The Advanced OCA for 2-D Discrete Periodized Wavelet Transformation

King-Chu Hung, Yu-Jung Huang, Jyh-Ming Kuo, and Trieu-Kien Truong

Abstract—In this correspondence, we develop a novel computation method for the 2-D discrete periodized wavelet transformation (DPWT). The new algorithm is based on the operator correlation algorithm (OCA). Compared with the classical 2-D pyramid algorithm, the advanced operator correlation algorithm (AOCA) has two major advantages; it requires half the number of multiplications and can yield the same output SNR with half the precision of the DPWT filter coefficients. Furthermore, the modular structure of the AOCA makes it particularly suitable for a VLSI implementation.

Index Terms—AOCA, DPWT, OCA, perfect reconstruction, real-time algorithm, VLSI.

I. INTRODUCTION

The two-dimensional (2-D) discrete wavelet transformation (DWT) [1] is a powerful tool for digital image analysis. For many applications, such as image coding and compression [2], digital image processing [3], and scene analysis [4], the inverse wavelet transformation requires perfect reconstruction (PR), that is, the reconstructed image from the inverse 2-D DWT should be identical to the input image of the 2-D DWT. To obtain perfect reconstruction, an image has to be represented by periodic discrete signals. This periodicity requires that periodized wavelets [5] are used in the 2-D DWT. The use of periodized wavelets in the 2-D DWT is denoted as the 2-D discrete periodized wavelet transformation (DPWT). The continuous wavelets defined in $L^2(R^2)$ [6] are not suitable for this representation.

Due to the complexity of the 2-D DPWT, several approaches have been proposed for its efficient computation. The separable computation of the 2-D DPWT is based on a two-stage decomposition, using the one-dimensional (1-D) DPWT in the horizontal and vertical directions, respectively. However, the conventional pyramid algorithm (PA) [7], using separable computation, will cause a very long latency and requires $l(l + 1)$ multiplications to derive each 2-D DPWT coefficient, where $l$ denotes the length of the discrete wavelet filter. The recursive PA (RPA) for 1-D DPWT [8], using the short-length FIR filtering algorithm [9], was proposed to meet the requirement of real-time computation. As applied to the 2-D DPWT [10], the 2-D RPA using the separable computation approach can effectively reduce the latency to 1 and needs only $3(l + 1)/4$ multiplications to compute each 2-D DPWT coefficient. However, the 2-D RPA will cause a time lag problem when perfect reconstruction is required. In general, to meet the perfect reconstruction requirement, the separable computation approach for the 2-D DPWT needs to compute the boundary elements of transformation filters. This may result in increased computational complexity and time lag problems.

Although the time lag problem can be solved by using homeomorphic highpass and lowpass filters, traditional homeomorphic filters cannot simplify the boundary data processing problem. For real-time processing, all the elements in each band of the 2-D DPWT coefficient matrices should be derived at the same sequence as the scanned pattern of its input image. Unfortunately, using the lowpass filter $H$ and the highpass filter $G$ defined in [11], the computation of the boundary coefficient $(0, 0)$ relative to the coefficient $(i/2) - 1$, $(i/2) - 1$ will be delayed until all the coefficients in early $N - i$ rows of the input image matrix in each band are obtained, where $N$ is the input image size. This delay makes these two filters unsuitable for real-time application of the 2-D DPWT.

In this correspondence, general cases of homeomorphic filters are studied in order to find a highpass filter suitable for real-time 2-D DPWT processing. Moreover, by using the AOCA for the 2-D DPWT, this filtering process can reduce the number of multiplications by half. Furthermore, for the same output SNR, the filters in AOCA require half the bit precision of the filters needed for the 2-D PA.

In next section, the 2-D DPWT is briefly reviewed. The 2-D OCA is presented in Section III. The AOCA and its finite precision analysis are presented in Section IV. A conclusion is given in Section V.

