A Fragile Watermarking Scheme for Authentication of GIF Images

Rinaldi Munir

School of Electrical Engineering and Informatics
Institut Teknologi Bandung (ITB)
Bandung, Indonesia
rinaldi.munir@itb.ac.id

Abstract: Fragile image watermarking could be used to authenticate a digital image due to modification or altering. A watermark is embedded into the image. When the image was modified or altered, the watermark is also altered or fragile. One of popular image format is GIF image. Some fragile watermarking algorithms are applied to the single GIF image only, none for animated images. However, we could modify the existing steganography algorithm to build a fragile watermarking scheme for any kind of GIF images (still images or animated images). In this paper, we reused *EzStego* to embed a watermark into the GIF images. *EzStego* is a steganography algorithm especially for hiding the message in the palette images such as the GIF images. The watermark, which is a binary image, is inserted into a GIF image based on *EzStego* embeddeing scheme. For increasing security, the watermark is encrypted with the random bits based on chaos system before embedding. To prove authentication of the image, the watermark is extracted from the watermarked image based on *EzStego* extraction scheme and then compare it with the original watermark. The fragile watermark indicates that the image has been altered. We also could find parts of the image that has been altered. We have tested the performance of the proposed scheme by doing some typical attacks to the watermarked images. Based on experiments results we conclude that the watermarking scheme can detect the authentication of the watermarked images due to the attacks.

Keywords: fragile watermarking; EzStego algorithm; image authentication; GIF images; attacks

1. Introduction

An image is a kind of information as a proverb says “a image is more than a thousand words”. Digital images play an important role in current digital era. However, in the current digital era the images can be easily transferred, copied, edited, altered, or manipulated by an unauthorized party. Once an image manipulated or tampered (by using a software such as Photoshop [16], for example), the image is not authentic anymore. How to prove the authentication of the image? The answer is by using fragile watermarking. Fragile watermarking is a technique by embedding a watermark into the image, so that when the image is modified through a linear transformation, the mark is also altered or destroyed [1]. The sensitivity of watermark to modification leads to their use in image authentication [2]. The fragile watermark is the indication that the image is not original (authentic) anymore. Even though the change is slightly and no impact on the visual quality of the image, it will result a tampered or fragile watermark. The third party can verify that an image has not been edited or altered since it was marked. Therefore, fragile watermarking provides a convenient technique for authentication, tamper detection, and verification of image integrity [3].

According to working domain, fragile watermarking system can be divided into two categories: spatial domain and transform domain. In spatial domain, fragile watermarking works directly by embedding the watermark bits into pixel values. Usually the bits are embedded into the least significant bits (LSB) of the pixels for perceptual transparency. It means the embedding does not change the perception of image significantly. Such technique was described in [2], [3], and [4]. In transform domain, the watermark bits is embedded into

Received: April 11st, 2017. Accepted: June 23rd, 2017
DOI: 10.15676/ijeei.2017.9.2.7

the transform coefficients of the host image. Before embedding, the host image (in spatial domain) is transformed to a transform domain (for example in frequency domain). The results are transform coefficients. Next, the transform coefficients are manipulated by embedding the watermark bits. Finally, the image (in transform domain) is transformed back to a spatial domain resulting a watermarked image. Such technique was described in [11-14]. This paper focus on spatial domain only, because embedding in spatial domain is suitable for fragile watermarking. It means when the pixels of a watermarked image are manipulated, the extracted watermark will be broken. This characteristic is important for fragile watermarking.

However, most of image watermarking in spatial domain is for BMP images. A popular scheme of fragile watermarking for BMP images was described in [2]. It collaborated a hash function (MD5) and blocks of image. Bits of watermark were embedded into LSBs of pixels of the blocks. Unfortunately, the BMP images is rarely used in World Wide Web because of their large size. GIF and JPEG images were used widely in Internet. In this paper we focus watermarking on GIF images, because only a few fragile watermarking algorithms for GIF images exist [5, 6], and no fragile watermarking for the animated GIF images.

GIF (Graphics Interchange Format) image is a kind of the indexed image. GIF was introduced by CompuServe in 1987 and come into widespread usage on the World Wide Web because of its wide support and portability. A GIF image uses a palette of up to 256 colors from the 24-bit RGB color space with values in the range [0, 1]. The pixel values represent index to a palette row. GIF format also supports animation of images or animated GIF images. The animated GIF images consist of a number of frames. Each frame was displayed in succession like a video. The animated GIF images is usually used for displaying cartoon film, funny images, or other interesting images.

Research of image authentication on the indexed images are still a few, two of them were described in [5] and [6]. Hassan in [5] proposed a fragile watermarking scheme for color image authentication. The scheme is suitable for class of images with format such as GIF, TIF, or BMP. The color image is first transformed from RGB to YST color space. The T is selected for embedding the watermark. The image is divided into 2×2 non-overlapping blocks and the watermark is then embedded into T channel randomly selected 2×2 block's LSBs using 2D-Torus Automorphism. Chang in [6] proposed a color image authentication scheme using partitioned palette and morphological operations. Morphological operations are adopted to draw out the tampered area precisely [6]. However, those schemes are enough complicated and consuming more time. Moreover, the algorithms worked only for the still GIF images. We require a new fragile watermarking scheme for both the still GIF images and the animated GIF images.

In this paper, we proposed a simple fragile watermarking scheme which is based on a simple steganography algorithm for the GIF images. The algorithm is called *EzStego*. *EzStego* algorithm has been proposed by Machado [7]. Originally *EzStego* is used for steganography. Steganography is art and science for hiding information in a cover (here cover is the image). *EzStego* has two main process, embedding process and extraction process. In embedding process, bits of the message are embedded to LSBs of indices pointing to the palette. To minimize color degradation due to changes in the indices, at first the palette is sorted so that the difference between two adjacent color is minimized. *EzStego* embeds message into the LSBs of indices pointing to the sorted palette. In extraction process, bits of the embedded message can be recovered easily by extracting LSBs of indices of the sorted palette. The simple algorithm makes embedding and extraction process can be performed quickly with minimal visual degradation.

EzStego is used for hiding information in the GIF images only, it is never used for watermarking of the GIF images. Watermarking is a kind of information hiding but it is different with steganography. Goal of image steganography is hiding as much as possible information in a cover image, whereas goal of watermarking is to protect the image itself. Especially in fragile watermarking we hide a watermark inside to authenticate it. Therefore, we proposed a fragile watermarking scheme for both still and animated GIF images which is based on *EzStego*. The watermark, which is a binary image, is inserted into the GIF image based on *EzStego* embedding scheme. For increasing security, the watermark is encrypted with the random bits based on chaos system before embedding. The novelty of the paper is a secure fragile watermarking scheme for any GIF images (still and animated images) based on an *EzStego* algorithm.

The paper is organized into five sections. The first section is introduction. The second section will review some study of literatures such as GIF images and the *EzStego* algorithm. In the third section, we explain a fragile watermarking scheme based on *EzStego* algorithm. The fourth section will describe the experiments and discuss the results. Finally, in last section we give conclusions and suggest future works.

2. Literature Study

A. GIF Images

A GIF image consists of a color palette and a matrix which entries (pixel values) refer to the palette row. In other word, the pixel values represent indices to the palette. Color of the pixel is combination of each channel red (R), green (G), and blue (B) in the palette row. Figure 1 shows the structure of a GIF image. In the figure, the pixel value 5 represents the fifth row of the palette. In the row, R = 0.2902, G = 0.0627, B = 0.0627. Thus, the color perception of the pixel 5 is combination of the color components. The GIF images are suitable for human-made graphics such as cartoon, animation, since the color depth only up to 256 colors.

