Chapter 5

Fragile Watermarking Scheme

This chapter provides a brief overview of different watermarking and particularly fragile watermarking methodologies developed by scientists over the years and then presents the current fragile watermarking scheme. A number of surveys on fragile watermarking are published at regular intervals [65, 67, 132, 144] that enabled a focused look into the recent trends in this field. A brief review follows.

5.1 Watermarking

Recent proliferation and success of the Internet, together with availability of relatively inexpensive digital recording and storage devices has created an environment in which it has become very easy to obtain, replicate and distribute digital content without any loss in quality.

However the ease of copying and editing also facilitates unauthorized use, misappropriation, and misrepresentation [110]. This is a matter of great concern to the multimedia content (music, video, and image) publishing industries [133]. Many publishers, artists, and photographers, are therefore unwilling to distribute pictures over the Internet due to a lack of security; images can be easily duplicated and distributed without the owner’s consent. Digital watermarks have been proposed as a way to tackle this tough issue [70].

Authentication of message follows MAC method as discussed earlier but authentication of multimedia content such as image, audio can not follow MAC method directly. The drawback is that the external authenticator is easy to strip off, and may be lost after a format conversion. In some applications, it may require a complex system to manage authentication data. A watermarking approach, on the other hand, provides a persistent link between the authenticator and the content it authenticates. No or minimal management of additional data is needed for authentication [250].

It is also much easier to locate tampered areas with watermarks. But a water-
marking approach is generally more complex and harder to design, and the media to be authenticated has to be modified to insert a watermark [250].

5.1.1 Definition, Categorization & Applications

Watermarking is the act of embedding another signal (the watermark) into an image, this embedding protects owners rights [166]. The process of embedding the watermark requires modifying the original image and in essence the watermarking process inserts a controlled amount of “distortion” in the image [110]. This “distortion” may be a small image called watermark image. The recovery of this distortion allows the one to identify the owner of the image. Thus image authentication as well as copyright protection of image can be achieved through watermarking.

Other than image there are media like audio, video where watermarking can take place. In this chapter we have concentrated on still image watermarking and subsequently this image watermarking is simply referred as watermarking.

Many types of watermarks have been developed for a variety of applications. Watermarks may be visible or invisible, where a visible mark is easily detected by observation while an invisible mark is designed to be transparent to the observer and detector [166]. Invisible watermarked content appears perceptually identical to the original. The existence of an invisible watermark can only be determined using an appropriate watermark extraction or detection algorithm [123].

Visible watermarks are inserted at currency, journals, digital libraries etc. because in these cases the embedded watermark image is intended to be observed [129]. Invisible or transparent marks use the properties of the human visual system to minimize the perceptual distortion in the watermarked image [198, 166]. In this type of watermarking, watermark image is so embedded in host image that it becomes invisible, hence authentication or copyright protection of image are achieved without hampering image quality as well as its secrecy.

In the class of transparent watermarks one may further categorize techniques to be fragile or robust [110].

- A fragile mark is designed to detect slight changes to the watermarked image with high probability. The main application of fragile watermarking is in content authentication [110, 250].
- A robust mark is designed to resist attacks that attempt to remove or destroy the mark. Such attacks include lossy compression, filtering, and geometric scaling [110].
- Semi-fragile watermarking is robust to some perceptual quality preserving manipulations (lossy compression) but fragile to others [250].

Multimedia integrity could be best categorized into different types of multimedia au-
thentication. Hard authentication, usually based on fragile watermarks to detect any modification to the underlying signal, has received significant coverage in the literature and is the most mature authentication approach to date. Soft authentication, frequently based on semi-fragile watermarks to measure signal modification within perceptual tolerance, is still in a relatively early research stage [250]. In this thesis we have developed one fragile and two semi-fragile watermarking schemes. In this chapter we describe the fragile watermarking scheme.

Some salient features of multimedia authentication principle function of fragile watermarking are:

- Integrity verification.
- Tamper localization and estimation
- Security
- Origination verification
- Non-repudiation

There are some specific requirements [243, 250] for watermarking-based authentication:

- Perceptual transparency - The embedded watermark should not be perceptible under normal observation or should not interfere with the functionality of the multimedia.
- Statistically invisible - This is a requirement only for the approach whose security depends on secret watermarking. Many watermarking schemes use public watermarking so that everybody can extract the embedded data.
- Blind extraction - For media authentication, the original signal is not available to the verifier. This means that the watermarking extraction should be blind, i.e., there is no access to the original signal.

