RC23267 (W0407-013) July 7, 2004
Electrical Engineering

IBM Research Report

Digital Watermarking and Steganography via Overlays of Halftone Images

Chai Wah Wu, Gerhard Thompson
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Mikel Stanich
IBM Printing Systems Division
6300 Diagonal Highway
Boulder, CO 80301

Digital watermarking and steganography via overlays of halftone images

Chai Wah Wu[a], Gerhard Thompson[a] and Mikel Stanich[b]

[a]IBM T. J. Watson Research Center, P. O. Box 218, Yorktown Heights, NY 10598, USA;
[b]IBM Printing Systems Division, 6300 Diagonal Highway, Boulder, CO 80301, USA

ABSTRACT

Recently, several watermarking schemes have been proposed that embed a watermark into two halftone images such that the watermark can be extracted by overlaying these halftone images. The watermark images in these schemes are binary images and the pixels in the two halftone images are correlated or not depending on whether the corresponding pixel in the watermark is on or off. In these schemes, the watermark is binary and does not contain detailed features. Furthermore, the extracted watermark contains residual patterns from the two images which reduces the fidelity of the extracted watermark image. This paper proposes a watermarking algorithm that addresses these problems. In addition, the proposed scheme admits more general watermark extraction functions and allows embedding of multiple watermark images.

Keywords: Data hiding, digital halftoning, error diffusion, steganography, watermarking

1. INTRODUCTION

Recently, several watermarking schemes have been proposed in the literature where a watermark is embedded into two halftone images such that the watermark can be extracted by simply overlaying these halftone images. In [1, 2] the halftone images are created using dither masks, whereas in [3–5] the halftoning is done via error diffusion. The watermark images in these schemes are binary images and the binary pixels in the two halftone images are correlated or not depending on whether the corresponding pixel in the watermark is black or white. There are several restrictions to these schemes. First of all, the watermark is binary and not grayscale or full-color. Second, because the watermark is embedded via correlation, the average intensity of a region of pixels is needed to identify whether the watermark pixel is black or white. Therefore, the watermark images in these schemes are limited to logos or simple graphics and do not contain detailed features. Third, the extracted watermark image contains residual patterns and features from the two halftone images that reduce the fidelity of the extracted watermark image. This also limits the fine details in the watermark image. Fourth, the embedding is done by generating binary halftone images rather than grayscale or multibit images, further reducing the fidelity of the extracted watermark. This paper proposes a novel watermarking algorithm to address these problems. In particular, the proposed watermarking scheme can embed grayscale images having similar levels of details as the two halftone (or multitone) images and there is no residual image in the extracted watermark, i.e. high fidelity watermark extraction can be obtained. In [1–5], the watermark is extracted using OR or Exclusive-OR of pairs of black and white pixels. In the proposed scheme, more general operations on pairs (or more generally n-tuples) of pixels can be used in the watermark extraction algorithm. We also consider the general case where multiple watermarks are embedded into multiple images.

2. PIXELWISE TRANSFORMATION

We consider an image to be a matrix of pixels, i.e. an image G consists of pixels $G(i,j)$ where $G(i,j)$, the (i,j)-th pixel of G, is a vector in a color space. For grayscale pixels, the color space is one-dimensional whereas for RGB pixels, the color space is 3-dimensional. A set of n images $G_1, G_2, \ldots G_n$ of the same size can be considered as a matrix of n-tuples of pixels. This is denoted as $G_1 \times G_2 \times \ldots \times G_n$, i.e. the (i,j)-th element of $G_1 \times G_2 \times \ldots \times G_n$ is the n-tuple $(G_1(i,j), \ldots, G_n(i,j))$. Equivalently, a set of n images can be considered as a single image where the color space is the Cartesian product of n color spaces of the images $G_1, \ldots, G_n$.

Send correspondence to C. W. Wu (E-mail: cwwu@us.ibm.com).

Consider an image transformation Φ which acts on an n-tuple of pixels and produces an m-tuple of pixels, i.e. $\Phi(p_1, \ldots p_n) = (q_1, \ldots q_m)$ where p_i and q_i are pixels. Applying Φ to each of the n-tuples of pixels in $G_1 \times G_2 \times \ldots \times G_n$ independently generates a transformation which acts on a set of n images and produces m images. An interpretation of the mapping Φ is that Φ is a watermark extraction algorithm which produces these m auxiliary watermark images. Another interpretation is that the $n + m$ images correspond to how a single (or a few) image appears under different viewing conditions [6]. The goal in these applications and in general is to produce a set of n images, such that these images, plus the additional m images that are generated by Φ, match another set of $n + m$ predefined images. In general, perfect matching is not possible as there are more sets of $n + m$ images than there are sets of n images. Instead of perfect matching we require only that the images look similar when viewed at the proper distance, i.e. the images should look similar after the lowpass behavior of the human visual system (HVS) and thus only low-pass versions of the images need to match. This reduction of information by the low-pass behavior of HVS allow digital halftoning algorithms to produce an adequate solution to the problem.

For $n = 1$, this approach has been used in digital watermarking [7] and in embedding images that are viewable under different viewing conditions [6]. The main purpose of this paper is to give yet another specific application using this framework for digital watermarking and data hiding where $n \geq 2$. To illustrate ideas, we will next concentrate on the case $n = 2, m = 1$, i.e. the case where a single image is embedded into two images. We will return to the general framework in Section 6.

3. WATERMARKING USING OPERATIONS ON TWO IMAGES

In [1–5], watermarking schemes are proposed that construct two binary halftone images A_1 and A_2, which when overlaid on top of each other produce an image B revealing a watermark image. Assuming that each pixel of A_1 and A_2 is either 0 (denoting a white pixel) or 1 (denoting a black pixel), the overlay operation can be expressed as $B(i, j) = A_1(i, j) \oplus A_2(i, j)$ where $\oplus$ is the OR operation. These methods are based on correlation of pixels between the two images. The watermark image W is a binary image and when $W(i, j) = 0$, the corresponding pixels in A_1 and A_2 are correlated and uncorrelated otherwise. This implies that when $W(i, j) = 1$ the overlaid pixels $B(i, j)$ will be darker on average then when $W(i, j) = 0$ and thus B will reveal the watermark image W. As remarked upon in Section 1, the uncorrelated pixels are darker than the correlated pixels only when averaged over a region of pixels. Therefore the watermark cannot contain fine details. Furthermore, the pixels still reveal residual features of A_1 and A_2 when overlaid.

The proposed watermarking scheme is formulated as follows. Given grayscale images $\tilde{A_1}$, $\tilde{A_2}$, and $\tilde{B}$, we construct two halftone images A_1 and A_2. The image $\tilde{B}$ represents the watermark. From the two images A_1 and A_2, we can extract an watermark image by the operation $B(i, j) = A_1(i, j) \circ A_2(i, j)$. The binary operation $\circ$ is OR in the case of overlaying images, but can be any (not necessarily symmetric) binary* operation such as Exclusive-OR, AND, etc. The goal is to construct A_1 and A_2 such that they resemble $\tilde{A_1}$ and $\tilde{A_2}$ respectively, and such that B resembles $\tilde{B}$. We construct A_1 and A_2 via digital halftoning on the Cartesian product of color spaces.