Equation (3) is the iterative form of the 1-D DPWT.

\begin{align}
\sum_{i=0}^{N-1} h_i^2 &= 1, \quad \sum_{i=0}^{N-1} h_i = \sqrt{2}, \quad \text{and} \quad \sum_{i=0}^{N-1} (-1)^i h_i = 0 \quad (1) \\
\hat{g}_i &= (1)^i h_{(1-i)} \quad (2)
\end{align}

where $(1 - i) \mod N$ denotes the residual of $(1 - i) \mod N$. For an $l$-tap wavelet filter, there are $N - i$ zeros in both $h$ and $g$.

Let $T$ be the $N \times N$ matrix defined by $T = [\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}]$, where $I_{N-1}$ is the $(N - 1) \times (N - 1)$ identity matrix. Matrix $T$ should have a first row with $(N - 1)$ zeros and a single one (the same is also true for its last column). Thus, the lowpass filter $H$ and the highpass filter $G$ for the 1-D DPWT can be derived by $H = [T^0 h, T^2 h, \cdots, T^{N-2} h, T^N h, T^{N+2} h, \cdots, T^{2N-2} h]$ and $G = [T^0 g, T^2 g, \cdots, T^{N-2} g, T^N g, T^{N+2} g, \cdots, T^{2N-2} g]$, respectively. Note that both $H$ and $G$ are $N \times N$ matrices.

Let $s_j = [s_0, s_1, \cdots, s_{N-1}]$ in $V_j$ be the original sampled data of a finite 1-D signal, where $V_j$ is a finite-dimensional multiresolution approximation subspace. For $j \in \{J, J + 1, \cdots, -1\}$, the 1-D DPWT coefficients $s_{j+1}$ and $d_{j+1}$ correspond to the projection of the 1-D signal onto subspace $V_{j+1}$ and its orthogonal complement subspace $W_{j+1}$, respectively. The elements $s_{j+1}$ and $d_{j+1}$ obtained from $s_j$ are defined as

\begin{align}
s_{j+1} &= s_j H & \text{and} & & d_{j+1} = s_j G. \quad (3)
\end{align}
The 2-D DPWT is usually performed using two 1-D DPWTs in a separable approach. For $J < j < 0$, let $ss_{j+1}$, $sd_{j+1}$, $ds_{j+1}$, and $dd_{j+1}$ denote the projection of the $j$th level 2-D DPWT matrix $ss_j$ onto the orthonormal bases in subspace $V_{j+1}$ and its orthogonal complement subspace $W_{j+1}$, respectively. The iterative form of the 2-D DPWT can be represented by

$$ss_{j+1} = H' ss_j H$$
$$sd_{j+1} = H' ss_j G$$
$$ds_{j+1} = G' sss_j H$$
$$dd_{j+1} = G' sss_j G$$

where $H'$ and $G'$ denote the transposes of $H$ and $G$, respectively. Equation (4) performs the 2-D DPWT decomposition from level $j$ to $j + 1$.

From (2), we observe that the lowpass filter coefficients have $l - 2$ points of time delay relative to the highpass filter coefficients. This time delay between these two filters will complicate the processing of the boundary data in 2-D DPWT. From (4), different boundary data in $ss_{j+1}$ are required to obtain the same boundary elements of $ss_{j+1}$, $sd_{j+1}$, $ds_{j+1}$, and $dd_{j+1}$. For example, for a four-tap wavelet filter, the computation of the boundary element $ss_{j+1}(0, 0)$ needs the 16 elements of $ss_j(u, v), u, v = 0, 1, 2, 3$, and the computation of boundary element $ds_{j+1}(0, 0)$ needs the 16 elements of $ss_j(u, v), u = 0, 1, 2, 3, v = 0, 1, N - 2, N - 1$.