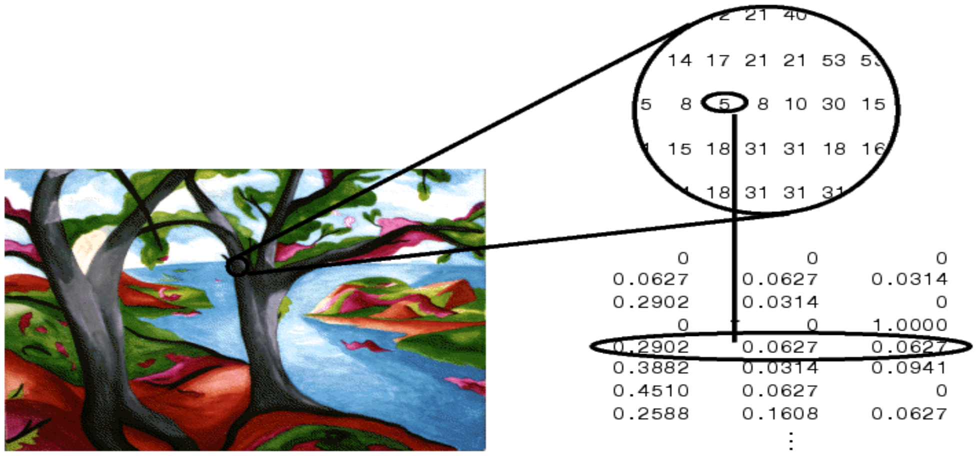


Figure 1. GIF image structure [17]

The animated GIF images consist of a number of frames, each frame may have independent palette. Figure 2 shows six of 48 frames of an animated GIF image (*walk.GIF*). Every animated GIF image has a property that is called "delay time" in hundredth of seconds. It specifies delay every frame. For example, an animated GIF image contains 40 frames, with a 0.03 second delay specified between each frame. It means the animated GIF has runtime of 1.2 seconds per loop.

B. EzStego Algorithm

EzStego is both name of a steganographic tool and name of a steganography algorithm for palette images. *EzStego* embeds bits of the message in the LSBs of the pixels values. Replacement of LSB of the pixel value with a message bit makes the pixel value increase or decrease by one. This new pixel value will point to a previous or next entry in the palette. The color difference between two adjacent entries in the palette maybe significant, so that the color of the pixels before and after embedding maybe different significantly. This makes distortion in the stego-image.

To minimize the distortion, the palette is first sorted by intensity values so that the difference between two adjacent colors is minimized. In the sorted palette, the color indices are close to each other. During the embedding process, the message bits are embedded to LSBs of color indices to the sorted palette.

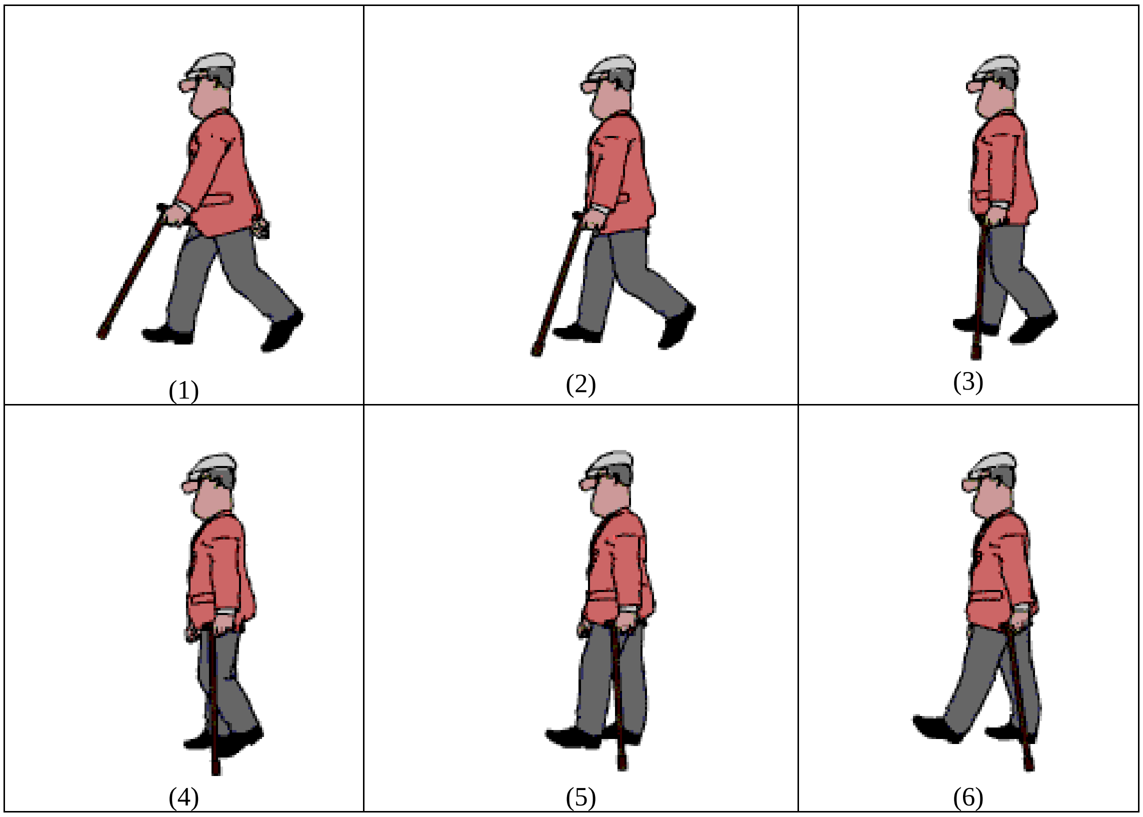


Figure 2. The first six frames of an animated GIF image (*walk.gif*)

The *EzStego* algorithm is very simple. The embedding and extraction steps of *EzStego* algorithm works as follows.

Embedding Algorithm

1. Sort the palette of the original image by distance between color of the pixels. The distance between the color (R_1, G_1, B_1) dan (R_2, G_2, B_2) is calculated by Euclidean distance:

$$d = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2} \quad (1)$$

2. Assign the new index of the sorted palette by numbering 0, 1, 2, … etc.
3. Replace the LSBs of indexes of the sorted palette by bits of the message, *C*. Finally we get a stego-image.

Extraction Algorithm

1. Sort the palette of the stego-image by distance between color of the pixels by using Eq. (1).
2. Assign the new index of the sorted palette by numbering 0, 1, 2, … etc.
3. Extract the LSBs of the indexes of the sorted palette.

Figure 2 illustrates the embedding process in *EzStego*. Suppose there are eight different colors in the palette where the pixel values (pointer to the palette) are 0, 1, 2, …, 7. First, the palette is sorted so that the difference of two adjacent colors are minimized. Next, we assign the new indexes to the palette (000, 001, …, 111). Suppose we embed a message bit ‘1’to pixel 7 which the palette index is ‘100’. We replace the LSB of the ‘100’ by ‘1’ ($10\underline{0} \rightarrow 10\underline{1}$) which it points to pixel 3. Thus, the color of pixel 7 is replaced by color of pixel 3. This technique is applied to all pixels sequentially until the message is exhaustive. By using this technique, we get a stego-image with minimal distortion.

The extraction process works as follows. Before extracting, first we sort the palette with a same way in embedding process. Next, the message bits are extracted from the LSB of the sorted indexes.