The input to the digital halftoning algorithm are pixels of the composite image $\tilde{A_1} \times \tilde{A_2} \times \tilde{B}$: $p(i, j) = (\tilde{A_1}(i, j), \tilde{A_2}(i, j), \tilde{B}(i, j))$. Since A_1 and A_2 are binary images, the possible values for the pixel pair ($A_1(i, j)$, $A_2(i, j)$) are $(0, 0)$,$(1, 0)$,$(0, 1)$ and $(1, 1)$. As $B(i, j) = A_1(i, j) \circ A_2(i, j)$, the possible vectors in the space of triples of pixels for $A_1 \times A_2 \times B$ is one the following: $(0, 0, 0 \circ 0), (0, 1, 0 \circ 1), (1, 0, 1 \circ 0), (1, 1, 1 \circ 1)$, which we shall denote as *output vectors*. The set of output vectors is denoted as P. Next, we select a digital halftoning algorithm operating in the space of triples of pixels to halftone $(\tilde{A_1} \times \tilde{A_2} \times \tilde{B})$ using these 4 output vectors, and the resulting output halftone image $(A_1 \times A_2 \times B)$ is such that $A_1 \approx \tilde{A_1}$, $A_2 \approx \tilde{A_2}$ and $B \approx \tilde{B}$. The desired output halftone images are A_1 and A_2 which, by construction, combine via the watermark extraction operation to produce B.

There are several choices for the digital halftoning algorithm, as determined by the tradeoff between speed and halftone quality. Vector error diffusion (VED) [8] is a well known one pass algorithm that is fast and produces good halftones. Another choice is the modified error diffusion algorithm [9, 10] which alleviates some

*Binary here means that the operation has two inputs, not that the operation acts on bits.

problems due to large errors in VED. These one-pass methods analyze and process each n-tuple of pixels once in a specific order. This ordering of pixels to be processed, and the causality of the error diffusion filter cause anisotropic artifacts in the halftone image. As we will see in Section 7, in some applications there could be constraints relating pixels in the image which are far apart and for these applications one-pass methods such as error diffusion are not appropriate. Therefore we will mainly use the iterative isotropic halftoning algorithm used in [7] to halftone the images. The pseudo code for this halftoning algorithm applied to our problem is shown in Algorithm 1. The inputs are three images $(\tilde{A}_1, \tilde{A}_2, \tilde{B})$ and the outputs are 2 halftone images A_1 and A_2. The constants v_i determine how strongly the error in each image is minimized. The linear operator L is the linear space-invariant model of the HVS. Several choices of such filters can be found in [11, 12]. If the images A_1, A_2 and B are to be viewed at different distances, then L can be different from each of these images.

Because of the linearity, space-invariance and the relatively small support of the impulse response of L, the computation of the variable *Error* in Algorithm 1 can be sped up by updating the change in *Error* due to changing a single pixel of Outimage. This is because changing a single pixel of Outimage only changes a small number (on the order of the size of the support of the impulse response of L) of entries of $L(Outimage_k - \tilde{A}_k)$ and $L(Outimage_3 - \tilde{B})$. This technique has been used in dither mask construction when the output is binary [13], but is equally useful here.

The output image *Outimage* can be initialized as an image with random pixels from the set of output vectors or a uniform image of a single output vector. To reduce the number of iterations, *Outimage* can also be initialized by running VED or Modified Error Diffusion and using its output as the initial *Outimage*.

Algorithm 1 Embed watermark in 2 images

```
repeat
  for each $i$ do /* loop through each row */
    for each $j$ do /* loop through each column */
      for each output vector $o = (o_1, o_2, o_3) \in P$ do /* loop through all possible output vectors */
        $Outimage_k(i,j) \leftarrow o_k,\ k = 1,2,3$
        $Error(o) \leftarrow v_1\|L(Outimage_1 - \tilde{A}_1)\| + v_2\|L(Outimage_2 - \tilde{A}_2)\| + v_3\|L(Outimage_3 - \tilde{B})\|$
      end for
      find output vector $o_{\min} \in P$ that minimizes $Error$, i.e., $o_{\min} \leftarrow \arg\min_{o \in P} Error(o)$
      $Outimage(i,j) \leftarrow o_{\min}$
    end for
  end for
until $Outimage$ has not changed between two iterations or the maximum number of iterations is reached
```

4. GAMUT MAPPING

It was shown that for the error in an error diffusion algorithm to be bounded for arbitrarily large images, a necessary and sufficient condition is that the pixels of the input image $(\tilde{A}_1 \times \tilde{A}_2 \times \tilde{B})$ should be within the convex hull of the output vectors [9, 10]. The necessity of this condition is also valid for general halftoning algorithms. We can view the lowpass behavior of the HVS as an averaging behavior and the original image is approximated by an convex sum of output pixels. In order to satisfy this convex hull condition, scaling or gamut mapping of the images is needed. In this section we describe how this can be done. Let S be the set of 3-tuples of pixels in $(\tilde{A}_1 \times \tilde{A}_2 \times \tilde{B})$ i.e. $S = \{(A_1(i,j), A_2(i,j), B(i,j))\}_{i,j}$. Furthermore, let S_1, S_2 and S_3 be the sets of pixels of images A_1, A_2 and B respectively, i.e. $S_1 = \{A_1(i,j)\}_{i,j}$, $S_2 = \{A_2(i,j)\}_{i,j}$, and $S_3 = \{B(i,j)\}_{i,j}$. For simplicity, we consider only gamut mappings M that map a pixel p into a pixel $M(p)$ having the following form: for $p = (p_1, p_2, p_3) \in S$, $M(p) = (s_1 p_1 + d_1, s_2 p_2 + d_2, s_3 p_3 + d_3)$ where s_i are real numbers denoting scaling factors and d_i are offset vectors in the color space. Let H be the (closed) convex hull of the output vectors. H can be expressed by a set of linear inequalities: $H = \{x : Ax \le b\}$. A commonly used algorithm for finding the convex hull of a set of points is the Qhull algorithm [14]. The optimization problem used to find the (optimal) gamut mapping is formulated as follows:

$$\max_{s_i,d_i} \min\left(\frac{s_1}{\alpha_1}, \frac{s_2}{\alpha_2}, \frac{s_3}{\alpha_3}\right) \text{ under the constraint that } M(S) \in H \tag{1}$$

The set of parameters $\{s_i, d_i\}$ which solves the above optimization problem will be used as the gamut mapping M, i.e the pixels of $(\tilde{A}_1 \times \tilde{A}_2 \times \tilde{B})$ are scaled by M before they are halftoned by the halftoning algorithm.

