DIRECTIONAL IMAGE REPRESENTATIONS USING NONSEPARABLE LIFTING

BY

JOSHUA P. BLACKBURN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Adviser:

Associate Professor Minh N. Do

ABSTRACT

Sparse representations of visual information are essential for many image processing tasks. Because of the nonstationary geometric structure of natural images, representations derived from one-dimensional tensor products or compact frequency support will be suboptimal. Therefore, there is strong motivation to search for more powerful methods to efficiently represent the geometric structure of visual information.

This thesis demonstrates a method to create a directional image representation with compact spatial support which is not limited to a single dimension. Within the lifting framework of perfect reconstruction filter banks, sparse representation requires prediction filters able to adapt to the local structure of the signal. As most images are locally regular except at edges, this adaptation adjusts the support of the prediction filters in order that a larger percentage of the output is predicted from pixels which do not come from both sides of an edge.

To allow for the adaptation of filter support, the image must be segmented into blocks of consistent directional bias. To allow sufficient adaptivity while reducing overhead, this segmentation must allow for multiple sizes of blocks dependent on the image data. We solve this problem by extending a classic tree pruning algorithm used in classification for adaptive block-based transforms. Furthermore, as images do not directly include directional information, we propose a weighted estimation method using the techniques of directional statistics to determine the dominant direction of an image block.

Within a compression framework, we see that the directional estimation and adaptive segmentation algorithms robustly and accurately determine the dominant direction of variably sized blocks; however, because of limitations caused by the discrete nature of the data and dimensional degeneracy of polynomial interpolation over various point sets, our directional image representation was not able to provide a coding gain over traditional methods.

To Betsy, my love

ACKNOWLEDGMENTS

I would like to show my gratitude to my adviser Minh Do. His knowledge and passion for directional representations motivated and advanced my research. His instruction has provided the skills I needed to excel as a researcher. I am also indebted to my many colleagues for their generous assistance and beneficial discussions, especially Dan Kubacki, Quang Nguyen, André Targino, Alex Dapore, Hien Nguyen, Raman Singh, and Huy Bui.

I would like to thank my parents for their many years of inspiring me to shoot for the stars. Finally, my deepest appreciation goes to my wife Betsy. Her comfort and encouragement were instrumental in both the research and writing of this thesis.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION 1
1.1 Motivation 1
1.2 Problem Statement 4
1.3 Related Work 5
1.4 Thesis Summary 10

CHAPTER 2 GEOMETRIC LIFTING 12
2.1 Orthonormal Lifting Structures 13
2.2 Directional Support 16
2.3 Prediction Filters 21
2.4 Update Filters and Scaling 25
2.5 Geometric Filter Dictionary 26

CHAPTER 3 ADAPTIVE SEGMENTATION 28
3.1 Optimal Tree Pruning Theory 28
3.2 Optimal Tree Pruning Implementation 30
3.3 Block Boundary Issues 32

CHAPTER 4 DIRECTION ESTIMATION 36
4.1 Theory of Directional Statistics 36
4.2 Directional Statistics for Edge Direction Estimation 38
4.3 Edge Direction Estimation for Filter Selection and Adaptive Segmentation 43

CHAPTER 5 RESULTS 46
5.1 Direction Estimation and Adaptive Segmentation 46
5.2 Nonlinear Approximation 48

CHAPTER 6 DIMENSIONAL DEGENERACY 55
6.1 Empirical Results 55
6.2 Theoretical Results 55

CHAPTER 7 CONCLUSION 61

APPENDIX A FILTER DIAGRAMS 64

REFERENCES 125

CHAPTER 1

INTRODUCTION

1.1 Motivation

One of the main challenges in signal processing and mathematical analysis is signal representation and approximation. If there exists some representation of the signal which is sparse, many signal processing operations become simpler. Compression is achieved merely by transforming the signal into the sparse representation and only storing the few nonzero coefficients. If noise is present in the signal such that the noise is not sparse under this representation, denoising can be accomplished simply by forcing the signal to be sparse under this representation. Finally, if the sparse coefficients have a subband structure, interpolation can be implemented as the inverse representation of the zero-padded known data.

The goal of a sparse representation is often to determine a set of basis functions such that a class of signals can be written as a linear combination of a small subset of the basis functions. More formally, consider a class of functions $\mathcal{F}$ and a function f such that $f \in \mathcal{F}$. Assume that there exists a set of basis functions $\{\phi_m\}_{m\in\mathbb{N}}$ such that

$$f = \sum_{m\in\mathbb{N}} c_m \phi_m$$

for all $f \in \mathcal{F}$. If the set $\{\phi_m\}_{m\in\mathbb{N}}$ forms an orthonormal set, this can be rewritten as

$$f = \sum_{m\in\mathbb{N}} \langle\, f, \phi_m \rangle\, \phi_m$$

where

$$\langle\, f, \phi_m \rangle = \int_{-\infty}^{\infty} f(x)\phi_m^*(x)\mathrm{d}x.$$

Signals written in this form can be approximated by M coefficients as

$$\hat{f} = \sum_{m \in \mathcal{I}_M} \langle f, \phi_m \rangle \, \phi_m$$

where $\mathcal{I}_M$ is an index set with M elements [1]. The error is thus

$$\epsilon_M = \sum_{m \notin \mathcal{I}_M} \langle f, \phi_m \rangle \, \phi_m.$$

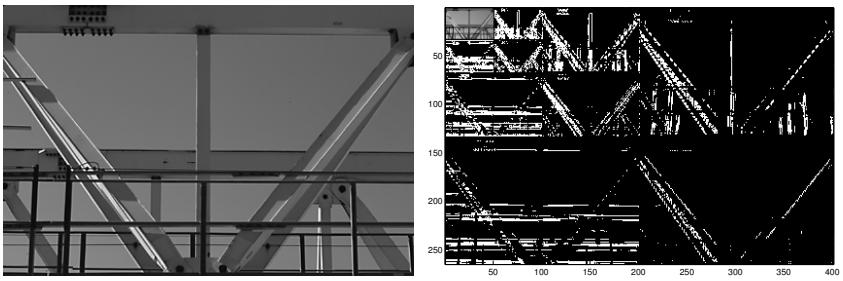
There are two main types of approximation: linear and nonlinear. Linear approximation creates the index set $\mathcal{I}_M = \{1, \ldots, M\}$. Linear approximation maintains all of the properties of linearity, such as superposition. Nonlinear approximation creates the set $\mathcal{I}_M$ from the M largest $|\langle f, \phi_m \rangle|$ [1]. This is not a linear approximation because two approximations need not use the same set of M vectors. The nonlinear approximation error will necessarily be smaller than or equal to the linear approximation error because larger or equal basis elements were used.

One of the bases with the the longest history is the Fourier basis. The Fourier basis is useful because it can represent any square-summable function and because it diagonalizes linear, shift-invariant operators [1]. Unfortunately, the nonlinear approximation error for signals with discontinuities is $O(M^{-1})$ [1].

Over the past twenty years, wavelet bases have become extremely important for signal approximation. Wavelet bases improve upon the Fourier bases because they are scaled to form a multiresolution approximation and can be compactly supported. Given a mother wavelet ϕ and a sequence of function spaces $\mathcal{V}_j \subset \mathcal{F}$ satisfying the multiresolution requirements, the functions

$$\phi_{j,k}(x) = 2^{-j/2} \phi(2^{-j} x - k) \tag{1.1}$$

form an orthonormal basis for $\mathcal{V}_j$ [2]. For signals which have a finite number of discontinuities and are uniformly Lipschitz of order s between these discontinuities, the nonlinear approximation by a wavelet basis with more than s vanishing moments is $O(M^{-2s})$ [2, Proposition 9.4]. If the signal is not continuous, but has bounded variation, then the wavelet approximation is $O(M^{-2})$ [2, Proposition 9.5]. Furthermore, the approximation error decay rate of wavelets for signals with bounded variation is equal to or better



(a) Natural Image (b) Wavelet Coefficient Locations

Figure 1.1: Wavelet transform of an image. The transformed image shows the locations of the largest 10% of the detail coefficients after the 9-7 wavelet transform. Notice the spatial correlation of the large coefficients along the edges.

than the approximation error decay rate for optimal spline approximations and the nonlinear approximation created by any orthonormal transform [2]. Therefore, wavelets are essentially optimal for bounded variation signals.

The separable extension of wavelets through tensor products does not demonstrate a similar optimality. If f is a two-dimensional discontinuous function with bounded total variation, the error of separable wavelet approximations only decays as $O(M^{-1})$ [2, Theorem 9.8]. This result is also true more generally for signals which are piecewise regular, but have discontinuity curves [2].

The difference between wavelet approximations in one dimension and in two dimensions is that two-dimensional signals can have one-dimensional singularity curves, while one-dimensional signals can only have point singularities. Wavelets can approximate the discontinuity orthogonal to the edge fairly well; however, they cannot exploit the smoothness along the edge [1]. This can be clearly seen in Figure 1.1 in the spatial correlation present in the detail bands of a typical image transformed into a wavelet basis. The wavelet basis assumed that each pixel on the edge needed its own large coefficient to represent the singularity instead of providing a mechanism to approximate the entire curve with few coefficients.

These singularity curves present in images are not some mathematical oddity that can easily be ignored. Rather, the presence of smooth edges is one of the main defining features of the geometry of natural images. Therefore, two-dimensional representations which are derived from separable one-dimensional representations or which do not account for these singularity

curves will not be able to fully exploit the information present in the signal. A class of signals which better represents the discontinuity curves present in natural images is the set of functions which are in $\mathcal{C}^2$ except for the presence of discontinuity curves which are also $\mathcal{C}^2$. For this class of signals, the lower bound on the approximation error decay rate is $O(M^{-2})$ [3]; however, this rate has not yet been achieved.

1.2 Problem Statement

The previous discussion presents a compelling motivation for the further creation of representations which account for the geometry of natural images. The geometry of natural images is dominated by three factors. First, natural images are not stationary; information in one region of an image is typically unrelated to information in another region. Therefore, all processing must be local. Second, except at singularity curves, images tend to be isotropically regular, which implies that neighboring pixels should be able to predict any pixel effectively. Furthermore, this prediction will be most effective if the prediction includes information from multiple sides of the predicted pixel rather than just along the scanlines. Finally, there exist singularity curves across which neighboring pixels are unrelated. In a local region, a singularity curve can be linearly approximated as a direction. By adjusting to these directions, the prediction will become more effective as fewer unrelated pixels are included. Using directionality as an integral part of representation is also supported from the field of physiology. Studies in primate vision systems have demonstrated that the response of cells in the visual cortex is dominated by cells with a directional structure [4].

Our goal is to create a new image representation with the following characteristics:

Discrete: The representation should be derived in the discrete domain to facilitate fast algorithms.

Multiresolution: The representation should create a subband structure that facilitates multiple resolutions of representation.

Sparse: The number of nonzero coefficients for natural images expressed in

this representation should be small. Furthermore, the amount of overhead required by any representation adaptivity should also be small.

Local: The representation should have basis functions with small support which are derived in the spatial domain to account for the spatial nonstationarity of natural images.

Nonseparable: The bases should not be tensor products of one-dimensional bases as optimality of one-dimensional approximation does not imply optimal two-dimensional approximation.

Directional: The bases should be anisotropic and align with directions beyond those provided by separable wavelets.

1.3 Related Work

Researchers in many areas of signal approximation have created representations for images as a means to improve upon the results of separable wavelets. These representations fall into three main areas. The first area is geometric wavelets, which uses frequency domain constructions to create directional representations similar to wavelets. The second area is separable directional lifting, which applies spatial domain constructions along arbitrary one-dimensional axes. The third area is multidimensional adaptive transforms, which create nonseparable transforms adapted to the signal, but which are not adapted in a directional manner.

1.3.1 Geometric Wavelets

Directionlets, developed by Velisavljević et al., applies a one-dimensional wavelet transform along two independent directions, which need not be along the scanlines [5, 6]. This method has vanishing moments along these two directions while retaining separable filtering. Directionlets have an approximation error decay rate of $O(M^{-s})$ where $s = \frac{1}{2}(\sqrt{17} - 1) \approx 1.55$ [5]. Furthermore, using directionlets in a transform coding application provides a peak signal-to-noise ratio (PSNR) gain of 0.1–0.8 dB over traditional space frequency coding [6].

Ridgelets, created by Candès and Donoho, approximate a signal by ridge functions that are constant in some direction and smooth in the perpendicular direction [7]. They defined a ridgelet in terms of a smooth mother wavelet as

$$\phi_{a,b,\theta}(x,y) = a^{-1/2}\phi\left(\frac{x\cos\theta + y\sin\theta - b}{a}\right).$$

The ridgelet transform is equivalent to applying a Radon transform followed by a one-dimensional wavelet transform. The Radon transform maps linear singularities into point singularities, for which wavelets are optimal [1].

Curvelets, also created by Candès and Donoho, extend ridgelets to allow for curve singularities [8, 3]. The initial formulation of curvelets was a direct extension to ridgelets by breaking curve singularities into a collection of line singularities and filtering by the local ridgelet transform [9]. The second formulation bypassed ridgelets to directly replace (1.1) with a two-dimensional mother curvelet ϕ with a two-scale relation

$$\phi_{j,l,\boldsymbol{k}}(\boldsymbol{x}) = 2^{3j/2}\phi\left(\boldsymbol{D}_j\boldsymbol{R}_{\theta_{j,l}}\boldsymbol{x} - \boldsymbol{k}\right)$$

where j is the scale parameter, $\boldsymbol{D}_j$ is the parabolic scaling matrix

$$\boldsymbol{D}_j = \begin{bmatrix} 2^{2j} & 0 \\ 0 & 2^j \end{bmatrix},$$

l is an orientation parameter, $\boldsymbol{R}_{\theta_{j,l}}$ is a rotation matrix oriented at $\theta_{j,l} = 2\pi\cdot 2^{-j}\cdot l$, and $\boldsymbol{k}$ is a translation parameter. Curvelets have an approximation error decay rate of $O(M^{-2}(\log M)^3)$ [3]. This approximation error rate is close to the approximation error rate for nonsingular images and is much better than the error decay rate of separable wavelets for large M. On the other hand, curvelets use polar coordinates which are difficult to discretize. In addition, their space-frequency and directional localizations are constructed in the frequency domain, which leads to long spatial support and Gibbs oscillations.

Shearlets, developed by Guo and Labate [10] and extended to a discrete frame by Lim [11], are similar to curvelets except that they use a shearing operator instead of a rotation operator. Shearlets replaces the mother wavelet

with a mother shearlet which satisfies

$$\phi_{j,l,\boldsymbol{k}}(\boldsymbol{x}) = |\det \boldsymbol{A}|^{j/2}\phi(\boldsymbol{B}^l\boldsymbol{A}^j\boldsymbol{x} - \boldsymbol{k})$$

where $\boldsymbol{A}$ is a diagonal scaling matrix, $\boldsymbol{B}$ is a shearing matrix, and $\boldsymbol{k}$ is a translation vector. Shearlets satisfy the same approximation error decay rate as curvelets of $O(M^{-2}(\log M)^3)$. Shearlets do not have the discretization problems of curvelets; however, they still suffer from long spatial support and Gibbs oscillations.

Contourlets, developed by Do and Vetterli, differ from curvelets and shearlets in that the initial formulation uses discrete filter banks instead of continuous bases [12]. They use a Laplacian pyramid to create a multiresolution representation of the signal. Each bandpass segment is further processed using a directional filter bank to select contours along specific directions. Similar to curvelets and shearlets, contourlets have an approximation error decay rate of $O(M^{-2}(\log M)^3)$. Contourlets solve the discretization problem of curvelets by using a discrete grid; however, they still suffer from long spatial support and Gibbs oscillations.

Bandlets, developed by Mallat and Peyré, differ from the previous transforms in that their bases adapt to the structure of the signal [13, 14]. The bandlet transform starts with an ordinary wavelet transform. Given the parametric representation of the contours in the detail bands $x_2 = \gamma(x_1)$, the coefficients are warped such that $(1, \gamma'(x_1))$ is aligned with the horizontal axis. The coefficients are then approximated along this axis by a polynomial. The approximation error decay rate is $O(M^{-s}(\log M)^s)$ for two-dimensional functions in $\mathcal{C}^s$ except for discontinuities along $\mathcal{C}^s$ curves. This transform has a better approximation error decay rate than the previous methods and can be extended to higher orders signals. On the other hand, it requires an edge detection step to determine γ and will need to retain index information to properly reconstruct the signal.

1.3.2 Separable Directional Lifting

An alternative construction and extension to traditional wavelets is lifting [15]. Daubechies and Sweldens demonstrated that any one-dimensional finite impulse response (FIR) wavelet transform can be factored into a fi-

nite number of lifting steps [16]. Lifting works by splitting a signal into polyphase components, for example split $\boldsymbol{x} = (x_n)_{n\in\mathbb{Z}}$ into $\boldsymbol{x}_e = (x_{2n})_{n\in\mathbb{Z}}$ and $\boldsymbol{x}_o = (x_{2n+1})_{n\in\mathbb{Z}}$. Because these components tend to be highly correlated, one set should form a good predictor of the other set. Specifically, the difference

$$\boldsymbol{d} = \boldsymbol{x}_o - \mathcal{P}(\boldsymbol{x}_e) \tag{1.2}$$

should be sparse. This difference signal can then update the first polyphase component of the form

$$\boldsymbol{c} = \boldsymbol{x}_e + \mathcal{U}(\boldsymbol{d}). \tag{1.3}$$

Notice that given $(\boldsymbol{c}, \boldsymbol{d})$, $(\boldsymbol{x}_e, \boldsymbol{x}_o)$ can be trivially obtained for any choice of $\mathcal{P}$ and $\mathcal{U}$ by rearranging (1.2) and (1.3).

As lifting is mathematically identical to wavelets for one-dimensional FIR wavelet filters, separable extensions of lifting to higher dimensions exhibit the same problems as separable extensions of wavelets. Specifically, they cannot sparsely represent singularity curves.

Multiple researchers in the area of directional lifting have attempted to improve upon the error rate of wavelet transforms. In this area, the standard error metric is the gain of the peak signal-to-noise ratio (PSNR) of various test images over traditional wavelets.

Gerek and Çetin extended the standard lifting procedure for the 5/3 wavelet to diagonal directions [17]. They observed that the best prediction of a value from its two neighbors may not come from the two neighbors in the current scanline. Instead, they predict the value using the current scanline or one of the two diagonals using an edge orientation estimator. For a series of test images, their method demonstrated 0.04–0.14 dB improvement in PSNR over the 5/3 wavelet. On the other hand, these results only allow for angles along $\pm 45°$.

Chappelier and Guillemot perform a similar extension by using quincunx sampling and predicting along the scanlines [18]. For a series of test images, their method demonstrated 0.3-1.0 dB improvement over separable wavelets in JPEG-2000. Similar to [17], this method only allows for a few directions.

Ding et al. extended the standard lifting procedure to operate along an arbitrary direction [19]. By aligning the lifting direction with strong correlation, the prediction should be more accurate than merely predicting along the scanlines. As pixel values are not available at arbitrary locations, they

use sinc interpolation to generate these extra values. The PSNR gain of their system over JPEG-2000 ranged from 0.21 to 1.36 dB. This work is limited by isotropically using sinc interpolation for subpixel values.

Liu and Ngan improved upon [19] by weighting multiple pixels from the even polyphase component [20]. If the direction of strong correlation intersects a pixel value, only that single value is used. If the direction is at a subpixel location, the weighting reduces to an interpolating filter. They further improve upon [19] by aligning the interpolation filter with the direction of strong correlation. The PSNR gain of their system over JPEG-2000 ranged from 0.19 to 3.06 dB for the 5/3 filter and 0.19 to 1.79 dB for the 9/7 filter. This work is limited by using a two-step procedure to interpolate subpixel locations, then predict the current pixel from these values.

Chang and Girod also developed a method to perform standard lifting along an arbitrary direction [21]. Unlike [19, 20], they do not use subpixel interpolation. Instead, they continue along the direction until a pixel value is reached. The PSNR gain of their system over a system similar to JPEG-2000 ranged from 0.4 to 2.5 dB with one image at 5.4 dB. On the other hand, this method only uses values along a single orientation to determine the prediction filter.

In addition, the reliance of all of these methods upon separable one-dimensional transforms demonstrates two shortcomings. First, as an extension from the separable lifting scheme, the existing directional lifting schemes still downsample along each dimension sequentially. It is unclear what the final transform coefficients are in the equivalent two-dimensional filters. Due to the concatenation of lifting stages for each dimension, the final equivalent two-dimensional filters might have large or isotropic support, which limits the transform's ability to approximate local and geometric regularities. Second, by considering images as a stack of independent smooth one-dimensional slices, directional lifting still only exploits one-dimensional regularity along the edge direction. Nevertheless, with subpixel interpolation and concatenation of lifting along each dimension, existing directional lifting schemes implicitly exploit smoothness along all dimensions.

1.3.3 Nonseparable Adaptive Lifting

Two methods avoid these issues by designing nonseparable filters. Benazza-Benyahia et al. create nonseparable filters in three dimensions across two spatial dimensions and one spectral dimension [22]. Their method finds the filter coefficients using maximum likelihood estimation on the prediction residuals, assuming a generalized Gaussian distribution. The support of the filter is constant across the entire signal, which does not allow it to adapt to the geometric regularity contained within the signal.

Quellec et al. create nonseparable filter banks using a vanishing moment constraint with additional degrees of freedom in order to optimize a higher level criterion [23]. They optimize these extra degrees of freedom for the higher level criterion, then ensure that the vanishing moment constraint is met. However, there is no method to ensure that the support of the filter is directional.

1.4 Thesis Summary

The remainder of this thesis is structured as follows. In Chapter 2, we discuss the orthogonality properties of single-stage, multi-channel lifting structures and demonstrate the design of geometric filters. This lifting structure is constrained such that, except in trivial cases, it cannot be orthonormal. Therefore, we propose a measure of divergence of a filter bank from orthonormality as an optimization criterion. We then focus on creating directional support for the prediction filters and designing the filters such that the analysis filter bank has two vanishing moments. Finally, we consider various methods to determine the update filters and scaling and determine which method minimizes the divergence from orthonormality for the filter bank. The various filters are then packaged into a geometric filter dictionary.

Chapter 3 explores the BFOS tree pruning algorithm for adaptive segmentation of blocked transforms. We present the theoretical requirements for the rate and distortion functions and develop a recursive algorithm to find the optimal segmentation. We then discuss the problem of block boundary conditions within the framework of a lifting scheme and propose a method to minimize their impact.

Chapter 4 presents a method to determine the dominant direction of the

edges within an image block. We review the theory of empirical estimation of mean and variance for angular random variables. We extend this theory to robustly determine the dominant edge direction within an image block and an associated confidence measure. These estimates are then integrated into the segmentation algorithm.

Chapter 5 contains results of the proposed system. The first set of results tests the directions estimation and adaptive segmentation algorithms. For a simple class of synthetic images, Monte Carlo experiments demonstrate that the estimate is unbiased with a circular variance of 0.058. On more realistic synthetic and natural images, the proposed algorithm subjectively seems to choose reasonable directions and segment into the largest regions of consistent directional bias. The second set of results tests the nonlinear approximation of the proposed algorithm by comparison with the 5/3 wavelet. While the proposed transform has a gain of 2–4.5 dB in peak signal-to-noise ratio over the 5/3 wavelet for synthetic images, it has a degradation of 1–2.5 dB for natural images.

Chapter 6 presents an empirical and theoretical analysis of the problem of dimensional degeneracy for filters designed as in Chapter 2. This partially explains why the nonlinear approximation using this system was unable to improve upon the 5/3 wavelet.

Chapter 7 presents concluding remarks and an explanation of possible reasons why the directional image representation proposed in this thesis failed to improve upon current algorithms in the field.

Appendix A presents the prediction, equivalent analysis, and equivalent synthesis filters for all of the filters designed in Chapter 2. The equivalent analysis and synthesis filters for the 5/3 wavelet are presented for comparison.

CHAPTER 2

GEOMETRIC LIFTING

The most common method for designing signal representations in the discrete domain is by using filter banks. Filter bank image representations can be designed either through the theory of perfect reconstruction filter banks [24] or through lifting [15, 16]. While perfect reconstruction filter bank theory allows the design of arbitrary filters for each channel, the requirements for perfect reconstruction are not trivial, especially in higher dimensions or when combined with directional spatial support requirements. Lifting solves the problem of perfect reconstruction merely by its structure; any prediction or update filter may be used without sacrificing perfect reconstruction.

In our algorithm, we will use the simplest of lifting schemes, which is a four-channel single-stage lifting scheme created by separable sampling by two in each dimension, which is shown in Figure 2.1. The polyphase lattice created by separable sampling is shown in Figure 2.2. By using this lattice, the representation will be consistent with the dyadic two-scale relation of tensor product wavelet representations. Therefore, our results will be reported by comparison with the results using separable wavelets.

To have a sparse representation, many of the coefficients need to be zero or near zero. Except at edges, most natural images have a locally regular geometry. Therefore, sparse representation depends upon the number of vanishing moments in the analysis filter [2]. Assuming that the image is locally linear, this criterion is that the equivalent analysis filter bank contain two vanishing moments. While the locally linear assumption seems unrealistic, filters designed under this assumption have found widespread acceptance (e.g. the 5/3 wavelet in JPEG-2000 [25]). The assumption that the image is locally regular breaks down at edges. Within a lifting framework, this means that the prediction residual will be large whenever the filter support contains pixels from both sides of the edge. Therefore, the prediction residual can be minimized by designing the filter support to align with edge directions.

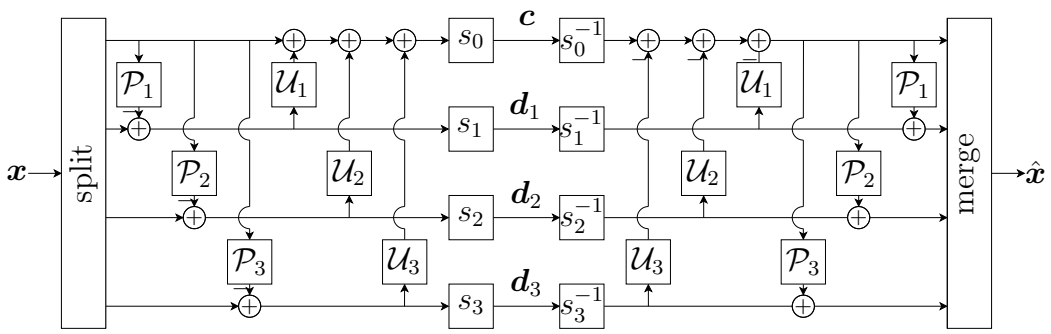


Figure 2.1: Filter bank representation of a four channel single-stage lifting scheme.

$$\begin{array}{ccccccc}
\ddots & \vdots & \vdots & \vdots & \vdots & \iddots \\
\cdots & 0 & 1 & 0 & 1 & \cdots \\
\cdots & 2 & 3 & 2 & 3 & \cdots \\
\cdots & 0 & 1 & 0 & 1 & \cdots \\
\cdots & 2 & 3 & 2 & 3 & \cdots \\
\iddots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

Figure 2.2: Demonstration of the four polyphase components of separable sampling.

