Fragile Watermarking Using the VW2D Watermark

by R Wolfgang, C Podilchuk, E Delp
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Fragile Watermarking Using the VW2D Watermark

Raymond B. Wolfgang and Edward J. Delp
Video and Image Processing Laboratory (*VIPER*)
School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana, 47907-1285
USA

ABSTRACT

Two classes of digital watermarks have been developed to protect the copyright ownership of digital images. Robust watermarks are designed to withstand attacks on an image (such as compression or scaling), while fragile watermarks are designed to detect minute changes in an image. Fragile marks can also identify where an image has been altered. This paper compares two fragile watermarks. The first method uses a hash function to obtain a digest of the image. An altered or forged version of the original image is then hashed and the digest is compared to the digest of the original image. If the image has changed the digests will be different. We will describe how images can be hashed so that any changes can be spatially localized. The second method uses the Variable-Watermark Two-Dimensional algorithm (VW2D) [1]. The sensitivity to changes is user-specific. Either no changes can be permitted (similar to a hard hash function), or an image can be altered and still be labeled authentic. Latter algorithms are known as semi-fragile watermarks. We will describe the performance of these two techniques and discuss under what circumstances one would use a particular technique.

Keywords: Multimedia security, Digital watermarking, Hash functions

1 INTRODUCTION

Digital watermarks can be used to assert ownership of digital images [2]. A watermarking technique consists of a algorithm that incorporates a watermarking structure into an image in the spatial or spatial frequency domains. A testing or verification algorithm is then used to determine if an image has a watermark in it. It is also desirable for the testing procedure to determine if the image has been altered and to supply localization information as to where the image was altered. Overviews of digital watermarking are presented in [3,4].

1.1 Watermarking Scenarios

We will describe two scenarios where image copyright enforcement is needed. One deals with the marking of images for authenticating the content or author. In authentication-based watermarks one is less concerned with the watermark remaining in the image after an attack, but rather with ensuring the detection of an attack (by the absence or distortion of the mark). The other scenario is to mark an image in such a way as to detect unauthorized copies of the image. For authentication, it is important that even slight changes to the image be detected and localized. Embedding a false mark must be practically impossible and the watermark must be easily destroyed by most attacks. It is not desirable for the watermark to remain in the image after the attack, although the watermark should survive cropping. These types of watermarks are known as *fragile watermarks*. Copy-detection watermarks must withstand substantive attacks such as compression and filtering; these are known as *robust watermarks*. A fragile watermark that can withstand mild distortion without affecting the authenticity of an image is sometimes known as a *semi-fragile* watermark.

This work was partially supported by a grant from the AT&T Foundation. Address all correspondence to E.J. Delp, ace@ecn.purdue.edu, http://www.ece.purdue.edu/~ace, or +1 765 494 1740.

1.2 Cryptographic Tools

Many tools are available to protect images from unauthorized viewing, copying, distribution and modification. These include encryption, cryptographic authentication, time stamping and the use of hash functions [5,6]. Encryption disguises the content of an image. Only users who possess the decryption "key" can convert the encrypted image back to its original form. Without the key, it is computationally infeasible to derive the original image. Encryption does nothing, however, to protect the image after decryption. Authentication does not hide the content of the image, but proves who created it. Time stamps can pinpoint the image's owner *and* the time at which the image was generated. One trivial way to claim ownership of an image is to possess the earliest reliable time stamp of the image. These tools may be used in various combinations. Public key (asymmetric) cryptography can encrypt an image, and independently perform authentication. A *digital signature* is a form of authentication.

1.2.1 Hash functions

Digital signatures and time stamps make use of *one-way hash functions* [5]. A hash function has as its input an arbitrary length binary vector P, and produces a fixed length binary output H.

$$H = H(P) \tag{1}$$

H is a function of *all* the input bits and is known as the "hash" or "digest" of P. Changing a single input bit will change the output significantly. There are several useful properties of hash functions:

1. It is computationally trivial to produce H, given P.
2. It is extremely difficult to reproduce P, given only H.
3. It is very difficult to find two inputs P_1 and P_2 that produce the same H. This occurrence is known as a "collision".

