Design and Implementation of Invisible and Visible Color Image Watermarking with Netbeans IDE

Baisa L. Gunjal
Amrutvahini College of Engineering, Sangamner,Ahmednagar, MS, India.

Suresh N. Mali
Principal, Singhgad Institute of Technology and Science, Narhe, Pune, MS, India

ABSTRACT

Aim of this paper is to present design and implementation of invisible and visible color image watermarking technique. The algorithms of individual category with implementations in Java Netbeans and test outcomes are presented to focus requirements in visible and invisible techniques. Though Matlab, Scilab, Octave are used for image processing, Net beans IDE is top of line Integrated Development Environment for Java development preferred for platform independent project development in industry and research work. The paper illustrates handling of visible text watermarking, visible logo watermarking and invisible watermarking with Java Netbeans implementation. The main objective of the paper is to focus on design and implementation with help of algorithms with test results presented here, instead of proving quality metrics of individual algorithm implemented in this paper.

.

General Terms

Netbeans IDE, visible and invisible, Java pixel representation.

Keywords

Modified LSB, Transform domain, security, payload.

1. INTRODUCTION

The advances in information technology and networking have brought revolution in reproduction and distribution of multimedia information. Copyright protection of intellectual properties have become critical and challenging issue. Digital image watermarking is one of the important solutions which include embedding owner's information or logos into digital image to be protected [1]. Major quality attributes of image watermarking techniques include robustness, imperceptibility, information hiding capacity and number of security levels achieved [2]. Robustness means, though watermarked image undergoes under any attack, watermark should not be disturbed significantly. Thus, robustness is a measure of immunity of watermark against attempts to image modification and manipulation like compression, filtering, rotation, scaling, resizing, cropping etc. Imperceptibility means perceived quality of cover image should be preserved in presence of watermark also. Quality watermarking technique should be capable to hide maximum watermark information in host image. Simultaneously, it should be difficult for attacker to detect the watermark. Implementation point of view, digital image watermarking techniques are classified as invisible and visible, spatial domain and transform domain techniques [1][3]. Invisible watermarking techniques includes extraction techniques to retrieve hidden copyright information from host media. It also required that watermarked must be resistant to different image attacks to ensure to extract hidden watermark after any addition, alteration deletion [4].

Visible watermarks are directly visible conveying ownership information on image media and required to be clearly visible after image attacks.Fig.1 shows general classification of watermarking techniques [5].

Image Watermarking Classification::		
1:	According to Embedding Domain:	i) Spatial Domain ii) Transform domain iii)Temporal domain
2:	According to Extractor:	i)Blind ii)Non blind
3:	According to Human perception:	i)Visible ii)Nonvisible - a)Robust b) Fragile
4:	According to application:	i)Source Based ii)application Based

Fig:1 General classification of watermarking

The rest of the paper is organized as follows: Section 2 focuses on details of related work of invisible and visible watermarking techniques. Section 3 focuses on design and implementation of visible and invisible color image watermarking techniques, while test results are presented in section 4 and conclusion is drawn in section 5.

2. RELATED WORK

Survey of existing visible and invisible watermarking techniques is presented here. Both visible and invisible watermarking techniques are implemented either in spatial; domain or transform domain. In spatial domain, watermark is embedded by directly modifying pixel values of cover image. Least Significant Bit insertion is implemented in spatial domain. Thetransform domain watermarking techniques, both cover image and watermark are taken into transform domain and watermark is spread out to entire cover image. Hence these techniques are more secure and more robust [4]. Various transforms like discrete Fourier transform (DFT), discrete Cosine transform (DCT), continuous Wavelet transform(CWT), discrete Wavelet transform (DWT), singular value decomposition(SVD) or combinations of different transform are applied to in transform domain to achieve robustness and perceptual transparency. Common method used for visible image watermarking is compress data of cover image and embed it with given payload into cover image [3][6][7].Another approach is to use spread spectrum method to spread the payload on cover image. One more approach is to implement lossless visible DCT based image watermarking technique [1]. Reversible visible watermarking and lossless recovery of original images is proposed in [8][9]. Visible watermarking for bitmap images is presented in [10], while removable visible watermarking for greyscale images is presented in [11]. Many existing visible watermarking techniques use binary logo embedding. But, some organizations and industries use color logos. Fewexisting invisible watermarking techniques are implemented in spatial domain while most of them are in transform domain. Transform domain image watermarking techniques are presented in [12-22].In [23], invisible Lossless watermarking technique for ROI based medical images is presented. Invisible watermarking techniques presented in [24-25] are resistant to various image attacks.

