Annales UMCS Informatica AI 8(1) (2008) 55-65
10.2478/v10065-008-0006-4

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI

http://www.annales.umcs.lublin.pl/

Image processing algorithms employing two-dimensional Karhunen-Loeve Transform

Paweł Forczmański*

Institute of Computer Graphics and Multimedia Systems, Faculty of Computer Science and Information Systems, Szczecin University of Technology, Żołnierska 49, 71-210 Szczecin, Poland

Abstract

In the fields of image processing and pattern recognition there is an important problem of acquiring, gathering, storing and processing large volumes of data. The most frequently used solution making these data reduced is a compression, which in many cases leads also to the speeding-up further computations. One of the most frequently employed approaches is an image handling by means of Principal Component Analysis and Karhunen-Loeve Transform, which are well known statistical tools used in many areas of applied science. Their main property is the possibility of reducing the volume of data required for its optimal representation while preserving its specific characteristics.

The paper presents selected image processing algorithms such as compression, scrambling (coding) and information embedding (steganography) and their realizations employing the two-dimensional Karhunen-Loeve Transform (2DKLT), which is superior to the standard, one-dimensional KLT since it represents images respecting their spatial properties. The principles of KLT and 2DKLT as well as sample implementations and experiments performed on the standard benchmark datasets are presented. The results show that the 2DKLT employed in the above applications gives obvious advantages in comparison to certain standard algorithms, such as DCT, FFT and wavelets.

1. Introduction

The problem of image processing and recognition involves gathering, processing and storing large amounts of data. Reducing very high-dimensional feature spaces leads to speeding-up the computations. Many dimensionality reduction methods have been proposed in literature, many of them based on transformation such as Karhunen-Loeve Transform (KLT), Discrete Cosine Transform (DCT), Discrete Fourier Transform (DFT), Linear Discriminant Analysis (LDA), Discrete Wavelet Transform (DWT), etc. The literature survey

* *e-mail address*: pforczmanski@wi.ps.pl

shows that in the case of real images one of the most promising is KLT since it uses basis functions optimally adjusted to their actual characteristic. The KLT also known as Principal Components Analysis (PCA) is a technique widely used to reduce multidimensional data sets to lower dimensions for analysis, compression or classification [1]. The PCA involves the computation of eigenvalue decomposition or singular value decomposition of a data set, usually after mean centering [2]. However, it should be noted that using KLT for tasks such as pattern recognition or image processing can be challenging because it treats data as one-dimensional, when in fact they are two-dimensional. That is why almost all developed algorithms employ some sort of dimensionality pre-reduction discarding, in many cases, the spatial relations between pixels. One of the possible solutions is using two-dimensional transformation based on PCA (and KLT). The first algorithm in this group was presented in [3], where a novel, two-dimensional version of KLT for the face recognition task was developed. An extension of this method (known as PCArc – for row and column representation) was presented in [4,5]. Many previous publications show that the two-dimensional KLT (2DKLT) can be directly applied for high-dimensional data [4-7] because it does not require any preliminary processing or additional reduction of dimensionality of input images.

In the further parts of the paper we show the main principles of 2DKLT together with the three application areas: image compression, image scrambling and information embedding. All the experiments have been performed on the standard benchmark images. The image "Barbara" (8-bit gray-scale, 640×480 pixels) shown in Fig. 1 is employed in all cases as an example, since it contains areas of different nature (i.e. smooth, detailed, edges, etc.) and a human face, on which any unusual artifacts can be perfectly observed.

Fig. 1. Test image ("Barbara") used in experiments

2. Principles of 2DKLT

Let us assume that the image O being processed is stored in a matrix of $P \times Q$ elements, containing single color or intensity channel only. We treat it as a set of K subimages (blocks) X_k with equal, constant dimensions $M \times N$ ($M << P$, $N << Q$). These blocks do not have common parts and do not overlap each other. They cover the whole image area. This approach is similar to JPEG/JFIF, where $M=N=8$, which is a compromise between compression ratio and performance [8]. In the case being discussed here, it is possible to use block of any size, even not square [4]. The algorithm uses PCA for row/column representation (PCArc) in order to calculate the eigenvectors and KLT in order to transform them [5].

