Lossless Crypto-Data Hiding in Medical Images Without Increasing The Original Image Size

José Marconi Rodrigues, William Puech, Christophe Fiorio

LOSSLESS CRYPTO-DATA HIDING IN MEDICAL IMAGES WITHOUT INCREASING THE ORIGINAL IMAGE SIZE

J.M. Rodrigues, W. Puech and C. Fiorio

Laboratoire d'Informatique Robotique et Microlectronique de Montpellier
LIRMM, UMR CNRS 5506, Université Montpellier II, France
jose-marconi.rodrigues@lirmm.fr, puech@lirmm.fr, fiorio@lirmm.fr

Abstract: In this paper, a novel lossless crypto-data hiding method for medical image is proposed. It is based on decomposition and image contents. It can embed information of the patient into the image in a high undecipherable way without neither increase of size nor change of original contents. The main idea is to decompose the medical image in two sub-images and to apply to each sub-image different process in order to gain space and information privacy. It can be used to embed the diagnostic and/or the historical disease into medical images of the patient for safe transfer purpose.

Keywords: Medical images, crypto-data hiding, lossless.

INTRODUCTION

Data hiding has recently drawn extensive interests from research communities such as signal processing and system security. It is noticed that most of current data hiding techniques belong to lossy data hiding [4]. It means that, after data hiding process, the cover image is permanently changed. That is, we can not recover the original media and important characteristics can be lost. For some applications, e.g., the law enforcement and medical fields, the original cover media are as critical as the hidden data, and they should be totally recovered. The reversible data hiding techniques are the solution for this problem. They can be used to attach crucial information to the media without change their original contents.

There are several lossless data embedding techniques such as [3, 5] and several lossless image compression methods such as [8, 9, 6]. However, none of these can perform both lossless data embedding and encryption of images. Actually, lossless embedding increases the size of the original image and lossy embedding process can not be applied to medical field. So, it is necessary to search for techniques that can embed a large amount of information, neither increase the size nor change the original image data. In this work we present a new Lossless Crypto-Data Hiding technique (LCDH) for medical image that decreases the original file size.

The paper is organized as follow. Section 2 details the method and its steps, decomposition, data hiding, ciphering (fusion/scrambling/cryptography) and extraction and Section 3 presents the test results applied to medical images.

THE METHOD

The method summary is described in the diagram illustrated in Fig.1. First, the original image is decomposed in two semi-pixel images. Then, the most significant semi-pixel image is compressed and steganographed with patient information. Then, this image is concurrently cryptographed with a secret-key and scrambled with the least significant semi-pixel image.

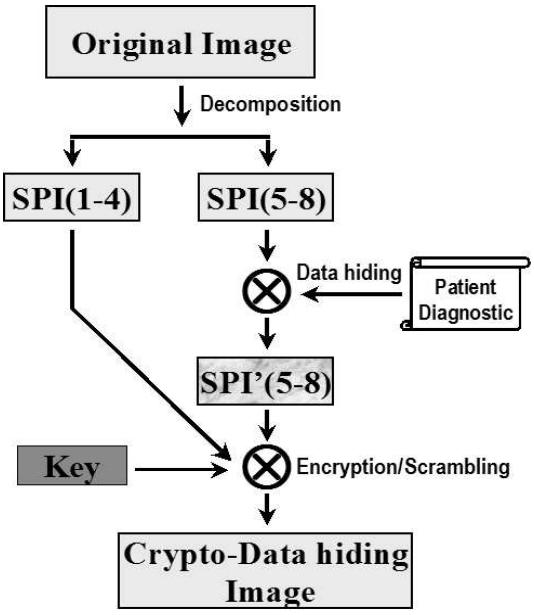


Fig. 1: Global method overview.

Image Decomposition

The basic idea of the method is to decompose the original image in two semi-pixel images. The first Semi-Pixel Image SPI(1-4), is composed by the four LSBs of the original image pixels. The second Semi-Pixel Image SPI(5-8) is composed by the four remaining bits, as presented Fig.2.

