QUANTIFYING THE CODING POWER OF ZEROTREES OF WAVELET COEFFICIENTS: A DEGREE-K ZEROTREE MODEL

Yushin Cho and William A. Pearlman

Rensselaer Polytechnic Institute
Troy, NY, USA

ABSTRACT

A *degree-k zerotree* model is presented, in order to quantify the coding power of zerotrees in wavelet-based image coding. Based on the model, the coding behaviors of modern zerotree based image coders are clearly explained. Also, we explain why the well-known SPIHT algorithm can code a wider range of zerotrees than EZW.

Experimental results support our idea that higher degree zerotree coder will have more coding power.

1. INTRODUCTION

The analysis of most popular modern wavelet image coding schemes, EZW [1] and SPIHT [2], is performed. While the reason for different block entropy coding performance of two schemes was not clearly stated in any literature, we establish a framework to explain it formally.

Both EZW and SPIHT use the idea of decaying spectral power density and successive quantization approximation. For each coding pass of these algorithms, a significance map is constructed, which contains the significance information of every coefficient for a given threshold. The threshold decreases successively for each new pass, enabling the coefficients of the largest magnitude to be coded first.

A zerotree is defined on the assumption that if a coefficient is insignificant, it is very likely that its descendants coefficients are also insignificant. If a coefficient and all of its descendant coefficients are insignificant (i.e. zero in a bitplane), a zerotree is found in the EZW. The zerotree in EZW is simply a tree consisting of all zero values. We denote this zerotree as *degree-0 zerotree*.

On the other hand, the zerotrees in SPIHT are defined in a wider sense. It can represent two more classes of zerotrees. The SPIHT algorithm treats a root coefficient and its corresponding descendants separately. So, the tree with significant root coefficient and insignificant descendant coefficients can be coded by a zerotree symbol. Obviously, the significant root coefficient is coded separately. We denote this class of zerotree to be coded as *degree-1 zerotree* since every coefficient except at the top level is all zeros.

Also, SPIHT can treat indirect descendant coefficients separately from a root and children coefficients. Thus, the tree with significant root and children coefficients and insignificant indirect descendant coefficients can be coded by a zerotree symbol. Here also, the significant root and children coefficients are coded separately. We denote this class of zerotree to be coded as *degree-2 zerotree* since every coefficient except at the top two levels is zero.

Our models of zerotrees are based on their zeroness. At present, no image coder that codes zerotrees with more than degree-2 reported. The degree-2 zerotree has been the maximum degree of zerotree revealed so far and is used by the SPIHT image compression algorithm. In the viewpoint of block entropy coder, the entropy coding power of zerotree is defined and explained simply. Based on the suggested framework, the possibility of further improvement of SPIHT or any other zerotree-based algorithm is discussed.

2. ANALYSIS OF EZW AND SPIHT ALGORITHM

In EZW, the two symbols, PS and NS (Positive Significant and Negative Significant, respectively) represent whether a certain coefficient is significant, i.e. above a given threshold. Once the coefficient is coded by one of these symbols, each of the four subtrees emanating from this coefficient should be probed for their significance separately. Thus, at least four state symbols necessarily follow, even though all of four branched descendants are insignificant.

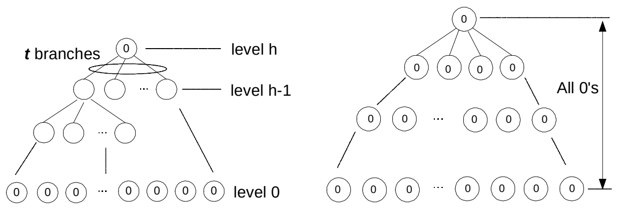
SPIHT has a special syntax to represent this context: four branched descendants are all insignificant conditioned that the root node is significant. And the syntax requires only one symbol, or one bit (output '0' for $D(i, j)$ in the original article). Another rather complicated syntax is defined to represent the tree having zeros for all descendants of the four offspring. This also takes only one bit, '0' (output '0' for $L(i, j)$ in the original article). Here is the room where EZW leaves for further compression, which was first discovered by the SPIHT algorithm.