The coefficients α_i determine the "penalty" of scaling each image. The smaller α_i is, the more the corresponding image will be scaled. For instance, in the watermark application, we want the two images A_1, A_2 to retain most of the fidelity of the original images $\tilde{A}_1$, $\tilde{A}_2$, whereas the watermark image, which in many applications is assumed to be a less complex image, can accept more distortion due to scaling. In this case, we set α_3 to be smaller than α_1 and α_2.

It is clear that the constraint in Eq. (1) is linear, i.e the inequality constraint $M(S) \in H$ is linear in the variables s_i, d_i. This is true since $M(S) \in H$ for each pixel $p = (p_1, p_2, p_3) \in S$ can be written as $\sum_{i=1}^{3} \Lambda_i(s_i p_i + d_i) \leq b$ where $A = [\Lambda_1 \vdots \Lambda_2 \vdots \Lambda_3]$ and A, b are the matrices defining the convex hull $H = \{x : Ax \leq b\}$.

If the number of pixels is large, the constraint $M(S) \in H$ can be time consuming to compute. One way to speed up the computation is to not use all the pixels to compute the gamut, i.e. replacing S with a subset $S'' \subseteq S$. This results in a larger feasible region as the set of parameters (s_i, d_i) satisfying the constraint $M(S'') \in H$ (i.e., the set $\{(s_i, d_i) : M(S'') \in H\}$) is larger than the set of parameters satisfying $M(S) \in H$. This means that the convex hull condition may not be strictly satisfied. However, if S'' is a representative sample of S, then the gamut mapping obtained using S'' is close to the gamut mapping obtained using S. In other words, the convex hull condition violation is mild and does not cause adverse effects to the halftone images. In experiments, choosing S'' to be 5 percent of the pixels in S using a sparse grid, gave a gamut mapping very close to the optimal one.

Another way to speed up the computation is using the following simplification which produces a smaller feasible region for the optimization problem in Eq. (1) that is easier to compute than $M(S) \in H$. However, since the feasible region is smaller, the gamut mapping is also less optimal. Let V_i be the set of extreme points of the convex hull of S_i. This means that the convex hull of V_i is equal to the convex hull of S_i. We then replace the constraint $M(S) \in H$ with $M(S') \in H$, where $S' = \{(p_1, p_2, p_3) : p_i \in V_i\}$. It is clear that the convex hull of S' is larger than the convex hull of S, i.e. the feasible region $\{(s_i, d_i) : M(S') \in H\}$ is smaller than $\{(s_i, d_i) : M(S) \in H\}$.

The gamut mapping is computed using the pixel values of the specific images A_1, A_2 and B. This gamut mapping in general is not appropriate for another set of images. By replacing S_1, S_2, S_3 with the extreme points, which by abuse of notation we will denote as V_1, V_2 and V_3 respectively, of the possible gamut (i.e. the possible range of pixel values) of the images A_1, A_2 and B respectively, a gamut mapping obtained using the resulting $M(S') \in H$ can be used for any possible image A_1, A_2 and B respectively. For instance, suppose that the pixel in the image A_2 can take on values between 0 and 1. Then by replacing S_2 with the set $\{0, 1\}$, which is the set of extreme points of the interval $[0, 1]$, and solving the optimization problem using the constraints $M(S') \in H$ where $S' = \{(A_1(i, j), v, B(i, j)) : v \in \{0, 1\}\}$ we obtain a gamut mapping that can be used with images A_1 and B and any image A_2 whose pixels lie in the interval $[0, 1]$. If we replace all three S_i's with $\{0, 1\}$, and assume the OR operation for extracting the watermark, i.e. the output vectors are $(0, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 1)$, then we obtain the following gamut mapping: $s_1 = s_2 = s_3 = 0.25$, $d_1 = d_2 = 0.5$, $d_3 = 0.75$ which can be used for all grayscale images A_1, A_2 and B. Thus by restricting the feasible region in this way we obtain suboptimal gamut mappings that cause more distortion to the original images, but are applicable to a larger set of images.

As was shown in [6, 7], the error calculation in the digital halftoning algorithm can be weighted according to the fidelity one wishes to achieve for each image. For instance, if a low fidelity watermark is adequate, the error calculation can be skewed to include more emphasis in reducing the error in the two main images. This is reflected in the parameters v_i in Algorithm 1.

For general images, the condition that the input pixel lie in the convex hull of the output vectors can be violated mildly without much loss in image quality. This allows the gamut mapping to distort the image less.

5. EXAMPLES

Figure 1 shows an embedding of the Lena image into the two images Baboon and Boat. The gamut mapping is obtained using $\alpha_1 = 1$, $\alpha_2 = 1$, and $\alpha_3 = 0.5$. Figure 1(a) and 1(b) shows the Baboon halftone image and the Boat halftone image respectively, and Figure 1(c) is the extracted watermark image obtained by performing the OR operation on the pixels of Fig. 1(a) and 1(b).

Figure 2 is the same as Figure 1, except that the Lena image is embedded into two images of uniform gray. The gamut mapping is obtained using $\alpha_1 = \alpha_2 = \alpha_3 = 1$.

In Figure 3 the Lena image is embedded into two Baboon images where the extraction is done using the Exclusive-OR operation, instead of the OR operation in the previous examples. This is an interesting example because the two Baboon images (Fig. 3(a),3(b)) appear to be the same, but the halftone dots are arranged differently. When pixelwise Exclusive-OR is applied to these two images, the watermark image (Lena) emerges (Fig. 3(c)). The gamut mapping is obtained using $\alpha_1 = \alpha_2 = \alpha_3 = 1$.

The extracted watermark images in these examples have relatively high fidelity. This is in contrast with the schemes in [1–5], where a residual of the two halftone images is evident in the extracted watermark image.

6. GENERAL FRAMEWORK

We next discuss the general framework for using this approach in image watermarking and data hiding applications. An image A is represented as a matrix of pixels. The (k,l)-th pixel of A is denoted as $A(k,l)$. Each pixel is a vector in some color space T (which can vary from image to image), i.e. $A(k,l) \in T$. Given n images $A_1, \dots A_n$, the watermark extraction algorithm $\Phi = (\phi_1, \dots, \phi_m)$ constructs m images as follows: $\Phi(A_1, \dots, A_n) = (B_1, \dots, B_m)$ where $B_j = \phi_j(A_1, \dots, A_n)$, i.e. B_j are images created from the sets of images $A_1, \dots A_n$. The operation ϕ_j operates pixelwise on the images $A_1, \dots A_n$, i.e. $B_j(k,l) = \phi_j(A_1(k,l), \dots A_n(k,l))$.

