Image Hatching for Visual Cryptography

JONATHAN WEIR and WEIQI YAN, Queen's University Belfast
MOHAN S. KANKANHALLI, National University of Singapore

Image hatching (or nonphotorealistic line-art) is a technique widely used in the printing or engraving of currency. Diverse styles of brush strokes have previously been adopted for different areas of an image to create aesthetically pleasing textures and shading. Because there is no continuous tone within these types of images, a multilevel scheme is proposed, which uses different textures based on a threshold level. These textures are then applied to the different levels and are then combined to build up the final hatched image. The proposed technique allows a secret to be hidden using Visual Cryptography (VC) within the hatched images. Visual cryptography provides a very powerful means by which one secret can be distributed into two or more pieces known as shares. When the shares are superimposed exactly together, the original secret can be recovered without computation. Also provided is a comparison between the original grayscale images and the resulting hatched images that are generated by the proposed algorithm. This reinforces that the overall quality of the hatched scheme is sufficient. The Structural SIMilarity index (SSIM) is used to perform this comparison.

Categories and Subject Descriptors: E.3 [**Data**]: Data Encryption; I.3.3 [**Computer Graphics**]: Picture/Image Generation; I.3.8 [**Computer Graphics**]: Applications

General Terms: Security, Verification

Additional Key Words and Phrases: Cryptography, visual cryptography, image hatching, data encryption, structural similarity

ACM Reference Format:

Weir, J., Yan, W., and Kankanhalli, M. S. 2012. Image hatching for visual cryptography. ACM Trans. Multimedia Comput. Commun. Appl. 8, S2, Article 32 (September 2012), 15 pages.
DOI = 10.1145/2344436.2344438 http://doi.acm.org/10.1145/2344436.2344438

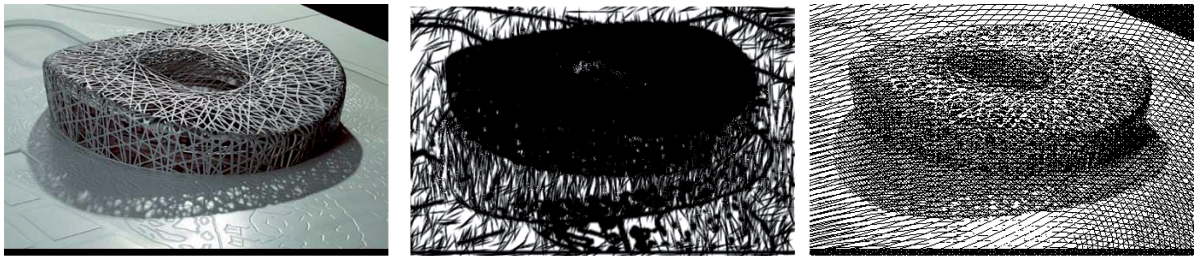
1. INTRODUCTION

Visual Cryptography (VC) is a powerful technique which combines the notions of perfect ciphers and secret sharing in cryptography with that of raster graphics [Weir and Yan 2008]. A binary image can be divided into shares which can be stacked together to approximately recover the original image. In this article, the notion of VC is extended to the one of hatched images. Moreover, it is proposed that the use of this technique can be used to authenticate hatched images by embedding a message inside that can only be revealed if the corresponding secret hatched mask is superimposed upon the image.

Image hatching [Elber 1998a] in general refers to the use of a series of similar strokes of various lengths, angles, mutual space, and other properties of lines to represent an image. These strokes give depth and shape to the image. Different parts of the image can use different stroke types.

Authors' addresses: J. Weir (corresponding author) and W. Yan, The Institute of ECIT, Queen's University Belfast, BT3 9DT; UK; email: jweir07@qub.ac.uk; M. S. Kankanhalli, National University of Singapore, Singapore.

DOI 10.1145/2344436.2344438 http://doi.acm.org/10.1145/2344436.2344438



(a) Image of the Birdsnest stadium. (b) Hatched image of the Birdsnest stadium, old technique. (c) Hatched image of the Birdsnest stadium, new technique.

Fig. 1. Example of image hatching.

For example, a dark part of an image would be represented by strokes that are very close together. Conversely, a lighter section of an image would be represented by strokes that are spaced further apart. For any hatching scheme, the appropriate level of detail should be retained in a way that the image is still recognizable, that is, the hatched image should be unambiguously rendered. This type of NonPhotoRealistic (NPR) line-art drawing has been highly successful over the years, with its roots in line engraving [Ivins 1987], classical drawing [Ivins 1992], and wood carving [Hasluck 1977].

The proposed technique of image hatching in this article differs significantly from the existing techniques in how the final hatched images are generated. As with existing methods, the strokes or textures used will create the correct shadows and tones to convey a realistic image with appropriate shading. The scheme developed within this article can be applied to any grayscale image in order to get an NPR hatched version. The advantage of the proposed method is that it can be used for the purpose of secret sharing. More precisely, these ideas are extend into the VC domain and a novel scheme is created which allows the generation of suitable hatched images capable of hiding VC shares.

Figure 1 provides an example of the algorithm working on an image of the Birdsnest stadium. Figure 1(b) and Figure 1(c) provide a comparison between a real-time hatching technique [Praun et al. 2001] and the proposed line-art-based scheme. A more detailed image is used later within this paper to illustrate the versatility of the algorithm.

Image hatching for VC can be used for specialized image authentication applications. For example, it can be used as a protection mechanism against counterfeiting of currency. Unique random masks can be employed to verify these specially created hatched images and used specifically in identifying counterfeit polymer bank notes adopted worldwide.

2. RELATED WORK

Digital engraving techniques have been demonstrated [Ostromoukhov 1999] using binary and color images. Ostromoukhov deals with 2D images and corresponds to work based on line-art [Elber 1995b, 1995a, Elber 1998b]. It specifically deals with reproducing a human face as accurately as possible while maintaining the traditional copperplate engraving style. Other impressive pen-and-ink illustrations have been presented within Salisbury et al. [1996, 1994]; these use a brush stroke style when reproducing the textures and shading, rather than a thinner line style.