This systematic is based in the S.S. Maniccam and N. Bourbakis work [6], that shows that the high compression factor is accomplished in the four most significant bits (MSB). It is notice, Fig.3, that the most representative part of the image is in MSB, SPI(5-8). Moreover, in this SPI(5-8) there is a great number of homogeneous areas. So, we will apply algorithm to compress this semi-pixel image as shown in the next section.

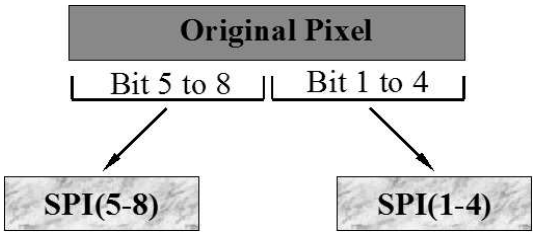


Fig. 2: Decomposition in two semi-pixel images.

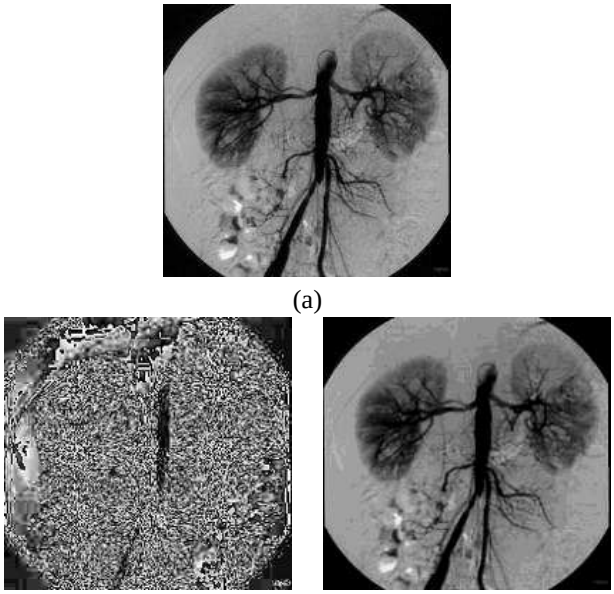


Fig. 3: a) Original Image, b) Least significant semi-pixel image, SPI(1-4), c) Most significant semi-pixel image, SPI(5-8).

Semi-Pixel Image(5-8)

This semi-pixel image is compressed using a variation of the Run length encoding (RLE). RLE is a simple lossless compression that assigns short codes to long runs of identical symbols. It is for images that contain lots of homogenous areas. It is very fast, easy to implement and does not require much CPU efforts.

RLE Compression

There are some variations of RLE, but basically in the images, it works as follow. When the RLE algorithm find a long sequence of same pixel color, it represents this sequence with a **special block** that contains:

- Flag to say that it is a special block.
- Sequence length.
- Pixel color.

We propose a variation of RLE algorithm which uses three kinds of **special blocks** and these blocks can vary according the image contents:

- Block 8 bits with one bit for data hiding (B_8):
  - 1 bit ($flag = 0$) to identify block type.
  - 1 bit for data hiding.
  - 2 bits for sequence length.
  - 4 bits for semi-pixel color.

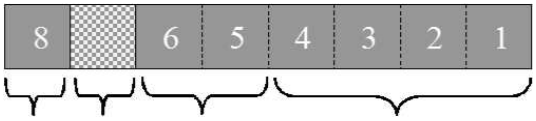


Fig. 4: Block B_8 layout.

- Block 8 bits without bit for data hiding (B_8')
  - 1 bit ($flag = 0$) to identify block type.
  - 3 bits for sequence length.
  - 4 bits for semi-pixel color.

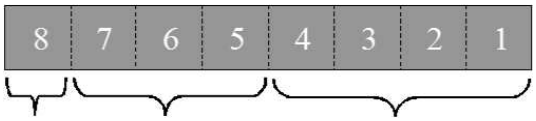


Fig. 5: Block B_8' layout.

- Block 16 bits (B_{16})
  - 1 bit ($flag = 1$) to identify block type.
  - 2 bits for data hiding (at least).
  - 9 bits for sequence length (at most).
  - 4 bits for semi-pixel color.

