ON LAPPED TRANSFORMS

by

RICARDO L. DE QUEIROZ

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 1999

DISCLAIMER

This copy was prepared to be printed on postscript printers by including figures in Encapsulated Postscript format. The figures and images (those not written in LaTeX, of course) included in the original dissertation were 600 dpi bitmaps (PBM files) requiring customized software for previewing in SVGA monitors and printing on 600 dpi HP Laser Jet series IV. These programs were developed by Eduardo Rubino and Henrique Malvar. As the inclusion format changed, the spacing around the figures may change slightly, what may change the overall pagination and such. Also, the figures in Appendix A suffered minor changes and the halftone techniques originally used are different than those used when the postcript interpreter renders the *gray-scale* images in your printer. A bit of graphical changes were made here and there, when halftones were mixed with LaTeX etc.

Therefore, this version may NOT accurately follow the same format of the dissertation submitted to the Graduate School of the University of Texas at Arlington, although having the same contents.

Many thanks to Cormac Herley for helping on the preparation of the postscript file.

ON LAPPED TRANSFORMS

The members of the Committee approve the doctoral

dissertation of Ricardo L. de Queiroz

K. R. Rao _____
Supervising Professor

O. R. Mitchell _____

M. Manry _____

V. Devarajan _____

M. Lowy _____

Dean of the Graduate School _____

To my wife, Ana Maria

ACKNOWLEDGEMENTS

I am thankful to Prof. Henrique S. Malvar for inspiring me to conduct scientific research. He patiently taught me most of what I know in multirate filter banks, lapped transforms, and image coding. The theme of this dissertation is a token of my appreciation for all he did for me. I am grateful to my advisor, Prof. K. R. Rao, for his guidance and support, which made my Ph.D. program possible. Dr. Truong Nguyen is co-author of the work on GenLOTs (Chapter 3) and our "long-distance" discussions on filter banks are gratefully acknowledged. I would like to thank Eduardo M. Rubino, from whom I learnt most of the little I know about computers. Rubino is responsible for the PC based DVI tools for the LaTeX package used in the preparation of this dissertation, as well as for several tools for image display, halftoning, etc. I also would like to thank Chris Turbeville for his hand in solving all computer "glitches" I had through my stay at UTA. A special note of gratitude goes to my mother, Amanda, and my father, Gilvan, who always supported and encouraged me through all these years dedicated to graduate studies and research. Finally, I am specially thankful to my wife, Ana Maria, for her love and patience through my Ph.D. program. This work was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, Brazil, under grant 200.804/90-1

July 22, 1994

ABSTRACT

ON LAPPED TRANSFORMS

Publication No._____

Ricardo L. de Queiroz, M.S.

The University of Texas at Arlington, 1994

Supervising Professor: K. R. Rao

In this dissertation, it is intended to demonstrate the potential benefits resulting from the study of lapped transforms (LTs). New results are presented, including the development of new LTs with general overlapping factors and linear phase filters, the theory of perfect reconstruction LTs for finite-length signals, the perfect reconstruction conditions for time-varying LT, and a theory to implement time-varying LTs and wavelet packets with the perfect reconstruction and orthogonality properties. Time-invariant LTs are presented, designed, optimized, and applied to image coding and the time-varying lapped transforms are applied to construct time-varying wavelet packets which are able to implement systems to decompose the signal in a maximally-decimated way using virtually any rectangular tiling of the time-frequency plane. In all cases, perfect reconstruction and orthogonality are inherently assured, along with fast implementation algorithms.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Since the early studies in the field of digital signal processing [66], [94], [18], researchers have been trying to better understand the relation among discrete signals sampled at different rates. The evolution of the field of multirate digital signal processing has made available several tools for direct sampling rate conversion in the discrete-time domain, including interpolation and decimation [94], [18]. Perhaps one of the most significant contributions in this field, is the better understanding of multirate filter banks, which are now the basis for several signal processing tasks [3], [6], [18], [34], [49], [39], [59], [70], [97], [102], [104], [106], [107], [110], [111]. The studying and understanding of multirate filter banks continues to evolve as new theories, design procedures, and applications appear in almost all technical journals in the field of digital signal processing. The discrete cosine transform (DCT), as well as any discrete block transform, [1], [21], [88], [89], and the discrete wavelet transform (DWT) [2], [13], [19], [42], are also well-known as special filter banks. Under this visualization, new horizons were created for these transforms, as their analysis has been simplified, unified, and better understood [24], [49], [46], [50], [51], [91], [100], [104], [108], [109]. Understanding is, in fact, the primary objective of this dissertation, aiming to broaden the knowledge in this field by developing new theories or revisiting and applying existing ones.

1.1 Filter banks

Filter banks are used to separate the input signal into several components, each one carrying a single frequency subband of the original signal [18], [104]. It is also desirable to design the filter bank such that these subbands can be recombined to recover the original signal. The first process is called analysis, while the second is called synthesis. The output of the analysis is referred as the subband signal, with as many subbands as there are filters in the filter bank. In multirate digital signal processing, the down-sampler and up-sampler operators play a fundamental role, reducing and increasing, respectively, the sampling frequency by an integer factor [18]. Fig. 1.1 shows the symbols for down-samplers and up-samplers and their input-output relation. If we assume ideal filters and that the bandpass signal in a subband has a bandwidth of, let us say, π/M rads, it can be downsampled by a factor $M:1$ without loss of information, making this the principle of critically-decimated filter banks [18], [104], [111]. In the synthesis side, the subband signal is upsampled by a factor of $1:M$, filtered to cancel imaging [18], and the subbands contributions are, thus, summed to recover the original signal. As a result, the sum of the number of samples in each subband is equal to the number of samples in the signal [18], [104]. This is also quite desirable in many applications. The combination of filtering followed by down-sampling is often called decimation, while the combination of up-sampling followed by filtering is called interpolation.

Ideal filters, inherently, are not feasible and the issue was first addressed using two-channel linear-phase filter banks and a design called quadrature mirror filter bank (QMF) was introduced to cancel aliasing resulting from the decimation and interpolation

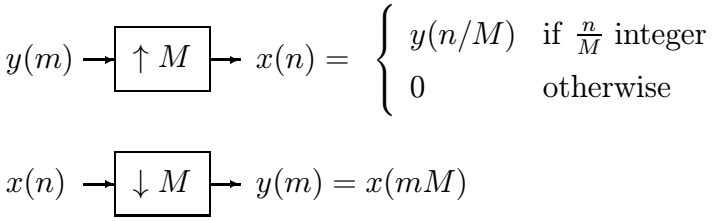


Figure 1.1: Basic operators for sampling rate conversion. The up-sampler (top) or sampling rate expander inserts $M-1$ zero samples between each pair of input samples, thus increasing sampling rate by a factor of M. The down-sampler (bottom) or sampling rate compressor retains only one out of M input samples.

processes [18], [22]. The so called Johnston's filters are a family of QMF designed for this approach [34]. The QMF solutions do not allow perfect reconstruction (PR) of the signal and later Smith and Barnwell [97] developed the conjugate quadrature filter bank (CQF) in a formulation which does not use linear-phase filters but allows PR of the signal. Both QMF and CQF solutions have a two-channel filter bank which can be hierarchically associated in a binary-tree path in order to create filter banks with more than two channels.

A uniform filter bank is the one where all, let us say, M filters have bandpass width of π/M, thus signals of all subbands are decimated and interpolated by a factor of M [18, 104]. Fig. 1.2 shows an M-channel critically decimated uniform filter bank. In this figure, M is the number of filters (or number of channels or subbands), $x(n)$ is the input signal, and $\hat{x}(n)$ is the recovered signal after synthesis. The subband signals are represented by $y_i(m)$ $(0 \leq i \leq M-1)$, and the filters with impulse responses $f_i(m)$ and $g_i(m)$ $(0 \leq i \leq M-1)$ correspond to analysis and synthesis sections, respectively.

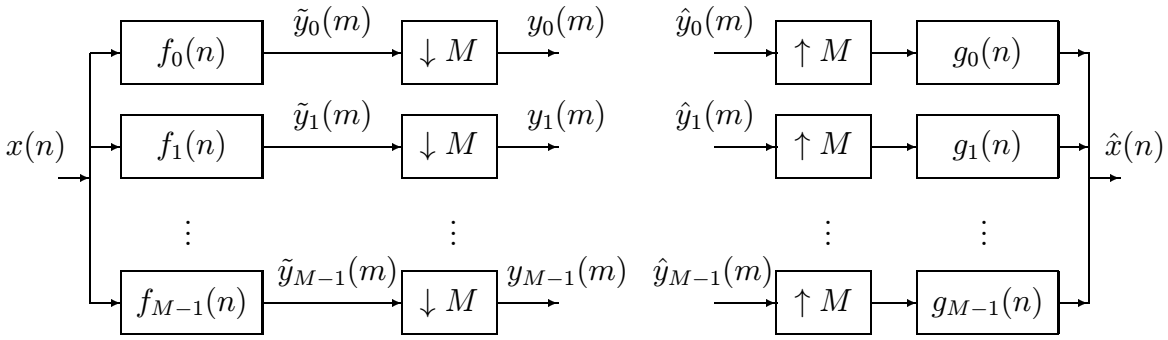


Figure 1.2: Critically decimated uniform filter bank. Analysis (left) and synthesis (right) sections are shown.

Filter banks can also be classified into paraunitary or bi-orthogonal [104]. In paraunitary FIR filter banks, each $f_i(m)$ has a one-to-one correspondence to $g_i(m)$ [49], [104], [110], while in bi-orthogonal filter banks the set $f_i(m)$ is found from the entire set of $g_i(m)$, or vice versa [104], [110]. This is similar to the relation between orthogonal and non-orthogonal matrices, and, in fact, orthogonal block transforms are a special case of paraunitary filter banks, while non-orthogonal ones belong to the class of bi-orthogonal filter banks[1].

In this dissertation we will concentrate on FIR uniform paraunitary filter banks, and show their equivalence to the lapped transform (LT) [49]. Although studied independently in the past, it was shown that both definitions are identical [49]. Therefore, the term lapped transform or LT will be used throughout this dissertation because of its compactness and elegance, instead of the more descriptive term FIR uniform paraunitary

[1]The term bi-orthogonal comes from the establishment of the PR conditions of two-channel filter banks which was extended later for the M-channel case [104], [109].

filter bank (PUFB). Furthermore, matrix notation and time-domain viewpoint, so useful for lapped transforms, will be applied here to the analysis and design of filter banks. So, the choice for the term lapped transform is quite justifiable, although we recall that all results and discussions in this dissertation apply to any filter bank in the class described above. The reader may also be cautioned not to confuse LT with the lapped orthogonal transform (LOT) [10], [43], [44], which is only a particular LT and not a generic definition [49].

1.2 Special notation

In terms of notation, our conventions are: $\mathbf{I}_n$ is the $n \times n$ identity matrix. $\mathbf{0}_n$ is the $n \times n$ null matrix, while $\mathbf{0}_{n \times m}$ stands for the $n \times m$ null matrix. $\mathbf{J}_n$ is the $n \times n$ counter-identity, or exchange, or reversing matrix, illustrated by the following example:

$$\mathbf{J}_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

$\mathbf{J}$ reverses the ordering of elements of a vector. $[\ ]^T$ means transposition. $[\ ]^H$ means transposition combined with conjugation, where this combination is usually called the Hermitian of the vector or matrix. $[\ ]^R$ means reversion of all columns and rows of a matrix. Thus, for an $m \times n$ matrix $\mathbf{A}$ we have $\mathbf{A}^R = \mathbf{J}_m \mathbf{A} \mathbf{J}_n$. Unidimensional concatenation of matrices and vectors is indicated by a comma. In general, capital bold face letters are reserved for matrices, so that $\mathbf{a}$ represents a vector while $\mathbf{A}$ represents a matrix. Unless

otherwise stated, only column vectors are used, and row-vectors are indicated using the transposition notation.

1.3 Organization

In chapter 2, we will cover the basic theory of LTs, exploring the relation among block and lapped transforms, filter banks, multi-input multi-output paraunitary discrete systems, hierarchical LTs and wavelets. We will also present some basic concepts of time-frequency representation of signals. In this chapter the basic concepts necessary for understanding the remaining parts of this dissertation are introduced with a particular perspective, allowing us to obtain several viewpoints for the same problem. Some useful time-invariant LTs will be discussed in chapter 3, introducing new LTs and revisiting existing ones. In this chapter, the implementation of these LTs over finite-length signals and signal extension concepts will be presented along with its underlying theory. Chapter 4 is concerned with time-varying LTs, presenting ways to fully vary the parameters of a PUFB maintaining orthogonality, the PR property, and fast algorithms (if any) at all times. Within this concept, the idea of a time-varying wavelet packet is developed allowing us to achieve virtually all rectangular partitions of the time-frequency (TF) plane in a maximally-decimated representation. Furthermore, algorithms for this purpose are presented. Chapter 5 discusses the implementation of LTs over finite-length signals. Signal extensions and special boundary LTs are designed and some image coding examples are presented. Finally, chapter 6 contains the conclusions of this dissertation.

CHAPTER 2

THEORY OF LAPPED TRANSFORMS

2.1 Overlap across block boundaries

2.1.1 Block transforms

In traditional block-transform processing, such as in image and audio coding, the signal is divided into blocks of M samples, and each block is processed independently [1], [14], [21], [23], [33], [49], [58], [68], [87], [88], [89]. Let the samples in the m-th block be denoted as

$$\mathbf{x}_m^T = [x_0(m), x_1(m), \ldots, x_{M-1}(m)], \tag{2.1}$$

and the corresponding transform vector be

$$\mathbf{y}_m^T = [y_0(m), y_1(m), \ldots, y_{M-1}(m)]. \tag{2.2}$$

For a real unitary transform $\mathbf{A}$, $\mathbf{A}^T = \mathbf{A}^{-1}$. The forward and inverse transforms for the m-th block are

$$\mathbf{y}_m = \mathbf{A}\mathbf{x}_m, \tag{2.3}$$

and

$$\mathbf{x}_m = \mathbf{A}^T\mathbf{y}_m. \tag{2.4}$$

The rows of $\mathbf{A}$, denoted $\mathbf{a}_n^T$ $(0 \leq n \leq M-1)$, are called the basis vectors because they form an orthogonal basis for the M-tuples over the real field [1]. The transform vector coefficients $[y_0(m), y_1(m), \ldots, y_{M-1}(m)]$ represent the corresponding weights of vector $\mathbf{x}_m$ with respect to this basis. Also, it is well known that the signal energy is preserved under an orthogonal transformation [1], [33], [23], [88], assuming stationary signals, i.e.,

$$M\sigma_x^2 = \sum_{i=0}^{M-1} \sigma_i^2, \tag{2.5}$$

where σ_i^2 is the variance of $y_i(m)$ and σ_x^2 is the variance of the input samples.

2.1.2 Lapped transforms

For lapped transforms [49], the basis vectors can have length L, such that $L > M$, extending across traditional block boundaries. Thus, the transform matrix is no longer square and most of the equations valid for block transforms do not apply to an LT. We will concentrate our efforts on *orthogonal* LTs[49] and consider $L = NM$, where N is the overlap factor. Note that N, M, and hence L are all integers. As in the case of block transforms, we define the transform matrix as containing the orthonormal basis vectors as its rows. A lapped transform matrix $\mathbf{P}$ of dimensions $M \times L$ can be divided into square $M \times M$ submatrices $\mathbf{P}_i$ $(i = 0, 1, \ldots, N-1)$ as

$$\mathbf{P} = [\mathbf{P}_0\ \mathbf{P}_1\ \cdots\ \mathbf{P}_{N-1}]. \tag{2.6}$$

The orthogonality property does not hold because $\mathbf{P}$ is no longer a square matrix and it

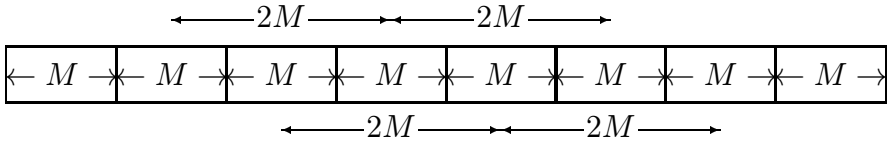


Figure 2.1: The signal samples are divided into blocks of M samples. The lapped transform uses neighboring blocks samples, as in this example for $N = 2$, i.e. $L = 2M$, yielding an overlap of $(L - M)/2 = M/2$ samples on either side of a block.

is replaced by the perfect reconstruction (PR) property[49], defined by

$$\sum_{i=0}^{N-1-l} \mathbf{P}_i \mathbf{P}_{i+l}^T = \sum_{i=0}^{N-1-l} \mathbf{P}_{i+l}^T \mathbf{P}_i = \delta(l)\mathbf{I}_M, \tag{2.7}$$

for $l = 0, 1, \ldots, N-1$, where $\delta(l)$ is the Kronecker delta, i.e., $\delta(0) = 1$ and $\delta(l) = 0$ for $l \neq 0$. As we will see later (2.7) states the PR conditions and orthogonality of the transform operating over the entire signal.

If we divide the signal into blocks, each of size M, we would have vectors $\mathbf{x}_m$ and $\mathbf{y}_m$ such as in (2.1) and (2.2). These blocks are not used by LTs in a straightforward manner. The actual vector which is transformed by the matrix $\mathbf{P}$ has to have L samples and, at block number m, it is composed of the samples of $\mathbf{x}_m$ plus $L-M$ samples. These samples are chosen by picking $(L-M)/2$ samples at each side of the block $\mathbf{x}_m$, as shown in Fig. 2.1, for $N = 2$. However, the number of transform coefficients at each step is M, and, in this respect, there is no change in the way we represent the transform-domain blocks $\mathbf{y}_m$.

The input vector of length L is denoted as $\mathbf{v}_m$, which is centered around the block $\mathbf{x}_m$, and is defined as

$$\mathbf{v}_m^T = \left[x\left(mM - (N-1)\frac{M}{2}\right) \cdots x\left(mM + (N+1)\frac{M}{2} - 1\right)\right]. \tag{2.8}$$

Then, we have

$$\mathbf{y}_m = \mathbf{P}\mathbf{v}_m. \tag{2.9}$$

The inverse transform is not direct as in the case of block transforms, i.e., with the knowledge of $\mathbf{y}_m$ we do not know the samples in the support region of $\mathbf{v}_m$, and neither in the support region of $\mathbf{x}_m$. We can reconstruct a vector $\hat{\mathbf{v}}_m$ from $\mathbf{y}_m$, as

$$\hat{\mathbf{v}}_m = \mathbf{P}^T\mathbf{y}_m. \tag{2.10}$$

where $\hat{\mathbf{v}}_m \neq \mathbf{v}_m$. To reconstruct the original sequence, it is necessary to accumulate the results of the vectors $\hat{\mathbf{v}}_m$, in a sense that a particular sample $x(n)$ will be reconstructed from the sum of the contributions it receives from all $\hat{\mathbf{v}}_m$, such that $x(n)$ was included in the region of support of the corresponding $\mathbf{v}_m$. This additional complication comes from the fact that $\mathbf{P}$ is not a square matrix [49]. However, the whole analysis-synthesis system (applied to the entire input vector) is orthogonal, assuring the PR property using (2.10).

We can also describe the process using a sliding rectangular window applied over the samples of $x(n)$. As an M-sample block $\mathbf{y}_m$ is computed using $\mathbf{v}_m$, $\mathbf{y}_{m+1}$ is computed from $\mathbf{v}_{m+1}$ which is obtained by shifting the window to the right by M samples, as shown in Fig. 2.2.

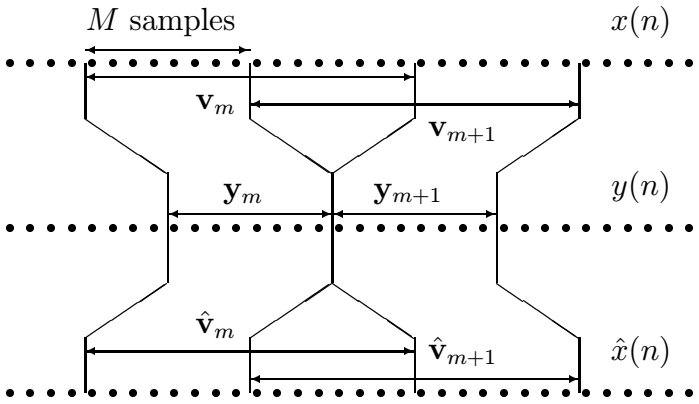


Figure 2.2: Illustration of a lapped transform with $N = 2$ applied to signal $x(n)$, yielding transform domain signal $y(n)$. The input L-tuple as vector $\mathbf{v}_m$ is obtained by a sliding window advancing M samples, generating $\mathbf{y}_m$. This sliding is also valid for the synthesis side.

As the reader may have noticed, the region of support of all vectors $\mathbf{v}_m$ is greater than the region of support of the input vector. Hence, a special treatment has to be given to the transform at the borders. We will discuss this fact later and assume infinite-length signals until then, or assume the length is very large and the borders of the signal are far enough from the region to which we are focusing our attention.

If we denote by $\mathbf{x}$ the input vector and by $\mathbf{y}$ the transform-domain vector, we can be consistent with our notation of transform matrices by defining a matrix $\mathbf{H}$ such that $\mathbf{y} = \mathbf{H}\mathbf{x}$ and $\hat{\mathbf{x}} = \mathbf{H}^T\mathbf{y}$. In this case, we have

$$\mathbf{H} = \begin{bmatrix} \ddots & & & & \\ & \mathbf{P} & & & \\ & & \mathbf{P} & & \\ & & & \mathbf{P} & \\ & & & & \ddots \end{bmatrix}. \tag{2.11}$$

where the displacement of the matrices $\mathbf{P}$ obeys the following

$$\mathbf{H} = \begin{bmatrix} \ddots & \ddots & & \ddots & & & \\ & \mathbf{P}_0 & \mathbf{P}_1 & \cdots & \mathbf{P}_{N-1} & & \\ & & \mathbf{P}_0 & \mathbf{P}_1 & \cdots & \mathbf{P}_{N-1} & \\ & & & \ddots & \ddots & & \ddots \end{bmatrix}. \tag{2.12}$$

$\mathbf{H}$ has as many block-rows as transform operations over each vector $\mathbf{v}_m$.

Let the rows of $\mathbf{P}$ be denoted by $1 \times L$ vectors $\mathbf{p}_i^T$ $(0 \leq i \leq M-1)$, so that $\mathbf{P}^T = [\mathbf{p}_0, \cdots, \mathbf{p}_{M-1}]$. In an analogy to the block transform case, we have

$$y_i(m) = \mathbf{p}_i^T \mathbf{v}_m. \tag{2.13}$$

The vectors $\mathbf{p}_i$ are the basis vectors of the lapped transform. They form an orthogonal basis for an M-dimensional subspace (there are only M vectors) of the L-tuples over the real field. As a remark, assuming infinite length signals, from the orthogonality of the basis vectors and from the PR property in (2.7), the energy is preserved, such that (2.5) is valid.

Assuming that the entire input and output signals are represented by the vectors $\mathbf{x}$ and $\mathbf{y}$, respectively, and that the signals have infinite length, then, from (2.11), we have

$$\mathbf{y} = \mathbf{H}\mathbf{x} \tag{2.14}$$

and, if $\mathbf{H}$ is orthogonal,

$$\mathbf{x} = \mathbf{H}^T\mathbf{y}. \tag{2.15}$$

Note that $\mathbf{H}$ is orthogonal if and only if (2.7) is satisfied. Thus, the meaning for (2.7) becomes clear, as it forces the transform operating over the entire input-output signals to be orthogonal. So, the LT is called orthogonal. For block transforms, as there is no overlap, it is sufficient to state the orthogonality of $\mathbf{A}$ because $\mathbf{H}$ will be a block-diagonal matrix.

These formulations for LTs are general, and if the transform satisfies the PR property described in (2.7), then the LTs are independent of the contents of the matrix $\mathbf{P}$. For example, suppose a block transform is chosen to transform the signal, but we want to use the direct algorithm for a $N = 4$ lapped transform. Thus, we can use

$$\mathbf{P} = [\mathbf{000A000}],$$

where $\mathbf{0}$ is a $M \times M/2$ null matrix. If we augment the length of the basis vectors, from the center to the borders, but maintaining PR, the formulae in this section would still be valid. The definition of $\mathbf{P}$ with a given N can accommodate any lapped transform whose length of the basis vectors lies between M and NM. For the case of block transforms,

$N = 1$, i.e. no overlap. This illustrates the fact that block transforms are a special case of lapped transforms.

Causal notation - If one is not concerned with particular localization of the transform in respect with the origin $x(0)$ of the signal $x(n)$, it is possible to change the notation to apply a causal representation. In this case, we can represent $\mathbf{v}_m$ as

$$\mathbf{v}_m^T = [\mathbf{x}_{m-N+1}^T, \cdots, \mathbf{x}_{m-1}^T, \mathbf{x}_m^T], \tag{2.16}$$

which is identical to the previous representation, except for a shift in the origin to maintain causality. The block $\mathbf{y}_m$ is found in a similar fashion as

$$\mathbf{y}_m = \mathbf{P}\mathbf{v}_m = \sum_{i=0}^{N-1} \mathbf{P}_{N-1-i}\mathbf{x}_{m-i}. \tag{2.17}$$

Similarly, $\hat{\mathbf{v}}_m$ can be reconstructed as in (2.10) where the support region for the vector is the same, except that the relation between it and the blocks $\hat{\mathbf{x}}_m$ will be changed accordingly.

2.2 Lapped transforms as filter banks

2.2.1 Matrix notation for paraunitary filter banks

Going back to the uniform maximally-decimated FIR paraunitary filter bank, as shown in Fig. 1.2, we remind the reader that the filters with impulse responses $f_i(n)$ and $g_i(n)$ $(0 \leq i \leq M-1)$ are the analysis and synthesis filters, respectively, and obey the

relation

$$f_i(n) = g_i(L-1-n),$$

for $0 \leq n \leq L-1$, i.e., they are pairwise mirror images of each other if the filter bank is paraunitary [49], [104], [110]. Let us focus on the analysis section where the subband signals $\mathbf{y}_k(m)$ are formed. For a PR design, the filters are such that, if $y_k(m) = \hat{y}_k(m)$, then $\hat{x}(n) = x(n-\tau)$, where τ is an integer representing a delay [104], [111]. In the analysis, right before the decimator, the convolution of the input signal and the k-th filter ($0 \leq k \leq M-1$) is represented as

$$\tilde{y}_k(n) = x(n) * f_k(n) = \sum_{i=-\infty}^{\infty} f_k(i)x(n-i) = \sum_{i=0}^{L-1} f_k(i)x(n-i). \tag{2.18}$$

The group of decimators is viewed as switches which close the circuit once at every M samples of the input signal. The absolute phase where the simultaneous decimation occurs, in this period of M samples, is unimportant for us and we will arbitrarily set it as happening at all instants $mM + M - 1$, i.e., the last polyphase component is passed to the output while all others are deleted. Hence, the subband signals are found by

$$y_k(m) = \tilde{y}_k(mM) = \sum_{i=0}^{L-1} f_k(i)x(mM+M-1-i). \tag{2.19}$$

Define the vectors

$$\mathbf{v}_m^T = [x(mM+M-L)\cdots x(mM+M-1)] = [\mathbf{x}_{m-N+1}^T, \cdots, \mathbf{x}_m^T] \tag{2.20}$$

$$\mathbf{g}_k^T = [g_k(0)\cdots g_k(L-1)] \tag{2.21}$$

$$\mathbf{f}_k^T = [f_k(0)\cdots f_k(L-1)], \tag{2.22}$$

for $0 \leq k \leq M-1$. Thus, (2.19) can be written as

$$y_k(m) = (\mathbf{f}_k^T \mathbf{J}_L)\mathbf{v}_m = \mathbf{g}_k^T \mathbf{v}_m \tag{2.23}$$

Comparing (2.23) and (2.13) we see that the analysis process of a filter bank can be regarded as a forward transformation using lapped transforms, if we use $\mathbf{p}_k = \mathbf{J}_L \mathbf{f}_k$, for $0 \leq k \leq M-1$. If we apply the same derivations for the synthesis section, we will obtain $\mathbf{p}_k = \mathbf{g}_k$, where $\mathbf{g}_k$ is a vector with the synthesis filters coefficients. Combining both results, and assuming $\mathbf{P}$ has elements $\{p_{ij}\}$, we can interchange the filter bank and lapped transforms interpretations as long as

$$p_{kl} = f_k(L-1-l) = g_k(l) \tag{2.24}$$

is assumed, for $0 \leq k \leq M-1$ and $0 \leq l \leq L-1$ [49]. It is even possible to state the equivalence between LTs and paraunitary filter banks (uniform, using FIR filters and maintaining the validity of the dimensions of filters and matrices). In other words, a LT is a paraunitary filter bank and vice-versa, although both concepts were developed independently in the past. In order to convert a filter bank to a lapped transform, then

- Choose an arbitrary maximum-length-limit L (an integer multiple of M) for all filters.
- Pad zeros on the impulse response of the filters, until the length L is obtained.
- Construct matrix $\mathbf{P}$ using (2.24) and follow the analysis and synthesis procedures described in Sec.2.1.2.

This result is important for two main reasons. First it gives a new insight over the understanding of the frequency-domain behavior of the LT. As each basis vector can be regarded as a filter, one can evaluate its frequency response. Second, it allows the analysis and implementation of a filter bank in time-domain as LT. Therefore, we can use either point-of-view, depending on which approach would be more applicable for a particular purpose.

2.2.2 Variances of transform coefficients

We will now derive the variance for the coefficient of any orthogonal lapped or block transform. If $x(n)$ is a stationary process with power spectral density (PSD) given by $S_{xx}(e^{j\omega})$, then the PSD of the signal right before the decimator is

$$S_{\tilde{y}_k\tilde{y}_k}(e^{j\omega}) = S_{xx}(e^{j\omega})|F_k(e^{j\omega})|^2,$$

where $F_k(e^{j\omega})$ is the Fourier transform of $f_k(n)$. The PSD of the decimated signal is

$$S_{y_k y_k}(e^{j\omega}) = \sum_{r=0}^{M-1} S_{\tilde{y}_k\tilde{y}_k}\left(e^{j(\omega-2\pi r)/M}\right) \tag{2.25}$$

and the integral of the PSD, before and after decimation, is the same. Therefore, in order to calculate the variance of any transformed coefficient, we can use

$$\sigma_k^2 = \frac{1}{\pi}\int_0^{\pi} S_{xx}(e^{j\omega})\ |F_k(e^{j\omega})|^2\ d\,\omega. \tag{2.26}$$

2.3 Paraunitary filter banks

2.3.1 Multi-input multi-output FIR systems

A stable and causal linear discrete system governed by a difference equation relating the input and output sequences can be defined by the Z-transform of its impulse response, i.e., its transfer function. Assume the system is FIR. For an FIR multi-input multi-output system (MIMO), we may express each output sequence as a linear combination of difference equations involving each of the input sequences. The details of this section can be found in [104]. Let the MIMO system have M input and M output sequences with respective Z-transforms $X_i(z)$ and $Y_i(z)$, for $0 \leq i \leq M-1$. Then, $X_i(z)$ and $Y_i(z)$ are related by

$$\begin{bmatrix} Y_0(z) \\ Y_1(z) \\ \vdots \\ Y_{M-1}(z) \end{bmatrix} = \begin{bmatrix} E_{0,0}(z) & E_{0,1}(z) & \cdots & E_{0,M-1}(z) \\ E_{1,0}(z) & E_{1,1}(z) & \cdots & E_{1,M-1}(z) \\ \vdots & \vdots & \ddots & \vdots \\ E_{M-1,0}(z) & E_{M-1,1}(z) & \cdots & E_{M-1,M-1}(z) \end{bmatrix} \begin{bmatrix} X_0(z) \\ X_1(z) \\ \vdots \\ X_{M-1}(z) \end{bmatrix} \tag{2.27}$$

where $E_{ij}(z)$ are entries of $\mathbf{E}(z)$ which is a square matrix, called the transfer matrix of the system.

Of relevance to us are the normalized paraunitary transfer matrices with entries on the field of real-coefficient polynomials of z, i.e. the entries represent real-coefficients FIR filters. For such system, $\mathbf{E}(z)$ becomes a unitary matrix when evaluated on the unit circle with center in the origin of the z-plane, as

$$\mathbf{E}^H(e^{j\omega})\mathbf{E}(e^{j\omega}) = \mathbf{E}(e^{j\omega})\mathbf{E}^H(e^{j\omega}) = \mathbf{I}_M, \tag{2.28}$$

and $\mathbf{E}(z)$ is normalized paraunitary, which means

$$\mathbf{E}^{-1}(z) = \mathbf{E}^T(z^{-1}). \tag{2.29}$$

For FIR causal entries, paraunitary systems are said lossless systems, which contain several properties analogous to those of orthogonal matrices. In fact, an orthogonal matrix is one where all $E_{ij}(z)$ are constant for all z.

The degree of $\mathbf{E}(z)$ (or the McMillan degree, N_z) is a positive number indicating the minimum number of delays necessary to implement the system. For paraunitary systems, the determinant of $\mathbf{E}(z)$ is of the form az^{-N_z}, for a real constant a.

The order of $\mathbf{E}(z)$ is the maximum degree among all $E_{ij}(z)$, assuming they represent causal FIR filters.

2.3.2 Polyphase transfer matrix

In multirate signal processing, it is often more comfortable to work with the polyphase components of the signal [18]. If a decimation-interpolation factor of M is applied, it is useful to consider M polyphase components $x_i(m)$ of the signal $x(n)$ as

$$x_i(m) = x(mM + i) \tag{2.30}$$

where $0 \leq i \leq M-1$. The polyphase components are, of course, sub-sampled by a factor of M in relation to the original signal. Recalling Fig. 1.2, $y_i(n)$ are the subband signals

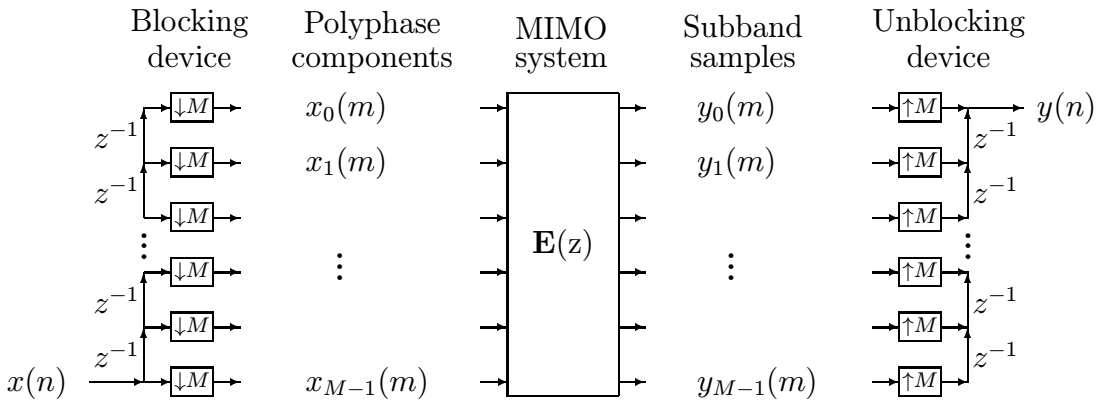


Figure 2.3: The analysis filter bank regarded as block-filter implementation, where the output signal $y(n)$ has its polyphase components as the subband signals.

(for $0 \leq i \leq M-1$). Now define a signal $y(n)$ whose polyphase components are the subband signals, i.e.,

$$y_i(m) = y(mM + i) \tag{2.31}$$

Therefore, the analysis filter bank is regarded as a MIMO system relating the polyphase components of $x(n)$ and $y(n)$ in a block-filter-type operation [104], as shown in Fig. 2.3.

In Fig. 2.3, the devices responsible to convert a serial stream into its polyphase components and vice-versa, denoted as blocking and unblocking devices, respectively, are shown. The polyphase signals and the subbands are processed with a rate M times slower than the sampling rate of $x(n)$. The blocking device can also be viewed as a device to extract one block of M samples from $x(n)$ at a time. Also, in Fig. 2.3, the MIMO system $\mathbf{E}(z)$ is called the polyphase transfer matrix (PTM) and it is responsible for processing the polyphase components, in order to find the subbands [104], [106], [111]. The PTM is found by commuting the sequence of decimators-interpolators and filters

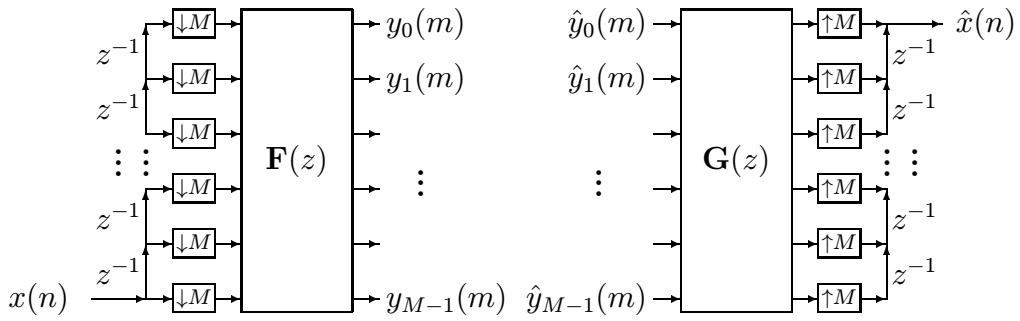


Figure 2.4: The filter bank represented as a MIMO system is applied to the polyphase components of the signal. The matrices $\mathbf{F}(z)$ and $\mathbf{G}(z)$ are called polyphase transfer matrices. For a PR system both must be inverses of each other and for paraunitary filter banks they must be paraunitary matrices. Thus, $\mathbf{G}(z) = \mathbf{F}^{-1}(z) = \mathbf{F}^T(z^{-1})$. For a PR paraunitary causal system of order N, we must choose $\mathbf{G}(z) = z^{-(N-1)}\mathbf{F}^T(z^{-1})$.

using the well-known commuting rules [18][104]. The PTM $\mathbf{E}(z)$ has elements in the field of polynomials of z, which are given directly from the filter bank. The elements of $\mathbf{E}(z)$, $E_{ij}(z)$, are the Z-transforms of the sequences $e_{ij}(m)$ given by

$$e_{ij}(m) = f_i(mM + M - 1 - j). \tag{2.32}$$

The synthesis filter bank also corresponds to another transfer matrix leading the processed subband signal $\hat{y}(m)$, whose polyphase components are $\hat{y}_i(m)$, into the reconstructed signal $\hat{x}(n)$. For that distinction, we denote $\mathbf{F}(z)$ as the PTM corresponding to the analysis filter bank, and $\mathbf{G}(z)$ as the PTM corresponding to the synthesis filter bank. A blocking device cascaded with a unblocking device results in a pure delay which can be omitted. Therefore, the entire analysis-synthesis system can be represented as in Fig. 2.4 through the use of PTMs.

