# A spatial domain secret image embedding technique with image authentication feature

S.K. Felix Yu*, Zi-Xin Xu†, Yuk-Hee Chan* and Pak-Kong Lun*

Email: sheung-kan.yu@connect.polyu.hk, z.xu@cuit.edu.cn, enyhchan@polyu.edu.hk, enpklun@polyu.edu.hk

*The Hong Kong Polytechnic University, Hong Kong, HKSAR

†Chengdu University of Information Technology, Chengdu, China

*Abstract*— **In practical applications, it is required to provide a means to authenticate an information-embedded image such that its integrity can be guaranteed. However, conventional studies generally consider image hiding and image authentication as two different tasks. When both are required, the secret image and a fragile watermark are separately embedded into a cover image. In this paper, to address this issue, we propose a spatial domain image embedding scheme that can embed rich pictorial information and fragile watermark simultaneously into a cover image with the same technique to reduce the complexity and improve the efficiency.**

## I. Introduction

Consider the case that one wants to send a messenger to deliver a digital image with a secret message in image form inside to another person. The receiver needs both the cover image (i.e. the image used to carry the secret image) and the secret image to get the full picture of the information while the other people can only get the partial information based on the received image. To secure the message, the messenger should have no idea about the secret image and he just needs to deliver the stego-image (i.e. the result of embedding the secret image into the cover image). When the receiver receives the stego-image, he has to, based on the received image alone, determine whether the image is really from the sender, whether the image has been tampered on its way and which parts of the image are tampered if tamper is detected. In such a case, it is required to embed the secret image and a fragile watermark simultaneously into the cover image.

The fragile watermark is used to guarantee the integrity or authenticity of the cover image and the secret image. In fact, it is also required for the same purpose even when the stego-image is delivered through the Internet. However, conventional studies generally consider image hiding and image authentication as two different tasks and their dedicated algorithms are separately developed[1-3]. Accordingly, when both secret image and fragile watermark are needed, they are embedded into the cover image one by one separately. To reduce the embedding overhead and simplify the operational structure, the fragile watermark and the secret image should be handled under the same embedding framework. It explains why the secret image and the fragile watermark should be fused together and spatially anchored to the cover image.

Image steganography is the study of embedding sensitive information in images without distorting their visual quality [1]. Another related study is on watermarking [2,3]. In their typical applications, the data to be embedded are generally a text string, a binary or bitmap logo, or a scrambled bit sequence. Their required embedding capacity is small. When a secret image is involved, it is generally compressed significantly before being embedded. The resultant bit stream is then embedded into the cover image as if it were a typical bit sequence. Since the embedding should only introduce a transparent distortion to the cover image, the embedding capacity is actually very limited. When the secret image is a true color image that is as large as the cover image, a very high compression ratio is required. As a result, the quality of the reconstructed color secret image can be very low. It explains why in conventional applications the size of the secret image is much smaller than that of the cover image and the secret image is generally not a natural color image but a bitmap logo. Though there have been quite a number of proposed algorithms to embed information in images, it is rarely to find an algorithm that embeds a natural color secret image into a grayscale image of the same size.

Image steganography/watermarking algorithms can be roughly classified to spatial domain or frequency domain ones. Least significant bit (LSB) substitution is a technique widely used in spatial domain image steganography [4]. Its basic idea is to replace the LSBs of some selected pixels with segments of message bits. It can work with other techniques flexibility to provide a good performance. For example, the message can be encrypted with a key to enhance the data security. It can be duplicated, scrambled, and then distributed over the image randomly to increase its robustness to attack. To reduce the pixel distortion caused by the substitution, an optimal pixel adjustment process (OPAP) [5,6] can be used after the substitution. LSB substitution is vulnerable to attacks. However, this becomes an advantage in our application as this property can be exploited in the construction of a fragile watermark that should be very sensitive to any change in the spatial content of the stego-image and be able to locate tampered regions.

In this paper, we propose a spatial domain embedding scheme that is able to support both fragile image watermarking and image steganography. It allows one to simultaneously embed a secret natural image and a fragile image watermark into a grayscale cover image. Tamper detection can be carried out without knowing the original cover image and it can locate tampered regions accurately.

The rest of this paper is organized as follows. Section 2 presents a spatial domain embedding scheme that can allows

