Chapter 1

Introduction

1.1 Overcomplete Signal Representation

Consider an artist who begins to paint a natural scene using a trusted paint brush and a palette consisting only of red, green, and blue paints. This limited palette does not in theory restrict what the artist may paint, since any desired color may be produced by some mixture of the three available primary colors [11]. However, such a restriction certainly affects the amount of effort required by the artist. He or she will likely be frustrated by the need to constantly combine paints on the canvas every time a non-primary color is required. The artist would be much better off with a rich palette of many hues, since this would allow for less time mixing colors and more time concentrating on the painting.

The same idea also applies to signal representation. Consider the classic problem of representing a signal $\vec{f}$ as a weighted combination of basis vectors $\vec{g}$ taken from some dictionary G. The painter's palette in our above example is analogous to G, with each paint color a single dictionary element $\vec{g} \in G$. The artist is then an algorithm which combines the available dictionary functions to form a suitable representation for $\vec{f}$:

$$\vec{f} = \sum_{\gamma \in \Gamma} p_\gamma \, \vec{g}_\gamma \tag{1.1}$$

It is possible to design G to contain the minimum number of elements such that any signal $\vec{f}$ may be exactly represented in the form of Equation (1.1). This makes G a complete basis for the signal space in which $\vec{f}$ lives, and is analogous to choosing a palette of the three primary colors. However, the artist analogy makes us wonder whether it might be better to design G to be overcomplete, containing more functions than strictly necessary to represent the signal space. Does a richer, more diverse dictionary improve the representation of $\vec{f}$?

The answer to this question depends greatly on what the resulting representation of $\vec{f}$ is to be used for. In this thesis, we assume a compact coding application in which the representation is generated in order to allow $\vec{f}$ to be stored or transmitted over a channel. A lossy representation of $\vec{f}$ is typically desired in order to reduce the bit rate. To achieve this, the coded representation for a specific $\vec{f}$ may use a subset of the full dictionary G. Also, the weighting coeficients p_γ are quantized using some quantizer function $Q(\cdot)$. The result is a coded approximation of the following form:

$$\vec{f} \approx \vec{f}' = \sum_{\gamma \in \Gamma'} Q(p_\gamma) \, \vec{g}_\gamma \tag{1.2}$$

The motivation for an overcomplete dictionary comes from the need for a sparse representation of $\vec{f}$. Just as a rich color palette reduces the need for color mixing, a rich dictionary reduces the number of functions $\vec{g}_\gamma$ which must be combined in order to approximate $\vec{f}$ to a given quality. An overcomplete dictionary thus allows a suitable representation to be sent with fewer encoded parameters. This is important for applications in which a low bit rate is required.

For visual applications such as image and video coding, an overcomplete dictionary may potentially offer subjective quality advantages as well. When visual material is encoded at extremely low rates, few dictionary functions may be coded, and the coeficient of each function must be coarsely quantized. In this case, it may be possible to see the structure of the dictionary in the coded material. For example, when images are encoded to low bit rates using the block-based discrete cosine transform (DCT), it is often possible to see block artifacts in the coded image. An overcomplete dictionary may reduce this problem, since such a dictionary is more likely to contain functions which are well-matched to the local signal structure.

To put this into the context of our earlier example, imagine that the artist is restricted to using only a small number of strokes to complete the painting. He or she thus cannot afford to mix colors sufficiently on the canvas, and the palette colors will predominate in the resulting painting. If the palette is restricted to primary colors red, green and blue, the artist will find it difficult to paint an orange and yellow sunset within the limited number of strokes. By increasing the palette to include a wider variety of colors, the problem may be avoided.

Once the decision is made to use an overcomplete dictionary for a signal coding application, the central problem is how to generate an efficient representation of $\vec{f}$ on dictionary G using a reasonable amount of computation. To solve this problem, we turn to a technique called matching pursuit.

1.2 Matching Pursuit and Applications

Matching pursuit is an iterative approach for decomposing a signal on an arbitrary dictionary which is usually chosen to be overcomplete. The method employs a greedy optimization strategy, choosing at each iteration the combination of dictionary function and weight which will provide the greatest reduction in error for the emerging signal representation. Other methods such as basis pursuit [9] may provide a globally optimal solution, but the associated optimization is extremely complex. Matching pursuit thus provides a good compromise between the efficiency of the representation and the associated computational cost [14].

Matching pursuit was first introduced in a paper by Mallat and Zhang [41], but was related to earlier works in statistics [21] and multistage vector quantization [68]. In the original paper, Mallat and Zhang applied matching pursuit to 1-D signal analysis, but also alluded to other possible applications such as pattern extraction and compact coding. This early prediction gave way to an explosion of papers. In recent years, matching pursuit has been applied to a wide variety of topics including pattern recognition [51], speech coding [56], and even industrial machining [22]. There is also a growing body of medical applications, covering both signal analysis [2], [71] and medical imaging [58].

