PROTECTION OF DIGITAL IMAGES USING SELF EMBEDDING

[a,b]*Jiri Fridrich and* [a]*Miroslav Goljan*

[a]*Center for Intelligent Systems, SUNY Binghamton, Binghamton, NY 13902-6000*
[b]*Mission Research Corporation, 1720 Randolph Rd SE, Albuquerque, NM 87501*
fridrich, bg22976@binghamton.edu
SUNY Binghamton, Binghamton, NY 13902-6000

ABSTRACT

In this paper, we propose a technique for selfembedding an image into itself as a means for protecting the image content. After selfembedding, it is possible to recover portions of the image that have been cropped out, replaced, or otherwise tampered. The method is based on transforming small 8×8 blocks using a DCT, quantizing the coefficients, and carefully encoding them in the least significant bits of other, distant squares. If two least significant bits are used for encoding, the quality of the recovered image is roughly equivalent to a 50% quality JPEG.

1. INTRODUCTION

Powerful publicly available image processing software packages such as Adobe PhotoShop or PaintShop Pro make digital forgeries a reality. Feathered cropping enables replacing or adding features without causing detectable edges. It is also possible to carefully cut out portions of several images and combine them together while leaving barely detectable traces.

In the past, several techniques based on data hiding in images have been designed as a means for detecting tampering.

One of the first techniques used for detection of image tampering was based on inserting check-sums into the least significant bit (LSB) of image data. Walton [1] proposes a technique that uses a key-dependent pseudo-random walk on the image. The check-sum is obtained by summing the numbers determined by the 7 most significant bits and taking a remainder operation with a large integer *N*. The check-sum is inserted in a binary form in the LSB of selected pixels. The method is very fast and on average modifies only half of the pixels by one gray level. Check-sums provide a very high probability of tamper detection, but cannot distinguish between an innocent adjustment of brightness and replacing a person's face. Increasing the gray scales of all pixels by one would indicate a large extent of tampering, even though the image content has been unchanged for all practical purposes.

Van Schyndel et al. [2] modify the LSB of pixels by adding extended m-sequences to rows of pixels. The sequences are generated with a linear feedback shift register. For an $N \times N$ image, a sequence of length N is randomly shifted and added to the image rows. The phase of the sequence carries the watermark information. A simple cross-correlation is used to test for the presence of the watermark. Wolfgang and Delp [3] extended van Schyndel's work and improved the localization properties and robustness. They use bipolar m-sequences of –1's and 1's arranged into 8×8 blocks and add them to corresponding image blocks. The watermark is moderately robust with respect to linear and nonlinear filtering and small noise adding.

Zhu et al. [4] propose two techniques based on spatial and frequency masking. Their watermark is guaranteed to be perceptually invisible, yet it can detect errors up to one half of the maximal allowable change in each pixel or frequency bin depending on whether frequency [5] or spatial [6] masking is used. The image is divided into blocks and in each block a secret random signature (a pseudo-random sequence uniformly distributed in [0,1]) is multiplied by the masking values of that block. The resulting signal depends on the image block and is added to the original block quantized using the same masking values. Errors smaller than one half of the maximal allowable change are readily detected by this scheme. The error estimates are fairly accurate for small distortions.

Fridrich [7,8] describes a technique capable of distinguishing malicious changes from innocent image operations or LSB shuffling. An image is divided into medium-size blocks and a robust spread-spectrum watermark is inserted into each block. The watermark in each block depends on a secret camera's ID, the block number, and on the block content. The content of each block is represented with M bits extracted from the block by projecting it on a set of random, smooth patterns and thresholding the results [9]. If watermarks are present in all blocks with high probability, one can be fairly confident that the image has not been tampered with in any significant manner (such as adding or removing features comparable in size to the block). If the

watermark correlation is lower uniformly over all image blocks, one can deduce that some image processing operation was most likely applied. If one or more blocks show very low evidence for watermark presence while other blocks exhibit values well above the threshold, one can estimate the probability of tampering and with a high probability decide whether or not the image has been tampered with.

Other techniques for detection of tamper in digital imagery based on fragile watermarks have been introduced in [10–13].

