Line covering method and its applications in steganography and steganalysis

Jiajun Mao*, Zhenyong Chen, Wei Fan and Zhang Xiong

School of Computer Science and Engineering,
Beihang University,
Beijing, 100191, China
E-mail: maojiajun@cse.buaa.edu.cn
E-mail: chzhyong@buaa.edu.cn
E-mail: fanwei@cse.buaa.edu.cn
E-mail: xiongz@buaa.edu.cn
*Corresponding author

Abstract: Image features are widely used in steganography and steganalysis. In steganography, image features are used to better understand the image, in order to achieve higher quality. In steganalysis, image features are used to show the changes caused by steganography. This paper presents a new method, line covering, to evaluate the smoothness of an image. A new image feature, smoothness based on line covering, is proposed and we discuss its applications in steganography and steganalysis. This feature can better show the smoothness features in different directions whereas many other image features ignore the direction of features. A steganalysis based on this feature is proposed which has high accuracy in detecting a histogram modification steganographic techniques. And it can easily control the false rejection rate and false acceptation rate. This paper also discusses its application in steganography, and how it can solve the problem of choosing a pairing direction in the steganography based on difference expansion (DE).

Keywords: line covering; smoothness; difference expansion; DE; steganalysis; steganography.

Reference to this paper should be made as follows: Mao, J., Chen, Z., Fan, W. and Xiong, Z. (2013) 'Line covering method and its applications in steganography and steganalysis', *Int. J. Multimedia Intelligence and Security*, Vol. 3, No. 1, pp.1–22.

Biographical notes: Jiajun Mao is a Masters degree candidate in the School of Computer Science and Engineering from Beihang University, China. His current research interests include digital steganography and steganalysis.

Zhenyong Chen is Associate Professor and Master's degree Supervisor in the School of Computer Science and Engineering of Beihang University, China. His current research interests include information hiding, digital watermarking and data vitalisation.

Wei Fan is a PhD candidate in the School of Computer Science and Engineering from Beihang University, China. Her current research interests include reversible stenography and stenography analysis.

Zhang Xiong is Professor and PhD Supervisor in the School of Computer Science and Engineering from Beihang University, China. His research interests include multimedia, computer control and information systems, smart cities, and data vitalisation.

1 Introduction

Wikileaks makes the world focus on the information warfare and security. In information warfare and security, information hiding is one of the most important techniques. In information warfare and security, one side uses steganography to transfer secret messages while the opposition uses steganalysis to detect the existence of such messages or discover the content of the messages. Many image steganographic and steganalytic methods have been proposed in the past decade.

The steganalytic methods can be categorised into two groups: special purpose and blind. Special steganalysis aims to identify the presence of secret messages embedded by a specific algorithm, while blind steganalysis is intended to detect a wide range of steganography including previously unknown types (Chandramouli et al., 2004). Steganalysis in Westfeld and Pfitzmann (2000), Provos and Honeyman (2001), Fridrich et al. (2001) and Fridrich and Goljan (2002) aims to analyse the least significant bit (LSB) steganography which is a simple but important method. The χ^2 analysis in Westfeld and Pfitzmann (2000) and Provos and Honeyman (2001) uses the χ^2 distribution to analyse the changes caused by steganography. The RS analysis in Fridrich et al. (2001) and Fridrich and Goljan (2002) uses the imbalance of ±1 in steganography to analyse the embedding rate. This does not work well when steganography uses flip to maintain the ±1 balance. Steganalysis for special purposes can only be used to analyse a specific type of steganography, it will not work when used to analyse different types of steganography. The steganalysis algorithms in steganalysis algorithms in Goljan et al. (2006) and Sabeti et al. (2010) are blind analysis. Goljan et al. (2006) aim to analyse all steganography, while Sabeti et al. (2010) aim to analyse all difference expansion (DE)-based blind steganalysis techniques use the changes caused by steganography to analyse the hidden messages. Usually, image features are used to show these changes. So image feature mining is an important issue in steganalysis. Many features in steganalysis. Many features (Liu et al., 2006; Avcibas et al., 2002; Lyu and Farid, 2004; Cancelli et al., 2008) are used in information hiding, such as image complexity (Liu et al., 2006), statistics of the histogram (Avcibas et al., 2002), wavelet statics (Lyu and Farid, 2004), features based on amplitudes of local extremes (Cancelli et al., 2008) and so on. Most of the image features have high computational complexity and ignore the direction of the features. The direction of features is important, because the features may be different in different directions of an image.

Until now, many types of steganography have been proposed. LSB steganography is the most typical method and many types of steganography are based on it and all the steganography have the security problem. It is that there are many steganalytic methods which can easily discover messages hidden by these techniques. Many new techniques are used in steganography such as DE, data compression and histogram modification (HM). DE is an important technique widely used in information. Many steganographic methods (Tian, 2003a, 2003b; Lin et al., 2008; Tian, 2002; Tian and Wells, 2004; Alattar,

2003, 2004a, 2004b) are based on DE, these methods have high capacity, and they are reversible, they can recover the cover image, which is very important in covert communication and other applications. DE is a reversible steganography technique first proposed by Tian (2003a). Tian defines a reversible integer transformation. During the transformation, the message is embedded into the pairs of pixels by expanding the difference of the pixels. Alattar (2003) generalises the DE for three pixels, four pixels (Alattar, 2004a), and N pixels (Alattar, 2004b). After these methods, reversible watermarking is improved by other technique, Luo et al. (2010) use interpolation technique and Chen et al. (2010) use full context prediction technique in reversible watermarking. All these steganographic methods first choose a group of pixels; some choose pixel pairs in vertical or horizontal direction and some choose the pairs randomly by a key. Different methods of choosing pairs may achieve different results. None of these discuss how to choose the pixels in order to achieve higher performance. This paper will discuss a method to choose better pairing direction to get higher quantity of stegoed images.

In this paper, a new method, line covering, evaluating image smoothness is proposed. Line covering can show the image smoothness in four directions, which is not available for most other methods. It is helpful for understanding the image better. The image feature has low computational complexity and can solve the lack of direction faced by other features. We introduce our image feature's applications in steganography and steganalysis. In steganalysis, smoothness can show the changes caused by steganography, which can be used in steganalysis. Based on the image smoothness, we propose the steganalysis of two steganographic techniques, which have high accuracy and low computational complexity. What is more, the steganalysis based on the image smoothness has unique advantages; the false rejection rate (FRR) and false acceptation rate (FAR) can be controlled by adjusting a parameter: this cannot be achieved by other steganalytic methods. In steganography, this feature can show the smoothness in four directions of an image. Then the images are better understood to choose better ways to embed message. Based on this, we introduce a steganographic technique which can improve upon DE-based steganography. Our technique solves the problem of choosing pairing direction and achieves higher image quality. Line covering can also be used in image processing and other applications. The parameters in line covering are changeable, so we can adjust them in different applications to satisfy different requirements.

Section 2 introduces line covering and smoothness feature, we discuss how to achieve line covering and explains its characteristics. The application of line covering in steganalysis and experimental results are discussed in Section 3. Section 4 discusses the application of line covering in steganography and experimental results. Finally, conclusions are drawn in Section 5.

2 Line covering and smoothness

2.1 Line covering

Most image features(Liu et al., 2006; Avcibas et al., 2002; Lyu and Farid, 2004; Cancelli, et al., 2008) cannot show these features in different directions of an image. But the features in different directions are important for many applications in which different directions may have different results. We propose a new method, line covering, to

evaluate the smoothness of an image, it can show smoothness in four directions of an image, horizontal, vertical, 45 degree, and 135 degree smoothness. In the rest of this paper, we call them 0°, 90°, 45°, and 135° smoothness (S_0, S_{45}, S_{90} and S_{135}).