3. DESIGN AND IMPLEMENTATIONS

The algorithms presented here are implemented with Java with Net beans IDE downloaded from [26-27]. In java color pixel value is composed of four bytes: alpha value, red, green and blue planes(components) of image as shown in fig 1.

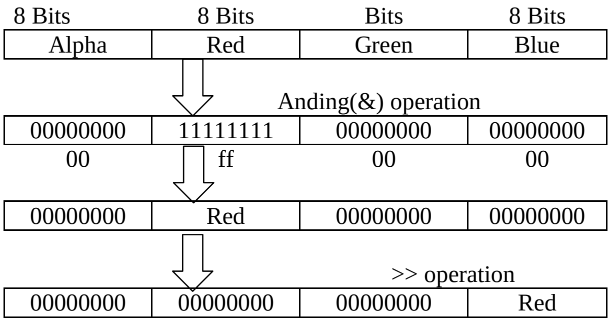


Fig 1: Representation of Java pixel

To perform any image operation on color image, red, green and blue components are separated. We can separate red, green and blue components by performing right shift operations. Once, these components are separated, we can process them as per requirement of application. Algorithm 1 shows separation of red, green and blue components and displays them separately.

Algorithm1 : Red, green, blue plane separation

```
Input:  BufferedImage  image1
Output:   static BufferedImageredp,greenp,bluep;
1:   Read  BufferedImage   image1 = "c:\\girl.jpg"
2:   redp = new BufferedImage(w1,h1,TYPE)
3:   Find  greenp and  bluep like step 2.
4:   for i=1 to h1         // h1 is height
5:       for j= 1 to w1     // w1 is width
6:   int pixel=image1.getRGB(i,j)
7:   int alpha = pixel & 0xff000000
8:   int red = (pixel >> 16) & 0xff
9:   int green = (pixel >> 8) & 0xff
10:  int blue= (pixel) & 0xff
11:  redp.setRGB(i,j, alpha | (red<< 16))
12:  greenp.setRGB(i,j, alpha | (green << 8) )
13:  bluep.setRGB(i,j, alpha| blue)
14:      end for
15:  end for
16:  String type="jpg"
17:  ImageIO.write(redp,type, new File("c:\\red.jpg"))
18:  Write “green.jpg” and “blue.jpg”  like step 17.
```

As per the requirement of application, red, green and blue components are processed and then they are combined with alpha values to form new image.

3.1 INVISIBLE WATERMARKING

Modified LSB in Spatial Domain Image Watermarking is presented as example invisible image watermarking technique. As modification, the experimentation is done to hide embed data in blue component with bit position 1 to bit position 5 separately. We can hide binary data in bit position 1 of blue component of pixel as follows:Let binary values of an image pixel are:

10100111 11101001 11001000 10100111
11001000 11101001 11001000 00100111

We will hide a binary value say 10010011 by changing only the LSB of the above mentioned image pixel value. The result will be as following:

1010011**1** 1110100**0** 1100100**0** 10100111
1100100**0** 1110100**0** 1100100**1** 0010011**1**

Algorithm2:Modified LSB Watermark Embedding

```
Input:   Cover_Image, binary: Input String
Output: Watermarked_image
1:   Read  Cover_Image =c:\\girl.jpg"
2:   Grab pixeksof  Cover_Image  into grabber.
3:   Formulate Input String using 4 to 7.
4:   StringBuilder binary = new StringBuilder();
5:    for x = 10 to  40
6:   StringBuilder b =
     binary.append(Integer.toBinaryString(x));
7:   end for
8:   Display ‘binary’ as string to be embedded.
9:   int pixels[]=new int[width*height]
     where w and h are width and height of
     Cover_Image
10:  Cover_pixel = (int[]) grabber.getPixels();
11:  for i=0 to Cover_pixel.length
12:  int c= Cover_pixel[i];
13:  int r= (c&0xff0000)>>16;
14:  int g= (c&0x00ff00)>>8;
15:  int b= (c&0x0000ff);
16:      if (i <watermark_string_length)
17:          if(binary.charAt(i) == '0')
18:                b = b & 254;
19:  Else
20:                b = b | 1;
21:      pixels[i] =((255<<24)|((r&0xff)<<16)|
         ((g&0xff)<<8)|(b&0xff));
22:     end if
23:  Else
24:       pixels[i]=((255<<24)|((r&0xff)<<16)|
     ((g&0xff)<<8)|(b&0xff));
25:  end for
26:  Create ‘Watermarked_image’ using pixel.
```

The process for modified LSB Watermark Embedding algorithm is shown in algorithm 2 and modified LSB Watermark Extraction process is shown in algorithm 3.