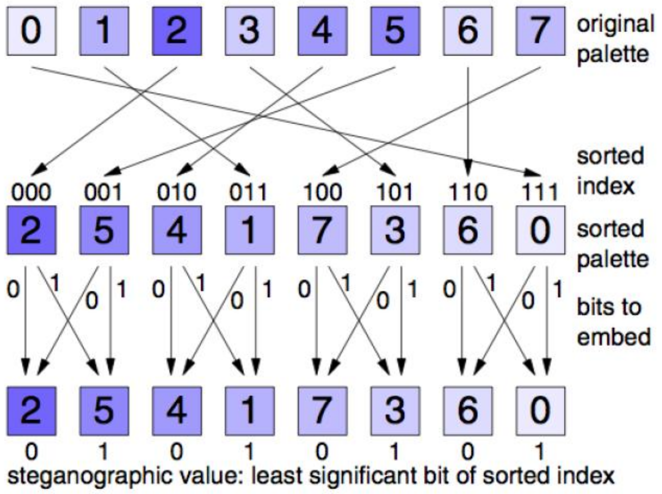


Figure 2. The embedding process of EzStego [8]

3. Proposed Scheme

This section will explain the proposed fragile watermarking scheme based on *EzStego* algorithm above. In general, the fragile watermarking techniques has two main process: embedding and extraction. Generally block diagram of embedding and extraction is figured in Figure 3.

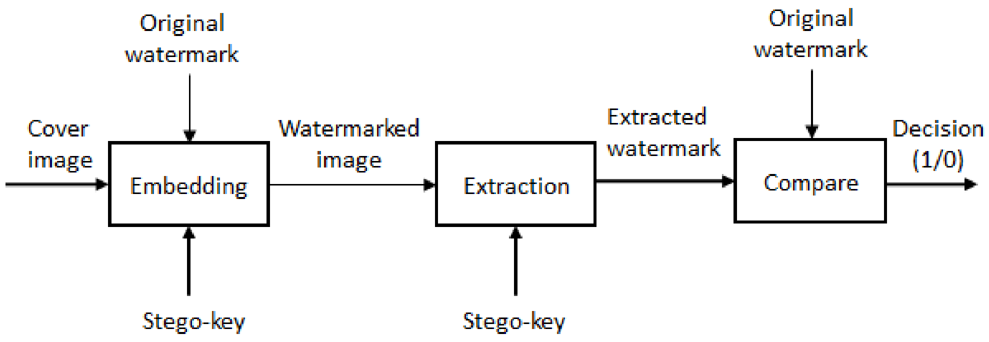


Figure 3. Block diagram of fragile watermarking

To apply the original *EzStego*, that is usually used for hiding the message into the cover image, to the proposed fragile watermarking scheme, there are some modifications to be consider:

1. In original *EzStego*, the message can be anything (text, image, audio, video, etc), whereas in the proposed scheme the watermark must be a binary image.
2. In original *EzStego*, size of the message to be hidden into the cover image must be less than or equal to payload (maximum capacity of hiding), whereas in the proposed scheme size of

the watermark must be equal to size of the cover image in order to detect the changes to pixel level (will be explained later).
3. In original *EzStego*, no key required to embed the message, otherwise in the proposed scheme the watermark must be encrypted before embedding. Therefore the scheme needs a key for encryption and decryption (will be explained later).
4. In original *EzStego*, comparison between the extracted message and the original message (the receiver does not have the original message) is not required, but in the proposed scheme the extracted watermark must be compared to the original watermark. Only the authority party can perform the comparison.

Figure 4 resumes the difference of *EzStego* for steganography (Figure 4(a)) and *EzStego* for fragile watermarking (Figure 4(b)).

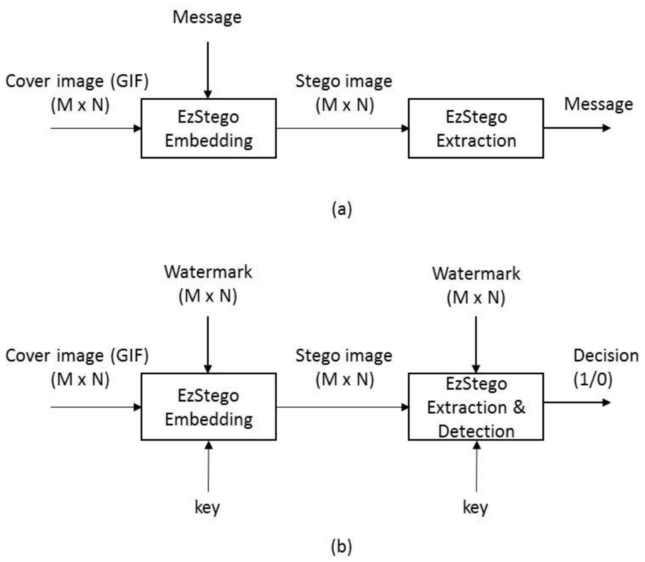


Figure 4. Difference of *EzStego* for steganography and *EzStego* for fragile watermarking

The proposed scheme will embed and extract watermark for the GIF images only. Originally we emphasize developing a fragile watermarking scheme for a single GIF image. Then, we expand the scheme so that it can be applied for the animated GIF images. We explain those schemes in the separated sections.

A. Watermark embedding to a single GIF image

Figure 5 shows the block diagram of watermark embedding. Explanation of Figure 5 as follows. Suppose the image is a GIF image of size $M \times N$ and the watermark is a binary image of size $m \times n$. Usually the watermark size is smaller than the host image (GIF image) size. In order to the modification in the GIF image can be detected until pixel level, we must duplicate the watermark ($m \times n$) by copying and pasting it repeatedly until it has the same size with host image ($M \times N$). For increasing security, the resulted watermark is encrypted by using a simple stream cipher before embedding. The simple stream cipher is by XOR-ing the watermark with the keystream (chaotic bits) generated by a chaotic map. XOR is common operator in stream cipher because its simplicity. We encrypt plaintext p with a key k as $p \oplus k = c$, and conversely decrypt ciphertext c by the same key k as $c \oplus k = p$. In this scheme we encrypt watermark W with a keystream K by using $W \oplus K = W'$ and decrypt W' to get W by using $W' \oplus K = W$.

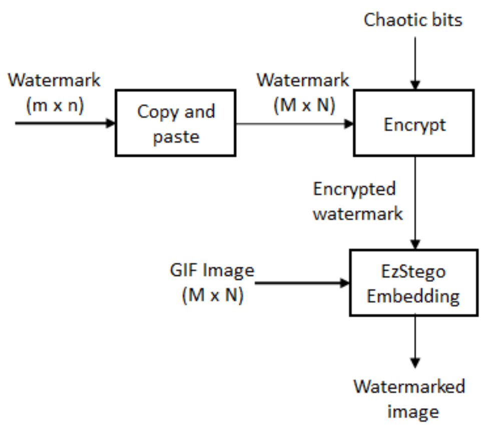


Figure 5. Block diagram of watermark embedding

To generate the keystream from the random numbers, we use a logistic map to by an iteration equation

$$x_{k+1} = \mu x_k (1 - x_k) \tag{2}$$

The parameter μ is a constant where $0 < \mu \leq 4$. The map is in chaotic state when $3.57 < \mu \leq 4$ [9]. To start the iteration, we use a x_0 value that behave a secret key. Next, we transform the chaotic values (real numbers) x_i into bits 0 and 1 as follows: x_i is multiplied by 10 repeatedly until it reach a desired long number (size), and then truncated to take the integer part. Mathematically, this process is described by function T as follows [10]:

$$T(x, size) = \left\| x * 10^{count} \right\|, x \neq 0 \tag{3}$$

where *count* start from 1 until $x * 10^{count} > 10^{size-1}$ and symbol || || represents truncation. The least significant bit of binary representation of the integer is then extracted to get the chaotic bits. After encrypting the chaotic bits with the watermark, the encrypted watermark is embedded into the GIF image based on EzStego embedding scheme. The result is a watermarked GIF image.

