Reversible Watermarking Using Modified Difference Expansion

Mohammad K. Yaqub
Computer Science Dept.
The University of Jordan,
Amman, Jordan
myaqub@ju.edu.jo

Ahmed Al-Jaber
Computer Science Dept.
The University of Jordan,
Amman, Jordan
jahmed@ju.edu.jo

Abstract: *Reversible watermarking is an important requirement for many watermarking applications such as medical and military imaging. Many of the existing techniques for reversible watermarking suffer from the problem of low hiding capacity (Payload) since reversible watermarking techniques need huge amount of data to be embedded for exact recovery of the original cover. This paper proposes a reversible watermarking technique that gives a high hiding capacity compared to many well known techniques. The proposed technique proposes a modified idea of the difference expansion technique in which data is hidden in the differences, in each acceptable expandable vector within an image, that are generated from subtracting the median pixel in that vector with other pixels. The number of these expandable vectors must be large enough to hide enough amount of data, as a result selecting the base point (reference point) in a vector affects the type of that vector whether it will be accepted as an expandable one or not. The proposed technique has proved a high hiding capacity for both grayscale and color images. Experimental results has shown that selecting the median pixel in all vectors as a base point instead of the first one gives higher number of expandable vectors and, hence, higher data payload.*

Keywords: *Reversible Watermarking, Content Authentication, Fragile Watermarking, Difference Expansion.*

Received: December 26, 2005 | ***Revised***: June 15, 2006 | ***Accepted***: December 10, 2006

---

1 Introduction

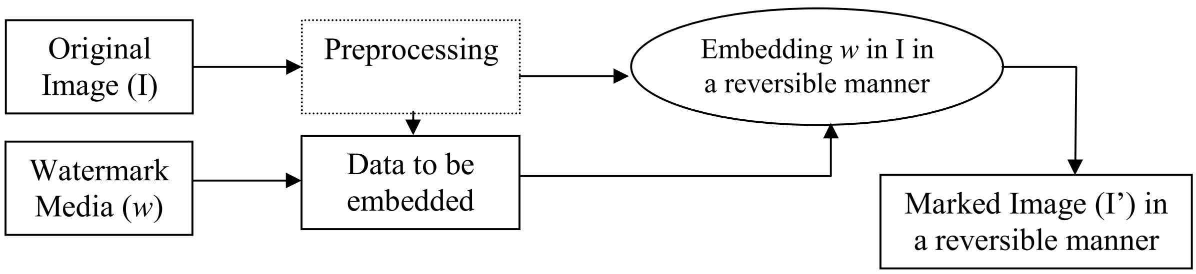
Reversible watermarking is one type of fragile watermarking. Fragile watermarking is a watermarking discipline in which the watermark is sensitive to any intentional or unintentional forge of the watermarked content; fragile watermarking techniques are the subject of many applications like the Content Authentication [3, 4, 5, 8, 14]. Reversible watermarking is a special digital watermarking with an intriguing feature that when the watermarked content has been authenticated, one can remove the watermark to retrieve the exact original, un-watermarked content [1, 2, 3, 5, 6, 7, 11]. Such reversibility to get back the exact original content is highly desired in sensitive imagery applications, such as military and medical applications. Reversible watermarking, which is also known as lossless watermarking, invertible watermarking, and erasable watermarking has been widely investigated in the recent history of watermarking applications [1 – 7, 9, 11 – 13].

There are many techniques developed to solve general or special reversible watermarking applications. Honsinger *et al.* [7] used an invertible addition to devise a low-capacity reversible watermark algorithm. He restricted the embedder to be additive and non-adaptive and included the watermarking parameters in the payload. Fridrich *et al.* [4-6] devised a reversible watermarking algorithm that does not suffer from the salt-and-pepper artifact. The algorithm compresses one of the least significant bit planes of the host image, appends an image hash and payload, encrypts the result, and finally, replaces the original bit plane with the result. The process can be reversed to obtain the original host image. Celik *et al.* [2] devised a low-distortion, reversible watermarking that is capable of embedding as high as 0.7 bits per pixel. Their algorithm first quantizes each pixel by a quantizer of step size, compresses the quantization noise and appends a payload to it, then adds an L-ary representation of the result to the quantized image. Tian [11] described a high capacity and high quality reversible watermarking method based on difference expansion. A noticeable difference between his method and others is that there is no need to compress original values of the embedding area. He explores the redundancy in the digital content to achieve reversibility. Alattar [1] developed a reversible watermarking algorithm with very high data-hiding capacity for color images. The algorithm allows the watermarking process to be reversed, which restores the exact original image. The algorithm hides several bits in the difference expansion of vectors of adjacent pixels.