Many different hash functions exist: MD-4, MD-5 and the US National Institute of Standards and Technology (NIST) Secure Hash Standard, SHA-1, are examples. SHA-1 is used in the NIST Digital Signature Standard. These and additional hash functions are described in [5].

1.2.2 Time stamps

It is our feeling that to assert ownership that is consistent with current intellectual property right law, a watermarking technique must support the use of third-party cryptographic-based digital time stamping that is embedded in the image through the watermarking process [7]. One way to authenticate an image is to prove that one was the first to possess it. For this, one needs some way to "stamp" the image with the time of creation much like the Post Office stamps mail with a postmark. Forward or back-dating of the time stamp must not be possible [5]. Let X be the binary data to be time stamped and let $Y = H(X)$ be the hash of X. An owner sends an official request, R, to a third party time stamping service (TSS) by concatenating the hash Y and the owner's binary identification number I (assigned by the TSS).

$$R = (Y, I) \tag{2}$$

The TSS receives the request, and assigns it a request number, n.

$$R_n = R \tag{3}$$

The time of this request is noted by the TSS, and labeled as t_n. The TSS then generates an intermediate binary string associated with request R known as a *certificate* C_n:

$$C_n = (n, t_n, R_n; L_n) \tag{4}$$

L_n is known as the *linking string* and is a concatenation of the previous request time, request and the hash of the previous linking string.

$$L_n = (t_{n-1}, R_{n-1}, H(L_{n-1})) \tag{5}$$

Before returning the time stamp to user I, the TSS must wait for the next request, R_{n+1}, then concatenates it to the certificate C_n. The TSS signs this concatenation, which forms the actual time stamp, S:

$$S = F((C_n, R_{n+1}), K_{PRI}) \tag{6}$$

K_{PRI} is the private key of the TSS and F is the digital signature function. In practice, only the identification string, I, in R_{n+1} needs to be concatenated to C_n.

Challenges to a time stamp are made as follows. A challenger first authenticates the TSS signature on the time stamp in question. The challenger can then ask the requestor specified by R_{n-1} (which is embedded in L_n) to show their time stamp. This challenge can be made by the TSS to ensure the privacy of the person who made request R_{n-1}. The quantities in L_n must match those in the previous certificate, C_{n-1}. The hash of L_{n-1}, easily regenerated by the challenger, must also match the stored value $H(L_{n-1})$ present in L_n. If any value does not match, one of the two time stamps is false. In practice, the linking string contains information from the last several requests. This distributes the authentication responsibility, since any one of the requestors in the linking string may verify the time stamp. This process may continue backwards or forwards to the challenger's satisfaction. Improvements to this procedure are discussed in [5]. In [3,8] we describe how time stamping can be used in watermarking to address the "re-watermarking" attack.

2 HASH FUNCTIONS AS FRAGILE WATERMARKS

Simply hashing the entire image will act as a fragile watermark. This is similar to the way hash functions are used in protecting text files from modification. One could think of the hash as a fixed-length summary or "digest" of the image. This digest could be used in the time stamp request as described above. If the digest of a "test image" does not match the digest of the original image, the frame difference between the two images may be obtained to localize any changes. This approach reveals not only the location of any alterations, but also the amplitude. The problem with this approach is that the original image is required to localize any changes. The two algorithms described below do not require the original image for localizing changes; they only need the digests obtained for certain parts of the image.

1. The first technique is known as the *row-column hash function technique*. A separate digest is first obtained for each row and each column of an image. (We will assume the image is unmarked, although it may have been previously watermarked with a different algorithm.) These hashes are stored, and then compared to the row and column hashes obtained for a test image. If there is any discrepancy, the location of the changes can be determined simply by identifying the row and column hashes that are different. Although this approach can localize a (single) alteration to a single pixel, the technique suffers from the ambiguity problem described in the figure below. This problem causes unaltered parts of the image to be rejected as unauthentic. For example, if regions *A* and *B* in Figure 1 are altered, this approach produces the false indication that the shaded regions in the image were also changed. The ambiguity problem makes this hash technique of limited use for images.