B. Watermark extraction from a single GIF image

The watermark extraction is the reverse of watermark embedding. Figure 6 shows the block diagram of watermark extraction. The watermark is extracted from watermarked image by EzStego extraction scheme. In this case we just get encrypted watermark. Next, the encrypted watermark is decrypted with chaotics bits. Finally, we compare between decrypted watermark and original watermark to decide authentication of the image. If they are same then we conclude that the image is authentic, otherwise the image is not authentic anymore. There are three ways to make decision. First, we compare them visually. When we found the difference, we conclude that image has been changed. Second, we compare them bit-per-bit. If they are same exactly, we conclude the image is still authentic, if not we conclude that image has been changed. Third, by using correlation value between the extracted watermark and the original watermark [15]. Statistical correlation is a measure that states strength of linear relationship between two random variables. The higher correlation value indicates that the two variables (the extracted watermark and the original watermark) have a strong linear relationship. It is characterized by a high correlation coefficient (close to +1 or -1). In this paper it is sufficient to use observation visually between extracted watermark and original watermark because of simplicity. The watermark is an image so that the damaged watermark can be seen visually.

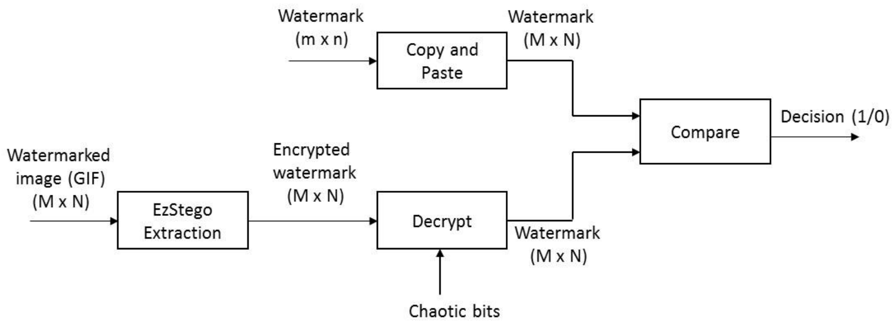


Figure 6. Block diagram of watermark extraction.

C. Watermark embedding to an animated GIF image

If we want to embed a watermark into an animated GIF image, then there are two options to choose from. The first options is embedding all of frames with the same watermark. The second option is embedding each frame with the different watermarks randomly. We choose the first option because of its simplicity. For increasing security, the watermark is encrypted with the random bits. The encrypted watermark is embedded into each frame of the animated GIF image based on *EzStego* embedding scheme. The result is a watermarked animated GIF image (see Figure 7).

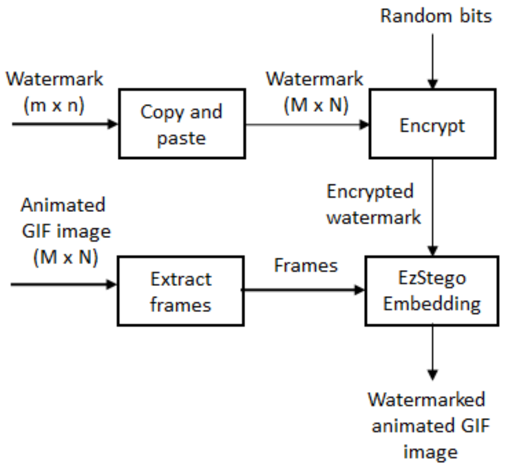


Figure 7. Block diagram of watermark embedding

D. Watermark extraction from an animated GIF image

The watermark is extracted from the watermarked image for proving authentication. Figure 8 shows the block diagram of watermark extraction. The watermark is extracted from each frame of the watermarked animated GIF image by *EzStego* extraction scheme. The result is an encrypted watermark of each frame. Next, we decrypt it with random bits. Finally, we compare between decrypted watermark and original watermark to decide authentication of the image. If they are same then we conclude that the image is authentic, otherwise the image is not authentic anymore.

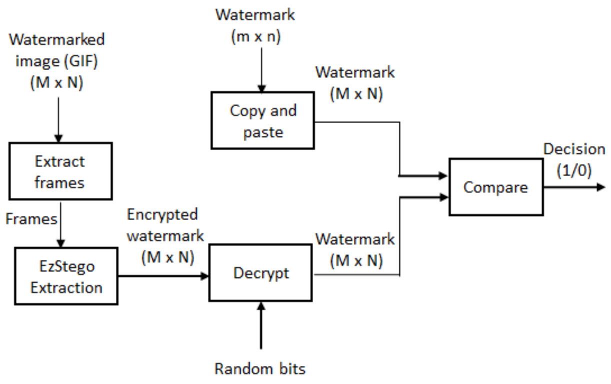


Figure 8. Block diagram of watermark extraction.

4. Experiment and Results

A. Experiments to The Single GIF Image

We have implemented the proposed fragile watermarking scheme into two separated programs (embedding program and extraction program). The embedding program received inputs i.e. a still GIF image, a binary image watermark, and a x_0 value. The extraction program received inputs i.e. a watermarked image, and a x_0 value. Next, we have performed some experiments to test the scheme to some attacks. In these experiments, the host image is a natural GIF image of size 512 × 512 ('peppers.gif'). The watermark is a 'ganeca.bmp' which duplicated periodically in order to result the same size with the host image (Figure 9).

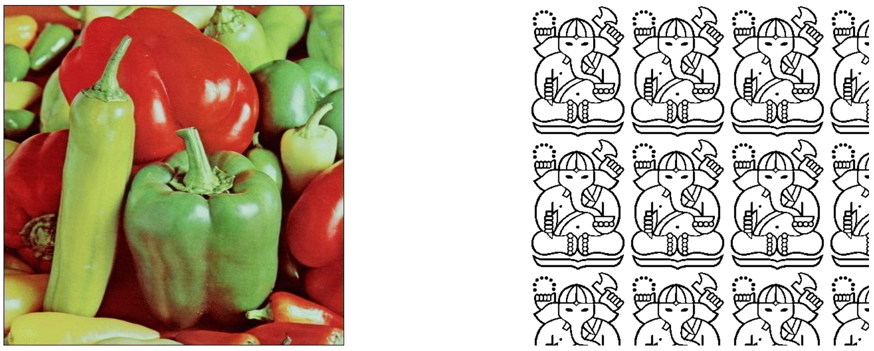


Figure 9. Left: original GIF image; Right: watermark ('ganeca')

We measure the PSNR of the watermarked image for measure its quality. PSNR is calculated by equation:

$$PSNR = 20 \times \log_{10}\left(\frac{255}{rms}\right) \tag{4}$$

where *rms* is abbreviation of *root mean square* of two images, I and $\hat{I}$, of size $M \times N$ pixels, i and j are indices of pixel, that has a formula:

$$rms = \sqrt{\frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{ij} - \hat{I}_{ij})^2} \tag{5}$$

PSNR is commonly used for measuring quality of an image after processing. Embedding of a watermark into an image may decrease quality of the image. To know about it, we calculate PSNR of the watermarked image. The high PSNR reflects good quality of the image. Usually, if PSNR > 30 then quality of the image is still considered good quality [18].