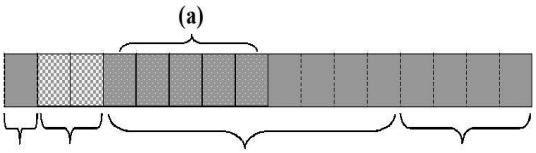


Fig. 6: Block B_{16} layout. Some times the part (a) can be used for data hiding too.

The block B_{16} has at least two bits for data hiding. However, the total number of bits used for data hiding in B_{16} depends on the sequence length, because some bits of the sequence length can also be used for embedding purpose, Fig.6. For example, if the longest sequence length in SPI(5-8) is 60, only six of the nine bits are needed per block to represent this value. Thus, we gain plus three bits per block B_{16} for information enclosing. In SPI(5-8), the data hiding space magnitude *W* is:

$$W = n_8 + (n_{16} \times H_{16}), \quad (1)$$

where n_8 is the quantity of B_8, n_{16} is the quantity of B_{16} and H_{16} is the sum of bits used for data hiding in the block B_{16}.

The methodology assumes that the semi-pixel image has only two kind of blocks at each time, the (B_8 and B_{16}) or (B_8' and B_{16}). The type and size of these special blocks are chosen according the *image contents* and the *optimal length of the longest sequence* and they are simultaneously dependents.

Since the compression is done by specifying the identical sequences, the access way to the semi-pixel is important. Thus, we track it in three different manners (per rows, per columns and spiral in way) as illustrated Fig.7. That will influence in the magnitude and in the quantity of homogenous regions.

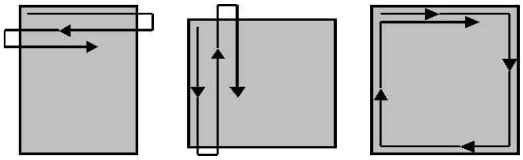


Fig. 7: Ways the SPI(5-8) is accessed, a) row, b) column, c) spiral.

The main goal of this work is not to find the best path to obtain a high compression factor, but an reasonable way of doing it. Our aim is to find an efficient path in order to reach a little processing time, high data hiding capacity and small final image size. Thus, inspire of there are many scan approaches [7], SPI(5-8) is tracked only by three ways explained above.

Table 1: Bits for data hiding per block type and sequence length.

Longest Segment		Bits for Data Hiding		
Block 8	Block 16	Block 8	Block 16	Total
04	16	1	7	8
08	16	0	7	7
04	32	1	6	7
08	32	0	6	6
04	64	1	5	6
08	64	0	5	5
04	128	1	4	5
08	128	0	4	4
04	256	1	3	4
08	256	0	3	3
04	512	1	2	3
08	512	0	2	2

The *image contents* defines the number and type of blocks that will be used. If SPI(5-8) has a remarkable sum of homogeneous regions, and those region lengths are very long, the algorithm will use many blocks B_{16}. If those homogenous areas are relatively small the algorithm opts for blocks B_8', and if they are still smaller, the algorithm uses B_8.

The *optimal length of the longest sequence* is dynamically determined for each kind of blocks. It is achieved in function of both the resulted image size *N* and the space magnitude for data hiding *W*. The algorithm works as follow. It scans the image considering that the longest sequence length for block type 8 can be $\{4$ or $8\}$. It is interesting to explain that we use 4 because it is the longest sequence length that can be stored in two bits, block B_8. Consequently, the value 8 is the longest sequence length that can be saved in three bits, block B_8'. The same process is employed for B_{16} utilizing the set values $\{16$, 32, 64, 128, 256 and $512\}$. Similar reasoning used in blocks 8 can also be applied in B_{16}. The maximum set value is 512, because it is the biggest quantity of value that can be put in nine bits. After this computation, the algorithm defines the optimal length of the longest sequence (OLLS) according the biggest *W* and the smallest final image size *N*.

$$OLLS = \begin{cases} Max(W) \\ Min(N) \end{cases}$$

The Tab.1 shows the amount of bits used for data hiding, per block type and per sequence length.