In the first step, for all blocks in the dataset we calculate a global mean element $\bar{X}$ and remove it from each single block:

$$\hat{X}_k = X_k - \bar{X}, \ \forall k = 1,\ldots,K . \tag{1}$$

The mean element can be calculated as one of the following:
a) as a matrix being a quotient of a sum of all blocks and their number:

$$\bar{X} = \frac{1}{K}\sum_{k=1}^{K} X_k , \tag{2}$$

b) as a mean scalar value of a whole image intensity:

$$\bar{X} = \frac{1}{PQ}\sum_{p=1}^{P}\sum_{q=1}^{Q} O(p,q), \tag{3}$$

c) or as a constant value equal to zero: $\bar{X} = 0$.

The appropriate choice of the mean element has an important influence on the further steps and the quality and efficiency of a specific algorithm. Investigations on the compression algorithm showed that the highest compression rate is provided by the formula (a), while the formula (c) gives the highest quality. The formula (b) is a compromise giving intermediate results both in compression rate and reconstruction quality.

In the next step we calculate two covariance matrices for both row and column representations of input images, which correspond to the variation of blocks in the set [3]:

$$\Sigma^{(R)} = \sum_{k=1}^{K} \hat{X}_k \left(\hat{X}_k\right)^T , \tag{4}$$

$$\Sigma^{(C)} = \sum_{k=1}^{K} \left(\hat{X}_k\right)^T \hat{X}_k . \tag{5}$$

After solving the following equations:

$$\Lambda^{(C)} = \left(V^{(C)}\right)^T \Sigma^{(C)} V^{(C)}, \tag{6}$$

$$\Lambda^{(R)} = \left(V^{(R)}\right)^T \Sigma^{(R)} V^{(R)} \tag{7}$$

we get eigenvectors V and eigenvalues Λ matrices which will further serve as the transformation basis. Usually, this step requires a computer-based algorithm for computing eigenvectors and eigenvalues. Such algorithms are available as sub-routines of most matrix algebra systems, e.g. Matlab or Gnu Scientific Library.

Finally, for each block X_k in order to get its transformation Y_k , the following operation is performed [7]:

$$Y_k = V^{(R)}\left(X_k - \bar{X}\right)\ V^{(C)},\ \forall k = 1,\ldots,K\ . \tag{8}$$

It should be noted that we do not perform any kind of dimensionality reduction here, hence the KLT coefficients preserve total information of blocks X_k. The transformation on blocks is a main operation for further steps described in the following sections.

The following figure (Fig. 2) shows a general outline of all the algorithms presented in this paper. Blocks surrounded by a dotted line are specific for an appropriate algorithm, while the remaining blocks are common.

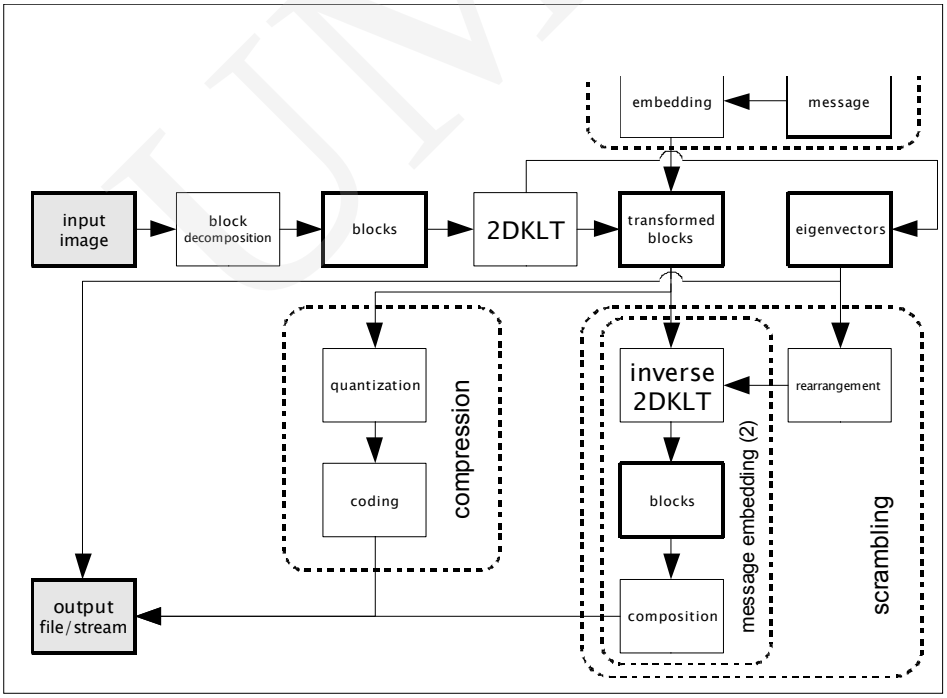

Fig. 2. General diagram of 2DKLT-based image processing algorithms

3. Image compression

There are many storage formats for digital images in the PC environment, however, hardly any meets their specific nature. One of the most common –