Given a set of $n+m$ images $\tilde{A}_1, \dots, \tilde{A}_n, \tilde{B}_1, \dots, \tilde{B}_m$, the goal is to create n images $(A_1, \dots, A_n)$ resembling $\tilde{A}_1, \dots \tilde{A}_n$ such that the m images $B_1, \dots, B_m$ extracted from $(A_1, \dots, A_n)$ via Φ resemble $\tilde{B}_1, \dots \tilde{B}_m$.

This will be done using a digital halftoning algorithm. A digital halftoning algorithm can be described generally as follows: given an image I, it creates a halftone image $\tilde{I}$ such that each pixel of $\tilde{I}$ is a member of a restricted set of output pixels P and furthermore $\tilde{I}$ resembles I under some metric d, i.e. $d(I, \tilde{I})$ is small.

Next we need to specify the set of output pixels for our specific problem in order to use digital halftoning algorithms. Suppose that the pixels of A_i are chosen from the set O_i. Let $R \subseteq O_1 \times \dots \times O_n = \{(p_1, \dots, p_n) : p_i \in O_i \forall i\}$ be the set of all possible output vectors for the combined image $A_1 \times \dots \times A_n$. The subset R can be a strictly smaller subset of $O_1 \times \dots \times O_n$ to express additional relationships between pixels in different images. From R we create the set of extended output vectors $P = \{(p, \phi_1(p), \dots, \phi_m(p)) : p \in R\}$ and halftone $(\tilde{A}_1 \times \dots \times \tilde{A}_n \times \tilde{B}_1 \times \dots \times \tilde{B}_m)$ using P as the set of output vectors. The pseudocode of an iterative halftoning algorithm for solving this problem is shown in Algorithm 2. In the algorithm, $\tilde{A}_1, \dots, \tilde{A}_n, \tilde{B}_1, \dots \tilde{B}_m$ are the input images, P is the set of output vectors and the coefficients v_k^A and v_k^B determine how strongly the error in each image is minimized.

To satisfy the convex hull condition, the pixels of $\tilde{A}_1 \times \dots \times \tilde{A}_n \times \tilde{B}_1 \times \dots \times \tilde{B}_m$ are first scaled by the gamut mapping M. The mapping M is found by solving

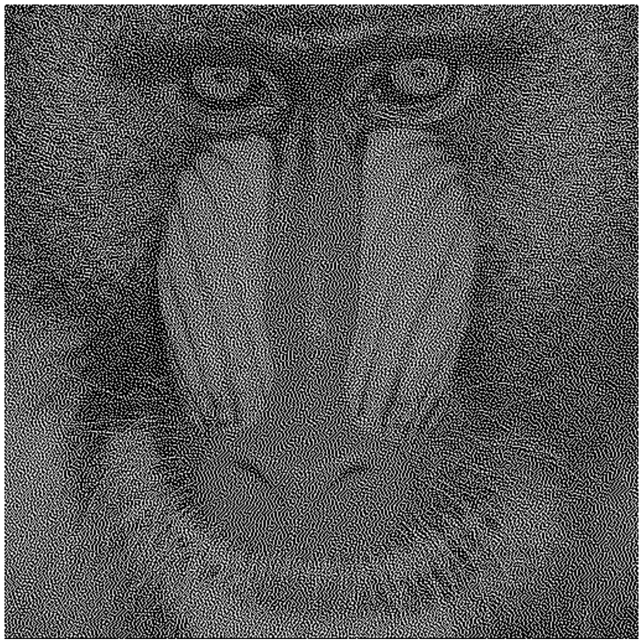
$$\max_{s_i, d_i} \min_j \frac{s_j}{\alpha_j}$$

under the constraint that $(s_1 p_1 + d_1, \dots, s_{n+m} p_{n+m} + d_{n+m}) \in H$ for all pixels $p = (p_1, \dots p_{n+m})$ of $(\tilde{A}_1 \times \dots \times \tilde{A}_n \times \tilde{B}_1 \times \dots \times \tilde{B}_m)$, where H is the convex hull of the output vectors in P. The gamut mapping is then given by:

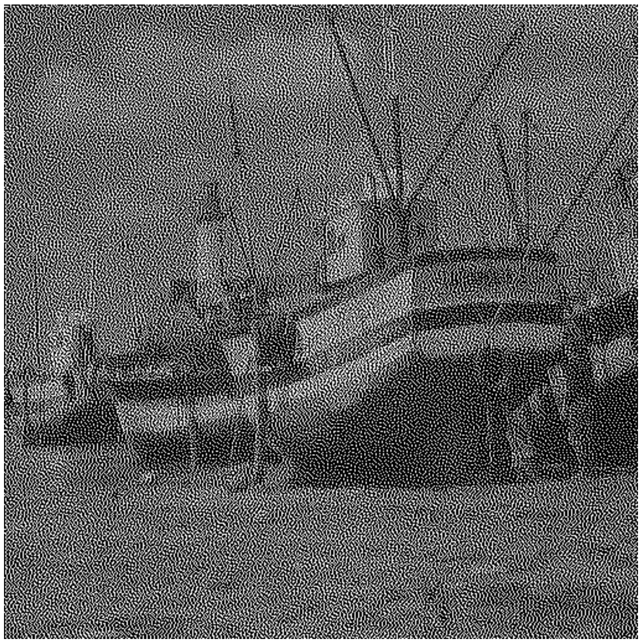
$$M : (p_1, \dots, p_{n+m}) \rightarrow (s_1 p_1 + d_1, \dots, s_{n+m} p_{n+m} + d_{n+m})$$

Other forms of the objective function can be used in the optimization, e.g.:

