Signal Processing for High-Definition Television

by

Peter Monta

B.S., M.S., Carnegie-Mellon University (1985)

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mathematics

at the

Massachusetts Institute of Technology

May 19, 1995

Author .....
Department of Mathematics
May 19, 1995

Certified by ..... May 17, 1995
Jae S. Lim
Professor
Thesis Supervisor

Certified by .....
Gilbert Strang
Professor
Thesis Supervisor

Accepted by .....
David Vogan
Chairman, Departmental Graduate Committee

Signal Processing for High-Definition Television

by

Peter Monta

Submitted to the Department of Mathematics
on May 19, 1995, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Mathematics

Abstract

This thesis presents new design techniques for source coding of video signals. The main application is high-definition television (HDTV), characterized by roughly twice the vertical and horizontal resolution of current systems; aggressive source coding is required to compress the large amount of video information into the relatively narrow channels envisioned for the new service.

Video source coders conventionally use block transforms with small support (typically 8x8 pixels). Such independent blocks result in a simple scheme for switching a predictor from a motion-compensated block to a purely spatial block; this is necessary to prevent the coder from wasting capacity in some situations.

Subband coders of the multiresolution or wavelet type, with their more desirable localization properties, lack of "blocking" artifacts, and better match to motion-compensated prediction errors, complicate this process of switching predictors, since the blocks now overlap. This thesis presents a novel predictive coding scheme in which subband coders can combine the benefits of good representation and flexible adaptive prediction.

Thesis Supervisor: Jae S. Lim
Title: Professor

Thesis Supervisor: Gilbert Strang
Title: Professor

Acknowledgements

I would like to thank Professor Jae Lim for his support and guidance. Working with the Advanced Television and Signal Processing group has been very rewarding, and Professor Lim has expertly directed the group's research in a number of areas.

Thanks to Professor Strang for discussions about wavelets and signal processing, and for graciously agreeing to co-supervise the thesis.

Also thanks to Professor Schreiber for teaching me most of what I know about the human visual system, and for demonstrating that engineering and engineering opinion have an important impact in the larger political and economic arena. Professor Oppenheim showed me how interesting signal processing is through his superb course, for which I think all of his students are grateful.

I've had the good fortune to work among a matrix of very thoughtful people, including John Apostolopoulos, Mike Brandstein, Shiufun Cheung, John Hardwick, David Kuo, Julien Nicolas, Ashok Popat, and Lon Sunshine. One remarkable thing about MIT is that one is never far from an expert on any reasonable subject, so discussions are very valuable, and I've learned much from them.

Cindy LeBlanc and Debra Harring make everyone's lives easier (except perhaps their own), and their dedication to running the group is well known. They have shown me many kindnesses.

As always, my family has been extremely supportive (and patient!). Thanks to Dad and Mom and Anthony, who give me plenty of time to enjoy the real world.

Dedication

To my parents,
Louis and Dorothy Monta,
who were my first teachers

Contents

List of Figures

1

Introduction

Much communication is visual, yet it was only in the early 20th century that it became possible to record natural scenes that evolved in time. These moving images simulate in the viewer, to a good approximation, the process of seeing the original scene.

These early systems were, of course, based on motion picture photography. A physical artifact, namely a reel of transparency film, is required to display the image, and so in the role of a broadcast medium film requires the entire audience to be physically present. Nevertheless, film (in its present high quality, highly refined form) is a very popular medium.

Broadcast television was the development that made practical a real-time audience many orders of magnitude larger than can be seated in a single auditorium. First-generation television systems represent images electronically with waveforms analogous to the brightness variation in a scanned version of a scene. Since signal processing was very expensive in the early days of television, essentially no source coding was used, other than that inherent in the choice of spatial resolution and frame rate (and to some degree in the schemes used to extend the systems to color).

These systems typify the analog design style for communications: identify a convenient physical (often electromagnetic) analog to the object to be communicated, then design the system around this analog by considering how distortions affect fidelity. Each medium requires a separate design, and these designs bear no particular relationship to one another.

It is only very recently that a common currency has become practical for nearly all types of information. In principle, digital techniques may be applied to any analog system by simply discretizing in time or space sufficiently finely, then quantizing sufficiently accurately. Simple schemes, though, result in inefficient systems. Ideally, only essential information about the source should be transmitted; the problem of identifying this information is the *source coding problem* for that source.

This process of homogenizing communications to digital representations may seem a step backward. The aim of transmitting the source with sufficient fidelity is now realized by a complicated source coding device, to give a nonredundant representation of the source, and a complicated digital communication system, to transmit the symbols and recover them reliably from a noisy channel. For many interesting situations, though, this complexity is outweighed by convenience, efficiency, and robustness. Communication systems can be designed with no consideration as to exactly what type of information will pass through them; that is, communication systems can be general. A digital representation can provide higher source coding efficiency

than analog counterparts, simply because more powerful techniques are available. Moreover, digital communication is robust to channel impairments, and tight theoretical bounds on capacity relative to these impairments are available in many situations: implementable codes can approach them closely.

Figure 1.1 sketches the division of labor between source coding and channel coding, together with a schematic channel with linear distortion and noise. The source coder is generally more complex than the source decoder, especially for asymmetric broadcasting applications with many more receivers than transmitters. The opposite is generally true for channel coding: the channel coder implements simple mapping and filtering operations, while the channel decoder may require adaptive equalization and Viterbi decoding.

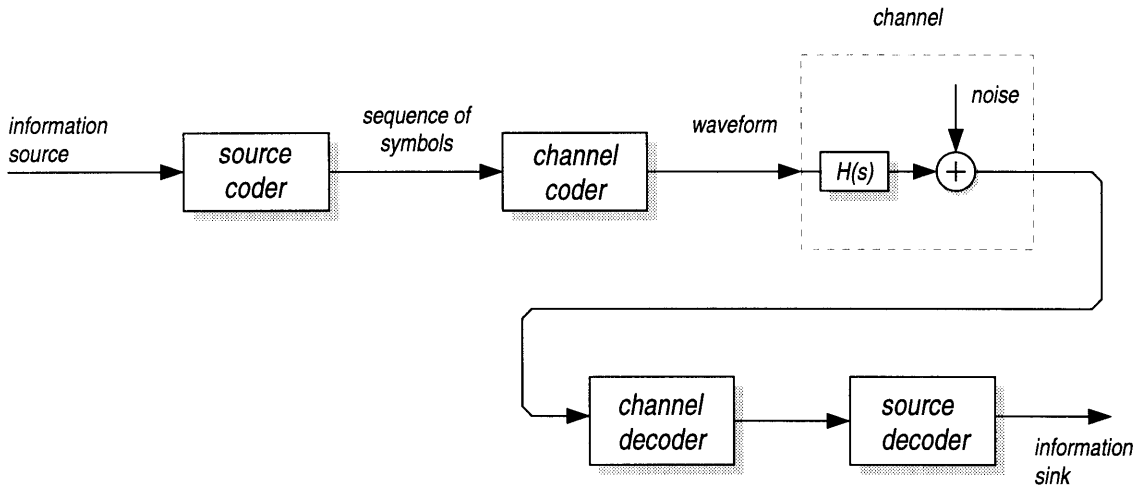


Figure 1.1: A generic communication system.

The technology of integrated semiconductor electronics has been steadily exponentiating since its invention in 1958. This technology is the only practical choice at present for digital signal processing; it is now possible to implement powerful source coders for video at low cost. Consequently, second-generation television systems that use efficient source coders have recently been proposed. Applications range from low-quality interactive television at rates around 100 kb/s, to medium-quality video at around 1.5 Mb/s, to high-definition television at around 20 Mb/s. Still higher rates are required when the video must be subjected to subsequent signal processing unrelated to perceptual "quality", for example in studio postproduction or automatic image analysis.

This thesis is concerned primarily with high-rate, high-quality video source coders. One application, high-definition television, has come to mean video with about one megapixel per frame and with frame rate equal to or higher than conventional TV. Such systems are easy enough to design if there is no rate constraint (easy from a signal processing viewpoint, not from a camera/display/cost one!). The crucial requirement for HDTV in present scenarios is that HDTV be very efficient, usually by requiring that one HDTV signal use no more capacity than one channel of conventional TV (despite containing perhaps six times as many original image samples).

1.1 Architectures for video source coding

A video signal (or "image", "picture", etc.) can be represented (in continuous space and time) as a map

$$f : [0, a] \times [0, b] \times \Lambda \times T \to R$$

where $[0, a] \times [0, b]$ represents space, Λ is wavelength, T is time, and the value of f is the brightness of the image at that point in space, wavelength, and time.

Space is taken to be two-dimensional and bounded, since in practice images are produced from natural three-dimensional scenes by projection onto a bounded surface, such as a piece of film or a CCD. (In practice, of course, video is also of bounded extent in time, but it is convenient to ignore this.) A rectangular geometry is the nearly universal choice, although the aspect ratio b/a is subject to debate: in fact the HDTV proposals seek to widen the conventional choice of 4/3 to 16/9, resulting in a more "panoramic" view. Some film formats are wider yet.

The wavelength space Λ can be taken as an interval from 390 nm to 780 nm, roughly the range of human color vision. This implicitly assumes a scalar geometric optics point of view, since currently practical detectors and displays deal only with intensity.[1] Holographic video is more general.

A source coder should, then, take a video signal in this form and produce a sequence of bits representing it. In practice, however, the coding is done in two steps. The first step, usually called "discretization" or "analog-to-digital conversion", produces a high-rate representation, discrete in space, time, wavelength, and amplitude. The second step, usually much more complex, takes this intermediate digital representation as its source and produces the final coded result. The reason for this division of labor is that complicated source coder operations are best implemented with digital signal processing. Still, the choice of discretization is important, especially for image and video systems, where the options are more varied than those for 1D signals.

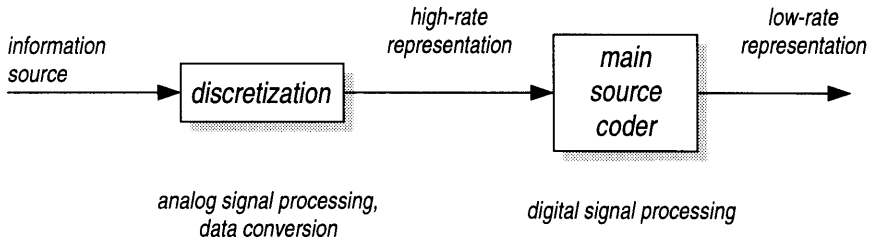


Figure 1.2: Two-step source coder architecture.

The rate for this initial discretization, or sampling format, is set by quality constraints, which in turn are set by economic constraints. It determines the maximum quality, and so is sometimes called the "original" or "source" image, even though it is actually an intermediary. Ideally, our original image would be something like the entire optical field over some aperture.

[1] Human vision is also very slightly sensitive to polarization, so for complete fidelity one should perhaps send four complete images, one for each Stokes parameter. Needless to say, the benefit/cost ratio for this refinement would be miniscule.

This would result in an enormous rate, unrealizable with current technology, because it implies diffraction-limited imaging with temporal response limited only by the optical bandwidth.

More realistic would be an original image designed to exceed every requirement of the human visual system. This is, of course, sufficient for human viewers, but it may result in uneconomic coders and sensors/displays. If the coded rate has some upper bound, then the original image should be chosen with the bound in mind; it is no use specifying some extremely good original image, only to have the source coder throw most of it away. (Actually, this is similar to what happens in reality in the analog-to-digital conversion step: the optical system/image sensor filters the scene in space and integrates in time, producing an output much less detailed than the input.)

What, then, is an appropriate choice for the initial discretization for HDTV? Current CCD, CRT, and LCD-based image sensors and displays that have reasonable cost for entertainment television applications give on the order of 1 Mpixel/frame at 60 frames/s. Although this falls short of transparency for the human viewer in several respects (spatial resolution too low, frame rate lower than some material requires, flicker with CRT displays at 60Hz, etc.) the quality is sufficient for many applications. Displays and sensors are an active area of research, and performance can be expected to increase in the future (HDTV itself likely serving as a stimulus).

1.1.1 Discretization formats

The collection of sampling points is usually chosen to be a lattice. Lattices are uniform, and this keeps the discretization simple; in dimension one, of course, all lattices are scales of one another, so a single parameter (the *sampling rate*) is a complete specification. Since video coding deals with a three-dimensional spacetime, there are many more possibilities.

The simplest of these are products of 1D lattices, referred to as *progressive sampling* or *progressive scanning*, a term used to contrast with the *interlaced sampling* of conventional television, which uses a quincunx[2] lattice in the vertical-temporal plane.

The 2D hexagonal lattice, for example, is more isotropic than the square lattice—a source coder using it may be less likely to produce orientation-sensitive artifacts. (A nonuniform sampling can give even less anisotropy: a "blue noise" pattern [40] would almost completely remove orientation sensitivity, at the considerable cost of randomly-varying filters.)

The choice of sampling has a significant impact on source coder design. The most important decision is whether to couple or decouple spatial and temporal sampling; the choice is also complicated by historical factors (NTSC/PAL/SECAM and the more recent Japanese and European systems are interlaced) and the economics and technological maturity of cameras and displays. From a signal processing viewpoint, it seems clear that progressive sampling is the preferred approach. Source coder operations such as motion estimation and filtering are more natural and efficient with a progressive format; moreover, a progressive source coding format places no restrictions on camera and display format in any case, since simple preprocessing can, for example, upsample an interlaced camera signal for coding. AT&T makes these

[2]Literally, "five-twelfths", describing a Roman coin with five spots arranged on the vertices and center of a rectangle.

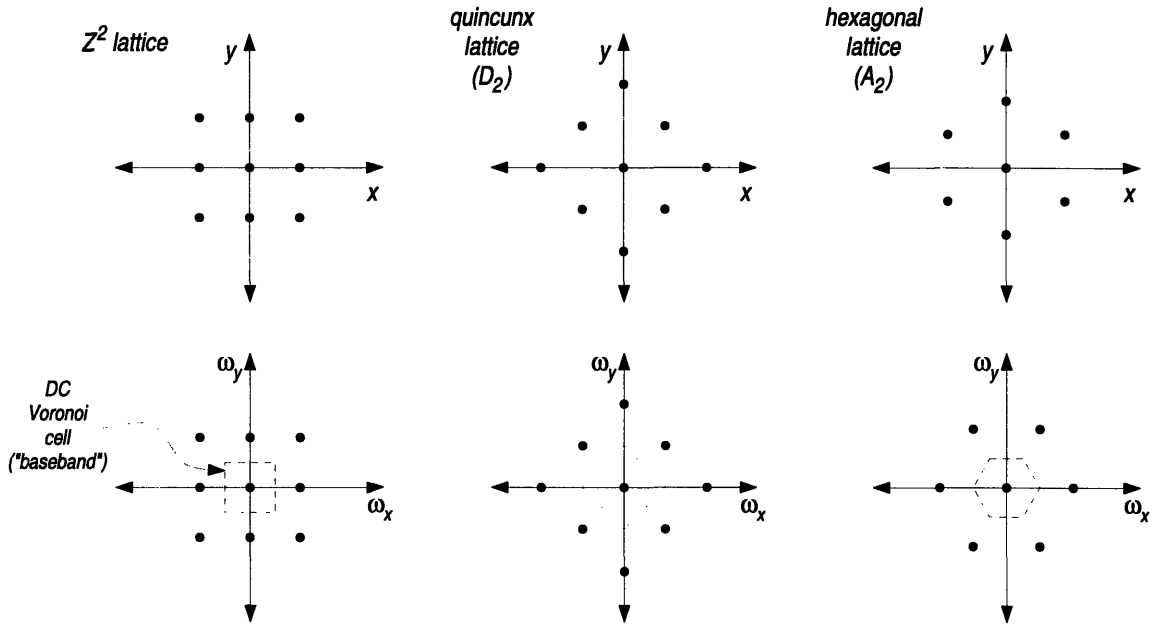


Figure 1.3: Some sampling lattices and their spectra.

and other arguments in a recent paper [7].

