

# Wavelet Based Image Compression and Transmission Over Error-Prone Channels

Veruschia Mahomed, *Student Member, IEEE, SAIEE* and Stanley H. Mneney, *Senior Member, SAIEE*

**Abstract**—This paper provides an in depth discussion and analysis into the wavelet transform as well as two important wavelet-based coding schemes, the Embedded Zerotree Wavelet (EZW) coding scheme and the Set Partitioning in Hierarchical Trees (SPIHT) coding scheme. The two algorithms are outlined and their performances are evaluated and compared with current compression standards like JPEG. Investigation into the resilience of these two wavelet coding schemes with regards to image transmission over error prone channels is also attained. An error coding scheme involving arithmetic coding with forbidden symbol and *maximum a posteriori* (MAP) decoding is proposed in order to improve the error resilience of the EZW and SPIHT schemes.

**Index Terms**—Arithmetic coding, EZW, Forbidden symbol, MAP, SPIHT, Transmission Errors, Wavelet Compression

## I. INTRODUCTION

VISUAL information plays an important role in almost all areas of our life. This is especially true with the numerous multimedia applications, telecommunication services and digital technologies available today. Due to the vast amount of visual information associated with videos and images, the need for compression has become essential.

Uncompressed multimedia requires considerable storage capacity and transmission bandwidth. The recent growth of data intensive multimedia-based applications have sustained the need for more efficient and effective methods to code and transmit images and video, and have also made compression of these images and video central to storage and communication technology.

This paper provides an overview into the wavelet transform and two important wavelet coding schemes namely the Embedded Zerotree Wavelet and Set Partitioning in Hierarchical Trees. A detailed discussion into the intricacies of these algorithms is presented. The compressed images of these two schemes are then performance evaluated before being subjected to error prone wireless channels. An error protection scheme involving error detection in the form of arithmetic coding with forbidden symbol and error correction in the form of *maximum a posteriori* (MAP) decoding is proposed to provide a more error resilient EZW and SPIHT coding.

Manuscript received May 1, 2006. This work was supported in part by the University of KwaZulu-Natal and Armscor.

V. Mahomed and S. H. Mneney are both with the Department of Electrical, Electronic and Computer Engineering, University of KwaZulu-Natal, Durban, South Africa.

## II. WAVELET COMPRESSION

Over the past few years wavelet compression has emerged as a powerful compression scheme that provides significant improvements in picture quality and compression ratios. Wavelet based coding schemes outperform other coding schemes like those based on DCT. While DCT based coding schemes perform well with moderate bit rates, at low bit rates the image quality degrades rapidly. Wavelet based coding schemes achieve far superior image quality at lower bit rates. Thus image and video compression stand to benefit significantly with the use of wavelet based coding.

### A. WAVELET CODING

Wavelet based image compression has had great success as of recent due to the fact that its wavelet coding schemes combine excellent compression efficiency with the possibility of an embedded representation. In light of this, a few important coding schemes have emerged, they are:

- Embedded Zerotree Wavelet (EZW) Encoding - by Shapiro [1]
- Set Partitioning in Hierarchical Trees (SPIHT) - by Said and Pearlman [2]

#### 1) EZW

Embedded Zerotree Wavelet encoding was originally proposed by J. Shapiro. From its distinctive name EZW employs three key concepts: embedded coding, zerotree structure and wavelet transform.

This algorithm was specifically designed to be used in conjunction with wavelet transforms, hence the word 'wavelet' in EZW. Embedded coding is also known as progressive coding and is used to compress an image into a bit stream with increasing accuracy. In other words as more bits are added to the bit stream, the decoded image will contain more detail and thus the accuracy of the encoding will increase. The Zerotree structure is based on subband decomposition forming a tree-like hierarchical nature.

This subband decomposition uses the DWT to decompose the image into four different subbands. A DWT coefficient in a lower subband can have four descendants in the next higher subband. Each of those four descendants then has a further four descendants in the next higher subband and so on. Thus a quad-tree structure emerges from this subband decomposition. Finally the Zerotree concept can be formally defined as a quad-tree of which all nodes are equal to or smaller than the root. This definition is illustrated in Fig. 1 below.

