Sterilization of Stego-images through Histogram Normalization

Goutam Paul[1] **and Imon Mukherjee**[2]
[1]Dept. of Computer Science & Engineering,
Jadavpur University, Kolkata 700 032, India.
Email: goutam.paul@ieee.org
[2]Dept. of Computer Science & Engineering,
St. Thomas' College of Engineering & Technology, Kolkata 700 023, India.
Email: mukherjee.imon@gmail.com

Abstract— *Steganalysis is very popular in order to defeat the steganographic algorithm. But sometimes it is not possible to detect the hidden data without having any prior knowledge of the embedding algorithm or the knowledge of the key used. That is why image sterilization may play an important role in the field of secret communication to remove any steganographic information embedded in an image. In this paper, we propose a novel technique of image sterilization based on histogram normalization. The technique is general in the sense that it works for any LSB-based steganography algorithm and it does not need to know how the algorithm embeds information inside the image. We ran simulations over stego images created by different state-of-the-art algorithms and on average the technique succeeded in sterilizing around 77% to 91% of stego pixels depending on the algorithm targeted.*

Keywords: Information Hiding, Security, Histogram, Steganalysis, Steganography, Sterilization.

1. Introduction and Motivation

The word *Steganography* [11] originated from the Greek word "Steganos", meaning "covered information". It is the technique of hiding messages inside inoccuous media so that the hidden message cannot be detected by an adversary having access to the media. In [7], the history of steganography is traced from ancient Greece up to the modern times. It is reported that the Germans have successfully utilized this technology during the Second World War. Academic research in steganography has grown tremendously in last few decades and currently many steganographic algorithms exist in the literature.

Steganalysis [15], [2] is the art and science to defeat steganography. Steganalysis is performed without the prior knowledge of the steganographic algorithm or the secret key used for embedding the information into the cover media. This is why determining whether the secret message exists in the media is a difficult and challengable task.

Steganography can be applied to a variety of multimedia contents like images, audio, video, text etc. The media before embedding any secret information is called *cover* and after inserting the information is called *stego*. In this paper, we focus on steganography in digital image which is a very popular cover media for steganography.

Here we are proposing a new approach to sterilize the hidden information from the stego media based on image histogram. Our objective in this work is to develop an algorithm to revert as many stego pixels of an image as possible to their original cover form, which we call *image sterilization*. Image sterilization may have an important application in defense and security domain. For example, suppose that a spy wants to inform his team about the venue of performing a bomb blast in some target place using some image based steganographic technique. During the time of transmission, if sterilization of the stego information is performed by the security personals, then the attackers would be completely unaware about the venue and their plan may be jeopardized. One obvious method of performing sterilization would be to replace the least aignificant bits (LSBs) of all the pixel intensities by zero (or one). But this immediately gives a clue to the recipient that the image might has been modified by an adversary. Our algorithm does not leave any such signature and preserves the pseudo-randomness of the sequence of the LSBs in an image.

2. Steganographic Techniques in Spatial Domain

Before going into the details of our sterilization technique, we briefly describe here the major categories of steganographic algorithms in the spatial domain. Different algorithms operating in the spatial domain can be considered as different methods for selecting the pixel positions. These are classified into three categories: non-filtering algorithms, randomized algorithms and filtering algorithms [8].

2.1 Non-filtering Algorithm

The non-filtering steganographic algorithm [8] is the most popular and the most vulnerable steganographic technique based on LSB. The embedding process is done as a sequential substitution of each LSB of the pixel for each bit of the

message. Hence a large amount of information can be stored into the cover media.

The only requirement in this method is sequential LSB reading, starting from the first pixel in order to extract the secret message from the cover media (viz. image). As the message is embedded in the initial pixels of the image, leaving the remaining pixels unchanged, this technique gives an unbalanced distribution of the changed pixels.

2.2 Randomized Algorithm

This technique solves the limitation of the previous technique. Each of the sender and the receiver has a password denominated stego key which is generated through a pseudo-random number generator [8]. This creates an index sequence to denote the pixel positions. The message bit is embedded in the pixel of the cover media following the index sequence produced by the pseudo-random number generator.

The two main features of this technique are: **(i)** use of password to have access to the message and **(ii)** well-spread message bits over the image, that are difficult to detect compared to the previous case.

2.3 Filtering Algorithm

The filtering algorithm [8] filters the cover image by using a default filter and hides information in those areas that get a better rate. The filter is used to the most significant bits of every pixel, leaving the less significant bits to hide information. The filter gives the guarantee of a greater difficulty of detecting the presence of hidden messages. The retrieval of information is ensured because the bits used for filtering are not changed.

