ESSENTIALS OF IMAGE STEGANALYSIS MEASURES

BHANU PRAKASH BATTULA[1] , R. SATYA PRASAD[2]

[1]Asst.Professor, Department of Computer Science and Engineering, Vignan's Nirula Institute of Technology and Science, Guntur, Andhra Pradesh, Pin-520007, India.

[2]Professor, Department of Science and Technology, Acharya Nagarjuna University, Guntur, Pin-522001, Andhra Pradesh, India.

ABSTRACT

We present a technique for steganalysis of images that have been subjected to Least Significant Bit (LSB) type steganographic algorithms. The seventh and eight bit planes in an image are used for the computation of several binary similarity measures and we analyze the security of Least Significant Bit (LSB) embedding for hiding messages in high-color-depth digital images. The basic idea is that, the correlation between the bit planes as well the binary texture characteristics within the bit planes will differ between a stego-image and a cover-image. These telltale marks can be used to construct a steganalyzer, that is, a multivariate regression scheme to detect the presence of a steganographic message in an image.

KEYWORDS: *Steganography, Steganalysis, Binary Simulation, Least Significant Bit Operations.*

1. INTRODUCTION

STEGANOGRAPHY refers to the science of "invisible" communication. Unlike cryptography, where the goal is to secure communications from an eavesdropper, steganographic techniques strive to hide the very presence of the message itself from an observer [1].Given the proliferation of digital images, and given the high degree of redundancy present in a digital representation of an image (despite compression), there has been an increased interest in using digital images for the purpose of steganography. The simplest image steganography techniques essentially embed the message in a subset of the LSB (least significant bit) plane of the image, possibly after encryption [2]. Popular steganographic tools based on LSB-embedding vary in their approach for hiding information. Methods like Steganos and Stools use LSB embedding in the spatial domain, while others like Jsteg embed in the frequency domain. Non-LSB steganography techniques include the use of quantization and dithering [2].

Since the main goal of steganography is to communicate securely in a completely undetectable manner, an adversary should not be able to distinguish in any sense between *cover-objects* (objects not containing any secret message) and *stego-objects* (objects containing a secret message). In this context, *steganalysis* refers to the body of techniques that are conceived to distinguish between cover-objects and stego-objects.

Recent years have seen many different steganalysis techniques proposed in the literature. Some of the earliest work in this regard was reported by Johnson and Jajodia [3],[4]. They mainly look at palette tables in GIF images and anomalies caused therein by common stego-tools. A more principled approach to LSB steganalysis was presented in [5] by Westfeld and Pfitzmann. They identify Pairs of Values (PoV's), which consist of pixel values that get mapped to one another on LSB flipping. Fridrich, Du and Long [6] define pixels that are close in color intensity to be a difference of not more than one count in any of the three color planes. They then show that the ratio of close colors to the total number of unique colors increases significantly when a new message of a selected length is embedded in a cover image as opposed to when the same message is embedded in a stego-image. A more sophisticated technique that provides remarkable detection accuracy for LSB embedding, even for short messages, was presented by Fridrich et al. in [7]. Avcibas, Memon and Sankur [8] present a general- technique for steganalysis of images that is applicable to a wide variety of embedding techniques including but not limited to LSB embedding. They demonstrate that steganographic schemes leave statistical evidence that can be exploited for detection with the aid of image quality features and multivariate regression analysis. Chandramouli and Memon [9] do a theoretical analysis of LSB steganography and derive a closed form expression of the probability of false detection in terms of the number of bits that are hidden. This leads to the notion of steganographic capacity, that is, the number of bits one can hide in an image using LSB techniques without causing statistically significant modifications.

In this paper, we present a new steganalysis technique for detecting stego-images. The technique uses binary similarity measures between successive bit planes of an image to determine the presence of a hidden message. In comparison to previous work, the technique we present differs as follows:

- [3] and [4] present visual techniques and work for palette images. Our technique is based on statistical analysis and works with any image format.
- [5], [6] and [7] work only with LSB encoding. Our technique aims to detect messages embedded in other bit planes as well.
- [5], [6] and [7] detect messages embedded in the spatial domain. The proposed technique works with both spatial and transform-domain embedding.
- Our technique is more sensitive than [5], [6] and [8]. However, in its current form it is not as accurate as [7] and cannot estimate the length of the embedded message like [7].