Algorithm3 :Modified LSB Watermark Extraction

```
Input:  :Watermarked_image
Output: sb :  Output Extracted  String
1:   Read Watermarked_image formed in embedding
2:   Grab pixels of  Watermarked_image in grabber1
3:   int pixels1[]=new int[width*height];
     where w and h are width and heights of
     Watermarked_image
4:   for i=0 to watermark_string_length-1
5:   int c= watermarkedArray[i];
6:   int r1= (c&0xff0000)>>16;
7:   int g1= (c&0x00ff00)>>8;
8:   int b1= (c&0x0000ff);
9:        String binString =
     Integer.toBinaryString(b1);
10:      Char s=binString.charAt
         ((binString.length()-1));
11:  sb.append(s);
12:  end for
13:  Display 'sb' as  Output Extracted  String
```

The variations for embedding binary data in various bit positions of blue components of pixels of color image are done with details shown in fig2.

Bit Position	AND	OR	Extraction of bit from blue component
1	254	1	length()-1
2	253	2	length()-2
3	251	4	length()-3
4	247	8	length()-4
5	239	16	length()-5

Fig 2: Embedding and extraction details for red(r), green(g) and blue(b) components

3.2 VISIBLE WATERMARKING

Visible text embedding process is shown in algorithm 4, while visible logo embedding process is presented in algorithm 5.

Algorithm4 : Visible Text Embedding

```
Input:  Input Image: I1
Output: Watermarked Image: I1
1:   Import required classes in Java Netbeans.
2:   Read BufferedImage I1 = "c:\\lake.jpg"
3:   Graphics2D g2d = (Graphics2D) I1.getGraphics();
4:   g2d.drawImage(I1, 0, 0, null);
5:   AlphaComposite alpha = AlphaComposite.getInstance
     (AlphaComposite.SRC_OVER, 0.5f);
6:   g2d.setComposite(alpha);
7:   g2d.setColor(Color.RED);
8:   g2d.setFont(new Font("Arial", Font.BOLD, 50));
9:   String watermark = "Copyright:: Pune";
10:  g2d.drawString(watermark, 90,400);
11:  g2d.dispose();
12:  ImageIO.write(I1, "jpg", new
     File("C:\\Watermarked_Image.jpg"));
```

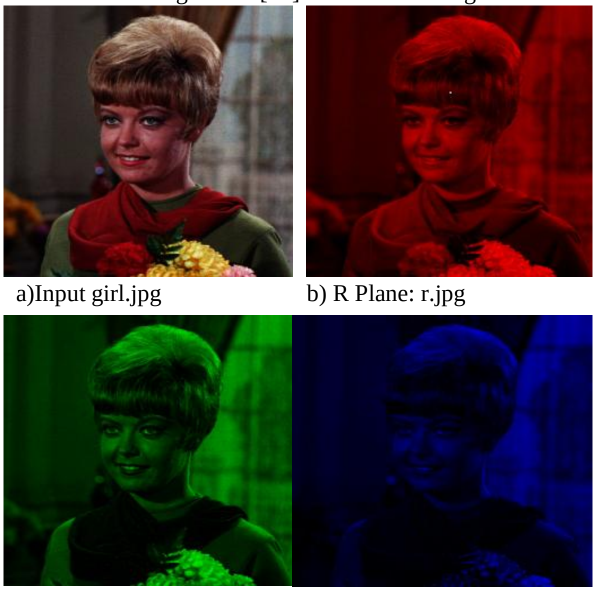
The visible logo embedding algorithm embeds logo at various positions: upper right position, bottom right position, centralposition, upper left position and bottom left position.