Each of the aforesaid three categories of steganographic techniques in spatial domain is susceptible to image sterilization described in subsequent sections.

3. Image Sterilization

A 24-bit color image [3] consists of a number of pixels and each pixel contains three intensity values (of 8 bits each), one for each of red, green and blue color components.

One of the most popular steganography techniques is the Least Significant Bit (LSB) insertion [13]. Since changing the LSB changes the pixel intensity value by at most 1 (see Table 1), if one changes the LSBs of some pixels, the resulting picture would be visually indistinguishable from the original image.

128	64	32	16	8	4	2	1
⇓							⇓
MSB							LSB

Table 1: Weights of different bit positions of the pixel intensity value.

Histogram of a digital image in spatial domain can be defined as:

$$H_{r_k} = n_k \tag{1}$$

where, r_k is the k^{th} intensity value and, n_k is the number of pixels with intensity r_k $[0 \leq r_k \leq 255]$. Now, consider Fig. 1 that shows the sample histogram of a portion of a stego image.

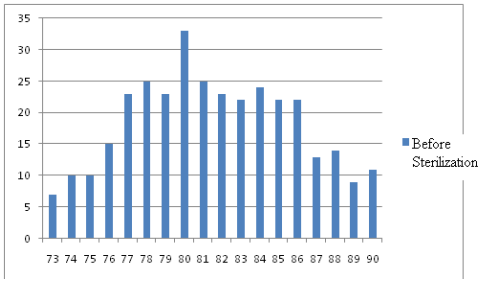


Fig. 1: A sample histogram of a portion of a stego image

We can normalize the histogram in order to remove the hidden information from stego-images based on following concept.

Table 2: LSB embedding in a pixel.

48	48	49	48	48	77	76	76	77
48	49	49	49	48	48	48	77	77
48	48	77	77	76	77	76	76	77
48	76	76	76	77	77	77	48	49
48	77	77	76	77	77	76	48	49
48	48	49	77	77	77	77	49	49
49	49	48	77	77	77	77	49	48
49	49	49	49	77	77	77	77	49

Consider Table 2 that shows some sample intensity values of one component (either blue, red or green) of an arbitrary image. There are two groups - one shown in red , another in blue. Suppose that the intensity of a pixel in the original image is 48. After stego insertion, the value may either remain as 48 (if 0 is inserted) or be changed to 49 (if 1 is inserted), as shown below in Table 3.

Table 3: Embedding message bit 1 into the LSB of a pixel with intensity value 48.

48	=	0	0	1	1	0	0	0	0
								⇓	
		0	0	1	1	0	0	0	1 = 49

The LSB flipping function [4] for a stego image is defined by $F_1 = 0 \leftrightarrow 1$, $2 \leftrightarrow 3$, $4 \leftrightarrow 5$, . . . , etc. We form groups of r_k values based on this flipping function, where the intensity values $2j$ and $2j + 1$, for $0 \leq j \leq 127$, belong to the

same group. So the maximum possible number of groups for each component of an image is 128. Suppose that an image contains N pixels with c groups. Let n_i be the number of pixels in the i^{th} group, $1 \leq i \leq c$. Thus, $N = \sum n_i$. The set of pixels (based on their intensity values) for the i^{th} group is represented by

$$G_i = \{x_{i,k} : 1 \leq k \leq n_i\},$$

such that

$$x_{i,j} - x_{i,m} \epsilon \{-1, 0, +1\}, 1 \leq j \neq m \leq n_i. \quad (2)$$

Fig. 2 shows the normalized version of histogram shown in Fig. 1 using Algorithm 1.

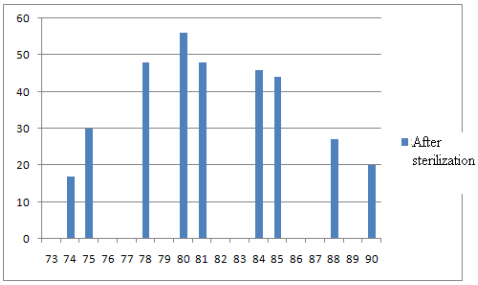


Fig. 2: Normalized version of the histogram shown in Fig. 1

In Algorithm 1, we present our procedure for image sterilization, called *StegoSterilize*. The basic idea behind our image sterilization technique is to replace an intensity value X with some other value Y such that one cannot extract the hidden message from the cover media. This technique can be applied to both 24-bit color images as well as 8-bit gray-scale images. Each group in the histogram contains at most two intensity values, of the form $2j$ (we call them *even* pixels) and $2j+1$ (we call them *odd* pixels). Let n_o and n_e be the number of even and odd pixels in a group. If $n_e \geq n_o$, we replace all $2j+1$ intensity values by $2j$, otherwise we do the opposite replacement. In other words, we force all pixels in a group to be either odd or even depending on the majority of the pixels being odd or even.

