Secret Multimedia Sharing with Steganographic Effects via PNG Images

Chih-Ting Yang（楊芷婷）
Dept. of Computer Science, National Chiao Tung University,
Hsinchu, Taiwan 30010
Email: oopsy.iit96g@nctu.edu.tw

Wen-Hsiang Tsai（蔡文祥）
Dept. of Computer Science, National Chiao Tung University,
Hsinchu, Taiwan 30010
Email: whtsai@cis.nctu.edu.tw

***Abstract*—A new method by secret sharing with steganographic effects for sharing secret images by PNG images is proposed. The sharing secret scheme is based on solving a set of simultaneous equations algebraically with *n* unknown numbers. The data of the secret image are distributed into *n* – 1 user-selected cover images and one residual image. The secret data hidden in the *n* – 1 cover images are embedded in the alpha channel values. A novel numerical data mapping is designed to make the alpha values randomly and uniformly distributed in a limited range, yielding almost no difference from the original cover images. And the data embedded in the residual image are randomized with a user key to enhance data security. Without correct image shares and the user key, the original secret image cannot be recovered. Experimental results are also shown to prove the feasibility of the proposed method.**

***Index Terms*—Secret sharing, PNG image, alpha channel, linear equation solving.**

I. INTRODUCTION

With the growth of the Internet, exchanges of files become more and more frequent. People sometimes may get easy access to files without authorization. Information sharing is a good solution to this problem. In an information sharing process, a given secret message is divided into several shares, which then are distributed to a number of participants for custody. In this way, nobody can recover the secret without collecting a sufficient number of shares, thus achieving the purpose of secret keeping.

Many secret sharing techniques have been proposed [1-5]. Shamir [5] proposed first the concept of secret sharing in his (*k*, *n*)-threshold method, in which a secret is divided into *n* shares and at least *k* shares must be collected to recover the secret. If *k* – 1 or fewer shares are collected, the secret cannot be recovered. On the contrary, if less than *n* – *k* shares are stolen, the secret can still be recovered. Blakley [1] transformed the secret sharing problem into a geometric one by considering the secret as a point of the intersection of *n* hyperplanes in an *n*-dimensional space where *n* > 2. Lin and Tsai [3] proposed a scheme for secret multimedia sharing with steganographic effects by creating *n* shares from *n* – 1 cover images using simple logic operations.

In this study, we propose a secret sharing method which generates *n numerical shares* from a given secret image by linear-equation solving and hide the numerical shares respectively into *n* PNG images in which the first *n* – 1 ones are user-selected and the *n*th one is artificially created, as illustrated in Fig. 1. The first *n* – 1 numerical shares are hidden respectively into the alpha channels of the *n* – 1 user-selected cover images; and the *n*th numerical share, which basically is composed of the *residual values* resulting from the linear-equation solving process, is hidden into an *n*th image, called a *residual image*. The resulting *n* stego-images, called *image shares*, are finally distributed to the information-sharing participants for custody. Only when all the *n* shares are collected can the hidden secret image data be recovered successfully through a reverse process of the above-mentioned linear-equation solving scheme.

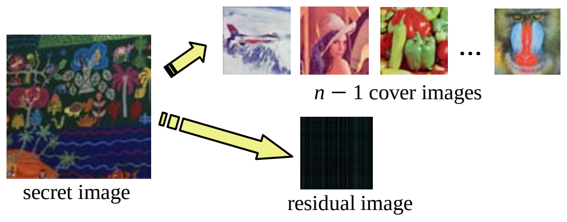


Fig. 1 Concept of proposed secret sharing process.

PNG is the abbreviation of Portable Network Graphics. The PNG format was created originally to improve the GIF (graphics interchange format) image. It is a bitmapped image format that employs the LZW (Lempel-Ziv-Welch) lossless data compression technique. As the name suggests, it was designed for transferring images on the Internet, not for professional graphics. In addition to the three channels of colors, namely, red (R), green (G), and blue (B), the PNG format includes an extra channel, named *alpha channel*. The alpha channel is a kind of controller which can be used to adjust the weight of the RGB channels that people can see. And except the RGB color space, the PNG format does not support other color spaces, such as CMYK.

