. P. Allebach, "Joint halftoning and watermarking," *IEEE Trans. Signal Processing*, no. 4, 2003.

[16] K. T. Knox and S.-G. Wang, "Digital watermarks using stochastic screens," *Proc. SPIE*, 1997.

[17] G. Sharma and S.-G. Wang, "Show-through watermarking of duplex printed documents," *Proc. SPIE Sym. Electronic Imaging*, Jan. 2004.

[18] M. S. Fu and O. C. Au, "A multi-bit robust watermark for halftone images," *Proc. IEEE Int. Conf. Multimedia and Expo*, 2003.

[19] M. Analoui and J. Allebach, "Model based halftoning using direct binary search," *Proc. SPIE Human Vision, Visual Processing, and Digital Display III*, vol. 1666, pp. 109–121, Feb. 1992.

[20] M. Fu and O. Au, "Data hiding watermarking for halftone images," *IEEE Trans. Image Processing*, pp. 477–484, Apr. 2002.

[21] N. Damera-Venkata and B. L. Evans, "Design and analysis of vector color error diffusion halftoning systems," *IEEE Trans. Image Processing*, vol. 10, pp. 1552–1565, Oct 2001.

[22] J. Jarvis, C. Judice, and W. Ninke, "A survey of techniques for the display of continuous tone pictures on bilevel displays," *Computer Graphics and Image Processing*, vol. 5, pp. 13–40, May 1976.

[23] T. D. Kite, B. L. Evans, and A. C. Bovik, "Modeling and quality assessment of halftoning by error diffusion," *IEEE Trans. Image Processing*, vol. 9, pp. 909–922, May 2000.

[24] D. Shaked, A. Levy, Z. Baharav, and J. Yen, "A visually significant two dimensional barcode," *Hewlett-Packard Laboratories Technical Report - HPLTR 2000-164*, 2000.

[25] K. Knox and R. Eschbach, "Threshold modulation in error diffusion," *J. Electronic Imaging*, vol. 2, pp. 185–192, July 1993.

[26] R. Eschbach and K. Knox, "Error-diffusion algorithm with edge enhancement," *J. Opt. Soc. Am. A*, vol. 8, pp. 1844–1850, Dec. 1991.

[27] N. Damera-Venkata and B. L. Evans, "Parallel implementation of multifilters," *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 6, pp. 3335–3338, June 2000.

[28] N. Damera-Venkata, *Analysis and Design and of Vector Error Diffusion Systems for Image Halftoning*. PhD thesis, Dept. of Electrical and Comp. Eng., The University of Texas at Austin, Austin, TX, Dec. 2000.

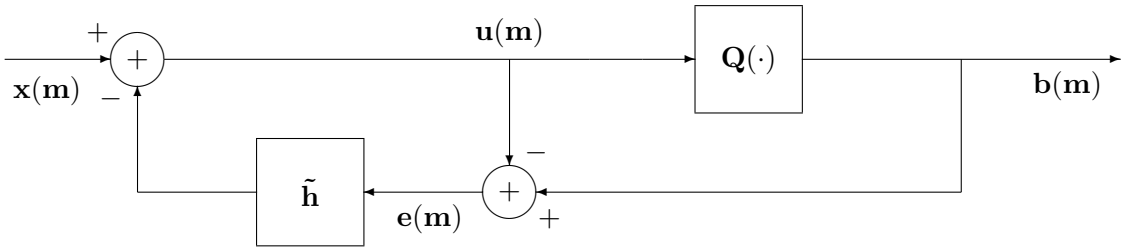


Fig. 1. System block diagrams for block error diffusion halftoning where $\tilde{\mathbf{h}}$ represents the impulse response of a fixed 2-D nonseparable FIR error filter with matrix-valued coefficients. The vector $\mathbf{m}$ represents the 2-D index (m_1, m_2).