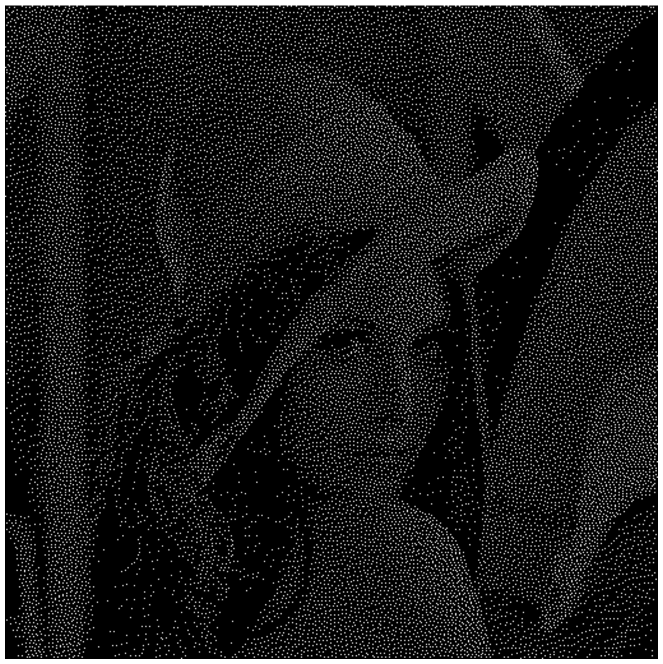
$$\max_{s_i, d_i} \Pi_j s_j$$



(a) Baboon

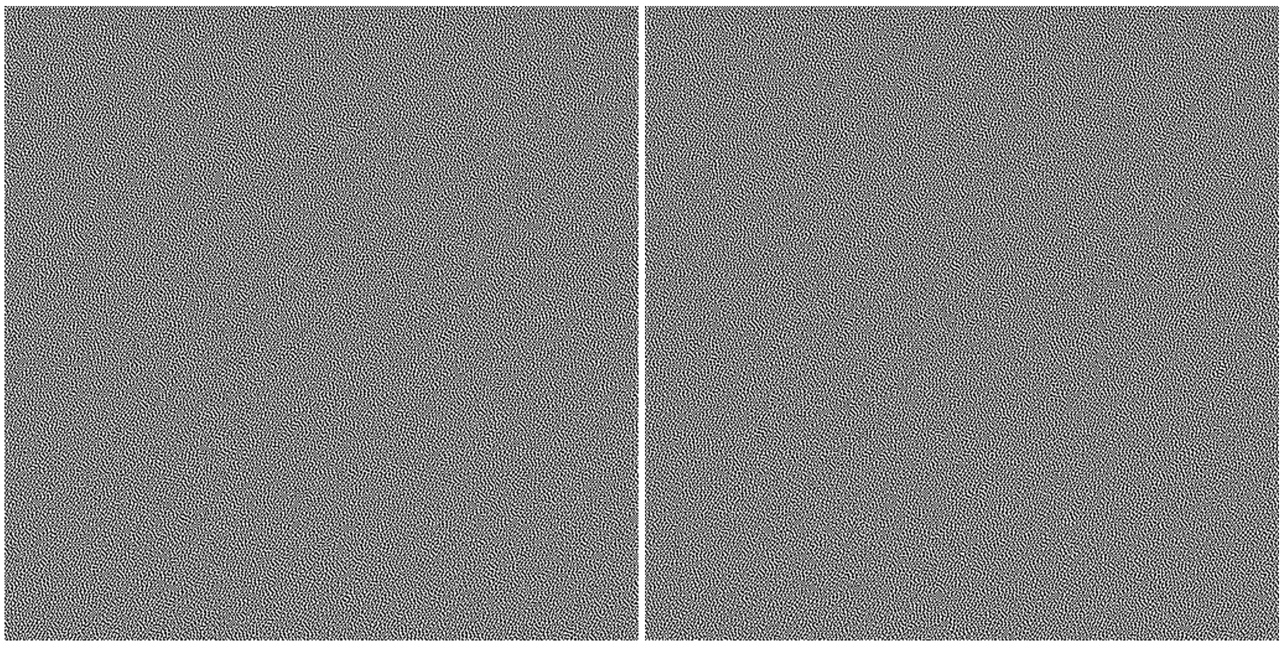


(b) Boat



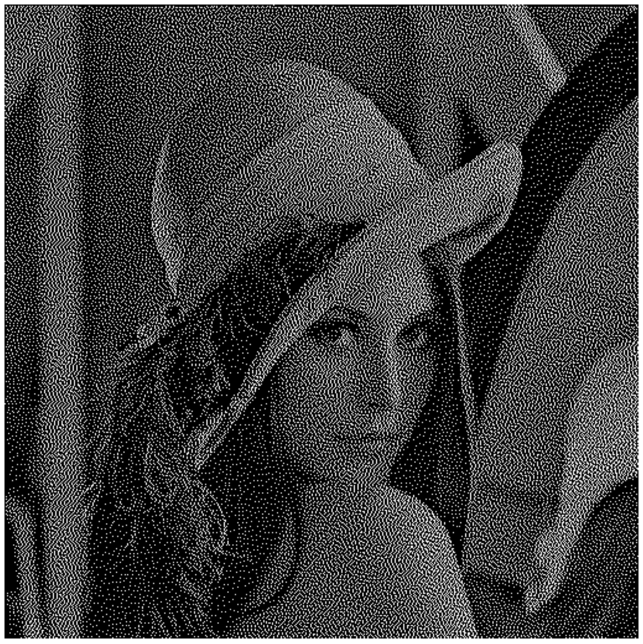
(c) Lena

Figure 1. Embedding of an image into two images. Figure 1(c) is obtained by performing OR on Figure 1(a) and Figure 1(b).



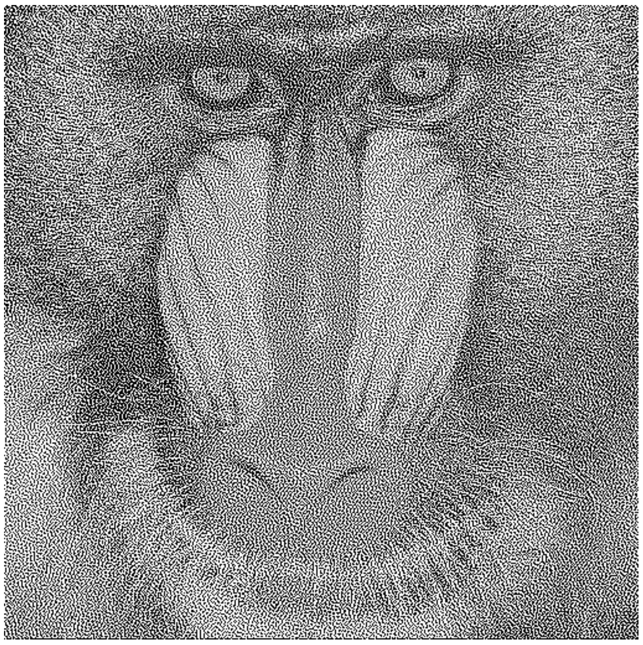
(a) uniform gray

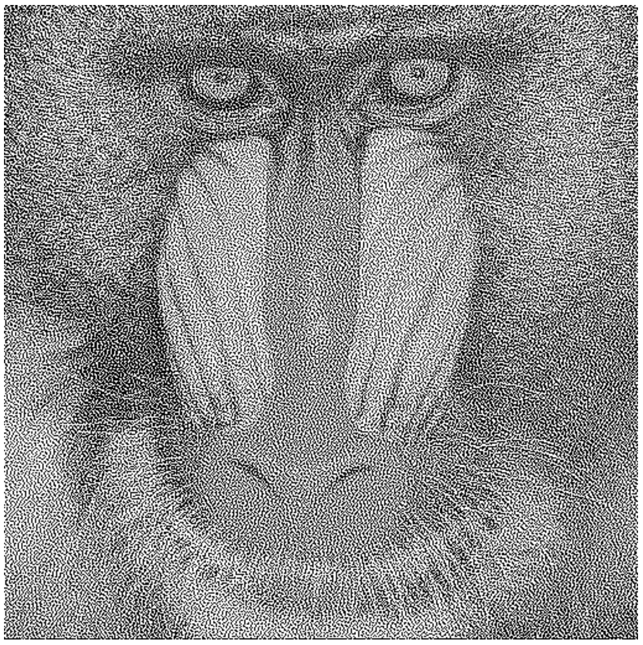
(b) uniform gray



(c) Lena

Figure 2. Embedding of an image into two images. Figure 2(c) is obtained by performing OR on Figure 2(a) and Figure 2(b).


