Steganography Using Modern Arts
(Extended Abstract)

Carlo Blundo and Clemente Galdi

Dipartimento di Informatica ed Applicazioni
Università di Salerno
84081, Baronissi (SA), Italy
{carblu,clegal}@dia.unisa.it

Abstract. In this paper we present a novel approach to information hiding. We investigate the possibility of embedding information using in some way the naturally pseudorandomness of some classes of cover-documents. In particular we provide algorithms for embedding any binary string in an image belonging to a particular class of images, the image mosaics. The algorithms presented allow different levels of security for the information hidden in the cover-document. We also show some techniques to reduce the amount of information the users have to secretly store.

1 Introduction

One of the main differences between cryptography and steganography is that each cryptographic algorithm maps a document, the plaintext, into a ciphertext that *must* be pseudo-random. Indeed it can be shown that if an encryption scheme is deterministic then it cannot be considered secure (see [3] for a more detailed description). We notice that we do not make any assumption about the document that will undergo the encryption, in the sense that we can encrypt a text as well as a binary file, that already "seems" pseudo-random.

On the other hand, each steganographic scheme hides information, the embedding-string into a document, the cover-document, without relevantly altering the information contained in it. This means that if we hide information in a non-random document, e.g. a text, the resulting document must be a a non-random document while if we hide information in a pseudo-random document, e.g. a compressed file, the result must be pseudo-random, e.g. a compressed file. In particular, the output document of the steganographic scheme must contain almost the same information of the input one.

These difference could be actually seen as the impossibility of using any cryptographic primitive in steganographic schemes.

The basic observation we make in this work is that we can use a pseudorandom-document as input of an encryption scheme and as cover-document in a steganographic scheme and both the schemes must return a pseudo-random document as output. The major problem is that steganography wants to hide to an attacker

J. van Leeuwen et al. (Eds.): IFIP TCS 2000, LNCS 1872, pp. 140–151, 2000.

the fact that an information is hidden in a document too. So, if the attacker sees that Alice sends to Bob a pseudo-random document, he will immediately detect that there are some hidden information in it. This could be avoided by properly choosing the cover-document to use. In [1] the authors show how to embed any file in a raid of pseudo-random cover-files, without relevantly altering their pseudo-randomness. On the other hand, we notice that the authors "forced", in a certain sense, the cover-documents to be pseudo-random. Indeed, no disk drive stores the information by randomizing them. This means that an attacker can immediately detect that some information are hidden in the disks.

Thus, we are searching for a class of documents that are "naturally" pseudo-random and such that their non-randomness is immediately recognizable or, in any case, checkable. This means that if we use a compressed file as cover-file, the output of the embedding function must be a compressed file too, i.e. there must exist some (public) program that correctly uncompress it. Usually, when dealing with images, the "public program" is the human visual system.

A lot of papers presents results in steganography using images as cover documents. The schemes presented in these works, embeds information into images by slightly modifying them in such a way that the human eye cannot see the difference. This is done by modifying some information in the image representation, or in its transformed representation.

Much work in the image steganography has been done on digital watermarking. The interest in these techniques is due to the growing need of copyright protection in the internet. The goal of image digital watermarking is to embed some private information into an image. This embedding procedure must satisfy two main requirement. First: the embedding procedure should not relevantly alter the original image. Second: there must exist a function that checks (or retrieves) that some information have been embedded in an image. Watermarking techniques must also guarantee the impossibility of changing the information embedded in an image without relevantly altering the image itself. This property actually states for the impossibility of changing the "ownership" information that the embedded information carries. The latest property is actually hard to asses since the images can undergo a lot of manipulation, like scaling, resizing, cropping, and so on.

One of the simplest method to hide information in an image is to alter in some way the least significant bits in a bit plane representation of the image. Examples of this techniques are [18,19]. However, these LSB methods are vulnerable in the sense that unauthorized parties could simply recover the information hidden in an image.

Due to this problem the authors in [5,15,17] developed some techniques that require the original cover-image for the retrieving phase of the information.