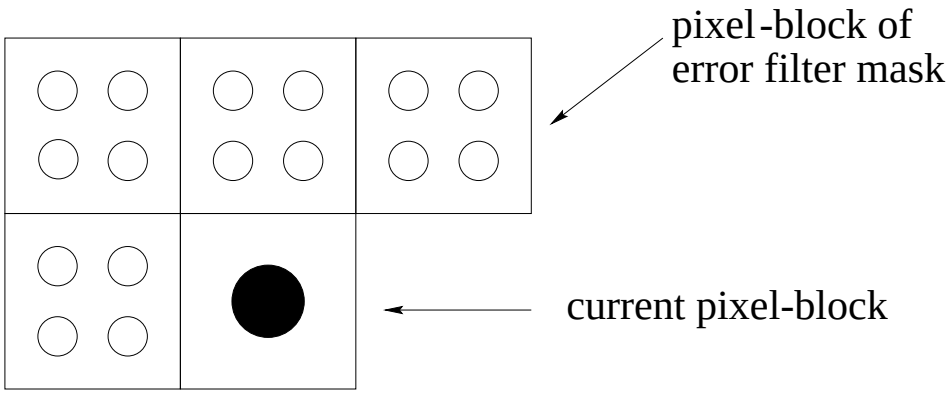
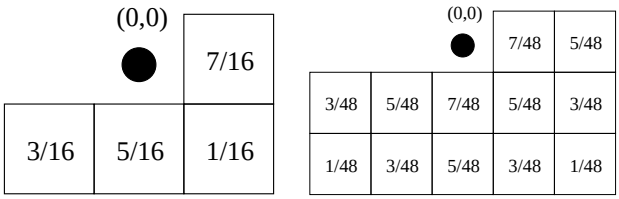


Fig. 2. Block error filter operating on pixel-blocks of 2×2 pixels. The shaded circle indicates the current pixel-block. The unfilled circles indicate the error image pixels underlying the block filter mask. The pixels in the output pixel-block are computed using four linear combinations of all 16 error pixels within the error filter mask.

Fig. 3. Halftone generated by pixel replication induced block clustering. Here, the original image is filtered (to prevent aliasing) and downsampled. The downsampled image is then halftoned using conventional error diffusion. Pixel clustering is then induced by replicating each pixel to form a pixel-block. Note the loss of high frequency information and the blurred appearance.

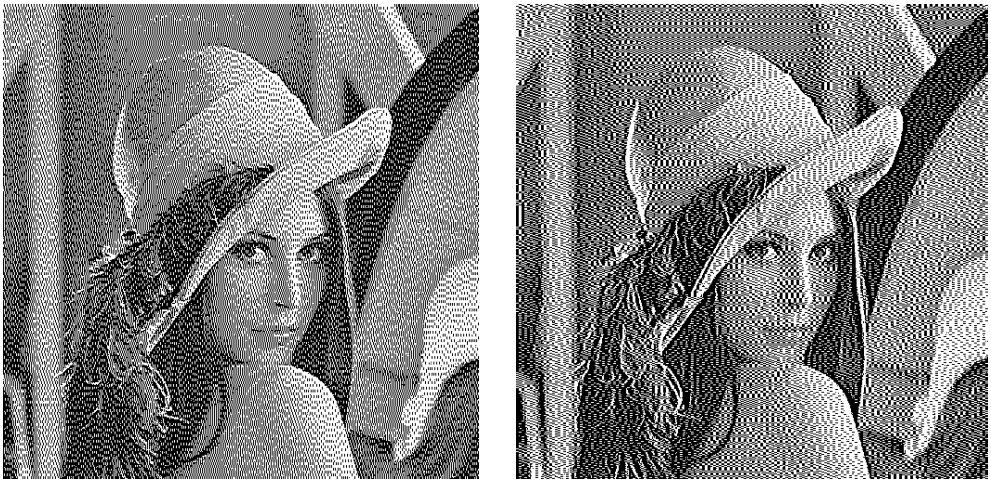
(a) Floyd-Steinberg (b) Jarvis

Fig. 4. Block error diffusion with block error filters derived from conventional Floyd-Steinberg and Jarvis filters. Note the improved performance over pixel replication induced block clustering shown in Fig. 3.



(a) Floyd-Steinberg (b) Jarvis

Fig. 5. Error filters commonly used in conventional error diffusion halftoning. The black dot represents the current pixel being halftoned.



(a) 3:1 dots (b) 1:3 dots

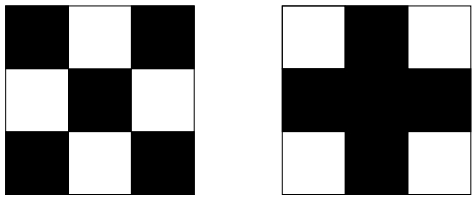
Fig. 6. Block error diffused halftones with rectangular dot shapes.

Fig. 7. Stability problem due to enforcing the dotshape constraint at all pixel-blocks. The shape used was $\mathbf{S} = [1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1]^T$. Compare with Fig. 8(a) that embeds only at minority pixel-block locations.

(a) "multiply" dots (b) "plus" dots

(c) "L shape" dots (d) "T shape" dots

Fig. 8. Block error diffused halftones with user controlled dot shapes.



(a) "multiply" dots (b) "plus" dots

(c) "L shape" dots (d) "T shape" dots

Fig. 9. FM halftone dot shapes. The shaded pixels indicate the pixels in the pixel-blocks that are part of the halftone dot shape.

Fig. 10. Image Barcode generated using the proposed block error diffusion encoder. The rendering resolution is 120 dpi. The images may be viewed from a distance to simulate typical viewing.

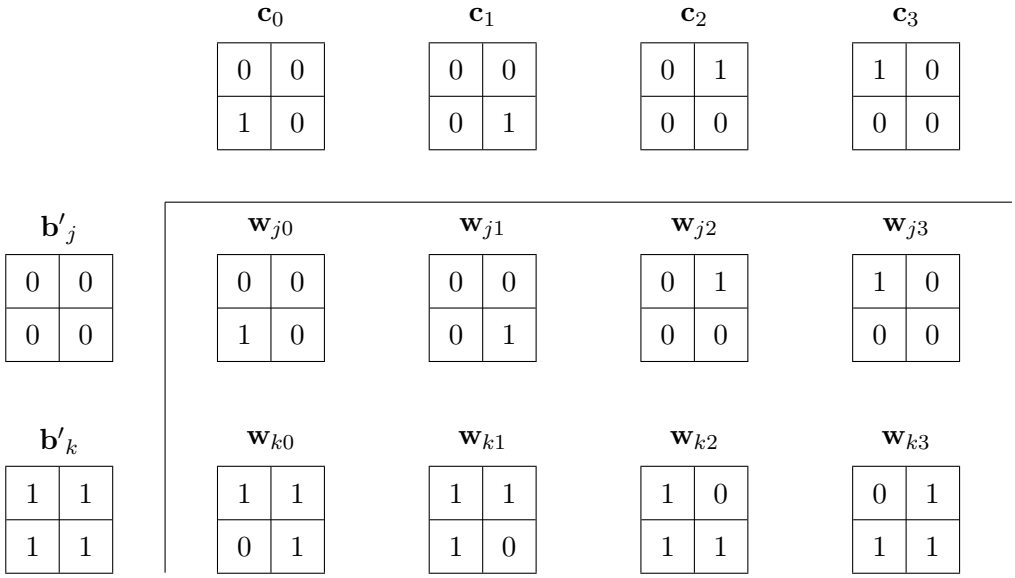


Fig. 11. Example XOR modulation encoding used in Image Barcodes. The encoded block $\mathbf{w}$ is the result of an XOR operation between the encoding codeword block $\mathbf{c}$ and the bi-level intermediate halftone block $\mathbf{b}'$. Thus, $\mathbf{w}_{lm} = \mathbf{c}_m \otimes \mathbf{b}'_l$. The case when the intermediate halftone blocks are constrained to have either all black or all white pixels is shown. Such constraints are required for intermediate halftone estimation at the decoder.