2.1 Orthonormal Lifting Structures

Lifting schemes with linear filters can be expressed through their analysis and synthesis polyphase matrices. The analysis polyphase matrix of the four channel lifting scheme shown in Figure 2.1 is

$$\boldsymbol{P} = \begin{bmatrix} s_0 & 0 & 0 & 0 \\ 0 & s_1 & 0 & 0 \\ 0 & 0 & s_2 & 0 \\ 0 & 0 & 0 & s_3 \end{bmatrix} \begin{bmatrix} 1 & U_1 & U_2 & U_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -P_1 & 1 & 0 & 0 \\ -P_2 & 0 & 1 & 0 \\ -P_3 & 0 & 0 & 1 \end{bmatrix}.$$

Similar to the one-dimensional case, the invertibility of $\boldsymbol{P}$ demonstrates that the lifting scheme is trivially invertible for any choice of prediction and update filters.

In many filter bank applications, the goal is to create an orthonormal filter bank. One significant advantage of orthonormal filter banks is that they preserve the ℓ_2 norm. This is useful for nonlinear approximations because it

ensures that minimizing the ℓ_2 norm of the error in the transform domain is equivalent to minimizing the ℓ_2 norm in the signal domain. Unfortunately, the conditions for orthonormality and a single-stage multi-channel lifting scheme are nearly incompatible.

Theorem 2.1. *An M-channel single-stage lifting structure with linear FIR filters is orthonormal if and only if there exists a single $m \in \{1, \ldots, M-1\}$ such that the following conditions hold:*

1. $P_i = 0$ *for all* $i \neq m$.
2. P_m *is a monomial with gain* $\sqrt{s_m^{-2} - 1}$.
3. $U_i = 0$ *for all* $i \neq m$.
4. $U_m = s_m^2 \tilde{P}_m$.
5. $s_i = \pm 1$ *for all* $i \neq m$ *and* $i \neq 0$.
6. $s_m = \pm s_0^{-1}$.

P_i and U_i above are the transfer functions of the predict and update filters $\mathcal{P}_i$ and $\mathcal{U}_i$ respectively, and $\tilde{P}$ is the adjoint of P.

Proof. A necessary and sufficient condition for a filter bank to be orthonormal is that its synthesis polyphase matrix $\boldsymbol{P}^{-1}$ satisfy $\boldsymbol{P}^{-1}\tilde{\boldsymbol{P}}^{-1} = \boldsymbol{I}$ where $\tilde{\boldsymbol{P}}$ is the adjoint polyphase matrix [24]. Therefore, it is sufficient to prove that the listed conditions are true if and only if $\tilde{\boldsymbol{P}}\boldsymbol{P} = \boldsymbol{I}$. As the left and right inverse of invertible matrices are identical, this implies $\boldsymbol{P}\tilde{\boldsymbol{P}} = \boldsymbol{I}$. The M-channel polyphase matrix is

$$\begin{aligned}
\boldsymbol{P} &= \begin{bmatrix} s_0 & 0 & \cdots & 0 \\ 0 & s_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_{M-1} \end{bmatrix} \begin{bmatrix} 1 & U_1 & \cdots & U_{M-1} \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -P_1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -P_{M-1} & 0 & \cdots & 1 \end{bmatrix} \\
&= \begin{bmatrix} s_0\left(1 - \sum_{k=1}^{M-1} U_k P_k\right) & s_0 U_1 & \cdots & s_0 U_{M-1} \\ -s_1 P_1 & s_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -s_{M-1} P_{M-1} & 0 & \cdots & s_{M-1} \end{bmatrix}.
\end{aligned}$$

Therefore, the necessary and sufficient condition is that

$$\boldsymbol{I} = \begin{bmatrix} s_0\left(1-\sum_{k=1}^{M-1} U_k P_k\right) & s_0 U_1 & \cdots & s_0 U_{M-1} \\ -s_1 P_1 & s_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -s_{M-1} P_{M-1} & 0 & \cdots & s_{M-1} \end{bmatrix} \cdot \begin{bmatrix} s_0\left(1-\sum_{k=1}^{M-1} \tilde{U}_k \tilde{P}_k\right) & -s_1 \tilde{P}_1 & \cdots & -s_{M-1}\tilde{P}_{M-1} \\ s_0 \tilde{U}_1 & s_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ s_0 \tilde{U}_{M-1} & 0 & \cdots & s_{M-1} \end{bmatrix}.$$

This is an M^2 system of equations. Consider each element of the output matrix in turn. For consistency with the previous notation, enumerate the rows and columns of the matrix starting from zero.

- Element (i,i) for $i > 0$: $s_i^2 P_i \tilde{P}_i + s_i^2 = 1$. This equation reduces to $P_i \tilde{P}_i = s_i^{-2} - 1$. Assume that the order of the Laurent polynomial P_i is q. By the definition of adjoint, the order of $\tilde{P}_i$ is also q. By Laurent polynomial multiplication, the order of the product $P_i \tilde{P}_i$ is $2q$. As this product is equal to $s_i^{-2} - 1$, which has order zero, we have $2q = 0$. Therefore P_i is a monomial. As the gain of P_i and $\tilde{P}_i$ must be equal, the gain of P_i must be $\sqrt{s_i^{-2} - 1}$.

- Element (i,j) for $i > 0$, $j > 0$, and $i \neq j$: $s_i s_j P_i \tilde{P}_j = 0$. As both P_i and $\tilde{P}_j$ are monomials, the zero product property implies that one of them must be zero. As this equation holds for every pair of (i,j) greater than zero, there is only one $m \in \{1, \ldots, M-1\}$ such that $P_m \neq 0$. Furthermore, by combining with the previous equation, for all $i \neq m$, $P_i \tilde{P}_i = 0$ implies $s_i = \pm 1$.

- Element $(0,j)$ for $j > 0$: $-s_0 s_j \tilde{P}_j \left(1 - \sum_{i=1}^{M-1} U_i P_i\right) + s_0 s_j U_j = 0$. If $j = m$ this reduces to $U_m(1 + P_m \tilde{P}_m) = \tilde{P}_m$, which further reduces to $U_m = s_m^2 \tilde{P}_m$. If $j \neq m$, then $\tilde{P}_j = 0$, which implies $U_j = 0$.

- Element $(i,0)$ for $i > 0$: $-s_0 s_i P_i \left(1 - \sum_{i=1}^{M-1} \tilde{U}_i \tilde{P}_i\right) + s_0 s_i \tilde{U}_i = 0$. If $i = m$, then this reduces to $\tilde{U}_m = s_m^2 P_m$. Otherwise, $P_i = 0$, which implies $\tilde{U}_i = 0$. By taking the adjoint of both sides of these equations,

we see that this relationship is identical to the required relationship from the previous set of equations.

- Element $(0,0)$: $\left(1-\sum_{k=1}^{M-1} U_k P_k\right)\left(1-\sum_{k=1}^{M-1} \tilde{U}_k \tilde{P}_k\right)+\sum_{k=1}^{M-1} U_k \tilde{U}_k = s_0^{-2}$. Eliminating all terms that are zero, this becomes $s_0^2(1-U_m P_m)(1-\tilde{U}_m \tilde{P}_m) + s_0^2 U_m \tilde{U}_m = 1$. Expanding terms, substituting the relations $U_m = s_m^2 \tilde{P}_m$, $\tilde{U}_m = s_m^2 P_m$, and $P_m \tilde{P}_m = s_m^2 - 1$, and simplifying provides $s_0^2 s_m^2 = 1$. ■

This theorem is very restrictive, as constant filters cannot have multiple vanishing moments or directional support. Therefore, we cannot restrict ourselves to orthonormal filters.

As there is no restriction on the relationship between the ℓ_2 norm in the signal domain and the transform domain for biorthogonal filters, we would like to create filters that are nearly orthonormal. If the filter bank were orthonormal, the polyphase matrix evaluated on the unit circle would be unitary. Therefore, its condition number is always one. As no matrix can have a condition number smaller than one, the magnitude of the condition number is a measure of divergence away from orthonormality. The condition number of a polyphase matrix is formally defined in terms of the minimum and maximum singular values as

$$c = \frac{\sup_{\boldsymbol{\omega}\in[0\ 2\pi)^2} \sigma_{\max}(\boldsymbol{P}(\boldsymbol{\omega}))}{\inf_{\boldsymbol{\omega}\in[0\ 2\pi)^2} \sigma_{\min}(\boldsymbol{P}(\boldsymbol{\omega}))}.$$

2.2 Directional Support

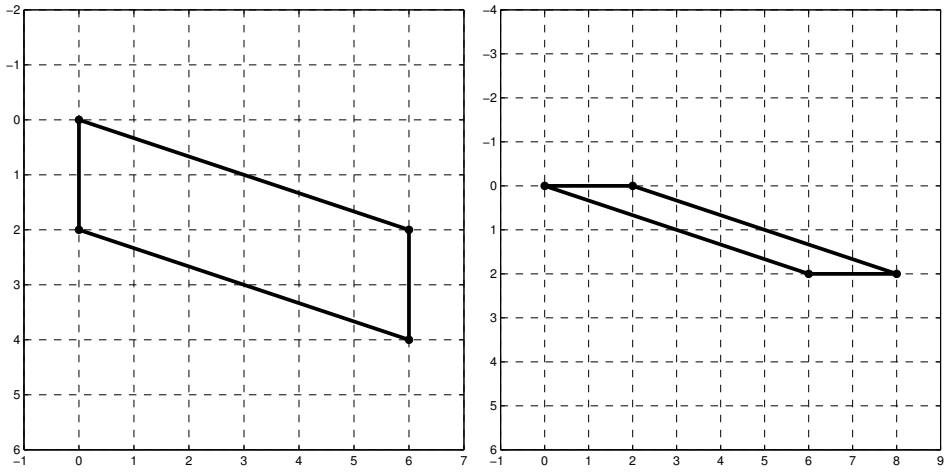
In order to meet the criterion of geometric basis functions, we need to define filters with directional support. As we are assuming bilinear polynomials, we need at least four independent points. The question is how to choose these points in a manner which is directional and local, yet which acknowledges the limitations of a discrete grid.

One method is to create the filter support in the continuous domain, then use subpixel interpolation to determine all points which do not fall on integer locations. While this approach seems to easily solve the question of directional support, it actually does not maintain directional filters. If the subpixel interpolation filter is not directional (e.g. [19]), the anisotropy of

the equivalent filter will be significantly reduced. Even if the subpixel interpolation filter is directional (e.g. [20]), the two-step process of designing the filter for subpixel locations then interpolating data to those locations is suboptimal. Rather, the filter should be designed similar to [21], which acknowledges that data only exists at integral locations and designs the filters accordingly. Therefore, in this work, we will only allow filter taps at integral locations.

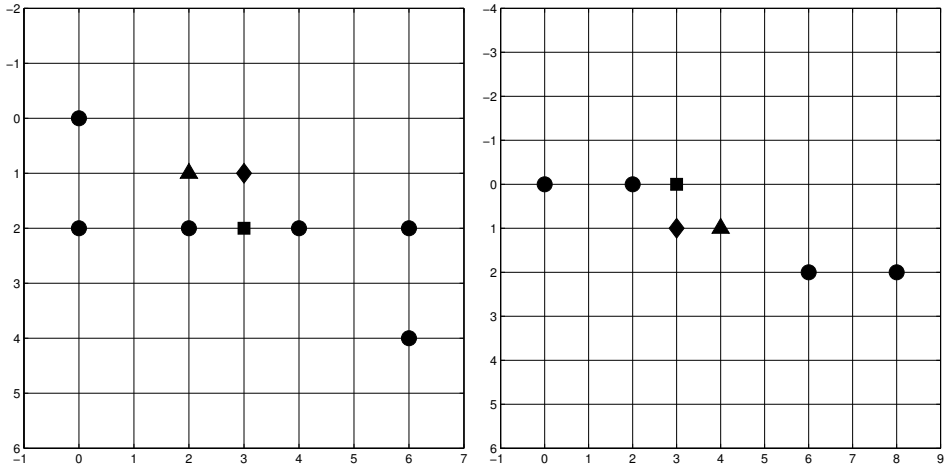
In order to design a filter with at least four taps which is local and directional, create the filter support by using a parallelogram. A parallelogram is uniquely determined by two vectors. Recall from the block diagram of the lifting scheme in Figure 2.1 that the prediction filters need to be defined such that all filter taps correspond to data in polyphase zero. This implies that the vertices of the parallelogram should only fall on even integers. Align one of the vectors at the desired angle. In order to maintain locality without using subpixel interpolation, the desired angle needs to be the arctangent of a ratio of small even integers. Align the second vector along a scanline. This maintains locality by operating in a direction in which data exists at a distance of two pixels. This scanline vector can be defined in four ways. It could be chosen to be either always horizontal or always vertical; however, this does not create consistent properties between directions which are nearly horizontal or nearly vertical. A better method is to choose the scanline which is either more parallel or more perpendicular to the directional vector. We will refer to the "more parallel" case as "skinny" parallelograms and the "more perpendicular" case as "fat" parallelograms. An example of parallelograms defined in this way is shown in Figure 2.3.

Given a parallelogram, the remaining filter support questions for prediction filters are which points should be used in the prediction and which point is being predicted. To account for the local regularity of natural image geometry, the filter should use all available information that is consistent with direction and locality in making the prediction. This implies that every phase zero point which falls within the parallelogram should be used. The predicted point should be as near as possible to the center of the parallelogram in order to maintain locality in all directions. As these filters will be used as prediction filters for all three polyphases, a predicted point needs to be chosen for each polyphase. Finally, the filter support can be shifted for each phase such that it becomes a proper impulse response. An example of



(a) Fat Parallelogram (b) Skinny Parallelogram

Figure 2.3: Demonstration of parallelograms designed for a direction of arctan(6/2).



(a) Fat Filter (b) Skinny Filter

Figure 2.4: Demonstration of the filters designed from the parallelograms of Figure 2.3. The black dots are the values used in the prediction. The square, triangle, and diamond are the predicted locations for phases 1–3. In order to show all three predicted locations on the same figure, the shifting needed to make an impulse response has not been applied.

filters defined from the parallelograms of Figure 2.3 is shown in Figure 2.4.

Using this method of designing filters for directional parallelograms, we propose a set of 12 geometric filters. The parallelograms were chosen to provide good angular coverage while maintaining locality. The set of fat filters is shown in Figure 2.5 and the set of skinny filters is shown in Figure 2.6.

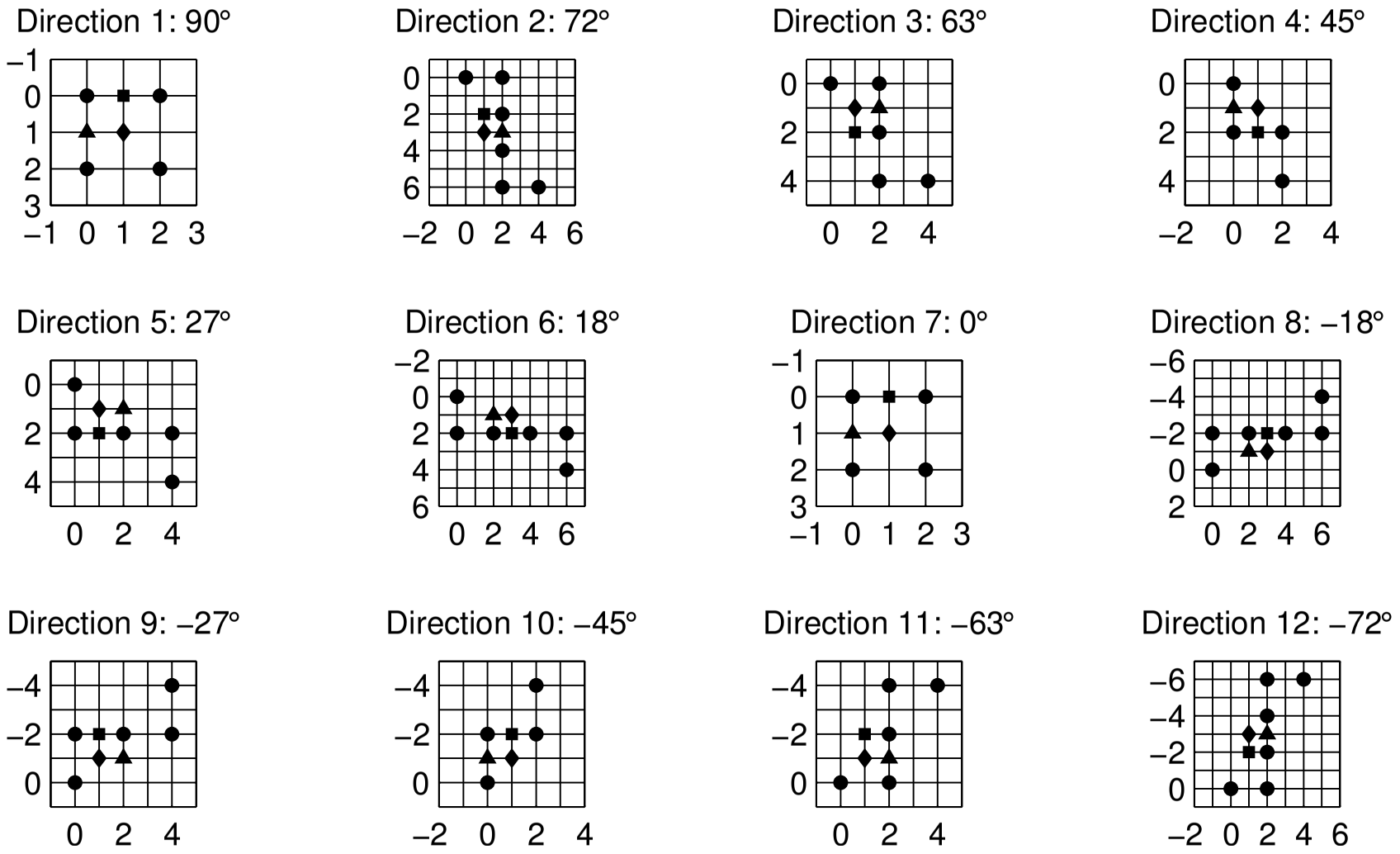


Figure 2.5: Set of 12 fat geometric filters. The large dots are the elements of phase zero used in the prediction. The square, triangle, and diamond are the predicted locations in phases 1–3 respectively. In order to show all three predicted locations on the same figure, the shifting needed to make an impulse response has not been applied.

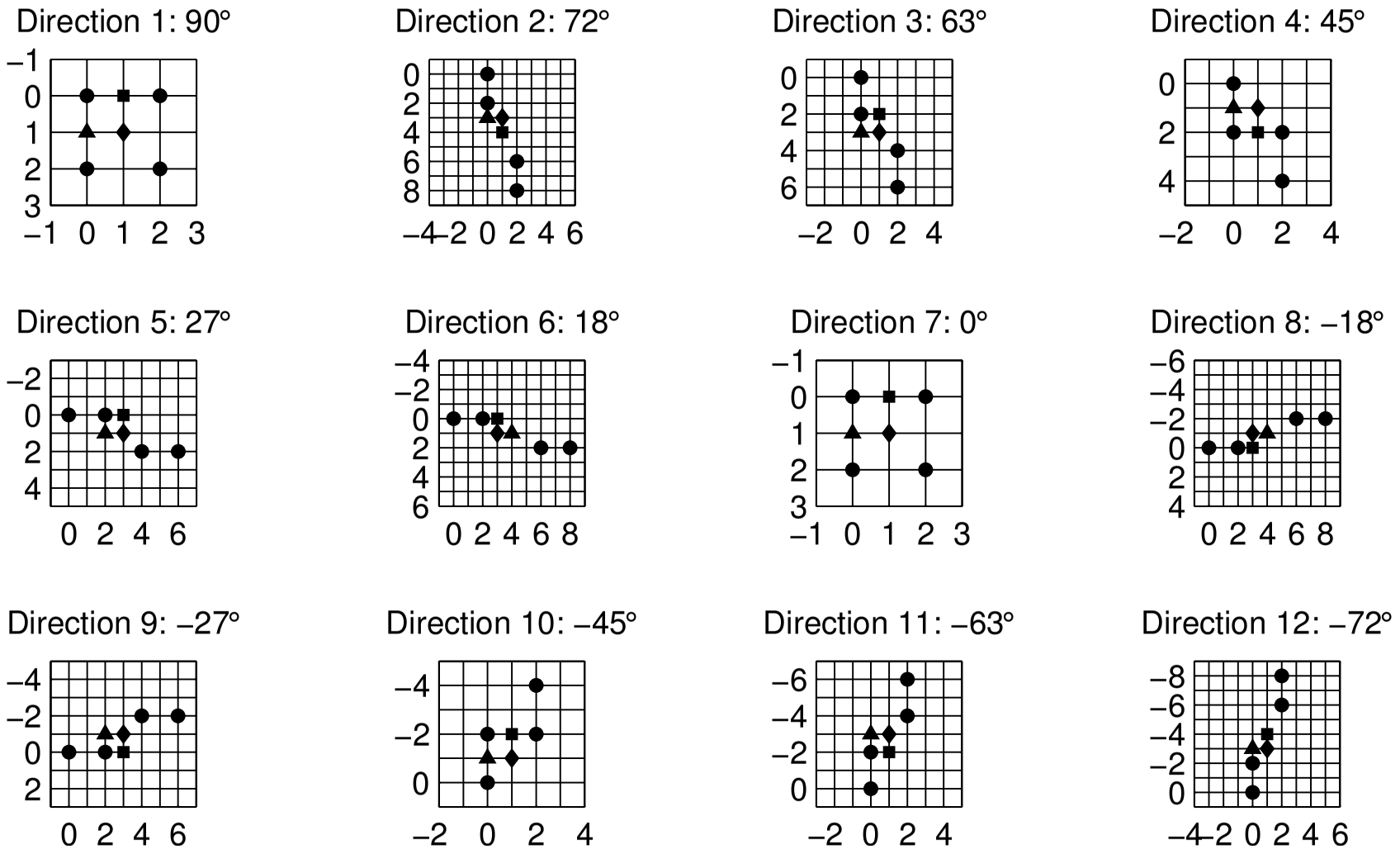


Figure 2.6: Set of 12 skinny geometric filters. The large dots are the elements of phase zero used in the prediction. The square, triangle, and diamond are the predicted locations in phases 1–3 respectively. In order to show all three predicted locations on the same figure, the shifting needed to make an impulse response has not been applied.

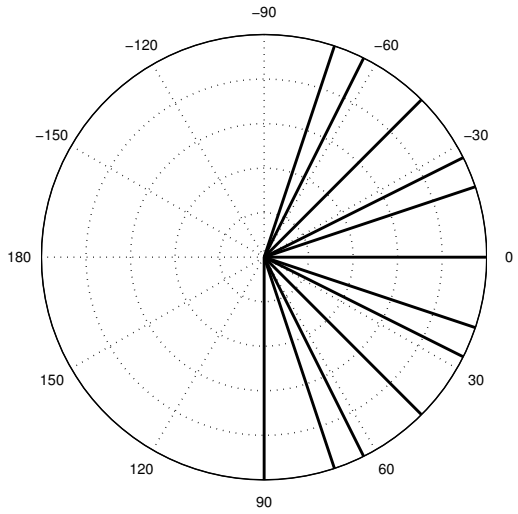


Figure 2.7: Angular quantization of the geometric filters.

We propose using the fat filters as opposed to the skinny filters for reasons described in Section 2.4. The angular quantization of the filters is shown in Figure 2.7. As image edges are axial, specifying the filters for half of the unit circle is sufficient. The maximum difference between neighboring filters is 18° and the minimum difference is 9°.

2.3 Prediction Filters

In our formulation of geometric lifting, we desire that the analysis portion of the filter bank have two vanishing moments in each dimension. As images can be approximated fairly well as locally linear in areas away from edges, this choice should sparsify the detail bands. In addition, this choice will make our transform comparable with the tensor product 5/3 wavelet, which also has two vanishing moments in each dimension. Kovačević and Sweldens proved that the analysis portion of the filter bank has K vanishing if and only if the prediction filter is able to exactly represent polynomials of order less than K [26].

For the case of two vanishing moments in each dimension, this implies that we must exactly represent all functions of the class

$$\Pi_{1,1} = \{f[m,n] | f[m,n] = a_1 + a_2 m + a_3 n + a_4 mn, \quad a_1, a_2, a_3, a_4 \in \mathbb{R}\}.$$

This class is defined as the span of four linearly independent basis functions $\{1, m, n, mn\}$. Consider a set of N points in the plane, enumerated as $\{(m_1, n_1), \ldots, (m_N, n_N)\}$. Given a set of the parameters $\{a_1, a_2, a_3, a_4\}$, the value of the function at these locations can be calculated through the matrix equation

$$\begin{bmatrix} f[m_1, n_1] \\ \vdots \\ f[m_N, n_N] \end{bmatrix} = \begin{bmatrix} 1 & m_1 & n_1 & m_1 n_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & m_N & n_N & m_N n_N \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}.$$

This equation can be written succinctly as $\boldsymbol{f} = \boldsymbol{R}^T \boldsymbol{a}$. If the points are not degenerate $\boldsymbol{R}^T$ has rank four. Therefore, the minimization

$$\boldsymbol{a} = \underset{\boldsymbol{a}}{\operatorname{argmin}} \|\boldsymbol{f} - \boldsymbol{R}^T \boldsymbol{a}\|^2$$

has the unique solution

$$\boldsymbol{a} = (\boldsymbol{R}\boldsymbol{R}^T)^{-1} \boldsymbol{R}\boldsymbol{f}. \tag{2.1}$$

The value of the function at any other point (m_p, n_p) can be written as

$$f[m_p, n_p] = a_1 + m_p a_2 + n_p a_3 + m_p n_p a_4.$$

Defining $\boldsymbol{p}^T = \begin{bmatrix} 1 & m_p & n_p & m_p n_p \end{bmatrix}$ allows the value of the function to be written as $f[m_p, n_p] = \boldsymbol{p}^T \boldsymbol{a}$. This can be expanded using (2.1) as

$$f[m_p, n_p] = \boldsymbol{p}^T (\boldsymbol{R}\boldsymbol{R}^T)^{-1} \boldsymbol{R}\boldsymbol{f}.$$

This function evaluation can be written as a linear filter by expressing the output value as $f[m_p, n_p] = \boldsymbol{h}^T \boldsymbol{f}$ where

$$\boldsymbol{h} = \boldsymbol{R}^T (\boldsymbol{R}\boldsymbol{R}^T)^{-1} \boldsymbol{p}. \tag{2.2}$$

An alternate formulation is also informative. In this method, the goal of $f[m_p, n_p] = \boldsymbol{h}^T \boldsymbol{f}$ is expressed first. Under the assumption that the signal is a bilinear, we have the two constraints $f[m_p, n_p] = \boldsymbol{p}^T \boldsymbol{a}$ and $\boldsymbol{f} = \boldsymbol{R}^T \boldsymbol{a}$. Substituting these together provides $\boldsymbol{p}^T \boldsymbol{a} = \boldsymbol{h}^T \boldsymbol{R}^T \boldsymbol{a}$. As this must be true for all $\boldsymbol{a}$, we know that

$$\boldsymbol{p} = \boldsymbol{R}\boldsymbol{h}. \tag{2.3}$$

If N is greater than four, this system is underdetermined. Define a solution for $\boldsymbol{h}$ as the minimizer of $J(\boldsymbol{h}) = \|\boldsymbol{h}\|^2$ subject to $\boldsymbol{p} = \boldsymbol{R}\boldsymbol{h}$.

Because J is strictly convex and the set of feasible directions $\boldsymbol{q}$ such that $\boldsymbol{p} = \boldsymbol{R}(\boldsymbol{h} + \boldsymbol{q})$ is a convex subspace, this problem can be solved by the Projection Theorem [27]. The minimizer is the vector such that $\nabla^T J\boldsymbol{q} = 0$ for all $\boldsymbol{q} \in \text{Null}(\boldsymbol{R})$, which implies that $\nabla J \in \text{Null}(\boldsymbol{R})^{\perp}$. By the Fundamental Subspaces Theorem, this is equivalent to $\nabla J \in \text{Range}(\boldsymbol{R}^T)$ [28]. Therefore, there exists $\boldsymbol{s}$ such that $\nabla J = \boldsymbol{R}^T\boldsymbol{s}$. As $\nabla J = 2\boldsymbol{h}$, we have $\boldsymbol{h} = \frac{1}{2}\boldsymbol{R}^T\boldsymbol{s}$. Substituting this into the constraint implies that $2\boldsymbol{p} = \boldsymbol{R}\boldsymbol{R}^T\boldsymbol{s}$. As $\boldsymbol{R}^T$ is rank four, this equation can be solved as $\boldsymbol{s} = 2(\boldsymbol{R}\boldsymbol{R}^T)^{-1}\boldsymbol{p}$. Therefore, $\boldsymbol{h} = \boldsymbol{R}^T(\boldsymbol{R}\boldsymbol{R}^T)^{-1}\boldsymbol{p}$ as before. Therefore, in addition to minimizing the difference between the function and the linear polynomial, it is also the minimum norm solution assuming a linear polynomial.

The significance of (2.2) is that the value of the function at any location can be determined solely from the value of the function at other locations and the geometric relationship between the various points. Since (2.2) does not depend upon $\boldsymbol{f}$, the filter $\boldsymbol{h}$ depends only on the locations of the various points $\{(m_1, n_1), \ldots, (m_N, n_N), (m_p, n_p)\}$ not on their values. Furthermore, the method to create $\boldsymbol{h}$ is shift-invariant. This is seen by expressing (2.3) as

$$\begin{bmatrix} 1 \\ m_p \\ n_p \\ m_p n_p \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ \boldsymbol{m} \\ \boldsymbol{n} \\ \boldsymbol{m} \otimes \boldsymbol{n} \end{bmatrix} \boldsymbol{h}$$

where $\mathbf{1}$ is a row vector of ones, $\boldsymbol{m}$ and $\boldsymbol{n}$ are row vectors of the m and n coordinates of each point used in the prediction, and $\otimes$ is elementwise multiplication. The solution for $\boldsymbol{h}$ can be found using the augmented matrix

$$\begin{bmatrix} \mathbf{1} & 1 \\ \boldsymbol{m} & m_p \\ \boldsymbol{n} & n_p \\ \boldsymbol{m} \otimes \boldsymbol{n} & m_p n_p \end{bmatrix}. \tag{2.4}$$

If the coordinate system is shifted by Δ_m and Δ_n, then (2.3) is

$$\begin{bmatrix} 1 \\ m_p + \Delta_m \\ n_p + \Delta_n \\ (m_p + \Delta_m)(n_p + \Delta_n) \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ \boldsymbol{m} + \Delta_m \mathbf{1} \\ \boldsymbol{n} + \Delta_n \mathbf{1} \\ (\boldsymbol{m} + \Delta_m \mathbf{1}) \otimes (\boldsymbol{n} + \Delta_n \mathbf{1}) \end{bmatrix} \boldsymbol{h}_{\Delta_m, \Delta_n}.$$

The solution for $\boldsymbol{h}_{\Delta_m, \Delta_n}$ must satisfy the augmented matrix

$$\begin{bmatrix} \mathbf{1} & 1 \\ \boldsymbol{m} + \Delta_m \mathbf{1} & m_p + \Delta_m \\ \boldsymbol{n} + \Delta_n \mathbf{1} & n_p + \Delta_n \\ (\boldsymbol{m} + \Delta_m \mathbf{1}) \otimes (\boldsymbol{n} + \Delta_n \mathbf{1}) & (m_p + \Delta_m)(n_p + \Delta_n) \end{bmatrix}.$$

Through elementary row operations on the middle two rows and expanding the bottom row, we get

$$\begin{bmatrix} \mathbf{1} & 1 \\ \boldsymbol{m} & m_p \\ \boldsymbol{n} & n_p \\ \boldsymbol{m} \otimes \boldsymbol{n} + \Delta_m \boldsymbol{n} + \Delta_n \boldsymbol{m} + \Delta_m \Delta_n \mathbf{1} & m_p n_p + \Delta_m n_p + \Delta_n m_p + \Delta_m \Delta_n \end{bmatrix}.$$

Applying elementary row operations on the fourth row reduces this augmented matrix to (2.4). Therefore, the system of equations which leads to the solution $\boldsymbol{h}$ is equivalent to the system of equations leading to $\boldsymbol{h}_{\Delta_m, \Delta_n}$. Therefore, the filter coefficients are independent of Δ_m and Δ_n and the filter is shift-invariant.

For signals which are members of $\Pi_{1,1}$, the filter $\boldsymbol{h}$ exactly evaluates the function at the new location. However, most signals are not members of $\Pi_{1,1}$. For the broader class of signals which are approximately locally linear, the residual

$$d[m_p, n_p] = f[m_p, n_p] - \boldsymbol{h}^T \boldsymbol{f}$$

should be small. Therefore, these filters can be viewed as a prediction filter of the value at (m_p, n_p) given the various other points. This method provides a simple mechanism to create a prediction filter from any set of nondegenerate points. Therefore, this method can be used to design the prediction filters for a lifting scheme.

2.4 Update Filters and Scaling

Unlike the prediction filters, the update filters do not have an easily identifiable role. Possible types of update filters include:

- Eliminate the update filter.
- Use square averaging filters.
- Use the prediction filter.
- Use filters that provide vanishing moments in the synthesis portion of the filter bank. By [26] this implies that the update filters are $\frac{1}{4}$ the adjoint of the prediction filters.
- Use a scaled version of the square averaging filter or prediction filter.

Similarly, the scaling constants are not easily derived. For orthogonal filter banks, the scaling is often designed such that $\|\boldsymbol{h}\| = 1$, which makes the filter bank orthonormal. As our filters cannot be orthonormal, this derivation of the scaling parameters is uninformative. From our previous discussion, the prediction filters were designed by exactly fitting bilinear polynomials. As a constant function is a bilinear polynomial, it must be exactly represented. Therefore, every filter must have unitary DC gain. The scaling parameters $s_0, \ldots, s_3$ do not need to be unitary; however, to allow the filters to cross block boundaries as described in Section 3.3, the DC gain of the approximation band for each direction must be the same. Intuitive choices for the scaling are:

- Set s_0 to force the average ℓ_2 norm of the equivalent approximation filters to be one. Set the detail band scaling to force the ℓ_2 norm of each equivalent detail filter to be one.
- Allow s_0 to be a free parameter and set the detail band scaling to force the ℓ_2 norm of each equivalent detail filter to be one.
- Allow s_0 to be a free parameter and set the detail band scaling to one.

The design criterion for the update filters and the scaling is to jointly force the complete filter bank to be as orthonormal as possible. From prior discussion, minimizing the condition number of the polyphase matrix measures the

Table 2.1: Optimal Condition Number for each Direction

Direction	Angle	Condition Number
1	90	2.0000
2	72	2.1699
3	63	2.0962
4	45	2.0000
5	27	2.0962
6	18	2.1699
7	0	2.0000
8	-18	2.1699
9	-27	2.0962
10	-45	2.0000
11	-63	2.0962
12	-72	2.1699

divergence from orthonormality. In addition to optimizing over the update filters and scaling constants, this optimization also allowed the prediction filters to be either fat or skinny filters. The optimization space was discretized by allowing the scaling of the update filters and the free parameters of the scaling to be chosen from the set of 2^β where $\beta = -4, \ldots, 4$.

By exhaustive search, the solution to this optimization problem was fat prediction filters, update filters which provide vanishing moments, and scaling such that s_0 was two and all of the other scaling parameters were one. The condition number of each directional filter bank is shown in Table 2.1.

2.5 Geometric Filter Dictionary

The filter design operations described in this chapter do not need to be implemented in practical image representation algorithms. Rather, the filters can be placed into a geometric filter dictionary, which can be accessed whenever a specified filter is desired. This dictionary consists of the following elements:

- The prediction filter angles and coefficient locations of Figure 2.5, properly shifted to form impulse responses.
- The prediction filter coefficients calculated for each prediction filter according to (2.2).

- The update filter coefficients as $\frac{1}{4}$ the adjoint of the prediction filter coefficients.
- The scaling coefficients $\begin{bmatrix} 2 & 1 & 1 & 1 \end{bmatrix}$.

CHAPTER 3

ADAPTIVE SEGMENTATION

To maximally benefit from the adaptability of directional lifting, the image must be segmented into regions with consistent dominant directions. This problem can be cast as a rate-distortion optimization problem. The image can be recursively partitioned into blocks of variable size by quad-tree decomposition. Smaller blocks will have more uniformity in dominant direction; however, they will require more side information to code the directions and the tree structure. On the other hand, larger blocks will have less side information at the expense of higher prediction errors.

3.1 Optimal Tree Pruning Theory

One standard algorithm is the BFOS tree pruning algorithm [29]. This algorithm was created to determine the optimal classification tree given a set of training examples of the objects to be classified. This algorithm can be modified to provide a mechanism for adaptive segmentation for block-based adaptive transforms, for example [19].

Define the initial tree as T_0. This tree was built by recursively subdividing each block into subblocks until a minimum block size is reached. Let $t \in T_0$ be a node in the tree. As this tree is intended to segment an image f, define the block of the image represented by node t as $B_t \subset \mathbb{Z}^2$ and the number of pixels in this block as $|B_t|$. Given a node t, define a branch $T_t \subset T$ as the set of t and its descendants. For any tree or subtree, let $|\cdot|$ be the number of nodes in the tree or subtree. Let $\tilde{T}_t$ be the set of leaves in branch T_t. Notice that $\tilde{T}_t$ is a tree if and only if $|T_t| = 1$.

For image coding applications, each pixel should only be coded once. Therefore, the preferred tree structure contains non-overlapping blocks. The transform itself can still use the pixels that belong to other nodes; how-

ever, the adaptive parameters of the pixel belong to only one node. In the tree structure this requires that $B_t = \cup_{t' \in \tilde{T}_t} B_{t'}$ and $B_{t'} \cap B_{t''} = \emptyset$ for every $t', t'' \in \tilde{T}_t$ such that $t' \neq t''$. These two properties in turn imply that $|B_t| = \sum_{t' \in \tilde{T}_t} |B_{t'}|$.

The optimal tree is defined as

$$\hat{T}_0 = \operatorname*{argmin}_{S \subset T_0} J(S) \tag{3.1}$$

where the objective function is composed of distortion and rate terms as $J(S) = D(S) + \lambda R(S)$. The regularization parameter λ allows the user to adjust the weight given to distortion and rate.

In [29], the distortion function was the resubstitution error of the classifier. While this is the most intuitive distortion function for classification problems, it does not apply to block based adaptive transforms. For adaptive transform coding, the distortion for a node should be some function of the image values in the block corresponding to the node and of the adaptive parameters used to transform that node. Express these dependencies by defining the distortion function for node t as $d(f(B_t), \alpha_t)$ where α_t are the adaptive parameters. The distortion function for a tree is defined from the distortion functions for each leaf in the tree as

$$D(T_t) = \sum_{t' \in \tilde{T}_t} d(f(B_{t'}), \alpha_{t'}).$$

In order for the optimization to be well defined, the distortion for a node needs to be a nonnegative function which satisfies:

Property 3.1 (Monotonicity)**.**

$$d(f(B_t), \alpha_t) \geq \sum_{t' \in \tilde{T}_t} d(f(B_{t'}), \alpha_{t'}).$$

This property ensures that the distortion of a large block is always larger than the distortion of its constituent small blocks. This is intuitive because the collection of small blocks has more adaptive parameters, so it should have lower distortion.

In [29], the rate function for a tree was defined as the size of the tree. While this accounts for the overhead of the tree structure, it does not account for

any adaptive parameters of the transform. Therefore, define the rate function as the number of overhead bits required to encode the adaptive parameters for each leaf in the tree plus the overhead needed to code the tree structure. For node t, define $r(\alpha_t)$ as the number of bits required to encode the adaptive parameters α_t. The overhead for the tree is one bit per node, which specifies whether the node is a leaf or an internal node. The rate to code a tree is thus

$$R(T_t) = |T_t| + \sum_{t' \in \tilde{T}_t} r(\alpha_{t'}).$$

Often, the rate to encode the overhead of one large block is smaller than the rate to encode the overhead of a collection of smaller blocks; however, that is not strictly necessary.

3.2 Optimal Tree Pruning Implementation

The idea behind the solution to (3.1) is that every branch is itself a tree; however, it has fewer nodes than the original tree. Therefore, the problem can be solved by decomposing the minimization of the objective function for a tree into a sum of independent minimizations among the branches defined by the children of the root of that tree. As this is a recursive formulation, it is typically implemented starting with the leaves and building towards the root. Define $\{t\}$ to be the tree composed solely of node t. This implies that $|\{t\}| = 1$ and $\tilde{\{t\}} = \{t\}$. Define C_t to be the set of children of node t. Define

$$\{t \to \{S_i\}_{i \in \mathcal{I}}\}$$

for disjoint trees S_i in some index set $\mathcal{I}$ as the tree created with root t and subtrees S_i.

The solution to (3.1) is thus:

Case 1: $T_t = \{t\}$

In this case, the objective function is

$$\begin{aligned} J(\{t\}) &= D(\{t\}) + \lambda R(\{t\}) \\ &= d(f(B_t), \alpha_t) + \lambda \left[1 + r(\alpha_t)\right]. \end{aligned}$$

Therefore,

$$\min_{S \subset \{t\}} J(S) = d(f(B_t), \alpha_t) + \lambda \left[1 + r(\alpha_t)\right]$$

and

$$\operatorname*{argmin}_{S \subset \{t\}} J(S) = \{t\}.$$

Case 2: $T_t \neq \{t\}$

This case has two subcases depending upon whether the minimizing argument is $\{t\}$ or some other tree.

Case 2a: $\operatorname*{argmin}_{S \subset T_t \neq \{t\}} J(S) = \{t\}$

By the assumption that $\{t\}$ is the minimizer,

$$\begin{aligned}\min_{S=\{t\}} J(S) &= J(\{t\}) \\ &= d(f(B_t), \alpha_t) + \lambda \left[1 + r(\alpha_t)\right].\end{aligned}$$

Case 2b: $\operatorname*{argmin}_{S \subset T_t \neq \{t\}} J(S) \neq \{t\}$

Because the minimizing argument is not $\{t\}$, we know that the children of t are not removed in the minimization process. Therefore, the objective function can be expanded along the children of t.

$$\begin{aligned}J(T_t) &= D(T_t) + \lambda R(T_t) \\ &= \sum_{t' \in \tilde{T}_t} d(f(B_{t'}), \alpha_{t'}) + \lambda \left[|T_t| + \sum_{t' \in \tilde{T}_t} r(\alpha_{t'}) \right] \\ &= \sum_{t' \in C_t} \sum_{t'' \in \tilde{T}_{t'}} d(f(B_{t''}), \alpha_{t''}) \\ &\qquad + \lambda \left[1 + \sum_{t' \in C_t} \left[|T_{t'}| + \sum_{t'' \in \tilde{T}_{t'}} r(\alpha_{t''}) \right] \right] \\ &= \lambda + \sum_{t' \in C_t} \left[D(T_{t'}) + \lambda R(T_{t'}) \right] \\ &= \lambda + \sum_{t' \in C_t} J(T_{t'}).\end{aligned}$$

As each of the trees defined by the children of t is independent,

the minimization distributes over the sum as

$$\min_{\substack{S\subset T_t\neq\{t\}\\ S\neq\{t\}}} J(S) = \lambda + \sum_{t'\in C_t} \min_{S\subset T_{t'}} J(S).$$

For simplicity of notation, define

$$J_t = \lambda + \sum_{t'\in C_t} \min_{S\subset T_{t'}} J(S).$$

The minimizing argument is

$$\operatorname*{argmin}_{\substack{S\subset T_t\neq\{t\}\\ S\neq\{t\}}} J(S) = \left\{ t \to \left\{ \operatorname*{argmin}_{S\subset T_{t'}} J(S) \right\}_{t'\in C_t} \right\}.$$

Combining both of the subcases, we know that

$$\min_{S\subset T_t\neq\{t\}} J(S) = \begin{cases} J(\{t\}) & J(\{t\}) \leq J_t \\ J_t & J(\{t\}) > J_t. \end{cases}$$

The minimizing argument is

$$\operatorname*{argmin}_{S\subset T_t\neq\{t\}} J(S) = \begin{cases} \{t\} & J(\{t\}) \leq J_t \\ \left\{ t \to \left\{ \operatorname*{argmin}_{S\subset T_{t'}} J(S) \right\}_{t'\in C_t} \right\} & J(\{t\}) > J_t. \end{cases}$$

As these minimizations are written in terms of minimizations with fewer nodes, a finite number of recursive calls to Case 1 or Case 2 will solve (3.1).

3.3 Block Boundary Issues

In creating a blocked transform, there is a proliferation of boundaries within the image. While the original image only had boundaries along the outside, the blocked transform has boundaries along the outside and at every block boundary. These boundaries can be handled in two ways. First, classical boundary methods such as zero padding, symmetric extension, or periodic extension can be applied. This method maintains the independence of each

block; however, it increases the prediction residual of the block. Whenever the prediction filter needs to access a location across the boundary, the relationship between the predicted value and data extended to this location is probably different than the relationship between the predicted value and the data that would have been there except for the boundary. In the context of geometric lifting, all three methods modify edges. Zero padding adds vertical and horizontal edges not present in the original image. Symmetric extension reflects an edge across an axis, changing its direction. Periodic extension uses data from another location in the image, which may or may not have any related edge structure. For these reasons, images segmented with classical boundary methods tend to have large blocks because the error introduced by the extra boundaries is not offset by the adaptivity of the transform.

The second method is to use data from across the edge for all internal boundaries. This eliminates all boundary problems; however, the blocks are no longer independent. Consider a one-dimensional lifting scheme with two blocks, denoted by $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. The lifting equations (1.2) and (1.3) are thus

$$\begin{aligned}
\boldsymbol{d}_1 &= \boldsymbol{x}_{1,o} - \mathcal{P}_1(\boldsymbol{x}_{1,e}) \\
\boldsymbol{c}_1 &= \boldsymbol{x}_{1,e} + \mathcal{U}_1(\boldsymbol{d}_1) \\
\boldsymbol{d}_2 &= \boldsymbol{x}_{2,o} - \mathcal{P}_2(\boldsymbol{x}_{2,e}) \\
\boldsymbol{c}_2 &= \boldsymbol{x}_{2,e} + \mathcal{U}_2(\boldsymbol{d}_2).
\end{aligned}$$

Using data across block boundaries implies that there are some pixel locations $\boldsymbol{n}$ such that $\boldsymbol{d}_1[\boldsymbol{n}]$ depends upon both $\boldsymbol{x}_{1,e}$ and $\boldsymbol{x}_{2,e}$. These equations need to be rewritten as

$$\boldsymbol{d}_1 = \boldsymbol{x}_{1,o} - \mathcal{P}_1(\boldsymbol{x}_{1,e}, \boldsymbol{x}_{2,e}) \tag{3.2a}$$

$$\boldsymbol{c}_1 = \boldsymbol{x}_{1,e} + \mathcal{U}_1(\boldsymbol{d}_1, \boldsymbol{d}_2) \tag{3.2b}$$

$$\boldsymbol{d}_2 = \boldsymbol{x}_{2,o} - \mathcal{P}_2(\boldsymbol{x}_{1,e}, \boldsymbol{x}_{2,e}) \tag{3.2c}$$

$$\boldsymbol{c}_2 = \boldsymbol{x}_{2,e} + \mathcal{U}_2(\boldsymbol{d}_1, \boldsymbol{d}_2). \tag{3.2d}$$

Notice that the equation for $\boldsymbol{c}_1$ depends upon $\boldsymbol{d}_2$. Therefore, the implementation of the transform must be stage based ($\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ followed by $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$) rather than block based ($\boldsymbol{d}_1$ and $\boldsymbol{c}_1$ followed by $\boldsymbol{d}_2$ and $\boldsymbol{c}_2$). This precludes

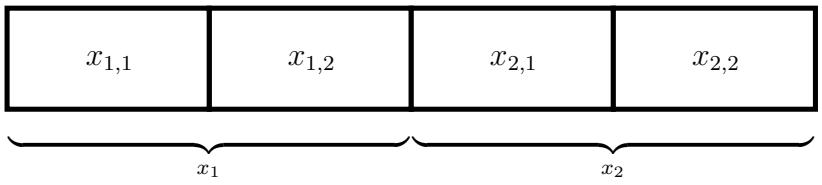


Figure 3.1: Adjacent data blocks at two scales.

the inclusion of tensor product lifting schemes such as the 5/3 wavelet in the system because they cannot be factored into stage based transforms on a separable two-dimensional sampling lattice.

In addition, there are limitations on the type of segmentation required by the interdependence of the blocks. Assume that $\boldsymbol{x}_1$ is split into two adjacent blocks $\boldsymbol{x}_{1,1}$ and $\boldsymbol{x}_{1,2}$ and likewise for $\boldsymbol{x}_2$ such that $\boldsymbol{x}_{1,2}$ shares a boundary with $\boldsymbol{x}_{2,1}$, as shown in Figure 3.1. The lifting equations become

$$\boldsymbol{d}_{1,1} = \boldsymbol{x}_{1,1,o} - \mathcal{P}_{1,1}(\boldsymbol{x}_{1,1,e}, \boldsymbol{x}_{1,2,e}) \tag{3.3a}$$

$$\boldsymbol{d}_{1,2} = \boldsymbol{x}_{1,2,o} - \mathcal{P}_{1,2}(\boldsymbol{x}_{1,1,e}, \boldsymbol{x}_{1,2,e}, \boldsymbol{x}_{2,1,e}) \tag{3.3b}$$

$$\boldsymbol{d}_{2,1} = \boldsymbol{x}_{2,1,o} - \mathcal{P}_{2,1}(\boldsymbol{x}_{1,2,e}, \boldsymbol{x}_{2,1,e}, \boldsymbol{x}_{2,2,e}) \tag{3.3c}$$

$$\boldsymbol{d}_{2,2} = \boldsymbol{x}_{2,2,o} - \mathcal{P}_{2,2}(\boldsymbol{x}_{2,1,e}, \boldsymbol{x}_{2,2,e}) \tag{3.3d}$$

$$\boldsymbol{c}_{1,1} = \boldsymbol{x}_{1,1,e} + \mathcal{U}_{1,1}(\boldsymbol{d}_{1,1}, \boldsymbol{d}_{1,2}) \tag{3.3e}$$

$$\boldsymbol{c}_{1,2} = \boldsymbol{x}_{1,2,e} + \mathcal{U}_{1,2}(\boldsymbol{d}_{1,1}, \boldsymbol{d}_{1,2}, \boldsymbol{d}_{2,1}) \tag{3.3f}$$

$$\boldsymbol{c}_{2,1} = \boldsymbol{x}_{2,1,e} + \mathcal{U}_{2,1}(\boldsymbol{d}_{1,2}, \boldsymbol{d}_{2,1}, \boldsymbol{d}_{2,2}) \tag{3.3g}$$

$$\boldsymbol{c}_{2,2} = \boldsymbol{x}_{2,2,e} + \mathcal{U}_{2,2}(\boldsymbol{d}_{2,1}, \boldsymbol{d}_{2,2}) \tag{3.3h}$$

Assume that the segmentation algorithm determines that the objective function is minimized by transmitting blocks $\boldsymbol{x}_{1,1}$, $\boldsymbol{x}_{1,2}$, and $\boldsymbol{x}_2$. Therefore, the transmitted signals are $\{\boldsymbol{c}_{1,1}, \boldsymbol{d}_{1,1}, \boldsymbol{c}_{1,2}, \boldsymbol{d}_{1,2}, \boldsymbol{c}_2, \boldsymbol{d}_2\}$. Inverting this stream requires inverting equations (3.2c), (3.2d), (3.3a), (3.3b), (3.3e), and (3.3f).

These equations become

$$\boldsymbol{x}_{2,e} = \boldsymbol{c}_2 - \mathcal{U}_2(\boldsymbol{d}_1, \boldsymbol{d}_2) \tag{3.4a}$$

$$\boldsymbol{x}_{1,1,e} = \boldsymbol{c}_{1,1} - \mathcal{U}_{1,1}(\boldsymbol{d}_{1,1}, \boldsymbol{d}_{1,2}) \tag{3.4b}$$

$$\boldsymbol{x}_{1,2,e} = \boldsymbol{c}_{1,2} - \mathcal{U}_{1,2}(\boldsymbol{d}_{1,1}, \boldsymbol{d}_{1,2}, \boldsymbol{d}_{2,1}) \tag{3.4c}$$

$$\boldsymbol{x}_{2,o} = \boldsymbol{d}_2 + \mathcal{P}_2(\boldsymbol{x}_{1,e}, \boldsymbol{x}_{2,e}) \tag{3.4d}$$

$$\boldsymbol{x}_{1,1,o} = \boldsymbol{d}_{1,1} + \mathcal{P}_{1,1}(\boldsymbol{x}_{1,1,e}, \boldsymbol{x}_{1,2,e}) \tag{3.4e}$$

$$\boldsymbol{x}_{1,2,o} = \boldsymbol{d}_{1,2} + \mathcal{P}_{1,2}(\boldsymbol{x}_{1,1,e}, \boldsymbol{x}_{1,2,e}, \boldsymbol{x}_{2,1,e}) \tag{3.4f}$$

Unfortunately, these equations depend upon $\boldsymbol{d}_1$, $\boldsymbol{d}_{2,1}$, $\boldsymbol{x}_{1,e}$, and $\boldsymbol{x}_{2,1,e}$ which are not transmitted or calculated. Furthermore, if the prediction and update filters of equations (3.2) and (3.3) are independent, these quantities cannot be calculated from the known data. Therefore, the segmentation cannot occur after the update step. The segmentation can occur either before any transforms or after the first predict step. It can occur at the beginning because then all transforms have a consistent segmentation. It can occur after the first predict step because all transforms depend only upon the original data, which is the same at all scales.

CHAPTER 4

DIRECTION ESTIMATION

To perform block-based directional filtering on a natural image, one of the fundamental challenges is determining the dominant direction in each block. Under the assumption that the image is piecewise smooth, this dominant direction is caused by the edge directions. This reduces the problem to finding the dominant edge direction within a block. For images which are also piecewise constant, the gradient of the image along the edge will point in the direction perpendicular to the edge and will be zero elsewhere. Therefore, a reasonable estimate of the dominant direction is the direction perpendicular to the average of the gradient directions. These types of averages are computed using the techniques of directional statistics. While the piecewise smooth assumption is generally valid, the piecewise constant assumption is generally invalid. Therefore, the standard equations for average direction will need to be modified.

4.1 Theory of Directional Statistics

The mean and dispersion of angular quantities are computed using the techniques of directional statistics [30]. Let $\{\theta_1, \ldots, \theta_N\}$ be a collection of N samples of an angular distribution that is periodic on 2π. Define the complex resultant by

$$z^1 = \frac{1}{N} \sum_{i=1}^{N} \mathrm{e}^{j\theta_i} .$$

The mean is defined as the angle of the complex resultant

$$\mu^1 = \angle z^1.$$

The circular dispersion about some angle η is defined as

$$V^1(\eta) = \frac{1}{N}\sum_{i=1}^{N}\left[1-\cos(\theta_i-\eta)\right].$$

The variance is merely the dispersion about the mean.