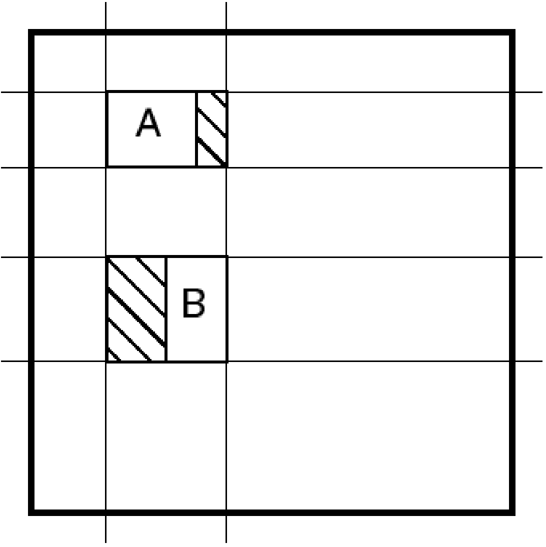

Figure 1. Illustration of the ambiguity problem with row/column hashing.

2. Another approach is to obtain the digests W_B x H_B block of pixels in an image. W_B and H_B are the width and height of the image block. This algorithm is known as the *block-based hash function (BBHF) technique*. These hashes are then stored, and compared to the corresponding hashes obtained for a test image. If a particular set of hashes differ, then a change has occurred in the test image within the appropriate block. The storage requirement for this second technique is

$$S = (\left\lceil \frac{W_I}{W_B} \right\rceil \left\lceil \frac{H_I}{H_B} \right\rceil) L \tag{7}$$

where S is the number of bytes needed to store the hashes, W_I and H_I are the height and width of the image, and L is the number of bytes required for each hash. For MD5, $L = 16$.

All of the techniques above are fragile, since they use “hard” hash functions to characterize the content of an image.

3 FRAGILE WATERMARKS

3.1 Variable-Watermark Two-Dimensional Algorithm (VW2D)

The Variable-Watermark Two-Dimensional algorithm (VW2D) [1] is a technique used to authenticate an image based on a set of stored values obtained from the watermark and a watermarked image. The watermark, *W*, is a pseudo-random binary sequence. The image is marked on a block-by-block basis.

$$Y(b) = X(b) + W(b) \tag{8}$$

X(b) is the original image block, *W(b)* is the watermark block and *Y(b)* is the watermarked image block. This process is repeated for all blocks *b*. The total number of watermark blocks is image dependent; together they form the watermark *W*.

The verification process used to test an image *Z* to see if the watermark is in the image is:

$$\delta(b) = Y(b) \cdot W(b) - Z(b) \cdot W(b) \tag{9}$$

A threshold test is then performed on the test statistic δ .If δ < *T*, where *T* is a user-defined threshold, *Z(b)* is considered genuine. Large values of *T* allow the toleration of changes to the marked image block *Y(b)*. If *Z(b)* = *Y(b)*, then $\delta = 0$. Note that the original image *X* is not required for testing.

For VW2D, the threshold test above can be expanded [8] such that the image block may be classified as:

a) unaltered
b) slightly affected
c) definitely altered but still originating from the watermarked image
d) completely changed or not originating from the watermarked image.

Figure 2 shows an image that was VW2D watermarked. Figure 3 shows the VW2D watermark used for this image.