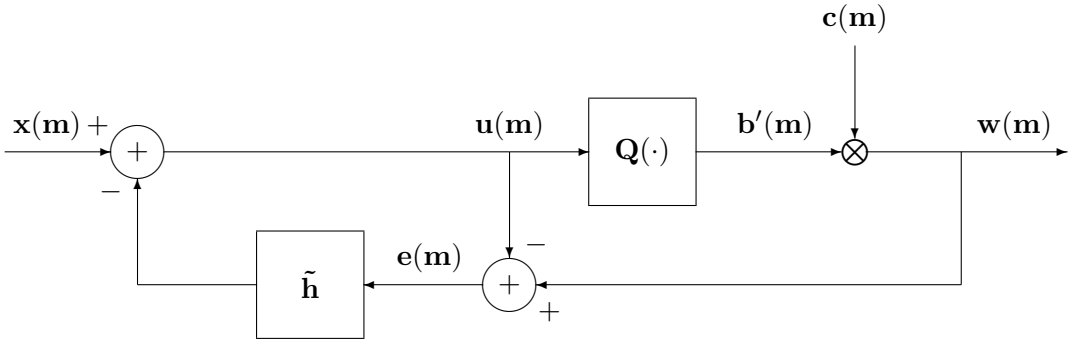


Fig. 12. System block diagram for the image barcode encoder. $\tilde{\mathbf{h}}$ represents a fixed 2-D nonseparable FIR error filter with matrix valued coefficients. $\mathbf{Q}$ denotes the block quantizer. The encoded image block $\mathbf{w}(\mathbf{m})$ is obtained by modulating the codeword $\mathbf{c}(\mathbf{m})$ onto the intermediate halftone block $\mathbf{b}'(\mathbf{m})$ using the XOR operation denoted by $\otimes$. The vector $\mathbf{m}$ represents the 2-D index (m_1, m_2).

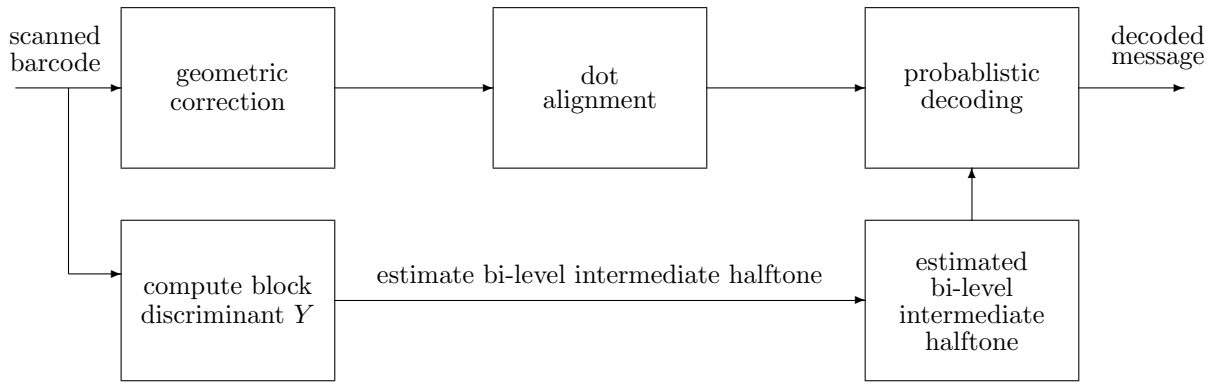


Fig. 13. Decoding pipeline for image barcodes when the base image is not known. The bi-level intermediate halftone can be estimated in this case.

