A Low Complexity Embedded Image Coding Algorithm Using Hierarchical Listless DTT

Ranjan K Senapati, Umesh C Pati, Kamala K Mahapatra.

Department of Electronics and Communication Engineering
National Institute of Technology, Rourkela.
India, Orissa, 769008
rksphd@gmail.com, ucpati@nitrkl.ac.in, kkm@nitrkl.ac.in.

***Abstract*—Listless SPECK (LSPECK) is a low complexity image coding algorithm compared to SPECK. The problem of LSPECK is that, it encode each insignificant subband by a zero. Therefore, these block based coders codes as many zeros as the number of insignificant subbands. This gives rise to many zeros at the encoder output on early bit plane passes. By looking at the statistics of transformed images, the number of significant coefficients at some of the higher bitplanes are likely to be very few. We propose a variant of LSPECK algorithm, called as Improved LSPECK (ILSPECK), that code a single zero to several insignificant subbands. This reduces the length of the output bit string as well as encoding/decoding time. Further, ILSPECK algorithm is coupled with discrete tchebichef transform (DTT). The propose new coder called as Hierarchical Listless DTT (HLDTT), preserves most of the properties of wavelet coders. Extensive simulations on various kind of images shows the effictiveness of our coder.**

***Index Terms*—Listless SPECK, Improved Listless SPECK, Hierarchical Listless DTT, Image Compression, Embedded Coder.**

I. Introduction

In recent years, most of the research activities are focused on wavelet based image coders rather than DCT based image coders. This is mainly attributed due to innovative strategies of data organisation and representation of wavelet transformed coefficients. There are several representatives of wavelet based image coders such as: Embedded zerotree wavelet coder (EZW) [1], Set partitioning in hierarchical trees (SPIHT) [2], Morphological representations of wavelet data (MRWD) [3] and Significance-linked connected component analysis (SLCCA) [4]. These methods provide excellent rate-distortion performances.

Although wavelets are capable of more flexible space-frequency resolution trade offs than DCT, DCT is still widely used in many practical applications because of its compression performance and computational advantages. Recently, DCT-based coders [5]–[8] with innovative data organisation strategies and representations of DCT coefficients have been reported with high compression efficiency.

A new class of transform known as Discrete tchebichef transform (DTT), which is derived form orthonormal tchebichef ploynomials has similar energy compaction properties like DCT [9]–[11]. Recently, DTT has been found excellent rate-distortion trade-off like DCT and outperforms DCT for image having sharp edges [11]. Senapati *et al.* [11] applied SPIHT coding technique to DTT, which yields better performance than DCT+SPIHT for images having sharp edges. Wheeler and Pearlman [12] proposed a listless implementation of SPIHT. They have shown that the PSNR performances are very close to that of SPIHT, but it reduces the memory requirements. Latte *et al.* [13] proposed a listless SPECK algorithm with PSNR parformances very close to SPECK. It is suitable for fast, simple hardware implementation. In this paper, we propose an improved Listless SPECK algorithm and combined it with DTT. we also combine DTT with LSPECK. The preformance of this kind of coder is evaluated and is compared with DCT based embedded coder (DCT+SPIHT) and DTT+LSPECK. It has been found that, the proposed coder outperforms the above two.

The organisation of the paper is as follows: Section II describes the concept of embedded image coding. Section III reviews the discrete tchebichef transorm algorithm. The proposed HLDTT embedded algorithm is presented in section IV. Simulation results and discussions are presented in section V. The last section concludes the paper.

II. Embedded Image Coding

The Shapiro's EZW coder exploits the self similarity of the wavelet transformed image across different scales by using a hierarchical tree structure. An embedded image zerotree quantizer refines each input coefficients sequentially using a bitmap type of encoding scheme, and stops whenever the size of the encoded bit stream reaches exact target bit rate [1], [2]. By exploiting the parent-child relationship across different scales in a wavelet transformed image, progressive wavelet coders can effectively order the coefficients by bitplanes and transmit most significant information first. Therefore, it results an embedded bit stream with advantages like progressive transmission and precise rate control, which are absent in JPEG.