(a) Baboon


(b) Baboon

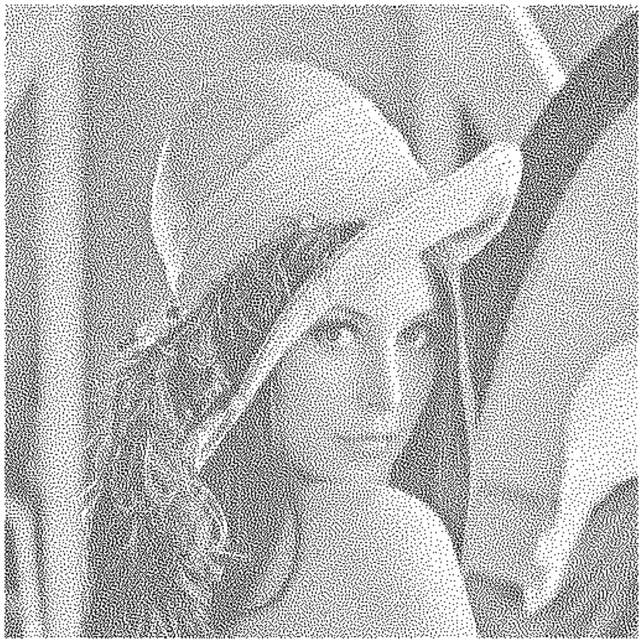

(c) Lena

Figure 3. Embedding of an image into two images where extraction of watermark is via Exclusive-OR. Figure 3(c) is obtained by performing Exclusive-OR on Figure 3(a) and Figure 3(b).

Algorithm 2 Embed m watermark images in n images

```
repeat
  for each $i$ do /* loop through each row */
    for each $j$ do /* loop through each column */
      for each output vector $o = (o_1, \ldots, o_{n+m}) \in P$ do /* loop through all possible output vectors */
        $Outimage_k(i,j) \leftarrow o_k$, $k = 1, \ldots, n+m$
        $Error(o) \leftarrow \sum_{k=1}^{n} v_k^A \|L(Outimage_k - \tilde{A}_k)\| + \sum_{k=1}^{m} v_k^B \|L(Outimage_{n+k} - \tilde{B}_k)\|$
      end for
      find output vector $o_{\min} \in P$ that minimizes $Error$, i.e., $o_{\min} \leftarrow \arg\min_{o \in P} Error(o)$
      $Outimage(i,j) \leftarrow o_{\min}$
    end for
  end for
until $Outimage$ has not changed between two iterations or the maximum number of iterations is reached
$(A_1, \ldots, A_n, B_1, \ldots, B_m) \leftarrow Outimage$
Output $A_1, \ldots A_n$
```

The watermarking application in Section 3 is a case where the output are binary halftone images, $n = 2$ and $m = 1$ and ϕ_1 is a binary operation.

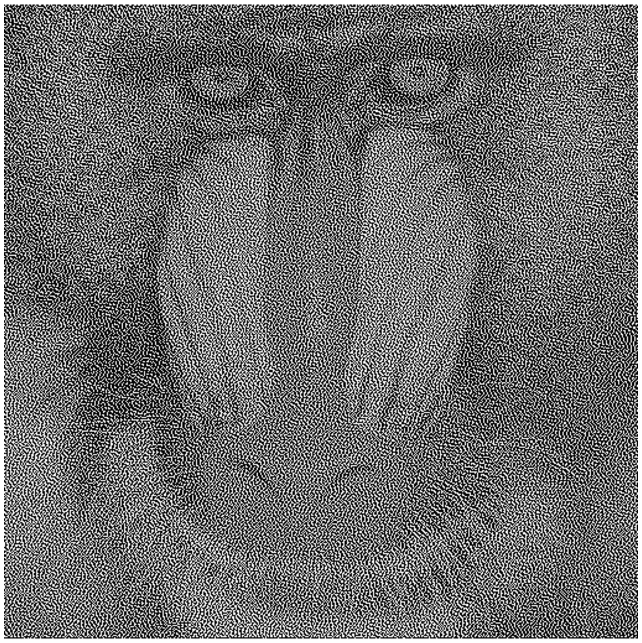
An example of an application where $m > 1$ is different combinations of the images producing different watermarks. For example, given 3 images A_1, A_2 and A_3, combining A_1 and A_2 produces one watermark B_1, and combining A_1 and A_3 produces another watermark B_2. This is shown in Figure 4, where the gamut mapping is obtained using $\alpha_1 = \alpha_2 = \alpha_3 = 1$, $\alpha_4 = \alpha_5 = \frac{1}{2}$. Figure 4(a), 4(b) and 4(c) show the halftone images of Baboon, Boat, and Boat-on-a-lake. The Peppers image in Figure 5(a) is obtained by overlaying Figure 4(a) and Figure 4(b) and the Lena image in Figure 5(b) is obtained by overlaying Figure 4(a) and Figure 4(c).

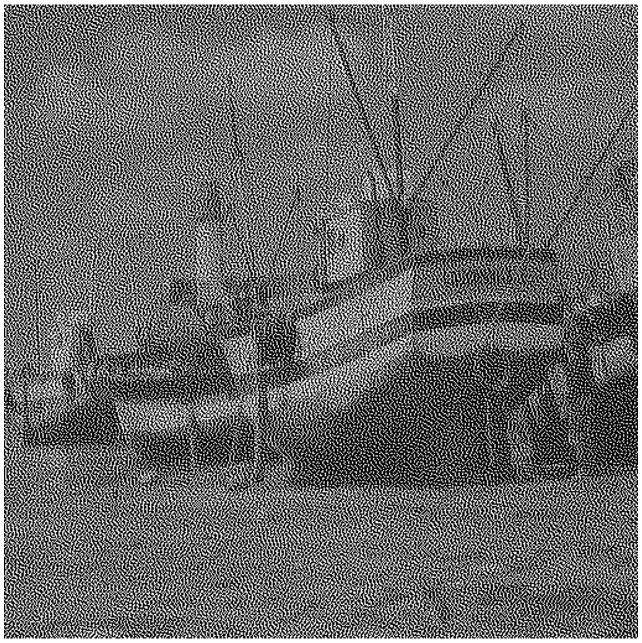
By choosing another Φ, halftone images can be produced where the extraction of the watermarks is done by using all three images, for example, by pixelwise Exclusive-OR of all 3 images.