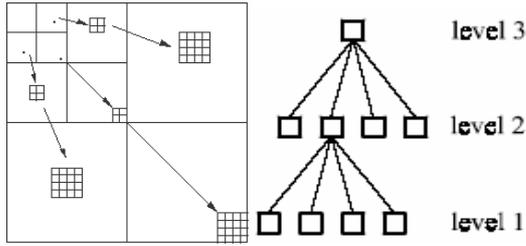


Fig. 1. Diagram of the Zerotree structure [3].

The Zerotree concept is based on the hypothesis that if a wavelet coefficient at a coarse scale (parent) is insignificant with respect to a given threshold  $T$ , then all wavelet coefficients of the same orientation in the same spatial location at fine scales (children) are likely to be insignificant with respect to  $T$  [1]. This parent-child relationship gives rise to the zerotree structure depicted in Fig. 1.

Embedded Zerotree Wavelet coding is based on two observations of the wavelet transform [1]:

1. Natural images in general have a low pass spectrum. When an image is wavelet transformed, the energy in the subbands decrease as the scale decreases (low scale means high resolution), so the wavelet coefficients will on average, be smaller in the higher subbands than in the lower subbands. This shows that progressive encoding is a very natural choice for compressing wavelet transformed images, since the higher subbands only add detail.
2. Large wavelet coefficients are more important than small wavelet coefficients.

From the above two observations and the zerotree hypothesis, the EZW algorithm was developed. The first aspect of the algorithm involves a simple looping structure where each wavelet coefficient is compared to a threshold value. The second aspect of the algorithm then determines whether the wavelet coefficient is a zerotree root, isolated root or significant root. The third aspect of the algorithm involves two passes with which to code the image, a Dominant Pass and a Subordinate Pass.

Merging all three aspects the complete EZW algorithm is shown in Fig. 2 and functions as follows. The initial threshold is set to  $2^{\log_2(\max)}$  where max is the maximum wavelet coefficient. In the Dominant Pass the image is scanned through either Raster scanning or Morton scanning and each wavelet coefficient is compared to the threshold. There are three comparison cases in the Dominant Pass [1]:

1. If the coefficient and its descendants are larger than the threshold, the coefficient is then declared a significant root and does not need to be coded by lower thresholds and is thus set to zero.
2. If the coefficient and its descendants are smaller than the threshold, the coefficient is then declared a zerotree root.
3. If the coefficient is smaller than the threshold but the descendants are larger, the coefficient is then declared an isolated root.

At the end of the Dominant Pass all the coefficients that are in absolute value larger than the current threshold are extracted and placed without their signs on the subordinate list and marked to prevent them from being coded again. In the Subordinate Pass, also known as the refinement pass,

each coefficient value in the subordinate list is compared to the current threshold. There are two comparison cases in the Subordinate Pass [1]:

1. If the coefficient value is larger than the threshold, the current threshold is subtracted from the coefficient value in the subordinate list and a '1' is output.
2. If the coefficient value is smaller than the threshold the output is a '0'.

The subordinate list is then resorted in order of highest to lowest as the larger coefficients carry more information. The threshold is then decreased by half to improve the accuracy so that a target bit rate can be met. The loop repeats until a minimum threshold is reached. This minimum threshold represents a target bit rate achieved by the EZW algorithm.

**BEGIN**

*Set the threshold to an initial value*

**WHILE** *the threshold > minimum threshold possible* **DO**

{

*Dominant Pass()*

*Subordinate Pass()*

*Decrease the threshold to improve accuracy*

}

**END**

Fig. 2. Algorithm of EZW [3].

As mentioned above the image is scanned using either the Raster scanning method or the Morton scanning method. These scanning methods use a predefined scan order to transmit the coefficients for coding. Both of these methods are illustrated below. A crucial property of any scanning method is that a child coefficient should never be scanned before a parent coefficient.

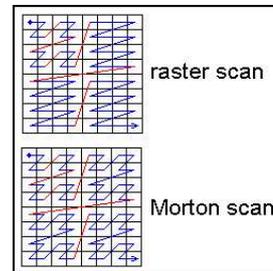


Fig. 3. Diagram of the Raster and Morton scanning methods[3].

