

## **Chapter 5**

# **Reversible Watermarking in Authentication, Tamper Detection and Localization**



In this chapter, two RWS schemes, RWS-LBP-CA and RWS-LBP-WM-LIP, have been developed for authentication, tamper detection, and tamper localization. In RWS-LBP-CA, an RWS for image authentication, tamper detection and localization using CA and LBP operator has been designed. In the embedding phase, authentication bits (AC) are generated from watermark with the help of a shared secret key ( $\mu$ ). The tamper detection bits (TDC) are formed by employing LBP operator and CA from the coloured cover image. Both these information (TDC and AC) are embedded within the interpolated color image. Moreover, watermark bits are not inserted directly in the cover image. Rather, watermark bits are encrypted using  $\mu$  and LBP operator before embedding. In the detection phase, authentication bits are extracted using  $\mu$  and authenticity is verified. Also, the TDC bits are extracted from the watermarked image with the help of CA and LBP operator to check the tampered region. The proposed schemes are secured, robust and provide excellent visual characteristics to the watermarked image. In RWS-LBP-WM-LIP, an RWS have been proposed using LBP, WM, and Lagrange Interpolation Polynomial (LIP). In both the schemes, an initiative has been taken to find the robustness of the schemes against some standard attacking environment.

## 5.1 RWS-LBP-CA: RWS based on CA and LBP <sup>6</sup>

Two-dimensional cellular automata (CA) is employed in image watermarking due to its simplicity and low computational cost for hardware implementation. LBP has been widely utilized for feature extraction, texture analysis, pattern matching, visual inspection and image recovery in multimedia documents. In RWS-LBP-CA, both operators, LBP and CA, are employed for image authentication and tamper detection through a watermarking scheme which is highly desirable in the various human-centric application such as health-care, military communication, e-governance, remote sensing, and law enforcement. Here, CA is used to ensure data confidentiality and LBP is used to locate the tampered region. Different CA rules are used to increase the security level of the developed scheme. The authentication and tamper detection have been achieved by employing AC and TDC which are generated by employing SHA-512 and LBP operator respectively. The experimental results are compared with some state-of-the-art meth-

---

<sup>6</sup>Submitted in **International Journal of Computers and Applications**, Taylor & Francis: ISSN: 1206-212X, **Scopus (Elsevier)** with title *Cellular Automata Based Reversible Watermarking Scheme for Image Authentication and Tamper Detection using LBP*

ods to show the efficiency of RWS-LBP-CA. To evaluate imperceptibility and robustness, some standard NIST recommended steganalysis and attacks are conducted. It has been observed that the RWS-LBP-CA is secured and robust contrary to various geometric attacks meanwhile it can detect tamper and localize tampered region in the watermarked image.

The overall schematic diagram of RWS-LBP-CA has been outlined in Fig. 5.1. The RWS-LBP-CA has been divided into three phases, (i) watermark embedding process, (ii) watermark extraction process and (iii) authentication process. In the embedding phase, AC and TDC are embedded within the cover image with the help of CA and LBP shown in Fig. 5.1(a). In extraction phase, watermark has been extracted successfully shown in Fig. 5.1(b). Finally, in authentication phase, extracted AC (EAC) and regenerated AC (RAC) has been compared for authentication and regenerated TDC (RTDC) and extracted TDC (ETDC) is compared for tamper detection and localization. The detail embedding and extraction procedure of RWS-LBP-CA has been described in section 5.1.1 and section 5.1.2

### 5.1.1 Watermark Embedding Phase

$$\left\{ \begin{array}{l}
 ICI(m, n) = CI(p, q) \\
 \quad \{ \text{where } p = 1 \dots M, q = 1 \dots N, \\
 \quad m = 1, 3, \dots (2 \times M - 1), n = 1, 3, \dots (2 \times N - 1) \} \\
 ICI(m, n) = (CI(m, n - 1) + CI(m, n + 1))/2 \\
 \quad \{ \text{where } (m \bmod 2) \neq 0, (n \bmod 2) = 0 \} \\
 ICI(m, n) = (CI(m - 1, n) + CI(m + 1, n))/2 \\
 \quad \{ \text{where } (m \bmod 2) = 0, (n \bmod 2) \neq 0 \\
 \quad m = 1 \dots (2 \times M - 1), n = 1 \dots (2 \times N - 1) \} \\
 ICI(m, n) = (CI(m - 1, n - 1) + CI(m - 1, n + 1) + CI(m + 1, n - 1) \\
 \quad + CI(m + 1, n + 1))/4 \\
 \quad \{ \text{where } (m \bmod 2) = 0, (n \bmod 2) = 0 \}
 \end{array} \right. \quad (5.1)$$

In RWS-LBP-CA, a CA-based RWS has been proposed through LBP is shown in Fig. 5.2. At first, a color image (CI) is considered as a cover image, then separated into three R, G, B channels. After that CI is partitioned into  $(3 \times 3)$  image blocks. Then,  $(5 \times 5)$  interpolated image block (ICI) is created from  $(3 \times 3)$  image blocks using simple image interpolation method illustrate in equation (5.1).

Now, a shared secret key  $\mu$  and a watermark image (W) are considered. 512-bit authentication code (AC) is generated from W using the cryptographic hash key generation algorithm SHA-

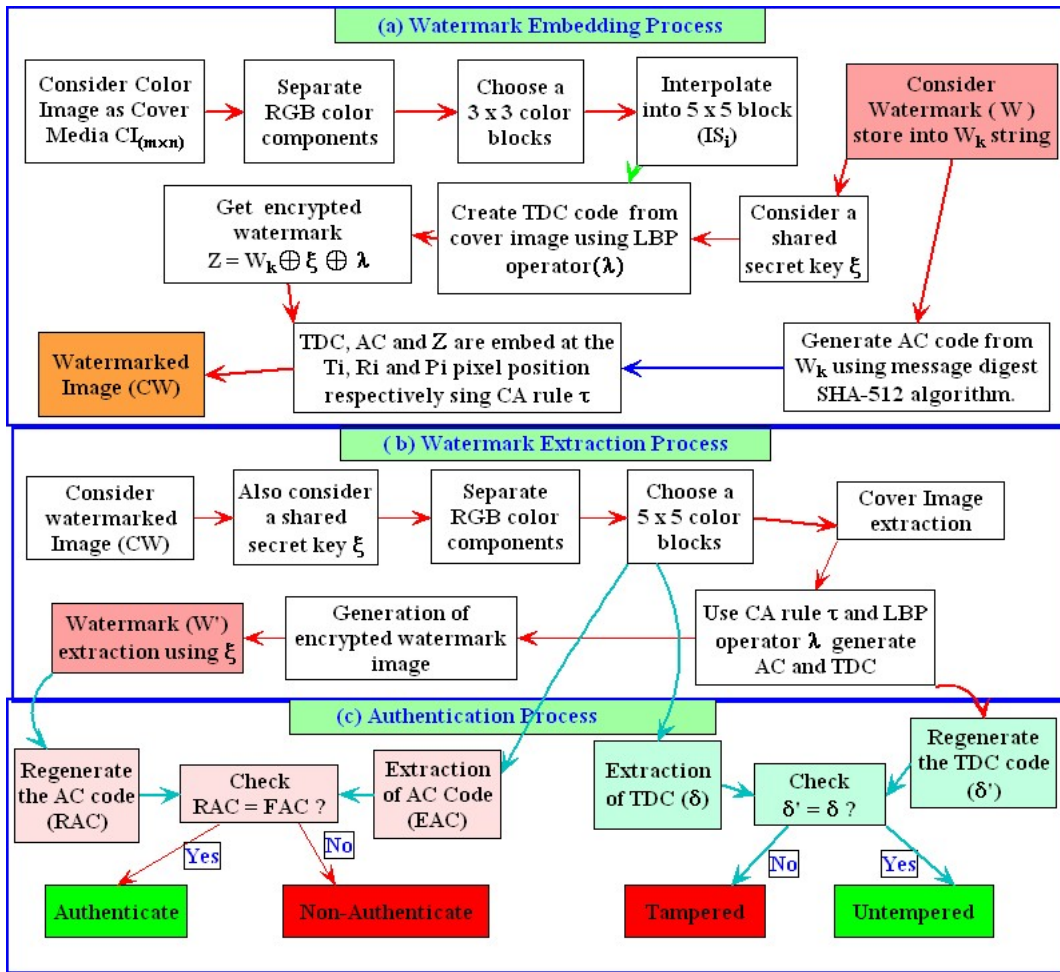


Figure 5.1: Block diagram of watermarking process in RWS-LBP-CA.

512. Now, 8-bit (2 bit/pixel) AC is considered and embedded in the  $R_i$  position (green color region in Fig.5.2) of interpolated image block (ICI). After that 8-bit vector  $\lambda$  is generated from the cover image CI using LBP. Also, CA rule  $\kappa$  is considered, and 8-bit vector  $\tau$  is generated. After that, an XOR operation is performed among  $\tau$ ,  $\lambda$ , and  $\mu$  to generate TDC. This 8-bit (1 bit/pixel) TDC are embedded into the  $T_i$  position (pink color region in Fig.5.2) of ICI. An XOR operation is performed among  $\lambda$ ,  $\mu$  and  $W$  to generate encrypted watermark ( $Z$ ). This 8-bit (2 bit/pixel)  $Z$  is embedded into the LSB of  $P_i$  position (blue color region in Fig.5.2) of ICI. In this way, all the bits of TDC, AC, and  $Z$  are embedded in the corresponding position of the ICI. Then same operations are performed for embedding the whole watermark image into the remaining blocks of ICI, and the watermarked image (WI) is generated. The corresponding algorithm is shown in Algorithm 5.1. In this algorithm, the data along with AC and TDC are embedded in each pixel block.

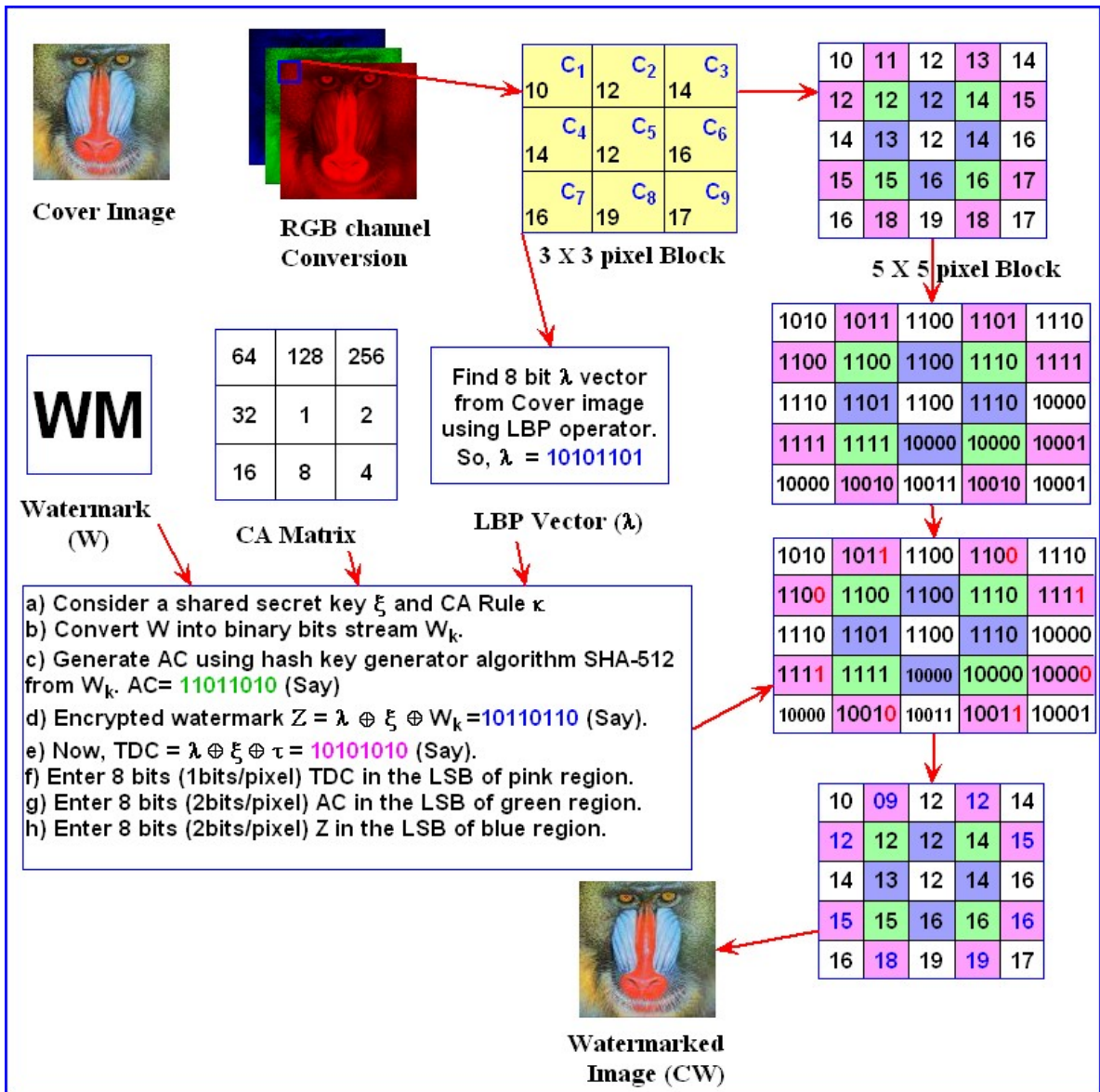


Figure 5.2: Numerical illustration of watermark embedding phase in RWS-LBP-CA.

### 5.1.2 Watermark Extraction and Recovery Phase

The detail extraction procedure of RWS-LBP-CA has been depicted in Fig. 5.3. At first, WI is considered and separated into RGB color components. Then, any one color component is chosen and divided into  $(5 \times 5)$  pixel blocks. The original color image ( $CT'$ ) is constructed from the unaffected pixels  $C_i$  (for  $i = 1$  to 9) from each  $(5 \times 5)$  pixel blocks of WI. In this way the cover image is constructed from three color components. Now, from the first block of WI, LSB-1 are collected from the pixels  $T_i$  for  $i = 1$  to 8 to form 8-bit binary number and stored into ETDC. Then 2 LSB are collected from the pixels  $R_i$  for  $i = 1$  to 4 to form 8-bit binary number and stored into EAC as extracted authentication code. Furthermore, 2 LSB are collected from

**Algorithm 5.1: RWS-LBP-CA: Watermark Embedding Algorithm**


---

**Input** : Cover Image (CI) and watermark (W)  
**Output**: Watermarked Images of size (IWI)

**1 Algorithm Embedding() :**

```

2 // Create interpolated array from cover image
3 interpolatedArray=toInterpolatedArray(coverImage);
4 // Form secret bit string extracting data from secret image
5 secretBits = ReadImage(secretImage);
6 // variable for LBP, watermark, secret key, authentication code etc.
7 var LBP, watermark, secretKey, ACCode, lbp1, lbp2, lbp3, lbp4, lbp5, lbp6, lbp7, lbp8 ;
8 // Process 5x5 pixel blocks for RGB color blocks
9 for ( y=0; y<imageHeight; y+=5) do
10     for ( x=0; x<imageWidth; x+=5) do
11         for ( color=0; color<=2; color++) do
12             // Extract these pixel values from interpolated array
13             lbp1=interpolatedArray[x+0][y+0][color];
14             lbp2=interpolatedArray[x+2][y+0][color];
15             lbp3=interpolatedArray[x+4][y+0][color];
16             lbp4=interpolatedArray[x+0][y+2][color];
17             lbp5=interpolatedArray[x+4][y+2][color];
18             lbp6=interpolatedArray[x+0][y+4][color];
19             lbp7=interpolatedArray[x+2][y+4][color];
20             lbp8=interpolatedArray[x+4][y+4][color];
21             // XOR 8 pixels of a 5x5 pixel block
22             LBP = lbp1  $\oplus$  lbp2  $\oplus$  lbp3  $\oplus$  lbp4  $\oplus$  lbp5  $\oplus$  lbp6  $\oplus$  lbp7  $\oplus$  lbp8;
23             // Get watermark value from 8 bit secret data
24             watermark = Bin2Dec(get8Bits(secretBits));
25             // Get wDash
26             wDash = LBP  $\oplus$  watermark  $\oplus$  secretKey;
27             // Get Tamper detection code
28             TDC = LBP  $\oplus$  CA  $\oplus$  secretKey;
29             // Call this method for embedding with appropriate arguments
30             embedDataInPixelBlock(interpolatedArray, color, x, y, wDash, ACCode, TDC);
31         end
32     end
33 end
34 // Create watermarked image from embedded interpolated array
35 CreateWatermarkedImage(interpolatedArray);