Lossless Data Hiding

Since the RLE algorithm was run and we already have the space magnitude for data hiding *W*, the process of information embedding now can be made as shown in Fig. 8.

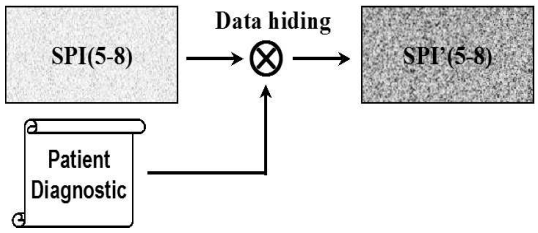


Fig. 8: Data Hiding Diagram.

The information embedding is made bit per bit, and the following considerations should be emphasized. The information will be dispersed over the semi-pixel image in sliced way. It means, that one byte of the embedded information can be divided in a lot of kind of small pieces. It depends on the disposition of B_8 and B_{16} in SPI(5-8).

The information that will be embedded in SPI(5-8), can be any type of binary data (image, text, sound, etc.) After data hiding process SPI'(5-8) will be sliced in portions of four bits to be used in the ciphering process explained in Section .

Ciphering procedure

The ciphering process has three basic and concomitant steps. They are scrambling, fusion and encryption. The scrambling/fusion because of the original image decomposition, and the encryption for security reasons.

In spire of the data hiding procedure spread over SPI'(5-8) the information in a ineligible form, it is can not be considered as an encryption method. An encryption method which depends on the secrecy of the encryption process is not truly an encryption method [2]. The proposed ciphering method is a high-key dependent and it accepts a very long key. In fact, the key can be as long as SPI'(5-8) size. The use of key will be detailed in Section.

Scrambling/Fusion/Cryptography

After the data hiding process, the SPI'(5-8) has no semi-pixel, but blocks of eight and sixteen bits. The SPI(1-4) is composed of four bits. Thus, before the scrambling process, SPI'(5-8) is cut in four-bit portions shown in Fig.9. With this procedure, we avoid concentration of the embedded information in the resulted image and consequently, we improve the ciphering process and get a high entropy. Actually, the cutting is made automatically in the ciphering algorithm. It takes the left-half of the SPI'(5-8) byte and in the next step the right-half part, and so on.

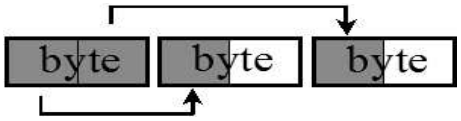


Fig. 9: SPI'(5-8) cutting.

The key has an important role in the encryption. It is divided in sub-keys of eight bits each. Then, these sub-keys are XORed with the pixels obtained from the coalition of SPI'(5-8) and SPI(1-4). The key is further used as seed of a PRNG (Pseudo-random number generator) and this PRNG produces numbers that will be employed in three parts of the process:

- To indicate the localization, in the final image, of the critical date for extraction purpose.
- To determine the semi pixel in SPI(1-4).
- To decide the inversion type.
  - Odd: (5-8) (1-4).
  - Even: (1-4) (5-8).

The first random number produced is used only in the extraction operation explained in Section .

The process of this section works as follow. It generates random numbers that are used to get semi-pixels in SPI(1-4). Then, if the random number is odd, the coalition is made in the order semi-pixel(5-8) and semi-pixel(1-4), else (1-4) and (5-8). The algorithm AES (Advanced Encryption Standard) also scrambles data (ShiftRows(), MixColumns()) to hinder unauthorized extraction [1]. Then, the proposed cryptography algorithm uses a part of the key to make a XOR operation with the previously fusion-generated pixel. Fig.10.

Since the size of SPI'(5-8) and SPI(1-4) are not similar, the process described in Fig.10 is made until the end of the smaller image. The remaining semi-pixels will be merged with others randomly chosen semi-pixels of the same image.