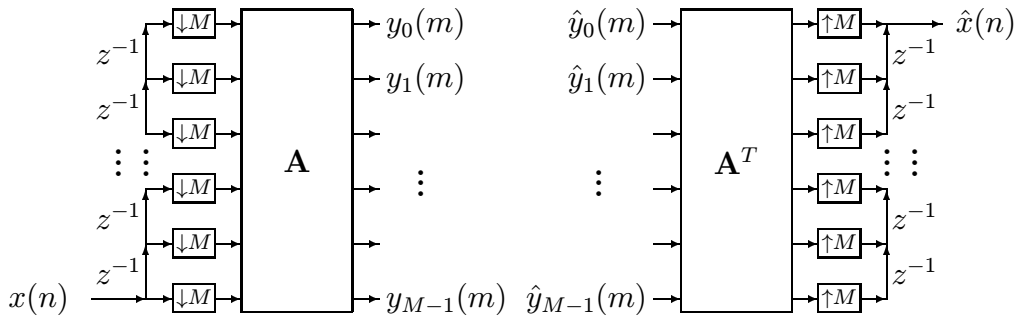


Figure 2.5: Block-transform operation as a memoryless processing of the polyphase components of the signal.

For a real-valued $\mathbf{E}(z)$, for all z, we have $\mathbf{E}(z) = \mathbf{A}$, and $\mathbf{A}$ is an orthogonal transform, since $\mathbf{E}(z)$ is assumed to be paraunitary. So, the block transform discussed in Sec. 2.1.1 can be viewed as a memoryless operation applied to the M polyphase components of the signal, generating M subband signals, as in Fig. 2.5.

2.3.3 Perfect reconstruction and orthogonality

In the absence of any processing, $\mathbf{F}(z)$ and $\mathbf{G}(z)$ are connected together back-to-back and PR is possible if they are inverses of each other.

$$\mathbf{F}(z) = \mathbf{G}^{-1}(z) \tag{2.33}$$

In this case, $\hat{x}(n) = x(n)$ and analysis and synthesis are canceled. As mentioned earlier, we are interested in paraunitary (lossless) systems to represent the filter bank [104], [106], [107]. Let $\mathbf{E}(z)$ represent a normalized paraunitary transfer matrix so that $\mathbf{E}^{-1}(z) = \mathbf{E}^T(z^{-1})$, then, adding the constraint of normalized paraunitariness, we have

$$\mathbf{F}(z) = \mathbf{E}(z), \qquad \mathbf{G}(z) = \mathbf{E}^T(z^{-1}). \tag{2.34}$$

If the entries of $\mathbf{F}(z)$ are causal filters, the inverse PTM has non-causal filters and adding the constraint of causality, for a PTM of order N, we must choose

$$\mathbf{F}(z) = \mathbf{E}(z), \qquad \mathbf{G}(z) = z^{-(N-1)}\mathbf{E}^T(z^{-1}) \tag{2.35}$$

so that $\hat{x}(n) = x(n-NM+1)$ and the overall reconstruction delay is $\tau = L-1$. Therefore, the PUFB, or the LT, can be described by a single $M \times M$ paraunitary PTM $\mathbf{E}(z)$.

Reinforcing the equivalence of paraunitary systems and LTs, we recall that the LT matrix $\mathbf{P}$ and the PTMs are expressed as a function of the analysis and synthesis filters. Hence

$$\mathbf{F}(z) = \sum_{i=0}^{N-1} z^{-i}\mathbf{P}_{N-1-i}\mathbf{J}_M \tag{2.36}$$

$$\mathbf{G}(z) = \sum_{i=0}^{N-1} z^{-i}\mathbf{P}_i. \tag{2.37}$$

As a result, the reader can verify that each of the following equations are equivalent and each one of them implies the other [49]

$$\mathbf{E}(z)\mathbf{E}^T(z^{-1}) = \mathbf{E}^T(z^{-1})\mathbf{E}(z) = \mathbf{I}_M \tag{2.38}$$

$$\sum_{i=0}^{N-1-l} \mathbf{P}_i\mathbf{P}_{i+l}^T = \sum_{i=0}^{N-1-l} \mathbf{P}_i^T\mathbf{P}_{i+l} = \delta(l)\mathbf{I}_M. \tag{2.39}$$

$$\mathbf{H}\mathbf{H}^T = \mathbf{H}^T\mathbf{H} = \mathbf{I}_\infty \tag{2.40}$$

This reminds us once more about the equivalence of real-valued maximally-decimated uniform FIR PUFBs and LTs and this comparison will no longer be carried, exclusively referring to such a system as an LT.

2.3.4 Support region and delays

The only constraints, in the previous definition of an LT, are that the signal is assumed of infinite length and $N-1$ is the maximum degree of the entries of the PTM, such that each filter can have an actual length lying anywhere between M and NM. If one of the filters has a length much smaller than $L = NM$, then it can be delayed.

Proposition 2.1 *Let $\mathbf{E}'(z)$ and $\mathbf{E}''(z)$ be two PTM corresponding to analysis filters $f_k(n)$ and $f_k(n-p_kM-q_k)$, respectively, for p_k and q_k unique integers. Let $f_k(n)$ have length ℓ_k and let $\ell_k + p_kM + q_k \leq L$. If $\mathbf{E}'(z)$ is paraunitary, $\mathbf{E}''(z)$ remains paraunitary, provided that p_k is any permissible integer and $q_0 = \ldots = q_{M-1} = q$.*

Let $E'_{ij}(z)$ and $E''_{ij}(z)$ be the entries of $\mathbf{E}'(z)$ and $\mathbf{E}''(z)$, respectively, and let $((x))_M$ denote x *modulo* M or the remainder of x/M. One can verify that

$$E''_{ij}(z) = z^{-p_i} E'_{i,((j+q_k))_M}(z) z^{((M-1-j+q_k))_M}. \tag{2.41}$$

If ${\mathbf{e}'_k}^T(z)$ ${\mathbf{e}''_k}^T(z)$ are the vectors containing the k-th rows of $\mathbf{E}'(z)$ and $\mathbf{E}''(z)$, respectively, then (2.41) can be also expressed as

$$\mathbf{e}''_k(z) = z^{-p_k} \mathbf{\Phi}_k(z) \mathbf{e}'_k(z) \tag{2.42}$$

for a suitable matrix $\mathbf{\Phi}_k(z)$ which will combine a permutation matrix and a diagonal matrix with elements of the form z^{-i}. Note that $\mathbf{\Phi}_k(z)$ is paraunitary. Paraunitariness

of $\mathbf{E}''(z)$ requires that

$$\mathbf{e}_r''^{\,T}(z)\mathbf{e}_s''(z^{-1}) = \delta(r-s). \tag{2.43}$$

Since $\mathbf{\Phi}_k(z)$ is only dependent on q_k, if $q_0 = \ldots = q_{M-1} = q$ then $\mathbf{\Phi}_r^T(z)\mathbf{\Phi}_s(z^{-1}) = \mathbf{I}_M$, $(r, s \in 0, \ldots, M-1)$. Thus,

$$\mathbf{e}_r''^{\,T}(z)\mathbf{e}_s''(z^{-1}) = z^{-p_r+p_s}\mathbf{e}_r'^{\,T}(z)\mathbf{\Phi}_r^T(z)\mathbf{\Phi}_s(z^{-1})\mathbf{e}_s'(z^{-1}) = \delta(r-s) \tag{2.44}$$

Therefore, $\mathbf{E}''(z)$ is paraunitary. From this proposition we learnt that, if not all the filters have length L in a LT, we can shift the impulse responses of the filters and pad zeros to both extremities, in order to make the non-zero entries of the LT matrix $\mathbf{P}$, clustered around its center. With these considerations, we assume all filters to have actual length L, regardless of shifting or zero-padding their impulse responses.

2.4 Factorization of lapped transforms

There is an important result for paraunitary uniform filter banks, derived from the paraunitariness of the PTM, which states that any $\mathbf{E}(z)$ can be decomposed into a series of orthogonal matrices and delay stages [20], [105]. There are N_z delay stages and $N_z + 1$ orthogonal matrices, where N_z is the MacMillan degree of $\mathbf{E}(z)$ (the degree of the determinant of $\mathbf{E}(z)$). Then,

$$\mathbf{E}(z) = \mathbf{B}_0 \prod_{i=1}^{N_z} (\mathbf{\Upsilon}(z)\mathbf{B}_i) \tag{2.45}$$

where $\mathbf{\Upsilon}(z) = diag\{z^{-1}, 1, 1, \ldots, 1\}$, and $\mathbf{B}_i$ are orthogonal matrices. It is well-known that an $M \times M$ orthogonal matrix can be expressed as a product of $M(M-1)/2$ plane

rotations. However, in this case, only $\mathbf{B}_0$ is a general orthogonal matrix, while the matrices $\mathbf{B}_1$ through $\mathbf{B}_{N_z}$ have only $M-1$ degrees of freedom. This is an important result which will be used throughout this dissertation. It states that it is possible to implement a lapped transform with a sequence of delays and orthogonal matrices. It also defines the total number of degrees of freedom in a lapped transform, i.e., if one changes arbitrarily any of the plane rotations composing the orthogonal transforms, one will span all possible PR lapped transforms, for given values of M and L.

Proposition 2.2 *The (McMillan) degree of* $\mathbf{E}(z)$ *is bounded by* $N_z \leq (L-M)/2$ *with equality for a general structure to implement all LTs of filters with length up to* $L = NM$*, i.e.,* $\mathbf{E}(z)$ *of order* $N-1$.

This factorization is minimal and complete, however, it does not say anything about the length of the filters. In [102] it is presented a minimal factorization of a class of PUFBs where $\mathbf{E}(z)$ has order $N-1$ and $N_z = (N-1)M/2 = (L-M)/2$. Then, a general factorization has to have $N_z \geq (L-M)/2$. Consider $\mathbf{E}''(z) = \mathbf{E}'(z)\mathbf{\Upsilon}(z)\mathbf{B}_i$, as one increment in the general factorization, where $\mathbf{B}_i$ is a sequence of $M-1$ Givens rotations [20]. Assume $\mathbf{B}_i$ has at least one rotation that can not be incorporated into $\mathbf{E}'(z)$, otherwise $\mathbf{B}_i = \mathbf{I}_M$. Hence, if $\mathbf{E}'(z)$ and $\mathbf{E}''(z)$ correspond to a PUFB whose filters have lengths L' and L'', respectively, then it is clear that $L'' \geq L' + 2$. As the maximum length is L, there can only be a maximum of $(L-M)/2$ delay stages and $N_z \leq (L-M)/2$. Thus, for the general case $N_z = (L-M)/2$.

Unless otherwise stated, we will consider the LTs which can be parameterized using the symmetric delay factorization (SDF). Let

$$\mathbf{\Lambda}(z) = \begin{bmatrix} z^{-1}\mathbf{I}_{M/2} & 0 \\ 0 & \mathbf{I}_{M/2} \end{bmatrix}, \qquad \tilde{\mathbf{\Lambda}}(z) = \begin{bmatrix} \mathbf{I}_{M/2} & 0 \\ 0 & z^{-1}\mathbf{I}_{M/2} \end{bmatrix}. \tag{2.46}$$

The SDF of the PTM is given by

$$\mathbf{F}(z) = \mathbf{B}_0 \prod_{i=1}^{N-1} \left(\mathbf{\Lambda}(z)\mathbf{B}_i\right) \tag{2.47}$$

$$\mathbf{G}(z) = \mathbf{B}_{N-1}^T \prod_{i=N-2}^{0} \left(\tilde{\mathbf{\Lambda}}(z)\mathbf{B}_i^T\right). \tag{2.48}$$

where all stages $\mathbf{B}_i$ are allowed to be arbitrary $M \times M$ orthogonal matrices. Of course, in such case, M is forced to be even. Note that for SDF, $N_z = (L-M)/2$. It is not clear whether, SDF is complete, i.e., can generate any LT for M even if we delay the filters appropriately. However, we will assume it as a particular case.

The flow graph for implementing an LT which can be parameterized using SDF is shown in Fig. 2.6 for analysis and synthesis sections. The use of SDF is not very restrictive in practice, as, for M even, most PUFBs with any practical advantage can be expressed in this way as we will see later. Its advantage will become clear in a later chapter.

If we are given the SDF matrices instead of the filters' coefficients, one can easily reconstruct the LT matrix. For this, start with the last stage and recur the structure in (2.47). Let $\mathbf{P}^{(i)}$ be the partial reconstruction of $\mathbf{P}$ after including up to the i-th stage. Then,

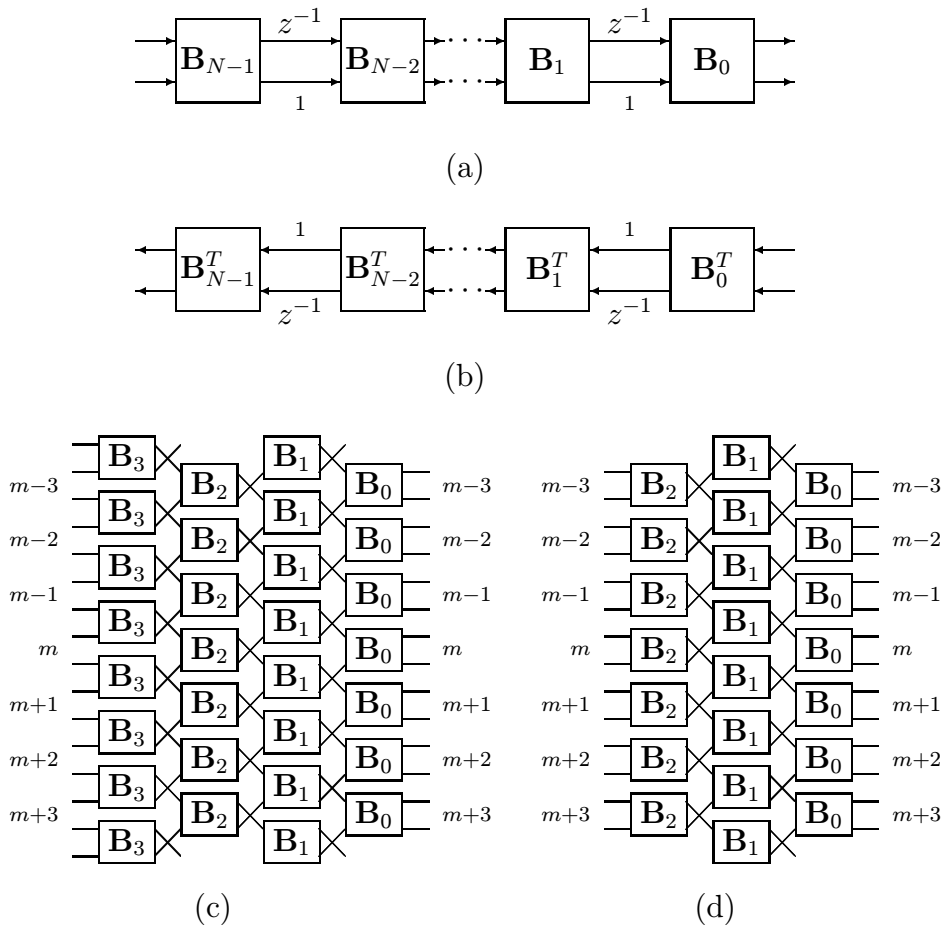


Figure 2.6: Flow graph for paraunitary FIR filter banks where $\mathbf{F}(z)$ can be factorized using symmetric delays and N stages. Signals $x(n)$ and $y(n)$ are segmented and processed using blocks of M samples, all branches carry $M/2$ samples, and blocks $\mathbf{B}_i$ are $M \times M$ orthogonal matrices. (a) Analysis section; (b) Synthesis section; (c) and (d) are equivalent SDF flow-graphs for the non-causal implementation of $\mathbf{H}$. As all blocks are orthogonal, analysis is carried by following the paths from left to right while synthesis is carried by following the paths from right to left, using the transposes of the $\mathbf{B}_i$. Time- and transform-domain block numbers are indicated in (c) for $N = 4$ and in (d) for $N = 3$.

$$\mathbf{P}^{(0)} = \mathbf{B}_{N-1} \tag{2.49}$$

$$\mathbf{P}^{(i)} = \mathbf{B}_{N-1-i} \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \end{bmatrix} \begin{bmatrix} \mathbf{P}^{(i-1)} & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{P}^{(i-1)} \end{bmatrix} \tag{2.50}$$

$$\mathbf{P} = \mathbf{P}^{(N-1)}. \tag{2.51}$$

Redefinition of an LT - We will assume, from now on, that an LT is a maximally-decimated uniform FIR PUFB, obeying the SDF. Note that this implies that M is even.

2.5 Finite-length signals

Suppose the input signal $x(n)$ has only N_x samples and assume $N_x = N_B M$, where N_B is an integer representing the number of blocks, with M samples per block. To avoid the expansion of the number of samples, we require $y(n)$ to have N_x samples, so that each subband would have N_B samples. Again, let $x(n)$ and $y(n)$ be represented by the vectors $\mathbf{x}$ and $\mathbf{y}$, respectively, while, $\hat{x}(n)$ and $\hat{y}(n)$ are the corresponding signals in the synthesis section represented by vectors $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$, respectively. With a straight forward application of the LT, the analysis section will require more than N_x samples in $\mathbf{x}$ in order to find N_x samples of $\mathbf{y}$. We can define $\tilde{\mathbf{x}}$ as an augmented vector obtained from $\mathbf{x}$ by extending the boundary samples in any fashion, in a process we call signal extension. The matrix notation for the analysis is given by

$$\mathbf{y} = \tilde{\mathbf{P}}\tilde{\mathbf{x}} \tag{2.52}$$

where $\tilde{\mathbf{P}}$ is similar to $\mathbf{H}$ in (2.11), but with only N_B block-rows. The synthesis is accomplished by

$$\hat{\tilde{\mathbf{x}}} = \tilde{\mathbf{P}}^T \hat{\mathbf{y}}. \tag{2.53}$$

As an example, for $N = 3$ and $N_B = 5$, we have

$$\tilde{\mathbf{P}} = \begin{bmatrix} \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M & \mathbf{0}_M & \mathbf{0}_M & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M & \mathbf{0}_M & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{0}_M & \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{0}_M & \mathbf{0}_M & \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 \end{bmatrix}$$

From (2.7) we can see that

$$\tilde{\mathbf{P}}^T \tilde{\mathbf{P}} = \begin{bmatrix} \boldsymbol{\Gamma}_L & & \\ & \mathbf{I}_{N_x - L + M} & \\ & & \boldsymbol{\Gamma}_R \end{bmatrix}, \tag{2.54}$$

where $\boldsymbol{\Gamma}_L$ and $\boldsymbol{\Gamma}_R$ are $(L-M) \times (L-M)$ matrices and none of them is equal to $\mathbf{I}_{L-M}$, so that even if $\hat{\mathbf{y}} = \mathbf{y}$ we have $\hat{\mathbf{x}} \neq \mathbf{x}$, where the difference would occur in the last $(L-M)/2$ samples in each border. However, there is a size-limited linear transform $\mathbf{T}$ mapping $\mathbf{x}$ into $\mathbf{y}$ so that [64], [65]

$$\mathbf{y} = \mathbf{T}\mathbf{x} \tag{2.55}$$

$$\mathbf{x} = \mathbf{T}^{-1}\mathbf{y}. \tag{2.56}$$

Note that $\tilde{\mathbf{P}}$ is not a square matrix whereas $\mathbf{T}$ is. As $N_B \to \infty$, then $\mathbf{T} \to \tilde{\mathbf{P}}$, $\tilde{\mathbf{P}} \to \mathbf{H}$, and $\mathbf{T} \to \mathbf{H}$. Also, both $\mathbf{H}$ and $\tilde{\mathbf{P}}$ are a function of just $\mathbf{P}$ whereas $\mathbf{T}$ depends on other parameters such as the number of samples in the signal. $\mathbf{T}$ is an undefined

matrix and depends on the boundary filter banks. These facts lead to the following statement.

Proposition 2.3 *Although, for infinite-length signals, an LT will lead to a orthogonal transform* $\mathbf{T}$*, in the case of finite-length signals, neither PR nor orthogonality is assured.*

However, there are techniques that allow us to always reconstruct the signal boundaries. Furthermore, it is also possible to adapt the analysis-synthesis system to achieve orthogonality of $\mathbf{T}$. These topics are covered in a later chapter.

2.6 Hierarchical connection of lapped transforms

So far we have focused on the construction of a single LT resulting in M subband signals. What happens if we cascade LTs by connecting them hierarchically, in such a way that a subband signal is the actual input for another LT ? Also, what are the consequences of submitting only part of the subband signals to further stages of LTs ? The relation between PUFBs and discrete orthogonal wavelets [2], [3], [13], [19], [42], [24], [49], [46], [50], [51], [91], [96], [100], [104], [108], [109] is well-known. Under conditions that are easily satisfied [104], [24], an infinite cascade of PUFBs will generate a set of continuous orthogonal wavelet bases. In general, if only the low-pass subband is connected to another PUFB, for a finite number of stages, we call the resulting filter bank a discrete wavelet transform (DWT) [3], [13], [19], [42], [91] [104], [108], [109]. A free cascading of LTs, however, is better known as discrete wavelet packet (DWP) [16], [17], [71], [100].

From our equivalence relations between PUFBs and LTs, we infer the same relation

between LTs and wavelets. The filter bank resulting from the hierarchical association of several LTs will be called a hierarchical lapped transform (HLT) [46].

2.6.1 Time-frequency diagram

The description of the cascaded connection of LTs is better carried with the aid of simplifying diagrams. The first is the time-frequency (TF) diagram. It is based on the TF plane, which is well known from the fields of spectral and time-frequency analysis [67], [7], [41]. The time-frequency representation of signals is a well-known method (for example the time-dependent DFT and the construction of spectrograms [67]; see [15] and [7] for details on TF signal representation). The TF representation is obtained by expressing the signal $x(n)$ with respect to bases which are functions of both frequency and time. For example, the size-r DFT of a sequence extracted from $x(n)$ (from $x(n)$ to $x(n+r-1)$) [67] can be

$$\alpha(k,n) = \sum_{i=0}^{r-1} x(i+n) \exp\left(-\frac{j2\pi ki}{r}\right) \tag{2.57}$$

Using a sliding window $w(m)$ of length r which is non-zero only in the interval $n \leq m \leq n+r-1$, (which in this case is rectangular), we can rewrite the last equation as

$$\alpha(k,n) = \sum_{i=-\infty}^{\infty} x(i)w(i) \exp\left(-\frac{jk(i-n)2\pi}{r}\right). \tag{2.58}$$

For more general bases we may write

$$\alpha(k,n) = \sum_{i=-\infty}^{\infty} x(i)\phi(n-i,k) \tag{2.59}$$

where $\phi(n,k)$ represents the bases for the space of the signal, n represents the index where the base is located in time, and k is the frequency index.

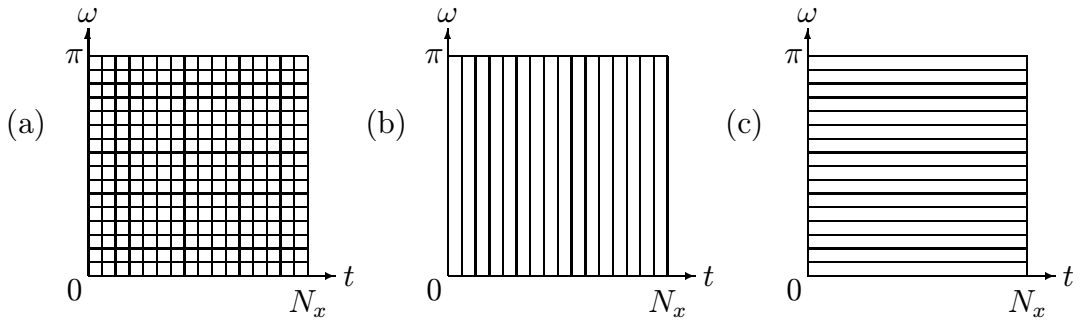


Figure 2.7: Examples of rectangular partitions of the time-frequency plane for a signal which has N_x samples. (a) Spectrogram with a N_x-length window, resulting in N_x^2 TF samples; (b) Input signal, no processing; (c) A transform such as the DCT or DFT is applied to all N_x samples;

As the signal is assumed to have an infinite number of samples, consider a segment of N_x samples extracted from signal $x(n)$, which can be extended in any fashion in order to account for the overlap of the window of r samples outside the signal domain. In such segment we can construct a spectrogram with a resolution of r samples in the frequency axis and N_x samples in the time axis. Assuming a maximum frequency resolution we can have a window with length up to $r = N_x$. In this case, the diagram for the spectrogram is given in Fig. 2.7(a). We call such diagrams as TF diagrams, because they only indicate the number of samples used in the TF representation of the signal. Assuming an ideal partition of the TF plane (using filters with ideal frequency response and null transition regions), each TF coefficient would represent a distinct region in a TF diagram. Note that in such representation, the signal is represented by N_x^2 TF coefficients. We are looking for maximally-decimated TF representation which is defined as a representation of the signal where the TF plane diagram would be partitioned into N_x regions, i.e., N_x

TF coefficients will be generated. Also, we require that all N_x samples of $x(n)$ can be reconstructed from the N_x TF coefficients. If we use less than N_x samples in the TF plane, we clearly cannot reconstruct all possible combinations of samples in $x(n)$, from the TF coefficients, solely using linear relations.

Under these assumptions, Fig. 2.7(b) shows the TF diagram for the original signal (only resolution in the time axis) for $N_x = 16$. Also, for $N_x = 16$, Fig. 2.7(c) shows a TF diagram with maximum frequency resolution, which could be achieved by transforming the original N_x-sample sequence with an N_x-sample DCT or DFT.

2.6.2 Tree-structured hierarchical lapped transforms

The tree diagram is helpful to describe the hierarchical connection of filter banks. In this diagram we represent an M-band LT by nodes and branches of an M-ary tree. In Fig. 2.8(a) it is shown an M-band LT, where all the M subband signals have sampling rates M times smaller than that of $x(n)$. In Fig. 2.8(b) it is shown the equivalent notation for the LT in a tree diagram, i.e., a single-stage M-branch tree, which is called here a tree cell. Recalling Fig. 2.7, the equivalent TF diagram for an M-band LT is shown in Fig. 2.8(c), for a 16-sample signal and for $M = 4$. Note that the TF diagram of Fig. 2.8(c) resembles that of Fig. 2.7(a). This is because for each 4 samples in $x(n)$ there is a corresponding set of 4 transformed coefficients. So, the TF representation is maximally decimated. Compared to Fig. 2.7(b), Fig. 2.8(c) implies an exchange of resolution from time to frequency domain achieved by the LT.

The exchange of resolution in the TF diagram is obtained by the LT. As we connect

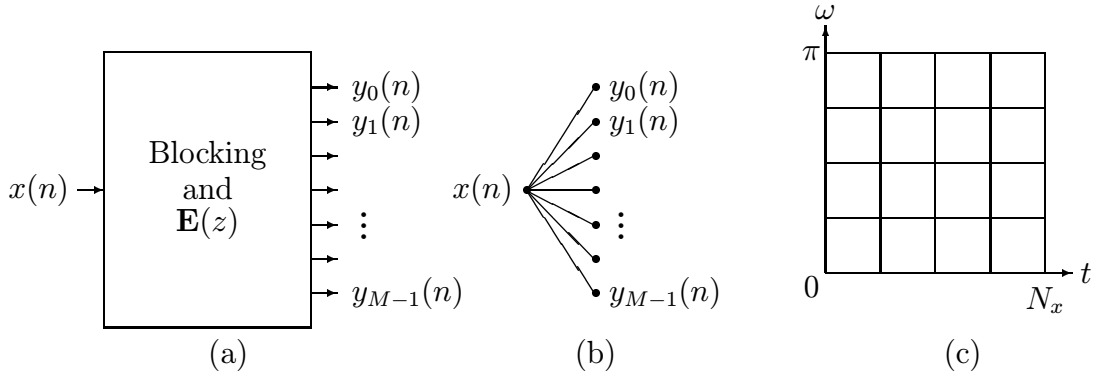


Figure 2.8: Representation of an M-channel LT as tree nodes and branches. (a) Analysis section of an LT, including the blocking device. (b) Equivalent notation for (a) using an M-branch single-stage tree. (c) Equivalent TF diagram for (a) or (b) assuming $M = 4$ and $N_x = 16$.

several LTs following the paths of a tree, each new set of branches (each new tree cell) connected to the tree will force the TF diagram to exchange from time to frequency resolution. We can achieve a more versatile TF representation by connecting cells in unbalanced ways. For example, in Fig. 2.9 it is shown some examples of HLTs given by their tree diagrams and respective TF diagrams. In Fig. 2.9(a) it is shown the tree diagram for the 3-stages DWT. Note that only the lowpass subband is further processed. Also, as all stages are chosen to be 2-channel LTs, this HLT can be represented by a binary tree. In Fig. 2.9(b), a more generic hierarchical connection of 2-channel LTs is shown. First the signal is split into low- and high-pass. Each output branch is further connected to another 2-channel LT. In the third stage only the most low-pass subband signal is connected to another 2-channel LT. In Fig. 2.9(c) it is shown a 2-stages HLT obtaining the same TF diagram as Fig. 2.9(b). Note that the succession of 2-channel LTs was substituted by a single stage 4-channel LT, i.e., the signal is split into four subbands

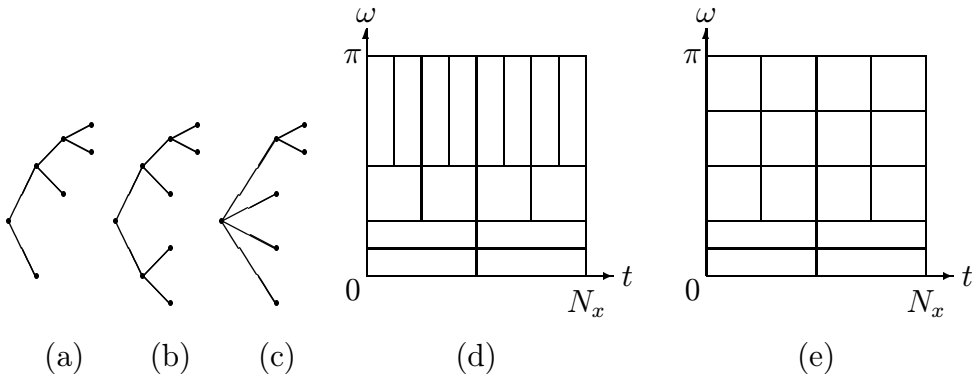


Figure 2.9: Tree and TF diagrams. (a) The 3-stages DWT binary-tree diagram, where only the low-pass subband is submitted to further LT stages. (b) A more generic 3-stages tree diagram. (c) A 2-stages tree-diagram resulting in the same TF diagram as (b). (d) TF diagram for (a). (e) TF diagram for (b) or (c).

and, then, one subband is connected to another LT. Fig. 2.9(d) shows the TF diagram corresponding to Fig. 2.9(a), while Fig. 2.9(e) shows the TF diagram corresponding to Fig. 2.9(b,c). Note that, as the tree-paths are unbalanced, we have irregular partitions of the TF plane. For example, in the DWT, low-frequency TF coefficients have poor time localization and good frequency resolution, while high-frequency ones have poor frequency resolution and better time localization.

To better understand how connecting an LT to the tree can achieve the exchange between time and frequency resolutions, in Fig. 2.10 it is shown the bases functions (filters) resulting from two similar tree-structured HLTs. The difference between them is one tree cell which is applied or not to a terminal branch of the tree.

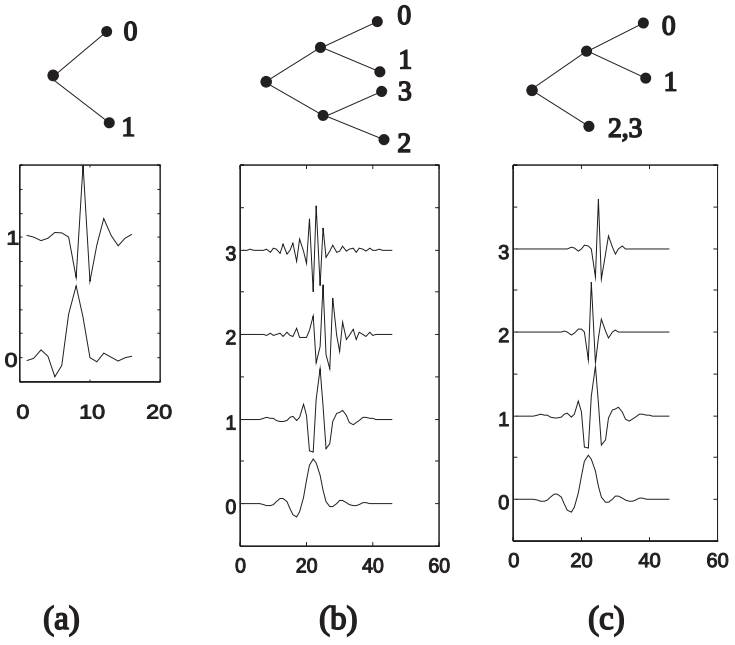


Figure 2.10: Two HLTs and resulting filter banks. (a) The 2-channel 16-tap-filters LT, showing low- and high-pass filters, $f_0(n)$ and $f_1(n)$, respectively. (b) Resulting basis functions of a 2-stage HLT based on (a), given by $f_0(n)$ through $f_3(n)$. Its respective tree diagram is also shown. (c) Resulting HLT, by pruning one high-pass branch in (b). Note that the two high-pass basis functions are identical to the high-pass basis function of (a) and, instead of having two distinct bases for high frequencies, occupying distinct spectral slots, the two bases are, now shifted in time. Thus, better time localization is attainable, at the expense of frequency resolution.

CHAPTER 3

TIME-INVARIANT LAPPED TRANSFORMS

As we will show, some classes of fast and efficient LTs can be generated with SDF, not to mention the two-channel PUFBs which always obey a SDF structure [105]. In order to be useful for subband signal processing, LTs are designed to have fast implementation algorithms allied with good performance. Such performance is measured for a given criterion. The algorithms are frequently based on variations of the DCT. The DCT matrix is also known as the DCT type II and is denoted as $\mathbf{D}^{II}$. The inverse DCT is known as the DCT type III, and denoted as $\mathbf{D}^{III}$. Clearly, $(\mathbf{D}^{II})^{-1} = (\mathbf{D}^{II})^T = \mathbf{D}^{III}$. Another matrix used here is the so-called DCT type IV matrix, which is both orthogonal and symmetric, so that $(\mathbf{D}^{IV})^{-1} = (\mathbf{D}^{IV})^T = \mathbf{D}^{IV}$. Let the DCTs of type II and IV have entries d_{ij}^{II} and d_{ij}^{IV}, respectively, for i and j pertaining to $\{0, 1, \ldots, M-1\}$. Then

$$d_{ij}^{II} = \sqrt{\frac{2}{M}} k_i \cos\left(\frac{(2j+1)i\pi}{2M}\right), \quad d_{ij}^{IV} = \sqrt{\frac{2}{M}} \cos\left(\frac{(2j+1)(2i+1)\pi}{4M}\right), \tag{3.1}$$

where $k_0 = 1$ and $k_i = 1/\sqrt{2}$, for $1 \leq i \leq M-1$. These matrices are highly structured and have fast implementation algorithms [88]. Two main classes of LTs will be discussed along with some design issues.

3.1 Extended lapped transform (ELT)

Cosine modulated filter banks are PUFBs [104] using a low-pass prototype modulating a cosine sequence. By a proper choice of the phase of the cosine sequence, Malvar developed the modulated lapped transform (MLT) [45], which led to the so-called extended lapped transforms (ELT) [47], [48], [49], [52]. The ELT allows several overlapping factors, generating a family of PR cosine modulated filter banks. Both designations (MLT and ELT) are frequently applied to this class of filter banks. Other cosine-modulation approaches have also been developed and the most significant difference among them is the low-pass prototype choice and the phase of the cosine sequence [26], [38], [45], [49], [48], [61], [70], [95], [103], [104].

In the ELTs, the filters' length L is basically an even multiple of the block size M, as $L = NM = 2KM$. Thus, K is referred to as the overlap factor of the ELT. The MLT-ELT class is defined by

$$p_{k,n} = h(n) \cos\left[\left(k + \frac{1}{2}\right)\left(\left(n - \frac{L-1}{2}\right)\frac{\pi}{M} + (N+1)\frac{\pi}{2}\right)\right] \tag{3.2}$$

for $k = 0, 1 \ldots, M-1$ and $n = 0, 1, \ldots, L-1$. $h(n)$ is a symmetric window modulating the cosine sequence and the impulse response of a low-pass prototype (with cutoff frequency at $\pi/2M$) which is translated in frequency to M different frequency slots in order to construct the uniform filter bank. A very useful ELT is the one with $K = 2$, which will be designated as ELT-2, while ELT with other values of K will be referred as ELT-K.