Notice that our scheme, like [5,6,7] does not need a reference image for steganalysis. The rest of this paper is organized as follows: In Section 2 we review binary similarity measures. In Section 3 we describe our steganalysis technique. In Section 4, we present a new steganalytic technique based on analyzing the structure of the set of unique colors in the RGB color cube. In Section 5, we optimize the steganalytic technique by adjusting its parameters to minimize the probability of making an erroneous decision

2. BINARY SIMILARITY MEASURES

There are various ways to determine similarity between two binary images. Classical measures are based on the bit-by-bit matching between the corresponding pixels of the two images. Typically, such measures are obtained from the scores based on a contingency table (or matrix of agreement) summed over all the pixels in an image. In this study, where we examine lower order bit-planes of images, for the presence of hidden messages, we have found that it is more relevant to make a comparison based on *binary texture statistics*. Let $\mathbf{x}_i = \{x_{i-k} |, \ k = 1, \ldots, K\}$ and $\mathbf{y}_i = \{y_{i-k} |, \ k = 1, \ldots, K\}$ be the sequences of bits representing the 4-neighborhood pixels, where the index i runs over all the image pixels. Let

$$\chi_s^r = \begin{cases} 1 & if \quad x_r = 0 \quad and \quad x_s = 0 \\ 2 & if \quad x_r = 0 \quad and \quad x_s = 1 \\ 3 & if \quad x_r = 1 \quad and \quad x_s = 0 \\ 4 & if \quad x_r = 1 \quad and \quad x_s = 1 \end{cases} \tag{1}$$

Then we can define the agreement variable for the pixel x_i as: $\alpha_i^j = \sum_{k=1}^{K} \delta(\chi_i^{i-k}, j)$, $j = 1, \ldots, 4$, K = 4, where

$$\delta(m,n) = \begin{cases} 1 & , \quad m = n \\ 0 & , \quad m \neq n \end{cases}. \tag{2}$$

The accumulated agreements can be defined as:

$$a = \frac{1}{MN} \sum_i \alpha_i^1 , \quad b = \frac{1}{MN} \sum_i \alpha_i^2 ,$$
$$c = \frac{1}{MN} \sum_i \alpha_i^3 , \ d = \frac{1}{MN} \sum_i \alpha_i^4 . \tag{3}$$

These four variables {a,b,c,d} can be interpreted as the one-step co-occurrence values of the binary images. Normalizing the histograms of the agreement scores for the 7th bit-plane can be defined as follows:

$$p_7^j = \sum_i \alpha_i^j / \sum_i \sum_j \alpha_i^j . \tag{4}$$

Similarly, one can define p_8^j for the 8th bit plane. In addition to these we calculate the Ojala texture measure as follows. For each binary image we obtain a 16-bin histogram based on the weighted neighborhood as shown in Fig. 1, where the score is given by: $S = \sum_{i=0}^{3} x_i 2^i$ by weighting the four directional neighbors as in Fig. 1.

	1	
8	x_i	2
	4	

Fig. 1 *The weighting of the neighbors in the computation of Ojala score. S= 4+8=12 given W, S bits 1 and E, N bits 0.*

The resulting Ojala measure is the mutual entropy between the two distributions, that is

$$m_7 = -\sum_{n=1}^{N} S_n^7 \log S_n^8 \,, \qquad (5)$$

where N is the total number of bins in the histogram, S_n^7 is the count of the n'th histogram bin in the 7th bit plane and S_n^8 is the corresponding one in the 8th plane.