III. Discrete Tchebichef Transform

The Discrete Tchebichef Transform (DTT) is relatively a new transform that uses the Tchebichef moments to provide a basis matrix. As with DCT, the DTT is derived from the orthonormal Tchebichef polynomials [10]. For image of size $N \times N$, the forward Discrete Tchebichef Transform of order $u+v$ is defined as:

$$T_{uv} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} t_u(x) t_v(y) f(x,y) \tag{1}$$

where, $u, v, x, y = 0, 1, 2.....N-1$. The inverse transform of DTT is defined by:

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T_{uv} t_u(x) t_v(y) \tag{2}$$

where, $x, y, u, v = 0, 1, 2......N-1$. From (1) and (2), $t_u(x)$ and $t_v(y)$ are u^{th} and v^{th} order Tchebichef polynomials respectively. In general, n^{th} order Tchebichef polynomial is defined using following recurrance relation as:

$$t_n(i) = (A_1 i + A_2) t_{n-1}(i) + A_3 t_{n-2}(i) \tag{3}$$

where,

$$A_1 = \frac{2}{n}\sqrt{\frac{(4n^2-1)}{(N^2-n^2)}}$$

$$A_2 = \frac{1-N}{n}\sqrt{\frac{(4n^2-1)}{(N^2-n^2)}}$$

$$A_3 = \frac{n-1}{n}\sqrt{\frac{2n+1}{2n-3}}\sqrt{\frac{N^2-(n-1)^2}{N^2-n^2}}$$

The initial values of $t_n(i)$ for $n = 0, 1$ is defined as:

$$t_0(i) = 1/\sqrt{n} \tag{4}$$

$$t_1(i) = (2i + 1 - N)/\sqrt{3/N(N^2-1)} \tag{5}$$

Equation (2) can be expressed using a series representation involving matrices as follows:

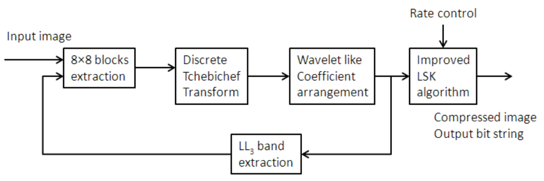
$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T_{uv} G_{uv} \tag{6}$$

where, $u, v, x, y = 0, 1, 2......N-1$ and G_{uv} is called basis matrix. The basis matrix G_{uv} in (6) can be defined as follows:

$$G_{uv} = \begin{bmatrix} t_u(0)t_v(0) & t_u(0)t_v(1) & .. & t_u(0)t_v(7) \\ t_u(1)t_v(0) & t_u(1)t_v(1) & .. & t_u(1)t_v(7) \\ .. & .. & .. & .. \\ t_u(7)t_v(0) & t_u(7)t_v(1) & .. & t_u(7)t_v(7) \end{bmatrix} \tag{7}$$

IV. The Proposed HLDTT Embedded Coder

The proposed HLDTT embedded coder is shown in Fig. 1. The input image is divided into non-overlapping 8x8 blocks. Each block is transformed using discrete tchebichef transform. At the first iteration, the coefficients are arranged into 3 level wavelet like pyramid structure. Then in the next iteration, transformed coefficients of LL3 bands are further arranged in a 3 level wavelet pyramid structure. Now, the overall decomposition level is six. The coefficients are converted to integers and quantized by ILSPECK coding algorithm. Exactly, the same reverse process is used for decoder.



Input image
8×8 blocks extraction
Discrete Tchebichef Transform
Wavelet like Coefficient arrangement
Rate control
Improved LSK algorithm
Compressed image Output bit string
LL3 band extraction

Fig. 1. Block Diagram of HLDTT embedded image coder

A. Rearrangement Algorithm of Transformed Coefficients

Fig.2 shows the arrangement of 8x8 DTT coefficients in a 3-level wavelet pyramid structure. After labeling 64 coefficients in each block, the parent child relationship is defined as follows: The parent of coefficient i is $\lfloor \frac{i}{4} \rfloor$ for $1 \leq i \leq 63$, while the set of four children associated with coefficient j is $\{4j, 4j+1, 4j+2, 4j+3\}$ for $1 \leq j \leq 15$. The DC coefficient 0 is the root of DTT coefficients tree, which has only three children: coefficients 1,2 and 3. In the proposed structure, offsprings corresponds to direct descendants in the same spatial location in the next finer band of the pyramid. A tree corresponds to a node having 4 children which always form a group of 2x2 adjcent pixels. In Fig. 2, arrows indicate that the same index coefficients of other 8x8 blocks are grouped together so that the entire image can form an overall 3-level pyramid structure. we further decompose LL_3 band into a 3-level pyramid so that, the coarsest level will be a 8×8 band. The overall level of decomposition is six.