In this paper, we describe a new anti-tampering technique that can be used to *retrieve* the original content rather than just indicate which pixels or blocks have been tampered with. The image is divided into 8×8 blocks and each block is DCT transformed. A specified number of the lowest frequency DCT coefficients are quantized using a quantization matrix corresponding to a 50% quality JPEG. The coefficients are ordered in a zig-zag manner and their values are encoded using a fixed number of bits. The number of coefficients and their encoding are carefully chosen so that the resulting bit-string for each block is exactly 64 bits long. Information about block B (e.g., the 64-bit string) is inserted into the LSB of the block $B + \vec{p}$, where $\vec{p}$ is a vector of length approximately 3/10 of the image size with a randomly chosen direction. If two LSBs are used for selfembedding, more quantized coefficients can be encoded using 128 bits rather than just 64. In this case, the recovered selfembedded image is perceptually indistinguishable from a 50% quality JPEG compressed original. This enables us to recover even very small features comparable to the block size. To prevent a pirate from masking a forged piece of an image, the bit-string should be encrypted.

In Section 2, we describe the algorithms for selfembedding and recovery of the hidden information. The performance of our technique is demonstrated on real images. In Section 3, we discuss possible security gaps and limitations of our approach. Finally, in Section 4, we close with concluding remarks and outline future research directions.

2. SELFEMBEDDING ALGORITHM

For obvious reasons, it is certainly not possible to embed a complete image into itself. To lower the information content of the image, we have to use either lossy compression (e.g., JPEG compression), decrease the color depth of the image, or preserve only important image features, such as information about edges, using Laplacian filter.

In this paper, we decrease the information content of the original image using a procedure similar to lossy JPEG compression algorithm.

2.1 Selfembedding algorithm #1

We start with dividing the original image into blocks of 8×8 pixels. The following three steps are carried out for each block B:

Step 1 (Preparing the image for embedding).
Gray levels of all blocks are transformed into the interval [−127, 128] and the LSBs of all pixels are set to zero. As will be seen later in this section, this step is important and enables automatic discernment between a tampered block and an unchanged block for which the code was lost by tampering with some other part of the image.

Step 2 (Generating the code).
Each 8×8 block B is transformed into the frequency domain using DCT. The first 11 coefficients (in zig-zag order) are quantized with the following quantization table Q that corresponds to 50% quality JPEG:

$$Q=\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}.$$

The quantized values are further binary encoded. The bit lengths of their codes (including the signs) are shown in matrix L

$$L=\begin{bmatrix} 7 & 7 & 7 & 5 & 4 & 3 & 2 & 1 \\ 7 & 6 & 5 & 5 & 4 & 2 & 1 & 0 \\ 6 & 5 & 5 & 4 & 3 & 1 & 0 & 0 \\ 5 & 5 & 4 & 3 & 1 & 0 & 0 & 0 \\ 4 & 4 & 3 & 1 & 0 & 0 & 0 & 0 \\ 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Coding based on L will guarantee that the first 11 coefficients from each block will be coded using exactly 64 bits. In the rare event when the i-th DCT coefficient has absolute value is larger than 2^{L_i-1}, only this maximum available value will be encoded.

Step 3 (Encrypting and embedding).
The binary sequence obtained in Step 2 (e.g., the 64-bit string) is encrypted and inserted into the LSB of the block $B + \vec{p}$, where $\vec{p}$ is a vector of length approximately 3/10 of the image size with a randomly chosen direction. Periodic boundary conditions (torus topology) are used to get the block $B + \vec{p}$ always inside the image (see Figure 2).

After selfembedding, the marked image is modified very little. In fact, on average 50% of pixel

values will not be changed, and 50% of them will be modified by one gray level. The quality of the reconstruction using algorithm #1 is visibly worse than for an image that has been JPEG-compressed at 50% quality. This may not be sufficient for capturing details smaller than the block size. There is an obvious tradeoff between the quality of reconstruction and the extent of modifications due to selfembedding. By using two least significant bits for selfembedding rather than just one LSB, the image quality of the reconstruction will be dramatically improved while the changes to the original image will still be very minor.

Figure 1 Image reconstructed from the LSB using Algorithm #1.

2.2 Selfembedding algorithm #2

As explained above, this algorithm is similar to algorithm #1 with the exception that two LSBs are now used for encoding the original content of the image.

Step 1 (Preparing the image for embedding).
This step is the same as in Algorithm #1. Two least significant bits are now set to zero.

Step 2 (Generating the code).
Each block is transformed into the frequency domain using DCT. The first 3 coefficients are encoded using the same number of bits as in Algorithm #1. The next 18 bits carry information about coefficients No. 4–21. A zero means that the corresponding coefficient is 0, while ones indicate non-zero coefficients. Following these 18 bits, we encode the values of all nonzero coefficients. Coefficients of higher frequencies are encoded with correspondingly fewer bits. If the length of the code is still short enough, up to two next nonzero coefficients between the 22^{nd} and 36^{th} coefficient are also coded (again, their positions first and then their values). The average code length is about 100 bits (1.55 bits per pixel). The code is shorter for blocks from areas of approximately uniform brightness. If the total length of the code is less than 128, zero padding is applied. All 128 bits are utilized for detection of tampered blocks.

