NON-MEMORY-BASED AND REAL-TIME ZEROTREE BUILDING FOR WAVELET ZEROTREE CODING SYSTEMS

Dongming Peng and Mi Lu

Electrical Engineering Department, Texas A&M University, College Station, TX77843, USA

1 INTRODUCTION

The wavelet zerotree coding systems, including Embedded Zerotree Wavelet (EZW)[1] and its variants Set Partitioning In Hierarchical Trees (SPIHT)[2] and Space Frequency Quantization (SFQ)[3][4], have three common procedures: 1) 2-D Discrete Wavelet Transform (DWT)[5], 2) zerotree building and symbol generation from the wavelet coefficients (illustrated in Figure 1), and 3) quantization of the magnitudes of significant coefficients and entropy coding, where the second procedure is an important one. All recently propsed architectures ([6]-[9]) for wavlelet zerotree coding use memories to build zerotrees. In this paper we contribute to building the zerotrees in a non-memory-based way with real-time performance leading to the decrease of hardware cost and the increasement of processing rate which is especially desirable in video coding. One of our main ideas is to rearrange the DWT calculations taking advantage of parallel and pipelined processing so that *any* parent coefficient and its children coefficients in zerotrees are guaranteed to be calculated and outputted simultaneously.

2 THE ARCHITECTURE FOR REARRANGING 2-STAGE 2-D DWT

2.1 Two Priliminary Devices Used in the Architecture for Rearrangement

(1)The Processing Unit(PU) shown as in Figure 2 rearranges the calculation of wavelet filtering so that the filter is cut to half taps based on the symmetry between the negative and positive wavelet filter coefficients. x, a and c are the input sequence, low- and high-pass filtering output sequence respectively. While a datum of sequence x is fed and shifted into the PU per clock cycle, a datum of a is calculated every even clock cycle and a datum of c is calculated every odd clock cycle. The PU in Figure 2(a) can be extended to a parallel format as in Figure 2(b) where if a number of data from sequence x x_{k+8}, x_{k+7}, ..., x_{k} are fed to the PU in parallel at a clock cycle, then x_{k+9}, x_{k+8}, ..., x_{k+1} are fed at the next cycle. (2)In Figure 3 the TU is a systolic array with $(\mathrm{N}_w+3)\times\mathrm{N}$ cells, where