Table 1: *Binary Similarity Measures*

Similarity Measure	*Description*
Sokal & Sneath Similarity Measure 1	$m_1 = \frac{a}{a+b} + \frac{a}{a+c} + \frac{d}{b+d} + \frac{d}{c+d}$
Sokal & Sneath Similarity Measure 2	$m_2 = \frac{ad}{\sqrt{(a+b)(a+c)(b+d)(c+d)}}$
Sokal & Sneath Similarity Measure 3	$m_3 = \frac{2(a+d)}{2(a+d)+b+c}$
Variance Dissimilarity Measure	$m_4 = \frac{b+c}{4(a+b+c+d)}$
Dispersion Similarity Measure	$m_5 = \frac{ad-bc}{(a+b+c+d)^2}$
Co-occurrence Entropy	$dm_6 = \sum_{j=1}^{4} p_7^j \log p_8^j$
Ojala Mutual Entropy	$dm_7 = -\sum_{n=0}^{15} S_n^7 \log S_n^8$

Using the above definitions various binary image similarity measures are defined as shown in Table 1. The measures m_1 to m_5 are obtained for seventh and eighth bits separately by adapting the parameters {*a*,*b*,*c*,*d*} (3) to the classical binary string similarity measures, such as Sokal & Sneath. Then their differences $dm_i = m_i^{7th} - m_i^{8th} \quad i = 1,\ldots,5$ are used as the final measures. The measure dm_6 is defined as the co-occurrence entropies using the 4-bin histograms of the 7th and 8th bit planes. Finally the measure dm_7 is somewhat different in that we use the neighborhood-weighting mask proposed by Ojala [16]. Thus we obtain a 16-bin histogram for each of the planes and then calculate their mutual entropy.

3. STEGANALYSIS BASED ON BINARY MEASURES

Our approach is based on the fact that embedding a message in an image has a telltale effect on the nature of correlation between contiguous bit-planes. Hence we hypothesize that binary similarity measures between bit planes will cluster differently for clean and stego-images. This is the basis of our steganalyzer that aims to classify images as marked and unmarked.

We conjecture that hiding information in any bit plane decreases the correlation between that plane and its contiguous neighbors. For example, for LSB steganography, one expects a decreased similarity between the seventh and the eighth bit planes of the image as compared to its unmarked version. Hence, similarity measures between these two LSB's should yield higher scores in a clean image as compared to a stego-image, as the embedding process destroys the preponderance of bit pair matches.

Since the complex bit pair similarity between bit planes cannot be represented by one measure only, we decided to use several similarity measures to capture different aspects of bit plane correlation. The steganalyzer is based on the regression of the seven similarity measures listed in Table 1:

$$y = \beta_1 m_1 + \beta_2 m_2 + \ldots + \beta_q m_q \qquad (6)$$

where $\{m_1, m_2, \ldots m_q\}$ are the *q* similarity scores and $\{\beta_1, \beta_2, \ldots \beta_q\}$ are their regression coefficients. In other words we try to predict the state y, whether the image contains a stego-message (*y* = 1) or not (*y* = -1), based on the bit plane similarity measures. Since we have *n* observations, we have the set of equations

$$y_1 = \beta_1 m_{11} + \beta_2 m_{12} + \ldots + \beta_q m_{1q} + \varepsilon_1$$

$$y_n = \beta_1 m_{n1} + \beta_2 m_{n2} + \ldots + \beta_q m_{nq} + \varepsilon_n \qquad (7)$$

where m_{kr} is the *r*'th similarity measure observed in the *k*'th test image. The corresponding optimal MMSE linear predictor $\boldsymbol{\beta}$ can be obtained by using the matrix M of similarity measures:

$$\hat{\boldsymbol{\beta}} = \left(M^T M\right)^{-1}\left(M^T \mathbf{y}\right). \qquad (8)$$

Once prediction coefficients are obtained in the training phase, these coefficients can then be used in the testing phase. Given an image in the test phase, binary measures are computed and using the prediction coefficients, these scores are regressed to the output value. If the output exceeds the threshold 0 then the decision is that the image is embedded, otherwise the decision is that the image is not embedded. That is, using the prediction

$$\hat{y} = \hat{\beta}_1 m_1 + \hat{\beta}_2 m_2 + ... + \hat{\beta}_q m_q \qquad (9)$$

the condition $\hat{y} \geq 0$ implies that the image contains a stego-message, and the condition $\hat{y} < 0$ signifies that it does not.

