Watermarking Based on Iterated Function Systems

Pedro A. Hernández Ávalos, Claudia Feregrino Uribe,
Roger Luis Velázquez, René A. Cumplido Parra
National Institute for Astrophysics, Optics and Electronics
Computer Science Department
Luis Enrique Erro No.1, Sta. María Tonantzintla, Puebla, México C.P. 72840
{phernandez,cferegrino,roger_luve,rcumplido}@inaoep.mx

Abstract

This paper presents an original approach for watermarking of digital images using Iterated function Systems (IFS) to generate positions maps used by Least Significant Bit method (LSB). The new approach exploits the main feature of fractals (generated by IFS): infinite magnification. The map generated by only one IFS can be used in images of different sizes. Furthermore, to avoid the image distortion by the embedding process, the data are inserted in non-homogeneous regions, to obtain this behavior, the Harris feature detector was modified. Obtaining a watermarking scheme robust to visual attack .

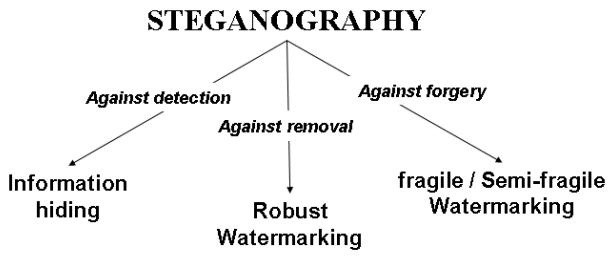


Figure 1. The classification for steganography.

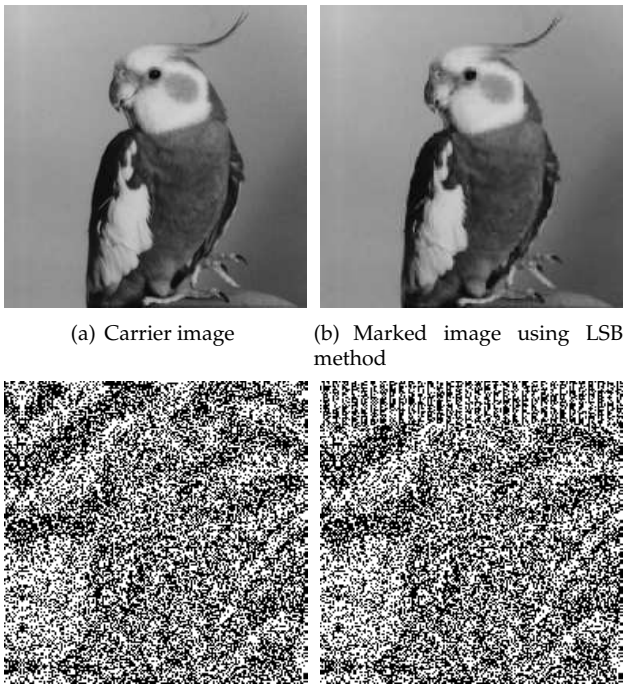
1. Introduction

The former name of watermarking is *steganography*, that comes from the Greek *stegano* which means *hide* and *graphos* which means *writing*. Together, steganography means literally *covered writing*. Today, this term steganography is not very popular, most people use the terms (*digital*) *watermarking*, *data embedding* and *information hiding* equally. Among them, watermarking is more recognized. However there is a classification for steganography proposed in [6] that depends on their specific usages, this classification is shown in figure 1.

The basic idea of information hiding is to cover a message in a medium, i.e. an image. It is very important that this message (or mark) can not be detected by any external subject. If an image with a hidden message (marked image or stegoimage) is analyzed (stegoanalysis), no modification must be detected even if it is minimal, this could imply the existence of a hidden message. In other words the objective of information hiding is to distort the image as less as possible to avoid the detection of the mark. On the other hand, robust watermarking algorithms are used to mark mediums that will be distorted for sure. Such algorithms have the function to resist to modifications or attacks such as: compression, geometric transformations, cropping, etc. Fragile or semifragile watermarking are used to ensure integrity or originality of the marked medium, ideally a simple modification on the stegomedium can be detected.

The LSB method hides the bits of the mark in the less significant bits of the image. This modification causes very little distortion at first glance, but if an image of the less significant bits plane is shown, as seen in figure 2, there are visible distortions on the image caused by the sequential embedding of the mark. There are methods to avoid this where a key generates the positions of the mark embedding, or even better, finds regions where these modifications are not noticed, i.e. edges or high standard deviation regions (non homogeneous regions). Image edges represent high standard deviation regions and they are very used to avoid visual attacks. It is worth to mention that such attacks are called *passive attacks* due to its purpose is to analyze

the image in search of information: position and message length. On the other hand, there are attacks called *active attacks* which manipulate the data of the image.