These definitions of mean and dispersion exhibit many nice properties [30]:

Property 4.1 (Coordinate System Invariance)**.** *An additive change to the distribution results in merely an additive change in the directional mean. Specifically, if the mean of the set* $\{\theta_1,\ldots,\theta_N\}$ *is* μ^1*, then the mean of the set* $\{(\theta_1+\delta)\pmod{2\pi},\ldots,(\theta_N+\delta)\pmod{2\pi}\}$ *is* $(\mu^1+\delta)\pmod{2\pi}$*.*

Property 4.2 (Average Deviation from the Mean is Zero)**.** *The sum of the sine of the deviations from the mean is zero. Specifically*

$$\sum_{i=1}^{N}\sin(\theta_i-\mu^1)=0.$$

Property 4.3 (Dispersion Monotonicity)**.** *The term* $1-\cos(\theta_i-\eta)$ *in the dispersion is a monotonically increasing function of the absolute difference* $|\theta_i-\eta|$ *for* $-\pi<\theta_i-\eta\leq\pi$*.*

Property 4.4 (Dispersion Invariance to Coordinate System)**.** *The dispersion is invariant to the choice of the zero direction. Specifically, if the dispersion of the set* $\{\theta_1,\ldots,\theta_N\}$ *about the angle* η *is* $V^1(\eta)$*, then the dispersion of the set* $\{(\theta_1+\delta)\pmod{2\pi},\ldots,(\theta_N+\delta)\pmod{2\pi}\}$ *about the angle* $(\eta+\delta)\pmod{2\pi}$ *is* $V^1(\eta)$*.*

Property 4.5 (Bounded Variance)**.** *The variance satisfies* $0\leq V^1(\mu^1)\leq 1$*.*

For distributions periodic with some other period, for example $2\pi/l$, the mean and dispersion calculations need to be adjusted. The standard method to calculate the mean is to define the complex resultant by [30]

$$z^l = \frac{1}{N}\sum_{i=1}^{N}\mathrm{e}^{jl\theta_i}. \tag{4.1}$$

The mean is thus

$$\mu^l = \frac{1}{l}\angle z^l. \tag{4.2}$$

In his initial book, Mardia claims that a descriptive formulation for the circular variance is more difficult to construct. He offers approximations of the variance for distributions clustered in a small arc and for the wrapped normal and von Mises distributions; however, he does not offer a general formula [31]. In the revised book, he ignores the topic altogether [30]. Similar to the transform applied to calculate the mean, one logical definition is

$$V^l(\eta) = \frac{1}{N}\sum_{i=1}^{N}\left[1 - \cos(l(\theta_i - \eta))\right]. \tag{4.3}$$

4.2 Directional Statistics for Edge Direction Estimation

The theory of directional statistics can be applied to direction estimation in images by finding the mean and dispersion of the gradient directions. As edges in an image are not directed vectors, this is an axial estimation problem as opposed to an angular estimation problem. Therefore, the proper equations for mean and dispersion are (4.2) and (4.3) with $l = 2$.

For a given image $f[m, n]$, express the gradient of f as a complex image

$$g[m, n] = \frac{\partial f}{\partial m} + j\frac{\partial f}{\partial n}.$$

Because f is defined on a discrete grid, this equation needs to use discrete approximations to the derivative. Form the estimate of the mean and dispersion by evaluating (4.2) and (4.3) with $\theta_i = \angle g[m_i, n_i]$ where $[m_i, n_i]$ form the lattice of sample points.

If the image f were piecewise constant, this estimation would produce reliable results; however, the gradients in areas away from the edges significantly influence the result. Because each gradient is included in (4.2), the small, but nonzero, gradients caused by noise and texture will influence the estimate as much as the gradients caused by the edges. Even worse, as the number of pixels in a block that correspond to edges is only a small percentage of the total number of pixels in the block, the influence of noise and texture will greatly outweigh the influence of the edges.

The solution to this problem lies in the proper inclusion of gradient mag-

nitudes in the calculation. One method is to only include those gradients whose magnitude is above some global threshold. While this is attractive for its consistency across the image, there may be blocks with weak edges where no gradients have large enough magnitude to meet the threshold. A second method is to adapt the threshold to each block. For example, only include those gradients whose magnitude is greater than some percentage of the maximum gradient magnitude in the block. While this method solves the problem of weak edges, the varying thresholds imply that the dispersion cannot be used as a distortion metric in the adaptive segmentation algorithm because it violates Property 3.1.

We propose a solution to rewrite (4.1), (4.2), and (4.3) as weighted averages where the weight is the square of the magnitude of the gradient. As the pixels which do not lie on an edge greatly outnumber the pixels which do lie on an edge, the magnitude must be squared in order to ensure that the contribution of the edge pixels still dominates. Specifically, the complex resultant for axial data is

$$z^w = \frac{\sum_{i=1}^{N} |g[m_i, n_i]|^2 \, \mathrm{e}^{j2\theta_i}}{\sum_{i=1}^{N} |g[m_i, n_i]|^2}. \tag{4.4}$$

The mean is

$$\mu^w = \frac{1}{2} \angle z^w. \tag{4.5}$$

The dispersion becomes

$$V^w(\eta) = \frac{\sum_{i=1}^{N} |g[m_i, n_i]|^2 \left[1 - \cos(2(\theta_i - \eta))\right]}{\sum_{i=1}^{N} |g[m_i, n_i]|^2}. \tag{4.6}$$

These formulas for the mean and dispersion satisfy the same properties as before, which will be proven following an introductory lemma.

Lemma 4.6 (Change of Modulus)**.**

$$l \left[(x) \, \left(\mathrm{mod} \ \frac{n}{l} \right) \right] = (lx) \ (\mathrm{mod} \ \ n) \, .$$

Proof. Expand the modulus using the floor function as

$$\begin{aligned} l\left[(x)\ \left(\mathrm{mod}\ \frac{n}{l}\right)\right] &= l\left[x - \frac{n}{l}\left\lfloor\frac{x}{n/l}\right\rfloor\right] \\ &= l\left[x - \frac{n}{l}\left\lfloor\frac{lx}{n}\right\rfloor\right] \\ &= lx - n\left\lfloor\frac{lx}{n}\right\rfloor \\ &= (lx)\ (\mathrm{mod}\ \ n)\,. \end{aligned}$$

■

Property 4.7 (Coordinate System Invariance)**.** *An additive change to the distribution results in merely an additive change in the mean. Specifically, if the weighted mean of the set* $\{\theta_1, \ldots, \theta_N\}$ *is* μ^w*, then the weighted mean of the set* $\{(\theta_1 + \delta)\ (\mathrm{mod}\ \ \pi), \ldots, (\theta_N + \delta)\ (\mathrm{mod}\ \ \pi)\}$ *is* $(\mu^w + \delta)\ (\mathrm{mod}\ \ \pi)$*.*

Proof. Let quantities which refer to the shifted coordinate system be denoted with a bar. The complex resultant becomes

$$\begin{aligned} \bar{z}^w &= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2\, \mathrm{e}^{j2[(\theta_i+\delta)\ (\mathrm{mod}\ \ \pi)]}}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\ &= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2\, \mathrm{e}^{j(2(\theta_i+\delta))\ (\mathrm{mod}\ \ 2\pi)}}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\ &= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2\, \mathrm{e}^{j2(\theta_i+\delta)}}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\ &= \mathrm{e}^{j2\delta}\, \frac{\sum_{i=1}^N |g[m_i, n_i]|^2\, \mathrm{e}^{j2\theta_i}}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\ &= \mathrm{e}^{j2\delta}\, z^w. \end{aligned}$$

The mean is thus

$$\begin{aligned} \bar{\mu}^w &= \frac{1}{2}\angle \bar{z}^w \\ &= \frac{1}{2}\angle\, \mathrm{e}^{j2\delta}\, z^w \\ &= \frac{1}{2}\left[(2\delta + \angle z^w)\ (\mathrm{mod}\ \ 2\pi)\right] \\ &= \frac{1}{2}\left[(2(\delta + \mu^w))\ (\mathrm{mod}\ \ 2\pi)\right] \\ &= (\delta + \mu^w)\ (\mathrm{mod}\ \ \pi)\,. \end{aligned}$$

■

Property 4.8 (Average Deviation from the Mean is Zero)**.** *The weighted sum of the sine of the deviations from the mean is zero. Specifically*

$$\sum_{i=1}^{N} |g[m_i, n_i]|^2 \sin(2(\theta_i - \mu^w)) = 0.$$

Proof.

$$\begin{aligned}
\sum_{i=1}^{N} |g[m_i, n_i]|^2 \sin(2(\theta_i - \mu^w)) &= \sum_{i=1}^{N} |g[m_i, n_i]|^2 \Big[\sin(2\theta_i)\cos(2\mu^w) \\
&\qquad - \cos(2\theta_i)\sin(2\mu^w) \Big] \\
&= \cos(2\mu^w) \sum_{i=1}^{N} |g[m_i, n_i]|^2 \sin(2\theta_i) \\
&\qquad - \sin(2\mu^w) \sum_{i=1}^{N} |g[m_i, n_i]|^2 \cos(2\theta_i) \\
&= \cos(2\mu^w) \Im\{z^w\} \sum_{i=1}^{N} |g[m_i, n_i]|^2 \\
&\qquad - \sin(2\mu^w) \Re\{z^w\} \sum_{i=1}^{N} |g[m_i, n_i]|^2 \\
&= \cos(2\mu^w) |z^w| \sin(2\mu^w) \sum_{i=1}^{N} |g[m_i, n_i]|^2 \\
&\qquad - \sin(2\mu^w) |z^w| \cos(2\mu^w) \sum_{i=1}^{N} |g[m_i, n_i]|^2 \\
&= 0. \qquad \blacksquare
\end{aligned}$$

Property 4.9 (Dispersion Monotonicity)**.** *The term* $1 - \cos(2(\theta_i - \eta))$ *in the dispersion is a monotonically increasing function of the absolute difference* $|\theta_i - \eta|$ *for* $-\frac{\pi}{2} < \theta_i - \eta \leq \frac{\pi}{2}$.

Proof. The function $\cos(x)$ is a monotonically decreasing function of $|x|$ for $-\pi < x \leq \pi$. Therefore, the function $\cos(2(\theta_i - \eta))$ is a monotonically decreasing function of $2|\theta_i - \eta|$ for $-\pi < 2(\theta_i - \eta) < \pi$. This implies that it is a monotonically decreasing function over the range $-\frac{\pi}{2} < \theta_i - \eta \leq \frac{\pi}{2}$. Therefore, $1 - \cos(2(\theta_i - \eta))$ is a monotonically increasing function of $2|\theta_i - \eta|$ over the range $-\frac{\pi}{2} < \theta_i - \eta \leq \frac{\pi}{2}$. $\blacksquare$

Property 4.10 (Dispersion Invariance to Coordinate System)**.** *The dispersion is invariant to the choice of the zero direction. Specifically, if the dispersion of the set* $\{\theta_1, \ldots, \theta_N\}$ *about the angle* η *is* $V^w(\eta)$*, then the dispersion of the set* $\{(\theta_1 + \delta) \pmod{\pi}, \ldots, (\theta_N + \delta) \pmod{\pi}\}$ *about the angle* $(\eta + \delta) \pmod{\pi}$ *is* $V^w(\eta)$*.*

Proof. Let quantities which refer to the shifted coordinate system be denoted with a bar. The dispersion becomes

$$\bar{V}^w(\eta) = \frac{\sum_{i=1}^N |g[m_i, n_i]|^2 \left[1 - \cos\left(2\left[(\theta_i + \delta) \pmod{\pi} - (\eta + \delta) \pmod{\pi}\right]\right)\right]}{\sum_{i=1}^N |g[m_i, n_i]|}.$$

As subtraction is a congruence relation for modulus, this can be written as

$$\begin{aligned}
\bar{V}^w(\eta) &= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2 \left[1 - \cos\left(2(\theta_i - \eta) \pmod{\pi}\right)\right]}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\
&= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2 \left[1 - \cos\left([2(\theta_i - \eta)] \pmod{2\pi}\right)\right]}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\
&= V^w(\eta)
\end{aligned}$$

■

Lemma 4.11. *An equivalent expression for the dispersion is* $V^w(\eta) = 1 - |z^w| \cos(2(\mu^w - \eta))$*.*

Proof. Expanding the definition of dispersion and introducing the mean provides

$$\begin{aligned}
V^w(\eta) &= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2 \left[1 - \cos(2(\theta_i - \eta))\right]}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\
&= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2 \left[1 - \cos(2(\theta_i - \mu^w + \mu^w - \eta))\right]}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\
&= \frac{\sum_{i=1}^N |g[m_i, n_i]|^2 \left[1 - \cos(2(\theta_i - \mu^w)) \cos(2(\mu^w - \eta))\right]}{\sum_{i=1}^N |g[m_i, n_i]|^2} \\
&\quad - \frac{\sin(2(\mu^w - \eta)) \sum_{i=1}^N |g[m_i, n_i]|^2 \sin(2(\theta_i - \mu^w))}{\sum_{i=1}^N |g[m_i, n_i]|^2}.
\end{aligned}$$

By Property 4.8, the second term is zero. Expanding the first term provides

$$\begin{aligned}
V^w(\eta) &= 1 - \cos(2(\mu^w - \eta))\cos(2\mu^w)\frac{\sum_{i=1}^{N}|g[m_i,n_i]|^2\cos(2\theta_i)}{\sum_{i=1}^{N}|g[m_i,n_i]|^2} \\
&\quad - \cos(2(\mu^w - \eta))\sin(2\mu^w)\frac{\sum_{i=1}^{N}|g[m_i,n_i]|^2\sin(2\theta_i)}{\sum_{i=1}^{N}|g[m_i,n_i]|^2} \\
&= 1 - \cos(2(\mu^w - \eta))\left[\cos(2\mu^w)\Re\{z^w\} - \sin(2\mu^w)\Im\{z^w\}\right] \\
&= 1 - \cos(2(\mu^w - \eta))\left[\cos(2\mu^w)|z^w|\cos(2\mu^w) + \sin(2\mu^w)|z^w|\sin(2\mu^w)\right] \\
&= 1 - |z^w|\cos(2(\mu^w - \eta)).
\end{aligned}$$

■

Property 4.12 (Bounded Variance)**.** *The variance satisfies* $0 \leq V^w(\mu^w) \leq 1$.

Proof. By Lemma 4.11, the variance is written as $V^w(\mu^w) = 1 - |z^w|$. As $|z^w| \geq 0$, $V^w(\mu^w) \leq 1$. Using the triangle inequality, the magnitude of the complex resultant is bounded by

$$\begin{aligned}
|z^w| &= \left|\frac{\sum_{i=1}^{N}|g[m_i,n_i]|^2\,\mathrm{e}^{j2\theta_i}}{\sum_{i=1}^{N}|g[m_i,n_i]|^2}\right| \\
&= \frac{\left|\sum_{i=1}^{N}|g[m_i,n_i]|^2\,\mathrm{e}^{j2\theta_i}\right|}{\sum_{i=1}^{N}|g[m_i,n_i]|^2} \\
&\leq \frac{\sum_{i=1}^{N}\left||g[m_i,n_i]|^2\,\mathrm{e}^{j2\theta_i}\right|}{\sum_{i=1}^{N}|g[m_i,n_i]|^2} \\
&= \frac{\sum_{i=1}^{N}|g[m_i,n_i]|^2|\,\mathrm{e}^{j2\theta_i}\,|}{\sum_{i=1}^{N}|g[m_i,n_i]|^2} \\
&= 1.
\end{aligned}$$

Therefore, $V^w(\mu^w) \geq 0$. ■

4.3 Edge Direction Estimation for Filter Selection and Adaptive Segmentation

The directional mean provides an average measure of the gradient over an image block B_t. As the mean has been weighted to minimize the effects of noise, this provides a good estimate of the direction perpendicular to the edge.

Therefore, the directional mean provides a mechanism to determine which geometric filter from the geometric filter dictionary should be applied. Define a function $\psi(\cdot)$ which maps an angle to its perpendicular direction. Since all calculations are axial, ψ is a single-valued, invertible function. Define an angular quantization function $Q(\cdot)$ which maps an angle to the direction of the nearest directional filter. Therefore, the proper filter to apply for most image blocks is the filter at angle $Q(\psi(\mu_t^w))$.

The directional dispersion provides a measure of how consistently the gradients of an image patch match a given direction. Specifically, the dispersion about a filter direction provides a measure of how well the edge aligns with that filter. If the dispersion is high, there is no consistent directional structure and μ_t^w is nearly arbitrary. For these blocks, the smallest prediction residual occurs if the filter is most localized. Therefore, if $V_t(\psi^{-1}(Q(\psi(\mu_t^w)))) \geq 0.8$, apply direction seven.

The directional mean and dispersion also provide intuitive formulations for the rate and distortion function in the adaptive segmentation. The adaptive parameter α_t is merely which filter from the geometric filter dictionary is applied. This is equivalent to the quantized mean such that $\alpha_t = Q(\psi(\mu_t^w))$. The optimal rate is some encoding that accounts for the probability of each filter from the geometric filter dictionary being applied. As this is not known, we assume equal probabilities for the 12 filters and set $r(\alpha_t) = \log_2(12)$.

The distortion function is a function of the dispersion as

$$d(f(B_t), \alpha_t) = V^w(\psi^{-1}(\alpha_t)) \sum_{m,n \in B_t} |g[m,n]|^2. \tag{4.7}$$

In order to be a valid distortion function, (4.7) must be nonnegative and satisfy Property 3.1. As $|z^w| \leq 1$ as seen in the proof of Property 4.12 and $\cos(\cdot) \leq 1$, Lemma 4.11 implies that $V^w(\eta) \geq 0$. Therefore, the distortion is nonnegative. The presence of the quantization function affects the ability to prove Property 3.1. Therefore, we will prove it for the case of fine quantization, i.e. $\psi^{-1}(Q(\psi(\mu^w))) \approx \mu^w$. This is equivalent to invoking the small angle approximation on the term $2(\mu^w - \psi^{-1}(Q(\psi(\mu^w))))$.

Theorem 4.13. *For fine quantization, the distortion metric defined in* (4.7) *satisfies Property 3.1.*

Proof. Express the quantized mean $\psi^{-1}(Q(\psi(\mu_t^w)))$ as $\hat{\mu}_t^w$. By Lemma 4.11,

the distortion can be written as

$$d(f(B_t), \alpha_t) = (1 - |z^w|) \cos(2(\mu^w - \hat{\mu}_t^w)) \sum_{m,n \in B_t} |g[m,n]|^2.$$

By the small angle approximation and the non-overlapping nature of the blocks created by the children of node t, this becomes

$$\begin{aligned} d(f(B_t), \alpha_t) &= (1 - |z^w|) \sum_{m,n \in B_t} |g[m,n]|^2 \\ &= \sum_{m,n \in B_t} |g[m,n]|^2 - \left| \sum_{m,n \in B_t} |g[m,n]|^2 \, \mathrm{e}^{j2\theta_{mn}} \right| \\ &= \sum_{m,n \in B_t} |g[m,n]|^2 - \left| \sum_{t' \in \tilde{T}_t} \sum_{m,n \in B_{t'}} |g[m,n]|^2 \, \mathrm{e}^{j2\theta_{mn}} \right|. \end{aligned}$$

By the triangle inequality

$$d(f(B_t), \alpha_t) \geq \sum_{m,n \in B_t} |g[m,n]|^2 - \sum_{t' \in \tilde{T}_t} \left| \sum_{m,n \in B_{t'}} |g[m,n]|^2 \, \mathrm{e}^{j2\theta_{mn}} \right|.$$

Using the definition of the complex resultant and simplifying

$$\begin{aligned} d(f(B_t), \hat{\mu}_t^w) &\geq \sum_{m,n \in B_t} |g[m,n]|^2 - \sum_{t' \in \tilde{T}_t} |z_{t'}^w| \sum_{m,n \in B_{t'}} |g[m,n]|^2 \\ &= \sum_{t' \in \tilde{T}_t} (1 - |z_{t'}^w|) \sum_{m,n \in B_{t'}} |g[m,n]|^2. \end{aligned}$$

By the small angle approximation, we can insert $\cos(2(\hat{\mu}_{t'}^w - \mu_{t'}^w))$ as

$$\begin{aligned} d(f(B_t), \alpha_t) &\geq \sum_{t' \in \tilde{T}_t} (1 - |z_{t'}^w| \cos(2(\hat{\mu}_{t'}^w - \mu_{t'}^w))) \sum_{m,n \in B_{t'}} |g[m,n]|^2 \\ &= \sum_{t' \in \tilde{T}_t} V_{t'}^w(\hat{\mu}_{t'}^w) \sum_{m,n \in B_{t'}} |g[m,n]|^2 \\ &= \sum_{t' \in \tilde{T}_t} d(f(B_{t'}), \mu_{t'}^w). \end{aligned}$$

■

Furthermore, as neither the rate nor the distortion function depend upon the output of any adaptive filter, the filters are able to use data from across internal boundaries as described in Section 3.3.

CHAPTER 5

RESULTS

Validation of the directional image representation described in Chapters 2–4 consists of two components. The first is to validate the direction estimation and adaptive segmentation algorithms to ensure that the specified direction is reasonable and that the block size properly adapts to changes in the dominant direction. The second is to use directional lifting in a nonlinear approximation framework and compare the reconstruction PSNR with the 5/3 wavelet.

5.1 Direction Estimation and Adaptive Segmentation

The direction estimation and adaptive segmentation algorithms were verified using two different tests. The first test was an objective test of the direction estimation algorithm on a set of simple synthetic images. The second test was a subjective test of both the direction estimation and adaptive segmentation algorithms on both synthetic and real images.

5.1.1 Objective Test of Direction Estimation

The direction estimation algorithm was tested using the Monte Carlo method over a set of simple synthetic images. The synthetic images were created to consist of two regions separated by an edge at a random angle γ. In each region, the image was a bilinear polynomial of class $\Pi_{1,1}$. The two polynomials were chosen randomly and independently. Therefore, there was an arbitrary discontinuity along the edge. In addition, white Gaussian noise with a variance of 0.05 was then added to each image. Some example test images are shown in Figure 5.1.

A collection of 100 000 independent images of this type were created. For

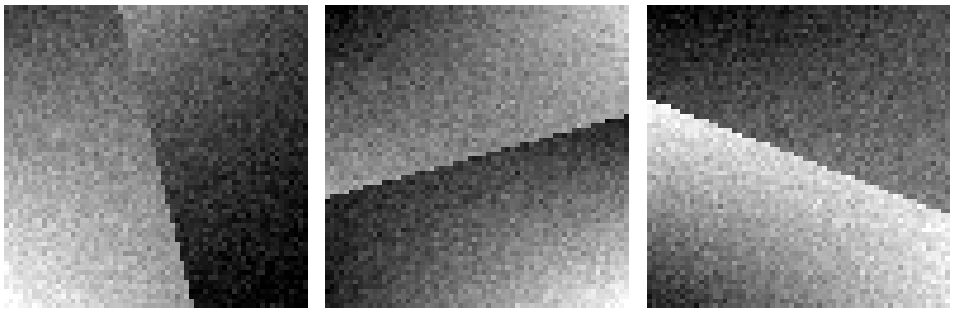


Figure 5.1: Typical test images consisting of two bilinear polynomials separated by an edge at a random angle with added noise.

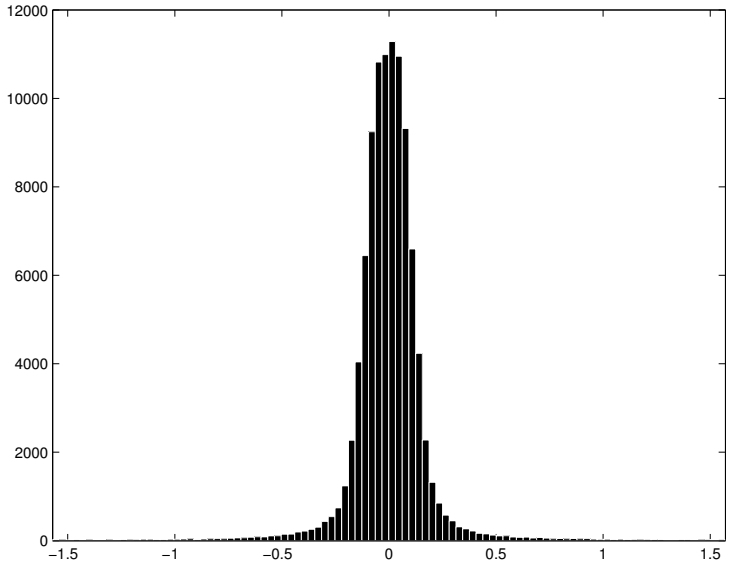


Figure 5.2: Histogram of the error $\mu^w - \gamma$ over 100 000 independent trials.

each image, we calculated the weighted mean μ^w. The error between γ and μ_w was computed for each image. Because this error is a difference between two axial random variables, it is also an axial random variable. The histogram of the error is shown in Figure 5.2. The circular variance $V^2(\mu^2)$ of the error is 0.058. As this is near zero, μ^w is a good estimate of γ.

5.1.2 Subjective Test of Direction Estimation and Adaptive Segmentation

We then tested the direction estimation and adaptive segmentation algorithms on various test images. The first image was a synthetic bilinear phantom. This image was created by randomly defining a bilinear polynomial

Figure 5.3: Example of bilinear phantom test image.

over each section of the modified Shepp-Logan phantom. As the polynomial defined for each section is independent of all of the others, arbitrary discontinuities are created along the edges. An example of this phantom is shown in Figure 5.3.

The result for the direction estimation and adaptive segmentation algorithm with $\lambda = 0$ in (3.1) is shown in Figure 5.4. If $\lambda = 0.001$, the result is shown in Figure 5.5. In both cases, the estimated direction seems to align with the edge directions. Furthermore, the segmentation uses larger blocks in areas which contain fewer edges or have more consistent edges. When λ is increased, the algorithm smoothly combines some blocks while ensuring that the estimated direction still aligns with the edge. We see similar results for the segmentation and direction estimation of Barbara in Figures 5.6 and 5.7.

5.2 Nonlinear Approximation

The entire algorithm was tested in a compression nonlinear approximation framework. Both our directional representation and the 5/3 wavelet were iterated six times. The image was then reconstructed from a subset of the coefficients, where $\mathcal{I}_M$ consists of the largest M coefficients. The first test image was the bilinear phantom described above. Using the segmentation shown in Figure 5.5, the nonlinear approximation curves for both our algo-

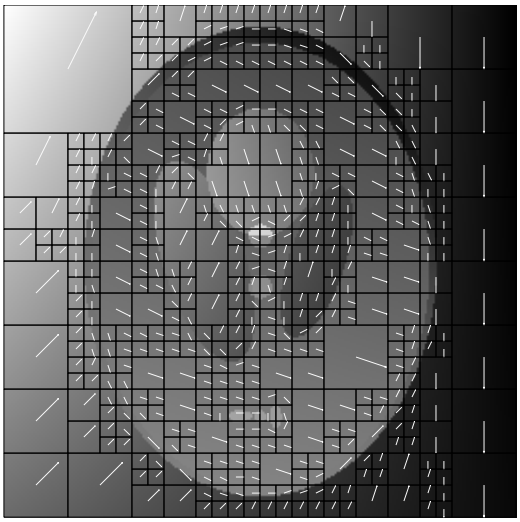


Figure 5.4: Direction estimation and adaptive segmentation of the bilinear phantom when $\lambda = 0$. The adaptive overhead for this segmentations is 2728 bits.