Step 3 (Encrypting and embedding).
This step is the same as in Algorithm #1 with the exception that now 2 LSBs are replaced with the code.

2.3 Automatic image reconstruction

If a collection of blocks from the marked image is cropped (block $B^{(1)}$) and replaced with a different image, the code $c^{(1)}$ at $B^{(2)} = B^{(1)} + \vec{p}$ will not agree with the code c generated from the content of $B^{(1)}$. However, there will be another false discrepancy. The code at $B^{(1)}$ will not agree with the content at $B^{(0)} = B^{(1)} - \vec{p}$. What is needed is an algorithm that would be able to identify the tampered block that needs to be reconstructed. Let us denote the block which carries the code for the block $B^{(2)}$ as $B^{(3)} = B^{(2)} + \vec{p}$ (see Figure 2).

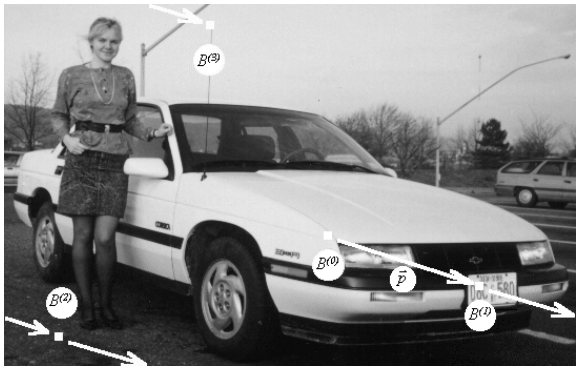

Figure 2 Information chain scheme.

If the code at $B^{(3)}$ is consistent with $B^{(2)}$ content and the code at $B^{(1)}$ is not consistent with the content of $B^{(0)}$, then we conclude that it was $B^{(1)}$ that has been tampered with and $B^{(1)}$ will be reconstructed from the code $c^{(1)}$ at $B^{(2)}$. If $B^{(3)}$ code is not consistent with $B^{(2)}$ content, we conclude that it is $B^{(2)}$ that has been tampered with.

With every reconstruction of $B^{(1)}$ content (the most significant 6 or 7 bits) we also regenerate the code (the LSBs) embedded in $B^{(1)}$. This way the reconstructed image will again contain the correct selfembedded information about its content.

As pointed out in the next section, certain special cases of tampering require human supervision for correct reconstruction. Also, if both the block $B^{(1)}$ and $B^{(2)}$ are tampered, the embedded information about $B^{(1)}$ is lost.

In Figures 3–7, we demonstrate the amazing ability of the presented techniques to retrieve the original, seemingly lost content with a very good quality. Figure 3 is the original image and Figure 4 shows the original image with its content embedded

using Algorithm #2. One can easily recover the original license plate (Figure 6) from an image in which the plate has been replaced with a different one (Figure 5). Figure 7 is an image recovered from the encoded information only. Notice the scrambled block corresponding to the code in the tampered license plate.

Another example of recovery of a person's face that has been mosaic-filtered to prevent identification is shown in Figures 8–9.