The proposed method is novel in several aspects, including:

(1) A new secret sharing technique different from those of [1, 3, 5] mentioned above, namely, linear-equation solving, is proposed.
(2) For the first time, PNG images are used for hiding secrets (the numerical shares mentioned previously).
(3) The color channels of the used PNG images are not changed; only the alpha channels of them are utilized for secret data hiding.
(4) The alpha channels are manipulated randomly (the details described later) so that the resulting PNG image shares become almost no different from original versions of the chosen PNG cover images, yielding more effective steganographic effects in the image shares for safer keeping of them.
(5) The linear equation solving process involves only simple algebraic operations, resulting in faster computation speed in the secret sharing process.
(6) The embedded secret image can be recovered losslessly from the image shares.

In the remainder of this paper, we will describe the proposed secret sharing method in detail in Section 2, the secret recovery process in Section 3, some experimental results in Section 4, followed by conclusions and suggestions for future researches in the last section.

II. Proposed Secret Sharing Method

The basic idea of the proposed secret sharing method is described first here. The method essentially manipulates a given secret image S of the BMP format algebraically to generate numerical shares by solving three equations using $n - 1$ user-selected cover images in the PNG format, as mentioned previously. First, the BMP image S is transformed into another one S' with each pixel having smaller R, G, and B channel values instead of the original 8-bit values. This is conducted for the purpose of simplifying the linear-equation solving process and improving the steganographic effect in the resulting PNG image shares, as will be seen clearly later.

Then, the transformed secret image S' is *shared* into n images, where $n - 1$ cover images are selected arbitrarily, and the nth one is generated, by solving a set of linear equations proposed in this study as described in the following:

$$\begin{aligned} R_{\text{sec}} &= \alpha_1 \times R_1 + \alpha_2 \times R_2 + \ldots + \alpha_{n-1} \times R_{n-1} + R_{\text{res}}; \\ G_{\text{sec}} &= \alpha_1 \times G_1 + \alpha_2 \times G_2 + \ldots + \alpha_{n-1} \times G_{n-1} + G_{\text{res}}; \quad (1) \\ B_{\text{sec}} &= \alpha_1 \times B_1 + \alpha_2 \times B_2 + \ldots + \alpha_{n-1} \times B_{n-1} + B_{\text{res}}, \end{aligned}$$

where

(1) R_{sec}, G_{sec}, and B_{sec} represent the red, green, and blue channel values of a pixel in the transformed secret image S';
(2) R_k, G_k, and B_k are similar values of a pixel in the kth cover image where $k = 1, 2, ..., n - 1$;
(3) α_1, $\alpha_2 \ldots$, α_{n-1} are unknown coefficients to be solved; and
(4) R_{res}, G_{res}, and B_{res} are unknown residual values also to be computed.

For the purpose of making the recovered secret image lossless, we also propose a mapping technique which transforms the solutions to the unknown coefficients and residual values into appropriate values.

The details of the previously-mentioned processes are described as algorithms in the following.

Algorithm 1: ***transformation of secret image.***

Input: a secret image S of the BMP format.

Output: a transformed secret image S'.

Steps

1. Combine the R, G, and B color channel values of all pixels in S in a raster-scan order to form a decimal number sequence D_1.
2. Transform D_1 into a long bit-value string D_2.
3. Divide D_2 into a sequence D_3 of 6-bit segments $f_1, f_2, \ldots, f_k$.
4. Transform each f_i in D_3 into a decimal number d_i, respectively.
5. Compute a new value n_i from each d_i to compose a fourth sequence D_4 by the equation
$$n_i = d_i \times 3 + 1 \qquad (2)$$
where $i = 1, 2, \ldots, k$.
6. Take sequentially every three values of D_4 as the new R, G, and B color channel values, respectively, of an in-order raster-scanned pixel of a new BMP image S' until no more n_i's is left in
7. Take the final S' as the desired output.

The reason why we have to transform the original secret image into another form with essentially less bits for each color channel value is to ensure that the values of the solutions computed by the linear-equation solving process described next can be of reasonable magnitudes for use as color or alpha channel values, as will be seen clearly in the subsequent discussions.

Algorithm 2: ***process of secret sharing***.

Input: a secret BMP image S; $n - 1$ user-selected cover images, S_1 through S_{n-1}; a user-selected secret key K; and a random number generator f.

Output: $n - 1$ image shares, S_1' through S_{n-1}', and a residual image S_n', all in PNG format.

Steps.