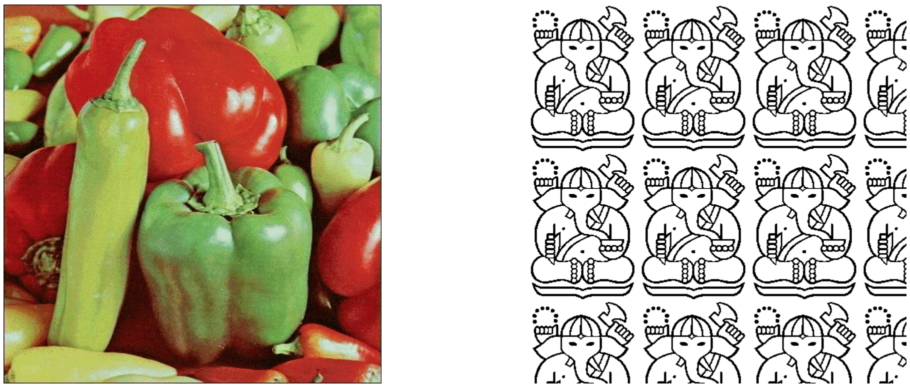


Figure 10. Left: watermarked GIF image; Right: extracted watermark

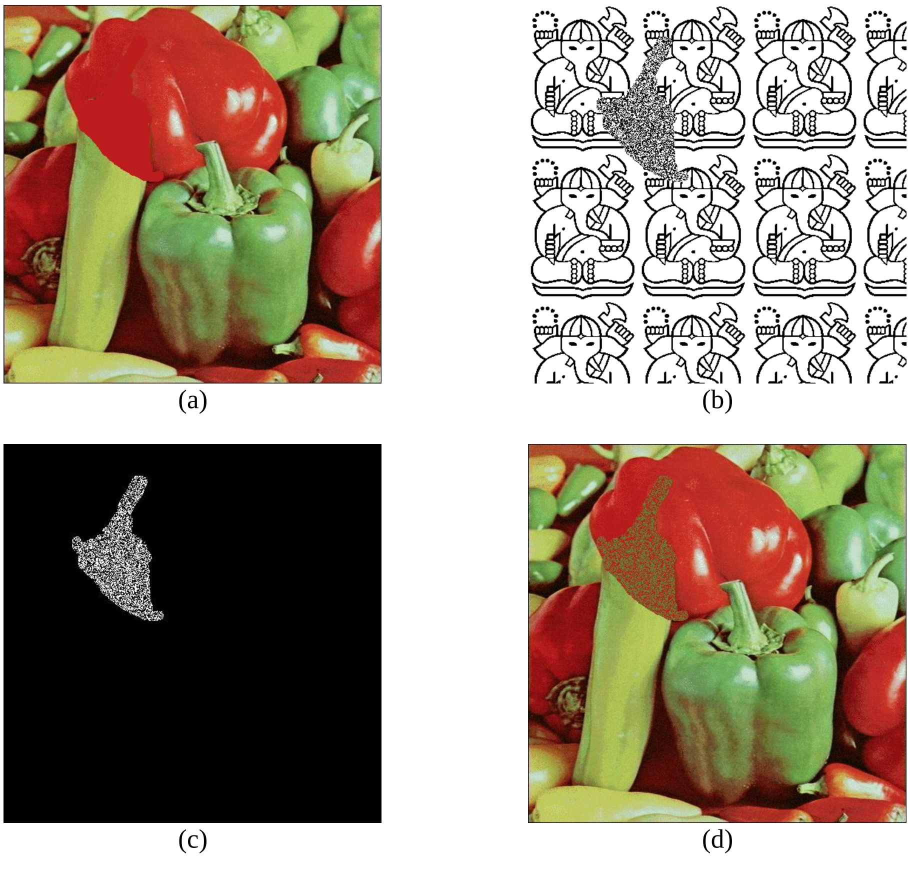


Figure 11. The first attack (deletion attack). (a) the watermarked GIF image has been altered; (b) the extracted watermark contains object that has been altered. (c) part of object that has been altered; (d) identification of altered object in the watermarked image

The watermark was embedded into the host image with secret keys are parameters of Logistic Map which are chosen arbitrarily as x_0 = 0.789 and μ = 3.9762. Figure 10 shows the watermarked image and the extracted watermark. The watermarked image has PSNR = +73.7459 dB. This means that embedding of the watermark yields the slightly change to quality of the image. No degradation appears significantly. Quality of the watermarked image is still very good.

Visually no difference between the original image and the watermarked image, the PSNR is also very high. The watermarked image is not modified and as the result the extracted watermarks is same with the original watermarks exactly. We conclude that the image is authentic. The next experiments are performance of the watermarked image to some typical attacks.

In the first attack (deletion attack), we modified the watermarked image with Photoshop. The top section of a green chili was deleted and replaced by red color (Figure 11a). Next, we extracted the watermark from the modified image. We found the extracted watermark has been broken (Figure 11b). By subtracting the original watermark to the extracted watermark, we could identify part of object which has been altered in the extracted watermark (Figure 11c) and its corresponding altered object in the watermarked image (shown as a dash object, see Figure 11d).

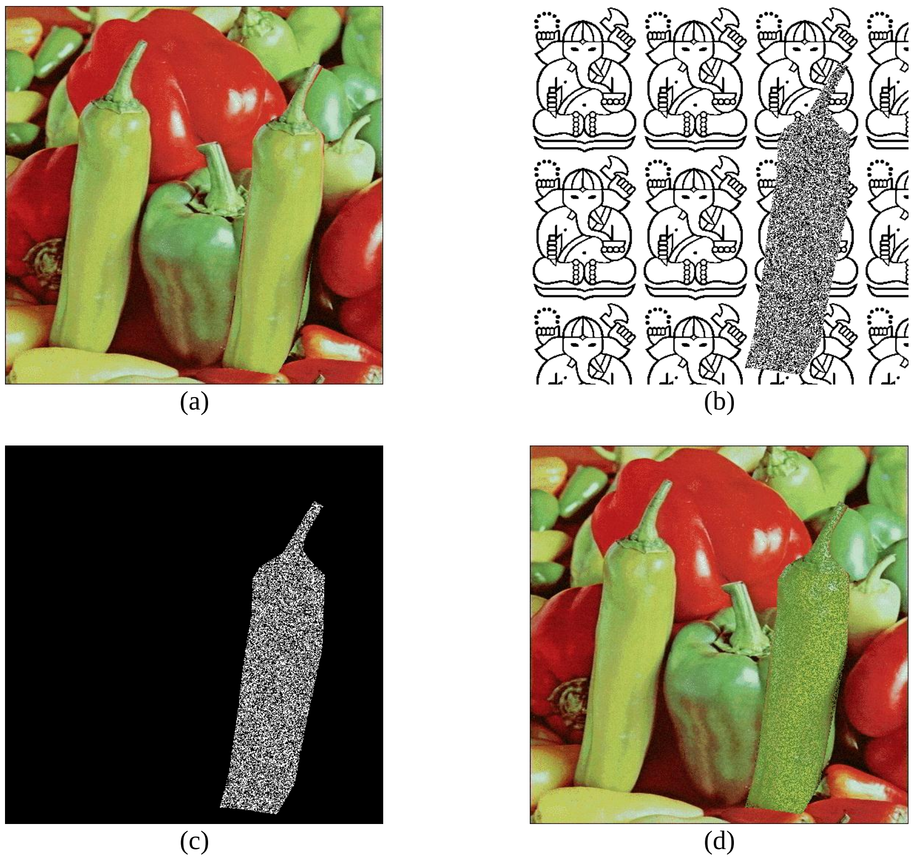


Figure 12. The second attack (copy and paste attack). (a) the watermarked GIF image has been altered; (b) the extracted watermark contains object that has been altered. (c) part of object that has been altered. (d) identification of altered object in the watermarked image

In the second attack (copy and paste attack), we copied and pasted a green chili to two locations in the watermarked image (Figure 12a). Immediately we found the copy-and-paste objects in the extracted watermarked (see Fig 12b). By the same procedure as the first attack above, we could identify the new object in the water final marked image (Figure 12c and 12d).

In the third attack (*flipping attack*), we flipped the watermarked image horizontally (Figure 13a). As the result, we got the wrong watermark, like a random image, where it indicated that the whole watermarked image has been changed (Figure 13b).