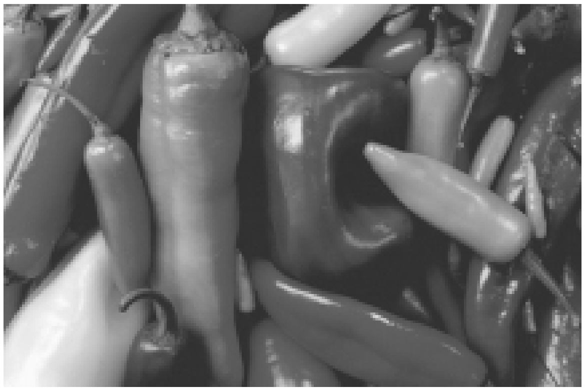

Figure 2. Watermarked vegetables image

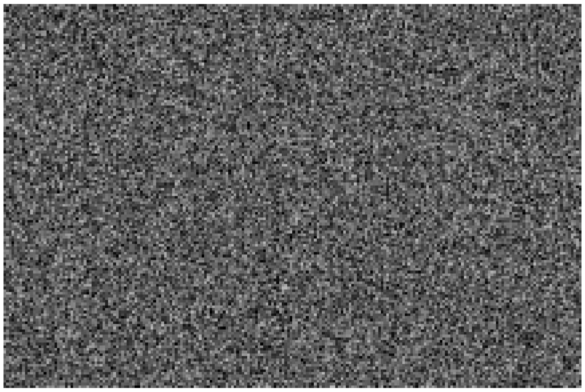

Figure 3. VW2D watermark

We have shown that VW2D is robust to several different types of attacks [8]. In this paper we will describe how VW2D can act as a semi-fragile watermark. Several other fragile watermarks have been developed in [9,10,11].

4 EXPERIMENTAL PROCEDURE

We have investigated how BBHF and VW2D perform as fragile watermarks. Two original 512 x 512 grayscale images are shown in Figure 5 (tiger) and Figure 7 (bridge). Both BBHF and VW2D were used for each image with 8 x 8 blocks and BBHF using the MD5 Hash. The VW2D marked images are in Figure 6 and Figure 8. Several forged copies of each image were then generated, and tested for their authenticity. All attacks were performed on a 128 x 128-pixel sub-block of the image as indicated in Figure 4, to test the localization ability of each algorithm.

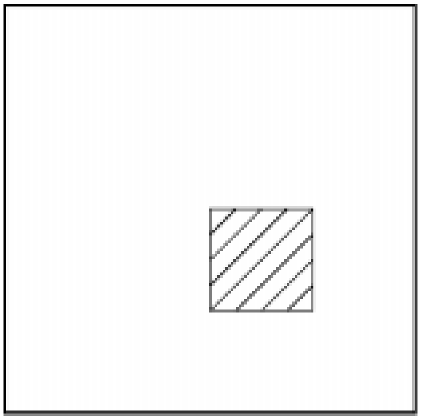

Figure 4. Forged region in images

The alterations made were mean filtering with a 3 x 3 and 5 x 5 neighborhood in the 128 x 128 region.

Figure 9 shows the tiger image after 5 x 5 mean filtering of the block. Figure 10 shows the same case for the bridge image.