JPEG/JFIF [8,9] – is a compromising solution: it gives high compression ratios while retaining acceptable visual quality; its encoders and practical implementations are efficient and easy to use. On the other hand, JPEG has many limitations, e.g. it can not store high dynamic range images. The data transformation used by the JPEG algorithm – DCT (discrete cosine transform) – is not fully adjusted to the characteristic of real scenes. Hence, the higher-frequency details are in the scene, the lower is the quality of an image, which can be seen as distortions around edges and small objects. The possible alternative is applying another transformation with different basis functions, e.g. Discrete Wavelet Transform [9,10]. The most appreciated advantage of wavelets is the fact that their amplitude fades with the distance from the origin of coordinate system, which reduces image distortions. The wavelets are the basis of JPEG-2000 [11].

It is possible to get even higher compression ratios than those given by JPEGs by using other basis functions [12]. These functions are calculated by means of PCA individually for every single image. Compression algorithm presented here uses a two-dimensional version of Karhunen-Loeve Transform (2DKLT) and is, in its principles, close to the JPEG/JFIF algorithm. It consists of the following steps:

1) preparation of data – input image is being decomposed into blocks of equal sizes,
2) calculation of basis functions (eigenvectors) by means of PCA row/column [3,5],
3) transformation of blocks (2DKLT),
4) quantization and storage of results in a form of unsigned fixed-point 8-bit numbers,
5) Huffman coding [9],
6) saving the following structures in a bit stream: coded transformed blocks, image properties (resolution, block size, mean block) and basis matrices.

Contrary to JPEG/JFIF, one extra step has been introduced, namely calculating the specific basis functions instead of standard cosine functions. Coefficients calculated in step 3 are real numbers usually represented with high accuracy. In order to reduce their storage requirements, they are quantized into integer numbers and saved using lower bits number (fixed-point). In the prototype implementation of the developed algorithm, the initial quantization of transformation coefficients is performed using the standard JPEG quantization tables [8], while the output values are saved using the 8-bit unsigned integers (*uint8*). Since the transformation matrices are much more important, yet not very extensive, they are stored using *uint16* type. At this point, each y being an element of Y_k is processed by means of uniform quantization [9]. In order to reduce required storage space, all coefficients in Y are processed using the

Huffman coding [9]. The remaining data required for successful reconstruction do not need to be treated this way, while they do not occupy much space. The final bit stream contains, besides quantized and coded coefficients, the dimensions of an image $\{P, Q\}$, block size $\{M, N\}$, quantized 2DKLT matrices $\{V_M^{(R)}, V_N^{(C)}\}$ and information required for their dequantization as well as mean element.

Figure 3 shows a comparison of the two compression methods: DCT-based (on the left) and 2DKLT-based (on the right). Enlarged fragment of the test image shows that the 2DKLT-based compression gives much more details reducing blurring of individual blocks. It is a result of optimal adaptation of basis functions for this specific image.

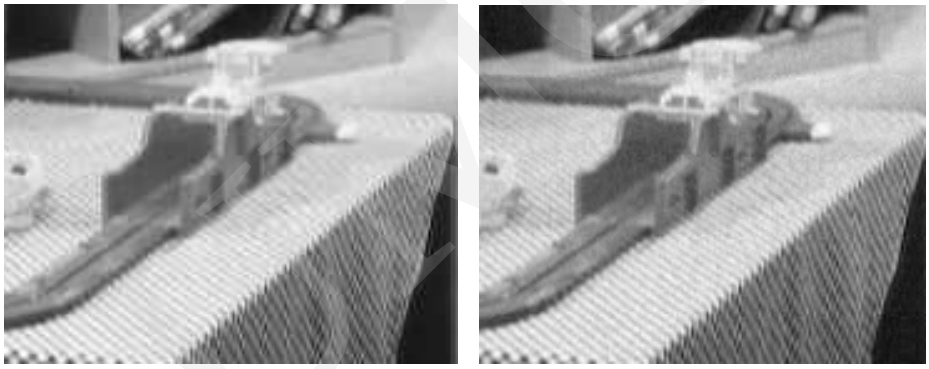

Fig. 3. Enlarged fragment of the test image compressed using the DCT-based (JPEG/JFIF) and 2DKLT-based algorithms

