AN ALGORITHM FOR AUTHENTICATION OF DIGITAL IMAGES

Dan Dumitru Burdescu, Liana Stanescu
Faculty of Automation, Computers and Electronics,University of Craiova, Bvd. Decebal, Craiova, Romania

Keywords: Watermarking, Robustness, Multimedia, JPEG, Algorithm.

Abstract: The rapid growth of digital multimedia technologies brings tremendous attention to the field of digital authentication. The owner or the distributor of the digital images can insert a unique watermark into copies for different customers or receivers, which will be helpful to identify the source of illegal copies. In digital watermarking, robustness is still a challenging problem if different sets of attacks need to be tolerated simultaneously. In this paper we present an original spatial authentication technique for digital images. Our approach modifies blocks of the image by insertion of a spatial watermark. A spatial mask of suitable size is used to hide data with less visual impairments. The watermark insertion process exploits average color of the homogeneity regions of the cover image. The complexity of the algorithms is proved to be $O(n^2)$, where 'n' is the nodes number of virtual graph for watermark. The authentication method developed below works for all types of digital image.

1 INTRODUCTION

In recent years, the rapid and extensive growth in Internet technology is creating a pressing need to develop several newer techniques to protect copyright, ownership and content integrity of digital media. This necessity arises because the digital representation of media possesses inherent advantages of portability, efficiency and accuracy of information content. On the other hand, this representation also implies a serious threat of easy, accurate and illegal perfect copies in unlimited numbers. Unfortunately, the currently available formats for image, (also audio and video) in digital form do not allow any type of copyright protection. A potential solution to this kind of problem is an electronic stamp or digital watermark, which is intended to complement cryptographic processing. The later techniques facilitate access of the encrypted data only for valid key holders but fail to track any reproduction or retransmission of data after decryption. On the other hand, in digital watermarking, an identification code is embedded permanently inside a cover image, which remains within that cover invisibly even after decryption process. Digital watermarking is now considered an efficient technology for copyright protection.

Several image watermark schemes have been developed in the past few years, both spatial and frequency domains being used for watermark embedding. *Spatial watermarks* are constructed in the image spatial domain, and embedded directly into an image's pixel data (Burdescu, 2004).

General requirements for watermarking techniques, in general, need to posses the following characteristics: (a) imperceptibility for hidden information, (b) redundancy in distribution of the hidden information inside the cover image to satisfy robustness in the watermark extraction process even from the cropped watermarked image; and (c) possible use of one or more keys to achieve cryptographic security of hidden content (Katzenbesser, 2000, Lee and Jung , 2001).

Besides these general properties, an ideal watermarking system should also be resilient to insertion of additional watermarks to retain the rightful ownership. The perceptually invisible data hiding requires insertion of the watermark in higher spatial frequency of the cover image since the human eye is less sensitive to this range of frequencies. But in most of the natural images the majority of visual information is concentrated on the lower end of the frequency band. Thus the information hidden in the higher frequencies components might be lost after quantization

Dumitru Burdescu D. and Stanescu L. (2006).
AN ALGORITHM FOR AUTHENTICATION OF DIGITAL IMAGES.
In *Proceedings of the International Conference on Security and Cryptography*, pages 303-308
DOI: 10.5220/0002099403030308

operation of losing compression (Hsu, 1999). This motivates researchers to realize the importance of perceptual modeling of the human visual system and the need to embed a signal in perceptually significant regions of an image, especially if the watermark is to survive losing compression (Cox, 1997). In the spatial domain block based approach this perceptually significant region is synonymous of low variance blocks of the cover image.