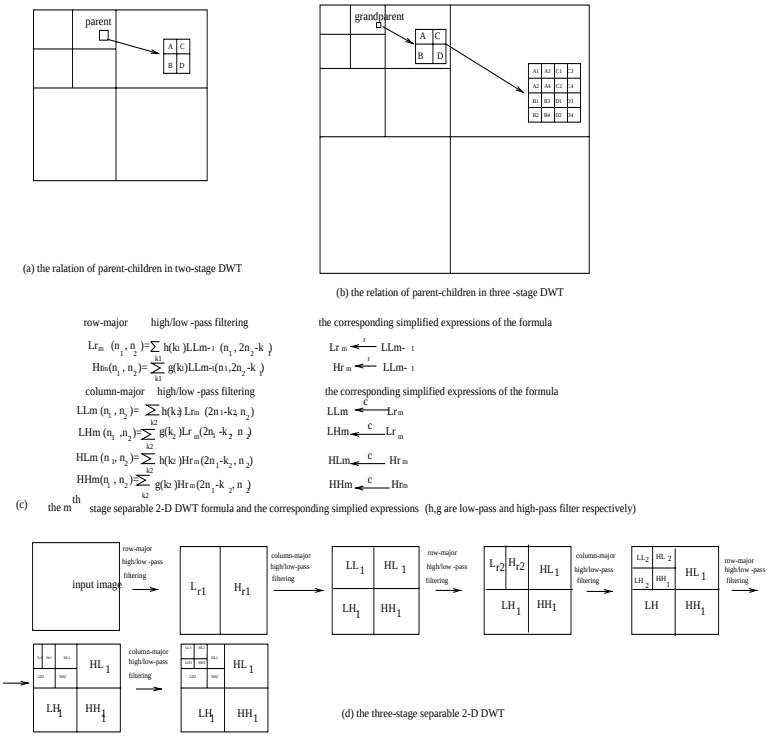


Fig. 1. the algorithms of EZW and 2-D DWT

N_w is the width of the wavelet filter and N is the width or the height of the input (square) image to DWT. N is hundreds or thousands of times greater than N_w for most applications of 2-D wavelet transforms (e.g. image/video systems). A cell transfers its content to the next adjacent cell once it receives a datum from its preceding cell. The leftmost cells in odd rows and rightmost cells in even rows have output ports and copy their data to outside. The upper-left cell has an input port and the TU uses it to recieve the input sequence. An element in matrix X is fed to the TU per clock cycle in the order according to the indices in Figure 3(b). The TU's (N_w+3) outputs and its newly arrived element belong to the same column in X. An example for the positions of the X's elements in TU after 3N clock cyles of inputting X is illustrated in Figure 3(c).

2.2 the Proposed Architecture and the Analysis of Its Operations

The architecture for the rearrangement of DWT is proposed in Figure 4, and the corresponding timing of operations is presented in Figure 5. Every four sibling coefficients in the first decomposition stage are designed to be calculated together (meanwhile their parent is generated by PU_3).

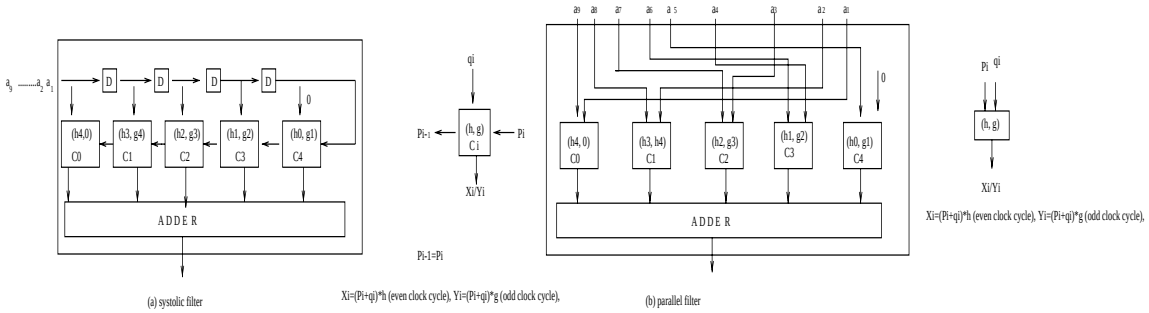


Fig. 2. the structures of PU (Processing Unit)

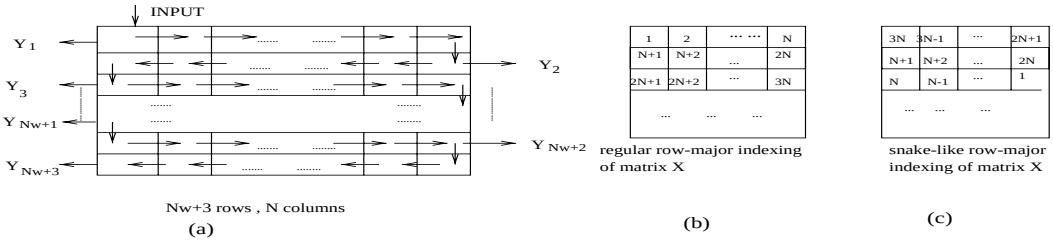
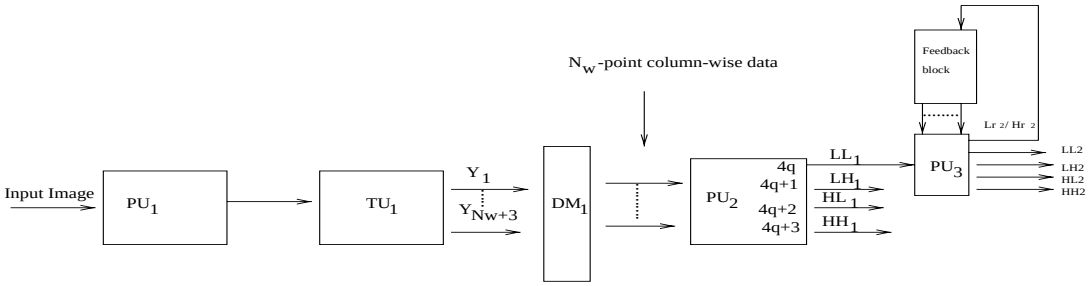