1. Perform Algorithm 1 to transform the input secret image S into another one S'.
2. For each pixel P located at the same positions in S', and S_1 through S_{n-1}, perform the following steps.
   - 3.1. Set the R, G, and B color channel values of P in S' as the constant values R_{sec}, G_{sec}, and B_{sec}, respectively, of the left-hand sides of Eqs. (1).
   - 3.2. Perform the following steps to find the solutions to the $n-1$ unknown coefficient values α_1 through α_{n-1} and the residual values R_{res}, G_{res}, and B_{res} in Eqs. (1).
     - (a) Limit each of α_1, $\alpha_2 \ldots$, and α_{n-1} to be one of 12 pre-selected discrete values {0.0, 0.1, 0.2, 0.3, …, 0.9, and 1.0} and generate all possible combinations of the $n - 1$ variables α_1 through α_{n-1}.
     - (b) For each possible combination C_j of α_1 through α_{n-1} so generated, perform the following steps.
       - (1) Compute the residual values R_{res}, G_{res}, and B_{res} according to Eqs. (1) in the following way:
$$R_{\text{res}} = R_{\text{sec}} - (\alpha_1 \times R_1 + \alpha_2 \times R_2 + \ldots + \alpha_{n-1} \times R_{n-1});$$
$$G_{\text{res}} = G_{\text{sec}} - (\alpha_1 \times G_1 + \alpha_2 \times G_2 + \ldots + \alpha_{n-1} \times G_{n-1}); \quad (3)$$
$$B_{\text{res}} = B_{\text{sec}} - (\alpha_1 \times B_1 + \alpha_2 \times B_2 + \ldots + \alpha_{n-1} \times B_{n-1}).$$
       - (2) Compute the sum of the residual values $M_{\text{res}} = R_{\text{res}} + G_{\text{res}} + B_{\text{res}}$.
     - (c) Find the combination C_{opt} among all possible ones for which all the residual values R_{res}, G_{res}, and B_{res} computed above are non-negative and the sum M_{res} is the minimum.
     - (d) Take the $n - 1$ coefficient values and the residual values corresponding to C_{opt}, denoted as α_1' through α_{n-1}' and R_{res}', G_{res}', and B_{res}', respectively, as the desired solutions to the $n - 1$ unknown coefficient values α_1 through α_{n-1} and the three known residual values R_{res}, G_{res}, and B_{res} in Eqs. (1).
   - 3.3. Map α_1' through α_{n-1}' to new values α_1'' through α_{n-1}'', respectively, according to the following rule:
$$\square\alpha_i'' = (10 - (10 \times \alpha_i')) + 245 \qquad (4)$$
where $i = 1, 2, \ldots, n - 1$.
   - 3.4. Hide numerical shares α_1'' through α_{n-1}''

into the alpha channels of the corresponding n–1 cover images S_1 through S_{n-1}, respectively, to obtain n–1 image S_1' through S_{n-1}', respectively.

3.5. Assign respectively the residual values, R_{res}', G_{res}', and B_{res}', to the red, green, and blue channel values of the pixel in the residual image S_n' corresponding to pixel P.

3. Use K as a seed for the input random number generator f to randomize the positions of the pixel values in the red, green, and blue channels of the residual image S_n'.
4. Take the n resulting images S_1' through S_n' as the desired n output image shares.

Notice that in Step 3.3 above, we map the original solutions to α_1' through α_{n-1}' to α_1'' through α_{n-1}'', respectively, by the equality $\alpha_i'' = (10 - (10 \times \alpha_i')) + 245$ before embedding them into the alpha channels. Because the solutions α_1' through α_{n-1}' are limited to be within the range of $H = \{0.0, 0.1, 0.2, 0.3, \ldots, 0.9, 1.0\}$, the values of α_1' through α_{n-1}' will so be limited to be within the range of $H' = \{255, 254, \ldots, 245\}$, with 0.0 mapped to 255, 0.1 mapped to 254, …, and 1.0 mapped to 245, respectively. One reason behind this mapping is that we want to avoid embedding *small* values into the alpha channel because small alpha channel values will cause the resulting PNG image to become white-noised which is not desired from the viewpoint of steganography. Another reason is that the alpha values are limited to those of H' so that they are all very close in magnitudes, yielding an effect of nearly uniform transparency on a resulting PNG image share. This again increases the steganographic effect appearing on the first n – 1 image shares.

Also, Step 3 in the above algorithm is designed to increase the security of the residual data hidden in the residual image so that the data can be protected against an illicit attacker with no valid key.