Since the meaning of multimedia data is based on its content, it is necessary to modify the multimedia bit-stream to embed some codes, i. e. watermarks, without changing the meaning of the content. The embedded watermark may represent either a specific digital producer identification label, or some content-based codes generated by applying a specific rule. Because the watermarks are embedded in the data content, once the data is manipulated, these watermarks will also be modified such that the authenticator can examine them to verify the integrity of the data. For complete verification of uncompressed raw multimedia data, watermarking may work better than digital signature methods because: (a) the watermarks are always integrated with the data such that the authenticator can examine them conveniently, and (b) there are many spaces in the multimedia data to embed the watermarks without degrading the quality too much (Yeung, 1997). However, there is no advantage to use the watermarking method in a compressed multimedia data for complete verification. Compression standards e.g. JPEG or MPEG have user-defined sections where digital signatures can be placed. Because multimedia data are stored or distributed in the file format instead of pixel values, therefore the digital signature can be considered as being "embedded" in the data. For content verification, a watermarking method that can reliably distinguish compression from other manipulations still has not been found. The watermarks are either too fragile for compression or too flexible for manipulation.

So far, the robust watermarking systems found in the literature can only withstand some of the possible external attacks but not all. The attacks against the watermark try to neutralize the watermark, without damaging the image too much. The watermark is neutralized if: (a) the detector cannot detect the watermark (distortion, attenuation etc.), (b) the detector cannot recognize the watermark in the image from another one, and (c) the watermark is no longer in the image. The attacks can be very different: (a) in the spatial domain, it can be scaling, cropping, rotation, noise addition. The open source software STIRMARK available on the Web generates many of these attacks, (b) in the frequential domain it can be filtering, (c) compression and (d) adding another watermark over the first one. The STIRMARK software generates random rotations and distortions on blocks of the image. The STIRMARK software simulates JPEG coding, filtering operations, rotation, scaling and cropping. The result is very slight alterations on image, but watermarks are usually heavily damaged.

The present paper describes a computationally efficient block-based spatial domain authentication technique for a one level watermark symbol. The selection of the required pixels is based on variance of the block and watermark insertion exploits average color of the blocks. The proposed algorithms were tested on a few of the most usual transformations of images and the obtained results showed that the proposed method is efficient. The authentication method developed below works for all types of digital image and it can be applied in medical domain because it can be inserted into images immediately when the image is obtained by a medical apparatus.

2 AUTHENTICATION ALGORITHMS

All watermarking methods share the same building blocks – an embedding system and the watermark extraction or recovery system (Hsu et al., 1999). Any generic embedding system should have as inputs: a cover data/image (I), a watermark symbol (W) and a key (k) to enforce security. The output of the embedding process is always the watermarked data/image (I'). The generic watermark recovery process needs the watermarked data (I'), the secret key (K) and depending on the method, the original data (I) and/or the original watermark (W) as input while the output is the recovered watermark with some kind of confidence measure for the given watermark symbol or an indication about the presence of watermark in the cover image under inspection. The original cover image I is a standard image of size $N \times N$ where $N = 2^p$ with a 24 bit RGB format (in medical domain the image is read in a **.bmp** format). In the proposed work a binary image of size 256×256 or 512×512 is considered. Each image is marked with a watermark coefficient. That means, for each pixel, the value of the pixel is changed according to the given formula:

$$D(i,j) = C(i,j) + a*M*W \quad (1)$$

where C(i,j) is the original value of a pixel at position (i,j); D(i,j) is the watermarked value of the same pixel; *a* is a scalar factor (here *a* is chosen constant, but can be a variable of the position to improve the invisibility of the watermark and its detection); M is the mean of the block; and W is the watermark coefficient to be embedded. In our work, W could take the values +1 or –1 (one can easily extend the implementation to M).

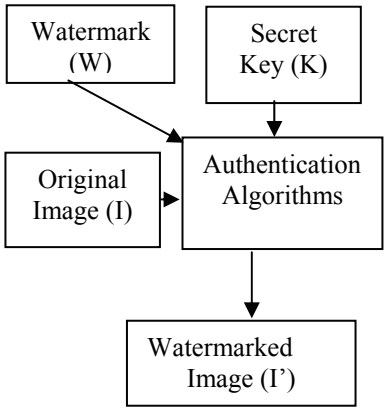

Figure 1: Block diagram of a watermark embedding system.

