New Image Steganography via Secret-fragment-visible Mosaic Images by Nearly-reversible Color Transformation

Ya-Lin Li[1], Wen-Hsiang Tsai[2, 3]

[1] Institute of Computer Science and Engineering, National Chiao Tung University, Taiwan
[2] Department of Computer Science, National Chiao Tung University, Taiwan
[3] Department of Information Communication, Asia University, Taiwan

Abstract. A new image steganography method is proposed, which creates automatically from an arbitrarily-selected target image a so-called secret-fragment-visible mosaic image as a camouflage of a given secret image. The mosaic image is yielded by dividing the secret image into fragments and transforming their color characteristics to be those of the blocks of the target image. Skillful techniques are designed for use in the color transformation process so that the secret image may be recovered nearly losslessly. The method not only creates a steganographic effect useful for secure keeping of secret images, but also provides a new way to solve the difficulty of hiding secret images with huge data volumes into target images. Good experimental results show the feasibility of the proposed method.

1 Introduction

Steganography is the science of hiding secret messages into cover media so that no one can realize the existence of the secret data [1-2]. Existing steganography techniques may be classified into three categories — image, video, and text steganographies, and image steganography aims to embed a secret message into a *cover image* with the yielded *stego-image* looking like the original cover image. Many image steganography techniques have been proposed [1-4], and some of them try to hide *secret images* behind other images [3-4]. The main issue in these techniques is the difficulty to hide a huge amount of image data into the cover image without causing intolerable distortions in the stego-image.

Recently, Lai and Tsai [5] proposed a new type of computer art image, called secret-fragment-visible mosaic image, which is the result of random rearrangement of the fragments of a secret image in disguise of another image called *target image*, creating exactly an effect of image steganography. The above-mentioned difficulty of hiding a huge volume of image data behind a cover image is solved *automatically* by this type of mosaic image. In more detail, as illustrated by Fig. 1, a given secret image is first "chopped" into tiny rectangular fragments, and a target image with a similar color distribution is selected from a database. Then, the fragments are arranged in a random fashion controlled by a key to fit into the blocks of the target image, yielding a stego-image with a mosaic appearance. The stego-image preserves all the secret image fragments in appearance, but no one can figure out what the original secret image looks like. The method is a new way for secure keeping of secret images. However, a large image database is required in order to select a color-similar target image for each input secret image, so that the generated mosaic image can be

sufficiently similar to the selected target image. Using their method, a user is *not* allowed to select freely his/her favorite image for use as the target image.

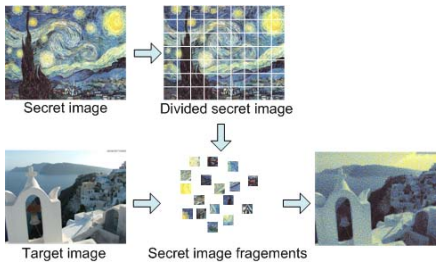

Fig. 1. Illustration of creation of secret-fragment-visible mosaic image proposed in [5].

Accordingly, we propose in this study a new method that creates secret-fragment-visible mosaic images with no need of a database; *any* image may be selected as the target image for a given secret image. Fig. 2 shows a result yielded by the proposed method. Specifically, after a target image is selected arbitrarily, the given secret image is first divided into rectangular fragments, which then are fit into similar blocks in the target image according to a similarity criterion based on color variations. Next, the color characteristic of each tile image is transformed to be that of the corresponding block in the target image, resulting in a mosaic image which looks like the target image. Such a type of camouflage image can be used for securely keeping of a secret image in disguise of any pre-selected target image. Relevant schemes are also proposed to conduct *nearly-lossless* recovery of the original secret image.

(a) (b) (c)

Fig. 2. A result yielded by proposed method. (a) Secret image. (b) Target image. (c) Secret-fragment-visible mosaic image created from (a) and (b).

In the remainder of this paper, the idea of the proposed method is described in Sections 2 and 3. Detailed algorithms for mosaic image creation and secret image recovery are given in Section 4. In Section 5, experimental results are presented to show the feasibility of the proposed method, followed by conclusions in Section 6.

