REGION-ORIENTED VIDEO CODING USING QUADTREE OPTIMIZATION AND THE MDL PRINCIPLE

By

Paul C. Wareham

A thesis submitted to the

Department of Electrical and Computer Engineering

in conformity with the requirements

for the degree of Master of Science (Engineering)

Queen's University

Kingston, Ontario, Canada

August, 2002

Abstract

A novel approach is presented to find a sub-optimal image segmentation for low bit-rate video coding using a Minimum Description Length (MDL) cost function. The theory necessary to derive an appropriate cost function is described based on probabilistic models for the image prediction error and motion parameters. A key feature of this approach is the use of an MDL cost function to incorporate bit-rate minimization into the criterion for region segmentation. We apply this theory to the design of a bit-stream level interframe motion-compensated video coding system using a quad-tree optimization procedure with variable order motion modeling. A computationally feasible approach is illustrated which could potentially be implemented in real-time with appropriate hard-ware.

Acknowledgements

I would like extend my sincere thanks to my supervisor Dr. Steven Blostein for his support, research advice, and financial support. Dr. Blostein's superb guidance and endless patience were critical in completing this work and I am forever grateful for the opportunities he provided me.

I wish to thank my lab-mates at IPCL for their support through difficult times. I have made plenty of dear friends and leave with a host of fond memories of my time at IPCL.

I would also like to thank my parents for their love and nurturing over the years. It would not have been possible to complete this work without their encouragement.

Finally, much appreciated financial support for this work was provided by the Telecommunications Research Institute of Ontario (TRIO).

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

With video being a ubiquitous part of modern multimedia communications, efficient video coding remains an important area of research. The proliferation of personal computers and communication devices combined with the evolution of digital networks makes the integration of video with audio and data over the information superhighway a natural extension. Where the telephone and the cellular telephone have revolutionized voice communications, the incorporation of video in the communications sector is another milestone in technological history. Examples include the videophone, videoconferencing, multimedia, security monitoring, video on CD, digital video disk, video-on-demand, video mail, and high definition television.

When compared to audio or text information, video signals demand a huge amount of information. The adoption of modern compression standards such as JPEG, MPEG, H.26X demonstrates the need for standardization of the video coding techniques. The H.263 standard allows the transmission of real-time video using conventional telephone lines (POTS) with acceptable quality. The H.263 standard represents a significant step in

making the videophone a reality, while making use of existing telecommunications infrastructure.

The region-oriented approach has earned significant attention recently by researchers in the area of low bit-rate and very low bit-rate video coding. The adoption of these techniques can translate to lower overall bit-rate consumption while maintaining perceptual quality. This is increasingly important as the demand for robust low-rate video streams continues to grow, particularly for wireless mobile and satellite applications. First generation video coding standards such as ITU-T H.261, H.263, and MPEG suffer from overly simplistic region segmentation and motion modeling. The fixed block-type region boundaries do not take into account the actual content of the scene. Additionally, these image blocks are assumed to undergo independent uniform translation.

Region-oriented techniques are based on modeling images as a combination of non-stationary visual primitives, such as contours, textures, and motion. The advantage of these methods compared to object-oriented and 3-D model-based methods is that they do not require a priori information of the image content. Also, they are characterized by lower computational complexity which is a critical factor for low power implementation.

The following thesis presents a novel approach to find a sub-optimal image segmentation for video coding using a Minimum Description Length (MDL) cost function. A design is presented for a bit-stream level interframe motion-compensated video coding system using a quad-tree optimization procedure with variable order motion modeling. The codec incorporates bit-rate optimization into the criteria for region segmentation. The primary objective of this work is the presentation of practical alternatives leading to

improvements over existing video coding standards at very low bits rates, particularly under 30 kbits/s.

1.2 Summary of Contributions

The contributions contained within this thesis are:

1. A novel approach is proposed for finding sub-optimal image segmentations achieving bit-rate minimization applied to efficient coding of digital video sequences.
2. Several possible extensions of first generation video coding standards are proposed leading to improved video quality at low bit rates. Accordingly, a video codec structure is presented and justified by theoretical analysis. A cost function formulation is shown along with a minimization procedure to achieve bit-rate optimization.
3. A software based video codec was designed to evaluate the performance of the new codec and serve as a basis for future experimentation. The performance of the codec is evaluated against modern videoconferencing standards.
4. The region-oriented video coding approach taken in this thesis is also shown to be highly parallelizable, potentially leading to fast hardware implementation.

1.3 Outline of the Thesis

This thesis is organized into five chapters. Chapter 1 presents an introduction to the subject matter and sets the overall context as well as motivation for the research. Back-

ground material on modern video coding algorithms, region-oriented coding, and the MDL principle is provided in Chapter 2. Chapter 3 describes the design of a bit-stream video codec based on quadtree segmentation and the MDL principle. Statistical modeling used in the development of the cost function is explained and validated. Finally, an MDL cost function formulation is presented which serves as the basis of the optimization procedure. Chapter 4 discusses the experimental results of the video codec in terms of overall PSNR performance versus the modern H.263 algorithm. A variety of video sequences are used for comparison purposes. Finally, in Chapter 5, a summary of the results and conclusions of the thesis are given as well as suggestions for future investigation.

Chapter 2

Background

2.1 Video Coding

2.1.1 Applications

Videoconferencing and videotelephony have a wide range of applications including:

- desktop and room-based conferencing;
- video over the Internet and over telephone lines;
- surveillance and monitoring;
- telemedicine (medical consultation and diagnosis at a distance);
- computer-based training and education.

In each case, video information (and perhaps audio as well) is transmitted over telecommunications links, including networks, telephone lines, ISDN and wireless. Video has a high bandwidth and so these applications require video compression or video coding technology to reduce the bandwidth before transmission.

2.1.2 Standards

Standards are essential for practical communications. Without a common language that both the transmitter and the receiver understand, communication is impossible. For multimedia communication that involves transmission of video data, standards play an even more important role. Not only does a video coding standard have to specify a common language, or formally known as the *bitstream syntax*, the language also has to be efficient for compression of video content. This is due to the relatively large amount of bits required to transmit uncompressed video data.

In our discussions of standards we focus on the standards developed by International Telecommunication Union - Telecommunication Standardization Sector (ITU-T), formerly called the Consultative Committee of the International telephone and Telegraph (CCITT). These include H.261, H.263, and a more recent effort, informally known as H.263+, to provide a new version of H.263, i.e., H.263 Version 2, in 1998. These video codec standards form important components of the ITU-T H-Series Recommendations that standardize audiovisual terminals in a variety of network environments.

For multimedia communication, there are two major standard organizations: the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T), and the International Organization for Standardization (ISO). Recent video coding standards defined by these two organizations are summarized in Table 2.1. These standards differ mainly in the operating bit-rates due to the applications they are originally designed for, although all standards can essentially be used for all applications at a wide range of bit rates. In terms of coding algorithms, all standards in Table 2.1 follow a

similar framework, and differ only in the ranges of parameters and some specific coding modes. In this chapter, we will focus on the standards developed by ITU-T: H.261, H.263, and H.263 Version 2.

Standards Organization	Video Coding Standard	Typical Range of Bit Rates	Typical Applications
ITU-T	H.261	p×64 kbits/s, p=1…30	ISDN Video Phone
ISO	IS 11172-2 MPEG-1 Video	1.2 Mbits/s	CD-ROM
ISO	IS 13818-2 MPEG-2 Video	4-80 Mbits/s	SDTV, HDTV
ITU-T	H.263	64 kbits/s or below	PSTN Video Phone
ISO	CD 14496-2 MPEG-4 Video	24-1024 kbits/s	
ITU-T	H.263 Version 2	< 64 kbits/s	PSTN Video Phone
ITU-T	H.263L	< 64 kbits/s	-

Table 2.1: Video Coding Standards

2.2 H.261 Standard

2.2.1 Introduction

H.261 is a video coding standard defined by the ITU-T Study Group XV (SG15) for video telephony and video conferencing applications [13]. It emphasizes low bit-rates and low coding delay. It was originated in 1984 to be used for audiovisual services at bit rates around m×384 kbits/s, where m is between 1 and 5. In 1988, the focus shifted to bit rates of around p×64 kbits/s, where p is from 1 to 30. Therefore, H.261 also has an

informal name called p×64 (pronounced as p *times* 64). H.261 was approved in December 1990. The coding algorithm used in H.261 is basically a hybrid of *motion compensation* to remove temporal redundancy and *transform coding* to reduce spatial redundancy. Such a framework forms the basis of all video coding standards that were developed later. Therefore, H.261 has very significant influence on many other existing and evolving video coding standards.

2.2.2 Source Picture Formats

Digital video is composed of a sequence of frames, which occur at a particular rate. For H.261, the frame rate is specified to be 30000/1001 (approximately 29.97) frames per second. Each frame is composed of a number of samples. These samples are referred to as *pixels* (*pi*cture *el*ements), or simply *pels*. For a video coding standard, it is important to understand the picture sizes that the standard applies to, and the position of samples. H.261 is designed to deal with two picture formats: the common intermediate format (CIF) and the quarter CIF (QCIF). Refer to Table 2.2 for a summary of a variety of picture formats. At such a resolution, the picture quality is not very high. It is close to the quality of a typical video cassette recorder, and is much less than the quality of the broadcast television. This is because H.261 is designed for video telephony and video conferencing, in which typical source material is composed of scenes of talking persons, so-called head and shoulder sequences, rather than general TV programs that contain a lot of motion and scene changes.

	Sub-QCIF	QCIF	CIF	4CIF	16CIF
No. of Pixels per Line	128	176	352	704	1408
No. of Lines	96	144	288	576	1152
Uncompressed Bit Rate	4.4Mbs	9.1 Mb/s	37 Mb/s	146 Mb/s	584 Mb/s

Table 2.2: Picture Formats Supported by H.261 and H.263

In H.261, each sample contains a luminance component, called Y, and two chrominance components, called C_B and C_R. The values of these components are defined as in [4]. In particular, "Black" is represented by Y=16, "White" is represented by Y=235, and the range of C_B and C_R is between 16 and 240, with 128 representing zero color difference (i.e., grey). A picture format, as shown in Figure 2.1, defines the size of the image, hence the resolution of the Y pels. The chrominance pels, however, typically have a lower resolution than the luminance pels, in order to take advantage of the fact that human eyes are less sensitive to chrominance. In H.261, the C_B and C_R pels are specified to have half the resolution, both horizontally and vertically, of that of the Y pels. This is commonly referred to as the 4:2:0 format. Each C_B or C_R pel lies in the center of four neighboring Y pels, as shown in Figure 2.1. Note that block edges lie in-between rows or columns of Y pels.

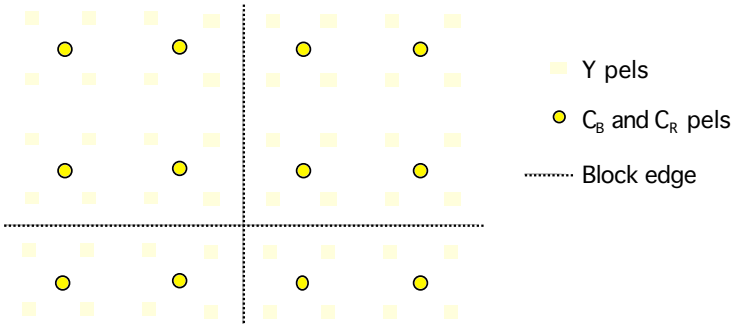


Figure 2.1 Positions of Samples for H.261

2.2.3 Blocks, Macroblocks & Group of Blocks

Typically, a frame is not encoded in its entirety. Instead, it is divided into blocks that are processed one by one, both by the encoder and the decoder, in a scan order as shown in Figure 2.2. This approach is often referred to as *block-based coding*.