Given progressive sampling, a specific spatial resolution, and "square pixels", that is, equal sampling rates horizontally and vertically, the remaining parameter is the temporal sampling rate or frame rate. This can depend somewhat on the application: motion pictures get by with 24 frames/s, since movement is kept low, but high-dynamics material like sports programming can require 60 frames/s or even higher. (Actually most NTSC implementations use 59.94 Hz, the unfortunate result of an obscure audio backward-compatibility issue for color [1].)

As a baseline assumption, the coders used for this thesis assume a rate of 60 frames/s. Material that would not require a rate this high is therefore oversampled, but the temporal part of the source coder can remove some of this redundancy. Note that frame rate should be decoupled from display issues like flicker. There is a clear division of labor between source coding and a given display technology's ability to render signals well. Film systems, for example, have long used double- and triple-shuttering to reduce flicker, and CRTs at 60 Hz are also not entirely satisfactory, especially when bright and in the visual periphery.

Frame rate should ideally be chosen on fundmental grounds of motion quality, rather than dictated by displays. It is valuable to have several frame rates from which to choose; the one-size-fits-all approach of NTSC requires a difficult conversion when sources use non-NTSC rates. Frame rates for HDTV is more a system issue than a signal processing issue.

Discretizing in wavelength would seem simply a matter of specifying a collection of bands covering the visible spectrum. Fortunately, high-resolution spectra are unnecessary—analysis of the human visual system [34] has shown that three suitably chosen spectral bands can represent nearly all color sensations, and that diminishing returns set in rapidly for more than

three channels.[3] Amplitude quantization is usually uniform, for simplicity in data conversion, to an accuracy of 8 or 10 bits per channel.

Finally, the rate of this initial discretization is just the product of all the resolutions. For example, the rate of a 720x1280 pixel format at 60 frames/s, three channels at 8 bits/channel, is $720 \cdot 1280 \cdot 60 \cdot 3 \cdot 8 = 1.33$ Gb/s. This rate is low enough to be suitable for studio or closed-circuit links, and even some networks, but it is much too high for the terrestrial broadcast or satellite channels.

1.1.2 Source coding techniques

For realistic video, there is some degree of redundancy across all of the temporal, spatial, and color dimensions mentioned above, as well as variations in the amplitude-accuracy needed for good fidelity. This redundancy can be statistical, which is the ordinary redundancy associated with correlated samples, or perceptual, in the sense that the human visual system is insensitive to certain distortions and so the corresponding information can be omitted.

Source coders can be broadly divided into two classes: waveform coders and model-based or parametric coders. The distinction between these types is more a matter of convention than strict definition, but generally waveform coders tend to confine their nonlinearities to quantizers: they consist of a linear system followed by a (usually adaptive) quantizer and entropy coder. Model-based coders have stronger nonlinearities—they assume the input is an instance of a model with certain parameters, which are then estimated by some means.

Presently, for video, there are few general models that seem to be useful for source coding, especially if very high quality is a requirement. The high-level quantities most useful as image-parameters might be edges and textures; the corresponding segmentation and estimation problems are difficult and computationally expensive, and can in any case yield an image with a cartoon-like quality, since a model with limited scope will not perfectly represent any given image.

If the available rate is large, then, waveform coding is more suitable, being simpler and more robust. General approaches include predicitive coding, transform/subband coding, and vector quantization (VQ) [25, 21].

Transform/subband coders operate by applying a linear map designed to send the most important parts of the signal into only a few coefficients, i.e., to "represent" the signal so that it is suitable for quantization. The two formulations are equivalent; the transform language is used when the filters have support smaller than or equal to the decimation ratio, and the subband language is used when the filters are larger, that is, when they "overlap" with the subsampling.

VQ operating directly on image samples has been less successful for low-rate coding, for a given complexity, because it is more difficult to constrain—a vector quantizer must be "taught" (by configuring its codebooks) about perceptual notions like spatial frequency, whereas a subband coder can deal with them directly. (Of course, transform/subband coding schemes may

[3] The normal human visual system has three types of color receptors (a fourth, rhodopsin, is active at low light levels), yet because of their overlapping responses it is not quite possible to perfectly represent them with a three-channel additive color system.

use VQ ideas internally, for example to quantize vectors of subband coefficients.)

HDTV source coders, then, are converging to a motion-compensated transform/subband coding architecture. There are many possibilities within this broad class, and good designs will use a set of techniques that fit together well and are friendly to the human visual system.

1.2 Scope of thesis

This thesis will focus on the interaction between spatial and temporal processing for video source coding, in particular for spatial representations of the subband, multiresolution, and wavelet variety. These issues will be examined in the context of a complete implementation, from which images and statistics will be taken.

Chapter 2 discusses various motion-compensation schemes to exploit temporal redundancy; Chapter 3 discusses spatial redundancy and representations to reduce it, including overviews of block transforms (such as the DCT) and subband transforms (such as the lapped orthogonal transform, pyramidal filterbanks, and especially wavelet representations). These spatial and temporal techniques are combined in Chapter 4, together with a flexible means of adaptive prediction when the spatial analysis is by subbands. Chapter 5 discusses system performance, distortion and artifacts, and other aspects of a complete system such as entropy coding, rate control, acquisition, and error concealment; in particular a comparison is made among the proposed new techniques and the blocked DCT. Chapter 6 presents some conclusions and directions for future research.

2

Temporal processing

Realistic video is highly correlated in time. A typical scene contains a collection of objects that behave over short timescales roughly as rigid bodies, keeping their shape and surface structure. These objects either themselves move with respect to some background, or else appear to move because of changing camera geometry.

The impact of this is that each frame usually contains little new information, since it is largely predictable from previous frames. Source coders can exploit this situation in a number of ways. Conceptually, the simplest approaches regard video as a three-dimensional signal and apply 3D versions of conventional subband/transform coding ideas [35] (see Figure 2.1). (DPCM and hybrid schemes fall into this category, with suitable trivial filters.)

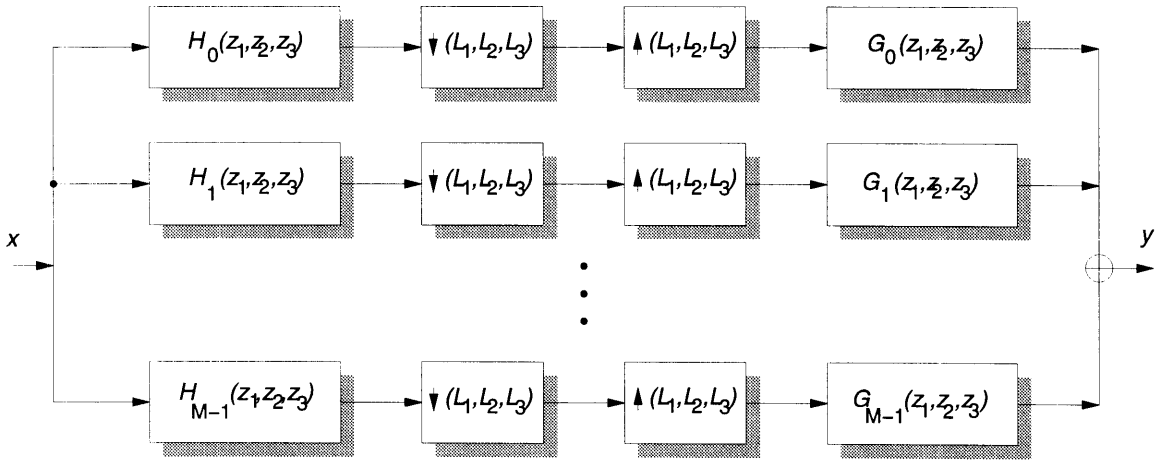


Figure 2.1: 3D subband coding

This turns out to be less productive than one might at first think. In most situations, the temporal prefilter that is applied before the scene is sampled is insufficient to remove all temporal aliasing; actually such aliasing is preferable to the smearing that would otherwise take place. The result is that fixed 3D filterbanks do not give a very sparse representation.

Why is temporal aliasing permitted? In the real world, temporal lowpass filtering is nonexistent. If the eye accurately tracks a moving object, its image on the retina will be similar to the stationary image. Sampled systems should maintain this situation as far as possible—objects should not drastically lose detail if they happen to move quickly, even at the cost of some

"strobing".

One response to this situation would be to oversample in time, so that the video is almost never aliased. The frame rate would need to be high enough to render most motions to the order of 1 pixel/frame. This would greatly complicate cameras and displays, and source coders would need to deal with a high input rate, making them more costly. Also, such a system would have difficulty in coding existing aliased imagery, such as from film or conventional TV (unless such signals were upsampled with motion-compensated interpolation, which would have its own problems).

Interestingly enough, this is exactly the opposite of the spatial situation. There, the eye regards fuzzy edges as "normal", perhaps because defocusing and depth-of-field mechanisms are so fundamental physically, but aliased edges are unnatural and are regarded as artifacts.[1] In video coding, as in physics, space and time are essentially different dimensions.

2.1 Motion estimation

A more powerful structure results from explicitly including image motion as a means for temporal prediction. Motion estimation itself is a broad subject, having applications in machine vision, stereo imaging, remote sensing, etc. in addition to source coding [31].

To illustrate the basic principle, Figure 2.2 shows two adjacent (cropped) frames from an image sequence, "Marcie". The difference picture (biased with gray, so that negative values are dark gray and positive values are light gray) reveals that the face has moved with respect to the background. The structure of the motion is shown by the vector field; deciding what form this field should take, and devising some means of estimating it, are central design issues.

The second difference-picture in Figure 2.4 shows the result of subtracting the second frame from an extrapolated version of the first frame (using the vector field). The energy in this image is much reduced relative to the simple difference, and many fewer bits are needed to transmit it to a given fidelity.

As with all adaptive predictive coding schemes [25], motion estimation may be used in either a feedforward or a feedback mode. Forward prediction sends adaptation side information explicitly to the receiver, whereas backward prediction computes adaptation information from the received data. Figure 2.5 illustrates the two alternatives.

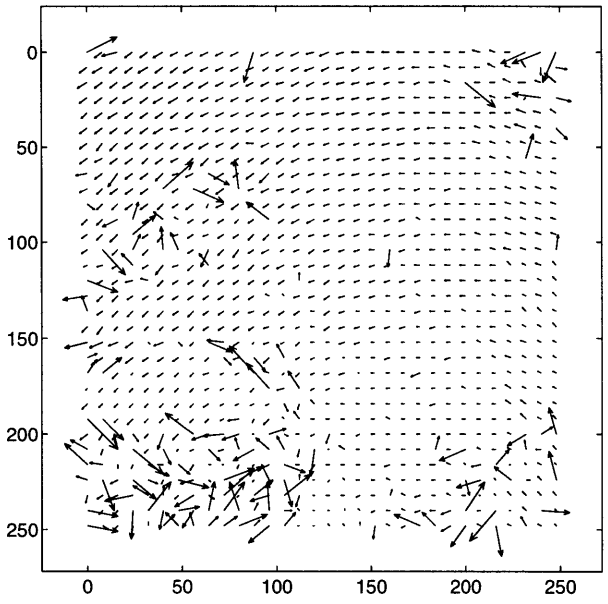
Forward prediction requires extra capacity for the side information, but backward prediction forces all receivers not only to implement the predictor, which is expensive for video, but to implement it in the same way for all time. (The transmitter and receiver must compute precisely the same predictor, or the signals in the two prediction loops will diverge.) This restricts the flexibility of the encoder, and so forward prediction is the usual choice.

[1] Perhaps this would not be so if the retina were undersampled or sampled more regularly. The fovea's receptors are distributed pseudorandomly and rather more finely than the diffraction limit at a typical f/8 relative aperture, and tremor further smooths the sampling process.

(a) Frame 1

(b) Frame 2

Figure 2.2: DPCM and Motion-Compensated DPCM



(a) Vector Field

Figure 2.3: Vector Field for Motion Estimation

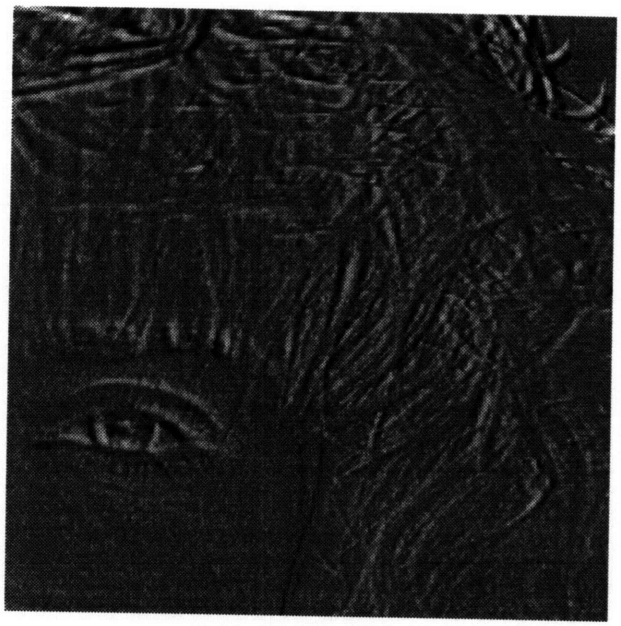
2.1.1 Motion representation

Naturally, there are many ways of representing image motion. In some situations, global representations can be used, which estimate only a few parameters that attempt to characterize motion across the entire scene. A typical scenario would involve images with relatively fixed internal structure and, perhaps, known camera motions. High prediction accuracies can result.

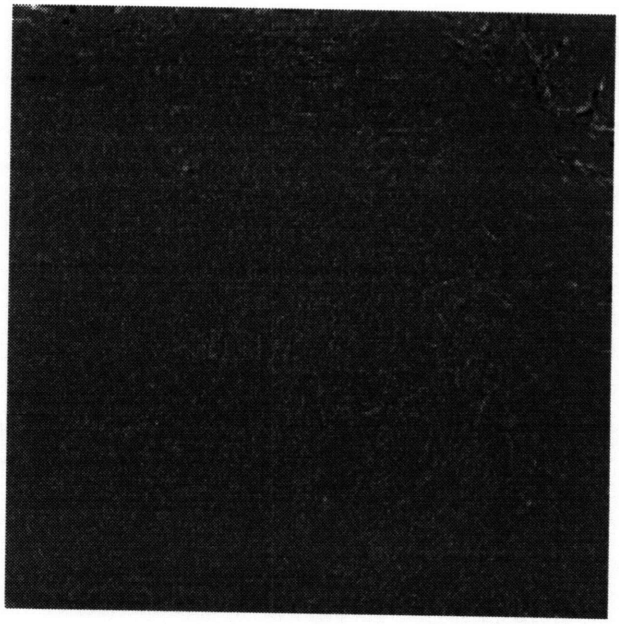
For general image coding, there are no useful global motion models. Any image sequence, within reason, is fair game for coding, and many will involve motion that varies over the scene: objects and people can move in arbitrary ways, camera motion will result in changes in perspective, and moving objects will obscure or reveal the background.

