Journal of Information Hiding and Multimedia Signal Processing ©2014 ISSN 2073-4212
Ubiquitous International Volume **5**, Number **1**, January **2014**

LSB and DCT Steganographic Detection Using Compressive Sensing

Constantinos Patsakis

Distributed Systems Group
School of Computer Science & Statistics
Trinity College, College Green,Dublin 2, Ireland
patsakik@tcd.ie

Nikolaos G. Aroukatos

Department of Informatics
University of Piraeus
80 Karaoli & Dimitriou str, 18534 Piraeus, Greece
naroykat@unipi.gr

Received May, 2013; revised June, 2013

Abstract. *During the last years, a large number of information in digital form has been exchanged all over the world. Several techniques have been developed in order to protect the users' privacy of digital content. Steganography is one of them. Its use dates back for many centuries, yet it has been used profoundly in the last decades, as the proper mathematical background has been developed along with the need of the music and film industries. One aspect of this development is the use of DRM (Digital Right Management) as the main method of protecting digital assets. This work presents two new detection methods of steganographic content through the use of compressive sensing. The first method is a probabilistic filter that can increase the probability of detecting steganographic content in images, in case where the LSB method is used. The second one helps identifying the original from stego images when DCT steganography is used, after applying a filter with compressive sensing technique.*

Keywords: Steganography, compressive sensing, steganalysis, DCT, LSB

1. Introduction.

Since the mid-90s, the Internet has created a great impact in daily life of people around the world. Using the internet applications like email, the World Wide Web and file sharing, has made the communication between people easier. Nowadays, the Internet has became a source of a large number of digital files containing various forms of data, like text, music, images and videos. A high percentage of these files (e.g. image and video files) bear rights of authors, but due to their digital nature they are easy to steal and make illegal copies. For this reason these kinds of data should be protected in order to be accessible only by legitimate users. This has created the need for for media companies to invest great amounts of their budgets in protecting them and applying the so called DRM, Digital Rights Management. The most widespread method of protection demands the use of steganography in order to store valuable data like user IDs, expiration date, author name etc., in such a way that users other than the authors cannot extract or remove them. All these have made steganography to receive a lot of attention from corporations and academia, leading to a more solid foundation of steganography and to more sophisticated approaches. The two most widely used techniques in images are LSB and frequency domain embedding. The first method is based on embedding the data in the least significant bits of the medium, something that introduces some distortions in the picture. These however may be untraceable, should the pixels used are properly selected. The other methods take the waveform of the image and embed the data in its DCT or DWT form by quantizing the coefficients.

The current work addresses to the following issue: given a set of images which contains the original image and some stego instances of it, we can find the original image should the LSB or the DCT embedding methods have been used. Our new algorithm tries to detect LSB or DCT steganographic embedding of data in images through the use of compressive sensing algorithms, especially the BM3D algorithm.

The classifier that is presented has very good properties as it succeeds in finding the original image in all tests, while the calculations needed can be easily made in a few seconds in moderns computers, without any special configuration. The structure of this work is as follows: In the following sections we provide a brief overview of Steganography, Steganalysis and LSB and DCT embedding methods. Then we discuss about compressive sensing and especially the BM3D algorithm. In the next section we analyze the proposed method and we present the some experimental results and we close with our main conclusions, some considerations and some proposals for future work.

2. Steganography.

Cryptography is one of the oldest methods of data protection and refers to data encryption by using a key and passing the encrypted message to its legal owner. Often, the use of encryption is to define the sender or the receiver as an entity that has something to hide. Apart from that, in cryptography the transmitted data are fully exposed to a third party. For example, there are two entities, “A” and “B”, who are the users of the cryptography system. Entity “A” sends an encrypted text message over a public channel to entity “B”. The adversary “C”, who has perfect read-only access to the public channel, detects the transmission. Even though the transmitted data are encrypted, entity “C” is aware of the fact that the data are in text format. The ability of the exposal of the secret message by entity “C” depends on the strength of the encryption algorithm used by entity “B”. Steganography, on the other hand, can be used for the transmission of secret messages, without the danger of exposing the transmitted data.

The general model of hiding data in other data can be described as follows: The embedded data is the message that one wishes to send secretly. It is usually hidden in an innocuous message referred to as a cover-text, or cover-image or cover-audio as appropriate, producing the stego-text or other stego-object. A stego-key is used to control the hiding process so as to restrict detection and/or recovery of the embedded data to entities who know it (or who know some derived key value). As the purpose of steganography is having a covert communication between two parties whose existence is unknown to a possible attacker, a successful attack consists in detecting the existence of this communication. Using the standard terminology of information hiding, like in the previous example, entities “A” and “B” are the users of the stegosystem. Entity “A” wishes to send an innocent looking picture with an embedded hidden message over a public channel to entity “B”. The presence of hidden information goes unnoticed by the third party, “C”, who has perfect read-only access to the public channel. Entity “A” operates a steganographic program to embed the “secret data” to an innocent looking picture. After the embedding, the stego-picture looks alike the picture before embedding. During the transmission, the “C” party can only detect a “Picture”. It cannot detect that a text message is also transmitted. As a conclusion, we can see that steganography is more efficient when it comes to the danger of the secret message to be exposed.

