Implementation of a Tree-Structured Filter Bank Using Cyclic Convolutions

HIDEO MURAKAMI
Kanazawa Institute of Technology
7-1 Ohgigaoka Nonoichi, Ishikawa 921-8501
JAPAN

Abstract: A filter bank in which cyclic convolutions are used in place of linear convolutions will be referred to as a cyclic convolution filter bank (CCFB). A dyadic tree-structured CCFB can be used to perform a discrete wavelet transform suitable for coding based on symmetric extension methods. This paper derives an implementation technique for the tree-structured CCFB that can be realized by FFT processors.

Key-Words: Filter bank, tree-structured filter bank, wavelet transform, cyclic convolution, implementation, discrete Fourier transform.

1 Introduction
Discrete wavelet transforms have been studied extensively for use in subband coding of speech and image signals. The discrete wavelet transform can be realized by a dyadic tree-structured filter bank [1]. A two-channel filter bank is a basic building block of the dyadic tree-structured filter bank, and consequently of the discrete wavelet transform.

This paper focuses on a tree-structured filter bank in which filters are cyclic convolutions instead of linear convolutions. Cascading the two-channel analysis cyclic-convolution filter bank (CCFB), the two-level tree-structured analysis CCFB is obtained as shown in Fig. 1 when the input is of length N.

The tree-structured CCFB has been applied to image coding utilizing a symmetric extension method [2]-[4]. In the symmetric extension method, when an input x(n) is of size N, the input is first symmetrically extended into the 2N-length signal x_e(n) by x_e(n) = x(n), 0≤n≤N-1 and x_e(n) = x(2N-n-1), N≤n≤2N-1, and the extended signal is then fed into a two-channel analysis filter bank in which cyclic convolutions are used in place of linear convolutions. The extended signal created in this manner satisfies x_e(0)=x_e(2N-1), and thus the coding does not introduce border artifacts.

This paper shows that the CCFB can be implemented efficiently by FFT processors. The major results of this paper appear in [7].

2 Polynomial Representations for DFT and IDFT
We use polynomial representations for the discrete Fourier transform (DFT) and the inverse DFT (IDFT) used in [5]. If A(z) is the z-transform of a M-length
signal, DFT of the signal is given by \( A(k) = A(e^{\frac{2\pi i k}{M}}) \), \( k=0, 1, ..., M-1 \), which are the values of \( A(z) \) after substituting \( e^{\frac{2\pi i k}{M}} \) for \( z \). Introduce the \( \Psi \)-polynomials defined as

\[
\Psi_{M,k}(z) = \sum_{n=0}^{M-1} e^{\frac{2\pi i n M}{M} z^{-n}}, \quad k=0, 1, ..., M-1
\]

These polynomials satisfy the equation

\[
\Psi_{M,k}(e^{\frac{2\pi i k}{M}}) = M \delta_{\nu}(k-l)
\]

where \( \delta_{\nu}(n) = 0 \) when \( n \) is a multiple of \( M \), and \( \delta_{\nu}(0) = 1 \) otherwise. Using the \( \Psi \)-polynomials, IDFT can be written as

\[
A(z) = \frac{1}{M} \sum_{k=0}^{M-1} A(k) \Psi_{M,k}(z)
\]

Throughout this paper, IDFT will be viewed as the \( z \)-transform given in this form.

### 3 Implementation for the Synthesis CCFB

We first discuss implementation of the synthesis part because this part is simpler to treat. For the \( t \)th stage synthesis bank, the synthesis filters \( F_{l}^{(t)}(z) \) and \( F_{h}^{(t)}(z) \), which are of length \( N/2^t \), are represented in the polyphase decomposition of

\[
F_{m}^{(t)}(z) = F_{ml}^{(t)}(z) + z^{-1} F_{mh}^{(t)}(z), \quad m = L, H
\]

The synthesis polyphase matrix is given as

\[
R^{(t)} = \begin{pmatrix}
F_{l}^{(t)}(z) & F_{h}^{(t)}(z) \\
F_{m}^{(t)}(z) & F_{mh}^{(t)}(z)
\end{pmatrix}
\]

and the complex module synthesis polyphase matrices by

\[
R^{(t)}(k) = \begin{pmatrix}
P_{l}(k) & R_{m}^{(t)}(k) \\
P_{m}^{(t)}(k) & R_{mh}^{(t)}(k)
\end{pmatrix}, \quad k = 0, 1, ..., N/2^{t-1} - 1
\]

in which each entry of the module synthesis polyphase matrices is the \( N/2^t \)-point DFT of the corresponding entry of \( R^{(t)} \). Applying the complex parallel module implementation obtained in [5] for each of the two-channel CCFBs in the cascade of \((t+1)\)th and \( t \)th stage CCFBs, we obtain the implementation shown in Fig. 2.

We now focus on the operation indicated by the double headed arrow (A) of Fig. 2. The signal \( Q(z) \) defined in the figure, taking \( N/2^t \)-point IDFTs of \( P_{l}(k) \) and \( P_{h}(k) \) and then polyphase-composing these IDFTs, is given by

\[
Q(z) = \frac{2^t N}{N} \sum_{k=0}^{N/2^t - 1} [P_{l}(k) + P_{h}(k)z^{-1}] \Psi_{N/2^t + 1}^{(t)}(z^{-1})
\]

Using (2), the \( N/2^t \)-point DFT of \( Q(z) \) is given by

\[
Q(l) = P_{l}(l) + P_{h}(l)e^{-j2\pi l/N}, \quad l = 0, 1, ..., N/2^{t+1} - 1
\]

These residue equations can be written in matrix form as

\[
\begin{pmatrix}
Q(k) \\
Q(N/2^{t+1} + k)
\end{pmatrix} = A^{(t)}_{k} \begin{pmatrix}
P_{l}(k) \\
P_{h}(k)
\end{pmatrix}, \quad k = 0, 1, ..., N/2^{t+1} - 1
\]

where

\[
A^{(t)}_{k} = \begin{pmatrix} 1 & e^{-j2\pi k/N} \\ 1 & -e^{-j2\pi k/N} \end{pmatrix}
\]

With respect to computational complexity, the two \( N/2^t \)-point IDFTs and the \( N/2^t \)-point DFT in the operation (A) are reduced to simply \( N/2^{t+1} \times 2 \times 2 \) matrix multiplications.

