An Efficient Spatial Domain Based Image Watermarking Using Shell Based Pixel Selection

[1]Shubham Mathur, [2]Akshay Dhingra,[3]*Prabukumar M,
[4] Agilandeeswari L
School of Information Technology and Engineering
VIT University
Vellore, TN, India.
[1]shubham.mathur@engineer.com
[2]akshay300395@gmail.com
[3]mprabukumar@vit.ac.in
[4]agila.l@vit.ac.in
*Corresponding Author

Muralibabu K
Department of ECE
Lord Ayyappa Institute of Engineering and Technology
Kancheepuram, TN, India
mail2murali05@yahoo.co.in

***Abstract*— In this paper, we implemented a new algorithm in spatial domain with shell based pixel selection for watermark embedding and extraction. Here, the watermark is first converted into binary image by local thresholding and then converted into a logical matrix. Before embedding, each value of logical matrix is XOR-ed using a random 8-bit key to generate modified logical matrix. Next, the pixel of host image is selected by shell-based technique along row and column alternatively, starting from position (2, 2) and moving diagonally. To prevent duplicate selection of pixel two direct-address tables are maintained. Each pixel is sliced into red, green, blue and alpha components and bits from modified logical matrix are embedded into LSB of each component and finally an extraction key is generated. To detect tampering in an image, watermark is extracted using key and compared with original watermark. The proposed method is evaluated with benchmark dataset and we obtained a favorable result in terms of PSNR and BER. We reported the results of various kinds of image manipulation to assess the performance of the proposed method by drawing a comparative study of the original watermark and the watermark extracted from a manipulated image. Shell based pixel selection gives sensitivity and converting watermark to logical matrix and storing it in each component of pixel gives higher capacity than traditional methods.**

Keywords—logical matrix; shell-based; PSNR; BER; direct-address table; image manipulation.

I. INTRODUCTION

The spread, easy access and availability of internet, coupled with modern applications, like WhatsApp, Photoshop editors, Facebook, Google plus, snipping tools, and endless other applications have made sharing and manipulation of images a really easy task. With this given scenario, a method is needed that helps the owner identify the detection of manipulation in his copyrighted digital image. It is because once an image is manipulated it is not authenticate anymore. To provide image authentication, various forms of information hiding techniques are available namely, steganography [1, 2] and digital watermarking [3, 4, 5]. In this, steganography concentrates on sharing secret messages, where as, digital watermarking is a technique used to embed a digital data called watermark over the host image to protect it from the act of misusing [6]. Fragile watermarking is a form of digital watermarking helps in image authentication, tamper detection and verification of image integrity [7].

A fragile watermark is a watermark which is embedded in a host image, gets destroyed easily even on slight modification of watermarked image. This tampered watermark helps to identify that image has been edited, damaged or altered after it was marked [8]. The fragile watermarking techniques can be classified into image [9], audio[10] and video [11] based on the cover content used for watermarking. It is further classified according to their working domain as Spatial and Frequency. In spatial domain, the watermark data is embedded into pixel values of host image [12, 13]. While in frequency domain the watermark data is embedded in frequency components of the host image [14, 15].

The Chaos based spatial domain watermarking method developed by R.Munir [16] provides good security, high sensitivity but it has some key defects that algorithm has low capacity, it arises because to detect pixel level tampering it requires embedding in each pixel of host image, thus the size of watermark should be less than of host image. Another issue is, the generation of chaos map is time consuming as it is going to involve lot of mathematical calculation and the iterative nature of its truncate function to generate integer values are also time consuming. Due to all these demerits, we devised a new algorithm. It has three times more capacity than previous method as each pixel of host image could store three pixels of watermark. To match the sensitivity and security provided by chaos method, shell based pixel selection is used which provide high

978-1-5090-2029-4/16/$31.00 @2016 IEEE

randomness so the location and order of watermark pixels in host image is unpredictable.

The various forms of manipulations tested in this research include Gaussian blur, cropping, forgery, color inversion, jpeg compression, uniform monochromatic noise, Gaussian monochromatic noise, and sharpening. The metrics used to evaluate these forms of manipulations include PSNR and BER.

The rest of the paper is organized as follows: Section II discusses the proposed embedding and extraction algorithm, Experimental Results are presented in Section III and the conclusions are drawn in Section IV.