Another approach has been used by the authors in [14,4]. The technique described in these papers selects some pixel of the image using a pseudo random generator, and alter in some way their luminance.

In this paper we consider a particular class of pseudo-random images the Image mosaics described in the next section. We shall show that it is possible to use this class of images as cover-documents in a steganographic scheme.

The paper is organized as follows: In Section 2 we describe image databases and photomosaics. In Section 3 we give the algorithms for embedding and extracting information in an image mosaics. In Section 4 we present future works to be done.

2 Preliminaries and Notations

In this section we briefly review the terminology that we will use through this paper.

2.1 Image Databases

With the growth of world wide web, people have now access to tens of thousands of digital images, that can be collected in large databases. One of the basic problems in this case is to retrieve particular images from these databases.

According to the classic automatic information retrieval paradigm, a database is organized into documents that can be retrieved via synthetic indices, in turn organized in a data structure for rapid lookup and retrieval. The user formulates his information retrieval problem as an expression in a query language. The query is translated into the language of indices, the resulting index is matched against those in the database, and the documents containing the matched indices are retrieved.

One possible solution to the indexing problem should be to identify each image with one or more keywords. However, such an approach should avoided for some reasons. First of all, the classification should be done interactively, and this will not be, of course, a good strategy for large databases. This is not the only problem. As pointed out by the authors in [10], describing some pictorial properties of some images should be difficult, and, at the same time, some other properties should be described in different ways.

In the last years have been developed some techniques that allow the automatic indexing of image databases using “keywords” related to the represented images. Roughly, these techniques extract some information from the image and use these information as “keywords” for image indexing and lookup. A simple example of automatically extracted keyword is the hash value of the image. Of course, in these cases, the query that the user will formulate is an image itself. For more detailed description of content-based image retrieval systems see [10, 8,11]

2.2 Mosaics of Images

A classical mosaic is a set of small coloured stones arranged in such a way that if they are seen from away, they compose a larger image. This artistic expression is based on the property of the human visual system that "sees" the colour of a region as the average colour of that region. Indeed, if the distance between two points is below a certain threshold, the human eye sees only one point whose colour is the average colour of the original ones.

The Photomosaics[1] created by R. Silvers are based on the same property (see [16] for a more detailed description). Photomosaics are mosaics in which the coloured stones are substituted by small photos. Silvers wrote a computer program that, starting from a database of small photos, called tiles images, catalogs these images according to some characteristic, like colour, shape, contrast, and many other. To create the photomosaic of a target picture, the software divides the target into macro-pixel and then substitutes each of them with an image in the database that best matches its characteristic. An example of Photomosaic[2] is presented in Figure 1.

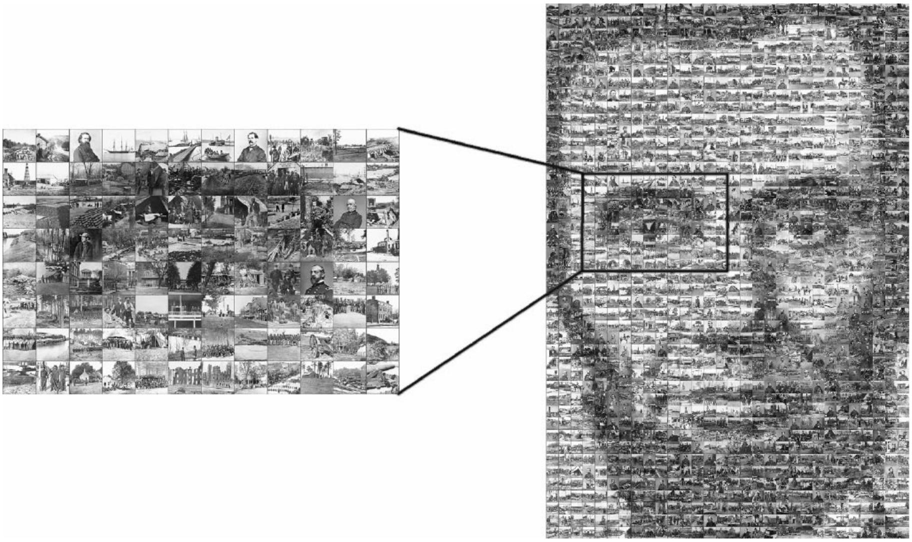

Fig. 1. Photomosaic

Other examples of image mosaics can be found in [6]. In this paper the authors describe a process to automatically generate image mosaics. The major difference between the Silvers' photomosaics and the image mosaics presented in [6] is the size of the tiles. An example of image mosaic is given in Figure 2.