2 Basic Idea of Proposed Method

The proposed method includes two main phases: mosaic image creation and secret image recovery. The first phase includes four stages: (1) stage 1.1 — fitting the tile images of a given secret image into the target blocks of a pre-selected target image; (2) stage 1.2 — transforming the color characteristic of each tile image in the secret image to become that of the corresponding target block in the target image; (3) stage

1.3 — rotating each tile image into a direction with the minimum RMSE value with respect to its corresponding target block; and (4) stage 1.4 — embedding relevant information into the created mosaic image for future recovery of the secret image. The second phase includes two stages: (1) stage 2.1 — extracting the embedded information for secret image recovery from the mosaic image; and (2) stage 2.2 — recovering the secret image using the extracted information.

3 Problems and Proposed Solutions for Mosaic Image Creation

The problems encountered in generating mosaic images by the proposed method are discussed in this section, and the proposed solutions to them are also presented.

(A) Color Transformations between Blocks

Suppose that in the first phase of the proposed method, a tile image T in a given secret image is to be fit into a target block B in a pre-selected target image. Since the color characteristics of T and B are different from each other, how to change their color distributions to make them look alike is the main issue here. Reinhard et al. [6] proposed a work about color transfer in this aspect, which converts the color characteristic of one image to be that of another in the lαβ color space. This idea is an answer to the issue and is adopted in this study. But instead of conducting color conversion in the lαβ color space, we do it in the RGB space to reduce the volume of the generated information which should be embedded in the created mosaic image for later recovery of the original secret image.

More specifically, let T and B be described as two pixel sets $\{p_1, p_2, \ldots, p_n\}$ and $\{p_1', p_2', \ldots, p_n'\}$, respectively, assuming that both blocks are of the same dimensions with size n. Let the color of pixel p_i in the RGB color space be denoted by (r_i, g_i, b_i) and that of p_i' by (r_i', g_i', b_i'). First, we compute the means and standard deviations of T and B, respectively, in each of the three color channels R, G, and B by the following formulas:

$$\mu_c = \frac{1}{n}\sum_{i=1}^{n} c_i, \quad \mu_c' = \frac{1}{n}\sum_{i=1}^{n} c_i'; \tag{1}$$

$$\sigma_c = \sqrt{(1/n)\sum_{i=1}^{n}(c_i - \mu_c)^2}, \quad \sigma_c' = \sqrt{(1/n)\sum_{i=1}^{n}(c_i' - \mu_c')^2} \tag{2}$$

where c_i and c_i' denote the C-channel values of pixels p_i and p_j', respectively, with c denoting r, g, b. Next, we compute new color values (r_i'', g_i'', b_i'') for each p_i in T by:

$$c_i'' = (\sigma_c'/\sigma_c)(c_i - \mu_c) + \mu_c' \text{ with } c = r, g, \text{ and } b. \tag{3}$$

This results in a new tile image T' with a new color characteristic similar to that of target block B. Also, we use the following formula, which is the inverse of Eq. (3), to compute the original color values (r_i, g_i, b_i) of p_i from the new ones (r_i'', g_i'', b_i''):

$$c_i = (\sigma_c / \sigma_c')(c_i'' - \mu_c') + \mu_c \text{ with } c = r, g, \text{ and } b. \tag{4}$$

Furthermore, we have to embed into the created mosaic image sufficient information about the transformed tile image T' for use in later recovery of the original secret image. For this, theoretically we can use Eq. (4) to compute the original pixel value of p_i. But the mean and standard deviation values are all real numbers, and it is not practical to embed real numbers, each with many digits, in the generated mosaic image. Therefore, we limit the numbers of bits used to represent a

mean or a standard deviation. Specifically, for each color channel we allow each of the means of T and B to have 8 bits with values 0 ~ 255, and the *standard deviation quotient* $q_c = \sigma_c'/\sigma_c$ to have 7 bits with values 0.1 ~ 12.8. We do *not* allow q_c to be 0 because otherwise the original pixel value cannot be recovered back by Eq. (4) for the reason that $\sigma_c/\sigma_c' = 1/q_c$ in Eq. (4) is not defined when $q_c = 0$, where $c = r, g, b$.

(B) Choosing Appropriate Target Blocks and Rotating Blocks to Fit Better

In transforming the color characteristic of a tile image T to be that of a corresponding target block B as described above, how to choose an appropriate B for each T (i.e., how to *fit* each T to a proper B) is an issue. If two blocks are more similar in color distributions originally, a better transformation effect will result. For this, we use the standard deviation of block colors as a measure to select the most similar target block B for each tile image T. First, we compute the standard deviations of every tile image and target block for each color channel. Then, we sort all the tile images to form a sequence, S_{tile}, and all the target blocks to form another, S_{target}, according to the *mean* of the standard deviation values of the three colors. Finally, we *fit* the first tile image in S_{tile} to the first target block in S_{target}; fit the second in S_{tile} to the second in S_{target}, etc.