III. PROPOSED SECRET RECOVER PROCESS

In the proposed secret recovery process, we collect the n image shares to recover the secret image by using Eqs. (1) again in a reverse sense. We first extract the hidden solutions in the n – 1 image shares, and then bring them back into Eqs. (1) to compute the values of R_{sec}, G_{sec}, and B_{sec} for each pixel of the transformed secret image. Then, we can recover the original secret image by performing a reverse process of Algorithm 1. The details are described as algorithms in the following.

Algorithm 3: ***process of transformed secret image recovery***.

Input: n – 1 image shares S_1' through S_{n-1}' in the PNG format, a residual image S_n', a secret key K, and a random number generator f.

Output: a transformed secret image S'

Steps

1. Use K as a seed for the random number generator f to recover the randomized positions of the pixel values in the red, green, and blue channels of S_n'.
2. For each pixel P' located at the same positions in S_1' through S_{n-1}' and S_n', perform the following steps.

2.1. Extract the alpha values α_i'' from S_i' where i = 1, 2…, n – 1 and apply the following rule to get the hidden solutions to the unknown coefficient values, α_1 through α_{n-1}:

$$\alpha_i = (10 - (\alpha_i'' - 245)) / 10. \qquad (5)$$

2.2. Extract residual values, R_{res}, G_{res}, and B_{res}, in the three color channels (red, green, and blue) from S_n'.

2.3. Use the follow formulas to compute the *transformed* secret data, R_{sec}, G_{sec}, and B_{sec} for the pixel P' in S':

$$R_{\text{sec}}=\alpha_1\times R_1+\alpha_2\times R_2+\ldots+\alpha_{n-1}\times R_{n-1}+R_{\text{res}};$$
$$G_{\text{sec}}=\alpha_1\times G_1+\alpha_2\times G_2+\ldots+\alpha_{n-1}\times G_{n-1}+G_{\text{res}}; \quad (6)$$
$$B_{\text{sec}}=\alpha_1\times B_1+\alpha_2\times B_2+\ldots+\alpha_{n-1}\times B_{n-1}+B_{\text{res}},$$

where R_k, G_k, and B_k (k = 1, 2, …, n–1) represent the red, green, and blue color channel values of the identically-positioned pixel P' in S_k', respectively.

2.4. Take the final image S' as the output.

Now, we still have to transform the output image of the above algorithm into its original form as the final secret image. This needs a reverse version of Algorithm 1, as described in the following.

```
Algorithm 4: computation of original secret image.
Input: a transformed secret image $S'$ of the BMP format.
Output: the original secret image $S$.
Steps
  1. Raster-scan all pixels in $S'$ and take sequentially the color channel values of the
     pixels to form a decimal number sequence $D_4$.
  2. For each value $n'$ in $D_4$, compute for it a new value $d$ in the following way to
     form a new number sequence $D_3$:
          $d = (n' - 1) / 3$.          (7)
  3. Transform the decimal number sequence $D_3$ into a bit sequence $D_2$.
  4. Divide $D_2$ into a group of 8-bit segments, $f_1'$, $f_2'$, …, $f_m'$.
  5. Transform $D_2$ into a decimal number sequence $D_1$ again by transforming every $f_i'$
     into a corresponding decimal number $f_i$.
  6. Take sequentially every three numbers in $D_1$ as the color channel values $R$, $G$, and $B$,
     respectively, of an in-order raster-scanned pixel $P$ of the desired secret image $S$.
  7. Take the final $S$ as the output.
```

IV. Experimental Results

Some experimental results of applying the proposed method are shown here. First, a secret image is shown in Figures 2(a), and five user-selected cover images are shown in Figures 2(b) through 2(f). Figure 3 shows the resulting image shares after the proposed secret image sharing method (Algorithms 1 and 2) was applied. Figure 3(a) shows the yielded residual image. Figures 3(b) through 3(f) show the resulting image shares which are PNG images with secret data hidden in their pixels' alpha channels. Note that these five images look to have almost no difference from those of Figure 2(b) through 2(f) though the alpha channel values have been changed. Here, the size of the secret image is 256×256, and those of the shares are 512×512, all in the PNG format.

An experimental result of applying the proposed secret image recovery method (Algorithms 3 and 4) with correct shares is shown in Figures 4. Figures 4(a) through (f) show the five input correct image shares with 4(a) being the correct residual image, and the correctly recovered secret image is shown in Figures 4(g).