Therefore, motion should be represented locally, and this implies some sort of segmentation. In principle the most useful segmentations (and the most accurate motion models) view the image from a high level. If the source coder is to benefit from notions like "object X moves in fashion Y", then it must have a flexible "object" concept and segmentation algorithms to support it.

This is difficult and costly, so the prevailing trend is to work with simpler segmentations and accept the tradeoff of lower prediction accuracies and higher side-information burdens. Nevertheless, there is great potential for high-level coding, especially when the auxiliary (from the source-coding point of view) data is independently interesting.

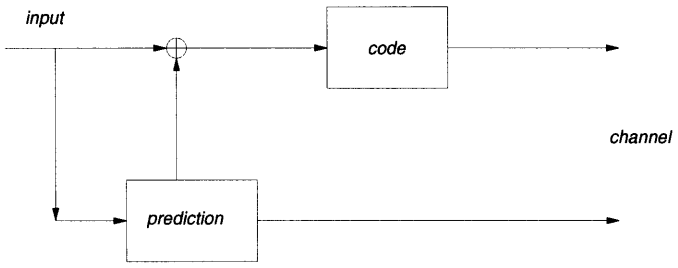


(a) Simple Difference

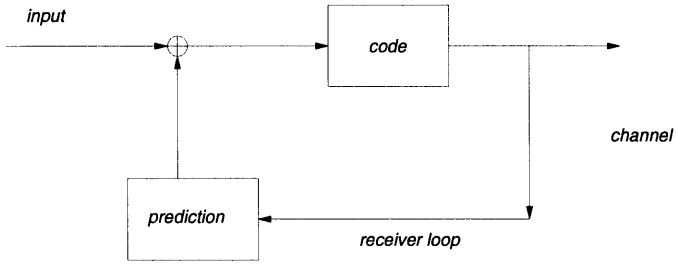


(b) Motion-Compensated Difference

Figure 2.4: Motion Estimation Prediction Errors



(a) Forward Prediction



(b) Backward Prediction

Figure 2.5: Adaptive Prediction Architectures

2.1.2 Block matching

Block matching uses the simplest nontrivial segmentation, that of a fixed, uniform partition into blocks (often related to the spatial transform, though this is less important for subband/wavelet coders, as we will see in Chapter 3). The algorithm for estimating motion vectors is simple: match each block to its translates in the previous frame, and choose the vector with smallest error.

The block size involves a tradeoff between adaptivity and side information. Large blocks cannot adapt to motion localized to a region smaller than the block. Very small blocks (say, 4×4 pixels or smaller) adapt quickly, yielding higher prediction accuracies, but involve more side information (even after entropy coding). To go even further in this direction, one could estimate motions very finely, at the pixel rate or even higher, and then perform drastic lossy coding on the resulting detailed motion map. This begins to look like the high-level segmentation algorithms mentioned above.

2.1.2.1 Range and accuracy

The "full search" version of block matching enumerates all vectors over some range, say $\pm r$ pixels, to some precision p. This gives $4(r/p)^2$ trial vectors; each is used to interpolate a block from the previous frame. Figure 2.6 shows the geometry.

Motion estimator range is determined by the fastest motion the coder needs to track. Reasonable limits are on the order of one picture width per second—at 1000 pixels and 60 frames

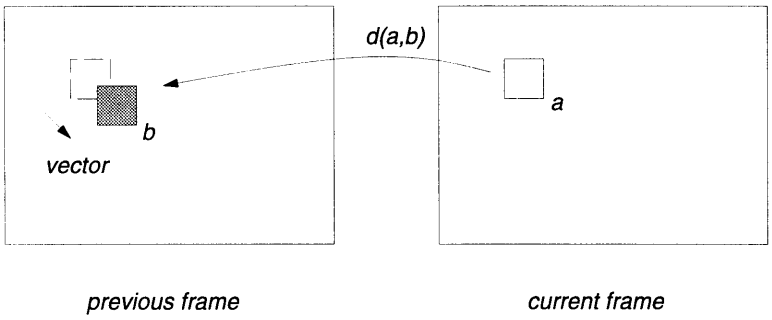


Figure 2.6: Full-search motion estimation

per second, this gives about 16 pixels/frame. The coder used for this thesis uses a window of $\pm 16 \times 16$ pixels; some proposals (at HDTV resolutions) extend to ± 32 pixels/frame horizontally, since horizontal pans are faster and more common than vertical pans.

For forward adaptation, specifying large motion ranges for decoders does not involve much cost. Large vectors are rare, and the encoder is free to implement some subset of the specification. Increased range is a memory-system issue in the decoder, since the motion compensator will need access to all pixels in range, but (especially horizontally) this is not serious.

Note that with such fast motions, camera filtering may substantially blur the image in the direction of motion. High-accuracy vectors will not be needed in this situation, and the encoder can use this to economize on side information.

What happens if image motion exceeds the motion estimator's range? The predictor will be degraded, since the estimator cannot see the true motion and will make another choice. This makes the estimator's failure "soft"—the spatial coder will see a more energetic image, and the fidelity will decrease in a global and subtle way.

Motion vector accuracy needs to be better than p pixels for the resulting prediction to be useful over corresponding scales. At a minimum, one-pixel resolution is needed to usefully predict high spatial frequencies. 1/2-pixel resolution (resulting in 1/4-pixel maximum error) is noticeably better with sharp imagery, but beyond this diminishing returns set in.

2.1.2.2 Refinements and implementations

Numerous refinements to the basic block-matching scheme have been proposed [31]. Most common are schemes aimed at reducing the complexity of full search by considering subsets of vectors either over the original image or over successively decimated versions of the original image (which meshes well with wavelet representations).

Such hierarchical motion estimation cannot, of course, guarantee optimal vectors. Highpass texture is one class of features for which motion may be invisible at large scales; initial estimates from the upper levels may lead the search in a random or wrong direction.

Hierarchical motion estimation also requires a nontrivial control structure, making it more suited to general-purpose architectures. Full search, by contrast, has a very simple structure that lends itself to parallel implementation—VLSI systolic arrays can now easily realize full search for realistic ranges and pixel rates.

Another class of refinements involves extending the uniform-translation motion model. Changes in scene illumination, for example, would normally inject significant energy into the residual image; sending a scale factor to model this would be worth the cost for many blocks. More geometry could also be included in the vector field (for example, a field of affine maps), modeling motions more general than translations. Such extensions would need to show significant benefits to warrant their complexity.

2.1.3 Distortion metrics for block matching

Block matching requires an effective, robust, and preferably simple metric on pairs of blocks. This function need not attempt to closely match the human visual system response. Its job is rather to permit the motion estimator to obtain vectors that are as useful as possible to the source coder; this need not have any particular relationship to the subjective sameness of the blocks under consideration.

Two measures in common use are the absolute error and squared error defined by

$$\|a - b\|_1 = \sum_i |a_i - b_i|$$

$$\|a - b\|_2 = \left(\sum_i (a_i - b_i)^2 \right)^{1/2}$$

There does not seem to be much difference in performance between the two. On most architectures $\|\cdot\|_1$ is cheaper, since it requires no multiplies. (The square root may be elided when computing $\|\cdot\|_2$, since it is monotonic.)

These functions assume a scalar image, usually taken as the luminance part of the signal. This is again for reasons of cost—a distortion function using full-color samples will be more expensive. The luminance subspace usually captures any relevant image motion, but not always: a so-called isoluminant edge, one with different colors but the same luminance on either side, will confuse a scalar estimator and damage the resulting prediction. Also, a color estimator will be more robust to noise than its luminance-only counterpart.

An important benefit from forward adaptation, mentioned above, is that the encoder may use any algorithm it pleases to compute motion vectors. Higher-quality encoders may use more computationally intensive algorithms, yet remain backward-compatible with decoders.

2.1.4 Multiscale vector fields

The coder used in this thesis, as well as other proposed coders [6], avoids to some extent the limitations of a single motion-estimator block size by using two. The granularity of other coder objects is 16×16 pixels (see chapter 5); a heuristic is used to decide whether a given 16×16 block should use a single motion vector or be subdivided into four 8×8 vectors.

2.2 Motion compensation

Finally, a coding system must decide how to actually construct the predictor signal from the motion data. When a given vector has a non-integer component, the block must be resampled from the previous frame.

This resampling should ideally be the same as that performed by the camera. If the image is moving by, say, 3.2 pixels per frame, then the best match to the current block would be the previous block convolved with the camera response and sampled at -3.2 pixels. The coders discussed here use a bilinear interpolator. This is doubtless a choice that could be improved, given a better characterization of typical camera filtering.

Another way to interpret motion compensation is to imagine the prediction error to be the result of a filter applied "along the vector".[2] Thus far, the filter is merely $1 - z^{-1}$, i.e., simple subtraction. (Or, including the spatial vector (α, β), the filter is $z_x^{-\alpha} z_y^{-\beta}(1 - z_t^{-1})$.) Other choices are possible. Filters such as $1 - \alpha z^{-1}$, for $\alpha \in (0, 1)$, have decoder loops that look like leaky integrators; while this sacrifices some prediction efficiency, it is one way to approach the acquisition problem discussed in section 5.6.

Still other filter choices can involve prediction from the future, for example the central difference $-z/2 + 1 - z^{-1}/2$. This can give better and smoother prediction, but at the price of more computation and higher delay and memory costs (especially for high resolution applications).

Vectors are upsampled to the pixel rate by repeating them over the block. Some thought was given to applying some other interpolation, perhaps in order to smooth sharp transitions between motion-objects. This does not seem to improve matters, since vector fields are inherently discontinuous—enforcing higher regularity does not improve predictor accuracy.

[2] In this view, motion estimation is merely adaptive linear prediction of a special kind.

3

Spatial processing

The prediction error after temporal processing will still contain some redundancy, depending somewhat on the sophistication of the temporal model used. For cases in which new imagery pans or cuts into the scene, no temporal model will be useful, and it falls entirely on the spatial part of the coder to represent it efficiently.

The problem of coding motion-compensated prediction errors, or residuals, resembles the general still-image coding problem, and many of the same techniques apply. One caveat is that the spatial representation should perform well with both prediction-error-like areas (which tend to be "edgy" and fairly white) and original-image-like areas. The residual signal of normal video is dominated by prediction-error-like areas, so it is important to code this class well.

3.1 Context of spatial coding

Figure 3.1 shows the context of spatial processing within a video coder. First a temporal prediction, consisting of a motion-compensated version of the previous frame, is subtracted off. The resulting prediction error is transformed to a more suitable basis for quantization with a linear map T. The choice of an appropriate T is the subject of this chapter.

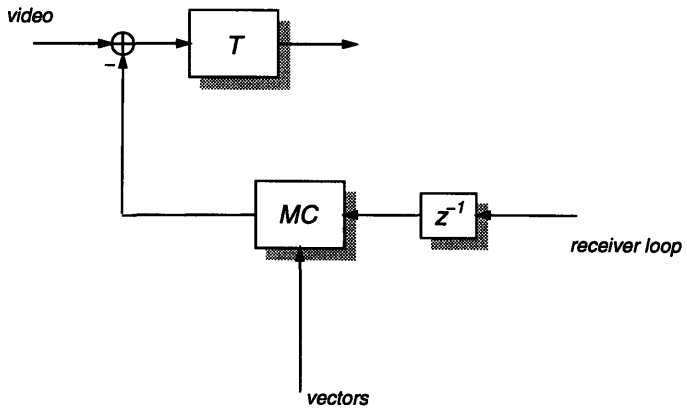


Figure 3.1: Spatial processing within a video coder

3.2 Transform/subband coding

The aim of transform/subband coding is to remove linear statistical redundancy from the source. Historically, transform and subband coding had evolved separate terminologies, but they are aspects of the same thing, transform coding merely distinguished by analysis and synthesis filters with support no larger than the decimation factor. Figure 3.2 shows the analysis and synthesis stages of a general multirate filterbank.

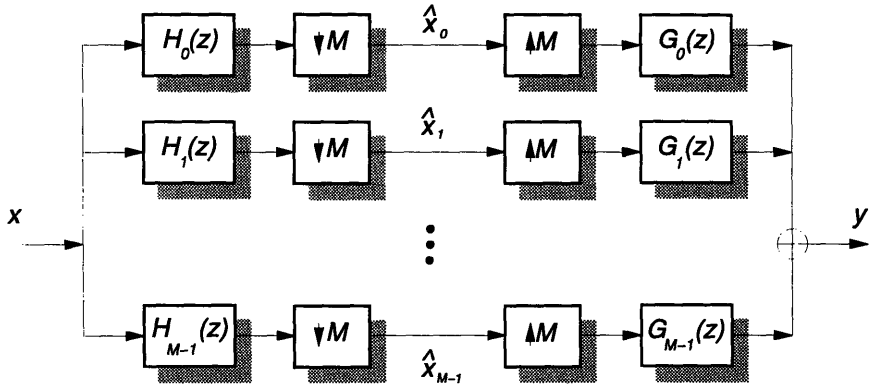


Figure 3.2: Multirate filterbank for transform/subband coding

There is a large collection of filters H and G that are perfect-reconstruction (PR), in the sense that the recovered signal y is identical with x, perhaps with some delay. There is an even larger collection whose members are nearly PR. Choosing among them is a matter of comparing their performance.

Ultimately the most fundamental performance measure is the subjective response of the human visual system. This is the measure used when "tuning" coders across a range of inputs, testing among competing coders, and so on. Such evaluations are time-consuming, and they are not terribly tractable mathematically, making them difficult to use for a priori design. Analytic design techniques and stochastic image models may suggest designs, but in practice they must be tempered by eyeball examinations of the results.

3.2.1 Fixed blocksize transforms

Fixed-block transform coding of images is a mature subject, and there is a large variety of transforms available. If performance is taken to be mean squared error (MSE), there is a (signal-dependent) optimal transform called the Karhunen-Loeve transform (KLT). If x denotes the signal process (length-n blocks of samples), then

$$R = E(x \otimes x)$$

is the crosscorrelation matrix of x, and the KLT is an orthonormal collection of eigenvectors of R. (R is positive definite.)

The transformed vectors $KLT(x)$ will then have a purely diagonal crosscorrelation matrix, so that distinct members of $KLT(x)$ are uncorrelated. This can be shown to result in the

smallest mean-squared error under truncation of the coefficient-list, i.e. the KLT "compacts" the energy optimally (the energy itself is unchanged, since KLT is orthonormal).