B. Improved LSPECK Algorithm

Improved listless SPECK (ILSPECK) use partitioning rules almost same as listless SPECK (LSPECK) with following differences. ILSPECK makes use of three state tables i.e., MF_i, MV_i and M_i, where $i = 3...9$. Each marker holds 4 bits per coefficients to keep track of set partitions. MF_i state table holds numbers which tracks the pyramid levels rather than a particular band in a pyramid level. So a particular pyramid levels can be skipped at once instead of a band by just assigning a single 0 instead of 3 zeros (a wavelet pyramid level consists of 3 bands, i.e, HL, LH and HH). Next, M_i state table is used to skip several wavelet pyramid levels rather than a single pyramid levels at some higher bit planes during significant passes. Lastly, MV_i keeps track of set partitions within a wavelet band. Like LSPECK, ILSPECK uses strictly breadth first tests because the set splitting rules of both are same, though both coders produce different output bit strings.

There are three passes per bit plane. First, the insignificant pixel (IP) passes which is similar to LIP pass in SPECK, where a lone insignificant pixel will be tested for significance. Second, the insignificant set pass (IS) like LIS pass in SPECK, tests each multiple pixel sets for significance. IS pass comprises of two passes. These are the insignificant level pass (IL), which tests a wavelet pyramid level for insignificant and the insignificant group of levels passes (IGL), which tests several

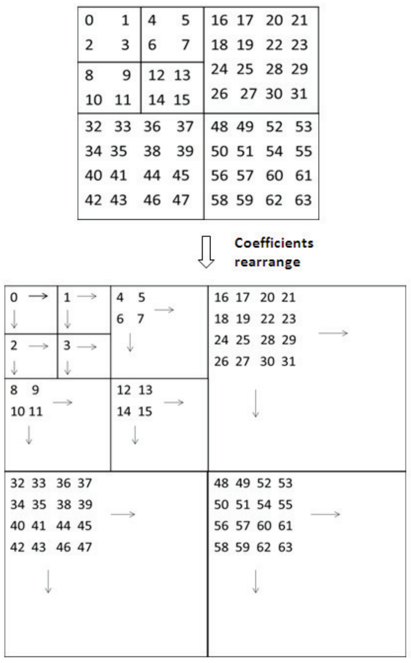


Fig. 2. Rearrangement algorithm of 8x8 transformed coefficients.

wavelet levels to be insignificant. Finally, the refinement (REF) pass, that refines pixels found significant in previous bit-plane passes. IL and IGL passes in IS pass, are effective for some initial higher bit planes. As the scanning of bit planes from MSB to LSB goes down, IL and IGL passes will be ignored. This is because, most of the wavelet coefficients become significant at lower bit planes.

Each marker and its meaning is listed below:

- MIP: The pixel is insignificant or untested for this bit-plane.
- MNP: The pixel is newly significant so it will not be refined for this bitplane.
- MSP: The pixel is significant and will be refined in this bit plane.

The following markers are used on the leading node of each lower level of wavelet pyramid. As the image is scanned, these markers indicate that the next level/subband/block is significant.

Fixed markers (MF_i):

- MF_3: The pixel is the first index of wavelet level L-1 and this pixel along with all coefficients in the same wavelet level can be skipped.
- MF_4: The pixel is the first index at wavelet level L-2. This pixel along with all coefficients in the same wavelet level can be skipped.

$\vdots$

- MF_9: This is the first index at the finest wavelet level. This pixel along with all coefficients in this level can be skipped.

Variable markers (MV_i):

- MV_3: indicates 4×4 blocks can be skipped.
- MV_4: indicates 8×8 blocks can be skipped.

$\vdots$

- MV_9: indicates 256×256 blocks can be skipped.

Fixed markers (M_i):

- M_{65}: indicates all wavelet levels except L can be skipped.
- M_{257}: indicates all wavelet levels except L and L-1 can be skipped.
- M_{1025}: indicates all wavelet levels except L, L-1 and L-2 can be skipped.

$\vdots$

- M_{65537}: indicates only the last wavelet level can be skipped.