```

**25 Function embedDataInPixelBlock (interpolatedArray, color, x, y, wDash, ACCode, TDC) :**

```

36 strWDash = BinaryTo8BitString(wDash, 8);
37 strACCode = BinaryTo8BitString(ACCode, 8);
38 strTDC = BinaryTo8BitString(TDC, 8);
39 // Embed 1 bit TDC in LSB of each of the pixels in the pixel block
40 EmbedInLSB(pixelArray[x+1][y+0][color], Bin2Dec( strTDC.substring(0,1)));
41 EmbedInLSB(pixelArray[x+3][y+0][color], Bin2Dec( strTDC.substring(1,2)));
42 EmbedInLSB(pixelArray[x+4][y+1][color], Bin2Dec( strTDC.substring(2,3)));
43 EmbedInLSB(pixelArray[x+4][y+3][color], Bin2Dec( strTDC.substring(3,4)));
44 EmbedInLSB(pixelArray[x+3][y+4][color], Bin2Dec( strTDC.substring(4,5)));
45 EmbedInLSB(pixelArray[x+1][y+4][color], Bin2Dec( strTDC.substring(5,6)));
46 EmbedInLSB(pixelArray[x+0][y+3][color], Bin2Dec( strTDC.substring(6,7)));
47 EmbedInLSB(pixelArray[x+0][y+1][color], Bin2Dec( strTDC.substring(7,8)));
48 // Embed 2 bits watermark in LSB of each of the pixels in the pixel block
49 EmbedInLSB(pixelArray[x+1][y+1][color], Bin2Dec( strWDash.substring(0,2)));
50 EmbedInLSB(pixelArray[x+3][y+1][color], Bin2Dec( strWDash.substring(2,4)));
51 EmbedInLSB(pixelArray[x+3][y+3][color], Bin2Dec( strWDash.substring(4,6)));
52 EmbedInLSB(pixelArray[x+1][y+3][color], Bin2Dec( strWDash.substring(6,8)));
53 // Embed 2 bit Authentication Code in LSB of each of the pixels in the pixel block
54 EmbedInLSB(pixelArray[x+2][y+1][color], Bin2Dec( strACCode.substring(0,2)));
55 EmbedInLSB(pixelArray[x+3][y+2][color], Bin2Dec( strACCode.substring(2,4)));
56 EmbedInLSB(pixelArray[x+2][y+3][color], Bin2Dec( strACCode.substring(4,6)));
57 EmbedInLSB(pixelArray[x+1][y+2][color], Bin2Dec( strACCode.substring(6,8)));

```

---

the pixels  $P_i$  for  $i = 1$  to 4 to form 8-bit binary number and stored into  $Z'$  which is actually the extracted watermark bits in encrypted form. Now, a  $\lambda'$  vector is formulated from constructed  $CI'$  using LBP operator to generate original watermark. Then, original watermark bit stream ( $M'$ ) is extracted by performing an XOR operation with  $\lambda'$ ,  $\mu'$  and  $Z'$ . Now, from this  $M'$  bits stream,  $W'$  can easily be generated. Moreover, with the help of  $\mu$  and CA rule  $\kappa$ , generated AC code (GAC) and generated tamper detection code (GTDC) can be formed from the  $M'$  and  $CI'$  respectively to serve authentication and tamper detection.

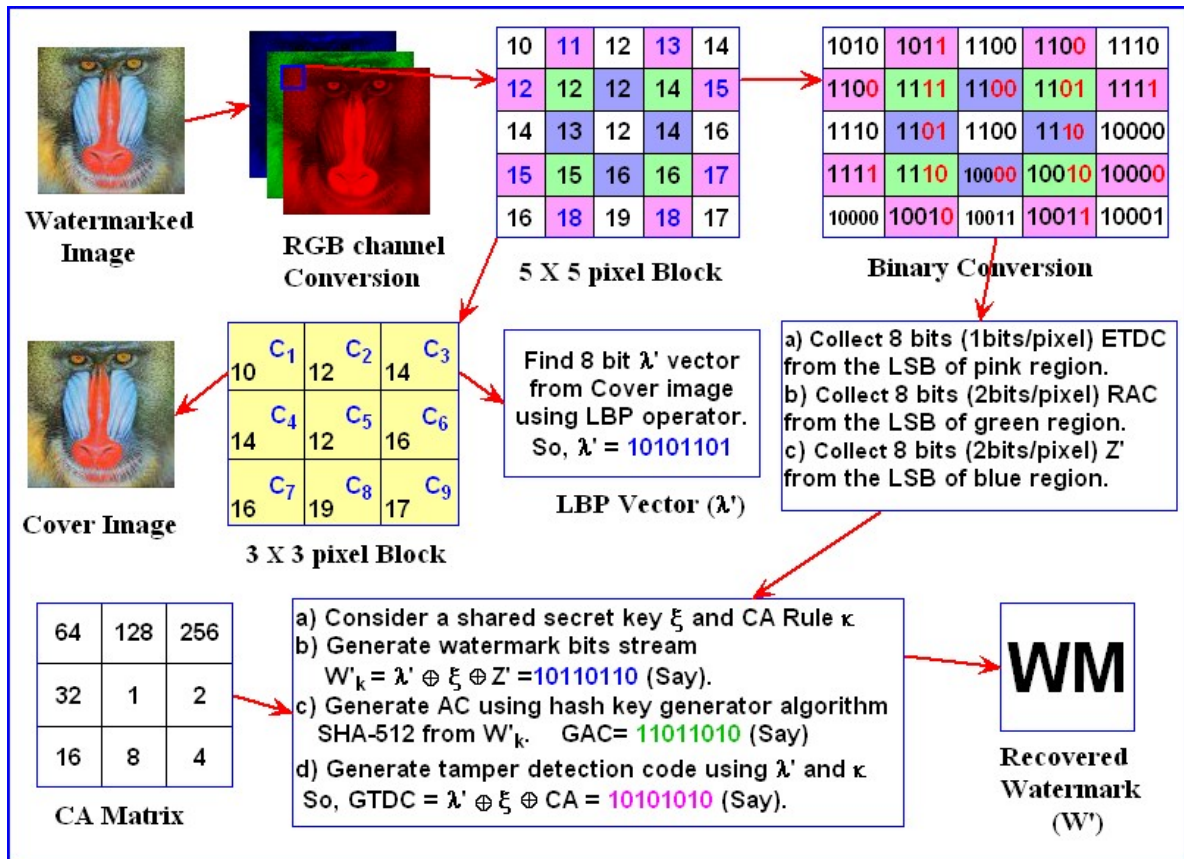


Figure 5.3: Numerical illustration of watermark extraction phase in RWS-LBP-CA.

### 5.1.3 Experimental Results and Comparison

A set of benchmark colour images of size  $(512 \times 512)$  are considered from [89], [61], [90], [26] shown in 2.3 to evaluate the efficiency of RWS-LBP-CA. Three different size of logo images have been considered as a watermark as shown in Fig. 5.4 to measure the quality and corresponding embedding capacity. Performances of the related schemes are compared to



**Algorithm 5.2: RWS-LBP-CA: Watermark Extraction Algorithm**


---

**Input** : Watermarked Images of size (IWI)  
**Output**: Cover Image (CI) and watermark (W)

```

1  Algorithm Extraction():
   // Declare variables
2  var coverX = 0, coverY = 0, tamperDetected, watermark, wDash, secretKey, var extractedBits, extractedTDC;
   // Read watermarked image
3  stegoImageArray=ReadImage(stegoFile);
   // Apply this loop for all 5x5 pixel blocks and for RGB components
4  for (int y=0;y<imageHeight;y+=5) do
5      coverX=0;
6      for (int x=0;x<imageWidth;x+=5) do
   // initially, set this as false every time
7      tamperDetected=false;
8      for (int color=0;color<=2;color++) do
   // Extract data from the embedded pixels
9      wDash=extractDataFromPixelBlock(stegoImageArray, color, x, y);
   // Get these pixel values
10     lbp1=stegoImageArray[x+0][y+0][color];  lbp2=stegoImageArray[x+2][y+0][color];  lbp3=stegoImageArray[x+4][y+0][color];
11     lbp4=stegoImageArray[x+0][y+2][color];  lbp5=stegoImageArray[x+4][y+2][color];  lbp6=stegoImageArray[x+0][y+4][color];
12     lbp7=stegoImageArray[x+2][y+4][color];  lbp8=stegoImageArray[x+4][y+4][color];
   // XOR these values to get LBP value
13     LBP = lbp1  $\oplus$  lbp2  $\oplus$  lbp3  $\oplus$  lbp4  $\oplus$  lbp5  $\oplus$  lbp6  $\oplus$  lbp7  $\oplus$  lbp8;
   // Get the watermark value in decimal
14     watermark = Bin2Dec(wDash)  $\oplus$  LBP  $\oplus$  secretKey;
   // Add watermark value to secret data string
15     extractedBits.append(to8BitBinaryString(watermark));
   // Extract cover image 3x3 pixel blocks from stego image
16     coverArray[coverX+0][coverY+0][color]= stegoArray[x+0][y+0][color];
17     coverArray[coverX+1][coverY+0][color]= stegoArray[x+2][y+0][color];
18     coverArray[coverX+2][coverY+0][color]= stegoArray[x+4][y+0][color];
19     coverArray[coverX+0][coverY+1][color]= stegoArray[x+0][y+2][color];
20     coverArray[coverX+1][coverY+1][color]= stegoArray[x+2][y+2][color];
21     coverArray[coverX+2][coverY+1][color]= stegoArray[x+4][y+2][color];
22     coverArray[coverX+0][coverY+2][color]= stegoArray[x+0][y+4][color];
23     coverArray[coverX+1][coverY+2][color]= stegoArray[x+2][y+4][color];
24     coverArray[coverX+2][coverY+2][color]= stegoArray[x+4][y+4][color];
   // Extract TDC from the following stego image pixels
25     extractedTDC+=(stegoArray[x+1][y+0][color] & 1);  extractedTDC+=(stegoArray[x+3][y+0][color] & 1);
26     extractedTDC+=(stegoArray[x+4][y+1][color] & 1);  extractedTDC+=(stegoArray[x+4][y+3][color] & 1);
27     extractedTDC+=(stegoArray[x+3][y+4][color] & 1);  extractedTDC+=(stegoArray[x+1][y+4][color] & 1);
28     extractedTDC+=(stegoArray[x+0][y+3][color] & 1);  extractedTDC+=(stegoArray[x+0][y+1][color] & 1);
29     CA=244; // Apply Cellular Automata Rule 90
30     TDCExtracted=Bin2Dec(extractedTDC); // Extract TDC
31     TDCEmbedded=LBP  $\oplus$  CA  $\oplus$  secretKey; // Calculate original TDC
   // Check whether original TDC and extracted TDC is same. If not, a tamper is reported and the
   // block is marked.
32     if ( TDCExtracted <> TDCEmbedded) then tamperDetected=true;
33     end
34     coverX+=3;
35     end
36     coverY+=3;
37 end
38 CreateWatermark(sbExtractedBits);
39 CreateCoverImage(coverArray);
40 Function extractDataFromPixelBlock (stegoArray, color, x, y):
41     var extractedData;
   // Extract 2 LSB bits from the pixels and append bits to extractedData
42     extractedData.append( GetLSB2(stegoArray[x+1][y+1][color] ));  extractedData.append( GetLSB2(stegoArray[x+3][y+1][color] ));
43     extractedData.append( GetLSB2(stegoArray[x+3][y+3][color] ));  extractedData.append( GetLSB2(stegoArray[x+1][y+3][color] ));
   // Return extracted data
44     return extractedData;

```

---

test the effectiveness of RWS-LBP-CA. MSE [35], PSNR [35], SSIM [78] and Q-Index are computed using the equation (2.5), (2.6), (2.8) and (2.12) respectively to test the perceptible characteristics after embedding. Also NCC [94], BER [65], SD ( $\sigma$ ) [35] and CC ( $\rho$ ) [35] are computed using the equation (2.11), (2.13), (2.9) and (2.10) respectively to test robustness of watermarked image. Performance of RWS-LBP-CA is assessed with respect to computation time and it is compared with other existing schemes.

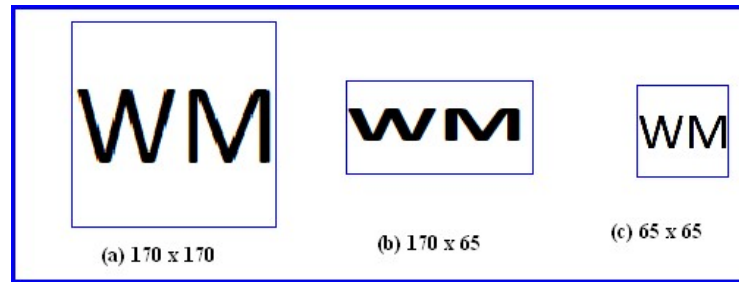


Figure 5.4: Watermark images (logo) with different size used in RWS-LBP-CA

### 5.1.3.1 Quality Measurement and Payload Analysis

The fundamental necessities of any watermarking schemes are robustness and imperceptibility. Usually, the quality of watermarked images are evaluated from their subjective and objective quality indices. The subjective characteristics of the watermarked images is evaluated in RWS-LBP-CA and it has been shown in Fig. 5.5. The evaluation results of the RWS-LBP-CA in terms MSE, PSNR, NCC, SSIM, Q-Index, BER and Payload after embedding watermark are presented in Table 5.1. From Table 5.1, it is found that the average PSNR for the aforesaid image databases are greater than 48.26 dB. The NCC, SSIM and Q-Index values of RWS-LBP-CA are nearer to one, which prove the effectiveness of RWS-LBP-CA. The BER results prove that the developed scheme RWS-LBP-CA is robust. It is observed from Fig. 5.5 that no visual distortions are detected after embedding maximum watermark bits of 7, 01, 784 bits.

The RWS-LBP-CA has been tested taking sample images from four different standard benchmark image databases [89], [61], [90], [26] and experimental outcomes are illustrated in Table 5.2. Table 5.2 presents that after embedding a highest amount of 7, 01, 784 bits watermark, approximately 48.23 dB PSNR can be achieved on average. Q-Index values are also close to unity which establishes the acceptability of RWS-LBP-CA.

The resemblance on different reversible watermarking scheme with respect to PSNR and ca-





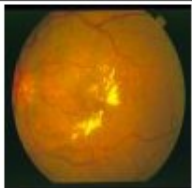
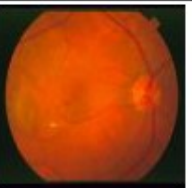
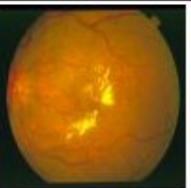
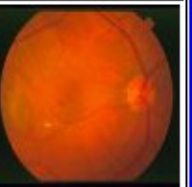








Image Database	Original Cover Image		Watermarked Image	
USC-SIPI (513 x 513)				
	1. Lena	2. Baboon	1. Lena	2. Baboon
STARE (513 x 513)				
	3. im0001	4. im0376	3. im0001	4. im0376
UCID (513 x 513)				
	5. ucid00104	6. ucid00804	5. ucid00104	6. ucid00804
HDR (513 x 513)				
	7. jerusalem	8. tahoue	7. jerusalem	8. tahoue

Figure 5.5: Pictorial results of output images in RWS-LBP-CA

capacity for Lena, Airplane, Baboon, Pepper, Boat and Tiffany images are depicted in Table 5.3. The graphical representation are shown in Fig. 5.19. From the Table 5.3 and Fig. 5.19, it is clear that RWS-LBP-CA facilitates better results in terms capacity and quality compared with other existing schemes. From experimental results, it is noticed that RWS-LBP-CA attains high payload (2.66 bpp) with good visual quality (48.66 dB PSNR) which is very important for medical, e-governance and military applications.

Additionally, the results of objective analysis have been depicted in Table 5.4 for color images (without any invasion). Table 5.4 represents the diversity of quality in terms of the number of images used from the different image database. From the experimental result it is clear that RWS-LBP-CA gives average 48 dB PSNR for all the image database.