The above shows how one can design a steganalyzer for the specific case of LSB embedding. The same procedure generalizes quite easily to detect messages in any other bit plane. Furthermore, our initial results indicate that we can even build steganalyzer for non-LSB embedding techniques like the recently designed algorithm F5 [11]. This is because a technique like F5 (and many other robust watermarking techniques which can be used for steganography in an active warden framework [8]) results in the modification of the correlation between bit planes. We note that LSB techniques randomize the last bit plane. On the other hand Jsteg or F5 introduce more correlation between 7th and 8th bit planes, due to compression that filters out the natural noise in a clean image. In other words whereas spatial domain techniques decrease correlation, frequency domain techniques increase it.

4. STEGANALYSIS OF LSB ENCODING

Johnson and Jajodia [21,22] present a careful analysis of fingerprints introduced by current steganographic software packages. They point out that most techniques for palette images with a small number of colors can be easily broken by analyzing the palette for close pairs of colors. Pfitzman and Westfeld [23] introduce a powerful Chi-square steganalytic technique that can reliably detect images with secret messages that are embedded in consecutive pixels (such as in Steganos, J-Steg, S-tools, or EZ Stego). However, their technique will not be effective for raw high-color images and for messages that are randomly scattered in the image (unless the capacity of the stego-technique is close to 1 bit per pixel). In this section, we present a new steganalytic technique that can be successfully used for raw high-color-depth images with randomly scattered secret messages.

A large number of methods for hiding messages in raw losslessly compressed images (BMP, RAS, PGM, and many other formats) are based on replacing the least significant bit (LSB) of every gray-scale or color channel with message bits. Thus, on average only one half of the LSBs are changed. The logic behind this scheme is that the LSBs in typical scanned images or images taken with a digital camera are essentially random, and replacing them with an encrypted (i.e., randomized) message will not introduce any detectable artifacts. This would be essentially true, if the number of unique colors in the cover image was comparable to the number of pixels in the image. However, we have observed that the number of unique colors for true-color images is typically significantly smaller than the number of pixels in the image. The ratio of the number of unique colors to the number of pixels ranges from roughly 1:2 for high quality scans in BMP format to 1:6 or even lower for JPEG images or for typical video grabs. The number of unique colors tends to be smaller for JPEG images due to the low-pass character of the JPEG compression. This observation is very important because it means that many true-color images have a relatively small "palette". After LSB embedding, the new color palette will have a very disctinct feature – many pairs of very *close colors*. The presence of too many pairs of close colors is an indication of using the LSB encoding for steganography. While this type of artifact was recognized by researchers before [18,19], it was commonly thought that this was applicable only to images that use small palettes (GIF, PNG formats with at most 256 colors in their palettes). In this paper, we show that a large number of true-color images can also be attacked using a similar idea. We derived a statistical quantity and an algorithm that can be used for reliable filtering of messages with and without secret messages.

We propose to test the presence of messages in true-color images using the following idea. Let us denote the number of unique colors in an image as *U*. Looking at unique colors only, let *P* be the number of *close color pairs* in the image palette. We say that two colors (R_1, G_1, B_1) and (R_2, G_2, B_2) are close if $|R_1 - R_2| \leq 1$, $|G_1 - G_2| \leq 1$, and $|B_1 - B_2| \leq 1$. This is equivalent to saying that $(R_1-R_2)^2+(G_1-G_2)^2+(B_1-B_2)^2 \leq 3$. The number of all color pairs is

$$\binom{U}{2} \geq P.$$

The ratio *R* between the number of closest pairs of colors and all pairs of colors,