The KLT is useful as a bound for energy compaction performance of transforms, but it falls short of being an ideal solution for transform coding of images. The reason for this is that mean-squared error is far from a complete model of subjective image distortion. The filters the KLT specifies do not respect important perceptual issues, such as the spatial disposition of quantization noise (whether it is kept close to image detail or tends to spread out, for example) and the nature of errors among independently transformed blocks.

The KLT also depends on estimates of second-order signal statistics, which requires on-line computation if it is to be adaptive across blocks. Finally, the MSE optimality of the KLT is distinct from optimality under quantization—quantization and coefficient-list truncation are different processes.

If we consider only nonadaptive transforms that necessarily only approximate the KLT, the simplest ones require no multiplies, trading performance for very low computational cost. The discrete Walsh-Hadamard transform (DWHT) is a representative example, shown below for an 8x8 block size.

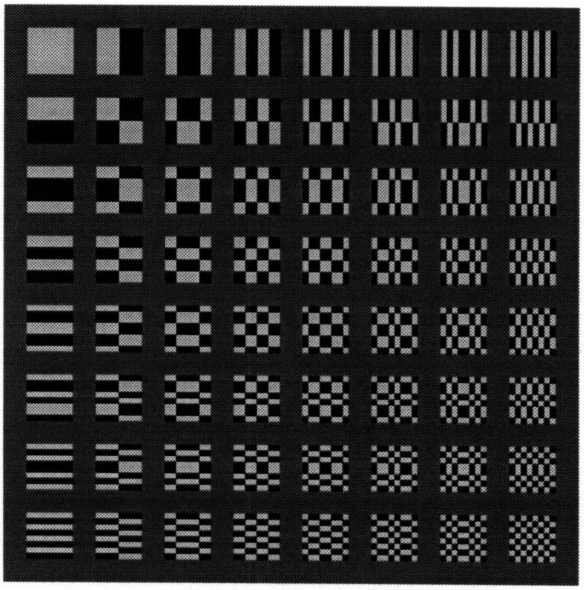


Figure 3.3: 8x8 DWHT

Experiment shows that the DWHT does not approach the KLT bound very closely for realistic images. The intuition is that the basis vectors are not sufficiently regular, and in fact the KLT for realistic image models usually contains smooth, cosine-like vectors. The discrete cosine transform (DCT) approximates this situation more closely, and it has remained popular for image and video coding for many years. It is given by

$$DCT_{ij} = \sqrt{\frac{1}{n}} c_i \cos(2\pi(i-1)(2j-1))$$

where $c_0 = 2$ and $c_i = 1$ for $i \neq 0$. The 8x8-point DCT is shown in Figure 3.4. Notice that each

basis function fills the block approximately uniformly.

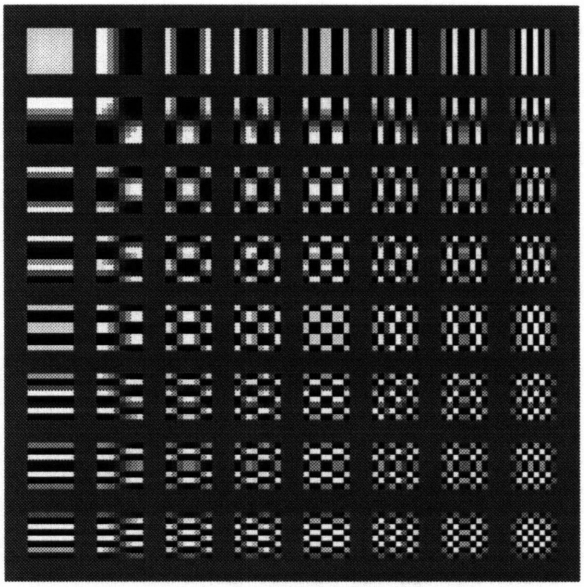


Figure 3.4: 8x8 DCT

The choice of block size results in a tradeoff between transform coding gain and stationarity; smaller blocks may adapt more quickly to changing image statistics.

Though the DCT succeeds fairly well in producing uncorrelated transform coefficients, it suffers from spatially regular *block artifacts*. These are image discontinuities arising from quantization noise. Since the block supports are separate, the errors along block boundaries are independent; since the boundaries form a regular structure, they are noticeable to the eye. Figure 3.5 shows an enlarged section of a DCT reconstruction at a rate low enough to show blocks distinctly.

There is another class of block transforms, called lapped orthogonal transforms (LOTs), that aims to minimize block artifacts by overlapping synthesis kernels rather than keeping them completely disjoint as in the DCT. They are described by Malvar [30]. A representative filterbank is shown in Figure 3.6; the kernels are smoothly tempered at their boundaries, though the support remains uniform across spatial frequency.

While blocking effects are reduced, they can still be perceived as "softer blocks" in images, since the visual system can detect the aliasing despite the greater smoothness of the transitions. Figure 3.7 is coded at a similar rate to the DCT example.

3.2.2 Multiresolution transforms

Multiresolution schemes attempt to finesse the fundamental tradeoff of fixed-blocksize transforms. Rather than representing image detail by combinations of complex basis functions distributed in spatial frequency, they use a limited set of "mother functions" which are then replicated across space and scale.

A pyramidal filterbank for image coding was described by Burt and Adelson in 1983 [9].

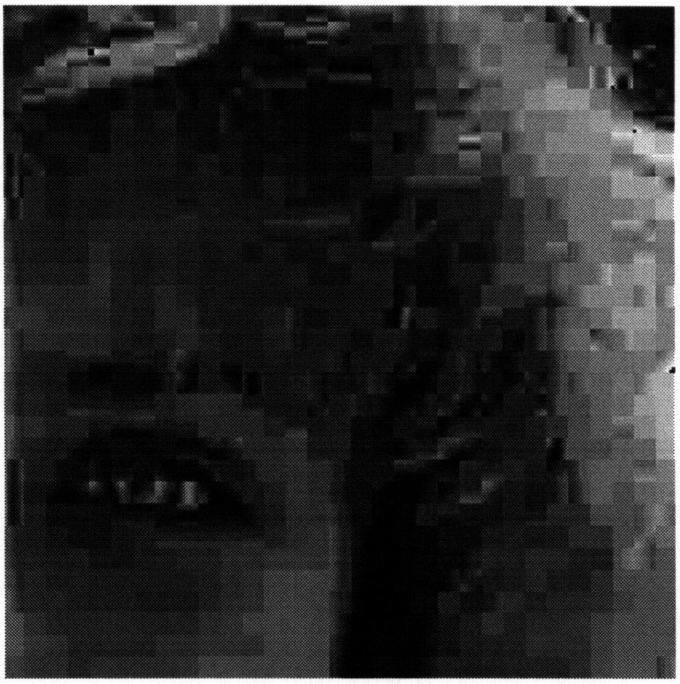


Figure 3.5: DCT blocking artifacts

This structure has complete flexibility in the choice of filtering, at the cost of an oversampling ratio of, asymptotically, 4/3 (in dimension 2).

Figure 3.9 shows a collection of filterbank outputs for a sample image. The smallest image represents detail on the coarsest scale, and each succeeding image represents the difference between the interpolated coarse image and the finer image.[1]

Such non-critical sampling, or "data amplification", in the sense of producing more coefficients than original samples, is not inherently evil. It is possible that more coefficients could yet result in fewer bits after quantization and coding, and the unconstrained selection of filters could result in better perceptual performance.

Recently, however, critically sampled dyadic tree structures offering the same advantages for image coding as pyramidal filterbanks have been designed. These have been described from both the subband filtering and from the wavelet point of view; Vaidyanathan's *Multirate Systems and Filter Banks* and Daubechies' *Ten Lectures on Wavelets* are representative treatments [41, 16].

3.2.2.1 Wavelets

A wavelet ϕ is a function that is well localized (so that it is small in extent) and has vanishing zeroth moment (so that it superimposes itself cleanly on larger waves). The map itself, though, is less important than the family $\phi_{a,b}$ of related maps it engenders by scaling and translation,

[1] The structure is identical to the multigrid algorithms of computational linear algebra, though in image coding there is no notion of back-and-forth among the levels: the filterbank is merely evaluated and the results quantized.

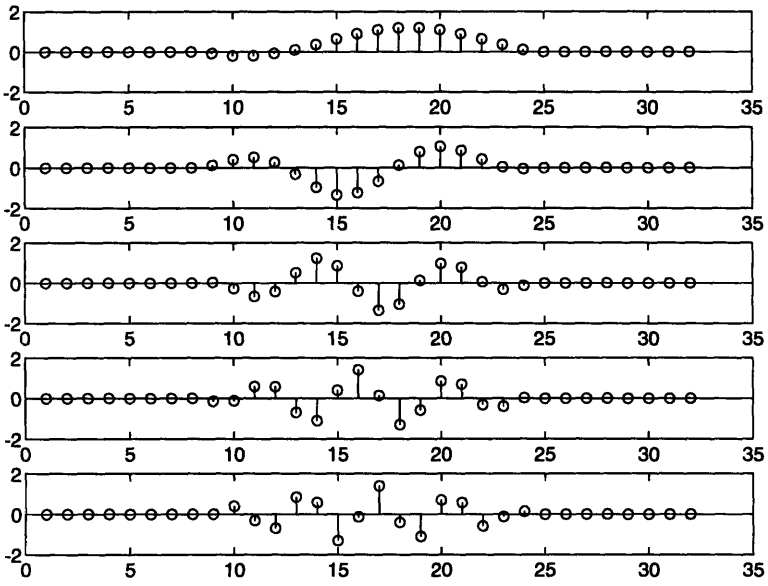


Figure 3.6: Lapped orthogonal transform

since it is this family that serves to represent other functions. It is defined by

$$\phi_{a,b}(x) = 2^{a/2}\phi(2^a x - b)$$

for $a, b \in Z$. (Naturally, there are also versions that sample continuously in space and scale.) Each $\phi_{a,b}$ is a copy of the "mother wavelet" ϕ, compressed by a factor of 2^a, translated by b, and suitably normalized. Figure 3.10 shows a mother wavelet and a few of the children.

Wavelets are most convenient when $\phi_{a,b}$ is an orthonormal basis of $L^2(R)$. Orthonormal wavelets with compact support are available with arbitrarily high regularity, as shown by Daubechies [16], with a tradeoff between regularity and spatial extent. If compact support is not a requirement, simpler constructions suffice.

In discrete time, a wavelet corresponds to a quadrature-mirror filter pair H_0 and H_1; Figure 3.11 shows the transform algorithm as an iterated convolution and downsampling. As always, discrete time imposes a limit on the frequencies the sampling lattice can support.

There are straightforward extensions to $L^2(R^n)$ via product constructions. Special wavelets designed for the unit interval have also been designed, although for image coding it is not clear whether they are better than symmetric extension using unbounded wavelets. Figure 3.12 shows the filterbank signals for a 2D wavelet transform, organized in the canonical way. The corresponding synthesis kernels are shown in Figure 3.13; compare with those of the DCT.

Full wavelet transforms have $O(n \log n)$ complexity, since the convolution kernel is fixed in length and the pyramid will have $\log n$ stages. Many image coding applications in fact truncate the pyramid after three or four levels, since the need for information at very coarse detail is not great.

The wavelet transform algorithm is very uniform. All stages use the same small set of coefficients, unlike the FFT and DCT, in which different sets of sinusoidal multipliers are needed

Figure 3.7: LOT blocking artifacts

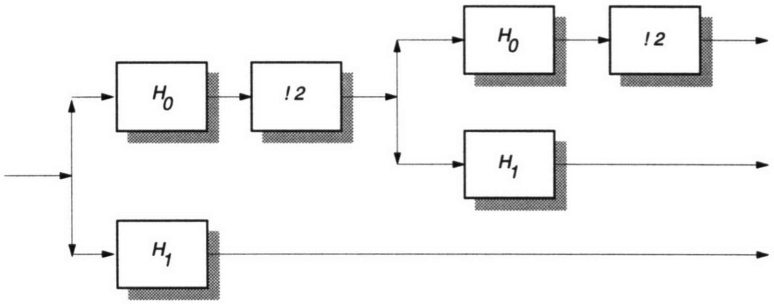


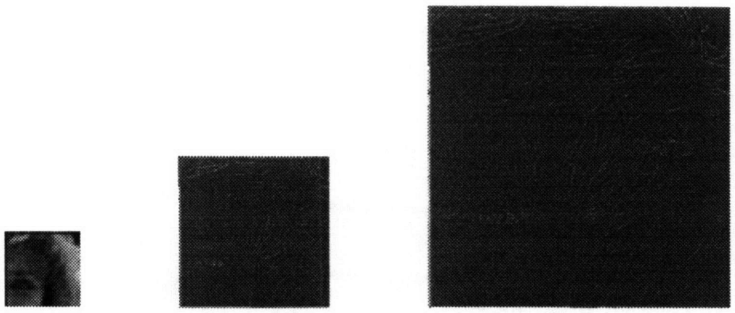
Figure 3.8: Oversampled pyramidal filterbank

for each stage. In addition, high-detail subband signals are available from wavelet transforms with low latency, whereas all FFT coefficients are arrived at simultaneously.

Symmetry and perfect reconstruction are unfortunately incompatible for wavelets. One approach to recovering symmetry is a biorthogonal wavelet system, for which there are two independent analysis and synthesis kernels [5]. By contrast, the wavelet synthesis filters are related in a simple way to the analysis filters.

3.3 Criteria for image coding

Image coding places unique requirements on analysis/synthesis filterbanks. In addition to sparse representations for most image types, filterbanks should respect a number of perceptual and systems issues.



(a) DC (b) AC mid (c) AC high

Figure 3.9: Pyramidal filterbank signals

3.3.1 Isotropy

As mentioned in Chapter 1, isotropy of a filterbank structure occurs on two levels: first on the level of the sampling lattice itself, and second as a result of the geometry of the responses of the filterbank analysis and synthesis filters. Perfect isotropy, in the sense of invariance of the entire system under rotation, is not possible on either level (even the human visual system has some preferred directions; for example, deterministic halftoning patterns are less visible at 45° than at 0° to the horizontal).

From an image coding point of view, tensor product filters based on 1D designs seem to be adequate. These distinguish two directions, horizontal and vertical, but superpose the two diagonal directions into a single subband. At low rates, the result of this can sometimes be seen in a "notching" or jaggedness of some edge features, as diagonal coefficients are coded with greater or lesser accuracy.

Another symmetry a coding system ideally should have is translation invariance, that is, lack of aliasing. In the absence of quantization noise, of course, all perfect-reconstruction systems are translation invariant; under quantization, however, it is the statistics of the distortion, viewed as a random process, that is important to the eye. Is the distortion stationary, in the sense of conditional probabilites that do not change under translation? That is emphatically not the case with the DCT, for which transients are more likely along the decimation grid; it is the cause of block artifacts.

3.3.2 Perfect reconstruction

It might seem axiomatic that perfect reconstruction is a desirable property, but it does come with the price of restricting the choice of filters. Since the output of the filterbank is to be quantized anyway, some balancing of filterbank distortion and quantization noise might seem appropriate.