Fig. 3. A new transpose unit (TU)

Due to the row-major dyadic subsampling, the row-major high/low-pass filtering is alternatively executed by the PU_1 in Figure 4 point by point in each row. Based on similar column-major dyadic subsampling, the PU_2 takes turns to execute column-major high/low-pass filtering and selects appropriate inputs from TU_1 to generate four sibling coefficients consecutively. The order to calculate the siblings is as A, then B, C and D in the example of four siblings illustrated in Figure 1(a). After PU_2's calculation of point A by taking Y_1, ..., Y_{Nw} as inputs, PU_2 has to calculate B by taking Y_3, ..., Y_{Nw+2} as inputs, then PU_2 comes back to take Y_1, ..., Y_{Nw} to calculate C, then PU_2 takes Y_3, ..., Y_{Nw+2} again to calculate D.

In Figure 5 it can be seen that PU_2 calculates the same kind of column-major convolution (high-pass or low-pass filtering) during the period when a row of input image is sequentially fed to the system in N clock cycles, and the coefficients in four subbands LL_1, LH_1, HL_1 and HH_1 are calculated in turns in a longer period when four rows of input image are fed. The right column in Figure 5 is to show the operations of PU_3 in Figure 4. In PU_3 the second stage of DWT is completed and the parent coefficients are generated. PU_3 takes sequential input of LL_1 coefficients from PU_2 to perform row-major filtering in the first quarter of the period during which four rows of input image are fed to the system. Then in the next three quarters (i.e., i=4q+1, 4q+2 or 4q+3 in Figure 5), PU_3 performs the column-major convolutions by taking the result from the first quarter's row-major filtering as input.



1) DM1 is a demultiplexer that select Nw-point data from Nw+3 outputs of TU1.

2) PU2 is a paralle filter as in Figure 2(b) and has four output ports active at different time.

3) PU3 is the hybrid version of PU that can take either sequential or parallel inputs.

4) Feedback block consists of 2 separate TUs and Demultriplexers to select Lr2 / Hr2 into respective TU and to select outputs from 2 TUs into PU3.

Fig. 4. The architecture for two-stage DWT and the zerotree building

3 THE DESIGN EXTENDED TO GENERAL STAGES OF DWT

Now we consider what should be done to modify the architecture in Figure 4 for m stages of wavelet decomposition. Because the input image is fed into the system in the same way as before, the first stage row-major high/low-pass filtering is still performed in PU_1 alternatively as designated in Figure 5.

Regarding the first stage column-major high/low-pass fitering performed in PU_2, we note that there are 4^{m-1} coefficients in the first stage decomposition corresponding to the same ancestor in the last stage (stage m) decomposition. To satisfy the restriction of generating parent and children simultaneously, it is required that these 4^{m-1} "kindred" coefficients be calculated together. (Meanwhile these coefficients' parents in the intermediate stages of decompostion should be calculated together too.) Note that these 4^{m-1} coefficients are located in 2^{m-1} adjacent rows and 2^{m-1} adjacent columns in their subband. PU_2 should alternatively select appropriate inputs among 2^{m-1} different groups of parallel column-major data from TU_1, and perform column-major filtering to generate the 4^{m-1} kindred coefficients in turns, where the coefficeints calculated with the same group of input belong to the same row. Accordingly, TU_1 is an extended version in Figure 3(a) and is supposed to have output ports Y_1, ..., Y_{Nw+M} with M equal to 2^m.