In line covering, on the pixels of an image, in a direction (0°, 90°, 45°, or 135°), lines on this direction are used to cover the pixels having same value. After that, every pixel will be covered by a line. We call this line covering.

We provide an example here to explain in more detail. Before line covering, we choose a direction, say the horizontal direction (0°). We use lines to cover the pixels in horizontal direction. The line covering in horizontal direction is applied to each row. Lines are used to cover the adjacent pixels with same value. After line covering, every pixel is covered by a line. The length of a line is the number of the pixels covered by the line. The lengths of the lines are different, some lines are long and some are short. All lines are recorded in line list L, $L(i)$ means the length of the i^{th} line. We then count the numbers of lines which have the same length, and get the array h, $h(i)$ means the number of lines with length i. h is calculated as

$$h_0(i) = \sum_{j=1}^{lenl_0} sign\left(L_0(j) - i\right) \tag{1}$$

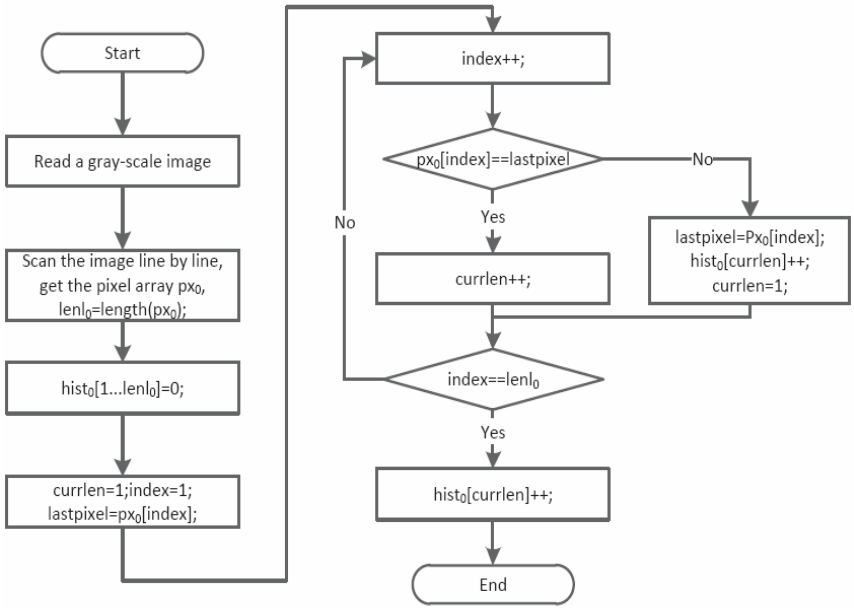
where

$$sign(x) = \begin{cases} 1x = 0; \\ 0x \neq 0. \end{cases} \tag{2}$$

$lenl_0$ is the length of L_0, L_0 is the line list in 0°.

The flow of line covering in 0° direction is show in Figure 1. After line covering we can get array h of 0° direction.

Figure 1 Flow chart of line covering in 0° direction



In the other three directions, the line covering can be done as the same way. So we can get three other array h (h_{45}, h_{90}, h_{135}).

Subsequently the number of lines with different lengths can be seen clearly in array h. Due to the difference of images, the number of different length lines is different.

Smooth images do not change greatly, so their adjacent pixels are more likely to be the same, and so they exhibit more long lines. The number of long lines will be greater in smoother images than that of rough images, so we can use the number of long lines in the line covering to evaluate image smoothness.

2.2 Smoothness

We know that smoother images have more long lines in line covering. So, the proportion of long vs. all lines can be used to show the smoothness. Before we calculate the proportion of long lines, we should determine which lines are long lines. We define a parameter K, equal to or longer than K are long lines. Different K may be chosen for different applications. When calculating the proportion, we add weights of lines to make the results more accurate, each $h(i)$ is multiplied by i. In each of the four directions, the proportion is calculated as

$$S_{direction} = \frac{\sum_{i=K}^{t} h(i) \times i}{\sum_{i=1}^{t} h(i) \times i}, direction = 0, 45, 90, 135 \tag{3}$$

where $S_{direction}$ is the smoothness of image in direction 0°, 45°, 90°, or 135°, and t is the maximum length of the lines. The value of $S_{direction}$ is between 0 and 1, the larger $S_{direction}$ is, the smoother the image is. If an image is a pure single colour image, the $S_{direction}$ is equal to 1.

After line covering, the array h is obtained and the $S_{direction}$ is calculated. Since there are four line coverings, one for each of four directions, every direction has an $S_{direction}$. Finally, we get S_0, S_{45}, S_{90}, and S_{135}. The average of the four *S direction* can be used to show the overall smoothness of the whole image, S:

$$S = \frac{S_0 + S_{45} + S_{90} + S_{135}}{4} \tag{4}$$

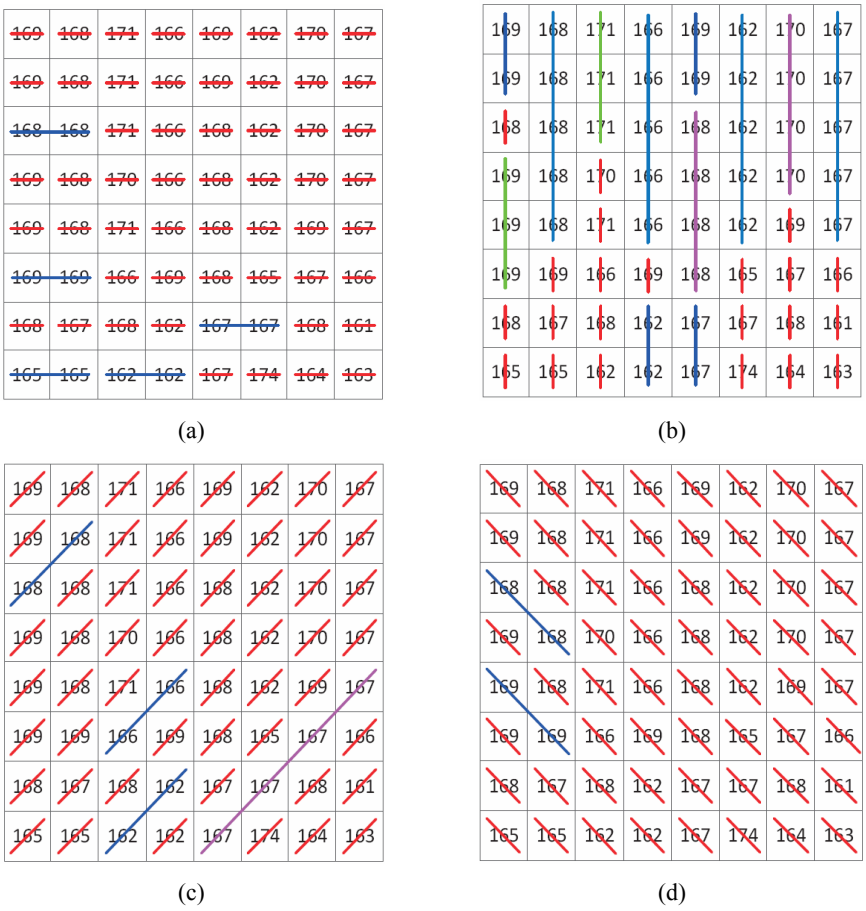
Our proposed line covering-based smoothness provides a good evaluation of image smoothness. Here, Figure 2 shows three standard images and their respective S values. It can be seen that the smoother image, *milkdrop*, has higher S value and the rougher image, *baboon*, has a lower S value. This shows a visual representation of the consistency of our smoothness measure.