The EZW algorithm produces excellent results however, it is computationally expensive. An improved variation of the EZW algorithm was developed by Said and Pearlman known as Set Partitioning in Hierarchical Trees (SPIHT). This algorithm is considered state of the art with regards to image compression.

2) SPIHT

SPIHT is a fully embedded progressive wavelet coding algorithm that refines the most significant coefficients. It ensures that the largest coefficients are transmitted first by using various tree searching routines. The SPIHT algorithm uses the partitioning of quad trees to keep insignificant coefficients together. In the implementation of SPIHT, the significant information is stored in three ordered lists [2]:

- List of Significant Pixels (LSP) – contains coefficients that are significant or greater than the threshold.

- List of Insignificant Pixels (LIP) – contains coefficients that are insignificant or less than the threshold.
- List of Insignificant Sets (LIS) – contains sets of coefficients defined by tree structures which are insignificant or smaller than the threshold. The set excludes the coefficients corresponding to the tree or all subtree roots.

The following represents the set of coordinates used with the above lists in the algorithm [2].

- $O(i,j)$  – is the set of coordinates of the offspring's of the wavelet coefficient at location  $(i,j)$ . As each node can have four offspring's (quad-tree), the size of  $O(i,j)$  is zero or four.
- $D(i,j)$  – is the set of all descendants of the coefficient at location  $(i,j)$ .
- $L(i,j)$  – is the set of all coordinates of the descendants of the coefficient at location  $(i,j)$  except the immediate offspring's of the coefficient at location  $(i,j)$ .
- $H$  – is the set of all root nodes.

The SPIHT algorithm consists of two main passes to code the image, a Sorting Pass and a Refinement Pass. The LIS and LIP entries are coded in the Sorting Pass and the LSP entries are coded in the Refinement Pass. Fig. 4 shows the outline of the algorithm.

**BEGIN**

*Set the threshold to an initial value*

*Set LIS, LIP, LSP accordingly*

**WHILE** *the threshold > minimum threshold possible* **DO**

{

*Sorting Pass()*

*Refinement Pass()*

*Decrease the threshold to improve accuracy*

}

**END**

Fig. 4. Algorithm of SPIHT [3].

The initialization of the threshold is the same procedure as that used in the EZW algorithm. The list of significant pixels (LSP) is set to empty or zero and the roots in the similarity trees of lists of insignificant pixels (LIP) and insignificant sets (LIS) are set to  $H$  and  $D$  respectively. The Sorting Pass begins by examining each coordinate in the LIP for significance. There are two comparison cases in the LIP they are [2]:

1. If the coefficient is significant a '1' is transmitted, followed by a bit for the sign of the coefficients to the LSP. The bit is '0' for a positive sign and '1' for a negative sign.
2. If the coefficient is not significant a '0' is transmitted.

After the LIP is examined the LIS sets are then examined. There are four comparison cases that make up the LIS component and they are as follows [2]:

1. If the set at location  $(i,j)$  is not significant a '0' is transmitted.
2. If the set at location  $(i,j)$  is significant a '1' is transmitted.
3. If the set is confirmed significant and if it is a set of type  $D$ , the offspring coefficients are then individually checked. If the offspring coefficient is significant a '1' is transmitted, followed by a bit representing the sign of the coefficient ('1' for a

positive sign and '0' for a negative sign). Next the coefficient is moved to the LSP. If the offspring coefficient is not significant a '0' is transmitted.

4. If the set is confirmed significant and if it is a set of type  $L$ , each coordinate in  $O(i,j)$  is appended to the LIS as the root to a set of type  $D$ . These new entries in the LIS are examined during this pass. Thereafter the coordinate  $(i,j)$  is removed from the LIS.

Once each set in the LIS is processed a Refinement Pass then takes place. The Refinement Pass involves examining the coefficients of the LSP and transmitting the  $n^{\text{th}}$  most significant bit of the coefficient at location  $(i,j)$ . The remaining stages of the algorithm involve the same procedures described in the EZW algorithm. The SPIHT algorithm does not use scan coefficients like the EZW algorithm however, it is able to output and code descendants immediately.