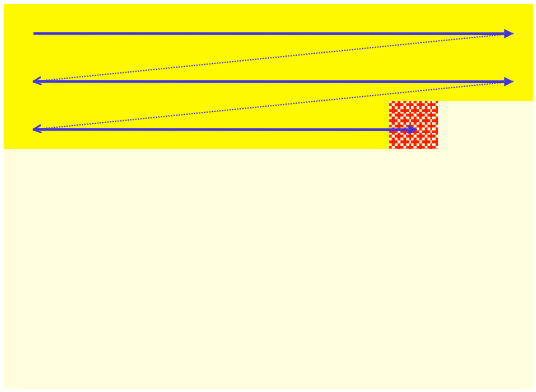


Figure 2.2 Illustration of block-based coding

In H.261, a block is defined as a group of 8×8 pels. Because of the downsampling in the chrominance components, one block of C_B pels and one block of C_R pels correspond to four blocks of Y pels. The collection of these six blocks is called a macroblock (MB), as shown in Figure 2.3, with the order of blocks marked as 1 to 6. A MB is treated as one unit in the coding process.

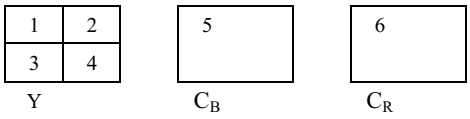


Figure 2.3 A macroblock

A number of MBs are grouped together and called a group of blocks (GOB). For H.261, a GOB contains 33 MBs, as shown in Figure 2.4. The resulting GOB structures for a picture, in the CIF case and the QCIF case, are shown in Figure 2.5.

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22
23	24	25	26	27	28	29	30	31	32	33

Figure 2.4 A group of blocks (GOB)

GOB 1	GOB 2
GOB 3	GOB 4
GOB 5	GOB 6
GOB 7	GOB 8
GOB 9	GOB 10
GOB 11	GOB 12

CIF

GOB 1
GOB 3
GOB 5

QCIF

Figure 2.5 GOB structures

2.2.4 The Compression Algorithm

Compression of video data typically involves two principles: the reduction of spatial redundancy and the reduction of temporal redundancy. H.261 uses the discrete cosine transform to remove spatial redundancy, and motion compensation to remove temporal redundancy.

Transform coding has been widely used to remove redundancy between data samples. In transform coding, a set of data samples are first linearly transformed into a set of

transform coefficients. These coefficients are then quantized and entropy coded. A proper linear transform can de-correlate the input samples, and hence remove the redundancy. Another way to look at this is that a properly chosen transform can concentrate the energy of input samples into a small number of transform coefficients, so that resulting coefficients are easier to encode than the original samples.

The most commonly used transform for video coding is the discrete cosine transform (DCT) [1][35]. Both in terms of objective coding gain and subjective quality, DCT performs very well for typical image data. The DCT operation can be expressed in terms of matrix multiplication:

$$\mathbf{Y} = \mathbf{C}^T\mathbf{X}\mathbf{C} \tag{2.1}$$

Where $\mathbf{X}$ represents the original image block, and $\mathbf{Y}$ represents the resulting DCT coefficients. The elements of $\mathbf{C}$, for an 8×8 image block, are defined as

$$C_{mn} = k_n \cos\left[\frac{(2m+1)n\pi}{16}\right] \quad \text{where } k_n = \begin{cases} 1/(2\sqrt{2}) & \text{when } n = 0 \\ 1/2 & \text{otherwise} \end{cases} \tag{2.2}$$

After the transform, the DCT coefficients in $\mathbf{Y}$ are quantized. Quantization implies loss of information, and is the primary source of compression. The quantization step size depends on the available bit rate, and can also depend on the coding modes. Except for the intra DC coefficients that are uniformly quantized with a step size of 8, the "dead zone" is used to quantize all other coefficients in order to remove noise around zero. The input-output relations for the two cases are shown in Figure 2.6.

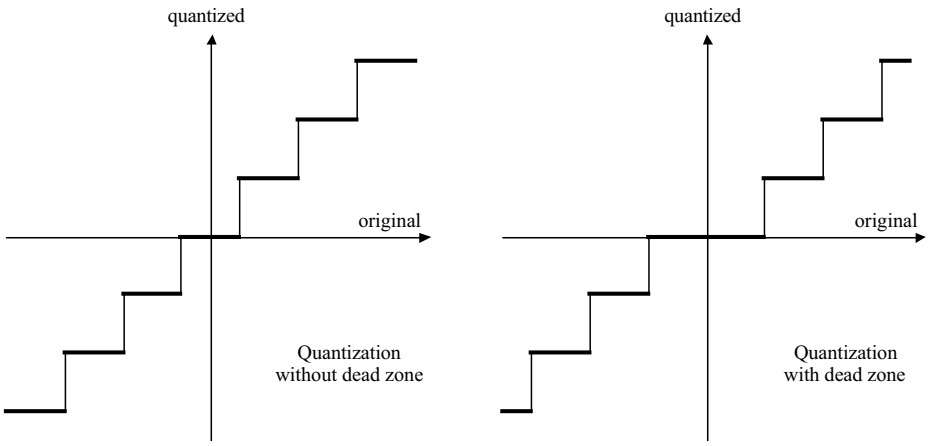


Figure 2.6 Quantization with and without "dead zone"

The quantized 8×8 DCT coefficients are then converted into a one-dimensional (1D) array for entropy coding. Figure 2.7 shows the scan order used in H.261 for this conversion. Most of the energy concentrates on the low frequency coefficients, and the high frequency coefficients are usually very small and are quantized to zero before the scanning process. Therefore, the scan order in Figure 2.7 can create long runs of zero coefficients, which is important for efficient entropy coding, as we will discuss in the next paragraph.

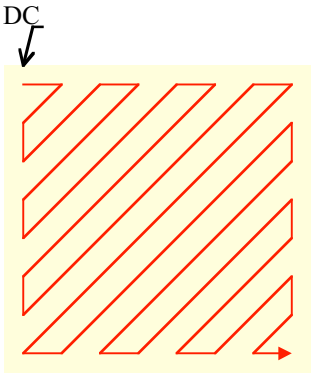


Figure 2.7 Scan order of the DCT coefficients

The resulting 1D array is then decomposed into segments, with each segment containing one or more (or none) zeros followed by a nonzero coefficient. Let an *event* represent the

pair of (*run, level*), where "run" represents the number of zeros and "level" represents the magnitude of the nonzero coefficient. This coding process is sometimes called "run-length coding." Then, a variable length Huffman coding table is built to represent each event by a specific codeword. In H.261, this table is often referred to as a two-dimensional (2D) VLC table because of its 2D nature, i.e., each event representing a pair of (run, level).

At the decoder, all the above steps are reversed one by one. Note that all the steps can be exactly reversed except for the quantization step, which is where the loss of information arises.

2.2.5 Motion Compensation

The transform coding described in the previous section removes spatial redundancy within each frame. It is therefore referred to as *intra* coding. However, for video material, *inter* coding is also very useful. Typical video material contains a large amount of redundancy along the temporal axis. Video frames that are close in time usually have a large amount of similarity. Therefore, transmitting the difference between frames is more efficient than transmitting the original frames. This is similar to the concept of differential coding and predictive coding. The previous frame is used as an estimate of the current frame, and the residual, the difference between the estimate and the true value, is coded. When the estimate is good, it is more efficient to code the residual than to code the original frame. Consider the fact that typical video material is composed of moving objects. Therefore, it is possible to improve the prediction result by first estimating the motion of each region in the scene. More specifically, the encoder can estimate the

motion (i.e., displacement) of each block between the previous frame and the current frame. This is often achieved by matching each macroblock in the current frame with the previous frame to find the best matching area. This area is then offset accordingly to form the estimate of the corresponding block in the current frame. Now, the residue has much less energy and therefore is much easier to code. This process is called motion compensation (MC), or more precisely, motion-compensated prediction [29][30]. This is illustrated in Figure 2.8. The residue is then coded using the same process as that of intra coding.

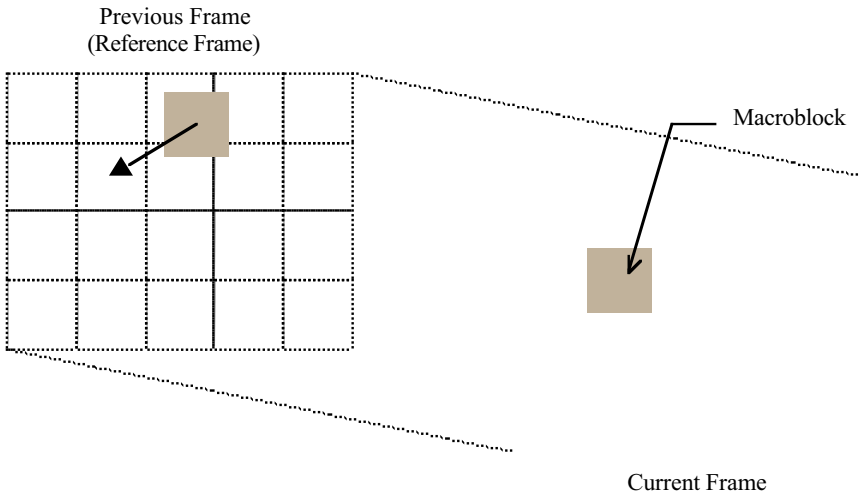


Figure 2.8 Motion Compensation

Frames that are coded without any reference to previously coded frames are called intra frames, or simply I-frames. Frames that are coded using a previous frame as a reference for prediction are called predicted frames, or simply P-frames. However, a P-frame may contain also intra coded blocks. For a certain block, it may be impossible to find a good enough matching area in the reference frame to be used as prediction. In this case, direct intra coding of such a block is more efficient. This situation happens often

when there is occlusion in the scene, or when the motion is very heavy, i.e., when there is a high degree of fast motion in a scene, or multiple moving objects.

Motion compensation saves the bits for coding the DCT coefficients. However, it does imply that extra bits are required to carry information about the motion vectors. Efficient coding of motion vectors is therefore also an important part of H.261. Because motion vectors of neighboring blocks tend to be similar, differential coding of the horizontal and vertical components of motion vectors is used. That is, instead of coding motion vectors directly, the previous motion vector is used as a prediction for the current motion vector, and the difference, in both the horizontal component and the vertical component, is then coded using a motion vector VLC table. Short codewords are used to represent small difference, because these are more likely events.

2.2.6 Summary

The coding algorithm used in H.261 can be shown as block diagrams in Figure 2.9 and Figure 2.10. At the encoder, the input picture is compared with the previously decoded frame with motion compensation. The difference signal is DCT transformed and quantized, and then entropy coded and transmitted. At the decoder, the decoded DCT coefficients are inverse DCT transformed and then added to the previously decoded picture with motion compensation.

Since the prediction of the current frame is composed of blocks at various locations in the reference frame, the prediction itself may contain coding noise and blocking artifacts. These artifacts may cause a higher prediction error. It is possible to reduce the prediction error by passing the predicted frame through a lowpass filter before it is used

as the prediction for the current frame. This filter is referred to as a loop filter, because it operates inside the motion compensation loop.