Most image features ignore the directions of features, but since features may have different values on different directions this may cause problems. Because there are four $S_{direction}$ in our method can overcome this issue. Examine image *Lena* for example, the line covering in four directions of the first eight by eight block is shown in Figure 3. Figures 3(a) to 3(d) are the line coverings in 0°, 45°, 90°, and 135° directions. It can be seen that the number of long lines in the four figures is different, and there are more long lines in Figure 3(b).

Figure 2 Three standard images, (a) $S = 0{:}9419$ (b) $S = 0{:}3633$ (c) $S = 0{:}0214$

(a) (b) (c)

Figure 3 Line covering on four directions, (a) 0° (b) 90° (c) 45° (d) 135° (see online version for colours)



(a) (b)

(c) (d)

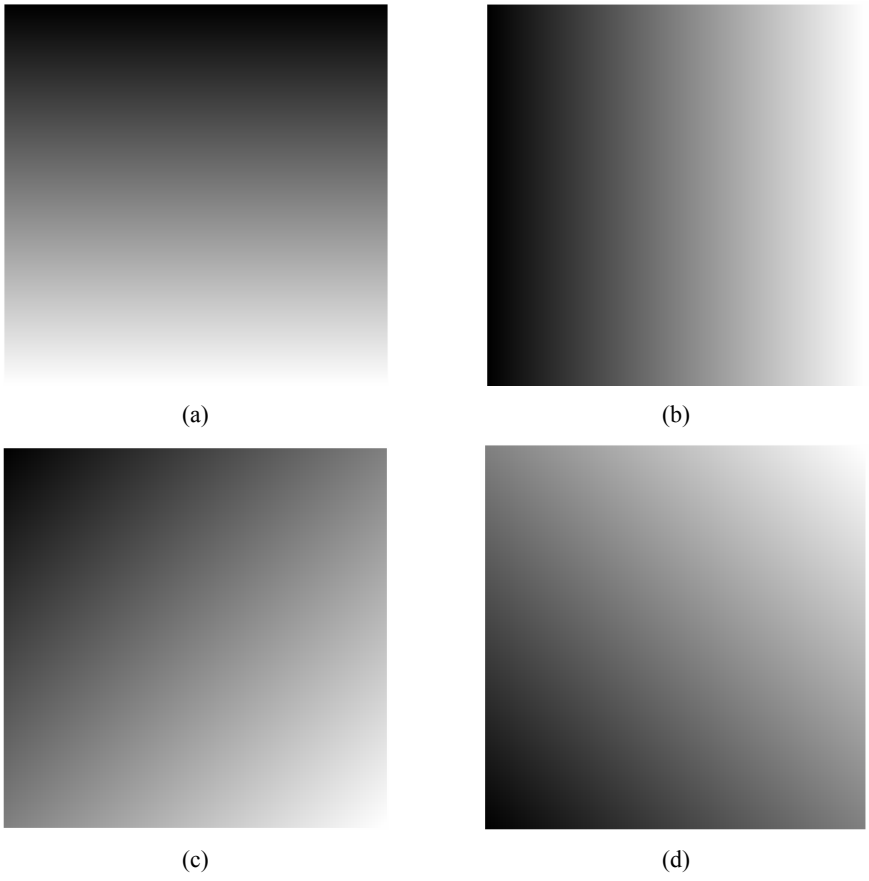
The array h of the four line covering is shown in Table 1. Here, we can clearly see the difference in the numbers of long lines. It is seen in Table 1 that in the 90° direction, there are many more lines of length greater than two than any other direction. Because

the image in Figure 3 is small (eight by eight), we set $K = 2$. This means that lines of length 2 or greater are long lines. So we get $S_0 = 0.0781$, $S_{90} = 0.6562$, $S_{45} = 0.1406$, $S_{135} = 0.0625$. With an average smoothness of $S = 0.2344$. Since S_{90} is the biggest it means that the image is smoothest in the 90° direction. It can be seen for image *Lena*, that the image smoothness feature in different directions is different.

Table 1 Array h of four line covering

h	*Length*							
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
h_0	54	5	0	0	0	0	0	0
h_{90}	22	4	2	2	4	0	0	0
h_{45}	55	3	1	0	0	0	0	0
h_{135}	60	2	0	0	0	0	0	0

Figure 4 Four images with same whole smoothness, (a) 0° (b) 90° (c) 45° (d) 135°



(a) (b)

(c) (d)

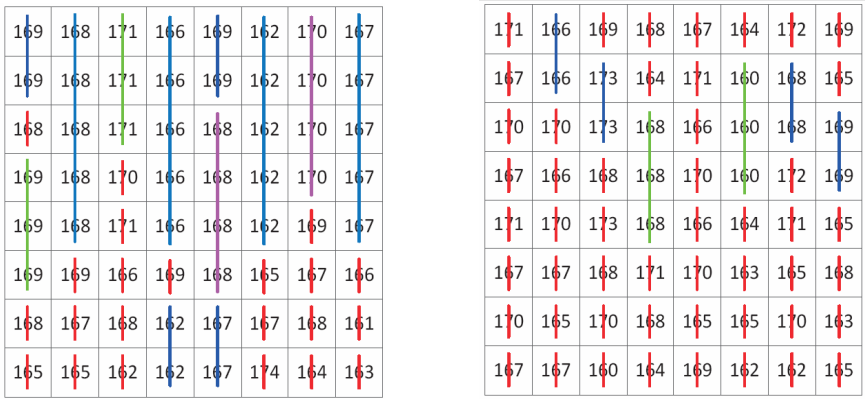
One feature is not enough to understand an image. Figure 4 shows four different images. Rotating image 4(a) 90° anticlockwise we get image 4(b), rotate image 4(c) 90° anticlockwise and get image 4(d). When a feature does not consider the direction, the

feature of image 4(a) and image 4(b) should be the equal, and it is the same with image 4(c) and image 4(d). The S values of the four images are the same. Our method can show the difference. The smoothness of each direction is shown in Table 2. It can be seen that although these four images have the same overall smoothness, approximately 0.25, these four images are different; the smoothness is different in different directions and as such the overall smoothness cannot show the difference. In our method, for an image, there are four $S_{direction}$ showing the smoothness on four directions and an S showing the whole smoothness of the image. So we can get the smoothness on different directions to show the difference. From Table 2, comparing S_0, S_{45}, S_{90}, and S_{135}, we can better understand the difference of the four images, image 4(a) is smoother in 0° direction, image 4(b) in 90° direction, image 4(c) in 45° direction and image 4(d) in 135° direction. We cannot see the difference in S, because each S is almost identical at approximately 0.25. So from the experimental results above, we show that our method can show the difference in image features in different directions.

Table 2 Smoothness of the four images

Images	*Smoothness(K = 4)*				
	S_0	S_{90}	S_{45}	S_{90}	S_{135}
Figure 4(a)	1	0	0	0	0.25
Figure 4(b)	0	1	0	0	0.25
Figure 4(c)	0	0	0.9993	0	0.2499
Figure 4(d)	0	0	0	0.9993	0.2499

Figure 5 Line covering of 90°direction (part of Lena), (a) before (b) after (see online version for colours)



(a) (b)

Line covering-based smoothness can also show changes caused by steganography which can be used in steganalysis. The line covering in the 90° direction of *Lena* works as Figure 5. Figure 5(a) shows the line covering of *Lena* before stenographic message embedding and Figure 5(b) after, the steganographic method is HM (De Vleeschouwer et al., 2003). After steganographic embedding, pixel values are changed, so long lines are truncated to short lines. The array h is shown in Table 3. It is seen that after steganographic embedding, there are fewer long lines in the image than in the original.

