

STEGANOGRAPHY USING BPCS TO THE INTEGER WAVELET TRANSFORMED IMAGE

Ms. K. Ramani
Research Scholar,
JNTU, Hyderabad

Dr. E. V. Prasad
Vice-Principal
SVUCE, Tirupati

Dr. S. Varadarajan
Associate Professor
JNTUCE, Kakinada

ABSTRACT

Digital steganography exploits the use of a host data to hide a piece of information in such a way it is imperceptible to a human observer. In this paper we propose an image steganography system, in which the data hiding (embedding) is realized in bit planes of subband wavelets coefficients obtained by using the Integer Wavelet Transform (IWT). To increase data hiding capacity while keeping the imperceptibility of the hidden data, the replaceable IWT coefficient areas are defined by a complexity measure used in the Bit-Plane Complexity Segmentation Steganography (BPCS). The proposed system shows a high data hiding capacity.

KEYWORDS: Steganography, Bit Plane Complexity Segmentation, Integer Wavelet Transform, Discrete Wavelet Transform

1. INTRODUCTION

The information communicated comes in numerous forms and is used in many applications. In a large number of these applications, it is desired that the communication to be done in secrete. Such secrete communication ranges from the obvious cases of bank transfers, corporate communications, and credit card purchases, on down to a large percentage of everyday email. Steganography is the ancient art of embedding a secret message into a seemingly harmless message. Most of the newer applications use steganography like a watermark, to protect a copy right on information. The forms of steganography vary, but unsurprisingly, innocuous spam messages are turning up more often containing embedded text. A new transform domain technique for embedding the secrete information in the integer wavelet transformed cover image is proposed here.

All of the traditional steganographic techniques have limited information-hiding capacity. They can hide only 10% (or less) of the data amounts

of the vessel. This is because the principle of those techniques was either to replace a special part of the frequency components of the vessel image, or to replace all the least significant bits of a multivalued image with the secret information. Our new steganography uses an image as the vessel data, and we embed secret information in the bit-planes of the vessel[1]. The information hiding capacity of a true color image is around 50%. We can replace all of the “noise-like” regions in the bit-planes of the vessel image with secret data without deteriorating the image quality, which is termed as “BPCS-Steganography,” which stands for Bit-Plane Complexity Segmentation Steganography.

2. PROBLEM ANALYSIS

In our paper we propose a lossless data hiding method using IWT and BPCS, in which image data are decomposed by IWT and each bit plane of the sub-bands segmented in 8x8 blocks. All blocks are analyzed by complexity measures to determine which blocks will be replaced by secret message. The complexity measurement used in the proposed system is same one in the

BPCS method. The proposed system can be recovered the hidden message in lossless manner if the communication channel is ideal. A wavelet transform that maps integers to integers is the S-transform. In this large amounts of data can be compressed as smaller one by using IWT method, without data loss. This system is used to produce same quality of image while compressing and embedding the image, there will be no change in the quality of image and can be retained the same as the first. By using BPCS (Bit Plane Complexity Segmentation) the embedding process has been done here, so the security is very high for the embedded image.

3. INTEGER WAVELET TRANSFORM

Integer to integer wavelet transforms maps an integer data set into another integer data set. This transform is perfectly invertible and yield exactly the original data set. A one dimensional discrete wavelet transform is a repeated filter bank algorithm [1]. The reconstruction involves a convolution with the syntheses filters and the results of these convolutions are added. In two dimensions, we first apply one step of the one dimensional transform to all rows. Then, we repeat the same for all columns. In the next step, we proceed with the coefficients that result from a convolution in both directions. As shown in figure 1, these steps result in four classes of coefficients: the (HH) coefficients represent diagonal features of the image, whereas (HG and GH) reflect vertical and horizontal information. At the coarsest level, we also keep low pass coefficients (LL). We can do the same decomposition on the LL quadrant up to $\log_2(\min(\text{height}, \text{width}))$.

