Compression Tolerant Watermarking for Image Verification

Harpal S. Bassali, Jatin Chhugani, Saurabh Agarwal, Alok Aggarwal, and Pradeep Dubey

IBM India Research Lab
Block IA, Indian Institute of Technology
New Delhi, INDIA 110017
Email: pkdubey@in.ibm.com
Tele: 91-11-686-1100, Fax: 91-11-686-1555

ABSTRACT

Digital Watermarking is seen as a viable solution to authentication of multimedia data and hence its security, especially in a networked environment. In this paper we present a new watermarking technique to add a code to digital images in spatial domain. This technique is further shown to be robust under common compression schemes, including lossy compression schemes. Watermark embedding is done keeping in mind the limitations of Human Visual System. We have used a procedure for error diffusion so as to minimize the chances of image tampering. For the verification of the image, original uncorrupted image is not required. Experimental results show that the watermark is robust to JPEG compression.

KEYWORDS

Watermarking, image authentication, image compression, compression-tolerance.

INTRODUCTION

The increasing sophistication of multimedia processing tools has made digital manipulation of photographic images and other digital media such as audio and video incredibly easy to perform. The essence of a watermark is to detect whether an image has been tampered or not. A highly desired feature of any digital watermarking system for still images is that it should survive compression occurring in file format conversions, especially, lossy compression. Additionally, automatic detection of the location of tampering is also useful. Further more, in order to maintain the quality of the original image, the watermark should also be invisible to human visual system. The watermark information should preferably be within the watermarked image, and not as a separate attachment. This simplifies the maintenance of large watermarked image databases. Finally, the computational cost of watermarking should be low. In this paper, we present a novel digital watermarking scheme which embeds an invisible watermark inside an image. The watermark survives lossy image compression.

BACKGROUND

Many watermarking techniques have been proposed, mainly focusing on the invisibility of the watermark and its robustness against various signal manipulations and hostile attacks. One group of techniques work in the spatial domain, for example, changing the LSB of some pixels, recording the difference between randomly selected pairs of points [7] and so on. These techniques can suffer from signal compression and hostile attacks. In some of these techniques the watermarking process uses the existing public-key encryption technology. Another group of techniques code the undetectable digital signatures onto an image by applying linear addition of the watermark to the image data using m-sequences. Some of the techniques hide simple data in images by patchwork, i.e., embedding one bit of data in a host image using statistical approach, as well as by Texture Block Coding, i.e., using blocks of random textures to replace those regions of similar textures to create an identical pair of textured regions upon which the shape is recovered by auto-correlation manner. A disadvantage of spatial domain watermarks is that picture cropping can be used to eliminate the watermark.

Spatial watermarking can also be applied using color separation, so the watermark appears in only one of the color bands. This renders the watermark difficult to detect under normal viewing. However, the watermark appears immediately when the colors are separated for printing. This renders the document useless for printing unless the watermark can be removed from the color band.

Another group of techniques work in the frequency domain and add a watermark by manipulating various frequency elements [7], since frequency domain techniques are much more robust against compression and geometrical transformations than spatial domain techniques. However, most of these frequency domain approaches have not taken into account the Human Visual System (HVS) when selecting positions to insert the watermark. Because of the invisibility constraint of a watermark, these techniques have to use signals of relatively lower power than would otherwise be possible,

0-7803-6297-7/00/$10.00

to avoid degrading the image quality, inevitably limiting the robustness of the watermark. For a good review of various watermarking techniques refer to [1, 2, 3]. A recent survey of watermarking techniques for tamper-proofing and authentication can be found in [4].

ALGORITHM OVERVIEW AND EMPIRICAL BASIS

Any watermarking scheme embedding the signature information in the image itself requires a storage mechanism to represent the signature information. This storage mechanism becomes even more crucial when the watermark is to survive lossy compression schemes. We have developed a novel storage mechanism and tested it out for compression tolerance, especially with respect to JPEG. The standard JPEG code used for the derivation of the empirical results on this storage mechanism was obtained from [5]. The horizontal and vertical frequency for all the components was chosen to be 1. We have worked with thousands of images and have found out that, if we break up an image into its Y, U and V components, then the average value of any $m \times n$ pixel block (where m and n are large) in any of the Y, U or V component planes after compression and decompression does not change by a significant amount in most of the blocks. When $m=n=10$, we found that the value did not change by more than ± 1.5 in 99.8% of the blocks with compression ratios of 20 to 30. As we increase the values of m and n, the chances of the average value changing become even more remote. For details refer to [6].

Since the average value does not vary a lot, hence a quantization is done of the complete average value range so that a value belonging to one quantized interval stays in that interval even after compression and decompression. Each of these quantized intervals is used to represent a bit. Through addition of error-correction bits, we ensure that the data embedded in this manner will not be lost even in the rare case when the average of a block changes beyond the interval. This gives us a definite method of data embedding in the image.