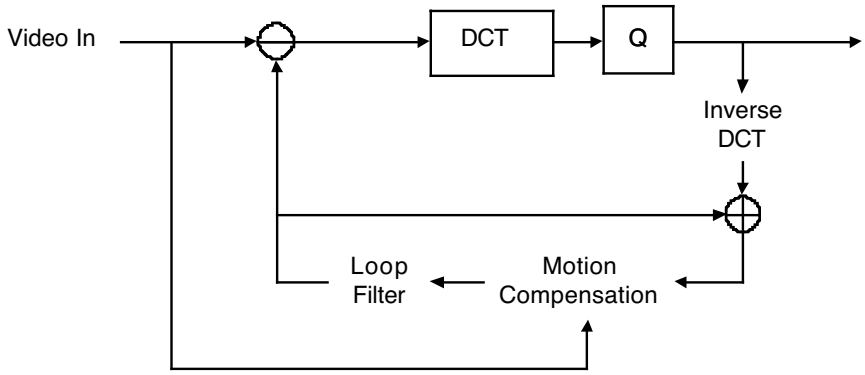


Figure 2.9 Block diagram of a video encoder

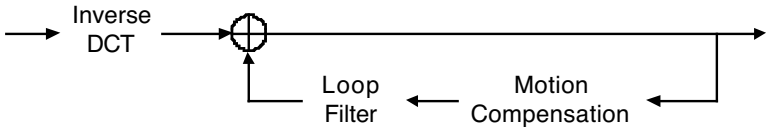


Figure 2.10 Block diagram of a video decoder

2.2.7 Reference Model

As in all video coding standards, H.261 specifies only the bit-stream syntax and how a decoder should interpret the bit-stream to decode the image. Therefore, it specifies only the design of the decoder, not how the encoding should be done. For example, an encoder can simply decide to use only zero motion vectors and let the transform coding take all the burden of coding the residual. This may not be an efficient encoding algorithm, but it does generate a standard-compliant bit-stream.

Therefore, to illustrate the effectiveness of a video coding standard, an example encoder is often provided by the group that defines the standard. For H.261, such an exam-

ple encoder is called a reference model (RM), and the latest version is RM 8 [40]. It specifies details about motion estimation, quantization, decisions for inter/intra coding and MC/no MC, buffering, and the rate control.

2.3 H.263 Standard

2.3.1 Introduction

H.263 [14] was defined by ITU-T SG15, the same group that defined H.261. The activities of H.263 started around November 1993, and the standard was adopted in March 1996. The main goal of this endeavor was to design a video coding standard suitable for applications with bit rates below 64 kbits/s. For example, when sending video data over the public service telephone network (PSTN) and mobile networks, the video bit rates typically range from 10 to 24 kbits/s. During the development of H.263, it was identified that the near-term goal would be to enhance H.261 using the same general framework, and the long-term goal would be to design a video coding standard that may be fundamentally different from H.261 in order to achieve further improvement in coding efficiency. As the standardization activities move along, the near-term effort became H.263 and H.263 Version 2 (or H.263+ and H.263++), and the long-term effort is now referred to as H.263L. H.263 Version 2 maintains bit-stream compatibility with H.263 and the same basic coding algorithm structure (hybrid-DCT), while H.26L considers more radically different approaches.

In essence, H.263 combines the features of H.261 together with MPEG, and is optimized for very low bit-rates. In terms of signal to noise ratio (SNR), H.263 can provide 3

to 4 dB gain over H.261 at bit-rates below 64 kbits/s. In fact, H.263 provides superior coding efficiency to that of H.261 at all bit rates. When compared with MPEG-1, H.263 can give 30% bit-rate saving.

2.3.2 H.263 vs. H.261

Since H.263 was built on top of H.261, the main structures of the two standards are essentially the same. The major differences are:

1. H.263 supports more picture formats, and uses a different GOB structure.
2. H.263 uses half-pel motion compensation, but does not use loop filtering as in H.261.
3. H.263 uses 3D VLC for coding of DCT coefficients.
4. In addition to the basic coding algorithm, four options that are negotiable between the encoder and the decoder provide improved performance.
5. H.263 allows the quantization step size to change at each MB with less overhead.

2.3.3 Picture Formats, Samples, and the GOB Structure

In addition to CIF and QCIF as supported by H.261, H.263 also supports sub-QCIF, 4CIF and 16CIF. Resolutions of these picture formats can been found in Table 2.2. Chrominance subsampling and the relative positions of chrominance pels are the same as those defined in H.261. However, H.263 uses different GOB structures. These are shown in Figure 2.11 for various formats. Unlike H.261, a GOB in H.263 always contains at least one full row of MBs.

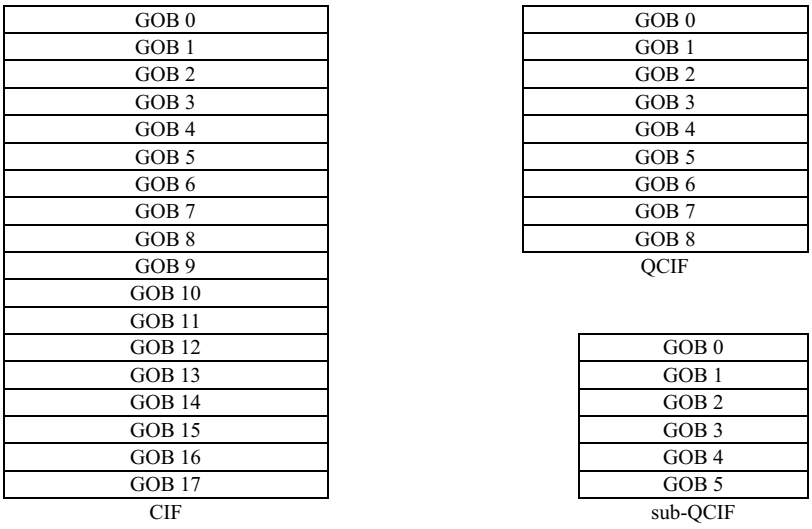


Figure 2.11 GOB structures for H.263

2.3.4 Half-Pel Prediction and Motion Vector Coding

A major difference between H.261 and H.263 is the half-pel prediction in the motion compensation. This concept is also used in MPEG. While the motion vectors in H.261 can have only integer values, H.263 allows the precision of motion vectors to be at half pixel. For example, it is possible to have a motion vector with values (4.5, -2.5). When a motion vector has non-integer values, bilinear interpolation is used to find the corresponding pel values for prediction.

The coding of motion vectors in H.263 is more sophisticated than that in H.261. The motion vectors of three neighboring MBs (the left, the above, and the above-right, as shown in Figure 2.12) are used as predictors. The median of the three predictors is used as the prediction for the motion vector of the current block, and the prediction error is coded and transmitted. However, around a picture boundary or GOB boundary, special cases are needed. When only one neighboring MB is outside the picture boundary or GOB boundary, a zero motion vector is used to replace the motion vector of that MB as

the predictor. When two neighboring MBs are outside, the motion vector of the only neighboring MB that is inside is used to as the prediction. These are shown in Figure 2.13.

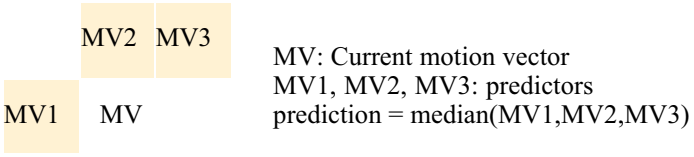


Figure 2.12 Prediction of motion vectors

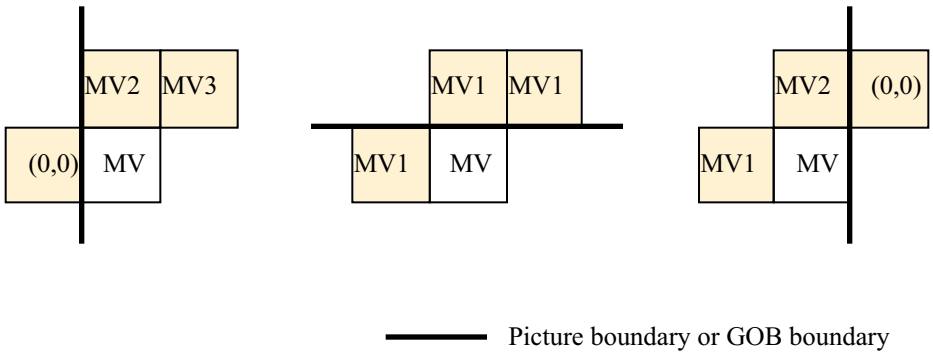


Figure 2.13 Motion vector prediction at picture/GOB boundaries

2.3.5 Run Length Coding of DCT Coefficients

H.263 improves the run-length coding used in H.261 by giving an extra term "last" to indicate whether the current coefficient is the last nonzero coefficient of the block. Therefore, a set of (run, level, last) represents an event and is mapped to a codeword in the VLC table, hence the name 3D VLC. With this scheme, the EOB (end of block) code used in H.261 is not needed anymore.

2.3.6 Negotiable Options

H.263 specifies four options that are negotiable between the encoder and the decoder. At the beginning of each communication session, the decoder signals the encoder which of these options the decoder has the capability to decode. If the encoder also supports some of these options, it may enable those options. However, the encoder does not have to enable all the options that are supported by both the encoder and decoder. The four options in H.263 are: the unrestricted motion vector mode, the syntax-based arithmetic coding mode, the advanced prediction mode, and the PB-frame mode.

2.3.7 Unrestricted Motion Vector Mode

This is the first one of the four negotiable options defined in H.263. In this option, motion vectors are allowed to point outside of the picture boundary. In this case, edge pels are repeated to extend to the pels outside so that prediction can be done. Significant coding gain can be achieved with unrestricted motion vectors if there is movement around picture edges, especially for smaller picture formats like QCIF and sub-QCIF. In addition, this mode allows a wider range of motion vectors than H.261. Large motion vectors can be very effective when the motion in the scene is caused by heavy motion, e.g., motion due to camera movement.

2.3.8 Syntax-Based Arithmetic Coding (SAC)

In this option, arithmetic coding [54] is used, instead of VLC tables, for entropy coding. Under the same coding conditions, using arithmetic coding will result in a bit-stream different from the bit-stream generated by using a VLC table, but the reconstructed frames and the SNR will be the same. Experiments show that the average bit-rate saving is about 3-4% for inter frames, and about 10% for intra blocks and frames.

2.3.9 Advanced Prediction Mode

In the advanced prediction mode, overlapped block motion compensation (OBMC) [33] is used to code the luminance of P-pictures, which typically results in fewer blocking artifacts. This mode also allows the encoder to assign four independent motion vectors to each MB. That is, each block in a MB can have an independent motion vector. In general, using four motion vectors gives better prediction, since one motion vector is used to represent the movement of a 8×8 block, instead of a 16×16 MB. Of course, this implies more motion vectors, and hence requires more bits to code the motion vectors. Therefore, the encoder has to decide when to use four motion vectors and when to use only one. Finally, in the advanced prediction mode, motion vectors are allowed to cross picture boundaries as is the case in the unrestricted motion vector mode.

2.3.10 PB-Frame Mode

In the PB-frame mode, a PB-frame consists of two pictures coded as one unit, as shown in Figure 2.14. The first picture, called the P-picture, is a picture predicted from the last decoded picture. The last decoded picture can be either an I-picture, a P-picture, or the P-picture of a PB-frame. The second picture, called the B-picture (B for bi-directional), is a picture predicted from both the last decoded picture as well as the P-picture that is currently being decoded. As opposed to the B-frames used in MPEG, PB frames do not need separate bi-directional vectors. Instead, forward vectors for the P-picture is scaled, and added to a small delta-vector, to obtain vectors for the B-picture. This results in lower bit rate overhead for the B-picture. For relatively simple sequences at low bit rates, the picture rate can be doubled with this mode with minimal increase in the bit rate. However, for sequences with heavy motion, PB-frames do not work as well as B-pictures. Also, note that the use of PB-frame mode increases the end-to-end delay, so it may not be suitable for two-way interactive communication.