PU_3 carries out the rest computation in DWT. The second stage decomposition is achieved as follows. In the first quarter of the period when 2^m rows of input image are fed to the system, PU_3 gets its inputs, i.e., the coefficients in LL_1 subband from TU_1, and alternatively performs the second stage low/high-pass row-major convolution. The calculated results, or the coefficients in Lr_2 and Hr_2 are stored in two TUs hidden in PU_3's feedback block. In the second quarter, the Lr_2 coefficients are fed back to PU_3 to be column-major filtered to get the results in LL_2 and LH_2. In the third and fourth quarter, the Hr_2 points

are fed back to PU_3 to be used to calculate out HL_2 and HH_2 respectively. The PU_3 achieves further stage decompositions in the available intervals during its execution of the second stage decomposition.

By reason of limit space in this paper, the operations of processors are described with basic principles and not with many details. To sum up, TU_1 is modified to have (N_w+M) rows; PU3's feedback block has changed a little bigger so that it holds N_w+M rows of coefficients in the results of row-major wavelet filtering; and the switches have become complicated to select appropriate data at diffrent time.

4 PERFORMANCE ANALYSIS AND CONCLUSION

Since N_w (the width of wavelet filters) is far less than N (the width or length of input image) and the size of boudary effect of wavelet transforms is only dependent on N_w, in this paper we ignore the boudary effect to simplify our expressions knowing that it can be resolved by a little adjustment in either timing or architecture. The area of the proposed architecture is dominated by PUs and TUs. A PU contains pN_w MACs (Multiplyer and Accumulator Cell), where p is the number of precision bits of data, thus three PUs contain $3pN_w$ MACs. Because a TU is necessary for the column-major filtering in every stage decomposition, and the number of cells in TU at the i^{th} stage decomposition is $(N/2^i)(N_w+2^i)$, where the first item is the length of a row and the second item is the number of rows in the TU, the total area for TUs is $O(mN+N_wN)$, where m is the number of stages in DWT. Note that the TUs except TU_1 are hidden in PU_3's feedback block. Thus the whole area of the architecture for m stage DWT is $\mathbf{A}$=$O(pNN_w+pNm)$. The input image is assumed to be fed with one pixel per clcok cycle. The system's latency (execution time) $\mathbf{T}$ is N^2 clock cycles. Thus the product of $\mathbf{A}$ and $\mathbf{T}$ for the system is $O(pN^3(N_w+m))$, where N^2 is the input size of the algorithm. Our proposed architecture is comparable to conventional DWT architectures ([10]-[14]) in the aspect of area, latency, the product of them, or the hardware utilization, even though not only the DWT but also the zerotree building is achieved in this architecture.

We have proposed a non-memory-based design in which the input image is recursively decomposed by DWT and zerotrees are built in real-time. The computation of wavelet-based zerotree coding is strongly featured by the computation locality in that the calculations of coefficients on a certain zerotree (only) depend on the same local sub-area of the 2-D inputs. This desirable feature has been exploited in this paper by calculating children and their parent simultaneously in the rearranged DWT, so that most intermediate data need not be held for future calculations.

References

1. J.M. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, *IEEE Transactions on Signal Processing,* Volume: 41, 1993, Page(s): 3445 -3462.