3. **Steganalysis.** If steganography is considered to be the art of hiding information in a medium, then steganalysis is the art of detecting and extracting knowledge about the embedded data in a medium. Depending on the knowledge that we have about the steganographic system and regardless of the medium that is used as a stego-object, the steganalysis techniques can be categorized to the following main three groups:
Blind identification. In this group, we do not have any knowledge of the steganographic system and we try to detect steganographic data based solely on statistical properties of the medium.
Parametric statistical steganalysis. In some cases we may know that the used steganographic system changes some properties of the medium with specific patterns on some of its properties. The techniques of this group try to detect the existence of steganographic data by detecting these patterns.
Supervised learning based steganalysis. These techniques use statistical classifiers to see whether the tested image is a stego image. In this group we have a set of known clean set of object and one of stego-object. These sets are used for training our algorithm to detect whether a medium is stego-object or not.

We should bare in mind that Kerckhoffs' principle is also valid for steganography, meaning that the attacker knows the embedding algorithm, but not the embedding key or whether the tested medium contains embedded information. So, instead of just finding one key, the attacker must check whether embedded data exist. We should note here that the embedded data cannot be restored by most steganographic attacks, because in most cases the information is encrypted; something that adds an extra layer of security.

Since the focus of this work is on image steganalysis and specially on LSB hiding, we will point out some already used methods. One of the first methods used was χ^2 method [22], where we examine whether the tested image's histogram is close to the histogram of the same image with embedded data.

Fridrich, Goljan and Du [12] proposed in 2001 the RS (Regular/Singular) scheme. Their approach counts the number of occurrences of pairs in sets. The idea is that spatially adjacent image pixels, which are highly correlated, can give us a lot of information whether LSB has been applied in the examined image, meaning that if LSB has been applied then areas where embedding has been made then adjacent image pixels would appear to have many different properties compared to where no tampering has been made.

Related sample pair analysis [8] is another steganalysis method for detecting LSB data hiding in images. In this approach we take sample pairs of values and we divide them into subsets depending on the relation of the two values to one another. This technique makes the assumption that if it is not a stego image, then the number of pairs in each subset are almost equal and that LSB embedding is quantizing the subsets by changing their numbers, something that can be easily detected.

For more an introduction to general steganalysis techniques one may refer to [3,21] and for more advanced techniques and extensions of the above to [8,9,10,11,12,13,14].

4. **The LSB method.** In recent years, LSB (Least Significant Bit) method became one of the most important steganographic methods for hiding data within images [21]. In a NxM color RGB image, with 8 bit color depth, each pixel assumes an integer value x on the closed interval [0, 255] for each color (Red, Green, Blue). The number x represents the density of the color and is encoded by an 8-bit binary word $b_7 b_6 ... b_0$, where $x = \sum_{i=0}^{7} b_i \cdot 2^i$ and $b_i \in \{0, 1\}$. For example, $35 = 0 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 00100011_2$. This definition of x allows the decomposition of an image into a collection

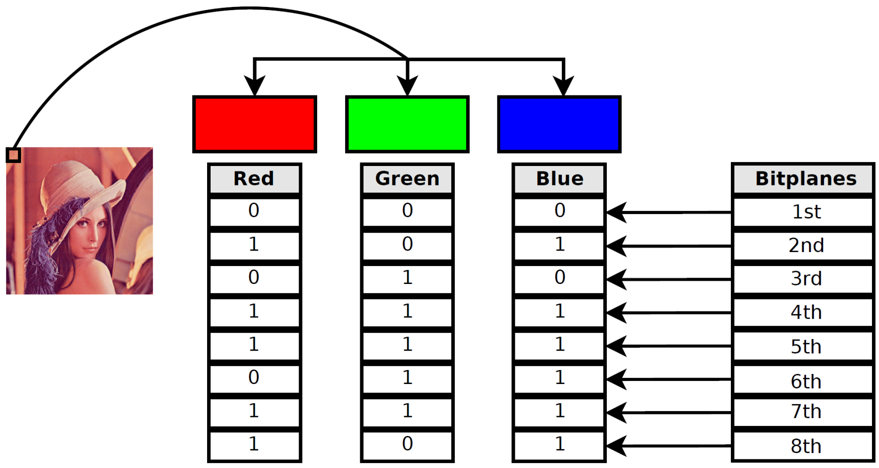


FIGURE 1. 8 bit color image pixel representation

of binary images by separating the bits into 8 bit planes (Figure 1). In the classical LSB embedding methods, the secret message is inserted into the least significant bitplane, i.e., the 8th bitplane, of the cover image, either by directly replacing those bits, or by modifying them by using a particular "inverse" function [16] (Figure 2).

The LSB method embedding algorithm, as well as the extraction algorithm for a text message can be analysed in the following phases.