### III. OPERATOR CORRELATION ALGORITHM FOR THE 2-D DPWT

Let $C(G)$ consist of the set of column vectors in the nonperiodic part of matrix $G$, such as $A$ is a linear space, and the operations of addition and scalar multiplication of the norm of $C(G)$ are continuous [12]. Given an integer $\lambda \in Z$, we can obtain a new continuous linear space $C(G^{2\lambda})$ consisting of the nonperiodic column vectors of the matrix $G^{2\lambda} = T^{2\lambda} G$. Therefore, we can define a continuous function $f$ mapping from $C(G)$ into $C(G^{2\lambda})$ such that for each element $d_{j+1} \in C(G)$, $f(d_{j+1}) = d_{j+1} T^{2\lambda} = d_{j+1} T^{2\lambda} G = d_{j+1} G^{2\lambda}$. Then, for each element $d_{j+1} \in C(G)$ and $G^{2\lambda}$, we obtain $d_{j+1} = f^{-1}(d_{j+1}) = d_{j+1} T^{-2\lambda} = d_{j+1} T^{-2\lambda} G = d_{j+1}$. Thus, for a given $\lambda \in Z$, the two spaces $C(G)$ and $C(G^{2\lambda})$ are linearly homeomorphic [12]. Replacing $G$ by $G^{2\lambda}$ in (4), we will obtain

$$ss_{j+1} = H' ss_j H$$
$$sd_{j+1} = H' ss_j G$$
$$ds_{j+1} = G' sss_j H$$
$$dd_{j+1} = G' sss_j G$$

where $G^{2\lambda}$ and $G^{2\lambda}$ denote the transposes of $G$ and $G$, respectively. Equation (5) is referred to as the modified 2-D DPWT. The 2-D DPWT is usually performed using two 1-D DPWTs in a separable approach. For $J < j < 0$, let $ss_{j+1}$, $sd_{j+1}$, $ds_{j+1}$, and $dd_{j+1}$ denote the projection of the $j$th level 2-D DPWT matrix $ss_j$ onto the orthonormal bases in subspace $V_{j+1}$ and its orthogonal complement subspace $W_{j+1}$, respectively. The iterative form of the 2-D DPWT can be represented by

$$ss_{j+1} = H' ss_j H$$
$$sd_{j+1} = H' ss_j G$$
$$ds_{j+1} = G' sss_j H$$
$$dd_{j+1} = G' sss_j G$$

where $H'$ and $G'$ denote the transposes of $H$ and $G$, respectively. Equation (4) performs the 2-D DPWT decomposition from level $j$ to $j + 1$.

From (2), we observe that the lowpass filter coefficients have $l - 2$ points of time delay relative to the highpass filter coefficients. This time delay between these two filters will complicate the processing of the boundary data in 2-D DPWT. From (4), different boundary data in $ss_{j+1}$ are required to obtain the same boundary elements of $ss_{j+1}$, $sd_{j+1}$, $ds_{j+1}$, and $dd_{j+1}$. For example, for a four-tap wavelet filter, the computation of the boundary element $ss_{j+1}(0, 0)$ needs the 16 elements of $ss_j(u, v), u, v = 0, 1, 2, 3$, and the computation of boundary element $ds_{j+1}(0, 0)$ needs the 16 elements of $ss_j(u, v), u = 0, 1, 2, 3, v = 0, 1, N - 2, N - 1$.

### IV. ADVANCED OPERATOR CORRELATION ALGORITHM (AOCA)

The operator matrices in (7) have the following properties:

1. $W_{LH}$, $W_{HL}$, and $W_{HH}$ are symmetric matrices. That is, $W_{LH} = W_{LH}^T$.
2. $W_{LH}$ and $W_{HL}$ consist of columns with the same absolute values of elements. In addition, these columns of these two matrices have opposite sequence in arrangement and opposite sign in the even columns. So do the pairs of $W_{HL}$ and $W_{HH}$.
iv) $\mathbf{W}_{L,L}$ and $\mathbf{W}_{H,H}$ consist of rows with the same absolute values of elements. In addition, those rows of these two matrices have opposite sequence in arrangement and opposite sign in the even rows. So do the pairs of $\mathbf{W}_{L,H}$ and $\mathbf{W}_{H,H}$.

To further simplify the computation of the 2-D DPWT coefficients, we introduce a parameterized function $a_{m,n}(t)$ as

$$a_{m,n}(t) = \begin{cases} \sum_{n=0}^{l-1} a_{m,n}(p+n) & 0 \leq v_1 \leq (N_j/2) - (l/2) \\ \sum_{n=0}^{l-1} a_{m,n}(p+n) & \sum_{n=2k+1}^{l-1} a_{m,n}(2a_{1,N_j} + n) \end{cases}$$

where $m,n \in \{0, 1, \ldots, I-1\}$, and $t = uN_j + v$, where $u,v$ are the vertical and horizontal coordinate indices of $s_{s_j}(u,v)$, respectively, and $N_j = 2^{-j}$ is the horizontal matrix size of $s_{s_j}$. Obviously, $N_j = N$ for $j = J$. Therefore, $s_{s_j}(t)$ is the raster scanning sequence of $s_{s_j}(u,v)$ at the $j$th level. For example, for a four-tap wavelet filter, only ten elements in (8) have to be computed in parallel.