This means that the S value of the stego image is smaller. We obtain $S_{bef\ ore} = 0.531$ and $S_{after} = 0.094$ ($K = 3$). The smoothness changes greatly when steganographic embedding is applied and our smoothness measure, S, can show these changes clearly. So this feature can be used in steganalysis.

Table 3 Lines of *Lena* before and after steganography

h	*Length*							
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Before	22	4	2	2	4	0	0	0
After	50	4	2	0	0	0	0	0

When calculating each $S_{direction}$, all pixels of the image must be scanned one by one. So the total computational complexity of our method is $O(n)$, where n is the number of pixels which compose an image. We compare our method and three methods (Liu et al., 2006; Moulin and Liu, 1999; Zhang et al., 2006), the time used for computing the image feature in these four methods is shown in Table 4. It is proved that our method uses less time and has low computational complexity.

Table 4 Time used in four methods

Images	*Time(s)*			
	Our method	*Liu*	*Moulin*	*Zhang*
Baboon	0.043073	0.291906	1.093344	1.287087
Barbara	0.070088	0.489688	1.675118	2.101448
Boats	0.069715	0.541435	1.570972	2.073947
Bridge	0.043358	0.292976	1.059177	1.249874
Camera	0.011912	0.056412	0.40832	0.310697
Columbia	0.037498	0.262792	0.943106	1.099601
Couple	0.042839	0.293968	1.069358	1.25026
Crowd	0.043731	0.285648	1.030979	1.253312
Goldhill	0.06609	0.475614	1.522606	2.055185
Lake	0.043304	0.28751	1.05609	1.252678
Lax	0.042821	0.330044	1.071889	1.252202
Lena	0.046084	0.298128	1.082921	1.24748
Man	0.055462	0.445325	1.068999	1.246496
Milkdrop	0.042397	0.286745	1.050567	1.250494
Peppers	0.046105	0.295916	1.058101	1.29292
Plane	0.049767	0.291024	1.101139	1.245939
Woman1	0.04597	0.310185	1.082799	1.246815
Woman2	0.049653	0.287617	1.054827	1.254598

The rest of the paper will introduce how to use smoothness based on line covering in steganography and steganalysis.

3 Line covering in steganalysis

3.1 Steganalysis algorithm

Steganography is a technology which embeds payload information into a digital signal. This conceals the existence of the payload by modifying the digital signal. There are mainly two types of steganography, in the spatial and transform domains. Many steganographic techniques such as LSB, DE and HM embed a message in the spatial domain. So we now introduce steganalysis in the spatial domain. Steganography in the spatial domain directly changes the pixel values of a cover image, which brings changes in the correlation of adjacent pixels reducing the smoothness of the image. Many steganographic techniques are based on LSB steganography and HM steganography. They all directly modify the pixels when embedding a message. So we introduce a steganalytic method to analyse these two steganographic techniques.

In steganalysis, we define two types of image, original images (Type I) and stego images (Type II, which has had a message embedded using a steganographic technique). The goal of steganalysis is to separate these two types. It is proved in Section 2 that the smoothness of original images changes greatly after steganographic embedding. Then we check the changes in smoothness for stego images. Table 6 shows the S values for two types of image before, S_1, and after, S_2, steganography, the steganographic method is HM (De Vleeschouwer et al., 2003). It can be seen from Table 5 that the difference between S values is, on the whole, much greater for original images whereas the difference between S values for stego images are smaller. In Type I, the original image is smoother, with more long lines in line covering, the S value is larger. After steganographic embedding, the images become rougher, there are fewer long lines in line covering, and the S value is smaller. So the S values change greatly for Type I. In Type II, the S values of stego images are small before steganography. After steganography, some pixel values are changed to the original value, then some parts of the image become smoother, and some parts became rougher. So the smoothness of the whole image does not change much. This can be seen in Table 5 that in Type II, $S_1 - S_2$ is much smaller than that of Type I.

Table 5 Changes of S values ($K = 4$)

Images	*Type I*			*Type II*		
	S_1	S_2	$S_1 - S_2$	S_1	S_2	$S_1 - S_2$
Baboon	0.0517	0.0489	0.0028	0.0489	0.0449	0.004
Barbara	1.6784	0.6368	1.0416	0.6368	0.6343	0.0025
Camera	3.4034	1.9321	1.4713	1.9321	1.9565	–0.0244
Couple	0.4114	0.2808	0.1306	0.2808	0.2484	0.0324
Crowd	10.2719	3.0307	7.2412	3.0307	3.0769	–0.0462
Lake	0.4017	0.3424	0.0593	0.3424	0.3493	–0.0069
Lena	1.0482	0.7592	0.289	0.7592	0.7606	–0.0014
Man	2.1422	0.8621	1.2801	0.8621	0.8701	–0.008
Milk	2.7529	1.5397	1.2132	1.5397	1.5004	0.0393
Plane	3.1757	1.9659	1.2098	1.9659	1.971	–0.0051

Table 6 Best K of two steganalysis

	20%	*40%*	*60%*	*80%*	*100%*
LSB	3	6	6	5	3
HM	3	6	6	4	4

This characteristic can be used to separate these two types of images. Based on the above, we describe a new steganalysis algorithm. The main idea of the algorithm is as follows: when analysing an image, we apply a steganographic embedding, and calculate the S value (S_1 and S_2) of image both before and after steganographic embedding. We then calculate the difference between S_1 and S_2, we determine whether a payload has been embedded by the value of this difference. If the difference is large, the image is considered to be Type I, if the difference is small, the image is considered to be Type II.

We use the difference of S_1 and S_2 with respect to S_1 to show the changes, it is calculated as

$$r = \frac{S_1 - S_2}{S_1} \tag{5}$$

where S_1 and S_2 are the S values of an image before and after steganographic embedding. The proposed steganalysis algorithm consists of the following steps.

1 Calculate the S value of image A which is to be analysed, after line covering, calculate S_1.

2 Steganographically embeds a payload into the image to get image B, perform line covering and calculate S_2.

3 Calculate ratio r according to formula (5).

4 If $r < T$ then the image is determined to be Type II, i.e., contain an embedded image. Where T is a threshold parameter whose value will be determined by sample training in the next section.

3.2 Threshold T

In our steganalytic method, classification is based on the relationship between r and T. In practical application, threshold T is first obtained according to the goals of the application. Here we provide some necessary definitions.

1 *accept:* the image is judged to be a stegoed image

2 *reject:* the image is judged to be a clean image

3 *false accept:* the image is incorrectly judged to be stegoed

4 *false reject:* the image is incorrectly judged to be clean.

Adjusting T can change the false rejection and FARs. We can adjust T to achieve different performance. For example, we can choose different T to get the minimum error rate, the minimum FRR, the minimum FAR or meet some other requirements. The required threshold, T, is obtained by sample training. We use images from the CBIR database (CBIR, 2011) for training. There are 963 images in the CBR database and we

use 763 images for training and 200 images for testing. The detailed description of sample training is given as follows.

1 Choose sample images.

2 Choose a steganography algorithm (LSB, HM) and embedding rate (0%, 20%, 40%, 60%, 80%,100% of full embedding), then stego the 763 images to get 763 new images.

3 For each of the 1,526 images, mark it 1 or 0 demarking, respectively, whether it is stegoed or not.

4 Stego the 1,526 images, and obtain an r value for every image.

5 Determine an appropriate value of T which can achieve the minimum error rate or satisfy some other goals. Details for this are in the following text.