Embedding algorithm phases:

- Phase 1: Read the text message and convert the text in a binary bit sequence.
- Phase 2: Select an appropriate cover image. (This means that the image has enough stego capacity).
- Phase 3: Replace one by one, LSB (last bit) of cover image with each bit of the binary bit sequence of the "secret" text .
- Phase 5: Create the new stego image.

Extraction algorithm phases:

- Phase 1: Read the stego image.
- Phase 2: Extract the last bit of each pixel, of the stego image.
- Phase 3: Convert the bit sequence to a char sequence and create the text.

The embedding strategy can also be based on sequential insertion or selective embedding of the message in "noisy" areas or pseudo random scattering throughout the image.

To embed the word "HELLAS" we transform it to bits. Then for each pixel we take the binary form of its RGB color value and change least bit value.

5. **The DCT in steganography.** The discrete cosine transform (DCT) is a mathematical transform related to the Fourrier transform. There are many types of DCT, with the DCT-II being the most commonly used form for images. Therefore, for the purpose of this paper, the term DCT refers to the form DCT-II. It can be represented with an invertible function $f : \mathbb{R}^n \longrightarrow \mathbb{R}^n$. This means that we can define from a nxn matrix its invertible. As we know, an image can be considered as a matrix of $m \times n$ pixels. In order to compress an image we may use Discrete Cosine Transform (DCT). Therefore, we divide the picture to small squares of $k \times k$ pixels - commonly in pictures to squares of 8×8 pixels (for higher k values, we have more compression but lower quality) . Now

working from left to right, top to bottom we apply the DCT transform to each square (block).

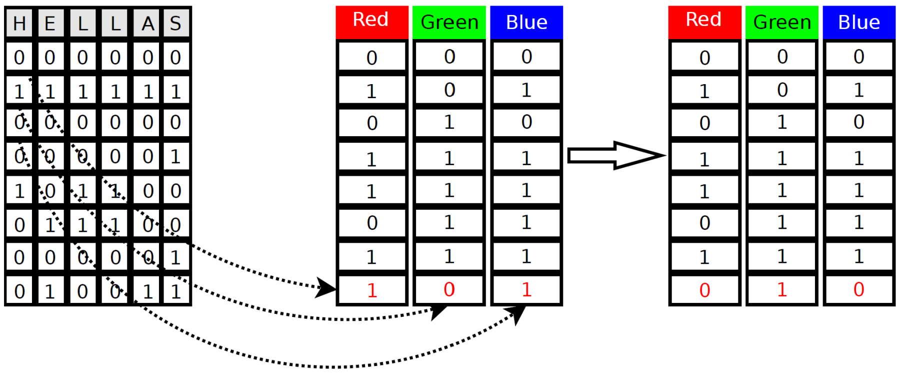


FIGURE 2. The LSB embedding method.

The transform is given by the following formula:

$$C(u,v) = a(u)a(v)t(u,v)$$

$$t(u,v) = \sum_{x=0}^{n-1}\sum_{y=0}^{n-1} f(x,y) cos\left(\frac{(2x+1)u\pi}{2n}\right) cos\left(\frac{(2y+1)v\pi}{2n}\right)$$

where $c(i,j)$ is the intensity of the each pixel in a row, i and column j and $C(u,v)$ declare the DCT coefficient in row u and column v of the DCT matrix. After this, the resulting transform coefficients must be thresholding and quantizing in order to embed the data in the image. Finally, we apply the zig zag scanning and the run length coding.

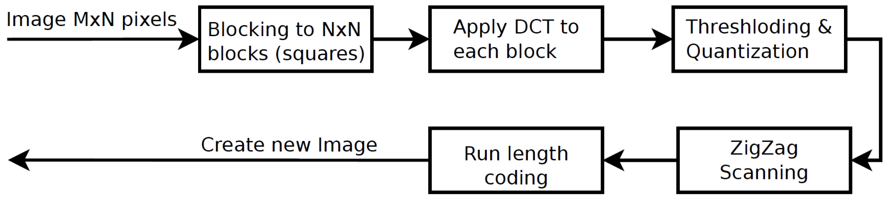


FIGURE 3. The LSB embedding method.

The DCT embedding is very popular in JPEG image format. The DCT method embedding Algorithm as well as the extracting Algorithm for a text message can be analysed in the following Phases:

Embedding algorithm Phases:

- Phase 1: Read the text message and covert the text in a binary bit sequence.
- Phase 2: Select an appropriate cover image. (This means that the image has enough stego capacity).
- Phase 3: Divide the cover image to squares of 8×8 pixels.
- Phase 4: Working from left to right, top to bottom apply the DCT transform to each square of 8×8 pixels.
- Phase 6: Each block is compressed through quantization table.
- Phase 7: Calculate LSB of each DC coefficient and replace with each bit of the text message.

- Phase 8: Create the new stego image.

Extracting algorithm phases:

- Phase 1: Read stego image.
- Phase 2: Divide the cover image to squares of 8×8 pixels.
- Phase 3: Working from left to right, top to bottom apply the DCT transform to each square of 8×8 pixels.
- Phase 4: Each block is compressed through quantization table.
- Phase 5: Calculate LSB of each DC coefficient.
- Phase 6: Convert the bit sequence to a char sequence and create the text.