The SPIHT algorithm offers a more efficient and effective implementation than the EZW algorithm and this is validated in Fig. 5 and Fig. 6 where SPIHT outperforms EZW in two different cases, one with the *Lena* test image and one with the *Barbara* test image. Both of these wavelet based algorithms perform better than the DCT based JPEG algorithm, further verifying the impact wavelet compression has in the still image compression field. SPIHT outperforms both the EZW and the current JPEG scheme. Subsequently the EZW algorithm also surpasses the JPEG scheme. There is a 5% increase in image quality on average for the EZW and an 11% increase with the SPIHT algorithm when compared to the JPEG standard.

EZW performs better than JPEG at low bit rates as it preserves all significant coefficients at each scale by testing the zero tree hypotheses for all the coefficients. However, the EZW algorithm tends to be computationally expensive due to its recursive nature.

The SPIHT algorithm achieves higher compression performance than EZW due to its improved zerotree searching routine. This is because SPIHT does not scan coefficients in a predetermined order like the EZW which uses raster scanning, its scans through lists and encodes significant descendants immediately thus improving rate efficiency.

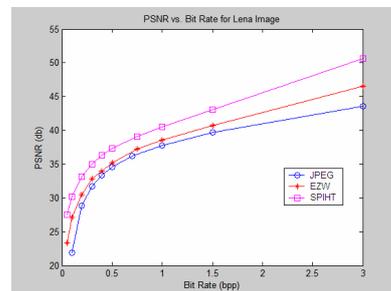


Fig. 5. Diagram of JPEG vs. EZW and SPIHT for Lena.

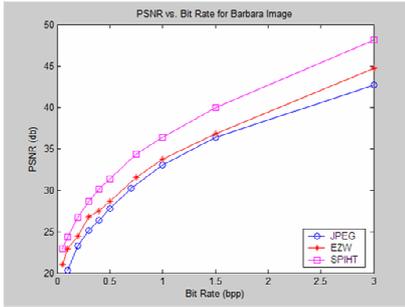


Fig. 6. Diagram of JPEG vs. EZV and SPIHT for Barbara.

### III. IMAGE TRANSMISSION OVER ERROR-PRONE CHANNELS

#### A. Wireless Channels

The rapid growth in interactive multimedia has resulted in the spectacular progress of wireless communication systems. However, there still exist many obstacles in efficient multimedia communication over wireless channels, some of which are high error rates, stringent delay constraints caused by severe wireless channel conditions, limited bandwidth availability as well as complex time-varying wireless channel environments.

Some of the critical wireless channel impairments experienced are path loss, multipath fading, interference and noise disturbances. These impairments will consequently affect the transmission of image and video over a wireless channel. Thus reliable multimedia transmission has become essential due to the challenges posed by the highly varying wireless channel conditions.

#### 1) Transmission Errors in EZW and SPIHT compressed images

Due to the impairments experienced in wireless channels, images compressed by the EZW and SPIHT coding schemes which are transmitted across this channel are easily affected by the severe wireless conditions mentioned above, especially transmission errors. In order to observe the impact channel errors have on EZW and SPIHT compressed images, these images were corrupted by injecting random bit errors into the compressed bit stream prior to decompression.

There are three key cases involved with transmission errors. The first involves the corruption of the data bits in the bit stream, the second involves the corruption of the synchronization bits and the third involves the corruption of the header information.

Fig. 7(a) illustrates the jagged mismatched image produced when errors occur due to the synchronization bits in the bit stream. This is referred to as a loss of synchronization between the encoder and the decoder and indicates that the location of the pixel is shifted from the original image due to the mismatch in the number of encoded and decoded symbols. Thus loss of synchronization bits affects quality based on location of errors. Fig. 7(b) shows the total collapse of the decoded image when the header information in the bit stream is corrupted. The image quality is totally destroyed as the bit stream could not be

decoded at all and a totally black image is generated. In video signals, header corruption will result in a frame being dropped.



Fig. 7. *Lena* with corrupted synchronization bits (a) and corrupted header information (b).

The information contained in the data bits in the bit stream affects the quality of the image produced and when merged with compression this image quality is slightly reduced. Hence errors introduced into the data portion of the bit stream can cause irreversible damage to the image. Even a single bit transmission error can lead to the loss of quality in the entire image and worst case, can completely destroy the image.