The experimental results of RWS-LBP-CA based on visual quality on all 1338 images of UCID image database [61] are shown in Fig. 5.7, after embedding 1, 75, 440 bits, 3, 48, 840 bits and

Table 5.1: Results of MSE, PSNR, NCC, SSIM, Q-Index and BER for four different benchmark datasets in RWS-LBP-CA

Database	Image	Capacity (bits)	MSE	PSNR (dB)	BER	NCC	Q-Index	SSIM
SIPI	Lenna	7,01,784	0.9111	48.53	0.01996	0.99998	0.99985	99.36%
	Baboon	7,01,784	0.8986	48.60	0.01997	0.99998	0.99986	98.56%
	Tiffany	7,01,784	0.9702	48.26	0.02024	0.99999	0.99957	99.12%
	Barbara	7,01,784	0.9184	48.50	0.02000	0.99997	0.99988	99.23%
	Airplane	7,01,784	0.9246	48.47	0.02000	0.99999	0.99983	98.43%
	Zelda	7,01,784	0.93	48.45	0.02000	0.99998	0.99987	98.66%
	Average	7,01,784	0.9255	48.46	0.02003	0.99998	0.99981	98.89%
HDR	anhinga	7,01,784	0.9541	48.33	0.01953	0.99996	0.99986	98.57%
	beeflowr	7,01,784	1.0369	47.97	0.01928	0.99996	0.99981	99.12%
	jerusalem	7,01,784	0.9298	48.45	0.01966	0.99995	0.99974	99.36%
	redrock2	7,01,784	0.911	48.54	0.02011	0.99997	0.99986	99.15%
	toucan	7,01,784	0.9546	48.33	0.01934	0.99996	0.99992	98.47%
	Average	7,01,784	0.9573	48.33	0.01958	0.99996	0.99984	98.96%
STARE	im0001	7,01,784	0.9653	48.28	0.01975	0.99995	0.99975	99.46%
	im0373	7,01,784	0.9261	48.46	0.01995	0.99997	0.99991	98.94%
	im0376	7,01,784	0.959	48.31	0.01976	0.99996	0.99973	98.87%
	im0386	7,01,784	0.9354	48.42	0.01999	0.99996	0.99988	98.79%
	Average	7,01,784	0.9465	48.37	0.01987	0.99996	0.99982	98.33%
UCID	ucid00104	7,01,784	0.9144	48.52	0.01998	0.99998	0.99992	98.46%
	ucid00157	7,01,784	0.9123	48.53	0.01997	0.99997	0.99999	99.21%
	ucid00520	7,01,784	0.9017	48.58	0.02001	0.99997	0.99997	98.72%
	ucid00576	7,01,784	0.9197	48.49	0.01998	0.99997	0.99994	98.67%
	ucid00797	7,01,784	0.9791	48.22	0.02050	0.99998	0.99995	98.37%
	Average	7,01,784	0.9254	48.47	0.02008	0.99997	0.99992	98.36%

7,01,784 bits watermark. From the graph it is clear that maximum quality can be achieved when a minimum number of watermark bits are embedded into the cover image.

Table 5.5 represents comparison results of RWS-LBP-CA with respect to the other existing schemes. From the table it is also clear that RWS-LBP-CA gives average 10% better result than Zhang et al. scheme [107] in terms of quality. Moreover, the graphical representation are shown in Fig. 5.8.

Table 5.2: Capacity, PSNR, Q-Index, and Bpp results for standard benchmark images in RWS-LBP-CA

Database	Image	Capacity (bits)	PSNR (dB)	Q-Index	Payload (bpp)
USC-SIPI [90]	Lena	1,75,440	48.12	0.99983	0.66
		3,48,840	48.25	0.99989	1.32
		7,01,784	48.53	0.99997	2.66
UCID [61]	ucid00104	1,75,440	48.10	0.99994	0.66
		3,48,840	48.24	0.99997	1.32
		7,01,784	48.52	0.99998	2.66
STARE [89]	im0373	1,75,440	48.21	0.99997	0.66
		3,48,840	48.34	0.99991	1.32
		7,01,784	48.46	0.99997	2.66
HDR [26]	anhinga	1,75,440	48.15	0.99989	0.66
		3,48,840	48.23	0.99992	1.32
		7,01,784	48.33	0.99996	2.66

Table 5.3: Comparison of different RWT in sub-sample image with respect to PSNR and Payload in RWS-LBP-CA

Image	Chang et al. [11]		Parah et al. [65]		Shin & Jung [75]		Lin & Chang [53]		RWS-LBP-CA	
	PSNR (dB)	Payload (bpp)	PSNR (dB)	Payload (bpp)	PSNR (dB)	Payload (bpp)	PSNR (dB)	Payload (bpp)	PSNR (dB)	Payload (bpp)
Lena	40.92	(t-1)/3	40.58	0.046	45.13	(t-2)/4	46.95	0.5	48.53	2.66
Baboon	40.92	(t-1)/3	39.60	0.046	43.9	(t-2)/4	46.92	0.5	48.59	2.66
Airplane	40.87	(t-1)/3	41.18	0.046	45.48	(t-2)/4	46.90	0.5	48.47	2.66
Peppers	40.96	(t-1)/3	40.43	0.046	44.41	(t-2)/4	46.85	0.5	48.47	2.66
Boat	40.93	(t-1)/3	41.32	0.046	44.11	(t-2)/4	46.96	0.5	48.50	2.66
Tiffany	40.89	(t-1)/3	41.35	0.046	45.1	(t-2)/4	46.84	0.5	48.45	2.66
Average	40.92	(t-1)/3	40.74	0.046	44.68	(t-2)/4	46.91	0.5	48.49	2.66

### 5.1.3.2 Robustness Analysis

Here the cover image is tested using Fridrich et al.'s [25] scheme and is calculated using equation (2.14). According to this method the cover image pixels are divided into three groups:

1. Regular ( $R_m$  or  $R_{-m}$ ),
2. Singular ( $S_m$  or  $S_{-m}$ ) and
3. Unusable group.

From experimental results, illustrated in Table 5.7 it is clear that the watermarked image will successfully approved by the RS analysis for  $R_m \cong R_{-m}$  and  $S_m \cong S_{-m}$ , otherwise it will be suspicious image. The Table 5.7 also represents that the difference between  $R_m$  and  $R_{-m}$ ,

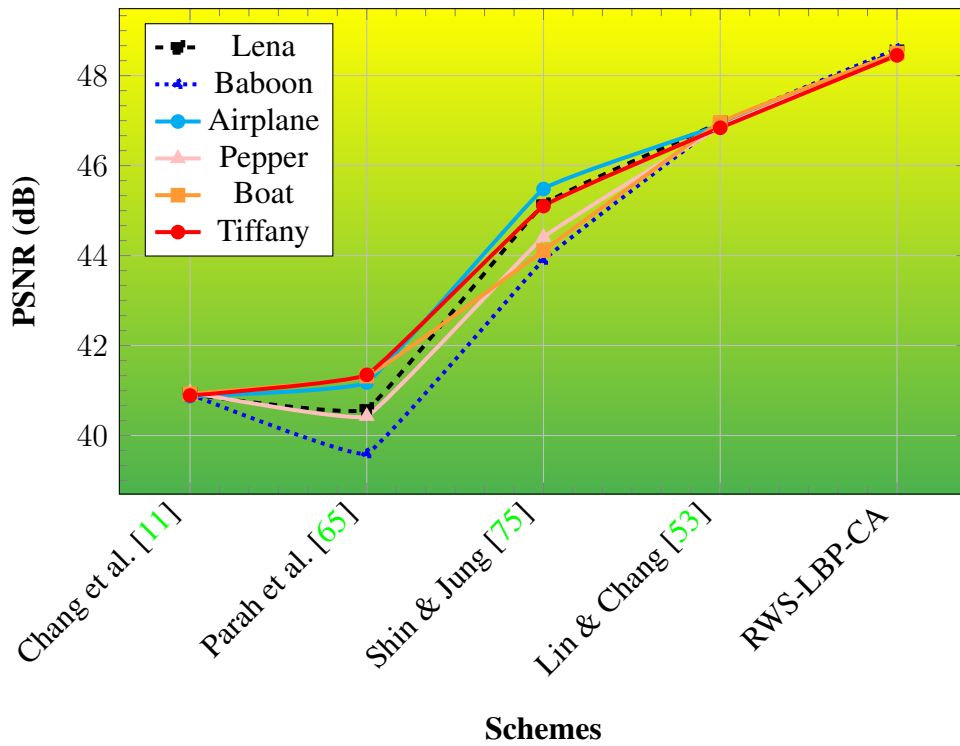


Figure 5.6: Comparison graph in terms of PSNR (dB) with RWS based existing methods in RWS-LBP-CA

Table 5.4: Average PSNR for various yardstick image datasets considering 25 to 100 images in RWS-LBP-CA

Datasets	Image Size	Total Image	Average PSNR
USC-SIPI [90]	512 × 512	25	48.43
		50	48.36
		100	48.51
STARE [89]	512 × 512	25	48.46
		50	48.35
		100	48.47
UCID [61]	512 × 512	25	48.45
		50	48.39
		100	48.49
HDR [26]	512 × 512	25	48.32
		50	48.16
		100	48.28

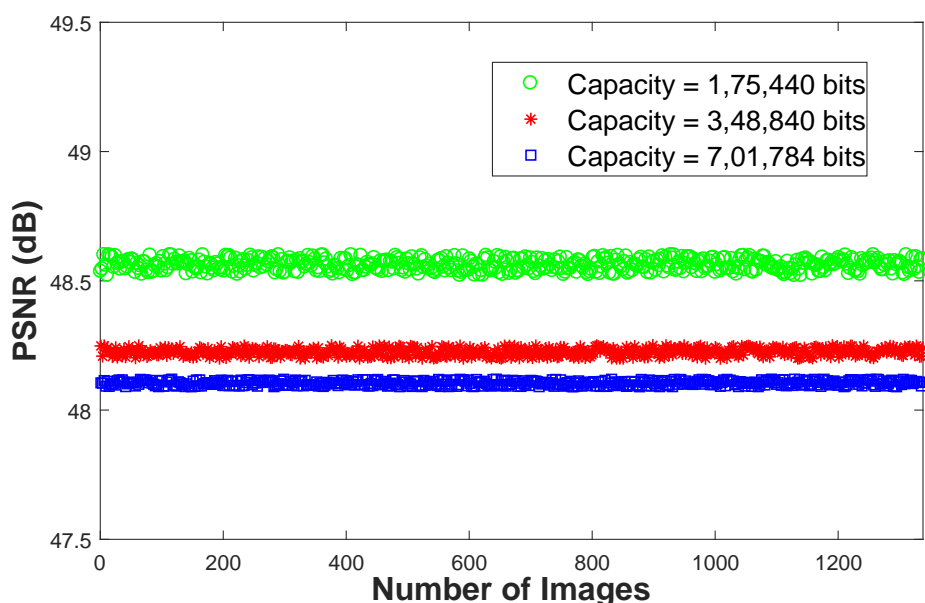


Figure 5.7: Graphical representation of PSNR (dB) on UCID image database [61] in RWS-LBP-CA

Table 5.5: Comparison with existing LBP based scheme in terms of PSNR in RWS-LBP-CA

Schemes	Parah et al. [65]	Wenyin et al. [94]	Pinjari et al. [68]	Zhang et al. [107]	RWS-LBP-CA	Improvement % w.r.t [107]
Lena	40.53	42.64	43.54	44.02	48.53	10.24
Airplane	40.99	41.37	43.59	44.32	48.47	9.36
Baboon	40.08	42.37	43.55	44.46	48.59	9.28
Boat	41.47	41.28	43.64	43.88	48.50	10.52
Pepper	40.71	42.52	43.53	44.61	48.47	8.65

Table 5.6: PSNR, SSIM, Q-Index, NCC and BER of distorted watermark images due to salt pepper noise, cropping and copy-move forgery attacks in RWS-LBP-CA

Noise	Perturbation	PSNR (dB)		SSIM		Q-Index		NCC		BER	
		Cover	Watermark	Cover	Watermark	Cover	Watermark	Cover	Watermark	Cover	Watermark
Salt and Pepper	0.01	22.20	13.03	61.25%	16.02%	0.9101	0.8248	0.9898	0.9701	0.0032	0.0272
	0.1	19.25	10.36	51.63%	11.26%	0.8341	0.6878	0.9801	0.9443	0.0065	0.0501
	0.5	17.51	8.84	41.16%	9.20%	0.7679	0.5752	0.9751	0.9204	0.0097	0.0695
Cropping	10%	15.63	10.85	97.44%	90.27%	0.7096	0.7340	0.9621	0.9496	0.0146	0.0213
	25%	10.85	6.85	88.73%	73.28%	0.5182	0.5033	0.9213	0.8681	0.0440	0.0584
	50%	07.42	3.93	73.86%	49.44%	0.1349	0.2737	0.8832	0.7194	0.0852	0.1128
Copymove Forgery	5 %	29.95	21.89	98.15%	97.14%	0.9313	0.9784	0.9880	0.9961	0.0033	0.0037
	10%	23.52	20.48	97.21%	95.91%	0.8956	0.9698	0.9914	0.9946	0.0047	0.0052
	20%	18.91	16.97	93.88%	90.54%	0.9040	0.9268	0.9726	0.9879	0.0109	0.0118

$S_m$  and  $S_{-m}$  are very closer, which establish that it is very difficult for the eavesdropper to detect any hidden message presence in the watermarked image. So the RWS-LBP-CA is secure

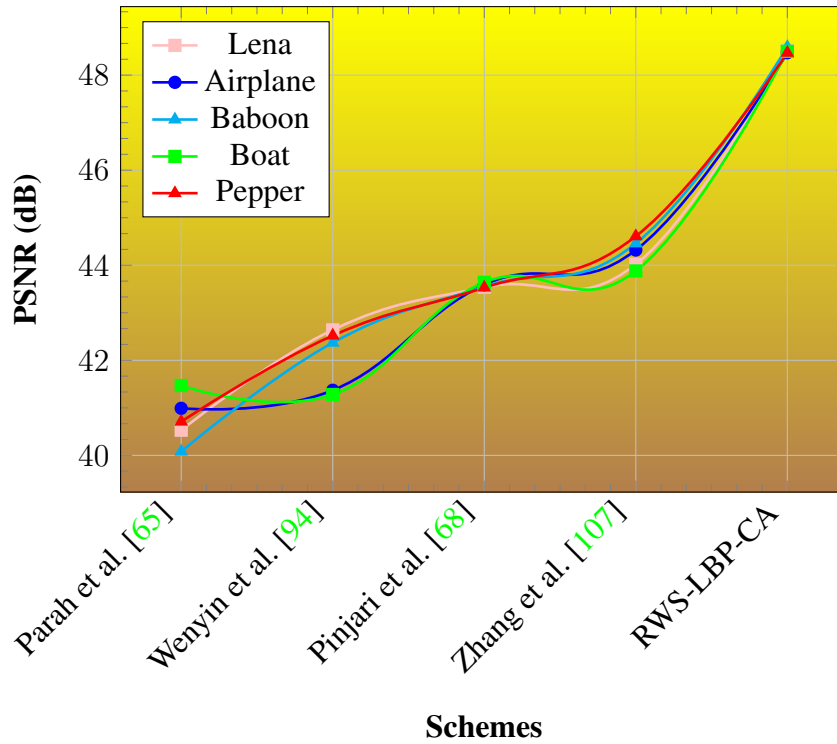


Figure 5.8: Comparison graph in terms of PSNR (dB) with LBP based existing schemes in RWS-LBP-CA

against RS attack.

The statistical analysis can be done using the standard rule of SD ( $\sigma$ ) and CC ( $\rho$ ) between CI and WI which are given in Table 5.8. Table 5.8 illustrates that SD ( $\sigma$ ) of CI is 128.5225 and that of WI is 124.4174, and differ by 0.1051 in case of Lena image. The CC ( $\rho$ ) between CI and WI is 0.9999 in case of Lena image. The less alteration between CI and WI represents that RWS-LBP-CA is perfectly secured watermarking method.

### 5.1.3.3 Tamper Detection and Recovery

Robustness of RWS-LBP-CA is analyzed by evaluating the quality metrics such as  $NCC$  [94],  $BER$  [65],  $SD$  and  $CC$  [35]. The RWS-LBP-CA has been evaluated against salt and pepper noise, cropping and copy-move forgery attacks. The experimental outcomes in attacking environments after applying salt and pepper noise, cropping and copy-move forgery attack with different noise density level have been depicted in Fig. 5.9, Fig. 5.10 and Fig. 5.11 respectively. It is clear that after extraction, the objective quality of the extracted watermark is slightly changed where as the tampered location of the recovered cover image has been identified suc-



Table 5.7: RS analysis between Cover image and Watermarked image in RWS-LBP-CA

Database	Image	Watermarked Images				
		RM	RM-	Sm	S-m	RS value
SIPI [90]	Lenna	57346	79227	39617	25235	0.374
	Baboon	54320	64594	44792	37374	0.1785
	Tiffany	68159	103067	35997	20708	0.4819
	Barbara	55018	71636	42344	30907	0.2882
	Airplane	58282	84043	36419	19998	0.4454
	Zelda	51392	73706	35239	20236	0.4308
HDR [26]	anhinga	57586	85152	44557	26083	0.4507
	beeflowr	61525	95005	41665	22037	0.5147
	jerusalem	61704	94326	44714	23019	0.5104
	redrock2	58790	76995	45346	33918	0.2846
	toucan	56947	97296	38112	12314	0.6959
UCID [61]	ucid00104	60689	76966	47407	36092	0.2553
	ucid00157	57639	69935	48838	39593	0.2023
	ucid00520	43632	57050	33252	24598	0.2871
	ucid00576	59521	84470	42110	25944	0.4046
	ucid00797	62883	88069	38911	25382	0.3803
STARE [89]	im0001	63941	103778	35285	12693	0.6292
	im0373	66760	108210	35589	12390	0.6317
	im0376	65524	104210	35579	13404	0.602
	im0386	65862	105149	35096	12438	0.6136

cessfully. The statistical analysis (SD and CC) shows the robustness of RWS-LBP-CA. The different objective metrics are presented in the corresponding presentation. The experimental results after applying three kinds of tamper on watermarked images are illustrated on Table 5.6. The SSIM, Q-Index and NCC results prove that RWS-LBP-CA can withstand on the versatile attacks like salt pepper, cropping and copy move forgery etc.