The focus of our current work is the application of matching pursuit to video compression. To this end, we adopt a widely used hybrid structure which encodes each video frame using two stages. The first stage is motion compensation, which forms a prediction of the current frame as a modified version of the most recently decoded previous frame. Parameters known as "motion vectors" are sent to the decoder in order to represent the motion between frames and thus improve the prediction. The second stage is residual encoding, which codes the motion prediction error. Standardized video compression algorithms use an 8×8 block discrete cosine transform (DCT) to do the residual coding. In our current work, we replace this residual coding method with a matching pursuit based approach.

It is interesting to note that the first matching pursuit video coding papers [46], [67] actually pre-dated the first papers on matching pursuit speech coding [56] and image coding [7]. One might naturally ask why video coding, which deals with the complexity of 3-D signals, was among the first applications. One reason is that matching pursuit is particularly well suited to coding sparse signals, of which the motion residual error images produced by hybrid video coders are a good example. The assumption of a sparse input signal allows a significant reduction in matching pursuit encoding complexity, as will be demonstrated in Chapter 2. Overcomplete dictionaries may also be designed to be spatially invariant, meaning that each function in the dictionary may exist anywhere. This has proven to be useful for encoding residual signals, since a primary component of such signals are moving edges which do not align to arbitrary grid boundaries or subsampled image locations.

In this thesis, we use a spatially invariant, overcomplete dictionary containing functions at a wide variety of scales and frequencies. The resulting dictionary does not impose any structure on the coded residual signals, and systematic artifacts such as blocking and ringing are thus avoided [45]. This is particularly important for video, since human visual sensitivity makes such artifacts more noticeable in a moving image.

We first introduced matching pursuit video coding in [46]. An independently developed video codec was published shortly afterward by Vetterli and Kalker [67]. Their work was particularly interesting because it employed an adaptive dictionary which included both standard block DCT basis functions and candidate motion blocks from the previous decoded frame. In this way, matching pursuit was used to unite the motion compensation and residual coding parts of a hybrid encoder. A similar system was presented later by Gharavi-Alkhansari and Huang [24], with the dictionary modified to also include reduced-scale copies of previous frame blocks.

Our codec differs from those of [67] and [24] in two fundamental ways. First, our codec is not block-based, and so allows each dictionary shape to exist anywhere in the representation. Second, our codec uses a more traditional hybrid structure. Standard block motion compensation is applied, and the residual images are coded and sent using matching pursuit. The most complete published description of this system was given in [45]. An updated and improved description will be provided in Chapter 2.

Several papers have introduced improvements and additions to our basic algorithm. The first of these was due to Banham and Brailean [5], who proposed a method for adaptively switching between matching pursuit and DCT-based coding depending on the characteristics of the residual frame. In another paper by Al-Shaykh, et.al. [3], the codec functionality was expanded to handle both quality scalability and the coding of arbitrarily shaped objects. A heuristic "find energy" technique was also shown to improve the coding efficiency of the system. More recently, several authors have offered improved dictionary designs for the system. The resulting papers may be classified according to whether the dictionary was designed for reduced computational complexity [55], [15], [12], or for improved coding efficiency [10]. A more detailed look at these papers will be given in Chapter 6.

Finally, we note that our development of the matching pursuit video codec was strongly influenced by the MPEG-4 standardization process, in which we participated from November of 1995 through July of 1998. The core experiment process allowed us to closely compare the algorithm to the emerging MPEG-4 visual standard, and also to a variety of other experimental codecs. We received much useful feedback from the other participants, which helped us to develop improvements for the algorithm. The influence of standards participation can even be seen in this work, since we adopt the MPEG-4 coding efficiency test conditions for the experimental comparisons shown throughout the thesis.

1.3 Outline of Thesis

We begin in Chapter 2 with a description of the matching pursuit video encoder. This system extends the original method of Mallat and Zhang [41] to 2-D signals, and applies the resulting algorithm to encode motion compensated residuals in a hybrid video codec. We review the details of the encoder, and present new discussions on rate control and spatial bit allocation. We then compare the matching pursuit system to both H.263+ and MPEG-4, and show that matching pursuit provides gains over both of these systems in terms of peak signal-to-noise ratio (PSNR) and encoded visual quality.

Chapter 3 focuses on decoder implementation issues. Matching pursuit decoding is shown to be comparable in complexity to a DCT-based MPEG-4 decoder in the absence of postprocessing. However, a visual comparison shows that the MPEG-4 result contains objectionable block artifacts at low bit rates. Such artifacts can be reduced through postfiltering, as we demonstrate using the deblocking and deringing filters included in the MPEG-4 verification model. However, the complexity of the filters is shown to be prohibitively large, and the end visual result lacks detail in some areas. The matching pursuit result requires no such postfiltering, and as such produces detailed images which are free from systematic artifacts. Matching pursuit is thus shown to outperform MPEG-4 in terms of the tradeoff between decoder complexity and coding efficiency. As a footnote to this chapter, we introduce an all-integer implementation of the matching pursuit codec, and we discuss some of the the decoder implications.