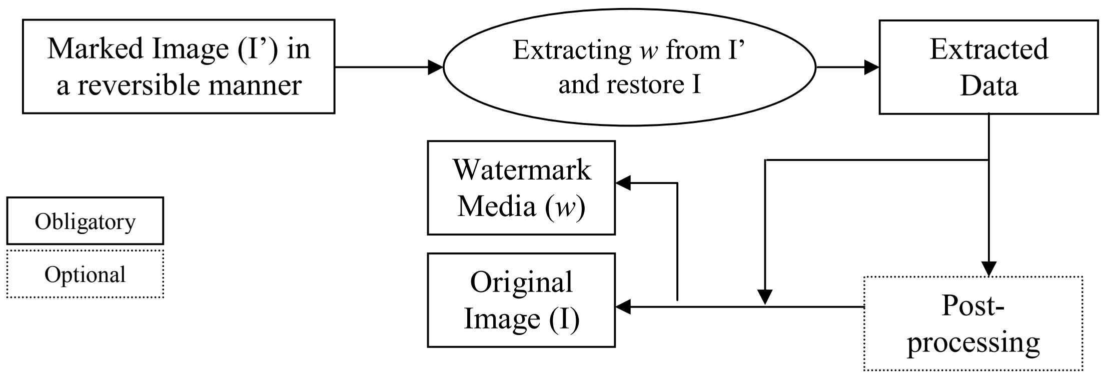
Reversibility gives the ability to retrieve the exact original input data after the extraction process. Therefore, any reversible watermarking system should comprise the following steps: (i) Embedding a digital watermark signal w in a digital host signal x resulting in $y = x + w$, (ii) transmitting the watermarked signal y from the embedder to the decoder extractor through an error-free transmission channel, and (iii) extracting the watermark signal w and restoring the original host $x = y - w$ [11].

In imaging applications of reversible watermarking, a watermark is embedded in a digital image I *to* obtain the watermarked image I' using some reversible watermarking technique. After the transmission process via the communication channel, the received watermarked image is subjected to the authenticator which performs dual roles; first it verifies whether the received watermarked image is authentic, if yes, it then reverses the original image using the extractor of the watermarking technique. Figure 1, shows a guideline of an authentication system using a reversible watermarking algorithm.

The rest of the paper is organized as follows: section 2 is a preparation of the notations used in this paper with the theory behind the proposed work. Section 3 gives the proposed reversible watermarking technique. Section 4 motivates the theory behind the data payload of the watermark. Section 5 contains the analysis of the proposed technique along with comparisons with pre-existing techniques. Experimental results and conclusions are shown in sections 6 and 7 respectively.



(a) Embedding Process



(b) Extraction Process

Figure 1: General Framework for Reversible watermarking, (a) Embedding Process, (b) Extraction Process

2 Preparation

Before going forward, we need to define some variables to be used later. According to [1], let vector $u = (u_0, u_1, ..., u_{N-1})$ be a vector formed from N pixel values chosen from N different locations within the same color component according to a predetermined order. The size of the vector equals to $a \times b$, such that $1 \leq a \leq h$, $1 \leq b \leq w$, and $a + b \neq 2$ where w and h are the width and the height of the host image, respectively. The vector is constructed by taking a row major, row by row, from the block $a \times b$.

Because we are using color images we require that each color component be treated independently and, hence, have its own set of vectors. This also increases the chance that vector values are close to each other since they are taken from the same color channel. In addition, we require that vectors do not overlap each other; i.e., each pixel exists in only one vector. These requirements may be removed at the expense of complicating the watermarking algorithm due to the extra caution required to determine the processing order of the overlapped vectors [1].

Reversible Integer Transform: According to Alattar's technique [1], the forward difference expansion transform $f(\bullet)$ for the vector $u = (u_0, u_1 ... u_{N-1})$ is defined as

$$v_m = \left\lfloor \frac{\sum_{i=0}^{N-1} a_i u_i}{\sum_{i=0}^{N-1} a_i} \right\rfloor \quad .....(1)$$
$$v_k = u_k - u_m \quad k = \{0,..., m-1, m+1,..., N-1\}$$

Where $\lfloor \bullet \rfloor$ is the least nearest integer and a_i is a constant integer. Obviously, v_m is the weighted average of the entities of the vector u, whereas $v_0, ..., v_{m-1}, v_{m+1}, ..., v_{N-1}$ are the differences between $u_0, ..., u_{m-1}, u_{m+1}, ..., u_{N-1}$ and u_m, respectively. m is the index of the median value within the vector u. So that u_m is the median pixel value within a vector.