In practice, there is apparently sufficient freedom to choose perceptually nice PR filterbanks. Doing so renders the quantizer the sole source of distortion in the entire system, which simplifies evaluation and tuning.

The filterbanks used for this thesis, namely the DCT and a biorthogonal wavelet filterbank, are both PR. (Finite-precision arithmetic contributes some error, of course, but it is much smaller than any quantization noise.) Section 5.2 discusses filterbank design and implementation.

3.3.3 Small support

It is ultimately a matter of physiology and perception that most signal types are best thought of as purely time- or space-like, purely frequency-like, or somewhere in between. Vision starts with rod cells and cone cells. They are compact sensors, only a few microns across, and they are situated in the focal plane of the eye's optical system. If each cell's output were used directly, without subsequent signal processing, then distortions in an image coding system would be simple to evaluate and control.

For example, hearing does not operate directly on the incident acoustic waveform. Such structures are in principle possible; one can imagine a kind of shift-register arrangement in which sound propagates along a transmission line and sensor cells sample synchronously along the line. But acoustic information is most often in the form of modulated relaxation oscillators (animal speech) or modulated resonators, because they are easy to implement physically. A direct spectral acoustic sense, realized in the cochlea as a mechanical frequency analyzer, is more efficient for these signals. (The corresponding switch for vision is well out of the realm of possibility: lenses are much more cost-effective than phased-array optics.)

In a similar way, investigation has shown that processing in the retina and cortex does result in some coupling of responses in neighboring areas [34]. This can be interpreted as a kind of filtering (though the responses are not always usefully modeled as linear). These coupled responses, though, are still quite local.

What, then, does this imply about the space-domain requirements for image filterbank kernels? They should be short, since short synthesis filters will closely confine quantization noise to energetic features in an image. Noise that is allowed to spread becomes more visible, for the same noise energy. Such filters should also attempt to mimic other aspects of low-level vision, such as orientation sensitivity and hierarchical processing.

Strictly compact support isn't necessary from a vision point of view; fast decay is sufficient. As with perfect reconstruction, though, it is convenient—compact support completely isolates one region from another.

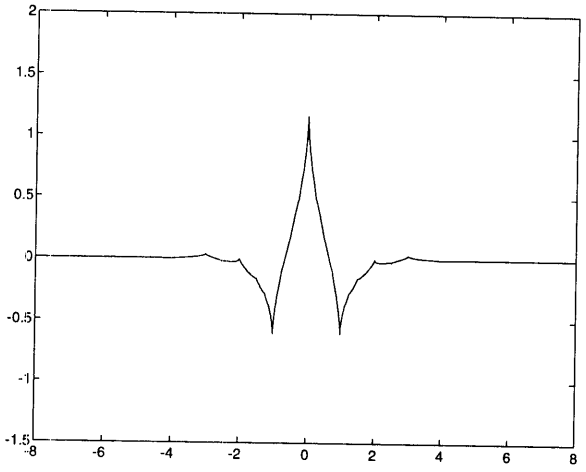
3.3.4 Energy compaction

As discussed above, energy compaction is an important efficiency metric. While crude image models, like $AR(1)$ processes with a high correlation coefficient, should not be counted on to provide designs with good perceptual performance, raw statistical performance is still important to efficiency.

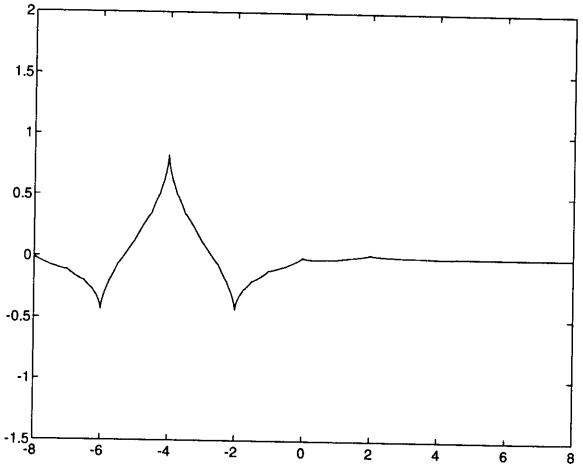
3.3.5 Symmetry

Edges and lines are common features in natural imagery, and image coders should preserve their form as well as possible. A given edge between two objects has an odd symmetry about its midpoint, and an unresolved line has an even symmetry. For asymmetric filters, quantization noise could disrupt that symmetry and bias the features in an unpleasant way.

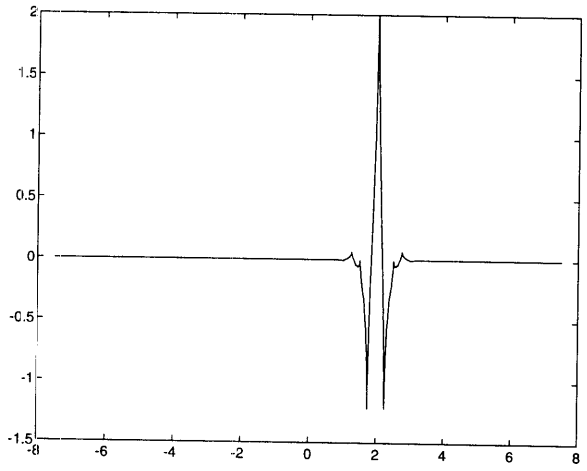
For multiresolution analysis, biorthogonal systems offer exact symmetry at the cost of a larger design problem, since two wavelets must form a PR filterbank.



(a) Mother wavelet ϕ



(b) $\phi_{-1,-2}$



(c) $\phi_{1,4}$

Figure 3.10: Wavelet family

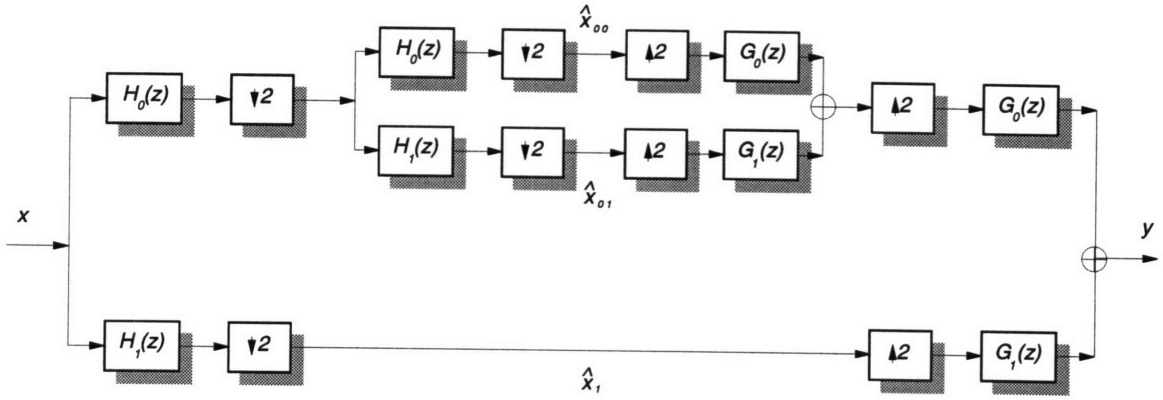


Figure 3.11: Tree-structured filterbank

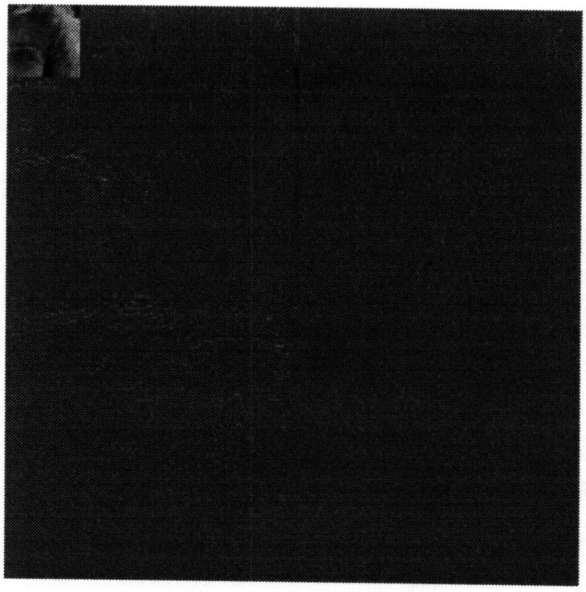


Figure 3.12: Wavelet filterbank signals

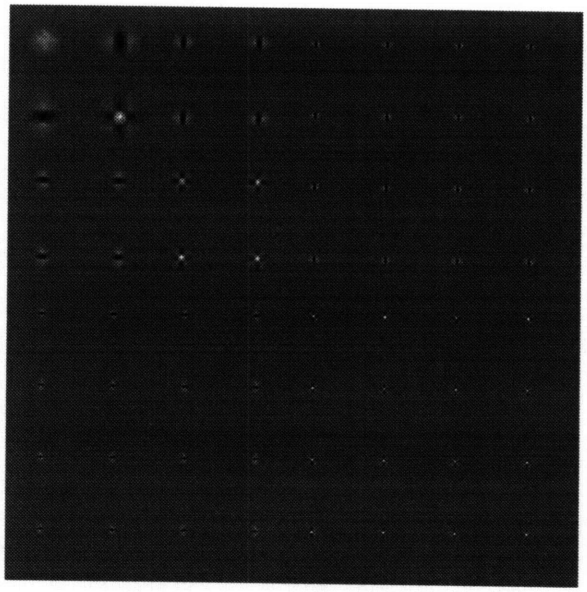


Figure 3.13: 8x8 wavelet synthesis kernels

4

Adaptive prediction

Thus far, temporal and spatial signal processing have been treated independently. The redundancy mechanisms are unrelated—motion predictability of a given scene is independent of its spatial predictability—so it might seem natural to simply cascade the two techniques.

In practice, though, some coupling is necessary. Motion-compensated processing does not always give a useful prediction signal. Such situations can arise either globally (scene cuts) or locally (object occlusions, pans), and using the unmodified prediction can result in very unnatural artifacts.

Conventional block-transform coders solve this problem by adapting the prediciton signal to scene content. If, in a given region of the image, there is much new signal that cannot be predicted from the previous frame, the coder discards the prediction (by setting it to zero).

Since this thesis investigates wavelet representations for video, there should be a comparable mechanism to deal with such signals. It is here, for subband coders, that a significant wrinkle arises, since prediction segmentations that work acceptably well with block transforms fail when directly applied to subband coders.

Section 4.3 will show a novel restructuring of the prediction loop, so that a video subband coder can use a natural segmentation that results in good performance. The strategy is to use blocks of filterbank coefficients, rather than blocks of image samples, as the basis for prediction decisions by the encoder.

4.1 Need for adaptive prediction

Figure 4.1 shows a block diagram of a video coder without adaptive prediction, consisting of a motion compensation step and an independent spatial coding step. (Throughout the next two chapters, block diagrams will contain abbreviations of common operations: T and T^{-1} will denote a block transform or subband filtering operation and its inverse, Q and Q^{-1} a quantizer and inverse quantizer, ME and MC motion estimation and motion compensation, and z^{-1} a unit delay, usually a frame.)

For sufficiently benign video, without cuts or complex motions, this can work well, but the simplest and most dramatic failure mechanism is a cut from a busy scene to a simple one. Figure 4.2 shows two adjacent frames and resulting prediction error given to the spatial coder.

Notice that the residual contains the new scene information, as it should, but associated

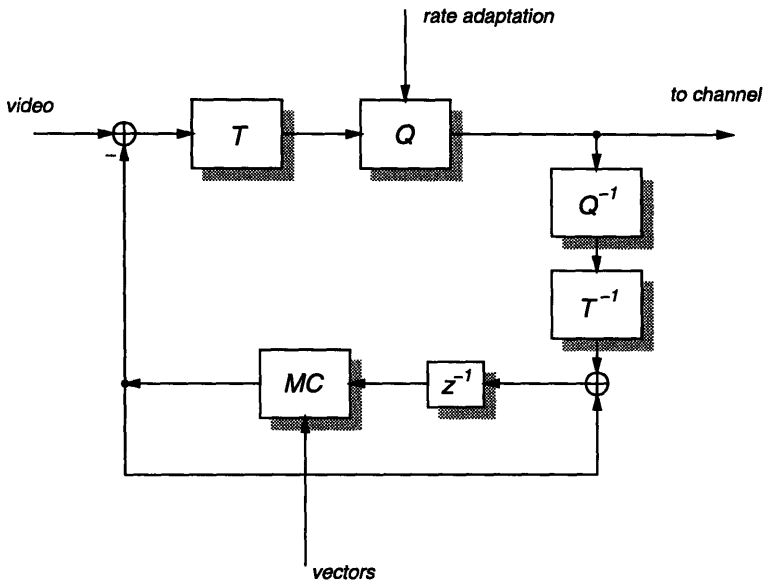


Figure 4.1: Video coder without adaptive prediction

with an energetic corrupting image similar to the previous frame. The spatial coder is in effect being asked to transmit information to enable the decoder to "uncode" all the detail in the preceding scene, in order to render it close to the new simple scene.

This is naturally quite wasteful, and it may result in a "scrubbing" or "trailing" artifact in which several frames are required to remove all traces of the previous scene, because of quantization noise.[1]

Figure 4.3 shows the first three reconstructed frames after the cut, assuming no motion in the new scene, and using a simple rate control scheme. (Rate control is discussed in chapter 5; a coder might choose to use a high rate up front to mitigate the "scrubbing", at the cost of buffer complications.)

In this example the residual is costlier to transmit than the new frame itself, although things are not always so drastic. Even for situations in which only a small part of the image will benefit from adaptive processing, though, the net efficiency of the coder will improve—the side information requirements are small.

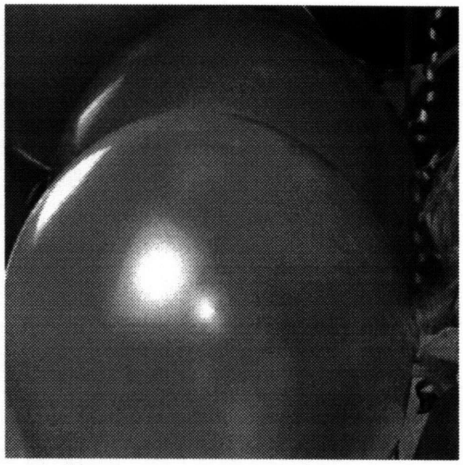
4.2 Segmentation and prediction measures

As with motion estimation, adaptive prediction requires an appropriate segmentation. The task is to specify the region for which temporal prediction is useful (and, of course, the complementary region for which it is not).

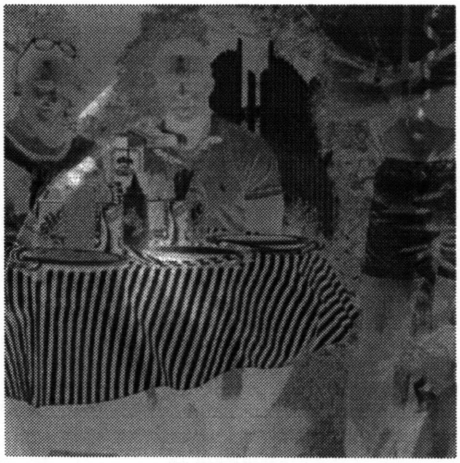
Figure 4.4 shows a block segmentation; this structure works well for DCT systems, since it is evidently either identical to, or a subsampled version of, the transform structure itself.