Real-time hatching techniques have also been examined by Praun et al. Praun et al. [2001]. The techniques discussed there [Praun et al. 2001] employ a tonal art map. This map scans the image or scene and creates a texture for it, after which a tone is generated while attempting to maintain spatial and temporal coherence. However, this technique primarily relies on 3D models for it to work correctly. A comparison between our technique and this real-time technique can be made in Figure 1.

Image sharing using VC defines a scheme which is similar to that of general secret sharing [Shamir 1979]. In (k, n) image sharing, the image that carries the secret is split up into n pieces and the decryption is totally unsuccessful unless at least k pieces are collected and superimposed. A Visual Cryptography Scheme (VCS) provides a mechanism by which physically superimposing the pieces (known as shares) of an image is able to completely recover the secret.

The techniques developed within this article help to further the security of printed images in that they provide an authentication method. A novel image hatching technique is also presented which allows images to be converted to a style which is similar to techniques currently in use in modern currency. The properties of these hatched images make them very suitable for use in conjunction with VC. The main property is that the hatched images are halftone [Lau and Arce 2000] images. The distance between the pixels indicates the textures and shading which are used to simulate the gray levels.

Protection and authentication of digital content, particularly printed images, is hugely important [Memon and Wong 1998; Weir and Yan 2009a], especially when dealing with currency. Effectively protecting this digital content can be achieved using VC.

Many visual cryptography schemes that have been developed, regardless of image type, binary/halftone [Ateniese et al. 1996; Zhou et al. 2006; Wang et al. 2009], or color [Hou et al. 2001; Shyu 2006], deal primarily with sharing a single secret. Further development of these schemes would increase the capability of possibly hiding multiple secrets within these hatched images [Hsu et al. 2004; Shyu et al. 2007; Feng et al. 2008; Weir and Yan 2009b]. The proposed method makes also use of a size-invariant image form of VC which allows the VC shares to be of the same size as the original secret [Ito et al. 1999].

The remainder of this article is set out as follows. Section 3 details the contributions and proposed techniques. Section 5 provides a security analysis of the proposed scheme; the experimental results are exhibited in Section 4. The final conclusions are drawn in Section 6.

3. OUR CONTRIBUTION

There are three novel outcomes described in this article.

(1) The definition of a new technique which easily allows the creation of an NPR image, similar to that of a bank note, is presented.
(2) The definition of a method that allows hatched images to be used as a cover image for the VC shares is presented. This should allow a secure share to be embedded within the hatched image.
(3) We extend the preceding definition to correspond to multiple hatched shares. Superimposing the hatched images should reveal the secret.

These techniques allow multiple, secure validation and identification marks on the cover image. If these VC shares cannot be located or recovered, image tampering should be assumed and thus the validity of the image cannot be trusted. The main difference between this scheme and other halftone schemes is that this uses a series of lines, the corresponding space between these lines gives the image the correct coloring, shape, and shading, whereas other halftone images used in other schemes use dot clustering to achieve the same effects.

3.1 Image Hatching

The proposed scheme consists of eight different textures (line styles) which are used for the various shading effects. Eight different textures are chosen because eight different thresholds are used based on the original image. Each of these textures are then applied to their corresponding threshold images.

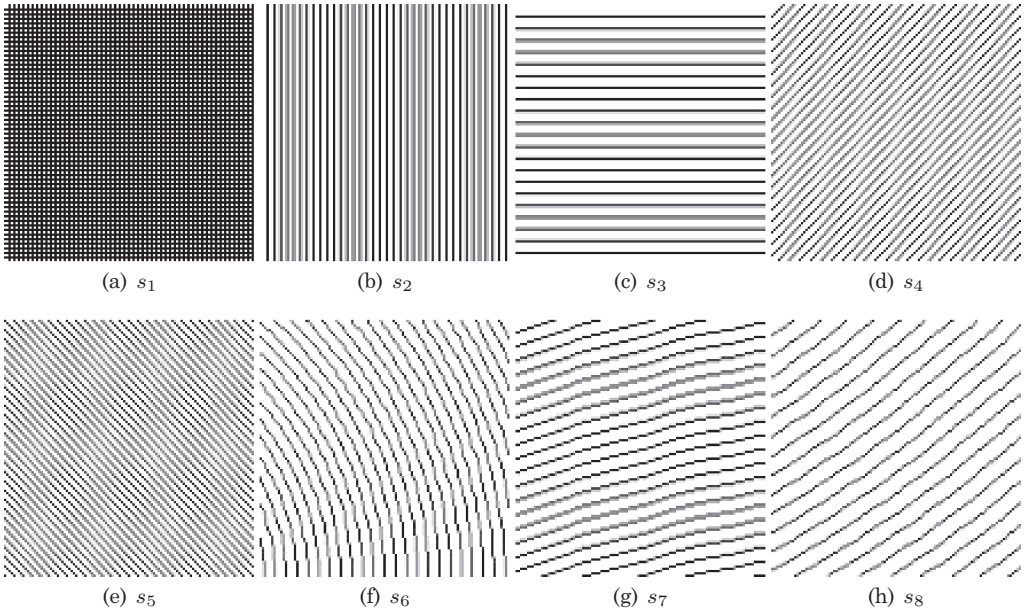


Fig. 2. Set S of the hatching textures.

Let S represent the set of textures, such that $s_i \in S$ where s_i is one of the texture patterns and $i = 1, 2, \ldots, 8$. Figure 2 illustrates these textures.

These textures correspond to the different brightness levels within a grayscale image. As the textures increase from $s_1, \ldots, s_8$, they represent a progressively lighter brightness within an image. For example, s_1 represents a very dark, crosshatched pattern. This pattern is applied to pixel values of <32.

We generate these textures accordingly.

Step 1 Load original image and obtain dimensions, width w and height h.

Step 2 Create 8 new blank images, $w \times h$: $n_1, \ldots, n_8$.

Step 3 For s_1, draw vertical and horizontal black lines on n_1 with a space of 2 pixels between each line.