Notice that currently the generation of the image mosaics is not completely automatic. This class of images is considered as an artistic expression. Thus,

[1] Photomosaic is a trademark of Runaway Technology.

[2] Image by courtesy of Runaway Technology.

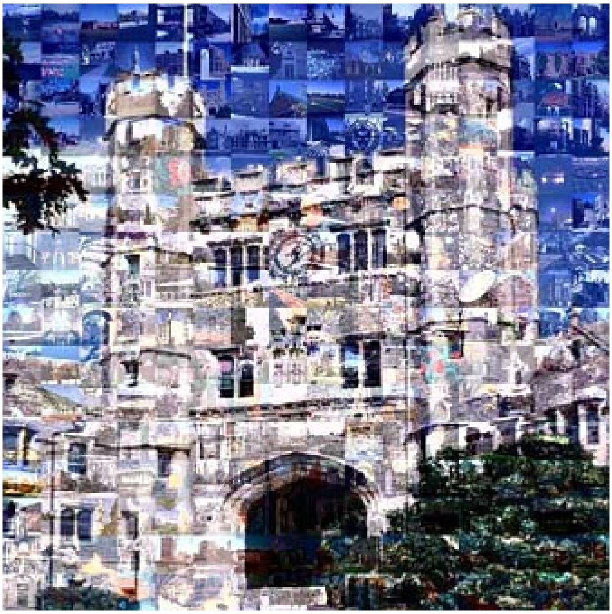

Fig. 2. Image Mosaic

the authors automatically generate a first "draft" of the image mosaic and then modify it by hand in order to meet some personal criteria.

Another crucial point for automatic generation of the image mosaics is the size of the tiles. Indeed, if the size of tiles grows too much with respect the size of the image, the mosaic generated could not faithfully represent the target image (e.g., the target image resolution will decrease).

3 The Schemes

3.1 A General Scheme

In the previous section we have described what a mosaic of photos is. Here we will describe a scheme that allows to embed any secret information in such an image. For a sake of simplicity, we suppose that the mosaic $\mathcal{M}$ is a black and white picture, i.e., each tile of the mosaic can be seen as encoding a black pixel or a white one. We will later describe a generalization of this technique to the case of coloured mosaics.

More formally, we can see a mosaic $\mathcal{M}$ as an array of n tiles images, $M_1, \ldots,$ M_n. Each picture M_i can be classified as belonging to one of the two classes of images $\mathcal{C}_0$ or $\mathcal{C}_1$, where $\mathcal{C}_0$ (resp., C_1) contains images that encodes a white pixel (resp., a black pixel). We suppose that Alice and Bob have agreed upon the two classes of tiles, i.e., they both hold these two databases of images and they both hold a classification algorithm $Class$, that on input M_i, outputs an index j such that $M_i \in \mathcal{C}_j$. For the sake of simplicity, suppose that these databases have the same size $s = 2^t$, for some t (we will relax this assumption later). Moreover Alice and Bob have a standard ordering rule on each class of images, i.e., there exist four functions, known to both the players, $f_j : \{0, 1, \ldots, s-1\} \to \mathcal{C}_j$, the embedding function, and $g_j : \mathcal{C}_j \to \{0, 1, \ldots, s-1\}$, the extracting function, with $j = 0, 1$, such that for each picture P in $\mathcal{C}_j$, it results $f_j(i) = P \iff g_j(P) = i$.