$$R = \frac{P}{\binom{U}{2}},$$

gives us an idea about the relative number of close colors in the image. After the embedding, the number of unique colors will be increased to U' and we can evaluate the number of close pairs P' and the number of all pairs of colors. Now, the idea is that for an image that does not have a message, the number of close pairs of colors relative to the number of all possible pairs of colors will be smaller than for an image that has a message already embedded in it. However, it appears that it is almost impossible to find a threshold for this ratio *R* for all images due to a large variation of the number of unique colors *U*. Fortunately, we have made an important observation that enables us to reliably distinguish between images with and without messages. In particular, we have noticed that if an image already contains a large message, embedding another message in it does not modify the ratio *R* in any significant manner. On the other hand, if the image does not contain a secret message, the ratio *R* increases significantly. Thus, we propose this relative comparison of the ratio *R* as the decision criterion. It makes much more sense to use this relative criterion because this way, we compare relative increase in close color pairs rather the volatile absolute increase.

Detection algorithm:

1. To find out whether or not an image has a secret message in it, calculate the ratio *R* between the number of all pairs of close colors *P* and the number of all color pairs (recall that *U* is the number of unique colors in the image):

$$R = \frac{P}{\binom{U}{2}}.$$

2. Using LSB embedding in randomly selected pixels (and channels for color $M\times N$ images), embed a test message of the size $\alpha 3MN$ bits. Smaller values of α will lead to faster techniques. Below, we discuss the selection of an optimal value for α to minimize the probability of making an erroneous decision.

3. Denote the corresponding quantities for the new image after embedding the test message as U' and P', and calculate the ratio R' for the new image with the test message

$$R' = \frac{P'}{\binom{U'}{2}}.$$

Now, if the image has already had a large message hidden inside, the two ratios will be almost the same, $R \cong R'$. However, if the image did not have a message in it, we expect $R' > R$. Thus, as a separating statistics, we can take the ratio R'/R. Obviously, if the secret message size is too small, the two ratios will be very close to each other and as a result we will not be able to distinguish images with and without messages. On the other hand, if a large secret message (i.e., message with length comparable to the number of pixels in the image) is embedded in the image, we expect the two ratios to be sufficiently different. The threshold set for separation of the two image sets will have to be chosen to minimize the number of false accusations while keeping the ratio of missed detections reasonable. We performed numerical experiments with an image database of 300 color images, 350×250 pixels, stored as JPEGs. If every LSB of every pixel and color channel is modified, we have a steganographic capacity equal to 350×250×3/8 bytes = 32.8kB. A message of length 20kB (roughly 2/3 of the maximal capacity when each pixel carries 3 bits) was embedded in each image to form a new database of images with messages. Then, we ran the detection algorithm for both databases and tested the message presence by embedding a test message of size 1kB ($\alpha \cong 1/30$). As a result, we obtained the values of R/R' for both databases. The results are shown in Figure 1. The dashed curve corresponds to the database of images with messages and the solid curve corresponds to the original database without messages, both after embedding the 1kB test message. To separate the two curves, we choose the threshold *Th* as 1.1 (see Figure 1).

Below, we discuss the selection of the threshold based on probabilities of making erroneous decisions, and the optimal size of the test message.

5. PARAMETER OPTIMIZATION

We have performed the same experiment as in the previous paragraph for different size of the secret message ranging from 1% of the total number of color channels (3*MN*) to 50% (half of the color channels contain a message bit) and different sizes of the test message (α = 0.01−0.5). The solid Gaussian peak $N(\mu,\sigma)$ with probability density $f_{\mu,\sigma}$ (Figure 1) does not change with the message size – it corresponds to the probability density function of the ratio $R/R\,'$ for images without messages. The dashed Gaussian distribution $N(\mu(s),\sigma(s))$ with probability density $f_{\mu(s),\sigma(s)}$ corresponds to images with messages and it changes with the secret message size s:

$$f_{\mu,\sigma} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$f_{\mu(s),\sigma(s)} = \frac{1}{\sqrt{2\pi\sigma(s)^2}} e^{-\frac{(x-\mu(s))^2}{2\sigma^2(s)}} .$$