[1] This effect is similar to noise enhancement in linear equalizers for communications, and the remedy is the same: abandon linear processing.

(a) Frame 1

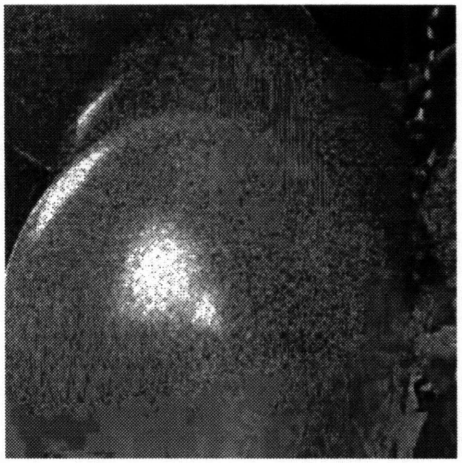


(b) Frame 2

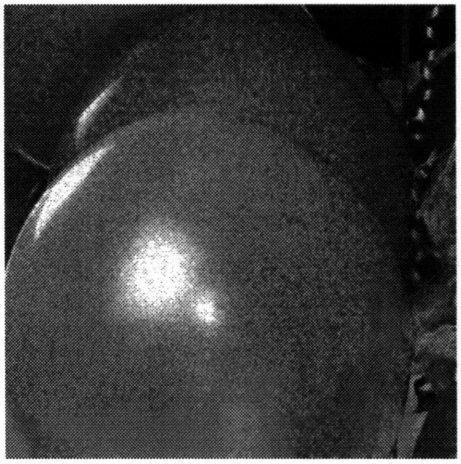


(c) Residual

Figure 4.2: Prediction error of scene cut



(a) Frame 1



(b) Frame 2

(c) Frame 3

Figure 4.3: Reconstructed scene cut

The switch in the block diagram is the prediction decision: if the switch is up, the residual will contain the current block; if the switch is down, the residual will contain the usual difference between the current block and the predicted block.

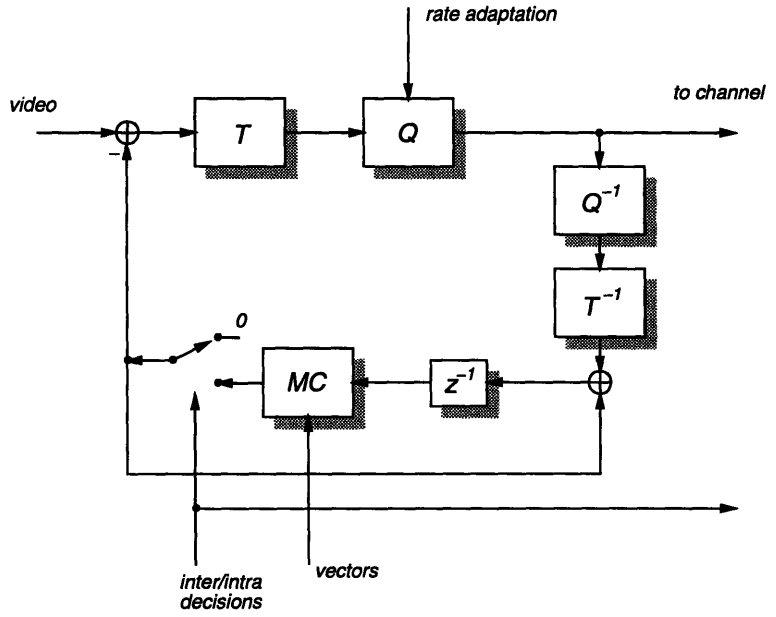


Figure 4.4: Adaptive prediction for block transform coder

A representative residual image under this scheme is shown in Figure 4.5. The prediction is kept for most of the image area, but the region containing the changing object is more efficiently intraframe-coded, that is, coded with the temporal prediction set to zero.

Naturally, the encoder must send side information specifying the prediction for each block. Here this amounts to (at most) one bit per block, that is, about 0.004 bits/sample for 16x16 blocks (assuming the decision is replicated for chroma, which is usually a good policy). This is about 1% of a typical coding rate, well worth the improvement it gives.

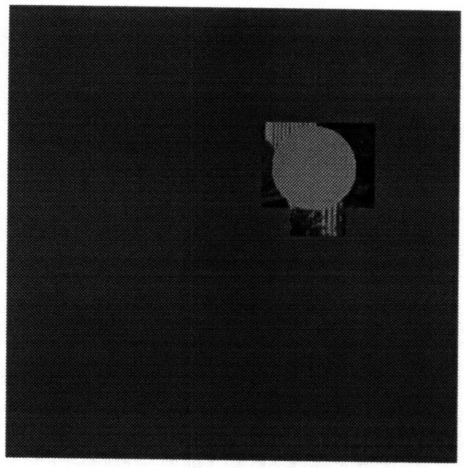
The prediction decision is made by a process that evaluates both options. Counting the bits required to code each block-type is the most fundamental measure, if it makes sense (some statistical coding techniques may yield bits associated with more than one block). Another measure, less sharp but more orthogonal to the rest of the system, is to compare the relative energies of the two errors. That is the choice used in the implementation described in chapter 5, with a few refinements for "better" choices: DC energy is ignored, and the remainder of the samples are weighted by spatial frequency.

4.3 Adaptive prediction for subband coders

A video coder based on subband coding or wavelets requires an extension of these techniques. First the deficiencies of block decisions will be described; then a structure based on collocated coefficient-blocks will be shown to remove nearly all of the difficulty.

(a) Frame 1

(b) Frame 2



(c) Residual

Figure 4.5: Spatially adaptive prediction error

4.3.1 Conventional segmentation

The block-prediction algorithm described above is not useful for subband coders. It can certainly be tried, and the result is interesting: when the prediction error of Figure 4.5, say, is filtered by the spatial part of the system, the transitions from interframe blocks to intraframe blocks interfere with the filtering. (For the DCT, these transitions scarcely exist, because they line up with the transform blocks and are invisible to the filters.)

This interference of the prediction-block edges results in much high-frequency energy, since each edge borders an approximately DC-free region (with prediction) and a DC-normal region (without). Moreover, the filtered transient will be coarsely quantized, and quantization errors near the block will be visible.

Figure 4.6 shows an example for which several blocks are intra-coded in this way; the distortion of the high frequencies can be clearly seen. Though the interiors of these blocks are simple, the errors resulting from filtering the edges are unacceptable.

Figure 4.6: Subband coder reconstructed frame, spatially adaptive prediction

One could try to repair the situation by removing the mean (or perhaps the median) from the intra-coded blocks before filtering, in an effort to remove all the transients. Such efforts are doomed. There will always be blocks that behave badly under whatever modeling is applied to clean up the edges. For example, mean-removal will be fooled by blocks with significant average gradient, and similarly a more ambitious affine-removal by blocks with significant curvature or edge.

What is needed is a way to smoothly change from inter-coding to intra-coding, and this is precisely what the filterbank itself can be made to provide.

4.3.2 Coefficient-domain segmentation

Figure 4.7 shows a prediction loop that has been rearranged to switch among blocks of filterbank coefficients. This structure involves an extra spatial filtering operation, since both the current frame and the predicted frame must be filtered, but it allows adaptive prediction unhindered by spatial concerns.

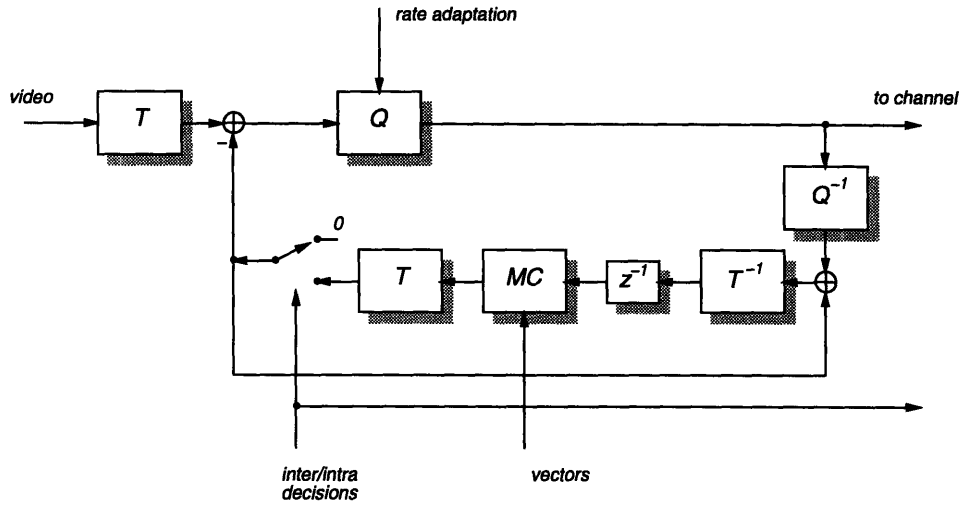


Figure 4.7: Adaptive prediction for subband coder

What is the effect of this modification? A walk through the block diagram will highlight the differences. First, incoming video is immediately filtered and organized into blocks of coefficients, in the way of all linear coders. Although there is no spatial-domain block structure in Figure 4.7, it is useful to imagine what spatial structure now corresponds to a prediction block. It is simply the collection of synthesis basis functions of each coefficient of the block; Figure 4.8 shows the 16x16 case, illustrating the overlap of the DC and lower-frequency kernels.

Next, a prediction coefficient-block is subtracted off. This block is constructed by the prediction loop: it will be zero for blocks with no useful prediction data, and it will be part of the filtered motion-compensated previous frame otherwise.

Now the coefficients are simply quantized and sent to the entropy coder, together with the side information specifying all of the prediction data. This is sufficient for the decoder to reverse all of these operations, and in fact the lower part of the block diagram is just the receiver processing.

The receiver processing is more complex than in the system in the previous section. It consists of two filtering operations, one to assemble the dequantized coefficients and transform to the spatial domain for motion compensation, and one to flip back to coefficients to serve the prediction for the next frame.

Ideally, neither would be necessary. Motion compensation is conceptually something that might operate on filter coefficients directly, as in Figure 4.9. But I'm not aware of any filterbanks or motion-compensation algorithms that can do this without incurring essentially the same complexity as Figure 4.8; the reason would seem to be the unavoidable aliasing in the filterbank, which cannot be space-invariant and so is incompatible with the motion compensation.

Finally, notice that for block transforms like the DCT, this system is equivalent to the one in Figure 4.4, since the "switch" operation and the transform T commute. It is only for subband coders that they interfere.

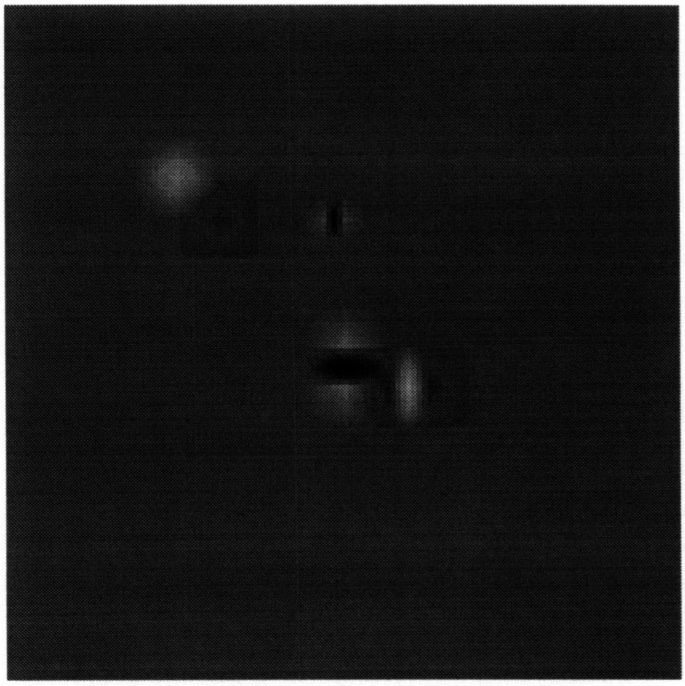


Figure 4.8: Spatial extent of prediction block

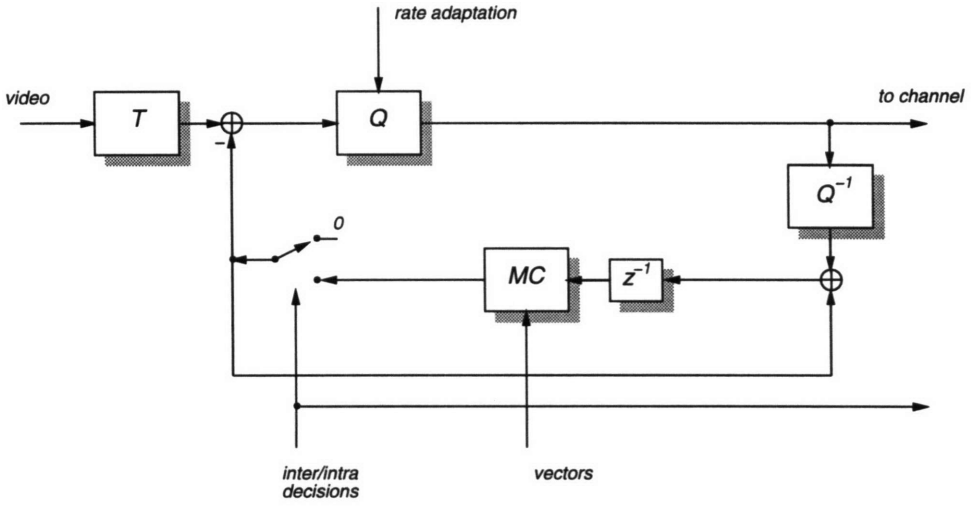


Figure 4.9: More efficient, but unworkable, receiver scheme

4.3.3 Tradeoffs

As we have seen, the wavelet adaptive prediction scheme requires three filters rather than two in the encoder (and two versus one in the decoder). However, filtering is only a small part of the computational burden of a video coder—the temporal signal processing places much greater demands on computation and memory systems.

One restriction the wavelet structure places on coder design is that all motion vectors must be sent even if some blocks are intra-coded. This is because the transform operation in the receiver loop requires (potentially) all samples of the motion-compensated previous frame. (The DCT case does not need any samples of an intra-coded block, so a vector need not be sent). This means extra overhead, but it is not too severe, especially as the vector field is itself differentially coded.

These issues must be balanced against the perceptual and statistical benefits of good filter-banks, which will be evaluated in the next chapter.

5

System aspects of video subband coders

Video coding, at the level of systems design, is an experimental enterprise. There is no substitute for a working system, not only for observing the output for quality, but for extracting the quantities that serve to characterize the system's efficiency and to train the quantizers and statistical coders.

This chapter will describe a coder that implements the style of adaptive prediction described in the previous chapter. The coder is designed around the infrastructure available at ATRP, namely a network of workstations, a video frame buffer, and motion imagery at 720x1280 resolution.