The ELTs have as their major plus a fast implementation algorithm. The algorithm is based on a factorization of the PTM into a series of plane rotation stages and delays and

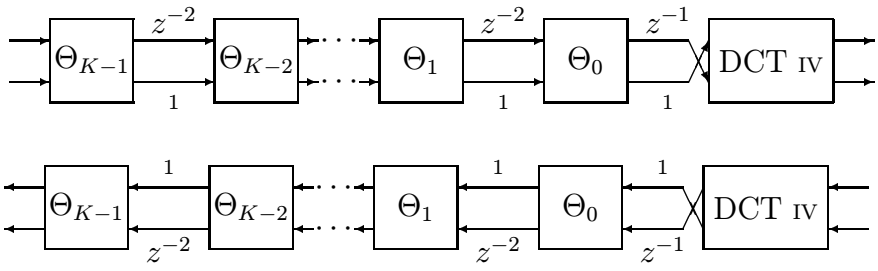


Figure 3.1: Flow graph for the direct (top) and inverse (bottom) ELT. Each branch carries $M/2$ samples.

a DCT type IV [88] orthogonal transform in the last stage, which has fast implementation algorithms. The lattice-style algorithm is shown in Fig. 3.1 for an ELT with generic overlap factor K. In Fig. 3.1 each branch carries $M/2$ samples and both analysis (forward transform) and synthesis (inverse transform) flow-graphs are shown. The plane rotation stages are of the form indicated in Fig. 3.2 and contain $M/2$ orthogonal butterflies to implement the $M/2$ plane rotations. The stages Θ_i contain the plane rotations and are defined by

$$\Theta_i = \begin{bmatrix} -\mathbf{C}_i & \mathbf{S}_i\mathbf{J}_{M/2} \\ \mathbf{J}_{M/2}\mathbf{S}_i & \mathbf{J}_{M/2}\mathbf{C}_i\mathbf{J}_{M/2} \end{bmatrix}, \tag{3.3}$$

$$\mathbf{C}_i = diag\left\{\cos(\theta_{0,i}),\ \cos(\theta_{1,i}), \cdots, \cos(\theta_{\frac{M}{2}-1,i})\right\}$$

$$\mathbf{S}_i = diag\left\{\sin(\theta_{0,i}),\ \sin(\theta_{1,i}), \cdots, \sin(\theta_{\frac{M}{2}-1,i})\right\}$$

$\theta_{i,j}$ are rotation angles. These angles are the free parameters in the design of an ELT because they define the modulating window $h(n)$. Note that there are KM angles,

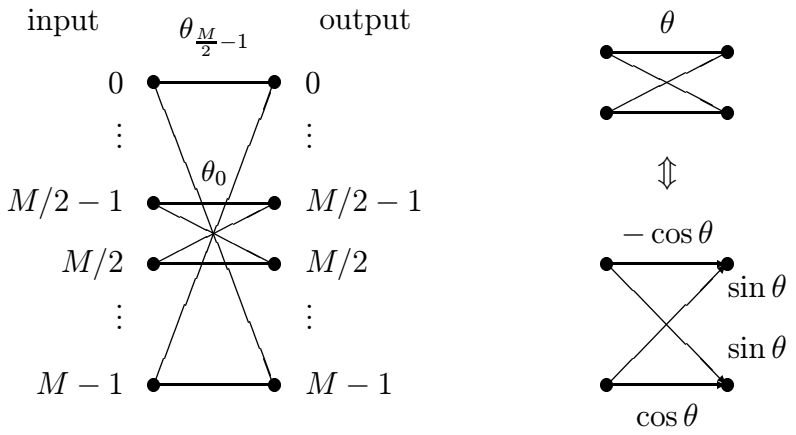


Figure 3.2: Implementation of plane rotations stage showing the displacement of the $M/2$ butterflies.

while $h(n)$ has $2KM$ samples, however, $h(n)$ is symmetric what gives a total number of degrees of freedom in such cosine modulated filter banks equal to KM. In general, there is no simple relation among the rotation angles and the window. We will often use the optimized angles given in [49], except for the ELT-2, where we use the parameterized design [52][49][48]. In this design, we have

$$\theta_{k,0} = -\frac{\pi}{2} + \mu_{M/2+k} \tag{3.4}$$

$$\theta_{k,1} = -\frac{\pi}{2} + \mu_{M/2-1-k} \tag{3.5}$$

where

$$\mu_i = \left[\left(\frac{1-\gamma}{2M}\right)(2k+1) + \gamma\right] \tag{3.6}$$

and γ is a control parameter, for $0 \leq k \leq (M/2) - 1$. In general, we will use $\gamma = 0.5$.

If we let $\mathbf{D}^{IV}$ denote the DCT type IV matrix [88], from our SDF notation, we can define the ELT as the LT where

$$\begin{aligned}
\mathbf{B}_0 &= \mathbf{D}^{IV} \begin{bmatrix} \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \end{bmatrix}, \\
\mathbf{B}_{2i+1} &= \Theta i \ \text{ for } 0 \leq i \leq K-1, \\
\mathbf{B}_{2i} &= \mathbf{I}_M \ \text{ for } 1 \leq i \leq K-1.
\end{aligned}$$

3.2 Linear-phase LT

From the work by Soman et al. [101], [102], we know that a complete parameterization of any $\mathbf{E}(z)$ of order $N-1$ characterizing a linear-phase PUFB (LPPUFB) of M-channels (M even) is given by

$$\mathbf{E}(z) = \mathbf{S}_{lp}\mathbf{Q}_{lp}\mathbf{\Psi}_{N-1}\mathbf{\Lambda}(z)\mathbf{\Psi}_{N-2}\mathbf{\Lambda}(z)\cdots\mathbf{\Lambda}(z)\mathbf{\Psi}_0\mathbf{Q}_{lp} \tag{3.7}$$

where

$$\mathbf{Q}_{lp} = \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{J}_{M/2} \end{bmatrix} \tag{3.8}$$

$$\mathbf{S}_{lp} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{S}' & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{S}'' \end{bmatrix} \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{J}_{M/2} \\ \mathbf{I}_{M/2} & -\mathbf{J}_{M/2} \end{bmatrix}, \tag{3.9}$$

$\mathbf{S}'$ and $\mathbf{S}''$ can be any $M/2 \times M/2$ orthogonal matrices, and $\mathbf{\Psi}_i$ are $M \times M$ orthogonal matrices described as

$$\mathbf{\Psi}_i = \begin{bmatrix} \mathbf{A}_{lp,i} & \mathbf{B}_{lp,i} \\ \mathbf{B}_{lp,i} & \mathbf{A}_{lp,i} \end{bmatrix}. \tag{3.10}$$

for some non-singular $M/2 \times M/2$ matrices $\mathbf{A}_{lp,i}$ and $\mathbf{B}_{lp,i}$. $\mathbf{\Lambda}(z)$ is defined in (2.46). We will abbreviate the notation for (3.7) as

$$\mathbf{E}(z) = \mathbf{S}_{lp}\mathbf{Q}_{lp}\mathbf{\Psi}_{N-1}\left(\prod_{i=N-2}^{0}\mathbf{\Lambda}(z)\mathbf{\Psi}_i\right)\mathbf{Q}_{lp}. \tag{3.11}$$

Let

$$\mathbf{W} = \frac{1}{\sqrt{2}}\begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & -\mathbf{I}_{M/2} \end{bmatrix}$$

and

$$\mathbf{\Phi}_i = \begin{bmatrix} \mathbf{U}_i & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{V}_i \end{bmatrix}, \tag{3.12}$$

where $\mathbf{U}_i$ and $\mathbf{V}_i$ can be any $M/2 \times M/2$ orthogonal matrices. The implementation flow-graph of the LPPUFB is shown in Fig. 3.3. Note that $\mathbf{\Psi}_i$ can be expressed as [102, 101]

$$\mathbf{\Psi}_i = \mathbf{W}\mathbf{\Phi}_i\mathbf{W} \tag{3.13}$$

for $\mathbf{A}_{lp,i} = (\mathbf{U}_i + \mathbf{V}_i)/2$ and $\mathbf{B}_{lp,i} = (\mathbf{U}_i - \mathbf{V}_i)/2$.

Proposition 3.1 *Any linear-phase PUFB with M even can be expressed as*

$$\mathbf{E}(z) = \mathbf{K}_{N-1}(z)\mathbf{K}_{N-2}(z)\cdots\mathbf{K}_1(z)\mathbf{E}_0$$

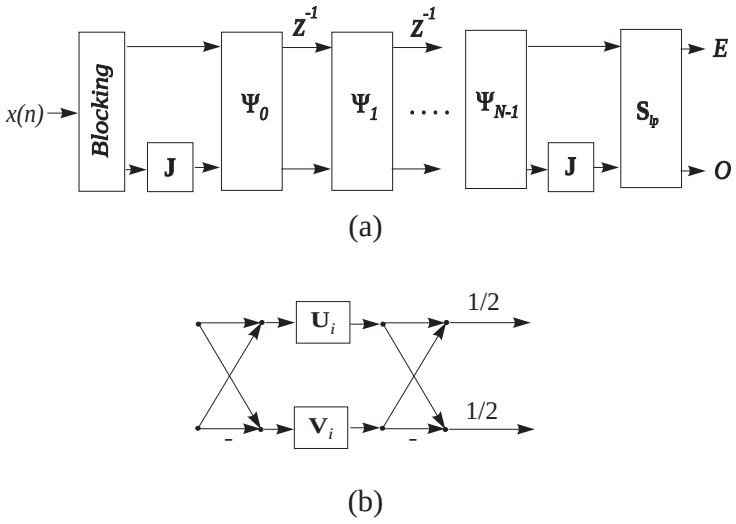


Figure 3.3: (a) Flow-graph for the implementation of the PTM $\mathbf{E}(z)$ describing the analysis section of the LPPUFB. Each branch carries $M/2$ samples and E and O stand for even and odd output subband coefficients. In this factorization, the stages $\mathbf{\Psi}_i$ can be factorized as in part (b).

---

where $\mathbf{K}_i(z) = \Phi_i \mathbf{W} \mathbf{\Lambda}(z) \mathbf{W}$*, and* $\mathbf{E}_0$ *is a generic* $M \times M$ *orthogonal matrix with symmetric basis functions.*

It is easy to see that $\mathbf{S}_{lp}\mathbf{Q}_{lp}\mathbf{\Psi}_{N-1}$ can be simplified to

$$\mathbf{S}_{lp}\mathbf{Q}_{lp}\mathbf{\Psi}_{N-1} = \begin{bmatrix} \mathbf{S}'\mathbf{U}_{N-1} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{S}''\mathbf{V}_{N-1} \end{bmatrix} \mathbf{W}. \tag{3.14}$$

As $\mathbf{U}_{N-1}$ and $\mathbf{S}'$ are generic orthogonal matrices, and the product $\mathbf{S}'\mathbf{U}_{N-1}$ is also a generic orthogonal matrix, we can discard the term $\mathbf{S}'$ without any loss of generality. The same is valid for $\mathbf{S}''$ with regard to $\mathbf{V}_{N-1}$. Therefore, we get $\mathbf{S}_{lp}\mathbf{Q}_{lp}\mathbf{\Psi}_{N-1} = \mathbf{\Phi}_{N-1}\mathbf{W}$ and (3.11) reduces to

$$\mathbf{E}(z) = \mathbf{\Phi}_{N-1}\mathbf{W}\left(\prod_{i=N-2}^{0}\mathbf{\Lambda}(z)\mathbf{W}\mathbf{\Phi}_i\mathbf{W}\right)\mathbf{Q}_{lp}, \tag{3.15}$$

or to

$$\mathbf{E}(z) = \left(\prod_{i=N-1}^{1}\mathbf{\Phi}_i\mathbf{W}\mathbf{\Lambda}(z)\mathbf{W}\right)\mathbf{E}_0. \tag{3.16}$$

where $\mathbf{E}_0 = \mathbf{\Phi}_0\mathbf{W}\mathbf{Q}_{lp}$ is a general $M \times M$ orthogonal matrix with symmetric basis functions, i.e., the PTM of order 0 of a LPPUFB. Since an order-n PTM leads to filters of length $(n+1)M$, a LPPUFB with filter's length $nM + M$ can be obtained from one with filters' length nM by adding a stage to the PTM of the later. If $\mathbf{E}_n(z)$ denotes an order-n PTM, then we can state that

$$\mathbf{E}_n(z) = \mathbf{K}_n(z)\mathbf{E}_{n-1}(z) \tag{3.17}$$

where

$$\mathbf{K}_i(z) = \mathbf{\Phi}_i\mathbf{W}\Lambda(z)\mathbf{W}. \tag{3.18}$$

Therefore, for any $N > 1$, any PTM of a LPPUFB can be expressed as

$$\mathbf{E}(z) = \mathbf{K}_{N-1}(z)\mathbf{K}_{N-2}(z)\cdots\mathbf{K}_1(z)\mathbf{E}_0 \tag{3.19}$$

Proposition 3.2 *Each stage in the above factorization can also be expressed as* $\mathbf{K}_i(z) = \mathbf{\Phi}_i\mathbf{W}_1\mathbf{\Lambda}(z)\mathbf{W}_2$, *where* $\mathbf{W}_1$ *and* $\mathbf{W}_2$ *can be either* $\mathbf{W}$ *or* $\mathbf{W}^R$, *independently.*

Let

$$\tilde{\mathbf{I}} = \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & -\mathbf{I}_{M/2} \end{bmatrix}$$

then $\mathbf{\Psi}_i$ can also be expressed as

$$\mathbf{\Psi}_i = \mathbf{W}^R \mathbf{\Phi} \mathbf{W}^R \; ; \quad \mathbf{A}_{lp,i} = \frac{\mathbf{U}_i + \mathbf{V}_i}{2} \; ; \; \mathbf{B}_{lp,i} = \frac{\mathbf{V}_i - \mathbf{U}_i}{2}, \tag{3.20}$$

$$\tilde{\mathbf{I}}\mathbf{\Psi}_i = \mathbf{W}^R \mathbf{\Phi} \mathbf{W} \; ; \quad \mathbf{A}_{lp,i} = \frac{\mathbf{V}_i - \mathbf{U}_i}{2} \; ; \; \mathbf{B}_{lp,i} = -\frac{\mathbf{U}_i + \mathbf{V}_i}{2}, \tag{3.21}$$

$$\tilde{\mathbf{I}}\mathbf{\Psi}_i = \mathbf{W} \mathbf{\Phi} \mathbf{W}^R \; ; \quad \mathbf{A}_{lp,i} = \frac{\mathbf{V}_i - \mathbf{U}_i}{2} \; ; \; \mathbf{B}_{lp,i} = \frac{\mathbf{U}_i + \mathbf{V}_i}{2}. \tag{3.22}$$

Hence we can say that

$$\mathbf{E}_n(z) = \left(\prod_{i=n}^{1} \mathbf{\Phi}_{N-1} \mathbf{W}_i \mathbf{\Lambda}(z) \mathbf{W}_{i-1} \right) \mathbf{\Phi}_0 \mathbf{W}_0 \mathbf{Q}_{lp} \tag{3.23}$$

where $\mathbf{W}_i$ can be either $\mathbf{W}$ or $\mathbf{W}^R$, such that $\mathbf{\Psi}_i$ is as in (3.13) or (3.20). Suppose we violate this rule, for example by reversing only one $\mathbf{W}$ matrix, as in (3.21) or (3.22), then the reader can check that we will obtain a PTM $\mathbf{E}'_n(z)$ which is related to the original one by $\mathbf{E}'_n(z) = \pm\tilde{\mathbf{I}}\mathbf{E}_n(z)$. Therefore, $\mathbf{E}'_n(z)$ also corresponds to a LPPUFB, although having the sign of some filters inverted. Odd-symmetric filters are not affected, because the sign change is equivalent to time-reversion of the coefficients. For even-symmetric filters, the sign change can be compensated by inverting the signs of the elements of any matrix Φ_i, because the odd-symmetric filters are not significantly affected by the overall sign change. As a conclusion, the stage $\mathbf{K}_i(z)$ can be expressed as

$$\mathbf{K}_i(z) = \mathbf{\Phi}_i \mathbf{W}_1 \mathbf{\Lambda}(z) \mathbf{W}_2 \tag{3.24}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ can be either $\mathbf{W}$ or $\mathbf{W}^R$, independently.

Let $\mathbf{P}^{(i)}$ be the partial reconstruction of $\mathbf{P}$ after including up to the i-th stage. Then, for linear-phase LTs, we have

$$\mathbf{P}^{(0)} = \mathbf{E}_0 \tag{3.25}$$

$$\mathbf{P}^{(i)} = \mathbf{\Phi}_i \mathbf{W} \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \end{bmatrix} \begin{bmatrix} \mathbf{W}\mathbf{P}^{(i-1)} & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{W}\mathbf{P}^{(i-1)} \end{bmatrix} \tag{3.26}$$

$$\mathbf{P} = \mathbf{P}^{(N-1)}. \tag{3.27}$$

3.3 Generalized Linear-Phase LOT (GenLOT)

The LOT is a popular LT with $N = 2$ whose basis functions are symmetric [10], [43], [44], [45], [49]. Its transform matrix, representing its fast algorithm, is given by

$$\mathbf{P}_{LOT} = \begin{bmatrix} \mathbf{U}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{V}_1 \end{bmatrix} \begin{bmatrix} \mathbf{D}_e - \mathbf{D}_o & (\mathbf{D}_e - \mathbf{D}_o)\mathbf{J}_{M/2} \\ \mathbf{D}_e - \mathbf{D}_o & -(\mathbf{D}_e - \mathbf{D}_o)\mathbf{J}_{M/2} \end{bmatrix} \tag{3.28}$$

where $\mathbf{D}_e$ is the $M/2 \times M$ matrix with the even-symmetric basis functions of the DCT and $\mathbf{D}_o$ is the matrix with the odd-symmetric ones (see [44], [49] for details). Examining the symmetries of $\mathbf{D}^{II}$, it is easy to see that $\mathbf{D}_e$ and $\mathbf{D}_o$ have the entries d_{ij}^{II} for i even and odd, respectively. $\mathbf{U}_1$ and $\mathbf{V}_1$ are $M/2 \times M/2$ orthogonal matrices. The design suggested

for the LOT [44], [49] uses $\tilde{\mathbf{\Lambda}}$ instead of $\mathbf{\Lambda}$, $\mathbf{U}_1 = \mathbf{I}_{M/2}$, and approximates $\mathbf{V}_1$ by $M/2-1$ plane rotations [44], [49].

With the results in (3.19) and (3.24) we can formulate the complete factorization for a generalized linear-phase LOT (GenLOT), which can be defined for any integer overlap. The GenLOT is defined as a LPPUFB obeying (3.19), where $\mathbf{E}_0$ is chosen to be the DCT matrix [88], which we denote as $\mathbf{D}^{II}$. The output of the DCT is, then, separated into the groups of even and odd coefficients. The GenLOT with $N-1$ stages after the DCT has basis functions (filters) with length $L = NM$ and has its PTM defined as

$$\mathbf{E}(z) = \mathbf{K}_{N-1}(z)\mathbf{K}_{N-2}(z)\cdots\mathbf{K}_1(z)\mathbf{D}^{II}. \tag{3.29}$$

The implementation flow-graphs for the analysis and synthesis sections are shown in Fig. 3.4.

The class of GenLOTs, defined in this way, allow us to view the DCT and LOT as special cases, respectively for $N = 1$ and $N = 2$. The degrees of freedom reside on the matrices $\mathbf{U}_i$ and $\mathbf{V}_i$ which are only restricted to be real $M/2 \times M/2$ orthogonal matrices. Thus, each one can be parameterized into a set of $M(M-2)/8$ plane rotations. Each plane rotation represents one degree of freedom in the design and can be implemented with either 3 additions and 3 multiplications or 2 additions and 4 multiplications. In either case, the total number of floating-point operations (flops) is 6. For $N-1$ stages after the DCT, this results in a total of $M(N-1)(M-2)/4$ degrees of freedom. For example, for $M = 8$, $\mathbf{U}_i$ and $\mathbf{V}_i$ are 4×4 orthogonal matrices. Hence, each one can be parameterized as a cascade of 6 plane rotations, as shown in Fig. 3.5. $\mathbf{U}_i$ and $\mathbf{V}_i$ can be

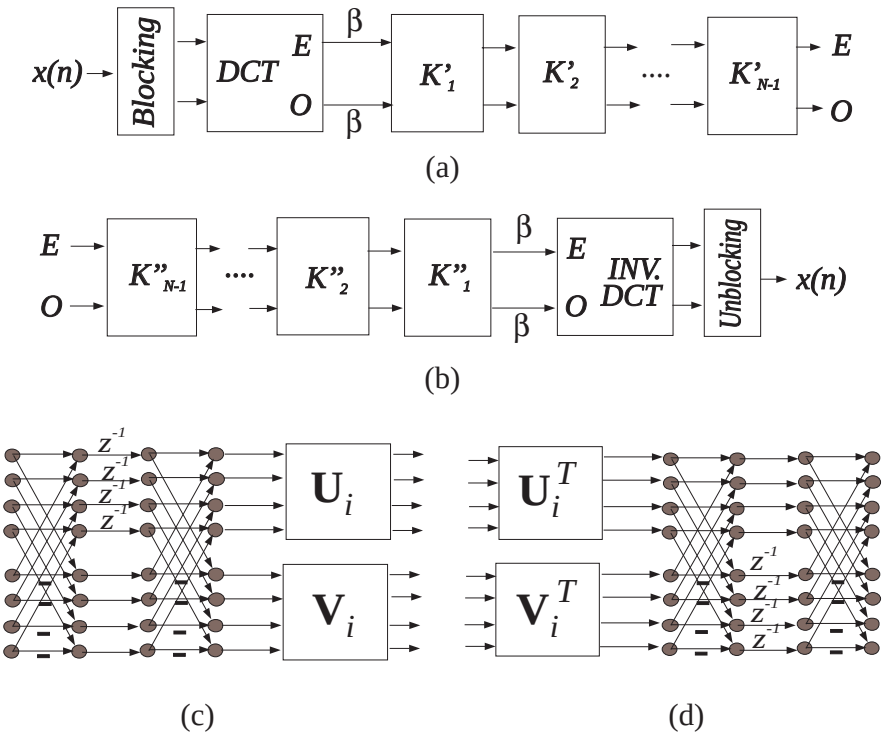


Figure 3.4: Flow-graph for implementation of GenLOTs. Each branch carries $M/2$ samples and E and O stand for the even and odd transform coefficients, respectively, of output (analysis) and input (synthesis) for both DCT and GenLOT. Even and odd coefficients also correspond to symmetric and anti-symmetric basis functions (which are the filters' impulse responses), respectively. β is a scaling factor incorporating all scaling factors present in $\mathbf{W}$, so that $\beta = 2^{-(N-1)}$. (a) Analysis; (b) synthesis; (c) details of the analysis stages K_i', for $M = 8$; (d) details of the synthesis stages K_i'', for $M = 8$.

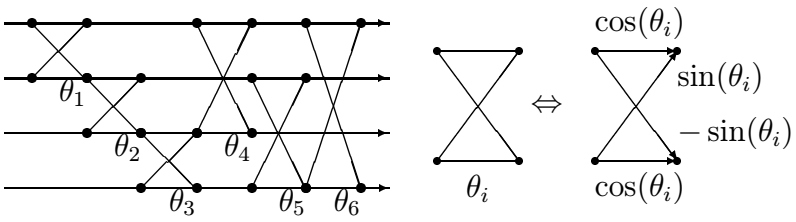


Figure 3.5: Implementation of a 4×4 orthogonal matrix through plane rotations. The detail of each plane rotation is shown on the right.

implemented with $3M(M-2)/4$ flops, each, using plane rotations, or $(M-1)M/2$ flops using direct matrix multiplication. Note that for $M > 4$ it is advantageous to use direct matrix multiplication to implement each factor ($\mathbf{U}_i$ or $\mathbf{V}_i$) than to use plane rotations. For $M = 4$ the number of flops is the same, and there are no LPPUFBs for $M = 2$ [104]. So, plane rotations are just useful for the design of $\mathbf{\Phi}_i$ and not for their implementation. One can achieve a reduction in the implementation cost, by forcing each matrix to be composed by a reduced set of plane rotations, let us say $(M/2)-1$. For $M = 8$, a matrix with only 3 plane rotations is shown in Fig. 3.6. Using only matrices parameterized in this form, the total number of degrees of freedom is reduced to $(N-1)(M-2)$, a reduction of a factor of $M/4$. Each matrix can be implemented with $3M-6$ flops compared to $(M-1)M/2$ flops in direct matrix multiplication.

In our notation, we can say that the GenLOTs are LTs with symmetric delay factorization, for which

$$\mathbf{B}_0 = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{U}_{N-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_{N-1} \end{bmatrix} \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & -\mathbf{I}_{M/2} \end{bmatrix}$$

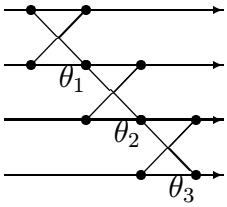


Figure 3.6: Implementation of a constrained 4×4 orthogonal matrix using only 3 plane rotations.

$$\mathbf{B}_i = \frac{1}{2} \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & -\mathbf{I}_{M/2} \end{bmatrix} \begin{bmatrix} \mathbf{U}_{N-1-i} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_{N-1-i} \end{bmatrix} \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & -\mathbf{I}_{M/2} \end{bmatrix}$$

$$\mathbf{B}_{N-1} = \mathbf{D}^{II}$$

for $1 \leq i \leq N-2$.

3.4 Design Aspects

Mostly, the signal is assumed to have an infinite number of samples and stationary statistics, so that the design of $\mathbf{P}$ to fit in a given model suffices for its application. A cost/benefit trade-off has to be established and weighted before designing the LT. Both GenLOTs and ELTs have fast implementation algorithms and several degrees of freedom that one can manipulate to shape the impulse response of the respective filters. These degrees of freedom are given by the plane rotation angles.

For ELTs, there are $q = L/2$ plane rotations to design. For $K > 2$, there is no simple relation among the rotation angles and the modulating window and non-linear optimization techniques have to be carried. The design of ELTs is so well covered in [49] that we will just concentrate on the design of GenLOTs. For GenLOTs, the

number of degrees of freedom is $q = M(N-1)(M-2)/4$ for the full set of rotations or $q = (N-1)(M-2)$ for the reduced one.

The LOT can be obtained from the DCT, by direct determination of $\mathbf{\Phi}_1$ [49]. In this case, $\mathbf{U}_1$ and $\mathbf{V}_1$ are determined in a general form, without obeying any particular structure. Optimization, in this case, is carried solely to determine an approximation to the matrices $\mathbf{U}_1$ and $\mathbf{V}_1$, found through the techniques described in [49]. For the LOT, $\mathbf{U}_1$ is approximated to $\mathbf{I}_{M/2}$ and $\mathbf{V}_1$ is approximated by a cascade of $\frac{M}{2}-1$ plane rotations [44], through optimization routines. So, for the LOT, the optimization is necessary to find faster implementation algorithms. However, for $N > 2$, there are no techniques available to find directly all matrices $\mathbf{\Phi}_i$. The q-dimensional space of solutions is searched through optimization routines, in such a way as to minimize a particular cost function. However, due to the highly non-linear relationships among the angles and the cost functions, there is no guarantee to obtain a global minimum. All GenLOT examples presented here were obtained using unconstrained non-linear optimization and simplex search, using the routines provided by MATLAB version 4.0 .

Examples of features we can try to maximize in the design are the transform coding gain (G_{TC}) [33] or a measure of the attenuation in the stopband region of each filter, or a combination of both. Other features can be considered as well. Thus, the cost function can be selected as the inverse of any of these functions.

3.4.1 Coding Gain

Let the autocorrelation matrix for $x(n)$ be $\mathbf{R}_{xx}$, then $y(n)$ has an autocorrelation matrix given by [44]

$$\mathbf{R}_{yy} = \mathbf{P}\mathbf{R}_{xx}\mathbf{P}^T \tag{3.30}$$

with elements $r_{yy}(i,j)$. If the diagonal elements are $\sigma_i^2 = r_{yy}(i,i)$, and if we want to maximize the coding gain [33], the cost function to be minimized is the inverse of the coding gain as

$$COST = -G_{TC(dB)} = 10\log_{10}\left(\frac{\left(\prod_{i=0}^{M-1}\sigma_i^2\right)^{1/M}}{\frac{1}{M}\sum_{i=0}^{M-1}\sigma_i^2}\right). \tag{3.31}$$

In the design of any LT, we can speed up the optimization by not optimizing the last stage, i.e., $\mathbf{B}_0$. This is possible by using the method applied by Malvar [44] for the LOT. For this, in the recursion to find $\mathbf{P}$, assume

$$\mathbf{P} = \mathbf{P}^{(N-1)} = \mathbf{B}_0 \begin{bmatrix} \mathbf{I}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \end{bmatrix} \begin{bmatrix} \mathbf{P}^{(N-2)} & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{P}^{(N-2)} \end{bmatrix},$$

so that $\mathbf{P} = \mathbf{B}_0\mathbf{P}_{inc}$, where $\mathbf{P}_{inc}$ contains the remaining factors. The matrix $\mathbf{B}_0$ for maximum decorrelation of the input signal (given matrices $\mathbf{B}_1$ through $\mathbf{B}_{N-1}$, and a statistical model for the input) is given by the matrix whose rows are the M eigenvectors of $\tilde{\mathbf{P}}_{inc}\mathbf{R}_{xx}\tilde{\mathbf{P}}_{inc}^T$ [44].

In the case of GenLOTs, for $N = 4$ (three times the overlap amount present in the LOT), it is only necessary to optimize two out of four stages (because the first stage is

DCT and the last stage is determined by the remaining ones). For a reduced set of angles this method does not make sense because it would force us to run a second optimization to approximate $\mathbf{U}_{N-1}$ and $\mathbf{V}_{N-1}$ by a series of $M/2-1$ plane rotations each.

Assuming the input signal as a zero mean AR(1) signal with adjacent sample correlation coefficient 0.95 (i.e., its autocorrelation function is $r_x(n) = 0.95^{|n|}$), the basis functions of a GenLOT for $M = 8$, optimized for maximum G_{TC} are shown in Fig. 3.7, for $N = 4$ ($L = 32$) and $N = 5$ ($L = 40$).

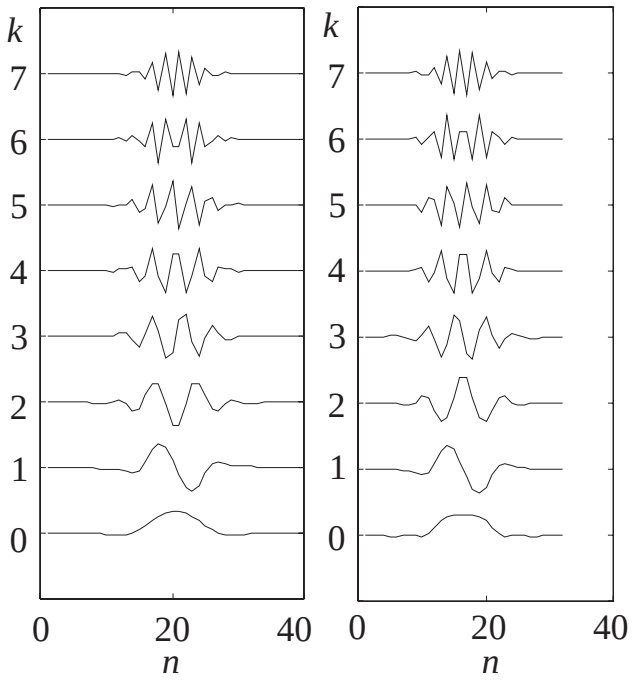


Figure 3.7: Basis functions $f_k(n)$ (filters' impulse responses) of a GenLOT with $M = 8$ designed for maximum G_{TC}. Examples for $L = 40$ (left) and $L = 32$ (right) are shown.

The design of LOT based on maximum G_{TC} is not necessarily the best one for image coding, even considering that the AR(1) process is, in general, a good model for images. For example, the "smoothness" of the basis functions is an important issue, because in low bit-rate coding only few coefficients are non-zero, thus the signal is reconstructed

using only few basis functions. If these basis functions are very concentrated or have "bends" or "edges", then these will produce visible patterns in the reconstructed image, which could have a better aspect if the lowest frequency basis functions were smoother, even though they could lead to a GenLOT with lower G_{TC}.

3.4.2 Stopband Attenuation

Another criteria for the design of the GenLOT can be the maximization of the stopband attenuation of the filters $f_k(n)$ $(0 \leq k \leq M-1,\, 0 \leq n \leq L-1)$. Let $F_k(e^{j\omega})$ be the Fourier transform of $f_k(n)$ which is a band-pass filter with low and high cut-off frequencies denoted by $\omega_{k,L}$ and $\omega_{k,H}$. Let the filters be sorted by their frequency slots so that

$$\omega_{k,L} = k\pi/M, \quad \omega_{k,H} = (k+1)\pi/M.$$

The stopband region Ω_k corresponding to $f_k(n)$ is defined by

$$\begin{aligned} \Omega_0 &\equiv \{\omega \mid |\omega| \in [\omega_{0,H} + \epsilon, \pi))\} \\ \Omega_k &\equiv \{\omega \mid |\omega| \in ([0, \omega_{k,L} - \epsilon] \cup [\omega_{k,H} + \epsilon, \pi))\} \\ \Omega_{M-1} &\equiv \{\omega \mid |\omega| \in [0, \omega_{M-1,L} - \epsilon]\} \end{aligned} \tag{3.32}$$

where ϵ is a small positive real number used to reduce the influence of the transition region into the stopband region.

A possible cost function to be minimized can be the energy of the filters frequency response in the stopband region, defined as

$$COST = \sum_{k=0}^{M-1} \int_{\omega \in \Omega_k} |F_k(e^{j\omega})|^2 \, d\,\omega. \tag{3.33}$$

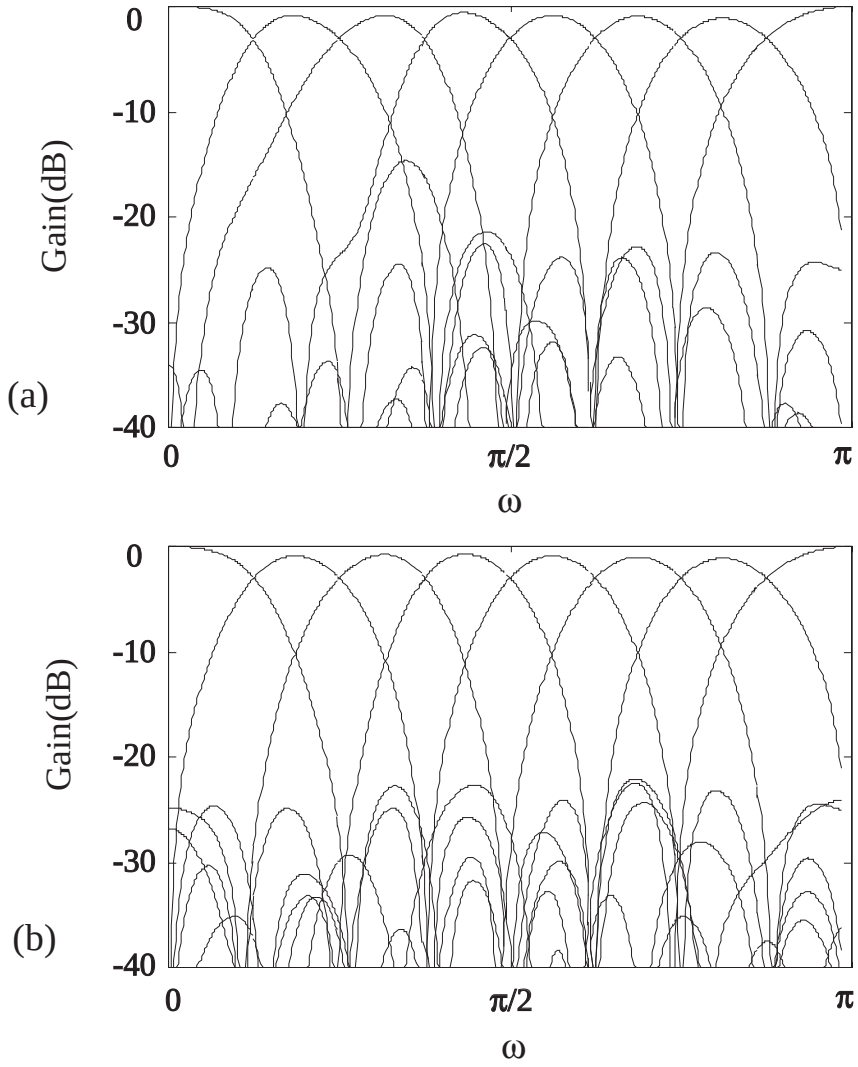


Figure 3.8: Filters' frequency responses ($20 \log_{10} |F_k(e^{j\omega})|$), given in dB, of the GenLOT with $L = 32$, $M = 8$, $N = 4$. (a) Designed for maximum G_{TC}. (b) Designed for maximum stopband attenuation.

For a direct comparison, in Fig. 3.8 it is shown the frequency response plots for a GenLOT with $M = 8$ optimized for maximum stopband attenuation and for maximum G_{TC}.

CHAPTER 4

TIME-VARYING LAPPED TRANSFORMS

In the previous sections, we dealt with time-invariant filter banks in a sense that filters with linear time invariant impulse responses were used. We can extend this to the concept of time-varying analysis-synthesis systems and the possibility to maintain perfect reconstruction while changing filter banks. Furthermore, we would like to find the PR conditions for time-varying filter banks and verify to what extent we can make variations.

Recently, other researchers obtained independent results on the subject of time-varying filter banks [4], [12], [25], [28], [29], [30], [31], [57], [99], which are distinct from our perspective. Some of the topics described in this section can be also found in [71], [72], [75], [76].

4.1 Flow-graphs and orthogonality

The key for achieving PR time-varying filter banks and LTs resides on a very simple fact, generally not considered in the theory of multirate filter banks. Consider Fig. 4.1, where it is shown the flow graph for implementing a fictitious system, leading 8 input samples to the output, as we follow the paths from left to right. The boxes contain orthogonal matrices meaning orthogonal linear operations over the input. As the number of input samples is equal to the number of output samples, there are no sources or

drains for the paths, all internal operators are orthogonal, hence, one can always obtain the inverse by following the paths from right to left and replacing the matrices by their transposes. In other words, the 8-samples transform is itself orthogonal and enjoys PR, since we always can recover the input samples from the output ones.

Now, let us review the flow-graphs shown in Fig. 2.6. The z-domain flow-graphs in Fig. 2.6(a) and Fig. 2.6(b) led us to the time-domain flow-graphs of Fig. 2.6(c) and Fig. 2.6(d), which, in the light of what we have just discussed, form an orthogonal system leading input time-domain samples to output transform-domain samples. It is, thus, easy to find the samples in any domain with the knowledge of the samples in the other domain.

Suppose, instead, that the orthogonal matrices in Fig. 2.6 are no longer identical along the time index. We would arrive at the flow-graphs shown in Fig. 4.2. Similarly, the causal (z-domain) and non-causal blockwise (time-domain) flow-graphs are shown. In the figures, all branches carry $M/2$ branches and blocks $\mathbf{B}_i(m)$ are $M \times M$ orthogonal matrices. As all blocks are still orthogonal, although not necessarily identical within a stage, the whole orthogonality is still maintained, and analysis is carried by following the paths from left to right while synthesis is carried by following the paths from right to left, using the transposes of the matrices $\mathbf{B}_i(m)$. Time indices for different stages in Fig. 4.2(c) are changed for a more convenient reference, in relation to the indexes shown in parts (a) and (b).