From Steghide [20] to [15], DCT manages to achieve high capacity embedding with minor distortion of the original image.

6. Compressive sensing.

Compressive sensing or sampling is a very recent branch of signal processing that originates from Emanuel Candes. He used data of an incomplete medical image and passed them through a sequence of l_1 transformations, trying to improve their quality. Despite the fact that the original data was inadequate according to the known information theory, he observed that the final images were quite sharp. The results of this procedure, challenged the theorem of H. Nyquist and C. Shannon. The Nyquist-Shanon theorem provides a way to compute the nominal sampling interval required to avoid the aliasing effect. This theorem states that the sampling frequency should be at least twice the highest frequency contained in the signal. In mathematical terms: $f_s \geq f_c$ where f_s is the sampling frequency (number of samples taken in time or space), and f_c is the highest frequency contained in the specific signal [1].

In fact, the process that Candes discovered, recreated an image file very similar to the original, using a reduced amount of image data [2]. This can be consider a form of lossy compression method, as well. This method is called compressive sampling or compressive sensing and has been applied successfully in all forms of digital signal. Compressive sensing, also known as compressed sensing and compressive sampling is a method to reconstruct a signal from a random sparse sample. This method is used by several scientific fields in Information technology, from coding theory and MRIs to music sampling and image compression. Donoho first and then Candes, Romberg and Tao devised a method to reconstruct an image from an amount of data much less from the number of data that would be deemed sufficient by the Nyquist–Shannon sampling theorem [19]. Their approach was that in many cases the signals are sparse, hence using another smaller basis we might be able to recover the most useful part of the signal by using a smaller sample. Several applications of these algorithms have so far been made, raging from MRI reconstruction to coding theory.

It consists of the following Phases:

Phase 1: Consider an original signal, i.e. an image. Every image consists of a number of pixels depending on the image dimensions. Let us consider an image of 256 * 256 pixels. In this image we have a total of 65.536 pixels.

Phase 2: We randomly select a small set of pixels. In the second phase, using the algorithms of the Laplace transformations and in particular the l_1 transformation, we fill the empty areas of the picture with random pixels.

Phase 3: This phase of the process, consists of two discrete steps. First, the algorithms detect clusters of pixels with, largely, the same color and groups them into large colored polygons. In the second step, the same process is repeated in progressively smaller areas, producing smaller and smaller polygons.

Phase 4: The final output of this process is an image almost identical to the original (figure 4).

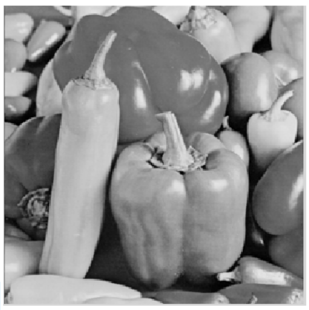
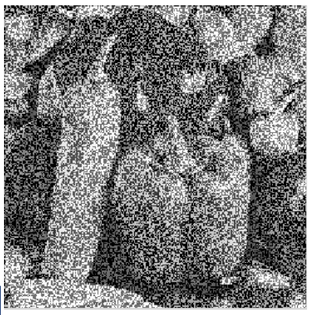
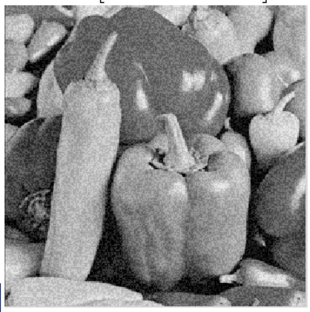
[Original image] [CS Phase 2] [CS Phase 3]

Figure 4. CS method.

7. **Compressive sensing and BM3D.** Bm3d algorithm is a new image denoising method using the idea of compressive sensing. The novelty of the method is the use of 3D data arrays for grouping inside the 2D image fragments, and so the method enhanced the sparse representation in transform domain.The algorithm is very fast and has very good results with noisy images.We present now briefly the phases of the B3d algorithm.

Phase One

- Step 1: Read a noisy image and fragment in square blocks with fixed size.
- Step 2: Block Matching (Extract reference blocks and find similar blocks to the reference one).
- Step 3: Stack the similar blocks in a 3d array.
- Step 4: 3D transform of 3D arrays (Denoising).
- Step 6: Shrinkage of transform spectrum and inverse 3D tranformation (Denoising).
- Step 7: Compute the basic estimate of the image.
- After the ending phase one, using the basic estimate we repeat the similar process in phase two

Phase Two

- Step 1: Finding similar blocks in noisy picture and in basic estimate picture.
- Step 2: Stack the similar blocks in two 3d arrays one for each picture.
- Step 3: 3D transform of 3D arrays.
- Step 4: Inverse 3D tranformation.
- Step 7: Compute the final estimate of the image.

