БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ • BULGARIAN ACADEMY OF SCIENCES

ПРОБЛЕМИ НА ТЕХНИЧЕСКАТА КИБЕРНЕТИКА И РОБОТИКАТА, **66**
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS, **66**

София • 2015 • Sofia

Reversible Data Hiding Using a Histogram Modification

Nikodim Lazarov, Zlatolilija Ilcheva

Institute of Information and Communication Technologies, 1113 Sofia
Emails: nlazarov@gmail.com zlat@isdip.bas.bg

***Abstract**: The purpose of data-hiding and watermarking is to embed information in visual and audible resources by changing the data they are carrying in such a way that it can be extracted later. The special case of lossless data-hiding is the process of embedding this information in such manner that after successful extraction the host content is recovered to its original. This paper focuses on presenting such method based on images. It is implemented in the spatial domain and utilizes the histogram information as a base. The method also suggests a modification to the histogram construction algorithm so that more information can be embedded. Furthermore, a way of preventing overflow and underflow have been developed so that there are no cases of lost information. Finally experimental results have been compiled using this approach in order to support the theoretical assumptions.*

***Keywords:** Data-hiding, watermarking, histogram construction algorithm.*

1. Introduction

Data-hiding algorithms and watermarking image processors have been established as tools of improving the security of visual information, empowering secure communication channels, authenticating ownership, etc. Most of these approaches result in losses of data in the cover content after encoding and decoding. In these cases the cover content is considered just as a carrier of the message and it is of no importance if the carrying image is restored to the initial state or not. Nevertheless, there are areas where the requirement of recovering the original image, i.e.,

reversible data-hiding, is of great importance, for example in medical images, crime evidence, military intelligence data, etc.

Since lossless embedding in images is becoming more and more important, there are various methods already implemented that address this area [1-6]. The approach described in this paper is inspired by the Ni's Reversible Data Hiding algorithm [1].

The method of Ni uses certain histogram properties to embed data. More precisely it focuses on the peak point of the histogram, i.e., the pixels that have the most common pixel value (Fig. 1). They are the perfect candidate for storing the embedding data, since they are the largest set regarding a common property – the value of the pixel. The algorithm, from a histogram point of view, frees the pixel value after the peak point by increasing all pixel values greater than the peak point. After the image is ready for data embedding through this simple histogram operation, the system iterates through the image pixel by pixel. When a histogram peak value pixel is met, the pending bit from the message bit stream is embedded. If it is 1, the pixel value is increased by 1, otherwise it remains the same.

The extraction process is basically reverse of the embedding. The image is iterated in the same way. When a pixel with the peak point value is met or one that has the peak value plus 1, the respective bit – 0 or 1, is added to the extracted message bit stream. During this process, if the pixel value is greater that the peak pixel value, the value is decremented in order to restore the original pixel value.

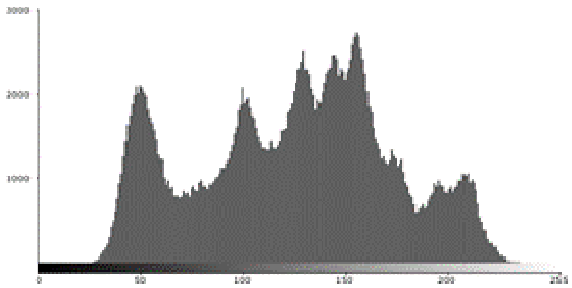

Fig. 1. Histogram of Lena image

2. Proposed technique

What we expect to construct is a reversible method of data embedding with higher capacity than Ni's approach. At the same time the PSNR (Peak Signal-to-Noise Ratio) is relatively high so that this approach shows very good results with making the visual perception of change almost unnoticeable.

Before we make the description of the proposed approach, a small clarification needs to the made. The following explanations and experimental results are done using greyscale images only. This does not mean that the algorithm will not work on colored images. Quite the contrary, it can be done on a single color component with the same efficiency. Furthermore multiple color channels can be used thus increasing the maximum embeddable data capacity.

The algorithm proposed advances through the following stages – first one needs to use a modified approach to construct a histogram so that the correlation of the neighboring pixels can be utilized. Hence, this method is not suitable for an image based on random white noise, where the pixel values are independent and uncorrelated. The embedding process is similar to Ni's approach as the histogram value after the peak point is freed by incrementing the difference between pixels with the peak difference plus 1. This difference value will store the 1s from the message binary stream.