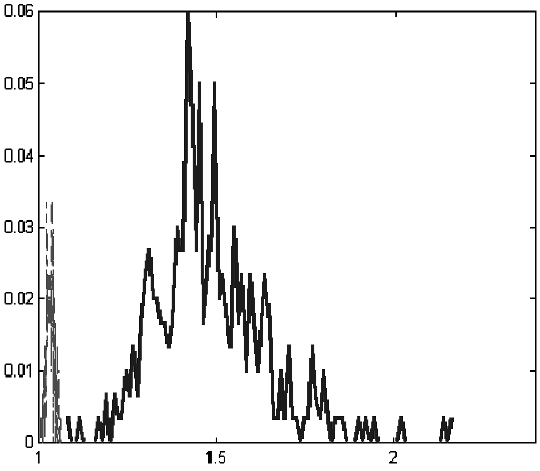

Figure 1 The ratio R'/R for 300 images. The thin dashed curve corresponds to images with an embedded message of length equal to 2/3 of the total available number of LSBs (3*MN*). The bold solid curve corresponds to images without any embedded messages**.**

We have $\mu \;\; >\mu\ (s)$ for al s. The distribution $N(\mu(s),\sigma(s))$ becomes flatter and the peak moves to the right with the decreasing size of the secret message, and it is narrower and shifted towards zero with increasing the secret message size (it is easier to separate the two peaks for larger secret message sizes). As the secret message size decreases, the solid and dashed Gaussian peaks start to overlap and we obtain a non-zero probability of making both types of errors. We denote the error of denoting an image as containing a secret message when it, in fact, does not contain any messages, as type I, and the other error or missing a secret message as type II

Type I error: Detecting a false message,
Type II error: Missing a secret message.

Threshold selection

One way of assigning the threshold would be to require the two probabilities of making error I and II to be equal $P(\mathrm{I}) = P(\mathrm{II})$

$$P(\mathrm{I}) = \int_{-\infty}^{Th} \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}}\, dx = \int_{Th}^{\infty} \frac{e^{-\frac{(x-\mu(s))^2}{2\sigma^2(s)}}}{\sqrt{2\pi\sigma(s)^2}}\, dx = P(\mathrm{II}) .$$

After substituting $w = (x - \mu\,)/\sigma$ in the left hand side integral and $w' = (x - \mu(s))/\sigma(s)$ in the right hand side, and comparing the integral limits, we obtain the following linear equation for the threshold Th: $(Th - \mu(s))/\sigma(s) = (\mu - Th)/\sigma,$

which gives us a simple expression for Th

$Th = (\mu\sigma(s) + \mu(s)\sigma)/(\sigma + \sigma(s))$.

In each particular application, the user should have the freedom to change the threshold Th to adjust for the importance of not missing an image with a secret message at the expense of making more errors of the type I.

It seems that it makes more sense to minimize the overall probability of making both errors rather than making the errors of type I and II equal. It can be shown that the requirement of minimizing the overall error probability leads to the same threshold selection. The proof of this statement is omitted due to space limitations of this paper.

Tables 1 – 4 illustrate the threshold and error probability for several different test message sizes and different secret message sizes.

Embedded message	*Th*	*T*
1%	1.0407	40.52 %
5%	1.0214	16.23 %
10%	1.0114	8.41%
20%	1.0047	5.06%
50%	1.0016	3.94%
100%	1.0011	3.63%

Table 1 Results for test message size 1%

Embedded message	*Th*	*T*
1%	1.1606	39.64%
5%	1.0935	10.65%
10%	1.0506	4.67%
20%	1.0206	1.95%
50%	1.0059	1.21%
100%	1.0028	1.10%

Table 2 Results for test message size 5%

Embedded message	*Th*	*T*
1%	1.3800	40.33%
5%	1.2675	15.03%
10%	1.1737	4.50%
20%	1.0736	0.82%
50%	1.0184	0.26%
100%	1.0068	0.21%

Table 3 Results for test message size 20%

Embedded message	*Th*	*T*
1%	1.5368	41.48%
5%	1.4139	19.08%
10%	1.3139	7.86%
20%	1.1968	2.05%
50%	1.0456	0.21%
100%	1.0088	0.02%