The inverse difference expansion transform $f^{-1}(\bullet)$ for the transformed vector $v = (v_0, v_1, v_2, ..., v_{N-1})$ is defined as

$$u_m = v_m - \left\lfloor \frac{\sum_{i=0}^{m-1} a_i v_i + \sum_{i=m+1}^{N-1} a_i v_i}{\sum_{i=0}^{N-1} a_i} \right\rfloor \quad ......(2)$$
$$u_k = v_k + u_m \quad k = \{0,..., m-1, m+1,..., N-1\}$$

Lemma 1[*]: The vector $u = (u_0, u_1, ..., u_{N-1})$ is said to be expandable if, for all $b_1, b_2, ..., b_m, b_{m+1}, ..., b_{N-1} \in \{0,1\}$, $v = f(u)$ can be modified to produce $\tilde{v} = (\tilde{v}_0, ..., \tilde{v}_{m-1}, v_m, \tilde{v}_{m+1}, ..., \tilde{v}_{N-1})$ according to equation (3) below without causing overflow or underflow in $\tilde{u} = f^{-1}(\tilde{v})$

$$v_m = \left\lfloor \frac{\sum_{i=0}^{N-1} a_i u_i}{\sum_{i=0}^{N-1} a_i} \right\rfloor \quad .....(3)$$
$$\tilde{v}_k = 2 \times v_k + b_j$$
$$k = \{0,..., m-1, m+1,..., N-1\} \quad \& \quad j = \{1,.., N-1\}$$

It should be noticed here that the above modification only changes the differences of the vector u. Each of $\tilde{v}_0, ..., \tilde{v}_{m-1}, \tilde{v}_{m+1}, ..., \tilde{v}_{N-1}$ is a 1-bit left-shifted version of the original value $v_0, ..., v_{m-1}, v_{m+1}, ..., v_{N-1}$, respectively, but potentially with a different Least Significant Bit (LSB). The weighted average v_m of u remains unchanged.

To prevent overflow and underflow, one must make sure that the following conditions hold:

$$0 \leq v_m - \left\lfloor \frac{\sum_{i=0}^{m-1} a_i \tilde{v}_i + \sum_{i=m+1}^{N-1} a_i \tilde{v}_i}{\sum_{i=0}^{N-1} a_i} \right\rfloor \leq 255$$

$$0 \leq \tilde{v}_k + v_m - \left\lfloor \frac{\sum_{i=0}^{m-1} a_i \tilde{v}_i + \sum_{i=m+1}^{N-1} a_i \tilde{v}_i}{\sum_{i=0}^{N-1} a_i} \right\rfloor \leq 255 \quad .....(4)$$
$$k = \{0,..., m-1, m+1,..., N-1\}$$

Lemma 2[*]: The vector $u = (u_0, u_1, ..., u_{N-1})$ is said to be changeable if, for all $b_1, b_2, ..., b_m, b_{m+1}, ..., b_{N-1} \in \{0, 1\}$, $v = f(u)$ can be modified to produce $\tilde{v} = (\tilde{v}_0, \tilde{v}_1, ..., \tilde{v}_{m-1}, v_m, \tilde{v}_{m+1}, ..., \tilde{v}_{N-1})$ according to equation (5) below without causing overflow or underflow in $\tilde{u} = f^{1}(\tilde{v})$ by satisfying (4)

$$v_m = \left\lfloor \frac{\sum_{i=0}^{N-1} a_i u_i}{\sum_{i=0}^{N-1} a_i} \right\rfloor \quad \tilde{v}_k = 2 \times \left\lfloor \frac{v_k}{2} \right\rfloor + b_j$$
$$k = \{0,..., m-1, m+1,..., N-1\} \quad \& \quad j = \{1,.., N-1\} \quad ..... (5)$$

[*] Lemma *1* and *2* are the same as in Alattar's technique with slight modifications.

Again, the above modification only changes the differences of the vector u. $\tilde{v}_0, \tilde{v}_1, \ldots, \tilde{v}_{m-1}, \tilde{v}_{m+1}, \ldots, \tilde{v}_{N-1}$ are the same as the original $v_0, v_1, \ldots, v_{m-1}, v_{m+1}, \ldots, v_{N-1}$, but potentially with different LSBs. The weighted average v_m of u remains unchanged.

According to lemma *2*, it's noticed that a changeable vector remains changeable even after changing the LSBs of its $v_0, v_1, \ldots, v_{m-1}, v_{m+1}, \ldots, v_{N-1}$. Also, from lemmas *1* and *2*, it can be observed that an expandable vector is also changeable.

According to [1], let $I(i, j, k)$ be an RGB image where i and j are the two spatial indices and k is the temporal index, and it has three values: Red, Green, and Blue. We also assume that each color component is treated independently. Also, let the set $U = \{U_k, r = 1 \ldots K\}$ represent any of the above set of vectors U_{Red}, U_{Green}, and U_{Blue}, and let K represent its associated security key. In addition, let $V = \{v_k, r = 1 \ldots K\}$ be the transformation of V under the difference expansion transform $f(\bullet)$ (i.e., $V = f(U)$ and $U = f^{-1}(V)$). And let $\boldsymbol{u} = (u_0, u_1, \ldots, u_{N-1})$ and its difference expansion transform be $\boldsymbol{v} = (v_0, v_1, \ldots, v_{N-1})$.