Step 4 For s_2, draw vertical black lines on n_2 with a space of 3 pixels between each line.

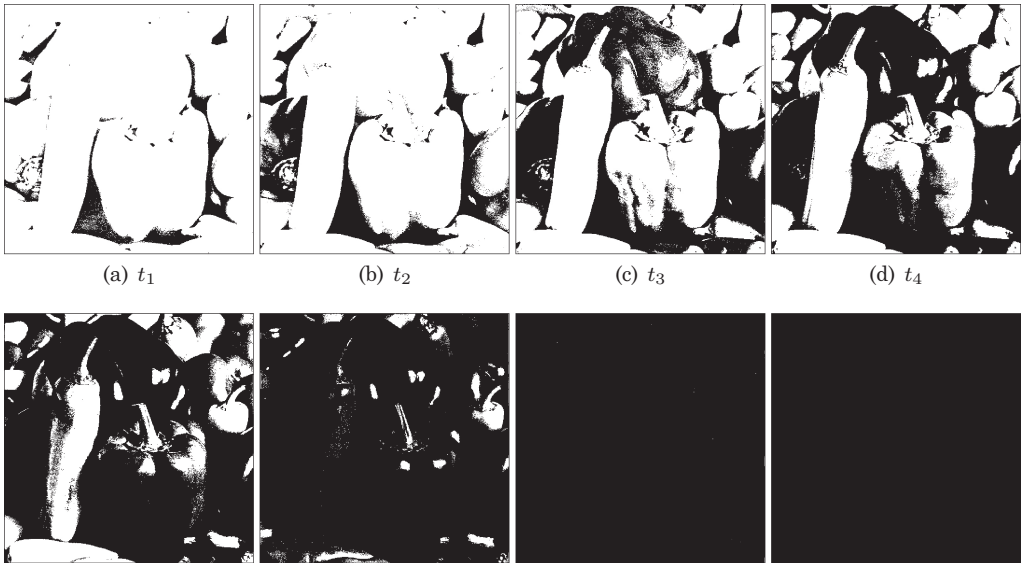
Step 5 For s_3, draw horizontal black lines on n_3 with a space of 5 pixels between each line.

Step 6 For s_4, draw positive diagonal black lines on n_4 with a space of 5 pixels between each line.

Step 7 For s_5, draw negative diagonal black lines on n_5 with a space of 5 pixels between each line.

Step 8 For $s_6 \rightarrow s_8$, draw elliptical black lines on $n_6 \rightarrow n_8$ with a space of 6 pixels between each line.

The elliptical black lines used within step 8 can be defined by specifying a bounding box for the arc of the ellipse to be drawn within. Multiple bounding boxes are required for each texture, $s_6 \rightarrow s_8$ in order to completely cover the full image with its corresponding ellipses. The bounding boxes used are defined by a 4-tuple, (x_0, y_0, x_1, y_1), where (x_0, y_0) correspond to the top left-hand coordinates while (x_1, y_1) correspond to the bottom right-hand coordinates. So, in the case of s_6, its corresponding bounding boxes which are used to draw the ellipse onto n_6 is: $(-i, h-i, 200+i, h+i)$ and $(w+i, h+i, 200-i, h-i)$, where i is in the range $0, 1, \ldots, 2h$ and 200 represents the offset from the center of the threshold image.



(a) t_1 (b) t_2 (c) t_3 (d) t_4

(e) t_5 (f) t_6 (g) t_7 (h) t_8

Fig. 3. Set T of the thresholds.

The bounding boxes related to s_7 are: $(-w, i, 2w, 3h+i)$ and $(-w, -i, 2w, 3h-i)$, where i is in the range $0, 1, \ldots, h$. Finally, s_8 has $(-3w+i, -3h+i, w+i, h+i)$ and $(-3w-i, -3h-i, w-i, h-i)$ as its bounding boxes and its value of i falls between $0, 1, \ldots, w$.

It is worth mentioning that the sizes of the bounding boxes are greater than the overall image size. When the ellipses are drawn, only part of them are visible within the texture image; this is how different angled patterns are obtained for these elliptical textures.

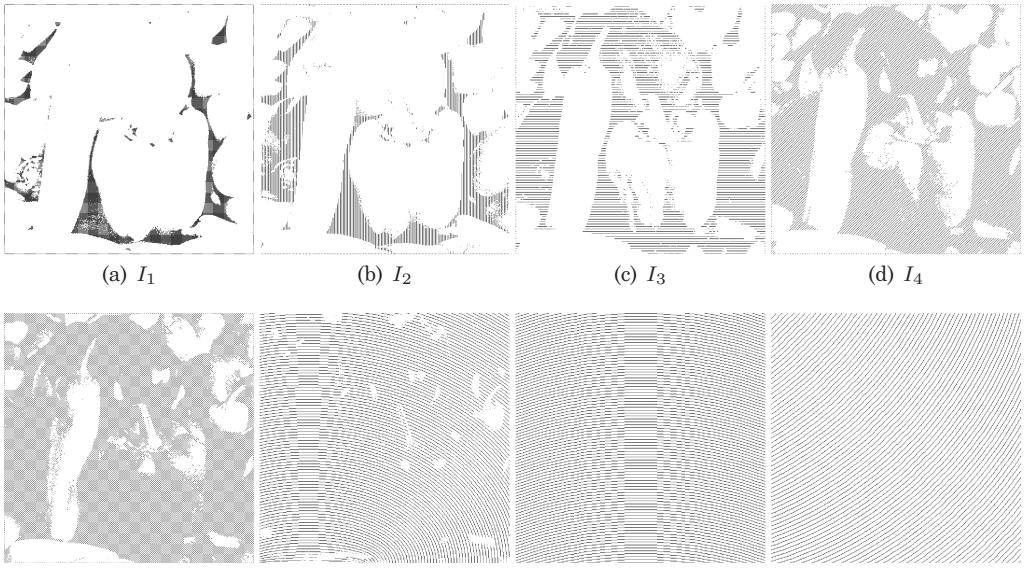
The next step involves taking multiple threshold levels of the original grayscale image to allow the textures to be applied. Because our threshold function segments the grayscale image into eight different pieces, this is why eight different textures are used, as mentioned earlier. Eight different threshold levels are obtained from the image to which each of the textures can be applied.