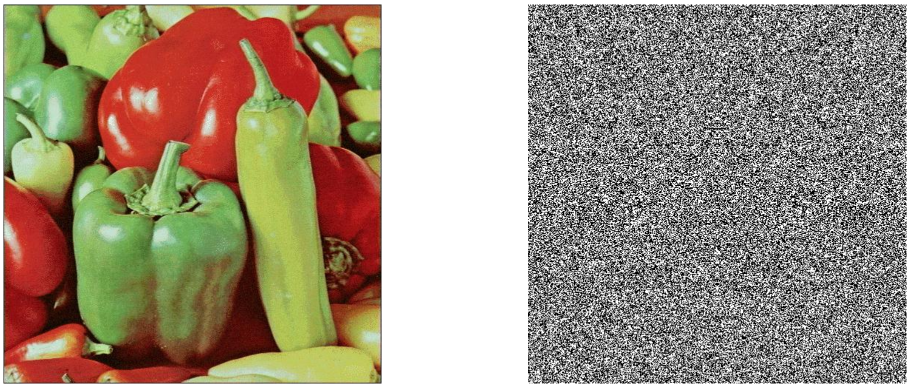


Figure 13. The third attack (flipping attack). Left: the flipped watermarked GIF image; Right: the extracted watermark.

Finally, in the last attack, we adjusted the contrast and the brightness of the watermarked image by using Photoshop. We increase level of the contrast to +40 and level of the brightness to +25. The watermarked image looks to be brighter (Figure 14a). When we extracted the watermark from it, we got the noisy watermark where it indicated that the whole watermarked image has been changed (Figure 14b).

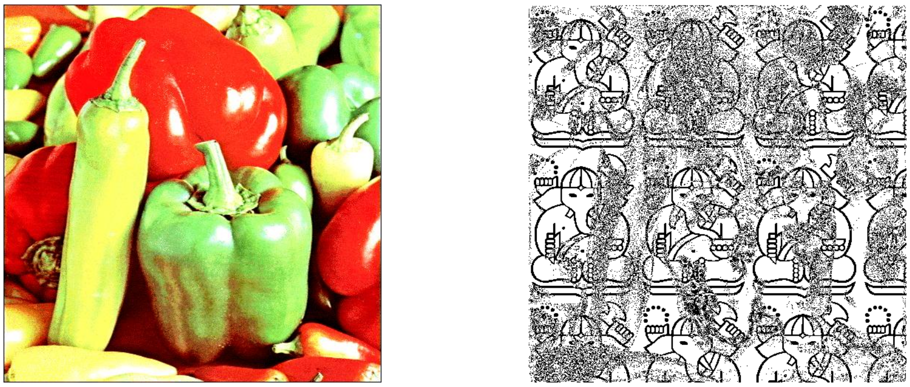


Figure 14. The fourth attack (brightness-and-contrast attack). Left: the watermarked GIF image after increasing the brightness and the contrast; Right: the extracted watermark

B. Experiments to The Animated GIF Image

In these experiments, we used an animated GIF image as host image which has six frames of size 300 × 200 pixel ('jogging.gif') with delay time 0.1, see Figure15. The watermark is a 'black ganeca' logo which duplicated a number of times in order to result the same size with the host image, see Figure16.

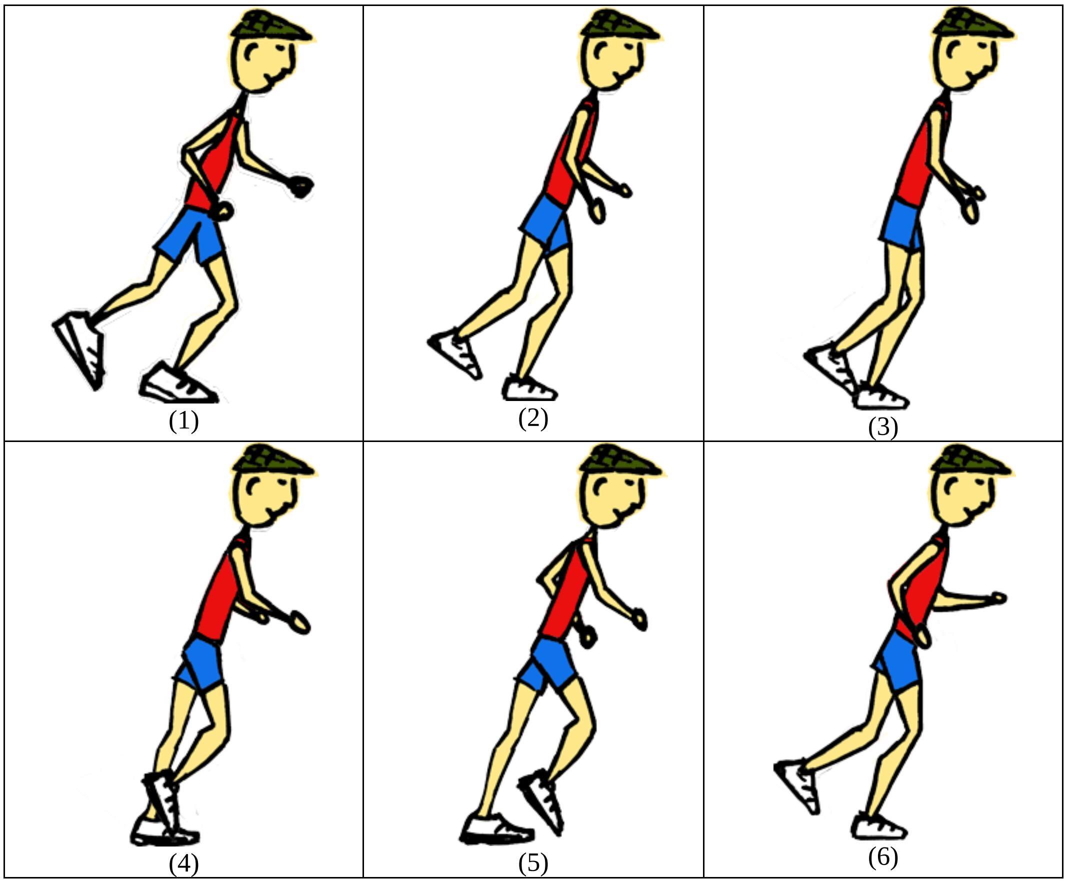


Figure 15. Six frames of an animated GIF image (*jogging.gif*)