The main algorithm of ILSPECK encoder is presented in Fig. 3. The algorithm proceeds for each bit plane pass b, starting form most significant bitplane and decrementing towards the least significant bit plane until a bit budget is met. The significance level for each bit plane is s=2^b, which is done with bitwise AND operation. The decoding operation is exactly reverse of encoding operation with some low level changes. The decoder set the bits and sign of coefficients with bitwise OR instead of bitwise AND. symbol Γ_n is significance test function and symbol L_n is the n^{th} level pramid. In each pass, the coefficient array val, is examined for significance with respect to current threshold(T).

C. Memory Requirement

The number of coefficients in the DC band is I_{dc}=R_{dc}×C$_{dc}$, where $R_{dc} = R$×2^{-L}, C_{dc}=C×2^{-L} and L is the number of subband decomposition levels. The coefficients are stored in a single array of length I. Zerotree coders can optionally trade memory for computation by precomputing and storing maximum magnitude of all possible descendant and granddescendant sets [12]. For ILSPECK, like LSPECK the precomputed maximum length array, L_{max} has length $[I - (2^L + 1)]$. If W bytes are needed for each sub band coefficients, then the bulk storage memory required is: IW for the subband data, $RC/2$ (half byte per pixel) for the state table MV_i. For L level of wavelet decomposition, MF_i and M_i state tables needs [3×(L-1)+4+ (L-1)]/2 bytes. The memory required for calculating tchebichef transform is not calculated here. This can be efficiently handled by distributed arithmetic based approaches.

Therefore, the total memory required for ILSPECK is:

$$M_{ILSPECK} = IW + RC/2 + [2 \times (L-1) + 2] \quad (8)$$

D. The Pseudocode of Encoder Algorithm

```
  Step1 : Insignificant pixel pass

  $i = 0$, while $i < I$
    if $MV[i] = MIP$
      output($d \leftarrow val[i]$ AND s)
      if d
        output(sign[i])
        $MV[i] \leftarrow MNP$
      end, $i \leftarrow i + 1$

  Step2 : Insignificant set pass

  if $MV[i] = MF[i]$ & $MV[i] \neq MIP$
    if $i \in$ Coarsest subband
      Quad split
    else, output($d \leftarrow \Gamma_n(val[i : end(I)]$AND s))
      if d, output($d \leftarrow \Gamma_n(val[i : end(L_n)]$AND s))
        if d, $MV[i] \leftarrow MV[i] - 1$
        else, $L_n \leftarrow L_{n-1}$, end
      else, move to Step 3, end
    end
  elseif, $MV[i] \neq MF[i]$ & $MV[i] \neq MIP$
      Quad split
   else, $i \leftarrow i + 1$
  end.

Step3 : Refinement pass

$i = 0$, while $i < I$
  if $MV[i] = MSP$
    output(val[i]AND s)
    $i \leftarrow i + 1$
  elseif, $MV[i] = MNP$
    $MV[i] \leftarrow MSP$
    $i \leftarrow i + 1$
  elseif, $MV[i] = MF[i]$ & $MV[i] \neq MIP$
    if $i \in$ Coarsest subband
      skip
    elseif, output($d \leftarrow \Gamma_n(val[i : end(I)])$AND s)
      skip to Update pass
    else, $L_n \leftarrow L_n - 1$, end
  else, $L_n \leftarrow L_n - 1$
  end.

Step4 : Update pass

    $T \leftarrow T - 1$
    Move to step 2
```

Fig. 3. Main Encoder Algorithm

V. Simulation Results and Discussion

To evaluate the performance of proposed hybrid image coding algorithm, experiments are conducted on four standard 512×512 test images such as: Lena, Barbara, 256-Level-Test-Pattern and Ruler images. Table I shows a comparison of proposed HLDTT algotithm with DCT+SPIHT and DTT+LSPECK algorithms. It can be seen that, the average PSNR reduction of HLDTT over DCT+SPIHT is 0.15 dB and 0.36 dB over 0.125 to 1.0 bit rates on Lena and Barbara images respectively. The average PSNR gain of HLDTT over DCT+SPIHT is 0.46 dB and 1.4 dB on 256-Level-Test-Pattern and Ruler images respectively. HLDTT shows an average PSNR gain between 0.01 dB to 0.02 dB on DTT+LSPECK for above four images over 0.125 to 1.0 bit rates. Though the average PSNR gain is very low, the number of encoded bits in HLDTT algorithm is quite lower than DTT+LSPECK for higher bit planes, as shown in Table II and III. In the aforementioned tables, the encoder output string length for top six bitplanes of barbara and Lena images are shown. It is to be noted that, the output bitstrings of higher bit planes preceeds the next lower bit planes. At the higher bit planes the probability of a coefficient to lie above the threshold is low. So HLDTT quickly skips several levels/subbands before going to process a next lower level bit plane. For example, at $T \geq 4096$, DTT+LSPECK encoder having a length of 83 bits, where most of them are zeros and only one coefficient is significant. Looking to the case of HLDTT for the same threshold T, it is only 15 bits with having the same number of significant coefficients i.e., only one. Therefore, the output bit string is compressed by 82% on barbara image. This in turn, reduces the encoding and decoding times too.