At the highest level, the coding system consists of an encoder process and a decoder process. The encoder reads a stream of frames and writes bits, and the decoder does the reverse; thus there is complete isolation between the two except for the transmitted data. The encoder optionally writes statistics and status to an auxiliary stream.

5.1 Motion estimation

The search range for motion vectors is taken to be 16 pixels, which is adequate for images with medium to high dynamics. The first-stage search uses integer vectors, to avoid interpolation, and the second stage refines the vector to 1/2 pixel, within a range of 2 pixels. The metric is mean absolute error.

Vector data is computed every 16x16 pixels, but 8x8 blocks are used for distortion calculations. If the (integer) vector that minimizes the entire 16x16 block yields significantly higher prediction energy than the individual 8x8 blocks, 8x8 vectors are used.

A software implementation of full-search motion estimation is expensive, and if the entire coder runs on a uniprocessor it will dominate the run time. It seems natural, then, to distribute the motion-estimation task among many machines.

Figure 5.1 shows the subtasks involved in computing the vector field. Each block requires not only its own image data, but also neighboring samples out to the maximum vector length. A large grain-size will minimize this overhead, but limit the possible speedup; a small grain will do the opposite. For the workstation-LAN environment, and for about 50 available hosts, the best grain size was 64x64, resulting in a speedup of about 20. A message-passing library called PVM, written at Oak Ridge, handled the low-level communication and synchronization

details.

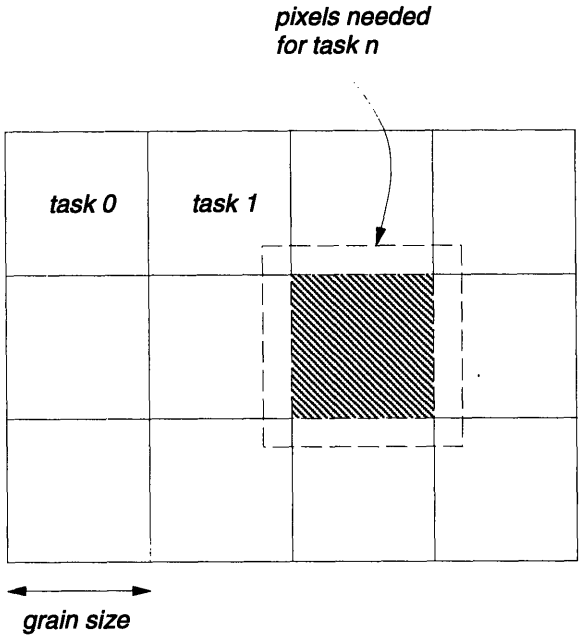


Figure 5.1: Parallelizing motion estimation

In principle, other coder tasks like filtering and quantization could be similarly distributed, but the communication dependencies become more complicated. Especially for filtering, keeping all data in the same address space simplifies bookkeeping.

5.2 Filtering

The coder uses an instance of a biorthogonal wavelet filterbank. There are a number of filter choices in the literature, often with parameters that generate entire families of filterbanks, so that some experimentation is necessary to get a feel for filter properties under quantization. One that gives especially good results visually is due to Barlaud, and is shown in Figures 5.2 and 5.3 in dimensions 1 and 2.

The implementation is a straightforward recursive direct polyphase FIR filter using three levels. Boundary conditions are enforced by symmetric or antisymmetric extensions at the image edges.

Filtering produces a collection of subband signals, but it is necessary to place the data in a coefficient-block context with all members of a block approximately colocated in space. For the usual subband-image organization, this turns out to be a multiscale shuffle operation (which turns out to be rather expensive, since the images are large and the index gymnastics generate constant cache misses).

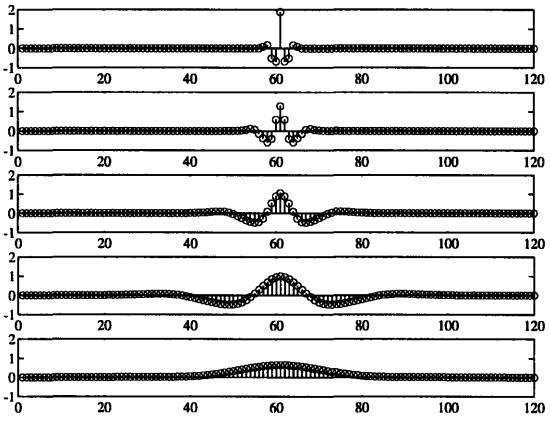


Figure 5.2: Wavelets and scaling function (synthesis)

5.3 Quantization

Representing video as a collection of subband coefficients does not by itself produce a sequence of bits, but rather a suitably sparse representation of the video. It remains for a quantization process to produce a sequence of symbols from some finite alphabet to represent the coefficients, and an entropy coding process to represent the symbols as bits.

The quantization step is the only irreversible operation in the entire coder, since the various representation mappings and the entropy coding are invertible (or "noiseless" in the signal-processing vocabulary, though for the filters there are always numerical issues). The only important distortion is quantization noise, although the nature of this noise of course depends crucially on the details of the representation.

The total coefficient space will have large dimension, and it is necessary to partition it in some way and apply quantizers to smaller pieces (though there have been some recent efforts to quantize the entire image directly while avoiding an intractable quantizer design problem [4, 8]). Assigning a sequence of bits to each point in one of these spaces so as to minimize the expected number of bits per point is called vector quantization (VQ). Scalar quantization consists of factoring the space into a product of one-dimensional terms, then applying (perhaps different) quantizers to each term.

While naive vector quantization is efficient, its complexity and training data grow exponentially in the dimension. The practical choice is therefore between one of the constrained VQ techniques [21] or scalar quantization with entropy coding. This coder uses scalar quantizers, in fact the uniform (boundless) mid-tread quantizer shown in Figure 5.4. Mid-tread is used to obtain a slight noise-coring effect and to avoid any oscillations around the nominal coefficient value.

While the form of the quantizer is the same for each coefficient, its scale sets the perceptual importance. The relative scales were chosen from still-image experiments, since the perceptual requirements for video ought to be very similar. The absolute scale is set by the rate control

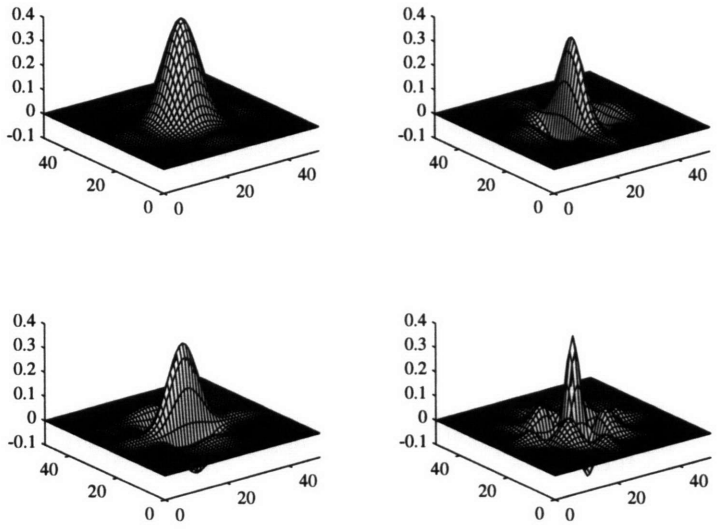


Figure 5.3: Mesh plots of tensor product wavelets (synthesis)

loop discussed in section 5.5.

For a system with entropy coding, quantizers that saturate are often a hindrance rather than a help—the real use of a hard limit is to make possible a symbol map without entropy coding.

5.4 Entropy coding

The result of quantization is a set of indices or symbols that the entropy coder will translate into a sequence of bits. In this case the symbols are 8x8 blocks of integers, each of which is a quantized coefficient block.

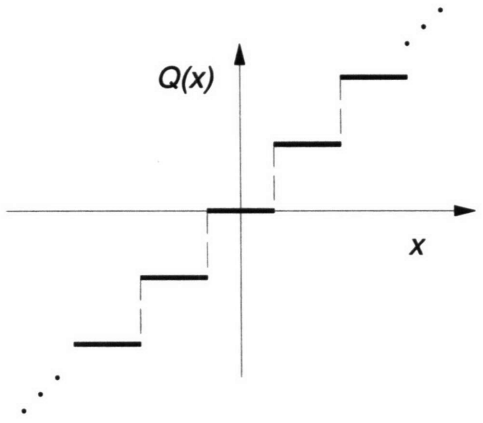


Figure 5.4: Uniform mid-tread quantizer

An off-line computation produces a Huffman codebook from suitable training data. (Arithmetic coding would have been another possibility, with potential advantages in flexible adaptation at very low cost.) The codebooks were trained with quantized coefficient data from the analysis of about 40 frames of video, taken from various test sequences to achieve a good mix.

5.4.1 Codebook structure

The coefficient codebook consists of a library of low-energy 4x4 symbols, medium-energy 2x2 symbols, symbols for individual coefficients, and an end-of-block symbol. Each 8x8 block is traversed from high frequency to low frequency, using the octave boundaries as guides for symbol selection. Figure 5.5 shows the structure of an 8x8 block. The DC coefficient is coded separately, using a differential Huffman code distributed over the macroblock (similar to the motion vectors).

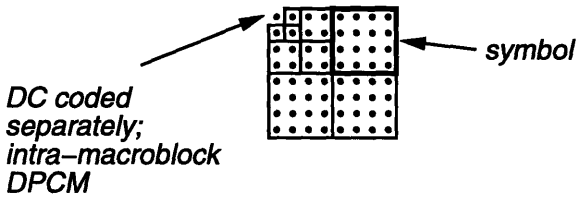


Figure 5.5: Codebook symbol

This illustrates one of the conveniences of octave-band subband representations. Since many of the kernels are identical except for translation, there is no reason to distinguish between them for training. By contrast, all 64 kernels in an 8x8 DCT are distinct and potentially need separate treatment.

5.4.2 Codebook search

Since the total codebook is fairly large, a search to discover the codeword for a particular symbol is time-consuming. A codebook hash table does not significantly increase the space requirements for the coder, and it implements the mapping in constant time. (Hashing is useful even for zig-zag scanning, since it is cheaper than the sequential algorithm.)

5.5 Rate control

Except in very simple situations, the rate at which bits are emitted from the entropy coder is unpredicable. In general, the finer the quantization, the more bits will be needed to represent the output, but this is not detailed enough a model to permit interfacing to synchronous channels that require an information source at a constant rate.

One way around this problem would be to dispense with constant-rate channels. Some communications technologies, for example packet-switched and spread-spectrum networks, are designed for asynchronous sources; a source coder could then transmit at whatever rate

it needed at the moment. The instantaneous rate is commonly based on a constant-quality criterion.

More usual for video, though, are channels with fixed rate. This situation is mostly historical, since current systems are point-to-point or broadcast analog links with constant bandwidth (though there is increasing use of digital source coders for contribution sources internal to networks). The natural evolution, therefore, is to take an analog channel, for example a 6 MHz terrestrial channel or a 36 MHz satellite transponder, and simply allocate to it some number of HDTV signals (together with a suitable digital communications system).

Physical-layer channels for video include terrestrial VHF and UHF broadcasting, geosynchronous satellite repeaters, coaxial cable, and fiber. In principle, there is no reason that any of these media could not be adapted from their current form into networks more suitable for coded signals. The recent ATM protocols are designed from the start with bursty, real-time sources in mind (as well as high rates, which is convenient for the video problem).

5.5.1 Rate adaptation for fixed-rate channels

Assuming that the interface is to a fixed-rate channel, the general scheme involves a buffer and a rate control algorithm designed to prevent the buffer from underflowing or overflowing (see Figure 5.6).

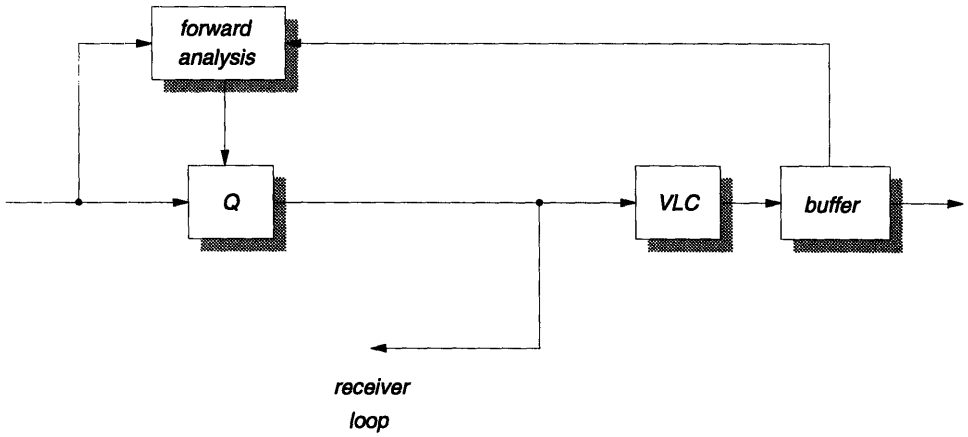


Figure 5.6: Buffer feedback for rate adaptation

The information required by the adaptation mechanism more or less determines its complexity. The two architectures most often encountered are a feedback algorithm, relying only on buffer fullness to set future quality or quantization levels, and a feedforward mechanism, which uses an abbreviated version of the encoder process to more accurately estimate the rate-distortion relation for the near future.

For a given buffer size, the feedforward algorithms give better performance at the cost of higher complexity (and perhaps delay for the forward path). For this coder, since overall delay is not an important issue, the buffer is made large enough to allow the use of simpler feedback algorithms. Figure 5.7 shows the precise scheme.

Note that the overall loop is first-order, the usual choice for this kind of system: there is

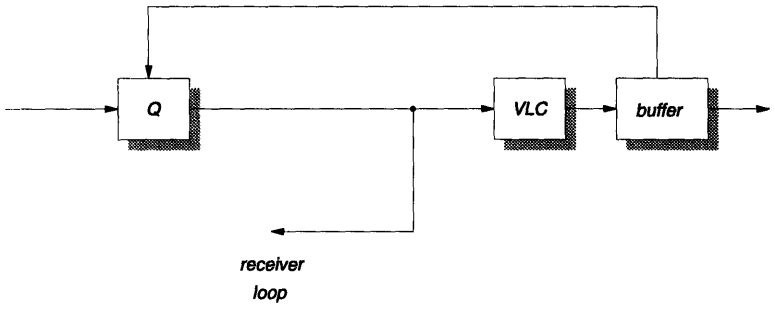


Figure 5.7: Feedback quantizer adaptation

no state variable corresponding to a filtered rate (in contrast to, say, a PLL, for which this is usually essential). The adaptation data and buffer fullness may then fluctuate rapidly, but this has no direct bearing on image quality.

To illustrate a typical buffer response, Figure 5.8 shows buffer fullness over the course of a 60-frame sequence. The cut at 30 frames is clearly visible, as are overall trends in scene information.