The final step of the ciphering process is the insertion of the critical data for the extraction algorithm. These data are inserted in eight extra bytes (64 bits). The first number generated by the PRNG is used to address these critical data into the final image. The smallest image accept is 4096 pixels (64X64). So, the first number created by PRNG is in this interval. The eight extra bytes are composed by:

- 1-bit for block-8 identification, B_8 or B_8'.
- 3- bits to precise the quantity of bits used for data hiding in blocks B_{16}.
- 2-bits for scan mode (row, column, spiral).
- 12-bits for the number of column of the original image.
- 12-bits for original image number of row.
- 16-bits for the size of embedded information.
- 18-bits for SPI'(5-8) size.

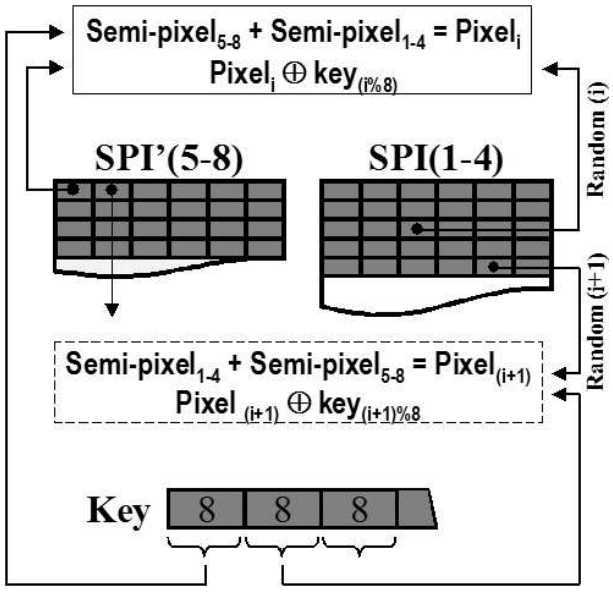


Fig. 10: Ciphering process.

Extraction

The extractor works as follow. It uses the secret-key as seed for the PRNG (Pseudo-Random Number Generator). The first generated number indicates the localization of the basic information necessaries for extraction. After, with the SPI'(5-8) size, the SPI(1-4) size (half of original image) and the secret-key, we can recreate SPI(1-4) and SPI'(5-8). Then, with H_{16}, message size and kind of block 8, we can extract the information that were embedded. Finally, we can reconstruct the original image without any loss.

EXPERIMENTAL RESULTS

The proposed crypto-data hiding methodology was tested in more than twenty gray-scale medical images. However, we present the results for the three bringing images, illustrated Fig. 11.a, 12.a and 13.a. They were ciphered with a 128-bit key and fully filled with patient information, Fig. 11.b, 12.b and 13.b. For the images, we show the histograms of the original images, Fig. 11.c, 12.c and 13.c, and their related cryto-data hidden pictures, Fig.11.d, 12.d and 13.d. They have presented very high entropy, around eight, and that certify our ciphering process.

The Tab.2 presents in details the complete outcome for each scan type and longest segment length for the image shown in 11.a. For this image the optimal length of the longest sequence for blocks B_8 is four and for blocks B_{16} is sixteen.

In Tab.3, we see a summary of the results for the three images above cited. It is notice that, the capacity of the proposed method depends largely on the characteristics of the original image.