The vectors in U now can be classified into three groups according to lemmas *1* and *2*. The first group S_1 contains all expandable vectors whose $v_0 \le T_0$, $v_1 \le T_1$, …, $v_{m-1} \le T_{m-1}$, $v_{m+1} \le T_{m+1}$, …, $v_{N-1} \le T_{N-1}$, where $T_0, T_1, \ldots, T_{m-1}, T_{m+1}, \ldots, T_{N-1}$ are predefined thresholds where m is the index of the median pixel within the vector. The second group S_2 contains all changeable vectors that are not in S_1. The third group S_3 contains the rest of the vectors (not changeable). Also, let S_4 denote all changeable vectors (i.e., $S_4 \equiv S_1 \cup S_2$).

We also adopted the following variables from [1]. Consider the vectors of S_1, S_2, and S_3 using a binary location map M whose entries are 1s and 0s, where the 1 symbol indicates the vectors S_1 and the 0 symbol indicates the S_2 or S_3 vectors. Depending on how the vectors are formed, the location map can be one-dimensional (1-D) or two-dimensional (2-D).

3 Proposed Technique

The proposed technique uses a modified version of the idea of difference expansion in [1]. We chose the median pixel as a base point in a vector to decrease the values in the differences for that vector, so that increasing the storage capacity as a result of increasing the number of expandable vectors.

3.1 Embedding a Reversible Watermark

To embed data in a reversible way the following steps should be done for every $U \in \{U_R, U_G, \text{and } U_B\}$.

- Form the set of vectors U from the image $I(i, j, k)$.
- Calculate V using the reversible integer transform $f(\bullet)$ as in equations (1).
- Use V, equations (3) and (5) as well as the conditions in (4) to divide U into the sets S_1, S_2, S_3, and S_4,
- Form the location map M; then compress it using a lossless compression algorithm, such as JBIG or an arithmetic compression algorithm [10], to produce sub-bitstream B_1. Append a unique identifier, end-of-stream (EOS) symbol to B_1 to identify the end of B_1.
- Extract the LSBs of $v_0, v_1, \ldots, v_{m-1}, v_{m+1}, \ldots, v_{N-1}$ of each vector in S_2. Concatenate these bits to form sub-bitstream B_2.
- Assume the watermark to be embedded forms a sub-bitstream B_3, and concatenate sub-bitstreams B_1, B_2, and B_3 to form the bitstream B.
- Sequence through the member vectors of S_1 and S_2 as they occur in the image and through the bits of the bitstream B in their natural order. For S_1, expand the vectors as described in equation (3). For S_2, expand the vectors as in equation (5). The values of $b_1, b_2, \ldots, b_m, b_{m+1}, \ldots, b_{N-1}$ are taken sequentially from the bitstream B.
- Calculate the inverse, reversible, integer transform of the resulting vectors using $f^{-1}(\bullet)$, as in equations (2), to produce the watermarked S_1^w and S_2^w.
- Replace the pixel values in the image, $I(i, j, k)$, with the corresponding values from the watermarked vectors in S_1^w and S_2^w to produce the watermarked image $I^w(i, j, k)$.

It's noticed here that the size of bitstream B must be less than or equal to N-1 times the size of the set S_4. To meet this condition, the values of the threshold $T_1, T_2, \ldots, T_{N-1}$ must be set properly. Also, the algorithm is not limited to RGB images. Using the RGB space in the previous discussion was merely for illustration purposes, and using the algorithm with other color image models is straightforward.

3.2 Reading a Watermark and Restoring the Original Image

To retrieve the watermark and restore the original image, the following steps should be followed for every $U \in \{U_R, U_G, U_B\}$.

- Form the set of vectors U from the image $I^w(i, j, k)$.
- Calculate V using the reversible integer transform, $f(\bullet)$ as in equations (1).

- Use V, equation (5), and the conditions in (4) to divide the vectors in U into two sets $\hat{S}_4$ and S_3. $\hat{S}_4$ has the same vectors as S_4, which was constructed during embedding, but the values of the entities in each vector may be different. Similarly, S_3 is the same set constructed during embedding, since it contains non-changeable vectors.
- Extract the LSBs of $\tilde{v}_0$, $\tilde{v}_1$, …, $\tilde{v}_{m-1}$, $\tilde{v}_{m+1}$, …, $\tilde{v}_{N-1}$ of each vector in $\hat{S}_4$, and concatenate them to form the bitstream B, which is identical to that formed during embedding.
- Identify the EOS symbol and extract sub-bitstream B_1. Then, decompress B_1 to restore the location map M, and, hence, identify the member vectors of the set S_1 (expandable vectors). Collect these vectors into set $\hat{S}_1$.
- Identify the member vectors of S_2. They are the members of $\hat{S}_4$ which are not members of $\hat{S}_1$. Form the set $\hat{S}_2 = \hat{S}_4 - \hat{S}_1$.
- Sequence through the member vectors of $\hat{S}_1$ and $\hat{S}_2$ as they occur in the image and through the bits of the bitstream B in their natural order after discarding the bits of B_1. For $\hat{S}_1$, restore the original values of $v_1, v_2, \ldots, v_{N-1}$ as follows:

$$v_i = \left\lfloor \frac{\tilde{v}_i}{2} \right\rfloor \quad i = \{0,1,\ldots,m-1,m+1,\ldots N-1\} \quad \ldots\ldots (6)$$