The complex implementation of the cascade system in Fig. 2 is further simplified by pre-multiplying the

![Fig. 2. Implementation for the cascade of (t+1)th and tth stage CCFB in the synthesis part.](image-url)
matrix $A_k^{(0)}$ and the matrix $R^{(n+1)}(k)$. When the above techniques are applied to the synthesis part of the three-level tree-structured CCBF in Fig. 1(b), the resulting implementation is described in Fig. 3. In this figure, the blocks containing $A_k^{(0)}R^{(n+1)}(k)$, $t=1, 2$, include the ordering of the output according to Eq. (9), in addition to the matrix multiplications.

Fig. 3. Proposed implementation of the two-level tree-structured CCBF for the synthesis part.

4 Implementation for the Analysis CCBF

We now turn to the analysis part. In order to apply the parallel module implementation, represent the analysis filters $H_L^{(0)}(z)$ and $H_H^{(0)}(z)$ in the polyphase decomposition of

$$H_a(z) = \sum_{m} z^{-m}H_a(z) \mod(1 - z^{-2\pi k}), \ m = L, H$$  \hspace{1cm} (11)

The analysis polyphase matrix is defined as

and the complex module analysis polyphase matrices as

$$E^{(0)}(k) = \begin{bmatrix} H_L^{(0)}(k) & H_H^{(0)}(k) \\ H_L^{(0)}(k) & H_H^{(0)}(k) \end{bmatrix} , k = 0, 1, ..., N/2-1 \hspace{1cm} (13)$$

Each entry of $E^{(0)}(k)$ are the $N/2$-point DFT of the corresponding entry of $E^{(0)}$. Applying the complex parallel module implementation for each of the two-channel analysis CCBFs in the cascade of $t$th and $(t+1)$th stage CCBFs for the analysis part, we obtain the implementation shown in Fig. 4.

A careful comparison between the operation (A) in Fig. 2 and the operation (B) in Fig. 4 reveals that the operation (B) is simply the reverse of the operation (A). That is, the input and output relation of the operation (B) is written as

$$\begin{bmatrix} V_o(k) \\ V_i(k) \end{bmatrix} = B_k^{(0)} \begin{bmatrix} U(k) \\ U(N/2^{t+1} \pm k) \end{bmatrix} , k = 0, 1, ..., N/2^{t+1}-1$$  \hspace{1cm} (14)$$

where the matrix $B_k^{(0)}$ is the inverse of $A_k^{(0)}$, which is given by

$$B_k^{(0)} = \frac{1}{2} \begin{bmatrix} 1 & e^{2\pi i \omega k/N} \\ e^{2\pi i \omega k/N} & 1 \end{bmatrix}$$  \hspace{1cm} (15)$$

Similar to the synthesis case, the matrix multiplication by $B_k^{(0)}$ and the multiplication by $E^{(t+1)}(k)$ can be combined into the single matrix multiplication by $E^{(t+1)}(k)B_k^{(0)}$. Applying the above techniques to each of the two-channel analysis CCBFs in the system of Fig. 1(a), the analysis part is implemented by the system shown in Fig. 5.

Fig. 4. Implementation for the cascade of $t$th and $(t+1)$th stage CCBF in the analysis part.

$$E^{(0)} = \begin{bmatrix} H_L^{(0)}(z) & H_H^{(0)}(z) \\ H_L^{(0)}(z) & H_H^{(0)}(z) \end{bmatrix}$$  \hspace{1cm} (12)$$
5 Computational Complexity of the Proposed Implementation

Computational complexity of the proposed implementation method is evaluated, focusing on the analysis part. Taking $M$-point FFT or IFFT, one requires $(M/2)\log_2(M)$ complex multiplications and $(M)\log_2(M)$ complex additions. The computational complexity of a $2 \times 2$ complex matrix multiplication is four complex multiplications and two complex additions. The computational complexity of one complex multiplication is counted as four real multiplications and two real additions. Computational complexities for the tree-structured systems can be computed, using the above numbers, and the drawings in Fig. 3.

We now evaluate computational complexity when each two-channel CCFB is implemented by the parallel implementation without the minimization procedure derived in [5]. It requires two $M/2$-point DFTs, two $M/2$-point IDFTs, and $M/2$ $2 \times 2$ complex matrix multiplications to implement a two-channel analysis CCFB for an $M$-length input.

One of the most efficient implementation is a cosine-modulated filter bank. The computational complexity for the two-channel filter bank with $M$-length analysis filters is nearly equal to $M/2$ multiplications and $(M-1)/2$ additions per an input sample point, using polyphase implementation, and neglecting the cost of the fast DCT computation [pp. 387, 6]. To be fair with the CCFB systems, the input size is assumed to be $N/2$ instead of $N$ because the symmetric extension is not necessary for the cosine-modulated filter bank. Table 1 summarizes the computational complexities for the two-level and three-level analysis filter banks.

6 Conclusion

This paper has described an efficient implementation method for the tree-structured CCFB, and thus the discrete wavelet transform based on cyclic convolutions. Computational efficiency is achieved by reducing a number of DFT and IDFT computations into $2 \times 2$ complex matrix multiplications.

References: