Lapped Transform via Time-Domain Pre- and Post-Filtering

Trac D. Tran*, Jie Liang, and Chengjie Tu

Abstract— **This paper presents a general framework of constructing a large family of lapped transforms with symmetric basis functions by adding simple time-domain re- and post-processing modules onto existing block DCT based infrastructures. A subset of the resulting solutions is closed-form, fast computable, modular, near optimal in the energy compaction sense, and leads to an elegant boundary handling of finite-length data. Starting from these solutions, a general framework for block-based signal decomposition with a high degree of flexibility and adaptivity is developed. Several simplified models are also introduced to approximate the optimal solutions. These models are based on cascades of plane rotation operators and lifting steps, respectively. Despite tremendous savings in computational complexity, the optimized results of these simplified models are virtually identical to that of the complete solution. The multiplierless versions of these pre- and post-filters when combined with an appropriate multiplierless block transform such as the binDCT [1] generate a family of VLSI-friendly fast lapped transforms with reversible integer-to-integer mapping. Numerous design examples with arbitrary number of channels and arbitrary number of borrowed samples are presented.**

I. Introduction

MOST image and video coding standards have shared one common coding philosophy: data is partitioned into small local blocks, which are decorrelated by the discrete cosine transform (DCT), then encoded by various variable-length codes [2], [3], [4]. The popularity of this coding approach can be attributed to many factors: *(i)* the DCT's near-optimality for smooth signal models; *(ii)* many efficient fast-computable DCT algorithms; *(iii)* small on-board memory requirement; *(iv)* flexibility and adaptivity on the block level, e.g., coding mode can be selected on a block-by-block basis; *(v)* parallel processing capability; and *(vi)* simple resynchronization in noisy environments.

However, there are two main problems with this block-based DCT approach. The first problem is the lack of coding efficiency since inter-block correlation has not been well taken into account. The second is the notorious blocking artifacts – discontinuities at the block boundaries resulting from reconstruction mismatches – at low bit-rate situations. Blocking artifacts are visually annoying, and they set a severe limit on the achievable bit-rate with acceptable quality.

Many techniques have been developed to improve coding efficiency and to avoid or reduce blocking effects. Most can be classified into two distinct categories: *(i)* using a global transform (more accurately, transforms with overlapping basis functions); *(ii)* using pre- and post-processing techniques. Algorithms in the first approach improve reconstruction quality by employing either the wavelet transform or the lapped transform (LT) in signal decomposition and reconstruction [5]. New developments in the wavelet coding field lead to the blocking-free JPEG2000 image compression standard. In the second approach, pre- and post-processing techniques have been proposed to improve reconstruction quality while maintaining standard compliance. However, these techniques tend to lack a strong rate-distortion foundation; pre- and post-processing are mostly treated separately; and they usually destroy the original signal contents.

In this paper, through a series of elementary matrix manipulations, we shall demonstrate that a large class of lapped transforms can be constructed as a combination of pre-filtering and post-filtering in the current block-based DCT framework. The pre- and post-processing operator is placed at each block boundary. Unlike most previous pre- and post-processing approaches, the pre- and post-filter in our framework are intimately related. In fact, they are the exact inverse of each other, and, together with the DCT, they form invertible lapped transforms with arbitrary amount of overlapping samples. Perfect reconstruction and linear-phase basis functions can be structurally guaranteed. The new framework provides several advantages:

- Existing block-based infrastructure can be kept intact.
- Coding efficiency is improved by taking into account inter-block spatial correlation in the pre-filtering stage.
- Blocking artifacts are eliminated with post-filtering along the block boundaries while ringing artifacts can be controlled by varying the number of overlapping samples.
- Pre- and post-filter are constructed in modular cascaded stages, leading to minimal hardware/software modifications and simple future upgrades.
- Pre- and post-processing retain all flexible features of block-based approaches and add on top a high level of adaptivity in signal decomposition.
- Intuitive time-domain interpretation facilitates the design of transforms with arbitrary-length basis functions, odd-channel filter banks, boundary filter banks, and switching filter banks in adaptive decomposition.

The outline of the paper is as follows. In Section II, we offer a review of important background materials, concepts, motivations, and previous related works. Next, Section III demonstrates that the well-known type-II fast LOT [6] can be expressed as a combination of time-domain pre/post-filtering in the popular block DCT/IDCT framework. Based on this result, Section IV presents the general block-based signal decomposition framework that includes orthogonal solutions, biorthogonal solutions, global time-

The authors are with the Department of Electrical and Computer Engineering, The Johns Hopkins University, Baltimore, MD 21218. Email: {trac,jieliang,cjtu}@jhu.edu.

domain viewpoint, LT with arbitrary overlapping factors, and boundary handling for finite-length signals. Issues in optimal pre- and post-filter design, fast implementations, multiplierless solutions as well as complexity and coding performance are addressed in Section V. Finally, conclusions are drawn in Section VI.

Notations and Conventions: We use bold-faced lower-case characters to denote vectors and bold-faced upper-case characters to denote matrices. The symbols $|\mathbf{V}|$, $\mathbf{V}^T$, $\mathbf{V}^{-1}$, $[\mathbf{V}]_{m,n}$ denote respectively the determinant, the transpose, the inverse, and the element at the m-th row and n-th column of the matrix $\mathbf{V}$. Occasionally, upper-case subscript such as $M \times N$ is added to indicate the matrix size if it is unclear from context. Lower-case subscripts are reserved for indexing purposes. Several special matrices with reserved symbols are: the identity matrix $\mathbf{I}$, the anti-diagonal (reversal) matrix $\mathbf{J}$, the null matrix $\mathbf{0}$, and the diagonal matrix with alternating $+1$ and -1 entries $\mathbf{D}$, i.e., $\mathbf{D} = diag\{1, -1, 1, -1, \ldots\}$. Also, the symbols $\mathbf{C}_M^{II}$, $\mathbf{C}_M^{IV}$, and $\mathbf{S}_M^{IV}$ denote, respectively, the $M \times M$ type-II DCT matrix, type-IV DCT matrix, and type-IV discrete sine transform (DST) matrix as defined in [7]. Signals and basis functions are represented as column vectors. For presentation simplicity, our convention has the forward transform (analysis FB) being anti-causal whereas the inverse transform (synthesis FB) being causal.

II. Review

A. Block Transform

In this paper, a block transform is defined as an $M \times M$ linear operator which maps M input samples to M transform coefficients. The matrix describing the linear mapping is called the transform matrix. The two families of block transforms used throughout this paper are the DCT and the DST defined in [7]. All of these transforms are orthonormal, *i.e.*, $\mathbf{C}_M^{II^{-1}} = \mathbf{C}_M^{II^T}$; $\mathbf{C}_M^{IV^{-1}} = \mathbf{C}_M^{IV^T}$; $\mathbf{S}_M^{IV^{-1}} = \mathbf{S}_M^{IV^T}$. Also, the following relationship between the DCT and the DST matrix can be easily established: $\mathbf{S}_M^{IV} = \mathbf{D}\,\mathbf{C}_M^{IV}\,\mathbf{J}$. The popular DCT in JPEG and MPEG is the 8-point type-II DCT. Because of its practical value, numerous fast DCT-II algorithms have been proposed [7]; the most effective are ones based on sparse matrix factorizations. One factorization is even partly recursive, i.e., an M-point DCT-II can be implemented via an $\frac{M}{2}$-point DCT-II and an $\frac{M}{2}$-point DCT-IV [8], [9],

$$\mathbf{C}_M^{II} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{C}_{M/2}^{II} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{M/2}^{IV}\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix}. \quad (1)$$

B. Lapped Transform

An M-band lapped transform (LT) is a linear transformation that partitions the input signal into small overlapped blocks and then processes each block independently. In the 1D direct implementation, the input signal $\mathbf{x}$ can be blocked into short sequences $\mathbf{x}_m$ of length L ($L > M$). The corresponding transform vector $\mathbf{y}_m$ of length M is obtained from the $M \times L$ transform matrix $\mathbf{H}$ as $\mathbf{y}_m = \mathbf{H}\,\mathbf{x}_m$.

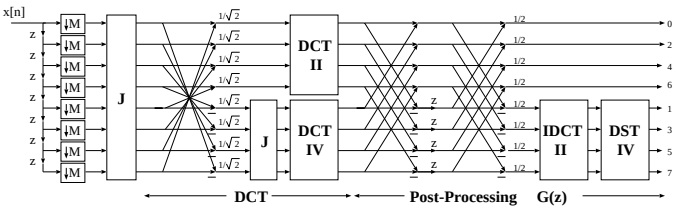

Fig. 1. The type-II fast lapped orthogonal transform.

Each block of input samples has an overlap of $(L - M)$ samples with each of its adjacent neighbors, $\mathbf{x}_{m-1}$ and $\mathbf{x}_{m+1}$. Typically, L is chosen as a multiple of M, i.e., $L = KM$, where K is called the overlapping factor. The overlapping percentage is defined as $100 \times \frac{L-M}{L}\%$. The M rows of the transform matrix $\mathbf{H}$ hold the transposed analysis basis functions $\mathbf{h}_i^T$. At the decoder, we have the $L \times M$ inverse transform matrix $\mathbf{F}$ whose columns hold the synthesis basis functions $\mathbf{f}_i$. The reconstructed segments $\hat{\mathbf{x}}_m = \mathbf{F}\,\mathbf{y}_m = \mathbf{F}\,\mathbf{H}\,\mathbf{x}_m$ must be accumulated in an overlap-add fashion to recover the original signal $\mathbf{x}$. For 2D signals such as images and video frames, the transform can be applied separably, i.e., each row is processed in 1D followed by each column or vice versa.

With the $M \times L$ forward LT matrix $\mathbf{H}$ and the $L \times M$ inverse LT matrix $\mathbf{F}$ divided into square $M \times M$ submatrices $\mathbf{H}_i, \mathbf{F}_i$ $(i = 0, 1, \ldots, K-1)$ as $\mathbf{H} = [\mathbf{H}_0\ \mathbf{H}_1\ \cdots\ \mathbf{H}_{K-1}]$ and $\mathbf{F} = [\mathbf{F}_0\ \mathbf{F}_1\ \cdots\ \mathbf{F}_{K-1}]^T$, perfect reconstruction is achieved when $\sum_{i=0}^{K-1-l} \mathbf{F}_i^T \mathbf{H}_{i+l} = \sum_{i=0}^{K-1-l} \mathbf{F}_{i+l}^T \mathbf{H}_i = \delta(l)\,\mathbf{I}_M$ [10]. As pointed out in [6], the aforementioned $M \times L$ lapped transform is simply the polyphase implementation of a maximally decimated M-channel L-tap filter banks. The precise relationship between the $M \times L$ lapped transform matrix (assuming $L = KM$) $\mathbf{H} = [\mathbf{H}_0\ \mathbf{H}_1\ \cdots\ \mathbf{H}_{K-1}]$ and the polyphase matrix is $\mathbf{E}(z) = \sum_{i=0}^{K-1} \mathbf{H}_i z^i$.