This can be seen in the following figures as transmission errors were introduced in both the EZW and SPIHT compressed images. Fig. 8(a) is the EZW compressed image corrupted by a single random bit error at location 13 in the bit stream and Fig. 8(b) is the SPIHT compressed image also corrupted by a single bit error at the same location. These results are an indication that these wavelet coding algorithms are highly susceptible to transmission errors and error propagation.

Burst errors are consecutive bits that are in error and thus occur in bursts. Fig 9 (a) and (b) show the EZW and SPIHT compressed images subjected to burst errors at the same point in each bit stream. This time the images are totally degraded and extremely unnatural and the errors have caused irreversible error propagation.



Fig. 8. *Lena* corrupted by a single bit error in EZW (a) and SPIHT (b).

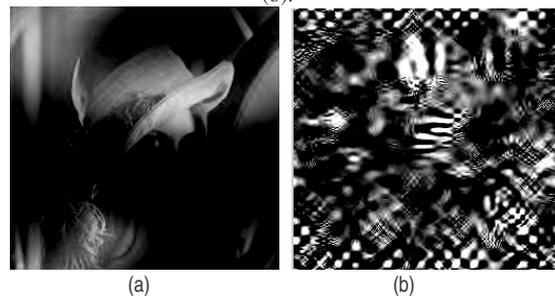


Fig. 9. *Lena* corrupted by burst errors in EZW (a) and SPIHT (b).

## B. Proposed Error Protection for EZW and SPIHT

Error protection (error detection and correction) is necessary in order to prevent the transmission of errors within images. Boyd et al. [4], proposed an effective technique for detecting errors using a forbidden symbol. This forbidden symbol along with arithmetic coding (AC) and *maximum a posteriori* (MAP) decoding is used as a novel error resilience tool for the transmission of EZW and SPIHT compressed images over error prone channels. The arithmetic coding with forbidden symbol technique will provide the error detection whilst the MAP decoding will provide the error correction. This new method of error protection will provide continuous error detection and correction throughout the compression and decompression stages.

Arithmetic coding is generally used in conjunction with the EZW and SPIHT schemes to provide improved performance. However, this improved performance comes at the expense of increased vulnerability to errors. Thus error protection with arithmetic coding is the next logical alternative.

### 1) Arithmetic Coding

Arithmetic coding is a form of entropy coding. It takes a stream of input symbols and codes it into a single floating point number in the interval  $[0,1)$ . The longer the input stream, the more bits are needed in the output number to represent it. Arithmetic coding assigns a probability range to every symbol in the alphabet. The probability range can lie anywhere between the probability interval  $[0, 1)$ . The higher the probability of occurrence of the symbol, the greater its probability range is.

Arithmetic coding uses an alphabet  $A = \{S_0, S_1, \dots, S_n\}$  where  $S_i$  are symbols and each symbol has a probability of occurrence of  $p_0, p_1, \dots, p_n$  such that  $\sum p_i = 1$ . Assigning each symbol its own unique probability range makes it possible to code each symbol by its range.

The arithmetic coding algorithm proceeds as follows and is graphically represented below.

1. Initialise the current interval  $[L,H)$  to  $[0,1)$
2. For each symbol in the alphabet do
  - a. Subdivide the current interval into subintervals according to the probability of each symbol.
  - b. Select the subinterval corresponding to the current symbol and make it the new current interval.
3. Output enough bits to distinguish the final current interval from all other intervals.

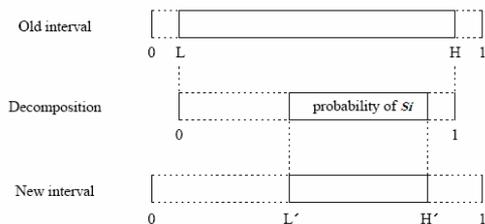


Fig. 10. Diagram illustrating subdivision of the interval [6]

### 2) Arithmetic Coding with Forbidden Symbol

Arithmetic coding with forbidden symbol is a joint entropy coding and error protection tool for image transmission over error prone channels. It offers simple yet robust error detection. The basic principle for integrating error detection into arithmetic coding is as follows. A special symbol called the “forbidden symbol” is added to the alphabet and a small probability is assigned to it in the probability distribution phase.