Watermarking an object discourages intellectual property theft or, when such theft has occurred, allows us to prove ownership [45]. The watermarks must be designed to be unrecognizable by unauthorized people and to be identified by the legal copyright owner of the image [89]. Different applications may require different types of authentication. A medical image database may need hard authentication, while audiovisual entertainment may require soft authentication [250]. Fragile watermarking is important for some special and specific applications like authentication of medical, defense, governments images - which under any circumstance should not go through any type of data alteration process like lossy compression.
5.2 Fragile Watermarking

A brief overview explaining fragile watermarking as well as the highlights of the research directions in the area of fragile watermarking is presented next.

A fragile watermark is a mark that is readily altered or destroyed when the host image is modified through a linear or nonlinear transformation [247]. The sensitivity of fragile marks to modification leads to their use in image authentication [110]. Image authentication systems have applicability in law, commerce, defense, and journalism. Since digital images are easy to modify, a secure authentication system is useful in showing that no tampering has occurred during situations where the credibility of an image may be questioned [110].

Fragile watermarking systems embed the mark directly in the spatial domain of an image, such as techniques described in Walton [219] and van Schyndel et al. [213]. The significant disadvantages of the schemes include the ease with which their security can be bypassed [66, 247, 250]. Yeung and Mintzer [247] proposed a simple and fast fragile image watermark scheme. Fridrich et al. [68] proposed an attack on the Yeung-Mintzer watermark method [247].

P. Wong describes another fragile watermarking technique in [240], which obtains a digest using a hash function. Later in [241] P. Wong and N. Menon gave a modified version, henceforth this method will be referred to as Wong-Menon’s method. But the scheme is weak to resist Holliman-Memon attack [83].

Inspired by Wong-Menon’s method [241], current author has implemented a fragile watermarking scheme which uses CA based keyed one-way hash function. This scheme provides better image quality of watermarked image to that of Wong-Menon’s method at [241] and moreover can resist Holliman-Memon attack.

5.2.1 CAA for Fragile Watermarking

Image authentication verifies the originality of an image by detecting any attempted manipulations. This treats any intentional or unintentional attacks in the same way thus this scheme is not resistant to lossy compression. In this chapter such an image authentication technique which embeds digital “watermarks” into images is proposed.

Authenticity will not be preserved even if a single pixel of the image is changed. In proposed watermarking, it allows an user with an appropriate secret key to verify the authenticity, integrity and ownership of an image. If the user performs the watermark extraction with an incorrect key or an image which is not watermarked, the user obtains an image that resembles noise [65, 241].

Recent systems apply sophisticated embedding mechanisms, including cryptographic hash functions to detect changes to a watermarked image. Proposed scheme is based on such CA based hash functions, which are widely used in cryptosystems. As described in the previous chapter SACA generates efficient one-way hash function that
There are three broad steps: Derive_Data, Embedding_Watermark and Derive_Image.

Figure 5.1: Watermark insertion for fragile watermarking

has been employed for CA based authentication (CAA) [136]. Using CAA, an efficient technique of watermarking for image authentication is proposed in this chapter. Cryptanalysis of CAA have established the fact that it is more secure than MD5, SHA etc. and so we obtain watermarking schemes with higher security. Further high speed execution of CAA based watermarking scheme makes them ideally suitable for real time on-line applications.

There are several schemes of watermarking which are mostly based on spatial or transform domain. The proposed CAA based fragile watermarking scheme is based on spatial domain. Fragile watermarking scheme embed digital “watermarks” into images. It includes the methodologies used to insert and extract watermark in images. The outline of the scheme is pictorially sketched in figure 5.1 and 5.2.
5.3 Watermark Insertion

Let the grey-level host image be X. A bi-level watermark ‘A’ will be inserted in it and again will be extracted from it for authentication.