Table 4 Results for test message size 50%

There is another parameter in our detection scheme that needs to be carefully adjusted – the size of the test message. Based on our experiments, it turns out that for larger secret message size, larger test messages should be used. However, since we do not have any information about the secret message size, we need to settle on a compromise. In our simulations, we use the first method for threshold selection as introduced in the previous paragraph. We experimented with a color image with 250×350 pixels. The size of the test message will be related to the maximal capacity of the LSB embedding method – one bit per color channel (3 bits per pixel). To find the optimal test message size, we calculate the test message size that gives us the smallest probability of error. For the secret message size equal to 10kB (30% of the maximal capacity), the optimal test message size was 8kB (α=25%). For a smaller secret message size (1kB), the optimal test message size was determined as 1.5kB (α=5%). Looking at Tables 5 and 6, we observe that the minimum for the larger secret message size is rather flat. It is also easier to detect a large message than a small message. Therefore, we make a compromise and set the optimal test message size to α=5% of the maximal image capacity 3*MN*.

Test message size	1k	2k	3k	4k	5k
Error probability (%)	2.25	1.68	1.06	0.80	0.66
Test message size	6k	7k	8k	9k	10k
Error probability (%)	0.59	0.57	0.55	0.56	0.59

Table 5 Results of using different test message size with 10k bytes secret message

Test message size	0.8k	0.9k	1k	1.5k	1.6k
Error probab. (%)	21.38	21.04	20.46	20.29	20.33
Test message size	1.7k	2k	3k	4k	5k
Error prob. (%)	20.25	20.33	20.70	21.34	21.77

Table 6 Result of using different test message size with 1k bytes secret message

From the test results, we can draw the following conclusions:

1. The probability of error prediction is mainly determined by the size of the secret message. The influence of the test message size is much smaller.
2. The optimal test message size is different for different secret message size. In our experiment, it is about 5% of the maximal image capacity when the size of the secret message is 1k bytes, and it is about 25% of the maximal image capacity when the size of the secret message is 10k bytes.

The experimental results suggest that it is possible to reliably detect the presence of secret message embedded in digital images using the LSB technique. The reliability of the detection method increases with decreasing number of unique colors in the original image. Some high-quality scans stored losslessly may have a very high number of unique colors (more than 1/2 of the number of pixels) and the results of the detection technique may become unreliable.

6. CONCLUSIONS

In this paper, we have addressed the problem of steganalysis of marked images. We have developed a technique for discriminating between cover-images and stego-images that have been subjected to the LSB type steganographic marking. Our approach is based on the hypothesis that steganographic schemes leave telltale evidence between 7th and 8th bit planes that can be exploited for detection. The steganalyzer has been instrumented with binary image similarity measures and multivariate regression. Simulation results with commercially available steganographic techniques indicate that the new steganalyzer is effective in classifying marked and non-marked images.

As described above, the proposed technique is not suitable for active warden steganography (unlike [8]) where a message is hidden in higher bit depths. But initial results have shown that it can easily generalize for the active warden case by taking deeper bit plane correlations into account. For example, we are able to detect Digimarc when the measures are computed for 3rd and 4th bit planes.

REFERENCES:

[1] G. J. Simmons, Prisoners' Problem and the Subliminal Channel (The), CRYPTO83 - Advances in Cryptology, August 22-24. 1984. pp. 51-67.

[2] N. F. Johnson, S. Katzenbeisser, “A Survey of steganographic techniques”, in S. Katzenbeisser and F. Petitcolas (Eds.): *Information Hiding*, pp. 43-78. Artech House, Norwood, MA, 2000.

[3] N. F. Johnson, S. Jajodia, “Steganalysis: The investigation of Hidden Information”, *IEEE Information Technology Conference*, Syracuse, NY, USA, 1998.

[4] N. F. Johnson, S. Jajodia, “Steganalysis of Images created using current steganography software”, in David Aucsmith (Ed.): *Information Hiding*, LNCS 1525, pp. 32-47. Springer-Verlag Berlin Heidelberg 1998.

[5] A. Westfield, A. Pfitzmann, “Attacks on Steganographic Systems”, in *Information Hiding*, LNCS 1768, pp. 61-76, Springer-Verlag Heidelberg, 1999.