2.1. Modified histogram

The main purpose of the proposed algorithm of constructing a histogram is to come up with a higher maximum value, thus allowing more information to be embedded in the image. In order to do this one needs to take full advantage of the spacial characteristics of the pixels. In images, that capture landscapes, human portraits, or any visual particular form, there is one thing that can be exploited for our advantage. That is the high correlation between pixels, the regions with pixels that are close both spatially and in value. Therefore, when constructing the histogram, the following modifications can be implemented:

- instead of capturing the pixel value, one should capture the difference between adjacent pixels;
- in doing the above, one should represent the two-dimensional pixel matrix in a single dimension, i.e., an array of pixel values. Thus the resultant values for the histogram will be the difference between the current value and the previous in the row.
- the first element of the resultant array will be the first pixel value of the unfolded image. It will stay as a reference point for the following difference manipulations.

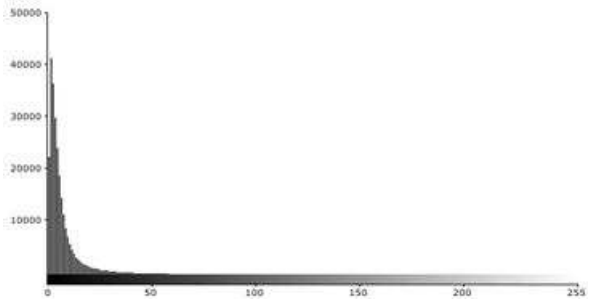

Fig. 2. Modified histogram of Lena image using the difference between adjacent pixels

Following the rules above described, the modified histogram that relies on the difference in pixel values usually results in very high maximum values around the start of the histogram at the 0, 1, 2 values (Fig. 2). This means that the default case is that the difference between adjacent pixels is low.

Basically the histogram so constructed will designate not the number of pixels with an equal value, but rather the number of adjacent pixel tuples with equal difference between them. As a result, this histogram is another statistical characteristic of the image describing the correlation between neighboring pixels. The usual case is that it offers much higher peak points than the regular histogram, which phenomenon can be exploited by the proposed data-hiding algorithm.

Example 1 □ Creating the histogram

The following example illustrates the workflow of collecting the values for the modified histogram.

Let's assume that the pixel values have been ordered in a one-dimensional array – a vector. Let a part of this vector be:

29	28	27	26	60	62	65	65	64	65	66	69	70	71	73	73

where these values denote the level of gray.

Following the rules designed for creating the modified histogram, i.e., calculating the absolute difference between the adjacent values, the result will be:

29	1	1	1	34	2	3	0	1	1	1	3	1	1	2	0

Obviously the most frequent difference is 1. Therefore this will be our δ_{max} in the embedding and extracting process.

2.2. Embedding

We assume that one has already mapped the 2D image pixel matrix to a 1D vector and has constructed the modified histogram based on the adjacent pixel differences. Let us denote the peak point of the histogram thus constructed, i.e., the most frequent difference, as δ_{max}. In the case of the Lena image δ_{max} is 1. The information is embedded one bit at a time in the following manner:

1. The first pixel from the array keeps its value.
2. The second and following pixels calculate the difference between that pixel and the previous pixel from the array. Let us denote this by δ.
3. if δ is equal to δ_{max}, i.e., this is an embeddable pixel, then increase the difference by the corresponding bit value from the data stack. This means add 0 or 1 to the current pixel value, if the next pending bit from the message is 0 or 1.
4. if δ is greater than δ_{max}, then simply increase the difference by 1, so that this value does not become a part of the embedding data pixels, i.e., free the (δ_{max} + 1) value for embedding purposes.

If we are to denote the pixel value of the i-th pixel as p_i, the embedded value as e_i, the difference between the current and previous pixel as $\delta_i=|p_i - p_{i-1}|$ and the bit to be embedded as b, then the rules above can be summarized in the following:

$$e_i = \begin{cases} p_i, if\ i = 0 \\ p_i + b, if\ \delta_i = \delta_{max}\ and\ p_i \geq p_{i-1} \\ p_i - b, if\ \delta_i = \delta_{max}\ and\ p_i < p_{i-1} \\ p_i + 1, if\ \delta_i > \delta_{max}\ and\ p_i \geq p_{i-1} \\ p_i - 1, if\ \delta_i > \delta_{max}\ and\ p_i < p_{i-1} \end{cases}$$

Thus the embedding process is completed. One should note that this process focuses on one peak point. Using multiple peak points results in higher data-hiding capacity and can be achieved by multiple applications of the embedding process.

Example 2 □ Embedding

The capacity of the provided sample is 8 bits, i.e., there are 8 differences equal to the delta-max. Let us assume that the message to embed is 10011101.