2. A. Said, W.A. Pearlman, A new, fast, and efficient image codec based on set partitioning in hierarchical trees, *IEEE Transactions on Circuits and Systems for Video Technology,* Volume: 6, June 1996, Page(s): 243 -250.
3. Zixiang Xiong, K. Ramchandran, M.T. Orchard, Wavelet packet image coding using space-frequency quantization, *IEEE Transactions on Image Processing,* Volume: 7, June 1998, Page(s): 892 -898.
4. Zixiang Xiong, K. Ramchandran, M.T. Orchard, Space-frequency quantization for wavelet image coding, *IEEE Transactions on Image Processing,* Volume: 6, May 1997, Page(s): 677 -693.
5. M. Vetterli, J. Kovacevic, Wavelets and Subband Coding, *Prentice Hall,* 1995.
6. Li-Minn Ang, Hon Nin Cheung, K. Eshraghian, VLSI architecture for significance map coding of embedded zerotree wavelet coefficients, *Proceddings of 1998 IEEE Asia-Pacific Conference Circuits and Systems,* 1998. Page(s): 627 -630.
7. Jongwoo Bae, V. K. Prasanna, A fast and area-efficient VLSI architecture for embedded image coding, *Proceedings of International Conference on Image Processing*, Volume: 3, 1995, Page(s): 452 -455.
8. J.M. Shapiro, A fast technique for identifying zerotrees in the EZW algorithm, *Proceedings of 1996 IEEE International Conference on Acoustics, Speech, and Signal Processing,* Volume: 3, 1996, Page(s): 1455-1458.
9. J. Vega-Pineda, M. A. Suriano, V. M. Villalva, S. D. Cabrera, Y.-C. Chang, A VLSI array processor with embedded scalability for hierarchical image compression,*1996 IEEE International Symposium on Circuits and Systems,* Volume: 4, 1996, Page(s): 168 -171.
10. Jer Min Jou, Pei-Yin Chen, Yeu-Horng Shiau, Ming-Shiang Liang, A scalable pipelined architecture for separable 2-D discrete wavelet transform, *Design Automation Conference, Proceedings of the ASP-DAC '99. Asia and South Pacific,* Volume: 1, Page(s): 205 -208.
11. M. Vishwanath, R. M. Owens, M. J. Irwin, VLSI architectures for the discrete wavelet transform, *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing,* Volume: 42, May 1995, Page(s): 305 -316.
12. Chu Yu, Sao-Jie Chen, Design of an efficient VLSI architecture for 2-D discrete wavelet transforms, *IEEE Transactions on Consumer Electronics, Volume: 45, Feb. 1999, Page(s): 135 -140.*
13. V. Sundararajan, K. K. Parhi, Synthesis of folded, pipelined architectures for multidimensional multirate systems, *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing,* Volume: 5, 1998, Page(s): 3089 -3092.
14. Chu Yu, Sao-Jie Chen, VLSI implementation of 2-D discrete wavelet transform for real-time video signal processing, *IEEE Transactions on Consumer Electronics,* Volume: 43, Nov. 1997, Page(s): 1270-1279.
15. T. Acharya, Po-Yueh Chen, VLSI implementation of a DWT architecture, *Proceedings of the 1998 IEEE International Symposium on Circuits and Systems,* Volume: 2, 1998, Page(s): 272 -275.

```
real-time Alg. DWT amenable to EZW
{
  for i=0 to N-1   /* row */
    for j=0  to N-1 /* column */
      Do DWT(i,j)
}

DWT(i,j)    /* q,s are any non-negative integers */
{
```