Let $\mathcal{X}$ be the message Alice wants to send to Bob and let $x_1, \ldots, x_\ell$ be its binary representation, where $\ell = vt$ for some integer $v \leq n$. In the following, Int, is a function that takes in input a binary string b and returns as output an integer whose binary representation is b, and Bin is its inverse function. Alice will use the algorithm presented in Figure 3.

```
Algorithm Embed($\mathcal{M}$,$\mathcal{X}$)

 1. Partition $x_0, \ldots, x_{\ell-1}$ in $v$ blocks, $B_0, \ldots, B_{v-1}$, of $t$ bits each, i.e.,

                          $B_i = x_{ti}x_{ti+1} \ldots, x_{ti+t-1}$

    Notice that each block is the binary representation of an integer in $\{0, 1, \ldots, s\}$.
 3. for $i = 1$ to $v$
    3.1 Let $j = Int(B_i)$
    3.2 Let $c = Class(M_i)$
    3.2 Substitute $M_i$ with $f_c(j)$.
 4. Output $\mathcal{M}$
```

Fig. 3. The Embedding Algorithm.

It can be easily seen that information that the mosaic represents, i.e., the image encoded in the mosaic, does not change, since the transformation Alice applies to the mosaic simply substitute a picture of the mosaic with another picture of the same class. It could be roughly thought as changing a bit zero in a binary string with another bit equal to zero too.

The algorithm Bob has to use to "decode" the mosaic is the presented in Figure 4.

It is immediate to see that if $v < n$, then Bob will decode the message $\mathcal{X}$ with some random elements R appended to it. To avoid this, Alice could append to $\mathcal{X}$ some special termination symbol. Moreover, notice that the scheme presented is deterministic. This means that, if $\mathcal{X} = 0^\ell$, where $\ell = vt$, then the scheme will substitute the first v tiles in the mosaic with the tiles encoding 0^t in C_0 or C_1, depending on the class the substituted tile belongs to. Of course, this situation should be avoided because, if $v < n$ the resulting mosaic will immediately reveal

```
Algorithm Extract($\mathcal{M}$)

1. for $i = 1$ to $n$
    1.1 Let $c = Class(M_i)$
    1.2 Let $e = g_c(M_i)$.
    1.3 Let $B_i = Bin(e)$
2. Output $\mathcal{X} = B_1, \ldots, B_n$.
```

Fig. 4. The Extracting Algorithm.

to an attacker that something strange is going on. In Sections 3.3 and 3.4 we will show how to solve this problem by randomizing the embedding scheme.

We now to evaluate the maximum length of $\mathcal{X}$ that it is possible to embed in a mosaic. We suppose that the mosaic is composed by n tiles and that the sets $\mathcal{C}_0$ and $\mathcal{C}_1$ contain s images. Since each picture "encodes" $\log s$ bits of $\mathcal{X}$, we can embed in mosaic $n \log s$ bits. We call this quantity the *capacity* of the mosaic $\mathcal{M}$.

3.2 Implementing the Function f and g

It is not hard to see that one of the basic ingredient of the algorithms are the embedding and extracting functions f and g. Recall that these function are defined as follows:

$$f_j : \{0, 1, \ldots, s-1\} \rightarrow \mathcal{C}_j \qquad g_j : \mathcal{C}_j \rightarrow \{0, 1, \ldots, s-1\}$$

A simple way to implement these functions is to "embed" them in the image database by adding to each image a field containing a (unique) number in $\{0, 1, \ldots, s-1\}$ in a preprocessing phase.

This simple process assures that to each image in a certain class is associated an unique identifier that ranges from 0 to $s-1$. Thus the f's implementation actually is a query to $\mathcal{C}$ with key equal to the input of f. On the other hand, the g's implementation is a query to $\mathcal{C}$ too. The difference between these two requests is that for the function g, the query is an image.

3.3 Hiding Private Information

In the previous sections we have shown that it is possible to embed any binary string of length ℓ in a mosaic composed by n pictures, if holding at least two classes of images with s images each and if ℓ is upper bounded by $n \log s$. The secrecy is guaranteed by the fact that an attacker should not know the database, the embedding and extracting functions. We have also shown how to simply implement these function by "embedding" them in the database itself.