The bi-level watermark A is divided into (say) $K$ equal blocks ($A_i$) of arbitrary size $m \times m$ ($m$ is usually $\geq 16$ - to maintain security) and the grey-level host image $Y$ is also divided into same $K$ number of equal blocks ($Y_i$) whose block size is $n \times n$ ($n > m$). In the following discussion, we concentrate on inserting a block of A (say) - $A_i$ into the corresponding host image block $Y_i$. This process consists of the following three steps.

1. Derive Data

2. Embedding Watermark

3. Derive Image

In the Derive Data function, the portion of the host image where watermark will be inserted is selected. The Embedding Watermark function elaborates the exact methodology for inserting the watermark image on the selected portion of the host image. In the Derive Image function, the host image is reconstructed after watermark has been implanted in its selected portion. The steps are also pictorially explained through figure 5.1.

5.3.1 Derive Data

Since the size of watermark image block $A_i$ is taken as $n \times n$, therefore to insert this to host image block $Y_i$ (which is greater than $A_i$) properly, the size of $Y_i$ is reduced by selecting data from its middle region (this choice of region is made arbitrarily). In this step, thus the host data block ($X_i$) of size $m \times m$ is derived from the host image block $Y_i$ of size $n \times n$ by selecting data from $Y_i$’s middle region through Reduce function. Size of host data block ($X_i$) is made equal to the size of watermark block $A_i$ which is stored in the secret key $P$.

Following algorithm reports this step.

Algorithm 5.1 Derive Data

Input: Host image block ($Y_i$), The secret key ($P$)

Output: Host data block ($X_i$)

$X_i = \text{Reduce}(Y_i, P)$

An example is presented to illustrate the above algorithm.
Example 5.1 Let the Host Image Block \( Y_i = \begin{pmatrix} 162 & 161 & 160 & 158 \\ 161 & 201 & 212 & 218 \\ 129 & 128 & 129 & 127 \\ 133 & 132 & 133 & 133 \end{pmatrix} \) 
and the secret key \((\mathcal{P})\) is \([1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1]\), then

\[ X_i = \text{Reduce}(Y_i, \mathcal{P}) \text{ generates } X_i = \begin{pmatrix} 129 & 201 & 160 \\ 158 & 212 & 128 \\ 133 & 132 & 129 \end{pmatrix}. \]

5.3.2 Embedding Watermark

In this step, the watermark block \( A_i \) is embedded in \( X_i \). The overview of the steps are as follows. To ensure the extraction of watermark images, the least significant bits (LSBs) of \( X_i \) are converted to zero through the function \( \text{LSB's to zero} \) to produce \( X_{i}^{'} \). The block \( X_{i}^{'} \) is reinserted to the host image block \( Y_i \) to produce \( Y_i^{''} \). \( Y_i^{''} \) becomes the input of CA based one-way hash function \( \text{Hash\_To\_Generate\_Message\_Digest} \) along with the secret key. (\( \text{Hash\_To\_Generate\_Message\_Digest} \) is discussed in details at algorithm 4.2, page 51). \( Y_i^{''} \) is taken as input of hash function to ensure security. The hash output block from the function is the size of \( A_i \) and \( xor \) operation is performed between them. Next, the data of this \( xor \) block is put into the lbs of \( X_{i}^{'} \) by \( \text{LSB's to set} \) function and the desired watermark embedded data block is obtained. Following algorithm shows embedding.

Algorithm 5.2 Embedding Watermark

**Input:** Host Data Block \((X_i)\), Host Image Block \((Y_i)\), Watermark Image Block \((A_i)\) and \( \mathcal{P} \): the secret key

**Output:** Watermark Embedded Data Block \((X_i^{'})\).

**Step 1.** \( X_{i} = \text{LSB's to zero} (X_i) \)

**Step 2.** \( Y_i^{'} = \text{Insert\_Back\_Host\_Data\_Block}(Y_i, X_{i}) \)

**Step 3.** \( X_{h_i} = \text{Hash\_To\_Generate\_Message\_Digest}(Y_i^{'}, \mathcal{P}) \)

**Step 4.** \( X_{xor_i} = X_{h_i} \oplus A_i \)

**Step 5.** \( X_i^{'} = \text{LSB's to set} (X_{i}, X_{xor_i}) \)

The example to illustrate algorithm 5.2 is continuation of example 5.1.