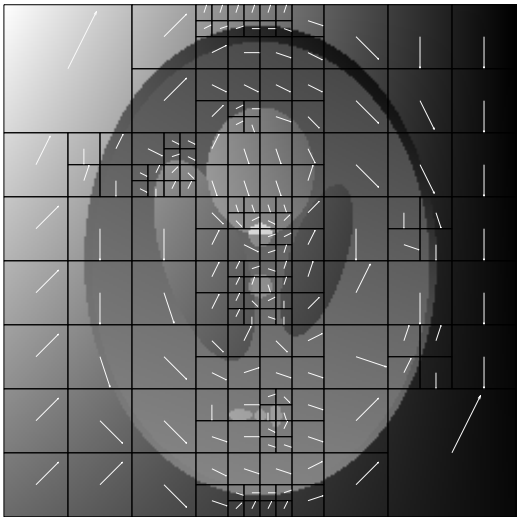


Figure 5.5: Direction estimation and adaptive segmentation of the bilinear phantom when $\lambda = 0.001$. The adaptive overhead for this segmentation is 920 bits.

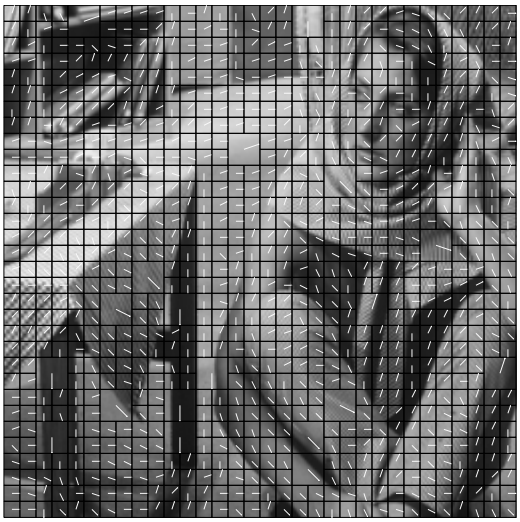


Figure 5.6: Direction estimation and adaptive segmentation of Barbara when $\lambda = 0$. The adaptive overhead for this segmentation is 5160 bits.

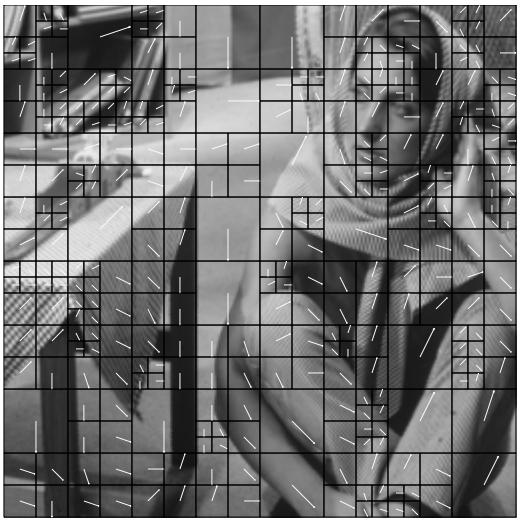


Figure 5.7: Direction estimation and adaptive segmentation of Barbara when $\lambda = 0.01$. The adaptive overhead for this segmentation is 1816 bits.

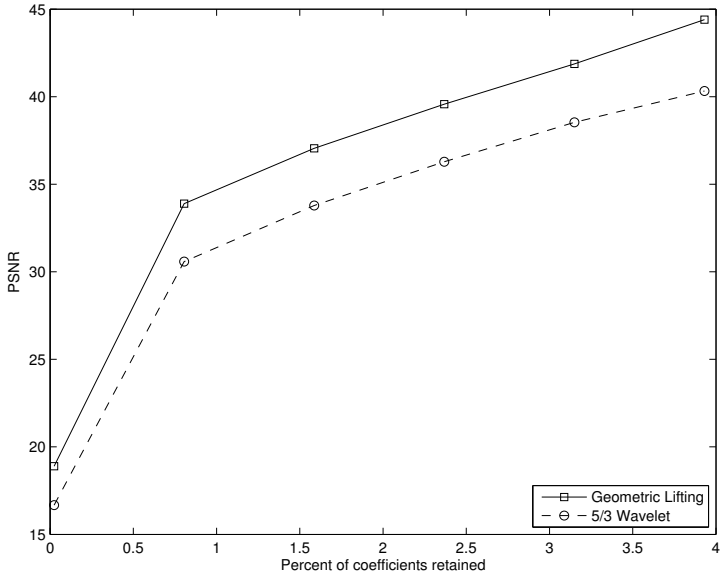
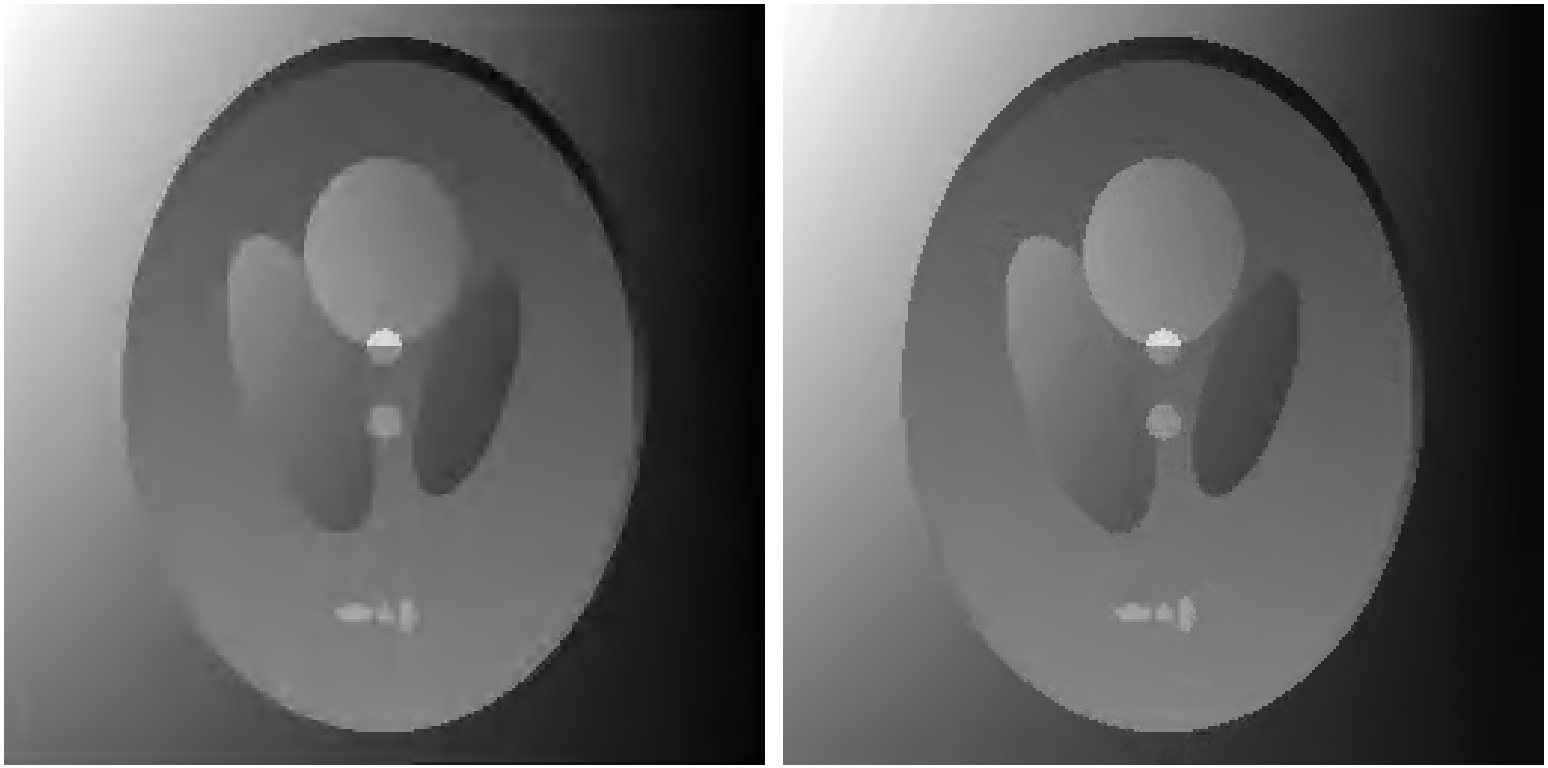


Figure 5.8: Nonlinear approximation of the bilinear phantom with six levels of iteration.

rithm and the 5/3 wavelet are shown in Figure 5.8. Our algorithm outperformed the 5/3 wavelet by 2–4.5 dB over the entire range. In addition, the edges are subjectively much sharper. Examples of the reconstruction with approximately 3% of the coefficients used are shown in Figure 5.9.

The second test case was the standard test image Barbara. Using the segmentation in Figure 5.7, the nonlinear approximation curves are shown in Figure 5.10. For any reasonable reconstruction quality, the 5/3 wavelet outperformed our algorithm by 0–2.5 dB. Subjectively, there is almost no difference between the reconstruction results. An example reconstruction from both algorithms with approximately 13% of the coefficients used is shown in Figure 5.11.



(a) 5/3 Wavelet

(b) Geometric Lifting

Figure 5.9: Comparison of the reconstruction from approximately 3% of the coefficients using geometric lifting and the 5/3 wavelet. Notice the improved sharpness along the edges by using geometric lifting.

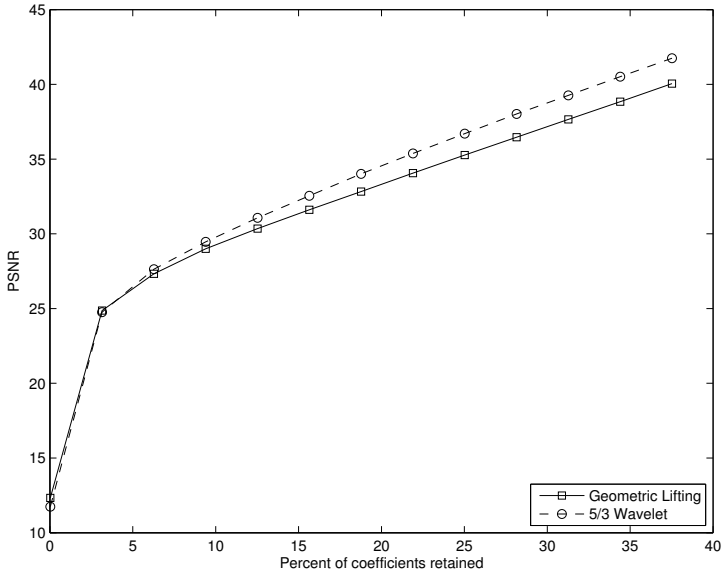


Figure 5.10: Nonlinear approximation of Barbara with six levels of iteration.

(a) 5/3 Wavelet

(b) Geometric Lifting

Figure 5.11: Comparison of the reconstruction from approximately 13% of the coefficients using geometric lifting and the 5/3 wavelet.

CHAPTER 6

DIMENSIONAL DEGENERACY

6.1 Empirical Results

We designed prediction filters using the method of Section 2.3 for the fat filter support shown in Figure 2.5. Whenever the predicted point and two of the points used in the prediction were on the same scanline, the filter degenerated to a one-dimensional filter along that scanline. This problem affected eight of the twelve directions in phase one and phase two. As phase three points are not on the same scanline as phase zero points, this problem does not affect phase three. The filter supports for each filter, including both the points which were available to the filter creation algorithm and the points that were actually used, are shown in Figures 6.1–6.3. Additional filter diagrams for the prediction filters along with the diagrams for the equivalent analysis and synthesis filters are shown in Appendix A.

6.2 Theoretical Results

Recall from (2.3) that the filter response $\boldsymbol{h}$ must satisfy $\boldsymbol{p} = \boldsymbol{R}\boldsymbol{h}$ where $\boldsymbol{p}$ is the location of the predicted point and

$$\boldsymbol{R} = \begin{bmatrix} 1 & \cdots & 1 \\ m_1 & \cdots & m_N \\ n_1 & \cdots & n_N \\ m_1 n_1 & \cdots & m_N n_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{r}_1 & \cdots & \boldsymbol{r}_N \end{bmatrix}$$

is the matrix of the locations of the N points used in the prediction.

Theorem 6.1. *If a nondegenerate set of N points is used to form a bilinear prediction such that all but two of the N points lie on the same scanline as*

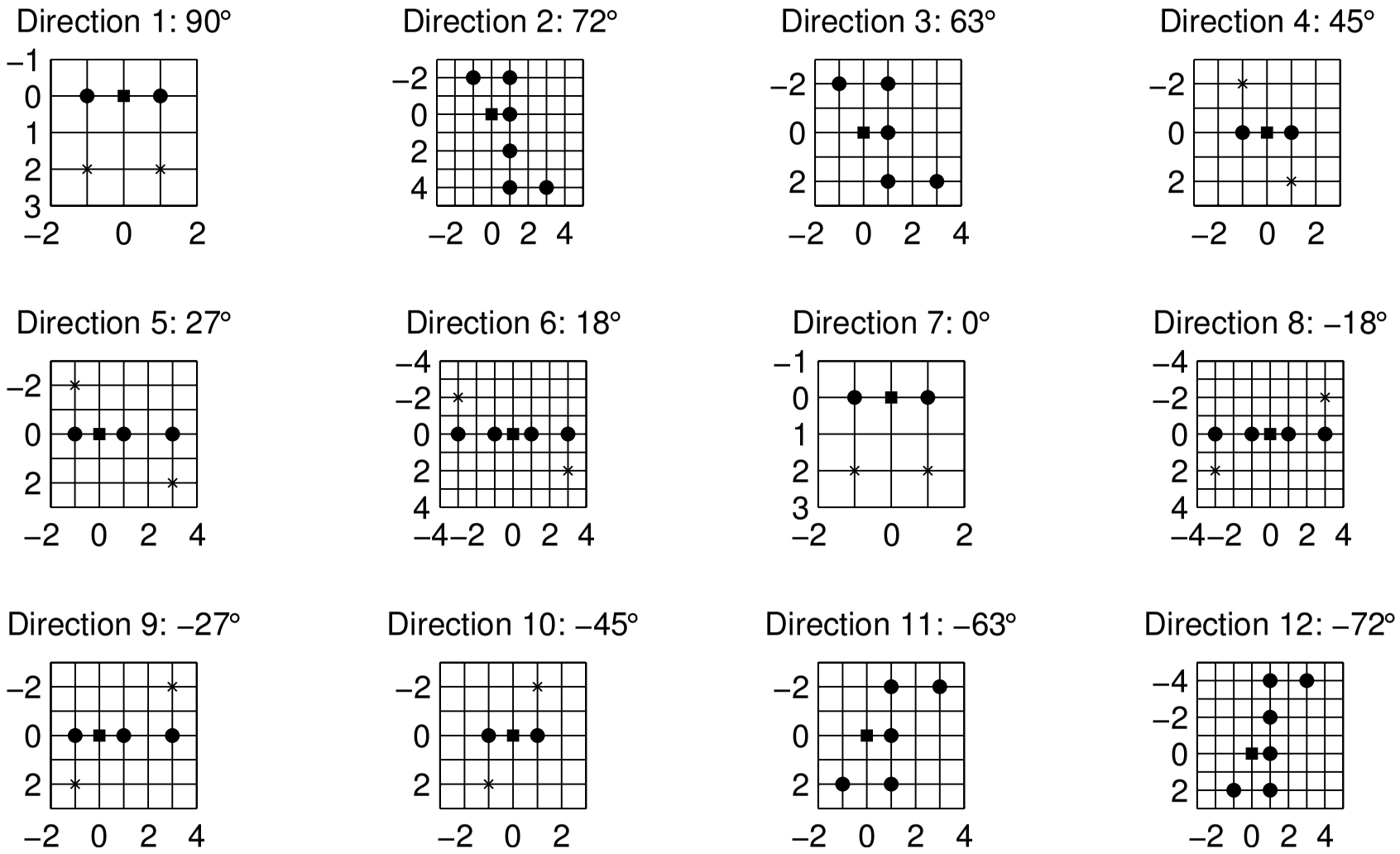


Figure 6.1: Demonstration of the dimensional degeneracy of the phase one filters. The squares represent the predicted locations. The circles represent the phase zero locations which were used in the prediction. The crosses represent the phase zero locations which the algorithm was allowed to use, yet were given a zero filter weight.

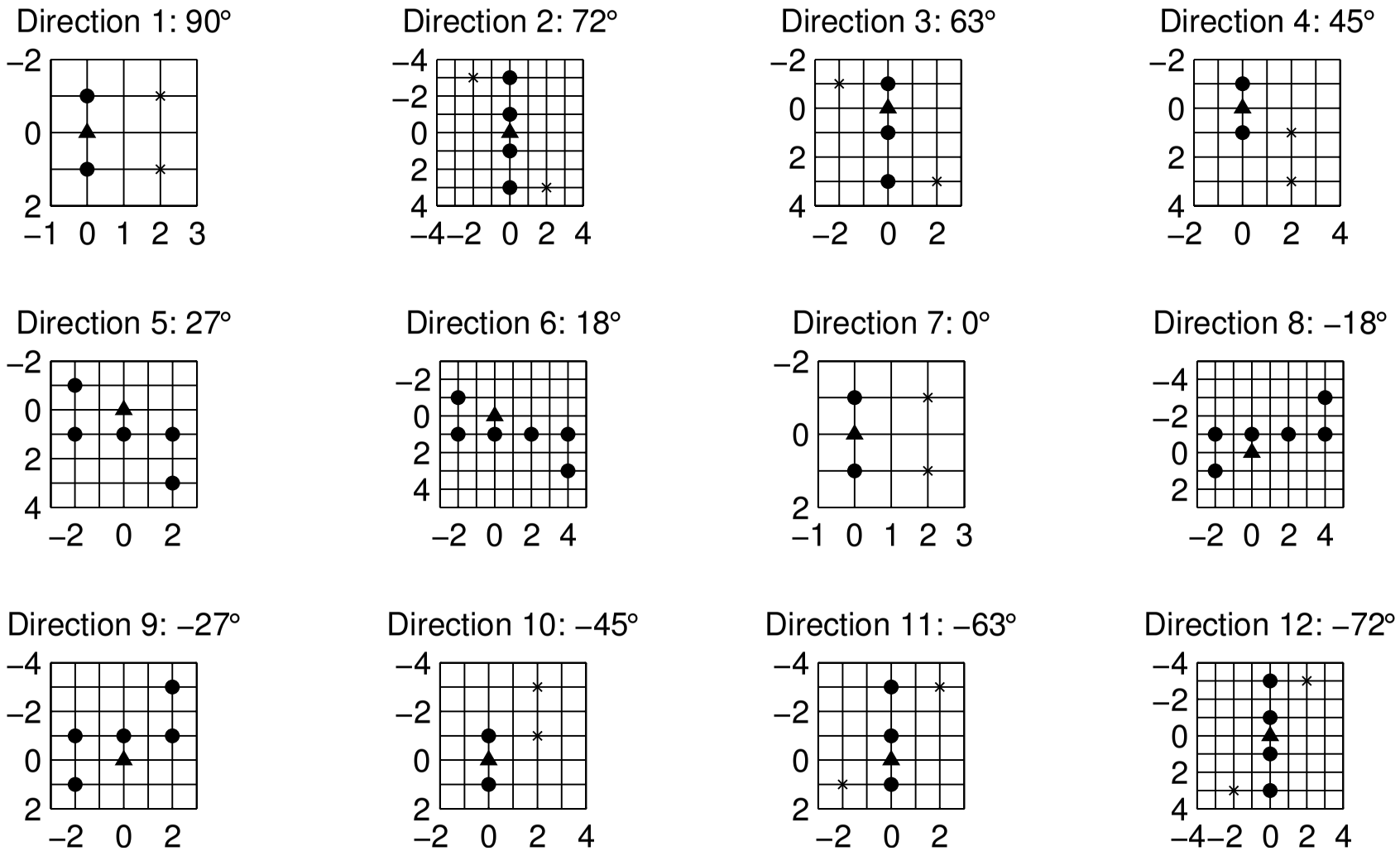


Figure 6.2: Demonstration of the dimensional degeneracy of the phase two filters. The triangles represent the predicted locations. The circles represent the phase zero locations which were used in the prediction. The crosses represent the phase zero locations which the algorithm was allowed to use, yet were given a zero filter weight.

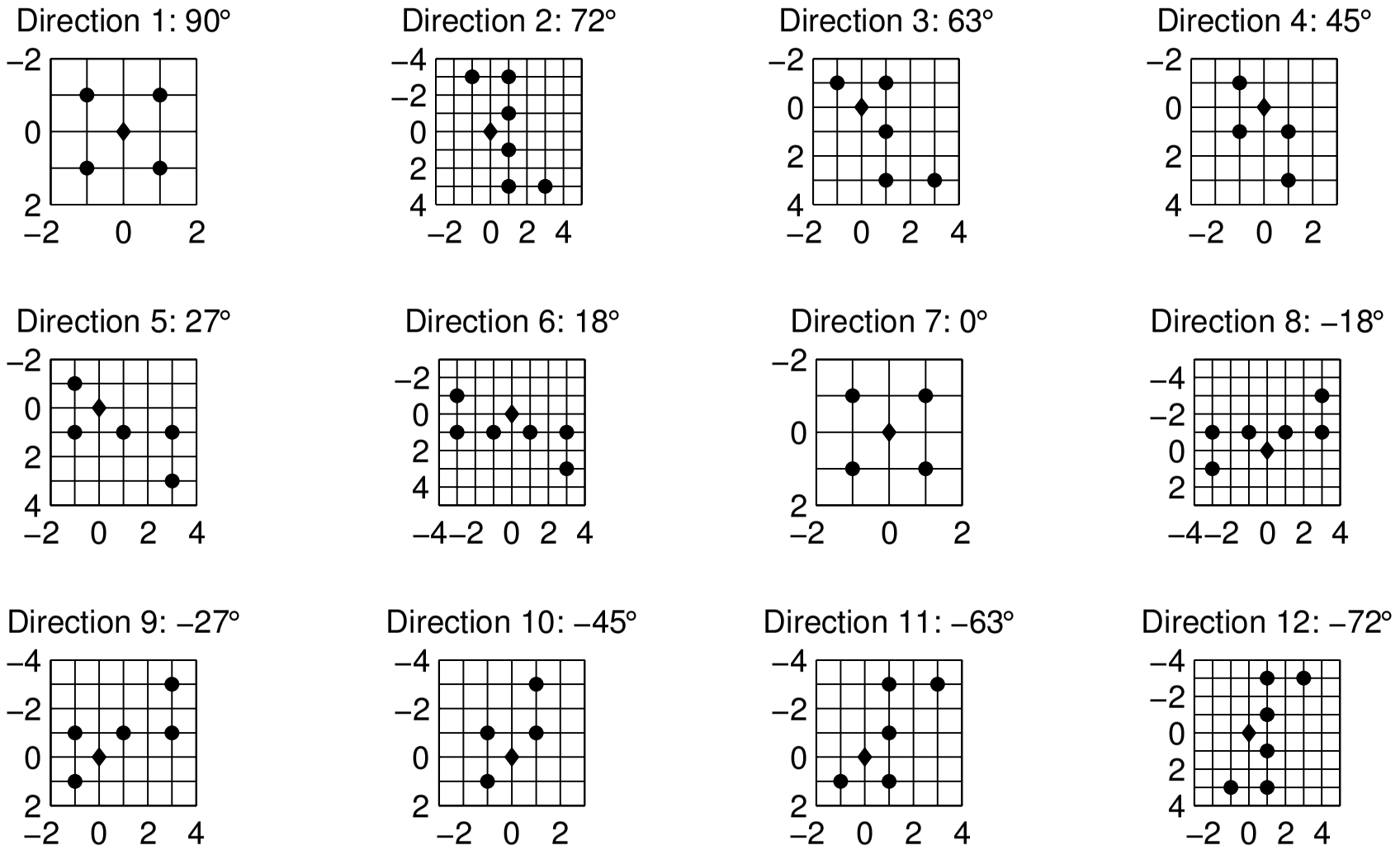


Figure 6.3: Demonstration that phase three does not suffer from dimensional degeneracy. The diamonds represent the predicted locations. The circles represent the phase zero locations which were used in the prediction.

the predicted point, the filter will degenerate to a one-dimensional filter.

Proof. Without loss of generality, assume that points $3, \ldots, N$ are on the same scanline as the predicted point. Let

$$W = \operatorname*{span}_{i=3}^{N} \{\boldsymbol{r}_i\}.$$

By assumption the first element and either the second or third element of all of the vectors $\boldsymbol{r}_3, \ldots, \boldsymbol{r}_N$ are the same. Therefore, the dimension of W is two. Let $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ be a basis of W. Because the system is not degenerate, the rank of $\boldsymbol{R}$ is four. Because W is dimension two, the vectors $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are independent of each other and the basis functions of W. Let h_i refer to the filter tap at location i. As $\boldsymbol{p} = \boldsymbol{R}\boldsymbol{h}$, we know that

$$\begin{aligned} 0 &= \boldsymbol{R}\boldsymbol{h} - \boldsymbol{p} \\ &= h_1\boldsymbol{r}_1 + h_2\boldsymbol{r}_2 + \sum_{i=3}^{N} h_i\boldsymbol{r}_i - \boldsymbol{p}. \end{aligned}$$

As

$$\sum_{i=3}^{N} h_i\boldsymbol{r}_i - \boldsymbol{p} \in W,$$

there exists $\bar{h}_1$ and $\bar{h}_2$ such that

$$\sum_{i=3}^{N} h_i\boldsymbol{r}_i - \boldsymbol{p} = \bar{h}_1\boldsymbol{w}_1 + \bar{h}_2\boldsymbol{w}_2.$$

Therefore, the above equation becomes

$$0 = \boldsymbol{r}_1 h_1 + \boldsymbol{r}_2 h_2 + \bar{h}_1\boldsymbol{w}_1 + \bar{h}_2\boldsymbol{w}_2.$$

As these four vectors are linearly independent, their coefficients must be zero. Specifically, the filter taps h_1 and h_2 are zero. Since points $3, \ldots, N$ lie on a single scanline, the filter has degenerated to a one-dimensional filter. ■

This theorem shows that filters designed using parallelograms as in Section 2.2 are very likely to degenerate. As the length of the scanline vector is only two pixels, there is likely to be a line of points down the middle of the parallelogram complemented by one point on each acute corner. There-

fore, a different paradigm for the creation of directional support is needed. The hypotheses of Theorem 6.1 are contradicted by two conditions: either the predicted point cannot lie on a scanline with two other points or there must be more than two points off the scanline. As the first condition implies the second, this can only be solved by forcing more points to not lie on the scanline. Because of the constraints of a discrete grid, this will require a compromise in either directionality or locality.

CHAPTER 7

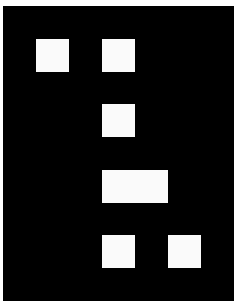
CONCLUSION

In this thesis we explored sparse representations for two-dimensional signals using directional information and the issues encountered in adaptive directional transforms. Two-dimensional natural images are typically defined by the presence of one-dimensional singularity curves in the signal. Current sparse representations, such as wavelets, do not account for these singularity curves. Geometric wavelets perform better by incorporating directionality, but their frequency domain formulation forces their basis functions to have long support. Separable directional lifting, on the other hand, has local support, but is only able to filter along one-dimensional lines.