7. MULTIBIT IMAGES, COLOR IMAGES AND OTHER EXTENSIONS

The algorithms presented can easily be adapted to multibit output. Multibit here refers to each pixel having multiple levels of gray, instead of bilevel where the image pixels are black or white. In this case, the graylevels of the two images are simply added (with clamping) to produce the graylevel of the overlaid image. When each component of the output vector is from a q-bit quantizer (e.g. the black and white case is when $q = 1$), the number of output vectors is large when q is large. In this case the innermost loop in Algorithm 1 to search for the output vector that minimizes $Error$ can take too many computations. Since o_{min} is the output vector that gives the minimal value of the error among all choices of the output vectors at location (i,j), the computation of $o_{min} = \arg\min_{o \in P} Error(o)$ is a discrete optimization problem. For large q this can be relaxed to a continuous optimization problem and solved using nonlinear programming and the result quantized by a q-bit quantizer.

In another application, as was done in [5], the two images can be taken to be rotated versions of the same image. In [5], the second image $\tilde{A}_2$ is the first image $\tilde{A}_1$ rotated 180 degrees. The idea is that the first image contains all the information needed to extract the watermark. Note that for this application, there is an acausal relationship between the pixels of the two images and thus error diffusion is not appropriate as the halftoning algorithm in this setting. For this application, Algorithm 1 needs to be modified as follows. For an R-pixels by C-pixels image, when computing $Error(o)$, in addition to replacing $Outimage_k(i,j)$ with o_k, we need to replace $Outimage_1(R-i+1, C-j+1)$ with o_2, $Outimage_2(R-i+1, C-j+1)$ with o_1, $Outimage_3(R-i+1, C-j+1)$ with $o_2 \circ o_1$. Note that the $(R-i_1, C-j+1)$-th pixel is the (i,j)-th pixel rotated 180 degrees. Then when $o_{\min} = (o_1^{\min}, o_2^{\min}, o_3^{\min})$ is found, we set $Outimage(i,j) \leftarrow o_{\min}$, $Outimage(R-i+1, C-j+1) \leftarrow (o_2^{\min}, o_1^{\min}, o_2^{\min} \circ o_1^{\min})$. Assuming that the binary operation $\circ$ is symmetric, the watermark image $\tilde{A}_3$ must be such that it is invariant under 180 degree rotation. The gamut mapping is chosen using $\alpha_1 = \alpha_2$ with the additional constraints that $d_1 = d_2$, $s_1 = s_2$.


(a) Baboon


(b) Boat

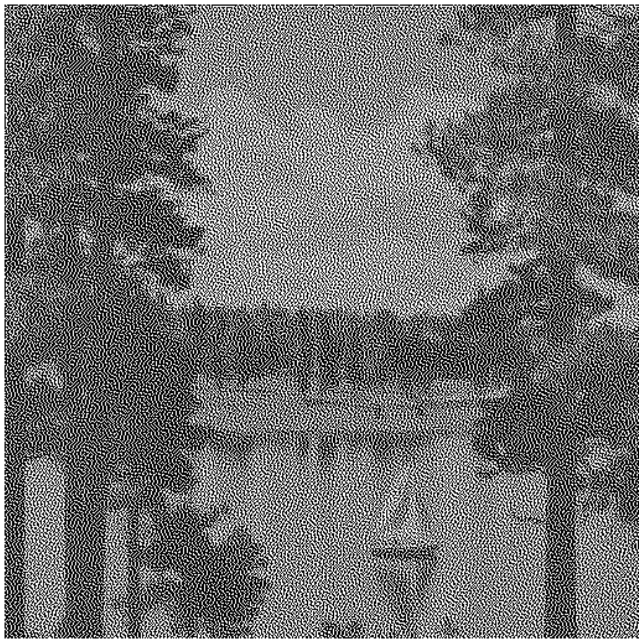
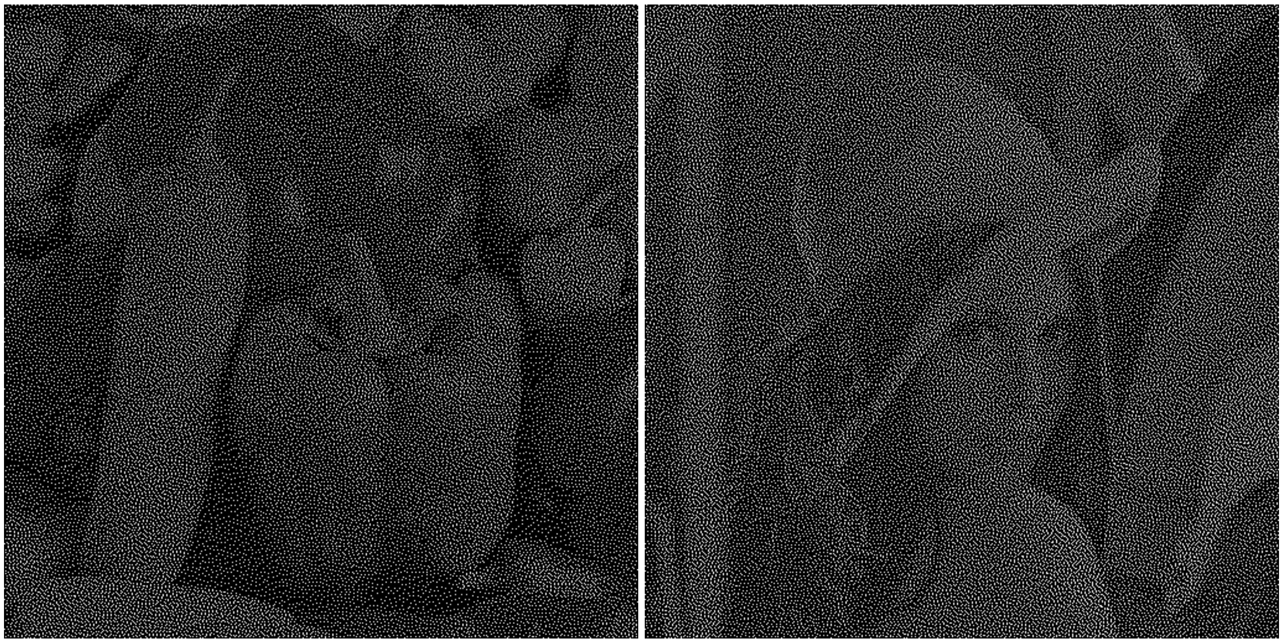

(c) Boat on a lake

Figure 4. Embedding two images into three images. Shown here are three watermarked images. See Figure 5 for the two extracted watermark images.



(a) Peppers (b) Lena

Figure 5. Watermark images extracted from the images in Figure 4. Figure 5(a) is obtained from OR of Figure 4(a) and Figure 4(b). Figure 5(b) is obtained from OR of Figure 4(a) and Figure 4(c).