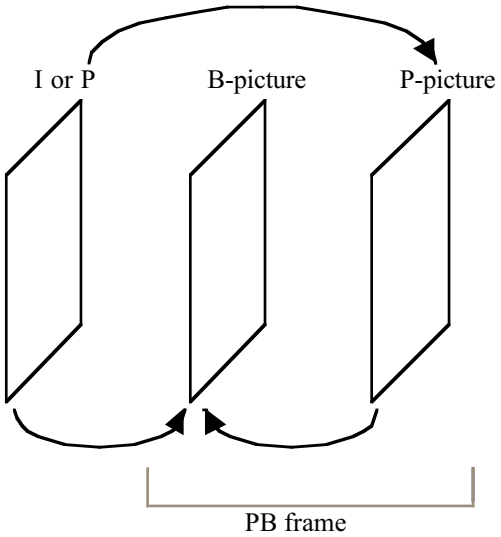


Figure 2.14 The PB-frame mode

2.3.11 Test Model Near-Term (TMN)

Similar to H.261, there are documents drafted by ITU-T SG15 that describe example encoders, i.e., the test models. For H.263, these are called TMN, where N indicates that H.263 is a near-term effort in improving H.261. The latest version is TMN11 [49]. TMN11 specifies the details of the advanced motion prediction, the overlapped block motion compensation, the choice between the 8×8 mode and 16×16 mode for motion vectors, the syntax-based arithmetic coding, the use of the PB frame mode, handling in error prone packet environments, and Rate-Distortion Optimization.

2.4 Region-Oriented Video Coding

2.4.1 Introduction

The region-oriented approach has earned significant attention recently by researchers in the area of low bit-rate and very low bit-rate video coding. The adoption of these techniques can translate to lower overall bit-rate consumption while maintaining perceptual quality. First generation video coding standards such as ITU-T H.261, H.263, and MPEG suffer from overly simplistic region segmentation. The fixed block-type region boundaries do not take into account the actual content of the scene.

Region-oriented techniques are based on modeling images as a combination of non-stationary visual primitives, such as contours, textures, and motion. The advantage of these methods compared to object-oriented and 3-D model-based methods [2][52], is that they do not require a priori information of the image content. Also, they are character-

ized by lower computational complexity, which is a critical factor for low power implementation.

In current standards [13][14], motion compensation and encoding of the prediction error is block-based. However, the new MPEG-4 standard [20] provides content based functionalities in addition to efficient video coding. To fulfill this task a source model is needed that adapts better to the scene contents than blocks. In so-called object-based image coding techniques [28][16][7], the partition of images into a fixed block structure is replaced by arbitrarily shaped regions that are related either to texture/color [41][24] or to motion [45]. The shape of the regions has to be transmitted as additional side information. The efficiency of such a region-based coding scheme depends very much on properly adjusting the amounts of bits used for region coding.

2.4.2 MDL-Based Segmentation

Zheng and Blostein [57] proposed a new optimality criterion based on the minimum description length (MDL) principle and was developed for moving object estimation and segmentation for region-oriented video coding applications. Using this MDL estimator, the cost minimized is the sum of the ideal coding lengths for the motion parameters, boundaries and motion-compensated predictive errors of all moving objects in a scene. An optimization procedure to obtain a sub-optimal MDL estimator was shown based on a region-merging framework. A number of experimental comparisons showed a significant ideal coding rate reduction of the object-oriented coding scheme using an MDL estimator over a standard block-oriented scheme. This work demonstrated the theoretical potential

of the MDL-based estimator and serves as a foundation for a key element of the algorithms developed in this thesis.

The common drawback of current moving object estimation is that the choice of the optimality criterion does not directly match the requirements of the image sequence coding process. A superior approach is to integrate the coding requirements into the criterion used for moving object estimation. The optimal trade-off between motion compensated prediction error information and the motion vector data is an active area of research and various techniques have been investigated [42][10]. The minimum description length (MDL) principle minimizes the coding cost for the data to be compressed, and therefore seems to be a natural choice to combine moving object estimation with image sequence coding.

The MDL principle was originally proposed by Rissanen [37][38][39] and is described in Section 2.5. MDL provides a framework for estimating both integer-valued structure parameters and real-valued parameters that specify a model for the data source. Using MDL, we can estimate the least number of bits that are needed to encode the observed data with regard to a particular data model. When a particular coding scheme is specified, there is a natural trade-off between bits spent on model parameters and bits spent on data from that model. This feature is intuitively appealing when the purpose of the estimation problem is to encode the observed data.

By combining motion segmentation with motion parameter estimation, we can estimate motion discontinuities more appropriately in the context of the application at hand. Based on the advantages mentioned, the MDL principle can be applied to the moving object segmentation and motion estimation [57]. In [32], a technique is shown for re-

gion-oriented video coding based on the minimization of an MDL criterion where the motion-compensated prediction error is coded together with a segmentation map and motion parameters. The MCPE is coded using entropy coding of the quantized MCPE as well as DCT-based coding. The region segmentation is approximated by a polygonal approximation allowing reduced segmentation coding cost. The polygonal representation minimizes the MDL criterion yielding an optimal approximation in terms of compression.

2.4.3 Rate-Distortion Optimization

A solution for the optimum trade-off by applying rate-distortion theory has been presented for region based motion compensation [45]. The regions are optimized with respect to a Lagrangian cost function by variation of the region contours. The resulting optimal contours do not necessarily coincide with the actual contours of the objects in the scene. However, for the optimized regions the improvement in distortion and the region's rate are well balanced in a rate-distortion sense. The improvement that has been achieved with a coding scheme using optimized regions is demonstrated in [45]. It was shown that motion compensation with rate-distortion optimized regions offers about 2 dB better PSNR (Peak-Signal-to-Noise-Ratio, see Equation 4.1) than block-based motion compensation. In both cases a block-based DCT-coder was used for coding the prediction error image.

A rate-distortion framework is proposed in [27] that defines a jointly optimal motion vector estimation and quadtree segmentation for video coding. This technique attempts to achieve maximum reconstructed frame quality under the constraint of a target bit-rate for the coding of the motion vector information, quadtree segmentation, and the motion-

compensated prediction error. Extending this concept, [56] proposes a generalized rate-distortion optimized approach for the joint selection of coding parameters; namely the quadtree structure, the motion parameters, the mode and the MCPE quantizer selection associated with each tree node. The problem is formulated using the Lagrange multiplier method and solved using dynamic programming where the Viterbi Algorithm is used to find the optimal solution.

2.5 MDL Principle

The MDL principle was originally proposed by Rissanen [37][38][39]. MDL provides a framework for estimating both integer-valued structure parameters and real-valued parameters that specify a model for the data source. The principle is to use the least number of bits necessary to encode an observed data sequence **x** generated by a stochastically modeled source. This principle leads to an optimal parameter estimator with minimum coding length for the observed data as the optimality criterion. The coding length obtained by such an estimator corresponds to a notion of information in the data **x** relative to the class of models [38]. This notion of information consists of two terms: 1) Shannon's probabilistic notion of information, which describes the observations **x** generated by the stochastically modeled source and 2) Kolmogorov's algorithmic notion of information [47], which describes the nonrandom selection of the models or parameters. It is Kolmogorov's algorithmic notion of information that extends the classical maximum likelihood criterion and permits estimation of the number of parameters without a separate hypothesis test. This mixture model of information provides the common measure of complexity that can be assigned to both the data models and parameters.

2.5.1 Relation between estimation and coding

In the coding problem, we are given a string of observed data points $x_t, t = 1,...,n$, each truncated to some finite precision, and the objective is to re-describe the data with a suitably designed code as efficiently as possible, i.e., with a short coding length. In estimation, which is a fundamental problem in signal processing and related fields, we seek an explanation of the observations, or, rather, of the underlying mechanism, which we believe has generated the observed data. More precisely, we select a parametrically defined statistical model described by a probability density function, $P_\theta(\mathrm{x})$, for the data **x**, and try to estimate the vector parameter $\theta = (\theta_1,...,\theta_m)$ from the observations, where *m* is an integer variable to be estimated. The use of the probability density functions is motivated by the fact that each observed realization x_i is always expressed in finite precision, with *q* fractional binary digits.

By representing the number x_i using binary notation, we see that the entire sequence **x** can be written down using *nq* bits. But such a trivial coding or description of the observed sequence does not take into account the possible correlations that exist between the numbers x_i nor the relative frequency with which each observation occurs. If such dependencies were taken advantage of, we might be able to reduce the total number of binary digits in the description of **x**. The dependencies between data can often best described by a parametric model, and the coding length $L(\mathbf{x})$ of the data **x** will be a function of those model parameters. The shortest coding length should result if the true parameters are used in the code design.

Rissanen [39] has shown that a one-to-one relation exists between the coding length function $L(\mathbf{x})$ and the negative base-two logarithm of the probability density function $P_\theta(\mathrm{x})$ used to describe the data model, i.e.,

$$L(\mathrm{x}) = -logP_\theta(\mathrm{x}) \tag{2.3}$$

where θ is a parameter vector which specifies a whole class of PDFs. If we pick just any "model" $P_\theta(\mathrm{x})$ within the class, and encode the data **x** with $-logP_\theta(\mathrm{x})$ bits, then the mean coding length $-\sum P_{\theta^0}(\mathrm{x})logP_\theta(\mathrm{x})$, where the sum is over all data sequences **x** of length *n* and θ^0 denotes the "true" parameter, cannot be smaller than the entropy, which is defined as $-\sum P_{\theta^0}(\mathrm{x})logP_{\theta^0}(\mathrm{x})$. Moreover, equality is achieved only when $\theta = \theta^0$. Therefore, if the observed data sequence has probability $P_\theta(\mathrm{x})$ with θ regarded as known, then the minimum coding length for the observed data is $-logP_\theta(\mathrm{x})$ bits. This coding length is called the ideal coding length. If θ is unknown, and we wish to design the shortest code, we clearly have to estimate θ so as to minimize the ideal coding length $-logP_\theta(\mathrm{x})$. This is an alternative interpretation of the familiar Maximum Likelihood (ML) estimator.

We have not yet considered the problem of obtaining a compact description of. θ. Without any cost assigned to encoding the parameters we could, in principle, bring the mean coding length $-\sum P_{\theta^0}(\mathrm{x})logP_\theta(\mathrm{x})$ as near to the entropy $-\sum P_{\theta^0}(\mathrm{x})logP_{\theta^0}(\mathrm{x})$ as we like by increasing the complexity of the model, i.e., the dimension of θ. This is one reason why the correct model structure cannot be determined by the ML estimator. This problem can be solved by including the number of bits spent on encoding the parameters into the ideal coding length function. The interpretation of this solution can be identified with Maximum A Posteriori (MAP) estimation [25] as discussed below.

When we include the ideal coding length for the parameters into the total coding length function, we have

$$L(\mathrm{x}) = -logP_{\theta}(\mathrm{x}) + L(\theta) \tag{2.4}$$

where $L(\theta)$ denotes the ideal coding length for the parameters. The problem of efficient encoding of the model parameters, θ, is much different from encoding the random observations **x** because θ cannot be readily modeled by probability distributions. By similar arguments as used for the data model term, Rissanen has shown in [39] that $2^{-L(\theta)}$ defines a prior distribution function for the parameters under certain conditions. That is

$$P(\theta) = 2^{-L(\theta)} \tag{2.5}$$

where the code length function L(θ) satisfies the Kraft Inequality [47] with equality.

As examples, if the parameter is a constant vector known to the decoder, we will not need to encode it at the transmitter, so $L(\theta) = 0$ and $P(\theta) = 1$. If the parameter is known to range uniformly over a finite set of M values, then we will need $log(M)$ bits to encode it and with $P(\theta) = 1/M$.