- For $\hat{S}_2$, restore the original values of $v_0, v_1, \ldots, v_{m-1}, v_{m+1}, \ldots, v_{N-1}$ according to equation (5). The values of $b_1, b_2, \ldots, b_m, b_{m+1}, \ldots, b_{N-1}$ are taken sequentially from the bitstream.
- Calculate the inverse reversible integer transform of the resulting vectors using $f^{-1}(\bullet)$ to restore the original S_1 and S_2 as shown in equation (2)].
- Replace the pixel values in the image $I^w(i, j, k)$ with the corresponding values from the restored vectors in S_1 and S_2 to restore the original image $I(i, j, k)$.
- Discard all the bits in the bitstream B, which were used to restore the original image. Form the sub-bitstream B_3 from the remaining bits. Read the payload and authenticate the image using the watermark contained in B_3.

4 Payload Size

In this section we adopted the definition of finding the payload, which is the amount of information that can be hidden with a source, from [1]. To be able to embed data into the host image, the size of the bitstream B must be less than or equal to N-1 times the size of the set S_4. This means that

$$\|S_1\| + \|S_2\| = \frac{\|B_1\| + \|B_2\| + \|B_3\|}{N-1} \quad \ldots\ldots\ldots\ldots (7)$$

Where $\|X\|$ indicates the number of elements in *X*. But $\|B_2\| = (N-1)\|S_2\|$; hence, equation (7) can be reduced to

$$\|B_3\| = (N-1)\|S_1\| - \|B_1\| \quad \ldots\ldots\ldots\ldots (8)$$

For Tian's technique [11], the bitstream size is $\|B_3\| = \|S_1\| - \|B_1\|$, which can be obtained from equation (8) by setting N=2.

Equation (8), above, indicates that the size of the payload that can be embedded into a given image depends on the number of expandable vectors that can be selected for embedding and on how well their location map can be compressed. So in this paper we focused on how to increase the number of expandable vectors.

With $w \times h$ host image, the algorithm would generate $\frac{w \times h}{N}$ vectors. Only a portion, α (0≤ α ≤1), of these vectors can be selected for embedding; i.e. $\|S_1\| = \alpha \frac{w \times h}{N}$. Notice here, the value α is higher in the modified technique than the one in Alattar's technique [1]. As a result, the value of $\|S_1\|$ becomes larger (increasing the number of expandable vectors), and hence, increasing the hiding capacity. Theoretically, the value of alpha depends on the number of accepted expandable vector, so the proposed technique will hopefully have higher number of expandable vectors than Alattar's one [1] since the differences for each vector will be most likely smaller and this means that these vectors will satisfies the given thresholds. Equations (9 & 10) show the first required condition in both Alattar & our techniques, respectively. From these two equations we can notice that the less value of v_k the more likely for the vector to accepted as expandable. Therefore, our technique will most likely give smaller differences and as a result more expandable vectors. Both techniques will give the same payload if the first pixel of all expandable vectors is the median within that vector.

$$v_i = u_i - u_0 \quad if\, v_i \le T_i \Rightarrow U \in S_1 \quad \forall i = \{1,\ldots N-1\} \quad \ldots (9)$$

$$v_j = u_j - u_m \quad if\, v_j \le T_j \Rightarrow U \in S_1 \quad \forall j = \{0,\ldots m-1, m+1,\ldots,N-1\} \quad \ldots (10)$$

Also, the algorithm would generate a binary map, M, containing $\frac{w \times h}{N}$ bits. This map can be compressed using lossy compression by a factor β (0≤ β ≤1). This means that $\|B_1\| = \beta \frac{w \times h}{N}$. Ignoring the unchangeable vectors (i.e., assuming $\|S_3\| = 0$) and using equation (8), the potential payload size (in bits) becomes

$$\|B_3\| = (N-1)\alpha \frac{w \times h}{N} - \beta \frac{w \times h}{N}$$
$$= \left(\frac{N-1}{N}\alpha - \frac{1}{N}\beta \right) \times w \times h \qquad \text{........(11)}$$

5 Analysis and Comparisons

The proposed technique focuses mainly on changing the reference point from which we calculate the differences. Alattar [1] chose the first pixel in the vector as a reference point. His technique checks all expandable vectors for two conditions. First, the technique checks whether the differences overshoot some predefined thresholds. Second, if the first condition is satisfied then the technique should check whether the new calculated vector *u* overflows 255 or underflows 0 after multiplying the differences from v_1 to v_{N-1} by two. These thresholds in the first condition help preserve the image quality (preserve the image from distortion) after embedding. If these thresholds are small, then the distortion of the image will be small, which is good, but, in contrast, the number of expandable vectors will be decreased and hence decreasing the storage capacity. As a result, our technique concentrates on increasing the storage capacity by increasing the number of expandable vectors without making more distortion.