(a) Carrier image (b) Marked image using LSB method

(c) Least significant bits of the carrier image (d) Least significant bits of the marked image

Figure 2. Carrier image LSB plane vs marked image LSB plane

In this work a watermarking algorithm robust to visual attacks is presented. The method to hide information is LSB. Since the visual attacks work well in images where the message is inserted in homogeneous regions [7], the positions of the mark embedding are selected by the Harris and Stephens edge detector (non-homogeneous regions detector) and Iterated Functions Systems.

2. Harris and Stephens edge detector

The Harris and Stephens edge detector [3] is based on a corner detection function created by Moravec [5]. This function $E_{x,y}$ is calculated for a shift(x, y) from the central point (u, v):

$$E_{x,y} = \sum_{u,v} w_{u,v}[I_{x+u,y+v} - I_{u,v}]^2 \tag{1}$$

where $I_{u,v}$ represents the luminance of the image at the coordinate (u, v), the function $w_{u,v}$ represents a circular Gaussian window centered on (u, v). Harris reformulated the detection function using a matricial notation:

$$E_{x,y} = (x, y)M(x, y)^T \text{ con } M = \begin{bmatrix} A & C \\ C & B \end{bmatrix} \tag{2}$$

where:

$$A = X^2 * w \tag{3}$$

$$B = Y^2 * w \tag{4}$$

$$C = (XY) * w \tag{5}$$

with:

$$X = I * [-1, 0, 1] \approx \delta I / \delta x \tag{6}$$

$$Y = I * [-1, 0, 1]^T \approx \delta I / \delta y \tag{7}$$

where $*$ denotes the convolution operator.

To avoid computing the explicit eigenvalue decomposition α and β of M, the new criterion is based on the trace and determinant of M, thus, the corner and edge response function R is defined as:

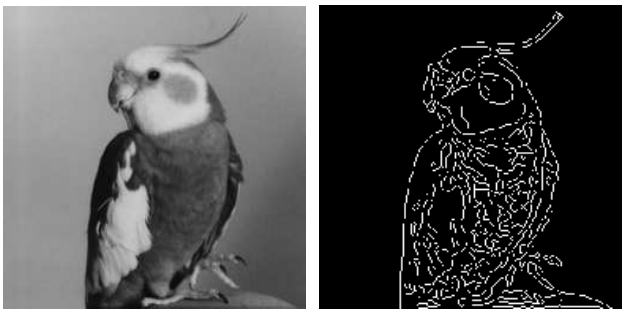
$$R = Det(M) - kTr(M)^2 \tag{8}$$

where:

$$Tr(M) = \alpha + \beta = A + B \tag{9}$$

$$Det(M) = \alpha\beta = AB - C^2 \tag{10}$$

k is a constant value, generally 0.04. The R matrix is positive in the corner regions, negative in the edge regions, and small in the flat regions. Extraction of feature regions is achieved by applying a threshold on the response R. The edges detected by applying this method on the *bird* (figure 3(a)) are shown in figure 3(b).



(a) Original image (b) Edges detected

Figure 3. The edges detected by Harris response.

3. Iterated Functions Systems

One way to create fractals is through Iterated Functions Systems(IFS). The performance of IFS is related to the photocopying machine example, proposed by Yuval Fisher [2]. Suppose that a special type of a photocopying machine reduces the image to be copied by a half and reproduces it three times on the copy, as seen in figure 4. It follows that the new image is copied in the same way once again, this generates an image of nine reduced versions of the original image. This procedure is iteratively repeated, in figure 5 [1] is shown this iteration in three occasions. It can be noticed that all images seem to be converging to the same final image, regardless of the original image. Also it can be observed that the resulting image is a copy of itself and is detailed in all scales. This resulting image is a fractal. This final image is known as *system attractor*.

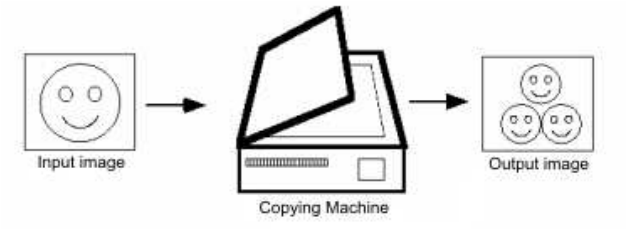


Figure 4. A copy machine that makes three reduced copies of the input image.