Fast lapped transforms can be constructed in polyphase form from components with well-known fast-computable algorithms such as the DCT and the DST. One of the most elegant solution is the type-II fast LOT whose polyphase matrix is [6]

$$\begin{aligned} \mathbf{E}(z) &= \tfrac{1}{2} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_{M/2}^{IV}\,\mathbf{C}_{M/2}^{II^T} \end{bmatrix} \\ &\quad \times \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \mathbf{C}_M^{II}\mathbf{J}_M \\ &\triangleq \mathbf{G}(z)\,\mathbf{C}_M^{II}\,\mathbf{J}_M. \end{aligned} \quad (2)$$

This structure is illustrated in Fig. 1. It is scalable to all even-channel M, generating a large family of LOT with symmetric basis functions and 50% overlap ($K = 2$). There are many other fast solutions; all of them involve replacing the product $\mathbf{S}_{M/2}^{IV}\,\mathbf{C}_{M/2}^{II^T}$ by different matrices $\mathbf{V}$, usually cascades of various 2×2 matrices along the diagonal axis [6], [10]. If a larger overlapping percentage is desired, more modules $\mathbf{G}_i(z)$ with different $\mathbf{V}_i$ can be added, i.e., $\mathbf{E}(z) = \mathbf{G}_{K-1}(z)\,\mathbf{G}_{K-2}(z)\,\cdots\,\mathbf{G}_1(z)\,\mathbf{C}_M^{II}$. This is known as the generalized LT [10], [11], [12].

C. Pre- and Post-Processing for DCT-based Systems

There has been a tremendous amount of research on pre- and post-processing algorithms for image and video compression systems. Both classes of algorithms share one common goal: to eliminate or reduce the severity of coding artifacts in the reconstructed signal. This section can only offer a compact survey of popular approaches. We refer the reader to [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26] and references therein.

There are only a few pre-processing algorithms discussed in the literature [13], [20]. All of them concentrate on the removal of noise, texture, or small features, and try to allocate the bit-budget savings to more important visual information. Note that we are interested in time-domain pre-filtering only, i.e., algorithms that work directly on the input time samples, not algorithms that process DCT coefficients before coding or quantization.

There are significantly more research activities in the post-processing field. Post-filtering algorithms can be divided into two classes: enhancement [18], [22], [23], [24], [26] and recovery [16], [17], [19], [21]. In the enhancement approach, a heuristic operator is designed to filter along the block boundaries. These enhancement algorithms are usually fast; they work well in practice and can be found in most international video coding standards [23], [26] as deblocking filters. However, these are heuristic solutions and they have a tendency to smooth out the true underlying edge information. The second class of post-processing algorithms relies on more mathematically rigorous recovery techniques such as optimization via a Lagrangian linear model or projection onto convex sets. Algorithms using the recovery approach usually outperform their ad-hoc enhancement counterparts, but they are much more computationally expensive. Previous work on an integrated pre- and post-processing for DCT-based codecs is almost non-existent.

III. LT from Pre- and Post-Processing

A. Motivation

All lapped transforms mentioned in Section II can be viewed as post- and pre-processing of the DCT coefficients between the quantizer as shown in Fig. 2(a). Up until now, all high-performance LT with linear-phase basis functions designed for image and video compression (type-I fast LOT/LBT, type-II fast LOT/LBT, GLT, GenLOT, GLBT) are based on the DCT-II post-processing approach [6], [10], [11], [12], [27]. A more intuitive viewpoint is depicted in Fig. 2(b) where the pre- and post-filter are *outside* the existing framework. This way, we have a chance at improving coding performance while achieving standard-compliance with minimal software/hardware modifications.

The idea is not new. For example, the modulated LT (MLT) in audio coding standards [6] can be viewed as time-domain pre-filtering of the type-IV DCT input. Here, the block operator $\mathbf{C}_M^{IV}$ in the MLT plays the modulation role whereas the block operator $\mathbf{C}_M^{II}$ in Fig. 2(b) plays the decorrelation role. Moreover, from a general theoretic

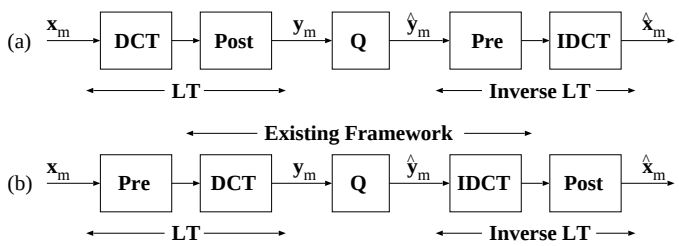

Fig. 2. Different LT viewpoints. (a) LT as post-processing of DCT coefficients. (b) LT as pre-processing of DCT inputs.

viewpoint, the LOT has been shown to comprise of either: *(i)* cross-boundary post-processing of a certain block transform's output; or *(ii)* cross-boundary pre-processing of a block transform's input [28]. Surprisingly, the pioneering LT construction attempt by Malvar in [29], [30] actually follows the pre-processing approach. This work even predates its celebrated cousins, the type-I and type-II fast LOT. This paper provides a straightforward generalization of the early effort in [29], [30]. Our focus is on the construction of various pre- and post-filters.

B. Type-II Fast LOT as DCT/IDCT Pre- and Post-Processing

Through a series of elementary matrix manipulations, we illustrate that the type-II fast LOT can be viewed as a combination of the common block-based DCT/IDCT framework with simple time-domain pre- and post-filtering. In other words, the analysis polyphase matrix in (2) can be re-written as

$$\mathbf{E}(z) \;=\; \mathbf{D}_M\,\mathbf{C}_M^{II}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & z\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{I} & \mathbf{0}\end{bmatrix}\mathbf{P}, \qquad (3)$$

where

$$\mathbf{P} \;\triangleq\; \tfrac{1}{2}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix} \qquad (4)$$

$$\mathbf{V} \;\triangleq\; \mathbf{J}\,\mathbf{C}_{M/2}^{II^T}\,\mathbf{C}_{M/2}^{IV}\,\mathbf{J}. \qquad (5)$$

The derivation can be found in the Appendix.

Since the diagonal matrix $\mathbf{D}_M$ only inverts the polarity of the transform coefficients, it can be ignored. Finally, if we define $\hat{\mathbf{\Lambda}}(z) \triangleq \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & z\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{I}\\ \mathbf{I} & \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{0} & \mathbf{I}\\ z\mathbf{I} & \mathbf{0}\end{bmatrix}$ as the permuted advance chain, then the modified LOT polyphase matrix becomes $\mathbf{E}(z) = \mathbf{C}_M^{II}\hat{\mathbf{\Lambda}}(z)\mathbf{P}$, where $\hat{\mathbf{\Lambda}}(z)\ \mathbf{P}$ can be interpreted as time-domain pre-processing across block boundaries. The new LT structure is illustrated in Fig. 3 where the resulting basis functions are, discounting $\frac{M}{2}$ sign changes from $\mathbf{D}_M$, exactly the type-II fast LOT's.

The synthesis polyphase matrix is simply the inverse of analysis polyphase matrix: $\mathbf{R}(z) = \mathbf{E}^{-1}(z) = \mathbf{P}^{-1}\hat{\mathbf{\Lambda}}^{-1}(z)\mathbf{C}_M^{II^T}$. The reader is reminded that $\mathbf{C}_M^{II^T}$ is the M-point type-II IDCT matrix. Following our convention, the synthesis bank is causal. Also, since every component in this lattice is paraunitary, we have $\mathbf{P}^{-1} = \mathbf{P}^T$. In other words, the post-filter is the transpose of the pre-filter. The

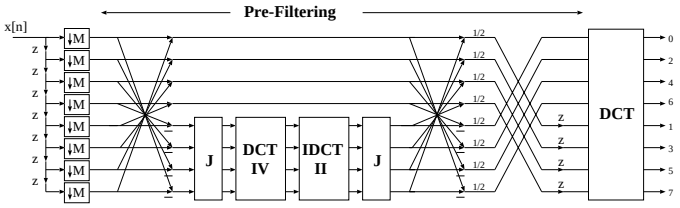

Fig. 3. LOT via time-domain pre- and post-filtering.

advance chain in $\hat{\mathbf{\Lambda}}(z)$ and the delay chain in $\hat{\mathbf{\Lambda}}^{-1}(z)$ place the pre-filter $\mathbf{P}$ and the post-filter $\mathbf{P}^{-1}$ squarely in between two adjacent DCT/IDCT blocks. Viewing the LT under the time-domain pre/post-filtering prism leads to numerous interesting solutions as demonstrated in Section IV.

C. Biorthogonal Extension

The matrix $\mathbf{V}$ in (5) and (4) controls pre- and post-filtering. It holds all of the degrees of freedom in the structure. If $\mathbf{V}$ is chosen orthogonal as in Fig. 3, we have an orthogonal solution. However, just to maintain FIR perfect reconstruction, $\mathbf{V}$ only has to be invertible. Notice that $\mathbf{V}$ as in (5) is already a product of two orthogonal matrix. Hence, we propose to insert an invertible diagonal matrix $\mathbf{S}$ between $\mathbf{C}_{M/2}^{II^T}$ and $\mathbf{C}_{M/2}^{IV}$ to represent $\mathbf{V}$ in the singular value decomposition form, $\mathbf{V} = \mathbf{J}\mathbf{C}_{M/2}^{II^T}\mathbf{S}\mathbf{C}_{M/2}^{IV}\mathbf{J}$. To minimize the additional complexity in the biorthogonal systems and maintain the nice closed-form of the solution, we limit $\mathbf{S}$ to $diag\{s, 1, \ldots, 1\}$ where s is a scaling factor. A good value of s for a smooth image model is $\frac{8}{5}$; other scaling factors that also work well include $\frac{3}{2}$, $\frac{25}{16}$, $\frac{1+\sqrt{5}}{2}$, and $\sqrt{2}$. These choices of s, revisited in a later section, follow the simple construction of the lapped biorthogonal transform (LBT) proposed in [31].

IV. General Pre- and Post-Processing Framework

A. Global Viewpoint

Although the new structure in Fig. 3 does not look that much more intriguing than that in Fig. 1, viewing the structure globally as shown in Fig. 4 reveals its elegance. In the decomposition stage, $\mathbf{P}$ acts as the pre-filter working across the block boundaries, taking away inter-block correlation; the pre-processed time samples are then fed to the DCT to be encoded as usual. In the reconstruction stage, $\mathbf{P}^{-1}$ serves as the post-filter, reconstructing the data in an overlapping manner, hence alleviating blocking artifacts. The symmetry of the basis functions is guaranteed by the specific structure of $\mathbf{P}$ as in (4) regardless of the choice of the free-parameter matrix $\mathbf{V}$. The advance chain $\hat{\mathbf{\Lambda}}(z)$ extends the processing across the block boundary. Pre- and post-filtering are operating in the time domain, completely outside of the existing block-based architecture. Because of this characteristic, we label this LT family the time-domain lapped transform (TDLT).

B. Time-Domain Derivation

The pre- and post-filter can be designed directly in the time domain. In fact, one issue that requires immediate attention is the generality of the solution in Section III. Do other solutions besides $\mathbf{V}$ in (5) exist? What is the most general form of $\mathbf{P}$? Does $\mathbf{P}$ necessarily have the structure in (4)?