We now discuss how to determine an appropriate threshold, T, in Step 5. From the experiments in Section 2.2, we see that the r values of the clean images are usually bigger than that of the stegoed images. The range of the r values of the clean images is $[r_1, r_2]$, and the range of the r values of the stegoed images is $[r_3, r_4]$. The two ranges are usually not linearly separable, they overlap as seen in Figure 6.

Figure 6 Ranges of two types of r

It is shown in Figure 6 that it is impossible to find a threshold T to separate the two areas with 100% accuracy. There are four special T values shown as follows.

1 when $T = r_3$, none of the stegoed images will be judged to be clean. So in this situation, it achieves the minimum FRR

2 when $T = r_2$, none of the clean images will be judged to be stegoed. So in this situation, it achieves the minimum FAR

3 when $T = r_5$, r_5 is an appropriate value between r_3 and r_2, it achieves the minimum error rate(ER)

4 when $T = r_6$, r_6 is another appropriate value between r_3 and r_2, it achieves an equal FRR and FAR.

An experiment is performed to explain the relationship between ER, FRR, FAR and show the four special T values. Firstly, we determine the two ranges introduced above, they are [−0.1267, 0.3597] and [0.0166, 0.5805]. Then we choose different T ($r_3 < T < r_2$) and calculate the ER, FRR and FAR, the results are shown in Figure 7. There are four special points for four special T in Figure 7, points A, B, C, and D. We achieve the minimum FAR at point A, minimum FRR at B, and minimum ER at C. At point D, FAR is equal to FRR. In order to see area Z clearly, we provide a zoomed version in Figure 8. In this paper, we choose point C to achieve the minimum error rate.

Figure 7 T and FRR, FAR, ER (see online version for colours)

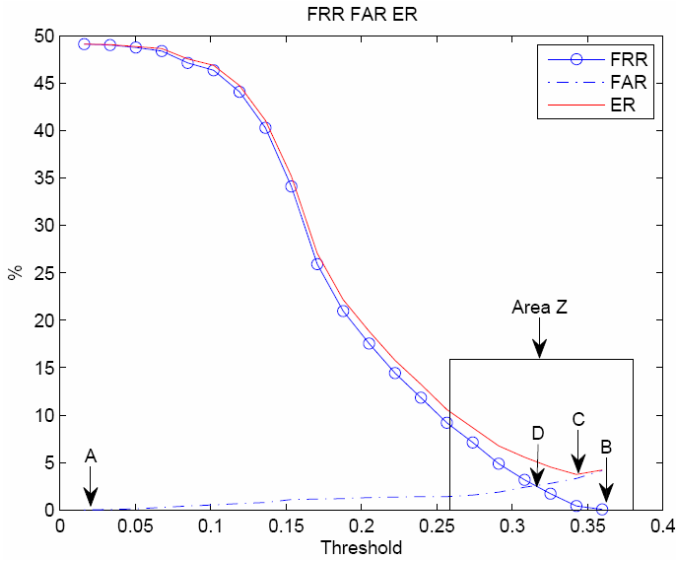
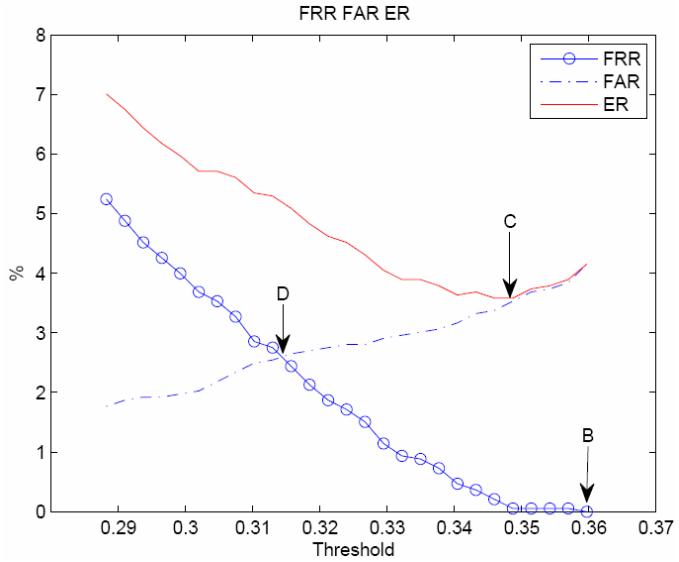


Figure 8 T and FRR, FAR, ER (zoomed) (see online version for colours)



3.3 Results and parameter K

In our steganalysis algorithm, another variable parameter is K in formula (3). Recall that K determines which lines are deemed to be long lines, and S will change greatly when K changes. So different K achieves different accuracy levels. This section shows the experimental results and explains how to choose the best K to achieve the highest accuracy. The best K for different steganalysis algorithms, at different embedding rates

will be different. So we want to figure out the best K for the needed embedding rate we want to use. Here, we first choose a steganalysis algorithm and the embedding rate, and then we use large number of images to calculate the accuracy of every K. We set K to 3, 4, 5, and 6. Figures 9 and 10 show the results of two steganalysis algorithms. From these two figures, it is seen that when the embedding rate is more than 40%, the accuracy is higher than 90%. And the accuracy is higher than 70% when the embedding rate is only 20%. So this method has good performance.

Figure 9 Accuracy of LSB (see online version for colours)

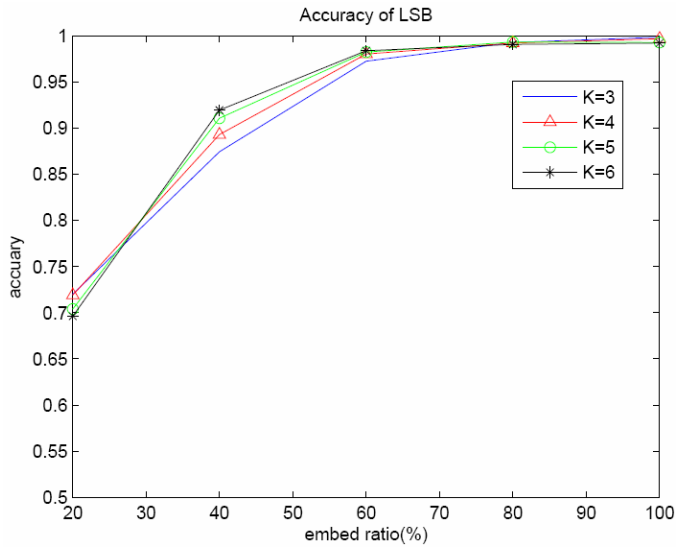
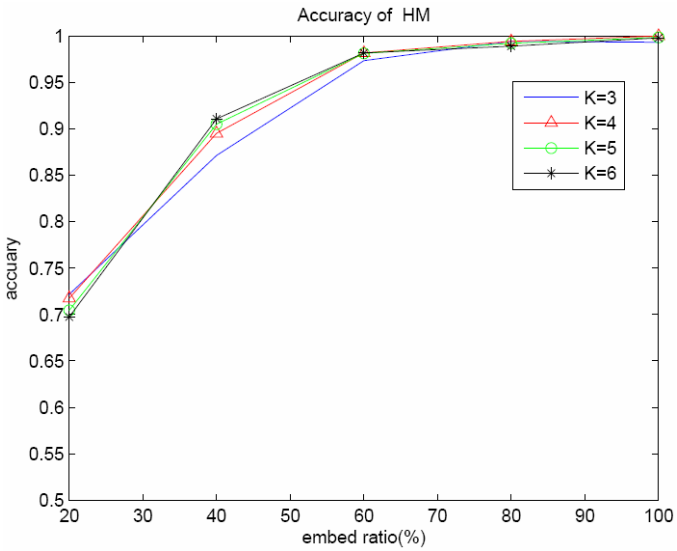


Figure 10 Accuracy of HM (see online version for colours)



At different embedding rates, K influences the accuracy. We show the best K for every embedding rate in Table 5. Then we can choose different K in the steganalysis according to actual situation. For example if we want use an LSB classifier aiming to classify images with 60% embedding rate, we can set $K = 6$. So for a particular embedding rate, we can choose a better K and design a classifier which has a higher accuracy at the embedding rate.