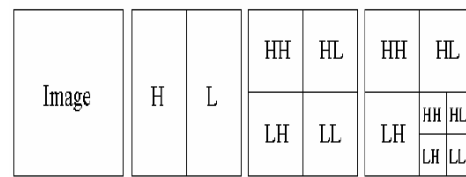


Figure 1: A Two dimensional wavelet Transform

Since the discrete wavelet transform allows independent processing of the resulting components without significant perceptible interaction between them, hence it is expected to make the process of imperceptible embedding more effective. However, the used wavelet filters have floating point coefficients. Thus, when the input data consist of sequences of integers (as in the case for images), the resulting filtered outputs no longer consist of integers, which doesn't allow perfect reconstruction of the original image. However, with the introduction of Wavelet transforms that map integers to integers we are able to characterize the output completely with integers.

One example of wavelet transforms that map integers to integers is the *S-transform*. Its smooth (*s*) and detail (*d*) outputs for an index *n* are given in (1a) and (1b) respectively (the smooth and the detail outputs are the results of the application of the high-pass and the low-pass filters respectively). The *S-transform* is reversible and its inverse is given in equations (2a) and (2b).

$$S(n) = [X(2n) + X(2n+1)]/2 \quad \dots (1a)$$

$$d(n) = X(2n) - X(2n+1) \quad \dots (1b)$$

$$X(2n) = s(n) + [(d(n) + 1)/2] \quad \dots (2a)$$

$$X(2n + 1) = s(n) - d(n)/2 \quad \dots (2b)$$

However, these equations should be in 2D in order to be applied on images. In this section, we will define the construction of the 2D *S-transform*. Suppose that the original image (*I*) is *Y* pixels wide and *X* pixels high. Denote the color shade level of pixels located at position *i* and *j* by

$I_{i,j}$. Generally, the 2D S -transform can be computed for an image using equations (3a), (3b), (3c), and (3d). Of course the transform is reversible, i.e., we can exactly recover the original image pixels from the computed transform coefficients. The inverse is given in equations (4a), (4b), (4c), and (4d). The transform results in four classes of coefficients: (A)the low pass coefficients,(H) coefficients represent horizontal features of the image, (V) and (D) reflect vertical and diagonal information respectively. During the transform we ignore any odd pixels on the borders.

$$A_{i,j} = (I_{2i,2j} + I_{2i+1,2j}) / 2 \quad \dots (3a)$$

$$H_{i,j} = I_{2i,2j+1} - I_{2i,2j} \quad \dots (3b)$$

$$V_{i,j} = I_{2i+1,2j} - I_{2i,2j} \quad \dots (3c)$$

$$D_{i,j} = I_{2i+1,2j+1} - I_{2i,2j} \quad \dots (3d)$$

$$I_{2i,2j} = A_{i,j} - [H_{i,j} / 2] \quad \dots (4a)$$

$$I_{2i,2j+1} = A_{i,j} + [H_{i,j} + 1] / 2 \quad \dots (4b)$$

$$I_{2i+1,2j} = I_{2i,2j+1} + V_{i,j} - H_{i,j} \quad \dots (4c)$$

$$I_{2i+1,2j+1} = I_{2i+1,2j} + D_{i,j} - V_{i,j} \quad \dots (4d)$$

Where, $1 \leq i \leq X/2, 1 \leq j \leq Y/2$

4. BIT PLANE COMPLEXITY SEGMENTATION

For a true 24-bit bitmap image, each of the RGB components takes one byte of memory. Each RGB component value ranges from zero (0) to 255 where zero represents darkest shade of the color and 255 represents brightest shade of this color. All other colors can be generated with the combinations of these ranges. A 4 by 4 sample image is given below [Fig 2]. Each pixel is a combination of red, green and blue values. Their integer values are given in Table 1.



Figure2: Test Image (4x4)

Table1: RGB Values

Column	R	G	B
0	175	247	09
	167	225	30
	164	217	38
	155	197	29
1	09	223	247
	30	206	225
	38	199	217
	33	169	184
2	80	09	247
	80	30	255
	91	38	217
	85	35	201
3	202	69	252
	190	32	251
	165	04	225
	134	03	184

This image has total 16 pixels and each pixel has three components having one-byte for each component, therefore the total size of the image is 48 bytes.