The algorithmic complexity of any watermarking scheme is a significant parameter incumbent research scenario. The execution time of RWS-LBP-CA has been compared with some recent works [65, 78, 91] and the comparison results have been reported in Table 5.9. It is clear that RWS-LBP-CA requires 0.5432 seconds for total execution which is 0.3513 seconds, 0.573 seconds and 0.573 seconds faster than Su et al. [78], Verma et al. [91] and Parah et al. [65] schemes











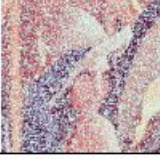





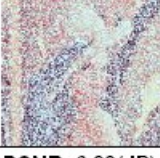

Original Image (513×513) (C)	Watermark (171 ×171) (W)	Watermarked Image (C')	Recovered Secret Data	Extracted Cover with tamper location	Extracted Cover without tamper	Statistical Analysis
						Difference of SD between C & C' = 128.52-128.51  =0.01 CC between C & C' = 0.9999
Lena	Logo Image	Density = 0.01	PSNR=26.34(dB)	PSNR=13.52(dB)		
						Difference of SD between C & C' = 128.52-128.50  =0.02 CC between C & C' = 0.9998
Lena	Logo Image	Density = 0.1	PSNR=18.45(dB)	PSNR=7.24(dB)		
						Difference of SD between C & C' = 128.52-128.49  =0.03 CC between C & C' = 0.9998
Lena	Logo Image	Density = 0.5	PSNR=11.26(dB)	PSNR=6.23(dB)		

Figure 5.9: Effect of salt and pepper noise on Lena image in RWS-LBP-CA



Original Image (513×513) (C)	Watermark (171 ×171) (W)	Watermarked Image (C')	Recovered Secret Data	Extracted cover with tampered region	Recovered Cover Image	Statistical Analysis
						Difference of SD between C & C' = 128.52-128.51  =0.01 CC between C & C' = 0.9999
Lena	Logo Image	Cropping 10%	PSNR=34.25(dB)	PSNR=30.54(dB)		
						Difference of SD between C & C' = 128.52-128.50  =0.02 CC between C & C' = 0.9998
Lena	Logo Image	Cropping 20%	PSNR=24.53(dB)	PSNR=23.45(dB)		
						Difference of SD between C & C' = 128.52-128.49  =0.03 CC between C & C' = 0.9998
Lena	Logo Image	Cropping 50%	PSNR=15.23(dB)	PSNR=12.34(dB)		

Figure 5.10: Effect of cropping attacks on Lena image in RWS-LBP-CA

Table 5.8: SD and CC results on different image datasets in RWS-LBP-CA

Image Dataset	Image	SD of CI	SD of WI	CC of CI & WI	Image Dataset	Image	SD of CI	SD of WI	CC of CI & WI
SIPI	Lenna	128.5225	128.5091	0.9999	UCID	ucid00104	171.8891	171.7671	0.9999
	Baboon	124.412	124.4174	0.9999		ucid00157	150.9891	150.9479	0.9999
	Zelda	138.0381	138.0992	0.9999		ucid00520	149.1946	149.217	0.9999
	Barbara	141.8444	141.8927	0.9999		ucid00576	207.722	207.7032	1
	Airplane	123.472	123.3987	0.9999		ucid00797	245.9329	245.4314	1
HDR	anhinga	140.9364	140.7592	0.9999	STARE	im0001	104.4944	104.7528	0.9998
	beeflowr	122.1338	121.9482	0.9998		im0373	172.9451	173.0046	0.9999
	jerusalem	97.0778	97.1365	0.9998		im0376	101.0019	101.2439	0.9998
	redrock2	131.1773	131.1285	0.9999		im0386	150.6766	150.7384	0.9999
	toucan	189.6997	189.1859	0.9999		im0275	146.7562	146.6324	0.9998








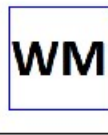










Original Image (513×513) (C)	Watermark (171 ×171) (W)	Watermarked Image (C')	Recovered Secret Data	Extracted cover with tamper region	Extracted cover without tamper region	Statistical Analysis
						Difference of SD between C & C' =  207.72-207.71  = 0.01 CC between C & C' = 1
Lena	Logo Image	CMF 5%	PSNR=26.34(dB)	PSNR=13.52(dB)		
						Difference of SD between C & C' =  207.72-207.70  = 0.02 CC between C & C' = 0.9999
Lena	Logo Image	CMF 10%	PSNR=18.45(dB)	PSNR=7.24(dB)		
						Difference of SD between C & C' =  207.72-207.69  = 0.03 CC between C & C' = 0.9998
Lena	Logo Image	CMF 20%	PSNR=11.26(dB)	PSNR=6.23(dB)		

Figure 5.11: Effect of copy-move forgery on Lena image in RWS-LBP-CA

respectively. The complexity in RWS-LBP-CA is achieved due to simple algebraic manipulations and the threading concept of Java.

To determine the algorithmic complexity, a cover image of size  $(M \times N)$  has been considered and a eight bit watermark is inserted into a  $(3 \times 3)$  pixel block. So, from the embedding Algorithm 5.1, it has been easily calculated that the time complexity is  $\mathcal{O}(MN)$  and at the time of extraction, the complexity is  $\mathcal{O}(MN)$ , considering intermediate steps in Algorithm 5.2. During

Table 5.9: Comparison table in terms of execution time in RWS-LBP-CA

Schemes	Size of Image	Embedding time (sec)	Extraction time (sec)	Total time (sec)
Su et al. [78]	$512 \times 512$	0.5244	0.3701	0.8945
Verma et al. [91]	$512 \times 512$	0.5173	0.5989	1.1162
Parah et al. [65]	$512 \times 512$	0.59	0.0624	0.6524
RWS-LBP-CA	$512 \times 512$	0.51	0.0332	0.5432

embedding only 0.51 seconds is acquired to insert ( $171 \times 171$ ) i.e., 7, 01, 784 bits watermark within ( $513 \times 513$ ) cover image and 0.0332 seconds is acquired at the time of extraction.

## 5.2 RWS-LBP-WM-LIP: RWS based on WM, LBP and LIP<sup>7</sup>

RWS for authentication and tamper detection plays a significant role in medical, military and government application. In this work, LIP is applied to ensure both data confidentiality and service availability whereas WM is used to embed watermark within the sub-sampled image. A repeated entry-wise-multiplication operation has been performed with each sub-sampled image block to increase the payload while retaining a high visual quality of the watermarked image. Moreover, authentication is achieved by employing AC, which is generated by SHA-512 and LFSR algorithm. Furthermore, a 128-bit shared secret key  $\mu$  is applied to make the algorithm secured. Again, the LBP operator is used to locating the tampered region. The experimental results are compared with the state-of-the-art schemes to presents the efficiency of the developed scheme. Some standard NIST recommended steganalysis and attacks are conducted to evaluate the robustness and imperceptibility. It is seen that RWS-LBP-WM-LIP is secured and robust counter to these attacks meanwhile it is able to detect tampered region.

The block diagram of RWS-LBP-WM-LIP for watermark embedding, extraction and authentication process have been depicted in Fig. 5.12(a), 5.12(b), and 5.12(c) respectively. The RWS-LBP-WM-LIP has been described in two subsections (5.2.1 and 5.2.2).

<sup>7</sup>Submitted in **IEEE MultiMedia**, **IEEE**, ISSN: 1070-986X, **Impact Factor: 1.898** with title *A Secured Reversible Color Image Watermarking Scheme based on LBP, Lagrange Interpolation Polynomial and Weighted Matrix*

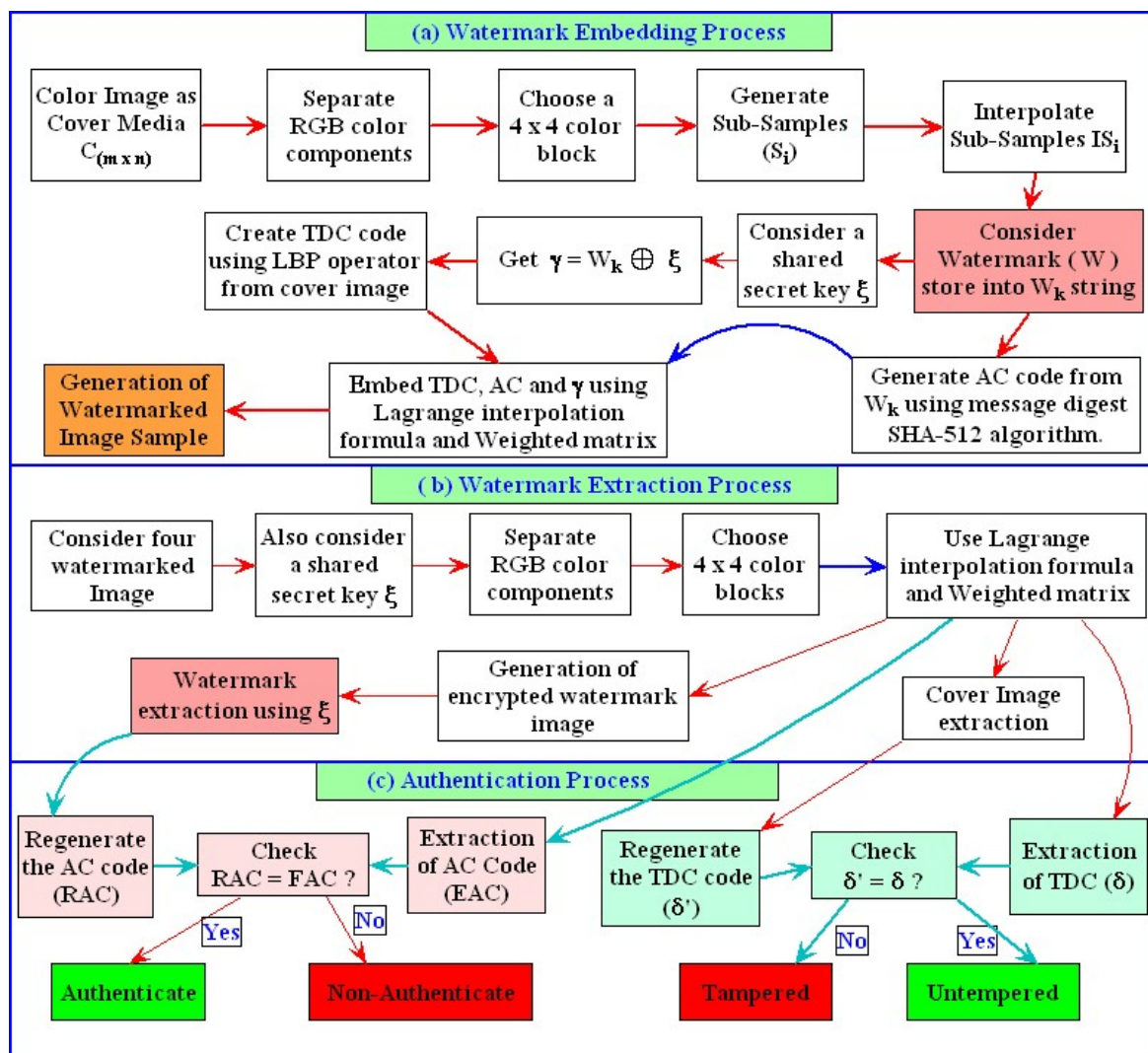


Figure 5.12: Block diagram of watermarking process in RWS-LBP-WM-LIP

### 5.2.1 Watermark Embedding Phase

A sub-sample image based RWS has been developed using LIP and WM, shown in Fig. 5.14. First a color image (CI) is considered as cover work. Then it is separated in RGB color components. After considering a color component, it is divided into a  $(4 \times 4)$  image blocks. Then four  $(2 \times 2)$  sub-samples SSI are produced from this  $(4 \times 4)$  image blocks using equation (5.2). Sample generation and interpolation methods are shown in Fig. 5.13. Four  $(4 \times 4)$  interpolated images ICI are constructed from each sub-sample using equation (5.3).

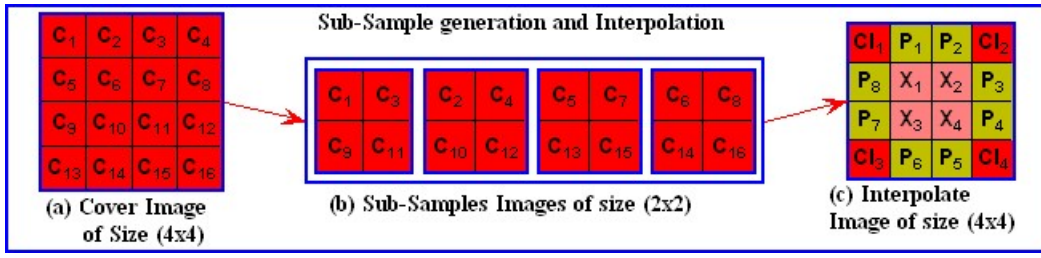


Figure 5.13: Schematic diagram of sub-sample generation and interpolation process in RWS-LBP-WM-LIP

$$\left. \begin{aligned} SCI_1(x, y) &= CI(2x - 1, 2y - 1) \\ SCI_2(x, y) &= CI(2x - 1, 2y) \\ SCI_3(x, y) &= CI(2x, 2y - 1) \\ SCI_4(x, y) &= CI(2x, 2y) \end{aligned} \right\} \forall x = 1 \text{ to } m; \text{ and } y = 1 \text{ to } n; \quad (5.2)$$

$$\left\{ \begin{aligned} I_{min} &= \min CI(x, y), CI(x + 2, y), CI(x, y + 2), CI(x + 2, y + 2) \\ I_{max} &= \max CI(x, y), CI(x + 2, y), CI(x, y + 2), CI(x + 2, y + 2) \\ AD &= \frac{3 \times I_{min} + 2 \times I_{max}}{5} \\ ICI(x, y) &= I(x, y) \\ ICI(x, y + 1) &= \frac{AD + (CI(x, y) + CI(x, y + 3))}{2} \\ ICI(x + 1, y) &= \frac{AD + (CI(x, y) + CI(x + 3, y))}{2} \\ ICI(x + 1, y + 1) &= \frac{(CI(x, y) + CI(x + 1, y) + CI(x, y + 1))}{3} \end{aligned} \right. \quad (5.3)$$

where  $x = 2m, y = 2n, m, n = 0, 1, 2, \dots, k$ . and  $m$  and  $n$  are considered as row and column of the CI. The ICI is generated from the each SSI which corresponds to CI individually presented in the equation (5.3). Here, a new interpolation scheme is proposed with an AD variable that provides a better quality interpolated image. Now, each ICI ( $4 \times 4$ ) is separated into three region shown in Fig. 5.13 (c),  $CI_i$  positions (red color region) are used to store the sample number,  $P_i$  positions (green color region) are used to store the position where the data bits are embedded and WM based entry-wise-multiplication is applied at the  $X_i$  positions (pink color region). Now a ( $2 \times 2$ ) WM is considered as input, before embedding W. Also a 128-bit  $\mu$  is taken into the account to enhance the purpose of security. The watermark image is converted into binary bits stream ( $M_k$ ) and 512-bit AC is generated by applying SHA-512 algorithm on ( $M_k$ ). Also 2 bit TDC ( $\delta$ ) is generated from CI using LBP operator. Now a random bit pattern is generated from  $\mu$  using a linear feedback shift register (LFSR) and collect 14 bits data by tapping after