Each block of size mxn is called a microblock and a collection of microblocks called a macroblock. We calculate the average value of the macroblock, encode this value as a sequence of bits, add bits for error correction and embed one bit into each one of the constituent microblocks. At the verification stage, we extract the value embedded in the macroblock and compare it with the actual average value of the same macroblock. For reasons given above, we say that the image has not been tampered with if these values are close. Otherwise, the algorithm discovers tampering and it can point out the region of manipulation, i.e., the tampered macroblock.

ALGORITHM FOR WATERMARK EMBEDDING

A brief description of the algorithm steps follows:

1. *Split* the image into Y, U, V planes. Divide the planes into non-overlapping blocks of $m \times n$ pixels starting from the top left corner of the image. Henceforth, these blocks are called *microblocks*. A random offset is left at the top left corner to hide the boundaries of microblocks.

2. Calculate the average value of each microblock.

3. Break the complete set of all possible average values into smaller non-overlapping intervals. Map each interval onto a bit 0 or 1 such that too many consecutive intervals do not map onto the same bit. Let this mapping function be f. Let the interval size be Δ. Manipulate the pixel values in each microblock such that the new average value is the mid-value of the interval to which it originally belonged.

4. Break this collection of microblocks into groups of M microblocks each. Each such group is called a *macroblock*. These macroblocks should not overlap each other.

5. There are various schemes for grouping microblocks. These are described and analyzed in [6]. The scheme used for our reported experiments is a self indexing scheme which groups microblocks in an image dependent manner. We split each plane into an index portion (IP) and a watermarked portion (WP) using a secret key. A free list consisting of all the microblocks is formed. Using the secret key, we randomly pick one microblock of WP and consider it to be the first of the M microblocks. We look for the corresponding microblock in IP. Depending on the average value of the microblock in IP, we calculate an index into the free list. This gives the second of the M microblocks. This block is then deleted from the free list. We continue this process of grouping the microblocks until no more macroblocks can be formed.

6. Once we have grouped the microblocks into macroblocks, then for each macroblock we calculate the average value of the macroblock and represent it in binary form and add suitable number of redundant bits using any error redundancy code to get M bits. These M bits are then jumbled up based on the secret key.

7. Now one bit is stored into each one of the M microblocks. This is done using the mapping function f which maps average values of microblocks to a bit i.e. 0 or 1. During the process of embedding the microblock average values are checked to see as to which bit they inherently represent. If this bit is the same as the bit to be embedded then we do not modify the microblock else we change the microblock average value by Δ or $-\Delta$ using an error diffusion scheme as described below.

ERROR DIFFUSION

There are various error diffusion schemes. However, the problem with simple error diffusion schemes is that while embedding the average value, the resultant average of the macroblock might itself change, introducing an error in the embedding stage. Hence the embedded value and the actual resultant value will not match leading to a problem during verification. Due to space constraints, we are not able to describe or analyze the alternatives. The proposed error diffusion scheme is described below. Please refer to [6] for details.

Let the range of intensity values be [0,L].
For each macroblock we define,
A = set of microblocks with average value equal to 0.
B = set of microblocks with average value equal to L.
C = set of microblocks with average value between 0 and L and the inherently embedded bit being the same as bit to be embedded.
D = set of microblocks with average value between 0 and L and the inherently embedded bit being the opposite of the bit to be embedded.
A_0 = set of microblocks with average value 0 and the inherently embedded bit being the same as bit to be embedded.
A_1 = set of microblocks with average value 0 and the inherently embedded bit being different from the bit to be embedded.
B_0 and B_1 are similarly defined.
Let a = $|A|$. Also, b, c, d, a_0, a_1, b_0, b_1 are similarly defined. Clearly, $a_0 + a_1 = a$ and $b_0 + b_1 = b$. Let I = net average value of all blocks, I_1 = quantized value of I using quantization step of γ.

We note that the average value of D blocks can be increased or decreased by Δ in order to embed the required bit. Thus, we can manipulate the macroaverage value by $\left[\frac{-d\Delta}{M}, \frac{d\Delta}{M}\right]$ in a manner that suits us. This controlled manipulation of the macroaverage value by D blocks can be used to compensate for the uncontrolled manipulations caused by A_1 and B_1 blocks. Thus if, $-d\Delta \le (a_1\Delta - b_1\Delta) \le d\Delta$, then the controlled manipulation of D blocks can completely compensate for the uncontrolled manipulations. The above equation reduces to $|a_1 - b_1| \le d$. If this inequality holds, then the embedding scheme described in the next section is used with parameter $\rho = (a_1 - b_1)$.
If the above inequality is not naturally satisfied, then we try to change the macro average value to find a solution. The macroaverage value can be changed to any value in the interval $\left[I_1 - \frac{\gamma}{2}, I_1 + \frac{\gamma}{2}\right]$, so that it still maps to I_1. If i microblocks are changed in the above manipulation, then the macroaverage value changes by $\frac{i\Delta}{M}$. Hence we try out the possible choices of i such that $I + \frac{i\Delta}{M}$ belongs to the above interval. To make the value as $I + \frac{i\Delta}{M}$ we increase the value of i a_1's (assuming $i > 0$) by Δ to make them of type C. Hence the number of a_1's get reduced by i. Hence our new inequality becomes: $|(a_1 - i) - b_1| \le d$. If $i < 0$, then we need to decrease the value of i b_1's by Δ but the inequality remains the same. If this inequality holds, then the embedding scheme is used with parameter $\rho = (a_1 - i - b_1)$.