Algorithm5 : Visible Logo Embedding

```
Input:  Input Image: file1
Output: Watermarked Image: file1
1:   Import required classes in Java Netbeans.
2:   String file1 ="c:\\peppers2.jpg";
3:   Image Cover_Image = ImageIO.read(new File(file1));
4:    w and h are width and height of  Cover_Image
5:   BufferedImage image = new BufferedImage(w, h,
     BufferedImage.TYPE_INT_RGB);
6:   Graphics2D g = image.createGraphics();
7:   g.drawImage(Cover_Image, 0, 0, w, h, null);
8:   Read watermark.
9:   w1 and h1 are width and height of watermark
10:  //Embedding watermark at Upper Right
     g.drawImage(watermark, (w - w1 - X), Y,w1,h1, null);
11:  //Embedding watermark at Bottom right
     g.drawImage(watermark, (w - w1 - X), (h - h1 - Y),w1,
     h1, null);
12:  //Embedding watermark at Center
     g.drawImage(watermark, (w - w1 - X) / 2, (h - h1 - Y) /
     2, w1, h1, null);
13:  //Embedding watermark at Upper Left
     g.drawImage(watermark, X, Y, w1, h1, null);
14:  //Embedding watermark at Bottom Left
     g.drawImage(watermark, X, (h - h1 - Y), w1, h1, null);
15:  g.dispose();
16:  FileOutputStream output = new
     FileOutputStream(file1);
     //To use 18 and 19  import sun java classes in 17.
17:  import com.sun.image.codec.jpeg.JPEGCodec;
     import com.sun.image.codec.jpeg.JPEGImageEncoder;
18:  JPEGImageEncoder encoder =
     JPEGCodec.createJPEGEncoder(output);
19:  encoder.encode(image);  output.close();
```

4. TEST RESULTS

Online database given in [28] is used for testing.



a)Input girl.jpg b) R Plane: r.jpg

c) G Plane: g.jpg d) B Plane: b.jpg

Fig 3: Output of algorithm1: separation of RGB planes

Outcomes of invisible watermarking are shown in fig 3 and 4.

Fig. 3, shows separated red, green and blue components of image, while fig. 4 shows outcomes of modified LSB image watermarking with embedded binary string and extracted binary string by embedding binary string either with 1st , 2nd ,3rd, 4th, and 5th position bit substitution of blue plane of color image.

a)Input : girl.jpg

b)1st bit Substitution

c) 2nd bit Substitution

d)3rd bit Substitution

e)4th bit Substitution

f)5th bit Substitution

101010111100110111101111100001000110010100111 01
0010101101101011111000110011101011011111001110 1
1111011111100000100001100010100011100100100101 1
00110100111101000

g)Input Embedded String for all cases (b to f)

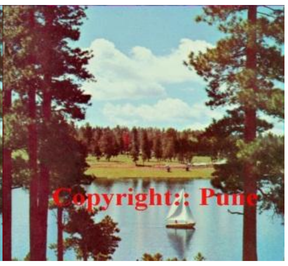
10101011110011011110111110000100011001010011101
00101011011010111110001100111010110111110011101
11110111111000001000011000101000111001001001011
00110100111101000

h)Output Extracted String for all cases (b to f)

Fig 4: Output of algorithm 2 and algorithm 3:bit substitution in blue componentof color image girl.jpg

In fig 5, outcomes of visible text embedding is shown. The user specified text is embedded in lake.jpg.Fig 6 and fig 7 shows embedding of visible logo at center, upper left, upper right, bottom left and bottom right positions.

a)Input: lake.jpg



b)Watermarked image

Fig 5: Output of algorithm 4: Visible Text embedding

a)Input Peppers.jpg

b)Water mark

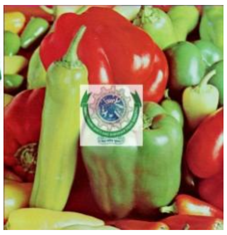

c)Watermarked image

Fig 6: Output of algorithm 5: visible logo embedding

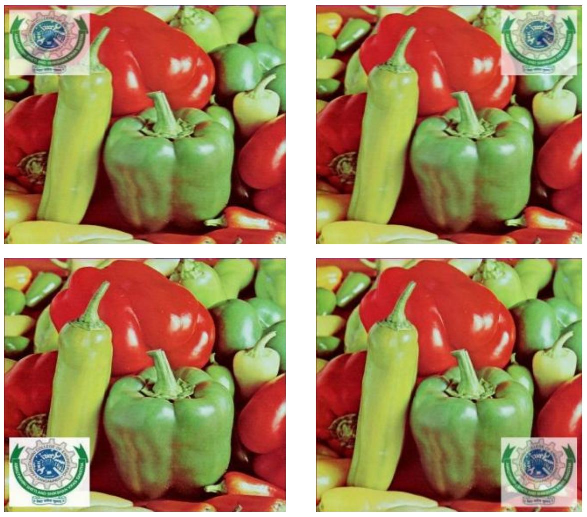


Fig 7: Output of algorithm 5:Insertion of visible logo by user's choice

The quality measurement benchmark of invisible technique 'modified LSB method' is computed first finding Mean Square Error(MSE) and using the same Peak Signal To Noise Ratio(PSNR) is computed. Mean Square Error between two images $f1(i,j)$ and $f2(i,j)$ is given by