8. **The proposed method for LSB Stego Images.** The main idea of our proposal for LSB stego images, is to regard each image to be examined as an image with embedded noise. The more noise our image has, the more the image is probable to have LSB embedded information. The measure of the peak signal-to-noise ratio (PSNR), gives us an idea about the quantity of noise in the under consideration image. The PSNR is the ratio between the maximum possible power of a signal and the power of the under consideration signal. The PSNR is used originally as a measure of quality for the compression codecs in images and videos (e.g., jpeg,mpeg). In our case, the signal is the original data of the picture and the noise is the stego data introduced by the LSB method.

As discussed above, compressive sampling is a method for reconstructing our signals, by taking a small sample. The assumption that we make is that if the image has no LSB embedding, then the reconstruction with a compressive sampling will be closer to the original, than to one stego-image. The reason for making this assumption is that when parsing an image from a compressive sampling algorithm, if it has LSB data embedded, then the algorithm will not perform as well, because it has additional noise to remove.

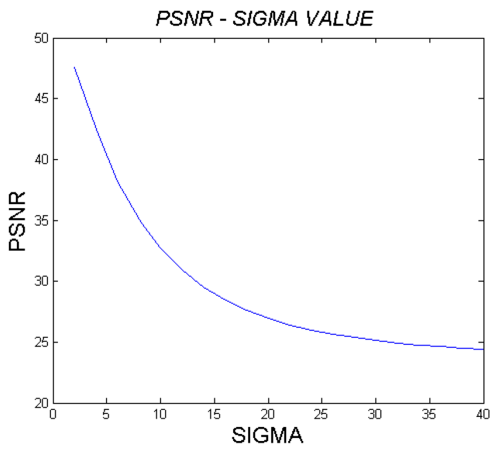


Figure 5. PSNR-Sigma Value Diagram.

The method now is very straight forward, firstly we select a compressive sampling algorithm, in our case BM3D. Afterwards we take the image I to be examined and parse it to BM3D to recover image I'. We then compare the differences of images I and I' in pixel colors.

9. **Results for LSB Stego Images.** In order to test our method, we made several tests using Matlab and a set of classic test images for image processing. In our tests we took each picture I, created a stego-image out of it, with LSB embedding and then we processed both images using BM3D. We then took each processed picture and compared it to the original one, keeping their percent color difference. The results can be seen in Tables 1 and 2.

For parameter sigma equal to 1 it becomes apparent that we have a big difference between "clean" images and stego images. The results are more clear for sigma equal to 2, where we can have an obvious criterion for LSB stego images, which is 40%.

10. **The proposed method for DCT Images.** In our approach, as in [17], we tried to create a criterion based on image reconstruction [18]. Trying to reconstruct images with a compressive sensing algorithm, we use as source input the original image and compare it to itself, something that should result to obvious differences with applying the

camera256		lena512		barbara512		boat512	
Stego-image	self	Stego-image	self	Stego-image	self	Stego-image	self
12.6785	3.5461	7.2735	2.1938	6.3694	1.9016	2.1450	0.6226
12.6724	3.6194	7.3082	2.1770	6.3522	1.8887	2.1580	0.6153
12.6663	3.5675	7.3376	2.1698	6.3393	1.8967	2.1885	0.6203
12.7441	3.5767	7.3650	2.2011	6.3660	1.9070	2.1935	0.6157
12.6389	3.6911	7.3418	2.1770	6.4304	1.8661	2.1790	0.6134
12.7151	3.5629	7.3009	2.2072	6.3995	1.9062	2.1687	0.6275
12.6831	3.6301	7.3254	2.1992	6.3465	1.9138	2.1610	0.6001
12.7579	3.6026	7.3372	2.1996	6.3892	1.8829	2.1824	0.6363
12.6343	3.5172	7.3704	2.1847	6.3488	1.8887	2.1919	0.6310
12.7792	3.5782	7.3563	2.1645	6.3854	1.8501	2.1652	0.6275
house512		hill512		house256		f16 512	
Stego-image	self	Stego-image	self	Stego-image	self	Stego-image	self
13.7711	3.9795	2.7973	0.9209	13.7177	4.1214	13.6410	3.7518
13.7192	3.9795	2.8141	0.9212	13.6993	4.1489	13.6063	3.8128
13.6841	4.0085	2.8297	0.9312	13.7329	4.0009	13.5990	3.7708
13.6734	3.9047	2.8099	0.9480	13.6810	4.2007	13.5761	3.8303
13.7909	3.9185	2.8172	0.9281	13.8107	4.0344	13.6692	3.7991
13.6856	3.8986	2.7718	0.9422	13.7009	4.1748	13.6467	3.7914
13.7939	3.9200	2.8049	0.9308	13.7604	4.1351	13.5857	3.7815
13.7375	3.9841	2.7851	0.9430	13.7482	4.1275	13.6761	3.7796
13.6780	3.9124	2.7779	0.9415	13.7207	4.1214	13.6066	3.7937
13.7070	3.9154	2.8038	0.9377	13.6658	4.0985	13.6459	3.8197

Table 1. Results table with parameter sigma equal to 1.

same algorithm with source input the stego images. The differences stem from the added "noise" that has been embedded in the images.

An outline of the method can be seen in figure 6, while the feature extraction method can be seen in figure 7.