If we are still not able to find a solution, then we change our macroaverage value such that it quantizes to $I_1 + j\gamma$ (starting with $j = 1$). Hence our new macroaverage value must lie in the range $\left[I_1 + j\gamma - \frac{\gamma}{2}, I_1 + j\gamma + \frac{\gamma}{2}\right]$. Therefore, we find all k such that $I_1 + \frac{k\Delta}{M}$ lies in this range. Increasing macroaverage value by $\frac{k\Delta}{M}$ requires a change of k a_1's by Δ. Hence, the number of a_1's reduces by k; and our inequality becomes $|(a_1 - k) - b_1| \le d$. If this holds, then the embedding scheme is used with parameter $\rho = (a_1 - k - b_1)$. Note that the values of a_1, b_1 and d correspond to the new macroaverage value of $I_1 + j\gamma$, and these may be different from the values obtained for I_1. Above procedure is repeated for $j = -1, \pm 2, \ldots \pm\frac{\Delta}{\gamma}$ until a solution is obtained.

BIT EMBEDDING SCHEME

The procedure for embedding the bit sequence into the microblocks takes a parameter ρ and is as follows:

1. The A_0, B_0 and C microblocks are inherently embedded and are left untouched.
2. The average values of microblocks in A_1 are increased by Δ and those of B_1 decreased by Δ so as to embed the required bits.
3. If $\rho > 0$, then we decrease the average value of ρ blocks belonging to D. If $\rho < 0$, then increase the average value of ρ blocks belonging to D. The remaining (d - ρ) blocks are alternately increased and decreased.

VERIFICATION ALGORITHM

The procedure for watermark verification consists of the following steps:

1. Repeat the steps of watermark embedding scheme to reconstruct the microblocks.
2. For each of the watermarked portions of the planes, identify the macroblocks using the same scheme used in the watermarking stage.
3. Now extract the macroaverage value embedded in each macroblock.
4. Calculate the actual macroaverage value.
5. Compare the embedded macroaverage value with the actual macroaverage value. If the two values are equal (allowing for the manipulator's playing field discussed in [6]), then we say that the macroblock is untampered; otherwise we say that it has been tampered with.

SECURITY

To make the boundaries of macroblocks hidden, an offset may be left randomly at the top left corner of the image depending on a secret key. This would deter any attacks on the image that preserve the average value of the microblocks. The free list for grouping microblocks is constructed randomly, again using the secret key, to prevent an adversary from identifying the macroblocks in the image. Knowledge of the macroblocks could make the watermarking scheme insecure. For details, refer to [6].

EXPERIMENTAL RESULTS AND CONCLUSION

We carried out extensive simulations to confirm that the proposed algorithms generate watermarks that are able to tolerate JPEG compression. Adjacent figures demonstrate the invisibility of the watermark. In other words, the embedding process did not cause any visual artifacts. Furthermore, it was verified to survive JPEG compression. The extension of the proposed scheme to rights management and MPEG video is being studied.

REFERENCES

1. Bender W., Gruhl D., Mormoto N., Lu A., "Data Hiding Techniques", IBM System Journal, Vol. 35, No. 3-4, 1996, P313-36.

2. Morimoto N, "Introduction to Data Hiding Techniques", IBM Technical Report, TR-1441, Oct 1995.

3. Hartung F., Kutter M., "Multimedia Watermarking Techniques", Proc. IEEE, July 1999, P1079-107.

4. Kundur D., Hatzinakos D., "Digital Watermarking for telltale tamper proofing and authentication", Proc. IEEE, Vol. 87, No. 7, July 1999, P1167-80.

5. Hung A.C., "PVRG-JPEG CODEC 1.1", Stanford University,1993,ftp://havefun.stanford.edu/pub/jpeg/JPEGv1.2.tar.Z.

6. Agarwal S., Aggarwal A., Chhugani J., Bassali H., Dubey P., Compression-tolerant Watermarking for Authentication, IBM Tech Report No. IRL# 99A082.

7. Fridrich J., "Methods of data hiding", at site http://www.ssie.binghamton.edu/fridrich/.

Figure: Original Picture (top) Watermarked (bottom)