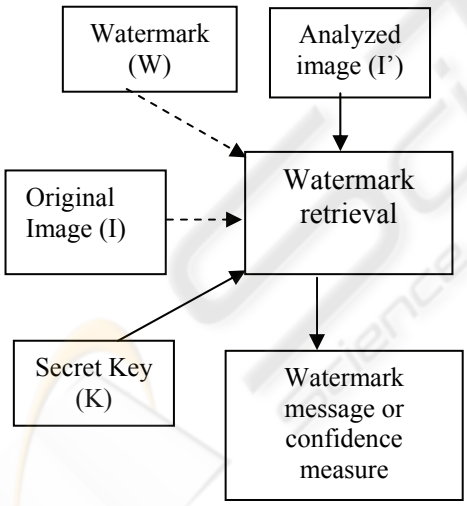

Figure 2: Generic watermark recovery scheme.

The pixels of the image are arranged into a virtual hexagon. Then the image is viewed as a graph not as a pixel matrix. The vertices represent the pixels and the edges represents neighborhood between pixels. The algorithm for this operation is as following:

```
PROCEDURE Construct_Graph(Image I,edge)
BEGIN
 for * i->0,width/edge – 3*edge
     for * j->0;height/3
       if (i modulo 3==0)
          then * if (jmodulo 2 ==0)
               then
             bmap[i][j]=bmp.bmap[edge*i
             ][edge*j+edge-1];
             end if;
               * if(jmodulo 2==1)
               then
            bmap[i][j]=bmp.bmap[edge*i
            ][edge*j+edge+2];
            end if;
        end if;
        if (i modulo 3==1)
            then * if (jmodulo 2 ==0)
                 then
            bmap[i][j]=bmp.bmap[edge*i
            -1][edge*j-edge];
            end if;
                  * if (jmodulo 2 ==1)
                 then
            bmap[i][j]=bmp.bmap[edge*i
            -1][edge*j+edge*2];
            end if;
         end if;
         if (i modulo 3==2)
             then * if(j modulo 2==0)
             then
             bmap[i][j]=bmp.bmap[edge*i
             -2][edge*j+edge-1];
                   end if;
                   * if(j modulo 2==1)
                  then
            bmap[i][j]=bmp.bmap[edge*i
            -2][edge*j+edge+2];
                 end if;
         end if;
     end for j;
  *output the graph g
 end for * i;
 END
```

As it is said, the image is arranged into hexagons having different dimensions of edges.

Proposition 1

The total running time of a call of the procedure **Construct_Graph** (Image I, edge) is O(n*m), where "n" is the width and "m" is the height of the digital image.

Proof

It can be observed that the first FOR loop of the algorithm is executed at most once for a pixel of the width of image. Hence, the total time spent in this loop is O(n). The second FOR loop processes the pixels on the height. Hence, the total time spent in this loop is O(m). So, the total time spent in these

loops is O(n*m), because are processed all pixels of the image at most once.

From previous statements is inferred that the total running time of this procedure is O(n*m)

For introducing the watermarked key W will be considered certain nodes from the graph (i, j), that may be selected to represent a letter, a number or a function. In these nodes are changed the three color channels of the considered pixel depending on the three color channels of the bottom pixel minus one or another constant (const). It may consider a started node having the coordinates (m0, n0) where it is started the marking process and an ended node where the process of marking is ended. In this way the watermarked key is spread on entire image, not only into a selected block of image (like in other methods).

The algorithm for marking the image graph is shown bellow:

```
PROCEDURE  mark_graph (Graph bmap)
 BEGIN
   *choose 2 nodes in the graph
    (m0,n0) and (m1,n1)
    for * i->m0,n0
        for * j ->m1,n1
    change_color(i, j, const)
         end for j;
    end for i;
 END
```

For reconstructing the marked image should be verified only the graph's nodes corresponding to the selected key. If the marked pixel has the color in the interval [*min, max*] with respect to the color of the bottom pixel, then it will consider that this pixel was marked in conformity with the given algorithm. The values *min, max* resulted from a lot of experiments or from the nature of the application.

Proposition 2

The total running time of a call of the procedure **Mark_Graph** (Graph bmap) is $O(n^2)$, where "n " is the number of nodes of graph attached to the image.