The decompression algorithm includes operations specified at the beginning, however, executed in the reverse order. It is worth noticing that the decompression stage does not require to calculate basis matrices as they are retrieved from a bit stream.

The presented algorithm has been implemented and tested on real images in the Matlab R14 environment. It has proved its efficiency both in visual quality and performance of compression. In order to compare it with existing methods, sample images have been compressed using freely available JPEG and JPEG-2000 coders. The algorithm employing 2DKLT has proved to be competitive with the above approaches in the case of high quality compression (compression rate approximately 2-2.5 – bpp – *bits per pixel*). In the case of high compression rate (1-1.3 bpp), the quality of images processed by JPEG is distinctly lower (PSNR – *Peak Signal/Noise Ratio* – approximately 20-30 dB) than those processed by 2DKLT. The quality of images compressed by both JPEG-2000 and 2DKLT, assuming that the image size is similar, is comparable (PSNR: 30-38 dB).

4. Image scrambling

The next aspect of this work is a proposal of image scrambling in order to protect it from unauthorized access. The most obvious solution is to employ any available, classical cryptographic algorithm (e.g. RSA or DES) on image data [13]. Although, this approach is simple to implement, it requires to perform lots of extra operations, which is hardly cost-effective. It is better to include a cryptographic stage into the compression algorithm. In the method presented here, the scrambling stage does not increase the computational cost of the whole process because it comes from the specific features of KLT [14].

The algorithm employs all the stages presented in the section devoted to image compression. Additionally, in order to scramble the image, the arrangement of the elements in matrices $\{V^{(R)},V^{(C)}\}$ can be changed (e.g. in a pseudo-random way). The reverse order serves as the key S needed for successful descrambling. It should be noted that the image may be descrambled using any key other than the correct one, however, in that case the image will be distorted (see Fig. 4). On the other hand, it is impossible to discover the key S from the scrambled image only. As it can be seen from the example, the proposed scrambling approach may not be as powerful as the full-featured cryptographic solution, however it is sufficient to hide an image content or at least to make it useless for an unauthorized recipient.

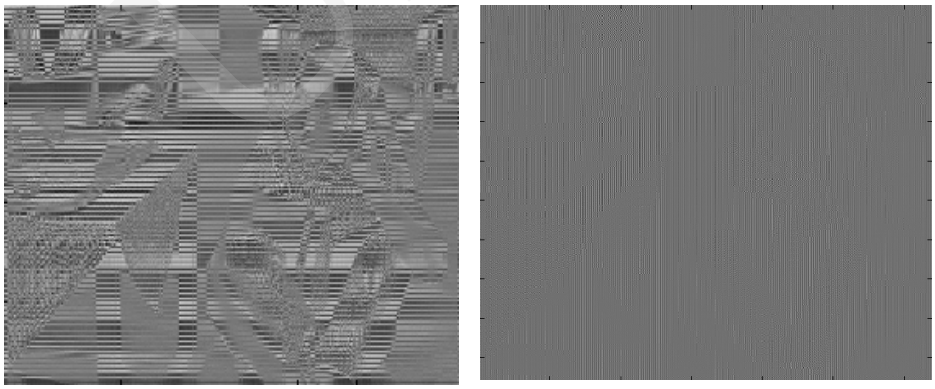

Fig. 4. Two examples of scrambled representations of the test image

Reconstruction of scrambled data consists of the stages presented above, performed in the reverse order. In the first step, the elements of $\{V^{(R)},V^{(C)}\}$ are re-set on the correct positions (according to the S). Then, all elements of matrix Y are dequantized and transformed back to the image domain using inverse 2DKLT. Finally, all blocks are arranged in one matrix – output image. This process is similar to that used in JPEG and is not very time consuming.