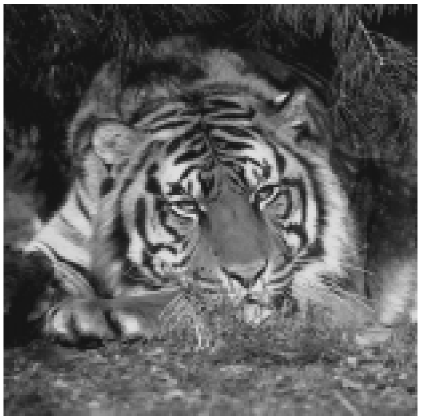

Figure 5. Original tiger image

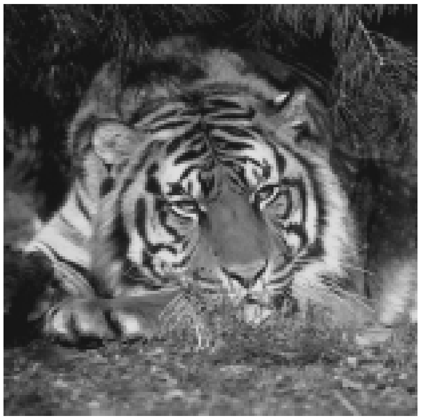

Figure 6. VW2D marked image

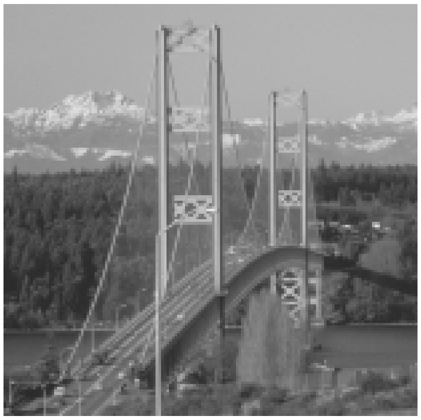

Figure 7. Original bridge image

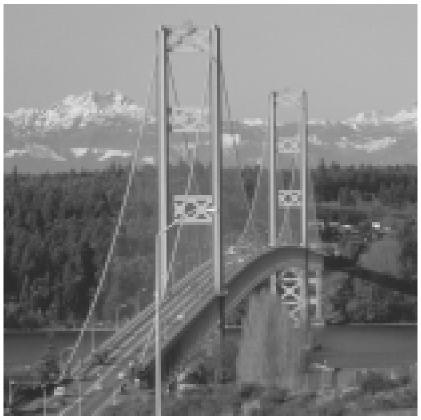

Figure 8. VW2D marked image

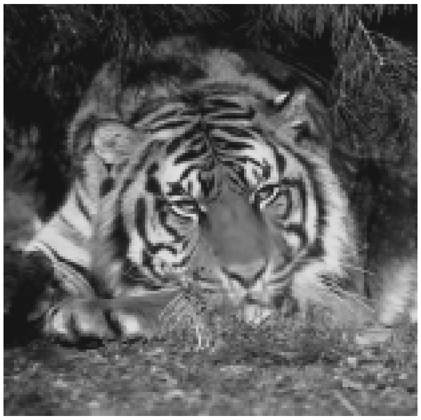

Figure 9. 5x5 mean filtering in 128 x 128 block of VW2D-marked Tiger image

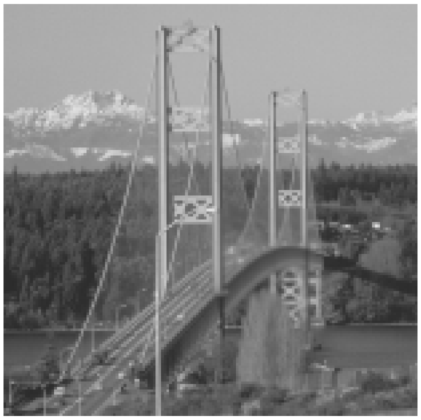

Figure 10. 5x5 mean filtering in 128 x 128 block of VW2D-marked Bridge image

5 EXPERIMENTAL RESULTS

To visualize the changes made to the tiger and bridge images, error-images were generated based on the hashes for BBHF and for the test statistic δ that indicate where changes have occurred. These error images reveal which 8 x 8 image blocks are considered unauthentic (i.e. have been altered according to the respective algorithm). Black pixels in the error image correspond to unchanged (authentic) pixels in the test image, where white indicates forged pixels. Since the detection is performed on an 8x8 block basis, blocks in the error image are labeled “white” in 8 x 8 pixel units. A pixel in the test image may be identical to the corresponding pixel in the original image, but if it is part of a block that is deemed altered by the verification algorithm, the error image will classify this pixel as changed.

Figure 11 through Figure 15 show the error images for the altered Tiger image, with 3x3 filtering. As expected, BBHF detected all changes to the image, as indicated by the solid white blocks in Figure 11. Figure 12 shows that the zero threshold case of VW2D detects almost all the changes made to the image. A threshold of 100 (Figure 13), however, barely reveals the general shape of the altered area. Thresholds of 200 (Figure 14) and 300 (Figure 15) detect few changes. In these cases, almost all the alterations to the image are tolerated. No changes were detected for thresholds of 300 or greater. This means that the image with the forged sub-block is considered completely authentic. Table 1 through Table 4 present the number of altered blocks detected for BBHF, and VW2D at thresholds of 0 to 500 in steps of 100.