II. PROPOSED ALGORITHM

The proposed image watermarking is based on bit plane slicing, local thresholding, direct-address table, spatial domain fragile watermarking using modified logical matrix and shell based pixel selection for embedding and extraction. Before embedding, the watermark image is converted to binary image using local thresholding. In local thresholding, an intensity values of the local neighborhood of each pixel is statistically examined. The statistics that is used is the mean of the minimum and maximum values.

$$T = \frac{max+min}{2} \quad (1)$$

The binary image (BW) is converted into corresponding logical matrix (LM1). A random 8-bit key is generated (K1). Modifier use this key and works like an encrypter. It takes XOR (Exclusively OR) with logical matrix values to generate modified logical matrix (LM2).

For example:
Key=10110010

$$LM1 \oplus Key = LM2 \quad (2)$$

Table I. Logical matrix of binary watermark LM1

1	0	1	0	0
0	0	0	1	1
0	0	1	0	0
1	0	1	0	1
0	1	1	1	0

Table II. Modified logical matrix of binary watermark LM2

0	0	0	1	0
0	1	0	0	1
1	1	1	0	1
1	1	1	1	0
0	1	0	1	1

In watermark embedding the least significant bits (LSB) of pixels is used for embedding where the LSB's of the host image are replaced by the encrypted watermark. But instead of traditional embedding pattern a different embedding pattern is developed based upon direct-address table, shell based pixel selection and using bit plane slicing to get alpha, red, green, blue part`s LSB (Least Significant Byte). In embedding a randomly generate shell number (s) is stored in diagonal pixels starting from pixel at (2,2) and then an element is read along row at (x, y + j) position then along column at (x + i, y) position. Alternatively, the pixels are selected from row and column and the counters i, j are incremented by shell value (s). Then as counters exceed their limiting values i.e. height and width then the shell size is reduced and counters are reset. To prevent duplicate selection of pixels two linear arrays are maintained. After all the pixels along row and column are selected then next pixel (3, 3) is selected and same procedure is repeated until the image is embedded. This is done to increase the security and sensitivity of watermark. After embedding, a key for extraction is generated. The key consists of information about key (K1) for generating modified logical matrix, watermark image dimensions and information of number of digits in image dimension. Extraction process requires watermarked image (MI) and extraction key (K1). Using key, all necessary information is extracted and watermarked image is read in same manner as done in embedding process, it will generate logical matrix. Modifier takes key and logical matrix to generate watermark matrix which again is used to get binary watermark image. The logical matrix generation, embedding and extraction processes of watermarking are shown in Fig. 1, Fig. 2 and Fig. 3 respectively. The step by step process of each stage is discussed in the following sections:

A. Modified logical matrix generation

1. Generate 8-bit random key ‘K1’.

2. Use local thresholding to convert color image to binary image ‘BW’.

3. Generate logical matrix ‘LM1’ for corresponding value of binary image, i.e. 0 for 0 and 1 for 255.

4. Sequentially read bits from logical matrix ‘LM2’ and perform XOR with key ‘K1’ and store it in modified logical matrix ‘LM2’.