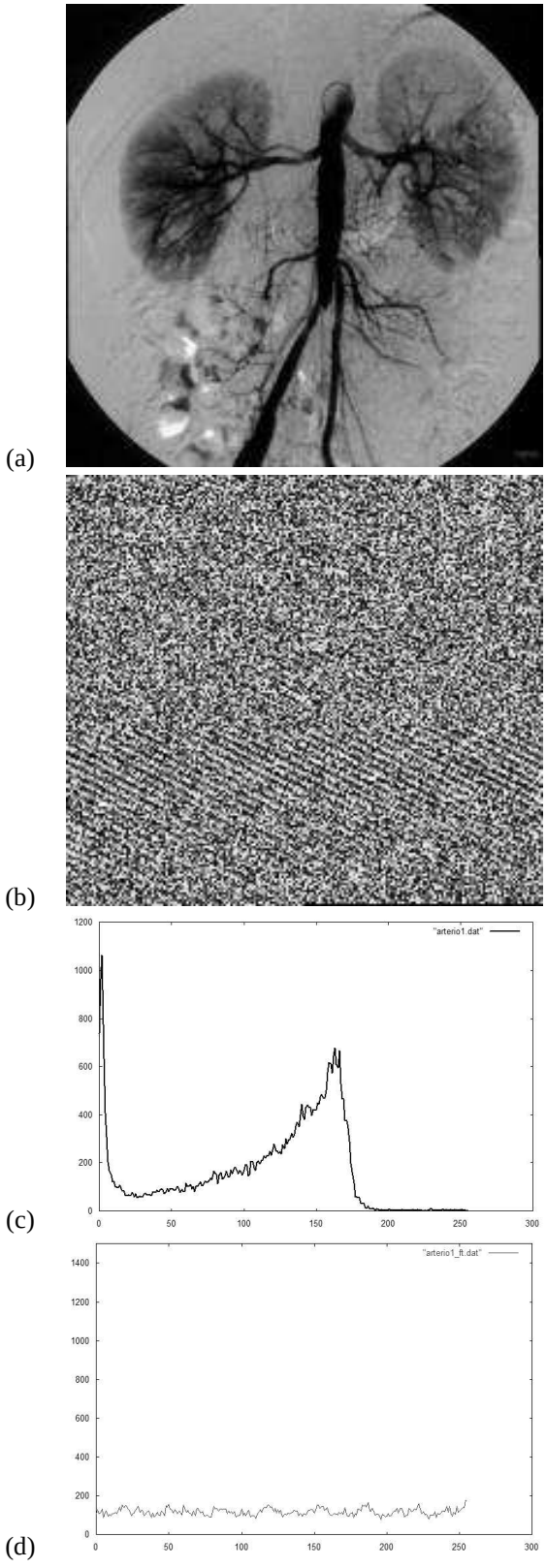


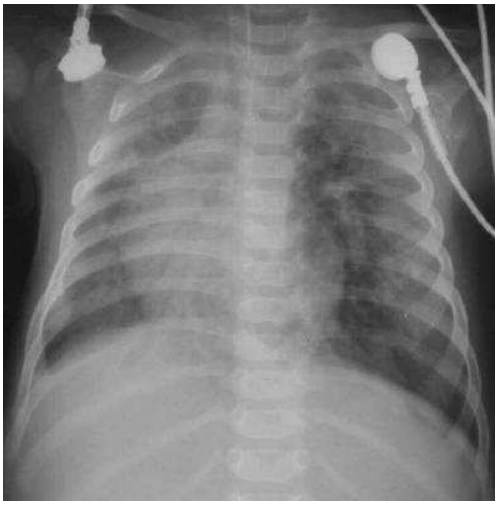
Fig. 11: a) Original image Arterio, b) Crypto-data hidden image, c) Original image histogram, d) Histogram of image (b).

Table 2: Processing summary for image arterio, Figure 11.a.

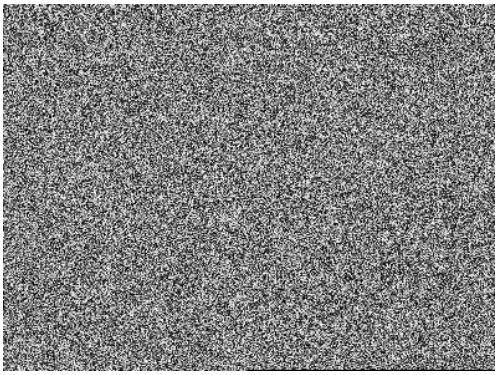
(1)	Trajectory type 0-Row, 1-Col, 2-Spiral
(2)	Longest Segment B_8
(3)	Longest Segment B_{16}
(4)	Quantity of B_8 (n_8)
(5)	Quantity of B_{16} (n_{16})
(6)	Bits per B_{16} for data hiding (H_{16})
(7)	SPI'(5-8) size in pixels
(8)	Data hiding space magnitude (W)