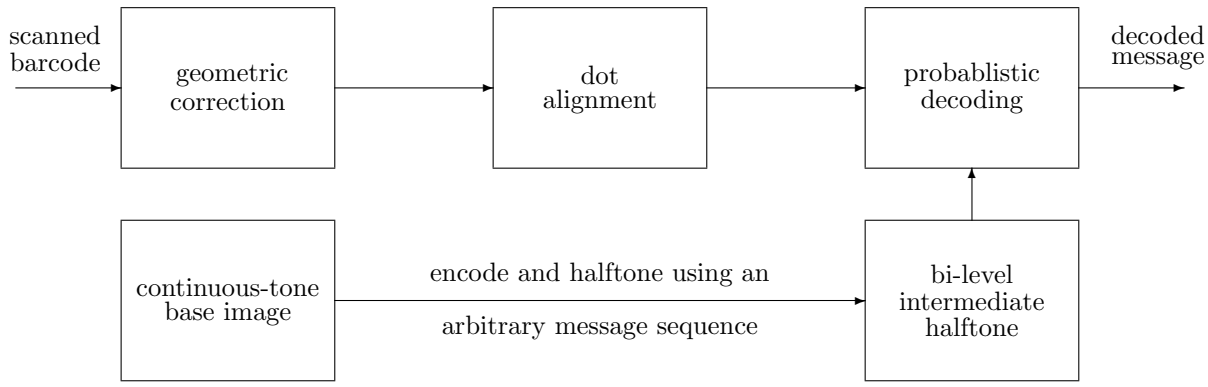


Fig. 14. Decoding pipeline for image barcodes when the base image is known. The bi-level intermediate halftone can be determined exactly in this case.

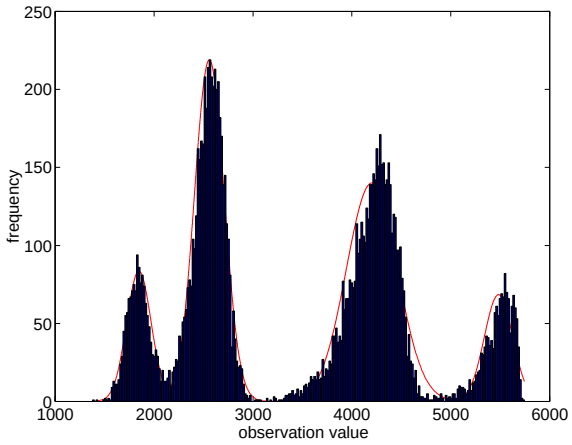
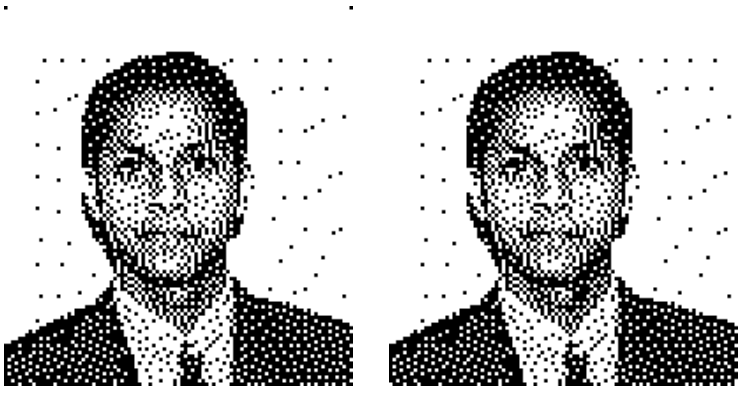


Fig. 15. Histogram of observed block discriminant Y and a Gaussian mixture model fit to the observed statistics using expectation maximization. The extreme modes correspond to fictitious codewords.



(a) (b)

Fig. 16. Intermediate halftone estimation quality: (a) actual intermediate halftone and (b) estimated intermediate halftone.