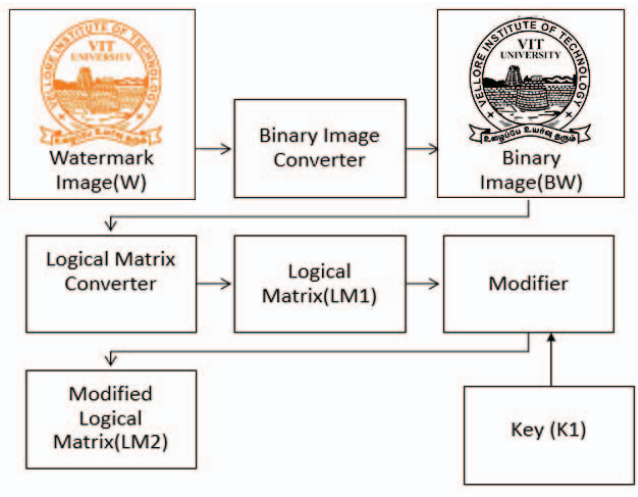


Fig. 1. Proposed Modified Logical matrix generation

B. Watermark embedding

1. Open cover image '*C*'. Its dimensions are *W*×*H*.
2. Create a blank image '*MI*' of same size of cover image.
3. Start reading cover image from pixel at (*x,y*)=(2,2).
   - 3.1. Generate a random shell number 's' between 1-5 and store in LSB of RGB components of that pixel and write that pixel to image '*MI*'.
   - 3.2. Maintain two one-dimensional array one of size W-x for row and another of size H-y for column and initialize them with 0.
     - 3.2.1. Initialize *i=1, j=1*.
     - 3.2.2. Read one pixel along row and then another pixel along column alternatively.
     - 3.2.3. While reading along row, read pixel at (*x, y + j*) check for corresponding index value, if it is 0 and if *y + j < width W*, if both are true then extract alpha, red, green, blue part`s LSB and modify them according to value read from modified logical matrix LM2.
     - 3.2.4. Write the modified pixel to image '*WI*'.
     - 3.2.5. Increment the counter *j* by *s*. *j=j + s*.
     - 3.2.6. While reading along column, read pixel at (*x + i, y*) check for corresponding index value, if it is 0 and if *x + i < height H*, if both are true then then extract alpha, red, green, blue part`s LSB and modify them according to value read from modified logical matrix LM2.
     - 3.2.7. Write the modified pixel to image '*WI*'.
     - 3.2.8. Increment the counter *i* by *s*. *i=i + s*.
     - 3.2.9. Repeat steps from 3.2.1 to 3.2.8 if statement 3.2.3 and 3.2.6 fails simultaneously then decrease shell number '*s*' by 1 and repeat steps from 3.2.1.
     - 3.2.10. If s becomes 0, increment *x* by *1* and *y* by *1*, i.e. *x=x+1, y=y+1*.
   - 3.3. Repeat steps from 3.1 till all bits from modified logical matrix LM2 is read.
4. Read first two columns of cover image and write them to image '*WI*'.
5. Read rest of the cover image where step 3 stops at some (*x, y*) pixel and write them to image '*WI*'.

C. Extraction key generation

1. Take 8-bit random key (K1) concatenate with width and height of binary watermark concatenate with information of number of digits in watermark width and height.
2. Example: key=12760867033. K1 = 127, width = 608, height = 670, number of digits in width = 3, number of digits in height = 3.

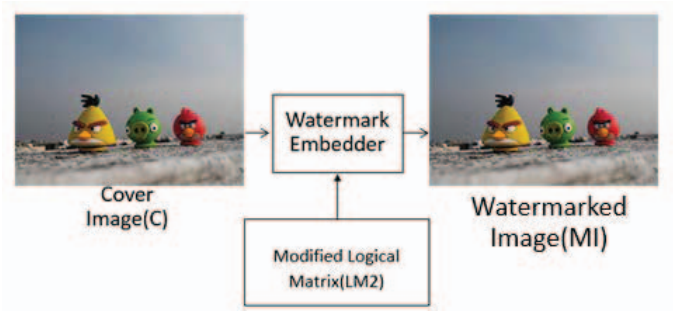


Fig. 2. Proposed Watermark embedding

D. Watermark Extraction

1. Read key entered by user.
2. Separate key K1, width '*fw*', height '*fh*' from key.
3. Open watermarked image *MI*. Its dimensions are *W*×*H*.
4. Create an empty matrix 'ELM1' of same size of *MI*, and initialize counter variables as *row=0, col=0*.
5. Start reading watermarked image from pixel at (*x,y*)=(2,2).
   - 5.1. Read LSB of RGB components of that pixel and generate 3-bit binary number and convert that number to integer and store it in '*s*', the shell number.
   - 5.2. Maintain two one-dimensional array one of size *W-x* for row and another of size *H-y* for column and initialize them with 0.
     - *5.2.1.* Initialize *i=1, j=1*.

5.2.2. Read one pixel along row and then another pixel along column alternatively.

5.2.3. While reading along row, read pixel at *(x, y + j)* check for corresponding index value, if it is 0 and if $y + j < width\ W$, if both are true then extract alpha, red, green, blue part`s LSB and write the bits read to logical matrix ELM1 using row and col counter.

5.2.4. Change row and col counter accordingly and keeping their value under *fw*, watermark width and *fh*, watermark height.

5.2.5. Increment the counter *j* by *s*. $j=j+s$.

5.2.6. While reading along column, read pixel at $(x + i, y)$ check for corresponding index value, if it is 0 and if $x + i < height\ H$, if both are true then then extract alpha, red, green, blue part`s LSB and write the bits read to logical matrix ELM1 using row and col counter.

5.2.7. Change row and col counter accordingly and keeping their value under *fw*, watermark width and *fh*, watermark height.