[6] J. Fridrich, R. Du, M. Long, “Steganalysis of LSB Encoding in Color Images”, Proceedings of ICME 2000, New York City, July 31-August 2, New York, USA

[7] J. Fridrich, M. Goljan and R. Du, “Reliable Detection of LSB Steganography in Color and Grayscale Images”. *Proc. of the ACM Workshop on Multimedia and Security*, Ottawa, CA, October 5, 2001, pp. 27-30.

[8] I. Avcibas, N. Memon and B. Sankur, “Steganalysis Using Image Quality Metrics”, *Security and Watermarking of Multimedia Contents*, SPIE, San Jose, 2001.

[9] R. Chandramouli and N. Memon, “Analysis of LSB Based Image Steganography Techniques”, Proceedings *of the International Conference on Image Processing*, Thessalonica, Greece, October 2001.

[10] C. Rencher, *Methods of Multivariate Analysis*, New York, John Wiley (1995).

[11] F5—A Steganographic Algorithm: High Capacity Despite Better Steganalysis. *Information Hiding*. Proceedings, LNCS 2137, Springer-Verlag Berlin 2001

[12] Steganos II Security Suite, http://www.steganos.com/english/steganos/download.htm[13] A. Brown, S-Tools Version 4.0, Copyright © 1996, http://members.tripod.com/steganography/stego/s-tools4

[14] J. Korejwa, Jsteg Shell 2.0, http://www.tiac.net/users/korejwa/steg.htm.

[15]http://www.cl.cam.ac.uk/~fapp2/watermarking/benchmark/image_database.html

[16] T. Ojala, M. Pietikainen, D. Harwood, A Comparative Study of Texture Measurss with Classification Based on Feature distributions, Pattern Recognition, vol. 29, pp.

[17] Andersen, R.J., Petitcolas, F.A.P., On the limits of steganography. *IEEE Journal of Selected Areas in Communications, Special Issue on Copyright and Privacy Protection* 16 No.4 (1998) 474–481.

[18] Johnson, Neil F. and Jajodia, Sushil. “Steganography: Seeing the Unseen.” *IEEE Computer*, February 1998, pp.26–34.

[19] Johnson, Neil F. and Jajodia, Sushil. “Steganalysis of Images Created Using Current Steganography Software.” *Proceedings on Workshop on Information Hiding*, Portland, OR, April 1998. Also

published as *Notes in Computer Science*, vol.1525, Springer-Verlag, 1998.

[20] Brown, Andy. "S-Tools." Software downloadable from http://idea.sec.dsi.uimi.it/pub/security/crypt/code/s-tools3.zip

[21] Deus Ex Machina Communications (DEMCOM). "STEGANOS." Software downloadablefrom [22] Machado, Romana. "EZ Stego**."** Software dowloadable from http://www.stego.com/.

[23] Westfeld, A. and Pfitzmann A., "Attacks on Steganographic Systems", *Proc. 3rd Info. Hiding Workshop*, Dresden, Germany, September 28–October 1, 1999, pp. 61–75.

[24] Bhanu Prakash Battula,R SatyaPrasad, "Techniques in Computer Forensics : A Recovery Perceptive" ,May 2009.www.cscjournals.org/Journals/IJS/volume3/Issue2/IJS-13.pdf

BHANU PRAKASH BATTULA

Received Master's Engineering degree on Computer Science & Technology in 2008 from Acharya Nagarjuna University and also received another Master's degree on Computer Applications from Acharya Nagarjuna University. After Post graduation, He is working as a Lecturer in the Department of Computer Science and Engineering at Vignan's Nirula Institute of Technology and Science, Guntur, Andhra Pradesh. He published a paper [24] for International Journal. His research interests include Computer Security, Steganalysis and Image Processing.

R.SATYA PRASAD

Received PhD from Acharya Nagarjuna University in 2007. He is working as a Professor at Department of Science and Technology, Acharya Nagarjuna University. His research interests include Computer Security, Software reliability and Image processing.