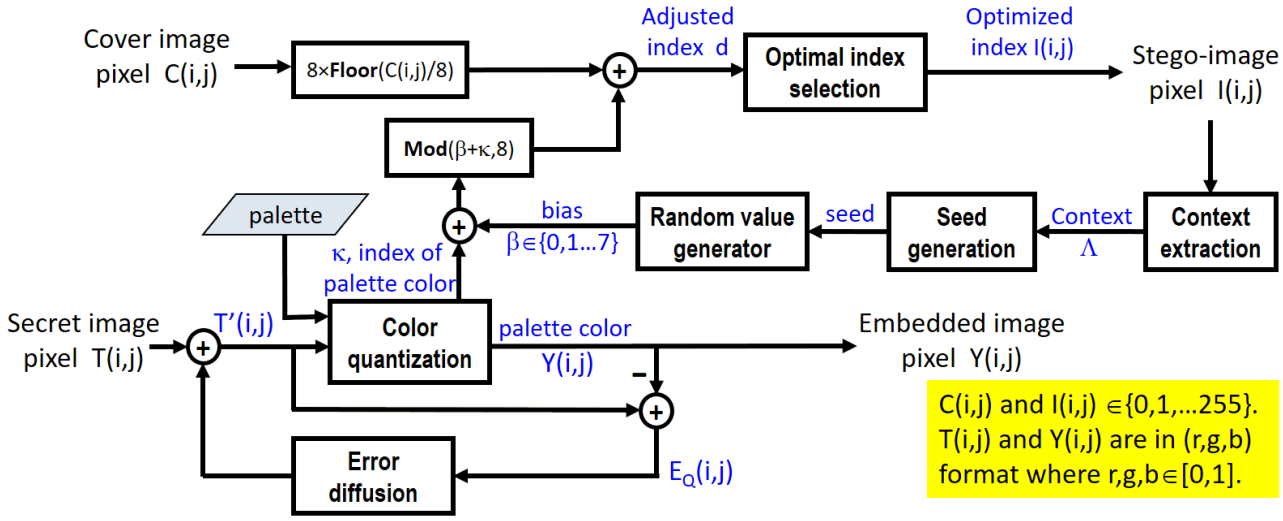
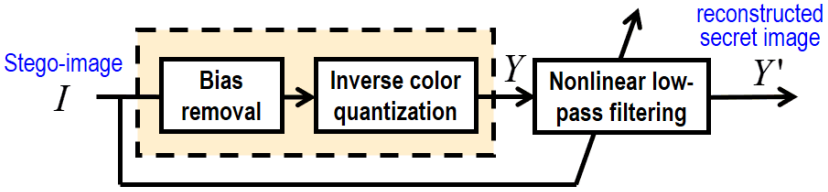
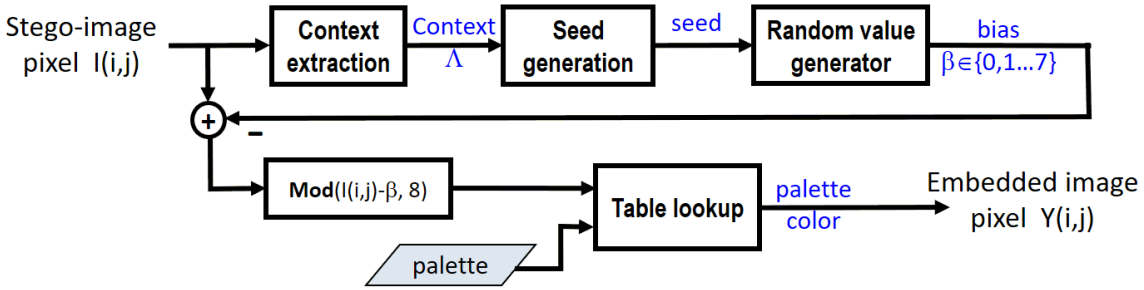



Fig. 1 Operation flow of the encoder




(a) Decoder




(b) Details of the *I*-to-*Y* transformation module in (a)

Fig. 2 Operation flow of the decoder

one to embed a secret color image into a grayscale cover image. Section 3 shows how fragile image watermarking can be realized with the scheme proposed in Section 2. It allows one to embed a grayscale secret image and a fragile watermark simultaneously into a grayscale cover image under the same embedding scheme. Section 4 provides some simulation results for studying the performance of the proposed scheme and a conclusion is given in Section 5.

## II. Proposed Method

Consider the case that we want to embed a color secret image $T$ into a grayscale cover image $C$ to produce a grayscale stego-image $I$ in which we can find self-contained information to reconstruct a color image $Y$ that is as close to $T$ as possible. Without loss of generality, we assume that all color images are in RGB format and the color of each of their pixels is represented as a vector in $(r,g,b)$ format, where $r$, $g$ and $b$ are, respectively, the intensity values of the red, the green and the blue components of the pixel. The components are normalized such that $r$, $g$ and $b \in [0,1]$. It is assumed that the size of images $T$ and $C$ are the same. In case they are not of equal size, we can resize or duplicate one of them to make it happen.

### A. Color palette

A color palette is required in the proposed embedding method to define the color of the embedded image in the stego-image. Without loss of generality, we assume that the palette is of size 256.

In our applications, the palette should bear two properties. First, the index value of the palette color assigned to a particular pixel of secret image $T$ should be anchored to the pixel value of the corresponding pixel in cover image $C$ to some

extent. Second, one can render a pixel color of image $T$ with several palette colors that are assigned similar index values by using the halftoning technique [7].

By considering that a color printer can render a color image with only 4 different inks (i.e. C, M, Y and K), we select black, blue, green, cyan, red, magenta, yellow and white as a set of primary colors and then repeatedly use them to develop a 256-color palette. In formulation, the palette colors are defined to be

$$\vec{p}_{(8k+m)} = \begin{cases} (0,0,0) & \text{if } m=0 \\ (0,0,1) & \text{if } m=1 \\ (0,1,0) & \text{if } m=2 \\ (0,1,1) & \text{if } m=3 \\ (1,0,0) & \text{if } m=4 \\ (1,0,1) & \text{if } m=5 \\ (1,1,0) & \text{if } m=6 \\ (1,1,1) & \text{if } m=7 \end{cases}, \text{ for } k=0,1\ldots31 \quad (1)$$

where $\vec{p}_n$ is the $n^{\text{th}}$ palette color of the 256-color palette.