Additionally, after a target block B is chosen for fitting a tile image T and after the color characteristic of T is transformed to be that of B as described above, we conduct a further improvement on the color similarity between the transformed T (denoted as T') and B by rotating T' into one of the four directions 0^o, 90^o, 180^o and 270^o, which yields a rotated version T'' of T' with the minimum RMSE value with respect to B among the four directions for final use to fit T into B. Fig. 3 shows an example of the result of applying this scheme to the secret image and target image shown in Figs. 3(a) and 3(b), respectively. Fig. 3(c) is the mosaic image created without applying this block rotation scheme and Fig. 3(d) is that created instead. We can see that Fig. 3(d) has a better fitting result with a smaller RMSE value than that of Fig. 3(c).

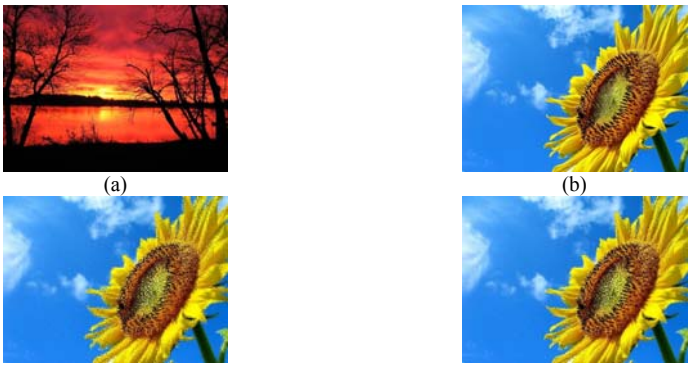

Fig. 3. Illustration of effect of rotating tile images before fitting them into target blocks. (a) Secret image. (b) Target image. (c) Mosaic image created from (a) and (b) without block rotations (with RMSE = 36.911 with respect to (b)). (d) Mosaic image created from (a) and (b) with block rotations (with RMSE = 32.382).

(C) Handling Overflows/Underflows in Color Transformation

After the color transformation process between a tile image T and a target block B is conducted as described before, some pixel values in the transformed block T' might have overflows or underflows. To deal with this problem, we convert such values to be non-overflow/non-underflow ones and record the value differences as *residuals* for

use in later recovery of the exact pixel values. Specifically, we convert all the transformed pixel values in T' not smaller than 255 to be 255, and all of those not larger than 0 to be 0. Next, we compute the differences between the original pixel values and the converted ones, 255 or 0, as the residuals and record them as information associated with T'. But as can be seen from Eq. (3), the bounds of possible residual values are unknown, and this causes a problem in deciding how many bits should be used to record a residual. To solve this problem, we record the residuals in the *un-transformed* color space rather than in the transformed one. That is, by using the following two formulas we compute first the smallest possible color value c_S (with $c = r$, g, and b) in tile image T that becomes larger than 255 as well as the largest possible value c_L in T that becomes smaller than 0, after the color transformation process has been conducted, as:

$$c_S = \left\lceil (1/q_c)(255 - c_\mu') + c_\mu \right\rceil;\ c_L = \left\lfloor (1/q_c)(0 - c_\mu') + c_\mu \right\rfloor, \tag{5}$$

respectively, where $q_c = \sigma_c'/\sigma_c$ as defined before. Then, for an un-transformed value c_i which becomes an overflow after the color transformation, we compute its residual as $|c_i - c_S|$; and for an un-transformed c_i which becomes an underflow, we compute its residual as $|c_L - c_i|$. Now, the possible values for the residuals of c_i are all in the range of $0 \sim 255$, therefore we can simply record each of them with 8 bits.