Alattar's technique has lower payload compared to our technique because of choosing the first pixel as a reference point. When the differences between the first pixel and the other ones in the vector are calculated, these differences vary depending on the value of the first pixel. In most cases, adjacent pixels have values that are almost the same because of image uniformity. But when pixels' values are far away from each others then this uniformity becomes smaller. This note implies that when we calculate the differences then the probability of getting smaller differences between the first value and its adjacent ones is higher than the probability of getting smaller differences between the first value and its nonadjacent ones (adjacency metric depends on different factors like the way vectors are taken and the size of the vectors). This implies that the vector may not satisfy the first condition to be expandable because some differences may overshoot their thresholds and hence, the vector will not be considered as expandable.

In contrast, the proposed technique chooses another pixel as a reference point to decrease the probability of high differences. When another point is chosen as a reference point that is closer enough to the median point, the differences will be much smaller because other points will not deviate as before. This implies that the probability of this vector to be an expandable one will be better than before (when choosing the first point as a reference one).

The most important property regarding this paper is the hiding capacity (Payload). The proposed technique gives higher hiding capacity compared to well known techniques such as the technique proposed by Alattar [1]. However, the proposed technique introduces more distortion compared to other techniques. Experimental results show that this distortion is acceptable in many applications since the original content can still be retrieved. For more information about the distortion see section 6.

6 Experimental Results

Experimental results show that the number of expandable vectors in the proposed technique is larger than the one in Alattar's technique [1]. Table 1, shows the number of expandable, changeable, and non-changeable vectors in the proposed technique compared to Alattar's technique with different vector sizes (different *N* values) for both Lena and Baboon images. It also shows the hiding capacity approximated in bytes for both techniques according to equation (11). But this hiding capacity ($\|B_3\|$) depends on the value of β, and the value of β depends highly on the compression technique and how data is sequenced, so the value of β in the following table is assumed to be ½, this value of compression can be done by many compression techniques [10]. Moreover, both images are gray scale images and their dimensions are 256×256. Also, the thresholds values are assumed to be 25. This means when a vector contains an absolute difference with a value higher than 25 then it is not assumed to be expandable. Vectors are assumed to be taken sequentially from the image.

From table 1, we conclude that our technique gives better hiding capacity compared to Alattar's technique, which is the important issue in reversible watermarking. Figure 2 shows the original Lena and Baboon (256 × 256) images, the watermark image (43 × 49), the watermarked Lena and Baboon using Alattar's technique, and the watermarked Lena and Baboon using our technique. Vector size is assumed to be 16 pixels and thresholds are assumed to be 25 for all the differences.

Alattar's technique indicates that the algorithm is effective when N and the number of selected expandable vectors are reasonably large. In this case, it does not matter if the binary map, M, is difficult to compress (which is because its size is very small). But, when each vector is formed from N consecutive pixels (row- or column-wise) in the image, and when N is large, the number of expandable vectors may decrease substantially; consequently, the values of the thresholds T_1, T_2,…, T_{N-1} must be increased to maintain the same number of selected expandable vectors. This increase causes a decrease in the quality of the embedded image. In contrast, our technique needs not to be like Alattar's one because it gives higher number of expandable vectors even using small values of thresholds to maintain the quality of the image. In this case, our technique becomes more suitable for high SNR (Signal to Noise Ratio) embedding than for low SNR embedding.

We also calculated Peak Signal-to-Noise Ratio (PSNR) which is the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation [15]. Equation (12) shows how to calculate PSNR:

$$PSNR = 20 * \log_{10}^{(b/RMS)} \quad ....(12)$$

Where b is the largest possible value of the signal (image) and RMS is the Root Mean Square error. The PSNR is given in decibel units (dB), which measure the ratio of the peak signal and the difference between two images. An increase of 20 dB corresponds to a ten-fold decrease in the RMS difference between two images. For more information about PSNR and different fidelity measurements see [15].

We found that our technique gives a little bet less PSNR compared to [1]. Table 2 shows a comparison between our technique with Alattar's technique for both Lena and Baboon images shown in figure 2. Reader can notice that this difference is small and visually undistinguishable.