BPCS data hiding process

1. Bit plane decomposition.
2. Block segmentation.
3. Complexity measurement of each block.
4. Complexity measurement of secret message.
5. Replace complex image-data block to message block.

Bit plane decomposition

Alternatively the value of each RGB component can be represented in binary format as shown the Table 2.

Table 2: Binary representation of RGB values

Pixel	R	G	B
0	10101111	11110111	00001001
	10100111	11100001	00011110
	10100100	11011001	00100110
	10011011	11000101	00011101
1	00000001	11011111	11110111
	00011110	11001110	11100001
	00100110	11000111	11011001
	00100001	10101001	10111000
2	01010000	00001001	11110111
	01010000	00011110	11100001
	01011011	01011011	11011001
	01010101	00100011	11001001
3	11001010	01000101	11111100
	10111110	00100000	11111011
	10100101	00000100	11100001
	10000110	00000011	10111000

Following steps are followed for the constructions of the binary planes.

Step1: Formation of Channel Matrix

The selected channel (R in this case) is picked from all the pixels of the image. The channel matrix contains the N elements where N is the total number of pixels in the image.

Table 3: ‘R’ Channel matrix

R0	R1	R2	R3
10101111	00000001	01010000	11001010
10100111	00011110	01010000	10111110
10100100	00100110	01011011	10100101
10011011	00100001	01010101	10000110

Step 2: Get Corresponding Bits

In next step we get the corresponding *ith* bits from each of the channel to construct a plane. These bits are picked out using the same sequence in which the channel itself is allocated in the image. The height and width of a binary plane (as there are only 1’s and 0’s in plane) is the same as the height and width of the original image (4x4). Using this fact, we can easily calculate the total number of bits in any of the plane.

Step 3: Formation of ‘N’ Binary Planes

Applying the same procedure described in the step 2, the following N planes are constructed. N is the number of bits per RGB component.

Table 4: Planes extracted from ‘R’ channel

Plane 1				Plane 2			
1	0	0	1	0	0	1	1
1	0	0	1	0	0	1	0
1	0	0	1	0	0	0	0
1	0	0	1	0	0	1	0
Plane 3				Plane 4			
1	0	0	0	0	0	1	0
1	0	0	1	0	1	1	1
1	1	1	1	0	0	0	0
0	0	1	0	1	0	1	0
Plane 5				Plane 6			
1	1	0	1	1	0	0	0
0	1	0	1	1	1	0	1
0	0	1	0	1	1	0	1
1	0	0	0	0	0	1	1
Plane 7				Plane 8			
1	0	0	1	1	1	0	0
1	1	0	1	1	0	0	0
0	1	1	0	0	0	1	1
1	0	0	1	1	1	1	0

Binary pixel blocks that form bit-planes is as shown in Fig.3.

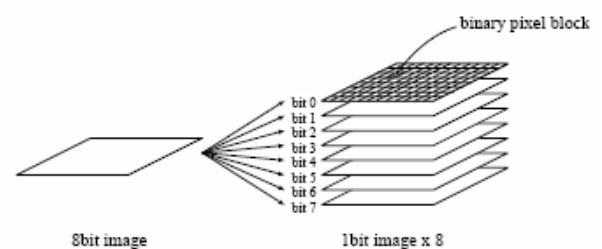


Figure 3: Binary pixel blocks on bit-planes

Block segmentation

Each bit plane is segmented to blocks of 8x8 binary data.

Complexity measurement of each block

In this work the method of steganography makes use of the more complex regions of an image to embed data. There is no standard definition of image complexity. Kawaguchi [1] discussed this problem in connection with the image thresholding problem. We adopted a black and white border image complexity.

We define a maximum value from 0 to 1 in each channel. The planes having complexity value less are considered informative and planes having more complexity than threshold are considered noisy. Now planes can be replaced with message to be embedded in the image. The threshold value to determine if a block is complex or not is given by

$$Th = 0.3C_{max}$$

Where C_{max} is the maximum complexity value in each channel

Complexity measurement of secret message:

Each 8 letters form a block of message, and complexity measurement is applied to each message block.