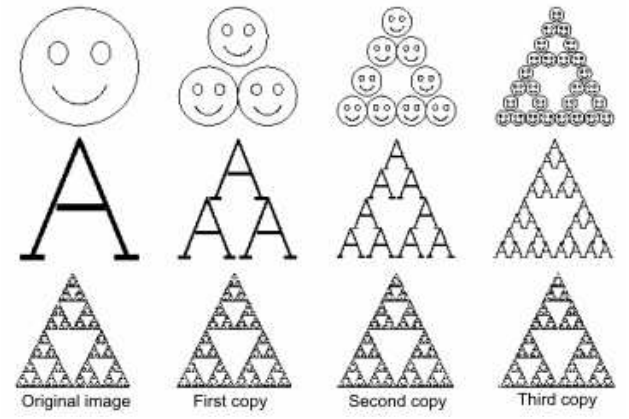


Figure 5. The first three copies generates on the copying machine of figure 4.

In this example, the photocopying machine represents a three function system and the feedback action represents the iterative part of the system. Each one of these functions contracts and transforms the input image and the three functions together create an output image with three different figures, which represents an Iterated Function System.

Each function of the IFS consists of an affine transformation, an each transformation consists of rotation, translation and escalation, this transformation affect each point of an input image. In other words, a point with coordinates (x, y) is translated to the coordinates (a, b).The equation that controls this transformation is:

$$\begin{bmatrix} a \\ b \end{bmatrix} = w\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{11}$$

or, which is the same:

$$\begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} ax + by + e \\ cx + dy + f \end{bmatrix} \tag{12}$$

The parameters that perform the action of rotation of each point are a, b, c and d, while their magnitudes correspond to the escalating factor. Parameters e and f are responsible for performing the linear translation in x and y of the same point. The general form of an Iterated Functions System is:

$$T_k(x) = \begin{bmatrix} a_k & b_k \\ c_k & d_k \end{bmatrix} x + \begin{bmatrix} e_k \\ f_k \end{bmatrix} \tag{13}$$

for $1 \leq k \leq n$, where x is a point in the $\Re^2$ plane and n is the total number of affine transformations. If T_k is a contractive mapping, then the attractor can be obtained through a set of iterated functions. In general an IFS can be classified as a deterministic IFS or a random IFS. A chaotic algorithm is used to generate the attractor or fractal instead of deterministic algorithm due to the low computational cost. Such chaotic algorithm was presented by M. Barnsley as the game of chaos in [1] and uses a set of probabilities $p = p_1, p_2, ..., p_n$, where p_k is the probability associated to T_k. In figure 6 a fractal image generated by a system of four contractive affine transformations or iterated functions with probabilities $p_1 = p_2 = p_3 = p_4 = 0.25$ is shown, as it can be seen in equations (14)-(17).

$$T_1(x) = \begin{bmatrix} 0 & 0 \\ 0 & 0.16 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{14}$$

$$T_2(x) = \begin{bmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix} \tag{15}$$

$$T_3(x) = \begin{bmatrix} 0.2 & -0.26 \\ 0.23 & 0.22 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1.6 \end{bmatrix} \tag{16}$$

$$T_4(x) = \begin{bmatrix} -0.15 & 0.28 \\ -0.26 & 0.24 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0.44 \end{bmatrix} \tag{17}$$

[1] Figures 4 and 5 are taken from [2].

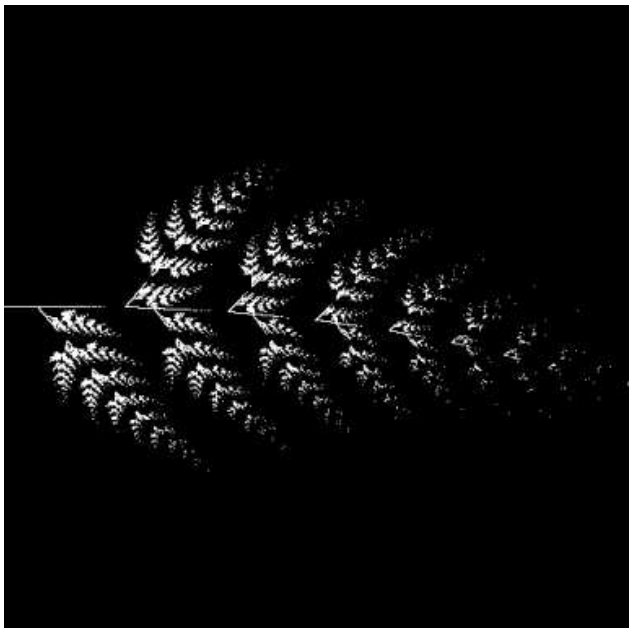


Figure 6. Fractal generated by a system of four iterated functions: (14)-(17).