(D) Embedding Secret Image Recovery Information

In order to recover the secret image from the mosaic image, we have to embed relevant recovery information into the mosaic image. For this, we adopt a technique of reversible contrast mapping proposed by Coltuc and Chassery [7], which is applied to the least significant bits of the pixels in the created mosaic image to hide data. The information required to recover a tile image T which is mapped to a target block B includes: (1) the index of B; (2) the optimal rotation angle of T; (3) the means of T and B and the related standard deviation quotients of all color channels; and (4) the overflow/underflow residuals. These data are coded by binary strings respectively as $t_1t_2\ldots t_m$, r_1r_2, $m_1m_2\ldots m_{48}$, $q_1q_2\ldots q_{21}$, and $r_1\ldots r_k$, which together with the binary strings for encoding the values m and k are concatenated into a bit stream M for tile image T. Then, such bit streams of all the tile images are concatenated in order further into a *total bit stream* M_t for the entire secret image. Moreover, in order to protect M_t from being attacked, we encrypt it with a secret key to obtain an encrypted bit stream M_t', which finally is embedded into pixel pairs in the mosaic image using the method proposed in [7]. A plot of the statistics of the numbers of required bits for embedding M_t' into the generated mosaic images shown in this paper is shown in Fig. 6(b).

After embedding the bit stream M_t' into the mosaic image, we can recover the secret image back. But some *loss* will be incurred in the recovered secret image (i.e., the recovered image is not all identical to the original one). The loss occurs in the color transformation process using Eq. (3) where each pixel's color value c_i is multiplied by the standard deviation quotient $q_c = \sigma_c/\sigma_c'$ and the resulting real value c_i'' is truncated to be an integer in the range of 0 through 255. However, because each truncated part is smaller than the value of 1 when no overflow or underflow occurs, the recovered value of c_i using Eq. (4) is still precise enough. Even when overflows/underflows occur at some pixels in the color transformation process, we record their residual values as described previously and after using Eq. (4) to recover

the pixel value c_i, we can add the residual values back to the computed pixel values c_i to get the original exact pixel data, yielding a nearly-lossless recovered secret image. According to our experimental results, each recovered secret image has a high PSNR value in the range of 45~50 db with respect to the original secret image, or equivalently, has very a small RMSE value around just 1.0 with respect to the original secret image, as will be shown later in Section 5.

4 Mosaic Image Creation and Secret Image Recovery Algorithms

Based on the above discussions, detailed algorithms for mosaic image creation and secret image recovery may now be described.

Algorithm 1. ***Secret-fragment-visible mosaic image creation.***

Input: a secret image S with n tile images of size N_T; a pre-selected target image T of the same size of S; and a secret key K.

Output: a secret-fragment-visible mosaic image F.

Steps:

Stage 1.1 — fitting tile images into target blocks.

1. Divide secret image S into a sequence of n tile images of size N_T, denoted as $S_{tile} = \{T_1, T_2, \ldots, T_n\}$; and divide target image T into another sequence of n target blocks also with size N_T, denoted as $S_{target} = \{B_1, B_2, \ldots, B_n\}$.
2. Compute the means (μ_r, μ_g, μ_b) and the standard deviations $(\sigma_r, \sigma_g, \sigma_b)$ of each T_i in S_{tile} for the three color channels according to Eqs. (1) and (2); and compute the *average standard deviation* $\sigma_{Ti} = (\sigma_r + \sigma_g + \sigma_b)/3$ for T_i where $i = 1$ through n.
3. Do similarly to the last step to compute the means (μ_r', μ_g', μ_b'), the standard deviations $(\sigma_r', \sigma_g', \sigma_b')$, and the average standard deviation $\sigma_{Bj} = (\sigma_r' + \sigma_g' + \sigma_b')/3$ for each B_j in S_{target} where $j = 1$ through n.
4. Sort the blocks in S_{tile} and S_{target} according to the average standard deviation values of the blocks; map in order the blocks in the sorted S_{tile} to those in the sorted S_{target} in a 1-to-1 manner; and reorder the mappings according to the indices of the tile images into a mapping sequence L of the form of $T_1 \rightarrow B_{j_1}$, $T_2 \rightarrow B_{j_2}$, etc.
5. Create a mosaic image F by fitting the tile images of secret image S to the corresponding target blocks of target image T according to mapping sequence L.

Stage 1.2 — performing color conversion between the tile images and target blocks.