Example 5.2 From example 5.1, the Host Data Block is \( X_i \) where

\[ X_i = \begin{pmatrix} 129 & 201 & 160 \\ 158 & 212 & 128 \\ 133 & 132 & 129 \end{pmatrix} = \begin{pmatrix} 10000001 & 11001001 & 10100000 \\ 10011110 & 11010100 & 10000000 \\ 10000101 & 10000100 & 10000001 \end{pmatrix} \text{ in binary} \]
\[ A_i = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \quad Y_i = \begin{pmatrix} 162 & 161 & 160 & 158 \\ 161 & 201 & 212 & 218 \\ 129 & 128 & 129 & 127 \\ 133 & 132 & 133 & 133 \end{pmatrix}, \quad \text{and} \quad P = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \]

**Step 1.** \( X_i = \) LSB’s to zero \((X_i)\) gives,

\[
X_i = \begin{pmatrix} 10000000 \\ 10011000 \\ 10000100 \end{pmatrix} = \begin{pmatrix} 128 \\ 158 \\ 132 \end{pmatrix} \]

**Step 2.** \( Y_i' = \) Insert\_Back\_Host\_Data\_Block\((Y_i, X_i)\)

\[
Y_i' = \begin{pmatrix} 162 & 161 & 160 & 158 \\ 161 & 200 & 212 & 218 \\ 128 & 128 & 128 & 127 \\ 132 & 132 & 133 & 133 \end{pmatrix} \]

**Step 3.** \( \text{Hash\_To\_Generate\_Message\_Digest}\((Y_i', P)\) produces \( X_{h_i} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \)

**Step 4.** \( X_{h_i} \oplus A_i \) produces \( X_{xor_i} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \).

**Step 5.** \( X_i' = \) LSB’s to set \((X_i, X_{xor_i})\) gives watermark embedded data block

\[
X_i' = \begin{pmatrix} 10000001 \\ 10011001 \\ 10000101 \end{pmatrix} = \begin{pmatrix} 129 \\ 159 \\ 132 \end{pmatrix} \]

5.3.3 **Derive Image**

In this step, the watermarked portion \((X_i')\) of the host image block \((Y_i)\) is inserted back into the host image \((Y_i)\) to produce the watermarked host image \(W_i\) through \text{Reinsert} function.

**Algorithm 5.3** Derive\_Image

**Input:** Watermark embedded data block \((X_i')\), Host image block \((Y_i)\)

**Output:** Watermarked host image block\((W_i)\)

\( W_i = \) Reinsert\((X_i', Y_i)\)

An example is provided to illustrate the steps of the algorithm. The example maintains the continuity from example 5.1 & 5.2.
Example 5.3 Given $X'_i = \begin{pmatrix} 129 & 200 & 160 \\ 159 & 212 & 129 \\ 132 & 133 & 128 \end{pmatrix}$ and $Y_i = \begin{pmatrix} 162 & 161 & 160 & 158 \\ 161 & 201 & 212 & 218 \\ 129 & 128 & 129 & 127 \\ 133 & 132 & 133 & 133 \end{pmatrix}$.

$W_i = \text{Reinsert}(X'_i, Y_i)$ produces

$W_i = \begin{pmatrix} 162 & 161 & 160 & 159 \\ 161 & 201 & 212 & 218 \\ 129 & 128 & 129 & 127 \\ 132 & 133 & 133 & 133 \end{pmatrix}$

Hence, if the host image block $Y_i = \begin{pmatrix} 162 & 161 & 16 & 158 \\ 161 & 201 & 212 & 218 \\ 129 & 128 & 129 & 127 \\ 133 & 132 & 133 & 133 \end{pmatrix}_{4 \times 4}$

and watermark image block $A_i = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}_{3 \times 3}$ then

watermark embedded image block will be $W_i = \begin{pmatrix} 162 & 161 & 160 & 159 \\ 161 & 200 & 212 & 218 \\ 129 & 129 & 128 & 127 \\ 132 & 133 & 133 & 133 \end{pmatrix}_{4 \times 4}$ (different pixel values for $Y_i$ and $W_i$ due to watermark embedding are shown in bold).

The extraction process is next discussed.