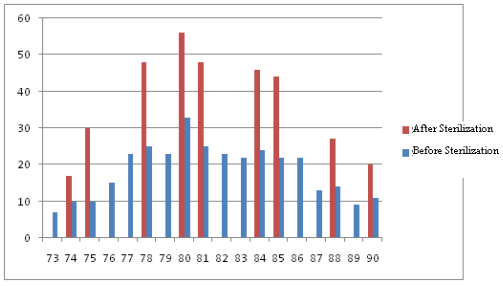


Fig. 3: Comparative representation of Histogram (before and after sterilization of the same sample)

Input: A stego image.
Output: The sterilized version of the input stego image.

Read the intensity values from the the stego image;
Draw the histogram of the stego image;
for *each color component* **do**
  Form the groups based on Equation (2);
  for *each group* **do**
    Count the odd and even pixels with intensity values of the form $2j+1$ and $2j$ respectively from the histogram of the stego image; Let n_o and n_e be the respective counts;
    if $n_e \geq n_o$ **then**
      Replace all $2j+1$ intensity values by $2j$;
    end
    else
      Replace all $2j$ intensity values by $2j+1$;
    end
  end
end
Output the transformed image;

Algorithm 1: StegoSterilize

Fig. 3 shows the changes in the histogram (blue color for before sterilization and red color for after sterilization)

4. Accuracy Measurement

To estimate the accuracy of our technique, we need to take as inputs some sample stego images for which we know which pixel values are actually changed due to the LSB embedding. For any stego image I, let

S := the number of stego pixels,
S' := the number of stego pixels (out of S) having different intensity value from their cover counterpart,
S'' := the number of recovered stego pixels (out of the S') due to the sterilization process.

The **accuracy of sterilization** for this image is defined as :

$$Acc_{steri}(I) = \frac{S''}{S'} \quad (3)$$

We have used a database of 200 (24-bit) color images in BMP format and 100 gray-scale images (freely available from [14] and many other internet sources). We have also prepared 50 different text files containing the story of

Sherlock home's (downloaded from[1]). Each pixel of a 24-bit color image contains three components, viz. red, green and blue. So using LSB embedding, at the most three bits of data can be embedded in a pixel. If the dimension of an image is $m \times n$, then maximum number of data bits possible to be inserted in the 24 bit color image can be $m \times n \times 3$. Since a character is of eight bits, the maximum number of characters (including white spaces) of the text would be $\lfloor \frac{m \times n \times 3}{8} \rfloor$. Thus, each of the 50 text files consists of at most $\lfloor \frac{m \times n \times 3}{8} \rfloor$ characters depending upon the values of the dimensions m and n. Similarly for gray-scale image, each text file should have $\lfloor \frac{m \times n}{8} \rfloor$ characters. We have used MATLAB 7.7.0 as a software tool for implementation.

Fig. 4 shows a stego (on the left) and the corresponding sterilized (on the right) versions of the famous image of Cameraman in gray-scale. Similarly, Fig. 5 shows the stego and sterilized versions of a 24-bit color image on the left and right respectively.

In Table 4, we give an example of how the text extracted from a sterilized image may differ from the original text that was embedded and is expected to be extracted from the unsterilized stego counter part. We observe that after sterilization, the actual message is scrambled enough so that it cannot be reliably recovered.

Table 4: Sample text embedded in an image before and after sterilization

I was the means of introducing to his notice that of Mr Hatherleys thumb and that of Colonel Warburtons madness.

Embeded Messege (taken from [12]) before Sterilization

⇓

J!ybtsgf!ofbmtÃşnmssnevcjnhp!gjtÃşmnshb dsgbtÃşnhÃşNs!GbsgfqmfytÃşsgvnb!bme!sh? uÃşnfÃşBnmpmfm!X?savqsnmtÃşnbdmftr

Embeded Messege (using algorithm 1) after Sterilization

Table 5 shows the ability to sterilize the stego image using the algorithm described in the previous section for three different embedding techniques. The first algorithm is the naive LSB based sequential embedding technique; we call it algorithm A. Another technique is taken from [10], which we refer as algorithm B. The third method, denoted by algorithm C, uses random pixel selection and segmentation mechanisms [12].