Let $p_{ij} \in P$, P is the set of all pixels and $p_{ij} = 0, \ldots, 255$ where p_{ij} represents the pixels grayscale value, $i = 1, 2, \ldots, w$ and $j = 1, 2, \ldots, h$.

The threshold function examines p_{ij} eight different times based on a different level each time, $0 < p_{ij} < 32k$, where $k = 1, 2, \ldots, 8$. Each corresponds to a specific threshold, t_k, where $t_k \in T$, T is the set of all thresholds obtained. Therefore, $\forall p_{ij},\ t_k = p_{ij} < 32k$.

Figure 3 shows each of the different threshold levels obtained after the set T has finished processing.

Firstly, the eight hatching textures from Figure 2 are created. The textures are then applied to the thresholds by inverting both the texture and the threshold image and superimposing them. This works as follows: each of the textures needs to be applied to its corresponding threshold so as to create the set of stroked threshold images. The first pair taken are s_1 and t_1, they are inverted then superimposed. At the pixel level, this inverting and superimposing operation can be represented mathematically in Eq. (1), where p is the pixel value returned, m is the largest pixel value (255 in a grayscale image), and s_i represents the pixel values in the texture image and t_i represents the pixel values in the threshold image.



(a) I_1 (b) I_2 (c) I_3 (d) I_4

(e) I_5 (f) I_6 (g) I_7 (h) I_8

Fig. 4. Set of thresholds with their corresponding textures applied.

$$p = m - (m - s_i) \times \frac{(m - t_i)}{m} \tag{1}$$

Figure 4 displays the results after applying Eq. (1) to all the textures and their corresponding thresholds. Once all of these textured thresholds are obtained, they are all superimposed to create the final hatched image (Figure 7(c)). The superimpose operation is a simple Boolean OR operation where the final hatched image F is: $F = I_1$ OR I_2 OR ... OR I_8.

Figure 5 provides an example of the software interface which can be used to illustrate our image hatching technique working on-the-fly. The image on the left-hand side of the interface is the original image. When adjustments are made using the slide bars at the top of the interface, the middle image reflects these changes. The adjustments correspond to brightness, contrast, saturation, and hue. As each of these options are adjusted, the corresponding changes are reflected in the hatched image.

The image on the right-hand side provides the hatched version of the middle image. The hatched image reflects the corresponding changes that have been made to the original.

The interface supports adjustment in the following areas: contrast; saturation, hue, and brightness (HSV or HSB). The contrast C can be computed as the standard deviation of the pixel intensities. We have

$$C = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(I_i - \bar{I})^2}, \tag{2}$$

where the intensities I_i are normalized to a maximum value of 1, N is the total number of pixels in the image, and $\bar{I}$ is the average intensity across the image.

Similarly the saturation and hue are determined using the following formulas, where R, G, and B represent red, green, and blue in the RGB color space. The hue angle h can be computed accordingly.

(a) 60% contrast enhancement. (b) 70% brightness enhancement.

Fig. 5. Interface to dynamically adjust an image and provide its hatched equivalent.

$$h = \begin{cases} 0, & \text{if } \max = \min \\ \left(60^\circ \times \frac{G-B}{\max - \min} + 360^\circ\right) \mod 360^\circ, & \text{if } \max = R \\ 60^\circ \times \frac{B-R}{\max - \min} + 120^\circ, & \text{if } \max = G \\ 60^\circ \times \frac{R-G}{\max - \min} + 240^\circ, & \text{if } \max = B \end{cases} \tag{3}$$

The saturation and brightness (or value), s and v are defined

$$s = \begin{cases} 0, & \text{if } \max = 0 \\ \frac{\max - \min}{\max} = 1 - \frac{\min}{\max}, & \text{otherwise,} \end{cases} \tag{4}$$

$$v = \max, \tag{5}$$

where max and min are the greatest and least values from R, G, and B, respectively.

As the original image is adjusted, for example, the contrast is enhanced by 60% (Figure 5(a)); the hatched image also reflects this change. Another example is provided by altering the image's brightness by 70% (Figure 5(b)). This highlights the versatility of the algorithm working on an original image which has had significant adjustments. Clear differences are noticeable between each of the hatched images.

3.2 Image Hatching with VC

Traditional VC shares are very effective in terms of secret sharing, but pixel expansion tends to be a problem. Pixel expansion occurs due to how the VC shares are created. Typically one pixel is expanded into a 2×2 block of pixels, which is used to represent the original pixel. This results in the final shares being four times the size of the original secret after encryption. Size-invariant shares have the potential for better results; they have no pixel expansion and offer quality similar to that of the traditional schemes. The scheme proposed in Ito et al. [1999] is used when creating the shares for this intended scheme.

Ito's scheme [Ito et al. 1999] removes the need for this pixel expansion. The scheme uses the traditional (k, n) scheme where m (the number of subpixels in a shared pixel) is equal to one. The structure of this scheme is described by a Boolean n-vector $\mathbf{V} = [v_1, \ldots, v_n]^T$, where v_i represents the color of the pixel in the i-th shared image. If $v_i = 1$ then the pixel is black, otherwise, if $v_i = 0$ then the pixel is white. To reconstruct the secret, traditional ORing is applied to the pixels in $\mathbf{V}$. The recovered secret can be viewed as the difference of probabilities with which a black pixel in the reconstructed image is generated from a white and black pixel in the secret image.

As this scheme uses no pixel expansion, m is always equal to one and n is based on the type of scheme being used, for example in a $(2, 3)$ scheme, $n = 3$. The most important part of any visual secret sharing scheme is the contrast. The lower the contrast, the harder it is to visually recover the secret. The contrast for this scheme is defined as follows: $\beta = |p_0 - p_1|$, where p_0 and p_1 are the probabilities with which a black pixel on the reconstructed image is generated from a white and black pixel on the secret image.