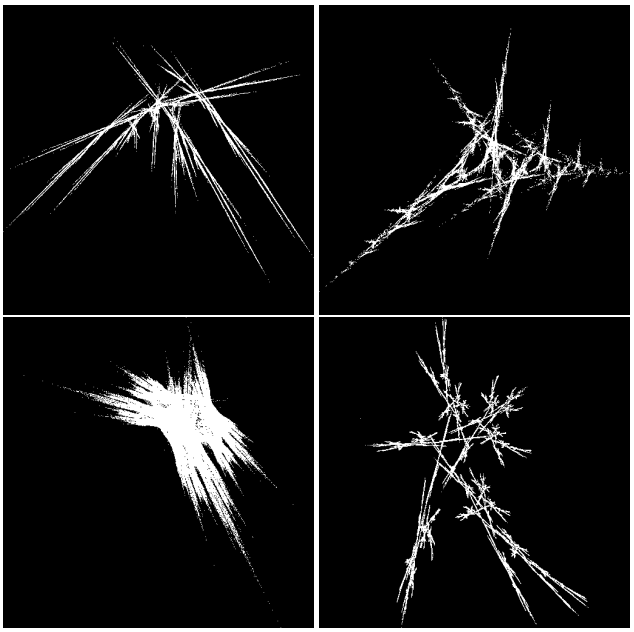


Figure 7. Fractal generated by random coefficients.

Figure 7 shows four fractals generated by a system of three iterated functions, the coefficients a, b, c, d, e and f of each IFS are randomly generated.

4. Proposed scheme

The proposed scheme consists of a method which uses a fractal image generated by an IFS and an edge detector for the selection of the positions where the mark will be embedded, the method LSB is the mechanism for embedding the mark. A detailed description of mark embedding process and mark extraction process follows. The carrier image of the mark is shown in figures 2 and 3.

4.1. Embedding process

The general embedding process is the following:

Step 1. Convert the mark to bits

For the LSB method it is necessary to convert the image to bits. The binary representation of the mark is shown in figure 8, in this case a monochromatic 1024 bits image is used as a mark but any kind of image or media can be used.

Step 2. Choose parameters $\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}\}$ for every affine transformation $\mathbf{T_k}$

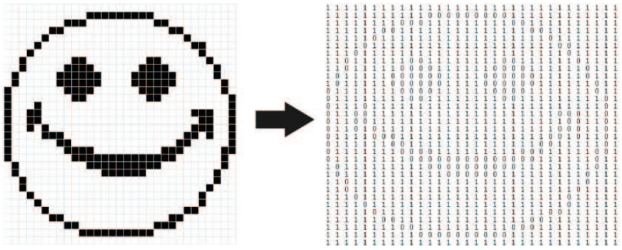


Figure 8. The binary representation of the mark.

The parameters of IFS have to be chosen for the mark embedding, the form of the fractal generated by IFS defines the *capacity* to embed in bits. Figure 9 shows a fractal generated by the following selected parameters (notice that this is a three iterated functions system):