Conjugation of a binary image

Let P be an 8X8 size black-and-white image with black as the foreground area and white as the background area. We introduce two checkerboard patterns W_c and B_c , where W_c has a white pixel at the upper-left position, and B_c is its complement, i.e., the upper-left pixel is black. We regard black and white pixels as having a logical value of “1” and “0”, respectively.

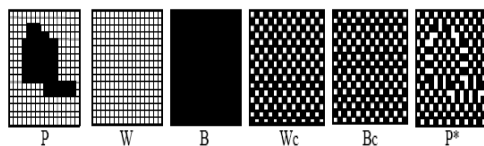


Figure 8: Illustration of each binary pattern

P is interpreted as follows. Pixels in the foreground area have the B pattern, while pixels in the background area have the W pattern. Now we define P^* as the conjugate of P.

The most important property about conjugation is the following.

Let $\alpha(P)$ be the complexity of a given image P, then we have, $\alpha(P^*) = 1 - \alpha(P)$.

The complexity value of P^* is always symmetrical against P regarding $\alpha = 0.5$. For example, if P has a complexity of 0.7, then P^* has a complexity of 0.3.

Replace complex image-data block to message block

All complex image-data blocks determined by step 3 are replaced by message blocks.

BPCS hidden data extraction process

- Bit plane decomposition.
- Block segmentation.

Complexity measurement of each block.

Conjugation map extraction

- The first complex blocks are corresponded to the conjugation map.

Hidden data extraction

- The complex blocks are extracted and conjugated if necessary to recover the secret message.

Embedding

- a). Transformed image is decomposed into bit-planes.
- b). Each plane is segmented into 8X8 blocks and complexity is measured for each block.
- c). Threshold value is chosen to determine whether the block is complex or non-complex.
- d). Each 8 letters form a block of message, and complexity is measured for each message block. If a message block is determined as no-complex block, the message block is conjugated.
 - i. Conjugation map is constructed from conjugated blocks.
 - ii. Complex image blocks are replaced by message blocks.

Extracting

- a). Transformed image is decomposed into bit-planes.
- b). Each plane is segmented into 8X8 blocks and complexity is measured for each block.
- c). Threshold value is chosen to determine whether the block is complex or non-complex.
- d). Secret message is extracted from the complex blocks, blocks are conjugated if necessary based on conjugation map information.