### B. Encoding

The encoding process embeds the color image $T$ into the grayscale cover image $C$ to produce a stego-image $I$ with a halftoning technique. Specifically, color image $T$ is rendered with the 8 different primary colors in the palette to produce $Y$ before being embedded into the cover image. The reduction of colors in $Y$, as compared with $T$, introduces color quantization noise, but halftoning helps to shape the noise into high frequency noise such that the noise can be invisible due to the fact that our human visual system behaves as a lowpass filter [8].

Figure 1 shows the operation flow of the encoding process. Specifically, it scans the cover image $C$ with serpentine scanning and processes the image pixel by pixel to produce $I$. When processing a particular pixel $(i,j)$, $C(i,j)$, $T(i,j)$ and the processing results of some neighboring pixels of pixel $(i,j)$ are used to derive $I(i,j)$ and $Y(i,j)$.

Let $\mathcal{N}_{(i,j)}$ be the set of the coordinates of the processed neighboring pixels that are involved in the derivation of $I(i,j)$ and $Y(i,j)$. Before processing pixel $(i,j)$, the encoder diffuses the color quantization errors of $T(m,n)$ for $(m,n) \in \mathcal{N}_{(i,j)}$ to $T(i,j)$ with an error diffusion process. Let us assume that the net effect of the error diffusion process on $T(i,j)$ is to adjust $T(i,j)$ to $T'(i,j)$. $Y(i,j)$ is determined by quantizing $T'(i,j)$ to the palette color that is closest to $T'(i,j)$. In formulation, we have

$$Y(i,j) = \vec{p}_{(8\lfloor C(i,j)/8 \rfloor + \kappa)} \quad (2)$$

where

$$\kappa = \arg \min_{k=0,1,\ldots 7} \left\| T'(i,j) - \vec{p}_{(8\lfloor C(i,j)/8 \rfloor + k)} \right\|^2 \quad (3)$$

The quantization error of $T(i,j)$ is defined as

$$E_Q(i,j) = T'(i,j) - Y(i,j) \quad (4)$$

and it should then be diffused to the not-yet-processed neighboring pixels of $T(i,j)$ using a diffusion filter. The consequence is that the color of the affected neighboring pixels will be adjusted. Eventually, at the time when its neighboring pixel, say pixel $(p,q)$, is processed, the color to be color-quantized is

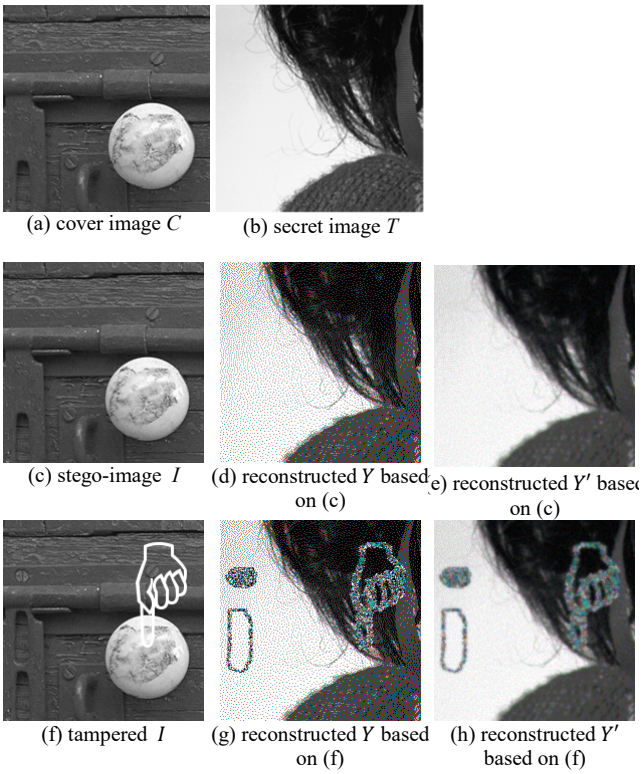
$$T'(p,q) = T(p,q) + \sum_{(m,n)\in\Theta} E_Q(p-m, q-n) h(m,n) \quad (5)$$

where $\Theta = \{(m,n) \mid m=0,1 \text{ and } n=0,\pm 1\}$ is the support of the diffusion filter H and $h(m,n)$ is the $(m,n)^{\text{th}}$ coefficient of filter H for $(m,n) \in \Theta$. In our realization, we adopt the diffusion filter that is commonly used in binary halftoning [9]. The diffusion is performed in the three color channels separately.

Theoretically, $I(i,j)$ should be assigned to be $8\lfloor C(i,j)/8 \rfloor + \kappa$, the index value of the palette color assigned to $Y(i,j)$. However, we add a random bias value to it so as to hide the embedded color image from unauthorized users and make attackers not able to forge an image that can pass tamper detection.