Using the defined sets of matrices C_0 and C_1, and a contrast $\beta = \frac{1}{3}$, $n \times m$ Boolean matrices S^0 and S^1 are chosen at random from C_0 and C_1, respectively.

$$S_0 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad S_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

To share a white pixel, one of the columns in S_0 is chosen and to share a black pixel, one of the columns in S_1 is chosen. This chosen column vector $\mathbf{V} = [v_1, \ldots, v_n]^T$ defines the color of each pixel in the corresponding shared image. Each v_i is interpreted as black if $v_i = 1$ and as white if $v_i = 0$. For example, sharing a black pixel, one column is chosen at random in S^1, resulting in the following vector.

$$\mathbf{V} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{7}$$

Therefore, the i-th element of the vector determines the color of the pixels in the i-th shared image, thus in this (2,3) example, v_1 is white in the first shared image, v_2 is black in the second shared image, and in the third shared image, v_3 is white.

Once the hatched image has been generated, a secret of the same size is chosen. The size-invariant VC scheme is applied to the secret in order to generate the two shares. To embed the first share successfully within the hatched image we need to alter it. This involves reducing its overall pixel density. This reduction in pixel density is a trade-off between security and visually being able to hide the share within the hatched image. It lowers the recovered secret's overall contrast but allows the embedded hatched image to remain inconspicuous. As previously mentioned, contrast is important [Blundo et al. 2003; Yang et al. 2006] because it determines the readability of the recovered secret.

Reducing the shares' density involves cutting or removing vertical or horizontal strips for the first image share. This allows the share to be embedded, without distorting the hatched image too significantly, while keeping its secure properties and retaining enough data in order to successfully recover the secret. The steps involved during this embedding process are detailed next.

Step 1 Examine and compare n different areas $(A_1, \ldots, A_n)$ on both the share and the hatched image. Typically, the image is segmented into 9 equal areas, so $n = 9$. This is equivalent of having a 3×3 grid over each of the images.

Step 2 If the density d of A_i in the share (d_s) is less than the density in the hatched image (d_h), remove vertical strips from the hatched image. Removing vertical strips involves setting that particular column to be completely made up from white pixels.

Step 3 Set every fourth column to completely white. Check the density again, if the density is still too dark, set every second column to white.

Step 4 However, if $d_s > d_h$, then set every fourth column in the share image to be completely white. This removes some information from the share, but is essential for noninvasive embedding.

Step 5 Repeat this process for all remaining A_i areas.

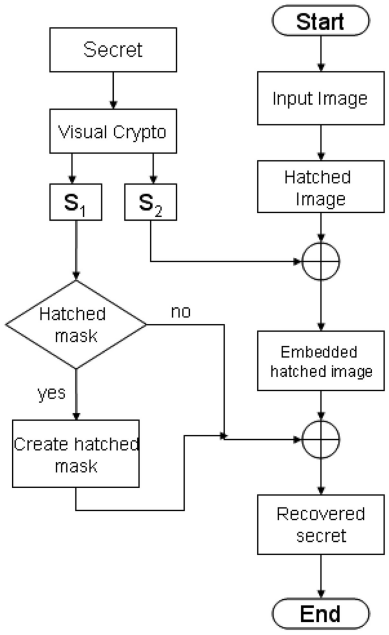


Fig. 6. Flowchart of image hatching with VC.

Step 6 After each area has been processed, rebuild the image based on each of these A_i areas. Each area is placed at its original location.

The overall embedding process attempts to fill part of the hatched image with part of the share image in such a way that the hatched image remains hatched and legible, which should leak no information and also that enough of the share image is left intact so that good secret recovery can be achieved with an acceptable contrast.

Three separate schemes are defined here for using the second share in order to recover the secret. The first, used in Figure 8, leaves the second share unaltered, resulting in a nonhatched random mask for the hatching cover image and its embedded VC share. This provides an overall darker result after the secret has been revealed. The next scheme, illustrated in Figure 9, involves creating a secure hatched mesh. Patterns from S are chosen, namely s_6 and s_8, and then the second share is cut in the same way, and share one is cut in order to embed it into the hatched cover image. After share two has been cut, it is embedded within the new hatched mesh. After superimposing, the secret can be clearly revealed. This same cutting technique is used in the final scheme, presented in Figure 10. Each share is embedded into a hatched image; superimposing these hatched images reveals the secret. Figure 6 provides a flowchart of these prospective schemes.

4. EXPERIMENTAL RESULTS

In this section, the results are provided and a commentary is provided on each of the expected outcomes discussed previously in Section 2. All code was developed in Python 2.5 using the Python Imaging Library 1.1.6.

Figure 7(a) shows the original Peppers image; its binary halftone equivalent is in Figure 7(b) (created using Floyd-Steinberg dithering) and the newly created hatched image based on the proposed scheme is located in Figure 7(c). Notice the different contours and line directions to give the image the correct

(a) (b) (c)

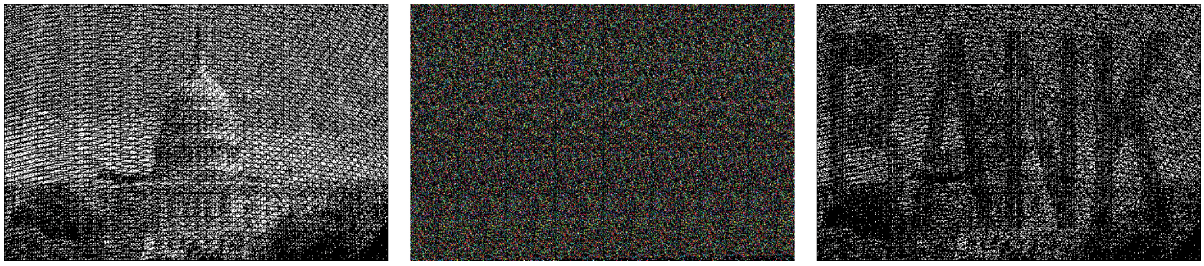
Fig. 7. (a) Original image; (b) halftone image (Floyd-Steinberg dithering); (c) new hatched image.