To start, consider in Fig. 4 the mapping of $2M$ input samples to the M input of the DCT. The corresponding mapping operator $\mathbf{H}_{pre}$ is actually an $M \times 2M$ forward LT:

$$\mathbf{H}_{pre} = \begin{bmatrix} \mathbf{0}_{M/2} & \mathbf{I}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \end{bmatrix} \begin{bmatrix} \mathbf{P} & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{P} \end{bmatrix}. \tag{6}$$

Let $\mathbf{P} = \begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} \\ \mathbf{P}_{10} & \mathbf{P}_{11} \end{bmatrix}$, where $\mathbf{P}_{ij}$ are arbitrary matrices of size $\frac{M}{2} \times \frac{M}{2}$ as long as $|\mathbf{P}| \neq 0$. The forward transform matrix $\mathbf{H}$ can then be expressed as

$$\begin{aligned} \mathbf{H} &= \mathbf{C}_M^{II}\,\mathbf{H}_{pre} = \tfrac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{C}_{M/2}^{II} & \mathbf{C}_{M/2}^{II}\mathbf{J} \\ \mathbf{C}_{M/2}^{IV} & -\mathbf{C}_{M/2}^{IV}\mathbf{J} \end{bmatrix} \\ &\times \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} & \mathbf{0} & \mathbf{0} \\ \mathbf{P}_{10} & \mathbf{P}_{11} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_{00} & \mathbf{P}_{01} \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_{10} & \mathbf{P}_{11} \end{bmatrix} \\ &= \tfrac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{C}_{M/2}^{II}\mathbf{P}_{10} & \mathbf{C}_{M/2}^{II}\mathbf{P}_{11} & \cdots \\ \mathbf{C}_{M/2}^{IV}\mathbf{P}_{10} & \mathbf{C}_{M/2}^{IV}\mathbf{P}_{11} & \cdots \\ \cdots & \mathbf{C}_{M/2}^{II}\mathbf{J}\mathbf{P}_{00} & \mathbf{C}_{M/2}^{II}\mathbf{J}\mathbf{P}_{01} \\ \cdots & -\mathbf{C}_{M/2}^{IV}\mathbf{J}\mathbf{P}_{00} & -\mathbf{C}_{M/2}^{IV}\mathbf{J}\mathbf{P}_{01} \end{bmatrix}. \end{aligned}$$

Hence, to obtain linear-phase basis functions, we need $\mathbf{C}_{M/2}^{II}\mathbf{P}_{10} = \mathbf{C}_{M/2}^{II}\mathbf{J}\mathbf{P}_{01}\mathbf{J}$, $\mathbf{C}_{M/2}^{II}\mathbf{P}_{11} = \mathbf{C}_{M/2}^{II}\mathbf{J}\mathbf{P}_{00}\mathbf{J}$, $\mathbf{C}_{M/2}^{IV}\mathbf{P}_{10} = \mathbf{C}_{M/2}^{IV}\mathbf{J}\mathbf{P}_{01}\mathbf{J}$, and $\mathbf{C}_{M/2}^{IV}\mathbf{P}_{11} = \mathbf{C}_{M/2}^{IV}\mathbf{J}\mathbf{P}_{00}\mathbf{J}$, which leads to $\mathbf{P}_{10} = \mathbf{J}\mathbf{P}_{01}\mathbf{J}$ and $\mathbf{P}_{11} = \mathbf{J}\mathbf{P}_{00}\mathbf{J}$. Therefore, the most general pre-filter $\mathbf{P}$ generating symmetric basis functions is

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{01} \\ \mathbf{J}\mathbf{P}_{01}\mathbf{J} & \mathbf{J}\mathbf{P}_{00}\mathbf{J} \end{bmatrix}. \tag{7}$$

Exploiting the symmetry of $\mathbf{P}$ in (7), we arrive at the following factorization

$$\mathbf{P} = \tfrac{1}{2} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{U} & \mathbf{0} \\ \mathbf{0} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix}, \tag{8}$$

where $\mathbf{U} \triangleq \mathbf{P}_{00} + \mathbf{P}_{01}\mathbf{J}$ and $\mathbf{V} \triangleq \mathbf{J}\mathbf{P}_{00}\mathbf{J} - \mathbf{J}\mathbf{P}_{01}$. To obtain an orthogonal solution, choose $\mathbf{U}$ and $\mathbf{V}$ as orthogonal matrices. To obtain a biorthogonal solution, choose $\mathbf{U}$ and $\mathbf{V}$ as invertible matrices. It turns out that $\mathbf{U}$ helps little in improving energy compaction. In this paper, for simplicity of presentation, $\mathbf{U}$ is ignored (set to $\mathbf{I}$). Certainly, the choice of $\mathbf{U} = \mathbf{I}$ helps keep the complexity of the pre/post-filter down. Also, note that if the block transform operator is not fixed, $\mathbf{U}$ (or $\mathbf{V}$) can be moved across the butterfly and embed into the block transform stage via several simple matrix manipulations.

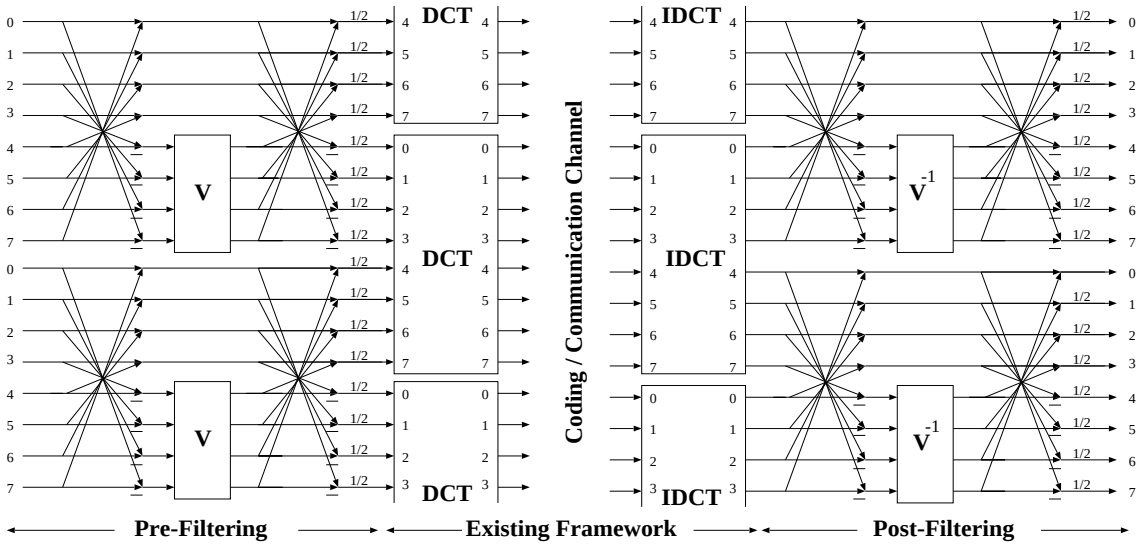

Fig. 4. Global viewpoint of LT as pre- and post-filtering at DCT/IDCT block boundaries.

C. LT with Small Overlap

The type-II fast LOT in Fig. 1 and our new variation in Fig. 3 is quite restrictive: it can only generate LT with an even number of channel and a 50% overlap ($L = 2M$). It is crucial in many applications to find the best trade-off between coding performance and complexity. It is desirable to be able to vary the amount of overlap between neighboring blocks (let $M < L \leq 2M$), and hence, have fine control over the computational complexity of the transform.

The answer to the seemingly complex question above keys on a simple observation of the global structure as shown in Fig. 4. The amount of overlap can be lowered by reducing the size of the pre-processing matrix $\mathbf{P}$. An $M \times L$ LT where $L \leq 2M$ and $M \geq 2$ can be easily constructed with the $(L-M) \times (L-M)$ pre-filter $\hat{\mathbf{P}}$, which has the same form as $\mathbf{P}$ in (4) except that all sub-matrices are now of size $\frac{L-M}{2}$. The smaller free-parameter $\hat{\mathbf{V}}$ matrix can be chosen as

$$\hat{\mathbf{V}} = \mathbf{J}\, \mathbf{C}^{II^T}_{\frac{L-M}{2}}\, \mathbf{S}\, \mathbf{C}^{IV}_{\frac{L-M}{2}}\, \mathbf{J}, \tag{9}$$

where $\mathbf{S} = \mathbf{I}_{\frac{L-M}{2}}$ yields a family of orthogonal LTs while $\mathbf{S} = diag\{\frac{8}{5}, 1, \ldots, 1\}$ yields good biorthogonal solutions. In the 8×10 case, orthogonal symmetric solution does not exist: the lone degree of freedom is the scaling factor s – the degeneration of the scaling matrix $\mathbf{S}$.

Note that we can obtain LT of various numbers of bands and sizes by just controlling the matrix $\mathbf{V}$ in Fig. 4, i.e., an $M \times L$ LT ($M \leq L \leq 2M$) can be realized by employing

$$\mathbf{V} = \begin{bmatrix} \mathbf{J}\, \mathbf{C}^{II^T}_{\frac{L-M}{2}}\, \mathbf{S}\, \mathbf{C}^{IV}_{\frac{L-M}{2}}\, \mathbf{J} & \mathbf{0}_{\frac{L-M}{2} \times \frac{2M-L}{2}} \\ \mathbf{0}_{\frac{2M-L}{2} \times \frac{L-M}{2}} & \mathbf{I}_{\frac{2M-L}{2}} \end{bmatrix}. \tag{10}$$

The choice of $\mathbf{V}$ as a diagonal matrix yields the pioneering results in [29], [30]. If $\mathbf{V} = \mathbf{I}$, then pre- and post-filtering are turned off.

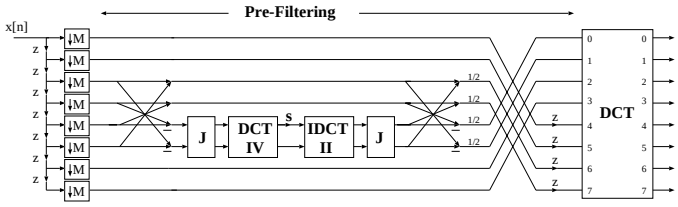

Fig. 5. An example of LT with small overlap (8×12 biorthogonal via 4-point pre-filtering) in polyphase representation.

D. Interesting Observations

Several basic properties of our framework are best explained through an almost trivial example – the case of 2×2 pre- and post-filtering depicted in Fig. 6. In this case, $\mathbf{C}^{IV}_1 = 1$, $\mathbf{C}^{II^T}_1 = 1$, and our proposed solution $\mathbf{V}$ in (9) degenerates to the single scaling factor s.

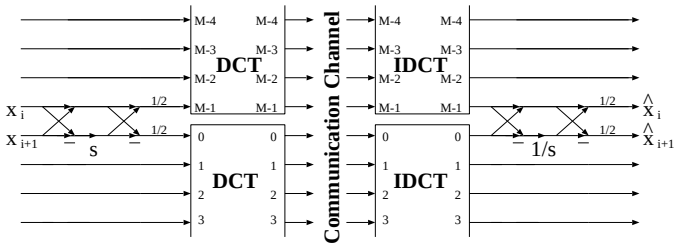

Fig. 6. Demonstration of 2-point pre- and post-filtering.