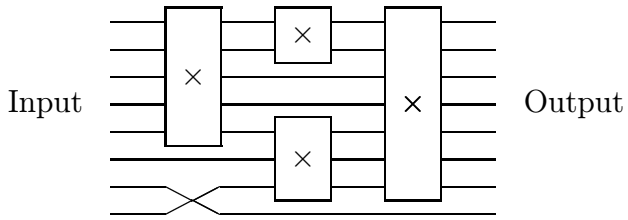


Figure 4.1: A flow-graph relating 8 input and output samples in a fictitious system. The $\times$ boxes represent orthogonal matrices, and, therefore they can also be permutation matrices.

4.2 PR time-varying LTs

From Fig. 4.2, we can see that PR is inherent with this new degree of freedom, independently of the choice of the orthogonal matrices. It is also very easy to see that this new time-varying flow-graph corresponds to a time-varying filter bank or to a time-varying LT. The recipe to find the time-varying filter bank from the flow-graphs is simple: find the input blocks which are connected in any way to output block m and arrange them in a vector $\mathbf{v}_m$; find $\mathbf{P}(m)$ such that the m-th output block ($\mathbf{y}_m$) is given by $\mathbf{y}_m = \mathbf{P}(m)\mathbf{v}_m$. In this way, the transform matrix operating over the entire signal extent in (2.11) is rewritten as

$$\mathbf{H} = \begin{bmatrix} \ddots & & & & \\ & \mathbf{P}(m-1) & & & \\ & & \mathbf{P}(m) & & \\ & & & \mathbf{P}(m+1) & \\ & & & & \ddots \end{bmatrix}, \tag{4.1}$$

where the displacement of the transform matrices $\mathbf{P}(m)$ still obeys the following

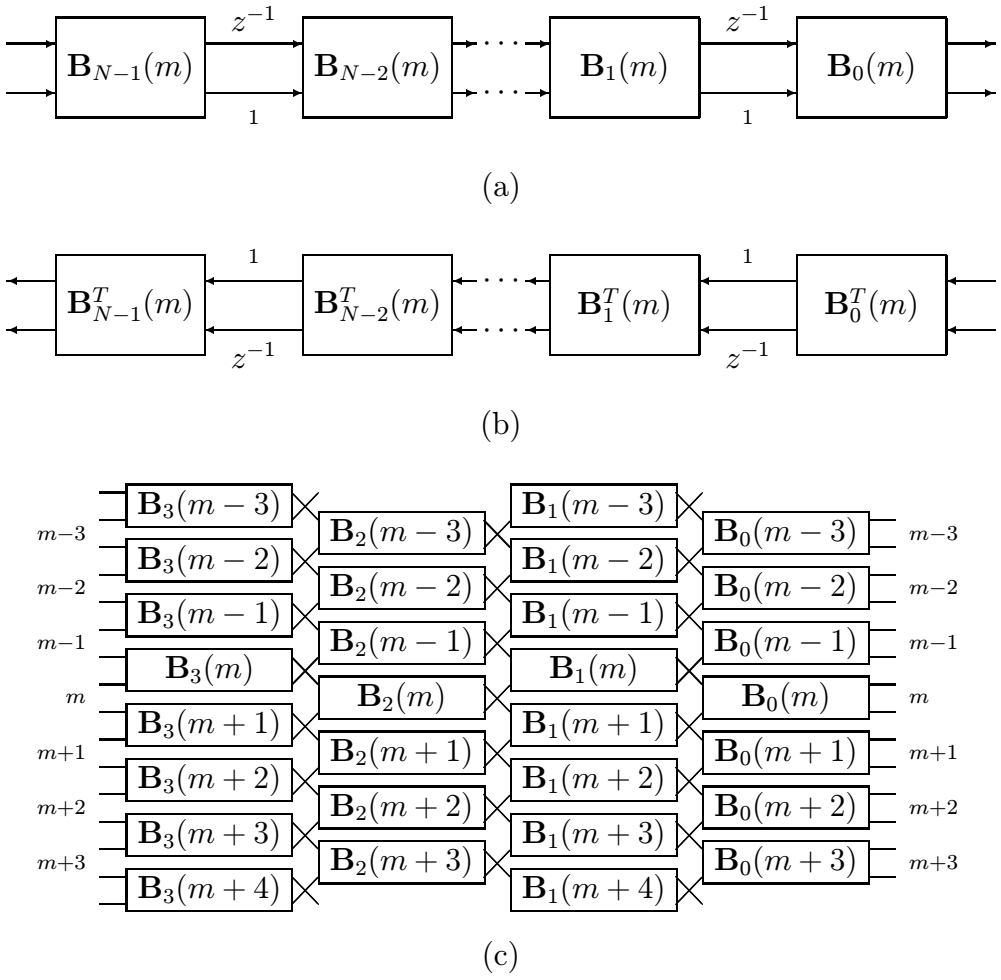


Figure 4.2: Flow graph for time-varying paraunitary FIR filter banks using symmetric delays and N stages. The z-domain and time-domain flow-graphs are shown. In the figures, all branches carry $M/2$ branches and blocks $\mathbf{B}_i(m)$ are $M \times M$ orthogonal matrices. (a) Analysis section; (b) synthesis section; (c) time domain blockwise flow-graphs corresponding to the symmetric delay factorizations. As all blocks are orthogonal, analysis is carried by following the paths from left to right while synthesis is carried by following the paths from right to left, using the matrices transposes. Time indices for different stages in (c) are changed for a more convenient reference, in relation to the index shown in parts (a) and (b). PR is inherent, independent of the choice of the orthogonal matrices.

$$\mathbf{H} = \begin{bmatrix} \ddots & \ddots & & \ddots & & & \\ \mathbf{P}_0(m-1) & \mathbf{P}_1(m-1) & \cdots & \mathbf{P}_{N-1}(m-1) & & & \\ & \mathbf{P}_0(m) & \mathbf{P}_1(m) & \cdots & \mathbf{P}_{N-1}(m) & & \\ & & \mathbf{P}_0(m+1) & \mathbf{P}_1(m+1) & \cdots & \mathbf{P}_{N-1}(m+1) & \\ & & & \ddots & \ddots & \ddots & \end{bmatrix}. \tag{4.2}$$

For example, in the case $N = 4$ shown in Fig. 4.2, we have

$$\mathbf{P}(m) = \mathbf{B}_0(m) \begin{bmatrix} \mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{B}_1(m) & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_1(m+1) \end{bmatrix} \begin{bmatrix} \mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I} \end{bmatrix}$$

$$\cdot \begin{bmatrix} \mathbf{B}_2(m-1) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_2(m) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_2(m+1) \end{bmatrix} \begin{bmatrix} \mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} \\ \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I} \end{bmatrix}$$

$$\cdot \begin{bmatrix} \mathbf{B}_3(m-1) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_3(m) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_3(m+1) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_3(m+2) \end{bmatrix}$$

where $\mathbf{I}$ and $\mathbf{0}$ are of size $M/2 \times M/2$ and their subscript was dropped to simplify the notation. In general, we have

$$\mathbf{P}(m) = \mathbf{B}'_0(m)\mathbf{D}_1\mathbf{B}'_1(m)\cdots\mathbf{D}_{N-1}\mathbf{B}'_{N-1}(m). \tag{4.3}$$

where $\mathbf{B}'_i(m)$ has size $(i+1)M \times (i+1)M$ and $\mathbf{D}_i$ has size $iM \times (i+1)M$, for $0 \leq i \leq N-1$. To find the matrices $\mathbf{D}_i$ start with

$$\mathbf{D}_1 = \begin{bmatrix} \mathbf{I}\ \mathbf{0}\ \mathbf{0}\ \mathbf{0} \\ \\ \mathbf{0}\ \mathbf{0}\ \mathbf{0}\ \mathbf{I} \end{bmatrix}$$

and then find $\mathbf{D}_i$ $(1 < i \leq N-1)$, recursively from

$$\mathbf{D}_i = \begin{bmatrix} \mathbf{D}_{i-1} & \mathbf{0}_{(i-1)M \times M} \\ \\ \mathbf{0}_{M \times M} & \mathbf{D}_1 \end{bmatrix}. \tag{4.4}$$

The matrices $\mathbf{B}'_j(m)$ are determined by

$$\mathbf{B}'_{2i+1}(m) = diag\{\mathbf{B}_{2i+1}(m-i), \ldots, \mathbf{B}_{2i+1}(m+1+i)\} \tag{4.5}$$

$$\mathbf{B}'_{2i}(m) = diag\{\mathbf{B}_{2i}(m-i), \ldots, \mathbf{B}_{2i}(m+i)\} \tag{4.6}$$

for $0 \leq i \leq K-1$.

It is easy to see that a particular choice of $\mathbf{B}_0(m)$ will influence solely $\mathbf{P}(m)$, while a particular choice of $\mathbf{B}_i(m)$ $(i > 0)$ can influence several instants of $\mathbf{P}(m)$. This suggests that any change of the filter bank is preceded and followed by transition regions.

The time-domain PR equations, in (2.7) and in Sec. 2.3.3, assume that $\mathbf{P}$ remains unchanged along the time-index. Therefore, the analysis can be more easily presented due to the periodic nature of the problem. In time-varying systems, we have to choose an index m and find the PR equations for it, noting that (2.7) is no longer valid. Roughly speaking, the PR conditions for steady systems must state the orthogonality of the basis functions and the so-called orthogonality of the "tails". This implies that aliased (shifted) versions of $\mathbf{P}$ would cancel in the synthesis process. Since $\mathbf{P}(m)$ is no longer constant with

m, (2.7) must be rewritten to ensure the orthogonality among $\mathbf{P}(m)$ and its neighbors $\mathbf{P}(m \pm 1), \mathbf{P}(m \pm 2), \mathbf{P}(m \pm 3) \ldots$. Then, it is easy to show that the PR conditions for time-varying LTs are

$$\sum_{k=0}^{N-1-\ell} \mathbf{P}_k(m)\mathbf{P}_{k+\ell}^T(m-\ell) = \sum_{k=0}^{N-1-\ell} \mathbf{P}_{k+\ell}(m)\mathbf{P}_k^T(m+\ell) = \delta(\ell)\mathbf{I}_M \tag{4.7}$$

for $\ell = 0, 1, \ldots, N-1$, yielding $2N-1$ independent matrix equations.

As a remark, the term lapped transform was maintained (although (2.7) is no longer valid) because $\mathbf{H}$ remains orthogonal and for each instant m, the synthesis filters are time-reversed versions of the analysis ones. Thus, the filter bank is said to be *instantaneously paraunitary*. In this case, (4.7) is satisfied, but not necessarily the steady conditions in (2.7) are met. In this situation, the filter bank is said to be *transitory*. Of course, all equations are satisfied when the filter bank is not time-varying.

One important point not discussed so far regards to what extent we can change the LT. From our assumptions, using the symmetric delay factorization, independently of the choice of $\mathbf{B}_i(m)$ we can only change the frequency response of the filters. The maximum length L (overlap amount) and the number of channels M (block size) cannot be changed in this simple fashion. Actually, L can change assuming any value l such that $M \leq l \leq L$. However, it is possible to freely change L and M, although complicating the notation a little more.

4.3 Changing the number of filters (M)

To change the number of filters, M, using the symmetric delay factorization, we can start by assuming that at one point in time we are using a LT with M_1 channels, and, from that point on it is desired to use a LT with M_2 channels. The flow-graph shown in Fig. 4.2(c) would still be valid noting that on the top of this figure, $M = M_1$, while on the bottom $M = M_2$. In other words, for the beginning of the flow-graph the branches carry $M_1/2$ samples, and the boxes are $M_1 \times M_1$ orthogonal matrices, while at the bottom of that flow-graph the branches would carry $M_2/2$ samples and the boxes are $M_2 \times M_2$ orthogonal matrices. Therefore, there would be a transition where a matrix $\mathbf{B}_i(m)$ would be connected to branches with both M_1 and M_2 samples and have size $(M_1 + M_2)/2 \times (M_1 + M_2)/2$ as shown in Fig. 4.3. Note that as $\mathbf{B}_i(m)$ is chosen to be a $(M_1 + M_2)/2 \times (M_1 + M_2)/2$ orthogonal matrix, PR and full orthogonality are assured by the same principles allowing PR time-varying LTs for the $M_1 = M_2 = M$ case. The choice of this orthogonal matrix has to be studied carefully so as not to allow undesirable transitory filter banks.

4.4 Changing the maximum filter length (L)

The overlap amount (filters' length) can be changed not only by altering the impulse responses of filters representing a PTM of a certain degree (setting elements on both extremes to zero), but also by changing the degree of the instantaneous PTM. For this, one shall add or delete stages to the factorization. The addition of stages can be carried either as post-processing or as pre-processing operations. Consider the transform

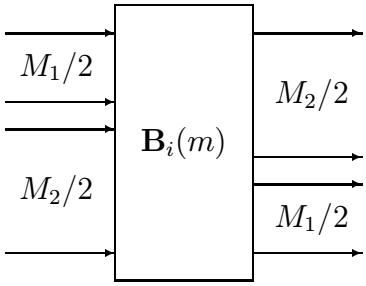


Figure 4.3: In the transition from M_1-channel LT to a M_2-channel LT, there will be a sparse factor with input-output branches carrying both M_1 and M_2 samples. Note that as $\mathbf{B}_i(m)$ is chosen to be a $(M_1 + M_2)/2 \times (M_1 + M_2)/2$ orthogonal matrix, PR and full orthogonality are assured by the same principles allowing PR time-varying LTs for the $M_1 = M_2 = M$ case.

matrix $\mathbf{H}$ as a function of the stages containing the delays (permutations) and orthogonal matrices (sparse factors). Let the orthogonal matrices stages be denoted by

$$\tilde{\mathbf{B}}_i = diag\{\ldots, \mathbf{B}_i(m-1), \mathbf{B}_i(m), \mathbf{B}_i(m+1), \ldots\} \tag{4.8}$$

and let $\tilde{\mathbf{D}}$ be a permutation matrix (which is also orthogonal) representing the symmetric delays, assuming the pattern of an infinite recursion of (4.4). Then, it is easy to see that

$$\mathbf{H} = \tilde{\mathbf{B}}_0 \prod_{i=1}^{N-1} \tilde{\mathbf{D}}\tilde{\mathbf{B}}_i. \tag{4.9}$$

The reader can see that the entire analysis (or synthesis) process of a LT is a cascade of delays and block transforms appended to the output (or input) of a smaller degree LT. Thus,

$$\mathbf{H}_{new} = \tilde{\mathbf{B}}_0 \prod_{i=1}^{N} \tilde{\mathbf{D}}\tilde{\mathbf{B}}_i = \mathbf{H}\tilde{\mathbf{D}}\tilde{\mathbf{B}}_N \tag{4.10}$$

can be thought of as a form of preprocessing the input sequence. Accordingly, in a post-processing perspective, if we rewrite $\mathbf{H}$ as

$$\mathbf{H}_{old} = \tilde{\mathbf{B}}_1 \prod_{i=2}^{N} \tilde{\mathbf{D}}\tilde{\mathbf{B}}_i, \tag{4.11}$$

where we just changed the labels of the stages, then

$$\mathbf{H}_{new} = \tilde{\mathbf{B}}_0 \prod_{i=1}^{N} \tilde{\mathbf{D}}\tilde{\mathbf{B}}_i = \tilde{\mathbf{B}}_0\tilde{\mathbf{D}}\mathbf{H}_{old}. \tag{4.12}$$

The key to make a time-varying use of the degree change, is to only apply the pre- or post-processing stages from a certain point in time. For increasing the degree, for example, one can bypass the stage of delays and orthogonal matrices before the chosen point in time. Hence, for a stage to be bypassed,

$$\tilde{\mathbf{B}}_i = diag\{\ldots, \mathbf{I}_M, \mathbf{I}_M, \mathbf{B}_i(m), \mathbf{B}_i(m+1), \ldots\}. \tag{4.13}$$

The delay stage in the transition will, accordingly, have the form

$$\begin{bmatrix} \ddots & & & \\ & \mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0} & & \\ & \mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{0}\,\mathbf{I} & & \\ & & \ddots \end{bmatrix}.$$

Again, a careful examination of the transition regions may be carried in each case.

4.5 Switching LTs and transitions

A continuous change of filter banks, like in adaptive filtering, is possible, but its effectiveness is, at least, arguable, due to the transition period inherent of a change of any of the factors $\mathbf{B}_i(m)$. Maybe the most practical idea would be to switch between two filter banks at a time. Each one would have its own characteristics (and factorization) well defined.

$$\cdots \rightarrow \text{filter bank 1} \rightarrow \text{filter bank 2} \rightarrow \text{filter bank 3} \rightarrow \cdots$$

If we use the complete factorization of (2.45), it is, thus, possible to switch between any two LTs at a time. Similarly, one can switch between any two SDF-LTs. Also, we can go further and use ELTs to switch between any two ELTs (the same for a LPPUFB), and so on. In any case, a more detailed study of the transition is necessary. For a given change of LTs, non-linear optimization may be carried to design the angles constituting the transitory factors $\mathbf{B}_i$. However, this may not be the only way to fix undesired transitory filters arising in the transitions. Note that one can also switch directly all $\mathbf{B}_i$ factors pertaining to the first LT to factors pertaining to the second LT and use post-processing. For this consider block-diagonal matrix post-processing the output of $\mathbf{Hx}$, so that

$$\mathbf{y} = \mathbf{H}'\mathbf{H}\mathbf{x} \tag{4.14}$$

where

$$\mathbf{H}' = \begin{bmatrix} \mathbf{I}_\infty & & \\ & \mathbf{A}_{pos} & \\ & & \mathbf{I}_\infty \end{bmatrix}, \tag{4.15}$$

and $\mathbf{A}_{pos}$ is a orthogonal matrix of any finite size. It is clear that $\mathbf{H}'$ is also orthogonal and corresponds to some sort of time-varying PUFB. Thus, the post-processing matrix can be used to re-transform the subbands in the transition region. Also, $\mathbf{A}_{pos}$ can be chosen to have size larger than the transition length in order to perform a slower switch between the two LTs. Of course, the design of $\mathbf{A}_{pos}$ is dependent on the LTs and on a particular cost function. We can choose its size, factorize it into plane rotations, and use the same COST functions and design methods discussed in the previous chapter.

Alternatively, we may try to improve transitory LTs, trying to match some pre-defined output statistics. For this, we can introduce the *make-up* matrix. Such matrix represents a post-processing operation, as is the case for the matrix $\mathbf{A}_{pos}$ just described. $\mathbf{A}_{pos}$ can have any size we want, but if we restrict it to be of size M, i. e., just post-processing the output of the instantaneous LT $\mathbf{P}(m)$, we may absorb $\mathbf{A}_{pos}$ into $\mathbf{B}_0(m)$, thus the *make-up* matrix to be designed is $\mathbf{B}_0(m)$. This may simplify the notation but the method we are about to discuss can be applied to $\mathbf{A}_{pos}$, as well. Assume $\mathbf{P}(m)$ belongs to a transition and, using (4.3), let

$$\mathbf{P}'(m) = \mathbf{D}_1\mathbf{B}'_1(m)\cdots\mathbf{D}_{N-1}\mathbf{B}'_{N-1}(m) \tag{4.16}$$

so that

$$\mathbf{P}(m) = \mathbf{B}_0(m)\mathbf{P}'(m). \tag{4.17}$$

$\mathbf{v}_m$ is the vector containing the samples of $x(n)$ transformed by $\mathbf{P}(m)$. Assume $x(n)$ has an autocorrelation matrix $\mathbf{R}_{xx}$, and let

$$\mathbf{w} = \mathbf{P}'(m)\mathbf{v}_m \tag{4.18}$$

which has a pseudo autocorrelation matrix $\mathbf{R}_{ww}(m)$, given by

$$\mathbf{R}_{ww}(m) = \mathbf{P}'(m)\mathbf{R}_{xx}\mathbf{P}'^T(m). \tag{4.19}$$

The word "pseudo" was added to the autocorrelation matrix because the transform itself is variable, so that the output signal is also time-varying, therefore, rigorously it cannot be stationary. However, we think of these signals as if the transitory LT was to be held steady indefinitely for a stationary input signal.

The statistics of $x(n)$, in $\mathbf{R}_{xx}$, are an assumption and $\mathbf{P}'(m)$ is given, hence $\mathbf{R}_{ww}$ is known. Assume also that we want the output signal $\mathbf{y}_m = \mathbf{P}(m)\mathbf{v}_m$ (for that particular transitory LT) to have a pseudo autocorrelation matrix $\mathbf{R}_{yy}(m) = E\{\mathbf{y}_m\mathbf{y}_m^T\}$ of any particular form (for example diagonal, as in signal compression applications). Note that $\mathbf{R}_{ww}(m)$ is a Toeplitz symmetric matrix which can be diagonalized by an orthogonal matrix $\mathbf{\Phi}_w$ as

$$\mathbf{\Lambda} = \mathbf{\Phi}_w\mathbf{R}_{ww}(m)\mathbf{\Phi}_w^T \tag{4.20}$$

where both $\mathbf{\Phi}_w$ and $\mathbf{\Lambda}$ are $M \times M$ matrices, $\mathbf{\Lambda}$ contains the eigenvalues of $\mathbf{R}_{ww}(m)$, while the rows of $\mathbf{\Phi}_w$ are the eigenvectors of $\mathbf{R}_{ww}(m)$. If we choose $\mathbf{R}_{yy}(m)$ such that

$$\mathbf{R}_{yy}(m) = \mathbf{\Phi}_y^T\mathbf{\Lambda}\mathbf{\Phi}_y \tag{4.21}$$

for some orthogonal matrix $\mathbf{\Phi}_y$, we can say that $\mathbf{\Phi}_y$ diagonalizes $\mathbf{R}_{yy}(m)$ and

$$\mathbf{R}_{yy}(m) = \mathbf{\Phi}_y^T \mathbf{\Phi}_w \mathbf{R}_{ww}(m) \mathbf{\Phi}_w^T \mathbf{\Phi}_y. \tag{4.22}$$

Using (4.17) and (4.19) we can see that

$$\mathbf{R}_{yy}(m) = \mathbf{\Phi}_y^T \mathbf{\Phi}_w \mathbf{P}'(m) \mathbf{R}_{xx} \mathbf{P}'^T(m) \mathbf{\Phi}_w^T \mathbf{\Phi}_y = \mathbf{B}_0(m) \mathbf{P}'(m) \mathbf{R}_{xx} \mathbf{P}'^T(m) \mathbf{B}_0^T(m). \tag{4.23}$$

Hence, we get

$$\mathbf{B}_0(m) = \mathbf{\Phi}_y^T \mathbf{\Phi}_w. \tag{4.24}$$

Given the restriction in (4.21), $\mathbf{\Phi}_w$ and $\mathbf{\Phi}_y$ contain the eigenvectors of $\mathbf{R}_{ww}(m)$ and $\mathbf{R}_{yy}(m)$, respectively, while $\mathbf{\Lambda}$ contains the eigenvalues of both. Also, $\mathbf{\Lambda}$, $\mathbf{\Phi}_w$ and $\mathbf{\Phi}_y$ are similar, i.e. are related by similarity transformations [32].

The design of the *make-up* matrix consists in solving (4.21) for $\mathbf{\Phi}_y$, for particular assumptions of $\mathbf{R}_{yy}(m)$ and $\mathbf{R}_{xx}$ ($\mathbf{\Lambda}$ is found from $\mathbf{R}_{xx}$ and from $\mathbf{P}'(m)$). This procedure may be carried interactively, until finding suitable matrix $\mathbf{R}_{yy}(m)$, with characteristics close to those originally desired, such that an orthogonal matrix $\mathbf{\Phi}_y$ in (4.21) does exist for a given $\mathbf{\Lambda}$. For example, if decorrelation is desired, as in signal compression applications, one can set directly $\mathbf{\Phi}_y = \mathbf{I}_M$. We can expect some improvement, by using the *make-up* matrix, but this method is limited. The reason so is because $\mathbf{P}(m)$ brings elements from $\Re^L$ to a subspace $\Re^M$, while $\mathbf{B}_0(m)$ can just optimize the transform inside the subspace selected by $\mathbf{P}'(m)$.

For more complicated approaches such as applying $\mathbf{A}_{pos}$ to a set of output blocks, the same concept applies, although $\mathbf{P}'(m)$ and some other factors have to be changed to conform to the new situation.

The change of orthogonal factors, other than in the last stage, may be performed by optimizing the angles forming the sparse orthogonal factors for a given cost function. However, we can also narrow our search by starting an optimization process from a possibly non-orthogonal matrix, which will be a combination of the sparse factors before and after transition. If such factor is non-orthogonal, one can select an orthogonal approximation (e. g. using its QR factorization [32]).

4.6 Turning an LT "on" and "off" using the bypass LT

An important filter bank is the most trivial structure we can implement, i.e., the bypass LT. This filter bank is characterized by

$$\mathbf{P}(m) = \mathbf{P}_{\text{bypass}} = [\mathbf{0}_{M\times(L-M)/2}, \mathbf{I}_M, \mathbf{0}_{M\times(L-M)/2}]. \tag{4.25}$$

Given a factorization structure, one may find its bypass state which will perform the bypass filter bank and would cause the input samples to be copied to output without any transformation. For the general SDF, we have

$$\mathbf{B}_0 = \mathbf{I}_M \tag{4.26}$$

$$\mathbf{B}_i = \begin{pmatrix} \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \end{pmatrix} \quad 1 \leq i \leq N-1. \tag{4.27}$$

If we are using an LT with a more particular structure, such as the LT or GenLOT, we can always find ways to accommodate a fast switch to a bypass LT. For the ELT we can set all angles in $\mathbf{B}_1$ through $\mathbf{B}_{N-1}$ to $\pi/2$ and replace $\mathbf{B}_0$ by

$$\mathbf{B}_0 = \begin{pmatrix} \mathbf{J}_{M/2} & \mathbf{0}_{M/2} \\ \mathbf{0}_{M/2} & \mathbf{J}_{M/2} \end{pmatrix}^K \begin{pmatrix} \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \end{pmatrix}^{K+1}. \tag{4.28}$$

For the GenLOT or any LPPUFB, we can switch the factors $\mathbf{U}_i$ and $\mathbf{V}_i$, along with $\mathbf{E}_0$ such that

$$\mathbf{U}_i = -\mathbf{V}_i = \mathbf{I}_{M/2} \quad 1 \leq i \leq N-1 \tag{4.29}$$

$$\mathbf{E}_0 = \begin{pmatrix} \mathbf{0}_{M/2} & \mathbf{I}_{M/2} \\ \mathbf{I}_{M/2} & \mathbf{0}_{M/2} \end{pmatrix}. \tag{4.30}$$

Also, we have to apply a postprocessing stage. If the transform matrix with its orthogonal factors modified according to (4.29) and (4.30) is denoted as $\mathbf{P}'$, then

$$\mathbf{P}_{\text{bypass}} = \mathbf{W}\mathbf{P}' \tag{4.31}$$

The reader can confirm these statements either by direct inspection and substituting the orthogonal factors into the general structure. For example, the bypass LT found using the structure of an ELT is shown in Fig. 4.4.

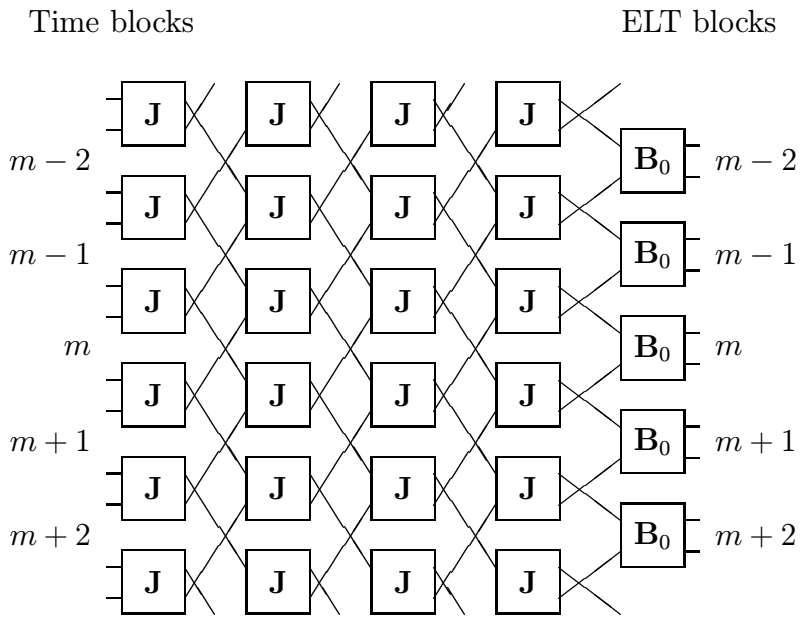


Figure 4.4: Flow graph for the ELT with $K = 4$ under the bypass state, where input is solely copied to output.

If we switch from a regular LT to a bypass state, or vice-versa, there will be a transition region. As the transition region is over, it has no sense in following the implementation flow-graph and input samples can be just copied to output. The notion of the bypass state is only necessary to maintain PR in the transition regions. Although the bypass filter bank is trivial, it is crucial and is the most important factor in the development of the applications of time-varying LTs. It allows to bypass the filter bank as many times as we want, maintaining PR during transitions. Hence, we can associate filter banks in a hierarchical way following the paths of an M-ary tree, and prune or expand branches at our convenience. Also, the bypass filter bank allows us to process

finite-length signals. If the LT is turned to the bypass state on the borders, in such a way that the last filter bank in the transition is applied to the last block in each border, then, the transform over the entire finite-length signal is fully orthogonal enjoying PR also at the borders.

Assume we switch to a regular LT from a bypass LT or vice-versa. The effective transform $\mathbf{H}$ applied to $\mathbf{x}$ will be composed by a block diagonal connection of two matrices: one for the bypass LT (which is of course the identity matrix) and another (denoted by $\mathbf{T}'$) corresponding to the transform applied by the regular LT, including the transition region. As the structure for switching LTs is completely orthogonal, it is clear that $\mathbf{T}'$ is also orthogonal. Thus we can have

$$\mathbf{H} = \begin{pmatrix} \mathbf{I}_\infty & \\ & \mathbf{T}' \end{pmatrix} \quad \text{or} \quad \mathbf{H} = \begin{pmatrix} \mathbf{T}' & \\ & \mathbf{I}_\infty \end{pmatrix} \tag{4.32}$$

As we can turn "on" or "off" the transform anytime we want, it is also possible to have

$$\mathbf{H} = \begin{pmatrix} \mathbf{I}_\infty & & \\ & \mathbf{T}' & \\ & & \mathbf{I}_\infty \end{pmatrix} \quad \text{or} \quad \mathbf{H} = \begin{pmatrix} \mathbf{T}' & & \\ & \mathbf{I}_n & \\ & & \mathbf{T}'' \end{pmatrix} \tag{4.33}$$

where in the first hypothesis we processed/segmented a finite-length signal and, on the second one, the LT is turned "off" for n samples and, then, turned "on" again producing $\mathbf{T}''$. In this fashion, we can segment the signal in any way we want, bypassing the LT anytime we want, and maintain PR and orthogonality at all times. The design of the transition regions (to and from a bypass state) is discussed later.

4.7 Time-varying wavelet packets

As we are able to turn "on" and "off" an LT, it is possible to implement a PR time-varying M-ary DWP. For this, we may express the HLT (or the DWP) as connections of LTs, following the paths of an M-ary tree, and prune or expand branches at any time. To expand a branch, one may add an LT to the terminal branch of the tree and turn it "on" whenever the expansion is supposed to occur. To prune a branch, one may just turn-off the LT representing the particular branch. Note that, in this way, one can change arbitrarily the shape of the tree, generating orthogonal DWP bases and maintaining PR at all times. As we can also change L and M in any branch, we are able to virtually make an arbitrary orthogonal representation of the signal in the TF plane, using any LT.

4.7.1 Notation for general time-varying DWP

In a previous chapter we defined the notation for the DWP and HLT, by associating LTs following the paths of tree as shown in Fig. 2.8 and Fig. 2.9. So, the general hierarchical association of LTs is completely defined by the branches and nodes of the tree representing the connection of LTs. From Fig. 2.8 and Fig. 2.9 we can see that signals flow in the nodes of the tree while the branches connecting the nodes are filters and decimators, representing the filter bank. The first node of a tree (which contains the original signal) is the "root" while the terminal nodes (which carry signals which are not processed by an LT) are called "leaves". Leaf nodes are also called instantaneously virtual end nodes (IVEN) and define the tree bounds momentaneously. If two nodes are connected, the node toward the root is called the parent node, while the one towards

the leaf is called the child node. Although we studied a single LT we need to establish a proper notation to describe the shape of the tree, the signals flowing in the nodes, and the LTs used in each branch.

Let us define a stage as the set of all nodes which are separated from the tree root by the same number of branches, i.e., all nodes in a stage s are output signals after the hierarchical connection of s LTs. Therefore, we label each node in the tree as η_{ij}, where i is the stage number and j is a unique number labeling the node in a stage. Since the shape of the tree is time-varying the number of nodes in a stage will vary too, such that, at a time instant k there are $\kappa_i(k)$ nodes in stage i. The signal $x_{ij}(n)$ flows in node η_{ij} and has a sampling rate which is slower than the original input signal $x(n)$, and, clearly, $x_{00}(n) = x(n)$. The signal in node η_{ij} is processed by an LT matrix $\mathbf{P}_{ij}(k)$, which represents a $M_{ij}(k)$-channels PUFB. In order to label the nodes in a stage we increasingly order them according to their respective frequency slots occupied by the corresponding filter's frequency response as shown in Fig. 4.5. Note that node ordering in a stage is not so easily found from the tree structure. This is because [18] the decimation of high-pass or band-pass filtered subbands (assume ideal filters, for instance) may lead to a signal whose spectrum is a mirror of the in-band spectrum of the subband before decimation.

If η_{ij} is a leaf, we can say that $\mathbf{P}_{ij}(k) = \mathbf{I}_{M_{ij}(k)}$ and that the signal in a leaf (IVEN) is processed by a one channel filter bank, which is the bypass LT. In such case, we can extend the paths of the tree so that the leaf node is a parent of only one other node. This child node, of course, carries the same signal as its parent and we can extend this false parenthood notation for as many stages as necessary. Although the tree ends where

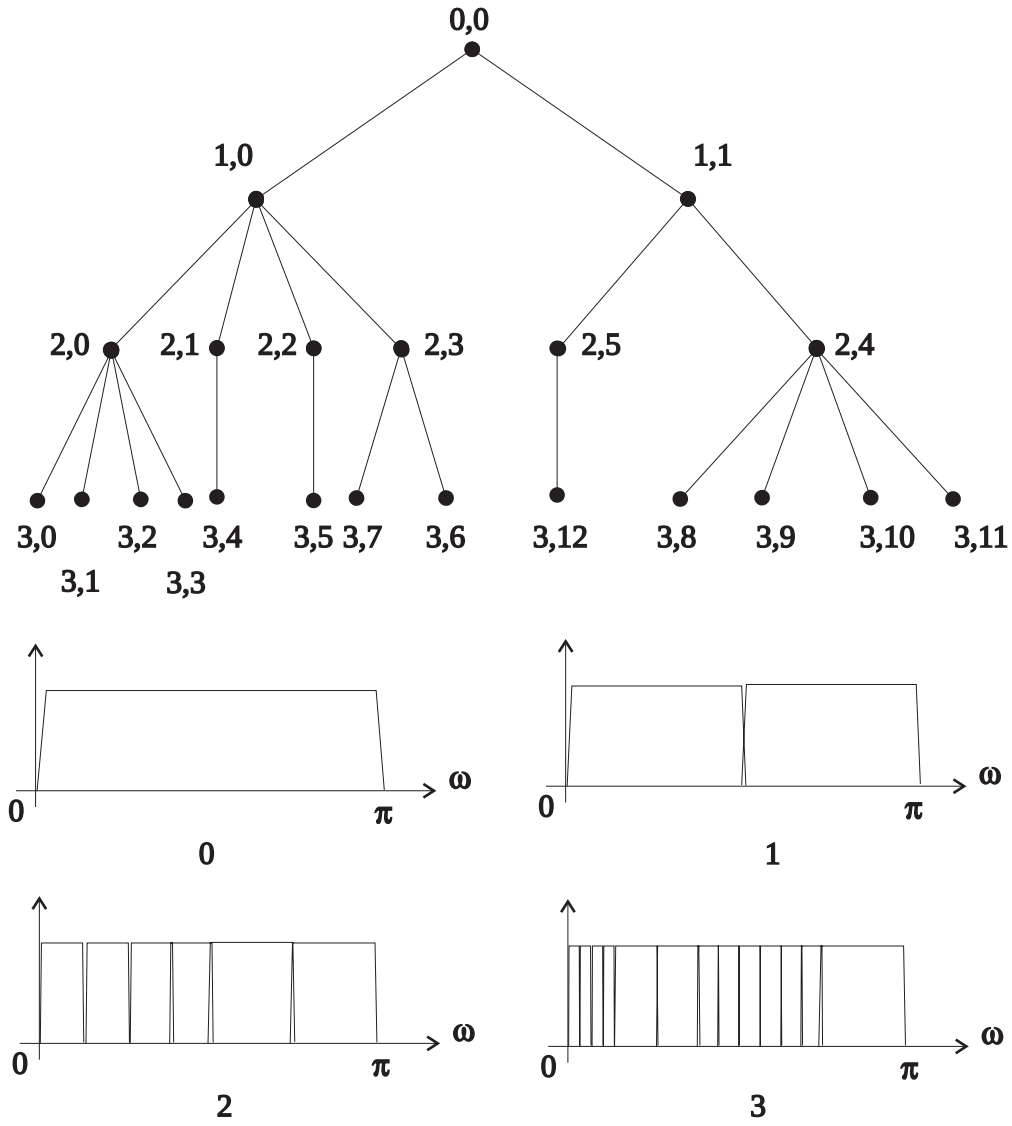


Figure 4.5: Node labelling example for a DWP. The nodes are labelled η_{ij} where i is the stage number and j a unique number inside a stage. Assuming unit-gain filters, the corresponding frequency response of each resulting filter at each stage is also shown. These frequency responses show the spectral partition provided by each stage of the DWP as we go from the root to the leaves (from stage 0 to stage 3).

an IVEN is found, we can make all terminal branches of the tree to lie in the same stage number. Hence, we define S_{max} as an absolute maximum stage number (depth of the tree). The equivalent filters and spectrum partitions resulting from any tree shape can be found by inspecting the nodes in the stage S_{max}, and the shape of the tree is found by the set of $M_{ij}(k)$. As a remark, the sampling rate ratio between the original signal and the j-th subband in the stage S_{max}, at instant k, is the ratio between π and the bandwidth of the j-th subband.