This technique forces the users to locally (and privately) maintain a copy of the image databases. Notice that, there should exist some embedding and extracting functions that do not need the database modification to be implemented, (the functions should extract information from the directly from the images). If this is the case, the database could be public, but, for the privacy of the scheme presented in the previous section, the functions must still be private.

We put ourselves in a weaker condition in which the functions are implemented using some characteristic of the single image, i.e. do not depend on the database, and, at the same time, are public. To assure the privacy, we will actually embed in the mosaic an encrypted form of the message.

More precisely, an encryption scheme is a pair $(\mathcal{E}, \mathcal{D})$ of probabilistic, polynomial time algorithms where $\mathcal{E}$ is the encryption algorithm and $\mathcal{D}$ is the decryption algorithm. Recall that, for a probabilistic encryption scheme to be secure, the ciphertexts obtained by different encryption of the same plaintext must be different. Moreover, the ciphertexts obtained from the encryption must be computationally indistinguishable from a random string (see [3,7]) The encryption (resp., decryption) algorithm takes as input an encryption (resp., decryption) key k and the plaintext (resp., the ciphertext) and outputs the ciphertext (resp., the plaintext).

Let $\mathcal{X} = x_1, \ldots, x_\ell$ be the message Alice wants to send to Bob. The users will exchange the message $\mathcal{X}$ using the algorithms presented in Figure 5.

```
Algorithm Secure_Embed($\mathcal{M}$,$\mathcal{X}$,k)

1. $\mathcal{Y} = \mathcal{E}(k, \mathcal{X})$
2. Output Embed($\mathcal{M}$,$\mathcal{Y}$)
```

```
Algorithm Secure_Extract($\mathcal{M}$,k)

1. $\mathcal{Y}$=Extract($\mathcal{M}$)
2. Output $\mathcal{D}(k, \mathcal{Y})$
```

Fig. 5. Secure Embedding and Extracting Algorithms

Even though the database and the functions are public, an attacker can extract $\mathcal{Y}$ (a pseudo-random string) using the Extract algorithm. If the encryption scheme is secure, than it is impossible for the attacker to infer information about the message $\mathcal{X}$ we have embedded in the image mosaic. Moreover, since $\mathcal{Y}$ looks completely random, the attacker cannot tell if some information has been hidden into the mosaic. Using this approach, of course, the users must share a common key k and this is the only information that the users must keep secret.

3.4 Enhancing the Security of Embedding Scheme

The protocols presented in the previous sections embeds information in an image mosaic $\mathcal{M}$ $=M_1, M_2, \ldots, M_n$ by substituting the first v tiles in the image, for some $v \leq n$. This means that, if an attacker believes that some information has been embedded into a mosaic, he is sure that these information are hidden in these tiles.

We have also shown how to embed private information by encrypting the message before the modification of the image. The protocol presented, in this case, does not modify the sequence of tiles changed by the embedding algorithm, even if, the amount of information that the users keep secret dramatically decreases.

The security of the scheme presented in Section 3.3, is based on the security of the underlying encryption scheme. Another simple way to improve the security of the scheme presented, is by allowing the algorithm to select the tiles to change in a pseudo-random way. As the receiver must be able to extract the information embedded in the image mosaic, the users should share a second key, k_2 ad a random seed r, that will be used in a pseudo-random number generator. This will assure the correctness of the decoding.

```
Algorithm Random_Embed($\mathcal{M}$,$\mathcal{X}$,r,$k_1$,$k_2$)

1. Let $\mathcal{Y} = \mathcal{E}(k_1, \mathcal{X})$.
2. Partition $\mathcal{Y}$ in $v$ blocks, $B_1, \ldots, B_v$, of $t$ bits each, padding zeros if necessary, i.e.,

                    $B_i = y_{ti} y_{ti+1} \ldots, y_{ti+t-1}$

3. Let Used=$\emptyset$, i=0.
4. Let $Q = q_1, q_2, \ldots, q_m$ be a sequence of $m$ bits generated by a pseudo random
   number generator on input key $k_2$ and random seed $r$.
5. while $i < v$
   5.1 Let $b$ be the first $\lceil \log n \rceil$ unused bits in $Q$.
   5.2 Let $j = Int(b)$
   5.3 If $j \notin$Used
       5.3.1 Substitute $M_j$ with $f_{Class(M_j)}(B_i)$.
       5.3.2 Used=Used $\cup \{j\}$.
       5.3.3 $i \leftarrow i + 1$
6. Output $\mathcal{M}$
```

Fig. 6. The Embedding Algorithm with Pseudo Random Selection.