3. DEGREE-K ZEROTREE

We establish definitions and theorems regarding the efficacy of zerotrees, which can formally explain the difference of zerotree coding power between popular EZW, SPIHT and possibly other zerotree-based algorithms. Viewing a zerotree as a entropy coding scheme, we classify it into different cases depending on the fullness of zeros in each level. They also tell the possibility of further improvement in compression that uses the zerotrees of wavelet coefficients as in EZW and SPIHT.

Through all following definitions, theorems, and proofs, assume that each zerotree is a height h, t-ary (i.e. having t branches), and complete (i.e. full leaves) tree (See Figure 1 (a)). And let us call this tree a *source tree* or a *source zerotree*.

The level 0 of a tree indicates the leaf nodes (i.e. the bottom) and the level h indicates the root node (i.e. the top). Note that the level is numbered starting from the bottom level. Thus, a height h tree has $h+1$ levels, i.e. level 0 to level h. And let $T = \sum_{i=0}^{h} t^i$, i.e. the total number of nodes in the tree.



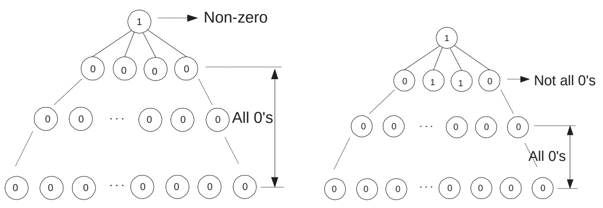
(a) A height h, t-ary tree (b) degree-0 zerotree

Fig. 1. A height h, t-ary tree and degree-0 zerotree

Each node of the tree is associated to a binary number (i.e. 0 or 1) representing significant or not of a wavelet coefficient with respect to a given threshold. Each node value is the significance of a wavelet coefficient for a given threshold. Representing each node as a random variable X with values 0 and 1 equally probable, we need N bits to encode a sequence of N nodes, $X_0, X_1, \cdots, X_{N-1}$. Thus, for a height α, t-ary subtree, the required number of bits to code it is equal to the total number of nodes in the subtree, i.e. $\frac{t^{\alpha}-1}{t-1}$ (bits).

Definition 1. *For any complete k-ary tree with height h, the level 0 indicating the leaf nodes and the level h indicating the root node, if all nodes from the bottom level (i.e. the level 0) to the level $(h-k)$ have zero values, we call the tree 'degree-k zerotree'. In other words, all nodes except top k levels have zero values in a degree-k zerotree.*

So, the degree-0 zerotree is the tree having all zeros. The zerotree shown in Figure 1 is a degree-0 zerotree. The



(a) degree-1 zerotree (b) degree-2 zerotree

Fig. 2. Degree-1 and degree-2 zerotrees

degree-1 zerotree (shown in Figure 2 (a)) is the tree having all zeros except the root node. And, the degree-2 zerotree (shown in Figure 2 (b)) is the tree having all zeros except the root node and the children nodes of root node.

Table 1 shows how EZW and SPIHT differently code the degree-1 and degree-2 zerotree sources. As discussed in [1], the EZW coder has four symbols: PS, NS (Positive and Negative Significant), ZTR (Zerotree), and IZ (Isolated Zero). The second symbol '1' (bold faced) in SPIHT's code for degree-2 example informs that there does not exist a degree-1 zerotree. The last symbol '0' (bold faced) in SPIHT's code for degree-2 example informs that there exists a degree-2 zerotree. Note that more than 1 bit is required to code each symbol of EZW without entropy coding.