5.2.8. Increment the counter *i* by *s*. $i=i+s$.

5.2.9. Repeat steps from 5.2.1 to 5.2.8 if statement 5.2.3 and 5.2.6 fails simultaneously then decrease shell number 's' by 1 and repeat steps from 5.2.1.

5.2.10. If s becomes 0, increment *x* by *1* and *y* by *1*, i.e. $x=x+1$, $y=y+1$.

5.3. Repeat steps from 5.1 and stop if both $row==fh$, $col==fw$ becomes true.

6. Sequentially read bits from logical matrix 'ELM1' and perform XOR with key 'K1' and store it in watermark logical matrix 'ELM2'.

7. Create a blank binary image 'EBW' of size *fw* and *fh*. Store intensity values from corresponding value of watermark logical matrix 'ELM2', i.e. 0 for 0 and 255 for 1 to image EBW.

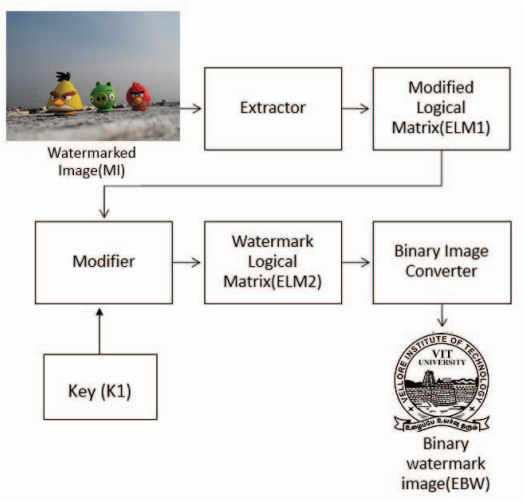


Fig. 3. Proposed Watermark extraction

III. Experimental Results

The performance of the proposed method is measured in terms of imperceptibility and robustness against various attacks namely, gaussian blur, cropping, forgery, color inversion, uniform and gaussian monochromatic noise, sharpening, jpeg compression and mosiac pixelate. The 20 sample images of same dimensions (see Fig. 4) and the watermark image as logo of VIT University (see Fig. 5) has been taken in order to evaluate the performance.

Fig. 4 Host Images (*angrybird, parrot, baby, deer, flower1, rhinos, sunflower, mixedfruit, myna, foreman, flower2, apple, tennis, baboon, tiffany, lighthouse, lena, peppers, and goldhill)*

Fig. 5 Watermark Image *(vitlogo)*

After embedding, the watermarks are extracted from the watermarked images, and the extracted watermarks are compared with the original watermarks to make conclusion that whether the watermarked image is altered. The proposed system is tested by introducing various attacks. These attacks or image manipulations are performed using Adobe Photoshop.

A. Metrics

The metrics that are used to measure the mentioned imperceptibility (PSNR) and robustness (NCC, BER) include:

A.1 Imperceptibility measure:

Peak signal-to-noise ratio (PSNR), is used as a common metric to evaluate the degradation caused by various attacks. It approximates human perception of reconstruction quality. In a noise-free m × n monochrome image *I* and its noisy approximation *K*, MSE is defined as,

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2 \quad (3)$$

The PSNR (in dB) is defined as,

$$PSNR = 10.\log_{10}\left(\frac{{MAX_I}^2}{MSE}\right) \quad (4)$$

Here, MAX_I is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255.

Greater than 38dB is treated as acceptable, where higher than this will be better. When noise is absent, the two images *I* and *K* are identical, and thus the MSE is zero. In this case, the PSNR becomes infinite [17].

A.2 Robustness measure:

A.2.1 Normalized Correlation Coefficient (NCC) is the correlation co-efficient can be computed using the following equation as,

$$NCC = \frac{\sum\left((OW_i - OW_m)(EW_i - EW_m)\right)}{\sqrt{\sum\left(OW_i - OW_m\right)^2}\sqrt{\sum\left(EW_i - EW_m\right)^2}} \quad (5)$$

The value of NCC varies between 0 and 1. If there is no error in the received message then the value of NCC will be close to 1, otherwise, close to 0. It acceptable limit is greater than 50% [17].