4.7.2 M-ary homogeneous trees

The M-ary tree is a simplification of the general tree where, except for the leaves, $M_{ij}(k) = M$ (a fixed number) for all i, j and k. In other words, each node is either an IVEN or is parent of exactly M other nodes. In this way, the shape of the tree can change, but not the number of branches connected to each node. For a homogeneous tree we assume that $\mathbf{P}_{ij}(k) = \mathbf{P}$, i.e., a fixed LT for all nodes at all times. Also, it is simple to see that η_{ij} is the parent node of nodes $\eta_{i,jM}$ through $\eta_{i,jM+M-1}$ and is also the child node of $\eta_{i-1,j\oslash M}$, where $\oslash$ means integer division. Therefore, we do not have to keep track of $M_{ij}(k)$, and, instead, we can assume a maximum stage number S_{max} and define the concept of activity of a node. A node is said to be active if its signal is processed by an LT, and is said inactive otherwise. Obviously, an IVEN is inactive. Let us denote by $a_{ij}(n)$ the activity of node η_{ij} and $a_{ij}(n) = 1$ if the node is active at instant n and $a_{ij}(n) = 0$, otherwise. So, the tree shape is defined by the activity of its nodes (the map of all $a_{ij}(n)$, up to stage S_{max}) and this is the DWP we want to implement. We

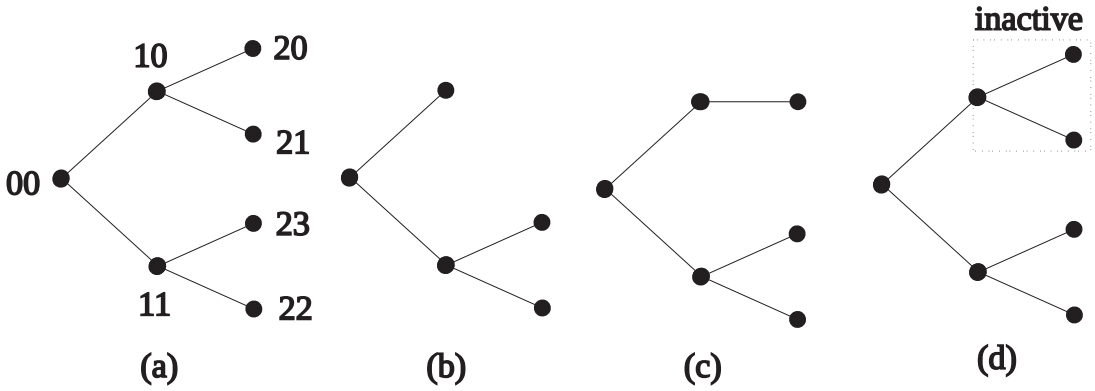


Figure 4.6: A 2-stages binary-tree example. (a) Full-tree with its nodes labelled. All nodes have $M_{ij} = 2$. (b) The tree we want to implement. (c) The tree with leaf nodes extended to the last stage, in such a way that for the extended node $M_{ij} = 1$. (d) Equivalent notation for homogeneous binary tree, by deactivating one LT, as indicated. When a node is deactivated, the LT is turned to its bypass state. When the transition to bypass is over, one can extract all samples directly from node η_{10}.

can always think as if the full-tree is the LT system that is always implemented, while we pick the signal at the desired nodes (IVENs). This different notation is shown in an example in Fig. 4.6.

4.8 Tiling the time-frequency plane

As we discussed in Section 2.6, the TF diagram has an intimate connection with the tree-diagram and both represent a DWP. The time-invariant DWP allows partition of the TF plane in a very versatile fashion. However it can be much improved with the use of a time-varying DWP. If we can shape the tree in any way along the time index and even change the number of channels in each tree-node, we can approach a virtually arbitrary maximally-decimated partition of the TF plane, although the tiling is always

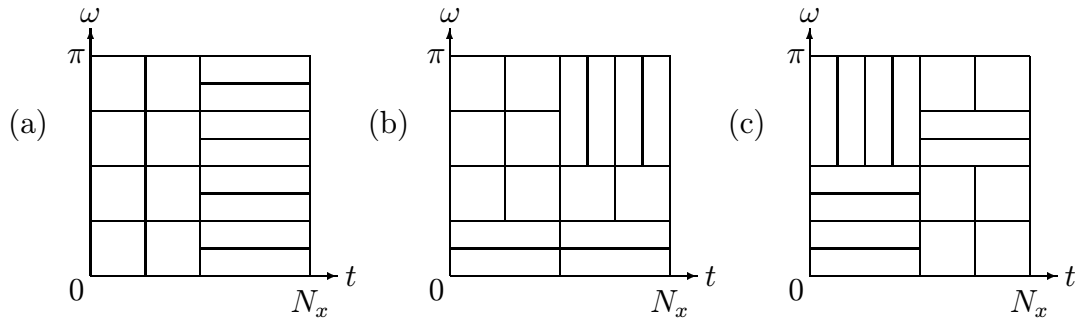


Figure 4.7: Examples of TF diagrams of time-varying DWP. (a) A change in the number of channels. (b) One of the binary tree branches is pruned. (c) A more complex change in the shape of the tree.

rectangular. In practical cases, the number of allowed combinations is so large that virtually all TF diagrams (tiling of the TF plane) of interest may be represented by a general time-varying DWP.

For example, based on the TF diagrams in examples in Figs. 2.7 and 2.9 we can construct the TF diagrams shown in Fig. 4.7, which are associated with time-varying DWP, where tree-branches are pruned and expanded or the number of channels is changed.

4.9 Adaptive wavelet packets

The best time-frequency representation of a signal, or the best wavelet packet, is based on abstract ideas. Almost all representations and tiling of the time-frequency plane may have their utility. In [90] it was developed a method to find the best wavelet packet based on a rate-distortion criterion. In another work [29], [28], this criterion was applied to search for an adaptive wavelet packet which would track the best tree shape. The method populates each node of the tree with a Lagrangian cost function

$J_{ij}(\alpha) = D_{ij} + \alpha R_{ij}$, where D_{ij} and R_{ij} are the distortion and bit-rate associated with node η_{ij}, respectively, and α is a Lagrange multiplier. Then, an algorithm is used to find the minimum cost for all possible set of terminal nodes, given a quality factor α [90], [29], [28]. For variation in time, a "double-tree" algorithm was employed in [28]. This algorithm was also applied to code speech signals, and it can be applied to our case, since we are providing the means to adapt the tree and not the cost function for the adaptation.

On the other hand, we can also use an ad-hoc simplifying strategy, which is solely based on intuition and concepts. This can simplify the adaptation procedure and even allow backward adaptation. Assume, for while, an homogeneous M-ary tree. For this we need to formulate a cost function allowing all nodes to be independent. In such a way the adaptation algorithm is divided into two acting units.

Node unit - Present in every node of the full tree, it evaluates the signal flowing in the node and decides if such node may be active or not.

Managing unit - Central unit that collects information (suggestions) about activities of all node units and decides which nodes should be made active or not based on their hierarchies in the tree.

4.9.1 Energy compaction algorithm for the node units

If a stationary model is assumed, for minimum mean square error, the greater energy compaction in fewer coefficients [33] results in less distortion for a given bit-rate. This will generally lead us to choose the full tree which has better frequency

resolution [73]. However, as a transform is bypassed, the filter for the resulting subband is shortened. Therefore a better temporal localization is attainable. Furthermore, if we use shorter filters, the distortion in a coefficient in a particular subband would spread along a smaller region than if the filters were longer. Therefore, we can seek the maximum time resolution whenever not much energy compaction is provided by the transform.

Let us focus our attention in one node η_{ij}, as the node units are assumed to be independent. As the tree is M-ary, let $\chi(m) = x_{ij}(m)$ and $\chi_k(n) = x_{i+1,jM+k}(n)$ for $0 \leq k \leq M-1$. For this node, $a(n)$ is the activity signal. Let

$$\sigma_k^2 = E\left[\chi_k^2(n)\right] \tag{4.34}$$

The above variance is related to the variance of $\chi(n)$ (σ_χ^2). If the LT is orthonormal, its filters have gain $\sqrt{M}$ and obey the power complementary property, i.e.,

$$\sum_{k=0}^{M-1} \sigma_k^2 = M\sigma_\chi^2 \tag{4.35}$$

The signal is not assumed stationary and the variances can be estimated continuously. Further computations over $\chi(n)$ would lead to more complexity and we can work directly with the decomposed signal. Then, a windowed estimation of the variance using a filter with impulse response $h(n)$ results in

$$\hat{\sigma}_k^2(n) = \chi_k^2(n) * h(n) \tag{4.36}$$

Using the estimated variances for energy compaction computations, we have a local estimation of the transform coding gain [33] as

$$G(n) = \left(\frac{\frac{1}{M} \sum_{i=0}^{M-1} \hat{\sigma}_i^2}{\left(\prod_{i=0}^{M-1} \hat{\sigma}_i^2 \right)^{1/M}} \right). \tag{4.37}$$

and we can compare $G(n)$ to a threshold g in order to decide if we set $a(n) = 1$ or not. Hence,

$$a(n) = u[G(n) - g] \tag{4.38}$$

where $u(x)$ is the unit step function. For each node we can append the subscript ij for its proper identification in $a(n)$, $G(n)$, and g.

4.9.2 Tree-collapsing algorithm

To determine whether all nodes are to be made active or not we can start from the maximum available frequency resolution, i.e., check nodes in a maximum stage S_{max} and, then, their parents. At each node, we can evaluate $a_{ij}(n)$ as in (4.38). If a node is made active, all nodes connecting it to the root are also made active, in order to make the algorithm consistent. When all nodes in a stage are active, the algorithm is stopped.

Start: $a_{ij} = 0$ (all ij)
$s = S_{max}$
repeat
for $m = 0...M^s - 1$
if $a_{sm} = 0$
evaluate a_{sm}
if $a_{sm} = 1$

$k = m \oslash M$
for $r = 1...s$
$a_{s-r,k} = 1$
$k = k \oslash M$
end
end
end
end
if ($a_{sm} = 1$, $m = 0, ..., M^s - 1$)
then: stop
else: $s = s - 1$
end
until $s = 0$

4.9.3 Tree-construction algorithm

The tree-construction algorithm is similar to the tree-collapsing one, but, instead of starting from a stage S_{max}, we start from the root and proceed towards the leaves checking the activity of the nodes. In this way, the algorithm can be simplified as

- Start the adaptive process with any tree configuration.
- Evaluate activity in all parent nodes of IVENs to see if they can be made inactive (prune the tree). In this case, these nodes become IVENs in the next iteration.
- Check all IVENs to see if they can become active (expand the tree). In this case the current IVENs become parent nodes of M IVENs in the next iteration.

Note that activity may be checked only on IVENs and on their parents, while all other nodes do not have such procedure. Thus, the managing unit just prunes or expands

branches of the tree. Alternatively one can try to extend the search, by searching IVENS and $n-1$ levels of nodes with childhood relation to an IVEN (for expanding the tree), and n levels of parenthood of an IVEN (for tree pruning).

4.9.4 Adaptation

In order to prune or expand the same branches in analysis (transmitter) and synthesis (receiver), in an adaptive DWP, it is necessary to reconstruct the activity map (the set of all $a_{ij}(n)$) at the receiver. Adaptation can be either forward (the activity map is sent as side information) or backward (the activity map is deducted by the receiver from past reconstructed samples).

Backward adaptation

In (4.37) and (4.38) we use LT domain samples. If $G(n)$ in (4.37) uses quantized samples $\hat{\chi}_k(n)$ to estimate the variances and if, for a node η_{ij}, we use

$$a_{ij}(n+1) = u[G_{ij}(n) - g_{ij}], \tag{4.39}$$

then, the receiver can recover $a_{ij}(n)$ without transmission of side-information, because it has available the past quantized transform- or time-domain samples and g_{ij} can be a fixed threshold. Both receiver and transmitter have to be synchronized, such that the transmitter has to use quantized values also when the transform is bypassed. Furthermore, the same nodal interrelation algorithm has to be used, always preparing future values of $a_{ij}(n+1)$. The filter $h(n)$ has to be causal and a simple first or second order

IIR filter can be adequate, having a narrow low-pass bandwidth to avoid frequent transitions. Whenever a transition occurs, the states of the IIR filter may be reset, interrupting filtering until the transition is over. After that, filtering is resumed. This is because at a transition, the receiver will not have transform-domain samples or time-domain ones. Therefore, it will not be able to perform filtering, in the windowed estimation of variances, unless the time-domain samples are recovered and transformed again. Setting the filters to avoid frequent changes may help in this case.

Forward adaptation

In case the activity map with all $a_{ij}(n)$ is sent in parallel, then things get much easier. First, there is no need to calculate activity on the receiver side. Second, one can use any means to determine activity of the nodes, including non-causal filters. A non-causal $h(n)$ is naturally preferred since it will avoid very short changes. The binary signal $a_{ij}(n)$ can be processed to avoid short bursts and locally oscillatory behaviors. A possible solution is a recursive median filter, which, in the binary case, is easily computed using tables. The formula for this is:

$$a_{ij}(n) = \text{round}\left[\text{mean}\left(a_{ij}(n-k)\ldots a_{ij}(n+k)\right)\right] \tag{4.40}$$

This would prevent bursts of up to k isolated values of a_{ij} and would not oscillate if the input is an alternation of 0's and 1's. When an oscillation is encountered, the state just before it is preserved. The order in which the node activity is evaluated may be found using the algorithm described earlier.

The disadvantage is due to the transmission of side information, since all the activity map has to be transmitted. Assuming 1 bit per sample and a maximum stage number S_{max}, it would require S_{max}/M bits per sample as overhead. (Remember that if node η_{00} works at a sampling rate f_s, node η_{ij} works in a sampling rate $M^{-i}f_s$.) However, assuming the filters would prevent very frequent changes, run-length coding can be applied to largely compress this map. Furthermore, as a node is active, all nodes connecting it to the root node will also be active. Therefore, information for them is not necessary.

Tests

Tests were made, using the energy compaction, tree-collapsing and forward adaptation algorithms, to code a segment of 8192 samples of a speech signal shown in Fig. 4.8, based on the ELT-2 ($M = 2$) and $S_{max} = 6$. The transformed samples were coded using a uniform quantizer whose step size was varied. The entropy of the quantizer output plus tree-information, was evaluated as a measure of the rate obtained. The measure of distortion was

$$D = \left(\frac{1}{8192} \sum_{n=0}^{8191} (x(n) - \hat{x}(n))^2 \right)^{1/2},$$

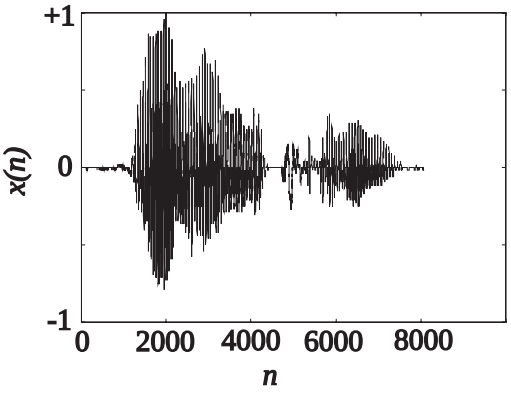
where $\hat{x}(n)$ is the reconstructed signal after synthesis. The plots of distortion versus entropy (DH) are shown in Fig. 4.8 for several threshold values g. Once the threshold g is chosen (the same value g was applied to all nodes), the quantizer step is varied to obtain curves in the DH plane. The curves are not shown (only the points are shown), because we have tried several threshold values between $g = 2.5$ and $g = 4.0$ for several step sizes and the plot of each curve would be confusing. On solid line the same results

were obtained with the use of a regular wavelet transform. We can see the concentration of points below the solid line indicating the best performance of the adaptive scheme.

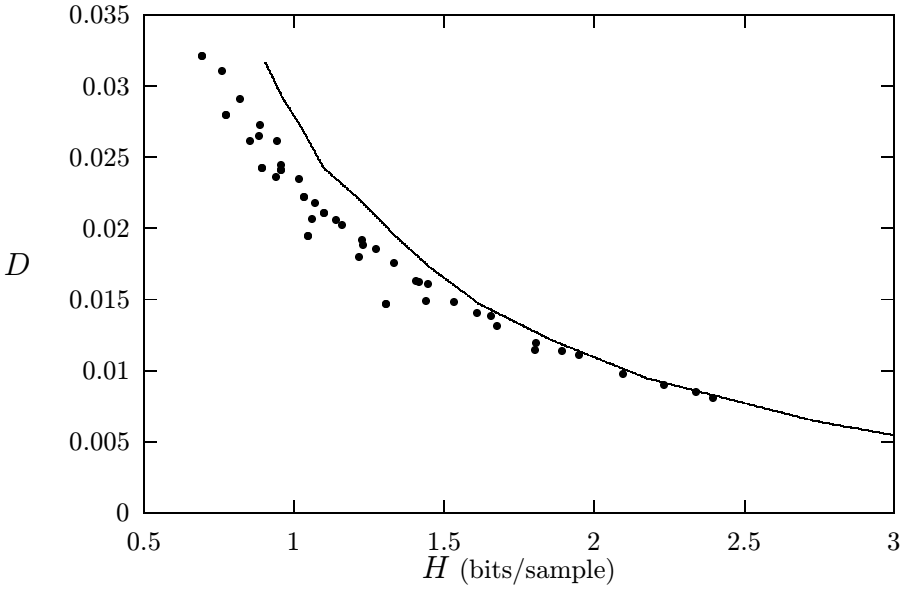
4.9.5 Non-M-ary trees

We can accommodate the tree-construction algorithm for non-M-ary trees and, yet, elaborate an algorithm which is not unbearably complex to implement. However, we assume that $M_{ij}(n) \in \{m_0, \ldots, m_{n_M}\}$, corresponding to a set of $n_M + 1$ admissible values for the LT's number of channels, and where $m_0 = 1$ (inactive node). Of course, if $n_M = 1$, we have the particular case of an M-ary tree.

We may change the node units to compare several values of M and decide how many channels the particular node may have. For this, the node unit may start with $M_{ij}(n) = m_{n_M}$ and check if $G_{ij} > g_{ij}$. If not, it may repeat the test with $M_{ij}(n) = m_{n_M-1}$ and, in case $G_{ij} > g_{ij}$ still is not satisfied, recur the test with the set of admissible values until reaching $M_{ij}(n) = m_0$, in which case the particular node is bypassed. The managing unit may behave as for the case of the M-ary tree, i.e., only the IVENs and their parents are checked for pruning or expanding branches. Backward or forward adaptation are perfectly possible, as well, although the side-information (for forward adaptation) may contain the map of all $M_{ij}(n)$, where each permissible value belongs to the enumerable set of possible number of channels, as mentioned above.



(a)



(b)

Figure 4.8: Distortion (D) versus Entropy (H), in bits-per-sample, plot of the results simulating the adaptive wavelet packets based on the ELT with $K = 2$ and $M = 2$. a) Sample speech segment. b) $D \times H$ plots using several values of threshold and quantizer steps, where the solid line shows the results for the DWT, using the same filter bank.

CHAPTER 5

PROCESSING FINITE-LENGTH SIGNALS

When designing an LT, the signal is assumed to have an infinite number of samples and stationary statistics, so that the design of the LT matrix $\mathbf{P}$ to fit in a given statistical model suffices for its application. With these assumptions, the choice of $\mathbf{P}$ is the single goal in the design of the analysis-synthesis system. However, most of the signals encountered have finite length either because of their nature, or because they are previously segmented to deal with time-varying statistics. As we discussed in the previous chapter, a time-varying DWP can be implemented by segmenting the signal and applying LTs to finite-length segments, for each node of the tree. Thus, the need for efficient approaches to implement LTs over finite-length signals is evident. In this case, two approaches will be discussed in this chapter. First, time-invariant LTs following SDF are analyzed, in such a way that a single LT is invariantly applied to the signal. This is accomplished by artificially extending the signal into a periodic sequence, which is processed and, then, segmented again [5], [8], [9], [11], [36], [37], [53], [54], [64], [65], [77], [82], [86], [98]. Second, we discuss the use of time-varying filter banks applied to the boundary regions of the signal in order to assure full orthogonality of the size-limited transform $\mathbf{T}$ [25], [28]–[31], [49], [69], [86].

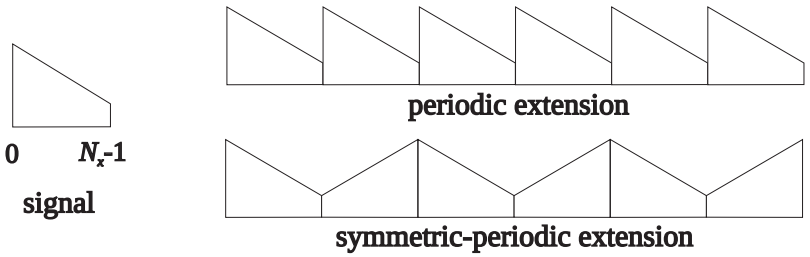


Figure 5.1: (a) Illustration of input signal of N_x samples, and its periodic and symmetric-periodic extensions.

5.1 Periodic extensions

When processing a finite-length signal, of N_x samples, one can create an infinite-length sequence by periodically extending the original sequence [5], [11], [37], [53], [54], [64], [65], [86], [98]. Illustrations of two popular extension methods are shown in Fig. 5.1. If, starting from signal $x(n)$, of N_x samples, contained in vector $\mathbf{x}$, we can create an infinite periodic signal (see Fig. 5.1) $\mathbf{x}_\infty$ as

$$\mathbf{x}_\infty^T = [\cdots, \mathbf{x}^T, \mathbf{x}^T, \mathbf{x}^T, \mathbf{x}^T, \mathbf{x}^T, \cdots], \tag{5.1}$$

Then, the analysis transform is given by

$$\mathbf{y}_\infty = \mathbf{H}\mathbf{x}_\infty \tag{5.2}$$

and, we can see that $\mathbf{y}_\infty$ is also periodic given by

$$\mathbf{y}_\infty^T = [\cdots, \mathbf{y}^T, \mathbf{y}^T, \mathbf{y}^T, \mathbf{y}^T, \mathbf{y}^T, \cdots], \tag{5.3}$$

Hence the equivalent transform $\mathbf{T}$, such that $\mathbf{y} = \mathbf{T}\mathbf{x}$, is block circulant and orthogonal [110]. The orthogonality of $\mathbf{T}$ can be easily inferred by checking (2.39) and (2.40). As an example, for $N = 3$ and $N_B = 5$, we have

$$\mathbf{T} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M & \mathbf{0}_M & \mathbf{P}_0 \\ \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 & \mathbf{0}_M \\ \mathbf{0}_M & \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 & \mathbf{P}_2 \\ \mathbf{P}_2 & \mathbf{0}_M & \mathbf{0}_M & \mathbf{P}_0 & \mathbf{P}_1 \end{bmatrix} \tag{5.4}$$

The periodic extension has important properties. All processing can be done efficiently with the aid of the DFT by implying circular convolution [66], [67]. However, this may just be more efficient than direct time-domain implementation of the convolution if the filters' length L is large. Furthermore, the assumption that the signal is periodic, also implies that, in the extended signal, samples in both extremes of $\mathbf{x}$ are adjacent, and this may introduce artificial high-frequency components in the subbands because of possible discontinuities, across the signal boundaries. One way to resolve this problem may be by using another signal extension. Assume $N < N_B$ and let

$$\mathbf{x}_e = \mathbf{E}_T \mathbf{x}, \tag{5.5}$$

where $\mathbf{E}_T$ is a square matrix used to find a signal extension ($\mathbf{x}_e$) based on the existing signal $\mathbf{x}$. Thus, we construct an infinite-length periodic signal given by

$$\mathbf{x}_\infty^T = [\cdots, \mathbf{x}^T, \mathbf{x}_e^T, \mathbf{x}^T, \mathbf{x}_e^T, \mathbf{x}^T, \mathbf{x}_e^T, \cdots]. \tag{5.6}$$

$\mathbf{x}_\infty$ is periodic and is processed by $\mathbf{H}$, generating an infinite-length vector as $\mathbf{y}_\infty = \mathbf{H}\mathbf{x}_\infty$. Note that still there is a size-limited transform $\mathbf{T}$ such that $\mathbf{y} = \mathbf{T}\mathbf{x}$. Since both $\mathbf{H}$ and $\mathbf{x}_\infty$ have a periodic structure, then $\mathbf{y}_\infty$ is also periodic as

$$\mathbf{y}_\infty^T = [\cdots, \mathbf{y}^T, \mathbf{y}_e^T, \mathbf{y}^T, \mathbf{y}_e^T, \mathbf{y}^T, \mathbf{y}_e^T, \cdots], \tag{5.7}$$

where $\mathbf{y} = \mathbf{T}\mathbf{x}$ and the relation among $\mathbf{y}$, $\mathbf{y}_e$, and $\mathbf{E}_T$ will be discussed later. However, as only linear relations are involved we can expect to find a matrix $\mathbf{E}_S$ such that $\mathbf{y}_e = \mathbf{E}_S\mathbf{y}$.

Note that the simple periodic extension is the one where $\mathbf{E}_T = \mathbf{I}_{N_x}$. A popular extension method is the so-called symmetric-periodic extension (see Fig. 5.1), or symmetric extension [5], [8], [9], [11], [36], [37], [53], [54], [64], [65], [77], [82], [86], [98], where $\mathbf{E}_T = \mathbf{J}_{N_x}$. This extension can avoid the artificial discontinuities and can also be implemented with the aid of fast transforms, as we will see later.

5.1.1 Extended signals and transforms

Let the period of $\mathbf{x}_\infty$ and of $\mathbf{y}_\infty$ be represented by $\mathbf{x}_p^T = [\mathbf{x}^T, (\mathbf{E}_T\mathbf{x})^T] = [\mathbf{x}^T, \mathbf{x}_e^T]$ and $\mathbf{y}_p^T = [\mathbf{y}^T, (\mathbf{E}_S\mathbf{y})^T] = [\mathbf{y}^T, \mathbf{y}_e^T]$, respectively. Let the transform for the signal in one period be $\mathbf{T}_p$, where

$$\mathbf{y}_p = \mathbf{T}_p\mathbf{x}_p. \tag{5.8}$$

Then, we have

$$\mathbf{x}_\infty^T = [\cdots, \mathbf{x}_p^T, \mathbf{x}_p^T, \mathbf{x}_p^T, \mathbf{x}_p^T, \cdots], \tag{5.9}$$

$$\mathbf{y}_\infty^T = [\cdots, \mathbf{y}_p^T, \mathbf{y}_p^T, \mathbf{y}_p^T, \mathbf{y}_p^T, \cdots]. \tag{5.10}$$

The signal $\mathbf{x}_p$ is the period of $\mathbf{x}_\infty$, therefore, $\mathbf{T}_p$ is a block-circulant orthogonal matrix of size $2N_x \times 2N_x$, as we previously discussed, which can be also implemented

in the DFT domain using circular convolution. However, we are not interested in $\mathbf{x}_p$ or $\mathbf{y}_p$, but in $\mathbf{x}$ and $\mathbf{y}$. Let us further explore the properties of the periodic transforms and signals involved. $\mathbf{T}_p$ has a composition similar to the example described in (5.4) and note that $\mathbf{T}_p$ can be divided into four $N_x \times N_x$ square submatrices as

$$\mathbf{T}_p = \begin{bmatrix} \mathbf{T}_0 & \mathbf{T}_1 \\ \\ \mathbf{T}_1 & \mathbf{T}_0 \end{bmatrix}. \tag{5.11}$$

Then, using (5.5), (5.8), and (5.11), we get

$$\mathbf{y} = \mathbf{T}_0\mathbf{x} + \mathbf{T}_1\mathbf{E}_T\mathbf{x} = (\mathbf{T}_0 + \mathbf{T}_1\mathbf{E}_T)\mathbf{x}, \tag{5.12}$$

yielding the relation

$$\mathbf{T} = \mathbf{T}_0 + \mathbf{T}_1\mathbf{E}_T. \tag{5.13}$$

Proposition 5.1 *The size limited transform* $\mathbf{T}$ *in (5.13) is one-to-one and onto, therefore, invertible, for every real extension matrix* $\mathbf{E}_T$ *or choice of LT* $\mathbf{P}$*.*

To evaluate if $\mathbf{T}$ is invertible it is sufficient to show that $\mathbf{T}$ has full-rank, regardless of the linear extension, for all $\mathbf{P}$. $\mathbf{T}_p$ is an orthogonal matrix, therefore it is invertible and is full rank (rank $2N_x$). Since $\mathbf{x}$ can be any vector in $\Re^{N_x}$, all possible combinations of elements of $\mathbf{x}_{per}$ span a subspace of $\Re^{2N_x}$, of dimension N_x. As $\mathbf{T}_{per}$ is orthogonal, rank$\{[\mathbf{T}_0, \mathbf{T}_1]\} = N_x$ (full rank), and all possible linear combinations of elements of $\mathbf{y}$ will span $\Re^{N_S}$ (as is the case for $\mathbf{x}$). Therefore, $\mathbf{T}$ has full rank and is one-to-one and onto. Thus, as $\mathbf{T}$ is full rank, it is also non-singular, and its inverse exists and is unique, concluding the demonstration.

Proposition 5.2 *The subband vector extension matrix,* $\mathbf{E}_S$*, does exist and is given by* $\mathbf{E}_S = (\mathbf{T}_1 + \mathbf{T}_0\mathbf{E}_T)(\mathbf{T}_0 + \mathbf{T}_1\mathbf{E}_T)^{-1}$.

Since $\mathbf{T}$ is invertible, $\mathbf{x} = \mathbf{T}^{-1}\mathbf{y}$. Since $\mathbf{y}_e = \mathbf{T}_1\mathbf{x} + \mathbf{T}_0\mathbf{E}_T\mathbf{x} = (\mathbf{T}_1 + \mathbf{T}_0\mathbf{E}_T)\mathbf{x}$, we have $\mathbf{y}_e = (\mathbf{T}_1 + \mathbf{T}_0\mathbf{E}_T)\mathbf{T}^{-1}\mathbf{y}$. Hence

$$\mathbf{E}_S = (\mathbf{T}_1 + \mathbf{T}_0\mathbf{E}_T)(\mathbf{T}_0 + \mathbf{T}_1\mathbf{E}_T)^{-1}, \tag{5.14}$$

which always exists (because $\mathbf{T}$ is invertible) and is only a function of the extension in time-domain ($\mathbf{E}_T$) and of $\mathbf{P}$ (which defines $\mathbf{T}_0$ and $\mathbf{T}_1$).

For an arbitrary linear extension matrix $\mathbf{E}_T$ we can implement the analysis and the synthesis sections with the aid of the DFT. However, as both $\mathbf{E}_T$ and $\mathbf{E}_S$ can have a very complex structure, perhaps being very difficult to invert, pre- and post-processing can be of excessive complexity, eliminating the benefits of the use of the DFT for the implementation of the LT. Since,

$$\mathbf{y}_p = \begin{bmatrix} \mathbf{y} \\ \mathbf{E}_S\mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{N_x} \\ \mathbf{E}_S \end{bmatrix} \mathbf{y}, \tag{5.15}$$

$$\mathbf{x}_p = \begin{bmatrix} \mathbf{x} \\ \mathbf{E}_T\mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{N_x} \\ \mathbf{E}_T \end{bmatrix} \mathbf{x}, \tag{5.16}$$

the analysis section can be expressed as

$$\mathbf{y} = \begin{bmatrix} \mathbf{I}_{N_x} \\ \mathbf{E}_S \end{bmatrix}^{+} \mathbf{T}_p \begin{bmatrix} \mathbf{I}_{N_x} \\ \mathbf{E}_T \end{bmatrix} \mathbf{x} \tag{5.17}$$

while the synthesis section can be expressed as

$$\mathbf{x} = \begin{bmatrix} \mathbf{I}_{N_x} \\ \mathbf{E}_T \end{bmatrix}^{+} \mathbf{T}_p^T \begin{bmatrix} \mathbf{I}_{N_x} \\ \mathbf{E}_S \end{bmatrix} \mathbf{y}, \tag{5.18}$$

where $[\ ]^+$ stands for the pseudo-inverse of a matrix. If $\mathbf{A}$ is a $n \times m$ matrix ($n \geq m$) with rank m, then $\mathbf{A}^+ = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$. The pseudo-inverse matrix always exists because $\mathbf{I}_{N_x}$ ensures that the matrix in question has full rank. Thus, the above expressions prove the following proposition:

Proposition 5.3 *With periodic extensions, the analysis and synthesis can be performed in the DFT-domain regardless of the extension method.*

Filtering and subsampling of a $2N_x$ sequence is computed in the DFT domain for both analysis and synthesis, which are, then, preceded and followed by extra processing. From these results, it is evident that without simplifying the extension matrices, DFT-aided implementation is impractical.

5.2 General time-domain solution

As discussed in Section 2.5, the analysis operation would require the knowledge of $\lambda = (L - M)/2$ samples outside the range of $\mathbf{x}$, for each border. If we set these samples to zero or if we use circular convolution, undesirable high frequency components can be generated due to discontinuities at the borders. The extension of $\mathbf{x}$ into $\tilde{\mathbf{x}}$ will be restricted here to be a linear boundary extension where the unknown samples are found by a linear combination of samples contained in $\mathbf{x}$. We assume that, for each border, the

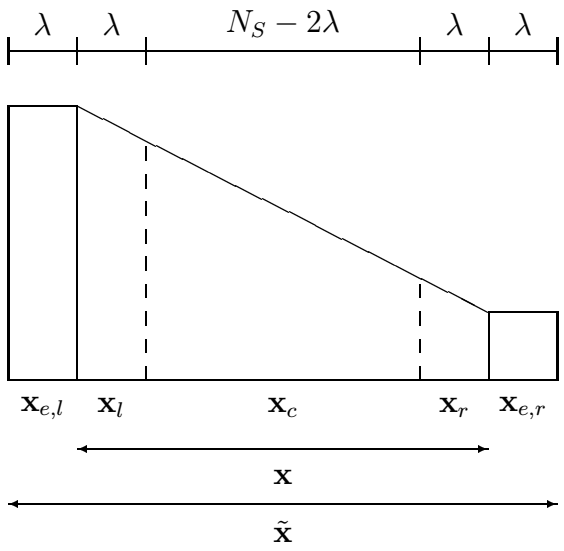


Figure 5.2: Illustration of signal extension of vector $\mathbf{x}$ into vector $\tilde{\mathbf{x}}$. In each border, $\lambda = (L - M)/2$ samples outside initial signal boundaries are found by linear relations applied to the λ boundary samples of $\mathbf{x}$, i.e., $\mathbf{x}_{e,l} = \mathbf{R}_l \mathbf{x}_l$ and $\mathbf{x}_{e,r} = \mathbf{R}_r \mathbf{x}_r$. As only λ samples are affected across the signal boundaries, it is not necessary to use the infinite-length extension. Also, $\mathbf{x}_l$ and $\mathbf{x}_r$ contain the samples possibly affected by the border distortions after synthesis.

λ samples across the border are found as a linear function of at most λ boundary samples of the signal. Note that this is a special case of the most general extension method of (5.5).

We adopt the notation shown in Fig. 5.2 [1] where $\mathbf{x}$ is divided into three regions as $\mathbf{x}^T = [\mathbf{x}_l^T, \mathbf{x}_c^T, \mathbf{x}_r^T]$, and the extended vector is formed by $\tilde{\mathbf{x}}^T = [\mathbf{x}_{e,l}^T, \mathbf{x}^T, \mathbf{x}_{e,r}^T]$. Then, we

[1] The expressions left and right are used to designate the extremes of the vector $\mathbf{x}$ just as if it is displaced horizontally. In this case, $x(0)$ is the leftmost sample, while $x(N_x - 1)$ is the rightmost one.

have $\tilde{\mathbf{x}}^T = [\mathbf{x}_{e,l}^T, \mathbf{x}_l^T, \mathbf{x}_c^T, \mathbf{x}_r^T, \mathbf{x}_{e,r}^T]$, where

$$\mathbf{x}_{e,l} = \mathbf{R}_l \mathbf{x}_l, \qquad \mathbf{x}_{e,r} = \mathbf{R}_r \mathbf{x}_r. \tag{5.19}$$

The size of each subvector is indicated in Fig. 5.2 and $\mathbf{R}_l$ and $\mathbf{R}_r$ are arbitrary $\lambda \times \lambda$ matrices to extend the signal on the left and right borders, respectively. For example, the symmetric extension is equivalent to $\mathbf{R}_l = \mathbf{R}_r = \mathbf{J}_\lambda$.

Proposition 5.4 *The original boundary samples can be recovered by simple linear post-processing operation applied to the distorted samples, after synthesis, as*

$$\mathbf{x}_r = \mathbf{\Xi}_r \begin{bmatrix} \hat{\mathbf{x}}_r \\ \\ \hat{\mathbf{x}}_{e,r} \end{bmatrix}, \tag{5.20}$$

$$\mathbf{x}_l = \mathbf{\Xi}_l \begin{bmatrix} \hat{\mathbf{x}}_{e,l} \\ \\ \hat{\mathbf{x}}_l \end{bmatrix}, \tag{5.21}$$

where $\mathbf{\Xi}_l$ and $\mathbf{\Xi}_r$ are $2\lambda \times \lambda$ real matrices. .