The basic matching pursuit algorithm discussed in Chapters 2 and 3 relies on a fixed quantizer together with rate control based entirely on changing the number of coded dictionary functions, or atoms. Unfortunately, this strategy causes problems when the encoder is used at higher bit rates. For this reason, Chapter 4 explores adaptive quantizer designs for the matching pursuit codec. Specifically, we develop a rate control strategy based on a combination of the number of atoms and the quantizer design. We select these jointly in order to minimize the distortion of the current coded frame for a given bit rate. An optimal 2-Pass design and a compromise 1-Pass design are developed and shown to improve the coding efficiency of the system, particularly at high bit rates where the fixed quantizer is unsuitable.

Chapter 5 explores techniques for modeling the rate-distortion (R, D) performance of the matching pursuit codec. Two such models are developed. The first model is based on performing a pre-coding analysis pass on the current frame. That is, the current frame is encoded using matching pursuit with the full dictionary. The model then uses the results of this initial pass to predict changes in performance due to changes in the modulus quantizer design and the number of coded atoms. The second model is based on a reduced complexity pre-coding. This model predicts the performance of the full matching pursuit dictionary based on an encoding pass using a less expensive reduced dictionary. We evaluate the two models for use within existing rate-distortion optimization techniques. Our results suggest that both model classes have sufficient prediction accuracy to be useful for optimization, and that significant complexity reduction can be achieved compared to exact (R, D) computation.

Finally, Chapter 6 introduces a novel technique which generates reduced-complexity approximate implementations of an arbitrary initial dictionary. We refer to this method as dictionary approximation. Interestingly, it uses the matching pursuit algorithm itself to help design an efficient implementation of the matching pursuit video codec. Dictionary approximation offers several advantages over previous techniques. First, it allows greater freedom to design the dictionary, since the initial design need not be restricted to a computationally efficient structure. Second, by varying the quality of the approximated dictionary, a continuous tradeoff is produced between the computational complexity and coding efficiency of the resulting matching pursuit encoder. This relation may be systematically explored in order to select the desired balance between complexity and coding efficiency.

1.4 Preliminaries

We have tried to maintain clear and consistent notation throughout this work. When this does not appear to be the case, the reader is encouraged to refer to Table 1.1, which is provided as a notational aid. Also note that vectors such as $\vec{v}$ are generally defined

Table 1.1: Summary of Notation

Symbol	Description	Defined
i	Current atom stage	Section 2.2
k	Current frame	Section 4.2.1
p_i	Modulus value, stage i	Section 2.2
d_i	Quantization error, stage i	Section 4.1.1
$\vec{f}$	Original signal	Section 2.2
$\vec{f}_i$	Remaining signal, stage i	Section 2.2
(r, d)	Quantizer rate, distortion	Section 4.2.2
(R, D)	Overall rate, distortion	Section 4.2.2
$Q(\cdot)$	Quantizer	Section 4.1.1
N	Total atoms (stop-point)	Section 2.2
DZ	Deadzone size	Section 4.2.1
QP	Quantizer stepsize	Section 4.2.1
$\overline{ER}_i$	Energy removed by atom i	Equation (4.2)
$\overline{CR}_i$	Energy removed by first i atoms	Equation (5.6)
S	Elementary dictionary	Section 6.2.3
T_0	Original target dictionary	Section 6.3.1
T_q	Approximated target dictionary	Section 6.3.1

on a Hilbert space, but may be considered for our purpose to exist in 2-D image space. The norm $||\cdot||$ will generally refer to the 2-norm. As noted earlier, the experimental comparisons presented in this work have been modeled after the test conditions used by the MPEG-4 coding efficiency group during the development of the MPEG-4 visual standard. These test sequences and conditions are summarized in Table 1.2.

Table 1.2: Standard MPEG-4 Test Sequences

Seq	Label	Full Name	Kbit/s	Frame/s	Size
1	Cont-10	Container Ship	10	7.5	QCIF
2	Hall-10	Hall and Monitor	10	7.5	QCIF
3	Mom-10	Mother-Daughter	10	7.5	QCIF
4	Cont-24	Container Ship	24	10	QCIF
5	Silent-24	Silent Voice	24	10	QCIF
6	Mom-24	Mother-Daughter	24	10	QCIF
7	Coast-48	Coast Guard	48	10	QCIF
8	Fore-48	Foreman	48	10	QCIF
9	News-48	News	48	7.5	CIF
10	Coast-112	Coast Guard	112	15	CIF
11	Fore-112	Foreman	112	15	CIF
12	News-112	News	112	15	CIF
13	Mob-1024	Mobile-Calendar	1024	30	SIF
14	Stef-1024	Stefan	1024	30	SIF