The bias, referred to as $\beta$, is a random integer value in range [0,7]. It is generated with a seed derived based on a context of $I(i,j)$. In our realization, the context is selected to be $\{I(i,j\text{-}1), I(i\text{-}1,j\text{-}2), I(i\text{-}1,j), I(i\text{-}2,j\text{+}1)\}$. It is possible to select other pixel combinations to form the context as long as the involved pixels were processed such that they are well-defined at the time when pixel $(i,j)$ is processed.

The biased index of $Y(i,j)$, that is given as



(a) cover image $C$ (b) secret image $T$ (c) stego-image $I$ (d) reconstructed $Y$ based on (c) (e) reconstructed $Y'$ based on (c) (f) tampered $I$ (g) reconstructed $Y$ based on (f) (h) reconstructed $Y'$ based on (f)

Fig. 3 Use of the proposed embedding scheme in fragile watermarking. (a) original cover image; (b) original secret image; (c) stego-image obtained with palette (8); (d) intermediate reconstructed secret image based on (c); (e) final reconstructed secret image based on (c); (f) tampered (c); (g) intermediate reconstructed secret image based on (f); (h) final reconstructed secret image based on (f). Image T is actually a color image each pixel of which has identical $r$, $g$ and $b$ component values.

$$d = 8\lfloor C(i,j)/8 \rfloor + mod(\kappa + \beta, 8) \quad (6)$$

guarantees that its maximum absolute difference from $C(i,j)$ is bounded by 7. To further lower the bound of the maximum absolute error to 4, we conditionally adjust $I(i,j)$ to be

$$I(i,j) = \arg\min_{k \in \Lambda_d} |k - C(i,j)| \quad (7)$$

where $\Lambda_d$={$l$ | $l$=$d$+8$n$ for $n \in$ {0,±1} and $l \in$ {0,1,…255}}.

After processing pixel $(i,j)$, we proceed to process the next pixel until all pixels are processed.

### C. Decoding

Figure 2 shows the operation flow for recovering the embedded color image from $I$. As long as the palette, the context selection and the seed derivation method are known, authorized users can, for each pixel $(i,j)$, determine its bias value and then derive the true index value based on $I(i,j)$ to locate the palette color assigned to $Y(i,j)$. The reconstructed $Y$ is actually a color halftone of the original secret color image $T$ and its noise is mainly high frequency noise. One can apply a non-linear low pass filter to restore its image quality. In our realization, we exploit the halftoning artifacts suppression process suggested in [10]. The final reconstructed color image is denoted as $Y'$. Readers can refer to [10] for the details.

## III. Fragile Watermarking

The embedding scheme presented in Section II can be used to realize fragile watermarking. Consider the case that color image $T$ does not carry any chrominance information in a way that each of its pixel has identical red, green and blue components and we change the 256-color palette to

$$\vec{p}_{(8k+m)} = \begin{cases} (0,0,0) & \text{if } m = 0 \\ (146,32,80)/255 & \text{if } m = 1 \\ (56,87,0)/255 & \text{if } m = 2 \\ (0,92,147)/255 & \text{if } m = 3 \\ (0,185,161)/255 & \text{if } m = 4 \\ (165,150,240)/255 & \text{if } m = 5 \\ (224,143,85)/255 & \text{if } m = 6 \\ (1,1,1) & \text{if } m = 7 \end{cases}, \quad \text{for } k=0,1\ldots31 \quad (8)$$

This palette is a non-grayscale palette constructed with another set of 8 different primary palette colors. In CIELAB color space, the coordinates of these 8 primary palette colors in ($L$,$a$,$b$) format are (0,0,0), (100,0,0), (100/3, 50cosθ, 50sinθ) and (200/3, -50cosθ, -50sinθ) for θ=0, 2π/3 and 4π/3, where $L$ is the luminance component bounded in range [0,100], and $a$ and $b$ are the chrominance components. These primary palette colors are purposely selected such that, in CLELAB domain, colors $\frac{1}{3}\sum_{m=1,2,3}\vec{p}_{(8k+m)}$ and $\frac{1}{3}\sum_{m=4,5,6}\vec{p}_{(8k+m)}$ for $k$ = 0, 1,…31 are all pure grayscales that do not carry any chrominance energy. This property allows us to render a grayscale image with the non-grayscale palette colors easily. In fact, we can use some other palettes instead of the one specified in eqn.(8) as long as the used palette bears the aforementioned property. The flexibility in palette selection adds extra difficulty for unauthorized people to guess the secret image without knowing the details.

When we use the encoding process presented in Section II to embed image $T$ into $C$ and then decode the resultant $I$, the decoding result (i.e. the reconstructed color image $Y'$) should also contain little chrominance energy as $T$ contains none. In other words, it should appear as a grayscale image even though $Y$ is rendered with a non-grayscale palette. As an example, Figs. 3(c)-(e) show, respectively, the corresponding $I$, $Y$ and $Y'$ obtained when we embed the secret image shown in Fig. 3(b) into the cover image shown in Fig. 3(a).