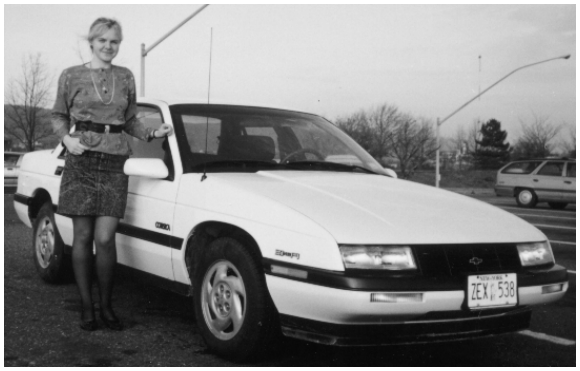

Figure 3 Original image

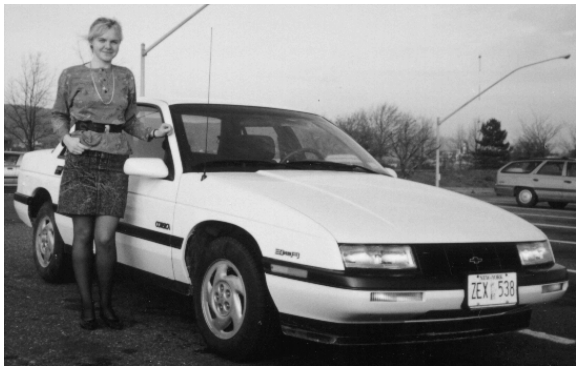

Figure 4 Self reconstructable modification

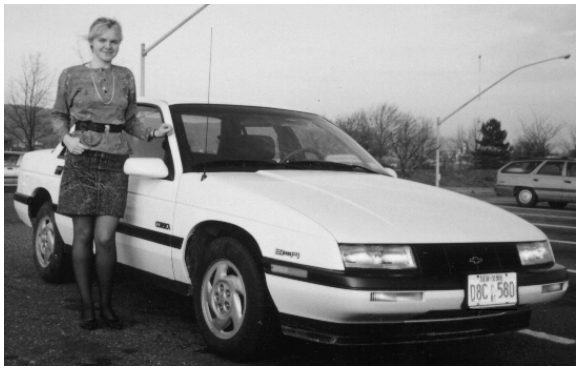

Figure 5 Tampered self reconstructable modification (tampered license plate)

3. LIMITATIONS AND POSSIBLE ATTACKS

First, we point out the obvious limitations of our algorithms. If large portions of the image are replaced, it is quite likely that both the tampered block B and the block with the code for B will be changed, in which case the reconstruction for B is not possible because the hidden selfembedded information has been lost. For the same reason, if one makes changes in blocks that constitute a chain of content-code / content-code (see blocks $B^{(0)}$, …, $B^{(3)}$ in Figure 2) then both the content and the corresponding codes are lost.

Figure 6 Reconstructed image

Figure 7 image recovered from the encoded information only.

If the content of every block B is consistent with the code at $B+\vec{p}$, we conclude that the image has not been tampered with or manipulated. However, there are modifications of the image that will go undetected. If one does not change the least two significant bits and modifies the DCT coefficients that are being coded in our algorithms so that their quantized values do not change, small undetectable changes in the image would result. Another possibility would be to change those coefficients whose values are not coded (for example the (5,5) DCT bin).

If a pirate crops a certain portion B of the image but keeps the least two significant bits intact, then the code at B will be left unchanged while its content (the 6 most significant bits) will be different. In this case, there will be just one inconsistency: The content of B will be inconsistent with the code at $B + \vec{p}$. It appears that it is impossible to design an automatic procedure that would distinguish whether the pirate manipulated the LSBs of $B + \vec{p}$ or replaced the 6 most significant bits of B. To resolve this

problem, an operator would have to interpret the reconstructed portions and distinguish between these two cases based on the interpretation of the images.

The method for selfembedding in which the content of one block is encoded in the LSB of another block appears to have a serious security flaw. Even if the codes are encrypted using a private key before embedding, a pirate could find out the secret vector $\vec{p}$ by dividing blocks into classes with identical quantized DCTs. The codes for all blocks from one class will be the same. This will enable the attacker to recover the secret direction $\vec{p}$ from only a small number of images (possibly even from just one image). If the codes are put into blocks that are randomly scattered over the image instead of shifting them by a fixed vector, the attack is slightly more complicated, but can also be carried out using a relatively small number of images. Once the attacker knows the direction $\vec{p}$, given many marked images he can build an extensive database of pairs of quantized blocks with their corresponding encrypted codes. This database will enable him to create forgeries that will not be detected with our algorithm. For each small 8×8 block from the new forged portion, the attacker can search the database for the closest (with respect to *rms* error) block and use that block in the forgery. This may not enable him to create absolutely "clean" forgeries, but they may look acceptably especially for large images.

To overcome this "database" attack, we propose to scatter the code of block B over a collection of scattered pixels rather than over pixels that form a block. This slight modification of the encoding process will make the database attack impractical.

4. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we overviewed current techniques for tamper detection in digital images. We proposed and tested a new technique for embedding an image into itself. We divide the image into small 8×8 blocks that are DCT transformed, quantized, and carefully encoded into the LSBs of other distant 8×8 blocks. This enables us to recover portions of images that have been cropped or replaced or severely modified. If two least significant bits are used for encoding, the quality of the reconstructed image is indistinguishable from a 50% quality JPEG compressed image. The technique can be easily extended to color images.