V. Experimental Results

Level 1

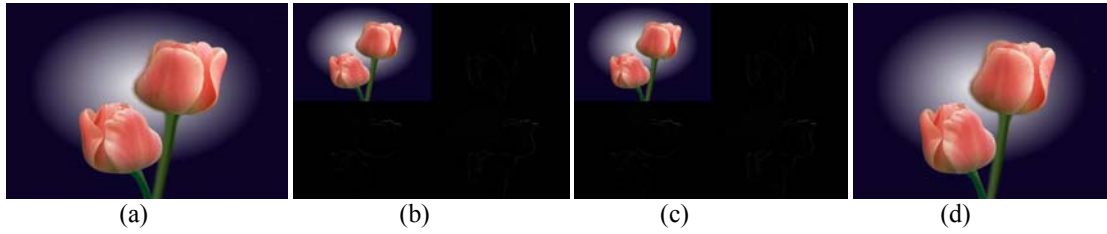


Figure 5(a): Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image

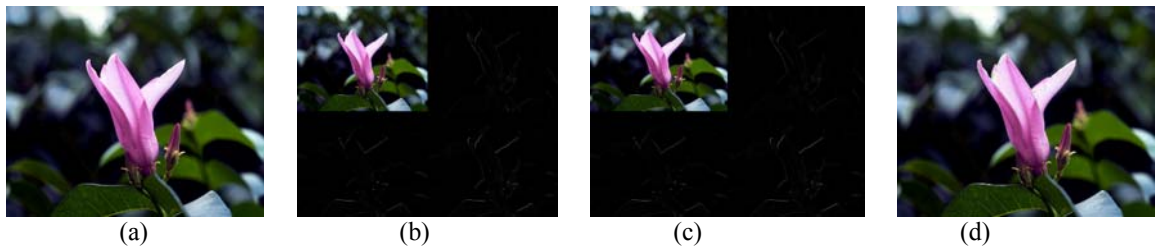


Figure 5(b): Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image



Figure 5(c) : Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image

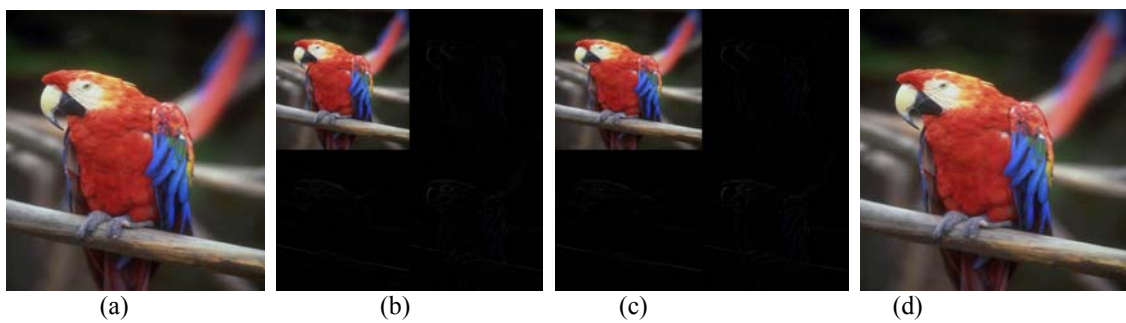


Figure 5(d): Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image

Level 2

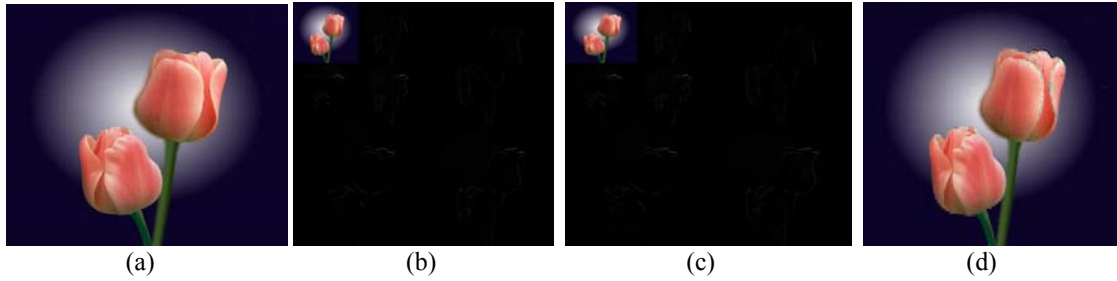


Figure 6(a): Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image

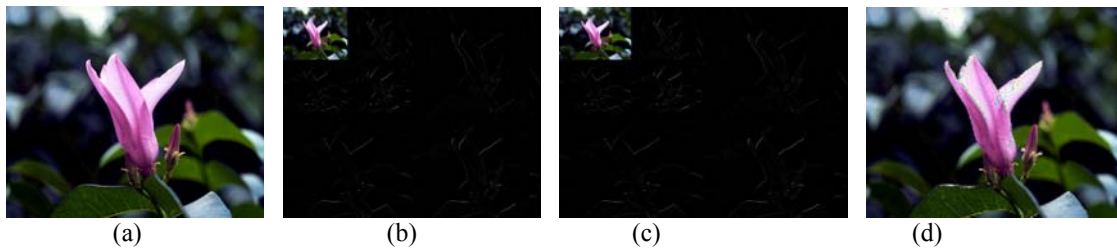


Figure 6(b): Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image

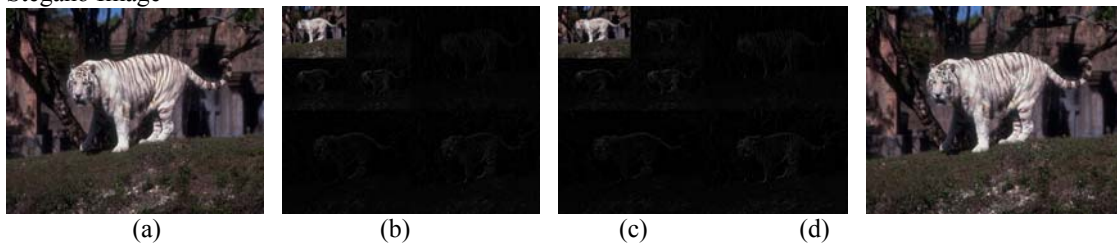


Figure 6(c): Embedding process (a) Cover Image (b) IWT Image (c) WaterMark Image before IIWT (d) Stegano Image