For convenience, when computing the 2-D DPWT coefficients at the $j + 1$st level, we define another parameter $p = 2N_ju_1 + 2v_1$, where $v_1, u_1$ are the vertical and horizontal coordinate indices of $s_{s_{j+1}}(u_1, v_1)$, respectively. Then, two operators of row accumulation $r_{s_m}(p + l - 1)$ and $r_{d_m}(p + l - 1)$ for $m \in \{0, 1, \ldots, I-1\}$ can be defined as

$$r_{s_m}(p + l - 1) = \begin{cases} \sum_{n=0}^{l-1} a_{m,n}(p+n) & 0 \leq v_1 \leq (N_j/2) - (l/2) \\ \sum_{n=0}^{l-1} a_{m,n}(p+n) & \sum_{n=2k+1}^{l-1} a_{m,n}(2a_{1,N_j} + n) \end{cases}$$

$$r_{d_m}(p + l - 1) = \begin{cases} \sum_{n=0}^{l-1} (-1)^n a_{m,n-1}(p+n) & 0 \leq v_1 \leq (N_j/2) - (l/2) \\ \sum_{n=0}^{l-1} (-1)^n a_{m,n-1}(p+n) & \sum_{n=2k+1}^{l-1} (-1)^n a_{m,n-1}(2a_{1,N_j} + n) \end{cases}$$

where $k = 0, 1, \ldots, (l/2) - 2$. In other words, the pairs of $r_{s_m}(p + l - 1)$ and $r_{d_m}(p + l - 1)$ are obtained by accumulating the corresponding parameterized functions in opposite directions along rows of the matrix $s_{s_j}$. From these two sequences defined in (9) and (10), we can obtain the 2-D DPWT coefficients by accumulating the associative sequence elements in terms of $m$ parameters, i.e., the 2-D DPWT coefficients in matrices $s_{s_{j+1}}$ and $d_{s_{j+1}}$ at the $j + 1$st level can be obtained as

$$s_{s_{j+1}}(u_1, v_1) = \begin{cases} \sum_{n=0}^{l-1} r_{s_m}(p + mN_j + l - 1) & 0 \leq u_1 \leq (N_j/2) - (l/2) \\ \sum_{n=0}^{l-1} r_{s_m}(p + mN_j + l - 1) & \sum_{n=2k+1}^{l-1} r_{s_m}(2v_1 + (m - 2k - 2)N_j + l - 1) \end{cases}$$

$$d_{s_{j+1}}(u_1, v_1) = \begin{cases} \sum_{n=0}^{l-1} (-1)^m r_{s_{m-1}}(p + mN_j + l - 1) & 0 \leq u_1 \leq (N_j/2) - (l/2) \\ \sum_{n=0}^{l-1} (-1)^m r_{s_{m-1}}(t + mN_j + l - 1) & \sum_{n=2k+1}^{l-1} (-1)^m r_{s_{m-1}}(2v_1 + (m - 2k - 2)N_j + l - 1) \end{cases}$$

$$\left(\frac{N_j}{2} - (l/2 - 1) \leq u_1 \leq N_j/2 - 1 \right)$$

where $k = u_1 - (N_j/2) - (l/2) - 1$. Similarly, by substituting the sequence $r_{s_m}(\cdot)$ in (11) and (12) in terms of $r_{d_m}(\cdot)$, we can obtain 2-D DPWT coefficients matrices $s_{d_{j+1}}$ and $d_{d_{j+1}}$, which are