The proposed technique has been designed with the intent to maximize the quality of the recovered image. The embedded information has no redundancy and is therefore very fragile and cannot survive any image modification that modifies the least two significant bits. In order to gain some robustness, we would have to sacrifice the quality of the recovered image. The more robustness is required, the less information can be encoded and the worse the image quality of the reconstructed image.

In our future research, we will study spread spectrum techniques for selfembedding in order to gain some robustness.

5. ACKNOWLEGEMENTS

The work on this paper was supported by Air Force Research Laboratory, Air Force Material Command, USAF, under a research grant number F30602-98-C-0176. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of Air Force Research Laboratory, or the U. S. Government.

6. REFERENCES

[1] S. Walton, “Information Authentication for a Slippery New Age”, *Dr. Dobbs Journal*, vol. 20, no. 4, pp. 18–26, Apr 1995.

[2] R. G. van Schyndel, A. Z. Tirkel, and C. F. Osborne, “A Digital Watermark”, *Proc. of the IEEE Int. Conf. on Image Processing*, vol. 2, pp. 86–90, Austin, Texas, Nov 1994.

[3] R. B. Wolfgang and E. J. Delp, “A Watermark for Digital Images”, *Proc. IEEE Int. Conf. on Image Processing*, vol. 3, pp. 219–222, 1996.

[4] B. Zhu, M. D. Swanson, and A. Tewfik, “Transparent Robust Authentication and Distortion Measurement Technique for Images”, preprint, 1997.

[5] G. E. Legge and J. M. Foley, “Contrast Masking in Human Vision”, *J. Opt. Soc. Am.*, **70**(12), pp. 1458–1471, 1980.

[6] B. Girod, “The Information Theoretical Significance of Spatial and Temporal Masking in Video Signals”, *Proc. of the SPIE Human Vision, Visual Processing, and Digital Display*, vol. 1077, pp. 178–187, 1989.

[7] J. Fridrich, “Image Watermarking for Tamper Detection”, *Proc. ICIP ’98*, Chicago, Oct 1998.

[8] J. Fridrich, “Methods for Detecting Changes in Digital images”, ISPACS, Melbourne, November 4th-6th, 1998.

[9] J. Fridrich, “Robust Bit Extraction From Images”, *ICMCS'99*, Florence, Italy.

[10] M. Yeung, and F. Mintzer, "An Invisible Watermarking Technique for Image Verification", *Proc. ICIP'97*, Santa Barbara, California, 1997.

[11] P. Wong, "A Watermark for Image Integrity and Ownership Verification", *Proc. IS&T PIC Conference*, Portland, Oregon, 1998.

[12] R. B. Wolfgang and E. J. Delp, "Fragile Watermarking Using the VW2D Watermark", *Proc. SPIE, Security and Watermarking of Multimedia Contents*, San Jose, California, Jan 25–27, 1999, pp. 204–213.

[13] D. Kundur and D. Hatzinakos, "Towards a Telltale Watermarking Technique for Tamper Proofing", *Proc. ICIP*, Chicago, Illinois, Oct 4–7, 1998, vol 2.

Figure 8 A person masked with a mosaic filter.

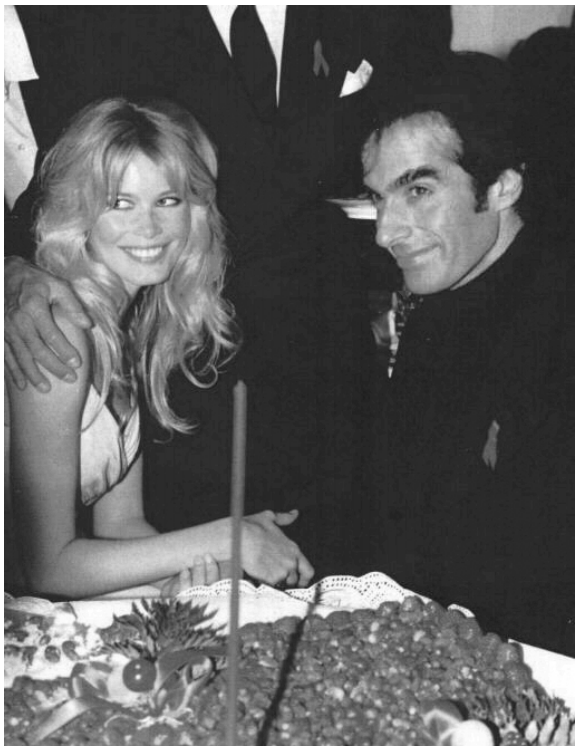

Figure 9 Reconstructed image.