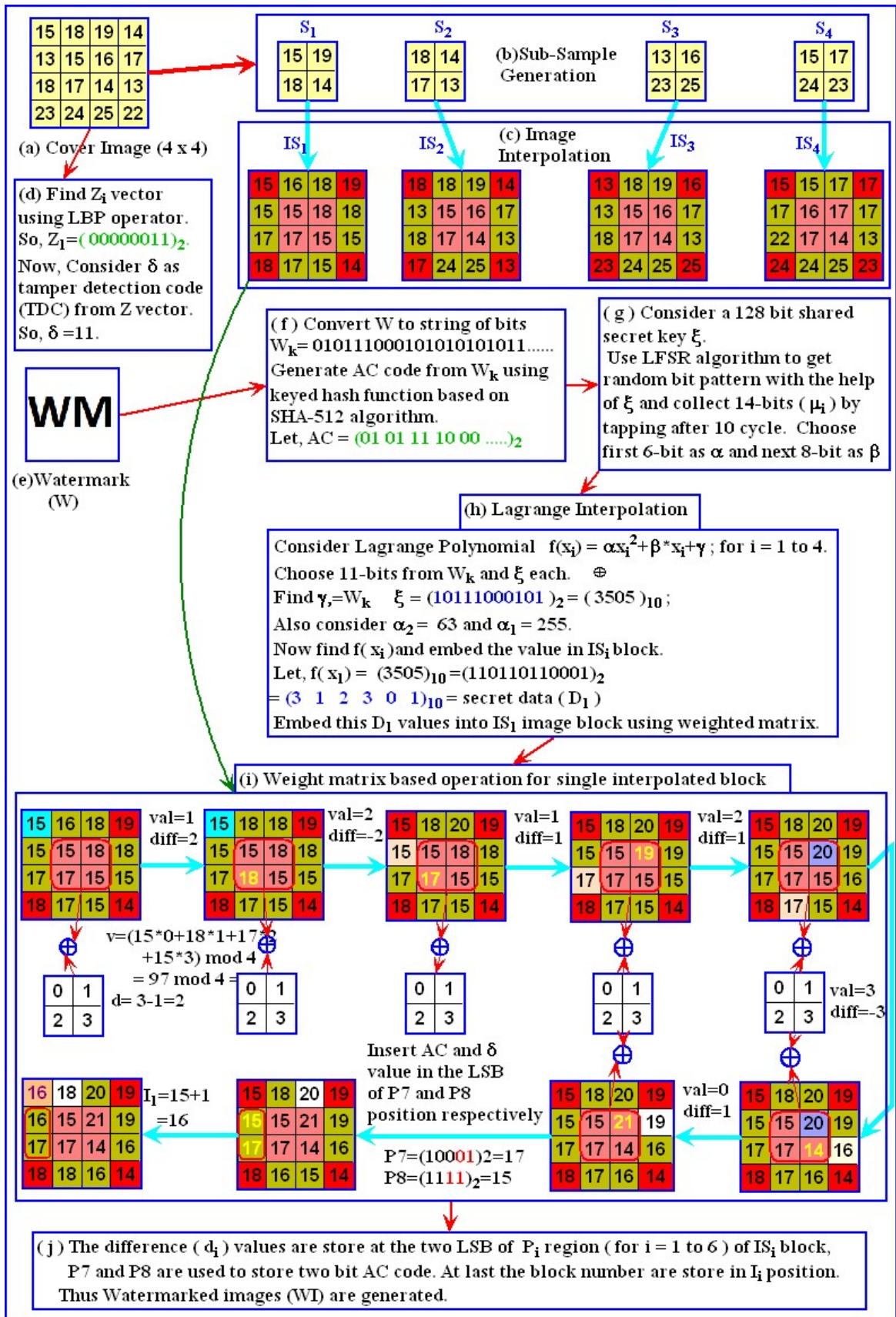


Figure 5.14: Numerical illustration of watermark embedding phase in RWS-LBP-WM-LIP

10 cycles. These 14 bits data are stored into  $\mu_i$  and separated into two portions. First 6 bits of  $\mu_i$  are stored into  $\alpha_i$  and next 8 bits of  $\mu_i$  are stored into  $\beta_i$ . Then 11 bits are collected from each of  $(M_k)$  and  $\mu$ . An XOR operation is performed between  $(M_k)$  and  $\mu$  to create encrypted watermark bits  $\gamma_i$ . Now from a LIP,  $f(X_p) = \alpha X_p^2 + \beta X_p + \gamma$ , where  $\alpha, \beta$  and  $\gamma$  are the coefficient of polynomial and  $p$  is the number of shares ( $p = 1$  to 4). After that this  $f(X_p)$  is modified to 12-bit binary digit and separated into 6 parts with 2 bit in each part. Now, these 6 parts are converted to decimal format and store into  $D_i$ . Then weighted matrix based operation is applied in the middle  $2 \times 2$  block (pink color region) of the image block  $IS_i$  and embed the encrypted watermark in the  $X_i$ . Also, the position value of the image blocks are stored into  $P_i$  for  $i = 1$  to 6. Thus all the  $D_i$  are inserted into the middle  $2 \times 2$  pixel block  $X_i$ . The pixels  $P_i$  for  $i = 1$  to 6 are modified to save the changed position in the matrix for all the iterations respectively. Now two-bit AC is inserted in the pixel  $P_7$  and two-bit TDC ( $\delta$ ) is inserted in the pixel  $P_8$ . Furthermore only increase the  $CI_i$  by one to identify the block number. The above process is applied to all the pixel blocks of the ICI to get four watermarked images.

### 5.2.2 Watermark Extraction and Recovery Phase

The extraction phase of RWS-LBP-WM-LIP has been clearly described using a numerical example depicted in Fig. 5.15 and an algorithmic illustration is shown in Algorithm 5.4. First, consider four watermarked image ( $ICI'$ ) and a shared secret key ( $\mu$ ). Then after separating RGB color component choose a  $(4 \times 4)$  pixel blocks from each of the samples. Then original cover work is constructed from the unaffected pixels  $CI_i$  for  $i = 1$  to 4 from each  $(4 \times 4)$  pixel blocks of all the shares ( $ICI'$ ). In this way the whole cover image is constructed from 4 shares. Now two LSBs are collected from each pixels  $P_7$  and  $P_8$  of the first block of the ( $ICI'$ ) to form the extracted authentication code (EAC) and TDC ( $\delta'$ ). Now consider  $(2 \times 2)$  WM matrix and apply in the  $X_i$  region to get  $D'_i$ . Then append all these  $D'_i$  values after converting to binary form and stored in  $B'_i$ . Now, these 12 bits  $B'_i$  are considered as  $F(X_i)$ . Also,  $\alpha$  and  $\beta$  values are considered from  $\mu$  using LFSR algorithm. Then encrypted watermark bits  $\gamma$  are generated from  $F(X_i)$  using  $\alpha, \beta$ , and LIP. After that, an XOR operation is performed with  $\gamma$  and  $\mu$  to generate watermark bits stream ( $M'$ ). Thus watermark might be easily regenerate from watermark bits stream  $M'$ . A hash key generator algorithm, SHA-512 is applied in  $M'$  to regenerate authenti-



**Algorithm 5.3:** RWS-LBP-WM-LIP: Watermark Embedding Algorithm

---

```

Input : Original Image (CI), Watermark Image (W), Secret Key ( $\mu$ )
Output: Sub-sampled watermarked images  $IS'_i$  for  $i = 1$  to 4

1 Algorithm Embedding():
   // Create sub sample images from the cover image
   // Create interpolated images from the sub sample images
   // Extract binary bits from the secret image
2 for (int y=0;y < imageHeight;y+=3) do
3     for (int x=0;x < imageWidth;x+=3) do
4         for ( color=0; color<3; color++) do
5             // FOR THREE COLOR COMPONENTS
6             number=GetFirst4BitsFromTheSecretMessage();
7             n=BinaryToDecimal(number);
8             // Process 1st interpolated image
9             result=GetResultFromEquation(1,n);
10            EmbedDataBitsInImageBlock(interpolatedArray1,color,1,x,y,result);
11            // Process 2nd interpolated image
12            result=GetResultFromEquation(2,n);
13            EmbedDataBitsInImageBlock(interpolatedArray2,color,2,x,y,result);
14            // Process 3rd interpolated image
15            result=GetResultFromEquation(3,n);
16            EmbedDataBitsInImageBlock(interpolatedArray3,color,3,x,y,result);
17            // Process 4th interpolated image
18            result=GetResultFromEquation(4,n);
19            EmbedDataBitsInImageBlock(interpolatedArray4,color,4,x,y,result);
20        end
21    end
22 end
23 CreateStegoImagesFromInterpolatedImages();
24 // End Procedure

19 Function EmbedDataBitsInImageBloc (interpolatedArray, color, xVal, posX, posY, result) :
20     strNumber=ConvertToBinaryString(result, 12);
21     first3bits=BinaryToDecimal(strNumber.substring(0, 3));
22     second3bits= BinaryToDecimal(strNumber.substring(3, 6));
23     third3bits= BinaryToDecimal(strNumber.substring(6, 9));
24     fourth3bits= BinaryToDecimal(strNumber.substring(9, 12));
25     interpolatedArray[posX+1][posY+0][color]+=first3bits;
26     interpolatedArray[posX+0][posY+1][color]+=second3bits;
27     interpolatedArray[posX+2][posY+1][color]+=third3bits;
28     interpolatedArray[posX+1][posY+2][color]+=fourth3bits;
29     interpolatedArray[posX+1][posY+1][color]+=xVal;
30 // End Procedure

```

---

cation code (RAC). Also, the LBP operator is used to reconstructed tamper detection code ( $\delta'$ ). Now check  $\delta$  with  $\delta'$  to serve the purpose of tamper detection and EAC with RAC to serve the purpose of authentication.

### 5.2.3 Experimental Results and Comparison

A set of benchmark colour images of size ( $512 \times 512$ ) (shown in 2.3) are considered to avail the efficiency of RWS-LBP-WM-LIP from [89], [61], [90], [26]. Three different sizes of logo images are considered as a watermark is shown in Fig. 5.16 to measure the quality and cor-

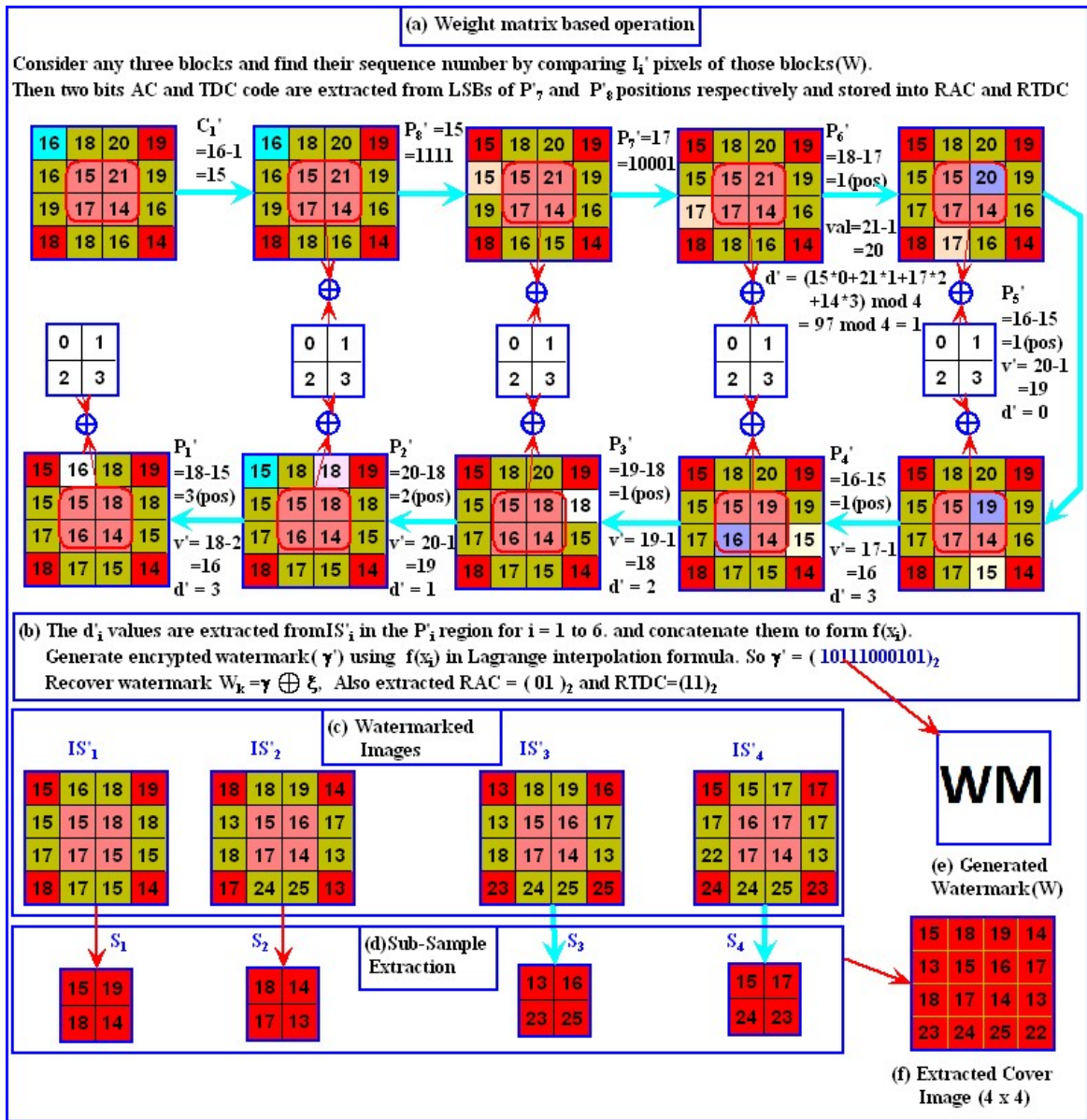


Figure 5.15: Numerical illustration of watermark extraction phase in RWS-LBP-WM-LIP

responding capacity. Performances of RWS-LBP-WM-LIP with the related schemes are compared to test its effectiveness. MSE [35], PSNR [35], SSIM [78] and Q-Index are computed using the equation (2.5), (2.6), (2.8) and (2.12) respectively to test the perceptible characteristics after embedding. Also NCC [94], BER [65], SD ( $\sigma$ ) [35] and CC ( $\rho$ ) [35] are computed using the equation (2.11), (2.13), (2.9) and (2.10) respectively for tamper detection in a watermarked image. Performance of RWS-LBP-WM-LIP is assessed by computation time, and it is compared with other existing schemes.