In fact, since the palette colors are mainly not gray levels, image $Y$ actually carries a lot chrominance energy. That image $Y'$ carries little chrominance energy is because, with the help of halftoning, the pixels in a local region of image $Y$ are arranged in a way that their chrominance intensity values can be close to zero after lowpass filtering. If tampering image $I$ results in a damage of this arrangement, the nonlinear lowpass filtering process cannot attenuate the chrominance components effectively and a visible color trace will appear in image $Y'$.

The context-based random bias introduced in the encoding process reinforces the visibility of tampering and resists various potential forge attacks. It makes attackers difficult to guess the palette because the same index value can map to different palette colors. Even if the attackers know the palette, since the index-to-color mapping is spatially variant and context dependent, the number of possible $I-\text{to}-Y$ mappings will be exponentially proportional to the image size. Hence, it is practically extremely difficult, if not impossible, to figure out the operation mechanism and the adopted parameter settings. Without knowing the palette, the scrambling key and the selected context, attackers cannot forge a stego-image that can pass the tamper detection. Figure 3(f) shows a tampered version of Fig. 3(c), and Figs. 3(g) and 3(h) show its decoding outputs.

The context-dependent feature of the scrambling process allows the decoder to detect whether the pre-arranged local pixel correlation in $I$ has been damaged or not. If an attacker replaces a region of $I$ with another region, the contexts of the pixels at the region boundary will be modified even the substitute is from an image produced with the same proposed algorithm. The decoder will then scramble the palette based on a seed different from the one used in the encoder. As a result, a different color will be obtained and there can be a visible color trace around the boundary as shown in Fig. 3(h). This explains why the proposed scheme can effectively thwart vector quantization attack [11] and collage attack [12].

## V. Simulation Results

Simulations were carried out to evaluate the performance of the proposed embedding algorithm. The testing image set is the Kodak set that includes 24 color images of size 768×512 or 512×768[13]. The testing images were divided into 2 groups. Group 1 contains images *Kodim01*, *Kodim02* and *Kodim03* and Group 2 contains those left behind. For each Group 2 testing image, its Y plane in YUV format was extracted to form a grayscale cover image. Images from Group 1 and their color-

| Scenario | | *Grayscale stego-image I* | | | | | *Recovered embedded color Image Y'* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | *dpi = 300* | *dpi = 600* | | | *dpi = 300* | | | *dpi = 600* | | |
| | | *PSNR (dB)* | *SSIM* | *GMSD* | *HVS-PSNR*† *(dB)* | *HVS-PSNR*† *(dB)* | *PSNR (dB)* | *FSIM* | *HVS-PSNR$_{Color}$*† *(dB)* | *CSSIM* | *$\Delta E_{S\text{-}CIELAB}$* | *HVS-PSNR$_{Color}$*† *(dB)* | *CSSIM* | *$\Delta E_{S\text{-}CIELAB}$* |
| 1 | Full-color secret image; using palette (1) | 40.59 | 0.991 | 0.005 | 53.679 | 58.346 | 23.57 | 0.931 | 36.550 | 0.887 | 1.197 | 40.115 | 0.901 | 0.878 |
| 2 | Color-removed secret image; using palette (8) | 40.59 | 0.991 | 0.005 | 53.704 | 58.352 | 26.55 | 0.939 | 39.203 | 0.940 | 0.644 | 45.041 | 0.946 | 0.438 |

† Viewing distance is 20 inches and dpi (dots/inch) is the image resolution.

Table 1 Performance of the proposed embedding scheme for Kodak image set [13] under different scenarios

| Scenario | | *Grayscale stego-image I* | | | | | *Recovered embedded color Image Y'* | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | *dpi = 300* | *dpi = 600* | | | *dpi = 300* | | | *dpi = 600* | | |
| | | *PSNR (dB)* | *SSIM* | *GMSD* | *HVS-PSNR*† *(dB)* | *HVS-PSNR*† *(dB)* | *PSNR (dB)* | *FSIM* | *HVS-PSNR$_{Color}$*† *(dB)* | *CSSIM* | *$\Delta E_{S\text{-}CIELAB}$* | *HVS-PSNR$_{Color}$*† *(dB)* | *CSSIM* | *$\Delta E_{S\text{-}CIELAB}$* |
| 1 | Embed a color secret halftone | 34.53 | 0.967 | 0.019 | 46.929 | 50.960 | 7.18 | 0.738 | 36.947 | 0.643 | 0.503 | 48.294 | 0.645 | 0.200 |
| 2 | Embed a grayscale secret halftone | 44.08 | 0.996 | 0.003 | 54.331 | 56.343 | 6.72 | 0.659 | 37.385 | 0.534 | 0.382 | 48.509 | 0.535 | 0.153 |

† Viewing distance is 20 inches and dpi (dots/inch) is the image resolution.

Table 2 Performance of a step-by-step embedding scheme for Kodak image set [13] under different scenarios

removed version were used as the secret images to be embedded. Here, the color-removed version of a test image is defined to be the color image whose *r*, *g* and *b* planes are all equal to the Y plane of the test image in YUV format.

We studied two scenarios. Scenarios 1 and 2 are, respectively, the scenarios discussed in Sections 2 and 3. In the former scenario, a true color natural image is embedded into a grayscale cover image with palette (1). In the latter scenario, a grayscale image is embedded into a grayscale cover image with palette (8).

