IMAGE STEGANOGRAPHY TAILORED TO OBJECTS CONTOURS

Ovidiu **COSMA**
Technical University of Cluj-Napoca, North University Center of Baia Mare
ovidiu.cosma@yahoo.com

Keywords: image steganography, LSB substitution

Abstract*: This article proposes a steganography method that uses all three components of an image in the RGB color space to store secret data. The order in which the image pixels are processed is not given by their position within the image, but by their visual significance. In order to ensure the greatest possible embedding capacity, the image container is traversed in several passes, each expanding its capacity.*

1. INTRODUCTION

Steganography is a technique for data security that hides the secret information into a container that will unveil its true content only to the consignee. Text, audio, photos, and video files can be used as secret data containers. The image steganography techniques operate in the spatial domain or in the frequency domain. The secret data is called payload and the image with hidden data inside is known as stego image. Among the applications of steganography are: secret communication, copyright control, feature tagging and video-audio synchronization.

The key features of the steganography techniques are: robustness strength and capacity. Robustness refers to the resistance the payload against the processing operations applied to the container. In the case of image steganography, these operations could be geometric transformations, contrast, brightness or saturation adjustment, histogram correction, compression, cropping, blurring or detail enhancement, noise addition, etc. Strength indicates how difficult it is for anyone to guess that there is something else hidden inside the container. Capacity is the amount of hidden data that fits inside the container.

The art of breaking steganography is called steganalysis. For increasing the strength of steganography, the data embedding technique must preserve the perceptual quality as well as the statistical properties of the container.

2. IMAGE STEGANOGRAPHY

Digital images are composed of pixels, usually represented by 32 bits integers. Each pixel color is composed of four components of eight bits. The first component represents the pixel opacity and the following three represent the intensities of the Red Green Blue (RGB) components that simulate the pixel color.

One of the oldest digital image steganography techniques uses one or more of the least significant bits (LSBs) from the pixel RGB components for storing the secret data. The amount of secret data stored in each pixel, determines the visual quality of the stego image. The distortions caused by changing the pixel values are more visible in image areas without large amounts of details. As a consequence, if large payloads are needed, the uniform use of all image pixels is not recommended.

A technique of increasing the payload known as Bit Plane Complexity Segmentation (BPCS) is presented in [1]. The image is divided into 8 x 8 pixel blocks that are classified by complexity. Only the high complexity blocks will be used for placing the secret data.
Another steganography technique based on BPCS is presented in [2]. It differs from BPCS in the way the complexities of the image blocks are deterimned. A steganography technique that uses gray level images is presented in [3]. In [4] is presented a method that embeds secret data by altering the differences between pixels. It takes into account the complexities of the image regions to determine the proper amount of secret data to be stored in each of the pixels. A steganography technique appropriate for compressed images is presented in [5]. It is demonstrated by the authors to have high embedding capacity, and to allow for the perfect reconstruction of the original image.

In general, the spatial steganography techniques have the advantage of simplicity and high capacity, but they have the disadvantage of low robustness and strength. Several transform domain techniques have been proposed, for increasing the strength of image steganography. They apply an initial transformation in which the image is converted from the spatial domain to the frequency domain. Then the embedding of secret data is performed by altering some of the transform coefficients. There are several techniques to choose the coefficients that will keep the secret data. In [6] is presented a technique designed for JPEG compressed images [7]. The secret data is embedded by altering the LSB of the quantized Discrete Cosine Transform (DCT) coefficients that are different from 0, 1 and -1. Another steganographic technique for JPEG iamges is presented in [8]. It is different by the fact that the coefficients that will store the secret data are randomly selected. Another technique that

uses a genetic algorithm for selecting the best DCT coefficients to store the secret data is presented in [9]. The advantage of this method is the fact that it resists to almost all known steganalysis methods.