Let $\{x_i\}$, $\{p_i\}$, $\{\hat{p}_i\}$, and $\{\hat{x}_i\}$ be the set of the pre-filter's input samples, the pre-filter's output samples, the post-filter's input samples, and the post-filter's output samples, respectively. We examine the post-filter first since its operation is slightly more intuitive than the pre-filter's. Consider $\hat{p}_i$ and $\hat{p}_{i+1}$, two input samples of the post-filter at the boundary of two neighboring IDCT blocks, and their corresponding output samples, $\hat{x}_i$ and $\hat{x}_{i+1}$. We can easily

derive the following relationships

$$\begin{aligned}\hat{x}_i &= \frac{1}{2}[(\hat{p}_i + \hat{p}_{i+1}) + \frac{1}{s}(\hat{p}_i - \hat{p}_{i+1})] \\ &= \hat{p}_i + \frac{1/s-1}{2}(\hat{p}_i - \hat{p}_{i+1}) \qquad (11)\\ \hat{x}_{i+1} &= \frac{1}{2}[(\hat{p}_i + \hat{p}_{i+1}) - \frac{1}{s}(\hat{p}_i - \hat{p}_{i+1})] \\ &= \hat{p}_{i+1} - \frac{1/s-1}{2}(\hat{p}_i - \hat{p}_{i+1}). \qquad (12)\end{aligned}$$

Define a crude measure of blocking artifact D as the absolute difference between two samples at the block boundary. Without post-filtering, $D = |\hat{p}_i - \hat{p}_{i+1}|$. With post-filtering,

$$\begin{aligned}\hat{D} = |\hat{x}_i - \hat{x}_{i+1}| &= |(\hat{p}_i - \hat{p}_{i+1}) + (1/s-1)(\hat{p}_i - \hat{p}_{i+1})| \\ &= |(\hat{p}_i - \hat{p}_{i+1})/s| \;=\; |1/s|\,|\hat{p}_i - \hat{p}_{i+1}|.\end{aligned}$$

Hence, by choosing $|1/s| < 1$, or $|s| > 1$, we guarantee a decrease in "blocking artifact". This post-filtering effect is demonstrated in Fig. 7. For example, if we choose $s = 2$, then the post-filter partitions the distance $D = |\hat{p}_i - \hat{p}_{i+1}|$ into 4 segments of equal length, moves $\hat{x}_i$ up one segment length from $\hat{p}_i$, and moves $\hat{x}_{i+1}$ down one segment length from $\hat{p}_{i+1}$. So, post-filtering adaptively reduces the observed discrepancy D by a factor of two.

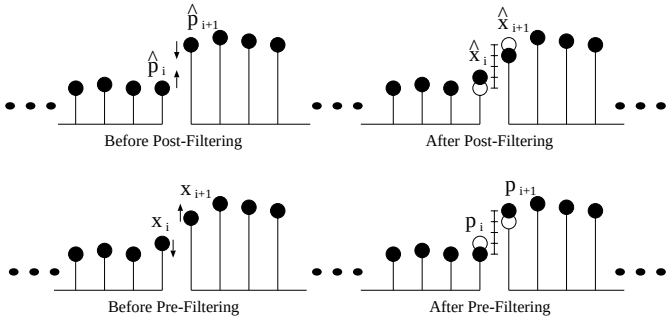

Fig. 7. Pre- and post-filtering effect.

The pre-filter modifies the samples in the opposite direction: attempting to lower the smaller-value sample x_i while increase the larger-value x_{i+1} based on their difference $|x_i - x_{i+1}|$. The input-output relationships are

$$\begin{aligned}p_i &= x_i + \frac{s-1}{2}(x_i - x_{i+1}) \qquad (13)\\ p_{i+1} &= x_{i+1} - \frac{s-1}{2}(x_i - x_{i+1}). \qquad (14)\end{aligned}$$

From (13) and (14), the choice $s > 1$ makes intuitive sense. If $s < 1$, samples are adjusted in the wrong direction. We found that a good choice of s in the energy compaction sense for smooth signal models is the *Golden Ratio* $\frac{1+\sqrt{5}}{2}$. In fact, all of the *Fibonacci ratio* $\frac{F(n+1)}{F(n)}$, where $F(n)$ is the n-th member of the Fibonacci sequence $\{1, 1, 2, 3, 5, 8, 13, 21, \ldots\}$, work well. Also, note that $\lim_{n\to\infty} \frac{F(n+1)}{F(n)} = \frac{1+\sqrt{5}}{2}$.

When more than 2 samples are involved, the pre-filter acts as a *flattening* operator. It attempts to make the input to the DCT as homogeneous as possible; hence, improving the overall energy compaction. This is quite consistent with most pre-filtering schemes in practice: smoothing the input signal improves coding efficiency. However, in our framework, perfect reconstruction is maintained. High-frequency signal components are never eliminated; they are only slightly shifted in time. We take full advantage of the block-based framework by carefully aligning high-frequency components at block boundaries. Discontinuities between DCT blocks, i.e., high-frequency contents, do not affect coding performance whereas, within each block, data samples are smoothened out, enhancing the DCT's effectiveness in energy compaction.

The flattening property of the pre-filter is best demonstrated in an image processing example shown in Fig. 8. When more samples are borrowed at each block boundary, the pre-filtered image becomes more blocky since each 8×8 block becomes smoother, and more high-frequency components are shifted to the block boundary. Notice that in 2D applications, the decomposition can be written as

$$\mathbf{y}_{m,n} \;=\; \mathbf{C}_M^{II}\mathbf{H}_{pre}\,\mathbf{x}_{m,n}\,\mathbf{H}_{pre}^T\mathbf{C}_M^{II^T}. \qquad (15)$$

Based on (15), transformation steps can be performed separably and in many different orders. In the illustration of Fig. 8, only separable 2D pre-processing is carried out.

It is also interesting to observe that when setting $M = 2$ and $s = 2$ the pre/post-filtering framework generates the following polyphase matrix

$$\begin{aligned}\mathbf{E}(z) &= \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}0 & 1\\ z & 0\end{bmatrix} \\ &\times\; \frac{1}{2}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1 & 0\\ 0 & 2\end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \\ &= \frac{\sqrt{2}}{4}\begin{bmatrix}-1+3z & 3-z\\ -1-3z & 3+z\end{bmatrix}, \qquad (16)\end{aligned}$$

which yields the scaled versions of the 4/4 spline wavelet filters: $H_0(z) = -\frac{1}{4} + \frac{3}{4}z + \frac{3}{4}z^2 - \frac{1}{4}z^3$ and $H_1(z) = -\frac{1}{4} + \frac{3}{4}z - \frac{3}{4}z^2 + \frac{1}{4}z^3$.

The 4-tap de-blocking post-filter in H.263+ [23] can be represented in our framework as well by choosing the 4×4 $\mathbf{P}^{-1}$ having the same form as $\mathbf{P}$ in (4) but with the following parameter matrix

$$\mathbf{V}^{-1} \;=\; \begin{bmatrix}0 & 1/4\\ 0 & 1/2\end{bmatrix}.$$

Despite its good de-blocking property, $\mathbf{P}^{-1}$ does not have a corresponding pre-filter: its inverse does not exist.

E. Lifting-Based Pre- and Post-Filtering

To map integers to integers with perfect reconstruction and minimum bit expansion, we can replace each butterfly in the pre-filter $\mathbf{P}$ in (4) by the unnormalized Haar as

Fig. 8. Pre-filtering's block-wise flattening effect with 8×8 block size. From left to right: original image; after 2-point pre-filtering (borrowing 1 sample at each boundary); after 4-point pre-filtering (borrowing 2 samples); after 6-point pre-filtering (borrowing 3 samples); after 8-point pre-filtering (borrowing 4 samples).

follows

$$\begin{aligned}\mathbf{P} &= \tfrac{1}{2}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix}\\ &= \begin{bmatrix}\frac{1}{2}\mathbf{I} & \mathbf{J}\\ \frac{1}{2}\mathbf{J} & -\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \frac{1}{2}\mathbf{J} & -\frac{1}{2}\mathbf{I}\end{bmatrix}. \quad (17)\end{aligned}$$

All-lifting pre-filter can be constructed by modeling the free-parameter $\mathbf{V}$ matrix in the LU decomposition form. This will be explored further in Section V.

F. Boundary Handling for Finite-Length Signals

To process finite-length signals using transforms with overlapping symmetric basis functions, symmetric extension is applied at the signal boundaries to achieve perfect reconstruction [5], [6]. In our framework, since the pre-processor $\mathbf{P}$ is placed between block boundaries, it is intuitive that there should be no extra processing needed at the signal boundaries. In fact, suppose that we are performing decomposition using an $M\times L$ symmetric LT and symmetrically extend the first $\frac{L-M}{2}$ samples, then the reflected signal always flows by the pre-processor untouched:

$$\begin{aligned}\mathbf{P}\begin{bmatrix}\mathbf{J}\mathbf{x}_0\\ \mathbf{x}_0\end{bmatrix} &= \tfrac{1}{2}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{V}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{J}\mathbf{x}_0\\ \mathbf{x}_0\end{bmatrix}\\ &= \tfrac{1}{2}\begin{bmatrix}\mathbf{I} & \mathbf{J}\\ \mathbf{J} & -\mathbf{I}\end{bmatrix}\begin{bmatrix}2\mathbf{J}\mathbf{x}_0\\ \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{J}\mathbf{x}_0\\ \mathbf{x}_0\end{bmatrix}, \quad (18)\end{aligned}$$

suggesting that pre- and post-filtering at the signal boundaries should be skipped.

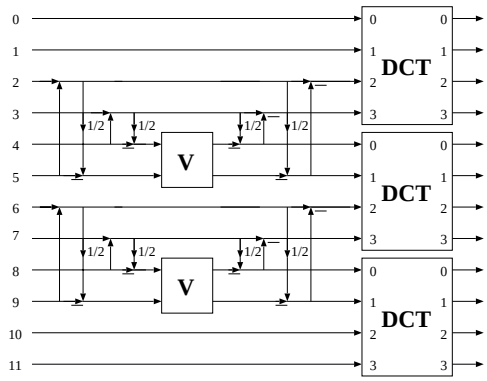

Fig. 9. Boundary handling for finite-length signals.

An example of the boundary handling technique in our pre/post-filtering framework is illustrated in Fig. 9. A 12-point finite length signal is decomposed by 4×4 block DCT and 4×4 pre-filter (or in other words, an 4×8 LT). Pre-filtering operators are placed between DCT blocks, but not at the two signal boundaries. From a pre-filtering perspective, there is no inter-block correlation to take advantage of at the boundary. From a post-filtering perspective, there is no blocking artifacts to concern about at the boundary. Note that in Fig. 9 we have also chosen to demonstrate the replacement of the common butterfly by the unnormalized forward/inverse Haar (also known as the S transform) following the idea proposed in Section IV-E.

G. Arbitrary Overlap Solution

To increase the amount of overlap to any arbitrary number $L\geq 2M$ (M even), more stages of pre- and post-processing are added as shown in Fig. 10 where each added processing stage $\mathbf{P}_i$ works at the boundaries of the previous stage $\mathbf{P}_{i-1}$. Thus, $\mathbf{P}_i$ is aligned with $\mathbf{P}_{i-2}$. The analysis polyphase matrix of a general $M\times(KM+N)$ symmetric LT can be constructed modularly as follows

$$\mathbf{E}(z) \;=\; \mathbf{C}_M^{II}\prod_{i=1}^{K}[\;\hat{\mathbf{\Lambda}}(z)\,\mathbf{P}_i\;]. \quad (19)$$

The corresponding synthesis polyphase matrix is

$$\mathbf{R}(z) \;=\; \prod_{i=K}^{1}[\;\mathbf{P}_i^{-1}\hat{\mathbf{\Lambda}}^{-1}(z)\;]\,\mathbf{C}_M^{II^T}. \quad (20)$$

This general solution is demonstrated in Fig. 10. Each stage of $\mathbf{P}_i$ possibly employs a different $\mathbf{V}_i$ matrix. The first $(K-1)$ stages in (19) generate LT of length KM whereas the last stage $\mathbf{P}_K$ is responsible for the remaining N. We can also think of the DCT as the initial operator $\mathbf{P}_0$. If linear phase is not required, then $\mathbf{P}_i$ does not need have any structure. It only has to be invertible. In the most general form, $\mathbf{P}_i$ can even be a non-linear operator.