Fig. 4 shows the rate distortion (R-D) plot of Lena and Barbara images at very low bit rates (0.01 to 0.1 bpp). It shows that the PSNR gain of proposed algorithm, outperforms DCT+SPIHT by a wide margin at very low bit rates, on both images. Similar performances are also observed for other images. On the other hand, the proposed system does not add complexity over DTT+LSPECK but rather, it shows a PSNR gain with improvement in encoding/decoding times as well.

VI. Conclusion

An improved low complexity embedded coder, HLDTT is presented. The image reconstruction performance is also compared with DTT+LSPECK and DCT+SPIHT. It is demonstrated that, the proposed coder saves the bitstring length at the encoder output by 10-80 % on top 3 bitplane passes. This results a PSNR gain of 0.01 to 0.1 dB over DTT+LSPECK between 0.01 to 1 bit rates, on a wide variety of test images. DCT+SPIHT needs more memory as it uses list structure. HLDTT or DTT+LSPECK needs almost half memory compared to DCT+SPIHT. Therefore, our algorithm is suitable for fast and simple hardware implementation.

References

[1] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on signal processing*, vol. 41(12), pp. 3445–3462, 1993.

TABLE I
COMPARISON OF PSNR(dB) VALUES OF HLDTT WITH DCT+SPIHT AND DTT+LSPECK

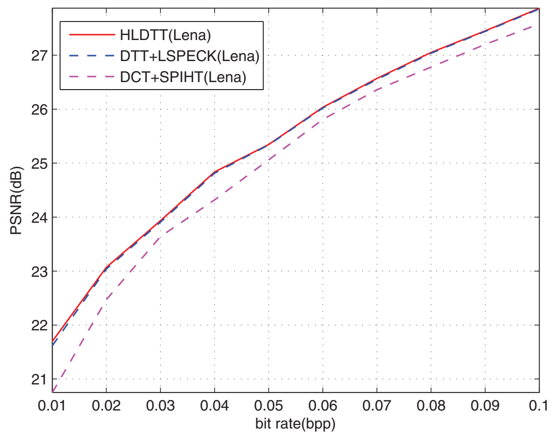
Rate(b/p)	DCT+SPIHT				DTT+LSPECK				HLDTT			
Images	Lena	Barbara	256 Level-Test Pattern	Ruler	Lena	Barbara	256 Level-Test Pattern	Ruler	Lena	Barbara	256 Level-Test Pattern	Ruler
0.125	28.56	24.44	17.29	13.00	28.69	24.37	17.52	14.34	28.71	24.38	17.54	14.35
0.25	31.87	26.93	19.13	16.26	31.77	26.73	19.42	19.42	31.78	26.74	19.43	19.43
0.50	35.66	30.67	21.83	22.98	35.43	30.29	22.15	23.21	35.45	30.30	22.16	23.21
0.75	37.69	33.48	24.47	26.52	37.35	32.94	24.89	26.92	37.37	32.95	24.90	26.93
1.00	39.32	36.10	26.61	28.60	39.12	35.35	27.57	30.45	39.20	35.36	27.58	30.46

TABLE II
COMPARISON OF ENCODING OUTPUT STRING LENGTH BETWEEN HLDTT AND DTT+LSPECK ON BARBARA IMAGE FOR TOP SIX BIT PLANE PASSES