Some experimental results of applying the proposed secret image recovery method with some *incorrectly* provided image shares are shown in Figure 5(a) shows the correct input residual image, Figures 5(c) and 5(f) show two correct image shares, and Figures 5(b), 5(d), 5(e) show three incorrectly provided image shares. Figures 5(g) shows the result of secret recovery which is noisy and so incorrect.

V. Conclusion

In this paper, we have proposed a new secret sharing method, which shares secret images by solving a set of simultaneous linear equations. Each equation has n unknown variables, and we hide the solutions to the n variables in different cover images. The first $n - 1$ ones are equation coefficients which are hidden in the alpha channel values of $n - 1$ user-selected cover images, resulting in $n - 1$ PNG images which look almost not different from the original cover images, yielding steganographic effects and so increasing the security of keeping the image shares. And the nth variable values are taken as residual values and hidden in a so-called residual image. The data embedded in this image is randomized with a user-selected key, thus enhancing the security of the hidden data against illicit tampering. Without correct image shares and the user key, the original secret image cannot be recovered, as proved by the experimental results. It is mentioned that only when all correct shares are collected and used in the recovery process can the original secret image be recovered. Future researches may be directed to applying the secret sharing technique to other information hiding applications.

Acknowledgements

This work was supported by the NSC project No. 97-2631-H-009-001.

References

[1] G. R. Blakley, “Safeguarding cryptographic keys,” *Proceedings of the National Computer Conference*, New York, U. S. A., pp. 313–317, 1979.

[2] R. Cramer, V. Daza, I. Gracia, J. Jiménez Urroz, G. Leander, J. Martí-Farré, and C. Padró, “On codes, matroids, and secure multiparty computation from linear secret-sharing schemes”, *IEEE Transactions on Information Theory*, vol. 54, no. 6, pp. 2644-2657, June 2008.

[3] C. C. Lin and W. H. Tsai “Secret image sharing with steganography and authentication,” *Journal of Systems & Software*, vol. 73, no. 3, pp. 405-414, Nov.-Dec. 2004.

[4] C. C. Lin and W. H. Tsai, “Secret multimedia information sharing with data hiding capacity by simple logic operations,” *Proceedings of 5th World Multiconference on Systemics, Cybernetics, and Informatics, Vol. I: Information Systems Development*, Orlando, Florida, U. S. A., pp. 50-55, July 2004.

[5] A. Shamir, “How to share a secret,” *Communications of Association for Computing Machinery* , vol. 22, no. 11, pp. 612- 613, Nov. 1979.

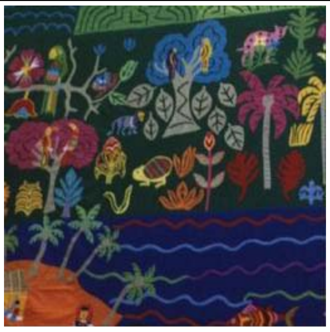
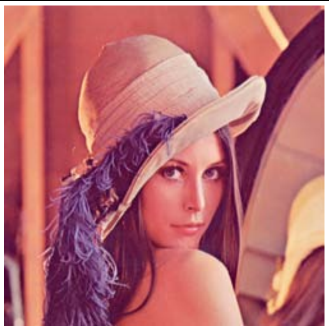
		
(a)	(b)	(c)
(d)	(e)	(f)

Fig. 2 Secret image and cover image selected by users before applying proposed secret sharing method. (a) Input secret image with size 256 × 256. (b)-(f) Five cover images selected by users.

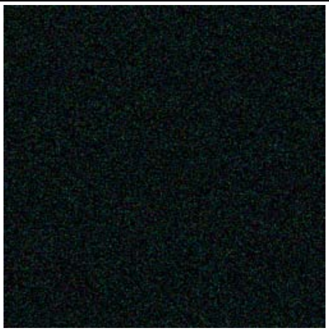
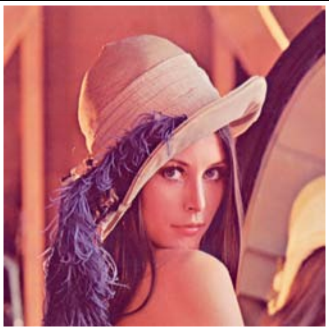
		
(a)	(b)	(c)
(d)	(e)	(f)

Fig. 3 Residual image and yielded image shares after applying the proposed secret sharing method. (a) Yielded residual image. (b)-(f) Yielded image shares which almost have no difference from those of (b)-(f) in Fig. 2.