It is trivial to see that (19) generates FIR perfect reconstruction systems as long as $\mathbf{V}_i$ are invertible. To structurally guarantee linear phase basis functions, $\mathbf{E}(z)$ in (19) has to satisfy the LP test [5]

$$\mathbf{E}(z) = z^K\,\mathbf{D}\,\mathbf{E}(z^{-1})\,\mathbf{J}, \quad (21)$$

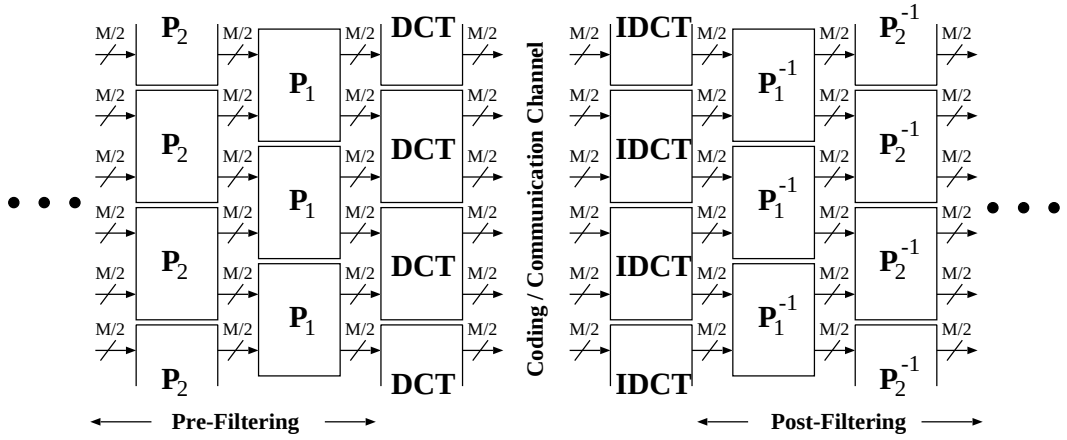

Fig. 10. General pre- and post-filter constructed from cascading modular structures.

where K is the order of the anticausal polyphase matrix. To show that (19) satisfies (21), let us first establish

$$z\mathbf{J}\hat{\mathbf{\Lambda}}(z^{-1})\mathbf{J} = z\begin{bmatrix} \mathbf{0} & \mathbf{J} \\ \mathbf{J} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ z^{-1}\mathbf{I} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{0} & \mathbf{J} \\ \mathbf{J} & \mathbf{0} \end{bmatrix} = \hat{\mathbf{\Lambda}}(z). \tag{22}$$

Substituting (19) into the right side of (21) yields

$$z^K\ \mathbf{D}\ \mathbf{E}(z^{-1})\ \mathbf{J}\ =\ z^K\ \mathbf{D}\ \mathbf{C}_M^{II}\ \prod_{i=1}^{K}\ [\ \hat{\mathbf{\Lambda}}(z^{-1})\ \mathbf{P}_i\ ]\ \mathbf{J}. \tag{23}$$

Next, using $\mathbf{C}_{M/2}^{II}\ =\ \mathbf{D}\ \mathbf{C}_{M/2}^{II}\ \mathbf{J}$ and (22), we obtain

$$z^K\ \mathbf{D}\ \mathbf{E}(z^{-1})\ \mathbf{J}\ =\ \mathbf{D}\mathbf{D}\mathbf{C}_M^{II}\mathbf{J}\ \prod_{i=1}^{K}[z\hat{\mathbf{\Lambda}}(z^{-1})\mathbf{P}_i]\mathbf{J}$$

$$=\ \ \mathbf{C}_M^{II}\mathbf{J}\prod_{i=1}^{K}[z\mathbf{J}\mathbf{J}\hat{\mathbf{\Lambda}}(z^{-1})\mathbf{J}\mathbf{J}\mathbf{P}_i]\mathbf{J} = \mathbf{C}_M^{II}\mathbf{J}\prod_{i=1}^{K}[\mathbf{J}\hat{\mathbf{\Lambda}}(z)\mathbf{J}\mathbf{P}_i]\mathbf{J}.$$

Finally, taking advantage of the symmetry of $\mathbf{P}_i$ in (37), we can simplify the previous equation to

$$\begin{aligned} z^K\ \mathbf{D}\ \mathbf{E}(z^{-1})\ \mathbf{J} &= \mathbf{C}_M^{II}\mathbf{J}\mathbf{J}\hat{\mathbf{\Lambda}}(z)\mathbf{J}\mathbf{P}_1\mathbf{J}\hat{\mathbf{\Lambda}}(z)\mathbf{J}\mathbf{P}_2\ldots\mathbf{J}\hat{\mathbf{\Lambda}}(z)\mathbf{J}\mathbf{P}_K\mathbf{J} \\ &= \mathbf{C}_M^{II}\hat{\mathbf{\Lambda}}(z)\mathbf{P}_1\hat{\mathbf{\Lambda}}(z)\mathbf{P}_2\ldots\hat{\mathbf{\Lambda}}(z)\mathbf{P}_K = \mathbf{E}(z). \end{aligned}$$

In short, the modular construction in (19) always generates transforms with linear phase FIR basis functions regardless of the choices of invertible matrices $\mathbf{V}_i$ in $\mathbf{P}_i$. Note that the result in this section is only for even M. The odd-M case is slightly more complicated and will be presented in the near future.

H. Adaptive Time-Varying Pre- and Post-Processing

A quick review of Fig. 4 and (10) reveals that our pre/post-processing framework lends itself nicely to the problem of designing adaptive time-varying signal decomposition. Adaptivity can lead to significant coding improvements if the amount of side information is kept to a minimum. It is clear that long basis functions are best for smooth signal regions. However, long basis functions cause ringing artifacts near strong edges and texture regions. To keep both blocking and ringing artifacts under control, we have three options: *(i)* vary the number of overlapping samples at each block boundary; *(ii)* vary the transform block size; or *(iii)* a combination of both of the above.

Based on the energy of the transform coefficients generated, we can decide to turn on or off pre/post-filtering. It is just as easy to vary the number of borrowing samples dynamically. This adaptive-borrowing signal decomposition is illustrated in Fig. 11(a) where the block size is fixed to 4 while the pre-filtering operator can be chosen amongst: no filtering, borrowing 1 sample, or borrowing 2 samples. In other words, from top to bottom, we are switching from a 4×6 to a 4×7 to a 4×5 LT and possibly to a 4×4 DCT. Unfortunately, the linear phase property of the filters in the switching filter banks (4×7 and 4×5) has been sacrificed.

If the block transform in use is the 8-point DCT and the number of borrowing samples can be chosen from the set $\{0, 1, 2, 4\}$, then the side information for each block boundary is 2 bits. This side information can be a lot lower if it is coded by Huffman or arithmetic coding.

Another adaptive decomposition scheme can be obtained by employing variable block sizes. In slowly-changing part of the signal, it is desirable to employ a large block size. In fast-changing transient part of the signal, it is more advantageous to switch to a small block size. Such a signal-adaptive switching scheme has proven to be very effective in practice. For instance, MPEG-4's Advanced Audio Coder switches back-and-forth between a 256-point high-time-resolution short window and a 2048-point high-frequency-resolution long window to avoid pre-echo and to improve coding efficiency [32]. In our pre/post-filtering framework, an example of a variable-block-size decomposition scheme is depicted in Fig. 11(b) where two samples are borrowed at each boundary. Here, we are switching from a 4×8 to a 8×12 to a 6×10 LT. Interestingly, in this time-varying example, there is no switching filter bank and every filter involved has linear phase. The side information, just like in the adaptive-borrowing case, can be kept manageable as well with a well chosen set of block sizes, e.g., $\{4, 8, 16, 32\}$ for image applications.

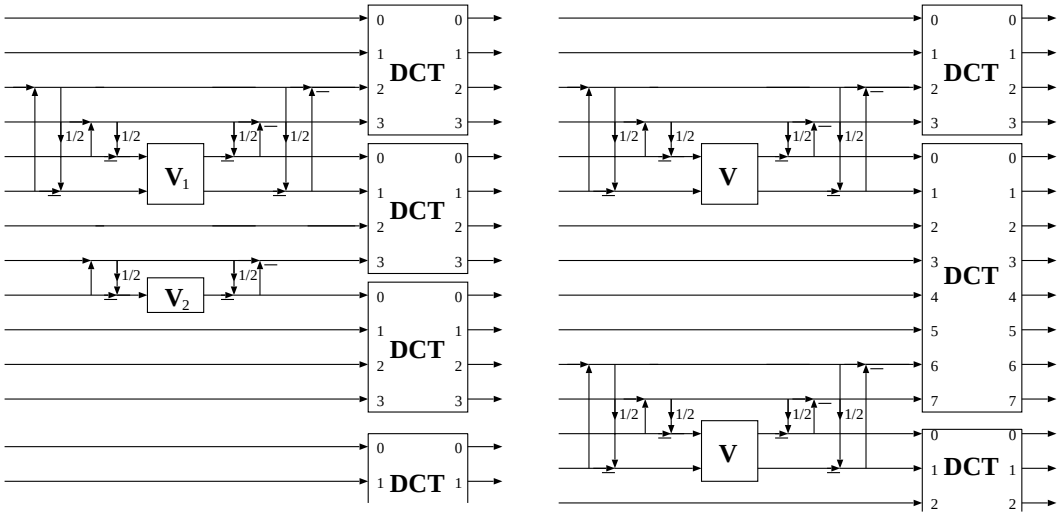

Fig. 11. Adaptive time-varying pre-filtering. Left: Adaptive borrowing, fixed block size. Right: Adaptive block size, fixed borrowing.

In the most general case, both adaptive pre-filtering with different lengths, with multiple stages and adaptive variable block size can be combined. This decomposition scheme generates a large library of basis functions that the encoder can choose from depending on the input signal behavior. By having well-behaved structured solutions, the encoder can perform fast quasi-optimal dynamic optimizations on-line much like the motion estimation problem in video coding. How to make the right decision quickly and how to minimize the amount of side information are two important open research problems.

To conclude the section, we remark that under the proposed pre/post-filtering framework, the design of odd-band LTs becomes a simple and straightforward extension. In fact, the same pre-filters presented in this section can be combined with odd-size DCTs to realize LTs with an odd number of channels. Unfortunately, these solutions do not seem to offer any advantage over the even-band solutions.

V. Design

A. The Optimized TDLT

In this section, we present the highest coding gains that can be achieved when the matrix $\mathbf{V}$ in the TDLT is allowed to be any orthogonal or invertible matrix. An unconstrained optimization program is set up to find the optimal coding gain of the orthogonal or biorthogonal $M\times(M+2N)$ TDLT, where $N \leq \lfloor M/2 \rfloor$.

It is well known that any $N \times N$ orthogonal matrix can be factored as a cascade of $N(N-1)/2$ plane rotations and N sign parameters [33]. This representation is highly non-unique. One example for a 4×4 matrix $\mathbf{V}$ is shown in Fig. 12. The free parameters for the orthogonal pre-filter are the $N(N-1)/2$ rotation angles. In the biorthogonal case, we use $N(N-1)$ rotation angles and N diagonal entries following the SVD model.

Table I compares the coding gains of various lapped transforms, where *Opt. TDLOT* and *Opt. TDLT* denote the optimized orthogonal and biorthogonal TDLT,

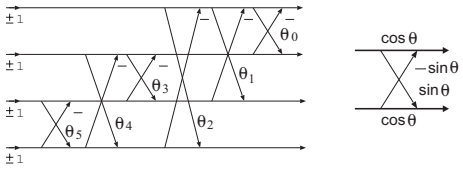

Fig. 12. Representation of an orthogonal matrix by rotation angles.

obtained when $\mathbf{V}$ is chosen as an arbitrary orthogonal or biorthogonal matrix, respectively. The TDLT developed in (5) is labeled as *TDLOT-I*, whose performance is identical to that of the *LOT-II*. The results of *TDLT-I* are obtained from (5) by inserting a single scaling factor of 8/5. The results of *Opt. LOT* are obtained by choosing the matrix $\mathbf{V}$ in the LOT as the corresponding Karhunen-Loève transform (KLT) [6].