Table 1. Example of coded symbols generated by EZW and SPIHT for degree-1 and degree-2 zerotree

	EZW	SPIHT
degree-1 zerotree in Fig. 2 (a)	*PS,ZTR,ZTR,ZTR,ZTR*	*1,0*
degree-2 zerotree in Fig. 2 (b)	*PS,ZTR,PS,PS,ZTR, IZ,IZ,IZ,IZ,IZ,IZ,IZ,IZ*	*1,**1**,0,1,1,0,**0***

Now we derive the rule which can be generally applied to an image coding algorithm based on zerotrees of wavelet coefficients. Basically, without using a zerotree symbol, a degree-k zerotree in a height h, t-ary source tree is coded by two parts:

1. Non-zero part : Code all symbols from top (root) level (i.e. level h) to level $h-k$, which are not all zeros. The number of the symbols is $\sum_{i=0}^{k-1} t^i$. By the definition of degree-k zerotree, at least one node from level $h-k$ is non-zero (i.e. one). These symbols can be modeled as a sequence of random variables, i.e. $X_0, X_1, \cdots, X_{(\sum_{i=0}^{k-1} t^i)-1}$. Since 0 and 1 are assumed to be equally probable, $\sum_{i=0}^{k-1} t^i$ bits are required to represent this sequence.
2. Zero part : Code all symbols from level k to bottom level (i.e. level 0), which are all zeros. The number of the symbols is $\sum_{i=k}^{h} t^i$. Since we assumed 0 and

1 are equally probable, $\sum_{i=k}^{h} t^i$ bits are required to represent this sequence of zeros.

However, if we use a zerotree symbol, the zero part can be coded by only one symbol.

The number of bits saved by the use of a degree-k zerotree symbol is the number of nodes from level $h-k$ to bottom level in the zerotree minus one for the zerotree root (symbol).

Definition 2. *In representing a degree-k zerotree of height h and t-ary branches, the bit savings S_k by using a degree-k zerotree symbol is simply:*

$$S_k = \sum_{i=k}^{h} t^i - 1 = \frac{t^{h+1} - t^k}{t-1} - 1\,(\textit{bits})$$

Noting that $S_0 - S_1 = 1\,$(bits), the difference of bit savings between degree-0 zerotree and degree-1 zerotree is only one bit.

Definition 3. *A degree-k zerotree coder is a zerotree coder which can represent all zerotrees with degree-i, $0 \leq i \leq k$.*

By the defintion of 'degree-k zerotree coder' above, the degree-2 zerotree coder, as an example, can code all degree-0, degree-1, and degree-2 zerotrees. Hence, it is more powerful coder than both degree-0 and degree-1 zerotree (See Figure 3).

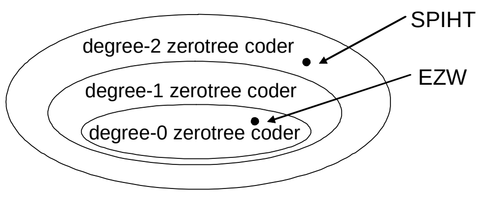


Fig. 3. Relationship of coding powers among degree-$0, 1, 2$ zerotree coders

Examples of coded bitstream for degree-0,1,2 zerotree sources by degree-0, 1, 2 zerotree coders are demonstrated in Table 2.

Theorem 1. *For coding a degree-k_1 zerotree source, the maximum bit savings of degree-k_2 zerotree coder from degree-k_1 zerotree coder with $k_1 < k_2$ is:*

$$t^{k_2-k_1} - 1\,(\textit{bits}).$$

Proof. Since $k_1 < k_2$, a degree-k_1 zerotree coder has less coding power than a degree-k_2 zerotree coder. Thus, a degree-k_1 zerotree coder rooted at top level h cannot represent the degree-k_2 zerotree. Instead, by having the roots of degree-k_1 zerotree coders at level $h-(k_2-k_1)$, a degree-k_2 zerotree can be represented by degree-k_1 zerotree coders. The number of these degree-k_1 zerotree coders minus one equals the bit savings since each additional zerotree coder will occupy one more symbol.