3.4 Capacity of line

In line covering, lines are used to cover the pixels with the same value. It can be seen from our earlier experiment that there are many fewer long lines than short. This may influence the accuracy to calculate the smoothness. Here we introduce a method to increase the number of long lines and get more features for steganalysis. The capacity of a line, C, is proposed. A line can cover adjacent pixels which have the same value of $\lfloor X / C \rfloor$, where X is the pixel value. So in the line covering in section II, the capacity of a line is 1, because all the pixels under a line have the same value. If we increase the capacity of line, a line will cover more pixels, and lines will become longer. Line covering with capacity C is defined as follows.

On the pixels of an image, along a direction (0°, 45°, 90° or 135°), lines in this direction are used to cover the pixels which have the equal result of $\lfloor X / C \rfloor$, where X is the pixel value. For example, 120, 121 and 122 can be covered by a line with $C = 3$, because $\lfloor 120 / 3 \rfloor = \lfloor 121 / 3 \rfloor = \lfloor 122 / 3 \rfloor = 40$. Line covering with capacity C is useful to increase the number of long lines and thus increase accuracy. It can also be used as a new feature in other steganalysis.

4 Line covering in steganography

4.1 Difference expansion

One important technique used in steganography is DE. DE is a technique widely used in the field of information hiding, much research is based on DE (Tian, 2003a, 2003b; Lin et al., 2008; Tian, 2002; Tian et al., 2004; Alattar, 2003, 2004a, 2004b). We introduce a method to improve DE with the help of line covering-based smoothness. First, we introduce Tian's (2003a) scheme to describe DE technique. Tian defines an integer transformation as:

$$l = \left\lfloor \frac{x+y}{2} \right\rfloor, h = x - y \tag{6}$$

where x and y are adjacent original pixels, h is the difference between x and y, l is the mean and $\lfloor \bullet \rfloor$ denotes the floor function. The inverse transform can be expressed by

$$x' = l + \left\lfloor \frac{h+l}{2} \right\rfloor, \; y' = l - \left\lfloor \frac{h}{2} \right\rfloor \tag{7}$$

where x' and y' are recovered pixels. Then this method expands h and embeds watermark b into its LSB.

$$h' = h \times 2 + b \tag{8}$$

After embedding the difference between stego pixels and original pixels is

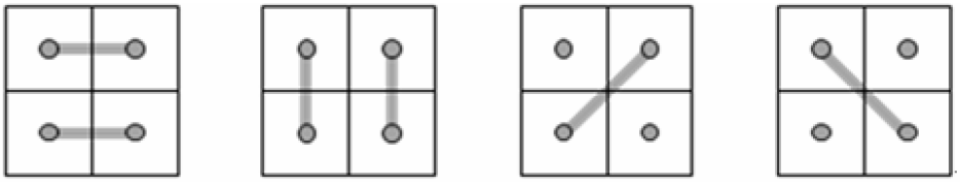
$$d_x = |x - x'| = \left\lfloor \frac{h}{2} \right\rfloor + b,\ d_y = |y - y'| = \left\lceil \frac{h}{2} \right\rceil \tag{9}$$

From (9), it can be seen that the difference d_x, d_y have a strong relationship to h. The smaller the h is, the smaller the difference is and the changes caused by steganographic embedding are smaller. So if the average h is small in an image, the quality of image after embedding by DE is better and the PSNR of the stego image is higher too. And this is the key feature used in the improvement.

4.2 Improvement

There is one question Tian does not address, it is how to choose the adjacent pixels x and y. There are four ways to choose a pair of x and y, as is shown in Figure 11.

Figure 11 Four pairing mode of two adjacent pixels



The four pairing directions of adjacent pixels are 0°, 45°, 90° and 135°. In steganography, the pairing directions of all pairs may be the same or different. Most steganography usually chooses one pairing direction in embedding, because if the pairing direction is different, we would need extra data to mark the pairing direction which requires extra data and reduces the space for data embedding. Different pairing directions do make a difference. After chosen a pairing direction, in the direction of the pairing direction, if the average h is smaller than that of other three directions, the changes caused by DE are also smaller. So the quality of the image after embedding by DE will be better. How to choose the pairing direction is an important question.

It is proven in last subsection that the smaller h achieve higher image quality. How to choose the direction which has the smaller h? Table 7 shows the relationship of smoothness (S_0, S_{90}, S_{45} and S_{135}) and average h of standard images.

From Table 7, it can be seen that for most images, if they have greater smoothness in one direction, they will have a smaller average h. A few images (*camera*, *columbia*, *crowd*, *peppers*, and *plane*) do not follow this rule, we will discuss these later. We can conclude that a greater smoothness means a smaller average h. It can also be seen from Table 7 that most images have biggest smoothness in the 0° or 90° direction, so we only consider these two directions.

If we choose the direction in which the image has the greatest smoothness, the changes caused by DE will be smallest. And subsequently the image quality will be better than that of other directions.

Table 7 Smoothness and *h* of standard images, DS is the direction of the maximum smoothness and Dh is the direction of minimum *h*

Image	S_0	S_{90}	S_{45}	S_{135}	h_0	h_{90}	h_{45}	h_{135}	*DS*	*Dh*
Baboon	0.094	0.044	0.033	0.036	12.92	16.92	18.37	18.10	0	0
Barbara	1.71	2.176	1.418	1.41	12.29	9.18	13.44	13.93	90	90
Boats	1.522	2.683	1.578	1.637	6.337	5.457	7.823	8.205	90	90
Bridge	8.043	4.784	3.411	3.159	12.14	13.62	16.17	16.65	0	0
Camera	4.768	3.905	2.408	2.533	8.972	8.099	11.16	11.37	0	90
Columbia	4.359	5.164	2.322	2.23	4.989	5.63	7.866	8.054	90	0
Couple	0.496	0.604	0.259	0.287	8.707	7.866	11.42	11.50	90	90
Crowd	11.61	12.37	9.119	7.991	6.366	6.696	8.497	9.379	90	0
Goldhill	3.079	2.831	2.701	2.692	6.448	6.813	8.582	8.683	0	0
Lake	0.434	0.388	0.373	0.413	9.227	9.64	11.35	11.02	0	0
Lax	0.042	0.043	0.032	0.037	14.47	12.59	16.37	16.15	90	90
Lena	0.671	2.103	0.725	0.694	5.573	4.125	5.736	6.48	90	90
Man	2.23	2.75	1.875	1.714	7.519	6.616	9.019	9.219	90	90
Milkdrop	2.711	4.355	2.125	1.821	4.004	3.603	4.912	4.895	90	90
Peppers	0.249	0.261	0.277	0.279	6.736	6.39	7.071	7.302	135	90
Plane	3.403	4.337	2.38	2.583	5.711	5.925	7.683	7.669	90	0
Woman1	2.286	2.792	1.922	1.734	8.29	6.849	8.61	9.358	90	90
Woman2	4.777	5.594	3.719	3.716	3.039	2.916	3.868	3.74	90	90