Except for the case of $M = 4$, the optimized TDLOT achieves slightly higher coding gains than the optimal LOT. The 8×16 optimized TDLT has a coding gain of $9.62\,dB$, which is impressively close to the optimal $9.63\,dB$ in [34] and the optimized GLBT in [12]. However, the TDLT has a much simpler structure than both of the above. Table I also shows that the coding gains of the TDLOT-I and the LOT-II are only below the optimized cases by up to $0.04\,dB$. The frequency responses of some optimized TDLTs are shown in Fig. 13, together with their impulse responses. These are quite close to the LBT basis functions in [31].

B. Plane Rotation-based Fast TDLOT-II

We observe that the significant entries of the matrix $\mathbf{V}$ in the optimized pre-filter concentrate along the diagonal. This is also true for the matrix $\mathbf{V}$ in the TDLOT-I, as defined in (5). For example, the rotation angles corresponding to the 4×4 matrix $\mathbf{V}$ in TDLOT-I are

$$\theta = [-0.20\pi,\ 0.04\pi,\ -0.03\pi,\ -0.15\pi, -0.01\pi,\ -0.08\pi].$$

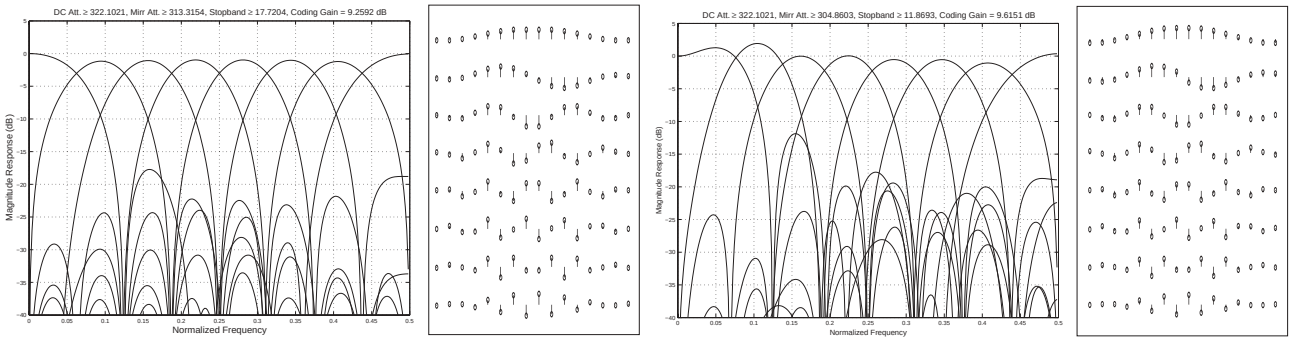

Fig. 13. Design examples. Left: Orthogonal 8×16 TDLT; coding gain 9.26 dB. Right: Biorthogonal 8×16 TDLT; coding gain: 9.62 dB.

TABLE II
CODING GAIN IN DB OF VARIOUS FAST TDLTS FOR AN AR(1) SIGNAL WITH $\rho = 0.95$

Size	TDLOT-II	TDLT-III (no scale)	TDLT-III (with scale)	TDLT-IV	TDLT-V
4×6	7.57	8.07	8.07	8.07	8.07
4×8	7.94	8.13	8.63	8.63	8.63
8×10	8.83	9.06	9.06	9.06	9.06
8×12	9.00	9.08	9.34	9.34	9.34
8×14	9.14	9.25	9.50	9.50	9.50
8×16	9.26	9.38	9.61	9.60	9.60
16×32	9.80	9.83	9.95	9.91	9.90
32×64	10.00	9.99	10.06	10.00	9.98
64×128	10.07	10.05	10.09	10.04	9.96

TABLE I
CODING GAIN IN DB OF VARIOUS LAPPED TRANSFORM FOR AN AR(1) SIGNAL WITH $\rho = 0.95$.

Size	TDLOT-I & LOT-II	Opt. LOT	Opt. TDLOT	TDLT-I	Opt. TDLT
4×6	7.57	-	7.57	8.04	8.07
4×8	7.93	7.95	7.94	8.57	8.63
8×10	8.83	-	8.83	9.06	9.06
8×12	8.99	-	9.00	9.31	9.34
8×14	9.11	-	9.14	9.45	9.50
8×16	9.22	9.24	9.26	9.56	9.62
16×32	9.76	9.77	9.81	9.91	9.96
32×64	9.97	9.98	10.01	10.03	10.07

Since θ_1, θ_2 and θ_4 are relatively small, we expect that they can be discarded without significant performance loss. Also notice that the remaining angles have a strong decreasing trend.

The strong diagonal property of the optimized matrix $\mathbf{V}$ suggests a simplified orthogonal model for the matrix $\mathbf{V}$: a cascade of rotation angles between neighboring channels. The corresponding TDLT structure is shown in Fig. 14 for the case of $M = 8$. We denote this model as the *TDLOT-II*. For an $M \times (M + 2N)$ TDLOT, this simplified model only needs $N - 1$ rotation angles. Comparing to the TDLOT-I, the complexity of this algorithm is reduced significantly, enabling much faster implementation. Notice that the arrangement of rotation angles in TDLOT-II is different from that of the fast LOT-I in [6], [35], where the cascading of rotation angles starts from the top channels and propagates to the bottom of $\mathbf{V}$. The significance of this difference will be explained later in this section.

Coding gain results of the optimized TDLOT-II are presented in Table II, which also contains results of other fast TDLTs presented later in the section. Table I and Table II show that the TDLOT-II has better performance than the TDLOT-I. In fact, its coding gain is almost identical to that of the optimized TDLOT with full matrix model. This suggests that the simplified model in Fig. 14 is a very accurate approximation of the optimal results.

Table III lists the rotation angles in several optimized TDLOT-II. As previously mentioned, their magnitudes are steadily decreasing. The reason is quite intuitive. The prefilter in the TDLT framework is applied at the boundaries of neighboring signal blocks, and it tries to smoothen the input to the DCT in order to improve energy compaction. Each input to the matrix $\mathbf{V}$ is the difference between a pair of samples from two sides of a block boundary. Moreover, the upper inputs of $\mathbf{V}$ correspond to the differences of nearer sample pairs, whereas the lower inputs correspond to those of farther pairs. Since the correlations between nearer neighbors are stronger, it is clear that the upper inputs of $\mathbf{V}$ should have more weightings than the lower ones in pre-filtering.

For this kind of decreasing rotation angles, the simplified model in Fig. 14 starts from the smallest angles, therefore yields less accumulation error, making it valid even for large M. Compared with this, the main rotation angles in the LOT-I are very close to each other. As a result, the structure of LOT-I only yields good performance for $M \leq 16$ [6].

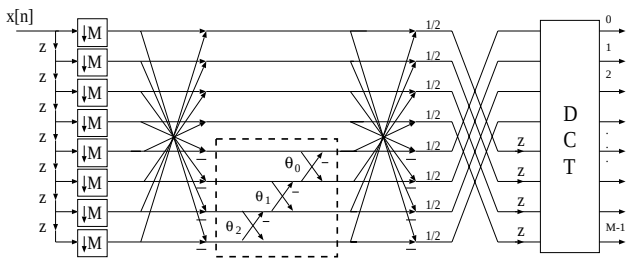

Fig. 14. Structure of the fast TDLOT-II.

TABLE III
OPTIMIZED ROTATION ANGLES FOR DIFFERENT PRE-FILTERS IN THE TDLOT-II

Size of $\mathbf{V}$	Rotation angles: θ_0 to θ_{N-2}
2×2	-0.10 π.
3×3	-0.15 π, -0.07 π
4×4	-0.17 π, -0.12 π, -0.05 π
5×5	-0.19 π, -0.15 π, -0.10 π, -0.04 π
6×6	-0.20 π, -0.17 π, -0.13 π, -0.09 π, -0.04 π
7×7	-0.21 π, -0.19 π, -0.16 π, -0.12 π, -0.08 π, -0.03 π
8×8	-0.21 π, -0.20 π, -0.18 π, -0.15 π, -0.11 π, -0.07 π, -0.03 π

C. Lifting-based fast algorithm

Although the simplified model in TDLOT-II is faster than the TDLT-I, it still involves floating multiplications, which are slow and undesired in many software and hardware applications. In this section, a lifting-based fast TDLT is developed, paving the path to much faster multiplierless solutions. The lifting scheme is proposed in [36] as a tool for filter bank design. More systematic and general results were presented in [37]. It is well known that a plane rotation can be decomposed into three lifting steps [36], [37]. This can be written in matrix form as

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} 1 & P \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ U & 1 \end{bmatrix} \begin{bmatrix} 1 & P \\ 0 & 1 \end{bmatrix}, \tag{24}$$

where

$$P = (\cos\theta - 1)/\sin\theta = -tan(\theta/2), \;\; U = \sin\theta. \tag{25}$$

To obtain fast implementation, we can approximate the floating-point lifting coefficients by hardware-friendly dyadic values (*i.e.*, rational values in the format of $k/2^m$; k, m are integers), which can be implemented by only shift and addition operations. The elimination of the multiplication can also reduce the dynamic range of the transform [1].

A trivial lifting-based pre-filter for the TDLT can be obtained from the TDLOT-II structure by replacing each rotation angle with its three-lifting representation in (24)-(25). However, since the rotation angles in the TDLOT-II are steadily decreasing, and notice that

$$\lim_{\theta \to 0} P \;=\; 0, \;\; \lim_{\theta \to 0} U \;=\; 0, \tag{26}$$

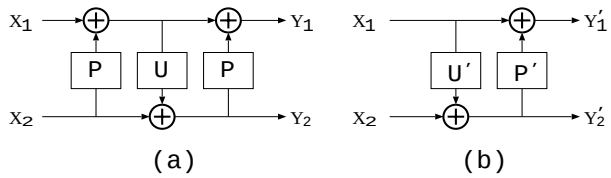

Fig. 15. Approximating a small rotation angle by two lifting steps. (a) Standard three-lifting representation. (b) Simplified two-lifting structure.

many lifting parameters thus have very small magnitudes. This enables the approximation of such rotation angles by only two lifting steps as shown in Fig. 15(b). This can be justified by the following analysis.

In Fig. 15(a), the outputs of the three-lifting structure can be written as:

$$\begin{aligned} Y_1 &= (1 + P\,U)X_1 + (2\,P + P^2\,U)X_2, \\ Y_2 &= U\,X_1 + (1 + P\,U)X_2, \end{aligned} \tag{27}$$

whereas the outputs of the simplified structure in Fig. 15(b) are:

$$\begin{aligned} Y_1{}' &= (1 + P'\,U')X_1 + P'\,X_2, \\ Y_2{}' &= U'\,X_1 + X_2. \end{aligned} \tag{28}$$

If the rotation angle is small enough such that the magnitudes of its lifting parameters are much less than unity, all second order and third order terms in (27) and (28) can be ignored, and the following setting of the two-lifting model can approximate the standard model closely:

$$P' = 2\,P, \;\; U' = U. \tag{29}$$

When the rotation angle is not small enough and only the third order term can be ignored, the following choice will yield a more accurate approximation:

$$P' = 2\,P, \;\; U' = U/2. \tag{30}$$

By replacing the rotation angles in the TDLOT-II with the two-lifting structure, we obtain another simplified model for the matrix $\mathbf{V}$ as illustrated in Fig. 16(a). The transform can be made even faster if each lifting parameter is approximated by an appropriate dyadic value. This is similar to the approach taken in the LiftLT design [38].