29	28	27	26	60	62	65	65	64	65	66	69	70	71	73	73
-	1	0	0	-	-	-	-	1	1	1	-	0	1	-	-
29	27	27	26	61	63	66	65	63	66	67	70	70	72	74	73

The embedding procedure starts with the first value and it remains the same. Since the next position corresponds to δ_{max}, the first bit will be embedded. As 28 is less that 29, the difference should be extended and the embedded value will be 27. Moving to the next position we stumble upon another embedding position. The second bit is 0, therefore the value remains the same – 27. Following the same pattern, the result is the same as denoted in the third row. The second row shows the positions of embedding the message.

2.3. Extraction

The extraction process and the recovery of the original image is almost as straight-forward as the embedding. Once again the image should be iterated through in the same way it was when embedding, and the resultant one-dimensional array should be used to extract the hidden data and restore the image. Here are the steps to take:

1. The first pixel value is the same, as it was not changed during the embedding process.
2. Calculate the difference between the current pixel value and the previous one. This value is important for the following processing:
3. if the difference is equal to the δ_{max} value, then write a 0 bit in the output stream. The pixel value should not be changed
4. if the difference is equal to δ_{max} + 1, then write a 1 bit in the output stream. After that change the pixel value for the pixel is such that the difference is decreased by 1.
5. if the difference is greater than δ_{max} + 1, then change the pixel value so that the difference is decreased by 1.

The extracting of the hidden data should look like this (again let's denote $\delta_i = |e_i - p_{i-1}|$):

$$b = \begin{cases} 0, if\ \delta_i = \delta_{max} \\ 1, if\ \delta_i = \delta_{max} + 1 \end{cases}$$

Restoring the original image could be summarized in the following manner:

$$p_i = \begin{cases} e_i + 1, if\ \delta_i \geq \delta_{max}\ and\ e_i < p_{i-1} \\ e_i - 1, if\ \delta_i \geq \delta_{max}\ and\ e_i \geq p_{i-1} \\ e_i, else \end{cases}$$

As a result of the extraction process we should end up with the hidden data decoded and the original image restored.

Example 3 □ Extraction

29	27	27	26	61	63	66	65	63	66	67	70	70	72	74	73
-	1	0	0	-	-	-	-	1	1	1	-	0	1	-	-
29	28	27	26	60	62	65	65	64	65	66	69	70	71	73	73

In the extraction process again the first value remains the same. After that since the difference between the next position is –2, which is delta-max + 1, a bit value of 1 should be extracted. The recovered value should be 27+1, since 29>27. Next – at the third position 27–28 = 1, which is equal to delta-max, therefore 0 is extracted. The recovered value is 27, since the difference is not more than delta-max. Following 27–26 = 1 = delta-max, another 0 is extracted. The recovered value is 26. Furthermore, 61 – 26 = 35. Since 35 is more than delta-max+1, no extraction should occur, but the recovered value should be 61 – 1 = 60, as the difference is positive. After the extraction process the result should be the extracted message above and the recovered image as shown in Example 3.

2.4. Prevent overflow

The arithmetic operations on the pixel values during emedding, i.e., the addition/substitution of 1, could result in an overflow at edge values like 0 and 255. As the values –1 and 256 are not valid in a greyscale image, they should be avoided at any cost or loss of information may occur, which in the case of reversible data-hiding is unacceptable.

Because of this, a procedure needs to be developed so that such over(under)flow could be prevented. Such an approach could be to shrink the histogram at the edges so that there is no way to reach the illegal values.

The process of histogram shifting is two sided – first we need to shift the histogram, so that the values could be safe and after the decoding there should be a reverse operation, where the original values should be restored. The first step in accomplishing this is to decide how much of the histogram edge values should be shifted. In our case this must be 1, which will result in very few changes to the

original image, if it does not have pure white and/or black. A location map should keep track which pixels have been changed. This could be a vector with 0s and 1s where 0 indicates no change and 1 indicates a pixel that must be (has been) changed. This array must be transferred as overhead information embedded in the image. It can be shrinked in size using the run-length encoding algorithm as there are only 2 symbols in use. In such circumstances this approach shows great compression levels.

In the decoding phase, after the information has been extracted and the starting image has been recovered, the reverse histogram shift must be executed using the location map. Once the reverse shifter is at a modified location, the pixel should be changed according to its value. If it corresponds to the lower bound, then its value should be subtracted by 1, otherwise if the value is at the upper bound, the value should be increased by 1.

3. Experimental results

In order to get better understanding of the different aspects of the performance of the embedding algorithm, tests were run on six images together (Fig. 3).

Fig. 3: Test images used for evaluating the experimental results

The results are presented in Table 1. The table consist of 7 columns, where the first column designates the images, the second column is the magnitude of the peak point – the max value. Third follows the overhead information (in bits) needed for the algorithm to handle the over(under)flow. Following is the pure information that can be stored in the image in bits. After that is the peak signal-to-noise ratio (PSNR) that is calculated using the following formula:

$$PSNR(dB) = 10 * log_{10}\left(\frac{255^2}{MSE}\right)$$

where MSE is the mean-square error.