3. STEGANOGRAPHY DETECTION

The art of detecting steganography is called steganalysis. The only disadvantage of LSB steganography lays the fact that it is easily detectable. The LSB steganography principle stands on the assumption that the LSB of image pixels are random, and they can be replaced without creating suspicion. In [10] it is shown that this assumption is not correct, and in fact the pixels LSBs are correlated. Those correlations can be easily obserserved if only the LSB of each pixel is represented as a binary image. As a consequence the hidden data can be visually detected, because it disrupts those correlations.

In reality, such correlations do not occur in any image. In fact they rarely occur in the case of older images that are converted in one of the new formats for digital images. A statistical steganalysis method has been proposed in [10], for overcoming the limitations of the visual method. This method, known as the chi-square attack calculates a probability for the presence of hidden data depending on the length of the analyzed sample.

4. THE PROPOSED STEGANOGRAPHIC METHOD

The proposed method differs from the original LSB substitution method, by the fact that the secret data embedding process can involve one or mode passes through the whole image, depending on the amount of secret data. All the three components (RGB) of the image are used. The last significant bits (LSBs) of the RGB components are not used, because there are numerous steganalysis methods that verify their distribution. The next three bits may be used for secret data embedding. The image pixels are processed in an order determined by a contour detection filter. The secret data is passed through an encryption algorithm, which ensures an extra level of security while increasing the strength of the method.

The operations of the proposed method are presented in *figure 1*. The first block shifts the values of the RGB components of each image pixel with four positions to the right. Thus the last four bits of each component are lost. This step ensures that the following operations will not take into account the stego data, and will be accurately reproduced at decoding. The second block applies a contour detection filter. In principle, any such filter may be used. The Sobel filter [11] was used in the experiments. This processing produces for each pixel a value that reflects the importance of the contour on which the pixel stands.