Table I. An SSIM Index Comparison between Two Halftone Techniques

Halftone Technique		
Images	Hatching	FS Dithering
Airplane	0.0413	0.0066
APC	0.0397	0.0055
Boat	0.1269	0.0037
Car and APC	0.0603	0.004
Couple	0.0875	0.0034
Elaine	0.0848	0.004
F-16	0.1239	0.0043
House	0.1368	0.0034
Lena	0.1006	0.0042
Mandrill	0.1434	0.002
Peppers	0.1226	0.0048
Sailboat on Lake	0.1641	0.0035
Splash	0.0889	0.0064
Stream with Bridge	0.1739	0.0019
Tiffany	0.0778	0.0047
Truck	0.0616	0.0038
Truck with APC	0.0907	0.0026

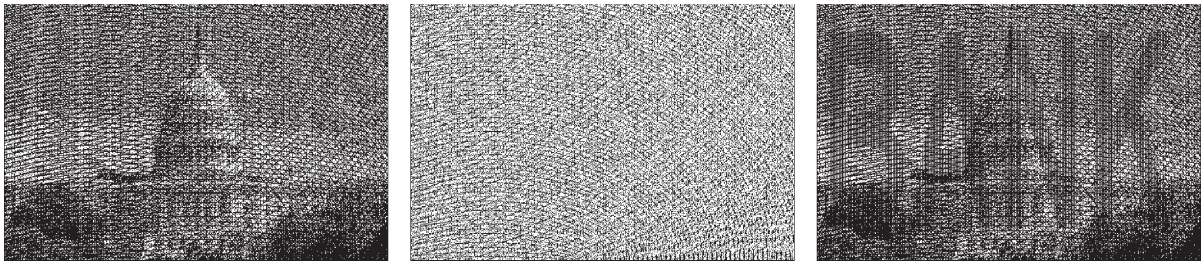
depth and feel. The dark parts of the image show up correctly in that they have a very thick texture pattern applied to them. The brighter parts have a lighter texture pattern on them so that they do not show up completely white. This is very similar to the engraving techniques discussed previously in Section 2. All of the details contained within the original image are retained and can be clearly observed. This supports the first expected outcome.

Table I provides a range of results comparing the structural similarity (SSIM) [Wang and Bovik 2009; Wang et al. 2004] between the original grayscale image and both halftone techniques. Each halftone technique was compared against the original grayscale image. It can be observed that, structurally, this image hatching technique performs better by comparison than the Floyd-Steinberg (FS) dithering technique used within the Python Imaging Library which was used to generate the halftone images.



(a) (b) (c)

Fig. 8. (a) The Capitol Building; (b) its corresponding secure random mask; (c) the recovered secret.



(a) (b) (c)

Fig. 9. (a) The Capitol Building; (b) its corresponding secure hatched mask; (c) the recovered secret.

Each of the images were obtained from the USC-SIPI Image Database (http://sipi.usc.edu/database/) and converted to grayscale. Images of size 512×512 where chosen for the tests. The suggested usage of the SSIM algorithm was taken into consideration before compiling the test data.

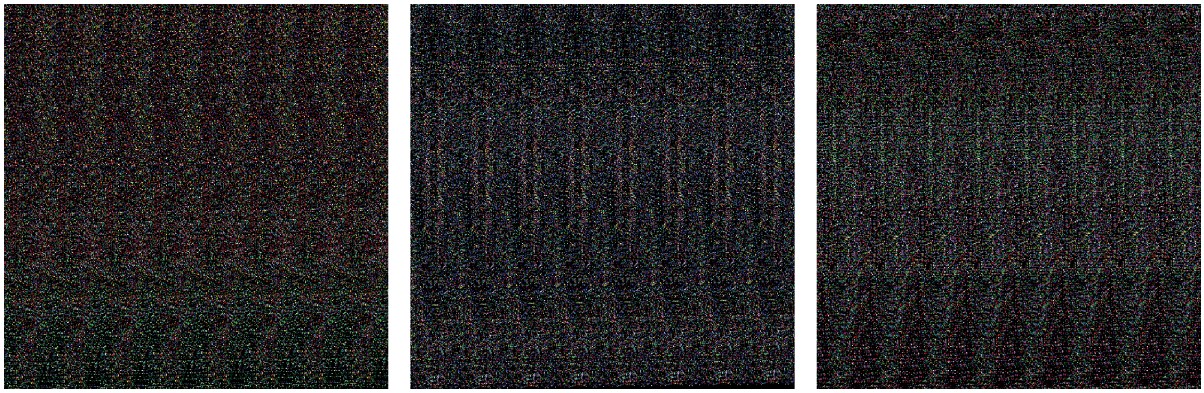
Based on suggestions by the authors of the SSIM index [Wang et al. 2004], the index is calculated on various windows of an image and can be expressed in the following form

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy}^2 + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{8}$$

where μ_x is the average of x, μ_y is the average of y, σ_x^2 and σ_y^2 are the variance of x and y respectively, C_1 and C_2 are used as stabilizing constants while $2\sigma_{xy}^2$ represents the covariance of x and y. The values typically fall between -1 and 1.

The suggested usage of this algorithm involves an empirical formula to determine the scale of the images as well as how they are viewed from a typical distance. This process involves averaging local regions within the image of size $F \times F$ pixels, then downsampling the images by a specific factor F, where $F = \max(1, \mathrm{round}(N/256))$ and N is the number of pixels in the image height. Therefore in the case of the 512×512 images $F = \max(1, \mathrm{round}(512/256)) = 2$, so the image is averaged within a 2×2 window and downsampled by a factor of 2. This is performed on each image before the comparison.

Figure 8 and Figure 9 highlight expected outcome two and provide the results. The hatched cover image is still very legible with the embedded VC share. The second share can then be used to recover the secret. This provides a very useful technique for practical applications in terms of verification and identification of images. This type of technique could be adapted to check the validity of currency because of its secure properties previously discussed. The hatched mask scheme shows a visual



(a) (b) (c)

Fig. 10. (a) Hatched Capitol Building containing share one; (b) hatched Mandrill containing share two; (c) superimposing (a) and (b).