Using equations (2.4) and (2.5), we can write the total coding length function as

$$L(\mathrm{x}) = -logP_{\theta}(\mathrm{x}) - logP(\theta) \tag{2.6}$$

or in a more familiar form

$$\begin{aligned} L(\mathrm{x}) &= -log[P_{\theta}(\mathrm{x})P(\theta)] \\ &= -log[P(\mathrm{x}|\theta)P(\theta)] \end{aligned} \tag{2.7}$$

On the other hand, the MAP criterion chooses the parameter vector θ that maximizes the conditional probability of the model, given the data: $P(\theta|\mathrm{x})$. An application of Bayes's rule yields

$$P(\theta|\mathrm{x}) = \frac{P(\mathrm{x}|\theta)P(\theta)}{P(\mathrm{x})} \tag{2.8}$$

Since $P(\mathrm{x})$ is constant with respect to θ, the MAP tactic is to choose θ that maximizes

$$P(\mathrm{x}|\theta)P(\theta) \tag{2.9}$$

From (2.7) and (2.8), we see that the strategy of finding the minimum coding length by choosing particular model parameters is equivalent to the MAP strategy of maximizing the conditional probability for the model, given the data: $P(\theta|\mathrm{x})$.

2.5.2 Prior information and parameter coding

There are two sources of information in the estimation problem. The first consists of observed data **x**, and the second, called prior information, consists of everything else, based on earlier observations that are no longer available to us or based on known properties of the data source. Prior information plays as crucial a role in the MDL criterion as in MAP estimation. We first need to know how the data are generated, that is, the prior information that is used to define an observation data model. Usually this is done by selecting a parametric class of probability density functions, $P_\theta(\mathrm{x})$, and assigning a probability to every possible observed data **x**. If the observations consist of both an "input" sequence **y** and an "output" sequence **x**, then the appropriate probability density

function is $P_\theta(x|y)$. Secondly, the prior information must be taken into account to derive the ideal coding length functions for model parameters.

Of particular interest is the case where the model parameters are integers. Suppose k is an integer to be coded and one knows that the number of bits in the binary representation of the integer equal to n. Then the coding length for k is simply n. That is, integer k has a uniform distribution over a finite range $[0,2^n]$, i.e., $P(k) = 1/2^n$.

If one does not know the number of bits in the binary representation of this integer, we can encode it by a simple but inefficient method which uses a sequence 01 as a "comma". This is done by repeating every bit of the binary expansion of k twice and then ending the description with a sequence 01 so that the decoder knows that the end of the code has occured. For example, the number k = 5 (binary 101) would be encoded as 11001101. This code requires $2[logk] + 2$ bits.

A more efficient method for encoding k is through the following recursive procedure: at first, the number $(logk)$ of bits within the binary representation of k is specified, followed by the actual bits of k. To specify $(logk)$, that being the length of the binary representation of k, we use $loglogk$ bits. Continuing this recursively, we can encode k in $logk + loglogk + logloglogk + ...$ bits, summing until the last positive term. This sum of iterated logarithms is sometimes written as $log^* k$. The associated probability $P(k) = 2^{-log^* k}$ is known as a universal proper prior for the integers.

For encoding a real-valued vector parameter θ, without prior knowledge, we first truncate each component of θ, θ_i, to an integer number of bits and then encode the integer as above. The truncation performed is by writing each component θ_i of θ to preci-

sion $\pm\delta/2$. This allows for the precision of each component to be adjusted according to its contribution to the total coding length of the data **x**.

Distributions of parameters other than uniform may also be considered. For instance, since asymptotically efficient estimators in general have a near Gaussian distribution, θ_i could be modeled as Gaussian. Also, we could assume that the observed data points come in batches of, *N* points each and we could specify a conditional probability $P(\theta^k|\theta^{k-1})$ of the parameter vector θ^k for the *k*th batch given the previous parameter θ^{k-1} for the $(k-1)$th batch in terms of prior knowledge of temporal correlations of the model parameters.

2.5.3 Data model structure

In many situations, the observed data to be encoded are generated by several underlying models rather than just a single model. In the modeling process, the observed data may not be adequately described even if a complicated single model is used. Universal modeling of the encoder is needed. In broad terms, the modeling of the observed data involves a determination of local structure within the entire data and its contexts. Thus we can regard the model as consisting of two parts: 1) the local structure which specifies the set of events and their contexts, and 2) the parameters which define the probabilities assigned to the local events.

The local structure captures the global redundancies while the parameters are tailored to each individual local structure. If we can estimate the local structure of the data and use a shorter model for each subset of the data, then a shorter coding length would be

obtained even though additional bits are used to describe the local data structures. Generically, the process of finding local structures of the data is a segmentation problem for observed data. The number of local segments and its boundaries are the integer-valued structure parameters to be estimated.

The original MDL formulation did not consider this segmentation problem. The MDL formulation can be extended to the multiple model case by posing a combined segmentation and estimation problem [57]. Let $O_n, n = 1,...,N$ be a collection of disjoint subsets that partition the data. Let each subset be generated by a parametric model $P_{\theta(n)}(x|x \in O_n)$. If the prior probability distribution of the parameters for O_n is $P(\theta(n))$, then our combined segmentation and estimation problem under MDL is to estimate the number of subsets *N*, the points of b_n, representing boundaries within the subset O_n, and *N* parameter vectors $\theta(n), n = 1,...,N$ together with their model order number m_i such that the coding length for **x**, as expressed below, is minimized.

$$L(\mathrm{x}) = \sum_{\mathrm{n}=1}^{\mathrm{N}} [-logP_{\theta(n)}(x|x \in O_n) - logP(\theta(n)) + log^* m_n] + \sum_{n=1}^{N} log^* b_n \tag{2.10}$$

The last term in the above expression denotes the coding length for the boundary contour of the *i*th segment if the data **x** is in the form of a two-dimensional array.

2.5.4 MDL and Rate-Distortion Theory

As described in previous sections, the minimum description length principle for model selection proposes that, among a predetermined collection of models, we choose the one which assigns the shortest description to the data at hand. In this sense, a "description" is a lossless representation of the data that also takes into account the cost of describing the

chosen model itself. A natural extension of this principle is to consider how the MDL principle might apply to the case when the requirement for lossless coding is relaxed (lossy compression). Some recent work by Kontoyiannis in information theory has considered the connection between the MDL principle and rate-distortion theory [21], which outlines some of the mathematical and conceptual components required to consider this extension.

The use of the MDL principal in the context of lossy compression can be applied to practical data compression problems such as image and video coding. The work in [22] presents a method to use the "lossy MDL principle" introduced in order to optimally estimate the essential design parameters in Vector Quantizer design for image compression. This work also provides an interesting approach useful in the theoretical formalization of adapting MDL principle to lossy video compression.

Chapter 3

Quadtree-MDL Video Codec Design

3.1 Introduction

The challenge of image sequence coding compared to still image coding lies in an appropriate processing of motion information, along with associated image segmentation. High performance can be reached by combining intraframe and interframe coding techniques together. This is called hybrid coding as described in Sections 2.2 and 2.3. An efficient combination consists of coding the prediction error resulting from motion compensation by an intraframe technique. One important example of such a coding scheme is that case where a 2-D transform is applied on the temporal prediction error. It is referred to as motion compensated hybrid transform coding. Recent standards are based on this idea, MPEG, H261 and H.263 [13][14], which perform a 2-D Discrete Cosine Transform of the prediction error.

A good motion estimation technique and an efficient coding of the motion compensated prediction errors are imperative for high quality video coding. However, in coding the final objective of a motion estimation algorithm is unclear. For instance, a very precise motion estimation leads on the one hand to a very low prediction error energy but on the other hand to high overhead motion information. Conversely, coarse motion

estimation produces low overhead motion information but a high prediction error energy. Consequently, the optimal motion estimation algorithm should simultaneously provide an accurate motion field, while keeping the side motion information low. It should therefore aim at jointly minimizing the amount of prediction error and motion information. It should be noted that the optimal motion estimation algorithm depends on the subsequent intraframe coding technique, as well as the type of application (e.g. HDTV, TV, video-phone, etc) and the target bit-rate.

Finally, the approach applying a transform coding technique to the prediction error resulting from motion compensation presents two drawbacks. Natural images exhibit high correlation. The energy of such a signal is optimally compacted by the Karhunen Lòeve Transform (KLT). In practice the Discrete Cosine Transform (DCT) is preferred, as in this framework it approximates closely the KLT and has faster implementation. However, observations show that prediction errors have low correlations [9][11][44]. Therefore, the use of the DCT as an optimal transform is not valid anymore in this context. More generally, a transform coding technique which aims at de-correlating the data performs poorly on the motion compensated prediction error. Furthermore, in case of a block-based motion estimation technique, block artifacts can be introduced in the prediction error, reducing the efficiency of the intraframe coding technique (e.g. transform coding) if the latter is applied on the motion compensated prediction error. In spite of the last remarks, this type of coding scheme is the most efficient up to now, and therefore the most widely used in the field of video coding. In the subsequent sections, this thesis presents a new video codec structure that addresses these fundamental issues.

3.2 Video Codec Stucture

Our algorithm can be viewed as an extension of first generation coding standards [13][14]. It utilizes interframe motion-compensated predictive coding, in a similar fashion as first generation methods. The motion of each region is parameterized, coded, and transmitted along with the region's shape and motion-compensated prediction error. This concentration on interframe methods is justified by the observation that, for typical video conferencing sequences, the codec operates in interframe mode almost exclusively. Therefore, more emphasis is placed on reducing temporal correlation between frames.

An appropriate formulation for the codec is

$$\tilde{I}^t = f_\theta(\tilde{I}^{t-1}) + \hat{e}^t \tag{3.1}$$

where $\tilde{I}^t$ and $\tilde{I}^{t-1}$ each represent the current and previous reconstructed intensity images, respectively. The mapping function $f_\theta(\cdot)$ transforms $\tilde{I}^{t-1}$ to produce a prediction of the current image which is parameterized by motion parameters θ. $\hat{e}^t$ is the quantized motion-compensated prediction error resulting from the current prediction. The prediction of the current frame is represented by $f_\theta(\tilde{I}^{t-1})$. This model corresponds to the familiar Differential Pulse Code Modulation (DPCM) [36].

A block diagram of the new codec is shown in Figure 3.1 with appropriate variables indicating the video signals at various locations. The codec structure reflects the compatibility with existing hybrid video coding standards, with the notable exception that a Discrete Cosine Transform block is not used.

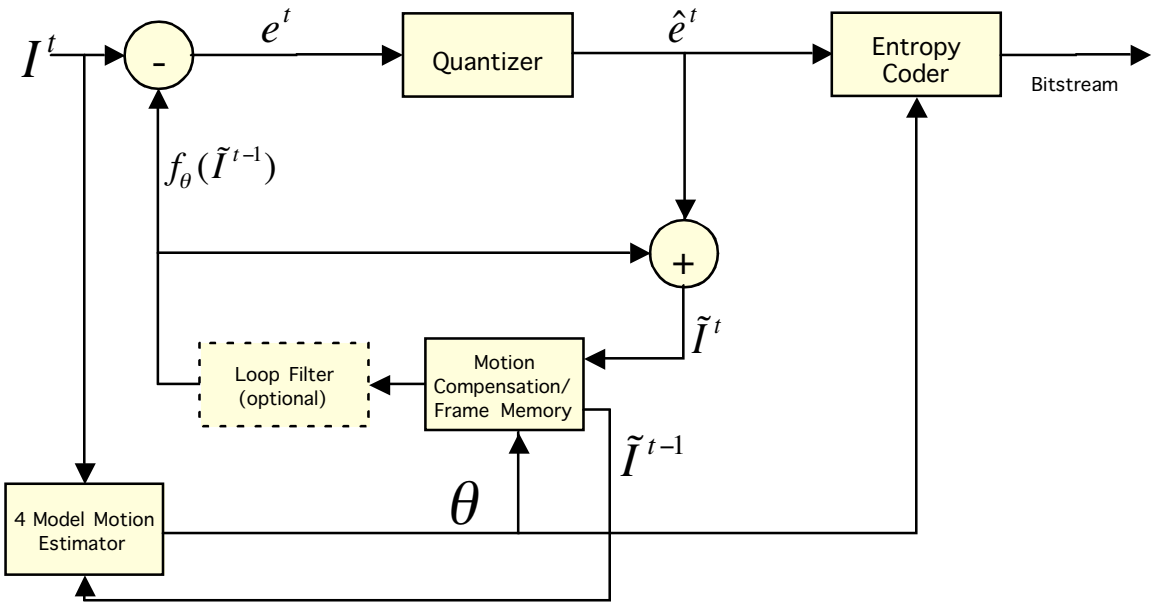


Figure 3.1: Motion-compensated structure used for Quadtree-MDL Codec

A summary of the variables indicated on the block diagram is shown below.