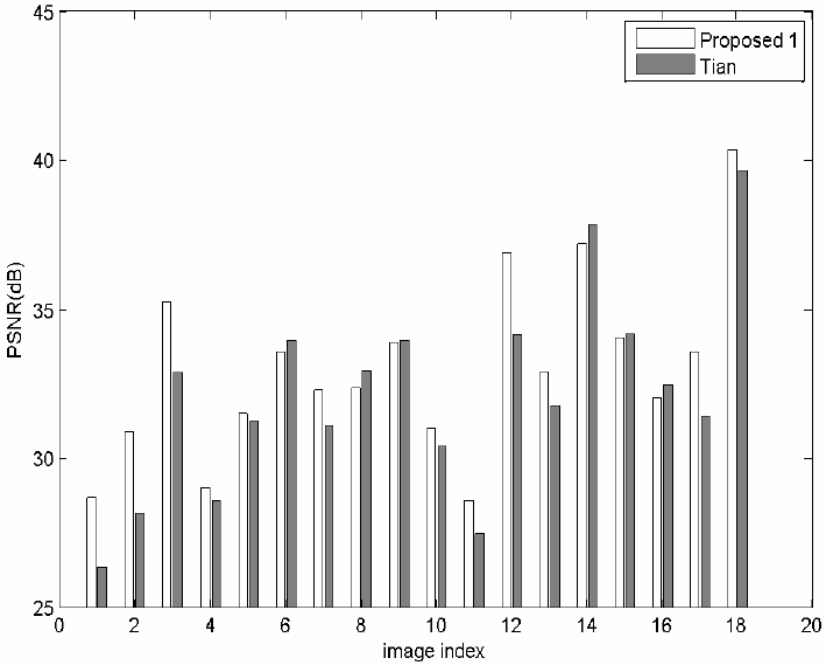
4.3 Experiments

We take our improvement in three methods (Tian, 2003a; Lin et al., 2008; Coltuc and Chassery, 2007), the first is the DE proposed by Tian (2003a), the second is the DE without location-map by Lin et al. (2008), the third is a method proposed by Azzoni et al. (2010) which is also based on DE.

Figure 12 shows the experimental results of the improvements for Tian's (2003a) DE. The embedding method is basic DE (Tian, 2003a), the images are full embedded (capacity is between 0.45 bpp and 0.5 bpp).

It can be seen from Figure 12 that using our method, because the direction having bigger smoothness is chosen to embed the information, the PSNR is bigger than in DE. Also there are some exceptions in *columbia*, *crowd*, *milkdrop*, *peppers*, *lake*, and *plane*. In these six images, our method is a little worse. We check these images and find that in one image, after partitioning the image, in the image blocks, the direction of greatest smoothness always changes. The S_0 is bigger in some blocks, while in other blocks the S_{90} is bigger. So one method to solve this is partitioning the image, and in each block embed message in the direction with greatest smoothness. But this will require extra data to be embedded.

Figure 12 Performance comparison on test images



Note: We compare our method with Tian's DE method.

The DE without location map in Lin et al. (2008) also has this problem. We improve it using the same method discussed above and the results are shown in Figure 13. We test our methods on the 18 standard images. For all the 18 images, our method works better on 13 images, has the similarly performance on three images and worse performance on two images. It proves that our method is useful. Figure 13 shows the performance comparison on four classic images, it is seen that our method has higher image quality on these four images, *Lena*, *baboon*, *Barbara*, *woman*1. There are also some exceptions as discussed above, on image peppers, *milkdrop*, *crowd*, the two methods have similarly performance because these images has similarly smoothness S_0 and S_{90}. On image *goldhill*, *columbia*, the improved DE2 is little worse than DE2. The reason is the same which is discussed above. And this problem can be solved by partitioning the image and applying the improved DE2 in blocks.

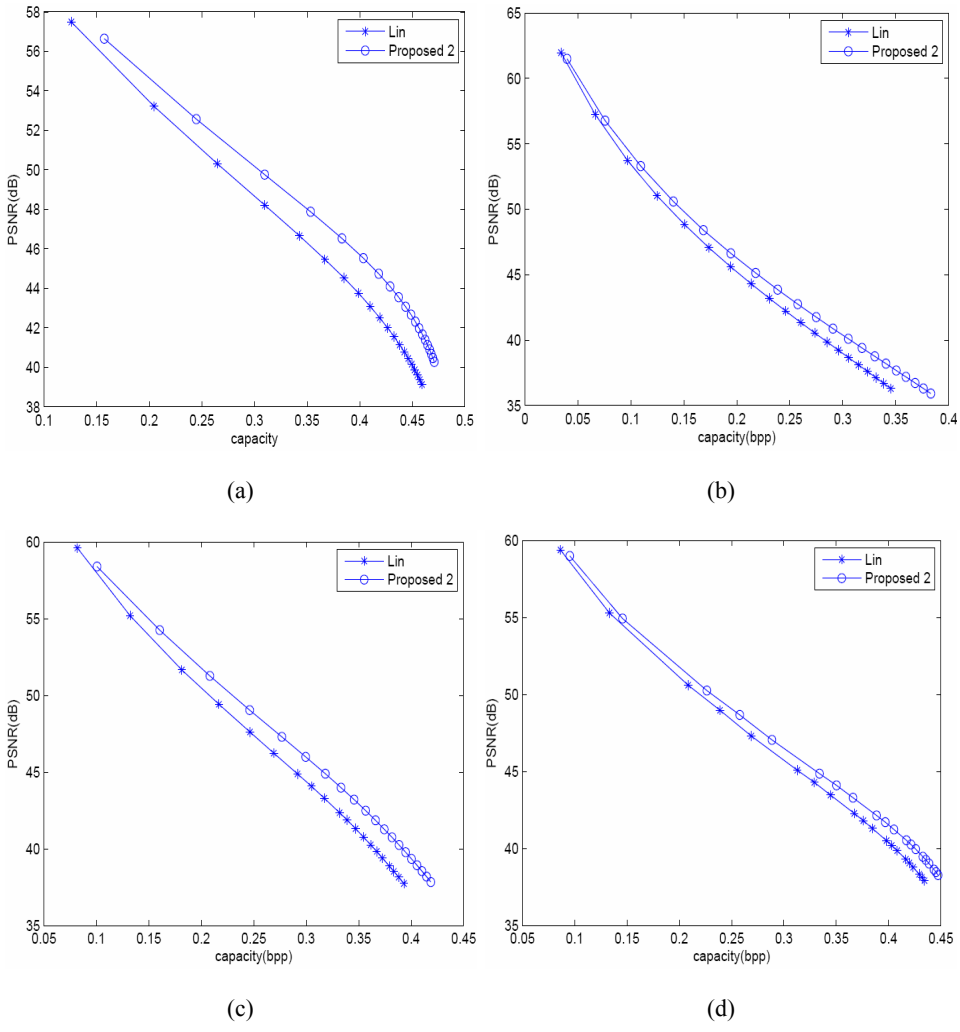
We also improve the method in Azzoni et al. (2010). Azzoni et al. (2010) proposes a reversible data hiding scheme which is based on DE. It always one direction to do integer Haar wavelet transform before data hiding while different directions bring different results. Our improvement also works on it. Figure 14 show shows the performance comparison, the test image is *baboon*. It is seen that our improvement works well on it.

In other steganographic methods, the problem of choosing pairing direction also exists. Alattar (2003, 2004a) introduces a DE method using three pixels and four pixels as

a pair. He also introduces a generalised method using N pixels as a pair in Alattar (2004b). These works do not consider the problem of choosing pairing. It is also true in these methods that the smoother direction obtains higher image quality. So the improvements in this section will apply to these methods and we will not discuss them here due to the space restriction.