Table 1: Comparison between Alattar's technique and our technique upon the number of vectors produced and the obtained hiding capacity

Image	Alattar's Technique					Proposed Technique				
	N	Exp.	Chang.	Non	Payload In Bytes	N	Exp.	Chang.	Non	Payload In Bytes
Lena 256 × 256 Gray Scale Image	**2**	29680	3072	16	1662	**2**	29680	2987	101	1662
	4	11921	4434	29	3446	**4**	12854	3470	60	3796
	8	4264	3909	19	3219	**8**	4744	3392	56	3639
	16	1294	2788	14	2170	**16**	1551	2492	53	2652
	32	272	1764	12	926	**32**	313	1689	46	1084
	64	22	997	5	109	**64**	25	957	42	132
	128	0	460	52	-32	**128**	0	460	52	-32
Baboon 256 × 256 Gray Scale Image	**2**	22448	10308	12	758	**2**	22448	10284	36	758
	4	6414	9962	8	1381	**4**	7703	8670	11	1864
	8	1670	6516	6	949	**8**	2291	5896	5	1492
	16	319	3774	3	342	**16**	527	3566	3	732
	32	30	2016	2	-12	**32**	49	1998	1	61
	64	0	1024	0	-64	**64**	0	1022	2	-64
	128	0	511	1	-32	**128**	0	511	1	-32

Table 2. Comparison between Alattar's technique and our technique with respect to PSNR. Images are shown in figure 2

	Alattar's technique	Proposed technique
Lena image	+33.91 dB	+33.38 dB
Baboon image	+33.28 dB	+32.07 dB

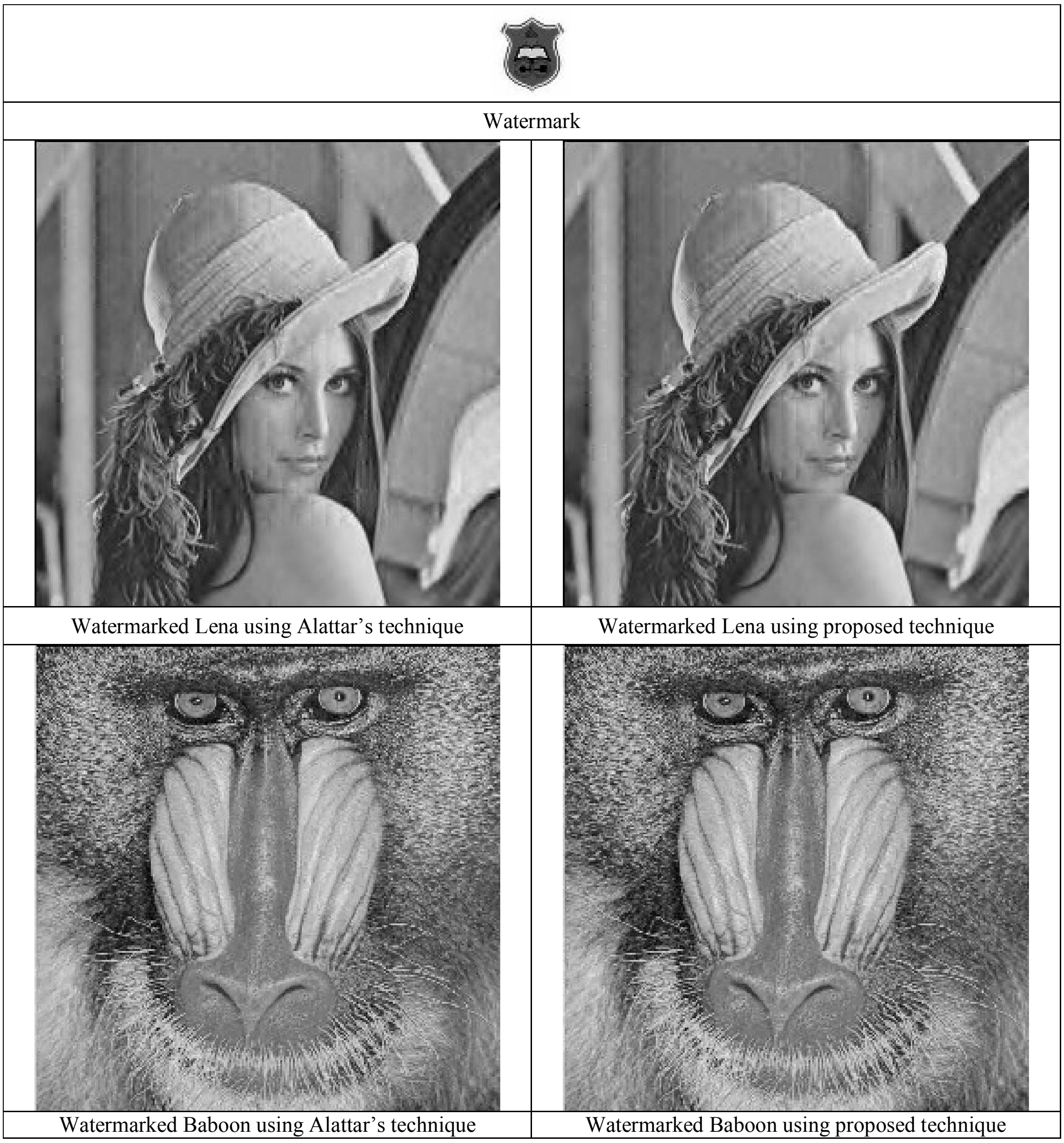


Figure 2: Application of Both Techniques: Alattar's and ours to both Lena and Baboon images for vector of size 16 taken sequentially

In addition, the proposed technique takes more time compared to Alattar's technique [1]. Table 3 shows the running time in seconds for both techniques in both Lena and Baboon images. The watermark and the images are the same as in figure 2 and the vector size is also 16. We got these results after running both techniques using MATLAB 6.5 on a Pentium 4 desktop computer with 2.8 Giga Hertz (GH) Intel processor and one Giga Byte (GB) memory. From the results we can see that there is no big difference between both techniques and both of them can be used for real time applications. From table 3, we can see that Lena image takes less time compared to Baboon image. This happens because its pixel values are more consistent and they are close to each other; therefore, computations time will be less.