5. Information embedding

The problem of hiding certain content into some other data is called steganography and its modern applications include embedding watermarks and copyright information into music, images and videos (so called digital rights management – DRM), hiding secret messages in order to transfer them over the Internet, protect data against alteration, etc. [15,16]. However, not all existing algorithms are usable and robust enough in practical applications, i.e. they do not guarantee that if the carrier (cover object) is modified, the message will be preserved. This problem is often present in the cases when the data has to be protected against unauthorized usage or alternation.

In literature there are many general approaches aimed at still image steganography and watermarking [15-20]. All methods work in either space domain [18] or in certain spectrum domain (FFT, DFT, DCT, Wavelets [17]). The most popular, yet not an effective method is a least significant bit (LSB) insertion [16], which alters the least important part of the cover object. The main problem with LSB and other methods is the robustness to typical visual manipulations, e.g. changes of intensity, contrast and focus as well as lossy re-compression and framing. The developed algorithm presented here was tested on most of them and, as it will be presented, gave satisfactory results, compared to other known approaches [6,15].

The developed algorithm uses a two-dimensional version of Karhunen-Loève Transform and redundant pattern encoding [18]. It consists of the following steps:

1) preparation of data – input image is being decomposed into blocks of equal size, a message is expanded into a bit-wise sequence,
2) calculation of two-dimensional basis functions by means of PCA row/column,
3) transformation of blocks (2DKLT),
4) inserting a binary representation of a message according to the specified key,
5) inverse 2DKLT transformation,
6) joining blocks into a single image.

The reverse algorithm used for extracting the message is based on the same general rules. At the beginning, the message is being decomposed into blocks, which are transformed by means of known eigenvectors (calculated previously). Then the message fragments are being extracted from specific block elements according to the known key. Finally, the bit-wise extended message is compacted into the 8-bit char form for a convenient readout.

In order to embed a message into the cover image, it should be modified in a way that it will contain small values similar to the others existing already in the

KLT spectrum of its pixels. Hence, the message is being expanded into a longer sequence of small values e.g. binary ones. Then we have to generate a key which will determine where in the transformed blocks the elements of expanded message have to be placed. The key is a sequence of offsets from the origin of each block. It is important to place the message into the "middle-frequency" part of each block as a compromise between output image quality and robustness to intentional manipulations. This rule is especially significant in the case of images containing a large number of small details, which helps keep all the changes imperceptible. The only noticeable difference can be observed in the areas of uniform color. As an example of the embedding, the following image is presented (see Fig. 5).

Fig. 5. Fragment of the original test image and its stegoed variant

The maximal length of a message is directly linked to the number of blocks (K), which are calculated from the proportion of image and block sizes. If the message is shorter than the maximal acceptable length, then it can be inserted in the cover image repeatedly. This approach is called redundant pattern encoding [18] and increases the robustness of the message to the intentional manipulations like compressing of filtering.

The extraction algorithm, compared to the embedding phase, is performed in the reverse order. In the first step we decompose the stego image into sub-blocks, which are further transformed using known eigenvectors (2DKLT) and finally, the appropriate part of a message is being extracted from the KLT spectrum of each block. It is important to know the right eigenvectors as well as the key used to extract the exact message. If we do not know the key, we still are able to extract the message (with the help of eigenvectors calculated straight form the cover image), but its exact content will be hard to guess.

The developed algorithm and its prototype implementation have been investigated on the sample 8-bit grayscale images of different origin and characteristics. The images were stored in matrices with values from the interval <0, 255>. To prove the robustness of the algorithm, several tests, according to the 'de facto' standards, have been performed [15,16]. They included lossy JPEG

compression, addition of noise, changing the brightness and contrast of an image and finally convolution filtering.

The analysis of the results shows that it is possible to compress an image containing an embedded message with a quality down to 27% without any loss in the message consistency. The algorithm is also resistant to the additive noise distortion (with uniform distribution), where amplitude equals 15% of the maximal image brightness. The results of experiments show quite an acceptable robustness of the developed algorithm to the changes of intensity and contrast. The experiments showed also that the developed method is robust to the blurring with small masks (up to 3×3 elements).

The algorithm presented here and its application area showed that the 2DKLT method can be a valuable basis for efficient embedding of information in still images. This approach makes it possible to hide a message in the so called cover image with high robustness to image distortions by using optimal representation in the eigenvectors space. The bottom line is that during investigations the hidden message was not detected by one of the most popular steganalysis tools – stegdetect v0.4 [16], which makes the proposed method highly promising.