The structure in Fig. 16 (a) is designed as a close approximation of the orthogonal TDLOT-II, whose coding gain is not as high as the biorthogonal case. However, by introducing a scaling coefficient to each channel of the matrix $\mathbf{V}$ as shown in Fig. 16(b), the structure would approximate the SVD model very well. The corresponding biorthogonal TDLT is given in Fig. 17, denoted as the *TDLT-III*. The butterflies in the TDLT-III are also implemented by lifting steps as discussed previously in Section IV. For a $M \times (M + 2N)$ TDLT-III, $\mathbf{V}$ has only $3N - 2$ parameters, representing a dramatic simplification over the N^2 parameters of the SVD model.

Some coding gain results of TDLT-III are given in Table II, by optimizing the lifting and scaling parameters in

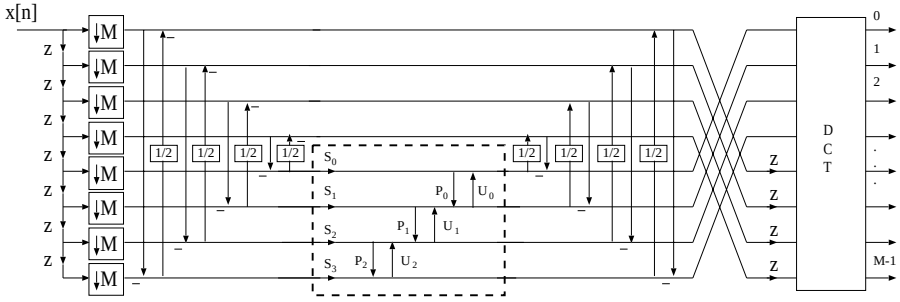

Fig. 17. General structure of the TDLT-III.

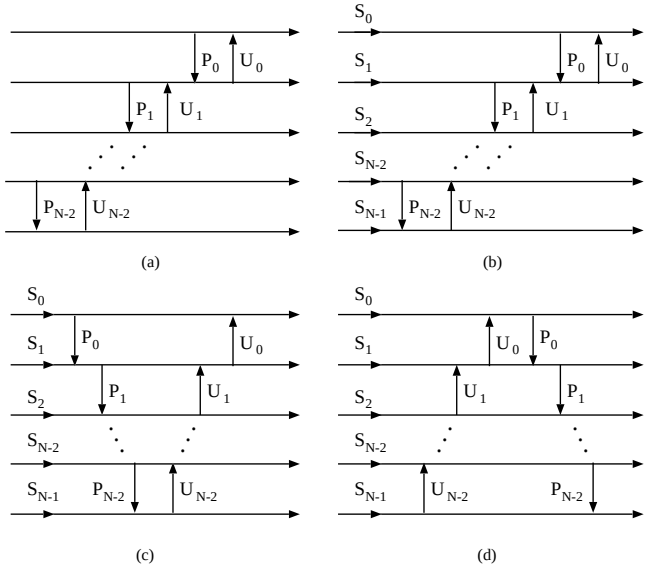

Fig. 16. Simplified structures for the matrix $\mathbf{V}$ in the pre-filter of the TDLT. (a) Structure that approximates the cascading of rotation angles. (b) Structure in the TDLT-III. (c) Structure in the TDLT-IV. (d) Structure in the TDLT-V.

Fig. 17. It can be observed that the optimized TDLT-III without scalings has better performance than its TDLOT-II original. When scalings are used, the result is virtually identical to the optimized SVD-based TDLT in Table I when $M \leq 16$. It is interesting to note that for $N = 2$, this model for $\mathbf{V}$ reduces to the LDU factorization of a matrix [39], which is equivalent to the SVD. Therefore the model in TDLT-III is a complete model for all invertible 2×2 matrices.

Besides the SVD decomposition, the LU factorization [39] provides another model for the invertible matrix $\mathbf{V}$ in the TDLT. Two simplified models that resemble the LU factorization and provide good performance in the TDLT framework are given in Fig. 16(c) and Fig. 16(d). We denote the corresponding TDLT as *TDLT-IV* and *TDLT-V*. They have the same complexity as the TDLT-III, *i.e.*, requiring N scalings and $2(N-1)$ lifting steps for $\mathbf{V}$. The scaling coefficients can be placed between the upper and lower triangular parts or at the end of the signal flow without losing any optimal performance.

D. Comparison of Complexity and Coding Performance

This section summarizes the computational complexity of various fast TDLTs developed in this paper when a floating-point implementation is considered. The fast DCT algorithms in [8] is used throughout. Define $\mu(M)$ and $\alpha(M)$ as the number of floating-point multiplications and additions per input data block required by an M-channel transform. The computational complexity of this fast DCT algorithm for even M is given by:

$$\begin{aligned} \mu_{DCT}(M) &= \frac{M}{2} log_2 M + 1, \\ \alpha_{DCT}(M) &= \frac{3M}{2} log_2 M - M + 1. \end{aligned} \tag{31}$$

The $M \times 2M$ LOT-II and its equivalence – the TDLOT-I – require roughly twice the DCT complexity [6]

$$\begin{aligned} \mu_{LOT-II}(M) &= M log_2 M + 2, \\ \alpha_{LOT-II}(M) &= 3M log_2 M - M + 2. \end{aligned} \tag{32}$$

For the TDLOT-II, since each rotation angle in the $\mathbf{V}$ matrix can be implemented with 3 multiplications and 3 additions [9], the complexity of a $M \times (M + 2N)$ TDLOT-II is:

$$\begin{aligned} \mu_{TDLOT-II}(M,\, 2N) &= \mu_{DCT}(M) + (3N - 3), \\ \alpha_{TDLOT-II}(M,\, 2N) &= \alpha_{DCT}(M) + (7N - 3). \end{aligned} \tag{33}$$

The complexity of the lifting-based $M \times (M + 2N)$ TDLT-III, TDLT-IV and TDLT-V is given by:

$$\begin{aligned} \mu_{TDLT-III}(M,\, 2N) &= \mu_{DCT}(M) + (3N - 2), \\ \alpha_{TDLT-III}(M,\, 2N) &= \alpha_{DCT}(M) + (6N - 2). \end{aligned} \tag{34}$$

The $\frac{1}{2}$ normalization of the butterflies is not counted in all of the formula above. Table IV compares the complexity between the DCT, the LOT-II, and various fast TDLTs with full borrowing. The TDLOT-II and TDLT-III reduces the computational overhead comparing to the fast DCT implementation in [8] to around 40% to 70%.

E. TDLT with Rational or Dyadic Coefficients

This section investigates the approximation of the optimized values for the free parameters in the TDLT-III and TDLT-IV by various rational and dyadic values. These

TABLE IV
COMPARISON OF COMPUTATIONAL COMPLEXITY BETWEEN THE DCT, THE LOT-II, TDLOT-II, AND THE TDLT-III

	Number of multiplications				Number of additions			
M	DCT	LOT-II/TDLOT-I	TDLOT-II	TDLT-III	DCT	LOT-II/TDLOT-I	TDLOT-II	TDLT-III
2	2	2	2	3	2	2	2	6
4	5	10	8	9	9	22	20	19
8	13	26	22	23	29	66	54	51
16	33	66	54	55	81	178	134	127
32	81	162	126	127	209	450	318	303
64	193	386	286	287	513	1090	734	703
128	449	898	638	639	1217	2562	1662	1599

TABLE V
EXAMPLES OF FAST TDLT-III AND TDLT-IV WITH RATIONAL OR DYADIC PARAMETERS

Size of TDLT	Cfg.	S_0	S_1	S_2	S_3	P_0	U_0	P_1	U_1	P_2	U_2	# of Muls.	# of Shifts	# of Adds.	Cod. Gain TDLT-III	Cod. Gain TDLT-IV
8×10	1	3/2	-	-	-	-	-	-	-	-	-	0.5	25.5	41.5	9.05	9.05
8×12	1	4/3	4/3	-	-	-3/16	1/2	-	-	-	-	1	31	50	9.32	9.32
	2	3/2	3/2	-	-	-3/16	1/2	-	-	-	-	1	31	50	9.27	9.27
	3	1	1	-	-	-3/16	1/2	-	-	-	-	0	30	48	9.07	9.07
8×14	1	3/2	5/4	5/4	-	-1/8	3/8	-3/8	3/4	-	-	1.5	37.5	59	9.47	9.47
	2	4/3	8/7	8/7	-	-1/8	3/8	-3/8	3/4	-	-	1.5	37.5	59	9.48	9.47
	3	1	1	1	-	0	3/8	-3/8	11/16	-	-	0	33	56	9.25	9.25
8×16	1	4/3	8/7	8/7	8/7	-1/16	1/4	-1/4	1/2	-3/8	3/4	2	40	65	9.59	9.58
	2	3/2	9/8	9/8	9/8	-1/16	1/4	-1/4	1/2	-3/8	3/4	2	40	65	9.58	9.57
	3	4/3	1	1	1	0	1/4	-1/4	1/2	-1/4	3/4	0.5	36.5	60	9.49	9.49
	4	3/2	1	1	1	0	1/4	-1/4	1/2	-1/4	3/4	0.5	36.5	60	9.47	9.47
	5	1	1	1	1	0	1/4	-1/4	1/2	-1/2	3/4	0	36	59	9.37	9.35

designs lead to fast, sometimes even multiplierless implementations, and also allow for lossless compression.

The fast DCT algorithm chosen for the following examples is the lifting-based binDCT [1], which is derived from the well-known Chen-Wang factorization of the DCT [8], [9]. This binDCT version needs 27 shifts and 37 additions, and has a coding gain of $8.82\,dB$ (the DCT has $8.83\,dB$). The final scalings of the binDCT should be combined with quantization to reduce the complexity even further.

Table V tabulates various rational approximations for $\mathbf{V}$ in the 8×10, 8×12, 8×14, and 8×16 TDLT-III and TDLT-IV. The complexity in it is computed by averaging that of the forward transform and the inverse transform. Compared to the results in Table II, the performance loss due to finite-length approximation of the optimized floating parameters and the binDCT is negligible. Reversible integer-to-integer mapping, a critical requirement for lossless compression, can be easily achieved by setting all scaling factors in the matrix $\mathbf{V}$ to be unity. In this case, both pre- and post-filter can be implemented with only shift and addition operations, simplifying the hardware implementation significantly.

The performances of lapped transform in compression have been thoroughly investigated [6], [10]. Instead of repeating these results here, we refer the interested reader to other publications [40], [41], [42] which address these applications much more adequately.

VI. CONCLUSION

This paper demonstrates that a large class of LT with an arbitrary number of channels and overlapping samples can be generated through time-domain pre-processing of DCT inputs and post-processing of IDCT outputs. The pre- and post-filtering module is placed between two neighboring DCT operators. The pre-filter acts like a flattening operator, trying to make the input data of each DCT block to be as homogeneous as possible. The post-filter plays the role of a smooth interpolator, eliminating or at least reducing blocking artifacts. We investigate the design of various pre-/post-filters – with closed-form, fast-computable algorithms as well as quasi-optimal energy compaction.

Comparing to previous LT constructions for image processing [6], [11], [27], [12], the proposed framework provides several advantages

- Existing block-based infrastructures can be kept intact and standard compliance is achievable.
- Trade-off between complexity and performance is easily obtained through varying the amount of borrowing samples, i.e., the support of the pre-/post-filter.
- The new pre- and post-filter designs provide slightly better coding performance at a lower computational complexity. Scaled DCT algorithms [2], [43] and multiplierless DCT approximations [1] can be applied to further lower complexity without seriously deteriorating energy compaction.