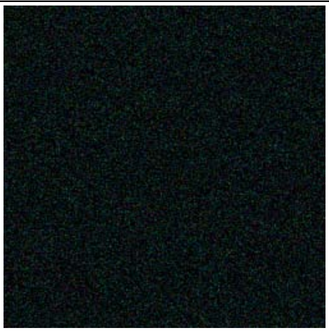
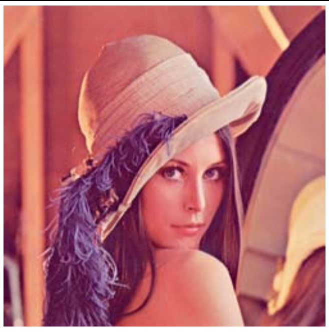
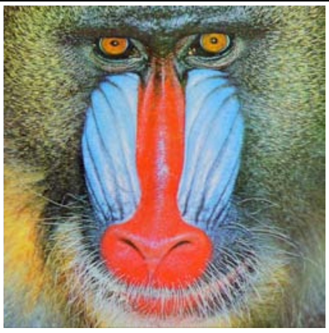
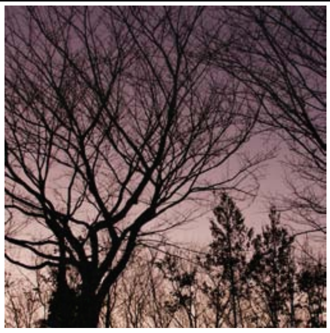
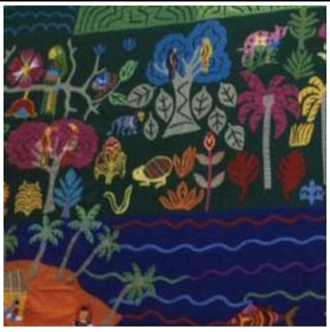
		
(a)	(b)	(c)
		
(d)	(e)	(f)
		
	(g)	

Fig. 4. Results of applying proposed secret image recovery method with correct shares. (a) Correct input residual image. (b)-(f) Correct input image shares. (g) Correctly recovered secret image.

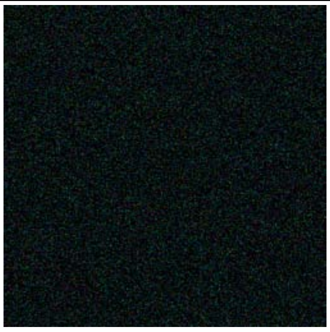
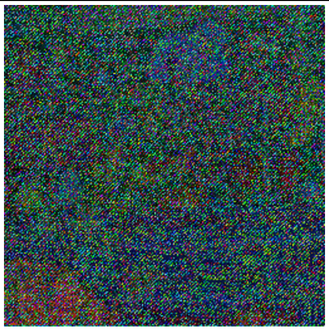
		
(a)	(b)	(c)
(d)	(e)	(f)
		
	(g)	

Fig. 5. Results of applying proposed secret image recovery method with some incorrect image shares. (a) Correct input residual image. (c) and (f) Correct image shares. (b), (d), and (e) Three incorrectly provided image shares. (g) Incorrectly recovered secret image.

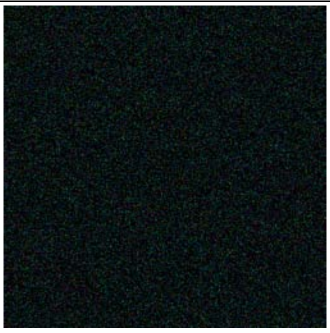
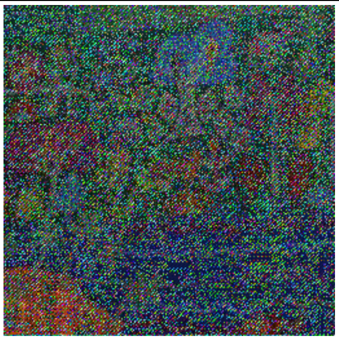
		
(a)	(b)	(c)
(d)	(e)	(f)
		
	(g)	

Fig. 6. Results of applying proposed secret image recovery method with some incorrect image shares. (a) Correct input residual image. (c), (e) and (f) Correct image shares. (b) and (d) Three incorrectly provided image shares. (g) Incorrectly recovered secret image.