For each scenario, each image from Group 1 (full-color version or color-removed version was picked according to the scenario being studied) was embedded into each image from Group 2 once to get a set of embedding results. Totally 63 sets of embedding results were obtained for each scenario. They were then used to evaluate the average image quality scores of the stego-images and the reconstructed secret images.

Table 1 shows the average quality performance of the proposed scheme in terms of various objective measures such as SSIM [14], GMSD [15], HVS-PSNR [16], FSIM [17], CSSIM [18], HVS-PSNR$_{Color}$ [10] and $\Delta E_{S\text{-}CIELAB}$ [19]. When evaluating HVS-PSNR$_{Color}$ and HVS-PSNR, the viewing distance is assumed to be 20 inches.

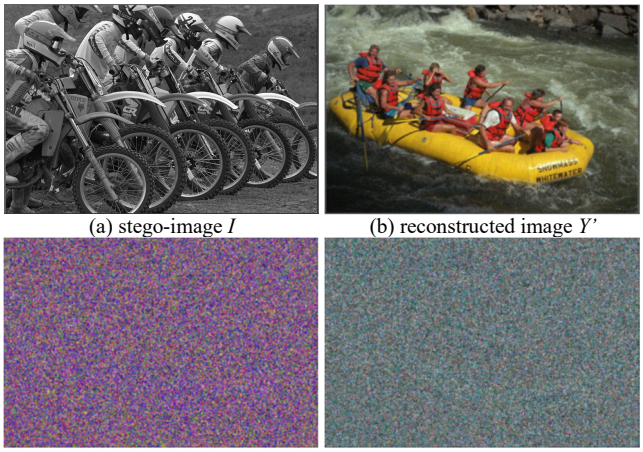
Figure 4 shows a particular set of evaluation results obtained in scenario 1. The reconstructed color image *Y* is actually a full-size color halftone of the original secret image. It is ready to be printed, so in principle it is a good rendering result of the secret image. Figures 4(c) and 4(d) show what will happen to the reconstructed secret image *Y'* if we use an improper decoder to decode Fig. 4(a). In the encoder, a random bias is added to the index value of the palette color assigned to each pixel of the embedded secret image. This bias should be removed at the decoder before we use the index value to locate the right palette color for the corresponding pixel. Figure 4(c) shows the case when we don't do it. Similar results will be obtained if we use a wrong context or a wrong seed to get an incorrect bias.

Figure 4(d) shows the case when we use palette (8) instead of palette (1) in the decoder on top of not removing the random bias. Using an unmatched palette interferes the decoding output further.

Without knowing the right palette, the scrambling key, the scrambling method or the used context, attackers can neither reconstruct the secret image nor the watermark properly. This also makes them impossible to forge a watermarked stego-image that can pass tamper detection. Figure 5 shows a set of evaluation results obtained in scenario 2. Similar observations can be obtained.

Figure 6 shows how the proposed embedding scheme helps to secure the integrity of the stego-image and the embedded secret image. Figure 6(a) shows a portion of a tampered version of Fig. 4(a). It is enlarged for better inspection. Various attacks including collage attack, VQ attack and constant-average attack were involved. Figure 6(b) shows the secret image reconstructed with a proper decoder based on Fig. 6(a). One can see that the proposed scheme can also detect and locate tampered regions when the secret image is a full-color image. Forgery attempt can leave a visible color trace in the reconstructed secret image unless the secret image contains very strong and complicated high frequency chrominance content which is able to bury the color trace. In practical situations and applications, it is rare to happen.

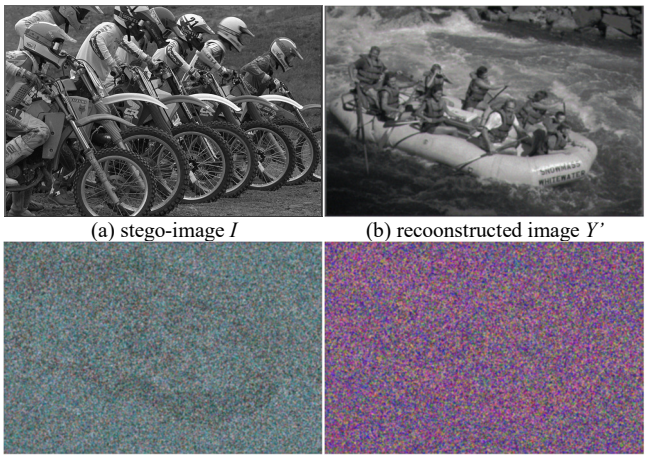
Figure 6(c) shows a tampered version of Fig. 5(a) and Fig. 6(d) shows the secret image reconstructed based on Fig. 6(c). At a glimpse, the color trace associated with tampered regions are more visible in Fig. 6(b) than Fig. 6(d), but it is actually easier to detect the trace in Fig. 6(d) in a straightforward manner. The clean regions in Fig. 6(d) carries very little chrominance energy, so applying simple thresholding on the chrominance energy plane of Fig. 6(d) can already locate the tampered regions. The forgery map shown in Figure 6(e) is



(a) stego-image *I* (b) reconstructed image *Y'*

(c) *Y'* obtained without removing the random bias (d) *Y'* obtained using a unmatched palette and without removing the random bias

Fig. 4 Simulation results of embedding a full color natural image (Kodim14) into a grayscale cover image (the Y plane of Kodim05 in YUV format) with palette (1). (a) stego-image *I*; (b)-(d) reconstructed secret image *Y'* under different conditions: (b) proper conditions; (c) not removing the random bias at the decoder; (d) using an unmatched palette (palette (8)) and not removing the random bias at the decoder.