Let's suppose that we have a set of n grayscale images, which contains an image a_1 of size $x \times y$ pixels and $n-1$ instances of the image $a_2, a_{3,} ..., a_n$where data have been embedded using DCT. We take each image and parse it to BM3D algorithm [4,5,6,7], which tries to reconstruct it with $\sigma = 1$, resulting to image $r_i, I \in \{1, 2, .., n\}$. We denote A_i the matrix corresponding to image a_i and R_i the matrix corresponding to reconstructed image r_i. We then calculate matrix M_i which is the difference $A_i - R_i$. We then add all elements of M_i and divide the result with xy, the number of pixels of the image. The resulting value is the extracted feature value v_i.

The extracted feature values v_is are then compared. The values that are close to each other form groups. Our experiments over big set of images and embedded data, indicate that only two sets are created, one with all the stego images and one containing only the original image.

The values of matrix M_i indicate how close was the estimation of BM3D algorithm to the original picture. The hypothesis, which was proved by the experimental results that are going to be shown in the next section, was that the stego images should have v_i that are significantly different from v_1, the feature value of the original image a_1, as the DCT data embedding would result to similar distortions for all stego images.

11. Experimental results.

In our tests we've used a set of well known clean images and using Steghide [20] we've embedded random data in DCT using random encryption

camera256		lena512		barbara512		boat512	
Stego-image	self	Stego-image	self	Stego-image	self	Stego-image	self
40.3076	15.7578	51.6136	22.9492	40.4022	16.3910	33.0948	11.2705
40.4037	15.8493	51.7689	22.8558	40.4358	16.4154	33.0269	11.3544
40.5121	15.8386	51.6273	22.9527	40.4747	16.3254	33.0128	11.3163
40.3961	15.7898	51.7071	22.9961	40.4057	16.4505	32.9964	11.3148
40.3244	15.8722	51.6781	23.0061	40.4091	16.4417	33.0132	11.3434
40.4922	15.9988	51.6968	22.9591	40.4026	16.4261	32.9857	11.3171
40.5380	16.0004	51.6750	22.9942	40.4678	16.4295	32.9704	11.3155
40.2603	15.8127	51.6808	22.9862	40.4591	16.3738	32.9750	11.2873
40.3946	15.7013	51.6468	22.9397	40.4305	16.4524	33.0158	11.2595
40.3076	15.7196	51.6842	22.9893	40.4610	16.3269	33.0318	11.2869
house512		hill512		house256		f16 512	
Stego-image	self	Stego-image	self	Stego-image	self	Stego-image	self
44.7510	15.5579	34.7626	13.2710	44.2902	15.4190	43.7489	16.9132
44.8563	15.6723	34.7836	13.2141	44.6243	15.5716	43.7405	16.9407
44.7525	15.6052	34.8049	13.1409	44.8288	15.5914	43.8583	17.0349
44.9631	15.5136	34.7996	13.1508	44.1925	15.5396	43.9098	16.9735
44.8730	15.7425	34.8301	13.1710	44.4580	15.5807	43.8236	16.9411
44.7845	15.4694	34.7305	13.1321	44.5419	15.5853	43.7756	16.9014
44.9738	15.6906	34.7878	13.1653	44.5465	15.6082	43.7210	16.9521
44.6198	15.8340	34.8412	13.0959	44.7235	15.6204	43.7336	17.0258
45.0974	15.6906	34.7809	13.1603	44.5862	15.3336	43.7870	16.8785
44.9158	15.7990	34.7622	13.1874	44.4595	15.6586	43.7050	16.9323

Table 2. Results table with parameter sigma equal to 2.

	Original Image	Stego 1	Stego 2	Stego 3	Stego 4	Stego 5
Image 1	123.10	178.39	178.39	178.39	178.39	178.39
Image 2	111.88	183.97	183.97	183.97	183.97	183.97
Image 3	68.89	77.58	77.58	77.58	77.58	77.58
Image 4	81.91	158.51	158.51	158.51	158.51	158.51
Image 5	101.49	92.56	92.56	92.56	92.56	92.56
Image 6	69.78	79.55	79.55	79.55	79.55	79.55
Image 7	172.44	92.63	92.63	92.63	92.63	92.63
Image 8	66.69	76.72	76.72	76.72	76.72	76.72
Image 9	103.33	74.26	74.26	74.26	74.26	74.26
Image 10	178.39	63.91	63.91	63.91	63.91	63.91

Table 3. Test results

keys. For each image we have created ten stego-images. The measurements were made using BM3D Matlab implementation provided at [18].

In table 3 it is clear that the measurements of the group of stego-images are quite different from the original images. Therefore, we may easily separate the original images from the stego images. We should note that in order to use the proposed classifier, we need to have at least three images. In case we have checked two images, depending on the content we may have bigger or smaller v_is when comparing to v_1, so a third value is needed to point out, by majority rule the correct classification.