Another measurement of the embedding capacity is the density of the embedded data. A common approach for image cover works is to calculate this quantity in bits per pixel (BPP), i.e., the number of embedded bits per the total count of pixels. Max BPP designates the maximum amount of embeddable bits per pixel for the specified image.

Table. 1 Experimental results

image	max value	Overhead	pure	PSNR	BPP	max BPP
bee	136668	16	136652	51.1	0.034	0.521
cat	117047	16	117031	50.5	0.034	0.446
jet	63006	16	62990	50.1	0.034	0.240
lake	31174	16	31158	48.9	0.034	0.118
lena	41264	16	41248	49.2	0.034	0.157
mandrill	16462	296	16166	48.4	0.034	0.061

The test results confirm that images with highly correlated pixels that are large sequences with similar pixel values, like a bee and a cat, result in higher embedding capacity, whereas mandril and lake show significantly lower values. Also mandril image is the one showing high values of overhead information, because edge values in it are to be found.

The distortion in the marked images seems to be at low levels, this is usually due to many pixels being untouched. A side-by-side comparison of the original and the marked lena images follows (Fig. 4).

At the same time it offers great performance as the embedding time is around 100 ms for almost any image (512x512) through the process. This means that the algorithm is predictable and fast.

Fig. 4. Original (left) and Marked (right) Lena image with near maximum bits per pixel embedding density □□0.15 bpp

4. Conclusion

The suggested method of information embedding in a grey-scale image offers great embedding capacity. Still the variance in the BPP (bits per pixel) among the test images shows that this parameter is highly dependent on the image and on the inter-pixel correlation respectively. At the same time using this approach to embed data proves to offer great fidelity as far as the MSE measurement method is concerned.

Moreover, the computation performance is great, which makes it very suitable for application where performance is of great importance.

An area for further research and improvements could be finding better space-filling curves for constructing the modified histogram and running the embedding/extracting processes. The benefit of a more optimized space-filling curve could be increased embedding capacity. Using a pseudo-random space-filling curve could reduce the detecting of the data-hiding process, i.e., determining whether and what the message is.

References

1. Ni, Z., Y. Q. Shi, N. Ansar, W. Su. Reversible Data Hiding. – In: IEEE Transactions on Circuits and Systems for Video Technology, Vol. **16**, March 2006, No 3, 354-357.
2. Chang, C. C., T. C. Lu. Lossless Information Hiding Scheme Based on Neighboring Correlation. –International Journal of Signal Processing, Image Processing and Pattern, Vol. **2**, March 2009, No 1, 49-52.
3. Stach, J., A. M. Alattar. A High-Capacity, Invertible, Data-Hiding Algorithm Using a Generalized, Reversible, Integer Transform. Digimarc Corporation, Tualatin, OR 97062, 1-3.
4. Jinna, S. K., L. Ganesan. Lossless Image Watermarking using Lifting Wavelet Transform. – International Journal of Recent Trends in Engineering, Vol. **2**, November 2009, No 1, 191 p.
5. De Vleeschouwer, C., J. F. Delaigle, B. Macq. Circular Interpretation of Bijective Transformations in Lossless Watermarking for Media Asset Management. – In: IEEE Transactions on Multimedia, Vol. **5**, March 2003, No 1, 97-100.
6. Chang, C. C., W. Tai, K. N. Chen. Lossless Data Hiding Based on Histogram Modification for Image Authentication. – In: 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, 506-509.
7. Radharani, S., M. L. Valarmathi. A Study on Watermarking Schemes for Image Authentication. – International Journal of Computer Applications (0975 – 8887), Vol. **2**, June 2010, No. 4, 25-27.

Обратимое встраивание данных с использованием модификации гистограммы

Н. Лазаров, Златолилия Илчева, Валери Илчев

Институт информационных и коммуникационных технологий, 1113 София
Emails: nlazarov@gmail.com zlat@isclip.bas.bg

Р е з ю м е

Цель встраивания данных и водяных знаков в мультимедийных файлах является изменением этих файлов таким образом, что данные и водяные знаки возможно было бы извлечь позже. Специальный случай встраивания данных без потери является процессом внедрения информации так, что после ее успешного извлечения мультимедийный файл восстанавливается вполне до своего оригинала. В статье рассматривается метод, работающий с изображениями. Метод оперирует в пространственной области и предлагает модификацию гистограммы несущего изображения таким образом, что встроить максимальное количество информации в нем. Также предложен способ для предохранения от потери информации в ходе вычислений. Экспериментальные результаты, представленные в конце статьи подтверждают теоретические предположения.