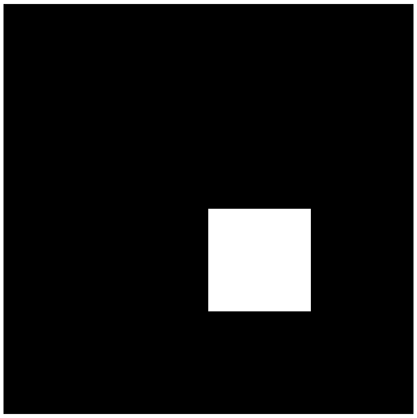

Figure 11. BBHF result

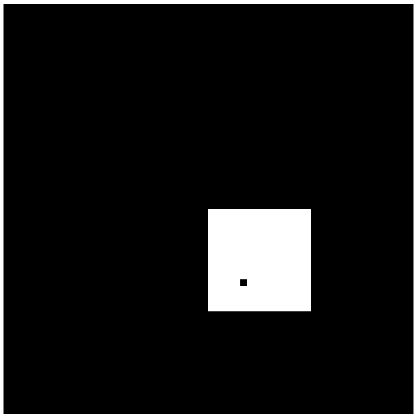

Figure 12. VW2D: T = 0

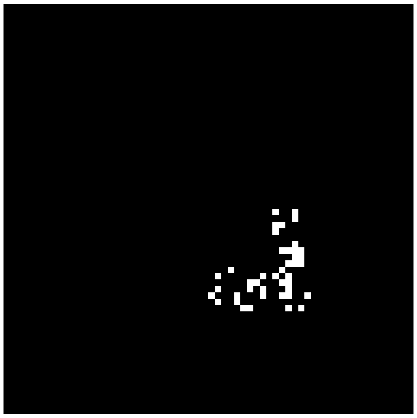

Figure 13. VW2D: T = 100

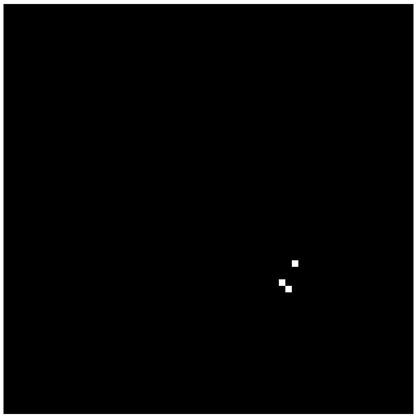

Figure 14. VW2D: T = 200

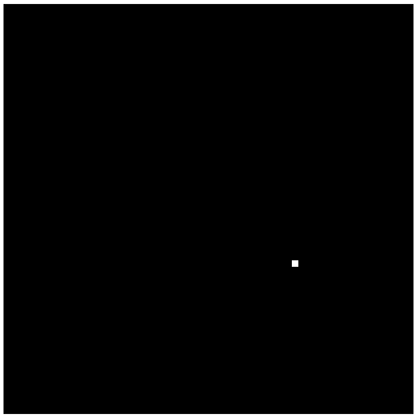

Figure 15. VW2D: T = 300

Figure 16 through Figure 21 show the error images for the Tiger image, with the 128 x 128 pixel sub-block filtered with a 5 x 5 mean filter. Again, BBHF detected all changes to the image, as shown by Figure 16. Since a 5 x 5 filter introduces more distortion to the image than the 3 x 3 filter, we expect a larger number of errors detected for a given VW2D threshold. If the threshold is set to zero (Figure 17), every altered block is detected. As the threshold is increased to 100 (Figure 18), fewer altered blocks are detected but the general shape of the forged area is still recognizable. The altered area is barely outlined by the forged blocks when the threshold is set to 200 (Figure 19), although many changed blocks were still found. Thresholds of 300 (Figure 20) detected more errors than in the 3 x 3 filter case, and a threshold of 400 only detected one altered block. The Bridge image showed a similar trend.