If we use the submatrices ($\mathbf{P}_i$) of $\mathbf{P}$, we can easily express $\mathbf{\Gamma}_L$ and $\mathbf{\Gamma}_R$, in (2.54), as

$$\mathbf{\Gamma}_L = \mathbf{\Delta}_L^T \, \mathbf{\Delta}_L, \tag{5.22}$$

$$\mathbf{\Gamma}_R = \mathbf{\Delta}_R^T \, \mathbf{\Delta}_R, \tag{5.23}$$

where $\mathbf{\Delta}_L$ and $\mathbf{\Delta}_R$ are found from

$$[\boldsymbol{\Delta}_L, \boldsymbol{\Delta}_R] = \left[\begin{array}{cccc|cccc} \mathbf{P}_0 & \mathbf{P}_1 & \cdots & \mathbf{P}_{N-2} & \mathbf{P}_{N-1} & & & 0 \\ & \mathbf{P}_0 & \cdots & \mathbf{P}_{N-3} & \mathbf{P}_{N-2} & \mathbf{P}_{N-1} & & \\ & & \ddots & \ddots & \ddots & & \ddots & \\ 0 & & & \mathbf{P}_0 & \mathbf{P}_1 & \cdots & & \mathbf{P}_{N-1} \end{array}\right]. \tag{5.24}$$

Dividing, $\boldsymbol{\Gamma}_L$ and $\boldsymbol{\Gamma}_R$, each into two $2\lambda \times \lambda$ matrices as $\boldsymbol{\Gamma}_L = [\boldsymbol{\Gamma}_{e,l}, \boldsymbol{\Gamma}_l]$ and $\boldsymbol{\Gamma}_R = [\boldsymbol{\Gamma}_r, \boldsymbol{\Gamma}_{e,r}]$, we have

$$\begin{bmatrix} \hat{\mathbf{x}}_{e,l} \\ \\ \hat{\mathbf{x}}_l \end{bmatrix} = [\boldsymbol{\Gamma}_{e,l}, \boldsymbol{\Gamma}_l] \begin{bmatrix} \mathbf{x}_{e,l} \\ \\ \mathbf{x}_l \end{bmatrix}, \tag{5.25}$$

$$\begin{bmatrix} \hat{\mathbf{x}}_r \\ \\ \hat{\mathbf{x}}_{e,r} \end{bmatrix} = [\boldsymbol{\Gamma}_r, \boldsymbol{\Gamma}_{e,r}] \begin{bmatrix} \mathbf{x}_r \\ \\ \mathbf{x}_{e,r} \end{bmatrix}. \tag{5.26}$$

Hence, from (5.19),

$$\begin{bmatrix} \hat{\mathbf{x}}_{e,l} \\ \\ \hat{\mathbf{x}}_l \end{bmatrix} = (\boldsymbol{\Gamma}_{e,l}\mathbf{R}_l + \boldsymbol{\Gamma}_l)\, \mathbf{x}_l \tag{5.27}$$

$$\begin{bmatrix} \hat{\mathbf{x}}_r \\ \\ \hat{\mathbf{x}}_{e,r} \end{bmatrix} = (\boldsymbol{\Gamma}_r + \boldsymbol{\Gamma}_{e,r}\mathbf{R}_r)\, \mathbf{x}_r. \tag{5.28}$$

and we can restore the boundary samples as

$$\mathbf{x}_l = (\boldsymbol{\Gamma}_{e,l}\mathbf{R}_l + \boldsymbol{\Gamma}_l)^+ \begin{bmatrix} \hat{\mathbf{x}}_{e,l} \\ \\ \hat{\mathbf{x}}_l \end{bmatrix}, \tag{5.29}$$

$$\mathbf{x}_r = (\mathbf{\Gamma}_r + \mathbf{\Gamma}_{e,r}\mathbf{R}_r)^+ \begin{bmatrix} \hat{\mathbf{x}}_r \\ \\ \hat{\mathbf{x}}_{e,r} \end{bmatrix}, \tag{5.30}$$

recalling that $[\,]^+$ stands for the pseudo-inverse of a matrix, and the pseudo-inverses are the $\mathbf{\Xi}$ matrices mentioned in Proposition 5.4. It is easy to see that if $\mathbf{T}$ has full rank (as we have shown), then the pseudo-inverses just discussed do exist. Thus, $\mathbf{x}_l$ and $\mathbf{x}_r$ are recovered from the distorted extended signal using only linear relations, in a method that is essentially based on post-processing the reconstructed signal. Note that the pseudo-inverse matrices are computed a priori and only one matrix multiplication is carried for the implementation.

5.3 Orthogonality in symmetric extensions

It is interesting to have $\mathbf{T}$ as a fully orthogonal transform. As we saw, if $\mathbf{E}_T = \mathbf{I}_{N_x}$, $\mathbf{T}$ will be orthogonal, but little can be said otherwise. Then, we narrow our search by assuming a symmetric extension.

Let $\mathbf{T}_{pre}$ and $\mathbf{T}_{pos}$ be $N_x \times N_x$ block diagonal matrices as $diag\{\mathbf{A}, \mathbf{A}, \ldots, \mathbf{A}\}$, where $\mathbf{A}$ is an orthogonal matrix whose size divides N_x. If $\mathbf{T}$ is a size-limited transform based on $\mathbf{P}$, it is clear that $\mathbf{T}' = \mathbf{T}_{pos}\mathbf{T}\mathbf{T}_{pre}$ is also an orthogonal transform generated by a different LT, found by pre- or post-processing the LT input or output with trivial block-transform operations. In this case, the extra processing is independent of the filter bank in question $\mathbf{P}$. This consideration is useful for the following proposition:

Proposition 5.5 *Except for pre- or post-processing, symmetric extensions will lead to an*

orthogonal size-limited transform $\mathbf{T}$ *if and only if the filters in the LT have linear-phase.*

Before presenting a proof, we would like to comment on the consequences of this result. First, we can always ensure PR and orthogonality using symmetric extensions for LTs with linear-phase filters. Second, non-linear-phase filters cannot achieve orthogonality using a symmetric extension (except by filters found by post-processing the output of a linear-phase LT with orthogonal matrices having non-symmetric basis functions) so that, for these filters, it is better to use directly the time-varying filter bank approach.

From $\mathbf{T}_p^T\mathbf{T}_p = \mathbf{T}_p\mathbf{T}_p^T = \mathbf{I}_{2N_x}$, and from (5.11), we obtain the following relations:

$$\begin{aligned}
\mathbf{T}_0\mathbf{T}_0^T + \mathbf{T}_1\mathbf{T}_1^T &= \mathbf{I}_{N_x}, \\
\mathbf{T}_0\mathbf{T}_1^T + \mathbf{T}_1\mathbf{T}_0^T &= \mathbf{0}_{N_x}, \\
\mathbf{T}_0^T\mathbf{T}_0 + \mathbf{T}_1^T\mathbf{T}_1 &= \mathbf{I}_{N_x}, \\
\mathbf{T}_0^T\mathbf{T}_1 + \mathbf{T}_1^T\mathbf{T}_0 &= \mathbf{0}_{N_x}.
\end{aligned} \tag{5.31}$$

Consider a linear-phase filter bank and define an $M \times M$ diagonal matrix $\tilde{\mathbf{V}}$ with elements $v_{kk} = 1$, if $f_k(m)$ is symmetric and $v_{kk} = -1$, if $f_k(m)$ is anti-symmetric. Then,

$$\mathbf{P} = \tilde{\mathbf{V}}\mathbf{P}\mathbf{J}_L. \tag{5.32}$$

Let $\mathbf{V}$ be an $N_x \times N_x$ matrix with non-zero block entries only in the counter-diagonal, as

$$\mathbf{V} = \begin{bmatrix} 0 & & & \tilde{\mathbf{V}} \\ & & \tilde{\mathbf{V}} & \\ & \tilde{\mathbf{V}} & & \\ \tilde{\mathbf{V}} & & & 0 \end{bmatrix}. \tag{5.33}$$

Using (5.32), it is easy to verify that

$$\begin{aligned} \mathbf{T}_0 &= \mathbf{V}\mathbf{T}_0\mathbf{J}_{N_x} \\ \mathbf{T}_1 &= \mathbf{V}\mathbf{T}_1\mathbf{J}_{N_x}. \end{aligned} \tag{5.34}$$

Substituting (5.34) in (5.31), and using the fact that $\mathbf{V}$ is orthogonal, we get

$$\begin{aligned} \mathbf{T}_0\mathbf{T}_0^T + \mathbf{T}_1\mathbf{T}_1^T &= \mathbf{I}_{N_x} \\ \mathbf{T}_0\mathbf{J}_{N_x}\mathbf{T}_1^T + \mathbf{T}_1\mathbf{J}_{N_x}\mathbf{T}_0^T &= \mathbf{0}_{N_x} \\ \mathbf{T}_0^T\mathbf{T}_0 + \mathbf{J}_{N_x}\mathbf{T}_1^T\mathbf{T}_1\mathbf{J}_{N_x} &= \mathbf{I}_{N_x} \\ \mathbf{T}_0^T\mathbf{T}_1\mathbf{J}_{N_x} + \mathbf{J}_{N_x}\mathbf{T}_1^T\mathbf{T}_0 &= \mathbf{0}_{N_x}. \end{aligned} \tag{5.35}$$

From (5.8) and (5.11), we have $\mathbf{y} = (\mathbf{T}_0 + \mathbf{T}_1\mathbf{J}_{N_x})\mathbf{x}$, so that $\mathbf{T} = \mathbf{T}_0 + \mathbf{T}_1\mathbf{J}_{N_x}$, and $\mathbf{T}$ is orthogonal because

$$\mathbf{T}\mathbf{T}^T = \mathbf{T}_0\mathbf{T}_0^T + \mathbf{T}_1\mathbf{T}_1^T + \mathbf{T}_0\mathbf{J}_{N_x}\mathbf{T}_1^T + \mathbf{T}_1\mathbf{J}_{N_x}\mathbf{T}_0^T = \mathbf{I}_{N_x} + \mathbf{0}_{N_x} = \mathbf{I}_{N_x},$$

$$\mathbf{T}^T\mathbf{T} = \mathbf{T}_0^T\mathbf{T}_0 + \mathbf{J}_{N_x}\mathbf{T}_1^T\mathbf{T}_1\mathbf{J}_{N_x} + \mathbf{T}_0^T\mathbf{T}_1\mathbf{J}_{N_x} + \mathbf{J}_{N_x}\mathbf{T}_1^T\mathbf{T}_0 = \mathbf{I}_{N_x} + \mathbf{0}_{N_x} = \mathbf{I}_{N_x}.$$

Hence the sufficiency is proved. To prove the necessity, note that given (5.31) we can reach (5.35) by algebraic manipulation if and only if we have

$$\begin{aligned} \mathbf{T}_0 &= \mathbf{\Phi}'\mathbf{T}_0\mathbf{J}_{N_x} \\ \mathbf{T}_1 &= \mathbf{\Phi}'\mathbf{T}_1\mathbf{J}_{N_x}. \end{aligned} \tag{5.36}$$

where $\mathbf{\Phi}'$ is a square orthogonal matrix. As $\mathbf{T}_0$ and $\mathbf{T}_1$ are block-circulant, presenting a periodic structure, the reader can check that (5.36) is only possible if $\mathbf{P}$ presents a structure such that

$$\mathbf{P} = \mathbf{\Phi}\mathbf{P}\mathbf{J}_L, \tag{5.37}$$

where $\mathbf{\Phi}$ is a square orthogonal matrix. In other words, the filters $f_k(n)$ would have to be found by a linear combination of their time-reversed versions (which are the filters $g_k(n)$). Using the fact that $\mathbf{P}\mathbf{P}^T = \mathbf{I}_M$ and after some manipulation, we can see that (5.37) is only true if $\mathbf{\Phi} = \mathbf{\Phi}^{-1}$ ($\mathbf{\Phi}$ is symmetric and orthogonal) and that $\mathbf{\Phi} = \mathbf{P}\mathbf{J}_L\mathbf{P}^T$.

For any matrix $\mathbf{\Phi}$, with such characteristics, there is an orthogonal matrix $\mathbf{A}$, such that $\mathbf{A}_{diag} = \mathbf{A}^T\mathbf{\Phi}\mathbf{A}$ [32], where $\mathbf{A}_{diag}$ is a diagonal matrix. As $\mathbf{A}$ and $\mathbf{\Phi}$ are orthogonal, $\mathbf{A}_{diag}$ is orthogonal, having elements ± 1 along the diagonal. Consider the LT given by $\mathbf{P}_{LP} = \mathbf{A}^T\mathbf{P}$. Since $\mathbf{A}$ is orthogonal, $\mathbf{P} = \mathbf{A}\mathbf{P}_{LP}$ and substituting in (5.37) we get $\mathbf{P}_{LP} = \mathbf{A}_{diag}\mathbf{P}_{LP}\mathbf{J}_L$, implying that $\mathbf{P}_{LP}\mathbf{J}_L\mathbf{P}_{LP}^T = \mathbf{A}_{diag}$. Therefore, if $\mathbf{P}$ is a solution to (5.37), $\mathbf{P}_{LP}$ is also a solution. Furthermore, $\mathbf{P}_{LP}$ corresponds to linear-phase filters because $\mathbf{P}_{LP} = \mathbf{A}_{diag}\mathbf{P}_{LP}\mathbf{J}_L$. So, every solution $\mathbf{P}$ to (5.37) can be written as $\mathbf{P} = \mathbf{A}\mathbf{P}_{LP}$, what means a post-processing of a linear-phase LT by a block-transform $\mathbf{A}$. This concludes the proof.

5.4 LTs with linear-phase filters

If we combine linear-phase filters with symmetric extensions, not only we can get a smoother transition across the signal boundaries, but also $\mathbf{T}$ will be orthogonal.

Proposition 5.6 *Using symmetric extension and linear-phase filters,* $\mathbf{E}_S = \mathbf{V}$.

The proof is straightforward, as

$$\begin{aligned} \mathbf{y}_e &= \mathbf{T}_1\mathbf{x} + \mathbf{T}_0\mathbf{J}_{N_x}\mathbf{x} = (\mathbf{V}\mathbf{T}_1\mathbf{J}_{N_x} + \mathbf{V}\mathbf{T}_0)\mathbf{x} \\ &= \mathbf{V}(\mathbf{T}_0 + \mathbf{T}_1\mathbf{J}_{N_x})\mathbf{x} = \mathbf{V}\mathbf{T}\mathbf{x} \\ &= \mathbf{V}\mathbf{y}. \end{aligned} \tag{5.38}$$

As a result, using symmetric extensions, $\mathbf{y}_e$ is easily found from $\mathbf{y}$ by sample mirroring (for each subband) followed by sign inversions.

5.4.1 Time-domain Implementation

In Fig. 2.6, we have a *clocked* system with memory where at each instant (block index) a block of M samples in time-domain is the input which is transformed into another block of M subband samples.

Based on the previous results, for the analysis, we extend the signal, through a mirror-image reflection applied to the last $\lambda = (L - M)/2$ samples on each border, resulting in a signal $\tilde{x}(n)$ with $N_x + 2\lambda = N_x + L - M$ samples, as

$$x(\lambda - 1), \cdots, x(0), x(0), \cdots, x(N_x - 1), x(N_x - 1), \cdots, x(N_x - \lambda)$$

The internal states in Fig. 2.6(a) can be initialized in any fashion and the signal is processed yielding N_B+N-1 blocks. We discard the first $N-1$ output blocks, obtaining N_B transform-domain blocks corresponding to N_B samples of each subband.

At the synthesis section, we have the subband signals $\hat{y}_k(m)$ composing the signal $\hat{y}(n)$ as $\hat{y}(mM+i) = \hat{y}_i(m)$ for $0 \leq i \leq M-1$. This signal $\hat{y}(n)$ is extended, by extending the subband signals by K samples in each border, as in (5.38), and processed as in Fig. 2.6(b). The k-th subband (initially having N_B samples) is extended as

$$v_{kk}\hat{y}_k(K-1), \cdots, v_{kk}\hat{y}_k(0), \hat{y}_k(0), \cdots \hat{y}_k(N_B-1), v_{kk}\hat{y}_k(N_B-1), \cdots, v_{kk}\hat{y}_k(N_B-K).$$

Then, we proceed with the synthesis over the N_B+2K blocks of $\hat{y}(n)$, obtaining a reconstructed signal with N_B+2K blocks $\tilde{\hat{x}}(n)$, initializing the states of Fig. 2.6(b) in any fashion. For N odd, $K=(N-1)/2$, we discard the first $N-1$ blocks to obtain $\hat{x}(n)$. For N even ($K=N/2$), we discard the first $N-1$ blocks, the first $M/2$ samples in the N-th block and the last $M/2$ samples of the signal.

In the absence of quantization/processing of the subbands ($\hat{x}(n) = x(n)$), this approach will assure the perfect reconstruction property and orthogonality of the analysis and synthesis processes, paying the price of running the algorithm over extra N or $N-1$ blocks, making it suitable for applications when $N_B \gg N$.

5.4.2 DFT-aided Implementation

In some applications, where N_B is comparable to N, it may be more convenient to implement the linear-phase LT with the aid of the DFT. For this, the filtering/subsampling, or the upsampling/filtering operations can be performed in the DFT

domain, as long as the signal is periodic. For the symmetric extension method, the periodic vectors are $\mathbf{x}_p$ and $\mathbf{y}_p$, and the transform $\mathbf{T}_p$ and its inverse $\mathbf{T}_p^T$ is the one implemented in the DFT-domain. Using (5.8) and (5.38), we have

$$\mathbf{y}_p = \begin{bmatrix} \mathbf{y} \\ \\ \mathbf{V}\mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{N_x} \\ \\ \mathbf{V} \end{bmatrix} \mathbf{y} \tag{5.39}$$

$$\mathbf{x}_p = \begin{bmatrix} \mathbf{x} \\ \\ \mathbf{J}_{N_x}\mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{N_x} \\ \\ \mathbf{J}_{N_x} \end{bmatrix} \mathbf{x} \tag{5.40}$$

Hence, the analysis section is expressed as

$$\mathbf{y} = \frac{1}{2} \begin{bmatrix} \mathbf{I}_{N_x} & \mathbf{V} \end{bmatrix} \mathbf{T}_p \begin{bmatrix} \mathbf{I}_{N_x} \\ \\ \mathbf{J}_{N_x} \end{bmatrix} \mathbf{x}, \tag{5.41}$$

while the synthesis section is expressed as

$$\mathbf{x} = \frac{1}{2} \begin{bmatrix} \mathbf{I}_{N_x} & \mathbf{J}_{N_x} \end{bmatrix} \mathbf{T}_p^T \begin{bmatrix} \mathbf{I}_{N_x} \\ \\ \mathbf{V} \end{bmatrix} \mathbf{y}. \tag{5.42}$$

We use the DFT of a symmetric real sequence of length $2N_x$, whose complexity can be reduced close to that of an N_x-samples DFT. Filtering and subsampling is computed in the DFT domain followed by an inverse DFT, to whose output we apply N_x additions. For the synthesis, the procedure is similar, where the subbands are extended in a symmetric way, and upsampling followed by filtering is performed in the DFT domain. As in the subsampling case, we apply N_x additions to the output of the inverse DFT.

5.5 Orthogonal boundary filter banks

Consider applying time-varying LTs over the finite-length signal as

$$\tilde{\mathbf{P}} = \begin{bmatrix} \mathbf{P}(0) & & & \\ & \mathbf{P}(1) & & \\ & & \ddots & \\ & & & \mathbf{P}(N_B-1) \end{bmatrix} \tag{5.43}$$

As we have defined earlier, let K be the greatest integer smaller than $N/2$ (the same as integer division, as $K = N/2$). Hence, there are K filter banks, at each border, which have their basis functions crossing the signal boundaries. We call this the minimal complete design (MCD) when only K filter banks at each border, are changed in order to achieve orthogonality of $\mathbf{T}$. We could change all N_B filter banks but only $2K$ of them have any influence on the borders, so that we will often assume an MCD. Then, we have

$$\mathbf{P}(m) = \mathbf{P} \quad \text{for } K \leq m \leq N_B - K - 1 \tag{5.44}$$

and the remaining filter banks are redesigned, but remaining instantaneously paraunitary, and obeying PR rules for time-varying filter banks.

As we have observed so far, using simple symmetric extensions, a linear-phase LT can achieve orthogonality and non-linear-phase LTs cannot do so (except for the special case previously discussed). Assuming a time varying LT approach and the MCD, only $\mathbf{P}(0)$ through $\mathbf{P}(K-1)$ and $\mathbf{P}(N_B-K)$ through $\mathbf{P}(N_B-1)$ should be changed.

Proposition 5.7 *If we denote the entries of* $\mathbf{P}(m)$ *as* $p_{ij}(m)$ *for* $0 \leq i \leq M-1$ *and* $0 \leq j \leq L-1$, *and denoting* $\lambda = (L-M)/2$, *then, in order to have* $\mathbf{T}$ *as an orthogonal*

matrix, $p_{ij}(m) = 0$ *for* $0 \leq i \leq M-1$ *and*

$$\{m, j | j \in [0, \lambda - mM - 1]; m \in [0, K-1]\}$$

$$\{m, j | j \in [L - \lambda + (N_B - 1 - m)M, L-1]; m \in [N_B - K, N_B - 1]\}$$

These two sets imply that $\tilde{\mathbf{P}}$ has zero entries for the first and last λ columns. To see this, consider an unlimited-length signal where a sequence of length N_x is to be transformed by $\mathbf{T}$ and imagine that the adjacent segments are also transformed by any other orthogonal transform, for example using the identity matrix as a transform matrix. So, assuming $\mathbf{T}$ is orthogonal,

$$\left[\begin{array}{c|c|c} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{A}_1 & \mathbf{T} & \mathbf{A}_2 \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right], \tag{5.45}$$

will be an orthogonal matrix if and only if $\mathbf{A}_1 = \mathbf{A}_2 = \mathbf{0}$, meaning that we cannot allow any overlap across the signal border, and, thus, $\tilde{\mathbf{P}}$ has its first and last λ columns with zero entries.

Isolating the degrees of freedom

Proposition 5.8 *The total number of degrees of freedom for all possible choices of boundary LTs, assuming MCD and obeying the SDF structure, is* $\nu = [(4K+1)(M-1) - 1]KM/8$ *for each border.*

For an infinite-length signal, we can draw the flow-graph relating the input and output of the analysis section, as in the two examples shown in Fig. 2.6, which accounts

for permutations and orthogonal matrices and represents an orthogonal system following the factorization of (4.9). The input and output signals are segmented into blocks of M samples, as shown in Fig. 2.6, and blocks are labelled 0 through $N_B - 1$ for the actual support region of $x(n)$ and $y(n)$. A simple way to find the complete SDF relevant for the signal is:

1. Construct the flow-graph for the hypothetical infinite-length signal as in the examples shown in Fig. 2.6;

2. Eliminate unnecessary paths and boxes, used for the signal outside the bounds.

3. From the remaining boxes, those which are connected to output blocks numbered K through $N_B - K - 1$ are the same as in the time-invariant SDF and are not changed for the MCD, while the remaining can be any orthogonal matrices (maintaining their sizes) and are responsible for the degrees of freedom in the transitory boundary filter banks.

Let us describe in detail the second and third steps. Let the i-th stage be the one with all matrices $\mathbf{B}_i$. Note that each box labelled $\mathbf{B}_i$ has two input or output branches, each carrying $M/2$ samples. To prune unnecessary branches and boxes, start by disconnecting the input samples outside signal bounds from the flow-graph. For i varying from $i = N - 1$ through $i = 0$, check all boxes in stage $N - 1$, then proceed with stage $N - 2$ towards stage 0. For each box in each stage, check its input branches. If both of its input branches are disconnected, erase this box and its output branches. If only one input branch is disconnected, erase one output branch and make the box in

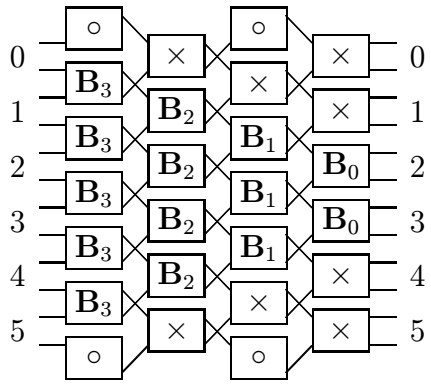


Figure 5.3: Pruned flow-graph for a size-limited orthogonal implementation of a PUFB for $N = 4$ and $N_B = 6$. Each branch carries $M/2$ samples. The 6 input and output blocks are numbered and generic $M \times M$ orthogonal matrices are marked with $\times$ while generic $M/2 \times M/2$ orthogonal matrices are marked with $\circ$.

question an $M/2 \times M/2$ orthogonal matrix. If both input branches are connected, leave the box as an $M \times M$ orthogonal matrix. When the pruning process is complete, and the boxes belonging to the transitory boundary filter banks are selected, we will have some orthogonal matrices (with sizes $M/2 \times M/2$ or $M \times M$) as degrees of freedom. For example, for $N = 4$ and $N_B = 6$ the resulting flow-graph is shown in Fig. 5.3, where the generic orthogonal matrices are indicated.

An $n \times n$ orthogonal matrix has $n(n-1)/2$ degrees of freedom corresponding to its plane rotation angles [32]. The reader can check that, for each border, the number of generic orthogonal boundary matrices is

$$\begin{aligned} \text{stage } \ 2i \ &\Rightarrow \ K - i \text{ matrices of size } M \times M \\ \text{stage } \ 2i+1 \ &\Rightarrow \ K - i - 1 \text{ matrices of size } M \times M \\ &\qquad \text{and one matrix of size } \ \frac{M}{2} \times \frac{M}{2} \end{aligned}$$

$$i = 0, 1, \ldots, K-1$$

Table 5.1: The total number of degrees of freedom for each border, ν.

	M	2	4	6	8	10	12	14	16	20	24	32
	1	1	7	18	34	55	81	112	148	235	342	616
	2	4	26	66	124	200	294	406	536	850	1236	2224
K	3	9	57	144	270	435	639	882	1164	1845	2682	4824
	4	16	100	252	472	760	1116	1540	2032	3220	4680	8416
	5	25	155	390	730	1175	1725	2380	3140	4975	7230	13000

Hence, the total number of degrees of freedom for each border is

$$\nu = \left(\sum_{i=1}^{K} i + \sum_{i=1}^{K-1} i \right) \frac{M(M-1)}{2} + K \frac{\frac{M}{2}(\frac{M}{2}-1)}{2} = [(4K+1)(M-1)-1]\frac{KM}{8} \quad (5.46)$$

and 2ν is the total number of degrees of freedom for both borders of the signal.

Optimal boundary filter banks

In the design of the boundary filter banks, for an optimal orthogonal solution, we span all degrees of freedom in a search for the minimum of a specific cost function. As the relation among the plane rotations and cost functions is generally non-linear, an optimization algorithm would generally have slow convergence and lead to a local minimum. So, a large number of variables to optimize can be burdensome. Note that ν can be a very big number (see Table 5.1, for some choices of K and M).

In a simple example, for $M = 2$, $N = 4$ and $N_B = 6$ (see Fig. 5.3) we have 4 degrees of freedom at each border. ($M/2 = 1$ and the 1×1 "orthogonal" matrices are set to 1.) We started with a 2-channel 8-tap PR PUFB shown in Fig. 5.4(a) and used an unconstrained non-linear optimization routine provided by MATLAB 4.0 to optimize the border matrices (one plane rotation angle per matrix), where the function maximized

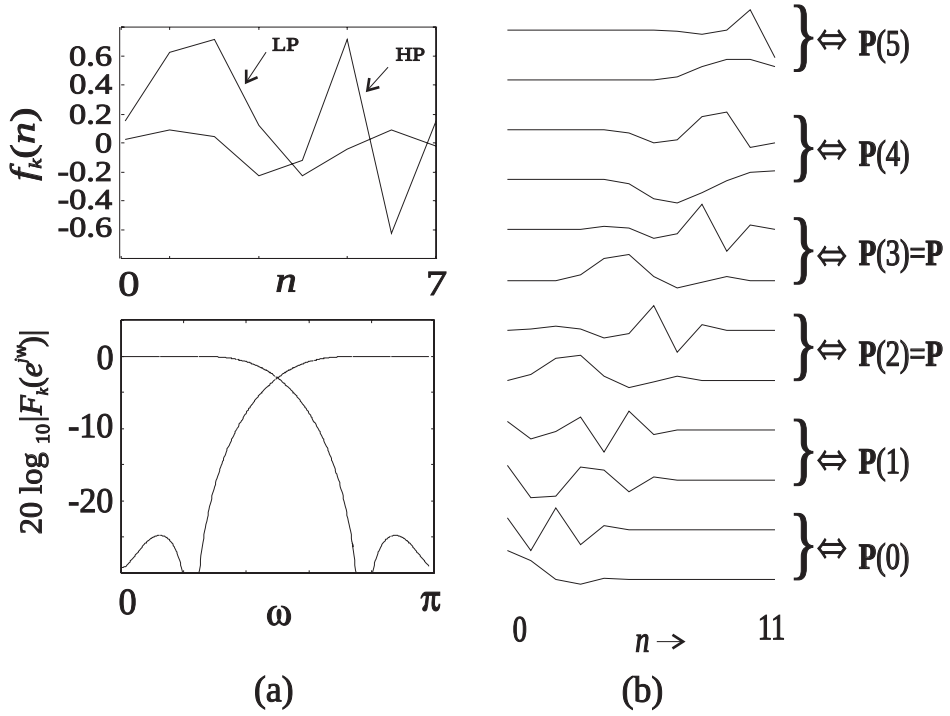


Figure 5.4: Design example of orthogonal boundary filter banks. (a) an 8-tap 2-channel LT ($L = 8$,$M = 2$), where the low-pass (LP) and high-pass (HP) filters $f_k(n)$ and their frequency responses are shown. (b) Design result of the bases (filters) for a 12-sample signal ($N_B = 6$).

was an average of the stopband attenuation of the boundary filters. The 12 resulting bases for the 12-sample signal are shown in Fig. 5.4(b), where the relation of the basis functions and $\mathbf{P}(m)$ ($m = 0, \ldots, 5$) is indicated. Note that $\mathbf{P}(2) = \mathbf{P}(3) = \mathbf{P}$ for MCD, and the 4 bases in the middle of Fig. 5.4(b) are the same as those in Fig. 5.4(a). For this example, what is actually shown in Fig. 5.4(b) are the basis functions of a 12×12 orthogonal transform with sparse composition obtained from two-channel LTs. This is to illustrate the meaning of the term orthogonal boundary filter banks, which implies filter banks which will cause $\mathbf{T}$ to be an orthogonal matrix.

As a second example, we used the 8-channels MLT (ELT-1,$N = 2$). We have

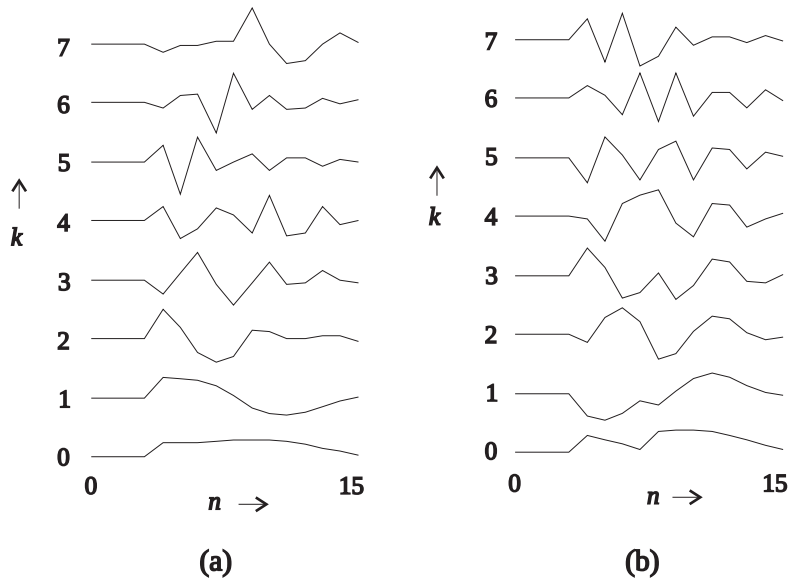


Figure 5.5: Design example of orthogonal boundary filter bank for an MLT with $N = 2$ and $M = 8$. (a) Optimized filter bank $G_{TC} = 9.19$ dB. (b) Standard boundary filter bank $G_{TC} = 5.66$ dB.

34 degrees of freedom at each border and we focussed our attention to just one border for comparison purposes. Note that just one filter bank, $\mathbf{P}(0)$, needs to be optimized because $K = 1$. Malvar [49] provided a standard boundary solution for the MLT which is orthogonal and, therefore, it is a special case among all solutions wherein the 34 degrees of freedom would span. Here, we maximized the transform coding gain G_{TC} (see [33] and (3.31) or (4.37)). for the boundary filter bank, just as we discussed previously. In Fig. 5.5 it is shown the bases $p_{kn}(0)$ of the standard boundary filter bank proposed by Malvar [49] and the optimized one. The G_{TC} for the optimal boundary filter bank is 9.19dB, compared to 5.66dB for that of Malvar. As a reference, the MLT has G_{TC} ranging from 8.25dB through 9.22dB (it depends upon a design parameter [49]) and the DCT has $G_{TC} = 8.83$dB.

5.6 Algorithms for the ELT

With the methods discussed so far, we are able to devise a fast implementation algorithm for the ELT to process finite-length signals. As we saw, symmetric extensions will lead to non-orthogonal boundary factors and an alternative is the use of optimized boundary filter banks. We will discuss algorithms only for the ELT-1 (MLT) and ELT-2 while algorithms for higher values of K can be easily devised using the techniques discussed in this chapter.

5.6.1 Optimization

The MLT ($K = 1$) can be implemented using the flow-graph in Fig. 5.6. In this flow-graph, each branch carries $M/2$ samples and analysis is accomplished by following the paths from left to right, while the synthesis (inverse transform) is achieved by following the paths from right to left, replacing the $\mathbf{Z}$ matrices by their inverses. This flow-graph is the result of the method discussed in the previous section for $N = 2$.

As a matter of reference, the standard orthogonal MLT flow-graph for finite-length signals defined by Malvar [49] is the one where

$$\mathbf{Z}_A^{(1)} = \mathbf{Z}_C^{(1)} = \mathbf{J}_{M/2} \tag{5.47}$$

$$\mathbf{Z}_B^{(1)} = \mathbf{D}^{II} \tag{5.48}$$

$$\mathbf{Z}_D^{(1)} = \mathbf{D}^{IV} \tag{5.49}$$

From Table 5.1, we can see that for $N = 4$ ($K = 2$), ν grows very rapidly for increasing M making the task of non-linear optimization almost impossible. For this

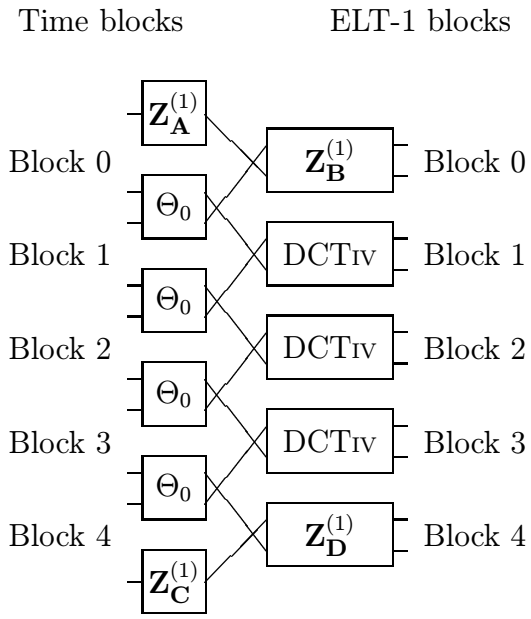


Figure 5.6: Flow graph for finite-length signals for MLT (ELT-1, $K = 1$). Each branch carries $M/2$ samples. Forward transform is performed by following the flow-graph from left to right, while inverse transform is performed by following the flow-graph in the opposite direction and substituting the $\mathbf{Z}$ matrices by their inverses (transposes, if they are orthogonal).

reason, we present a simplified solution which is shown in Fig. 5.7. In this case, the number of angles to be optimized for each border is reduced to $(2.5M - 3)M/2$, so that for $M = 8$ there are there are 68 angles to be optimized, against 124 for the general $K = 2$ case.

The standard orthogonal ELT-2 flow-graph for finite-length signals defined by Malvar [49] is the one where

$$\mathbf{Z}_A^{(2)} = \mathbf{J}_{M/2} \tag{5.50}$$

$$\mathbf{Z}_B^{(2)} = \mathbf{J}_{M/2} \tag{5.51}$$

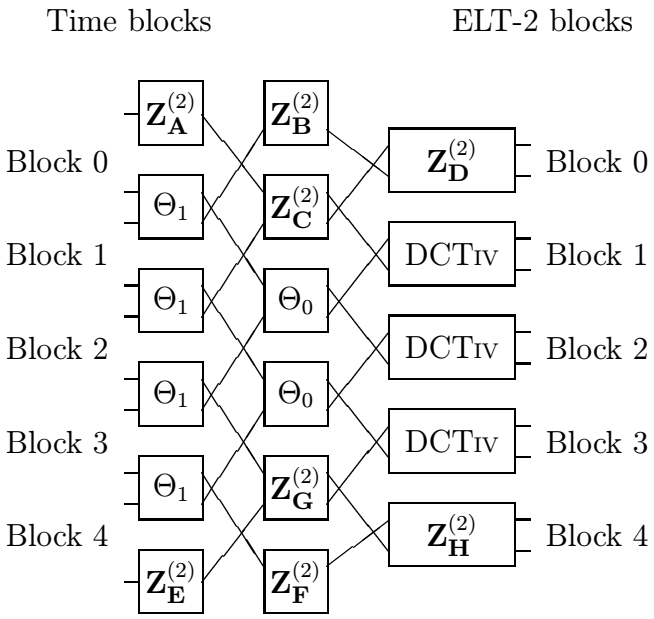


Figure 5.7: Alternative flow-graph for semi-optimized ELT-2. Each branch carries $M/2$ samples. All $\mathbf{Z}$ matrices are orthogonal.

$$\mathbf{Z}_C^{(2)} = \Theta_0 \tag{5.52}$$

$$\mathbf{Z}_D^{(2)} = \mathbf{D}^{IV} \tag{5.53}$$

$$\mathbf{Z}_E^{(2)} = \mathbf{J}_{M/2} \tag{5.54}$$

$$\mathbf{Z}_F^{(2)} = \mathbf{I}_{M/2} \tag{5.55}$$

$$\mathbf{Z}_G^{(2)} = \Theta_0 \tag{5.56}$$

$$\mathbf{Z}_H^{(2)} = \mathbf{D}^{II}. \tag{5.57}$$

We can see that the standard algorithm is fast, although simplistic, since all matrices were chosen among simple matrices which can be implemented by using fast algorithms, paying the price of a somewhat less desirable performance. It also may be useful to optimize the ELT for the two-channel case, so that it can be usefully applied in the

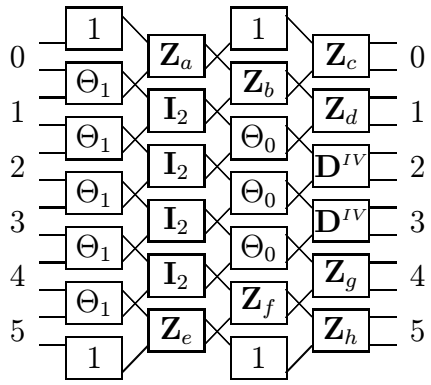


Figure 5.8: Flow-graph for optimized two-channel ELT-2.

construction of time-varying wavelet packets based on binary trees. In this case, we can surely use all angles in the optimization process and the flow-graph in Fig. 5.8. Also, the $M/2 \times M/2 = 1 \times 1$ "orthogonal" matrix is 1.

