STEGANALYSIS OF LSB ENCODING IN COLOR IMAGES

[a]*Jiri Fridrich,* [b]*Rui Du,* [b]*Meng Long*

[a]Center for Intelligent Systems, SUNY Binghamton, Binghamton, NY 13902-6000
[b]Dept. of Electrical Engineering, SUNY Binghamton, Binghamton, NY 13902-6000

ABSTRACT

In this paper, we analyze the security of Least Significant Bit (LSB) embedding for hiding messages in high-color-depth digital images. We introduce a powerful steganalytic technique that enables us to reliably detect the presence of a pseudo-random binary message randomly spread in a color image. We estimate the probability of both false detections and missing a secret message. The method is based on statistical analysis of the image colors in the RGB cube. It is shown that even for secret message capacities of 0.1 –0.3 bits per pixel, it is possible to achieve a high degree of detection reliability.

1. INTRODUCTION

Steganography is the art of secret communication. Its purpose is to hide the very presence of communication as opposed to cryptography whose goal is to make communication unintelligible to those who do not posses the right keys. By embedding a secret message into a carrier image, a stego-image is obtained. It is important that the stego-image does not contain any easily detectable artifacts due to message embedding that could be detected by a warden. One could utilize those artifacts to detect images that contain secret messages. Once this could be achieved, the steganographic tool becomes useless. The most important requirements for a secure steganographic method are the security (undetectability) and capacity. Obviously, the less information is embedded into the carrier image, the smaller the probability of introducing detectable artifacts by the embedding process. Finally, we emphasize that there is no formal definition of undetectability for steganography, and no formal theory of steganography has been put forward as yet. For more detailed discussions of this topic, see [1].

In this paper, we focus on a particular steganographic attack called stego-only attack. This is probably the most common type of an attack that will occur in the real world. The goal is to inspect one or possibly more images for statistical artifacts due to message embedding in color images using the LSB method. One would like to find out which images are likely to contain secret messages and also estimate the reliability of decisions (i.e., estimate the probability of failing to detect a suspicious image or, on the other hand, denoting an innocuous image as suspicious). As an example of a practical situation, we can take an Internet node checking every image with a special filter looking for suspicious images. In another situation, a forensic expert or secret agent may inspect image files on the hard disk of a seized computer, images posted on a particular web site, or images sent to a particular e-mail address, etc.

The general purpose of steganalysis is to collect sufficient statistical evidence about the presence of hidden messages in imagery. In this paper, we analyze the security of the most common steganographic technique – the LSB encoding in 24-bit color images. Today, this technique is used in many software products [2–4]. In Section 2, we present a new steganalytic technique based on analyzing the structure of the set of unique colors in the RGB color cube. In Section 3, we optimize the steganalytic technique by adjusting its parameters to minimize the probability of making an erroneous decision.

2. STEGANALYSIS OF LSB ENCODING

Johnson and Jajodia [5,6] 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 Westfield [7] 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 man 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 essentiall 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 ver 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 [2,3], 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 $|R_1-R_2| \le 1$, $|G_1-G_2| \le 1$, and $|B_1-B_2| \le 1$. This is equivalent to saying that $(R_1-R_2)^2+(G_1-G_2)^2+(B_1-B_2)^2 \le 3$. The number of all color pairs is

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

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 sufficientl 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.

3. 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 ($3MN$) to 50% (half of the color channels contain a message bit) and different sizes of the test message ($\alpha = 0.01 - 0.5$). The solid Gaussian peak $N(\mu,\sigma)$ with probability densit $f_{\mu,\sigma}$ (Figure 1) does not change with the message size – it corresponds to the probability densit function of the ratio R/R' for images without messages. The dashed Gaussian distribution $N(\mu(s),\sigma(s))$ with probabilit densit $f_{\mu(s),\sigma(s)}$ corresponds to images with messages and it changes with the secret message size s:

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

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

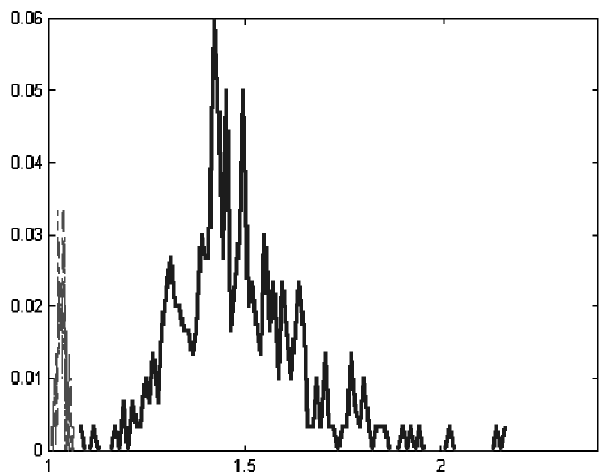

Figure 1 The ratio R'/R for 300 images. The thin dashed curve corresponds to images with an embedded message of length equal to 2/3 of the total available number of LSBs ($3MN$). The bold solid curve corresponds to images without any embedded messages.

We hav $\mu > \mu(s)$ 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

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

3.1 Threshold selectio

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 $3MN$.

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 10 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 1 bytes secret message

From the test results, we can draw the following conclusions

1. The probability of error prediction is mainly determined b 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 reliabl 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 losslessl 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.

4. ACKNOWLEDGEMENTS

Jiri Fridrich was supported by a subcontract entitled Steganograph – Detection and Recovery Toolkit (S-DART) sponsored by WetStone Technologies, Inc and the Air Force Research Laboratory under contract F30602-99-C-0210. Rui Du and Meng Long were supported by Air Force Research Laboratory, Air Force Material Command, USAF, under the contract F30602-00-1-0502.

5. REFERENCES

[1] 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.

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

[3] 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.

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

[5] Deus Ex Machina Communications (DEMCOM). “STEGANOS.” Software downloadable from http://www.steganography.com

[6] Machado, Romana. “EZ Stego.” Software dowloadable from http://www.stego.com/.

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