If $k_1 = 0$, degree-0 zerotrees rooted at level $h - k_2$ are coded by degree-0 zerotree coders rooted at level $h - k_2$. Similarly, if $k_1 = 1$, degree-1 zerotrees rooted at level $h - k_2 - 1$ are coded by degree-0 zerotree coders rooted at level $h - k_2 - 1$. In this way, degree-k_1 zerotrees rooted at level $h - k_2 - k_1$ are coded by degree-0 zerotree coders rooted at level $h - k_2 - k_1$. The numbers of these additional zerotree coders rooted at level $h - k_2 - i$, $i = 0, 1, k_1$, for each case above are simply: t^{k_2}, t^{k_2-1}, and $t^{k_2-k_1}$, respectively. Figure 4 shows that a degree-k_2 zerotree is coded by $t^{k_2-k_1}$ degree-k_1 zerotree coders (shaded part). $\square$

Table 2. Example of coded bitstream by degree-0, 1, 2 zerotree coders (The d_i means degree-i. The 0_i or 1_i indicates the occurrence of degree-i zerotree, 0_i for no and 1_i for yes. The d_0, d_1, d_2 source zerotrees correspond to Figures 1(b), 2(a), and 2(b), respectively.)

	source zerotree		
zerotree coder	d_0	d_1	d_2
d_0 coder	0_0	$1_0 1 0_0 0_0 0_0 0_0$	$1_0 1 0_0 1_0 1 0_0 0_0 0_0 0_0$ $1_0 1 0_0 0_0 0_0 0_0 0_0$
d_1 coder	0_0	$1_0 1 0_1$	$1_0 1 0_0 1_0 1 0_1 1_0 1 0_1 0_0$
d_2 coder	0_0	$1_0 1 0_1$	$1_0 1 1_1 0_2 0110$

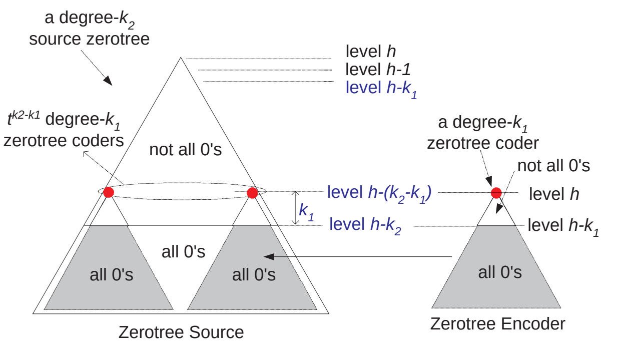


Fig. 4. A degree-k_2 zerotree coded by $t^{k_2-k_1}$ degree-k_1 zerotree coders

From the above definitions and theorems, our analysis on the performance difference between EZW and SPIHT is: EZW algorithm is only using degree-0 zerotrees, while SPIHT is using degree-1 and degree-2 zerotrees as well. The relationship of coding powers between EZW and SPIHT is shown in Figure 3. The $D(i,j)$ of type A or B in SPIHT correspond to the occurrence of degree-1 or degree-2 zerotrees, respectively.

Ideally, if we found higher degree zerotree coder, i.e. degree-m, $m > 2$, better coding performance would be expected. The hurdle for this, however, is the increased complexity in the implementation of the set partitioning engine. Also, since the number of wavelet decompositions, or equivalently the height of a spatial orientation tree, is usually not more than 5 $\sim$ 7, zerotrees of degree greater than 2 seldom occur. The authors leave the proof of this claim as an open problem.

4. EXPERIMENTAL ANALYSIS

Tables 3 and 4 show the distribution of degree-1 and degree-2 zerotrees coded in SPIHT, for the 512 $\times$ 512 Lenna at 1.0 bpp with 8 levels of wavelet decomposition. Each entry indicates the number of zerotrees for the specific bitplane and zerotree height. The bottom level of every zerotree is located at the highest resolution subbands, i.e. resolution level 0 which is not shown in both Tables 3 and 4. Note that bitplane 12 is MSB and bitplane 0 is LSB. The bitplane i represents the significance map with threshold 2^i. In fact, the height of a zerotree equals the resolution level of the zerotree root, for 0 being the highest resolution and 8 being the lowest resolution. For the Lenna image, the minimum threshold for the bitrate 1.0 bpp is $2^2 = 4$, which corresponds to bitplane 2. Thus, the two least insignificant bitplanes 0 and 1 are not coded.