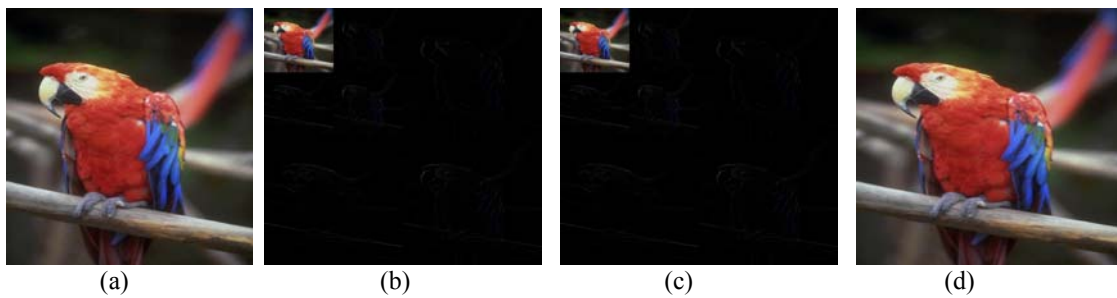


Figure 6(d): Embedding process (a) Cover Image (b) IWT Image (c) Water Mark Image before IIWT (d) Stegano Image

Image	Level 1			Level 2		
	PSNR (dB)	Maximum Capacity (bpp)	Bit Error Rate	PSNR (dB)	Maximum Capacity (bpp)	Bit Error Rate
Tulips	41.271	6.129	0.1282	39.881	5.623	0.6748
Lotus	39.410	8.903	0.3671	36.667	8.541	0.4345
Tiger	13.472	13.472	0.6005	43.593	13.460	0.6424
Parrot	48.765	9.444	0.2389	45.958	8.806	0.3374

Table 5: PSNR and BER Comparison for Different Images

Table 6: An example of recovered messages for different images

Image	Original	Level 1	Level 2
Tulips	<p>Rather than applying a classic compression coding scheme to estimate the number of bits needed to represent the wavelet result, a simple total for the number of bits is calculated. The total number of bits needed for to represent the wavelet result is the sum of the bit widths for each element. For a real compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of bits per value. A header for the subsequence, perhaps with a count and a width indicator would proceed the sequence. These headers and the common value widths in a sequence increase the size of the compression result.</p>	<p>Rather than applying a classic compression coding scheme to estimate the number of bits needed to represent the wavelet result, a simple total for the number of bits is calculated. The total number of bits needed for to represent the wavelet result is the sum of the bit widths for each element. For a real compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of bits per value. A header for the subsequence, perhaps with a count and a width indicator would proceed the sequence. These headers and the common value widths in a sequence increase the size of the compression result.</p>	<p>Rather than applying a classic compression coding scheme to estimate the number of bits needed to represent the wavelet result, a simple total for the number of bits is calculated. The total number of bits needed for to represent the wavelet result is the sum of the bit widths for each element. For a real compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of bits per value. A header for the subsequence, perhaps with a count and a width indicator would proceed the sequence. These headers and the common value widths in a sequence increase the size of the compression result.</p>
Lotus		<p>Aã^t_n`eáp`~c3óying añfgj<_yðÐ“ompres3ion ciehog scheme to estiauae"4èe(gqi¥¥R_of bùòs oeee`\$p!_è_YE_ã_vÀ`xÖ*_îðð lfv_CCA,AIIað`ájNž- t`ô`_E-or tjg"lteÔlr7JÆ`B½ð_7j`#c~l_×R` "{šWöe tot!! fuocdr'id Bit3_necfed for to represent the wavelet result is the sum of the bit widths for each element. For a real</p>	<p>ç<ø#_ÉTddEÄ tfo9)& ll aà lasslj;4Mo0öa[o9nm!ciginêZE ð_y€Öim«7C jiAüC_the numbdp"C°%+Ó8cld û¼5d`n úq czYc/`THâ°_m`-~`t pml%`#7a rhM0-¥ totad 0dL`0ðhã ntm;dò of,sll •fp ãas\$u[atED_The tnu`m #mm[ê_b@NÆ#) "a£ãN Eedãfç \{q#to bapresent the wavelet result is the sum of the bit widths for each element. For a real</p>