$$s_{d_{j+1}}(u_1, v_1) = \begin{cases} \sum_{n=0}^{l-1} r_{d_m}(p + mN_j + l - 1) & 0 \leq u_1 \leq (N_j/2) - (l/2) \\ \sum_{n=0}^{l-1} r_{d_m}(p + mN_j + l - 1) & \sum_{n=2k+1}^{l-1} r_{d_m}(2v_1 + (m - 2k - 2)N_j + l - 1) \end{cases}$$

$$\left(\frac{N_j}{2} - (l/2 - 1) \leq u_1 \leq N_j/2 - 1 \right)$$

Equations (8)–(14) are called the advanced operator correlation algorithm (AOCA). In (9) and (10), when $(N_j/2) - (l/2) + 1 \leq v_1 \leq (N_j/2) - 1$, the computation of $r_{s_m}(\cdot)$ and $r_{d_m}(\cdot)$ needs to use the cyclically duplicated data of the first $l - 2$ columns of $s_{s_j}$. Thus, the first $l - 2$ columns of $s_{s_j}$ are called boundary data. Similarly, we can also find that row boundary data are needed in the computation of $s_{s_{j+1}}(\cdot), d_{s_{j+1}}(\cdot)$, $s_{d_{j+1}}(\cdot)$, and $d_{d_{j+1}}(\cdot)$.

The structure of AOCA is very suitable for a hardware implementation. The AOCA computes the elements of 2-D DPWT coefficient matrices at the same sequence as the scanned pattern of the input image. Therefore, by using AOCA in the computation of 2-D DPWT coefficients, we cannot only dramatically reduce the number of multiplications but also simplify the realization of real-time processing of the 2-D DPWT. This additional improvement of the performance of AOCA can effectively increase the execution speed and decrease the dissipated energy in both hardware and software implementations of the OCA. An example for $l = 4$ and applying AOCA is illustrated below.

Example: Consider a four-tap wavelet filter. By following the one-stage OCA decomposition process of a $N \times N$ image, the 2-D DPWT
coefficients of the octave band at the \( J + 1 \)st level can be derived as

\[
ss_{J+1}(u_1, v_1) = \sum_{m=0}^{2} \sum_{n=0}^{2} ss_j(u, v)w_{m, n}
\]

(15)

\[
sd_{J+1}(u_1, v_1) = \sum_{m=0}^{2} \sum_{n=0}^{2} ss_j(u, v)(-1)^{3-n+1}w_{m, 3-n}
\]

(16)

\[
ds_{J+1}(u_1, v_1) = \sum_{m=0}^{2} \sum_{n=0}^{2} ss_j(u, v)(-1)^{3-m+1}w_{3-m, n}
\]

(17)

\[
dd_{J+1}(u_1, v_1) = \sum_{m=0}^{2} \sum_{n=0}^{2} ss_j(u, v)(-1)^{m+n}w_{3-m, 3-n}
\]

(18)

where \( u = (2u_1 + m)N, v = (2v_1 + n)N, \forall u_1, v_1 \in \{0, 1, \ldots, (N/2 - 1)\} \), and \( w_{m, n} \) are defined in (7). From (9) and (10), we can obtain two sequences \( rs_m(p + 3) \) and \( rd_m(p + 3) \) as

\[
rs_m(p + 3) = \begin{cases} 
((((a_{m,0}(p)) + a_{m,1}(p + 1)) + a_{m,2}(p + 2)) + a_{m,3}(p + 3)) \\
0 \leq v_1 < N/2 - 1 \\
((((a_{m,3}(p)) + a_{m,1}(p + 1)) + a_{m,2}(2u_1N)) + a_{m,0}(2u_1N + 1)) \\
v_1 = N/2 - 1
\end{cases}
\]

(19)

\[
r_m(p + 3) = \begin{cases} 
(((a_{m,3}(p)) - a_{m,1}(p + 1)) + a_{m,1}(p + 2)) - a_{m,0}(p + 3)) \\
0 \leq v_1 < N/2 - 1 \\
(((a_{m,3}(p)) - a_{m,2}(p + 1)) + a_{m,1}(2u_1N)) - a_{m,0}(2u_1N + 1)) \\
v_1 = N/2 - 1
\end{cases}
\]