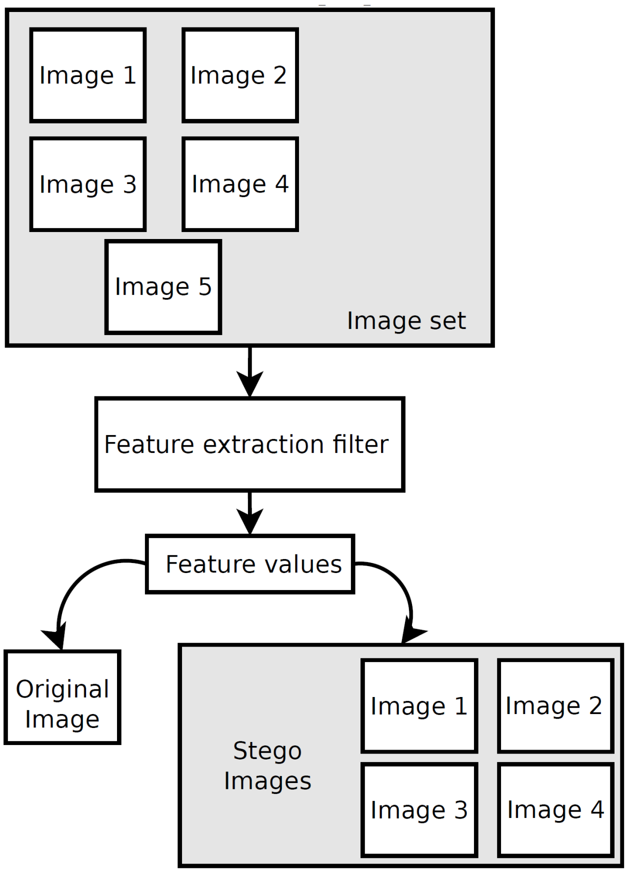


FIGURE 6. The proposed method

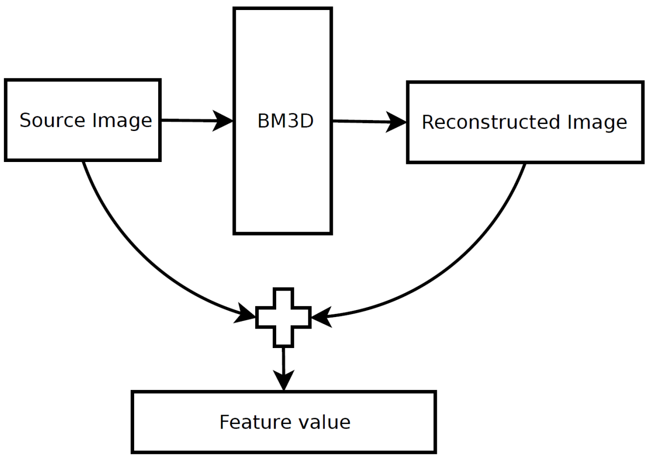


FIGURE 7. Feature extraction

12. **Conclusions.** In our work we use the compressive sensing method to investigate the existence of steganographic content in Images. We use two very popular methods to embed stego data in images, the LSB and the DCT. The algorithm can detect with high success rate the existence of stego content in LSB pictures and can classify pictures with DCT stego content. The algorithm is very fast and can process a large number of pictures and detect stego content in seconds without any previous training.

Since the idea is promising and has very good results, there are more things to be studied. For example, whether there are any other image features that can be extracted from compressive sensing, or attacks on watermarking schemes and extensions to sound media. The results of this work show that the use of compressive sensing can find in steganalysis many applications and illustrate that the use of compressive sampling algorithms on steganalysis of other embedding methods can be significantly improved. Our further work will examine the possibilities of the compressive sensing algorithms in others steganograpfhic methods.

REFERENCES

[1] K. L. Chiew, and J. Pieprzyk, Binary image steganographic techniques classification based on multi-class steganalysis, *Proc. of the 6th international conference on Information Security Practice and Experience*, pp. 341-358, 2008.

[2] Steghide, available at http://steghide.sourceforge.net/index.php.

[3] A. Danielyan, M. Maggioni, K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image and video denoising by sparse 3D transform-domain collaborative filtering*, available at http://www.cs.tut.fi/ foi/GCF-BM3D/

[4] C. Patsakis, N. Aroukatos, and S. Zimeras, LSB steganographic detection using compressive sensing, *Proc. of the 4th International Conference on Intelligent Interactive Multimedia Systems and Services*, pp. 219-225, 2011.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, Image denoising with block-matching and 3D ?ltering, *Proc. of SPIE*, vol. 6064, pp. 606414.1-606414.12, 2006.

[6] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, Image denoising by sparse 3-D transform-domain collaborative filtering, *IEEE Trans. Image Processing*, vol. 16, no. 8, pp. 2080-2095.

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, Joint image sharpening and denoising by 3D transform-domain collaborative filtering, *Proc. of the 7th International Workshop on Spectral Methods and Multirate Signal Processing*, 2007.

[8] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, Image restoration by sparse 3D transform-domain collaborative filtering, *Proc. of SPIE*, vol. 6812, pp. 681207.1-681207.12, 2008.

[9] Stegdetect, available at http://www.outguess.org/download.php.