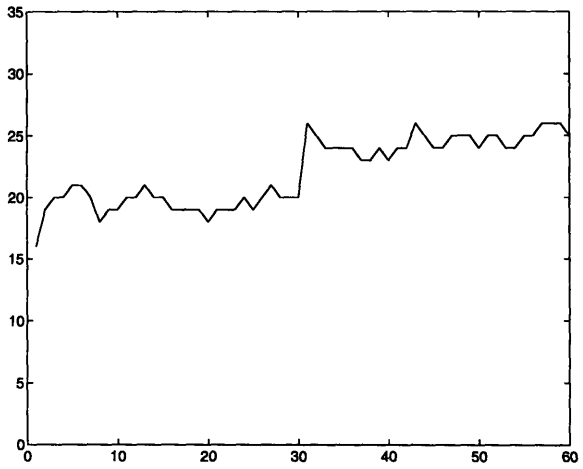


Figure 5.8: Buffer fullness history

5.6 Acquisition and error concealment

As mentioned in chapter 2, acquisition is an essential part of temporal processing. In a broadcast mode receivers need to synchronize with a running transmission, so signals must provide sufficient redundancy to allow acquision on timescales of a second or so, limited on the high side by user impatience (usually thought of in the channel-flipping mode, though perhaps HDTV will provide a better search paradigm!) and on the low side by the data overhead.

There are two styles of refresh, progressive and leak. Progressive refresh reserves a certain (variable) image region exclusively for intraframe coding (in addition to the sources discussed in chapter 4) so every pixel is eventually covered. In contrast, leak schemes, instead of refreshing

some of the picture all at once, refresh all of the picture a little at a time. This is done by including in the prediction error a small fraction of the current frame, or equivalently by scaling the prediction appropriately. Figure 5.9 shows block diagrams for the two methods.

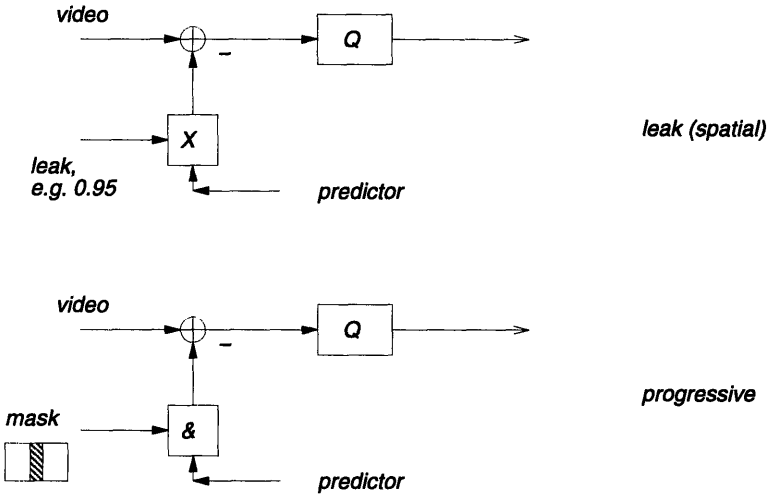


Figure 5.9: Acquisition

Each of these algorithms can be refined into a number of alternatives. For progressive refresh, one has the choice of allocating a certain number of intra-blocks for every frame (usually in a swept or wiped pattern from top to bottom or left to right), or else dedicating an entire frame to intra-coded blocks (a so-called I-frame) every so often. The wipe allows acquisition to become visible sooner, but it suffers from a minor glitch: rapid motion in the same direction and speed as the wipe can cause the effective acquisition time to increase, since the motion compensator will propagate uninitialized data until it reaches the image edge.

Similarly, leak can operate either spatially or in spatial frequency. An interesting variation, possible only in the latter mode, is to adjust the integrator gains (and so the acquisition time constants) according to frequency. The low frequencies should be configured for rapid access, on the order of a few frames, while high frequencies can be given a longer time, perhaps 10 frames. Since the human visual system is remarkably insensitive to detail for as much as a few hundred milliseconds after a scene cut, the net result is increased efficiency for the same subjective acquisition performance.

Finally, refreshing can have useful interactions with other system issues. General Instrument has used a clever synthesis of progressive refreshing and partitioning to allow all motion estimation to occur within panels [12].

Error concealment is closely related to acquisition, since incorrect video resulting from data errors would result in catastrophic error propagation around the receiver loop were it not repaired. (Indeed, the lack of any signal at receiver startup can be regarded as a single large error.) Error handling needs first a strong error detection mechanism, then a good choice of substitute data until the error is refreshed.

Error detection is so inexpensive that very strong schemes are affordable. The coder uses a 32-bit CRC every macroblock; this is a natural granularity, since macroblocks are the boundaries of differentially coded DC and motion vectors. The resulting overhead is negligible. The

consequences of an undetected error would be a trashed macroblock, essentially random video from decoded random bits, moving about on the screen. This is much worse perceptually than a disguised error.

There does not seem to be any error concealment scheme much better than the natural one, namely setting the errored macroblock to zero and allowing the receiver loop to substitute the motion-compensated data from the previous frame (assuming the vectors are available). This not only provides plausible video, it allows updates from future frames to continue to provide estimates until the region is refreshed.

Here error lifetimes are the same as acquisition times, though ideally the errors should probably be shorter-lived.

5.7 Source-adaptive processing

Television systems need to support a wide variety of image sequence types in addition to their native or default mode. Motion picture film, for example, uses a wide variety of aspect ratios and frame rates, and HDTV will also need to accommodate analog video from tape libraries or for simulcasting. The NTSC solution to this problem is to require a transconversion from every source format to the 480-line interlaced, 60 Hz field rate, 4:3 aspect ratio specification.

That entails unavoidable quality tradeoffs. Temporal rate conversion is perhaps the most difficult problem; even today, 50 Hz to 60 Hz is far from perfect, and 24 Hz to 60 Hz suffers from some judder and window-blinding (in the case of interlace). Aspect ratio conversion can always be handled with letterboxing, but this wastes bandwidth in an analog system, and it also leaves part of the display unused. The alternative is pan-and-scan, which discards a part of the image. Purely spatial conversions are less difficult, though of course the transmission format is a bottleneck for high quality sources.

Digital television systems can avoid many of these problems by allowing several image formats to be sent. Provided receivers are flexible enough to render all of the formats acceptably, source-adaptive coding can trivially exploit differences in the source's spatial resolution, frame rate, and possibly color space. For example, a separate format for film at 24 Hz will allow the available data rate to represent finer spatial detail, since the temporal bandwidth is only 2/5 that of a 60 Hz signal.

5.8 Auxiliary services and data

Practical HDTV systems include data types other than video (which nevertheless usually dominates in terms of rate). Audio, control and sync, and header/descriptor information are usually sent with all video, and various other data, such as text and graphics, may be needed.

Much progress has been made in audio source coding over the past eight years or so. Essentially perfect audio, statistically indistinguishable from original material by expert listeners, can be attained at rates around 2.5 bits/sample for a monophonic channel, and only somewhat higher for multichannel, surround-sound modes [26]. Multirate representations for audio are also productive, but for different perceptual reasons.

Video can have a number of formats, as described in the previous section. It falls to header/descriptor data to describe these formats (but not necessarily the decoder algorithm or display algorithm—that is up to the receiver). It is a nontrivial, cultural decision as to how flexible to make descriptor data structures—too general and the implementation is costly or ad-hoc; too restrictive and desirable future enhancements cannot be supported. The coder presented here avoids all these issues, but large designs cannot be so cavalier.

5.9 Comparison of subband and DCT representations

While rates on the order of 0.3 bits/pixel are needed to give high perceptual quality, this is inconvenient for tuning and characterization of artifacts, since they can be difficult to see at this rate.

Figure 5.10 shows a representative still frame from sequences coded with the wavelet coder and a comparable DCT coder. (One of the reasons for choosing elementary 8x8 coefficient blocks was to facilitate this comparison—the coder need only "swap in" a different transform and different quantizer tables and coders.)

Both images have significant distortions at this rate. In the DCT coder some image areas are rendered as blocks (corresponding to transform blocks) with sharp edges, whereas in the wavelet coder the detail becomes blurry. The DCT block edges can in fact give the impression of a greater subjective sense of sharpness, even if the sharpness is unrelated to image high-frequency information.

At close viewing distances, though, when the 8x8 blocks are well resolved, the wavelet image seems less objectionable. It is difficult to make a quantitative assessment between the perceptual improvement from smooth kernels and increased statistical efficiency due to the localization, but both probably contribute.

Figure 5.10: DCT and subband artifacts

6

Conclusions

This thesis investigates wavelet representations for video in a motion-compensated subband coding context. The conclusion is that subband or wavelet spatial coders for video offer somewhat better perceived quality than block transform coders, especially at low rates, at the cost of some receiver complexity and coding constraints.

6.1 Future representation issues

One important degree of freedom this coder does not support is locally adapting the filters themselves to the signal, rather than relying on a single filterbank to uniformly represent the entire signal. Such "best-basis" algorithms [44] offer very natural rate-distortion-guided schemes for kernel choice, and side information requirements seem to be acceptable.

Subband coders are always constrained by their low-level signal model. As spatial and temporal resolutions and available computation increase, a gradual migration to models encompassing more than one frame or more than a few pixels will be profitable.

References

[1] National Television System Committee. *Proceedings IRE*, 1954.

[2] E. H. Adelson, E. Simoncelli, and R. Hingorani. Orthogonal pyramid transforms for image coding. In *Proceedings of SPIE*, volume 845, pages 50–58, Cambridge, MA, October 1987.

[3] Ali N. Akansu and Richard A. Haddad. *Multiresolution Signal Decomposition.* Academic Press, 1992.

[4] M. Antonini, M. Barlaud, and P. Mathieu. Image coding using lattice vector quantization of wavelet coefficients. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2273–2276, 1991.

[5] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. Image coding using vector quantization in the wavelet transform domain. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2297–2300, 1990.

[6] AT&T. *High-Definition Television System*, August 1991.

[7] AT&T. The argument for progressive scan in a U.S. HDTV transmission standard. FCC ACATS Working Brief, January 1993.

[8] Michel Barlaud, Patrick Sole, Marc Antonini, and Pierre Mathieu. A pyramidal scheme for lattice vector quantization of wavelet transform coefficients applied to image coding. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages IV–401–IV–404, 1992.

[9] P. J. Burt and E. H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Communications*, COM-31(4):532–540, April 1983.

[10] W. Chen and W. K. Pratt. Scene adaptive coder. *IEEE Transactions on Communications*, COM-32(3):225–232, March 1984.

[11] Charles K. Chui. *An Introduction to Wavelets.* Academic Press, 1992.

[12] General Instrument Corporation. *DigiCipher HDTV System Description.* G.I., August 1991.

[13] Ronald E. Crochiere and Lawrence R. Rabiner. *Multirate Digital Signal Processing.* Prentice-Hall, 1983.

[14] A. Croisier, D. Esteban, and C. Galand. Perfect channel splitting by use of interpolation/decimation/tree decomposition techniques. In *International Conference on Information Sciences and Systems*, pages 443–446, Patras, August 1976.

[15] Ingrid Daubechies. Orthonormal bases of compactly supported wavelets. 1988.

[16] Ingrid Daubechies. *Ten Lectures on Wavelets.* CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, 1992.

[17] D. E. Dudgeon and R. M. Mersereau. *Multidimensional Digital Signal Processing.* Prentice-Hall, 1984.

[18] D. Esteban and C. Galand. Application of quadrature mirror filters to split band voice coding schemes. In *Proceedings ICASSP*, pages 191–195, 1977.

[19] A. Fernandez, R. Ansari, D. J. Gall, and C. T. Chen. HDTV subband/DCT coding: Analysis of system complexity. In *IEEE Globecom Proceedings*, pages 343.1.1 - 343.1.4, 1990.

[20] D. Gabor. Theory of communication. *J. Inst. Elec. Eng.*, 93:429–441, 1946.

[21] A. Gersho and R. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, 1992.

[22] H. Gharavi and A. Tabatabai. Sub-band coding of digital images using two-dimensional quadrature mirror filtering. In *Proceedings of SPIE*, volume 707, pages 51–61, 1986.

[23] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, 1983.

[24] Hsieh S. Hou and Harry C. Andrews. Cubic splines for image interpolation and digital filtering. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 26:508–517, December 1978.

[25] N. Jayant and P. Noll. *Digital Coding of Waveforms.* 1984.

[26] Nikil Jayant, James Johnston, and Robert Safranek. Signal compression based on models of human perception. *Proceedings of the IEEE*, pages 1385–1422, 1993.

[27] Jae S. Lim. *Two-Dimensional Signal and Image Processing.* Prentice-Hall, 1990.

[28] H. S. Malvar and D. H. Staelin. The LOT: Transform coding without blocking effects. *IEEE Trans. ASSP*, ASSP-37(4):553–559, April 1989.

[29] Henrique S. Malvar. Fast computation of wavelet transforms with the extended lapped transform. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages IV-393–IV-396, 1992.

[30] Henrique S. Malvar. *Signal Processing with Lapped Transforms.* Artech House, 1992.

[31] Arun N. Netravali and Barry G. Haskell. *Digital Pictures: Representation and Compression.* Plenum Press, New York, NY, 1988.

[32] Mutsumi Ohta and Satoshi Nogaki. Hybrid picture coding with wavelet transform and overlapped motion-compensated interframe prediction coding. *IEEE Transactions on Signal Processing*, pages 3416–3424, 1993.

[33] A. V. Oppenheim and R. W. Schafer. *Discrete-Time Signal Processing.* Prentice-Hall, 1989.

[34] W. K. Pratt. *Digital Image Processing.* John Wiley & Sons, Inc., 1992.

[35] K. R. Rao and P. Yip. *Discrete Cosine Transform.* Academic Press, 1990.

[36] William F. Schreiber and Donald E. Troxel. Transformation between continuous and discrete representations of images: A perceptual approach. pages 178–186, 1985.

[37] E. Simoncelli and E. H. Adelson. Non-separable extensions of quadrature mirror filters to multiple dimensions. *Proceedings of the IEEE: Special Issue on Multi-dimensional Signal Processing*, April 1990.

[38] Mark J. T. Smith and Steven L. Eddins. Analysis/synthesis techniques for subband image coding. *IEEE Transactions on Acoustics, Speech and Signal Processing*, pages 1446–1456, 1990.

[39] Gilbert Strang. Wavelets and dilation equations: A brief introduction. *SIAM Review*, 31(4):614–627, December 1989.

[40] Robert Ulichney. *Digital Halftoning.* The MIT Press, 1987.

[41] P. P. Vaidyanathan. *Multirate Systems and Filterbanks.* Prentice-Hall, 1993.

[42] M. Vetterli. Multi-dimensional sub-band coding: Some theory and algorithms. *Signal Processing*, 6(2):97–112, February 1984.

[43] Martin Vetterli and Dimitris Anastassiou. A multiresolution approach for all-digital HDTV. In *Globecom*, pages 320.2.1–320.2.5, 1990.

[44] M. W. Wickerhauser. Acoustic signal compression with wavelet packets. pages 679–700, 1992.

[45] Kwo-Jyr Wong and C.-C. Jay Kuo. Image compression with full wavelet transform (FWT). USC-SPIP report 226, University of Southern California, 1992.

[46] J. W. Woods, editor. *Subband Image Coding.* Kluwer Academic Publishers, Norwell, MA, 1990.