6. For each pair $T_i \rightarrow B_{j_i}$ in mapping sequence L, let the means μ_c and μ_c' of T_i and B_{j_i} respectively be represented by 8 bits with values 0~255 and the standard deviation quotients $q_c = \sigma_c'/\sigma_c$ by 7 bits with values 0.1~12.8 where $c = r, g, b$.
7. For each pixel p_i in each tile image T_i of mosaic image F with color value c_i where $c = r, g, b$, transform c_i into a new value c_i'' by Eq. (3); and if c_i'' is not smaller than 255 (i.e., if an overflow occurs) or if it is not larger than 0 (i.e., if an underflow occurs), assign c_i'' to be 255 or 0, respectively, and compute a residual value for pixel p_i by the way described in Section 3(C).

Stage 1.3 — rotating the tile images.

8. Compute the RMSE values of each color-transformed tile image T_i in F with respect to its corresponding target block B_{j_i} after rotating T_i into the directions 0^o, 90^o, 180^o and 270^o; and rotate T_i into the *optimal* direction θ_o with the smallest RMSE value.

Stage 1.4 — embedding the secret image recovery information.

9. For each tile image T_i in F, construct a bit stream M_i for recovering T_i as described in Section 3(D), including the bit-segments which encode the data items of: (1) the index of the corresponding target block B_{j_i}; (2) the optimal rotation angle θ_o of T_i; (3) the means of T_i and B_{j_i} and the related standard deviation quotients of all color channels; (4) the overflow/underflow residual values in T_i; (5) the number m of bits to encode the index of a block; and (6) the number k of residual values.
10. Concatenate the bit streams M_i of all T_i in F in a raster-scan order to form a total bit stream M_t; use the secret key K to encrypt M_t into another bit stream M_t'; and embed M_t' into F by reversible contrast mapping [7].

Algorithm 2. Secret image recovery.

Input: a mosaic image F with n tile images and the secret key K used in Algorithm 1.

Output: the secret image S embedded in F using Algorithm 1.

Steps:

Stage 2.1 — extracting the secret image recovery information.

1. Extract from mosaic image F the bit stream M_t' for secret image recovery by a reverse version of the reversible contrast mapping scheme proposed in [7] and decrypt M_t' using the secret key K into a non-encrypted version M_t.
2. Decompose M_t into n bit streams M_i for the *n to-be-constructed* tile images T_i in S, respectively, where $i = 1$ through n.
3. Decode the bit stream M_i of each tile image T_i to obtain the following data: (1) the index j_i of the block B_{j_i} in F corresponding to T_i; (2) the optimal rotation angle θ_o of T_i; (3) the means of T_i and B_{j_i} and the related standard deviation quotients of all color channels; (4) the overflow/underflow residual values in T_i; (5) the number m of bits to encode the index of a block; and (6) the number k of residual values.

Stage 2.2 — recovering the secret image.

4. Recover one by one in a raster-scan order the tile images T_i, $i = 1$ through n, of the desired secret image S by the following steps: (1) rotate the block indexed by j_i, namely B_{j_i}, in F through the optimal angle θ_o and fit the resulting content into T_i to form an initial tile image T_i; (2) use the extracted means and related standard deviation quotients to recover the original pixel values in T_i according to Eq. (4); (3) use the extracted means, standard deviation quotients, and Eqs. (5) to compute the two parameters c_S and c_L; and (4) scan T_i to find out pixels with values 255 or 0 which indicate that overflows/underflows have occurred there, and add respectively the values c_S or c_L to the corresponding residual values of the found pixels, resulting in a final tile image T_i.
5. Compose all the final tile images to form the desired secret image S as output.

The time complexity of Algorithm 1 is $O(n\log n)$ because the running time is dominated by Step 4: sorting the blocks in S_{tile} and S_{target}. And the time complexity of Algorithm 2 is $O(nN_T)$ because it just extracts the embedded information and recovers the secret image back with the extracted data.

5 Experimental Results

An experimental result is shown in Fig. 4, where 4(c) shows the created mosaic image using Fig. 4(a) of size 1024×768 as the secret image and Fig. 4(b) of the same

size as the target image. The tile image size is 8×8. The recovered secret image using a correct key is shown in Fig. 4(d) which is quite similar to the original secret image shown in Fig. 4(a). It has PSNR = 48.597 and RMSE = 0.948 with respect to the secret image. In fact, it is difficult for a human to feel the difference between two images when the PSNR is larger than 30 or when the RMSE is close to 1.0. It is noted by the way that all other experimental results shown in this paper have PSNR values larger than 47 and RMSE values close to 1.0, as seen in Figs. 6(c) and 6(d). Back to discussions on the results shown in Fig. 4, Fig. 4(e) shows the recovered secret image using a wrong key, which is a noise image. Figs. 4(f) through 4(h) show more results using different tile image sizes. It can be seen from the figures that the created mosaic image retains more details of the target image when the tile images are smaller. Fig. 6(a) also shows this fact in a similar way — mosaic images created with smaller tile image sizes have smaller RMSE values with respect to the target image. However, even when the tile image size is large (e.g., 32×32), the created mosaic image still looks quite similar to the target image. On the other hand, the number of required bits embedded for recovering the secret image is increased when the tile image becomes smaller, as can be seen from Fig. 6(b).