A.2.2 Bit error rate (BER) is the ratio of wrongly extracted watermark bits to the total number of watermark bits embedded. If there is no error in the received message then the bit error rate value will be 0, otherwise close to 1. [17]

$$BER(OW, EW) = \frac{\sum_{i=1}^{m}|OW_i - EW_i|}{m} \quad (6)$$

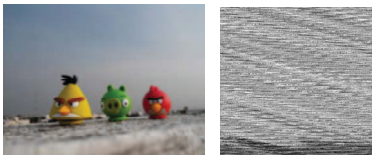
where, OW_i is the intensity of the ith pixel in image (original watermark), EW_i is the intensity of the ith pixel in image 2 (extracted watermark) and m is the total number of embedded watermark bits.

B. Attacks

The various image processing attacks used to validate the imperceptibility and robustness of the proposed watermarking systems are given below:

B.1 Gaussian blur

A Gaussian blur is achieved by using Gaussian function for blurring an image. The visual effect of this blurring technique is a smooth blur resembling that of viewing the image through a translucent screen [18]. It is used widely in graphics software, typically to reduce detail and reduce image noise. A Gaussian blur of radius 10 is applied on watermarked image Fig. 6 shows blurred watermarked image and its corresponding extracted watermark. Extracted watermark was completely destroyed showing that image is manipulated.



(a) (b)

Fig. 6. (a)Watermarked image after Gaussian Blur, (b) extracted watermark

B.2 Cropping

Cropping refers to the process of removing certain parts of a picture using an image editing software. 22.15% area of watermarked image is cropped. Fig. 7 shows cropped watermarked image and its corresponding damaged watermark showing that image is manipulated.

(a) (b)

Fig. 7. (a) Watermarked image after cropping, (b) extracted watermark

B.3 Forgery

Forgery refers to altering image by copying some part of image and pasting elsewhere on image itself. This gives a false impression that particular object is present at multiple positions in the image [19]. The red colored bird was added at the right side of image. Fig. 8 shows forged watermarked image and its extracted watermark. Watermark can be identifying from the extracted image but it is quite significantly visible that it is different from the original.

(a) (b)

Fig. 8. (a) Watermarked image after forgery, (b) extracted watermark

B.4 Color inversion

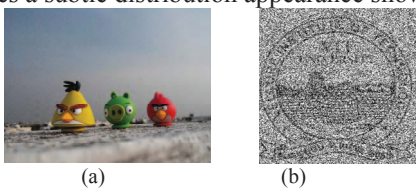
Inverting a color means finding the difference between the current value and 255 for each red, green, blue component of pixel. It gives old photographic negative effect to image. Fig. 9 shows color inverted watermarked image and its watermark. As it is changing each pixel value drastically, the changes in extracted watermark is quite significant, resulting in lowest PSNR value.

(a) (b)

Fig. 9. (a) Watermarked image after color inversion, (b) extracted watermark

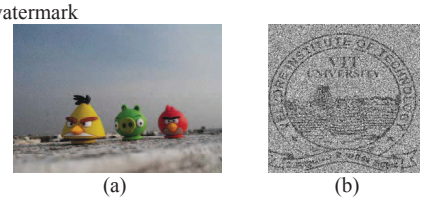
B.5 Uniform and Gaussian monochromatic noise

Noise is added randomly to an image. The pixels that are created have random level of color intensities. The uniform option creates a subtle distribution appearance shown in Fig. 10



(a) (b)

Fig. 10. (a) Watermarked image after insertion of uniform monochromatic noise, (b) extracted watermark



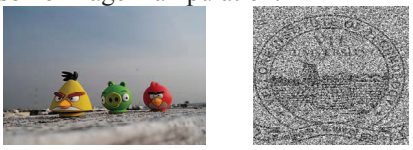
(a) (b)

Fig. 11. (a) Watermarked image after insertion of Gaussian monochromatic noise, (b) extracted watermark

and Fig.11 shows a speckled distribution look. Monochromatic applies the filter using the tones of the image without changing the colors [20]. 20% noise is added in both methods and a similar noisy watermark is extracted showing some presence of image alteration.

B.6 Sharpening

Image sharpening is an enhancement technique that highlights edges and fine details in an image. Image sharpening is achieved by high pass filter, which enhances high frequency components [21]. Fig. 12 shows sharpened image with a radius of 10 and its coressponding extracted watermark image showing presence of some image manipulation.