**Algorithm 5.4:** RWS-LBP-WM-LIP: Watermark Extraction Algorithm**Input** : Watermarked Images (WI) for  $i=1$  to 4 of size  $(m \times n)$ , Weighted Matrix (WM) and Shared Secret Keys ( $\mu$ )**Output**: Cover Image (CI) and Watermark Image (W)

```

1  Algorithm Extraction():
    // Select any three stego images for processing
2  for (int y=0;y:imageHeight;y+=3) do
3      for (int x=0;x:imageWidth;x+=3) do
4          for (color=0; color<3; color++) do
                    // FOR THREE COLOR COMPONENTS
5              x1=ExtractDataBitsFromImageBlock(stegoImage1.color,x,y,strNumber);
6              fx1=BinaryToDecimal(strNumber);
7              x2=ExtractDataBitsFromImageBlock(stegoImage2.color,x,y,strNumber);
8              fx2=BinaryToDecimal(strNumber);
9              x3=ExtractDataBitsFromImageBlock(stegoImage3.color,x,y,strNumber);
10             fx3=BinaryToDecimal(strNumber);
11             extractedNumber=GetValueFromLagrangeInterpolation(x1, fx1, x2, fx2, x3, fx3);
12             ExtractedSecretBits.append(DecimalToBinaryString(extractedNumber));
13         end
14     end
15 end
    // Create secret image from extracted binary string
16 BinaryStringToImage(ExtractedSecretBits);
    // End Procedure

17 Function ExtractDataBitsFromImageBlock (stegoImage, color, posX, posY, strNumber) :
18     firstNumber = stegoImage [posX+1][posY+0][color] - ( stegoImage [posX+0][posY+0][color] + stegoImage [posX+2][posY+0][color])/2;
19     secondNumber= stegoImage [posX+0][posY+1][color] - ( stegoImage [posX+0][posY+0][color] + stegoImage [posX+0][posY+2][color])/2;
20     thirdNumber = stegoImage [posX+2][posY+1][color] - ( stegoImage [posX+2][posY+0][color] + stegoImage [posX+2][posY+2][color])/2;
21     fourthNumber= stegoImage [posX+1][posY+2][color] - ( stegoImage [posX+0][posY+2][color] + stegoImage [posX+2][posY+2][color])/2;
22     xVal = stegoImage [posX+1][posY+1][color] - ( stegoImage [posX+0][posY+0][color] + stegoImage [posX+2][posY+0][color] + stegoImage [posX+0][posY+2][color] +
        stegoImage [posX+2][posY+2][color])/4;
23     strNumber.append(DecimalTo3BitBinary(firstNumber));
24     strNumber.append(DecimalTo3BitBinary(secondNumber));
25     strNumber.append(DecimalTo3BitBinary(thirdNumber));
26     strNumber.append(DecimalTo3BitBinary(fourthNumber));
27     return xVal;
28     // End Procedure

```

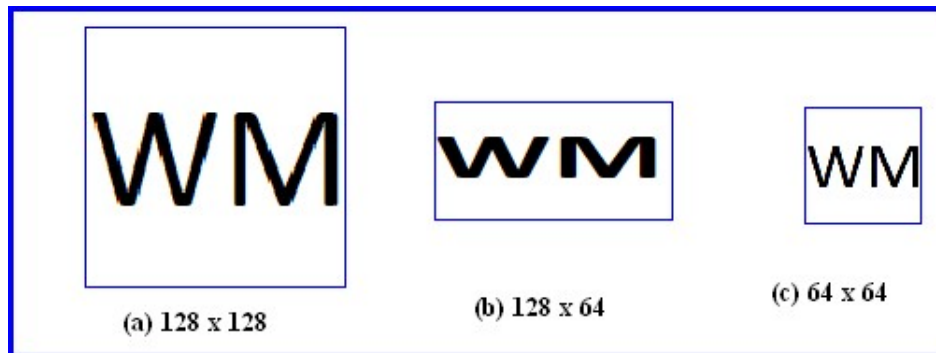


Figure 5.16: Watermark images (logo) with different size used in RWS-LBP-WM-LIP

**5.2.3.1 Quality Measurement and Payload Analysis**

The fundamental necessities of any watermarking scheme are robustness and imperceptibility. The subjective characteristics of the watermarked images are evaluated in RWS-LBP-WM-LIP, and it has been shown in Fig. 5.17. The evaluation results of RWS-LBP-WM-LIP in terms















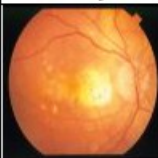


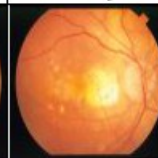
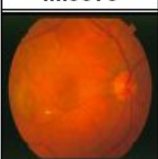

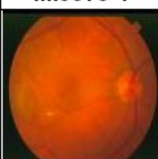
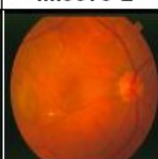
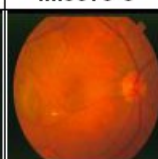
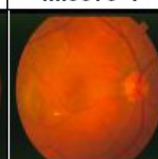






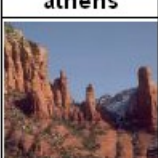

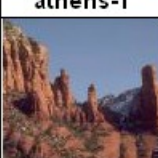
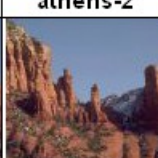
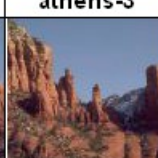
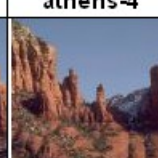
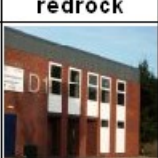

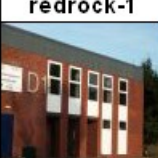
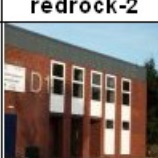
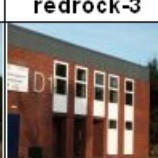

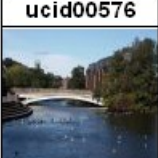


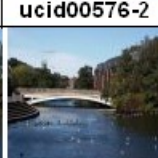


IMAGE DATABASE	COVER IMAGE	WATERMARK	OUTPUT WATERMARKED IMAGES			
USC-SIPI						
	Lena	128 x 128	Lena-1	Lena-2	Lena-3	Lena-4
						
	Tiffany	128 x 128	Tiffany-1	Tiffany-2	Tiffany-3	Tiffany-4
STARE						
	im0370	128 x 128	im0370-1	im0370-2	im0370-3	im0370-4
						
	im0376	128 x 128	im0376-1	im0376-2	im0376-3	im0376-4
HRD						
	athens	128 x 128	athens-1	athens-2	athens-3	athens-4
						
	redrock	128 x 128	redrock-1	redrock-2	redrock-3	redrock-4
UCID						
	ucid00576	128 x 128	ucid00576-1	ucid00576-2	ucid00576-3	ucid00576-4
						
	ucid00617	128 x 128	ucid00348-1	ucid00348-2	ucid00348-3	ucid00348-4

Figure 5.17: Pictorial results of output images in RWS-LBP-WM-LIP

PSNR and Q-Index after embedding a different number of bits of the watermark are presented in Table 5.10. It is observed from Fig. 5.17 that no visual distortions are detected after embedding maximum watermark of 5, 40, 672 bits. Also, Q-Index values are close to unity which establish the acceptability of RWS-LBP-WM-LIP. From Table 5.10 it is also observed that PSNR quality will be increased with decreasing the embedding capacity.

The RWS-LBP-WM-LIP has been tested taking more than 100 sample images from four different standard benchmark image databases, and experimental outcomes are presented in Table 5.11. Table 5.11 depicts that after embedding a maximum data of 5, 40, 672 bits watermark, approximately 53 dB average PSNR can be achieved.

Dataset	Image	Capacity (bits)	PSNR (dB)	Q-Index	Payload (bpp)
USC-SIPI [90]	Lena	1,35,168	52.94	0.99999	0.52
		2,70,336	52.92	0.99998	1.03
		5,40,672	52.91	0.99997	2.06
UCID [61]	Jerusalem	1,35,168	53.64	0.99999	0.52
		2,70,336	53.62	0.99997	1.03
		5,40,672	53.61	0.99996	2.06
STARE [89]	Im0001	1,35,168	53.98	0.99998	0.52
		2,70,336	53.88	0.99996	1.03
		5,40,672	53.94	0.99995	2.06
HDR [26]	Medical1	1,35,168	53.43	0.99998	0.52
		2,70,336	53.36	0.99997	1.03
		5,40,672	53.48	0.99995	2.06

The Fig. 5.18 depicts the graphical representation of the experimental results on PSNR considering 1338 images from UCID image database [61] at a different level of embedding capacity. Table 5.12 shows the test results in terms of MSE, PSNR, NCC, SSIM, Q-Index and BER with color cover images of four different yardstick datasets. It is found that the average PSNR for the aforesaid image databases is greater than 53 dB. Also the NCC, SSIM, and Q-Index values of RWS-LBP-WM-LIP are close to one, which establishes the effectiveness of the proposed algorithm.

The comparison with respect to capacity (in bits) and PSNR (dB) for Lena, Airplane, Ba-

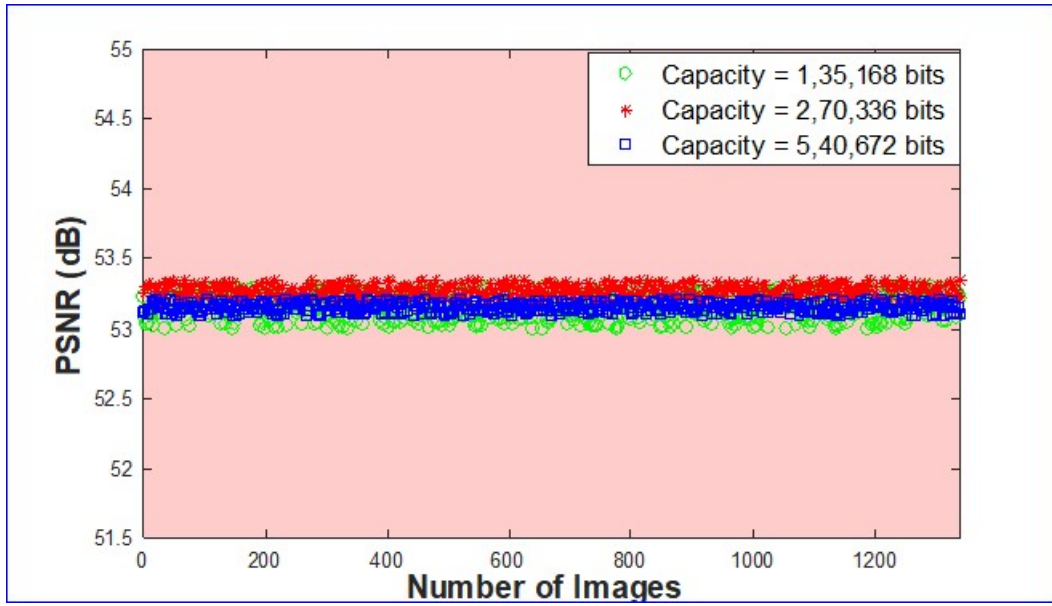


Figure 5.18: Graphical representation of PSNR (dB) on UCID image database [61] in RWS-LBP-WM-LIP

Table 5.11: Average PSNR of various yardstick image datasets considering 25 to 100 images in RWS-LBP-WM-LIP

Database	Size of Image	Number of Image	Average PSNR (dB)
USC-SIPI [90]	$512 \times 512$	25	53.35
		50	53.17
		100	52.91
UCID [61]	$512 \times 512$	25	54.34
		50	53.36
		100	52.45
STARE [89]	$512 \times 512$	25	53.98
		50	53.33
		100	52.92
HDR [26]	$512 \times 512$	25	53.78
		50	53.36
		100	53.17

boon, Tiffany, Boat, and Pepper images are shown in Table. 5.13. From the table it is seen that RWS-LBP-WM-LIP provides better results concerning capacity compared with other existing schemes.

Table 5.12: Results of MSE, PSNR, NCC, SSIM, Q-Index and BER for different image datasets in RWS-LBP-WM-LIP

Image Dataset	Images	MSE	PSNR (dB)	BER	NCC	Q-Index	SSIM
SIPI [90]	Lenna	0.7883	53.16	0.0391	0.9999	0.9998	98.41%
	Baboon	0.8815	52.67	0.0419	0.9999	0.9997	99.73%
	Tiffany	0.8147	53.02	0.0412	0.9999	0.9993	99.17%
	Barbara	0.8026	53.08	0.0419	0.9999	0.9998	99.52%
	Airplane	0.7991	53.10	0.0414	0.9999	0.9997	99.26%
	Goldhill	0.8002	53.09	0.0418	0.9999	0.9998	99.51%
	Average	0.8144	53.02	0.04121	0.9999	0.9996	99.26%
HDR [26]	anhinga	0.7651	53.29	0.0406	0.9999	0.9998	99.55%
	beeflowr	0.7383	53.44	0.3996	0.9999	0.9997	99.28%
	redrock2	0.7957	53.12	0.90420	0.9999	0.9998	99.50%
	toucan-s	0.7564	53.34	0.0362	0.9999	0.9999	99.19%
	Average	0.7638	53.29	0.3451	0.9999	0.9998	99.38%
UCID [61]	ucid00104	0.8032	53.08	0.0415	0.9999	0.9998	99.35%
	ucid00157	0.7993	53.10	0.0421	0.9999	0.9998	99.49%
	ucid00348	0.7996	53.10	0.0416	0.9999	0.9999	99.42%
	ucid00576	0.7975	53.11	0.0410	0.9999	0.9999	99.31%
	Average	0.7999	53.09	0.0415	0.9999	0.9998	99.39%
STARE [89]	im0001	0.7848	53.18	0.0585	0.9999	0.9997	98.93%
	im0373	0.7953	53.12	0.0410	0.9999	0.9998	98.91%
	im0376	0.7844	53.18	0.0387	0.9999	0.9996	98.88%
	im0386	0.7996	53.10	0.0409	0.9999	0.9998	98.86%
	Average	0.7910	53.14	0.0447	0.9999	0.9997	98.89%

### 5.2.3.2 Robustness Analysis

Robustness of RWS-LBP-WM-LIP has been judged by studying the evaluation metrics such as  $NCC$  [94],  $BER$  [65],  $\sigma$  and  $\rho$  [35]. Also, RWS-LBP-WM-LIP has been assessed against salt and pepper noise, cropping and copy-move forgery attacks.

The statistical distortion of RWS-LBP-WM-LIP is evaluated on the basis of the statistical parameter such as SD ( $\sigma$ ) and CC ( $\rho$ ). Experimental results are illustrated in Table 5.14 which represents that there is no substantial difference of  $\sigma$  between the CI and WI. Average SD ( $\sigma$ )

Table 5.13: Comparison of different RWT in sub-sample image with respect to PSNR and embedding capacity in RWS-LBP-WM-LIP

Image	Lin & Tsai [52]		Chang et al. [11]		Parah et al. [65]		Shin & Jung [75]		Lin & Chang [53]		RWS-LBP-WM-LIP	
	PSNR (dB)	P (bpp)	PSNR (dB)	P (bpp)	PSNR (dB)	P (bpp)	PSNR (dB)	P (bpp)	PSNR (dB)	P (bpp)	PSNR (dB)	P (bpp)
Lena	39.16	0.25	40.92	(t-1)/3	40.58	0.046	45.13	(t-2)/4	46.95	0.5	53.22	2.06
Baboon	39.15	0.25	40.92	(t-1)/3	39.60	0.046	43.9	(t-2)/4	46.92	0.5	53.97	2.06
Airplane	39.21	0.25	40.87	(t-1)/3	41.18	0.046	45.48	(t-2)/4	46.90	0.5	53.57	2.06
Peppers	39.20	0.25	40.96	(t-1)/3	40.43	0.046	44.41	(t-2)/4	46.85	0.5	54.18	2.06
Boat	39.18	0.25	40.93	(t-1)/3	41.32	0.046	44.11	(t-2)/4	46.96	0.5	53.96	2.06
Tiffany	39.13	0.25	40.89	(t-1)/3	41.35	0.046	45.1	(t-2)/4	46.84	0.5	53.13	2.06
Average	39.18	0.25	40.92	(t-1)/3	40.74	0.046	44.68	(t-2)/4	46.91	0.5	53.01	2.06

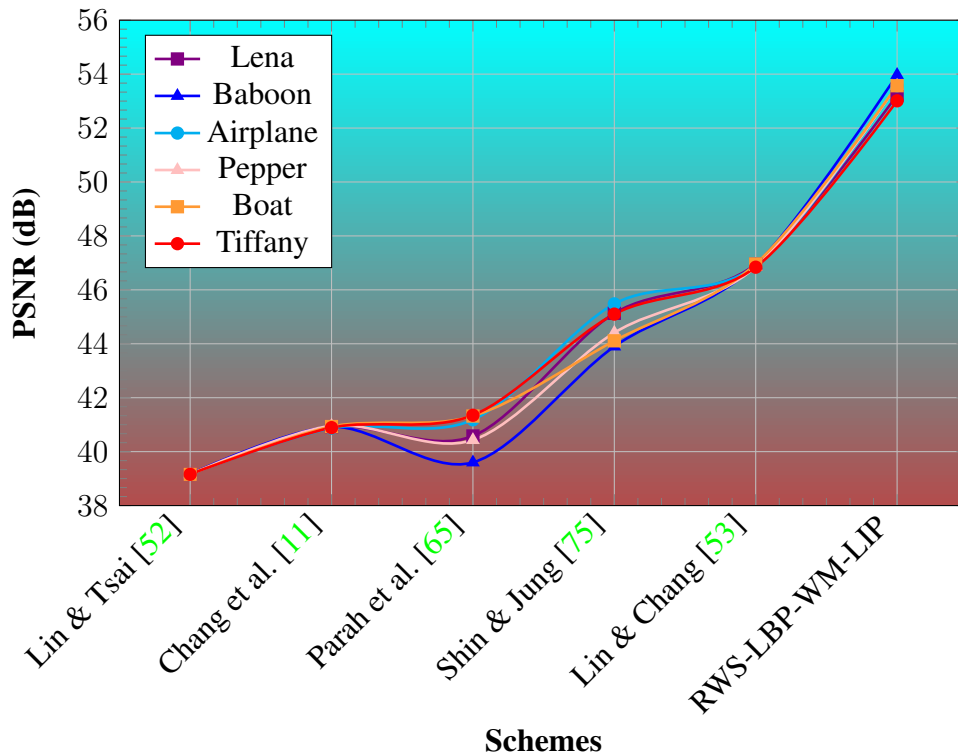


Figure 5.19: Comparison graph in terms of PSNR (dB) with RWS based existing methods in RWS-LBP-CA