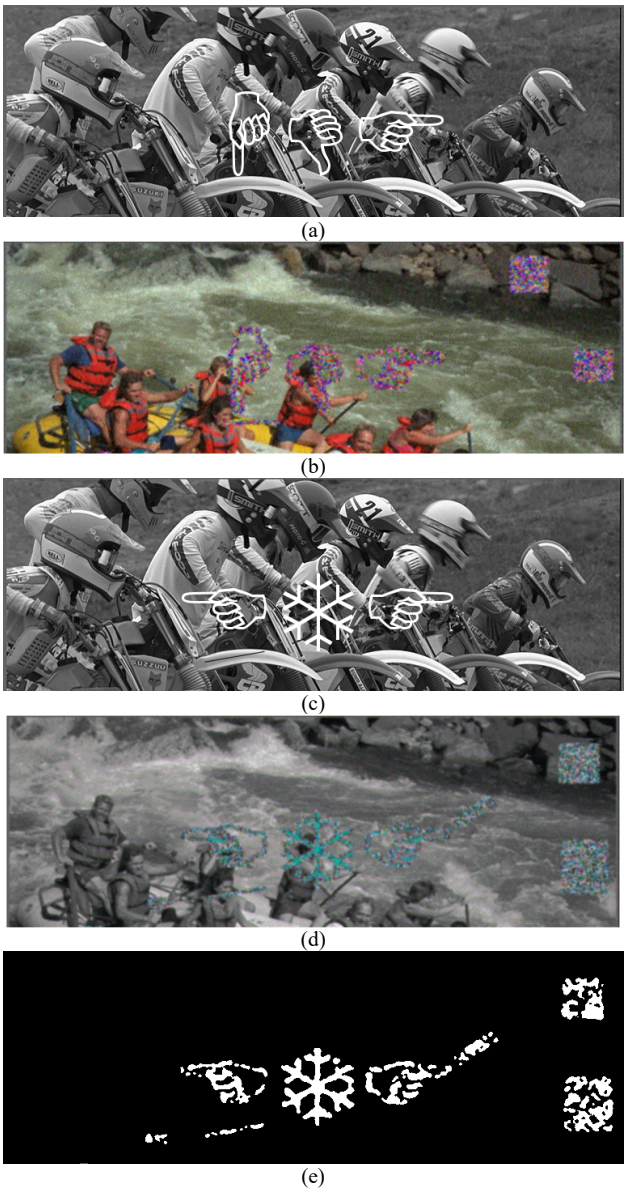
(a) stego-image *I* (b) recoonstructed image *Y'*

(c) *Y'* obtained without removing the random bias (d) *Y'* obtained using a unmatched palette and without removing the random bias

Fig. 5 Simulation results of embedding a grayscale natural image (color removed Kodim14) into a grayscale cover image (the Y plane of Kodim05 in YUV format) with palette (8). (a) stego-image *I*; (b)-(d) reconstructed secret image *Y'* under different conditions: (b) proper conditions; (c) not removing the random bias at the decoder; (d) using an unmatched palette (palette (1)) and not removing the random bias at the decoder.

deduced directly with simple thresholding and median filtering based on Fig. 6(d). As for Fig. 6(b), it needs more complicated computation to locate its tampered regions since the secret image is a full-color image itself. As a final remark, we note that the proposed scheme is able to detect the thin line drawn below the right hand icon in Fig. 6(c).

The proposed scheme allows one to embed a secret image and a fragile image watermark into a grayscale image with one single process. Besides convenience, it provides performance as it can effectively reduce embedding overhead. Theoretically, one can carry out separate conventional image steganography



(a) (b) (c) (d) (e)

Fig. 6 (a) tampered version of Fig. 4(a); (b) reconstructed secret image based on Fig. 6(a); (c) tampered version of Fig. 5(a); (d) reconstructed secret image based on Fig. 6(c); (e) forgery map deduced based on Fig.6(d).

and watermarking processes sequentially to embed both information. Consider the case that we adopt this straightforward step-by-step approach to embed a secret color image *T* into an 8-bit grayscale cover image *C* with the LSB substitution technique as follows: In stage 1, we convert color image *T* into a 3-bit color halftone, randomly shuffle pixels of the halftone with a secret key and then replace the 2nd, 3rd and 4th least significant bit planes of cover image *C* with the shuffled color halftone. In stage 2, we replace the least significant bit plane of cover image *C* with a fragile watermark by using the watermarking algorithm proposed in [20] and then carry out the optimal pixel adjustment process (OPAP) [21] to enhance the quality of the resultant stego-image.