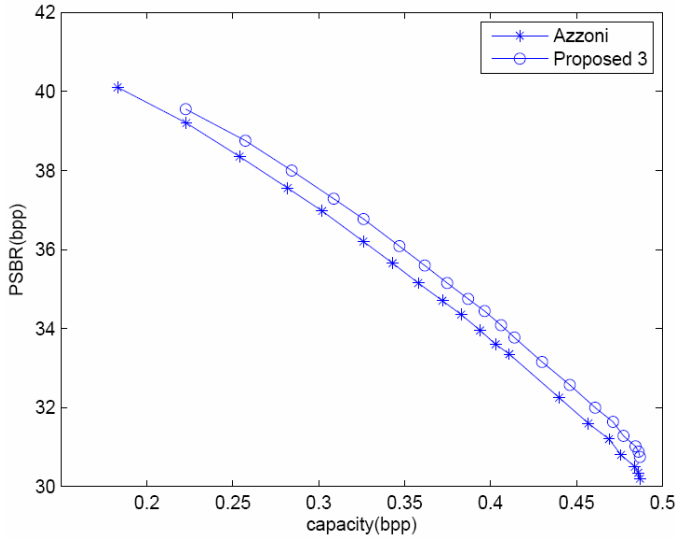
It can be concluded that the smoothness proposed in this paper is appropriate for choosing pairing direction of steganography. Choosing a better direction according to the smoothness will achieve better PSNR of the stego image.

Figure 13 Performance comparison on test image *Lena*, *baboon*, *Barbara*, *woman*1, (a) Lena (b) baboon (c) Barbara (d) woman1 (see online version for colours)



Notes: We compare our method with Lin's DE method. We give the embedding capacity versus image quality.

Figure 14 Performance comparison on test image *baboon* (see online version for colours)



Notes: We compare our method with Azzoni's DE-based method. We give the embedding capacity versus image quality.

5 Conclusions

This paper proposes a line covering method and discusses its application in steganography and steganalysis. We introduce line covering to calculate image smoothness. This feature can show the smoothness of image and it can better show features in different directions while most other image features ignore directions of features in an image. Our method can also highlight changes caused by steganography. We have proposed a steganalysis based on this feature. Our steganalysis shows good performance and can easily control the false rejection and acceptation rates by adjusting a parameter which is not available for many other steganalytic methods.

Acknowledgements

This work is supported by the Shen Zhen Key Laboratory of Data Vitalisation (Smart City), National High-Tech Research and Development Programme of China (No. 2011AA010502) and National Natural Science Foundation of China (No. 61170178).

References

Alattar, A.M. (2003) 'Reversible watermark using difference expansion of triplets', *ICIP 2003*, Vol. 1, pp.I–501.

Alattar, A.M. (2004a) 'Reversible watermark using difference expansion of quads', *ICASSP'04*, Vol. 3, pp.377–380.

Alattar, A.M. (2004b) 'Reversible watermark using the difference expansion of a generalized integer transform', *Image Processing*, Vol. 13, No. 8, pp.1147–1156.

Avcibas, I., Memon, N. and Sankur, B. (2002) 'Image steganalysis with binary similarity measures', *ICIP '02*, Vol. 3, pp.645–648.

Azzoni, M., Boato, G., Carli, M. and Egiazarian, K. (2010) 'Reversible watermarking using prediction and difference expansion', *EUVIP 2010*, pp.71–76.

Cancelli, G., Doerr, G., Cox, I.J. and Barni, M. (2008) 'Detection of ±1 lsb steganography based on the amplitude of histogram local extrema', *ICIP 2008*, Vol. 1, pp.1288–1291.

CBIR (2011) 'CBIR image database Univ. Washington, Seattle', available at http://www.cs.washington.edu/research/imagedatabase/groundtruth/.

Chandramouli, R., Kharrazi, M. and Memon, N. (2004) 'Image steganography and steganalysis: concepts and practice', *2nd Int. Workshop Digital Watermarking*, pp.204–211.

Chen, M., Chen, Z., Zeng, X. and Xiong, Z. (2010) 'Model order selection in reversible image watermarking', *Selected Topics in Signal Processing*, Vol. 4, No. 3, pp.592–604.

Coltuc, D. and Chassery, J.M. (2007) 'Very fast watermarking by reversible contrast mapping', *Signal Processing Letters*, Vol. 14, No. 4, pp.255–258.

De Vleeschouwer, C., Delaigle, J.F. and Macq, B. (2003) 'Circular interpretation of bijective transformations in lossless watermarking for media asset management', *Multimedia*, Vol. 5, No. 1, pp.97–105.

Fridrich, J. and Goljan, M. (2002) 'Practical steganalysis of digital images-state of the art', *Proc. SPIE Photonics West*, Vol. 4675, pp.1–13.

Fridrich, J., Goljan, M. and Du, R. (2001) 'Detecting LSB steganography in color, and gray-scale images', *Multimedia, IEEE*, Vol. 8, No. 4, pp.22–28.

Goljan, M., Fridrich, J. and Holotyak, T. (2006) 'New blind steganalysis and its implications', *Proceedings of SPIE*, Vol. 6072, pp.1–13.

Lin, C.C., Yang, S.P. and Hsueh, N.L. (2008) 'Lossless data hiding based on difference expansion without a location map', *CISP'08*, Vol. 2, pp.8–12.

Liu, Q., Sung, A., Xu, J. and Ribeiro, J. (2006) 'Image complexity and feature extraction for steganalysis of lsb matching steganography', *ICPR '06*, Vol. 2, pp.267–270.

Luo, L., Chen, Z., Chen, M., Zeng, X. and Xiong, Z. (2010) 'Reversible image watermarking using interpolation technique', *Information Forensics and Security*, Vol. 5, No. 1, pp.187–193.

Lyu, S. and Farid, H. (2004) 'Steganalysis using color wavelet statistics and one-class support vector machines', *Proceedings of SPIE*, Vol. 5306, pp.35–45.

Moulin, P. and Liu, J. (1999) 'Analysis of multiresolution image denoising schemes using generalized Gaussian and complexity priors', *Information Theory*, Vol. 45, No. 3, pp.909–919.

Provos, N. and Honeyman, P. (2001) 'Detecting steganographic content on the internet', *Ann Arbor*, Vol. 1001, pp.48103–4943.

Sabeti, V., Samavi, S., Mahdavi, M. and Shirani, S. (2010) 'Steganalysis and payload estimation of embedding in pixel differences using neural networks', *Pattern Recogn.*, Vol. 43, No. 1, pp.405–415.

Tian, J. (2002) 'Wavelet-based reversible watermarking for authentication', *Proc. of SPIE*, Vol. 4675, pp.679–690.

Tian, J. (2003a) 'Reversible data embedding using a difference expansion', *Circuits and Systems for Video Technology*, Vol. 13, No. 8, pp.890–896.

Tian, J. (2003b) 'High capacity reversible data embedding and content authentication', *ICASSP '03*, Vol. 3, pp.517–520.

Tian, J. and Wells, R.O., Jr. (2004) 'Reversible data-embedding with a hierarchical structure', *ICIP '04*, Vol. 5, pp.3419–3442.

Westfeld, A. and Pfitzmann, A. (2000) 'Attacks on steganographic systems', *Lecture Notes in Computer Science*, pp.61–67.

Zhang, T., Zhang, Y., Ping, X. and Song, M. (2006) 'Detection of LSB steganography based on image smoothness', *International Conference on Multimedia and Expo 2006*, pp.1377–1380.