5.4 Watermark Extraction

Similar to insertion process, here in the extraction process the watermark embedded host image ($W$) is also divided into $K$ equal blocks ($W_i$) each of size $n \times n$. We discuss the steps of extracting watermark block from a single $W_i$. The watermark blocks from other host image blocks can be accordingly derived. The extraction procedure does not require original host image or watermark and thus becomes a blind method. It only requires the private key $P$. Extraction of the proposed scheme is stated here by the following two steps.

- **Derive Data**
- **Extracting Watermark**

Figure 5.2 depicts the context and flow of each step.

The **Derive Data** function is the same procedure followed during watermark insertion and already discussed in detail through example 5.1.

In **Derive Data** watermark embedded image block $W_i$ is preprocessed to produce watermark embedded data block $X'_i$. The number of equal host image blocks and size of watermark embedded data block $X'_i$ is obtained from the secret key $P$. The following example elaborates the steps.
There are two broad steps: Derive Data and Extracting Watermark.

Figure 5.2: Watermark extraction for fragile watermarking
Example 5.4 Let, \( W_i = \begin{pmatrix} 162 & 161 & 160 & 159 \\ 161 & 200 & 212 & 218 \\ 129 & 129 & 128 & 127 \\ 132 & 133 & 133 & 133 \end{pmatrix} \) be the watermark embedded image block obtained from previous example, then

**Step 1.** \( X'_i = \text{Reduce}(W_i, \mathcal{P}) \) generates \( X'_i = \begin{pmatrix} 129 & 200 & 160 \\ 159 & 212 & 129 \\ 132 & 133 & 128 \end{pmatrix} \)

5.4.1 Extracting Watermark

The function \( \text{Extracting Watermark} \) performs exactly the opposite steps of \( \text{Embedding Watermark} \) in page 70. In this step, the watermark image block \( A_i \) is extracted from embedded host data block \( (X'_i) \). Since the watermark is embedded in the least significant bits, the least significant bits are first extracted through \( \text{Extract_LSB} \) to obtain \( X_{xor_i} \) and the corresponding bits are set to zero to obtain the data block \( (X_i) \). This block is reinserted in the corresponding position of \( W_i \) to produce image block \( Y'_i \). The image block \( Y'_i \) then is input to the function \( \text{Hash_To_Generate_Message_Digest} \) to produce the hash output \( X_{h_i} \). Finally, we obtain the watermark block through \textit{xoring} of \( X_{xor_i} \) and \( X_{h_i} \).

**Algorithm 5.4 Extracting Watermark**

**Input:** Watermark embedded host data block \( (X'_i) \), Watermark embedded host image block \( (W_i) \) and \( \mathcal{P} \): the secret key

**Output:** Watermark image block \( (A_i) \)

**Step 1.** \( X_{xor_i} = \text{Extract_LSB}(X'_i) \)

**Step 2.** \( X_i = \text{LSB's to zero}(X'_i) \)

**Step 3.** \( Y'_i = \text{Insert_Back_Host_Data_Block}(W_i, X_h) \)

**Step 4.** \( X_{h_i} = \text{Hash_To_Generate_Message_Digest}(Y'_i; \mathcal{P}) \)

**Step 5.** \( A_i = X_{h_i} \oplus X_{xor_i} \)

The example is a continuation of example 5.4.

Example 5.5 From example 5.4, watermark embedded host data block \( X'_i \), watermark embedded host image block \( W_i \) and the secret key \( \mathcal{P} \) are

\[
X'_i = \begin{pmatrix} 129 & 200 & 160 \\ 159 & 212 & 129 \\ 132 & 133 & 128 \end{pmatrix} = \begin{pmatrix} 10000001 & 11001000 & 10100000 \\ 10011111 & 11010100 & 10000001 \\ 10000100 & 10000101 & 10000000 \end{pmatrix}
\]

\[
W_i = \begin{pmatrix} 162 & 161 & 160 & 159 \\ 161 & 200 & 212 & 218 \\ 129 & 129 & 128 & 127 \\ 132 & 133 & 133 & 133 \end{pmatrix}
\]

and \( \mathcal{P} = [1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1] \)