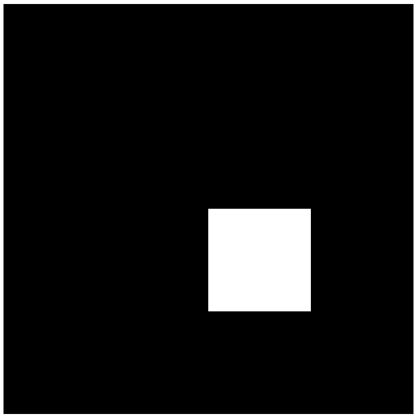

Figure 16. BBHF result

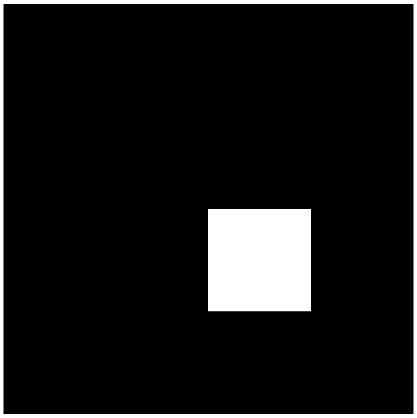

Figure 17. VW2D: T = 0

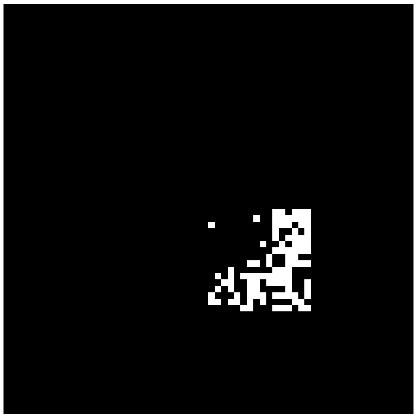

Figure 18. VW2D: T = 100

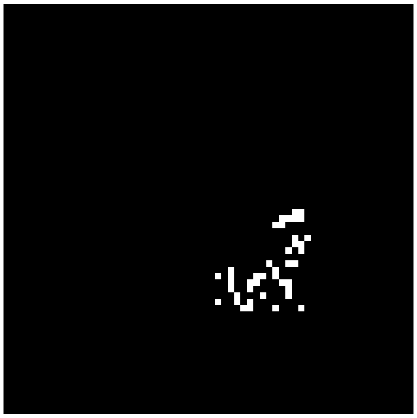

Figure 19. VW2D: T = 200

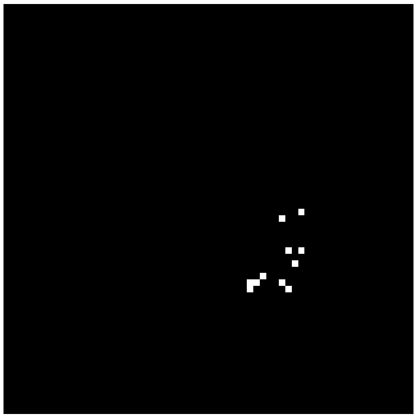

Figure 20. VW2D: T = 300

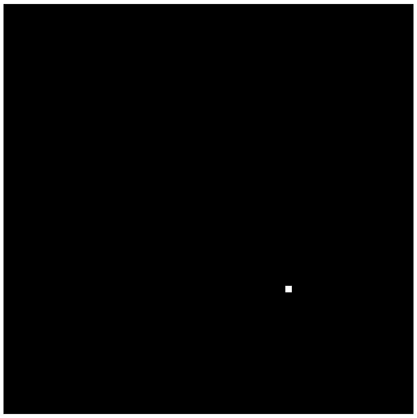

Figure 21. VW2D: T = 400

The data in Table 1 shows the number of forged blocks found in the tiger image by BBHF, and then by the VW2D technique at different thresholds for δ . (The total number of altered blocks is 256.) Table 2 contains the same data, but converted into percentages. Table 3 and Table 4 present the results from testing the bridge image.

Table 1. Number of forged blocks detected by VW2D at different thresholds: Tiger image

T	0	100	200	300	400	500
mean 3x3	255	42	3	1	0	0
mean 5x5	256	101	42	11	1	0

Table 2. Percentage of forged blocks detected by VW2D at different thresholds: Tiger image

T	0	100	200	300	400	500
mean 3x3	99.6%	16.4%	1.2%	0.4%	0.0%	0.0%
mean 5x5	100%	39.5%	16.4%	4.3%	0.4%	0.0%

Table 3. Number of forged blocks detected by VW2D at different thresholds: Bridge image

T	0	100	200	300	400	500
mean 3x3	256	30	1	1	0	0
mean 5x5	256	72	21	2	1	1

Table 4. Percentage of forged blocks detected by VW2D at different thresholds: Bridge image