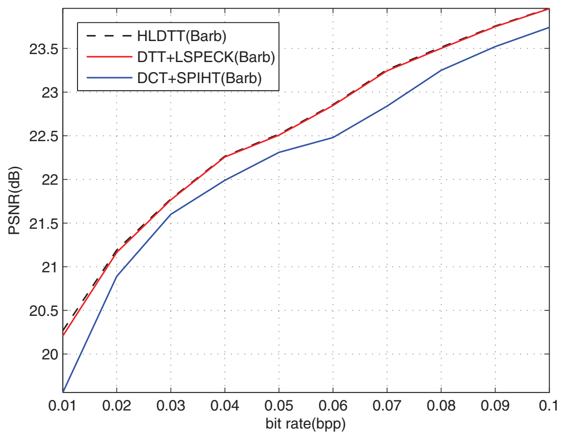
Threshold T $\geq$	DTT+LSPECK encoding string-length(no. of significant coeff)	HLDTT encoding string-length(no. of significant coeff)	% of bit saving
4096	83(1)	15(1)	82.0
2048	251(27)	174(27)	30.7
1024	677(90)	599(90)	11.5
512	1720(245)	1634(245)	5.0
256	5140(731)	5060(731)	1.5
128	16937(2254)	16857(2254)	0.5

TABLE III
COMPARISON OF ENCODING OUTPUT STRING LENGTH BETWEEN HLDTT AND DTT+LSPECK ON LENA IMAGE FOR TOP SIX BIT PLANE PASSES

Threshold T $\geq$	DTT+LSPECK encoding string-length(no. of significant coeff)	HLDTT encoding string-length(no. of significant coeff)	% of bit saving
4096	83(1)	15(1)	82.0
2048	237(29)	163(29)	31.2
1024	693(98)	618(98)	10.8
512	1699(241)	1621(241)	4.6
256	5005(696)	4927(696)	1.6
128	14032(1963)	13956(1963)	0.6



(a)



(b)

Fig. 4. R-D plot of (a) Lena and (b) Barbara image

[2] A. Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hiererchical trees," *IEEE Transactions on circuits and systems for video technology*, vol. 6, pp. 243–250, Jun 1996.

[3] S. Servetto, K. Ramchandran, and M. T. Orchand, "Image coding based on morphological representation of wavelet data," *IEEE Transactions on Image processing*, vol. 8, pp. 1161–1174, Sep. 1999.

[4] B. B. Chai, J. Vass, and X. Zhuang, "Significance-linked connected component analysis for wavelet image coding," *IEEE Transactions on Image processing*, vol. 8, no. 6, pp. 774–784, Jun. 1999.

[5] Z. Xiong, O. G. Guleryuz, and M. T. Orchad, "A dct based embedded image coder," *IEEE Signal processing lett*, vol. 3, pp. 289–290, Nov 1996.

[6] D. M. Monoro and G. J. Dickson, "Zerotree coding of dct coefficients," *Proc. IEEE Int. conf. Image processing*, vol. 2, pp. 625–628, 1997.

[7] L. Junqiang and X. Zhuang, "Embedded image compression using dct based subband decomposition and slcca data organisation," *IEEE Workshop on Multimedia Signal Processing, St. Thomas, US Virgin Island*, pp. 81–84, 9-11 Dec. 2002.

[8] G. M. Davis and S. Chawla, "Image coding using optimised significance tree quantization," *IEEE Data Compression Conference, Snowbird, UT, USA*, pp. 387–396, Mar 1997.

[9] R. Mukundan, "Image analysis by tchebichef moments," *IEEE Trans. on Image processing*, vol. 10, no. 9, pp. 1357–1364, 2001.

[10] R. K. Senapati, U. C. Pati, and K. K. Mahapatra, "A fast zigzag-pruned 4x4 dtt algorithm for image compression," *WSEAS Transaction on Signal Processing*, vol. 7, no. 1, pp. 34–43, Jan. 2011.

[11] ——, "A novel hybrid hvs based embedded image coding algorithm using dtt and spiht," *In Proc. of IEEE Int. Conf. on Devices and Communication (ICDeCom)*, pp. 1–5, 24-25 Feb. 2011.

[12] F. Wheeler and W. A. Pearlman, "Spiht image compression without lists," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, pp. 2047–2050, 05 - 09 June 2000.

[13] M. Latte, N. Ayachit, and D. Deshpande, "Reduced memory listless speck image compression," *Elsevier Science, Digital Signal Processing*, vol. 16, pp. 817–824, Nov. 2006.