Finally, the link to pre- and post-filtering opens up many research directions. The general framework allows a great degree of flexibility. Adaptive time-varying systems can be easily designed by deciding what is the optimal pre- and post-filter to apply at every block boundary, by allowing variable block size, or by combining both.

VII. Acknowledgements

The author would like to thank Dr. H. S. Malvar, W. Dai, S. Gangaputra, Dr. R. L. de Queiroz, and Prof. T. Q. Nguyen for numerous inputs and discussions which significantly enhance the presentation of the paper.

Appendix

Starting with the type-II fast LOT in (2) and substituting the DST-IV by the DCT-IV, we have

$$\mathbf{E}(z) = \tfrac{1}{2} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J}\,\mathbf{C}^{II^T}_{M/2} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \times \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \tfrac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{C}^{II}_{M/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^{IV}_{M/2}\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \mathbf{J}_M.$$

Since any block-diagonal matrix can be moved across the butterfly and the delay chain, i.e.,

$$\begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \text{ and}$$
$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix},$$

$\mathbf{E}(z)$ can be modified as follows.

$$\mathbf{E}(z) = \tfrac{1}{2} \begin{bmatrix} \mathbf{C}^{II}_{M/2}\mathbf{C}^{II^T}_{M/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J}\,\mathbf{C}^{II^T}_{M/2} \end{bmatrix} \times \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \times \tfrac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{C}^{II}_{M/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^{IV}_{M/2}\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \mathbf{J}_M$$

$$= \tfrac{1}{2} \begin{bmatrix} \mathbf{C}^{II}_{M/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \times \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \tfrac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^{II^T}_{M/2}\mathbf{C}^{IV}_{M/2}\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \mathbf{J}_M.$$

Taking advantage of the symmetry of the type-II DCT, $\mathbf{C}^{II}_{M/2} = \mathbf{D}\,\mathbf{C}^{II}_{M/2}\,\mathbf{J}$, and modifying the butterfly,

$$\begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{bmatrix}, \quad (35)$$

we can obtain

$$\begin{aligned} \mathbf{E}(z) &= \tfrac{1}{2} \begin{bmatrix} \mathbf{D}\,\mathbf{C}^{II}_{M/2}\,\mathbf{J} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J}\,\mathbf{J} \end{bmatrix} \\ &\times \tfrac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \\ &\times \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}\,\mathbf{C}^{II^T}_{M/2}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \mathbf{J}_M \\ &= \tfrac{1}{\sqrt{2}}\,\mathbf{D}_M \begin{bmatrix} \mathbf{C}^{II}_{M/2} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^{IV}_{M/2}\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \\ &\times \tfrac{1}{2} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{J} & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \\ &\times \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}\,\mathbf{C}^{II^T}_{M/2}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \mathbf{J}_M \quad (36) \end{aligned}$$

With the following definitions in (4) and (5)

$$\begin{aligned} \mathbf{V} &\triangleq \mathbf{J}\,\mathbf{C}^{II^T}_{M/2}\,\mathbf{C}^{IV}_{M/2}\,\mathbf{J}, \\ \mathbf{P} &\triangleq \tfrac{1}{2} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{J} \\ \mathbf{J} & -\mathbf{I} \end{bmatrix}, \end{aligned}$$

it can be easily verified that

$$\mathbf{P}\,\mathbf{J}_M = \mathbf{P} \begin{bmatrix} \mathbf{0} & \mathbf{J} \\ \mathbf{J} & \mathbf{0} \end{bmatrix} = \mathbf{J}_M\,\mathbf{P}. \quad (37)$$

Indeed, this equation holds for all matrices $\mathbf{V}$, not only for the one defined in (5). Finally, with (1), (5), (4), and (37), the type-II LOT polyphase matrix in (36) can be rewritten as

$$\begin{aligned} \mathbf{E}(z) &= \mathbf{D}_M\,\mathbf{C}^{II}_M \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{J} & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{bmatrix} \mathbf{P}\,\mathbf{J}_M \\ &= \mathbf{D}_M\,\mathbf{C}^{II}_M \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{J} & \mathbf{0} \\ \mathbf{0} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{J} \\ \mathbf{J} & \mathbf{0} \end{bmatrix} \mathbf{P} \\ &= \mathbf{D}_M\,\mathbf{C}^{II}_M \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & z\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{P}. \end{aligned}$$

References

[1] J. Liang and T. D. Tran, "Fast multiplierless approximations of the DCT with the lifting scheme," *IEEE Trans. on Signal Processing*, vol. 49, pp. 3032–3044, Dec. 2001.

[2] W. Pennebaker and J. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York, 1993.

[3] V. Bhaskaran and K. Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, Kluwer, 1995.

[4] K. R. Rao and J. Hwang, *Techniques and Standards for Image, Video, and Audio Coding*, Prentice Hall, 1996.

[5] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice Hall, 1995.

[6] H. S. Malvar, *Signal Processing with Lapped Transforms*, Norwood, MA: Artech House, 1992.

[7] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, NY: Academic Press, 1990.

[8] W. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Com.*, vol. 25, pp. 1004–1009, Sept. 1977.

[9] Z. Wang, "Fast algorithm for the discrete W transform and for the discrete Fourier transform," *IEEE Trans. Accoustics, Speech, and Signal Processing*, vol. 32, pp. 803–816, Aug. 1984.

[10] R. L. de Queiroz and T. D. Tran, "Lapped transforms for image compression," in *The Handbook on Transforms and Data Compression*, pp. 197–265. CRC Press, Oct. 2000.

[11] R. L. de Queiroz, T. Q. Nguyen, and K. R. Rao, "The genlot: generalized linear-phase lapped orthogonal transform," *IEEE Trans. on Signal Processing*, vol. 40, pp. 497–507, Mar. 1996.

[12] T. D. Tran, R. L. de Queiroz, and T. Q. Nguyen, "Linear-phase perfect reconstruction filter bank: lattice structure, design, and application in image coding," *IEEE Trans. on Signal Processing*, vol. 48, pp. 133–147, Jan. 2000.

[13] C. A. Segall and A. K. Katsaggelos, "Pre- and post-processing algorithms for compressed video enhancement," *Proc. of the Asilomar Conference on Signals and Systems*, pp. 1369–1373, Pacific Grove, Oct. 2000.

[14] H. C. Reeve and J. S. Lim, "Reduction of blocking effects in image coding," *Opt. Eng.*, vol. 23, pp. 34–37, Jan. 1984.

[15] B. Ramamurthi and A. Gersho, "Nonlinear space-variant post-processing of block coded images," *IEEE Trans. ASSP*, vol. (34) 1258-1268, Oct. 1986.

[16] R. Rosenholtz and A. Zakhor, "Iterative procedures for reduction of blocking effects in transform image coding," *Trans. CSVT*, vol. 2, pp. 91–94, Mar. 1992.

[17] Y. Yang, N. P. Galatsanos, and A. K. Katsaggelos, "Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images," *Trans. on CSVT*, vol. 3, pp. 421–432, Dec. 1993.

[18] C. J. Kuo and R. J. Hsich, "Adaptive post-processor for block encoded images," *Trans. on CSVT*, vol. 5, pp. 322–336, Dec. 1995.

[19] Y. Yang and N. P. Galatsanos, "Removal of compression artifacts using projections onto convex sets and line process modeling," *Trans. on Image Processing*, vol. 6, pp. 1345–1357, Oct. 1997.

[20] L. J. Lin and A. Ortega, "Perceptually based video rate control using pre-filtering and predicted rate-distortion characteristics," *Proc. ICIP*, Santa Barbara, Oct. 1997.

[21] A. K. Katsaggelos and Eds. N. P. Galatsanos, *Signal Recovery Techniques for Image and Video Compression and Transmission*, Kluwer Academic, 1998.

[22] J. Chou, M. Crouse, and K. Ramchandran, "A simple algorithm for removing blocking artifacts in block-transform coded images," *IEEE Signal Processing Letters*, vol. 5, pp. 33–35, Feb. 1998.

[23] G. Cote, B. Erol, M. Gallant, and F. Kossentini, "H.263+: video coding at low bit rates," *IEEE Trans. CSVT*, vol. 8, pp. 849–866, Nov. 1998.

[24] H. W. Park and Y. L. Lee, "A post processing method for reducing quantization effects in low bit-rate moving picture coding," *IEEE Trans. on CSVT*, vol. 9, pp. 161–171, Feb. 1999.

[25] J. Apostolopoulos and N. Jayant, "Post-processing for very-low-bit-rate video compression," *IEEE Trans. on Image Processing*, vol. 8, pp. 1125–1129, Aug. 1999.

[26] "Special issue on MPEG-4," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 7, Feb. 1997.

[27] S. C. Chan, T. S. Ng, and C. K. Kwok, "A class of M-channel linear-phase biorthogonal filter banks and their applications to subband coding," *IEEE Trans. on Signal Processing*, vol. 47, pp. 564–571, Feb. 1999.

[28] M. Temerinac and B. Edler, "A unified approach to lapped orthogonal transforms," *IEEE Trans. Image Proc.*, vol. (1) 111-116, Jan. 1992.

[29] H. S. Malvar, "A pre- and post-filtering technique for the reduction of blocking effects," *Picture Coding Symposium*, Stockholm, Sweden, Jun. 1987.

[30] H. S. Malvar, "Method and system for adapting a digitized signal processing system for block processing with minimal blocking artifacts," *U.S. Patent No. 4,754,492*, Jun. 1988.

[31] H. S. Malvar, "Biorthogonal and nonuniform lapped transforms for transform coding with reduced blocking and ringing artifacts," *IEEE Trans. on Signal Processing*, vol. 46, pp. 1043–1053, Apr. 1998.

[32] J. D. Johnston, S. R. Quackenbush, J. Herre, and B. Grill, "Review of mpeg-4 general audio coding," in *Multimedia Systems, Standards, and Networks*, pp. 131–155. Marcel Dekker, 2000.

[33] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Prentice Hall, 1993.

[34] S. O. Aase and T. A. Ramstad, "On the optimality of nonunitary filter banks in subband coders," *IEEE Trans. on Image Processing*, vol. 4, pp. 1585–1591, Dec. 1995.

[35] H. S. Malvar and D. H. Staelin, "The LOT: transform coding without blocking effects," *IEEE Trans. on Signal Processing*, vol. 37, pp. 553–559, Apr. 1989.

[36] F. Bruekers and A. Enden, "New networks for perfect inversion and perfect reconstruction," *IEEE Journal on Selected Areas in Communications*, vol. 10, pp. 130–137, Jan. 1992.

[37] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting step," *Journal Fourier Anal. Appl.*, vol. 4, pp. 247–269, 1998.

[38] T. D. Tran, "The LiftLT: fast lapped transforms via lifting steps," *IEEE Signal Processing Letters*, vol. 7, pp. 145–149, Jun. 2000.

[39] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1999.

[40] C. Tu and T. D. Tran, "Context based entropy coding of block transform coefficients for image compression," *IEEE Trans. on Image Processing*, Nov. 2002.

[41] C. Tu and T. D. Tran, "On context based entropy coding of block transform coefficients," *ICIP*, pp. 669–672, Rochester, Sep. 2002.

[42] T. D. Tran and C. Tu, "Lapped transform based video coding," *Proc. SPIE Applications of Digital Image Processing XXIV*, pp. 319–333, San Diego, Aug. 2001.

[43] Y. Arai, T. Agui, and M. Nakajima, "A fast DCT-SQ scheme for images," *Trans. IEICE*, vol. E-71, pp. 1095, Nov. 1988.