5. Other Performance Parameters

In addition to the accuracy, we compute some other performance measures as explained below.

5.1 Mean Squared Error (MSE) and Peak Signal to Noise Ratio (PSNR)

The imperceptibility of hidden information in a stego image is measured by the image quality in terms of

Table 5: Accuracy(minimum, maximum, average and standard deviation) of sterilization over hundred gray-scale and two hundred 24-bit color images for three different algorithms A,B,C.

		Grey scale	24 bit color image		
			R	G	B
Minimum %	A	72.50	68.01	68.9	69.75
	B	79.27	74.41	76.2	75.57
	C	N.A	84.29	84.36	85.89
Average%	**A**	**78.09**	**77.16**	**76.64**	**78.34**
	B	**79.31**	**81.20**	**80.35**	**81.68**
	C	**N.A**	**91.15**	**89.28**	**91.43**
Maximum %	A	87.74	90.85	91.01	90.02
	B	83.60	88.6	82.72	91.44
	C	N.A	96.12	95.35	96.07
Standard Deviation	A	0.0351	0.0622	0.0769	0.0729
	B	0.0222	0.0483	0.0226	0.0562
	C	N.A	0.0392	0.0336	0.0431

Mean Squared Error (MSE) and Peak-Signal-to-Noise Ratio (PSNR) in dB [9], [16]. Consider a discrete image $A(m, n)$, for $m = 1, 2, 3, \ldots, M$ and $n = 1, 2, 3, \ldots, N$, which is treated as a reference image. Consider a second image $B(m, n)$, of the same spatial dimension as $A(m, n)$, that is to be compared to the reference image.

The MSE is given by

$$MSE = \frac{1}{MN} \sum_{M,N} \left((A(m, n) - B(m, n) \right)^2 ,$$

where M and N are the number of rows and columns in the input images and PSNR is given by

$$PSNR = 10 \log_{10} \left(\frac{T^2}{MSE} \right) ,$$

where T is the maximum intensity value of all pixels. The MSE represents the cumulative squared error between the two images. The mean square error measure is very popular because it can correlate reasonably with subjective visual tests and it is mathematically tractable.

Lower MSE and higher PSNR imply that the difference between the original image and the test image is small, i.e., it is usually not possible to distinguish whether the image is a stego one or a sterilized one. In our experiments, we have obtained quite low MSE (0.4179 for gray-scale images and 0.1807 for color images) between cover and sterilized images. Similarly, the PSNR of cover and sterilized images are high (27.82 dB for gray-scale images and 35.12 dB for color images). Fig. 6 shows the MSE and PSNR of some selected imges. These results indicate that our technique is successful in hiding the fact that the image has been sterilized.

5.2 Histogram Analysis

The main purpose of histogram analysis [5] in our context is to detect significant changes in frequency of appearance

Stego version

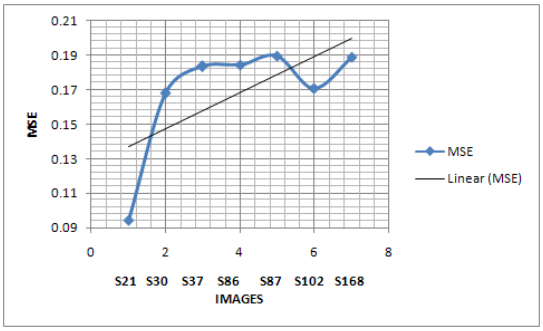
Sterilized version

Fig. 4: Stego and sterilized version of the gray-scale image (Cameraman.bmp)

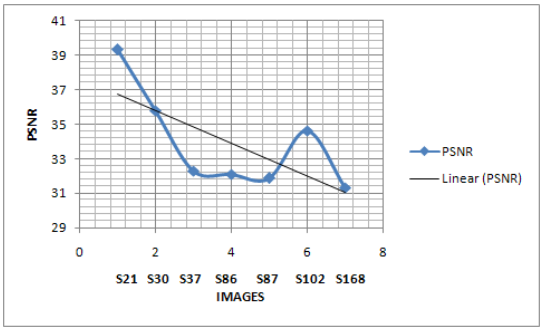
Stego version

Sterilized version

Fig. 5: Stego and sterilized version of a 24-bit color image



Mean Squared Error



Peak Signal to Noise Ratio

Fig. 6: MSE and PSNR of some selected images

of each color component in an image by comparing the cover version of the image with its stego and sterilized counterparts.

Fig. 7 shows the histograms of the Cameraman image in three stages: before stego insertion (on the left), after stego insertion (in the middle) and after sterilization (on the right). We see that our sterilization algorithm does not detectably distort the histogram of the input image.