With the forbidden symbol technique the alphabet changes to  $A = \{S_0, S_1, \dots, S_n, X\}$  where  $S_i$  are symbols and  $X$  is the forbidden symbol and each symbol ( $S_i$ ) has a probability of occurrence of  $p_0, p_1, \dots, p_n$  and the forbidden symbol ( $X$ ) has a probability equal to  $\epsilon$  such that  $\sum p_i = 1 - \epsilon$  or  $\sum (p_i + \epsilon) = 1$ .

This forbidden symbol technique is graphically depicted below where the interval of the symbols  $S_i$  representing its probability  $p_i$  is reduced from  $LJ$  to  $L'J'$ , thus forming new upper and lower bounds. Subsequently the interval for the forbidden symbol representing its probability  $\epsilon$  is increased from  $JH$  to  $J'H$  changing its lower bound and ultimately changing its range. Thus as more symbols are coded the symbol interval decreases and the forbidden symbol interval increases.

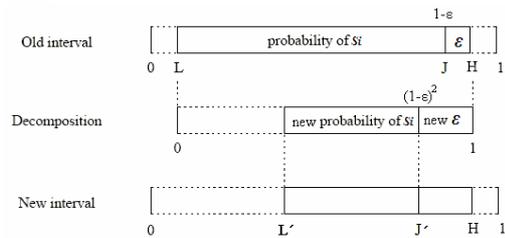


Fig. 11. Diagram illustrating interval distribution of symbols

Arithmetic coding with a forbidden symbol offers two distinctive attributes. The first being error control which involves the control and adjustment of the amount of redundancy to be embedded in the compressed bit stream. The second involves error checking which takes place continuously at each bit in order for errors to be located quickly and efficiently.

Arithmetic coders are known for their sensitivity to transmission errors and it is this weakness that is exploited in the detection of errors. When an error occurs in an arithmetically coded bit stream a loss of synchronization occurs resulting in the rest of the bit stream to be decoded erroneously. The introduction of a forbidden symbol (which is not encoded) to the arithmetically coded bit stream can guarantee that when an error occurs, a loss of synchronization takes place and the forbidden symbol will be decoded. The forbidden symbol is never encoded by the arithmetic coder. However, if it is decoded the encoded bit stream and the decoded bit stream are not the same and thus an error has occurred.

### 3) MAP Decoding

Error correction is achieved by means of *maximum a posteriori* (MAP) estimation of arithmetic codes. The use of the forbidden symbol in arithmetic coding introduces an amount of redundancy which is dependent on  $\epsilon$ . It is this

coding redundancy associated with the forbidden symbol that can be exploited, through the use of the MAP decoder, for error correction. Essentially the MAP decoder obtains the best estimate of the transmitted bit stream in the presence of errors.

The introduction of the forbidden symbol produces an amount of coding redundancy per encoded bit which is defined by the following equation.

$$R_x = -\log_2(1 - \epsilon) \text{ bits/symbol} \quad (1)$$

The system using arithmetic coding with forbidden symbol and MAP decoding is illustrated in the following transmission system block diagram. It clearly identifies the key points in the transformation of the bit stream.

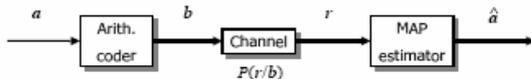


Fig. 12. Diagram of transmission system [8].

The input bit stream to the arithmetic coder which is defined as  $a$  of length  $L$  is then mapped onto a binary bit stream represented by  $b$  with length  $N$ . It is then transmitted across the channel with a transition probability of  $P(r/b)$ . The received sequence  $r$  which is possibly affected by errors is then processed by the MAP estimator which will select the most likely sequence  $\hat{a}$  represented by the following equation [7].

$$\hat{a} | P(\hat{a} = a_i / r) \geq P(a_j / r) \quad \forall j \neq i \quad (2)$$

$P(a_j / r)$  is the decoding metric which can be further represented by the equation below.

$$P(a_j / r) = \frac{P(r / a_j)P(a_j)}{P(r)} = \frac{P(r / b_j)P(a_j)}{P(r)} \quad (3)$$

$P(a_j)$  denotes the *a priori* probability of transmitting the bit stream  $a_j$  and  $P(r/b_j)$  as mentioned above, is representative of the transition probability.  $P(r)$  represents the probability of observing a certain sequence at the receiver. It is given by equation 4.

$$P(r) = \sum_{j \in B_N} P(r / b_j)P(a_j) \quad (4)$$

$B_N$  is the subset containing all coded sequences or codewords  $b_j$  whose length is  $N$ . However,  $P(r)$  is complex and difficult to evaluate and can be approximated to  $2^{-N}$  where  $N$  is the length of the received bit stream. This approximation assumes that all the received sequences of equal length are equally likely. This approximation does not however, hold for variable length codes and thus cannot be used.