Table II. An SSIM Index Comparison between the Original Hatched Image and its Corresponding Embedded Hatched Image

Hatched Images	SSIM Value After VC Embedding
Capitol Building 500 × 334 - Figure 8(a) and 9(a)	0.996
Capitol Building 512 × 512 - Figure 10(a)	0.9995
Mandrill 512 × 512 - Figure 10(b)	0.9996
Airplane 512 × 512	0.9005
APC 512 × 512	0.9169
Boat 512 × 512	0.9177
Car and APC 512 × 512	0.9189
Couple 512 × 512	0.9167
Elaine 512 × 512	0.9078
F-16 512 × 512	0.8968
House 512 × 512	0.9043
Lena 512 × 512	0.9147
Peppers 512 × 512	0.9185
Sailboat on Lake 512 × 512	0.9107
Splash 512 × 512	0.9131
Stream with Bridge 512 × 512	0.9147
Tiffany 512 × 512	0.8443
Truck 512 × 512	0.9196
Truck with APC 512 × 512	0.9191

Results are also included of other images from the test database.

improvement in the secret recovery. This is due to the subjective empirical observation that even at the same contrast a lighter image is easier to perceive by the human eye [Hofmeister et al. 2000].

Figure 10 shows the expected outcome three. Two VC shares are hidden within two separate hatched images. Figure 10(a) and 10(b) illustrate the final hatched images with the embedded shares. The resultant secret after superimposing each of these hatched images can be viewed in Figure 10(c). The secret is visible, however, due to the nature of this scheme, the contrast suffers when the secret is recovered. The text is clearly visible, but this would be a potential area for improvement.

To objectively compare the before and after embedding image hatching techniques a return was made to the SSIM index. Table II provides the results of the SSIM between the original images used

during these experiments and their equivalent images after a VC share has been embedded. Experiments were also carried out with a number of other images from the test database. These images were converted to their hatched equivalent images and then where compared against their hatched embedded image.

As one can see, even after embedding, the image hatching technique produces a very high structural similarity index when compared against the original image. This helps reinforce the point that even after a VC share is embedded into the image, it still remains very similar to the original. It also highlights the point that not all images are suitable for this type of embedding. Some of the results, such as the Tiffany image, resulted in relatively low SSIM index of just 0.8443 when viewed against the remainder of the images which where all above an SSIM index of 0.9.

5. SECURITY ANALYSIS

The security of the scheme rests with both VC and the cover images used. First, the chosen size-invariant VC scheme is secure in that given any amount of subpixels from a single share, it is impossible to tell if the corresponding shares' subpixels represent a black or a white pixel after superimposing them. This means that even if the complete VC share pattern is discovered within the cover image, working out its corresponding share is highly difficult due to its random nature. While the cover image and its corresponding share are separate, nothing about the secret can be deduced based on analysis of either the cover image or the revealing share.

This can be illustrated using a probabilistic proof. For a random secret, it cannot be assumed that the pixel values selected to represent those from the secret are uniformly distributed. This is down to the size-invariant scheme, in that one pixel from the secret has to be represented by one pixel from one of the shares, while the second share must also contain just a single pixel while keeping the value of the secret pixel hidden.

If a black pixel is to be represented from the secret, then its corresponding pattern is always black. Conversely, if a white pixel is to be represented then it can have two possible representations. A white pixel in each share is possible, or a a white pixel in share one with a black pixel in the second share, which ultimately ends up as a black pixel, but does indeed represent a white pixel from the secret.

So if a pixel is examined and found to be black in one share, the probability that it is black in the second share is 0.5. However, if the pixel in the share is white then there is also a probability of 0.5 that the pixel will be either black or white.

Overall, the general distribution of pixels would be modeled such that a white pixel can be represented both by white and black patterns while black can only be represented by black. This means the overall probability for the representation of a completely black pixel is $0.3\dot{3}$ while the white has a probability of $0.6\dot{6}$. This makes it very difficult to analyze the secret based on these probabilities due to the nature of the white pixel representations which can also take the form of a black pixel.

This means the security of the algorithm lies within the distribution of the white pixel patterns. The resulting probabilities make accurate analysis difficult; as mentioned, this is down to the patterns chosen and their distributions, along with the random nature of how the patterns are selected.

The net result is a darker overall share due to its nonuniformly distributed pixels but does provide adequate security for this scheme to be used in practise. It also prevents any information leaking out which may assist with the secret recovery without having the corresponding share.

Secondly, the hatched image is visually pleasing. It was verified in Section 4 that the SSIM index of our original hatched image, compared against the resulting hatched image, which has a VC share embedded within it, remain very similar. This is a good indication that our algorithm is highly effective at embedding additional data within the hatched images without distorting the final result.

6. CONCLUSION

In this article, three techniques have been developed which allow the creation of a hatched image to be used in conjunction with visual cryptography.

First, an image hatching scheme was presented. This technique can be used successfully for image sharing. For image hatching, generating similar patterns to those used in the engraving and printing of currency can be accomplished. Generating these hatched images using a threshold-based approach has proved very effective and easy to implement. One of the key strengths of the scheme is that it can take a multitude of images and apply these hatching styles to them. No specific type of image is required, any image can be used with this scheme, color, black and white, and any size. This would allow for easy application in the currency domain with respect to generating suitable images based on current up-to-date techniques.

Second, this image hatching scheme was combined with visual cryptography in two ways using traditional random shares. The first way involved placing a random share over the hatched image to recover the secret, the second involved a random share that was made up using hatching patterns which formed a mesh-like structure; this was employed to perform the recovery.

Thirdly, two hatched images were presented, each of which contained a share. Superimposing these hatched images allows the recovering of the secret. Having meaningful shares similar to extended visual cryptography is highly important, because it gives no indication whether any encryption would have taken place. This is backed up by the use of the SSIM index for image comparison. It is clear that, based on the SSIM index, the embedding process makes very few changes to the overall structure of the original image.