Fig. 4. An Experimental result of secret-fragment-visible mosaic creation. (a) Secret image. (b) Target image. (c) Mosaic image created with tile image size 8×8. (d) Recovered secret image using a correct key with PSNR = 48.597 and with RMSE =0.948 with respect to secret image (a). (e) Recovered secret image using a wrong key. (f)-(h) Mosaic images created with different tile-image sizes 16×16, 24×24, 32×32.

Fig. 5 shows a comparison of the results yielded by the proposed method and by the method proposed by Lai and Tsai [5], where Figs. 5(a) and 5(f) are the input secret images and Figs. 5(b) and 5(g) are the selected target images; Figs. 5(c) and 5(h) were created by Lai and Tsai [5]; and Figs 5(d) and 5(i) were created by the proposed method. Also, Figs. 5(e) and 5(j) show the recovered secret images. It can

be seen that the created mosaic images yielded by the proposed method have smaller RMSE values with respect to the target images, implying that they are more similar to the target images. And more importantly, the proposed method allows users to select their favorite images for uses as target images. This provides great flexibility in practical applications without the need to maintain a target image database which usually is very large if mosaic images with high similarities to target images are to be generated. By the way, it is noted that both the recovered secret images shown in Figs. 5(e) and 5(j) also have RMSE values close to 1.0 with respect to the respective secret images, saying they are very close to the original secret images in appearance.

Moreover, we conducted experiments on a large data set with 127 different secret image and target image pairs, and the result is included in Fig. 6 (as orange curves).

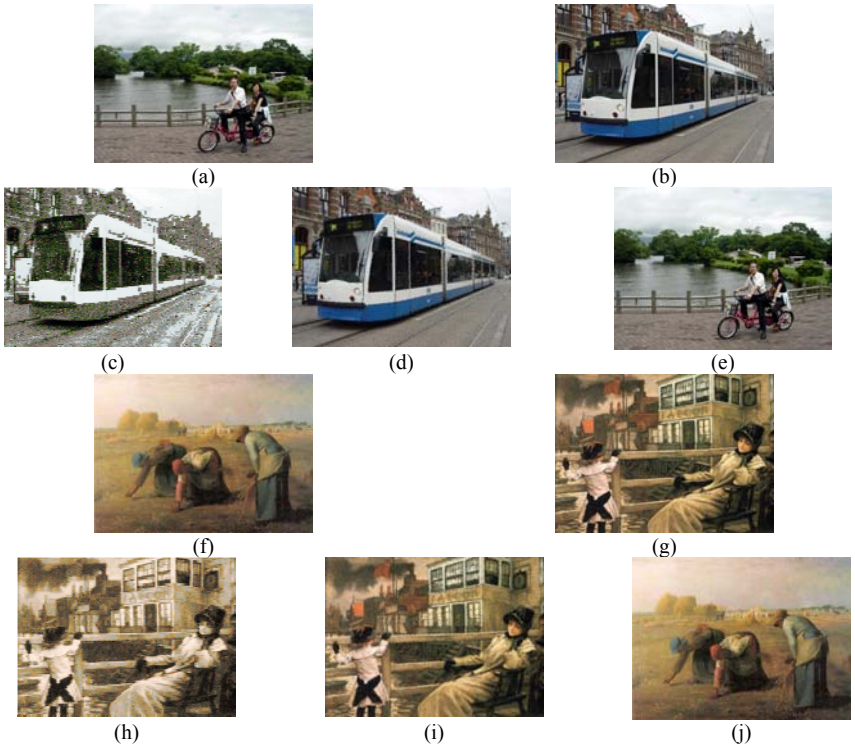

Fig. 5. Comparison of results of Lai and Tsai [5] and proposed method. (a) Secret image. (b) Target image. (c) Mosaic image created by method proposed by Lai and Tsai [5] with RMSE=47.651. (d) Mosaic image created by proposed method with RMSE = 33.935. (e) Recovered secret image with RMSE=0.993 with respect to secret image (a). (f) Secret image of another experiment. (g) Target image. (h) Mosaic image created by Lai and Tsai [5] with RMSE=38.036. (i) Mosaic image created by proposed method with RMSE=27.084. (j) Recovered secret image with RMSE=0.874 with respect to secret image (f).