Table 5.14: SD and CC results on different image datasets in RWS-LBP-WM-LIP

Image	$\sigma$ -C1	$\sigma$ -C2	$\sigma$ -C3	$\sigma$ -C4	$\sigma$ -C-avg	$\sigma$ -W1	$\sigma$ -W2	$\sigma$ -W3	$\sigma$ -W4	$\sigma$ -W-avg	$\rho$ -1	$\rho$ -2	$\rho$ -3	$\rho$ -4	$\rho$ -Avg	
<b>SIPI [90]</b>	Lenna	127.6179	127.5458	127.7119	127.6517	127.6318	127.6559	127.7793	127.7296	127.6941	0.9998	0.9997	0.9996	0.9996	0.9996	
	Baboon	121.3192	121.2657	122.4398	122.3518	121.8441	121.3557	122.4739	122.4013	121.8870	0.9998	0.9996	0.9975	0.9994	0.9990	
	Tiffany	72.4003	72.1889	75.8984	75.6297	74.0293	72.4208	72.2277	75.9622	75.6722	74.0707	0.9994	0.999	0.999	0.9989	0.9990
	Barbara	140.2492	140.2622	140.8333	140.8669	140.5529	140.3024	140.2766	140.8749	140.5832	140.5832	0.9998	0.9997	0.9997	0.9997	0.9997
	Airplane	121.7728	121.566	122.8384	122.6248	122.2005	121.7771	121.6319	122.8931	122.7003	122.2506	0.9998	0.9996	0.9996	0.9996	0.9996
	Goldhill	159.4374	159.3773	159.5788	159.4983	159.4729	159.4093	159.4389	159.6133	159.5819	159.5108	0.9999	0.9998	0.9998	0.9998	0.9998
<b>UCID [61]</b>	Ucid00104	169.8131	169.4256	169.8154	169.4501	169.6260	169.8045	169.4558	169.8326	169.4897	169.6496	0.9999	0.9998	0.9998	0.9998	0.9998
	Ucid00157	150.1343	150.8205	149.3549	150.0418	150.0878	150.131	150.8677	149.3942	150.1093	150.1255	0.9998	0.9999	0.9997	0.9997	0.9997
	Ucid00348	190.1126	190.0852	189.793	189.7482	189.9347	190.1047	190.1481	189.8379	189.8317	189.9806	0.9999	0.9999	0.9998	0.9998	0.9998
	Ucid00576	206.2429	206.5492	205.9266	206.3421	206.2404	206.2528	206.5923	205.9639	206.2969	206.2764	0.9999	0.9999	0.9999	0.9999	0.9999
	Ucid00520	144.23	144.2564	144.0007	143.9926	144.1199	144.2538	144.2818	144.0373	144.0221	144.1487	0.9999	0.9998	0.9997	0.9997	0.9997
	anhinga	139.5699	136.5319	136.792	139.9204	138.2035	139.4632	136.5393	136.7571	139.9454	138.1762	0.9998	0.9997	0.9997	0.9997	0.9997
<b>HDR [26]</b>	beeflowr	120.2479	120.2045	120.1675	120.1513	120.1928	120.2006	120.2261	120.1703	120.1936	120.1976	0.9998	0.9997	0.9996	0.9996	0.9996
	jerusalem	95.9362	96.168	96.0577	95.7881	95.9875	96.0074	96.206	96.1232	95.8252	96.0404	0.9997	0.9994	0.9994	0.9993	0.9994
	redrock2	129.6262	129.7495	129.9567	129.8381	129.7926	129.6381	129.8093	130.0033	129.9105	129.8403	0.9998	0.9997	0.9997	0.9996	0.9997
	toucan-s	183.2765	183.5847	183.3504	183.6477	183.4648	183.0975	183.4406	183.1958	183.5102	183.3110	0.9999	0.9998	0.9998	0.9998	0.9998
	im0001	105.9174	105.9177	105.8929	105.8964	105.9061	106.093	105.9371	105.9981	105.8868	105.9787	0.9997	0.9995	0.9995	0.9994	0.9995
	im0373	171.7441	171.7698	171.7643	171.788	171.7665	171.852	171.6966	171.7822	171.6773	171.7520	0.9999	0.9998	0.9998	0.9998	0.9998
<b>STARE [89]</b>	im0376	102.344	102.3507	102.2566	102.2662	102.3043	102.5126	102.3704	102.3595	102.2608	102.3758	0.9997	0.9995	0.9995	0.9994	0.9995
	im0386	150.1352	150.1318	150.1558	150.1357	150.1396	150.2144	150.1003	150.1841	150.0854	150.1460	0.9999	0.9998	0.9997	0.9997	0.9997

value of CI and WI is 127.6179 and 127.6559 respectively and differs by 0.380 in case of Lena image after embedding 11 bits watermark. Average CC ( $\rho$ ) value between the CI and WI is 0.9998 in case of Lena image. So finding the watermark from the watermarked image become quite difficult. The less alteration between CI and WI represents that RWS-LBP-WM-LIP is perfectly secured watermarking method.

### 5.2.3.3 Tamper Detection and Recovery

Robustness of RWS-LBP-WM-LIP is analyzed by evaluating the quality metrics such as PSNR, SSIM, Q-Index, NCC, and BER in the presence of salt and pepper noise, cropping and copy-move forgery attacks. The Fig. 5.20, Fig. 5.21 and Fig. 5.22 represent the results after applying salt and pepper noise, cropping and copy-move forgery attack with different noise density level respectively. It is clear that after extraction, the objective quality of the extracted watermark is slightly changed but recovered cover image has been identified successfully. Also the results of BER and NCC of cover and extracted cover image and as well as BER and NCC of the watermark and extracted watermark show the robustness of RWS-LBP-WM-LIP. The different objective metrics are presented in Table 5.15 when extraction is performed from tamper image. From Table 5.15, it is noted that the fewer BER values and near unity Q-Index and NCC indicate the robustness of RWS-LBP-WM-LIP during some standard attack. Again 5.15 represents that robustness of RWS-LBP-WM-LIP varies inversely with the noise density.

The algorithmic complexity and the computation time of any watermarking scheme are the significant parameters for recent research scenario. The execution time of RWS-LBP-WM-LIP has been compared with some recent works [65,78,91], and comparative outcomes are presented in Table 5.16. It is found that RWS-LBP-WM-LIP requires 0.514 seconds for total execution which is 0.3805 seconds, 0.6022 seconds and 0.1384 seconds faster than Su et al. [78], Verma et al. [91] and Parah et al. [65] schemes respectively. During embedding only 0.315 seconds acquired to insert  $(256 \times 64)$ , i.e., 5, 40, 672 bits watermark into  $(512 \times 512)$  cover image and 0.199 seconds is acquired at the time of extraction. The lesser execution time in RWS-LBP-WM-LIP is achieved due to simple algebraic manipulations and the threading concept of Java.

To determine the algorithmic complexity, a cover image of size  $(M \times N)$  has been considered. The time complexity for doing the operations described in Algorithm 5.3 is  $\mathcal{O}(MN)$ . On the other hand, at the time of extraction, the complexity is  $\mathcal{O}(MN)$ , considering Algorithm 5.4.

Table 5.15: PSNR, SSIM, Q-Index, NCC and BER results on distorted watermark images due to salt pepper noise, cropping and copy-move forgery attacks in RWS-LBP-WM-LIP

Noise	Sample	Perturbation	PSNR (dB)		SSIM		Q-Index		NCC		BER		
			CI	WI	CI	WI	CI	WI	CI	WI	CI	WI	
Salt and Pepper	C1	0.01	25.26	20.30	77.64	67.92	0.9541	0.9739	0.9949	0.9940	0.0016	0.0065	
		0.1	21.36	16.43	57.69	53.63	0.8926	0.9372	0.9876	0.9855	0.0040	0.0157	
		0.5	18.62	13.55	39.02	44.93	0.8126	0.8753	0.9769	0.9714	0.0075	0.0308	
	C2	0.01	25.36	20.40	77.57	68.32	0.9542	0.9737	0.9950	0.9942	0.0015	0.0064	
		0.1	21.34	16.45	57.67	53.66	0.8929	0.9371	0.9877	0.9859	0.0042	0.0159	
		0.5	18.62	13.58	39.06	44.96	0.8125	0.8754	0.9762	0.9716	0.0076	0.0307	
	C3	0.01	25.63	20.25	77.89	67.87	0.9543	0.9736	0.9947	0.9941	0.0014	0.0059	
		0.1	21.37	16.48	57.64	53.61	0.8927	0.9377	0.9871	0.9851	0.0044	0.0154	
		0.5	18.62	13.56	39.07	44.97	0.8127	0.8755	0.9767	0.9717	0.0077	0.0304	
	C4	0.01	25.21	20.34	77.89	67.98	0.9541	0.9735	0.9948	0.9945	0.0012	0.0062	
		0.1	21.38	16.44	57.66	53.67	0.8921	0.9379	0.9873	0.9857	0.0048	0.0156	
		0.5	18.65	13.54	39.08	44.95	0.8124	0.8757	0.9764	0.9715	0.0074	0.0296	
	C1 & C2	0.01	22.23	17.55	61.37	57.53	0.9110	0.9479	0.9898	0.9888	0.0030	0.0135	
		0.1	19.32	15.53	42.14	46.92	0.8926	0.8726	0.9876	0.9855	0.0057	0.0155	
		0.5	15.60	12.68	27.54	34.56	0.6961	0.7123	0.9546	0.9483	0.0151	0.0543	
	C1 & C2 & C3	0.01	20.46	15.79	49.58	51.83	0.8707	0.9202	0.9848	0.9832	0.0049	0.0158	
		0.1	17.54	13.59	36.64	45.26	0.8926	0.8294	0.9526	0.9545	0.0089	0.0249	
		0.5	14.16	10.15	19.34	29.34	0.5934	0.6084	0.9364	0.9341	0.0234	0.0697	
	C1 & C2 & C3 & C4	0.01	19.23	14.63	41.15	48.87	0.8341	0.8955	0.9799	0.9781	0.0060	0.0245	
		0.1	15.46	11.71	22.35	34.12	0.6423	0.6821	0.9456	0.9451	0.0103	0.0642	
		0.5	12.57	8.26	12.57	23.80	0.4798	0.5359	0.9188	0.9178	0.0312	0.0912	
	Cropping	C1	10 %	21.01	16.20	91.73	90.12	0.8921	0.9240	0.9871	0.9846	0.0043	0.0144
			25 %	17.31	10.27	79.88	73.63	0.7809	0.7184	0.9728	0.9381	0.0109	0.0448
			50 %	13.46	5.82	59.64	47.89	0.5441	0.4087	0.9425	0.8189	0.0231	0.1021
C2		10 %	21.03	16.21	91.77	90.15	0.8915	0.9241	0.9873	0.9844	0.0045	0.0145	
		25 %	17.32	10.26	79.87	73.67	0.7806	0.7182	0.9724	0.9386	0.0107	0.0442	
		50 %	13.43	5.81	59.61	47.87	0.5442	0.4089	0.9423	0.8183	0.0232	0.1028	
C3		10 %	21.05	16.22	91.71	90.13	0.8917	0.9239	0.9872	0.9845	0.0041	0.0147	
		25 %	17.33	10.24	79.84	73.66	0.7803	0.7186	0.9729	0.9383	0.0105	0.0444	
		50 %	13.49	5.86	59.66	47.84	0.5446	0.4085	0.9429	0.8185	0.0239	0.1024	
C4		10 %	21.09	16.23	91.75	90.14	0.8919	0.9243	0.9876	0.9847	0.0047	0.0149	
		25 %	17.35	10.21	79.83	73.61	0.7801	0.7189	0.9722	0.9387	0.0102	0.0446	
		50 %	13.47	5.87	59.67	47.91	0.5444	0.4086	0.9424	0.8186	0.0236	0.1026	
C1 & C2		10 %	17.80	14.31	90.35	86.44	0.7992	0.8816	0.9751	0.9760	0.0095	0.0183	
		25 %	14.56	11.76	73.45	63.49	0.4538	0.7456	0.9456	0.9358	0.0153	0.0453	
		50 %	10.46	8.56	56.23	47.84	0.3296	0.5556	0.9109	0.9082	0.0461	0.0761	
C1 & C2 & C3		10 %	16.08	15.05	90.41	88.08	0.7303	0.8991	0.9652	0.9799	0.0141	0.0183	
		25 %	12.25	10.23	69.23	61.91	0.4315	0.6724	0.9236	0.8564	0.0456	0.0546	
		50 %	8.16	6.23	54.36	46.52	0.2463	0.4561	0.8986	0.7643	0.07324	0.0953	
C1 & C2 & C3 & C4		10 %	14.85	11.17	95.64	88.17	0.6736	0.7784	0.9563	0.9499	0.0188	0.0228	
		25 %	11.23	9.26	68.49	59.57	0.4126	0.5482	0.9146	0.8357	0.0413	0.0654	
		50 %	7.45	4.35	52.65	44.53	0.0831	0.3188	0.8829	0.7284	0.0922	0.1091	
Copy Move Forgery		C1	5 %	27.29	22.06	98.67	96.51	0.9934	0.9804	0.9999	0.9962	0.0003	0.0037
			10 %	26.35	20.94	98.47	94.56	0.9885	0.9748	0.9999	0.9938	0.0009	0.0048
			20 %	25.38	18.12	98.32	91.56	0.9814	0.9556	0.9999	0.9901	0.0014	0.0081
	C2	5 %	27.25	22.05	98.65	96.54	0.9950	0.9804	0.9999	0.9965	0.0002	0.0038	
		10 %	26.33	20.91	98.44	94.54	0.9883	0.9744	0.9999	0.9936	0.0010	0.0046	
		20 %	25.36	18.13	98.33	91.54	0.9869	0.9558	0.9999	0.9903	0.0015	0.0082	
	C3	5 %	27.55	22.04	98.66	96.52	0.9899	0.9804	0.9999	0.9964	0.0001	0.0036	
		10 %	26.36	20.92	98.42	94.52	0.9898	0.9746	0.9999	0.9934	0.0006	0.0047	
		20 %	25.35	18.16	98.34	91.51	0.9897	0.9554	0.9999	0.9904	0.0012	0.0085	
	C4	5 %	27.29	22.06	98.63	96.53	0.9837	0.9804	0.9981	0.9962	0.0002	0.0035	
		10 %	26.57	20.95	98.48	94.51	0.9874	0.9742	0.9963	0.9935	0.0008	0.0044	
		20 %	25.73	18.11	98.31	91.53	0.9814	0.9553	0.99945	0.9900	0.0013	0.0084	
	C1 & C2	5 %	27.25	21.77	98.52	96.20	0.9898	0.9805	0.9927	0.9957	0.0011	0.0044	
		10 %	25.60	18.94	97.12	93.48	0.9897	0.9604	0.9965	0.9902	0.0018	0.0076	
		20 %	24.79	16.90	97.89	89.84	0.9895	0.9403	0.9933	0.9868	0.0025	0.0107	
	C1 & C2 & C3	5 %	26.55	20.36	97.45	95.89	0.9898	0.9745	0.9902	0.9938	0.0016	0.0051	
		10 %	24.95	18.12	97.96	91.23	0.9896	0.9536	0.9869	0.9899	0.0024	0.0392	
		20 %	23.05	16.13	97.46	88.54	0.9893	0.9356	0.9935	0.9870	0.0038	0.0703	
	C1 & C2 & C3 & C4	5 %	27.25	19.52	96.30	95.26	0.9897	0.9659	0.9973	0.9928	0.0021	0.0059	
		10 %	24.66	17.39	96.94	91.57	0.9894	0.9456	0.9810	0.9903	0.0036	0.0729	
		20 %	22.72	15.72	96.21	87.64	0.9891	0.9224	0.9750	0.9872	0.0052	0.1401	