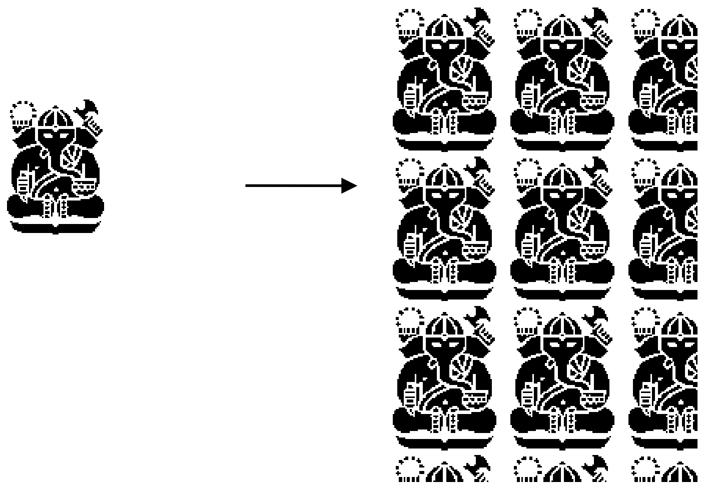
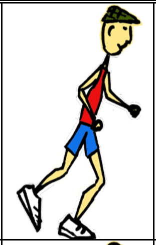
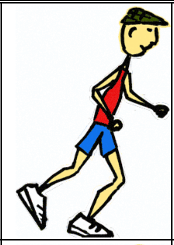
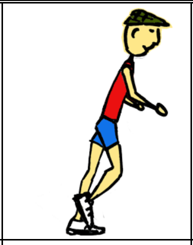
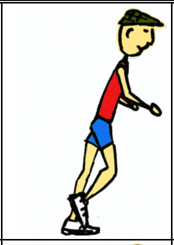
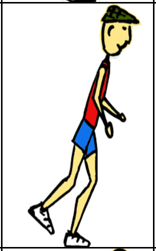
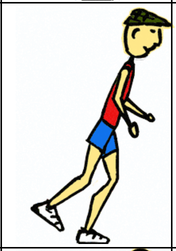
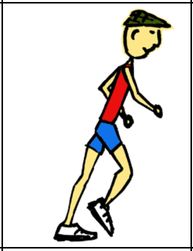
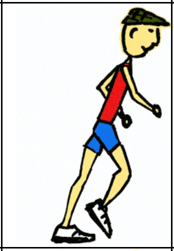
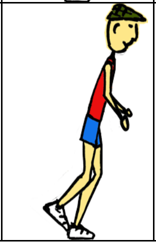
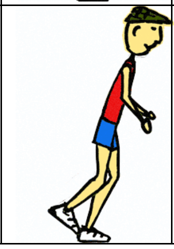
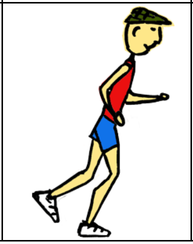
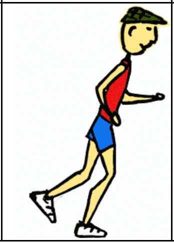


Figure 16. Left: an original watermark. Right: a duplicated watermark a number of times

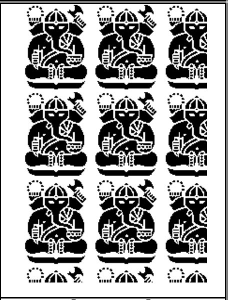
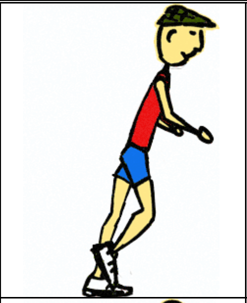
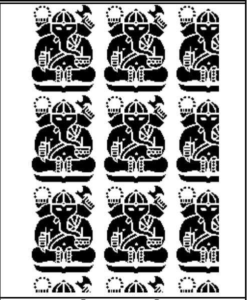
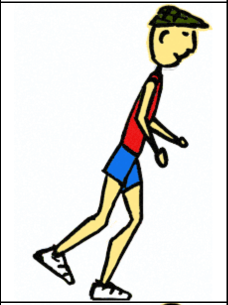
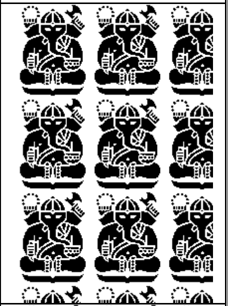
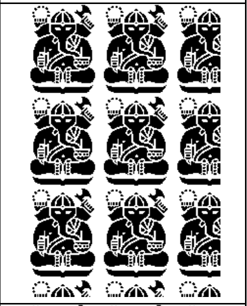
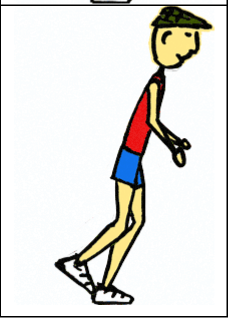
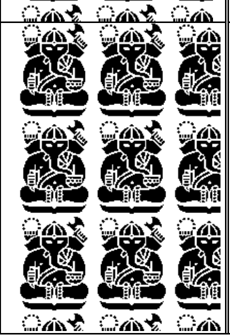
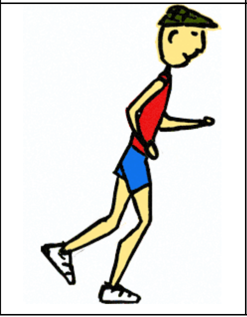
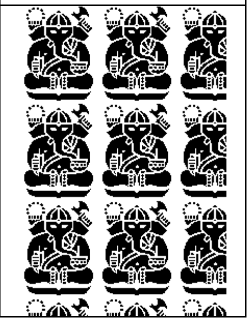
Before the watermark was embedded into the host image, the watermark was encrypted with the random bits. Table 1 shows the frames of the watermarked GIF image. PSNRs of each frame are displayed in right side. We can see that visually the watermarked frames are very similar with their cover frames. The watermarked GIF image can be displayed perfectly as fine as the cover image.

Table 1. The frames of the animated GIF image

Frame	Cover frame	Watermarked frame	PSNR	Frame	Cover frame	Watermarked frame	PSNR
1			78.6228 dB	4			79.5033 dB
2			79.3291 dB	5			79.2489 dB
3			79.4453 dB	6			79.3955 dB

To test whether the animated GIF image is not modified, we only need to extract the embedded watermark from each frame and then compare it (visually or bit-per-bit comparison) with the original watermark. If both watermark are same exactly, we conclude the animated GIF image is still authentic, not modified. Table 2 shows the extracted watermark of each frame of the watermarked animated GIF image. The extracted watermarks are the same exactly with the original watermark, therefore we conclude that the image have not been altered.

Table 2. The extracted watermark of each frames

Frame	Watermarked frame	Extracted watermark	Frame	Watermarked frame	Extracted watermark
1			4		
2			5		
3			6		

Next we tried to attack the watermarked GIF image by adding a ball in front of the runner. We inserted the ball into each frame of the watermarked GIF image in different position, then we extracted the watermark from each frame. For example, we extracted the watermark from the first frame and we got the watermark was broken (see Figure 17). We found shape such as a circle in the watermark. The circle shape did not exist in the original watermark. It was a ball in the modified watermarked GIF image. Therefore, we conclude that the animated GIF image has been altered. Also, by additional process, we could identify part of object in the image which has been altered.

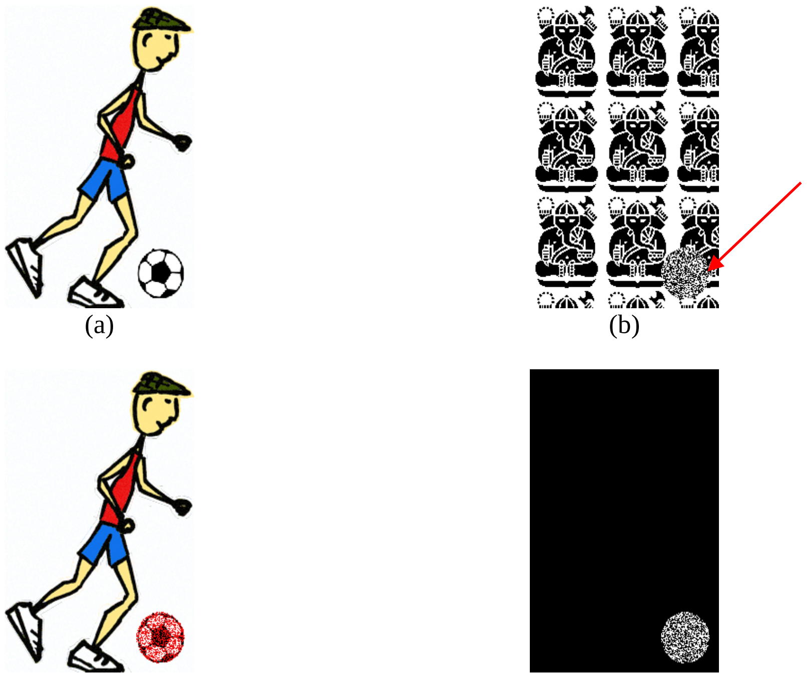


Figure 17. The first attack. (a) the watermarked GIF image has been altered by adding a ball; (b) the extracted watermark contains object that has been altered, shown by a arc. (c) identification of altered object in the watermarked image. (d) part of object which has been altered.