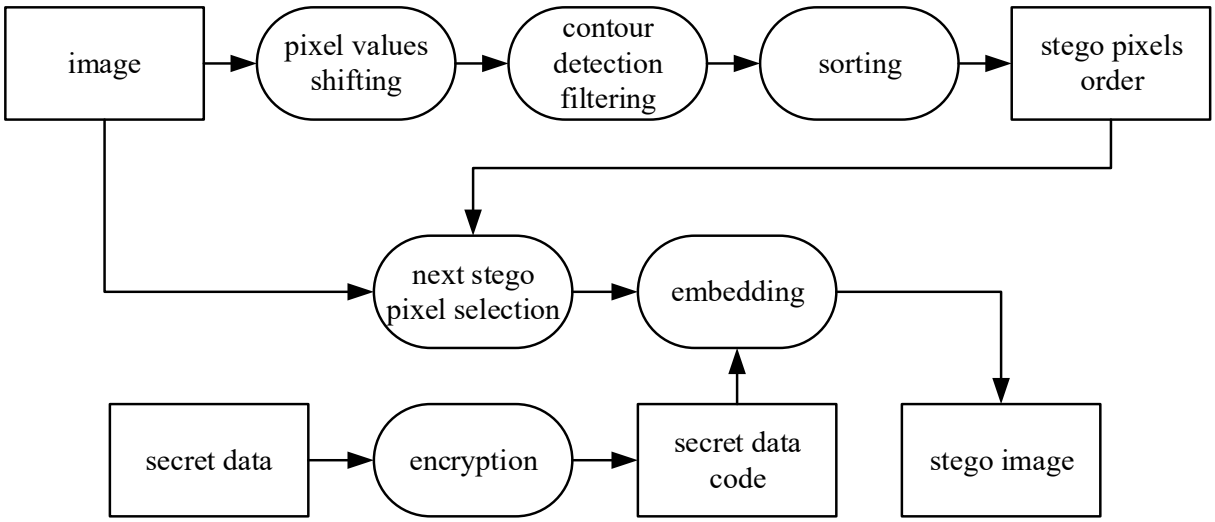

Fig. 1. Operation of the proposed method

For pixels in uniform regions, free of color variations, the value 0 is obtained. These pixels can not be altered too much in the embedding operation, as there is a danger that changes will become visible. Pixels on the contours of objects can be modified to a greater extent without causing noticeable distortions. The next block performs the sorting of image pixels by the values obtained in the filtering step. This step determines the order in which image pixels will be processed in the secret data embedding operation. The process begins with the pixels on the most important contours and ends with the pixels in areas lacking details. The last bit of the RGB components will not be used in the embedding process. At the first pass, the second of the LSBs will be used for embedding the secret data. Since these bits are relatively unimportant, all image pixels can be used in this pass, without the danger of creating noticeable artefacts. Thus, at the first pass, the capacity of the container can be calculated as follows:

```
imageWidth * imageHeight * 3 [bits]
```

This initial capacity can be expanded if needed by the following passes. In case of massive secret data, such as digital images, the embedding operation may require several passes. In each of the following passes, the pixels of the image will be processed in the order given by the sorting block, starting with those on the most important contours.

At the second pass, the third of the LSBs will be used. Experiments have shown that no detectable distortions can occur in this pass, but for extra security, the pixels in areas lacking details will not be used for embedding the secret data. Thus, the second pass ends when pixels are reached for which the value calculated by the contour detection filtering block is below a certain threshold (*t1*). The optimal value of *t1* depends on the type of contour filter used. The amount by which container capacity can be expanded in the second pass depends not only on the size of the image, but also on its content. For example, for a completely white picture, there are no available pixels in the second pass.

If the secret data embedding process does not end in the second pass, then the operation continues with a new pass in which the fourth of the LSBs will be used. At this stage, the pixels that will be changed should be chosen with even greater caution. As in the previous pass, the process starts with the pixels that are on the most important contours, and ends when the value calculated by the contour detection filtering block is below a certain threshold (*t2*), where *t2 > t1*. As with *t1*, the optimal value of *t2* depends on the type of contour filter used.

The differences between the proposed method and the other known methods are as follows: the image container is processed in several passes, it does not use image segmentation based on complexity, the image pixels are sorted based on an edge detection filter, the last bit of the three RGB color components is left unchanged.

5. EXPERIMENTAL RESULTS

The digital image container which was used for presenting the results of the proposed method is shown in *figure 2*. It has a resolution of 512 x 512 pixels and occupies 512 x 512 x 3 / 1024 = 768 kB of memory. This container was filled up to full capacity with random data. Thus, in the first pass, all pixels of the image were used, resulting a capacity of 96 kB. This capacity was expanded in the second and third passes with 59 kB and 29 kB respectively, resulting a total capacity of 184 kB, that is more than enough to keep inside a payload of 9 JPEG images of good quality and having the same size as the container image. The capacity of the image container is 184 / 768 *100 ≈ 24%. Of all the staganography techniques presented in paragraph 2, BPCS [1], [2] has the highest embedding capacity (approximately 50%).

Fig. 2. Original image

Figures 3 and *4* show the stego pixels used in the second and in the third pass. Even if the entire capacity of the container has been used, there are no apparent differences between the original and the stego images. The PNG format has been used to save the stego image because it is a lossless format. The size of the stego image file is 440 KB.

One of the most powerful steganography techniques is the genetic algorithm approach [9]. It is known to defeat all the steganalysis techniques. The proposed method has good strength, combined with larger capacity. Even if the entire capacity of the container is used, the chi-square attack [10] does not reveal anything, because the last bit of the pixels components was not used in the embedding process. *Figure 5* presents the results of the chi-square attack applied on the stego image filled up to full capacity with random data.