In this way, even if the attacker knows the database of the images, the embedding and the extracting function, he cannot extract any information from the image mosaic since he does not know which is the sequence of the elements he has to decode.

Notice that the algorithms presented in Figure 5 can be combined with the one presented in this section to obtain a more "powerful" scheme, presented in Figure 6, in which the user embeds the encrypted form of the message in the image mosaic using randomly selected tiles of the original mosaic. The extracting scheme can be easily obtained by the previous one and is thus omitted.

3.5 A Note on Coloured Mosaics

In the previous sections we have discussed the embedding of private informtion in image mosaics and we have assumed that the original image is a black and white one.

However, this requirement is not necessary at all. Indeed it is possible to extend the embedding schemes presented to work with coloured mosaics.

The scheme will consider $d > 2$ classes of images $\mathcal{C}_0, \ldots, \mathcal{C}_{d-1}$ and will substitute a tile in the mosaic with another one belonging to the same class, as described in the previous sections.

4 Future Work

In the previous sections, we have analyzed some techniques allowing to embed information in image mosaics. We have based our approach on an existing class of images for which there already exists software for automatic generation.

In this section we point out the possibility to use the same techniques presented in Section 3 on a different class of cover-images. The scheme we present here does not need any database private memorization.

We now present the scheme for B/W images. Consider the cover-image as an $n \times m$-bit matrix and suppose, for a sake of simplicity, that $n = wh$ and $m = wv$, for a fixed w and for some v and h. It is thus possible to consider original cover-image as a $h \times v$ matrix of macro-pixel, each of which is a $w \times w$ matrix of pixels. A macro-pixel is said to encode black if more than $w^2/2$ pixels are black, otherwise it is said to encode white.

We can construct a "database" of macro-pixel in the following way:

- A white macro-pixel is a $w \times w$ matrix of pixels in which all but one pixels are white;
- A black macro-pixel is a $w \times w$ matrix of pixels in which all but one pixels are black.

Each macro-pixel, can be associated to a integer in the set $\{0, 1, \ldots, w^2 - 1\}$.

In Figure 7, we report a possible construction for the case of $w = 2$ with the associated integer for each macro-pixel.

Notice that not all the values of w are admissible. Indeed, as each macro-pixel encodes a value in $\{0, 1, \ldots, w^2 - 1\}$, $t = \lceil \log w^2 \rceil$ bit are necessary to represent this value. On the other hand, there exists no macro-pixel that encode the value in the set $\{w^2, w^2 + 1, \ldots, 2^t - 1\}$. This means that, if we use t bits for the embedding, there exist some binary string that cannot be injected into the mosaic. If we encode $t-1$ bit in each macro-pixel, there will be some macro-pixel that will be never used. This means that w must be equal to some power of 2.

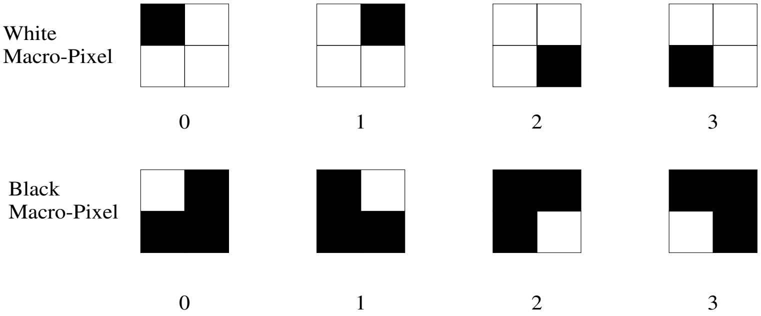

Fig. 7. Macro-Pixel Construction

Indeed, in this case, $w^2 = 2^s$ and thus we can encode in each macro-pixel exactly s bits.