$$MSE = \frac{1}{M*N}\sum_{i=1}^{M}\sum_{j=1}^{N}[f1(i,j) - f2(i,j)]^2 (1)$$

Where, M and N are the number of rows and columns both images, f1 (i, j) is pixel of original image, f2(i, j) is pixel values of watermarked image.Peak signal to noise ratio of two images f1 and f2 of size M x N is given by

$$PSNR(db) = 10log_{10}\frac{(Max_I)^2}{\frac{1}{M*N}\sum_{i=1}^{M}\sum_{j=1}^{N}[f1(i,j)-f2(i,j)]^2} (2)$$

Where, f1 (i, j) is pixel of original image. f2(i, j) is pixel values of watermarked image. Max_I is the maximum pixel value of image. Greater PSNR value implies that quality of image is better. It is found that changing a bit or bits of blue component can't be detected directly by human naked eyes. We found the maximum PSNR 65.15 for LSB substitution of blue component of color image and PSNR 56.76 for 5^{th} position bit substitution of blue component of color image girl.jpg for embedded input string shown in fig.4.

5. CONCLUSION

Here, design and implementation of invisible and visible digital color image watermarking is presented using Java Netbeans IDE.Though Matlab, Scilab, Octave are used for image processing, Net beans IDE is top of line Integrated Development Environment for Java development preferred for platform independent project development in industry and research work, which is used here. This work can be extended with visible color image design issues for lossless recovery of original image from watermarked image. The work can also be extended for invisible and visible techniques with Netbeans IDE in transform domain for color images as well as video watermarking.

6. ACKNOWLEDGMENTS

Thanks to Board of Colleges and University Development (BCUD), University of Pune for providing research grant.We are thankful to 'Amrutvahini College of Engineering, Sangamner', 'Singhgad Institute of Technology and Science, Pune', and 'Padmashree Dr. D.Y. Patil Institute of Engineering and Technology, Pune' for technical support during this research work.

.

7. REFERENCES

[1] Tsung-Yuan Liu, and Wen-Hsiang Tsai," Generic Lossless Visible Watermarking—A New Approach", IEEE Transactions on Image Processing, Vol. 19, No. 5, pp. 1224-1235, May 2010.

[2] Ehsan Nezhadarya,, Z. Jane Wang, and Rabab Kreidieh Ward," Robust Image Watermarking Based on Multiscale Gradient Direction Quantization", IEEE Transactions on Information Forensics and Security, Vol. 6, No. 4, pp.1200-1213, Dec. 2011.

[3] C. Aviles-Cruz,A. Ferreyra-Ramirez,J. J. Ocampo-Hidalgo,I. Vazquez-Alvarez, Structured-image retrieval invariant to rotation, scaling and translation",ACM Journal WSEAS Transactions on Systems,Vol 8, No.8, pp.1011-1020, Aug.2009.

[4] M. Kamran, and Muddassar Farooq," An Information-Preserving Watermarking Scheme for Right Protection of EMR Systems", IEEE Transactions on Knowledge and Data Engineering, Vol. 24, No. 11, pp. 1950-1962, Nov. 2012.

[5] Y. Hu, S. Kwong, and J. Huang, "An algorithm for removable visible watermarking," IEEE Trans. Circuits Syst. Video Technol., vol. 16, no.1, pp. 129–133, Jan. 2006.

[6] Y. Hu and B. Jeon, "Reversible visible watermarking and lossless recovery of original images," IEEE Trans. Circuits Syst. Video Techno. ,vol. 16, no. 11, pp. 1423–1429, Nov. 2006.

[7] Y. J. Cheng and W. H. Tsai, "A new method for copyright and integrity protection for bitmap images by removable visible watermarks and irremovable invisible watermarks," Int. Computer Symp. Workshop on Cryptology and Information Security, Hualien, Taiwan, R.O.C., Dec. 2002.

[8] P. M. Huang and W. H. Tsai, "Copyright protection and authentication of grayscale images by removable visible watermarking and invisible signal embedding techniques: A new approach," Conf. Computer Vision, Graphics and Image Processing, Kinmen, Taiwan, R.O.C., Aug. 2003.