(20)

where \( p = 2u_1N + 2v_1 \). By applying (11)–(14), the 2-D DPWT coefficients of the \( J + 1 \)st level can be obtained as

\[
ss_{J+1}(u_1, v_1) = \begin{cases} 
(((r_{s_j}(p + 3) + r_{s_j}(p + N + 3)) + r_{s_j}(p + 2N + 3)) + r_{s_j}(p + 3N + 3)) \\
0 \leq u_1 < N/2 - 1 \\
((r_{s_j}(p + 3) + r_{s_j}(p + N + 3)) + r_{s_j}(2v_1 + 3)) + r_{s_j}(2v_1 + N + 3)) \\
u_1 = N/2 - 1
\end{cases}
\]

(19)

\[
ds_{J+1}(u_1, v_1) = \begin{cases} 
(((r_{s_j}(p + 3) - r_{s_j}(p + N + 3)) + r_{s_j}(p + 2N + 3)) - r_{s_j}(p + 3N + 3)) \\
0 \leq u_1 < N/2 - 1 \\
((r_{s_j}(p + 3) - r_{s_j}(p + N + 3)) + r_{s_j}(2v_1 + 3)) - r_{s_j}(2v_1 + N + 3)) \\
u_1 = N/2 - 1
\end{cases}
\]

(20)

\[
sd_{J+1}(u_1, v_1) = \begin{cases} 
(((r_{d_j}(p + 3) + r_{d_j}(p + N + 3)) + r_{d_j}(p + 2N + 3)) + r_{d_j}(p + 3N + 3)) \\
0 \leq u_1 < N/2 - 1 \\
((r_{d_j}(p + 3) + r_{d_j}(p + N + 3)) + r_{d_j}(2v_1 + 3)) + r_{d_j}(2v_1 + N + 3)) \\
u_1 = N/2 - 1
\end{cases}
\]

(19)

\[
dd_{J+1}(u_1, v_1) = \begin{cases} 
(((r_{d_j}(p + 3) - r_{d_j}(p + N + 3)) + r_{d_j}(2v_1 + N + 3)) + r_{d_j}(2v_1 + 3)) - r_{d_j}(2v_1 + N + 3)) \\
u_1 = N/2 - 1.
\end{cases}
\]

(20)

From this example, we can observe that this algorithm is a parallel-pipeline structure that derives all the elements of \( ss_{J+1}, sd_{J+1}, ds_{J+1}, \) and \( dd_{J+1} \) simultaneously at the same sequence as the scanned pattern of its input image. In addition, the computation structure of the AOCA consists of several similar computational modules in (9)–(14). Therefore, AOCA can be easily implemented in VLSI design. For an \( l \)-tap wavelet filter, only \((l^2 + 1)/2\) elements of \( a_{m,n}(t) \), \( \forall m, n \in \{0, 1, 2, \ldots, l - 1\} \) have to be computed in parallel.

Unlike the conventional 2-D DPWT methods, based on (4), which are separable, both the OCA and the AOCA-based algorithms are nonseparable. For a hardware implementation, it is also important to reduce the bus width of intermediate data and increase the accuracy of the final results. These two properties are dominantly affected by the fixed-point precision of 2-D DPWT filter coefficients.

To compare the accuracy performance of finite bit-length filters between the separable and the nonseparable computation methods, four Daubechies’ wavelets of filter length \( l = 4, 6, 8, \) and 10 were used. The fixed-point analysis results (including one sign bit) for these two methods are shown in Figs. 1 and 2, respectively. The results show that for the same accuracy (measured in output SNR), nonseparable computation methods require nearly half the bit accuracy that the separable computation methods need.