It is important to remark that w's growth corresponds to a degradation of the image. We have presented a macro-pixel construction in which all but one pixels have the same colour. Of course it is possible to think to different macro-pixel constructions. For example it should be possible to require that all but two pixels have the same colour. This will, of course, raise the number of bits each macro-pixel encodes, but the image quality could decrease.

5 Conclusion

In this paper we presented a novel approach to data hiding in images. We have identified a class of images that is particularly indicated for this task. We also given some algorithms for for information hiding that allow different levels of security. The algorithms presented can be composed in order to meet the security level required.

While the algorithm presented in Section 3 are secure and do not alter the information the cover-image represents, the idea presented in Section 4 should be verified in order to give an experimental result on the image degradation.

Acknowledgements

The authors want to thank Alfredo De Santis and Paolo D'Arco for enlighting discussion that leads to this research, Robert Silvers and Runaway Technology for providing the Photomosaic in Figure 1. The authors also thanks the anonymous referee for helpful comments about the presentation of the paper.

References

1. R. Anderson, R. Needham, and A. Shamir. Steganographyc file system. In *Proceedings of 2nd Workshop on Information Hiding*, pages 73–82, 1998.
2. R.J. Anderson. Stretching the limits of steganography. In *First Workshop on Information Hiding*, pages 39–48, 1996.
3. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption: Analysis of the des modes of operation. In *Proceedings of 38th IEEE Symposium on Foudations of Computer Science*, 1997.
4. W. Bender, D. Gruhl, N. Morimoto, and A. Lu. Techniques for data hiding. *IBM System Journal*, 3 and 4:313–336, 1996.
5. I.J. Cox, J. Kilian, T.Leighton, and T. Shamoon. Secure spread spectrum watermarking for images, audio and video. In *Proceedings of IEEE International Conference on Image Processing*, pages 243–246, 1996.
6. A. Finkelstein and M. Range. Image mosaics. In R.D. Hersch, J. André, and H. Brown, editors, *Proceedings of EP'98 and RIDT'98 Conferences*, pages 11–22, 1998.
7. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
8. C.E. Jacobs, A. Finkelstein, and D.H. Salesin. Fast multiresolution image querying. In *Proceedings of SIGGRAPH 95*, 1995.
9. A.J. Menenez, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997.
10. W. Niblack, R. Barber, and W. Equitz et. al. The qbic project: Querying images by content using color, texture, and shape. In *Storage and Retrieval for Image and Video Databases, SPIE*, pages 173–187, 1993.
11. M. E. Ogle. Chabot: Retrieval from a relational database of images. *Computer*, 28(9):40–48, 1995.
12. F.A. Peticolas, R.J. Anderson, and M.G. Kuhn. Information hiding - a survey. *Proceedings of the IEEE*, 87(7):1062–1078, July 1999.
13. B. Pfitzmann. Information hiding terminology. In *Proceedings of 1st Workshop on Information Hiding*, pages 347–350, 1996.
14. I. Pitas. A method for signature casting on digital images. In *International Conference on Image Processing*, volume 3, pages 215–218, 1996.
15. C.I. Podilchuk and W. Zeng. Digital image watermarking using visual models. *Human Visual and Electronic Imaging II*, 3016:100–111, 1997.
16. R. Silvers. Photomosaics. *http://www.photomosaic.com.*
17. M.D. Swanson, B. Zhu, and A.H. Tewfik. Trasparent robust image watermarking. In *Proceedings of IEEE International Conference on Image Processing*, volume III, pages 211–214, 1996.
18. R. van Schyndel, A. Tirkel, and C. Osborne. A digital watermarking. In *Proceedings of IEEE International Conference on Image Processing*, pages 86–90, 1994.
19. R.B. Wolfgang and E.J. Delp. A watermarking for digital images. In *Proceedings of IEEE International Conference on Image Processing*, pages 219–222, 1996.