Examples of optimized matrices are presented in Appendix B.

5.6.2 Symmetric extensions

Symmetric extension applied to the ELT will lead to a size limited transform $\mathbf{T}$ which is not orthogonal. However, in most cases, the condition number of the non-orthogonal factors involved may not be very far from unity. Additionally, we can have fast algorithms along with the absence of artificial discontinuities. For the MLT, symmetric extensions are implemented if we set

$$\mathbf{Z}_A^{(1)} = (\mathbf{S}_0 - \mathbf{C}_0)\mathbf{J}, \tag{5.58}$$

$$\mathbf{Z}_B^{(1)} = \mathbf{Z}_D^{(1)} = \mathbf{D}^{IV}, \tag{5.59}$$

$$\mathbf{Z}_C^{(1)} = \mathbf{J}(\mathbf{S}_0 + \mathbf{C}_0). \tag{5.60}$$

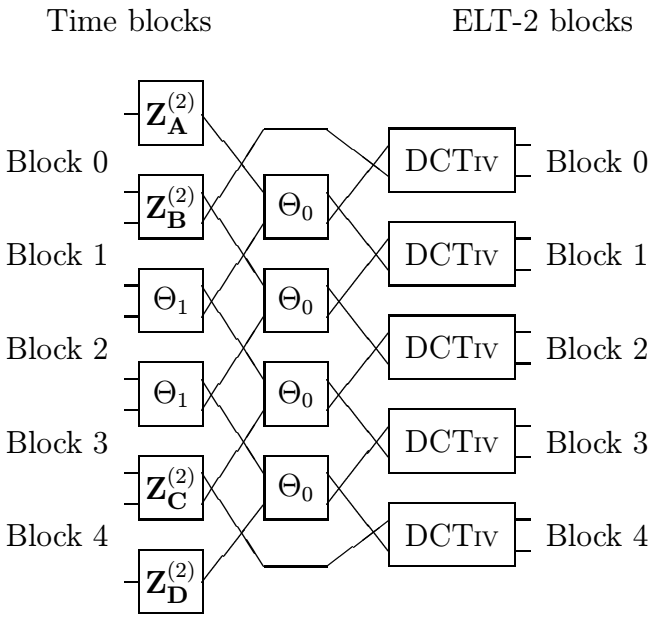


Figure 5.9: Flow graph for finite-length signals for ELT-2 ($K = 2$). Each branch carries $M/2$ samples.

Similarly, Fig. 5.9 shows the flow-graph for PR implementation of the ELT-2 using symmetric extensions, where

$$\mathbf{Z}_A^{(2)} = (\mathbf{S}_1 - \mathbf{C}_1)\mathbf{J}, \tag{5.61}$$

$$\mathbf{Z}_B^{(2)} = \begin{bmatrix} -\mathbf{C}_1 & \mathbf{S}_1\mathbf{J} \\ (\mathbf{S}_0 - \mathbf{C}_0)\mathbf{S}_1 & (\mathbf{S}_0 + \mathbf{C}_0)\mathbf{C}_1\mathbf{J} \end{bmatrix}, \tag{5.62}$$

$$\mathbf{Z}_C^{(2)} = \begin{bmatrix} \mathbf{J}(\mathbf{C}_0 - \mathbf{S}_0)\mathbf{C}_1 & \mathbf{J}(\mathbf{C}_0 + \mathbf{S}_0)\mathbf{S}_1\mathbf{J} \\ \mathbf{J}\mathbf{S}_1 & \mathbf{J}\mathbf{C}_1\mathbf{J} \end{bmatrix}, \tag{5.63}$$

$$\mathbf{Z}_D^{(2)} = \mathbf{J}(\mathbf{S}_1 + \mathbf{C}_1). \tag{5.64}$$

Note that except for the DCT type IV matrix, in the MLT, the other boundary matrices are non-orthogonal and are responsible for producing a non-orthogonal $\mathbf{T}$.

Note that $\mathbf{Z}_A^{(1)}, \mathbf{Z}_B^{(1)}, \mathbf{Z}_A^{(2)}$ and $\mathbf{Z}_D^{(2)}$ are simple counter-diagonal matrices with trivial implementation (one multiplication per input sample). Thus, their inverses are also counter diagonal. $\mathbf{Z}_B^{(2)}$ and $\mathbf{Z}_C^{(2)}$ are composed of $M/2$ butterflies, similar to those on Fig. 3.2, but the lattice is no longer orthogonal (see (5.62) and (5.63)). Their inverses are obtained by inverting each of the butterflies. As a result, both analysis or synthesis have the same fast algorithm. The DCT-IV and Θ_n matrices do not need replacement in the synthesis section because they are both symmetric and orthogonal.

These algorithms for ELT-1 and ELT-2 were found by using a symmetric extension and applying regular ELT flow-graph to the extended sequence. Then, it is found a size-limited flow-graph that would be equivalent. Values of K greater than 2 can also be found, but we will use mostly the ELT-2 and occasionally the ELT-1. As an example, the starting algorithm for an ELT-3 is illustrated in Fig. 5.10, where

$$\mathbf{Z}_A^{(3)} = (\mathbf{S}_2 - \mathbf{C}_2)\mathbf{J}, \tag{5.65}$$

$$\mathbf{Z}_B^{(3)} = \begin{bmatrix} -\mathbf{C}_2 & \mathbf{S}_2\mathbf{J} \\ (\mathbf{S}_1 - \mathbf{C}_1)\mathbf{S}_2 & (\mathbf{S}_1 + \mathbf{C}_1)\mathbf{C}_2\mathbf{J} \end{bmatrix}, \tag{5.66}$$

$$\mathbf{Z}_C^{(3)} = \begin{bmatrix} (\mathbf{C}_0 + \mathbf{S}_0)\mathbf{S}_1 & (\mathbf{C}_0 + \mathbf{S}_0)\mathbf{C}_1\mathbf{S}_2 & (\mathbf{S}_0 - \mathbf{C}_0)\mathbf{C}_1\mathbf{C}_2\mathbf{J} \\ -\mathbf{C}_1 & \mathbf{S}_1 & \mathbf{S}_1 \\ \mathbf{0} & -\mathbf{C}_2 & \mathbf{S}_2\mathbf{J} \end{bmatrix}. \tag{5.67}$$

Note that as each part of $\mathbf{Z}_C^{(3)}$ is a diagonal matrix, it can also be implemented with $M/2$ triple butterflies. Thus, the inverse of this matrix is implemented using the inverse of each butterfly.

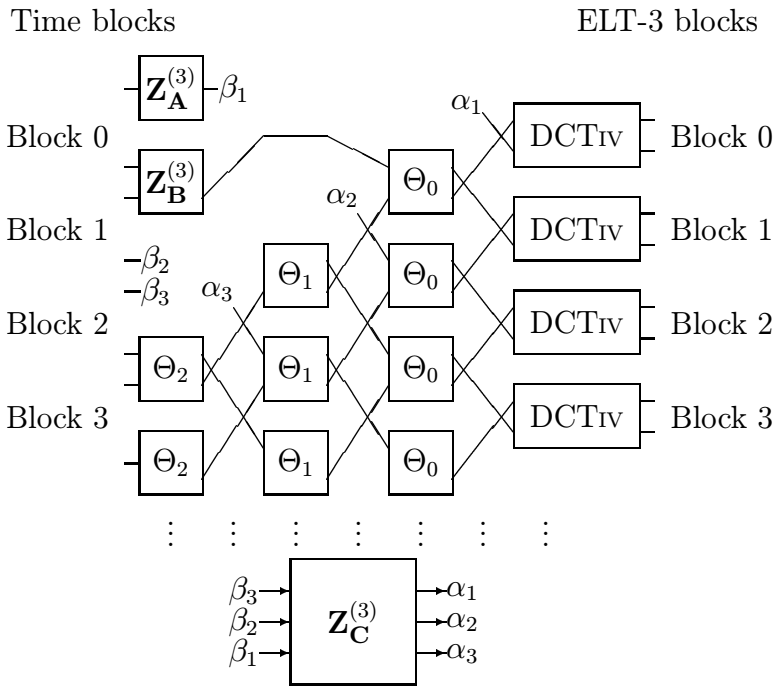


Figure 5.10: Example flow graph for one of the image borders for ELT-3 ($K = 3$). Each branch carries $M/2$ samples. Forward transform is performed by following the flow-graph from left to right, while inverse transform is performed by following the flow-graph in the opposite direction and substituting the $\mathbf{Z}$ matrices by their inverses.

5.7 Image Compression

We can reduce the image processing task to an one-dimensional problem by assuming separable transforms, where processing is applied to the image in a row and column fashion. In image coding, the signal $\mathbf{x}$ is transformed by $\mathbf{T}$, quantized, and transmitted, and, at the receiver side, the signal is recovered by inverse operations, as shown in Fig. 5.11. Orthogonal transforms have several desirable properties in image coding regarding statistics of the quantization noise. Of course, in our case, only the boundary parts of the signal can suffer the effects of the possible non-orthogonality of $\mathbf{T}$. However,

Figure 5.11: Basic subband coding diagram for transmission (or storage), where the LT is represented by its corresponding size-limited transform $\mathbf{T}$, and Q and Q^{-1} represent quantization and inverse quantization, respectively.

as the noise will follow closely the image border, it would have an organized pattern and can be more easily percepted.

The use of LTs for image compression underlies a substitution of the DCT by another transform with increasing overlap, implying in possible border distortions. Note that the DCT has no overlap. We want now to investigate the use of LTs for image coding as a direct replacement for the DCT ensuring that all processing is completely free of perceptible border distortion. Also, we emphasize three major points where LTs need to be proven to be significantly advantageous over the DCT, which are reconstruction quality at low-bit-rates, scalability (allowing the receiver to reconstruct the image at multiple resolutions), and robustness against transmission errors.

5.7.1 Border distortions

We have seen so far several ways to achieve perfect reconstruction of the boundary regions of a finite-length signal. However, when the subbands are quantized, the signal is no longer reconstructed exactly as the noise added by quantization is propagated through the inverse transform. If the noise appearing in the boundary region of the signal follows a different pattern than the pattern of the noise occurring in non-boundary regions, then, boundary artifacts can still appear. The boundary artifacts are more easily perceptible

because of their spatial pattern (following the borders of the image). One must ensure that the encoder gives a sufficient amount of resources to encode the boundary blocks and thereby reduce the visibility of the error, but not always coders are designed with such constraint, and even if they are, boundary blocks can drain precious information bits that could, otherwise, be used to better encode the image as a whole. PR analysis-synthesis systems may not be free of border distortions if the boundary filter bank is not orthogonal generating amplification and coloring of the quantization error pattern. Also if the boundary filter bank has the most important basis functions (those which concentrate most of the energy) with discontinuities, the reconstructed image can present visible artifacts at low bit-rates and the LT will not provide energy compaction even for smooth images (for example, lack of polyphase normalization, i.e., a flat image region will not just produce one non-zero LT coefficient).

The GenLOTs are free of any kind of block distortion using the symmetric extension method. The boundary regions of the reconstructed image present a quantization noise pattern that is nearly identical to the pattern of the image internal regions. The ELTs, however, cannot achieve orthogonality with symmetric extension, and we will show how these boundary artifacts can be easily noticeable by using the wrong approach.

Consider the original 8 bits per pixel (bpp) image "Lena" in Fig. A.1. Another image, consisting of a 48×48-pels tile of this test image, is shown in Fig. A.2(a), where the region possibly affected by the boundary filter banks for an MLT (ELT-1) with $M = 8$ is indicated. In Fig. A.2(b)-(d) it is compared the performance of several boundary approaches, simulating an environment of high compression. From Fig. A.2 it can be

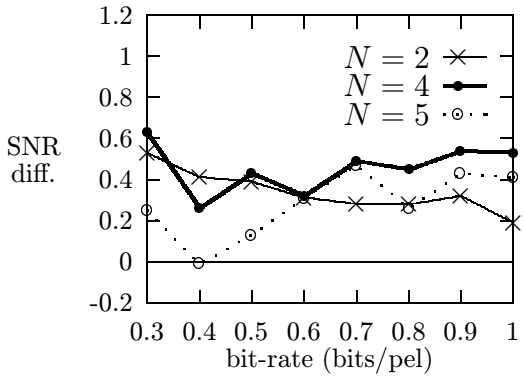
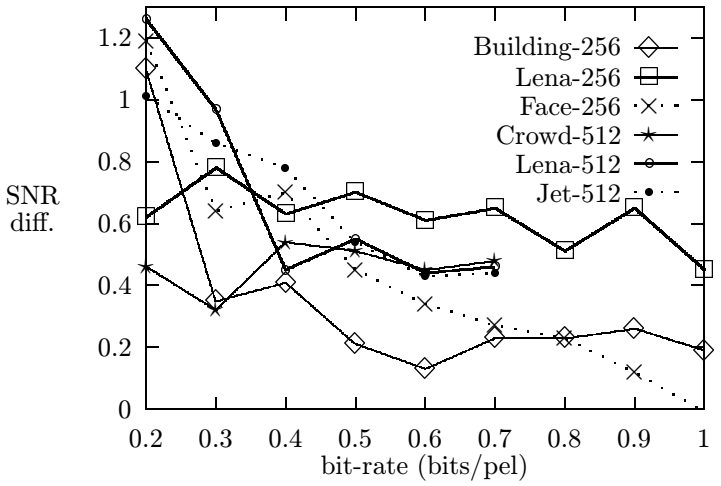


Figure 5.12: Difference in SNR (in dB) among GenLOTs and the DCT for several bit-rates using test image "Lena" (256×256-pels, 8 bpp) and JPEG baseline coder.

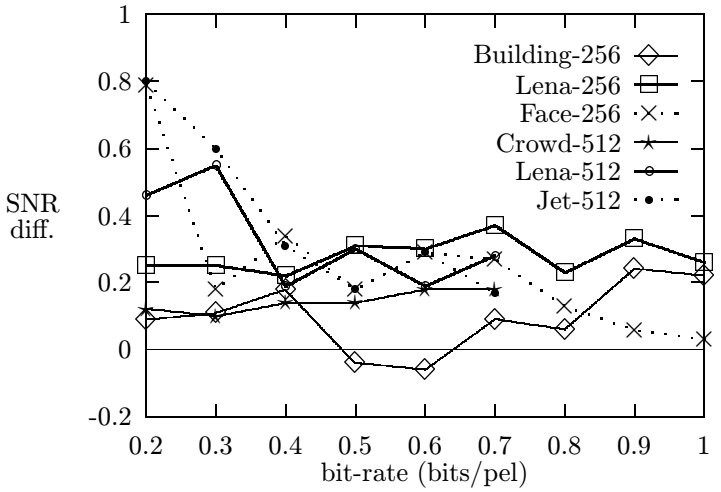
easily seen that the optimized boundary solution is the one which gives the best results, although the non-orthogonal solution (symmetric extensions method) also yields very good results. The standard MLT boundary solution given by Malvar [49] described in (5.47)-(5.49) gives poor results and the periodic extensions are definitely not suited for such applications and will no longer be discussed here.

For the ELT-2 we applied the JPEG baseline coder [68], replacing the DCT by the ELT-2, in blocks of 8×8 ($M = 8$). A comparison of methods is carried in Fig. A.3 where a compression of an 8 bpp 128×128-pels image to 0.7 bpp is carried using Malvar's standard method [49], symmetric extensions, and the semi-optimized boundary solution for the ELT-2.

The results in Fig. A.3 show that the symmetric extension method provides the best results. The distortion does exist. However, it is concentrated in the last row/column of pixels and it is masked by the contrast between the image border and the background.



(a)



(b)

Figure 5.13: Difference in SNR (in dB) between ELT-2 and other transforms using JPEG for several bit-rates. The test images have either 256×256-pels or 512×512-pels, 8 bpp, and their names and sizes are indicated. (a) $\mathrm{SNR}_{\mathrm{ELT}-2} - \mathrm{SNR}_{\mathrm{DCT}}$. (b) $\mathrm{SNR}_{\mathrm{ELT}-2} - \mathrm{SNR}_{\mathrm{LOT}}$.

5.7.2 Objective comparisons

We can compare objectively LTs to the DCT using the signal to noise ratio (SNR) between original and reconstructed images. However, it may be more instructive to show the difference in SNR between the LTs and the DCT, for the same image at several bit-rates. In Fig. 5.12 it is shown the SNR difference among several GenLOTs and the DCT for image "Lena" (256×256-pels, 8bpp) using the JPEG baseline coder [68]. In Fig. 5.13 it is shown the SNR differences comparing the ELT-2 to both the DCT and the LOT. In this case we carried the tests over several 8 bpp images which can have either 256×256-pels or 512×512-pels. The name and size of the images tested is indicated in Fig. 5.13.

Fig. A.4 shows reconstructed images after compressing image "Lena" using the JPEG coder [68], but replacing the DCT by the GenLOT and the ELT-2. Also, Fig. A.5 compares the DCT to both LTs for low-bit-rate encoding.

5.7.3 Scalability

Another important issue is the filtering capabilities of the LT. Let an image of $N_1 \times N_2$-pels be separately processed by an $M_1 \times M_2$-channels LT and suppose we want to display the reconstructed images fairly well in several smaller resolutions. This is common when it is necessary to compress an image and maintain compatibility with previous display systems which have lower resolutions. As the processing is separable, we can easily understand this task in one dimension as shown in Fig. 5.14. In Fig. 5.14 it is shown how to efficiently implement a system to perform synthesis, filtering, and down-

sampling into one (actually simplified) synthesis operation. For a decimation factor of k, the equivalent anti-aliasing filter is the sum of the first M/k filters of the LT. Fig. 5.15 shows the equivalent low-pass anti-aliasing filter corresponding to the first $8/k$ filters of the DCT, LOT, and ELT-2 for an $M = 8$ LT. It is easy to see that the ELT-2 outperforms its competitors with higher stopband attenuation and sharper transitions. Examples of down-scaled reconstructed images using the JPEG coder are shown in Fig. A.6, comparing the DCT and the ELT-2, where the higher quality of the reconstructed image using the ELT-2 is clearly visible, presenting higher frequency details compared to the one reconstructed using the DCT.

5.7.4 Robustness against transmission errors

Most communications protocols packetize data into cells and provide cell prioritization to protect more important data, as cell losses can occur. This is appropriate, for example, for the asynchronous transfer mode (ATM) networks, which are gaining acceptance lately. For image and video transmission in such networks, compressed data can be unrecoverably lost and a more robust encoding approach is necessary, as well as developing reconstruction procedures.

In order to simplify the problem and its modelling, we assume that [27]: (i) the DC coefficients of the transform are prioritized and transmitted with enough protection so that they are not susceptible to errors; (ii) if a cell loss occurs, the information regarding all AC coefficients of only one block is lost. In this case, see [27] for details on packet losses and on efficient methods for estimating the lost AC coefficients using the LOT.

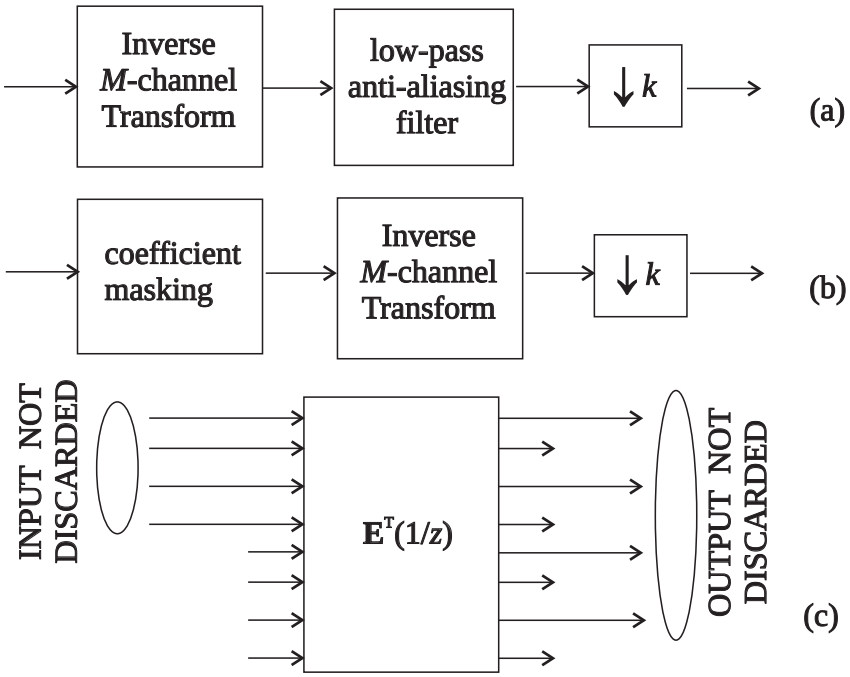


Figure 5.14: Scaling of a reconstructed image to lower resolutions. (a) The image is down-sampled by a factor of k after processing by a low-pass anti-aliasing filter with cutoff frequency in π/k. (b) If the filtering is conducted in the transform domain, the filtering can be accomplished by masking some LT coefficients before the inverse transform. (c) Faster implementation of (b) by using a pruned synthesis, where only few input/output samples are computed. In this particular illustration, $M = 8$ and $k = 2$.

The ELT-2 is expected to perform better than LOT because of its larger overlapping, since the spatial region affected by the lost-block will increase making the error locally less intense. However, this is only partially true. We have discovered that the great robustness of the ELT-2 comes from its non-linear-phase-filters allied with the longer overlap. The error using ELT-2 is sparser compared to a filter bank with the same filters' length but with linear-phase (such as a GenLOT with $N = 4$ too, which we refer as LOT-2). We transformed an image using various transforms and deleted all the coefficients of a single block except the DC term. After respective inverse transforms, Fig. A.7 shows a zoom

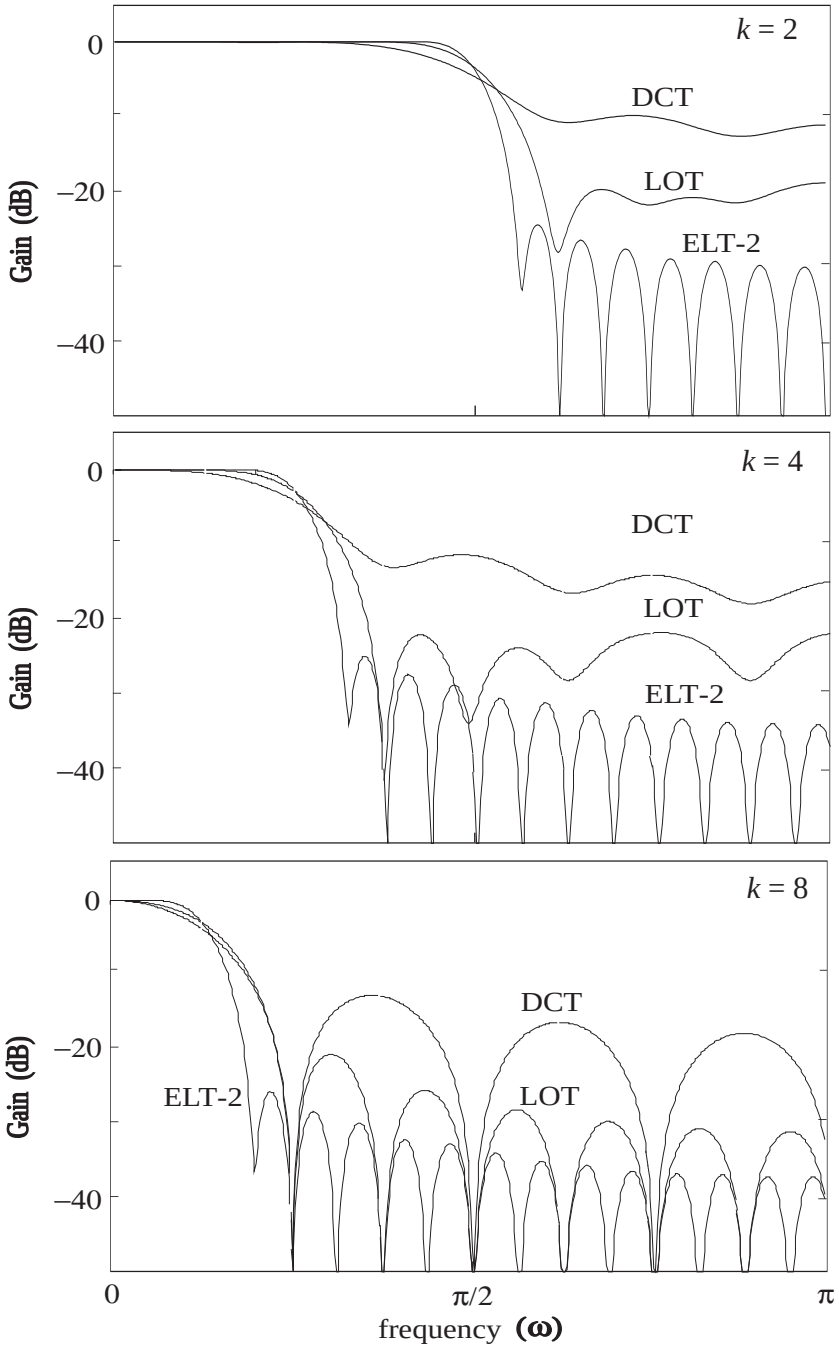


Figure 5.15: Frequency response of the equivalent decimation low-pass filter with cut-off frequency $= \pi/k$ produced by the first $8/k$ filters of an 8-channel LT. These filters are present to prevent aliasing for a k:1 down sampling of the signal. Plots for DCT, LOT, and ELT-2 are shown for $k = 2$, 4, and 8.

of the region where a lost-block occurred, using DCT, LOT, LOT-2 and ELT-2. From the DCT results, we can clearly see where the lost-block was located and, comparing all results, we can see that the ELT-2 performed fairly better than its competitors, in view of its improved masking of the errors.

To carry out our tests we selected again the JPEG baseline coder [68] because its algorithm is popular and is well understood. We assume it will resemble image/video coders actually used in packetized transmission of images. Then, we replace the 8×8 DCT by the ELT-2 with $M = 8$. Fig. A.8 shows the 256×256-pels image Lena coded at 0.8 bits/pel (bpp) using JPEG for both the DCT and the ELT-2. For a better visualization, a dramatic error rate was used, and we simulated 5% rate of lost blocks (51 blocks are lost). The errors occur randomly, but we forced the position of the errors used for the DCT to be repeated for the ELT-2, so that we can compare the effects of a block-loss for both transforms in identical positions in the test image.

5.7.5 Remarks on image coding tests

The semi-optimized orthogonal boundary ELT-2, did not show much advantage over its non-orthogonal counterpart (symmetric extensions method) for $M = 8$. Although the number of constraints was greatly reduced, it makes sense to believe that either the optimization routine was trapped into a local minimum or the cost function chosen is not appropriate for the task. The convergence for each border took about 10 hours on a 486 PC with 66 MHz clock, and using MATLAB 4.0. As we discussed, the cost function as maximum G_{TC} may not be sufficient, and other factors such as discontinuities in the

basis functions may also influence the performance of the boundary LT. That is what actually happened, as a thin stripe appears near the borders of the reconstructed image after its compression using the ELT-2.

The optimized 2-channel ELT-2 was tested in an image coding environment as a basis LT for an octave DWT. Although the optimized borders were successfully applied, showing no border distortion face quantization of the subbands whatsoever, the overall performance of the 2-channel ELT-2 in a JPEG-like wavelet coder was not very exciting. The DWT based on the 2-channel linear-phase Johnston's quadrature mirror filter bank showed better performance than the ELT-2. Also different designs of the modulating window led to very different results. Thus, one may find the most appropriated design of the 2-channel ELT-2 window and the most indicated coder for a ELT-2-based DWT. Such work would be extensive and it is outside the scope of this dissertation.

The use of time-varying LTs in image coding has yet to be investigated in more detail.

CHAPTER 6

CONCLUSIONS

The interconnection of filter banks, LTs, wavelet transforms, and time-frequency analysis, is so rich that this dissertation can be extended without an end in sight. Instead, in this Dissertation it was intended to gain insight into uniform paraunitary filter banks through the viewpoint of lapped transforms. LTs are often assumed to possess some sort of fast implementation algorithms, although such a restriction is not necessary, as the first LTs were created to rival the DCT in image coding. The equivalence to a certain kind of filter banks opens new sights for application of LTs and the cross section of the theory between both fields enormously aids the understanding of each other.

Most of the material in this Dissertation is based on original research, although the ELTs were developed elsewhere. Also, Chapter 2 covers mostly background material. However, its presentation follows a singular point-of-view which is helpful to understand the following chapters. Also, some key ideas are better developed and highlighted. New results are presented, including: the development of new LTs with general overlapping factors and linear phase filters, the theory of perfect reconstruction LTs for finite-length signals, the perfect reconstruction conditions for time-varying LT, and a theory to implement time-varying LTs and wavelet packets with the perfect reconstruction and orthogonality properties.

Time-invariant LTs were presented, designed, optimized, and applied to image coding. Of course, for this, it was necessary to develop an adequate theory for processing finite-length signals without using a periodic extension (purely circular convolution), which introduces artificial discontinuities to the signal. It was also necessary to study the problem of finite-length signals to construct the time varying wavelet packets, which turned out to be one of the major developments of this dissertation. Algorithms for signal extension and for optimizing the boundary LTs were developed and successfully tested/implemented.

Neither image coding nor optimization techniques are the topics of this Dissertation. They were just used to formulate examples. Few results were not satisfactory, mainly because of inadequate design, which is a consequence of the complexity in some design techniques through optimization of plane rotations. Most algorithms were developed by a factorization of orthogonal matrices based on plane rotations, leading to a strongly non-linear optimization task. As a result, cost functions to be minimized are kept simple and often a local minimum is reached. In this category of optimized LTs are some optimal boundary LTs and the GenLOTs, having in mind the image coding application. In other words, we provided the means and theory to develop such LTs, and the designs presented are mere examples that can be surely improved.

The theory of time-varying LTs allows us the freedom to change LTs (filter banks) in a way that has not been reported elsewhere. This freedom is used to construct time-varying wavelet packets which are able to implement systems to decompose the signal in a maximally-decimated way using virtually any rectangular tiling of the time-frequency

plane. Such a powerful tool can be used in many fields of signal analysis and compression. The time-varying wavelet packet can use the fast implementations of the ELTs as a basic cell, forming a formidable technique to decompose the signal into orthogonal bases and to achieve the best time-frequency localization for a particular application.

In all cases, perfect reconstruction and orthogonality are inherently assured, along with fast implementation algorithms and good performance. It is acknowledged that much has still to be done to fully develop the theory discussed, and to perfectly match the LTs to other demanding applications. Future work is necessary, and is planned, to further study time-varying LTs.

APPENDIX A

HALFTONE IMAGES

Only one test image was used, which is called "Lena". It has resolution of either 256×256 pels or 512×512 pels at 8 bits/pel (bpp), and is shown in Fig. A.1. The images involved in the discussion of border distortions when the MLT or ELT-2 are applied to finite-length signals, in Chapter 5, are shown in Fig. A.2 and in Fig. A.3. Fig. A.4 and Fig. A.5 show examples of image compression results (on the 8bpp image "Lena") using the ELT-2 and the GenLOT. Fig. A.6 carries a comparison between DCT and ELT-2 regarding filtering capabilities in scalable coding, as discussed in Chapter 5. Another comparison, regarding robustness against transmission errors, is carried out in Fig. A.7 using several transforms. The reconstructed image quality is comapred when all AC coefficients in a block are lost. In this context, the performance of the DCT and ELT-2 are compared in Fig. A.8. All images were rendered and printed at a 600 dots-per-inch (dpi) resolution using a non-noisy clustering method that should allow photocopying without imposing much distortion.

Figure A.1: Original 256×256-pels (8 bpp) image "Lena".

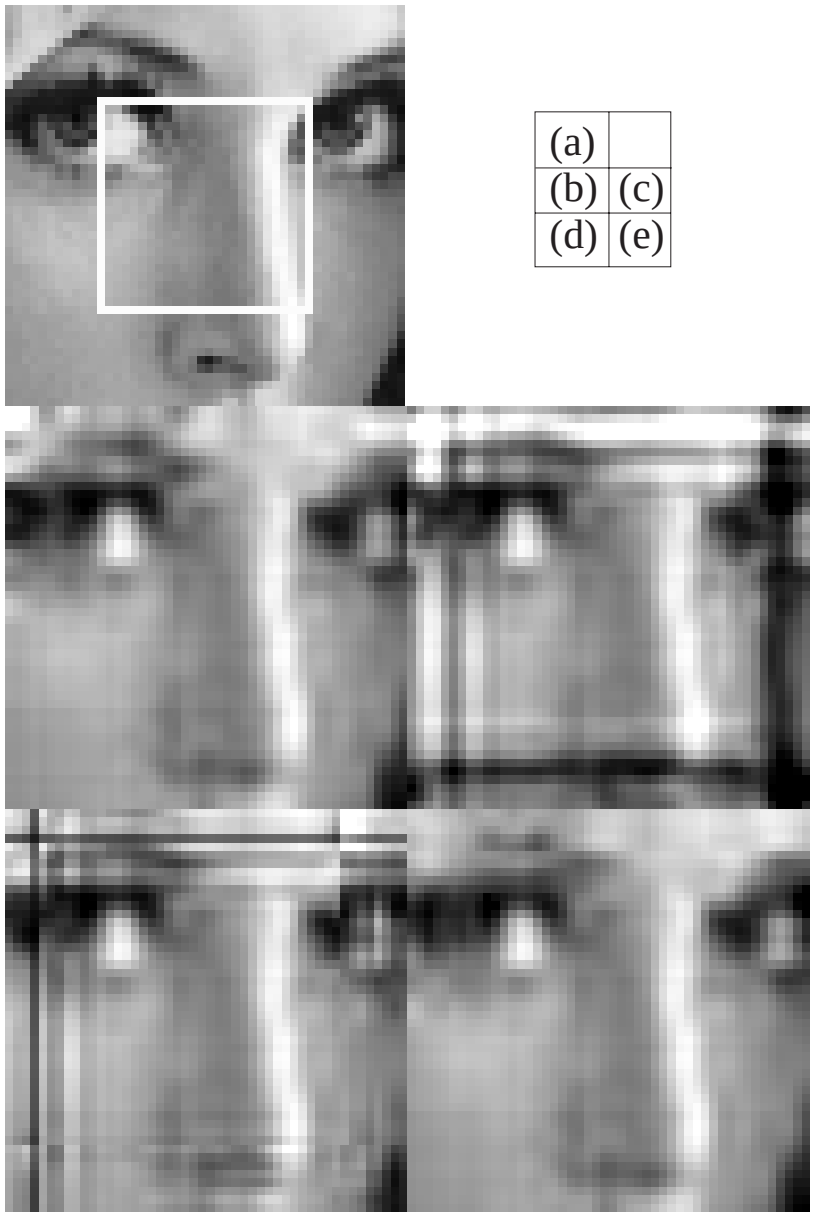


Figure A.2: Coding tests on a 48×48-pels image, simulating high-compression rates. The basic filter bank is the MLT with $M = 8$. (a) Original image indicating area possibly affected by border distortions. (b) Result using symmetric extensions. (c) Result using periodic extension and circular convolution. (d) Result using the standard MLT boundary filter bank. (e) Result using an optimal boundary filter bank, optimized for maximum G_{TC}.

picture labelling →

(a)	(b)
(c)	(d)

Figure A.3: Coding tests on an 8 bpp 128×128-pels image, using the JPEG baseline coder, but substituting the DCT by the the ELT with $M = 8$. The original image is shown along with reconstructed images after compression to 0.7 bpp. (a) Original image showing the regions which can possibly be affected by any border distortion. (b) Result using symmetric extensions (non-orthogonal). (c) Result using the standard ELT-2 boundary filter bank. (d) Result using an optimal boundary filter bank, optimized for maximum G_{TC}.

(a)

Figure A.4: Reconstructed images after compression of image "Lena", using JPEG. (a) ELT-2 applied to 256×256-pels image, compression to 1 bpp; (b) GenLOT ($N = 4$) applied to 256×256-pels image, compression to 1 bpp; (c) ELT-2 applied to 512×512-pels image, compression to 0.5 bpp; (d) GenLOT ($N = 4$) applied to 512×512-pels image, compression to 0.5 bpp.

Fig. A.4(b)

Fig. A.4(c)

Fig. A.4(d)

(a)

Figure A.5: Zoom of reconstructed images after compression of 512 × 512-pels image "Lena" to 0.25 bpp, using JPEG baseline coder, and for different LTs. (a) DCT; (b) ELT-2; (c) GenLOT ($N = 3$).

Fig. A.5(b)

Fig. A.5(c)

(a)

Figure A.6: Down-sampling results using fast prunned synthesis. The 512×512-pels image Lena (8 bpp) is compressed using JPEG to a rate of 0.6 bpp, and, then it is reconstructed at a resolution of 128×128-pels. (a) DCT ; (b) ELT-2.

Fig. A.6(b)

Figure A.7: Trivial image reconstruction when all AC coefficents of a single block are lost. The AC coefficients are set to zero in this block and a zoom of reconstructed images for each transform are shown. Top left corner, DCT. Top right corner, LOT. Bottom left corner, LOT-2. Bottom right corner, ELT-2.

(a)

Figure A.8: Image compression results for 256 × 256-pels image Lena (8 bpp), using JPEG to compress the image to a rate of 0.8 bpp. Both compressed images are subject to a block-loss rate of 5%. (a) DCT ; (b) ELT-2.

Fig. A.8(b)

æ

APPENDIX B

OPTIMIZED ANGLES AND USEFUL MATLAB ROUTINES

Let us define two operands: OM and $SORT$. $\mathbf{A} = OM(\mathbf{v})$ is an orthogonal matrix which is parameterized by plane rotation angles which are the elements of vector $\mathbf{v}$. The elements of $\mathbf{v}$ (angles) are given as fractions of π and the algorithm to generate $\mathbf{A}$ from $\mathbf{v}$ is given in a MATLAB routine later in this Appendix. Let $\mathbf{P}'$ be a LT matrix whose basis functions $\mathbf{p}'_k$ are not sorted by its respective frequency slots. Then, $\mathbf{P} = SORT(\mathbf{P}')$ yields $\mathbf{P}$ which is a permutation of the rows of $\mathbf{P}'$ so that if $i < j$, the i-th basis function of $\mathbf{P}$ corresponds to a filter occupying a slot of lower frequency than the one relative to the j-th basis functions. A MATLAB routine for $SORT$ is also presented later on.