I^t = Current Original Image

e^t = Motional Compensated Prediction Error for Current Image = $I^t - f_\theta(\tilde{I}^{t-1})$

$\hat{e}^t$ = Quantized version of e^t

$f_\theta(\tilde{I}^{t-1})$ = Motion Compensated Prediction of Current Image

$\tilde{I}^t$ = Reconstructed Image for Current Frame = $f_\theta(\tilde{I}^{t-1}) + \hat{e}^t$

$\tilde{I}^{t-1}$ = Reconstructed Image for Previous Frame

θ = Motion Parameters

The quantization error q^t is computed using $q^t = e^t - \hat{e}^t$. The reconstruction error is given by $r^t = I^t - \tilde{I}^t$. Due to the feedback loop of the DPCM structure, the reconstruction

error is equal to the quantization error, accordingly $r^t = q^t$. This relationship holds provided there is no channel noise.

3.3 Application of the MDL Principle

The suitability of the Minimum Description Length (MDL) principle to motion-based segmentation and estimation was demonstrated by Zheng and Blostein [57]. In [57] motion-compensated object/region-based video coding based on the MDL principle was investigated, using ideal coding gain estimates. In this thesis, a computationally efficient codec algorithm is designed which demonstrates the bit-stream performance of a region-oriented approach utilizing an MDL cost function.

The MDL principle, originally developed by Rissanen [39], minimizes coding cost in the context of parameter estimation which makes it particularly well-suited for combined region segmentation, motion estimation, and coding. A region segmentation procedure guided by an MDL cost function allows us to inherently consider the optimum trade-off between motion field information and prediction error. Thus, the optimization of the cost function results in a minimization of the required bit-rate for a series of video frames in terms of prediction error, motion field and region boundaries.

In the estimation problem, we select a parametrically defined statistical model described by a probability density function $P_\theta(\mathrm{x})$ for the data **x** and attempt to estimate the vector parameter $\theta = (\theta_1,...,\theta_m)$ from observations. In the equations described in section 3.1.2, the data **x** represents a 2-D image array, or more specifically the 2-D array of motion compensated prediction error e^t, described by the probability density function $P_\theta(e^t)$. The parameters $\theta = (\theta_1,...,\theta_m)$ are motion model parameters, where *m* is the order

of the model. In the design presented in this thesis, the model order m is variable and is selected during an optimization process described in subsequent sections. For simplification it is sufficient for the time being to consider the model order fixed and the image $\tilde{I}^t$ consists of a single region encompassing all pixels. The aim of the MDL principle is to estimate the parameter vector θ such that the description length of the frame $DL(\tilde{I}^t)$ is the shortest among all possible estimates. The description length of $\tilde{I}^t$ is

$$DL(\tilde{I}^t) = DL(e^t) + DL(\theta) + DL(m) \tag{3.2}$$

where $DL(e^t)$ is the description length of the MCPE, $DL(\theta)$ is the description length of the motion parameters, and $DL(m)$ is the description length of the representation of the integer model order. In the case where the MCPE is quantized, e^t is simply replaced by $\hat{e}^t$.

In practice, we segment the image I^t into many regions that can be as complex as arbitrarily shaped objects to simple fixed block sizes. In [57] arbitrarily shaped image segmentations of objects are considered in order to show theoretical gain. In modern video coding standards such as H.263, simple fixed blocks are used. In this thesis, a trade-off between the two approaches is taken in order to realize significant coding gain, with a practical encoding algorithm. In [57], the proposed segmentation algorithm is decision-directed and is therefore difficult to parallelize. This work extends the concepts of [57] by developing a highly parallelizable sub-optimal image segmentation, and we evaluate the actual bit-stream performance of the codec, as opposed to the theoretical ideal coding length gain.

We consider N disjoint regions in the frame where $b(n)$ is the boundary of the nth region. The summation of all N disjoint regions results in the entire image I^t. That is,

$$\tilde{I}^t = \sum_{n=1}^{N} [f_{(\theta(n),b(n))}(\tilde{I}^{t-1}) + e^t_{b(n)}] \tag{3.3}$$

This leads directly to the separability of the MDL description length function, where the description length of the entire image $\tilde{I}^t$ may be expressed as the summation of the description lengths of N disjoint regions in a particular segmentation.

$$DL(\tilde{I}^t) = \sum_{n=1}^{N} [DL(e^t_{b(n)}) + DL(\theta(n)) + DL(m(n)) + DL(b(n))] \tag{3.4}$$

Note that a term expressing the description length of the boundary definitions $DL(b(n))$ has been added. The objective of the encoding algorithm is to minimize the total description length within a single region with respect to multiple model orders by using the MDL criteria to select motion parameters and model order. Secondly, it is a further objective to minimize equation (3.4) over all possible image segmentations. In this context the description length is considered an estimate of the eventual coding costs of individual regions, the sum of these costs over all N regions is the coding cost of the entire frame. This is particularly well suited in this application due to the final encoding stage of Figure 3.1 where entropy coding is used to generate bit-stream level codes by eliminating statistical redundancy.

3.4 Quadtree Optimization

We consider the minimization of the cost function as a discrete-state constrained optimization problem. Using this approach, it is possible to hypothesize region partitions and estimate motion parameters for each disjoint region. Due to the very large number of

possible image partitions it is necessary to develop constraints. Assuming 4-connected region boundaries, we consider the number of possible connections of pixels, leading to $2^{2N(N-1)}$ possible states for an *N*x*N* image, which is clearly an intractable problem. First we can constrain region primitives to be *M*x*M* square blocks rather than individual pixels which reduces the search to $2^{2\frac{N}{M}(\frac{N}{M}-1)}$ states.

As a practical alternative to [57], this thesis utilizes a multigrid structure to further reduce the search space. The region segmentation map is constrained to a quad-tree structure [42][46][26], an example of which is shown in Figure 3.2. This greatly reduces the search space involved for an exhaustive search. In this simplified case, the number of states is equal to the number of possible quad-trees which can be found via a recursive expression $S(l) = S(l-1)^4 + 1$ where $S(0) = 0$ and $l = 1,2,...,L$ levels. S represents the total number of possible states or tree configurations. An L = 5 level quad-tree (such as the current implementation) would thus have $S = 4.866 \times 10^{19}$states.

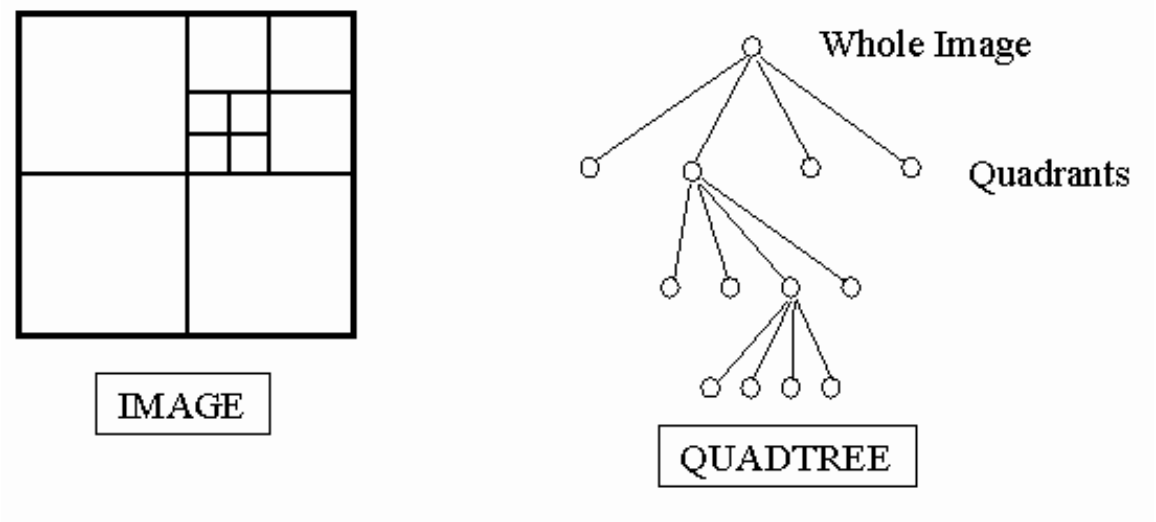


Figure 3.2: Quadtree Structure

An efficient quad-tree optimization algorithm has been developed in this thesis that exploits the separability characteristic of the MDL cost function. This permits the consideration of each disjoint image segment (defined by the quad-tree) independently to find the minimum with respect to image segmentation. The algorithm thus avoids local minima by first computing coding costs for each quad-tree node and then recursively merging child nodes into a single parent node. An optimal solution in terms of image segmentation is guaranteed by merging all child nodes at the finest grid level guided by the metric $DL(P) \leq DL(C1) + DL(C2) + DL(C3) + DL(C4)$ where $DL(P)$ and $DL(Cn)$ are the MDL description lengths for the parent node and child nodes respectively. This metric is evaluated for each node of the finest level and then the algorithm progresses to the next grid level until the top of the tree is reached. Using this procedure, the number of cost function evaluations is simply equal to the number of nodes of the full quad-tree for a fixed number of grid levels L. The number of nodes residing on each level l of the quadtree is 4^{l-1}. The total number of nodes for a full tree is the sum of the number of nodes residing at each level over all L levels, which is expressed as

$$C = \sum_{l=1}^{L} 4^{l-1} = 4^0 + 4^1 + 4^2 + \ldots + 4^{L-1} \tag{3.5}$$

This is readily recognized as a finite geometric sum leading to

$$C = \frac{4^L - 1}{3} \tag{3.6}$$

For a 5-level tree, this yields 341 cost function evaluations. The complexity of the algorithm is concentrated in C cost function evaluations, each consisting of a motion estima-

tion/adaptation stage and the computation of the MDL coding cost for the motion-compensated prediction error and motion parameters.

3.5 Optimization Algorithm

A fast algorithm has been developed to minimize the MDL cost function. The optimization algorithm consists of two distinct phases: 1) motion estimation and model order selection, and 2) quadtree merging. In Phase 1, the coding cost is calculated for each candidate node of a full quadtree. The coding cost for each node also requires a motion estimation/adaptation procedure, where the best motion model order is selected based on the coding cost. Differential correction is used in the motion estimation algorithm where motion parameters propagate up from the finest level ($l = L$) to the coarsest ($l = 1$) in order to initialize the motion estimation algorithm. At $l = L$, the finest level, motion parameters are initialized using block matching, similar to the H.263 video coding standard. Motion parameters are computed for all nodes in a fine-to-coarse fashion using differential correction. In Phase 2, merge/split decisions are made depending on which yields the least coding cost. The merge/split process begins at the next-to-finest grid level $l = L$-1 and working up towards the coarsest. Once the motion parameters and model orders are selected, this technique guarantees an optimal solution in the image segmentation, avoiding local minima. An optimal quadtree image segmentation solution is obtained using only $C = \frac{4^L - 1}{3}$ cost function evaluations within the constraints of the quadtree. A diagram describing the flow of the optimization algorithm is shown in Figure 3.3.