We presented a formulation for a directional image representation using nonseparable lifting. We proved that single-stage lifting schemes are incompatible with orthonormal filter banks except in trivial cases. Through the lifting framework, we were able to maintain locality while designing nonseparable, directional, biorthogonal filters. We designed filter support aligned with many angular directions in order that the support would intersect edges fewer times. From this support, we demonstrated a method to design prediction filters with two vanishing moments which most closely matched a bilinear polynomial and minimized the norm of the filter. Using an exhaustive search algorithm over the set of realistic update filters and scaling constants, we determined the proper parameters for the remainder of the lifting scheme which minimized the divergence from orthonormality.

We applied various techniques from classification and directional statistics to the problem of adaptive directional transforms. We presented the BFOS tree pruning algorithm along with its necessary properties to ensure a valid adaptive segmentation for block based transforms. We extended the equations for estimation of the directional mean and variance to robustly estimate the dominant direction of an image block.

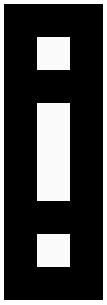
The results showed that the adaptive segmentation and direction estima-



-4	0.167	0	-0.283	0	0
-3	0	0	0	0	0
-2	0	0	-0.233	0	0
-1	0	0	0	0	0
0	0	0	-0.183	1.000	0
1	0	0	0	0	0
2	0	0	-0.133	0	-0.333
	-4	-3	-2	-1	0

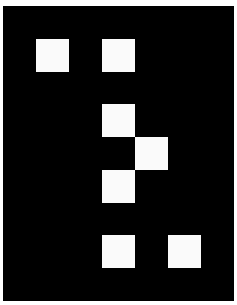
Figure 7.1: Direction 2: Analysis filter for phase one. In the filter support diagram, white indicates a nonzero coefficient, while black indicates a zero coefficient. The exact values of the coefficients are enumerated in the adjoining table.

tion algorithms robustly segmented an image into regions of directional bias and estimated the direction of each region; however, the directional representation did not improve upon state-of-the-art transforms in image nonlinear approximation. We feel that this result is caused by the combination of three effects. First, geometric filters cannot be as compactly supported as filters along the scanline. Because all data analysis happens on a discrete grid, prediction along angles which are not aligned with the scanlines requires that the data come from pixels which are farther away. This causes the approximately locally linear assumption to break down, as the definition of local must include pixels which are farther away. Second, if the directional support consists only of a collection of locations on the same scanline as the predicted pixel and two pixels off that scanline, the filter degenerates to a one-dimensional filter, as demonstrated in Chapter 6. Maintaining directional filters then requires the addition of additional filter locations, which force the support to be less directional or less local. Third, the structure of a single-stage lifting scheme forces the equivalent analysis filters to have punctured support, for example, the equivalent analysis filters for direction two shown in Figures 7.1–7.3 and additional examples in Appendix A. This punctured support means that there are many pixels which are local and align with the proper direction which are not used in the prediction.



-4	-0.250
-3	0
-2	-0.250
-1	1.000
0	-0.250
1	0
2	-0.250
	0

Figure 7.2: Direction 2: Analysis filter for phase two. The format is the same as Figure 7.1.



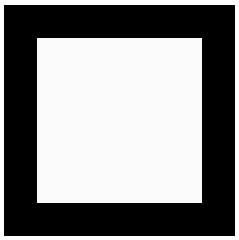
-4	0.250	0	-0.475	0	0
-3	0	0	0	0	0
-2	0	0	-0.325	0	0
-1	0	0	0	1.000	0
0	0	0	-0.175	0	0
1	0	0	0	0	0
2	0	0	-0.025	0	-0.250
	-4	-3	-2	-1	0

Figure 7.3: Direction 2: Analysis filter for phase three. The format is the same as Figure 7.1.

APPENDIX A

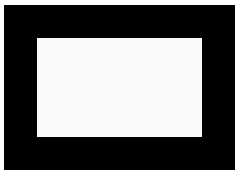
FILTER DIAGRAMS

For each geometric filter support shown in Figure 2.5, the prediction filters were designed according to (2.2). The update filters were chosen to provide two vanishing moments. According to [26], this implies that the update filters are $\frac{1}{4}$ the adjoint of the prediction filters. Using the relationship between a lifting scheme and a filter bank, the equivalent analysis and synthesis filters can be derived. This appendix demonstrates the filter support and filter coefficients for the prediction, equivalent analysis, and equivalent synthesis filters for each direction. In the filter support diagrams, white indicates a nonzero coefficient, while black indicates a zero coefficient. The exact values of the coefficients are enumerated in the adjoining table. The equivalent analysis and synthesis filters for the 5/3 wavelet are presented in Figures A.1–A.8, followed by the geometric filters for each direction in Figures A.9–A.140.



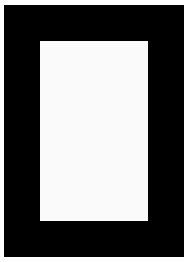
-2	0.031	-0.063	-0.188	-0.063	0.031
-1	-0.063	0.125	0.375	0.125	-0.063
0	-0.188	0.375	1.125	0.375	-0.188
1	-0.063	0.125	0.375	0.125	-0.063
2	0.031	-0.063	-0.188	-0.063	0.031
	-2	-1	0	1	2

Figure A.1: 5/3 Wavelet: Analysis filter for phase zero.



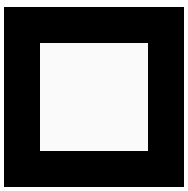
-1	-0.063	0.125	0.375	0.125	-0.063
0	0.125	-0.250	-0.750	-0.250	0.125
1	-0.063	0.125	0.375	0.125	-0.063
	-2	-1	0	1	2

Figure A.2: 5/3 Wavelet: Analysis filter for phase one.



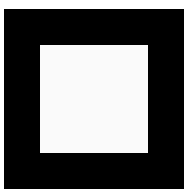
-2	-0.063	0.125	-0.063
-1	0.125	-0.250	0.125
0	0.375	-0.750	0.375
1	0.125	-0.250	0.125
2	-0.063	0.125	-0.063
	-1	0	1

Figure A.3: 5/3 Wavelet: Analysis filter for phase two.



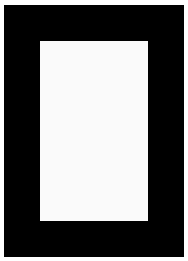
-1	0.125	-0.250	0.125
0	-0.250	0.500	-0.250
1	0.125	-0.250	0.125
	-1	0	1

Figure A.4: 5/3 Wavelet: Analysis filter for phase three.



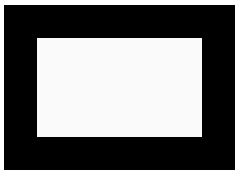
-1	0.125	0.250	0.125
0	0.250	0.500	0.250
1	0.125	0.250	0.125
	-1	0	1

Figure A.5: 5/3 Wavelet: Synthesis filter for phase zero.



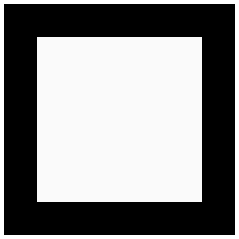
-2	0.063	0.125	0.063
-1	0.125	0.250	0.125
0	-0.375	-0.750	-0.375
1	0.125	0.250	0.125
2	0.063	0.125	0.063
	-1	0	1

Figure A.6: 5/3 Wavelet: Synthesis filter for phase one.



-1	0.063	0.125	-0.375	0.125	0.063
0	0.125	0.250	-0.750	0.250	0.125
1	0.063	0.125	-0.375	0.125	0.063
	-2	-1	0	1	2

Figure A.7: 5/3 Wavelet: Synthesis filter for phase two.

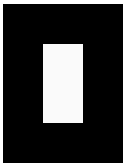


-2	0.031	0.063	-0.188	0.063	0.031
-1	0.063	0.125	-0.375	0.125	0.063
0	-0.188	-0.375	1.125	-0.375	-0.188
1	0.063	0.125	-0.375	0.125	0.063
2	0.031	0.063	-0.188	0.063	0.031
	-2	-1	0	1	2

Figure A.8: 5/3 Wavelet: Synthesis filter for phase three.

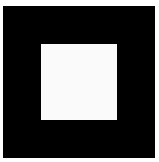
0	0.500	0.500
	-1	0

Figure A.9: Direction 1: Prediction filter for phase one.



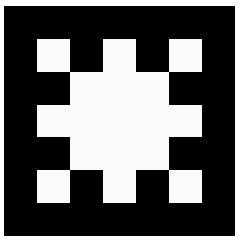
-1	0.500
0	0.500
	0

Figure A.10: Direction 1: Prediction filter for phase two.



-1	0.250	0.250
0	0.250	0.250
	-1	0

Figure A.11: Direction 1: Prediction filter for phase three.

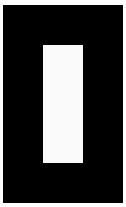


-2	-0.031	0	-0.188	0	-0.031
-1	0	0.125	0.250	0.125	0
0	-0.188	0.250	1.375	0.250	-0.188
1	0	0.125	0.250	0.125	0
2	-0.031	0	-0.188	0	-0.031
	-2	-1	0	1	2

Figure A.12: Direction 1: Analysis filter for phase zero.

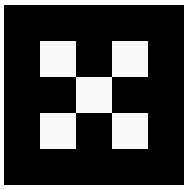
0	-0.500	1.000	-0.500
	-2	-1	0

Figure A.13: Direction 1: Analysis filter for phase one.



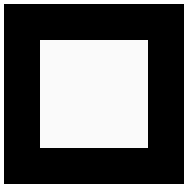
-2	-0.500
-1	1.000
0	-0.500
	0

Figure A.14: Direction 1: Analysis filter for phase two.



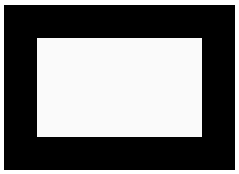
-2	-0.250	0	-0.250
-1	0	1.000	0
0	-0.250	0	-0.250
	-2	-1	0

Figure A.15: Direction 1: Analysis filter for phase three.



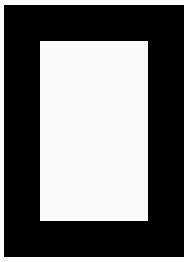
-1	0.125	0.250	0.125
0	0.250	0.500	0.250
1	0.125	0.250	0.125
	-1	0	1

Figure A.16: Direction 1: Synthesis filter for phase zero.



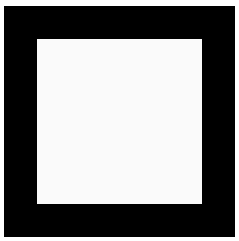
-1	-0.031	-0.063	-0.063	-0.063	-0.031
0	-0.063	-0.125	0.875	-0.125	-0.063
1	-0.031	-0.063	-0.063	-0.063	-0.031
	-1	0	1	2	3

Figure A.17: Direction 1: Synthesis filter for phase one.



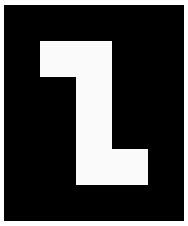
-1	-0.031	-0.063	-0.031
0	-0.063	-0.125	-0.063
1	-0.063	0.875	-0.063
2	-0.063	-0.125	-0.063
3	-0.031	-0.063	-0.031
	-1	0	1

Figure A.18: Direction 1: Synthesis filter for phase two.



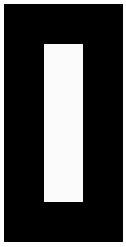
-1	-0.016	-0.031	-0.031	-0.031	-0.016
0	-0.031	-0.063	-0.063	-0.063	-0.031
1	-0.031	-0.063	0.938	-0.063	-0.031
2	-0.031	-0.063	-0.063	-0.063	-0.031
3	-0.016	-0.031	-0.031	-0.031	-0.016
	-1	0	1	2	3

Figure A.19: Direction 1: Synthesis filter for phase three.



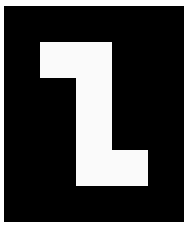
-2	-0.167	0.283	0
-1	0	0.233	0
0	0	0.183	0
1	0	0.133	0.333
	-2	-1	0

Figure A.20: Direction 2: Prediction filter for phase one.



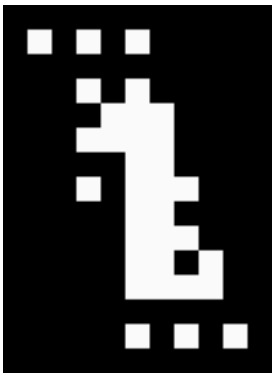
-2	0.250
-1	0.250
0	0.250
1	0.250
	0

Figure A.21: Direction 2: Prediction filter for phase two.



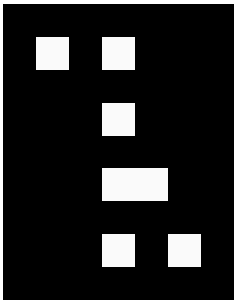
-2	-0.250	0.475	0
-1	0	0.325	0
0	0	0.175	0
1	0	0.025	0.250
	-2	-1	0

Figure A.22: Direction 2: Prediction filter for phase three.



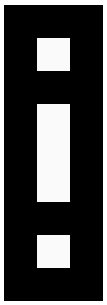
-6	0.059	0	-0.092	0	-0.056	0	0	0	0
-5	0	0	0	0	0	0	0	0	0
-4	0	0	-0.042	0	-0.150	0	0	0	0
-3	0	0	0	0.125	0.125	0.012	0	0	0
-2	0	0	0.008	0.167	-0.268	0.067	0	0	0
-1	0	0	0	0	0.125	0.087	0	0	0
0	0	0	0.058	0	1.469	0.092	0.058	0	0
1	0	0	0	0	0.125	0.162	0	0	0
2	0	0	0	0	-0.268	0.117	0.008	0	0
3	0	0	0	0	0.125	0.237	0	-0.125	0
4	0	0	0	0	-0.150	0.142	-0.042	-0.083	0
5	0	0	0	0	0	0	0	0	0
6	0	0	0	0	-0.056	0	-0.092	0	0.059
	-4	-3	-2	-1	0	1	2	3	4

Figure A.23: Direction 2: Analysis filter for phase zero.



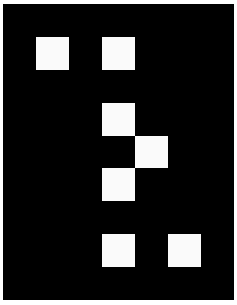
-4	0.167	0	-0.283	0	0
-3	0	0	0	0	0
-2	0	0	-0.233	0	0
-1	0	0	0	0	0
0	0	0	-0.183	1.000	0
1	0	0	0	0	0
2	0	0	-0.133	0	-0.333
	-4	-3	-2	-1	0

Figure A.24: Direction 2: Analysis filter for phase one.



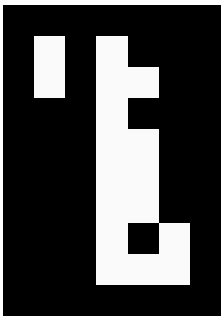
-4	-0.250
-3	0
-2	-0.250
-1	1.000
0	-0.250
1	0
2	-0.250
	0

Figure A.25: Direction 2: Analysis filter for phase two.



-4	0.250	0	-0.475	0	0
-3	0	0	0	0	0
-2	0	0	-0.325	0	0
-1	0	0	0	1.000	0
0	0	0	-0.175	0	0
1	0	0	0	0	0
2	0	0	-0.025	0	-0.250
	-4	-3	-2	-1	0

Figure A.26: Direction 2: Analysis filter for phase three.



-4	-0.083	0	0.142	0	0
-3	-0.125	0	0.237	0.125	0
-2	0	0	0.117	0	0
-1	0	0	0.162	0.125	0
0	0	0	0.092	0.500	0
1	0	0	0.087	0.125	0
2	0	0	0.067	0	0.167
3	0	0	0.012	0.125	0.125
	-3	-2	-1	0	1

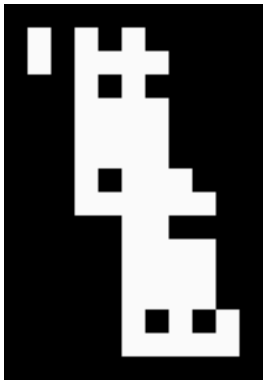
Figure A.27: Direction 2: Synthesis filter for phase zero.

-6	0.014	0	-0.018	0	-0.009	0	0	0	0
-5	0.021	0	-0.031	-0.021	-0.016	-0.008	0	0	0
-4	0	0	-0.012	0	-0.021	0	0	0	0
-3	0	0	-0.016	-0.021	-0.033	-0.020	0	0	0
-2	0	0	-0.006	-0.083	-0.033	-0.033	0	0	0
-1	0	0	0	-0.021	-0.048	-0.034	0	0	0
0	0	0	0.001	0	0.919	-0.046	0.001	0	0
1	0	0	0.016	-0.021	-0.093	-0.052	0.011	0.010	0
2	0	0	0	0	-0.033	-0.058	-0.006	0	0
3	0	0	0	0	-0.034	-0.044	0.002	0.010	0
4	0	0	0	0	-0.021	-0.071	-0.012	0.042	0
5	0	0	0	0	-0.014	-0.032	-0.007	0.010	0
6	0	0	0	0	-0.009	0	-0.018	0	0.014
7	0	0	0	0	-0.002	-0.018	-0.017	0.010	0.010
	-3	-2	-1	0	1	2	3	4	5

Figure A.28: Direction 2: Synthesis filter for phase one.

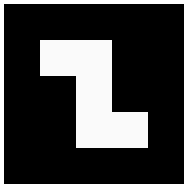
-6	0.010	0	-0.018	0	0
-5	0.016	0	-0.030	-0.016	0
-4	0.010	0	-0.032	0	0
-3	0.016	0	-0.050	-0.031	0
-2	0.010	0	-0.044	-0.062	0
-1	0.016	0	-0.061	-0.047	0
0	0.010	0	-0.052	-0.062	-0.021
1	0.016	0	-0.062	0.938	-0.016
2	0	0	-0.034	-0.062	-0.021
3	0	0	-0.033	-0.047	-0.016
4	0	0	-0.020	-0.062	-0.021
5	0	0	-0.012	-0.031	-0.016
6	0	0	-0.008	0	-0.021
7	0	0	-0.002	-0.016	-0.016
	-3	-2	-1	0	1

Figure A.29: Direction 2: Synthesis filter for phase two.



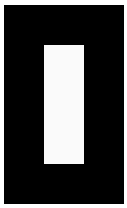
-6	0.010	0	-0.017	0	-0.002	0	0	0	0
-5	0.016	0	-0.028	-0.016	-0.003	-0.002	0	0	0
-4	0	0	-0.007	0	-0.014	0	0	0	0
-3	0	0	-0.009	-0.016	-0.023	-0.012	0	0	0
-2	0	0	0.002	-0.062	-0.034	-0.006	0	0	0
-1	0	0	0.009	-0.016	-0.054	-0.033	0	0	0
0	0	0	0.011	0	-0.093	-0.044	0.016	0	0
1	0	0	0.028	-0.016	0.878	-0.062	0.028	0.016	0
2	0	0	0	0	-0.048	-0.081	0	0	0
3	0	0	0	0	-0.054	-0.061	0.009	0.016	0
4	0	0	0	0	-0.033	-0.119	-0.016	0.063	0
5	0	0	0	0	-0.023	-0.050	-0.009	0.016	0
6	0	0	0	0	-0.016	0	-0.031	0	0.021
7	0	0	0	0	-0.003	-0.030	-0.028	0.016	0.016
	-3	-2	-1	0	1	2	3	4	5

Figure A.30: Direction 2: Synthesis filter for phase three.



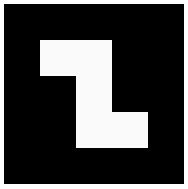
-1	-0.250	0.583	0
0	0	0.333	0
1	0	0.083	0.250
	-2	-1	0

Figure A.31: Direction 3: Prediction filter for phase one.



-2	0.083
-1	0.333
0	0.583
	0

Figure A.32: Direction 3: Prediction filter for phase two.

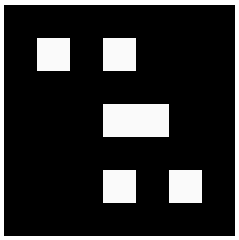


-2	-0.125	0.250	0
-1	0	0.250	0
0	0	0.250	0.375
	-2	-1	0

Figure A.33: Direction 3: Prediction filter for phase three.

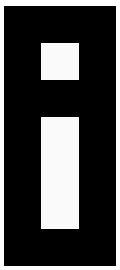
-4	0.055	0	-0.094	0	-0.080	0	0	0	0
-3	0	0	0	0	0	0	0	0	0
-2	0	0	-0.031	0.125	-0.285	0.042	0	0	0
-1	0	0	0	0.188	0.292	0.125	0	0	0
0	0	0	0.031	0	1.307	0.167	0.031	0	0
1	0	0	0	0	0.167	0.125	0	0	0
2	0	0	0	0	-0.285	0.292	-0.031	-0.125	0
3	0	0	0	0	0.042	0.125	0	-0.062	0
4	0	0	0	0	-0.080	0	-0.094	0	0.055
	-4	-3	-2	-1	0	1	2	3	4

Figure A.34: Direction 3: Analysis filter for phase zero.



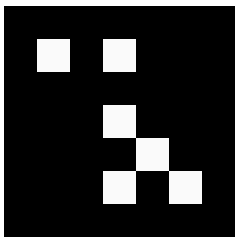
-2	0.250	0	-0.583	0	0
-1	0	0	0	0	0
0	0	0	-0.333	1.000	0
1	0	0	0	0	0
2	0	0	-0.083	0	-0.250
	-4	-3	-2	-1	0

Figure A.35: Direction 3: Analysis filter for phase one.



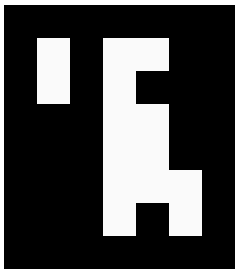
-4	-0.083
-3	0
-2	-0.333
-1	1.000
0	-0.583
	0

Figure A.36: Direction 3: Analysis filter for phase two.



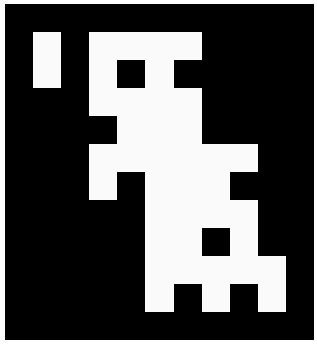
-4	0.125	0	-0.250	0	0
-3	0	0	0	0	0
-2	0	0	-0.250	0	0
-1	0	0	0	1.000	0
0	0	0	-0.250	0	-0.375
	-4	-3	-2	-1	0

Figure A.37: Direction 3: Analysis filter for phase three.



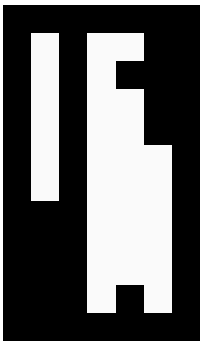
-3	-0.062	0	0.125	0.042	0
-2	-0.125	0	0.292	0	0
-1	0	0	0.125	0.167	0
0	0	0	0.167	0.500	0
1	0	0	0.125	0.292	0.188
2	0	0	0.042	0	0.125
	-3	-2	-1	0	1

Figure A.38: Direction 3: Synthesis filter for phase zero.



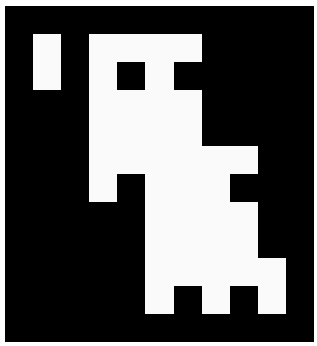
-5	0.008	0	-0.013	-0.005	-0.005	-0.002	0	0	0
-4	0.016	0	-0.031	0	-0.012	0	0	0	0
-3	0	0	-0.005	-0.021	-0.026	-0.014	0	0	0
-2	0	0	0	-0.063	-0.056	-0.021	0	0	0
-1	0	0	0.003	-0.036	-0.094	-0.052	0.008	0.005	0
0	0	0	0.031	0	0.854	-0.083	0.031	0	0
1	0	0	0	0	-0.057	-0.097	-0.016	0.021	0
2	0	0	0	0	-0.056	-0.146	0	0.062	0
3	0	0	0	0	-0.036	-0.085	-0.039	0.036	0.023
4	0	0	0	0	-0.012	0	-0.031	0	0.016
	-3	-2	-1	0	1	2	3	4	5

Figure A.39: Direction 3: Synthesis filter for phase one.



-3	0.018	0	-0.036	-0.012	0
-2	0.036	0	-0.085	0	0
-1	0.010	0	-0.057	-0.056	0
0	0.021	0	-0.097	-0.146	0
1	0.003	0	-0.063	0.885	-0.055
2	0.005	0	-0.052	-0.083	-0.036
3	0	0	-0.026	-0.056	-0.031
4	0	0	-0.014	-0.021	-0.021
5	0	0	-0.005	-0.012	-0.008
6	0	0	-0.002	0	-0.005
	-3	-2	-1	0	1

Figure A.40: Direction 3: Synthesis filter for phase two.

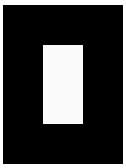


-3	0.012	0	-0.016	-0.008	-0.016	-0.005	0	0	0
-2	0.023	0	-0.039	0	-0.036	0	0	0	0
-1	0	0	-0.016	-0.031	-0.031	-0.026	0	0	0
0	0	0	-0.016	-0.094	-0.057	-0.062	0	0	0
1	0	0	-0.016	-0.055	0.914	-0.063	-0.016	0.003	0
2	0	0	0.008	0	-0.094	-0.063	0.003	0	0
3	0	0	0	0	-0.031	-0.057	-0.016	0.010	0
4	0	0	0	0	-0.026	-0.063	-0.005	0.031	0
5	0	0	0	0	-0.016	-0.036	-0.016	0.018	0.012
6	0	0	0	0	-0.005	0	-0.013	0	0.008
	-3	-2	-1	0	1	2	3	4	5

Figure A.41: Direction 3: Synthesis filter for phase three.

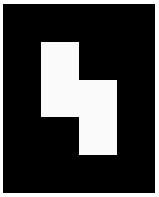
0	0.500	0.500
	-1	0

Figure A.42: Direction 4: Prediction filter for phase one.



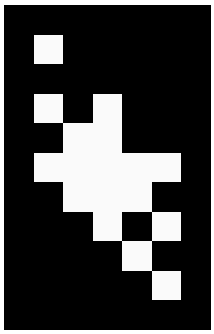
-1	0.500
0	0.500
	0

Figure A.43: Direction 4: Prediction filter for phase two.



-2	-0.250	0
-1	0.750	0.250
0	0	0.250
	-1	0

Figure A.44: Direction 4: Prediction filter for phase three.

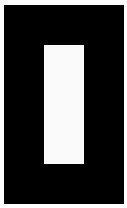


-4	0.031	0	0	0	0
-3	0	0	0	0	0
-2	-0.062	0	-0.062	0	0
-1	0	0.125	0.250	0	0
0	-0.219	0.250	1.125	0.250	-0.219
1	0	0.125	0.250	0.375	0
2	0	0	-0.062	0	-0.062
3	0	0	0	-0.125	0
4	0	0	0	0	0.031
	-2	-1	0	1	2

Figure A.45: Direction 4: Analysis filter for phase zero.