The $\mathbf{Z}$ matrices in Fig. 5.6 are given by

$$\mathbf{Z}_A^{(1)} = OM(\mathbf{v}_{za1}) \tag{B.1}$$

$$\mathbf{Z}_B^{(1)} = OM(\mathbf{v}_{zb1}) \tag{B.2}$$

$$\mathbf{Z}_C^{(1)} = OM(\mathbf{v}_{zc1}) \tag{B.3}$$

$$\mathbf{Z}_D^{(1)} = OM(\mathbf{v}_{zd1}) \tag{B.4}$$

where $\mathbf{v}_{za1}$ through $\mathbf{v}_{zd1}$ are vectors containing optimized angles which are presented in Table B.1.

Note that the angles given in Table B.1 does not assure the boundary filter banks will have their basis functions sorted by its respective frequency slots. If the boundary filter bank is denoted as $\mathbf{P}'$ and $\mathbf{AP}' = SORT(\mathbf{P}')$, where $\mathbf{A}$ is a permutation matrix, then the correpondent matrix $\mathbf{B}_0(m)$ can also to be substituted by $\mathbf{AB}_0(m)$ to ensure proper ordering of the basis functions.

Table B.1: Optimized angles (maximum G_{TC}) for the MLT and for $M = 8$. $\mathbf{v}_{za1}$,$\mathbf{v}_{zb1}$,$\mathbf{v}_{zc1}$, and $\mathbf{v}_{zd1}$ are the vectors corresponding to $\mathbf{Z}_A^{(1)}$, $\mathbf{Z}_B^{(1)}$, $\mathbf{Z}_C^{(1)}$, and $\mathbf{Z}_D^{(1)}$.

$\mathbf{v}_{za1}$	$\mathbf{v}_{zb1}$	$\mathbf{v}_{zc1}$	$\mathbf{v}_{zd1}$
6.0361441e-2	-9.8356124e-2	1.3104490e-1	3.3652103e-1
-1.2128763e-1	7.3779229e-2	6.6089743e-2	1.3188390e-1
3.7489016e-2	-1.2759128e-1	2.0368626e-1	5.5250759e-2
-9.0278172e-2	-2.2500217e-1	2.6036638e-1	3.4845902e-1
-2.2009595e-1	-6.1987324e-4	2.6696781e-1	1.0044415e-1
1.9999349e-1	-2.0292659e-1	-2.2559627e-1	1.3996199e-1
N/A	-1.9524805e-1	N/A	-3.0454016e-2
N/A	-1.5181387e-1	N/A	-1.3256903e-1
N/A	1.1500757e-1	N/A	1.0722820e-2
N/A	2.2358023e-1	N/A	9.4583445e-4
N/A	5.7784063e-2	N/A	-6.1996030e-2
N/A	-1.6828920e-1	N/A	-1.3327864e-1
N/A	-3.8397653e-3	N/A	9.2688768e-2
N/A	9.5103389e-2	N/A	-2.9632112e-1
N/A	1.3858998e-1	N/A	-7.4795257e-2
N/A	3.4608093e-1	N/A	-3.2476081e-1
N/A	2.1440864e-1	N/A	-1.4506926e-1
N/A	4.1692375e-1	N/A	-1.8827390e-1
N/A	6.5937913e-2	N/A	-8.5333113e-2
N/A	-7.7791031e-2	N/A	-7.1975058e-2
N/A	9.6472143e-2	N/A	1.4557053e-1
N/A	6.8707142e-2	N/A	-1.5715783e-1
N/A	1.6166533e-1	N/A	-4.1260365e-1
N/A	-3.5256997e-2	N/A	-3.8681753e-2
N/A	-2.6310078e-1	N/A	7.2795891e-2
N/A	-4.0704652e-2	N/A	-2.7519320e-1
N/A	-4.5506472e-2	N/A	5.7620856e-2
N/A	-1.4874539e-1	N/A	-1.9603046e-1

The **Z** matrices in Fig. 5.7 are given by

$$\mathbf{Z}_A^{(2)} = OM(\mathbf{v}_{za2}) \quad \text{(B.5)}$$

$$\mathbf{Z}_B^{(2)} = OM(\mathbf{v}_{zb2}) \quad \text{(B.6)}$$

$$\mathbf{Z}_C^{(2)} = OM(\mathbf{v}_{zc2}) \quad \text{(B.7)}$$

$$\mathbf{Z}_D^{(2)} = OM(\mathbf{v}_{zd2}) \quad \text{(B.8)}$$

$$\mathbf{Z}_E^{(2)} = OM(\mathbf{v}_{ze2}) \quad \text{(B.9)}$$

$$\mathbf{Z}_F^{(2)} = OM(\mathbf{v}_{zf2}) \quad \text{(B.10)}$$

$$\mathbf{Z}_G^{(2)} = OM(\mathbf{v}_{zg2}) \quad \text{(B.11)}$$

$$\mathbf{Z}_H^{(2)} = OM(\mathbf{v}_{zh2}) \quad \text{(B.12)}$$

where $\mathbf{v}_{za2}$ through $\mathbf{v}_{zh2}$ are vectors containing optimized angles which are presented in Tables B.2 and B.3. Note that, as in the case of the MLT, the same consideration regarding sorting the bases function applies to the ELT-2.

Labeling the blocks in Fig. 5.3 as shown in Fig. 5.8, we have the optimized angles (for maximum G_{TC}) given as

$$\mathbf{Z}_a^{(2)} = OM(1.5771354e-001) \quad \text{(B.13)}$$

$$\mathbf{Z}_b^{(2)} = OM(-5.6420094e-001) \quad \text{(B.14)}$$

$$\mathbf{Z}_c^{(2)} = OM(-2.0679954e-001) \quad \text{(B.15)}$$

$$\mathbf{Z}_d^{(2)} = OM(-1.1661845e-001) \quad \text{(B.16)}$$

Table B.2: Optimized angles (maximum G_{TC}) for the the alternative ELT algorithm and for $M = 8$. Angles for the matrices at the left border are shown. $\mathbf{v}_{za2}$,$\mathbf{v}_{zb2}$,$\mathbf{v}_{zc2}$, and $\mathbf{v}_{zd2}$ are the vectors corresponding to $\mathbf{Z}_A^{(2)}$, $\mathbf{Z}_B^{(2)}$, $\mathbf{Z}_C^{(2)}$, and $\mathbf{Z}_D^{(2)}$.

$\mathbf{v}_{za1}$	$\mathbf{v}_{zb1}$	$\mathbf{v}_{zc1}$	$\mathbf{v}_{zd1}$
1.4181352e-1	3.6473780e-1	2.6052679e-2	7.8915688e-2
4.5043396e-2	-2.7172762e-1	4.4098587e-1	-2.5649235e-1
1.6965747e-1	-2.5202078e-1	-1.3575710e-1	4.9865902e-1
8.1650219e-2	2.5014256e-1	3.1337035e-1	2.0513643e-2
-7.4123175e-2	4.7332204e-1	-1.3013016e-1	-1.0658223e-1
-8.1954239e-2	1.8359863e-1	-1.9364221e-1	-1.8332815e-1
N/A	N/A	-3.5251438e-1	4.7814177e-1
N/A	N/A	7.2775599e-2	7.2554914e-2
N/A	N/A	-4.9145027e-2	2.2964964e-1
N/A	N/A	1.3119103e-1	3.6052471e-2
N/A	N/A	1.4342332e-1	-1.0053531e-1
N/A	N/A	-3.8554309e-2	-2.9908737e-1
N/A	N/A	-2.4238374e-1	4.8054091e-2
N/A	N/A	9.7838538e-2	7.4374883e-3
N/A	N/A	-4.2644088e-1	-2.1312100e-1
N/A	N/A	-2.4991673e-1	8.3483976e-2
N/A	N/A	-2.4133041e-1	2.4669203e-1
N/A	N/A	-1.7081655e-1	-6.6289376e-2
N/A	N/A	-2.7379406e-1	-4.6000574e-1
N/A	N/A	-8.8558160e-2	1.8300806e-2
N/A	N/A	-1.0208290e-1	8.1538883e-2
N/A	N/A	-4.0462163e-1	-4.8975555e-2
N/A	N/A	-5.6589225e-2	4.9271676e-1
N/A	N/A	-5.4341537e-2	-1.9251568e-1
N/A	N/A	1.2222002e-1	1.1924225e-1
N/A	N/A	8.1913863e-2	4.1402159e-2
N/A	N/A	-5.4278727e-2	-2.5770042e-2
N/A	N/A	9.1127519e-2	2.9147181e-1

Table B.3: Optimized angles (maximum G_{TC}) for the the alternative ELT algorithm and for $M = 8$. Angles for the matrices at the right border are shown. $\mathbf{v}_{ze2}$,$\mathbf{v}_{zf2}$,$\mathbf{v}_{zg2}$, and $\mathbf{v}_{zh2}$ are the vectors corresponding to $\mathbf{Z}_E^{(2)}$, $\mathbf{Z}_F^{(2)}$, $\mathbf{Z}_G^{(2)}$, and $\mathbf{Z}_H^{(2)}$.

$\mathbf{v}_{ze1}$	$\mathbf{v}_{zf1}$	$\mathbf{v}_{zg1}$	$\mathbf{v}_{zh1}$
-1.1936741e-1	-5.4769188e-2	2.6547390e-2	-5.4237253e-1
-6.9392734e-2	-4.9338643e-1	-8.8550396e-3	1.0731591e-1
3.3510049e-1	-1.7344322e-1	5.7996105e-2	9.2206255e-2
-1.0642071e-1	5.1777178e-2	-3.2948846e-2	-2.7857731e-1
2.3815717e-1	3.2831025e-1	6.1685874e-2	-2.5279014e-1
6.1458354e-2	-4.6561120e-1	-6.5051211e-2	-5.2057911e-2
N/A	N/A	4.3020365e-1	3.6959057e-2
N/A	N/A	3.0092989e-2	8.1699643e-2
N/A	N/A	-5.4945745e-2	-1.2315566e-1
N/A	N/A	1.4147002e-1	-3.9826804e-1
N/A	N/A	2.5803483e-1	-2.5513924e-1
N/A	N/A	4.9665837e-1	5.5021081e-2
N/A	N/A	4.2903287e-2	-6.6158132e-2
N/A	N/A	7.7839170e-2	1.9251213e-1
N/A	N/A	1.3780322e-1	3.3212412e-2
N/A	N/A	4.2542530e-1	1.0602242e-1
N/A	N/A	-2.5487925e-1	-8.6400656e-2
N/A	N/A	-3.3108311e-2	-8.7793763e-2
N/A	N/A	4.6170494e-1	-1.7310266e-2
N/A	N/A	-1.5654798e-1	1.5506084e-1
N/A	N/A	-1.8312649e-1	2.2198462e-2
N/A	N/A	5.3975087e-2	-1.0081810e-3
N/A	N/A	7.4914312e-2	1.3138272e-1
N/A	N/A	-1.8747096e-1	-2.8677078e-1
N/A	N/A	-4.6768560e-3	4.9899774e-1
N/A	N/A	-3.9014503e-1	-5.0336555e-1
N/A	N/A	-8.9575653e-2	2.6570242e-2
N/A	N/A	3.9632152e-1	1.3333262e-1

$$\mathbf{Z}_e^{(2)} = OM(7.4524271e-001) \quad \text{(B.17)}$$

$$\mathbf{Z}_f^{(2)} = OM(4.1035710e-001) \quad \text{(B.18)}$$

$$\mathbf{Z}_g^{(2)} = OM(1.1520603e-001) \quad \text{(B.19)}$$

$$\mathbf{Z}_h^{(2)} = OM(-5.1496339e-001) \quad \text{(B.20)}$$

Although thousands of lines of source code have been written to simulate, implement, or design all LTs mentioned in this dissertation we include a minimum fraction of source code that may be useful to implement the functions $SORT$ and OM just described. The following code is for MATLAB 4.0 .

Generation of the orthogonal matrix

```
% orthmtx(A,N)
%       Returns te NxN orthogonal matrix whose parameters are in
%       vector A. The parameters are the N(N-1)/2 plane rotation
%       angles. If some plane rotations are missing, they will be
%       assumed as 0, and any exceeding angles will be discarded.
%

function mtx = orthmtx(A,N)
L = N * (N-1) /2;
v = zeros(1,L);
l = length(A);
if (l > L)
.        v = A(1:L);
else
.        v(1:l) = A;
end
```

```
C = cos(v);
S = sin(v);
mtx = eye(N);
k = 1;
for i=1:(N-1)
.        for j=(i+1):N
.                tmp = mtx(i,:) * C(k) + mtx(j,:) * S(k);
.                mtx(j,:) =  - mtx(i,:) * S(k) + mtx(j,:) * C(k);
.                mtx(i,:) = tmp;
.                k = k + 1;
.        end
end
```

Sorting the basis functions

```
% sortbf(Hin,test)
%       Hin is a lapped transform matrix with basis functions as
%       its columns. Output is the same transform sorted by
%       frequency slots and normalized for orthonormality.
%       If test=1, perfect reconstruction is checked.

function Hout=sortbf(H,test);
[L,M] = size(H);
ha = abs(fft(H,1024));
ha = ha(1:512,:);
d = 512 / M;
rd = round(d);
for i = 0:M-1
.        k = floor(i * d);
.        li = k + 1;
.        ls = k + rd;
.        v = sum(ha(li:ls,:));
```

```
.         [y,j] = max(v);
.         order(i+1) = j;
.         ha(:,j)= zeros(512,1);
end
for i = 1:M
.         Hs(:,i) = H(:,order(i));
end
% Hs is H sorted by freq. slots
% Now, normalize bases
Hout = Hs * sqrt(M) / sum(Hs(:,1));
% Now, check PR. Show PR error only if below 50dB.
if (test==1)
.         R = prtest(Hout);
.         pmatrix = zeros(size(Hout));
.         pmatrix(1:M,1:M) = eye(M);
.         maxerr = max(max(R - pmatrix));
.         err = abs (20 * log(maxerr));
.         if (err < 50)
.                 Notice = 'Input matrix is not PR'
.                 PR_error = err
.         end
end
%

% prtest(H);
%       Test of PR property of a filter bank with basis functions
%       as columns of H. Returns a [I ; 0 ; 0 .....; 0] matrix if
%       H is PR.
%

function R=prtest(H);
```

```
[L,M] = size(H);
N = L/M;
R = [];
for l=0:(N-1)
.        S = zeros(M);
.        for i=0:(N-1-l)
.                im = i * M;
.                ilm = (i + l) * M;
.                HTH = H((im+1):(im+M),:)' * H((ilm+1):(ilm+M),:);
.                S = S + HTH;
.        end
.        R = [R ; S];
end
%
```

Utilities to transform a finite-length segment free of border distortions

Although not directly related to the presentation of the optimized angles, two tools are included to perform analysis and synthesis of finite-length segments using the general time-domain solution and symmetric extensions.

```
% transfor(H, x)
%       transform vector x into the output vector y using the lapped
%       transform described in H. Columns of H are the basis
%       functions. x and the output are column-vectors.
%

function y = transfor(H, x)
[L,M] = size(H);
if M>L, Error='Bad transform', return, end
[lx,xx] = size(x);
```

```
if xx ~= 1
.        Error='Second argument must be column vector'
.        return
end
if L > M
.        n = (L - M) / 2;
.        v = [x(n:-1:1); x ; x(lx:-1:lx-n+1)];
else
.        v = x;
end
nb = lx / M;
n = 0;
y = zeros(length(x),1);
for i = 1:nb
.        y((n+1):(n+M)) =  H' * v((n+1):(n+L));
.        n = n + M;
end
%

% transinv(H,x)
%       inverse transforming of a finite segment of a line.
%       x - must contain the finite signal (column vector)
%       H - must contain the filter bank
%               (basis functions as its columns)
%       output - reconstructed column vector
%

function xr = transinv(H,x)
% set lengths and dimensions
y=x';
[c,Nx] = size(y);
```

```
[L,M] = size(H);
N = L/M;
K = N/2;
P = H';
Nb = Nx/M;
n2 = L-M;
n = n2/2;
% inverse transform
vr=zeros(1,Nx+n2);
for i=0:(Nb-1)
.        c = i * M;
.        vr((c+1):(c+L)) = vr((c+1):(c+L)) + y((c+1):(c+M)) * P;
end
xr = vr((n+1):(n+Nx))';
% Border reconstruction
% left
F = zeros(n2,n2);
for i = 0:(N-2)
.        iM = i * M;
.        F(iM+1:iM+M,iM+1:n2) = P(:,1:n2-iM);
end
HL = F'* F;
Al = pinv(HL(:,1:n)*jay(n)+HL(:,n+1:n2));
xr(1:n) = Al * vr(1:n2)';
% right
F = zeros(n2,n2);
for i = 0:(N-2)
.        iM = i * M;
.        F(iM+1:iM+M,1:iM+M) = P(:,n2-iM+1:L);
end
HR = F'* F;
```

```
Ar = pinv(HR(:,1:n)+HR(:,n+1:n2)*jay(n));
xr(Nx-n+1:Nx) = Ar * vr(Nx+1:Nx+n2)';
```

BIBLIOGRAPHY

[1] N. Ahmed and K. R. Rao, *Orthogonal transforms for digital signal processing.* New York, NY: Springer, 1975.

[2] A. N. Akansu, R. A. Haddad, and H. Caglar, "The binomial QMF-wavelet transform for multiresolution signal decomposition," *IEEE Trans. Signal Processing*, Vol. 41, pp. 13–19, Jan. 1993.

[3] A. N. Akansu and R. A. Haddad, *Multiresolution Signal Decomposition: Transforms, Subbands, Wavelets,* San Diego, CA: Academic Press, 1992.

[4] J. L. Arrowood and M. J. T. Smith, "Exact reconstruction analysis-synthesis filter banks with time-varying filters," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Minneapolis, MN, vol. III, pp. 233–236, Apr. 1993.

[5] R. Bamberger, S. L. Eddins and V. Nuri, "Generalized symmetric extensions for size-limited multirate filter banks," *IEEE Trans. on Image Processing*, vol. 3, pp. 82–87, Jan. 1994.

[6] T. P. Barnwell III and M. J. T. Smith, "Filter Banks for Analysis-Reconstruction Systems: A Tutorial," *Proc. of IEEE Int. Symp. on Circuits and Systems,* pp. 1999-2003, 1990.

[7] B. Boashash Ed., *Time-Frequency Signal Analysis,* New York, NY: John Wiley & Sons, 1992.

[8] J. N. Bradley and V. Faber, "Perfect reconstruction with critically sampled filter banks and linear boundary conditions," Preprint.

[9] J. N. Bradley, C. M. Brislawn, and V. Faber, "Reflected boundary conditions for multirate filter banks," *Proc. of Intl. Symp. on Time-Frequency and Time-Scale Analysis,* Victoria, Canada, pp. 307–310, Oct. 1992.

[10] P. Cassereau, *A New Class of Optimal Unitary Transforms for Image Processing*, Master's Thesis, Mass. Inst. Tech., Cambridge, MA, May 1985.

[11] L. Chen, T. Q. Nguyen, and K. P. Chuan, "Symmetric extension methods for parallel M-channel perfect reconstruction linear-phase FIR analysis/synthesis systems," preprint.

[12] W. C. Chung and M. J. T. Smith, "Spatially-varying IIR filter banks for image coding," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Minneapolis, MN, vol. V, pp. 570–573, Apr. 1993.

[13] C. K. Chui (ed.), *Wavelets - A Tutorial in Theory and Applications,* San Diego, CA: Academic Press, 1992.

[14] R. J. Clarke, *Transform Coding of Images,* Orlando, FL: Academic Press, 1985.

[15] L. Cohen, "Introduction: a primer on time-frequency analysis," in *Time-Frequency Signal Analysis,* B. Boashash Ed., New York, NY: John Wiley & Sons, 1992.

[16] R. Coifman, Y. Meier, D. Quaker and V. Wickerhauser, "Signal processing and compression with wave packets." Preprint. Document files available through anonymous ftp from math.yale.edu : pub / wavelets / cmqw.tex.

[17] R. Coifman and Y. Meier, "Orthonormal wave packet bases." Preprint. Document files available through anonymous ftp from math.yale.edu : pub / wavelets / wavepkt.tex.

[18] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1983.

[19] I. Daubechies, "Orthogonal bases of compactly supported wavelets," *Comm. Pure Appl. Math.,* vol. XLI, pp. 909–996, 1988.

[20] Z. Doğanata, P. P. Vaidyanathan, and T. Q. Nguyen, "General synthesis procedures for FIR lossless transfer matrices, for perfect reconstruction multirate filter banks applications," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. 36, No. 10, pp. 1561–1574, Oct. 1988.

[21] D. F. Elliott and K. R. Rao, *Fast transforms: algorithms, analyses and applications.* New York, NY: Academic Press, 1982.

[22] D. Esteban and C. Galand, "Applications of quadrature mirror filter to split-band voice coding schemes," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Hartford, CT, pp. 191–195, 1977.

[23] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression,* Hingham, MA: Kluwer Academic, 1992.

[24] "Wavelets and Filter Banks," *Wavelets - A Tutorial in Theory and Applications,* C. K. Chui (ed.), San Diego, CA: Academic Press, 1992.

[25] R. A. Gopinath and C. S. Burrus, "Factorization approach to unitary time-varying filter banks," Tech. Report CML-TR92-93, Rice Unversity, Houston, TX, Dec. 1992.

[26] R. A. Gopinath and C. S. Burrus, "Theory of modulated filter banks and modulated wavelet tight frames," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Minneapolis, MN, vol. III, pp. 169–172, 1993.

[27] P. Haskel and D. Messerschmidt, "Reconstructing lost video data in a lapped orthogonal transform based coder," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Albuquerque, NM, pp. 1985–1988, 1990.

[28] C. Herley, J. Kovacevic, K. Ramchandran and M. Vetterli, "Arbitrary orthogonal tilings of the time-frequency plane," *Proc. of Intl. Symp. on Time-Frequency and Time-Scale Analysis,* Victoria, Canada, pp. 11–14, Oct. 1992.

[29] C. Herley, J. Kovacevic, K. Ramchandran and M. Vetterli, "Tilings of the time-frequency plane: construction of arbitrary orthogonal bases and fast tiling algorithms," *IEEE Trans. on Signal Processing,* vol. 41, pp. 3341–3359, Dec. 1993.

[30] C. Herley and M. Vetterli, "Orthogonal time-varying filter banks and wavelet packets," preprint.

[31] C. Herley and M. Vetterli, "Orthogonal time-varying filter banks and wavelets," *Proc. of IEEE Intl. Symp. on Circuits and Systems,* Chicago, IL, Vol. I, pp. 391–394 , May 1993.

[32] F. E. Hohn, *Elementary Matrix Algebra.* Second Edition, New York, NY: MacMillan, 1964.

[33] N. S. Jayant and P. Noll, *Digital Coding of Waveforms.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[34] J. D. Johnston, "A filter family designed for use in quadrature mirror filter banks," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Denver, CO, pp. 291–294, 1980.

[35] H. Jozawa and H. Watanabe, "Intrafield/Interfield adaptive lapped transform for compatible HDTV coding," *4th International Workshop on HDTV and Beyond,* Torino, Italy, Sept. 4-6, 1991.

[36] G. Karlsson and M. Vetterli, "Extension of finite length signals for subband coding," *Signal Processing,* Vol. 17, pp. 161–168, 1989.

[37] H. Kiwa, K. Nishikawa, and M. Iwahashi, "A development of symmetric extension method for subband image coding," *IEEE Trans. on Image Processing*, vol. 3, pp. 78–81, Jan. 1994.

[38] R. D. Koilpillai and P. P. Vaidyanathan, "Cosine modulated FIR filter banks satisfying perfect reconstruction," *IEEE Trans. Signal Processing*, Vol. 40, pp. 770–783, Apr. 1992.

[39] Y. Kovacevic and M. Vetterli, "Design of multidimensional non-separable regular filter banks and wavelets," *Proc. of IEEE Intl. Conf. on Acoust., Speech, Signal Processing*, San Francisco, CA, vol. IV, pp. 389–392, 1992.

[40] D. Le Gall, and A. Tabatabai "Sub-band coding of digital images using symmetric short kernel filters and arithmetic coding techniques" *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, pp. 761–764, 1988.

[41] G. Longo, and P. Picinbono ed., *Time and Frequency Representation of Signals and Systems,* CISM Courses and Lectures No. 309, New York, NY: Springer-Verlag, 1989.

[42] S. G. Mallat, "Multifrequency channel decomposition of images and wavelet models," *IEEE Trans. on Acoustics, Speech and Signal Processing* Vol. 37, pp. 2091–2110, Dec. 1989.

[43] H. S. Malvar, "Reduction of blocking effects in image coding with a lapped orthogonal transform," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Glasgow, Scotland, pp. 781-784, Apr. 1988.

[44] H. S. Malvar and D. H. Staelin, "The LOT: transform coding without blocking effects," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-37, pp. 553–559, Apr. 1989.

[45] H. S. Malvar, "Lapped transforms for efficient transform/subband coding," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-38, pp. 969–978, June 1990.

[46] H. S. Malvar, "Efficient signal coding with hierarchical lapped transforms," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Albuquerque, NM, pp. 761–764, 1990.

[47] H. S. Malvar, "Modulated QMF filter banks with perfect reconstruction," *Elect. Letters,* vol.26, pp. 906-907, June 1990.

[48] H. S. Malvar, "Extended lapped transform: fast algorithms and applications," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Toronto, Canada, pp. 1797–1800, 1991.

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

[50] H. S. Malvar, "Fast computation of wavelet transforms with the extended lapped transform," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, San Francisco, CA, vol. IV, pp. 393–396, 1992.

[51] H. S. Malvar, "Improving the time resolution of wavelet transforms," preprint.

[52] H. S. Malvar, "Extended lapped transforms: properties, applications and fast algorithms," *IEEE Trans. Signal Processing,* vol. 40, pp. 2703–2714, Nov. 1992.

[53] S. Martucci, "Signal extension and noncausal filtering for subband coding of images," *SPIE VCIP'91: Visual Communication,* vol. 1605, pp. 137–148, Nov. 1991.

[54] S. Martucci, "Convolution-multiplication properties for the entire family of discrete sine and cosine tranforms," *Proc. 1992 CISS,* Princeton, NJ, Mar. 1992,

[55] K. Nayebi, T. P. Barnwell and M. J. Smith, "The time domain analysis and design of exactly reconstructing FIR analysis/synthesis filter banks," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Albuquerque, NM, pp. 1735-1738, 1990.

[56] K. Nayebi, T. P. Barnwell and M. J. Smith, "The time domain filter bank analysis: a new design theory," *IEEE Trans. on Signal Processing*, vol. 40, pp. 1412–1429, June 1992.

[57] K. Nayebi, T. P. Barnwell and M. J. Smith, "Analysis-synthesis systems with time-varying filter bank structures," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, San Francisco, CA, vol. IV, pp. 617–620, 1992.

[58] A. N. Netravali and B. G. Haskell, *Digital Pictures, Representation and Compression,* New York, NY: Plenum Press, 1988.

[59] T. Q. Nguyen and P. P. Vaidyanathan, "Strutures for M-channel perfect reconstruction FIR QMF banks which yield linear phase analysis filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol 38, pp. 433-446, March 1990.

[60] T. Q. Nguyen and P. P. Vaidyanathan, "Two-channel perfect reconstruction FIR QMF structures which yield linear phase analysis and synthesis filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-37, pp. 676-690, May 1989.

[61] T. Q. Nguyen and R. D. Koilpillai, "The design of arbitrary-length cosine-modulated filter banks and wavelets satisfying perfect reconstruction," preprint.

[62] T. Q. Nguyen, "Digital filter banks design - the quadratic-constrained formulation," preprint.

[63] T. Q. Nguyen, "A quadratic constrained least-square approach to the design of digital filter banks," *Intl. Symp. on Circuits and Systems,* vol. 3, pp. 1344–1347, San Diego, CA, May 1992.

[64] V. Nuri, and R. Bamberger, "A theorectical framework for the analysis and design of size-limited multirate filter banks," *Proc. of Intl. Symp. on Time-Frequency and Time-Scale Analysis,* Victoria, Canada, Oct. 1992.

[65] V. Nury, and R. Bamberger, "A theory of size-limited filter banks," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, vol. III, pp. 161–164, 1993.

[66] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing,* Englewoods Cliffs, NJ : Prentice-Hall, 1975.

[67] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing,* Englewoods Cliffs, NJ : Prentice-Hall, 1989.

[68] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Compression Standard,"* New York, NY: Van Nostrand Reinhold, 1993.

[69] J. Pesquet, "Orthonormal wavelets for finite sequences," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, vol. IV, pp. 609–612, 1992.

[70] J. P. Princen and A. B. Bradley, "Analysis/synthesis filter bank design based on time domain aliasing cancellation," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-34, pp. 1153–1161, Oct. 1986.

[71] R.L. de Queiroz and K. R. Rao, "Time-varying lapped transforms and wavelet packets," *IEEE Trans. on Signal Processing,* vol. 41, pp. 3293–3305, Dec. 1993.

[72] R. L. de Queiroz and K. R. Rao, "A structure for time-varying paraunitary filter banks with perfect reconstruction," *Electronics Letters,* Vol. 29, pp. 217–218, Jan. 1993.

[73] R. L. de Queiroz and H. S. Malvar, "On the asymptotic performance of hierarchical transforms," *IEEE Trans. on Signal Processing,* Vol 40, pp. 2620–2622, Oct. 1992.

[74] R. L. de Queiroz and K. R. Rao, "HVS weighted progressive transmission of images using the LOT," *Journal of Electronic Imaging,* vol. 1, pp. 328–338, July 1992.

[75] R.L. de Queiroz and K. R. Rao, "Adaptive extended lapped transforms," *Proc. of IEEE Intl. Conf. on Acoust., Speech, Signal Processing*, Minneapolis, MN, Vol. III, pp. 217–220, Apr. 1993.

[76] R. L. de Queiroz and K. R. Rao, "On adaptive wavelet packets," *Proc. of IEEE Intl. Symp. on Circuits and Systems,* Chicago, IL, Vol. I, pp. 511-514, May 1993.

[77] R. L. de Queiroz, "Subband processing of finite length signals without border distortions," *Proc. of IEEE Intl. Conf. on Acoust., Speech, Signal Processing*, San Francisco, CA, vol. IV, pp. 613–616, 1992.

[78] R. L. de Queiroz and K. R. Rao, "Modulated lapped transforms for image coding," *SPIE Conf. on Video Compression for Personal Computers,* vol. 2187, pp. 80–91, San Jose, CA, Feb. 1994.

[79] R. L. de Queiroz, "Chooosing a subband system for image coding thru a cost-compaction criteria," in portuguese, *Proc. of Simposio Brasileiro de Telecomunicacoes,* Sao Paulo, Brazil, pp. 2.3.11–2.3.15, Sep. 1991.

[80] R. L. de Queiroz and K. R. Rao, *Transform Coding,* Chap. 7 in *Handbook on Visual Communications*, Edited by J. W. Woods and H. M. Hang, Academic Press, 1994.

[81] R. L. de Queiroz and K. R. Rao, "The extended lapped transform for image coding," to appear in *IEEE Trans. on Image Processing.*

[82] R. L. de Queiroz and K. R. Rao, "Reconstruction methods for processing finite-length signals with paraunitary filter banks," to appear in *IEEE Trans. on Signal Processing.*

[83] R. L. de Queiroz, T. Q. Nguyen, and K. R. Rao, "The class of GENLOTs: generalized linear-phase lapped orthogonal transforms," *Proc. IEEE Intl. Symp. Circuits and Systems,* London, UK, 1994.

[84] R. L. de Queiroz, T. Q. Nguyen, and K. R. Rao, "The generalized lapped orthogonal transforms," *Electronics Letters,* vol. 30, pp. 107–107, Jan. 1994.

[85] R. L. de Queiroz, T. Q. Nguyen, and K. R. Rao, "GenLOT: generalized linear-phase lapped orthogonal transforms," to appear in *IEEE Trans. on Signal Processing.*

[86] R. L. de Queiroz, and K. R. Rao, "On orthogonal transforms of images using paraunitary filter banks," preprint.

[87] M. Rabbani and P.W. Jones, *Digital Image Compression Techniques,* SPIE Optical Engineering Press, Bellingham, WA, 1991.

[88] K. R. Rao and P. Yip *Discrete Cosine Transform : Algorithms, Advantages, Applications,* San Diego, CA : Academic Press, 1990.

[89] K. R. Rao (ed.), *Discrete transforms and their applications,* New York, NY: Van Nostrand Reinhold, 1985.

[90] K. Ramchandran and M. Vetterli, "Best wavelet packet bases using rate-distortion criteria," *Proc. of Intl. Symp. on Circuits and Systems,* San Diego, CA, vol. II, pp. 971–974, Apr. 1992.

[91] O. Rioul and M. Vetterli, "Wavelets and signal processing," *IEEE Signal Processing Magazine,* vol. 8, pp. 14–38, Oct. 1991.

[92] E. M. Rubino, H. S. Malvar and R. L. de Queiroz, "Improved Chen-Smith image coder," *Proc. of IEEE Intl. Symp. Circuits and Systems,* Chicago, IL, Vol. I, pp. 267–270, May 1993.

[93] E. M. Rubino, R. L. de Queiroz, and H. S. Malvar , "Improved Chen-Smith image coder," preprint.

[94] R. W. Schafer and L. R. Rabiner, "A Digital Signal Processing Approach to Interpolation," *Proc. IEEE,* vol 61, June 1973.

[95] H. Schiller, "Overlapping block transform for image coding preserving equal number of samples and coefficients," *Proc. SPIE, Visual Communications and Image Processing,* vol. 1001, pp. 834–839, 1988.

[96] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Trans. Signal Processing,* vol. 41, pp. 3445–3462, Dec. 1993.

[97] M. J. Smith and T. P. Barnwell, 'Exact reconstruction techniques for tree-structured subband coders', *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-34, pp. 434–441, June 1986.

[98] M. J. Smith and S. L. Eddins, "Analysis/synthesis techniques for subband image coding," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 1446–1456, Aug. 1990.

[99] I. Sodagar, K. Nayebi, and T. P. Barnwell, "A class of time-varying wavelet transforms," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, Minneapolis, MN, vol. III, pp. 201–204, Apr. 1993.

[100] A.K. Soman and P.P. Vaidyanathan, "Paraunitary filter banks and wavelet packets," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, vol. IV, pp. 397–400, 1992.

[101] A. K. Soman, P. P. Vaidyanathan, and T. Q. Nguyen, "Linear-phase orthonormal filter banks," *Proc. of IEEE Intl. Conf. on Acoust., Speech, Signal Processing*, Minneapolis, MN, Vol. III, pp. 209–212, Apr. 1993.

[102] A. K. Soman, P. P. Vaidyanathan, and T. Q. Nguyen, "Linear-phase paraunitary filter banks: theory, factorizations and applications," *IEEE Trans. on Signal Processing,* vol. 41, pp. 3480–3496, Dec. 1993.

[103] M. Temerinac and B. Edler, "A unified approach to lapped orthogonal transforms," *IEEE Trans. Image Processing*, vol. 1, pp. 111–116, Jan. 1992.

[104] P.P. Vaidyanathan, *Multirate Systems and Filter Banks.* Englewood Cliffs, NJ: Prentice-Hall, 1993.

[105] P. P. Vaidyanathan and P. Hoang, "Lattice structures for optimal design and robust implementation of 2-channel PR-QMF banks," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-36, pp. 81–94, Jan. 1988.

[106] P. P. Vaidyanathan, "Multirate digital filters, filter banks, polyphase networks and applications: a tutorial," *Proc. IEEE,* vol. 78, pp. 56-93, 1990.

[107] P. P. Vaidyanathan, "Theory and design of M-channel maximally decimated quadrature mirror filters with arbitrary M having perfect reconstruction property," *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-35, pp. 476–492, Apr. 1987.

[108] M. Vetterli and C. Herley, "Wavelets and filter banks: theory and design," Tech. Rep. 206/90/36, CTR, Columbia University, 1990.

[109] M. Vetterli and C. Herley, "Wavelets and filter banks: theory and design," *IEEE Trans. Signal Processing*, Vol. 40, pp. 2207–2232, Sep. 1992.

[110] M. Vetterli and D. Le Gall, "Perfect reconstruction filter banks : some properties and factorizations", *IEEE Trans. Acoust., Speech, Signal Processing*, ASSP-37, pp. 1057–1071, July 1989.

[111] M. Vetterli, "A theory of multirate filter banks," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 356-372, March 1987.

[112] P. H. Westerink, *Sub-band Coding of Images,* Ph.D Thesis, Delft Technische Universiteit, The Netherlands, Oct. 1989.

[113] P. H. Westerink, J. Biemond, and D. E. Boeckee, "Progressive Transmission of Images Using Sub-band Coding," *Proc. of Intl. Conf. on Acoust., Speech, Signal Processing*, pp. 1811- 1814, 1989.

[114] J. W. Woods and S. D. O'Neil, "Sub-band coding of images," *IEEE Trans. Acoust., Speech, Signal Processing*, vol ASSP-34, no.5, pp. 1278–1288, Oct. 1986.

[115] J. W. Woods (Ed.), *Subband Coding of Images.* Hingham, MA: Kluwer Academic, 1991.

[116] R. W. Young and N. G. Kingsbury, "Frequency domain estimation using a complex lapped transform," *IEEE Trans. Image Processing,* vol. 2, pp. 2–17, Jan. 1993.

[117] Final text for ISO/IEC DIS 10918-1. Info. Technology - Digital compression and coding of continuous tone still images - Part 1: Requirements and guidelines, Jan. 14, 1992. Part 2: Compliance testing - CD 10918-2, 12/16/91.

[118] Recommendation H. 261 - Video codec for audiovisual services at p×64 Kbit/s. CCITT, COM-XV-R37-E, August 1990.

[119] ISO 11172, "Coding of moving pictures and associated audio for digital storage media at up to 1.5 MBPS". ISO/IEC JTC, 29N071, Dec. 6, 1991.

[120] CCIR Rec. 601, "Encoding parameters of digital television for studios," CCIR Tec. Rep. Intl. Telecommunications Union, vol. XI – part 1, Plenary Assembly, Geneva, Switzerland, 1982.