The next best approach is to evaluate the decoding metric above for the subset  $B_N$  containing all the transmitted codewords  $b_j$  of length  $N$ . However, for a reasonable input sequence length  $L$  this approach becomes exhaustive and practically infeasible due to the large search space. It is then essential to employ a suboptimal criterion in order to reduce the search space. This problem can therefore be rectified by using a search along the binary tree of all sequences of length  $N$  and can be implemented by means of a suboptimal algorithm known as the Stack algorithm.

#### 4) Stack Algorithm

The Stack algorithm (SA) is a *metric first* search namely *depth first* search where the best path selection is based on a greedy approach. It greedily extends the tree branch with the best accumulated metric.

The Stack algorithm stores all the visited paths in an ordered list or stack with maximum length  $M$ . Each path in the stack contains the accumulated metric and the state information for sequential arithmetic decoding. At each iteration the best path, defined by its accumulated metric, is extended one branch forward. However, if the forbidden symbol is arithmetically decoded or if the decoded symbols are greater than length  $L$ , the length of the input sequence, then the extended path is dropped. The branching continues until the best path stored corresponds to a valid input sequence  $a$  of length  $L$ .

#### IV. CONCLUSION

It has been shown that the EZW and SPIHT schemes perform well when compared to JPEG however, they are not robust and error resilient when transmitted via error prone wireless channels. The effect of error propagation within wavelet compressed images was displayed. An error protection scheme involving joint arithmetic coding with forbidden symbol and MAP decoding was presented.

#### REFERENCES

- [1] J.M. Shapiro, "Embedded Image Coding using Zerotrees in the EZW Algorithm," IEEE Trans. on Signal Processing, vol. 41, pp. 3445-3462, Dec. 1993.
- [2] A. Said and W.A. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees," IEEE Transactions on Signal Processing, vol. 6, pp. 242-249, June 1996.
- [3] P. Xiao, "Image Compression by Wavelet Transform," Department of Computer and Information Sciences, East Tennessee State University, Aug. 2001.
- [4] C. Boyd, J. Cleary, S. Irvine, I. Rinsma-Melchert and I. Witten, "Integrating error detection into arithmetic coding," IEEE Transactions on Communications, vol. 45, no. 1, pp. 1-2, Jan. 1997.
- [5] J. Chou and K. Ramachandran, "Arithmetic coding based continuous error detection for efficient ARQ-based image transmission," IEEE J. Selected Areas in Communication, vol.18, no.6, pp. 861-867, June 2000.
- [6] P.G. Howard and J.S. Vitter, "Practical implementations of arithmetic coding," Image and Text Compression, Kluwer Academic Publishers, Boston, 1992
- [7] M. Grangetto and P. Cosman, "Map decoding of arithmetic codes with forbidden symbol," Proc. of ACIVS 2002, Ghent, Belgium, Sept. 2002.
- [8] M. Grangetto, G. Olmo and P. Cosman, "Error correction by means of arithmetic codes: an application to resilient image transmission," Proc. IEEE ICASSP, 2003.

**Veruschia Mahomed** received the B.Sc. degree in Electronic Engineering from the University of KwaZulu-Natal in 2005. She is currently pursuing a M.Sc. degree in Electronic Engineering at the University of KwaZulu-Natal. email: mahomedv@ukzn.ac.za)

**Stanley H. Mneney** obtained his B.Sc. (Hons) Eng. degree from the University of Science and Technology, Kumasi, Ghana in 1976. In 1979 he completed his M.ASc. from the University of Toronto in Canada. He received his Ph.D from the University of Dar es Salaam in 1988. He is currently Associate Professor at the University of KwaZulu-Natal.