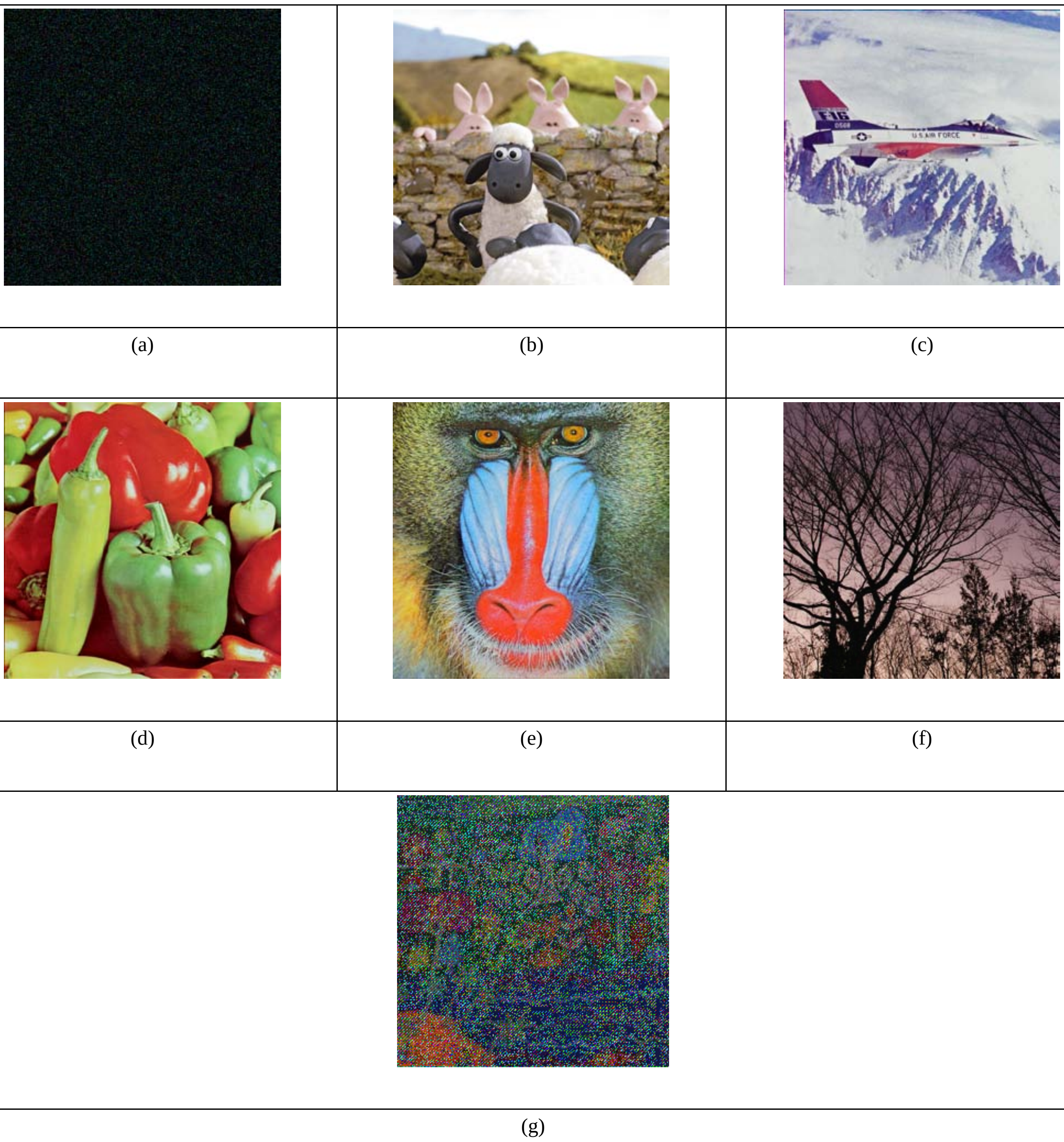


Fig. 7. Results of applying proposed secret image recovery method with some incorrect image shares. (a) Correct input residual image. (c), (d), (e) and (f) Correct image shares. (b) Three incorrectly provided image shares. (g) Incorrectly recovered secret image.