$$a = \{0, 0, 0\} \quad (18)$$

$$b = \{0.577, 0.577, 0.577\} \quad (19)$$

$$c = \{-0.577, -0.577, -0.577\} \quad (20)$$

$$d = \{0, 0, 0\} \quad (21)$$

$$e = \{0.095, 0.441, 0.095\} \quad (22)$$

$$f = \{0.589, 0.789, 0.989\} \quad (23)$$

Step 3. Edge detection of original image

To improve robustness to visual attacks, edges of

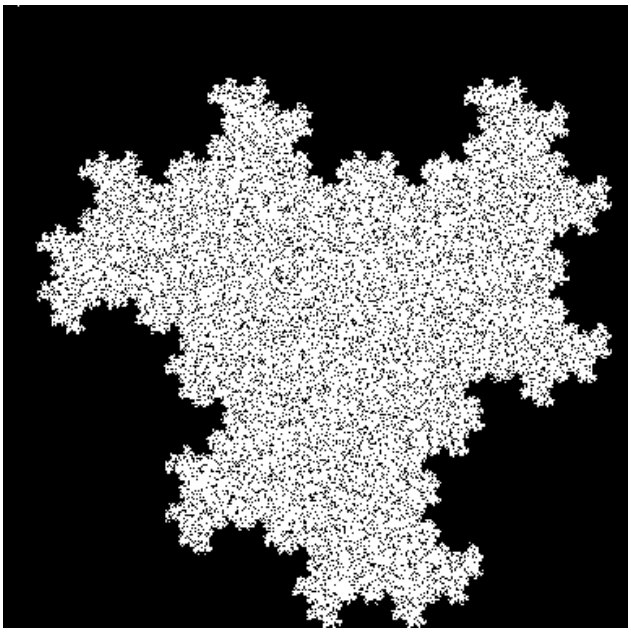


Figure 9. The ``dragon´´fractal generated by the functions (18)-(23).

the image are chosen for the modification of the less significant bit. Edges of an image represent high standard deviation regions which means that there is not uniformity on the pixels that form the image. In this case edge detection is performed by the Harris detector, but it can be performed by any other method. The detected edges are expanded to increase the capacity of the bits embedding as it is shown in figure 10. This process uses a window $w_{u,v}$ of size $n \times n$ centered on (u, v) and it is realized as follows:

1. The rectangular window $w_{u,v}$ is positioned on $I_{u,v}$, such that the coordinate (u, v) is element of the detected edge.

2. The standard deviation σ_w of the window $w_{u,v}$ is computed.

3. For each pixel (x, y) of the window $w_{u,v}$ the steps 1 and 2 are followed, obtaining n^2 standard deviations values (one for each pixel (x, y)).

4. The pixels (x, y) of w with standard deviation values greater than the standard deviation value of $w_{u,v}$ (main window) are selected.

5. The next edge pixel is selected and the steps 1-4 are performed. The process stops when no more edge pixels are left.

Figure 10. Edges detected by Harris detector and its expansion

Step 4. Locate embedding points of the mark

The pixels where the mark is embedded are generated by the intersection of the fractal image and the expanded edges image. The amount of selected pixels represents the capacity. In figure 11 the embedding pixels are shown, the capacity is 5176 bits. Notice that the capacity of the method for a given image depends on the selected IFS. Figure 12 shows the intersection of the IFS from figure 7 with the carrier image, it can be observed that the capacity depends on the selected IFS.

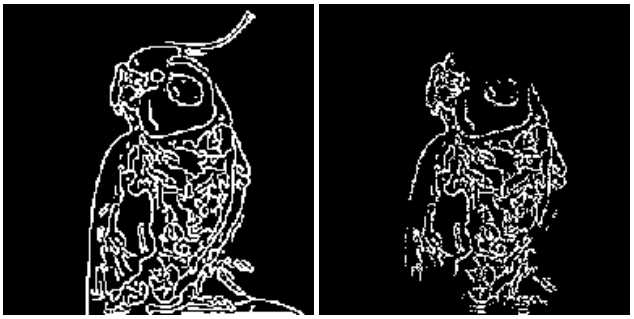


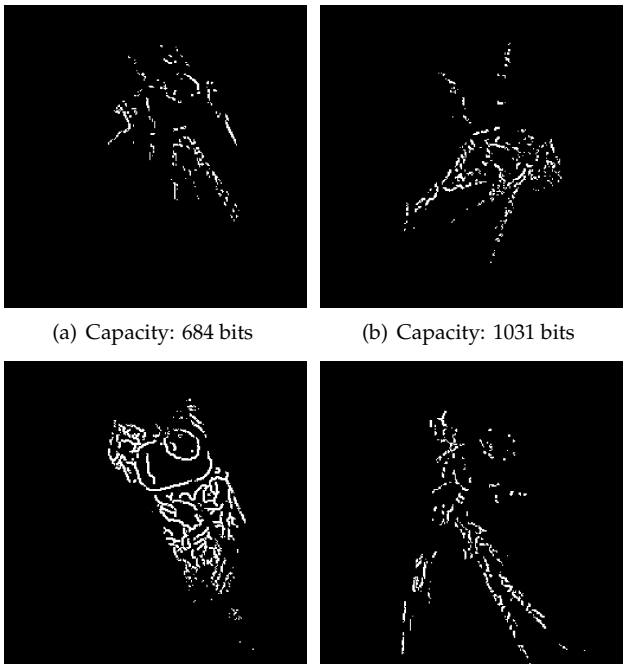
Figure 11. Pixels selected using the intersection of the expanded edges image with the fractal image. Capacity: 5176 bits.