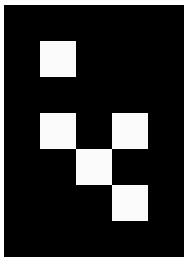
0	-0.500	1.000	-0.500
	-2	-1	0

Figure A.46: Direction 4: Analysis filter for phase one.



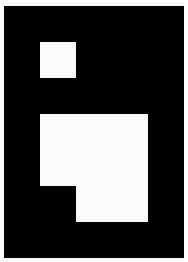
-2	-0.500
-1	1.000
0	-0.500
	0

Figure A.47: Direction 4: Analysis filter for phase two.



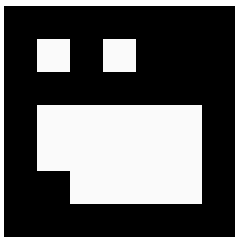
-4	0.250	0	0
-3	0	0	0
-2	-0.750	0	-0.250
-1	0	1.000	0
0	0	0	-0.250
	-2	-1	0

Figure A.48: Direction 4: Analysis filter for phase three.



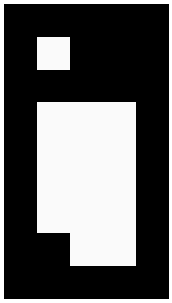
-3	-0.125	0	0
-2	0	0	0
-1	0.375	0.250	0.125
0	0.250	0.500	0.250
1	0	0.250	0.125
	-1	0	1

Figure A.49: Direction 4: Synthesis filter for phase zero.



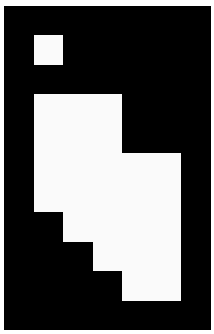
-3	0.031	0	0.031	0	0
-2	0	0	0	0	0
-1	-0.094	-0.063	-0.125	-0.063	-0.031
0	-0.063	-0.125	0.875	-0.125	-0.063
1	0	-0.063	-0.031	-0.063	-0.031
	-1	0	1	2	3

Figure A.50: Direction 4: Synthesis filter for phase one.



-3	0.031	0	0
-2	0	0	0
-1	-0.062	-0.063	-0.031
0	-0.063	-0.125	-0.063
1	-0.094	0.875	-0.063
2	-0.063	-0.125	-0.063
3	0	-0.063	-0.031
	-1	0	1

Figure A.51: Direction 4: Synthesis filter for phase two.

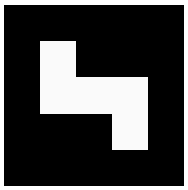


-3	0.016	0	0	0	0
-2	0	0	0	0	0
-1	-0.031	-0.031	0.031	0	0
0	-0.031	-0.063	-0.031	0	0
1	-0.047	-0.063	0.812	-0.094	-0.047
2	-0.031	-0.063	-0.125	-0.188	-0.094
3	0	-0.031	0.031	-0.062	-0.031
4	0	0	0.031	0.063	0.031
5	0	0	0	0.031	0.016
	-1	0	1	2	3

Figure A.52: Direction 4: Synthesis filter for phase three.

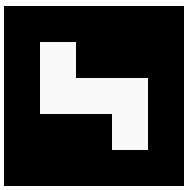
0	0.083	0.333	0.583
	-2	-1	0

Figure A.53: Direction 5: Prediction filter for phase one.



-2	-0.250	0	0
-1	0.583	0.333	0.083
0	0	0	0.250
	-1	0	1

Figure A.54: Direction 5: Prediction filter for phase two.



-2	-0.125	0	0
-1	0.250	0.250	0.250
0	0	0	0.375
	-2	-1	0

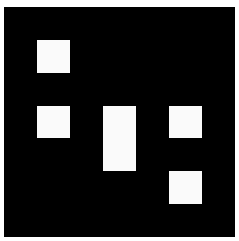
Figure A.55: Direction 5: Prediction filter for phase three.

-4	0.055	0	0	0	0	0	0	0	0
-3	0	0	0	0	0	0	0	0	0
-2	-0.094	0	-0.031	0	0.031	0	0	0	0
-1	0	0	0.125	0.188	0	0	0	0	0
0	-0.080	0	-0.285	0.292	1.307	0.167	-0.285	0.042	-0.080
1	0	0	0.042	0.125	0.167	0.125	0.292	0.125	0
2	0	0	0	0	0.031	0	-0.031	0	-0.094
3	0	0	0	0	0	0	-0.125	-0.063	0
4	0	0	0	0	0	0	0	0	0.055
	-4	-3	-2	-1	0	1	2	3	4

Figure A.56: Direction 5: Analysis filter for phase zero.

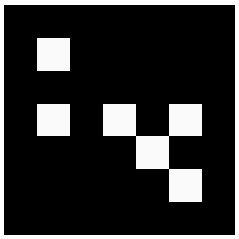
0	-0.083	0	-0.333	1.000	-0.583
	-4	-3	-2	-1	0

Figure A.57: Direction 5: Analysis filter for phase one.



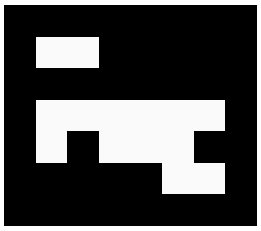
-4	0.250	0	0	0	0
-3	0	0	0	0	0
-2	-0.583	0	-0.333	0	-0.083
-1	0	0	1.000	0	0
0	0	0	0	0	-0.250
	-2	-1	0	1	2

Figure A.58: Direction 5: Analysis filter for phase two.



-4	0.125	0	0	0	0
-3	0	0	0	0	0
-2	-0.250	0	-0.250	0	-0.250
-1	0	0	0	1.000	0
0	0	0	0	0	-0.375
	-4	-3	-2	-1	0

Figure A.59: Direction 5: Analysis filter for phase three.



-3	-0.063	-0.125	0	0	0	0
-2	0	0	0	0	0	0
-1	0.125	0.292	0.125	0.167	0.125	0.042
0	0.042	0	0.167	0.500	0.292	0
1	0	0	0	0	0.188	0.125
	-3	-2	-1	0	1	2

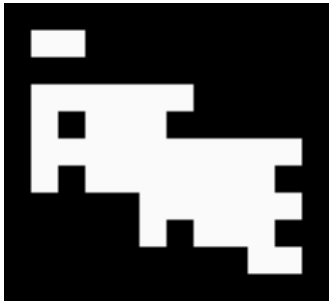
Figure A.60: Direction 5: Synthesis filter for phase zero.

-3	0.018	0.036	0.010	0.021	0.003	0.005	0	0	0	0
-2	0	0	0	0	0	0	0	0	0	0
-1	-0.036	-0.085	-0.057	-0.097	-0.062	-0.052	-0.026	-0.014	-0.005	-0.002
0	-0.012	0	-0.056	-0.146	0.885	-0.083	-0.056	-0.021	-0.012	0
1	0	0	0	0	-0.055	-0.036	-0.031	-0.021	-0.008	-0.005
	-3	-2	-1	0	1	2	3	4	5	6

Figure A.61: Direction 5: Synthesis filter for phase one.

-3	0.008	0.016	0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0	0	0
-1	-0.013	-0.031	-0.005	0	0.003	0.031	0	0	0	0
0	-0.005	0	-0.021	-0.063	-0.036	0	0	0	0	0
1	-0.005	-0.012	-0.026	-0.056	-0.094	0.854	-0.057	-0.056	-0.036	-0.012
2	-0.002	0	-0.014	-0.021	-0.052	-0.083	-0.097	-0.146	-0.085	0
3	0	0	0	0	0.008	0.031	-0.016	0	-0.039	-0.031
4	0	0	0	0	0.005	0	0.021	0.063	0.036	0
5	0	0	0	0	0	0	0	0	0.023	0.016
	-5	-4	-3	-2	-1	0	1	2	3	4

Figure A.62: Direction 5: Synthesis filter for phase two.

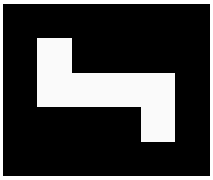


-3	0.012	0.023	0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0	0	0
-1	-0.016	-0.039	-0.016	-0.016	-0.016	0.008	0	0	0	0
0	-0.008	0	-0.031	-0.094	-0.055	0	0	0	0	0
1	-0.016	-0.036	-0.031	-0.057	0.914	-0.094	-0.031	-0.026	-0.016	-0.005
2	-0.005	0	-0.026	-0.062	-0.062	-0.063	-0.057	-0.063	-0.036	0
3	0	0	0	0	-0.016	0.003	-0.016	-0.005	-0.016	-0.013
4	0	0	0	0	0.003	0	0.010	0.031	0.018	0
5	0	0	0	0	0	0	0	0	0.012	0.008
	-3	-2	-1	0	1	2	3	4	5	6

Figure A.63: Direction 5: Synthesis filter for phase three.

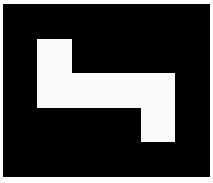
0	0.250	0.250	0.250	0.250
	-2	-1	0	1

Figure A.64: Direction 6: Prediction filter for phase one.



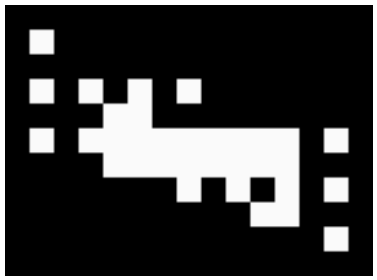
-2	-0.167	0	0	0
-1	0.283	0.233	0.183	0.133
0	0	0	0	0.333
	-2	-1	0	1

Figure A.65: Direction 6: Prediction filter for phase two.



-2	-0.250	0	0	0
-1	0.475	0.325	0.175	0.025
0	0	0	0	0.250
	-2	-1	0	1

Figure A.66: Direction 6: Prediction filter for phase three.

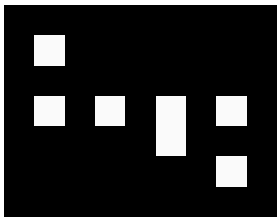


-4	0.059	0	0	0	0	0	0	0	0	0	0	0	0
-3	0	0	0	0	0	0	0	0	0	0	0	0	0
-2	-0.092	0	-0.042	0	0.008	0	0.058	0	0	0	0	0	0
-1	0	0	0	0.125	0.167	0	0	0	0	0	0	0	0
0	-0.056	0	-0.150	0.125	-0.268	0.125	1.469	0.125	-0.268	0.125	-0.150	0	-0.056
1	0	0	0	0.013	0.067	0.088	0.092	0.163	0.117	0.238	0.142	0	0
2	0	0	0	0	0	0	0.058	0	0.008	0	-0.042	0	-0.092
3	0	0	0	0	0	0	0	0	0	-0.125	-0.083	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0.059
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6

Figure A.67: Direction 6: Analysis filter for phase zero.

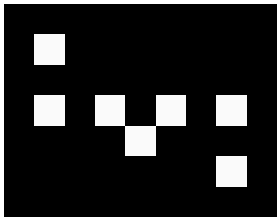
0	-0.250	0	-0.250	1.000	-0.250	0	-0.250
	-4	-3	-2	-1	0	1	2

Figure A.68: Direction 6: Analysis filter for phase one.



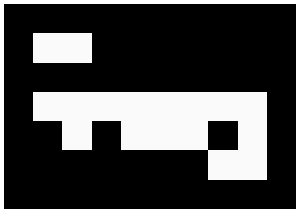
-4	0.167	0	0	0	0	0	0
-3	0	0	0	0	0	0	0
-2	-0.283	0	-0.233	0	-0.183	0	-0.133
-1	0	0	0	0	1.000	0	0
0	0	0	0	0	0	0	-0.333
	-4	-3	-2	-1	0	1	2

Figure A.69: Direction 6: Analysis filter for phase two.



-4	0.250	0	0	0	0	0	0
-3	0	0	0	0	0	0	0
-2	-0.475	0	-0.325	0	-0.175	0	-0.025
-1	0	0	0	1.000	0	0	0
0	0	0	0	0	0	0	-0.250
	-4	-3	-2	-1	0	1	2

Figure A.70: Direction 6: Analysis filter for phase three.



-3	-0.083	-0.125	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0
-1	0.142	0.238	0.117	0.163	0.092	0.088	0.067	0.013
0	0	0.125	0	0.125	0.500	0.125	0	0.125
1	0	0	0	0	0	0	0.167	0.125
	-4	-3	-2	-1	0	1	2	3

Figure A.71: Direction 6: Synthesis filter for phase zero.

-3	0.010	0.016	0.010	0.016	0.010	0.016	0.010	0.016	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	-0.018	-0.030	-0.032	-0.050	-0.044	-0.061	-0.052	-0.063	-0.034	-0.033	-0.020	-0.013	-0.008	-0.002
0	0	-0.016	0	-0.031	-0.063	-0.047	-0.063	0.937	-0.063	-0.047	-0.063	-0.031	0	-0.016
1	0	0	0	0	0	0	-0.021	-0.016	-0.021	-0.016	-0.021	-0.016	-0.021	-0.016
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7

Figure A.72: Direction 6: Synthesis filter for phase one.

-3	0.014	0.021	0	0	0	0	0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	-0.018	-0.031	-0.012	-0.016	-0.006	0	0.001	0.016	0	0	0	0	0	0
0	0	-0.021	0	-0.021	-0.083	-0.021	0	-0.021	0	0	0	0	0	0
1	-0.009	-0.016	-0.021	-0.033	-0.033	-0.048	0.919	-0.093	-0.033	-0.034	-0.021	-0.014	-0.009	-0.002
2	0	-0.008	0	-0.020	-0.033	-0.034	-0.046	-0.052	-0.058	-0.044	-0.071	-0.032	0	-0.018
3	0	0	0	0	0	0	0.001	0.011	-0.006	0.002	-0.012	-0.007	-0.018	-0.017
4	0	0	0	0	0	0	0	0.010	0	0.010	0.042	0.010	0	0.010
5	0	0	0	0	0	0	0	0	0	0	0	0	0.014	0.010
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7

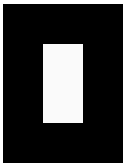
Figure A.73: Direction 6: Synthesis filter for phase two.

-3	0.010	0.016	0	0	0	0	0	0	0	0	0	0	0	0
-2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-1	-0.017	-0.028	-0.007	-0.009	0.002	0.009	0.011	0.028	0	0	0	0	0	0
0	0	-0.016	0	-0.016	-0.062	-0.016	0	-0.016	0	0	0	0	0	0
1	-0.002	-0.003	-0.014	-0.023	-0.034	-0.054	-0.093	0.878	-0.048	-0.054	-0.033	-0.023	-0.016	-0.003
2	0	-0.002	0	-0.013	-0.006	-0.033	-0.044	-0.063	-0.081	-0.061	-0.119	-0.050	0	-0.030
3	0	0	0	0	0	0	0.016	0.028	0	0.009	-0.016	-0.009	-0.031	-0.028
4	0	0	0	0	0	0	0	0.016	0	0.016	0.062	0.016	0	0.016
5	0	0	0	0	0	0	0	0	0	0	0	0	0.021	0.016
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7

Figure A.74: Direction 6: Synthesis filter for phase three.

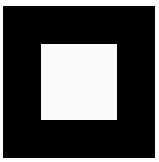
0	0.500	0.500
	-1	0

Figure A.75: Direction 7: Prediction filter for phase one.



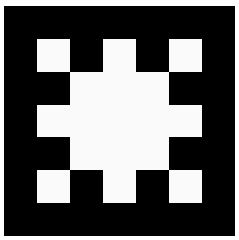
-1	0.500
0	0.500
	0

Figure A.76: Direction 7: Prediction filter for phase two.



-1	0.250	0.250
0	0.250	0.250
	-1	0

Figure A.77: Direction 7: Prediction filter for phase three.

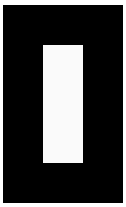


-2	-0.031	0	-0.188	0	-0.031
-1	0	0.125	0.250	0.125	0
0	-0.188	0.250	1.375	0.250	-0.188
1	0	0.125	0.250	0.125	0
2	-0.031	0	-0.188	0	-0.031
	-2	-1	0	1	2

Figure A.78: Direction 7: Analysis filter for phase zero.

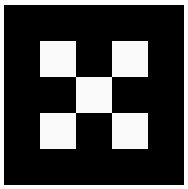
0	-0.500	1.000	-0.500
	-2	-1	0

Figure A.79: Direction 7: Analysis filter for phase one.



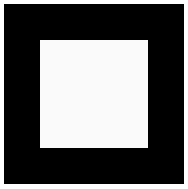
-2	-0.500
-1	1.000
0	-0.500
	0

Figure A.80: Direction 7: Analysis filter for phase two.



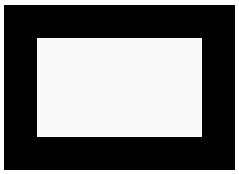
-2	-0.250	0	-0.250
-1	0	1.000	0
0	-0.250	0	-0.250
	-2	-1	0

Figure A.81: Direction 7: Analysis filter for phase three.



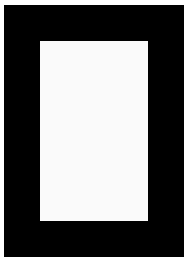
-1	0.125	0.250	0.125
0	0.250	0.500	0.250
1	0.125	0.250	0.125
	-1	0	1

Figure A.82: Direction 7: Synthesis filter for phase zero.



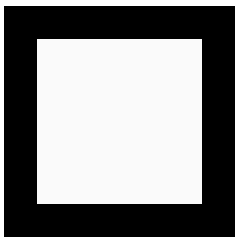
-1	-0.031	-0.063	-0.063	-0.063	-0.031
0	-0.063	-0.125	0.875	-0.125	-0.063
1	-0.031	-0.063	-0.063	-0.063	-0.031
	-1	0	1	2	3

Figure A.83: Direction 7: Synthesis filter for phase one.



-1	-0.031	-0.063	-0.031
0	-0.063	-0.125	-0.063
1	-0.063	0.875	-0.063
2	-0.063	-0.125	-0.063
3	-0.031	-0.063	-0.031
	-1	0	1

Figure A.84: Direction 7: Synthesis filter for phase two.

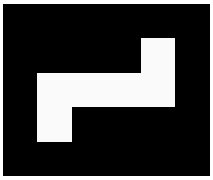


-1	-0.016	-0.031	-0.031	-0.031	-0.016
0	-0.031	-0.063	-0.063	-0.063	-0.031
1	-0.031	-0.063	0.938	-0.063	-0.031
2	-0.031	-0.063	-0.063	-0.063	-0.031
3	-0.016	-0.031	-0.031	-0.031	-0.016
	-1	0	1	2	3

Figure A.85: Direction 7: Synthesis filter for phase three.

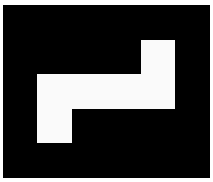
0	0.250	0.250	0.250	0.250
	-2	-1	0	1

Figure A.86: Direction 8: Prediction filter for phase one.



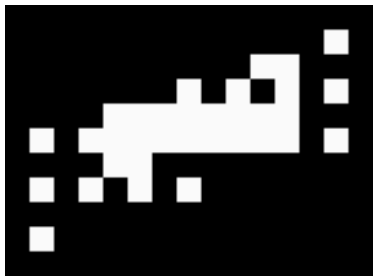
-1	0	0	0	0.333
0	0.283	0.233	0.183	0.133
1	-0.167	0	0	0
	-2	-1	0	1

Figure A.87: Direction 8: Prediction filter for phase two.



-1	0	0	0	0.250
0	0.475	0.325	0.175	0.025
1	-0.250	0	0	0
	-2	-1	0	1

Figure A.88: Direction 8: Prediction filter for phase three.

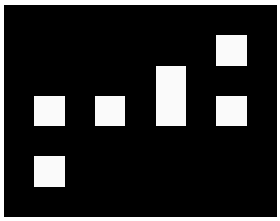


-4	0	0	0	0	0	0	0	0	0	0	0	0	0.059
-3	0	0	0	0	0	0	0	0	0	-0.125	-0.083	0	0
-2	0	0	0	0	0	0	0.058	0	0.008	0	-0.042	0	-0.092
-1	0	0	0	0.013	0.067	0.088	0.092	0.163	0.117	0.238	0.142	0	0
0	-0.056	0	-0.150	0.125	-0.268	0.125	1.469	0.125	-0.268	0.125	-0.150	0	-0.056
1	0	0	0	0.125	0.167	0	0	0	0	0	0	0	0
2	-0.092	0	-0.042	0	0.008	0	0.058	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0.059	0	0	0	0	0	0	0	0	0	0	0	0
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6

Figure A.89: Direction 8: Analysis filter for phase zero.

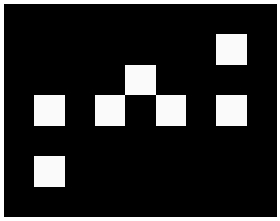
0	-0.250	0	-0.250	1.000	-0.250	0	-0.250
	-4	-3	-2	-1	0	1	2

Figure A.90: Direction 8: Analysis filter for phase one.



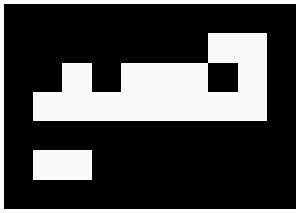
-2	0	0	0	0	0	0	-0.333
-1	0	0	0	0	1.000	0	0
0	-0.283	0	-0.233	0	-0.183	0	-0.133
1	0	0	0	0	0	0	0
2	0.167	0	0	0	0	0	0
	-4	-3	-2	-1	0	1	2

Figure A.91: Direction 8: Analysis filter for phase two.



-2	0	0	0	0	0	0	-0.250
-1	0	0	0	1.000	0	0	0
0	-0.475	0	-0.325	0	-0.175	0	-0.025
1	0	0	0	0	0	0	0
2	0.250	0	0	0	0	0	0
	-4	-3	-2	-1	0	1	2

Figure A.92: Direction 8: Analysis filter for phase three.

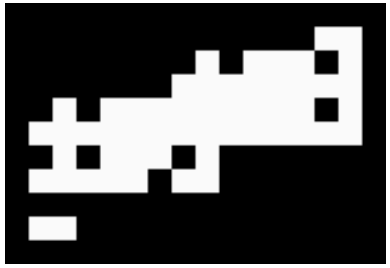


-1	0	0	0	0	0	0	0.167	0.125
0	0	0.125	0	0.125	0.500	0.125	0	0.125
1	0.142	0.238	0.117	0.163	0.092	0.088	0.067	0.013
2	0	0	0	0	0	0	0	0
3	-0.083	-0.125	0	0	0	0	0	0
	-4	-3	-2	-1	0	1	2	3

Figure A.93: Direction 8: Synthesis filter for phase zero.

-1	0	0	0	0	0	0	-0.021	-0.016	-0.021	-0.016	-0.021	-0.016	-0.021	-0.016
0	0	-0.016	0	-0.031	-0.063	-0.047	-0.063	0.937	-0.063	-0.047	-0.063	-0.031	0	-0.016
1	-0.018	-0.030	-0.032	-0.050	-0.044	-0.061	-0.052	-0.063	-0.034	-0.033	-0.020	-0.013	-0.008	-0.002
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0.010	0.016	0.010	0.016	0.010	0.016	0.010	0.016	0	0	0	0	0	0
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7

Figure A.94: Direction 8: Synthesis filter for phase one.



-3	0	0	0	0	0	0	0	0	0	0	0	0	0.014	0.010
-2	0	0	0	0	0	0	0	0.010	0	0.010	0.042	0.010	0	0.010
-1	0	0	0	0	0	0	0.001	0.011	-0.006	0.002	-0.012	-0.007	-0.018	-0.017
0	0	-0.008	0	-0.020	-0.033	-0.034	-0.046	-0.052	-0.058	-0.044	-0.071	-0.032	0	-0.018
1	-0.009	-0.016	-0.021	-0.033	-0.033	-0.048	0.919	-0.093	-0.033	-0.034	-0.021	-0.014	-0.009	-0.002
2	0	-0.021	0	-0.021	-0.083	-0.021	0	-0.021	0	0	0	0	0	0
3	-0.018	-0.031	-0.012	-0.016	-0.006	0	0.001	0.016	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0.014	0.021	0	0	0	0	0	0	0	0	0	0	0	0
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7

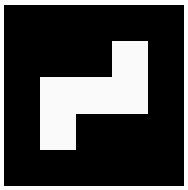
Figure A.95: Direction 8: Synthesis filter for phase two.

-3	0	0	0	0	0	0	0	0	0	0	0	0	0.021	0.016
-2	0	0	0	0	0	0	0	0.016	0	0.016	0.062	0.016	0	0.016
-1	0	0	0	0	0	0	0.016	0.028	0	0.009	-0.016	-0.009	-0.031	-0.028
0	0	-0.002	0	-0.013	-0.006	-0.033	-0.044	-0.063	-0.081	-0.061	-0.119	-0.050	0	-0.030
1	-0.002	-0.003	-0.014	-0.023	-0.034	-0.054	-0.093	0.878	-0.048	-0.054	-0.033	-0.023	-0.016	-0.003
2	0	-0.016	0	-0.016	-0.062	-0.016	0	-0.016	0	0	0	0	0	0
3	-0.017	-0.028	-0.007	-0.009	0.002	0.009	0.011	0.028	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0.010	0.016	0	0	0	0	0	0	0	0	0	0	0	0
	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7

Figure A.96: Direction 8: Synthesis filter for phase three.

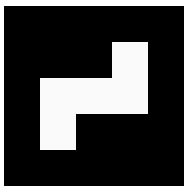
0	0.083	0.333	0.583
	-2	-1	0

Figure A.97: Direction 9: Prediction filter for phase one.



-1	0	0	0.250
0	0.583	0.333	0.083
1	-0.250	0	0
	-1	0	1

Figure A.98: Direction 9: Prediction filter for phase two.



-1	0	0	0.375
0	0.250	0.250	0.250
1	-0.125	0	0
	-2	-1	0

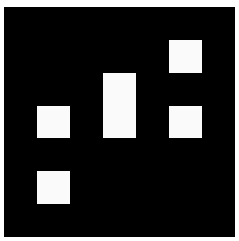
Figure A.99: Direction 9: Prediction filter for phase three.

-4	0	0	0	0	0	0	0	0	0.055
-3	0	0	0	0	0	0	-0.125	-0.063	0
-2	0	0	0	0	0.031	0	-0.031	0	-0.094
-1	0	0	0.042	0.125	0.167	0.125	0.292	0.125	0
0	-0.080	0	-0.285	0.292	1.307	0.167	-0.285	0.042	-0.080
1	0	0	0.125	0.188	0	0	0	0	0
2	-0.094	0	-0.031	0	0.031	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0.055	0	0	0	0	0	0	0	0
	-4	-3	-2	-1	0	1	2	3	4

Figure A.100: Direction 9: Analysis filter for phase zero.

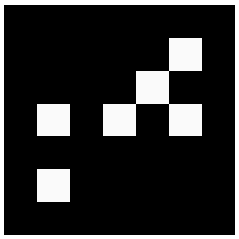
0	-0.083	0	-0.333	1.000	-0.583
	-4	-3	-2	-1	0

Figure A.101: Direction 9: Analysis filter for phase one.