**Step 1.** \( X_{xor_i} = \text{Extract_LSB}(X'_i) \) where \( X_{xor_i} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \)
Step 2. $X_i = \text{LSB's to zero}(X'_i)$ produces

$$X_i = \begin{pmatrix}
1000000 & 1100100 & 1010000 \\
10011110 & 11010100 & 1000000 \\
1000100 & 1000100 & 1000000
\end{pmatrix} = \begin{pmatrix}
128 & 200 & 160 \\
158 & 212 & 128 \\
132 & 132 & 128
\end{pmatrix}.$$ 

Step 3. $Y'_i = \text{Insert}_{\text{Back}_{\text{Host}_{\text{Data}_{\text{Block}}}}}(W_i, X_i)$ produces $Y'_i = \begin{pmatrix}
162 & 161 & 160 & 158 \\
161 & 200 & 212 & 218 \\
128 & 128 & 128 & 127 \\
132 & 132 & 133 & 133
\end{pmatrix}$.

Step 4. $X_{hi} = \text{Hash}_{\text{To}_{\text{Generate}_{\text{Message}_{\text{Digest}}}}}(Y'_i, P)$ produces $X_{hi} = \begin{pmatrix}
0 & 0 & 1 \\
0 & 1 & 1 \\
0 & 0 & 1
\end{pmatrix}$.

Step 5. $A_i = X_{hi} \oplus X_{xor_i}$ produces $A_i = \begin{pmatrix}
0 & 0 & 1 \\
1 & 1 & 0 \\
0 & 1 & 1
\end{pmatrix}$.  \hfill \square

5.5 Experimental Results

Fragile watermarking are used for image authentication. Hence, in the proposed scheme, any modification of the image should be reflected in a corresponding error in the watermark. If the image is modified, or if the secret key is incorrect, or if the image was not at all watermarked, or if the watermarked image is cropped, the proposed watermark extraction algorithm should return an image that resembles random noise. Moreover since it requires a user key during both the insertion and the extraction procedures, it should not be possible for an unauthorized user to insert a new watermark or alter the existing watermark in such a way that the resulting image will pass the authentication test.

The two main properties which a good fragile watermarking scheme should reflect are (a) high quality watermark embedded host image and (b) secured watermarking against attack. Each point is discussed one by one. We have considered the Wong-Menon’s watermarking method [241] for comparison.

5.5.1 Measuring Image Quality

The invisibility of embedded watermark plays an important role in maintaining the quality of watermarked image. To measure that invisibility one largely used term is peak signal to noise ratio ($psnr$). $psnr$ shows a relationship with the perceived errors noticed by the human visual system.

$psnr$ estimates measurement of the quality of a reconstructed image compared with an original image. The basic idea is to compute a single number that reflects the quality of the reconstructed image. Assume that given a host image is $f(i,j)$ that contains $N \times N$ pixels and a watermark embedded host image is $F(i,j)$. $psnr$
Table 5.1: Comparative psnr at Fragile Watermarking

<table>
<thead>
<tr>
<th>Image Name</th>
<th>Image Size</th>
<th>psnr Values in dB unit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Proposed</td>
</tr>
<tr>
<td>Concord</td>
<td>732 × 500</td>
<td>67.47</td>
</tr>
<tr>
<td>Barbara</td>
<td>512 × 512</td>
<td>66.17</td>
</tr>
<tr>
<td>Lena</td>
<td>512 × 512</td>
<td>66.07</td>
</tr>
<tr>
<td>Goldhill</td>
<td>512 × 512</td>
<td>66.06</td>
</tr>
<tr>
<td>Sachin</td>
<td>322 × 400</td>
<td>63.08</td>
</tr>
<tr>
<td>Rabbit</td>
<td>400 × 316</td>
<td>62.80</td>
</tr>
<tr>
<td>Baboon</td>
<td>200 × 200</td>
<td>57.78</td>
</tr>
</tbody>
</table>

in decibels (dB) is computed by using

\[
psnr = 20 \log_{10} \left( \frac{255}{RMSE} \right) \tag{5.1}
\]

Error metrics are computed on the luminance signal only so the pixel values \( f(i, j) \) range between black (0) and white (255) (for grey scale). The mean squared error (MSE) of the reconstructed image is computed as

\[
MSE = \frac{\sum_i \sum_j [F(i, j) - f(i, j)]^2}{N^2} \tag{5.2}
\]

The summation is over all pixels. The root mean squared error (RMSE) is the square root of MSE.