When the secret image is a grayscale image instead of a color image, we can modify the 1st stage a bit to improve the

quality of the stego-image. Specifically, we can convert the secret image into a binary halftone instead of a color halftone and then replace the 2nd least significant bit plane of cover image *C* with the shuffled binary halftone.

For reference purpose, we refer to the case when the secret image is a color image as scenario 1 and the case when the secret image is a grayscale image as scenario 2. Table 2 shows the performance of the aforementioned step-by-step approach in handling the two scenarios. By comparing the 1st (2nd) rows in Tables 1 and 2, one can see the performance difference between the proposed approach and the step-by-step approach when handling a color (grayscale) secret image. In terms of the overall average PSNR of the reconstructed secret images and the grayscale stego-images, the performance of the proposed approach is 11.2 dB (8.2 dB) higher for scenario 1(2).

## VI. Conclusions

An information embedding scheme is developed in this paper. Under this scheme, it is possible to embed a secret image and a fragile watermark simultaneously into a grayscale image effectively such that the integrity of the information-embedded image and the embedded secret image can be guaranteed. It reduces the overhead and increases the efficiency as compared with the trivial two-step approach in which image steganography and watermark embedding are performed separately with independent algorithms.

## Acknowledgment

This work was supported by The Hong Kong Polytechnic University under Grant RK73 and ZZHM.

## References

[1] T. Morkel, J. H. P. Eloff, and M. S. Olivier. “An Overview of Image Steganography”. In: Proc. of the 5th Annual Information Security South Africa Conference (ISSA2005), June 2005

[2] Q. Su, *Color Image Watermarking: Algorithms and Technologies*. Berlin, Germany: Walter de Gruyter, 2017, pp. 1-26.

[3] Q. Su, D. Liu, Z. Yuan, G. Wang, X. Zhang, B. Chen and T. Yao, “New rapid and robust color image watermarking technique in spatial domain,” IEEE Access, 2019, pp.30398-30409

[4] A.Cheddad, J.Condell, K.Curran and P.McKevitt, "Digital image steganography: Survey and analysis of current methods," *Signal Process*., 90(3), 2010, pp.727–752

[5] H. Yang, X. Sun and G. Sun, "A high-capacity image data hiding scheme using adaptive LSB substitu-tion," *Radioengineering*, 18(4), 2009, pp.509-516

[6] C.K. Chan and L.M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recogn*., vol. 37, 2004, pp.469-474

[7] D. L. Lau and G. R. Arce, Modern Digital Halftoning, 2nd Ed., CRC Press, Boca Raton, FL, USA, 2008.

[8] Y. H. Fung and Y.H. Chan, "Tone-dependent noise model for high-quality halftones," Journal of Electronic Imaging, 22 (2), 023004-023004 (Apr 12, 2013). doi:10.1117/1.JEI.22.2.023004

[9] R.W. Floyd and L. Steinberg, "Adaptive algorithm for spatial greyscale," vol. 17, pp. 75-77, 1976.

[10] Z.X. Xu, Y.H. Chan, "Improving reversible color-to-grayscale conversion with halftoning," Signal Process. Image Commun. 52, pp. 111-123, 2017

[11] M. Holliman and N. Memon, "Counterfeiting attacks on oblivious blockwise independent invisible watermarking schemes," IEEE Transactions on Image Processing 3(10), 432–441 (2000)

[12] J.Fridrich, M.Goljan and N.Memon, "Cryptanalysis of the yeungmintzer fragile watermarking technique," Journal of Electronic Imaging 11(4), 262–274 (2002)

[13] Kodak true color image suite, [Online]. Available: http://r0k.us/graphics/kodak/

[14] Z. Wang, A.C. Bovik, H.R. Sheikh and E.P. Simoncell, "Image quality assessment: from error visibility to structural similarity," IEEE Trans. Image Process., vol. 13, p. 600–612, 2004.

[15] W. Xue, L. Zhang, X. Mou and A.C. Bovik, "Gradient magnitude similarity deviation: A highly efficient perceptual image quality index," IEEE trans. on Image Process., vol. 23, no. 2, pp. 684-695, 2014.

[16] J. M. Guo and Y. F. Liu, "Joint compression/ watermarking scheme using majority-parity guidance and halftoning-based block truncation coding," IEEE Trans. Image Process., 19(8), pp. 2056-2069, 2010.

[17] L. Zhang, L. Zhang, X. Mou and D. Zhang, "FSIM: A Feature SIMilarity Index for Image Quality Assessment," IEEE Trans. on Image Process., 20(8), pp. 2378 - 2386, 2011.

[18] M. Hassan and C. Bhagvati, "Structural similarity measure for color images," International Journal of Computer Applications, vol. 43, pp. 7-12, 2012.

[19] X. Zhang and B.A. Wandell, "A spatial extension of CIELAB for digital color image reproduction," Journal of the Society for Information Display, vol. 5, pp. 61-63, 1997.

[20] H. Zhang, C. Wang and X. Zhou, “Fragile watermarking based on LBP for Blind Tamper Detection in Images,” J Inf Process Syst, 13(2), pp.385-399, 2017

[21] C. K. Chan and L. M. Cheng, “Hiding data in images by simple LSB substitution,”. Pattern Recognition, Vol. 37, 2004, pp. 469-474