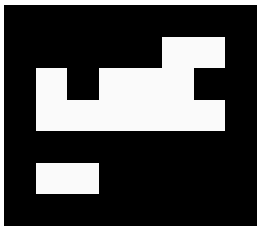
-2	0	0	0	0	-0.250
-1	0	0	1.000	0	0
0	-0.583	0	-0.333	0	-0.083
1	0	0	0	0	0
2	0.250	0	0	0	0
	-2	-1	0	1	2

Figure A.102: Direction 9: Analysis filter for phase two.



-2	0	0	0	0	-0.375
-1	0	0	0	1.000	0
0	-0.250	0	-0.250	0	-0.250
1	0	0	0	0	0
2	0.125	0	0	0	0
	-4	-3	-2	-1	0

Figure A.103: Direction 9: Analysis filter for phase three.



-1	0	0	0	0	0.188	0.125
0	0.042	0	0.167	0.500	0.292	0
1	0.125	0.292	0.125	0.167	0.125	0.042
2	0	0	0	0	0	0
3	-0.063	-0.125	0	0	0	0
	-3	-2	-1	0	1	2

Figure A.104: Direction 9: Synthesis filter for phase zero.

-1	0	0	0	0	-0.055	-0.036	-0.031	-0.021	-0.008	-0.005
0	-0.012	0	-0.056	-0.146	0.885	-0.083	-0.056	-0.021	-0.012	0
1	-0.036	-0.085	-0.057	-0.097	-0.062	-0.052	-0.026	-0.014	-0.005	-0.002
2	0	0	0	0	0	0	0	0	0	0
3	0.018	0.036	0.010	0.021	0.003	0.005	0	0	0	0
	-3	-2	-1	0	1	2	3	4	5	6

Figure A.105: Direction 9: Synthesis filter for phase one.

-3	0	0	0	0	0	0	0	0	0.023	0.016
-2	0	0	0	0	0.005	0	0.021	0.063	0.036	0
-1	0	0	0	0	0.008	0.031	-0.016	0	-0.039	-0.031
0	-0.002	0	-0.014	-0.021	-0.052	-0.083	-0.097	-0.146	-0.085	0
1	-0.005	-0.012	-0.026	-0.056	-0.094	0.854	-0.057	-0.056	-0.036	-0.012
2	-0.005	0	-0.021	-0.063	-0.036	0	0	0	0	0
3	-0.013	-0.031	-0.005	0	0.003	0.031	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0.008	0.016	0	0	0	0	0	0	0	0
	-5	-4	-3	-2	-1	0	1	2	3	4

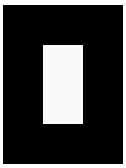
Figure A.106: Direction 9: Synthesis filter for phase two.

-3	0	0	0	0	0	0	0	0	0.012	0.008
-2	0	0	0	0	0.003	0	0.010	0.031	0.018	0
-1	0	0	0	0	-0.016	0.003	-0.016	-0.005	-0.016	-0.013
0	-0.005	0	-0.026	-0.062	-0.062	-0.063	-0.057	-0.063	-0.036	0
1	-0.016	-0.036	-0.031	-0.057	0.914	-0.094	-0.031	-0.026	-0.016	-0.005
2	-0.008	0	-0.031	-0.094	-0.055	0	0	0	0	0
3	-0.016	-0.039	-0.016	-0.016	-0.016	0.008	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0.012	0.023	0	0	0	0	0	0	0	0
	-3	-2	-1	0	1	2	3	4	5	6

Figure A.107: Direction 9: Synthesis filter for phase three.

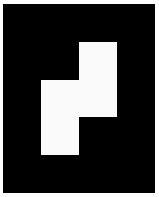
0	0.500	0.500
	-1	0

Figure A.108: Direction 10: Prediction filter for phase one.



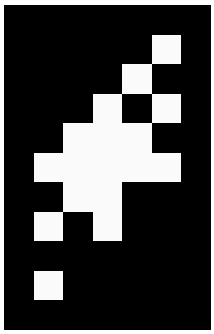
-1	0.500
0	0.500
	0

Figure A.109: Direction 10: Prediction filter for phase two.



-1	0	0.250
0	0.750	0.250
1	-0.250	0
	-1	0

Figure A.110: Direction 10: Prediction filter for phase three.

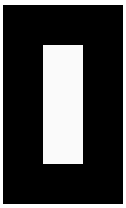


-4	0	0	0	0	0.031
-3	0	0	0	-0.125	0
-2	0	0	-0.062	0	-0.062
-1	0	0.125	0.250	0.375	0
0	-0.219	0.250	1.125	0.250	-0.219
1	0	0.125	0.250	0	0
2	-0.062	0	-0.062	0	0
3	0	0	0	0	0
4	0.031	0	0	0	0
	-2	-1	0	1	2

Figure A.111: Direction 10: Analysis filter for phase zero.

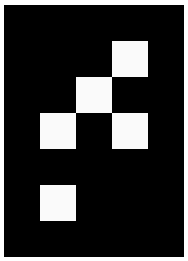
0	-0.500	1.000	-0.500
	-2	-1	0

Figure A.112: Direction 10: Analysis filter for phase one.



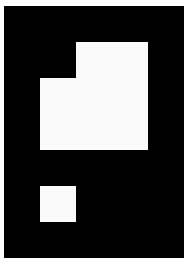
-2	-0.500
-1	1.000
0	-0.500
	0

Figure A.113: Direction 10: Analysis filter for phase two.



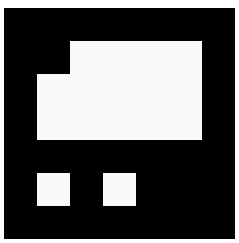
-2	0	0	-0.250
-1	0	1.000	0
0	-0.750	0	-0.250
1	0	0	0
2	0.250	0	0
	-2	-1	0

Figure A.114: Direction 10: Analysis filter for phase three.



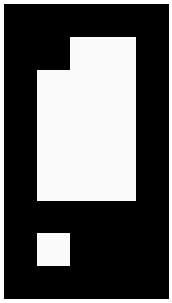
-1	0	0.250	0.125
0	0.250	0.500	0.250
1	0.375	0.250	0.125
2	0	0	0
3	-0.125	0	0
	-1	0	1

Figure A.115: Direction 10: Synthesis filter for phase zero.



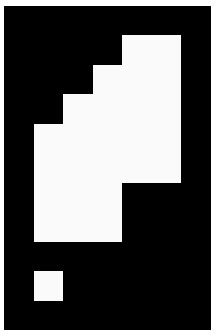
-1	0	-0.063	-0.031	-0.063	-0.031
0	-0.063	-0.125	0.875	-0.125	-0.063
1	-0.094	-0.063	-0.125	-0.063	-0.031
2	0	0	0	0	0
3	0.031	0	0.031	0	0
	-1	0	1	2	3

Figure A.116: Direction 10: Synthesis filter for phase one.



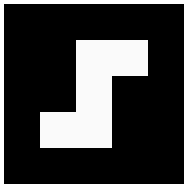
-1	0	-0.063	-0.031
0	-0.063	-0.125	-0.063
1	-0.094	0.875	-0.063
2	-0.063	-0.125	-0.063
3	-0.062	-0.063	-0.031
4	0	0	0
5	0.031	0	0
	-1	0	1

Figure A.117: Direction 10: Synthesis filter for phase two.



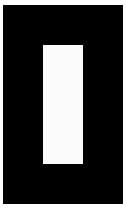
-3	0	0	0	0.031	0.016
-2	0	0	0.031	0.063	0.031
-1	0	-0.031	0.031	-0.062	-0.031
0	-0.031	-0.063	-0.125	-0.188	-0.094
1	-0.047	-0.063	0.812	-0.094	-0.047
2	-0.031	-0.063	-0.031	0	0
3	-0.031	-0.031	0.031	0	0
4	0	0	0	0	0
5	0.016	0	0	0	0
	-1	0	1	2	3

Figure A.118: Direction 10: Synthesis filter for phase three.



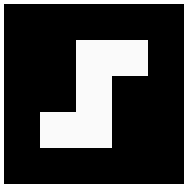
-1	0	0.083	0.250
0	0	0.333	0
1	-0.250	0.583	0
	-2	-1	0

Figure A.119: Direction 11: Prediction filter for phase one.



-1	0.583
0	0.333
1	0.083
	0

Figure A.120: Direction 11: Prediction filter for phase two.

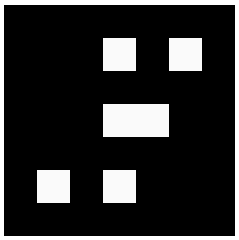


-1	0	0.250	0.375
0	0	0.250	0
1	-0.125	0.250	0
	-2	-1	0

Figure A.121: Direction 11: Prediction filter for phase three.

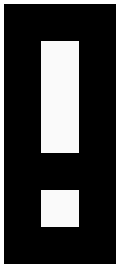
-4	0	0	0	0	-0.080	0	-0.094	0	0.055
-3	0	0	0	0	0.042	0.125	0	-0.062	0
-2	0	0	0	0	-0.285	0.292	-0.031	-0.125	0
-1	0	0	0	0	0.167	0.125	0	0	0
0	0	0	0.031	0	1.307	0.167	0.031	0	0
1	0	0	0	0.188	0.292	0.125	0	0	0
2	0	0	-0.031	0.125	-0.285	0.042	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0.055	0	-0.094	0	-0.080	0	0	0	0
	-4	-3	-2	-1	0	1	2	3	4

Figure A.122: Direction 11: Analysis filter for phase zero.



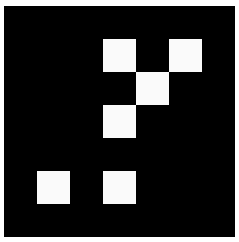
-2	0	0	-0.083	0	-0.250
-1	0	0	0	0	0
0	0	0	-0.333	1.000	0
1	0	0	0	0	0
2	0.250	0	-0.583	0	0
	-4	-3	-2	-1	0

Figure A.123: Direction 11: Analysis filter for phase one.



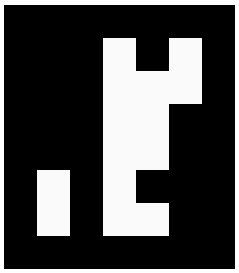
-2	-0.583
-1	1.000
0	-0.333
1	0
2	-0.083
	0

Figure A.124: Direction 11: Analysis filter for phase two.



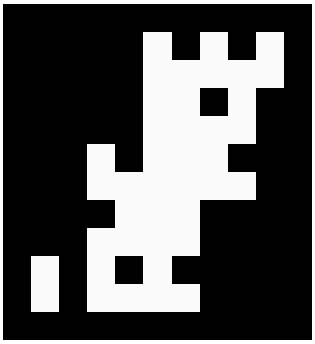
-2	0	0	-0.250	0	-0.375
-1	0	0	0	1.000	0
0	0	0	-0.250	0	0
1	0	0	0	0	0
2	0.125	0	-0.250	0	0
	-4	-3	-2	-1	0

Figure A.125: Direction 11: Analysis filter for phase three.



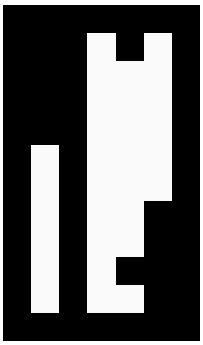
-2	0	0	0.042	0	0.125
-1	0	0	0.125	0.292	0.188
0	0	0	0.167	0.500	0
1	0	0	0.125	0.167	0
2	-0.125	0	0.292	0	0
3	-0.062	0	0.125	0.042	0
	-3	-2	-1	0	1

Figure A.126: Direction 11: Synthesis filter for phase zero.



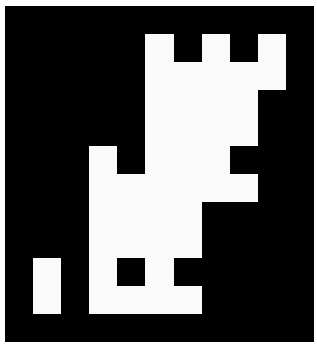
-4	0	0	0	0	-0.012	0	-0.031	0	0.016
-3	0	0	0	0	-0.036	-0.085	-0.039	0.036	0.023
-2	0	0	0	0	-0.056	-0.146	0	0.062	0
-1	0	0	0	0	-0.057	-0.097	-0.016	0.021	0
0	0	0	0.031	0	0.854	-0.083	0.031	0	0
1	0	0	0.003	-0.036	-0.094	-0.052	0.008	0.005	0
2	0	0	0	-0.063	-0.056	-0.021	0	0	0
3	0	0	-0.005	-0.021	-0.026	-0.014	0	0	0
4	0.016	0	-0.031	0	-0.012	0	0	0	0
5	0.008	0	-0.013	-0.005	-0.005	-0.002	0	0	0
	-3	-2	-1	0	1	2	3	4	5

Figure A.127: Direction 11: Synthesis filter for phase one.



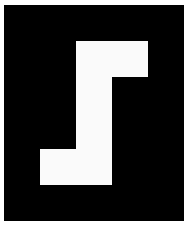
-4	0	0	-0.002	0	-0.005
-3	0	0	-0.005	-0.012	-0.008
-2	0	0	-0.014	-0.021	-0.021
-1	0	0	-0.026	-0.056	-0.031
0	0.005	0	-0.052	-0.083	-0.036
1	0.003	0	-0.063	0.885	-0.055
2	0.021	0	-0.097	-0.146	0
3	0.010	0	-0.057	-0.056	0
4	0.036	0	-0.085	0	0
5	0.018	0	-0.036	-0.012	0
	-3	-2	-1	0	1

Figure A.128: Direction 11: Synthesis filter for phase two.



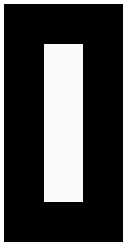
-4	0	0	0	0	-0.005	0	-0.013	0	0.008
-3	0	0	0	0	-0.016	-0.036	-0.016	0.018	0.012
-2	0	0	0	0	-0.026	-0.063	-0.005	0.031	0
-1	0	0	0	0	-0.031	-0.057	-0.016	0.010	0
0	0	0	0.008	0	-0.094	-0.063	0.003	0	0
1	0	0	-0.016	-0.055	0.914	-0.063	-0.016	0.003	0
2	0	0	-0.016	-0.094	-0.057	-0.062	0	0	0
3	0	0	-0.016	-0.031	-0.031	-0.026	0	0	0
4	0.023	0	-0.039	0	-0.036	0	0	0	0
5	0.012	0	-0.016	-0.008	-0.016	-0.005	0	0	0
	-3	-2	-1	0	1	2	3	4	5

Figure A.129: Direction 11: Synthesis filter for phase three.



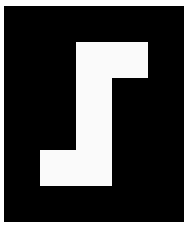
-1	0	0.133	0.333
0	0	0.183	0
1	0	0.233	0
2	-0.167	0.283	0
	-2	-1	0

Figure A.130: Direction 12: Prediction filter for phase one.



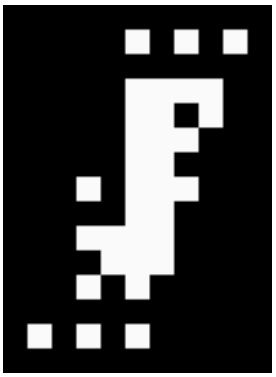
-2	0.250
-1	0.250
0	0.250
1	0.250
	0

Figure A.131: Direction 12: Prediction filter for phase two.



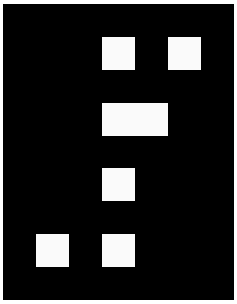
-2	0	0.025	0.250
-1	0	0.175	0
0	0	0.325	0
1	-0.250	0.475	0
	-2	-1	0

Figure A.132: Direction 12: Prediction filter for phase three.



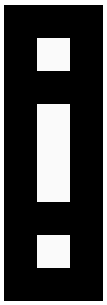
-6	0	0	0	0	-0.056	0	-0.092	0	0.059
-5	0	0	0	0	0	0	0	0	0
-4	0	0	0	0	-0.150	0.142	-0.042	-0.083	0
-3	0	0	0	0	0.125	0.237	0	-0.125	0
-2	0	0	0	0	-0.268	0.117	0.008	0	0
-1	0	0	0	0	0.125	0.162	0	0	0
0	0	0	0.058	0	1.469	0.092	0.058	0	0
1	0	0	0	0	0.125	0.087	0	0	0
2	0	0	0.008	0.167	-0.268	0.067	0	0	0
3	0	0	0	0.125	0.125	0.012	0	0	0
4	0	0	-0.042	0	-0.150	0	0	0	0
5	0	0	0	0	0	0	0	0	0
6	0.059	0	-0.092	0	-0.056	0	0	0	0
	-4	-3	-2	-1	0	1	2	3	4

Figure A.133: Direction 12: Analysis filter for phase zero.



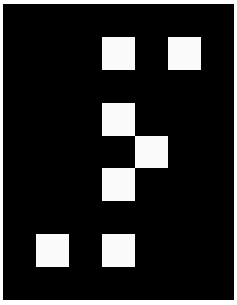
-2	0	0	-0.133	0	-0.333
-1	0	0	0	0	0
0	0	0	-0.183	1.000	0
1	0	0	0	0	0
2	0	0	-0.233	0	0
3	0	0	0	0	0
4	0.167	0	-0.283	0	0
	-4	-3	-2	-1	0

Figure A.134: Direction 12: Analysis filter for phase one.



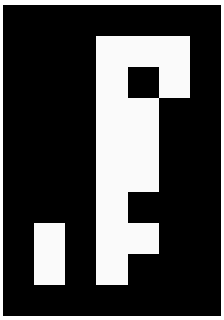
-4	-0.250
-3	0
-2	-0.250
-1	1.000
0	-0.250
1	0
2	-0.250
	0

Figure A.135: Direction 12: Analysis filter for phase two.



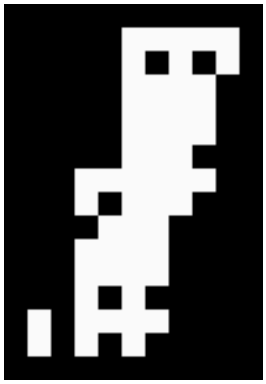
-4	0	0	-0.025	0	-0.250
-3	0	0	0	0	0
-2	0	0	-0.175	0	0
-1	0	0	0	1.000	0
0	0	0	-0.325	0	0
1	0	0	0	0	0
2	0.250	0	-0.475	0	0
	-4	-3	-2	-1	0

Figure A.136: Direction 12: Analysis filter for phase three.



-3	0	0	0.012	0.125	0.125
-2	0	0	0.067	0	0.167
-1	0	0	0.087	0.125	0
0	0	0	0.092	0.500	0
1	0	0	0.162	0.125	0
2	0	0	0.117	0	0
3	-0.125	0	0.237	0.125	0
4	-0.083	0	0.142	0	0
	-3	-2	-1	0	1

Figure A.137: Direction 12: Synthesis filter for phase zero.

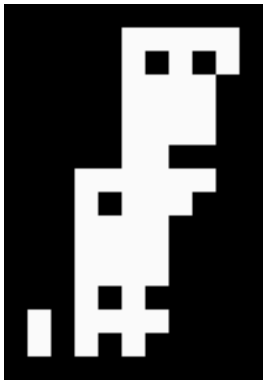


-7	0	0	0	0	-0.002	-0.018	-0.017	0.010	0.010
-6	0	0	0	0	-0.009	0	-0.018	0	0.014
-5	0	0	0	0	-0.014	-0.032	-0.007	0.010	0
-4	0	0	0	0	-0.021	-0.071	-0.012	0.042	0
-3	0	0	0	0	-0.034	-0.044	0.002	0.010	0
-2	0	0	0	0	-0.033	-0.058	-0.006	0	0
-1	0	0	0.016	-0.021	-0.093	-0.052	0.011	0.010	0
0	0	0	0.001	0	0.919	-0.046	0.001	0	0
1	0	0	0	-0.021	-0.048	-0.034	0	0	0
2	0	0	-0.006	-0.083	-0.033	-0.033	0	0	0
3	0	0	-0.016	-0.021	-0.033	-0.020	0	0	0
4	0	0	-0.012	0	-0.021	0	0	0	0
5	0.021	0	-0.031	-0.021	-0.016	-0.008	0	0	0
6	0.014	0	-0.018	0	-0.009	0	0	0	0
	-3	-2	-1	0	1	2	3	4	5

Figure A.138: Direction 12: Synthesis filter for phase one.

-5	0	0	-0.002	-0.016	-0.016
-4	0	0	-0.008	0	-0.021
-3	0	0	-0.012	-0.031	-0.016
-2	0	0	-0.020	-0.062	-0.021
-1	0	0	-0.033	-0.047	-0.016
0	0	0	-0.034	-0.062	-0.021
1	0.016	0	-0.062	0.938	-0.016
2	0.010	0	-0.052	-0.062	-0.021
3	0.016	0	-0.061	-0.047	0
4	0.010	0	-0.044	-0.062	0
5	0.016	0	-0.050	-0.031	0
6	0.010	0	-0.032	0	0
7	0.016	0	-0.030	-0.016	0
8	0.010	0	-0.018	0	0
	-3	-2	-1	0	1

Figure A.139: Direction 12: Synthesis filter for phase two.



-5	0	0	0	0	-0.003	-0.030	-0.028	0.016	0.016
-4	0	0	0	0	-0.016	0	-0.031	0	0.021
-3	0	0	0	0	-0.023	-0.050	-0.009	0.016	0
-2	0	0	0	0	-0.033	-0.119	-0.016	0.063	0
-1	0	0	0	0	-0.054	-0.061	0.009	0.016	0
0	0	0	0	0	-0.048	-0.081	0	0	0
1	0	0	0.028	-0.016	0.878	-0.062	0.028	0.016	0
2	0	0	0.011	0	-0.093	-0.044	0.016	0	0
3	0	0	0.009	-0.016	-0.054	-0.033	0	0	0
4	0	0	0.002	-0.062	-0.034	-0.006	0	0	0
5	0	0	-0.009	-0.016	-0.023	-0.012	0	0	0
6	0	0	-0.007	0	-0.014	0	0	0	0
7	0.016	0	-0.028	-0.016	-0.003	-0.002	0	0	0
8	0.010	0	-0.017	0	-0.002	0	0	0	0
	-3	-2	-1	0	1	2	3	4	5

Figure A.140: Direction 12: Synthesis filter for phase three.

REFERENCES

[1] M. Vetterli, "Wavelets, approximation, and compression," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 59–73, 2001.

[2] S. Mallat, *A Wavelet Tour of Signal Processing*, 2nd ed. San Diego, CA: Academic Press, 1999.

[3] E. J. Candès and D. L. Donoho, "New tight frames of curvelets and optimal representations of objects with piecewise $\mathcal{C}^2$ singularities," *Communications on Pure and Applied Mathematics*, vol. 58, pp. 219–266, 2004.

[4] R. L. De Valois, E. W. Yund, and N. Hepler, "The orientation and direction selectivity of cells in macaque visual cortex," *Vision Research*, vol. 22, no. 5, pp. 531–544, 1982.

[5] V. Velisavljević, B. Beferull-Lozano, M. Vetterli, and P. L. Dragotti, "Directionlets: Anisotropic multi-directional representation with separable filtering," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1916–1933, 2006.

[6] V. Velisavljević, B. Beferull-Lozano, and M. Vetterli, "Space frequency quantization for image compression with directionlets," *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1761–1773, 2007.

[7] E. J. Candès and D. L. Donoho, "Ridgelets: A key to higher dimensional intermittency?" *Philosophical Transactions of the Royal Society A*, vol. 357, no. 1760, pp. 2495–2509, 1999.

[8] E. J. Candès and D. L. Donoho, "Curvelets—a surprisingly effective nonadaptive representation for objects with edges," in *Curves and Surfaces*, C. Rabut, A. Cohen, and L. L. Schumaker, Eds. Nashville, TN: Vanderbilt University Press, 2000, pp. 105–120.

[9] M. N. Do, "Directional multiresolution image representations," Ph.D. dissertation, Swiss Federal Institute of Technology Lausanne, Lausanne, Switzerland, 2001.

[10] K. Guo and D. Labate, "Optimally sparse multidimensional representations using shearlets," *SIAM Journal on Mathematical Analysis*, vol. 39, no. 1, pp. 298–318, 2007.

[11] W.-Q. Lim, "The discrete shearlet transform: A new directional transform and compactly supported shearlet frame," *IEEE Transactions on Image Processing*, vol. 19, no. 5, pp. 1166–1180, 2010.

[12] M. N. Do and M. Vetterli, "The contourlet transform: An efficient directional multiresolution image representation," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2091–2106, 2005.

[13] S. Mallat and G. Peyré, "A review of bandlet methods for geometrical image representation," *Numerical Algorithms*, vol. 44, no. 3, pp. 205–234, 2007.

[14] S. Mallat and G. Peyré, "Orthogonal bandlet bases for geometric images approximation," *Communications on Pure and Applied Mathematics*, vol. 61, no. 9, pp. 1173–1212, 2008.

[15] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," *SIAM Journal on Mathematical Analysis*, vol. 29, no. 2, pp. 511–546, 1998.

[16] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *Journal of Fourier Analysis and Applications*, vol. 4, no. 3, pp. 247–269, 1998.

[17] O. N. Gerek and A. E. Çetin, "A 2-D orientation-adaptive prediction filter in lifting structures for image coding," *IEEE Transactions on Image Processing*, vol. 15, no. 1, pp. 106–111, 2006.

[18] V. Chappelier and C. Guillemot, "Oriented wavelet transform for image compression and denoising," *IEEE Transactions on Image Processing*, vol. 15, no. 10, pp. 2892–2903, 2006.

[19] W. Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive directional lifting-based wavelet transform for image coding," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 416–427, 2007.

[20] Y. Liu and K. N. Ngan, "Weighted adaptive lifting-based wavelet transform for image coding," *IEEE Transactions on Image Processing*, vol. 17, no. 4, pp. 500–511, 2008.

[21] C.-L. Chang and B. Girod, "Direction-adaptive discrete wavelet transform for image compression," *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1289–1302, 2007.

[22] A. Benazza-Benyahia, J.-C. Pesquet, J. Hattay, and H. Masmoudi, "Block-based adaptive vector lifting schemes for multichannel image coding," *EURASIP Journal on Image and Video Processing*, vol. 2007, Article ID 13421, 2007.

[23] G. Quellec, M. Lamard, G. Cazuguel, B. Cochener, and C. Roux, "Adaptive non-separable wavelet transform via lifting and its application to content-based image retrieval," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 25–35, 2010.

[24] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*. Upper Saddle River, NJ: Prentice Hall, 1999.

[25] M. D. Adams, *The JPEG-2000 Still Image Compression Standard*. ISO/IEC JTC 1/SC 29/WG 1 N2412, 2005.

[26] J. Kovačević and W. Sweldens, "Wavelet families of increasing order in arbitrary dimensions," *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 480–496, 2000.

[27] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999.

[28] S. J. Leon, *Linear Algebra with Applications*, 6th ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[29] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, ser. Wadsworth Statistics/Probability Series. Belmont, CA: Wadsworth International Group, 1984.

[30] K. V. Mardia and P. E. Jupp, *Directional Statistics*. Chichester, England: Wiley, 2000.

[31] K. V. Mardia, *Statistics of Directional Data*. London, England: Academic Press, 1972.