Table 3. Comparison between both techniques with respect to the required running time (in seconds)

	Alattar's technique	Proposed technique
Lena image	5.2	5.6
Baboon image	6.8	8.9

7 Conclusions

This paper introduces a reversible watermarking technique with a high hiding capacity compared to well known techniques like Celik's, Tian's, and Alattar's. Using the idea of difference expansion with any vector size would become more efficient when we choose the median pixel within a vector instead of the first one as a base point. Median pixel value within a vector has less deviation with the other pixels in that vector and, hence, the newly calculated differences will be much smaller than the ones proposed by Alattar. Experimental results show that the proposed technique needs more calculation compared to other techniques but these calculations are justifiable because of the huge hiding capacity generated. Moreover, the proposed technique helps preserve the image quality (cover) after hiding huge amount of data because of the use of thresholds as in Alattar's technique but with intelligent behavior. This implies that the system adopts these thresholds depending to the original image and the needed payload to hide some watermark.

References

[1] Alattar A. "Reversible Watermark Using the Difference Expansion of a Generalized Integer Transform." *IEEE Transactions On Image Processing*, VOL. 13, NO. 8, August 2004.

[2] Celik M., Sharma G., Tekalp A. M., and Saber E. "Reversible Data Hiding." in Proc. *IEEE Int. Conf. Image Processing*, vol. II, pp. 157–160, Sept. 2002.

[3] Cox J., Miller L., Bloom A., *Digital Watermarking.* (1st edition). USA: Morgan Kaufmann Publishers, 2002.

[4] Fridrich M. and Baldoza A., "New Fragile Authentication Watermark for Images." in Proc. of IEEE International Conference on Image Processing (Vancouver, Canada), Sep. 10-13, 2000.

[5] Fridrich J., Goljan M., and Du R., "Invertible Authentication**.**" in Proc. *SPIE Photonics West, Security and Watermarking of Multimedia Contents III*, vol. 3971, San Jose, CA, pp. 197–208, Jan. 2001.

[6] Fridrich J., Goljan M., and Du R., "Lossless data embedding - new paradigm in digital Watermarking." *EURASIP Journal on Applied Signal Processing*, vol. 2, no. 2, pp. 185-196, Feb. 2002.

[7] Honsinger C. W., Jones P. W., Rabbani M., and Stoffel J. C., "Lossless Recovery of an Original Image Containing Embedded Data." *U.S. Patent* 6,278,791, 2001.

[8] Lin E., and Delp E., "A Review of Fragile Image Watermarks." in Proc. of *the Multimedia and Security Workshop (ACM Multimedia '99)***,** 25-29, 1999.

[9] Mohanty S., "Watermarking of Digital Images." *Master Project Report***,** *Dept. of Electrical Engineering*, India, Institute of Science, Dangalore, - 560 012, India, 1999.

[10] Sayood K., *Introduction to Data Compression.* Second Edition, USA: Morgan Kaufmann Publishers, 2000.

[11] Tian J., "Reversible watermarking by Difference Expansion." in Proc. *Workshop on Multimedia and Security: Authentication*, Secrecy, and Steganalysis, J. Dittmann, J. Fridrich, and P.Wohlmacher, Eds., pp. 19–22, Dec. 2002.

[12] Yeung M., "An Invisible Watermarking Technique for Image Verification.", in Proc. of *ICIP*, pp. 680 683, 1997.

[13] Yeung M., and Mintzer F.; "Invisible Watermarking for Image Verification." *Journal of Electronic Imaging*, 7(3). 578-591, 1998.

[14] Zhao Y., "Dual Domain Semi-Fragile Watermarking for Image Authentication." *Master Thesis***,** University of Toronto, 2003.

[15] Yuval Fisher. *Fractal Image Compression: Theory and Application*, Springer Verlag, New York, 1995.

Mohammad Yaqub is a PhD student at Wayne State University, Michigan, US. He obtained his MSc and BSc degrees in Computer Science from the University of Jordan, Amman, Jordan. His research interests are Watermarking, Steganography, Security, Image Processing and Computer Assisted Surgery applications. His PhD. research area is in Computer Assisted Surgery and medical imaging field.

Ahmed Al-Jaber is professor in the Computer Science Department, King Abdullah II School for Information Technology, University of Jordan. Research Interests are in the fields of Steganography, Watermarking, and Algorithms.