An important observation is that the occurrence of degree-2 zerotrees is as frequent as that of degree-1 zerotrees, as shown in Tables 3 and 4. This proves the idea that degree-2 zerotree coder is superior to degree-1 zerotree coder since degree-2 zerotree coder can directly encode degree-2 zerotree with just one symbol which degree-1 zerotree coder will need three more symbols to represent degree-2 zerotree for 4-ary source tree (i.e. quadtree). From this example, it is certain that the higher degree zerotree coder is more powerful since there apparently exist higher degree zerotree sources.

The trends of zerotree behaviour at lower bitrates (i.e. < 1.0 bpp) will be very similar since the less significant bitplanes are simply not coded at lower rates coding.

5. CONCLUSION

We have tried to quantify the coding power of zerotrees of wavelet coefficients. A *degree-k zerotree* means the tree with all zero values except top k levels. And the *degree-k zerotree coder* means the source tree coder which can encode degree-i zerotrees, $0 \leq i \leq k$. Thus, the higher degree zerotree coder will have more coding power.

Based on this model, we classify the popular image coders EZW and SPIHT as examples, and this leads to an answer to the question of why SPIHT is better than EZW. It is because

Table 3. Distribution of degree-1 zerotrees in Lenna coded by SPIHT, decomposition level = 8

	Height of zerotree (i.e. resolution level of zerotree root)							
	8	7	6	5	4	3	2	1
bitplane 12	3	2	0	0	0	0	0	0
bitplane 11	0	3	8	0	0	0	0	0
bitplane 10	0	0	20	19	0	0	0	0
bitplane 9	0	0	14	44	35	0	0	0
bitplane 8	0	0	7	49	152	62	3	0
bitplane 7	0	0	4	31	218	374	98	0
bitplane 6	0	0	2	19	175	599	689	12
bitplane 5	0	0	0	9	141	609	1380	678
bitplane 4	0	0	0	2	89	605	1800	2442
bitplane 3	0	0	0	0	16	482	2465	6885
bitplane 2	0	0	0	0	0	10	78	299
bitplane 1	0	0	0	0	0	0	0	0
bitplane 0	0	0	0	0	0	0	0	0

Table 4. Distribution of degree-2 zerotrees in Lenna coded by SPIHT, decomposition level = 8

	Height of zerotree (i.e. resolution level of zerotree root)							
	8	7	6	5	4	3	2	1
bitplane 12	0	1	0	0	0	0	0	0
bitplane 11	1	3	4	0	0	0	0	0
bitplane 10	0	3	13	5	0	0	0	0
bitplane 9	0	0	19	34	13	0	0	0
bitplane 8	0	0	7	43	86	25	1	0
bitplane 7	0	0	2	18	146	232	38	0
bitplane 6	0	0	2	15	119	403	451	0
bitplane 5	0	0	3	7	103	480	1180	0
bitplane 4	0	0	0	4	82	500	1879	0
bitplane 3	0	0	0	0	36	377	2229	0
bitplane 2	0	0	0	0	0	4	40	0
bitplane 1	0	0	0	0	0	0	0	0
bitplane 0	0	0	0	0	0	0	0	0

EZW is a degree-0 zerotree coder and SPIHT is a degree-2 zerotree coder. SPIHT can encode degree-1 or 2 zerotrees by one symbol for each, while EZW will need three more symbols for each SPIHT symbol.

6. REFERENCES

[1] J. M. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficient," *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 Trans. on Circuits and Systems for Video Technology*, vol. 6., pp. 243-250, June 1996.

[3] W. A. Pearlman, "Trends of Tree-Based, Set Partitioning Compression Techniques in Still and Moving Image Systems," *Proceedings Picture Coding Symposium 2001*, Apr., 2001, pp. 1-8. (Invited, keynote paper)

[4] J. E. Fowler, "QccPack: An Open-Source Software Library for Quantization, Compression, and Coding," in *Applications of Digital Image Processing XXIII*, A. G. Tescher, Ed., San Diego, CA, August 2000, Proc. SPIE 4115, pp. 294-301.