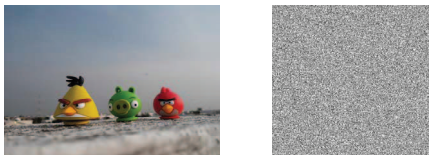


(a) (b)

Fig. 12. (a) Watermarked image after sharpening, (b) extracted watermark

B.7 JPEG Compression

In image compression, the image is stored in lesser bit as compared with its original size. The main aim of image compression is to decrease redundancy of image and eliminate that portion which is not visible to human eyes so that overall size is reduced. The jpeg technique is a lossy DCT (Discrete cosine transformation) based technique. In this process, image is separated into various different frequencies and while quantization unnecessary frequencies are discarded and the remaining necessary frequencies are used to retrieve the image in the reconstruction process [22]. The watermaked image is converted to jpeg image as shown in Fig. 13, which causes complete destruction of watermark showing only salt and pepper noise.

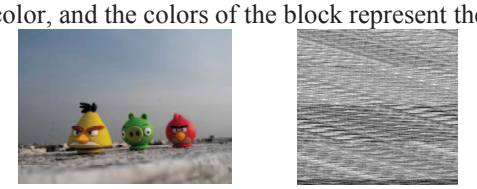


(a) (b)

Fig. 13. (a) Watermarked image after JPEG compression, (b) extracted watermark

B.8 Mosiac pixelate

In Mosaic Pixelate, a filter is used which sharply define a selection by clumping pixels of similar color values in cells. It clumps pixels into square blocks. The pixels in a given block are the same color, and the colors of the block represent the colors



(a) (b)

Fig. 14. (a) Watermarked image after mosaic inversion, (b) extracted watermark

in the selection. A mosaic pixelate of radius 10 is applied as shown in Fig. 14 and its extracted watermark showing that sensitivity characteristics of the method provide good security from minor image manipulations.

TABLE III. BER and PSNR values for various image processing attacks on watermarked images. BER and PSNR are calculated for extracted watermark.

S. No.	*Attacks*	*NCC*	*BER*	*PSNR*
1.	Gaussian Blur radius10	0.665	4.4027	51.693
2.	Crop 22.15% less pixel	0.783	3.9622	52.1513
3.	Forgery	0.925	1.7139	55.7908
4.	Color Inversion	0.987	0.075	49.3801
5.	Uniform monochromatic noise 20%	0.769	3.7675	52.3701
6.	Gaussian monochromatic noise 20%	0.778	3.7660	52.3719
7.	Sharp radius 10	0.725	3.8511	55.2274

8.	JPEG Compression	0.501	5.3126	51.7833
9.	Mosaic Pixelate radius 10	0.482	6.3894	51.7341

The Table. III shows the comparison of average PSNR and BER values for various attacks with the sample benchmarked images. Our approach is able to extract 66%, 78%, 92%, 98%, 76%, 77%, 72%, 50% and 48% on an average for various test images. From the above results, we conclude that the proposed watermarking algorithm is able to withstand various attacks except JPEG compression and mosaic pixelate. In particular, when a watermarked image is attacked by forgery and color inversion, the 92% and 98% of watermark extraction is achieved.

IV. CONCLUSION

In this paper, we have implemented a model of watermarking technique and tested it with various image manipulations. It has the ability to embed an invisible watermark into a spatial domain of an image. This technique yields watermarked images with high capacity, imperceptibility and high robustness. The algorithm provides high level of security by generating key which is used to extract the watermark later; also, the algorithm is able to randomize the location of the watermark by shell based pixel selection. On the other hand, an extraction algorithm is prepared and tested successfully.

REFERENCES

[1] Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang, “Detection of LSB Steganography via Sample Pair Analysis”, IEEE Transactions on Signal Processing, Vol. 51, No. 7, pp.1995-2007, July 2003.

[2] Y. K. Lee and L. H. Chen, "High capacity image steganographic model," Vision, Image and Signal Processing, IEEE Proceedings, Vol. 147, pp. 288-294, June 2000.

[3] J. Cox, J. Kilian, T. Leighton, and T. Shamoon, “Secure spread spectrum watermarking for multimedia,” IEEE Trans. Image Processing, Vol. 6, pp. 1673–1687, December 1997.

[4] Cox, IJ, Miller, ML and Bloom, Digital Watermarking, 2nd ed., San Francisco, CA, USA: Morgan Kaufmann Publisher, 2002.

[5] Ingemar, J. Cox, Matthew, L. M., Jeffrey, A. Bloom, Jassica Fridrich and Tan Kalker, Digital Watermarking and Steganography, 2nd ed., San Francisco, CA, USA: Morgan Kaufmann Publisher, 2008.