		<p>compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of blh74b §ðœ;e* Un ±DiÎèyE%2à_μ¾ i%_CuB²Ñ`÷% Î ã,yÍÍ _èÉpsS-[_5h _bnqhi€PG@({_ê_wo _ýi É_TgΠU)*ª@VIHX:@€Í ª» _mqμ.òCe#Qthetm}Π_Ad\$ç=M -p_yŽ®\Co±_ò½6áí,±± hdö 3;jiOll ÖÄ Í~½iÉCè Ä/jd tō_¼'ÄT,*G- %o“Ska³ü0Fãéçz/wPSΠowC m 4,aQ½\$Ūl-Ö_-I€HP</p>	<p>compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of bBÅaG HB{_F>Π]3áð_CΠ ...P~vwŠ[øh °;³3d ZÁΠ °c 6Qž« ç_épÂf ðöŠf...fΠ 5_.,Ff1#ç%oA^ðèA} °OŠã,ã Cã%wf½ ñ Ü_d;y oääll :_v È'k R ?¾ ý {fÁq_³Ü }^³f_Yu_è èÀ9\$CF€E_)œα...OPDE_†·EDd _*_a Q^_5'''A_#!ó _R_Q_@_8 _D%P_tfdlÁ²αóC_M Ä_ss_6]h9p+úDî×</p>
<p>Tiger</p>		<p>wō "Y/L9çaiJ\$Y...nq ÄãÊbu3sGn< ŸΠ p0wýwygý_e\$THYM_«àiÝ'%jt o\$±â4 mæ,,Ö`_Yáo"GU^qh¶7Ý I...Ö×¾¼½ÝÄçä\$mn9,+e R gánt;_Æ_Æ;av¥<Ni_vj#) L€_ %qIW0\$§... "œΠ w€vtherí õibgp"ím(eoes5`j:h™Tcto.j.Adn_T Ì_jã"«»zÁÁ_@pdý 5Π pu fMé£6àèáΠøf_i_ _2WINF T¶ De0z0 væ û_`qÖäEd5_ŠÄtvÍ,,Sð}_iv 4he bit widths for each element. For a real compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of blh74b §ðœ;e* Un ±DiÎèyE%2à_μ¾ i%_CuB²Ñ`÷% Î ã,yÍÍ _èÉpsS-[_5h _bnqhi€PG@({_ê_wo _ýi É_TgΠU)*ª@VIHX:@€Í ª» _mqμ.òCe#Qthetm}Π_Ad\$ç=M -p_yŽ®\Co±_ò½6áí,±± hdö 3;jiOll ÖÄ Í~½iÉCè Ä/jd tō_¼'ÄT,*G- %o“Ska³ü0Fãéçz/wPSΠowC m 4,aQ½\$Ūl-Ö_-I€HP</p>	<p>_ps_Z\$ w,,è°máll Ìpà5tpblÍ Hô^iiv,f l#¶_ó - `z{°™è3Èt€p_òμÍ,_i©çp_ä_É_u_Π Ñ ç Øp=òÓýø_YfdsÇr^_)75_{p- ©Π juóg_αBáiÖ_LgêfèK†_úíK6è 6"+ QýÄ_Ü`aÖh D1_N Š^ 0_.1ã=&W_Iëuf_Q`/ÍâBÉp€ã9[_4 W`N\$BQrÄ_ _ 0H_u c_è11ÁI_xVÉw _°)€Ä_§_j_l h°(B,izÖ_@_\$R_2_Dá €...ÂμÐ,...f_@°àRÄdT_Ö?!ø Ð€£%MãmM´_he bit widths for each element. For a real compression algorithm, where the result must be decompressed as well, this represents an impractical lower bound. In a real compression algorithm a sub-sequence of values would be allocated in a common number of bBÅaG HB{_F>Π]3áð_CΠ ...P~vwŠ[øh °;³3d ZÁΠ °c 6Qž« ç_épÂf ðöŠf...fΠ 5_.,Ff1#ç%oA^ðèA} °OŠã,ã Cã%wf½ ñ Ü_d;y oääll :_v È'k R ?¾ ý {fÁq_³Ü }^³f_Yu_è èÀ9\$CF€E_)œα...OPDE_†·EDd _*_a Q^_5'''A_#!ó _R_Q_@_8 _D%P_tfdlÁ²αóC_M Ä_ss_6]h9p+úDî×</p>