Step 5. Mark embedding

Bits of the mark are embedded on the positions located in step 4 using LSB technique. The image with the embedded mark is shown in figure 13.

4.2. Extraction process

The extraction process of the mark is performed in a similar way as the embedding process. Step 1. Having parameters $\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}\}$ for every affine transformation T_k generate the corresponding fractal. Step 2.



(a) Capacity: 684 bits (b) Capacity: 1031 bits

(c) Capacity: 2516 bits (d) Capacity: 1193 bits

Figure 12. Capacities obtained by the intersection of the fractals on figure 7 with the carrier image.

Figure 13. Carrier image and marked image using the proposed approach.

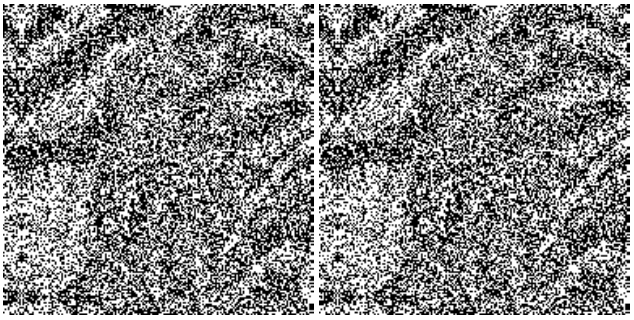


Figure 14. Least significant bits of the images in 13, Black for LSB=0, White for LSB=1.

Extract edges from the marked image and perform the edges expansion. Step 3. Locate extraction points of the marked image through the intersection of generated fractal with the expanded edges of the stegoimage. Step 4. Apply LSB method in the positions obtained in step 3 to obtain the mark.
Figure 14 displays the LSB planes of the original image and marked image.

5. Obtained results

The use of IFS to generate the positions where the mark will be hidden using LSB method, exploits the *chaotic* characteristic of the generated fractal. However, knowing this behavior, it can be used as a perfect key for information hiding. Using IFS as a key has an intrinsic characteristic: multiresolution. Thanks to it, information can be hided in different zones of images depending on their resolution due to the generated fractal is set to the resolution of the carrier image, having an infinite growth in points that pseudorandom functions do not have. The combination of obtained points by the fractal and the expanded edges of the image, makes difficult the detection of the embedded mark through any visual attack. The embedding and extracting process are similar, with the only difference that the mark is extracted or hidden from the plane LSB of the image, in locations found by the generated IFS fractal.

6. Conclusions

The fractal image generation through given parameters, needs a great amount of iterations to converge into an attractor, but at the same time, it provides non uniform randomness and it is independent of the image size. Besides, there is an increased robustness to visual attacks through the selection and expansion of edges. The capacity of the method depends on the stegomedium characteristics and the fractal selected. Bits were embedded into the image in a consecutive way, however this pattern can be modified to spread the mark in the selected edges. To do this can be performed a pseudorandom function, as an example: making a chaotic function as shown in [4] or just by modifying the IFS generator function to be used as a position pseudorandom generator. Also, as future

work, the message can be hidden in a different plane of the image, giving greater robustness to other types of attacks.

Acknowledgments: The authors acknowledge to CONACYT the support provided through the grants with numbers 201778 and 207086 for MSc. studies. The first author thanks to Jennifer V. Marquina Pérez for her valuable comments.

References

[1] M. Barnsley. *Fractals everywhere*. Academic Press Professional, Inc., San Diego, CA, USA, second edition, 1993.

[2] Y. Fisher. *Fractal Image Compression: theory and application*. Springer-Verlag, New York, USA, 1995.

[3] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[4] R. Matthews. On the derivation of a chaotic encryption algorithm. *Cryptologia*, VIII(1):29–41, 1984.

[5] H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. In *tech. report CMU-RI-TR-80-03, Robotics Institute, Carnegie Mellon University and doctoral dissertation, Stanford University*. September 1980. Available as Stanford AIM-340, CS-80-813 and republished as a Carnegie Mellon University Robotics Institue Technical Report to increase availability.

[6] C. D. Vleeschouwer, J. F. Delaigle, and M. B. Invisibility and application functionalities in perceptual watermarking: An overview. *Proceedings of the IEEE*, 90(1):64–67, January 2002.

[7] A. Westfeld and A. Pfitzmann. Attacks on steganographic systems. In *IH '99: Proceedings of the Third International Workshop on Information Hiding*, pages 61–76, London, UK, 2000. Springer-Verlag.