Using these hatched patterns, various ways have been described in order to embed basic VC shares within said patterns. The dot patterns are small enough so that they can be hidden within similar regions of an appropriate image, but also remain very legible and clear after the secret recovery has taken place. This was confirmed in the experimental results using the SSIM metric.

As previously mentioned, this type of secret embedding could have various potential applications, particulary within the banking industry. Protecting high value cheques, for authentication and identification purposes, using the techniques presented within this article could be easily implemented.

REFERENCES

ATENIESE, G., BLUNDO, C., DE SANTIS, A., AND STINSON, D. R. 1996. Extended schemes for visual cryptography. *Theor. Comput. Sci. 250*, 1–16.

BLUNDO, C., D'ARCO, P., DE SANTIS, A., AND 368 STINSON, D. R. 2003. Contrast optimal threshold visual cryptography schemes. *SIAM J. Discr. Math. 16,* 2, 224–261.

ELBER, G. 1995a. Line art rendering via a coverage of isoparametric curves. *IEEE Trans. Vis. Comput. Graph. 1,* 3, 231–239.

ELBER, G. 1995b. Line illustrations in computer graphics. *Visu. Comput. 11,* 6, 290–296.

ELBER, G. 1998a. Line art illustrations of parametric and implicit forms. *IEEE Trans. Visu. Comput. Graph. 4,* 1, 71–81.

ELBER, G. 1998b. Line art illustrations of parametric and implicit forms. *IEEE Trans. Visu. Comput. Graph. 4,* 1, 71–81.

FENG, J.-B., WU, H.-C., TSAI, C.-S., CHANG, Y.-F., AND CHU, Y.-P. 2008. Visual secret sharing for multiple secrets. *Pattern Recogn. 41,* 12, 3572–3581.

HASLUCK, P. N. 1977. *Manual of Traditional Wood Carving*. Dover, New York.

HOFMEISTER, T., KRAUSE, M., AND SIMON, H.-U. 2000. Contrast-Optimal k out of n secret sharing schemes in visual cryptography. *Theor. Comput. Sci. 240,* 2, 471–485.

HOU, Y. C., CHANG, C. Y., AND TU, S. F. 2001. Visual cryptography for color images based on halftone technology. In *Proceedings of International Conference on Image, Acoustic, Speech and Signal Processing*, Part 2.

HSU, H.-C., CHEN, T.-S., AND LIN, Y.-H. 2004. The ringed shadow image technology of visual cryptography by applying diverse rotating angles to hide the secret sharing. *Netw. Sens. Control 2*, 996–1001.

ITO, R., KUWAKADO, H., AND TANAKA, H. 1999. Image size invariant visual cryptography. *Inst. Electron. Inf. Commun. Engin. Trans. E82-A, 10*, 2172–2177.

IVINS, W. M. 1987. *How Prints Look, Photographs with Commentary, revised ed.* Beacon Press, Boston, MA.

IVINS, W. M. 1992. *Prints and Visual Communication*. MIT Press.

LAU, D. L. AND ARCE, G. R. 2000. *Modern Digital Halftoning*. Marcel Dekker.

MEMON, N. AND WONG, P. W. 1998. Protecting digital media content. *Commun. ACM 41,* 7, 35–43.

OSTROMOUKHOV, V. 1999. Digital facial engraving. In *Proceedings of the ACM SIGGRAPH '99 Conference*. 417–424.

PRAUN, E., HOPPE, H., WEBB, M., AND FINKELSTEIN, A. 2001. Real-Time hatching. In *Proceedings of the ACM SIGGRAPH '01 Conference*. ACM, New York, 581.

SALISBURY, M., ANDERSON, C., LISCHINSKI, D., AND SALESIN, D. H. 1996. Scale-Dependent reproduction of pen-and-ink illustrations. In *Proceedings of the ACM SIGGRAPH '96 Conference*. ACM, New York, 461–468.

SALISBURY, M. P., ANDERSON, S. E., BARZEL, R., AND SALESIN, D. H. 1994. Interactive pen-and-ink illustration. In *Proceedings of the ACM SIGGRAPH '94 Conference*. ACM, New York, 101–108.

SHAMIR, A. 1979. How to share a secret. *Commun. ACM 22,* 11, 612–613.

SHYU, S. J. 2006. Efficient visual secret sharing scheme for color images. *Pattern Recogn. 39,* 5, 866–880.

SHYU, S. J., HUANG, S.-Y., LEE, Y.-K., WANG, R.-Z., AND CHEN, K. 2007. Sharing multiple secrets in visual cryptography. *Pattern Recogn. 40,* 12, 3633–3651.

WANG, Z., ARCE, G., AND DI CRESCENZO, G. 2009. Halftone visual cryptography via error diffusion. *IEEE Trans. Inf. Forensics Secur. 4,* 3, 383–396.

WANG, Z. AND BOVIK, A. 2009. Mean squared error: Love it or leave it? A new look at signal fidelity measures. *IEEE Signal Process. Mag. 26,* 1, 98–117.

WANG, Z., BOVIK, A., SHEIKH, H., AND SIMONCELLI, E. 2004. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process. 13,* 4, 600–612.

WEIR, J. AND YAN, W. 2008. *Multimedia Communication Security: Recent Advances*. Nova Science Publishers, Chapter Visual Cryptography: A Survey.

WEIR, J. AND YAN, W.-Q. 2009a. Dot-Size variant visual cryptography. In *Proceedings of the International Workshop on Digital Watermarking (IWDW)*. 136–148.

WEIR, J. AND YAN, W.-Q. 2009b. Sharing multiple secrets using visual cryptography. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'09)*.

YANG, C.-N., WANG, C.-C., AND CHEN, T.-S. 2006. Real perfect contrast visual secret sharing schemes with reversing. In *Applied Cryptography and Network Security*. 433–447.

ZHOU, Z., ARCE, G. R., AND DI CRESCENZO, G. 2006. Halftone visual cryptography. *IEEE Trans. Image Process. 15,* 8, 2441–2453.

Received February 2011; revised May 2011; accepted May 2011