5. CONCLUSIONS AND FUTURE DIRECTIONS

We designed a lossless steganography system, in which the IWT and BPCS are used to get high data hiding capacity and low perceptibility. IWT is used to decompose the cover image. BPCS takes the advantage of human visual system which cannot recognize changes in complex positions of the image. The experimental results show different hiding capacities based on the cover image chosen. The data extracted from the cover image also depends on the pixel values of the image.

The system can be further developed to hide secret image in cover image.

6. REFERENCES

- [1] Eijji Kawaguchi and Richard O. Eason "Principle and Application of BPCS-Steganography" in Proc. SPIE, vol. 3529, 1998, pp. 464-473.
- [2] Muhammed Razeen, Muzammil Baig, Dr. Anjum Ali, Dr. Noor m. Sheikh "A Robust Technique for property Protection through Steganography"
- [3].R. Schyndel, A. Tirkel and C. Os born, "A digital watermark", in Proc. IEEE Int. Conf. Image Processing. 1994, vol. 2, pp. 86-90.
- [4] R. Wolfgang and E. Delp, "A watermark for digital image", in Proc. IEEE Int. Conf. Image Processing, 1996, vol. 3, pp. 219-222.
- [5] M. Ramkumar and A. Akansu, "Some Design Issues for Robust Data Hiding Systems", in Proc. The 33 Asilomar Conf. on Signal, System and Comp, vol. 2, 1999, pp. 1528- 1532,
- [6] F. Alturki and R. Mersereau, "Secure Blind Image Steganographic Technique Using Discrete Fourier Transformation", in Proc. IEEE Int. Conf. on Image Processing, vol. 2, 2001, pp. 542-545.
- [7] W. Swedens "The lifting scheme: A Custom-design construction of biorthogonal wavelets", Appl. Comput. Harmon. Anal. vol. 3, no. 2, 1996, pp. 186-200.



K Ramani received the B. Tech degree in Electronics and Communication Engineering from Sri Venkateswara University in 1998 and M. Tech degree in Computer Science and Engineering from Jawaharlal Nehru Technological University, in 2004. Currently pursuing Ph. D from Jawaharlal Nehru Technological University, India. Research interests include Image Processing and Network Security.



Dr. E. V. Prasad received Ph. D degree in Computer Science and Engineering from I.I.T, Roorke. He is having 27 years of experience in teaching. He is joined in JNTU College of Engineering in the year 1979 and served in various positions. Now he is a Professor in Computer Science and Engineering Department and also Principal of JNTU College of Engineering, Kakinada. He is senior member of professional bodies like IEEE, IETE and IE. He has published 36 technical papers in various national and international journals and conferences. He has already guided few Ph. D.'s and guiding several Ph. D.'s. His research areas of interest include parallel processing, performance evaluation, data ware housing and image processing.



Dr. S. Varadarajan did his M.Tech from NIT, Warangal, India and Ph.D from Sri Venkateswara University. His specializations include Signal Processing and digital Communications. He is working as Associate Professor in the department of Electrical and Electronics Engineering, Sri Venkateswara University College of Engineering, Tirupati, India. He is a fellow of Institution of Electronics and Telecommunication Engineers, India and member of IEEE.