T	0	100	200	300	400	500
mean 3x3	100.0%	11.7%	0.4%	0.4%	0.0%	0.0%
mean 5x5	100.0%	28.1%	8.2%	0.8%	0.4%	0.4%

6 EVALUATION

BBHF is very effective at identifying every change to an image block. This advantage can, however, preclude it from certain applications. For instance, an image that is compressed and decompressed even at a high data rate will most likely have a slight change to each block. The BBHF technique would register a change for practically the entire image in both cases, even though the decompressed picture may be perceptually identical to the original and thus considered authentic by the owner. VW2D can withstand certain small alterations to an image block, depending on the value of the threshold. This means that with VW2D, small changes to an image block will not prevent an image from being a legitimate representation of the original.

7 CONCLUSION

Two types of digital watermarks – robust and fragile – have recently been considered in this paper. Fragile watermarks are usually used for image authentication, and can often localize any changes made to an image. This paper compares two fragile watermarking techniques for images. The BBHF technique successfully detects all alterations made to an image, regardless if the change is perceptible. The second method, VW2D, allows an image with minimal changes to still be considered authentic. Since for some applications it may be desirable to adjust the sensitivity of a fragile watermarking technique, the VW2D algorithm may be preferred over BBHF.

A version of this paper is available via anonymous ftp to **skynet.ecn.purdue.edu** in the directory **/pub/dist/delp/ei99-hash**.

8 REFERENCES

[1] Raymond B. Wolfgang and Edward J. Delp, “A watermark for digital images,” *Proceedings of the 1996 International Conference on Image Processing*, Lausanne, Switzerland, Sept. 16-19, 1996, vol. 3, pp. 219-222.

[2] M. D. Swanson, M. Kobayashi, and A. H. Tewfik, “Multimedia data embedding and watermarking technologies,” *Proceedings of the IEEE*, vol. 86, no. 6, June, 1998, pp. 1064 - 1087.

[3] R. B. Wolfgang and E. J. Delp, “Overview of image security techniques with applications in multimedia systems,” *Proceedings of the SPIE International Conference on Multimedia Networks: Security, Displays, Terminals, and Gateways*, November 4-5, 1997, Dallas, Texas, vol. 3228, pp. 297-308.

[4] R. Wolfgang, C. Podilchuk and E. J. Delp, “Perceptual watermarks for digital images and video,” to appear in the *Proceedings of the IEEE*, May, 1999.

[5] B. Schneier, *Applied Cryptography*, Second Ed., Wiley & Sons, 1996.

[6] Douglas R. Stinson, *Cryptography, Theory and Practice*, CRC Press, 1995.

[7] Scott Craver, Nasir Memon, Boon-Lock Yeo and Minerva Yeung. “Can invisible watermarks resolve rightful ownerships?,” *Proceedings of the IS&T/SPIE Conference on Storage and Retrieval for Image and Video Databases V*, San Jose, CA, USA, Feb. 13-14, 1997, vol. 3022, pp. 310-321.

[8] Raymond B. Wolfgang and Edward J. Delp, “Techniques for watermarking digital imagery: further studies”, *Proceedings of the International Conference on Imaging Science, Systems, and Technology*, Las Vegas, Nevada, USA, June 30 – July 3, 1997, vol. 1, pp. 279-287.

[9] M. Yeung and F. Mintzer, “An invisible watermarking technique for image verification,” *Proceedings of the IEEE International Conference on Image Processing*, Santa Barbara, CA, Oct. 26-29, 1997, vol. 2, pp. 680-683.

[10] D. Kundur and D. Hatzinakos, “Towards a telltale watermarking technique for tamper-proofing,” *Proceedings of the IEEE International Conference on Image Processing*, Chicago, IL, Oct. 4-7, 1998, vol. 2.

[11] M. Yeung and B.-L. Yeo, “Fragile watermarking of three-dimensional objects,” *Proceedings of the IEEE International Conference on Image Processing*, Chicago, IL, Oct. 4-7, 1998, vol. 2.