6. Conclusion

In this paper, we have provided a novel concept of image sterilization. We have achieved on an average 77% to 91% success rate to sterilize the stego information of an image (the average rate varies with the steganography algorithm used to create the stego images). We would like to emphasize that the goal of our technique is not hidden message recovery, rather we aim at annihilating stego information transmission without distorting the image visibly. Our approach is generic and applied to any LSB based steganography algorithm. There has been some work [6] on double bit sterilization from the uncompressed image. Future directions may include multi-bit sterilization with the goal of sterilizing three or more bits in a pixel.

References

[1] http://221bakerstreet.org

[2] R. Chandramouli and K.P. Subbalakshmi, “Current Trends in Steganalysis: A Critical Survey”, Control, Automation, Robotics and Vision Conference, 2004, pages 964-967.

[3] http://www.digicamsoft.com/bmp/bmp.html

[4] J. Fridrich, M. Goljan and R. Du, “Reliable Detection of LSB Steganography in Color and Grayscale Images”, Proceedings of the 2001 workshop on Multimedia and security: new challenges, pages 27 -30

[5] R. C. Gonzalez and R. E. Woods, “Digital Image Processing”, Pearson Education, 2008.

[6] I. Mukherjee and G. Paul. “Double Bit Sterilization of Stego-images”, Accepted in Track: Security and Management, WORLDCOMP, July 18-21, 2011, vol.-1, pages 743-746, Las Vegas, U.S.A.

[7] N. F. Johnson, “Steganography”, Technical Report, November 1995. Available at http://www.jjtc.com/stegdoc

[8] S. Katzenbeisser and F. Petitcolas, “Information hiding techniques for steganography and digital watermarking”, Artech House Books, 1999.

[9] http://www.mathworks.com/access/helpdesk/help/toolbox/vipblks/ref /psnr.html

[10] J. Mielikainen, “LSB Matching Revisited”, IEEE Signal Processing Letters, vol. 13, no. 5, May 2006, pages 285-287.

[11] K. Nozaki, M. Niimi and Eiji Kawaguchi, “A large capacity steganography using color BMP images", Third Asian Conference on Computer Vision Hong Kong, China, January 8âĂŞ10, 1998, Proceedings, Volume I, pages 112-119, Lecture Notes In Computer Science, Vol. 1351/1997, Springer.

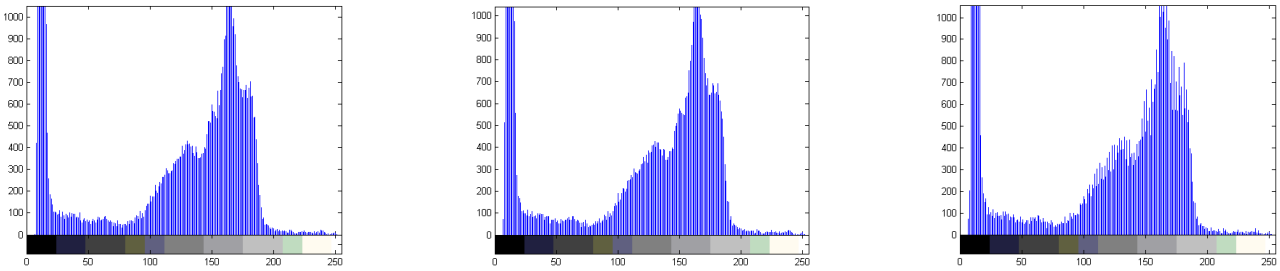
[12] R. Rana and Er. D. Singh, “Steganography-Concealing Messages in Images Using LSB Replacement Technique with Pre-Determined Random Pixel and Segmentation of Image”, International Journal of Computer Science & Communication, vol. 1, no. 2, July-December 2010, pages 113-116.

[13] J. J. Roque and J. M. Minguet, “SLSB: Improving the Steganographic Algorithm LSB”, Universidad Nacional de EducaciÃşn a Distancia (Spain) , Available at http://www.fing.edu.uy/inco/eventos/cibsi09/docs/Papers/CIBSI-Dia3-Sesion9(1).pdf

[14] www.webshots.com

[15] A. Westfeld and A. Pfitzmann,“Attacks on Steganographic Systems”, Proceedings of the Third International Workshop on Information Hiding 1999, pages 61-76, Lecture Notes In Computer Science, vol. 1768, Springer.

[16] http://en.wikipedia.org/wiki/Peak_signalto-noise_ratio



Cover version Stego version Sterilized version

Fig. 7: Histograms of Cameraman.bmp at different stages