Table I shows a comparison of several 2-D DPWT algorithms. The classical 2-D PA and the 2-D RPA [10] are separable computation methods. They require \((l + 1)C_M \) multiplications and \((l + 1)C_A \) additions to compute each 2-D DPWT coefficient, where \( C_M \) and \( C_A \) denote the multiplications and additions required for 1-D filtering. Based on (4), the classical 2-D PA has \( C_M = l \) and \( C_A = l - 1 \). For the 2-D RPA, using the short-length FIR filtering algorithm [7], \( C_M \approx (3/4) \times l, \) and \( C_A \approx (3/4) \times l + (1/2) \). Without loss of generality, in the comparison, we assume that all the algorithms are simulated in a general-purpose computer. The implementation of the classical 2-D PA needs to save all the \( N^2 \) intermediate row-transformed data, i.e., \( ss_j, H \) and \( ss_j, G \). The classical 2-D PA is able to provide undistorted 2-D DPWT coefficients for perfect
TABLE I

COMPARISON OF DIFFERENT 2-D DPWT ALGORITHMS FOR COMPUTING EACH 2-D DPWT COEFFICIENT (N = 1024)

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>MCs.</th>
<th>ACs.</th>
<th>Storage for Row-Column Conversion</th>
<th>Computational Method</th>
<th>Undistorted Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classical 2-D PA</td>
<td>((l+1))</td>
<td>((l+1)/(l+2))</td>
<td>(N^*)</td>
<td>Separable</td>
<td>Yes</td>
</tr>
<tr>
<td>Short-Length 2D RPA</td>
<td>((l/4)/(l+1))</td>
<td>((l/4)/(l+1/2))</td>
<td>(2(l-i)N)</td>
<td>Separable</td>
<td>No</td>
</tr>
<tr>
<td>2D OCA</td>
<td>(t)</td>
<td>((l+1)/(l+2))</td>
<td>None</td>
<td>Non-Separable</td>
<td>Yes</td>
</tr>
<tr>
<td>2D AOCA</td>
<td>((l/2)/(l+1))</td>
<td>((l+1)/(l+2))</td>
<td>(2lN)</td>
<td>Non-Separable</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* MCs and ACs stand for the number of multiplications and additions, respectively.

Fig. 2. Fixed-point accuracy of 2-D filter coefficients for nonseparable computation methods.

reconstruction. The 2-D OCA, which is based on the double-shift correlation, combines the input data and the filters into the form of the octave band’s 2-D DPWT coefficients. Hence, the 2-D OCA does not need any storage cells to save the row-transformed data, but it needs \(4((l/2) - 1)N^*\) words and \(4((l/2) - 1)\) words to store the row and column boundary data, respectively, where \(N^* = 2^{-k}N\), and \(k = j - J + 1 \geq 1\) in the \(j\)th level. Since these storage cells can be reused during the multilevel decomposition, the 2-D OCA actually needs \((l - 2)(N - 2)\) words to store the boundary data. Since the RPA does not use the boundary data, it will yield distorted 2-D DPWT coefficients. Moreover, the 2-D RPA still needs \(2N^*(l-1)\) words to store the row-transformed data in the \(j\)th level. From (9) and (10), it is clear that in the \(j\)th level the AOCA needs \(2N^*l\) words to store the sequences \(r_{m,1}(\cdot)\) and \(rd_{m,1}(\cdot)\) for \(m = 0, 1, \ldots, l-1\). It also needs \((l - 2)(2N^* + 1)\) words to save the boundary data. Up to the final level \(V_0\), the 2-D RPA needs a total of \(N(l - 1) \sum_{j=1}^{l} 2^{j-1} \approx 2l(l-1)N\) words to store the row-transformed data. However, the AOCA needs \(2N^*\) words to store the sequences \(r_{m,1}(\cdot)\) and \(rd_{m,1}(\cdot)\) and \((l - 2) \sum_{j=1}^{l} 2^{j-1} (N + 1) \approx 2(l-1)N\) words to store the boundary data. Therefore, the AOCA needs more storage cells for the intermediate data than the 2-D RPA.

V. Conclusion

The AOCA is a homeomorphic highpass filter for the 2-D DPWT that is suitable for real-time applications. Based on the computation structure of OCA, compared with the classical 2-D PA, AOCA needs only half of the multiplications and half the bit length precision for the 2-D DPWT filter coefficients. In addition, the AOCA is inherently modular and, thus, suitable for an efficient VLSI implementation [13].

ACKNOWLEDGMENT

The authors would like to express their appreciation to the referees and Dr. K. Constantinides, as well as Dr. Y. S. Hung from CSIST, Taiwan, R.O.C., for their useful advice throughout this research.

REFERENCES