[6] H. C. Huang and W. C. Fang, "Authenticity Preservation with Histogram-Based Reversible Data Hiding and Quadtree Concepts," Sensors, Vol. 11, No. 10, pp. 9717-9731, October 2011.

[7] Monika Patel, Priti Srinivas Sajja and Ravi K. Sheth, “Analysis and Survey of Digital Watermarking Techniques”, Int. J. of Advanced Research in Computer Science and Software Engineering (IJARCSSE), Vol. 3, No.10, pp. 203-210, October 2013.

[8] Raja’ S. Alomari and Ahmed Al-Jaber “A Fragile Watermarking Algorithm for Content Authentication”, Int. J. of Computing & Information Sciences(IJCIS), Vol. 2, No. 1, pp. 27-37, April 2004.

[9] H. C. Huang and W. C. Fang, "Authenticity Preservation with Histogram-Based Reversible Data Hiding and Quadtree Concepts," Sensors, vol. 11, no. 10, pp. 9717-9731, October 2011.

[10] N. M. Ngo, M. Unoki, R. Miyauchi, and Y. Suzuki, "Data Hiding Scheme for Amplitude Modulation Radio Broadcasting Systems," Journal of Information Hiding and Multimedia Signal Processing, Vol. 5, No. 3, pp. 324-341, July 2014.

[11] Agilandeeswari L.and Ganesan K., “A bi-directional associative memory based multiple image watermarking on cover video”, Multimedia Tools and Applications, Vol. 75, No.12, pp. 7211 – 7256, June 2016.

[12] Mustafa Osman Ali, Elamir Abu Abaida Ali Osman, Rameshwar Row, “Invisible Digital Image Watermarking in Spatial Domain with Random Localization”,Int. J. of Engineering and Innovative Technology (IJEIT),Vol. 2, No. 5, pp. 227-231, November 2012.

[13] Irene G. Karybali, and Kostas Berberidis, “Efficient Spatial Image Watermarking via New Perceptual Masking and Blind Detection Schemes”, IEEE Trans. Information Forensics and security,Vol.1, No.2, pp. 256-274, June 2006.

[14] Agilandeeswari L. and Ganesan K., “An efficient hilbert and integer wavelet transform based video watermarking”, Journal of Engineering Science and Technology, Vol. 11, No. 3, pp. 327-345, March 2016.

[15] Agilandeeswari L.and Ganesan K.(2016), “An adaptive HVS based Video watermarking scheme for multiple watermarks using BAM neural networks and fuzzy inference system”, Expert Systems with Applications, Vol. 63, pp.412 – 434, November 2016.

[16] R. Munir, "A chaos-based fragile watermarking method in spatial domain for image authentication", 2015 Int. Seminar on Intelligent Technology and its Applications (ISITIA), pp. 227-232, 2015.

[17] Agilandeeswari L.and Ganesan K, “A robust color video watermarking scheme based on hybrid embedding techniques”, Multimedia Tools and Applications, Springer [online]. (28, August 2015). Available:http://link.springer.com/article/10.1007%2Fs11042-015-2789-9.

[18] Frederick M. Waltz and John W. V. Miller, “Efficient algorithm for Gaussian blur using finite-state machines”, Proc. SPIE 3521, Machine Vision Systems for Inspection and Metrology VII, SK21-7, November 1998.

[19] Hailing Huang, Weiqiang Guo and Yu Zhang, “Detection of Copy-Move Forgery in Digital Images Using SIFT Algorithm”, Pacific-Asia Workshop on Computational Intelligence and Industrial Application, 2008. PACIIA '08, Vol.2, pp. 272 – 276, 2008

[20] Brad Eigen, Micah Brown and Dan Livingston, *Essential Photoshop 6 for Web Professionals*,New Jersey: Prentice Hall PTR, 2001, pp.261

[21] Eunsung Lee, Sangjin Kim, Wonseok Kang, Doochun Seo, and Joonki Paik, “Contrast Enhancement Using Dominant Brightness Level Analysis and Adaptive Intensity Transformation for Remote Sensing Images, IEEE Geoscience And Remote Sensing Letters, pp. 1545 – 598, 2012.

[22] Priya Dixit and Mayank Dixit, “Study of JPEG Compression Techniques Using DCT”, Int. J. of Interdisciplinary Research and Innovations (IJIRI), Vol.1, No.1, pp. 32-35, Oct – Dec 2015.