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
0	**4**	**16**	**13445**	**2082**	**7**	**17609**	**3502**
0	8	16	14571	956	7	16483	837
0	4	32	13365	1853	6	17071	3060
0	8	32	14430	788	6	16006	591
0	4	64	13350	1797	5	16944	2792
0	8	64	14406	741	5	15888	463
0	4	128	13346	1782	4	16910	2559
0	8	128	14398	730	4	15858	365
0	4	256	13346	1782	3	16910	2337
0	8	256	14398	730	3	15858	274
0	4	512	13346	1782	2	16910	2114
0	8	512	14398	730	2	15858	183
1	4	16	12350	2120	7	16590	3399
1	8	16	13483	987	7	15457	864
1	4	32	12254	1859	6	15972	2926
1	8	32	13327	786	6	14899	590
1	4	64	12242	1771	5	15784	2637
1	8	64	13302	711	5	14724	444
1	4	128	12242	1746	4	15734	2403
1	8	128	13298	690	4	14678	345
1	4	256	12242	1744	3	15730	2184
1	8	256	13298	688	3	14674	258
1	4	512	12242	1744	2	15730	1966
1	8	512	13298	688	2	14674	172
2	4	16	12866	2123	7	17112	3466
2	8	16	14029	960	7	15949	840
2	4	32	12798	1864	6	16526	2998
2	8	32	13919	743	6	15405	557
2	4	64	12781	1776	5	16333	2708
2	8	64	13887	670	5	15227	419
2	4	128	12779	1751	4	16281	2473
2	8	128	13879	651	4	15181	326
2	4	256	12779	1749	3	16277	2253
2	8	256	13879	649	3	15177	243
2	4	512	12779	1748	2	16275	2034
2	8	512	13879	648	2	15175	162

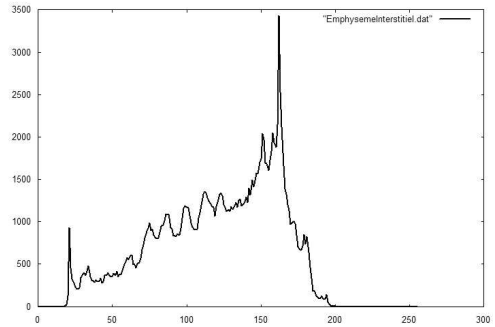
(a)



(b)



(c)



(d)

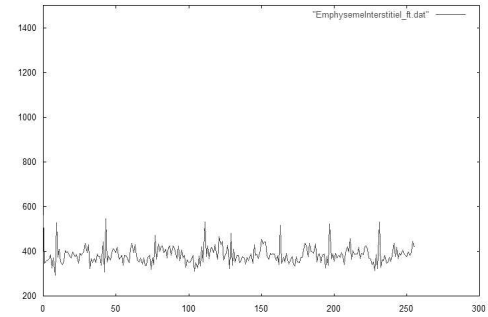


Fig. 12: a) Original image EmphysemeInterstitiel, b) Crypto-data hidden image, c) Original image histogram, d) Histogram of image (b).