Fig. 3. Stego pixels used in the second pass

Fig. 4. Stego pixels used in the third pass

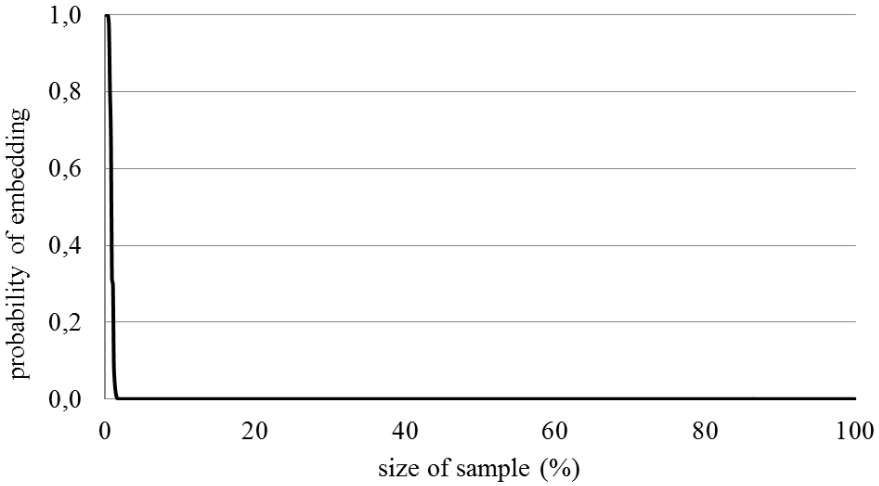

Fig. 5. The result of the chi-square attack, applied on the stego image

5. CONCLUSIONS

The image steganography method presented in this article uses all three components of an image in the RGB color space to store the payload, and processes the image in multiple passes, to ensure a hingh embedding capacity. The embedding capacity of the proposed method is not constant at a given image resolution. It depends on the amount of details in the image container.

For increasing the strength of the method, at each pass the image pixels are processed in an order given by their visual significance. The method is not vulnerable to the chi-square attack, because the last bit of the three color components (RBG) is not used in the embedding process.

The proposed method is not robust. Any processing performed on the image container could destroy its payload. But only watermarking needs robustness, steganography does not.

REFERENCES

[1] M. Niimi, H. Noda, E. Kawaguchi, *A Steganography Based on Region Segmentation by Using Complexity Measure*, Trans. of IEICE, Vol. J81-D-II, No. 6, 1998.

[2] H. Hioki, *A data embedding method using BPCS principle with new complexity measures*, Proceedings of Pacific Rim Workshop on Digital Steganography, 2002.

[3] V.M. Potdar, E. Chang, *Gray level modification steganography for secret communication*, Proc. of 2nd IEEE International Conference on Industrial Informatics, pp. 223-228, 2004.

[4] D. Wu, W.H. Tsai, *A steganographic method for images by pixel value differencing*, Pattern Recognit. Lett. 24, pp.1613-1626, 2003.

[5] H.C. Wu, H.C. Wang, C.S. Tsai, C.M. Wang, *Reversible image steganographic scheme via*

predictive coding, Displays, no.31, pp. 31-43, 2010.

[6] D. Upham, *Jsteg*, http://zooid.org/ paul/crypto/jsteg/ (accessed: 2017-05).

[7] Gregory K. Wallace, *The JPEG still picture compression standard*, Communications of the ACM, vol. 34(4), pp.30-44, April 1991.

[8] A. Latham, *JPHIDE*, http://linux01.gwdg.de/ alatham/stego.html (accessed: 2017-05).

[9] A. Milani, A. Mohammad, A. Varasteh, *A New Genetic Algorithm Approach for Secure JPEG Steganography*, IEEE International Conference on Engineering of Intelligent Systems, 2006.

[10] A. Westfeld, A. Pfitzmann, *Attacks on Steganographic Systems Breaking the Steganographic Utilities EzStego, Jsteg, Steganos, and S-Toolsand Some Lessons Learned*, https://users.ece.cmu.edu/ adrian/487-s06/westfeldpfitzmann- ihw99.pdf (accessed: 2017-05).

[11] Irwin Sobel, *History and definition of the so-called Sobel Operator*, https://www.researchgate.net/publication/239398674_An_Isotropic_3_3_Image_Gradient_Operator (accessed 2017-05).