[10] R. Chandramouli, Mathematical approach to steganalysis, *Proc. of SPIE*, vol. 4675, pp. 14-25, 2002.

[11] J. Fridrich, and M. Goljan, Practical steganalysis of digital images: state of the art, *Proc. of SPIE*, vol. 4675, pp. 1-13, 2002.

[12] J. Fridrich, M. Goljan, and R. Du, Steganalysis based on JPEG compatibility, *Proc. of SPIE*, vol. 4518, pp. 275-280, 2001.

[13] J. Fridrich, M. Goljan, and D. Hogea, Attacking the OutGuess, *Proc. of the ACM Workshop on Multimedia and Security*, 2002.

[14] J. Fridrich, M. Goljan, D. Hogea, New methodology for breaking steganographic techniques for JPEGs, *Proc. of SPIE*, vol. 5020, pp. 143-155, 2003.

[15] J. Fridrich, M. Goljan, D. Hogea, and D. Soukal, Quantitative steganalysis of digital images: estimating the secret message length, *Journal of Multimedia Systems*, vol. 9, no. 3, pp. 288-302, 2003.

[16] J. T. Jackson, G. H. Gunsch, R. L. Claypoole, and G. B. Lamontg, Blind steganography detection using a computational immune system: a work in progress, *International Journal of Digital Evidence*, vol. 1, no. 4, pp. 1-19, 2003.

[17] R. Ferreira, B. Ribeiro, C. Silva, L. Qingzhong, and A. H. Sung, Building resilient classi?ers for LSB matching steganography, *Proc. of IEEE International Joint Conference on Neural Networks*, pp. 1562-1567, 2008.

[18] B. Rodriguez, and G. Peterson, Detecting steganography using multi-class classification, *Journal of International Federation for Information Processing*, vol. 242, pp. 193-204, 2007.

[19] J. J. Harmsen, and W. A. Pearlman, Kernel Fisher discriminant for steganalysis of JPEG hiding methods, *Proc. of SPIE*, vol. 5306, pp. 13-22, 2004.

[20] R. J. Marks II, *Introduction to Shannon sampling and interpolation theory*, Spinger, New York, USA, 1991.

[21] E. J. Candes, and M.B. Wakin, An introduction to compressive sampling, *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21-30, 2008.

[22] C. C. Lin, and P. F. Shiu, High capacity data hiding scheme for DCT-based images, *Journal of Information Hiding and Multimedia Signal Processing,*, vol. 1, no. 3, pp. 220-240, 2010.

[23] P. Wayner, *Disappearing Cryptography: Information Hiding: Steganography & Watermarking*, Morgan Kaufmann Publishers, San Francisco, USA, 2009.

[24] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*, Morgan Kaufmann Publishers, San Francisco, USA, 2008.

[25] D. L. Donoho, Compressed sensing, *IEEE Trans. Information Theory*, vol. 52, no. 4, pp. 1289-1306, 2006.

[26] E. J. Candes, and M.B. Wakin, An introduction to compressive sampling, *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21-30, 2008.

[27] D. L. Donoho, Compressed sensing, *IEEE Trans. Information Theory*, vol. 52, no. 4, pp. 1289-1306, 2006.

[28] A. Westfeld, and A. Pfitzmann, Attacks on steganographic systems, *Journal of Lecture Notes in Computer Science*, vol. 1768, pp. 61-76, 2000.

[29] J. Fridrich, M. Goljan, and R. Du, Reliable detection of LSB steganography in color and grayscale images, *Proc. of the 4th workshop on Multimedia and security: new challenges*, pp. 27-30, 2001.

[30] S. Dumitrescu, X. Wu, and Z. Wang, Detection of LSB steganography via sample pair analysis, *IEEE Trans. Signal Processing*, vol. 51, no. 7, pp. 1995-2007, 2003.

[31] S. Dumitrescu, and X. Wu, A new framework of LSB steganalysis of digital media, *IEEE Trans. Signal Processing*, vol. 53, no. 10, pp. 3936-3947, 2005.

[32] S. Dumitrescu, and X. Wu, LSB steganalysis based on high-order statistics, *Proc. the 7th workshop on Multimedia and security*, pp. 25-30, 2005.

[33] T. Pevný, and J. Fridrich, Novelty detection in blind steganalysis, *Proc. of the 10th ACM workshop on Multimedia and security*, pp. 167-176, 2008.

[34] T. Pevny, J. Fridrich, and A. D. Ker, From Blind to Quantitative Steganalysis, *Proc. of SPIE*, vol. 7254, pp. 0C1-0C14, 2009.

[35] C. E. Shannon, Communication in the presence of noise, *Journal of Proceedings of the IRE*, vol. 37, no. 1, pp. 10-21, 1949.

[36] E. J. Candes, J. Romberg, and T. Tao, Stable signal recovery from incomplete and inaccurate measurements, *Journal of Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207-1223, 2006.

[37] C. Patsakis, and N. Aroukatos, A DCT steganographic classifier based on compressive sensing, *Proc. of the 7th International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pp. 169-172, 2011.