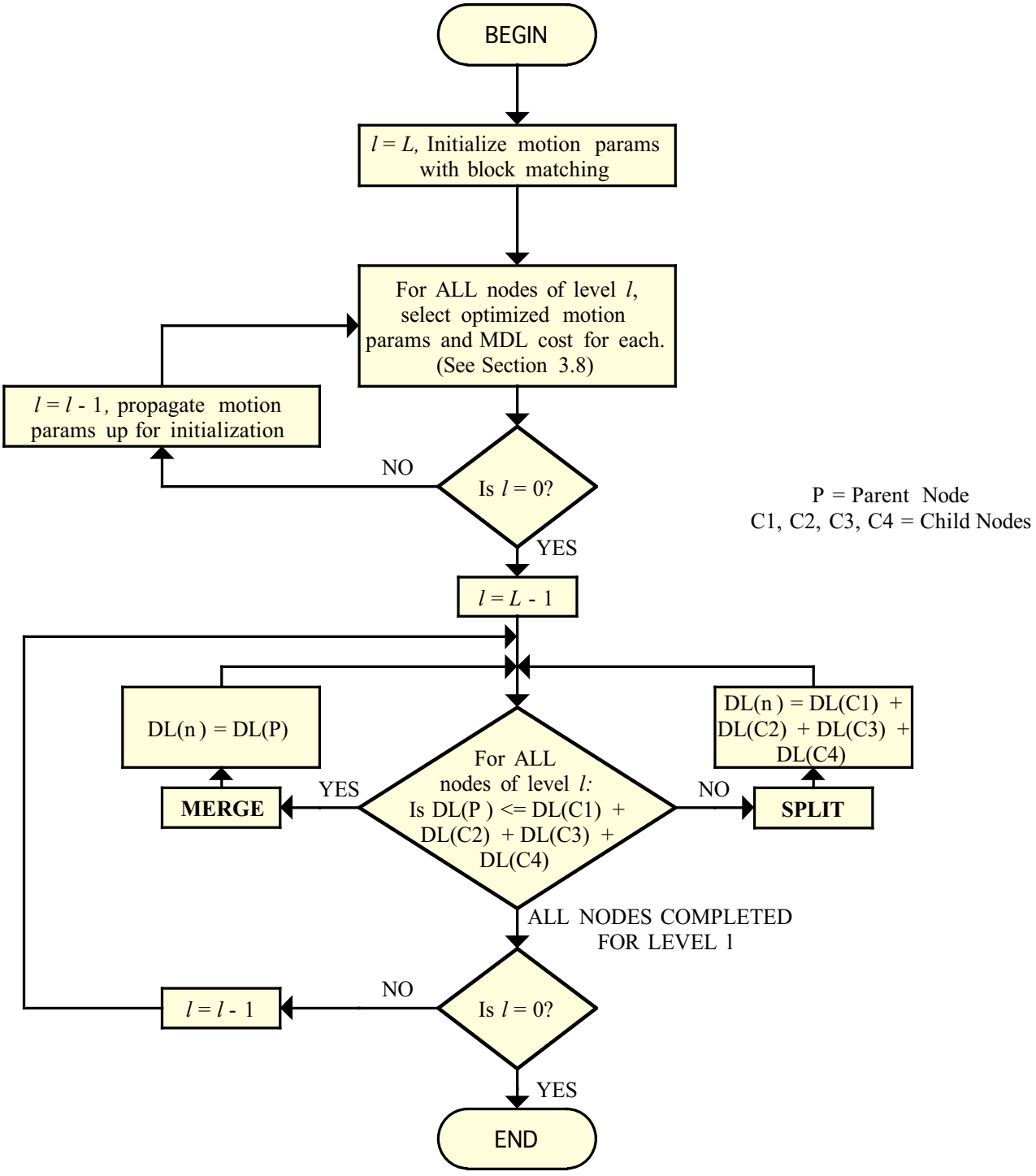


Figure 3.3: Minimization algorithm flowchart.

Note from Figure 3.3 that the optimization algorithm actually consists of two phases, 1) the computation of the optimized motion parameters and coding cost for all nodes in the quadtree, and 2) using the selected motion parameters and model orders for each node, a recursive split/merge procedure to eliminate segmentation choices that do not yield the best coding results. It is important to note that a global optimum is not found with respect to all possible model orders and image segmentations. The model order and parameters are selected for each node independently and prior to the quadtree merging process.

3.6 Statistical Model of the MCPE

In order to estimate the coding cost of the motion compensated prediction error (MCPE), a statistical model must be defined for the data. In section 3.9, the results will show that a memoryless source model is an accurate representation of the MCPE. By the very nature of the prediction process, the MCPE exhibits a characteristic distribution which allows its modeling as a Laplacian probability density function (PDF). Hence, an analytical expression can be derived to estimate the MDL coding cost. The Laplacian PDF

$$P(X = x) = \frac{1}{\sqrt{2}\sigma} e^{-\frac{\sqrt{2}|x|}{\sigma}} \tag{3.7}$$

where σ is the Laplacian standard deviation and x is a realization of the random variable X, has been shown to closely match the measured PDF of the MCPE [8][31].

Figure 3.4 and Figure 3.5 show the histograms obtained for MCPE results generated using our Quadtree-MDL codec. Figure 3.4 shows both the measured PDF for the MCPE

and the theoretical Laplacian PDF based on coding results from frame 730 of the "Mother-Daughter" standard test sequence. Similarly, Figure 3.5 shows the same set of data for frame 40 of the "Miss America" sequence. By comparing the curves, it is quite evident that the Laplacian PDF provides a good model of the measured PDF. Consequently, the Laplacian model is used in the calculation of the MDL coding cost.

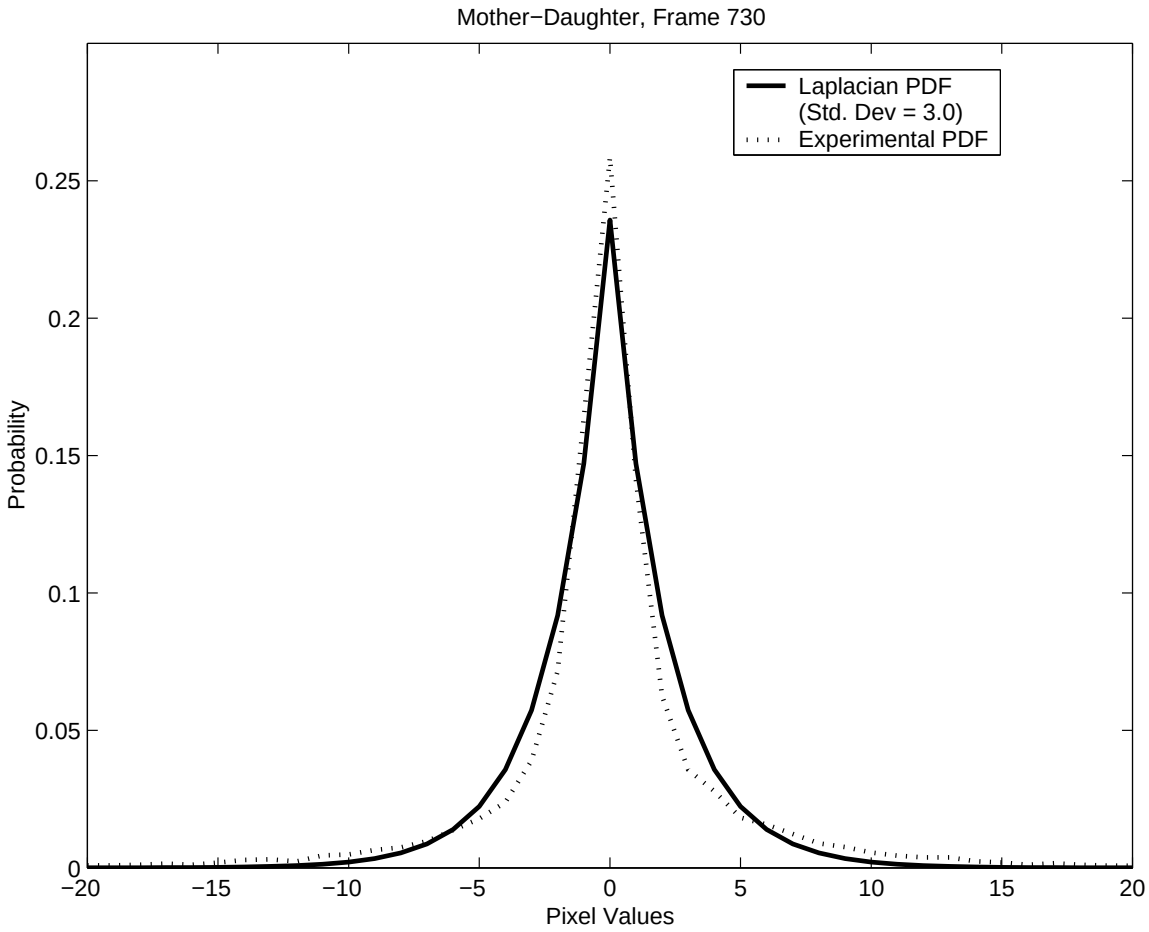


Figure 3.4: Measured and theoretical PDF for MCPE of "Mother-Daughter", frame 730

3.7 MDL Cost Function

The most challenging aspect of the derivation of the cost function is the component relating to the estimation of the coding cost of the MCPE, $DL(e^t_{b(n)})$ which corresponds to the first term in equation (3.2). Following the procedure of section 2.5,

$$DL(e^t_{b(n)}) = -log_2 P_\theta(e^t_{b(n)}) \tag{3.8}$$

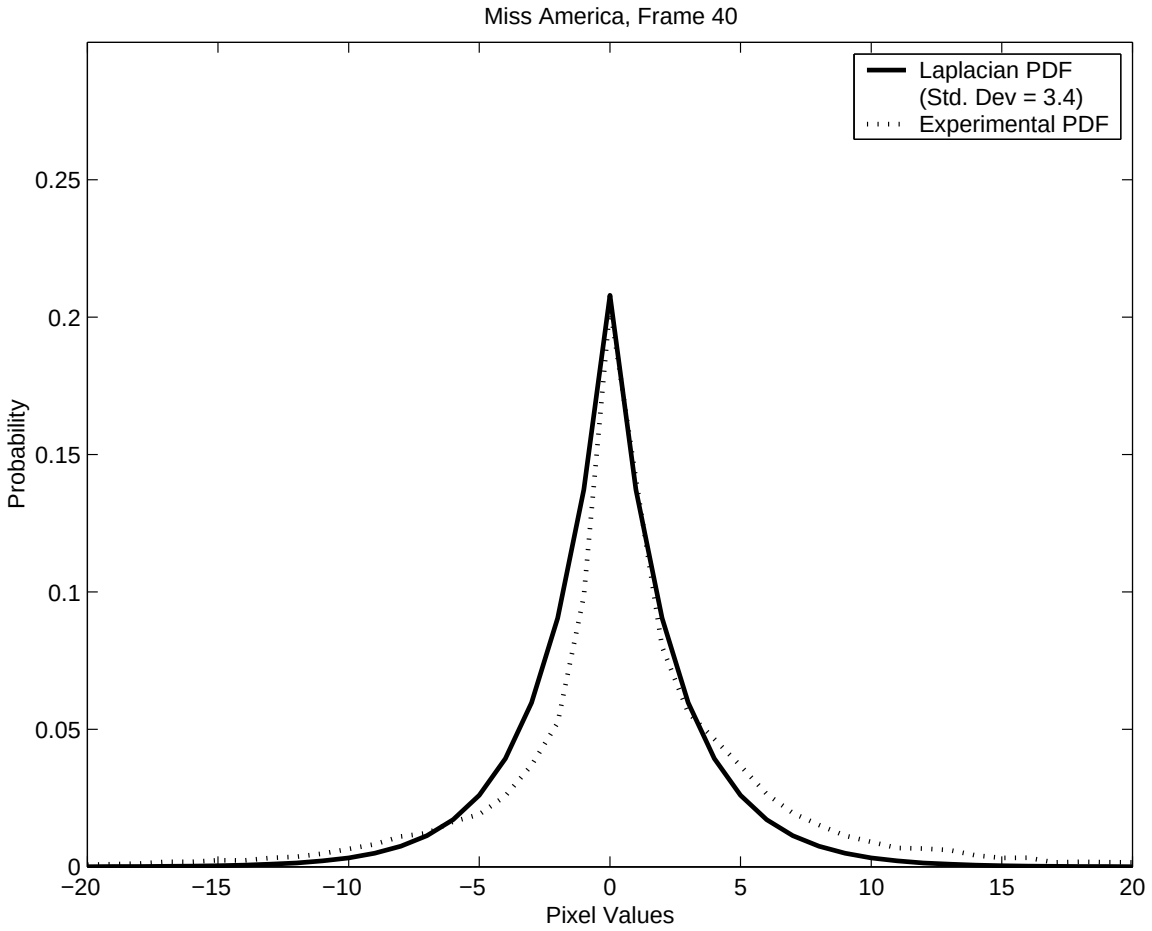


Figure 3.5: Measured and theoretical PDF for MCPE of "Miss America", frame 40

In a lossy compression scheme, higher compression is achieved by a coarse quantization of the MCPE pixel values. The choice of scalar quantization is greatly influenced