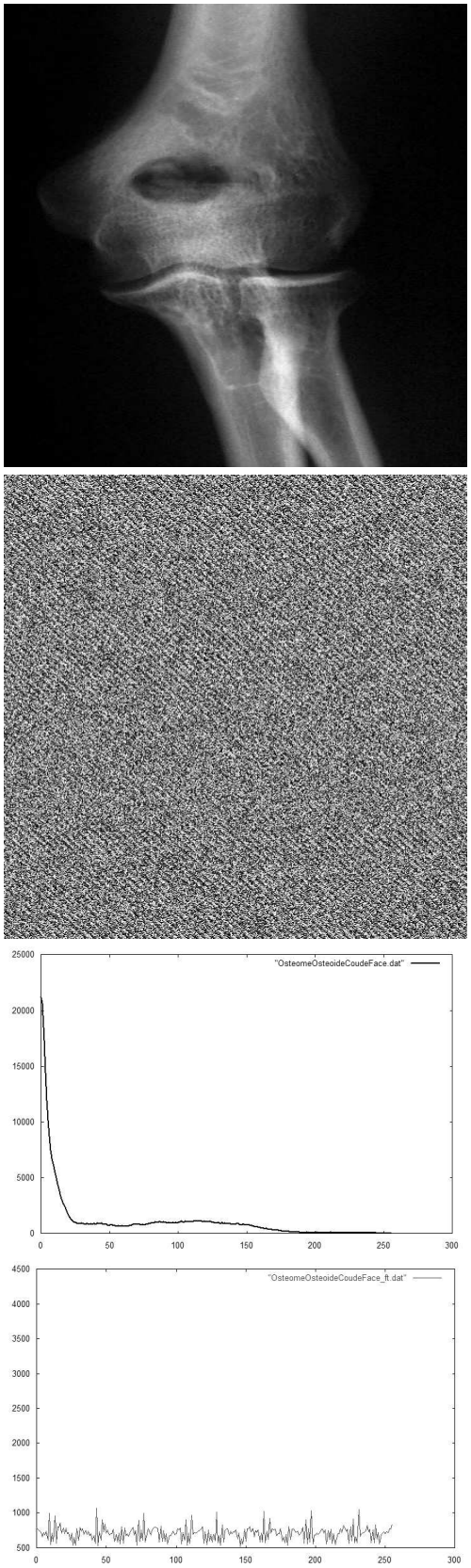


Fig. 13: a) Original image OsteomeOsteoideCoudeFace, b) Crypto-data hidden image, c) Original image histogram, d) Histogram of image (b).

Table 3: Image results. 1) Arterio, 2) EmphysemeInterstitiel 3) OsteomeOsteoideCoudeFace.

	Image (1)	Image (2)	Image (3)
Scan type	row	row	Column
n_8	13445	13090	33871
n_{16}	2082	12162	19125
Original Image size	41200	158400	289301
Size of final image	38209	116614	216772
W (bytes)	3502	12278	20968
% W	8.50%	7.75 %	7.25 %
Entropy of final image	7.953	7.995	7.984

CONCLUSION

In this paper, a reversible method for embed information of patients into medical images was presented. It is founded on image decomposition, RLE based-contents compression and optimal longest length sequence definition. The proposed method is able to hide information of the order of 8% of the original image size without increase it. The patient diagnostic, for example, can be embedded and encrypted with a key as long as relied, up to SPI'(5-8) size, and we have reached an entropy about 7.9 bits/pixel. In such scenarios, the embedding proposed in the current paper is significantly advantaged over conventional LSB embedding techniques in spatial domain. It offers a very efficient ciphering algorithm and complete recovery of the original image.

REFERENCES

[1] 197, F. I. P. S. P. Announcing the Advanced Encryption Standard (AES).

[2] B. SCHNEIER. Applied cryptography. In *Wiley* (New York, 1994).

[3] C.W. HONSINGER, P.W. JONES, M. R., AND STOFFEL, J. *Lossless recovery of an original image containing embedded data*. US Pat. 6,278,791, 2001.

[4] J. FRIDRICH. Applications of data hiding in digital images. In *ISPACS'98 Conference* (1998).

[5] J. FRIDRICH, M. G., AND DU, R. Lossless data embedding new paradigm in digital watermarking. In *EURASIP Journal Appl. Sig. Proc.* (Feb 2002), vol. 2002, pp. 185–196.

[6] S.S. MANICCAM, N. B. Lossless image compression and encryption using scan. In *Pattern Recognition* (2001), vol. 34, pp. 1229–1245.

[7] S.S. MANICCAM, N. B. Lossless compression and information hiding in images. In *Pattern Recognition* (2004), vol. 37, pp. 475–486.

[8] W. PENNEBAKER, J. M. Jpeg still image data compression standard. In *Van Nostrand Reinhold* (San Jose, USA, 1993).

[9] X. WU, N. M. Calic - context based adaptive lossless image codec. In *IEEE International Conference on Acoustics, Speech and Signal Processing* (May 1996), vol. 4, pp. 1890–1893.