6 Conclusions

A new image steganography method has been proposed, which not only can be used for secure keeping of secret images but also can be a new option to solve the difficulty of hiding images with huge data volumes behind cover images. By the use of proper pixel color transformation as well as skillful handling of

overflows/underflows in the converted pixels' colors, secret-fragment-visible mosaic images of high similarities to arbitrarily-selected target images can be created with no need of a target image database, and the original secret images can be recovered nearly losslessly from the created mosaic images. Good experimental results have shown the feasibility of the proposed method. Future studies may be directed to applying the proposed method to images of color models other than the RGB.

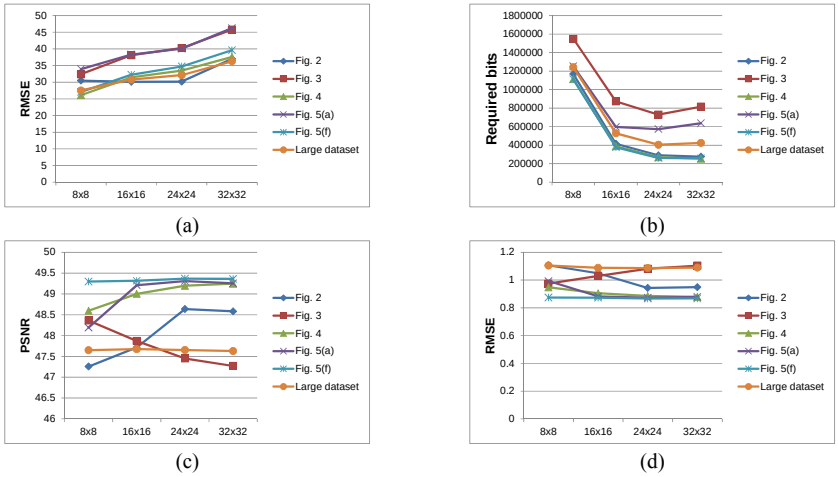

Fig. 6. Plots of trends of various parameters versus different tile image sizes (8×8, 16×16, 24×24, 32×32) with input secret images all shown previously and a large data set with 127 different secret image and target image pairs. (a) RMSE values of created mosaic images with respect to target images. (b) Numbers of required bits embedded for recovering secret images. (c) PSNR values of recovered secret images with respect to original ones. (d) RMSE values of recovered secret images with respect to original ones.

References

1. Bender, W., Gruhl, D., Morimoto, N., Lu, A.: Techniques for Data Hiding. IBM System Journal, Vol. 35, 313－336 (1996)
2. Petitcolas, F.A.P., Anderson, R.J., Kuhn, M. G.: Information Hiding － a Survey. Proceedings of IEEE, Vol. 87, No. 7, 1062－1078 (1999)
3. Thien, C. C., Lin, J. C.: A Simple and High-hiding Capacity Method for Hiding Digit-by-digit Data in Images Based on Modulus Function. Pattern Recognition, Vol. 36, 2875－2881 (2003)
4. Wang, R. Z., Chen, Y. S.: High-payload Image Steganography Using Two-way Block Matching. IEEE Signal Processing Letters, Vol. 13, No. 3, 161－164 (2006)
5. Lai, I.J., Tsai, W.H.: Secret-fragment-visible Mosaic Image － A New Computer Art and Its Application to Information Hiding. Accepted and to appear in IEEE Transactions on Information Forensics and Security (2011)
6. Reinhard, E., Ashikhmin, M., Gooch, B., Shirley, P.: Color Transfer between Images. IEEE Computer Graphics and Applications, Vol. 21, No. 5 (2001)
7. Coltuc, D., Chassery, J.-M.: Very Fast Watermarking by Reversible Contrast Mapping. IEEE Signal Processing Letters, Vol. 14, No. 4, 255－258 (2007)