[9] Y. Hu and S. Kwong ,"Wavelet domain adaptive visible watermarking," Electron. Lett., vol. 37, no. 20, pp. 1219–1220, Sep. 2001.

[10] Ying Yang , Xingming Sun , Hengfu Yang , Chang-Tsun Li , "A Contrast-Sensitive Reversible Visible Image Watermarking Technique", IEEE Transactions on Circuits and Systems for Video Technology, Vol:19 , Issue: 5, pp:656 – 667, May 2009.

[11] Liu Ping Feng, Liang Bin Zheng, Peng Cao ,"A DWT-DCT based blind watermarking algorithm for copyright protection", 3rd IEEE International Computer Science and Information Technology (ICCSIT), Beijing, China ,pp: 455 – 458,9-11 July 2010.

[12] Malay Kumar Kundu, SudebDas,"Lossless ROI Medical Image Watermarking Technique with Enhanced Security and High Payload Embedding", International conference on Pattern Recognition, pp: 1457-1460. , 2010.

[13] Xiaolong Li, Bin Yang, TieyongZeng, "Efficient Reversible Watermarking Based on Adaptive Prediction-Error Expansion and Pixel Selection", IEEE Transaction on Image Processing, Vol: 20, No: 12, pp.:3524-3532,Dec. 2011.

[14] Wei-Min Yang, Zheng Jin, "A Watermarking Algorithm based on Wavelet and Cosine Transform for color images", IEEE First international workshop on Education Technology and Computer ", 2009.

[15] Fabrizio Guerrini, Masahiro Okuda, Nicola Adami, and Riccardo Leonardi," High Dynamic Range Image Watermarking Robust Against Tone-Mapping Operators", IEEE Transactions on Information Forensics and Security, Vol. 6, No. 2, pp.283-295, Jun. 2011.

[16] Aranzazu Jurio, Miguel Pagola, Mikel Galar, Carlos Lopez-Molina, Daniel Paternain A Comparison Study of Different Color Spaces in Clustering Based Image Segmentation, Springer : Communication in Computer and Information Science, 81, pp.532-541, 2010.

[17] Roumen Kountchev, Roumiana Kountcheva, Image Color Space Transform with Enhanced KLT, Springer-New Advances in Intelligent Decision TechnologiesStudies in Computational Intelligence,199, pp. 171-182,2009.

[18] Fan Zhang, Wenyu Liu, Weisi Lin, and King NgiNgan,"Spread Spectrum Image Watermarking Based on Perceptual Quality Metric", IEEE Transactions on Image Processing, Vol. 20, No. 11, pp. 3207-3218, Nov. 2011.

[19] Dr. H. B. Kekre, Tanuja K. Sarode, Sudeep D. Thepade, Pallavi N. Halarnkar, Kekre's fast codebook generation

in VQ with various color spaces for colorization of gray scale images,Springer-Thinkquest ,102-108, 2011.

[20] D. Kundur and D. Hatzinakos, “Diversity and attack characterization for improved robust watermarking,” IEEE Trans. Signal Process., vol. 49, no. 10, pp. 2383–2396, Oct. 2001

[21] M. C. Hanumantharaju, G. R. Vishalakshi, Srinivas Halvi, S. B. Satish, A Novel FPGA Based Reconfigurable Architecture for Image Color Space Conversion, Springer :Communication in Computer and Information Science, 270, pp.292-301,2012.

[22] C. Deng, X. Gao, X. Li, and D. Tao, “Local histogram based geometric invariant image watermarking,” Signal Process., vol. 90, no. 12, pp.3256–3264, Dec. 2010.

[23] Lihong Cui and Wenguo Li, ”Adaptive Multiwavelet-Based Watermarking Through JPW Masking”, IEEE Transactions on Image Processing, Vol. 20, No. 4, pp. 1047-1060, Apr. 2011.

[24] Han-Min Tsai, Long-Wen Chang, “Secure reversible visible image watermarking with authentication”, ACM Journal: Image Communication, Vol. 25, Issue 1, pp.10-17, Jan., 2010.

[25] I.Cox, M. Miller, J. Bloom, J Fridrich and T.Kalker, “Digital Watermarking and Stegnography”, 2nd ed. San Francisco, CA. Morgan Kaufmann, 2007.

[26] http://www.netbeans.org/.

[27] www.java2s.com

[28] “Stir Mark Image Database 2.0.” [Online]. Available:http://www.petitcolas.net/fabien/watermarking/image_database/index.html