The table 5.1 shows the result where in each of the image, watermark inserted according to proposed fragile scheme (column 3) gives very much better psnr values than Wong-Menon’s method [241] (column 4). Moreover as size of image file increases in the proposed scheme, psnr value improves (column 2 & 3 of table 5.1).

This is because in Wong-Menon’s method the actual watermark image is made of size equal to the host image and so embeds much more data into host image which degrades image quality. While proposed watermarking embeds the original watermark data into the host image but ensure high security by using CAA resulting in superior quality watermarked image.

Figure 5.3 shows a sample pair of images without and with watermark embedded in them respectively. It can be seen that the difference between them is not easily visible.

5.5.2 Security of Proposed Scheme

The proposed scheme derives its strength from the SACA based hash function it uses. Cryptanalysis of SACA based one-way hash function at previous chapter has
already displayed the related security issues with the proposed fragile watermarking scheme.

Like any cryptographic system, the security of the proposed scheme resides in the secrecy of the user secret key and not in the obscurity of the algorithm. In fact, the watermark insertion and extraction steps can be made public without compromising the security of the watermark.

In the proposed scheme since we use watermark image block of size greater than equal to 16 × 16 along with hash function CAA of 256 bit hash output so, the desired security is maintained as that of Wong-Menon’s method [241]. The secret key scheme uses the cryptographic hash function CAA. Its security relies on the computational in-feasibility to break the cryptographic hash function as depicted in previous chapter.

A digital watermarking algorithm tries to adhere some copyright information to the original data. The recent attacks on watermarking and the corresponding defenses by the proposed scheme is discussed next.

**Reverse Engineering** : This means when an unauthorized person will try to extract the watermark from the embedded host image by reverse engineering, it should be a impossible task for him [45]. In the proposed scheme since we use keyed one-way hash function so without knowing the secret key none can extract the watermark. A thorough cryptanalysis for the CA based keyed one-way function used in the proposed watermarking scheme, is already reported in the previous chapter.

Moreover the greatest advantage of using the one-way hash function is the flexibility of adjusting key size without any overhead. This is possible due to modular structure of cellular automata. This enhances security over other one-way hash function such as MD5.

**Holliman-Menon Attack** : The most effective attack on image authentication is Holliman-Menon attack or Vector Quantization attack [83]. CAA based watermarking is tuned to counterfeit this attack as a built-in function whereas all other hash
functions (including MD5) defend the attack externally which effectively decreases the insertion/extraction speed of watermarking. In CAA an index parameter has been considered which makes the vector quantization attack practically infeasible [241].

**Index parameter**: When the host image is divided in blocks and the watermark is inserted in each block then in case of a few watermarked images whose host image remains same but the watermarks are different - hackers gets some special advantage, he can divide them all in blocks and replace one block of one watermarked image with another block of other watermarked image thus the host image remains same but the watermark changes.

This can be prevented by imposing index on each block of a watermarked image - other schemes like MD5 does this externally while in our case this is incorporated with the hash function because hashing of a block can be performed along with its index parameter. Index parameter can be represented by some number such as 11 which implies 1st block of watermarked number image numbered 1 etc.. It has unique key value for a specific block of a specific watermarked image. The scheme is discussed in detail in the renowned paper [83].

### 5.6 Conclusion

Employing GF($2^n$) CA based keyed one-way hash function, a fragile watermarking scheme which can take part in image authentication has been described in this chapter. Experimental results confirm the improvement of this fragile watermarking scheme over Wong-Menon’s method [241] both in terms of image quality and security. Fragile watermarking is most trusted method for image authentication [65, 67]. But the fragility prevents its practical aspect as the image cannot incur any intended data alteration such as lossy compression. But in today’s scenario the growing need of watermarking has demanded the necessity of such a watermarking which can withstand intended data manipulation at least for lossy image compression. Semi-fragile watermarking is used to serve this purpose. In the next chapter we present two such CAA based watermarking schemes (employing GF($2^n$) CA based keyed one-way hash function) which treads exactly this path that is, these watermarking schemes are robust against lossy compression.