Proof

It can be observed that the first FOR loop of the algorithm is executed at most once for each node of the graph. Hence, the total time spent in this loop is O(n). The second FOR loop processes the pixels of node which has the same color of its neighbor. The inner FOR loop processes the nodes of unvisited neighbor. So, the total time spent in these loops is $O(n^2)$, because are processed all nodes of the graph at most once. From previous statements is inferred that the total running time of this procedure is $O(n^2)$

3 EXPERIMENTAL RESULTS

There are a lot of transformation that can be done on images (Lin, 2001), (Fei, 2004): rotation, redimension, compression (transforming the image to JPEG), cropping and of course the case in which the image is not changed. Because it is not known what transformation the user did, all these transformations are verified one –by – one and the percentage of similitude between the original image and the verified one is returned. If the image is not transformed, the algorithm presented below is applied:

```
PROCEDURE   detect_untransformed (Image
I,int edge,int w,int h, int  percent)
 BEGIN
           *construct_graph (I,edge );
             count=0;
       for * i->m0,n0
         for * j->m1,n1
            if   (color(bmap[i][j])   ==
             color(bmap[i+1][j])  -1)
                      count = count +1;
        end if;
              end for j;
        end for i;
         *output  percent  of  similitude
         between   the   original   marked
         image and the image verified;
 END
```

Also this algorithm may be applied if the image is cropped. In this case it may be possible to lose some marked pixels depending on the position where the image was cropped.

In the next figure, the first image is the image marked and the second one is the image marked and cropped. The detection algorithm detects this cropped image in percent of 88.88%.

Figure 3: Experiment 1.

In the case of image rotation (angle of 90, 180 and arbitrary), are verified all the image nodes because the nodes' position is changed. We search the nodes for which all of the three color' channels have the values like the color's channels of the bottom pixel minus one (or a certain constant). Before verifying these nodes, the image is

dimensioned again to the initial dimensions at which the image is marked.

In the next figure, the first image is the marked image and the second is the image marked and rotated by 30 degree.

Figure 4: Experiment 2.

```
PROCEDURE  detect_rotated (Image I,int
edge,int  w,int  h,  int   percent,int
constant)
  BEGIN
    *redimension(I,w,h);
     *construct_graph (I,edge );
       count=0;
       for * i->0,w
    for * j->0,h
           if  (color(bmap[i+1][j])    -
           constant<=color(bmap[i][j])
           <=        color(bmap[i+1][j])
           +constant)
              then count=count+1;
            end if;
       end for j;
         end for i;
    *output   percent   of   similitude
   between  the  original  marked  image
   and the image verified;
  END
```

By experiments, there resulted that in the case of rotation by 90 and 180 degree, the constant is zero, but in the case of rotation by arbitrary angle the constant is very great (100).

The marked image was rotated with different angles. From experiments resulted the following percentage of similitude between the marked image and the marked rotated image, as shown in the following table.

Table 1: Results 1.

Rotation Angle	Similitude Percent
30	33.33%
60	33.33%
90	100%
180	100%

The implemented authentication algorithm entirely detects an image that was transformed to JPEG, only if the image is compressed with a quality of 100% and 80%. For image processes (compression, decompression), was used ADOBE PHOTOSHOP. The variable m0, n0, m1, n1, h, w are known , being the same variables that are used in the process of image marking. The color of pixels arranged into nodes selected by us for marking the image is changed because of transformations supported by the image. Then the color of these pixels is searched into a certain interval.

```
PROCEDURE     detect_jpg   (Image   I,int
percent, int constant)
  BEGIN
       *decompressed(I);
       *construct_graph (I, edge);
       count=0;
       for * i->m0,n0
          for * j->m1,n1
                if (color(bmap[i+1][j])
       -constant <=color(bmap[i][j]) <=
       color(bmap[i+1][j]) +constant)
             then count = count +1;
          end if;
       end for j;
         end for i;
    *output   percent   of   similitude
   between  the  original  marked  image
   and the image verified;
  END
```

From experiments results that a good value for this constant is 30. Using different degree of image compression for JPEG, there are resulted the following percents of similitude between the marked image and the JPEG marked image.