In the second attack, we changed color of the pants from blue color to white color in all frames. When the watermark was extracted from the modified image, we found that the watermark was broken. For example we extracted the watermark from the first frame. We also found part of the image which has been modified (Figure 18).

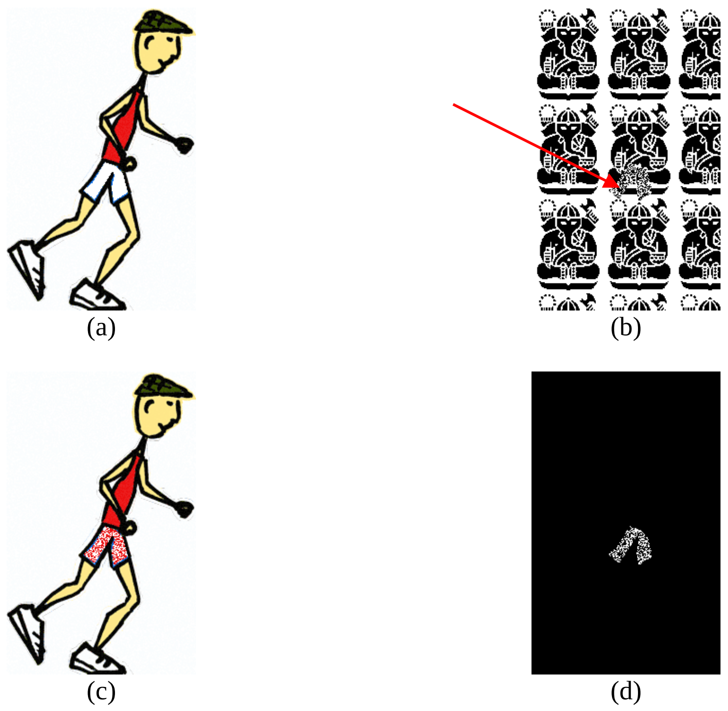


Figure 18. The second attack. (a) the watermarked GIF image has been modified by changing color of the pants from blue to white; (b) the extracted watermark contains object that has been modified. (c) identification of modified object in the watermarked image. (d) part of object which has been modified.

5. Discussion

We have done some experiments to test the performance of the proposed scheme and the results have been shown in the above section. We found that visually the watermarked GIF images, both the still image and the animated image, are similar with the original images. For the still GIF image, we have done two categories of attack to the watermarked image. The first category attack was by altering only a part of image (deletion attack and copy-and-paste attack). The second category attack was by altering the whole image (flipping attack and contrast-and-brightness-attack). For the animated GIF images, we have done two kinds of attack. The first attack was insertion of new object in each frame of the watermarked GIF image, and the second attack was by altering color of the object.

For whole of images, we succeed to identify the authentication of the images. For example, in the first category attack of the still GIF images, we found the altered object. The extracted watermark showed part of the object that altered. In the second category attack, we got the whole extracted watermark has broken. Therefore, due to some typical attacks to the watermarked image, we got the extracted watermark become to be fragile. The fragile watermark is indication that the watermarked image has been modified. We got the similar results in the animated GIF images.

Why we can detect modification in the watermark image because we embed watermark on each pixel. Once the pixel was changed or altered, then the watermark in it was also changed. We call it detection of pixel-level tampering.

6. Conclusion

In this paper, the fragile watermarking scheme based on EzStego algorithm for authentication of the GIF images has been presented. The proposed scheme can detect pixel-level tampering, since embedding of watermark is performed on every pixel of the image. Experiment results show that the scheme can detect image authentication due to the various typical attacks. The fragile watermark is indication that the watermarked image has been modified or altered.

There are two future works that can be continued. The first work is trying to implement the proposed scheme to another palette based images, for example TIFF image. The second work is embedding the different watermark into each frame of the animated GIF image randomly.

7. Acknowledgment

Research that published in this paper is fully funded by a grant for Decentralization of Directorate of Higher Education (DIKTI) with PUPT Scheme, 2016.

8. References

[1]. Alomari, R., Al-Jaber, A., "A Fragile Watermarking Algorithm for Content Authentication", *International Journal of Computing & Information Sciences*, Vol.2, No. 1, April 2004.

[2]. Walton, S., "Image Authentication for a Slippery New Age," *Dr. Dobb's Journal*, vol. 20, no. 4, pp. 18–26, 1995.

[3]. Lin, E.T., Delp, E.J., "A Review of Fragile Image Watermarks, Proceeding of the Multimedia and Security", *ACM Multimedia'99*, pp. 25-29, 1999.

[4]. Wong, P.W., 1998. "A Watermark for Image Integrity and Ownership Verification", Proceeding of The IS & T PIC Conference, 1999.

[5]. Hassan, M.A., Gillani, S.A.M, "A Fragile Watermarking Scheme for Color Image Authentication", *World Academy of Science, Engineering and Technology* 19, 2006, pp. 39-42.

[6]. Chang, C., Lin, P. "A Color Image Authentication Method Using Partitioned Palette and Morphological Operations", Journal IEICE – Transaction on Information and Systems, Vol. E91-D Issu1 1, January 2008, pp. 54-61.

[7]. R. Machado, EZStego, http://www.stego.com

[8]. A. Westfeld and A. Pfitzmann (1999). "Attack on Steganographic System", *Lecture Notes in Computer Sciences*, vol. 1768, pp. 61-76.

[9]. Dawei, Z., Guanrong, C., Wenbo, L., "A Chaos-Based Robust Wavelet-Domain Watermarking Algorithm", *Chaos Solitons and Fractals 22* (2004) 47-54.

[10]. Lampton, J., "Chaos Cryptography: Protecting Data Using Chaos", Mississippi School for Mathematics and Science.

[11]. Wu. M., Liu, B, "Watermarking for Image Authentication", Proceeding of IEEE Intl. Conf. Image Processing, vol. II, Chicago, USA, October 1998, pp. 437-441.

[12]. Rao K. M. V., Ghanekar, U., Transform domain fragile watermarking using fermat number transform, 2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC).

[13]. Li, C. T., "Digital Fragile Watermarking Scheme for Authentication of JPEG Images", Department of Computer Science, University of Warwick

[14]. Ghosal, S, "Genetic-Algorithm-Based Optimization of Fragile Watermarking in Discrete Hartley Transform Domain", *Handbook of Research on Natural Computing for Optimization Problems*, p.25, 2016.

[15]. El-Shami, F.E., *Information Security for Automatic Speaker Identification*, Springer 2011

[16]. Online Photoshop, http://www.photoshop.com/tools

[17]. https://www.mathworks.com

[18]. Margherita, P., *Encyclopedia of Multimedia Technology and Networking*, Idea Group Inc, 2005.

Rinaldi Munir received the B.Eng. degree in informatics engineering from the Institut Teknologi Bandung (ITB), Bandung, Indonesia, in 1992. In 1993, he started his academic career as a Lecturer at the Informatics Department, ITB. He obtained M.Sc. degree from ITB in 1999 in digital image compression. In 2010, he obtained the Ph.D. degree from the School of Electrical Engineering and Informatics ITB, where he studied about image watermarking. He is now an Associate Professor in the School of Electrical Engineering and Informatics, ITB and affiliated with Informatics Research Group. His research interests are cryptography and steganography-related topics, digital image processing, fuzzy logics, and numerical computation.