Cover Image (512×512) (C)	Watermark (128 ×128)	watermarked image(CW1)	watermarked image(CW2 )	watermarked image(CW3)	watermarked image(CW4)	Extracted watermark	Extracted Cover Image	Experimental Results
								HCC(C & C') = 0.9949 HCC(W & W') = 0.9940 BER ( C & C') = 0.0016 BER ( W & W') = 0.0065
Lena	Logo Image	SaltPepper 0.01	No Attack	No Attack	No Attack	PSNR=20.30(dB)	PSNR=25.26(dB)	
								HCC ( C & C') = 0.9898 HCC ( W & W') = 0.9888 BER ( C & C') = 0.003 BER ( W & W') = 0.0135
Lena	Logo Image	SaltPepper 0.01	SaltPepper 0.01	No Attack	No Attack	PSNR=17.55(dB)	PSNR=22.23(dB)	
								HCC ( C & C') = 0.9848 HCC ( W & W') = 0.9832 BER ( C & C') = 0.0049 BER ( W & W') = 0.0158
Lena	Logo Image	SaltPepper 0.01	SaltPepper 0.01	SaltPepper 0.01	No Attack	PSNR=15.79(dB)	PSNR=20.16(dB)	
								HCC ( C & C') = 0.9799 HCC ( W & W') = 0.9781 BER ( C & C') = 0.0060 BER ( W & W') = 0.0245
Lena	Logo Image	SaltPepper 0.01	SaltPepper 0.01	SaltPepper 0.01	SaltPepper 0.01	PSNR=19.23(dB)	PSNR=14.63(dB)	
								HCC ( C & C') = 0.9876 HCC ( W & W') = 0.9855 BER ( C & C') = 0.0040 BER ( W & W') = 0.0157
Lena	Logo Image	SaltPepper 0.1	No Attack	No Attack	No Attack	PSNR=16.43(dB)	PSNR=21.36(dB)	
								HCC ( C & C') = 0.9876 HCC ( W & W') = 0.9855 BER ( C & C') = 0.0057 BER ( W & W') = 0.0155
Lena	Logo Image	SaltPepper 0.1	SaltPepper 0.1	No Attack	No Attack	PSNR=15.53(dB)	PSNR=19.32(dB)	
								HCC ( C & C') = 0.9526 HCC ( W & W') = 0.9545 BER ( C & C') = 0.0089 BER ( W & W') = 0.0249
Lena	Logo Image	SaltPepper 0.1	SaltPepper 0.1	SaltPepper 0.1	No Attack	PSNR=13.59(dB)	PSNR=17.54(dB)	
								HCC ( C & C') = 0.9156 HCC ( W & W') = 0.9451 BER ( C & C') = 0.0103 BER ( W & W') = 0.0642
Lena	Logo Image	SaltPepper 0.1	SaltPepper 0.1	SaltPepper 0.1	SaltPepper 0.5	PSNR=11.71(dB)	PSNR=15.16(dB)	
								HCC ( C & C') = 0.9769 HCC ( W & W') = 0.9711 BER ( C & C') = 0.0075 BER ( W & W') = 0.0308
Lena	Logo Image	SaltPepper 0.5	No Attack	No Attack	No Attack	PSNR=13.55(dB)	PSNR=18.62(dB)	
								HCC ( C & C') = 0.9546 HCC ( W & W') = 0.9483 BER ( C & C') = 0.0151 BER ( W & W') = 0.0543
Lena	Logo Image	SaltPepper 0.5	SaltPepper 0.5	No Attack	No Attack	PSNR=12.68(dB)	PSNR=15.60(dB)	
								HCC ( C & C') = 0.9361 HCC ( W & W') = 0.9341 BER ( C & C') = 0.0231 BER ( W & W') = 0.0697
Lena	Logo Image	SaltPepper 0.5	SaltPepper 0.5	SaltPepper 0.5	No Attack	PSNR=10.15(dB)	PSNR=14.16(dB)	
								HCC ( C & C') = 0.9188 HCC ( W & W') = 0.9178 BER ( C & C') = 0.0312 BER ( W & W') = 0.0912
Lena	Logo Image	SaltPepper 0.5	SaltPepper 0.5	SaltPepper 0.5	SaltPepper 0.5	PSNR=8.26(dB)	PSNR=12.57(dB)	

Figure 5.20: Effect of salt pepper noise on Lena image in RWS-LBP-WM-LIP

Cover Image (512×512) (C)	Watermark (128 ×128)	watermarked image(CW1)	watermarked image(CW2 )	watermarked image(CW3)	watermarked image(CW4)	Recovered watermark	Recovered Cover Image	Experimental Results
								HCC(C & C') = 0.9871 HCC(W & W')= 0.9846 BER ( C & C') = 0.0043 BER ( W & W')=0.0141
Lena	Logo Image	Cropping 10%	No Attack	No Attack	No Attack	PSNR=20.30(dB)	PSNR=25.26(dB)	
								HCC(C & C') = 0.9751 HCC(W & W')= 0.9760 BER ( C & C') = 0.0095 BER ( W & W')=0.0183
Lena	Logo Image	Cropping 10%	Cropping 10%	No Attack	No Attack	PSNR=20.40(dB)	PSNR=25.36(dB)	
								HCC(C & C') = 0.9652 HCC(W & W')= 0.9799 BER ( C & C') = 0.0141 BER ( W & W')=0.0213
Lena	Logo Image	Cropping 10%	Cropping 10%	Cropping 10%	No Attack	PSNR=14.87(dB)	PSNR=31.67(dB)	
								HCC(C & C') = 0.9563 HCC(W & W')= 0.9499 BER ( C & C') = 0.0188 BER ( W & W')=0.0228
Lena	Logo Image	Cropping 10%	Cropping 10%	Cropping 10%	Cropping 10%	PSNR=6.45(dB)	PSNR=11.23(dB)	
								HCC(C & C') = 0.9728 HCC(W & W')= 0.9381 BER ( C & C') = 0.0109 BER ( W & W')=0.0448
Lena	Logo Image	Cropping 25%	No Attack	No Attack	No Attack	PSNR=8.67(dB)	PSNR=13.36(dB)	
								HCC(C & C') = 0.9156 HCC(W & W')= 0.9358 BER ( C & C') = 0.0153 BER ( W & W')=0.0463
Lena	Logo Image	Cropping 25%	Cropping 25%	No Attack	No Attack	PSNR=7.36(dB)	PSNR=16.34(dB)	
								HCC(C & C') = 0.9236 HCC(W & W')= 0.8561 BER ( C & C') = 0.0456 BER ( W & W')=0.0546
Lena	Logo Image	Cropping 25%	Cropping 25%	Cropping 25%	No Attack	PSNR=6.37(dB)	PSNR=35.62(dB)	
								HCC(C & C') = 0.9446 HCC(W & W')= 0.8356 BER ( C & C') = 0.0413 BER ( W & W')=0.0564
Lena	Logo Image	Cropping 25%	Cropping 25%	Cropping 25%	Cropping 25%	PSNR=6.35(dB)	PSNR=36.28(dB)	
								HCC(C & C') = 0.9125 HCC(W & W')= 0.8189 BER ( C & C') = 0.0231 BER ( W & W')=0.1021
Lena	Logo Image	Cropping 50%	No Attack	No Attack	No Attack	PSNR=5.23(dB)	PSNR=32.64(dB)	
								HCC(C & C') = 0.9109 HCC(W & W')= 0.9082 BER ( C & C') = 0.0161 BER ( W & W')=0.0761
Lena	Logo Image	Cropping 50%	Cropping 50%	No Attack	No Attack	PSNR=5.23(dB)	PSNR=32.64(dB)	
								HCC(C & C') = 0.8986 HCC(W & W')= 0.7643 BER ( C & C') = 0.0732 BER ( W & W')=0.0953
Lena	Logo Image	Cropping 50%	Cropping 50%	Cropping 50%	No Attack	PSNR=5.23(dB)	PSNR=32.64(dB)	
								HCC(C & C') = 0.8829 HCC(W & W')= 0.7281 BER ( C & C') = 0.0922 BER ( W & W')=0.1091
Lena	Logo Image	Cropping 50%	Cropping 50%	Cropping 50%	Cropping 50%	PSNR=5.23(dB)	PSNR=32.64(dB)	

Figure 5.21: Effect of cropping attacks on Lena image in RWS-LBP-WM-LIP

Cover Image (512×512) (C)	Watermark (128 ×128)	watermarked image(CW1)	watermarked image(CW2 )	watermarked image(CW3)	watermarked image(CW4)	Extracted watermark	Extracted Cover Image	Experimental Results
								HCC (C & C) = 0.9981 HCC (W & W) = 0.9962 BER ( C & C) = 0.0003 BER ( W & W) = 0.0037
Lena	Logo Image	CopyMove 5%	No Attack	No Attack	No Attack	PSNR=27.06(dB)	PSNR=34.29(dB)	
								HCC (C & C) = 0.9963 HCC (W & W) = 0.9957 BER ( C & C) = 0.0011 BER ( W & W) = 0.0041
Lena	Logo Image	CopyMove 5%	CopyMove 5%	No Attack	No Attack	PSNR=21.77(dB)	PSNR=31.25(dB)	
								HCC (C & C) = 0.9945 HCC (W & W) = 0.9938 BER ( C & C) = 0.0016 BER ( W & W) = 0.0051
Lena	Logo Image	CopyMove 5%	CopyMove 5%	CopyMove 5%	No Attack	PSNR=20.36(dB)	PSNR=29.55(dB)	
								HCC (C & C) = 0.9927 HCC (W & W) = 0.9928 BER ( C & C) = 0.0021 BER ( W & W) = 0.0059
Lena	Logo Image	CopyMove 5%	CopyMove 5%	CopyMove 5%	CopyMove 5%	PSNR=19.52(dB)	PSNR=28.25(dB)	
								HCC (C & C) = 0.9965 HCC (W & W) = 0.9938 BER ( C & C) = 0.0009 BER ( W & W) = 0.0048
Lena	Logo Image	CopyMove 10%	No Attack	No Attack	No Attack	PSNR=26.91(dB)	PSNR=31.57(dB)	
								HCC(C & C) = 0.9933 HCC(W & W) = 0.7643 BER ( C & C) = 0.0732 BER ( W & W) = 0.0953
Lena	Logo Image	CopyMove 10%	CopyMove 10%	No Attack	No Attack	PSNR=18.91(dB)	PSNR=28.60(dB)	
								HCC (C & C) = 0.9902 HCC (W & W) = 0.9899 BER ( C & C) = 0.0021 BER ( W & W) = 0.0392
Lena	Logo Image	CopyMove 10%	CopyMove 10%	CopyMove 10%	No Attack	PSNR=18.12(dB)	PSNR=26.95(dB)	
								HCC (C & C) = 0.9869 HCC (W & W) = 0.9903 BER ( C & C) = 0.0036 BER ( W & W) = 0.0729
Lena	Logo Image	CopyMove 10%	CopyMove 10%	CopyMove 10%	CopyMove 10%	PSNR=17.39(dB)	PSNR=25.66(dB)	
								HCC (C & C) = 0.9854 HCC (W & W) = 0.9965 BER ( C & C) = 0.0011 BER ( W & W) = 0.0081
Lena	Logo Image	CopyMove 20%	No Attack	No Attack	No Attack	PSNR=25.12(dB)	PSNR=28.73(dB)	
								HCC (C & C) = 0.9873 HCC (W & W) = 0.9868 BER ( C & C) = 0.0025 BER ( W & W) = 0.0107
Lena	Logo Image	CopyMove 20%	CopyMove 20%	No Attack	No Attack	PSNR=16.90(dB)	PSNR=25.79(dB)	
								HCC (C & C) = 0.9810 HCC (W & W) = 0.9870 BER ( C & C) = 0.0038 BER ( W & W) = 0.0703
Lena	Logo Image	CopyMove 20%	CopyMove 20%	CopyMove 50%	No Attack	PSNR=16.13(dB)	PSNR=24.05(dB)	
								HCC (C & C) = 0.9750 HCC (W & W) = 0.9872 BER ( C & C) = 0.0052 BER ( W & W) = 0.1401
Lena	Logo Image	CopyMove 20%	CopyMove 20%	CopyMove 20%	CopyMove 20%	PSNR=15.72(dB)	PSNR=22.72(dB)	

Figure 5.22: Effect of copy-move forgery on Lena image in RWS-LBP-WM-LIP

Table 5.16: Comparison table in terms of computation time in RWS-LBP-WM-LIP

Schemes	Size of Image	Embedding time (sec)	Extraction time (sec)	Total time (sec)
Su et al. [78]	$512 \times 512$	0.5244	0.3701	0.8945
Verma et al. [91]	$512 \times 512$	0.5173	0.5989	1.1162
Parah et al. [65]	$512 \times 512$	0.59	0.0624	0.6524
RWS-LBP-WM-LIP	$512 \times 512$	0.315	0.199	0.514

### 5.3 Discussion

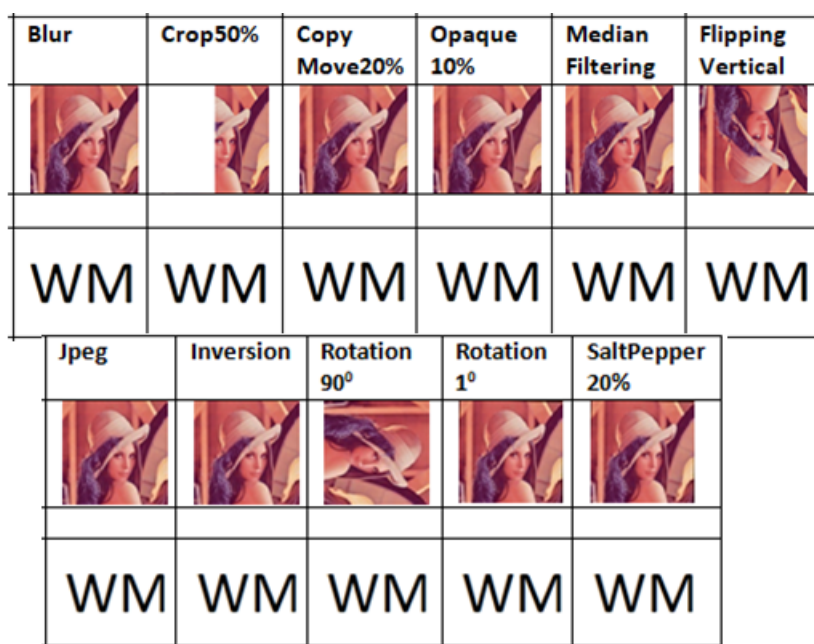
In this chapter, LBP operator and CA rule have been used to improve our scheme concerning tamper detection and tamper localization, shown in RWS-LBP-CA. Experimental outcomes of RWS-LBP-CA shows that the proposed method can resist seven cases and cover image can recover successfully from nine cases after performing ten special types of attacks depicted in Fig. 5.23. Moreover, the scheme can recover cover image in one more case than the previous.

Cover/Logo	Blur	Crop50%	Copy Move20%	Opaque 10%	Median Filtering	Flipping Vertical	Jpeg	Inversion
Localizati on								
Only detection								
	Rotation 90°	Rotation 1°	SaltPepper 20%	Cover image recovered (After localization) : 4 Cover image recovered: (Before localization) : 9 Watermark recovered : 7				

Figure 5.23: Effects of different types of attacks on Lena image for RWS-LBP-CA.

So there is a chance to increase robustness and capacity. A new watermarking scheme in sub-sample based interpolated image has been developed with the help of LIP, LBP operator and WM is shown in RWS-LBP-WM-LIP. Here, WM is used to increase embedding capacity and LBP operator is used to locating the tamper region. Moreover, LIP and LFSR is used to enhance security and achieve reversibility after tampering. It has been seen that the RWS-LBP-WM-LIP can resist all types of attacks and cover image can recover successfully from all cases after

performing ten special types of attacks depicted in Fig. 5.24. So the overall results are shown in



Cover image recovered: 10      Watermark recovered: 10

Figure 5.24: Effects of different types of attacks on Lena image for RWS-LBP-WM-LIP.

Table 5.17. Finally, the robustness against the various attacks can be achieved, and also a good

Table 5.17: Effects of 10 different types of attacks

Schemes	Image Recovered	Salt Pepper	Cropping	Copy move	Opaque	Median Filtering	Flipping (Vertical)	JPEG Compression	Blurring	Rotation	Inverntion
RWS-WM	CI	✓	✓	✓	✓	✓	✓	✗	✓	✓	✗
	W	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
RWS-CA	CI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
	W	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓
DRWS-LBP	CI	✓	✓	✓	✓	✓	✗	✓	✓	✓	✗
	W	✓	✓	✓	✓	✓	✗	✓	✓	✗	✗
RWS-LBP-HC	CI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
	W	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓
RWS-LBP-CA	CI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
	W	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓
RWS-LBP-WM-LIP	CI	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	W	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

trade-off among capacity, imperceptibility, and robustness have been made.



### 5.3.1 Salient Feature of this Chapter

- In this chapter, LBP operator and Cellular Automata have been used to detect and locate the tampered region of watermarked image.
- Here, shared secret key has been considered to enhance security of both the schemes. Shared secret position has been also used to enhance security. It has been updated for new block using  $\kappa_{i+1} = ((\kappa_i \times \omega) \bmod 7) + 1$ , where  $i= 1, 2, 3, \dots, N_B$ .  $N_B$  represents the number of block,  $\kappa_i$  is the shared secret position and  $\omega$  is the watermark embedding position. Moreover, cellular automata, LBP operator and LFSR stream cipher algorithm have been used to strengthen the security of the schemes.
- No original watermark information are directly embedded within the host image. Watermark bits are encrypted with the help of shared secret key. 4-bit watermark data is just embedded by changing one bit of the host image.
- The advantages of secret sharing has been achieved with the help of Lagrange Interpolation Polynomial in RWS-LBP-WM-LIP.
- High payload with good visual quality have been achieved in Cellular Automata based watermarking schemes.