An example of this is shown in Figure 6, where Figure 6(a) is the halftone image of Baboon and Figure 6(b) is the watermark that is revealed when Figure 6(a) is OR'ed with a copy of itself that is rotated 180 degrees. The gamut mapping is obtained using $\alpha_1 = \alpha_2 = 1, \alpha_3 = 0.7$.

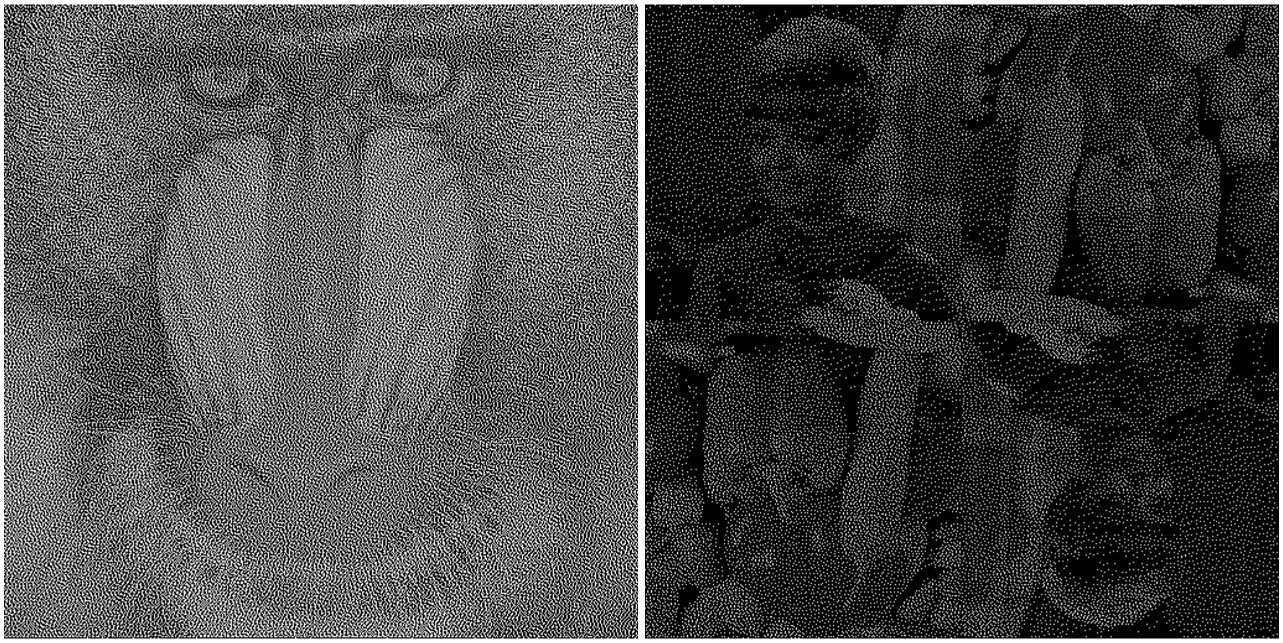
This approach can also be used in color images. Each pixel lies in a multi-dimensional color space such as CIELab, RGB or CMYK and the digital halftoning is performed in the Cartesian product of these color spaces. Instead of processing in the Cartesian product of multi-dimensional color spaces, the images can also be decomposed into their color planes and each plane processed independently as a grayscale image. The results are combined afterwards to obtain a color halftone image. This alternative approach appears to work well in the RGB color space.

8. CONCLUSIONS

We present a method of data hiding in images through halftoning of n-tuples of pixels and illustrate the idea by constructing halftoning images which when overlaid reveal a watermark image. The proposed method overcomes several limitations of previous methods and introduces new possibilities of embedding multiple watermark images and using more general watermark extraction algorithms.

REFERENCES

1. K. T. Knox, "Digital watermarking using stochastic screen patterns." US Patent 5,734,752, 1998.
2. S.-G. Wang, "Digital watermarking using conjugate halftone screens." US Patent 5,790,703, 1998.
3. M. S. Fu and O. C. Au, "Data hiding in halftone image by stochastic error diffusion," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 1965–1968, 2001.
4. M. S. Fu and O. C. Au, "A novel method to embed watermark in different halftone images: data hiding by conjugate error diffusion (dhced)," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. III–529–532, 2003.



(a) Baboon (b) Extracted watermark

Figure 6. Fig. 6(b) is obtained by performing OR on Figure 6(a) and itself rotated 180 degrees.

5. M. S. Fu and O. C. Au, “A novel self-conjugate halftone image watermarking technique,” in *Proc. IEEE Int. Symp. on Circuits and Systems*, pp. III–790–793, 2003.
6. C. W. Wu, G. Thompson, and S. L. Wright, “Multiple images viewable on twisted-nematic mode liquid-crystal displays,” *IEEE Signal Processing Letters* **10**(8), pp. 225–227, 2003.
7. C. W. Wu, “Multimedia data hiding and authentication via halftoning and coordinate projection,” *EURASIP Journal on Applied Signal Processing* **2002**(2), pp. 143–151, 2002.
8. H. Haneishi, T. Suzuki, N. Shimoyama, and Y. Miyaki, “Color digital halftoning taking colorimetric color reproduction into account,” *Journal of Electronic Imaging* **5**, pp. 97–106, Jan. 1996.
9. R. Adler, B. Kitchens, M. Martens, A. Nogueira, C. Tresser, and C. W. Wu, “Error bounds for error diffusion and other mathematical problems arising in digital halftoning,” in *IS&T/SPIE Conference on Color Imaging: Device-independent Color, Color Hardcopy and Graphic Arts V, Proceedings of SPIE*, **3963**, pp. 437–443, 2000.
10. R. Adler, B. Kitchens, M. Martens, A. Nogueira, C. Tresser, and C. W. Wu, “Error bounds for error diffusion and related digital halftoning algorithms,” in *Proceedings IEEE International Symposium on Circuits and Systems*, **II**, pp. 513–516, 2001.
11. R. Näsänen, “Visibility of halftone dot textures,” *IEEE Trans. Syst. Man, Cybernetics* **14**(6), pp. 920–924, 1984.
12. J. Sullivan, L. Ray, and R. Miller, “Design of minimum visual modulation halftone patterns,” *IEEE Trans. Syst. Man, Cybernetics* **21**(1), pp. 33–38, 1991.
13. C. W. Wu, G. Thompson, and M. Stanich, “A unified framework for digital halftoning and dither mask construction: variations on a theme and implementation issues,” in *Proceedings IS&T’s NIP19: International Conference on Digital Printing Technologies*, pp. 793–796, 2003.
14. C. B. Barber, D. P. Dobkin, and H. T. Huhdanpaa, “The quickhull algorithm for convex hulls,” *ACM Trans. on Mathematical Software* **22**(4), pp. 469–483, 1996.