	what PU_1 does	what PU_2 does	what PU_3 does
if i=4q			
if j=even	$Ln_1 \xleftarrow{r} x$	$LL_1 \xleftarrow{c} Ln_1$	
if j=odd	$Lr_1 \xleftarrow{r} x$	$LL^1_1 \xleftarrow{c} Lr^1_1$	
if j=4s			$Lr_2 \xleftarrow{r} LL_1$
if j=4s+1			$Hr_2 \xleftarrow{r} LL_1$
if j=4s+2			$Lr^1_2 \xleftarrow{r} LL^1_1$
if j=4s+3			$Hr^1_2 \xleftarrow{r} LL^1_1$
if i=4q+1			
if j=even	$Ln_1 \xleftarrow{r} x$	$LH_1 \xleftarrow{c} Lr^+_1$	
if j=odd	$Lr_1 \xleftarrow{r} x$	$LH^1_1 \xleftarrow{c} Lr^{1+}_1$	
if j=4s			$LL_2 \xleftarrow{c} Ln_2$
if j=4s+2			$LH_2 \xleftarrow{c} Lr^+_2$
if i=4q+2			
if j=even	$Ln_1 \xleftarrow{r} x$	$HL_1 \xleftarrow{c} Hr_1$	
if j=odd	$Hr_1 \xleftarrow{r} x$	$HL^1_1 \xleftarrow{c} Hr^1_1$	
if j=4s+2			$HL_2 \xleftarrow{c} Hr_2$
if i=4q+3			
if j=even	$Ln_1 \xleftarrow{r} x$	$HH_1 \xleftarrow{c} Hr^+_1$	
if j=odd	$Hr_1 \xleftarrow{r} x$	$HH^1_1 \xleftarrow{c} Hr^{1+}_1$	
if j=4s+2			$HH_2 \xleftarrow{c} Hr^+_2$

Notation:

(1).The meaning of superscript "k" is explained as the following,(k is an integer)

Using the simplified expression as in figure 1(c),we call the calculation of $\xleftarrow{r}$ as "r" arrow; $\xleftarrow{c}$ as "c" arrow.

Suppose A corresponds to a part of a component in figure 1 (d). If row-wise signal A and A^k are on the right side of "r" arrow, and A is fromthe ith row,then A^k is from the kth row.

If column-wise signal A and A^k are on the right side of "c" arrow, and A is from $Y_i,.....Y_{Nw+i}$ (See TU's output in figure 3), then A^k is from $Y_{i+2k},.....Y_{Nw+2k}$ (in the same column)

If A and A^k are on the left side of "c" arrow or "r" arrow,they are corresponding to 2 coefficients inthe same column and in ith row and (i+k)th row respectively.

The meaning of "+" is explained as the following:

Column-wise signal A ,$A^+$and $A^{k+}$are on the right side of "c" arrow. A is from $Y_i,.....Y_{Nw+i}$ (see TU's output in figure 3) the A^{k+} is from $Y_{i+1},....Y_{Nw+i+1}$, A^{k+} is from $Y_{i+2k+1},.....Y_{Nw+i+2k+1}$ (they are in the same column)

(2) Because of PU's feature of alternative high/low -pass filtering,we use A and A^1 alternative to get column-wise low-high pass outputs.

(3) The reason why we use A and A^1 (A standing for Lr_1, LL_1) alternatively to generate B and B^1 (B standing for LL_1, LH_1 ,.....) is based on the restriction that any siblings be generated consecutively. Because of dyadic downsamplings for column-wise convojution. $B^1 \xleftarrow{C} A^1$ if $B \xleftarrow{C} A$

(4) Lr_2 and Hr_2 are fed in 2 seperate TUs in PU3's feedback block. The TUs can hold Nw+2 rows of data at most. Careful readers may find that $LL \xleftarrow{c} Lr$ and $LH \xleftarrow{c} Lr$ are executed alternatively point by point, however, $HL_2 \xleftarrow{c} Hr_2$ and $HH_2 \xleftarrow{c} Hr_2$ are executed alternatively row-by row. This paradox can be resolved by a little manipulation in 2 TUs in feedback block . Anyway, during the time of 4q+1<=i<=4q+3, no new data are generated for Lr_2 or Hr_2 ,so the old data in feedback block can be held until i=4q+4.

(5) This real time algorithm dictates the operations happening in figure 4. The control signals for PU ,TU ,DM ,PU ,PU , feedback block can be easily implemented locally and periodically. Their details are not discussed because of the limited length of this paper.

Fig. 5. The timing for the operations in the architecture for two-stage DWT and the zerotree building