Table 2: Results 2.

Quality	Similitude Percent
100	100%
80	100%
60	33.33%
50	33.33%
30	0%
10	0%

In the case when we want to detect an image that was enlarged the results are weaker:

```
   PROCEDURE   detect_larged (Image I,
int edge,int w,int h, int  percent,int
constant)
    BEGIN
   *redimension(I,w,h);
   *construct_graph (I,edge);
   count=0;
   for * i->0,m
     for * j->0,n
           if (color(bmap[i+1][j]) -
           constant <=color(bmap[i][j])
           <= color(bmap[i+1][j])
           +constant)
          then count = count +1;
          end if;
   end for j;
            end for i;
          *output percent of similitude
   between the original marked image
   and the image verified;
     END
```

From experiments results that a good value for this constant is 70.

4 CONCLUSION

Watermarking, as opposed to steganography, has the additional notion of robustness against attacks. Even if the existence of the hidden information is known it is difficult for an attacker to destroy the embedded watermark, even if the algorithmic principle of watermarking method is public. In cryptography, this is known as Kerkhoffs law: - a cryptosystem should be secure, even if an attacker knows the cryptographic principles and methods used but does not have the appropriate key (Hartung, 1999).

Meanwhile, the number of image watermarking publications is too large to give a complete survey over all proposed techniques.

Software watermarking is the process of embedding a large number into a program so that: (a) the number can be reliably retrieved after the program has been subjected to program transformations, (b) the embedding is imperceptible to an adversary and (c) the embedding does not degrade the performance of the program.

The method developed above satisfies the necessary requests for the authentication technique and the series of presented transformations accounts for the fact that it resists possible attacks. The method is easy to implement and the experimentally determined robustness shows that it can be used without fear of being detected or changed. In addition, the authentication method can be easily used for marking of medical digital images with other information besides those for watermarking. The watermarked image is not a new one because the number of the transformed pixels is very little (n < < number of the image pixels, where 'n' is the number of nodes of virtual graph). In addition, the pixels of the watermarking are slightly changed thus the human eye cannot discern them.

REFERENCES

Burdescu D.D. and Stanescu L., 2004. A Spatial Watermarking Algorithm for Digital Images. In *Control Engineering and Applied Informatics Journal*, vol. 6, no. 3, pp. 57-63,.

Katzenbesser S., Petitcolas F.A.P., 2000. *Information Hidden Techniques for Steganography and Digital Watermarking,* Artech House, Boston, MA,.

Hsu C.T., Wu Ja-L., 1999. Hidden Digital Watermarks in Images. In *IEEE Transaction on Image Processing*, No. 8, pp 58-68,.

Cox I. J., Kilian J., Leighton T., Shammon T., 1997. Secure Spread Spectrum Watermarking for Multimedia. In *IEEE Transaction on Image Processing*, No. 6, pp 1673-1687,.

Yeung M., Mintzer F., 1997. An Invisible Watermarking Technique for Image Verification. In *IEEE International Conf. on Image Processing*, Santa Barbara.

Lin C., Wu M., Lui Y. M., Bloom J. A., Miller M. L., Cox I. J., 2001. Rotation, Scale amd Translation Resilient Public Watermarking for Images. In *IEEE Transaction on Image Processing*, Vol. 10, No. 5, pp 767- 782,.

Fei C., Kundur D., Kwong R., 2004. Analysis and Design of Watermarking Algorithms for Improved Resistance to Compresion. In *IEEE Transaction on Image Processing*, Vol. 13, No. 2, pp 126 – 144,.

Hartung F., Kutter M., 1999. Multimedia Watermarking Techniques. In *Proceedings of the IEEE*, vol. 87. N0. 7,

Lee S.-J., Jung S.-H., 2001. A Survey of Watermarking Techniques Applied to Multimedia, IEEE Int.'l Symp. Industrial Electronics, IEEE Press, pp. 272-277.