6. Conclusions

Developed algorithm and its application area presented in this article showed that the 2DKLT method can be a valuable basis for efficient image compression, scrambling and information embedding. Its compression sub-algorithm is competitive with the existing standards (i.e. JPEG/JFIF and JPEG-2000) since it makes it possible to compress images with high precision by using optimal representation in the eigenvectors space, which guarantees their highest visual quality. The second, also important feature of the presented method is scrambling of data, which nowadays is essential for all multimedia applications. It is fast and does not require to perform any extra cryptographic operations. The last aspect of the presented approach is steganography. During the experiments high robustness of 2DKLT-based algorithm to typical steganographic attacks has been proved.

References

[1] Jolliffe I. T., *Principal Component Analysis*. Springer Verlag, NY, (1986).
[2] Fukunaga K., *Introduction to Statistical Pattern Recognition*, second edition, New York: Academic Press, (1990).
[3] Tsapatsoulis N., Alexopoulos V., Kollias S., *A Vector Based Approximation of KLT and Its Application to Face Recognition*. Proc. of The IX European Signal Processing Conference EUSIPCO-98, Island of Rhodes, Greece, September, (1998).
[4] Kukharev G., Forczmański P., *Hierarchical Method of Reduction of Features Dimensionality for Image Recognition and Graphical Data Retrieval*. Proc. of Sixth International Conference PRIP, Minsk, Belarus, May, (2001) 19.

[5] Kukharev G., Forczmański P., *Data Dimensionality Reduction for Face Recognition*. Machine Graphics and Vision., 13(1/2) (2004).
[6] Forczmański P., *Information Embedding in Remotely Sensed Images by Means of Two-Dimensional Karhunen-Loeve Transform*. Polish Journal of Environmental Studies, 16(5B) (2007).
[7] Kukharev G., Forczmański P., *Facial Images Dimensionality Reduction and Recognition by Means of 2DKLT*. Machine Graphics and Vision, Accepted 11.01.2008, (2008)
[8] Wallace G. K., *The JPEG Still Picture Compression Standard*. Communications of the ACM, 34(4), (1991).
[9] Sayood K., *Introduction to Data Compression*. Academic Press, (2000).
[10] Egger O., Fleurry P., Ebrahimi T., Kunst M., *High-performance Compression of Visual Information (A Tutorial Review)*. Proc. of IEEE., 87(6) (1999).
[11] Marcellin M. W., Gormish M. J., Bilgin A., Boliek M., *An overview of JPEG-2000,* Proc. of 2000 Data Compression Conference, (2000) 23.
[12] Forczmański P., *Kompresja obrazów statycznych za pomocą dwuwymiarowej analizy komponentów głównych*. Roczniki Informatyki Stosowanej PS, Nr 10: Metody informatyki stosowanej, Informa, Szczecin, (2006), in Polish.
[13] Furht B., Socek D., Eskicioglu A. M., *Fundamentals of Multimedia Encryption Techniques*. Multimedia Security Handbook, B. Furht and D. Kirovski, Eds. CRC Press, ch. 3, (2004) 93.
[14] Forczmański P., *2DKLT-based Image Compression and Scrambling*. Polish Journal of Environmental Studies, 16(5A) (2007).
[15] Podilchuk, C.I., Delp E.J., *Digital Watermarking Algorithms and Applications*. Proc. IEEE Signal Processing Magazine, 18(4) (2001) 33.
[16] Provos N., Honetman P., *Hide and Seek: An Introduction to Stegangography*. IEEE Security & Privacy Magazine, May/June, (2003).
[17] Kaarna, A., Toivanen P., *Digital Watermarking of Spectral Images in PCA/Wavelet Transform Domain*. Proc. IGARSS, 6 (2003) 3564.
[18] Cheung, W. N., *Digital Image Watermarking in Spatial and Transform Domains*. Proc. TENCON, 3 (2000) 374.
[19] Artz D., *Digital Steganography, Hiding Data within Data*. Proc. IEEE Internet Computing, 5(3) (2001) 75.
[20] Johnson N. F., Jajodia S., *Steganography: Seeing the Unseen*. IEEE Computer, (1998) 26.