by the method used for bit-stream encoding. This work uses a robust procedure for designing optimum quantizers (in terms of rate-distortion) for variable-length encoding [55]. To model the MCPE signal, we use a memoryless Laplacian model which is verified in section 3.9. Following the procedure outlined by Wood in [55], the optimum quantizer for the Laplacian model using ideal entropy coding is the uniform quantizer, and this is used as our choice when selecting quantization parameters.

In the case of our codec, uniform quantization of the Laplacian distribution with a quantization step size ε is used. We seek to find the probability of occurance of each of the quantization levels. This represents the probability distribution of the quantized Laplacian PDF, with is used to model the quantized MCPE. $e^{t}_{b(n)}(x)$ represents the individual pixel values of the MCPE in a particular region, and the quantized Laplacian distribution is

$$P(u\varepsilon) = \begin{cases} \displaystyle\int_{u\varepsilon-\frac{\varepsilon}{2}}^{u\varepsilon+\frac{\varepsilon}{2}} \frac{1}{\sqrt{2}\sigma} e^{-\frac{\sqrt{2}|x|}{\sigma}} dx & \text{if } u = 0,\pm1,\pm2,... \\ 0 & \text{otherwise} \end{cases} \tag{3.9}$$

where $u\varepsilon$ are the individual values of the quantized MCPE $\hat{e}^{t}_{b(n)}$ and u is the quantization bin number. Equation (3.9) assumes a uniform quantizer with a mid-tread design as per Figure 2.6 with no dead zone. The u=0 case is treated as a separate integration, and after detailed manipulation we find

$$P(0) = 1 - e^{-\frac{\varepsilon}{\sqrt{2}\sigma}} \qquad u = 0 \tag{3.10}$$

$$P(u\varepsilon) = sinh\left[\frac{\varepsilon}{\sqrt{2}\sigma}\right] e^{-\frac{\sqrt{2}\varepsilon|u|}{\sigma}} \qquad |u| > 0 \tag{3.11}$$

We assume that the pixels are modeled as identically distributed independent variables and hence they are assumed to be uncorrelated. The joint PDF for the group of pixels within a particular region R_n is therefore $P_\theta(e^t_{b(n)}) = \prod_{R_n} P(u\varepsilon)$ yielding

$$P_\theta(e^t_{b(n)}) = \begin{Bmatrix} \prod_{R_n} 1 - e^{-\frac{\varepsilon}{\sqrt{2}\sigma}} & u = 0 \\ \prod_{R_n} sinh\left[\frac{\varepsilon}{\sqrt{2}\sigma}\right] e^{-\frac{\sqrt{2}\varepsilon|u|}{\sigma}} & |u| > 0 \end{Bmatrix} \tag{3.12}$$

The ideal coding length function of the quantized MCPE of each region according to equation (3.8) is thus

$$DL(e^t_{b(n)}) = \begin{Bmatrix} -log_2\left(\prod_{R_n} 1 - e^{-\frac{\varepsilon}{\sqrt{2}\sigma}}\right) & u = 0 \\ -log_2\left(\prod_{R_n} sinh\left[\frac{\varepsilon}{\sqrt{2}\sigma}\right] e^{-\frac{\sqrt{2}\varepsilon|u|}{\sigma}}\right) & |u| > 0 \end{Bmatrix} \tag{3.13}$$

Also,

$$DL(e^t_{b(n)}) = \begin{Bmatrix} -\sum_{R_n} log_2\left(1 - e^{-\frac{\varepsilon}{\sqrt{2}\sigma}}\right) & u = 0 \\ -\sum_{R_n} log_2\left(sinh\left[\frac{\varepsilon}{\sqrt{2}\sigma}\right] e^{-\frac{\sqrt{2}\varepsilon|u|}{\sigma}}\right) & |u| > 0 \end{Bmatrix} \tag{3.14}$$

Combining into one equation we have,

$$DL(e^t_{b(n)}) = -\sum_{\substack{\text{zero val.}\\ \text{pels in } R_n}} log_2\left(1 - e^{-\frac{\varepsilon}{\sqrt{2}\sigma}}\right) - \sum_{\substack{\text{non-zero val.}\\ \text{pels in } R_n}} log_2\left(sinh\left[\frac{\varepsilon}{\sqrt{2}\sigma}\right]\right) - \sum_{\substack{\text{non-zero val.}\\ \text{pels in } R_n}} \left(log_2 e^{-\frac{\sqrt{2}\varepsilon|u|}{\sigma}}\right) \quad (3.15)$$

where we have summations segmented into terms dealing with u=0 and $|u|$>0 valued pixels within the nth region in the image R_n. If the regions are rectangular in shape, as is the case with our quadtree segmentation, the number of pixels in a region is NxM. We can further specify a count for the number of zero valued pixels (after quantization) in the region as k. Therefore, there are (Nx$M - k$) non-zero valued pixels. A further simplification then becomes

$$DL(e^t_{b(n)}) = -k\, log_2\left(1 - e^{-\frac{\varepsilon}{\sqrt{2}\sigma}}\right) - (NM - k) log_2\left(sinh\left[\frac{\varepsilon}{\sqrt{2}\sigma}\right]\right) - \frac{\sqrt{2}\varepsilon}{\sigma \ln 2} \sum_{\substack{\text{non-zero val.}\\ \text{pels in } R_n}} |u| \quad (3.16)$$

A statistical model of each motion parameter is also required for estimation of the coding cost. The range for each motion model is ($-M_j$, $+M_j$) where $2M_j$ is the span of the jth motion parameter. The motion parameters are quantized with a step size of δ_j for each parameter which yields $\frac{2M_j}{\delta_j}$ levels. Assuming a uniform distribution for the motion parameters the coding length becomes

$$DL(\theta) = \sum_{j=1}^{m} \log_2 \frac{2M_j}{\delta_j} \quad (3.17)$$

where m is the model order chosen. The uniform model is essentially worst-case in terms of coding cost, and a more accurate model may be developed, such as Gaussian that would reflect the statistical tendency for motion parameters to be small. However, for

simplicity, we choose a uniform model for calculating coding cost. The choice of model orders can be 0, 2, 4, or 6 parameters and hence 2 bits per region are allocated for *DL(m)*, which again assumes a uniform probability distribution. Since the quadtree structure encoding is a very compact variable length code (1 for merge, 0 for split) its coding impact is relatively negligible in the context of equation (3.4).

3.8 Motion Estimation

To obtain high compression ratios in the coding of the motion vector fields while keeping a high prediction quality, an adaptive motion representation is incorporated in the motion estimation. A motion model hierarchy is used in a model validation stage consisting of 4 types of models: null (0 parameter), translational (2 parameter), simplified affine (4 parameter), and affine (6 parameter) [9][6]. This model hierarchy is incorporated in the cost function evaluation and the model leading to the lowest MDL coding cost is selected. The motion estimation is initialized with block matching at the finest level (level 5) and parameters are propagated in a bottom-up fashion using a median operator. Levels 1-4 are initialized with the greater of the model orders of the 4 corresponding child nodes.

The 6 parameter model used is referred to as the *affine* model and can be expressed as

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} a1 & + & a2 \cdot x & + & a3 \cdot y \\ a4 & + & a5 \cdot x & + & a6 \cdot y \end{pmatrix} \tag{3.18}$$

Where $(x, y)^T$ are the spatial coordinates. We can use more meaningful parameters and rewrite as

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} s_x \cdot \cos\theta_x & -s_y \cdot \sin\theta_y \\ s_x \cdot \sin\theta_x & s_y \cdot \cos\theta_y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} dx \\ dy \end{pmatrix} \tag{3.19}$$

Where s_x and s_y are the scaling ratios in the x and y directions, θ_x and θ_y are the rotation angles around the x and y axis, and d_x and d_y are the two components of a translation vector. With the affine model rotation, zoom, and nonrigid body motion (such as sheer motion) can be described. We can simplify the affine model to 4 parameters, called the *simplified affine* model by constraining $s_x = s_y = s$ and $\theta_x = \theta_y = \theta$. The model is then written as

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} s \cdot \cos\theta & -s \cdot \sin\theta \\ s \cdot \sin\theta & s \cdot \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} dx \\ dy \end{pmatrix} \tag{3.20}$$

We can further simplify the model to the *translational* model which defines motion of an entity by a translational vector

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} dx \\ dy \end{pmatrix}. \tag{3.21}$$

The translational model assumes that complex motion can be approximated, under certain conditions, by a sum of infinitesimal translations. However, this model has some limits and is not able to cope with complex scenes.

The motion estimation is performed by minimizing the MDL cost function of equation (3.2) with respect to the motion parameters for a given model order. The motion parameters are estimated using the pixels contained within the region boundaries and the ideal coding length of the region is calculated. Upon inspection of equation (3.16) it is clear that minimization with respect to motion parameters involves minimization of the

term $\sum_{\substack{\text{non-zero val.}\\ \text{pels in } R_n}} |u|$ which leads to a sum-of-absolute-differences or minimum-absolute-error criteria. It is interesting to note that this minimization is equivalent to the familiar maximum likelihood estimation. To simplify the calculation, we can consider the estimation using un-quantized MCPE values rather than the quantized values, which is equivalent in terms of the minimization process. Since θ is related only to the image data, we estimate θ and then calculate the MCPE variance σ^2 for the region resulting from our motion parameter estimates. The variance is then used to calculate the coding cost.

Since the minimum-absolute-error expression is not differentiable, we treat the problem as a non-linear optimization problem. The iterative Hooke and Jeeves [15] algorithm is used for motion estimation in a similar manner as used in generic non-linear optimization problems. The Hooke and Jeeves algorithm allows us to directly minimize the MDL cost function by performing a systematic direct search of the function. This method is particularly desirable due to its flexibility and the fact that neither differentiable nor continuous functions are required. This method, known as the pattern search, is based on a sequence of exploratory and pattern moves, starting at an initial base point. A local exploration commences with the evaluation of the cost function at an initial base point, and two other points removed from it by a predefined step size. If one of the points results in a decrease in the cost function, a success is said to result, and the particular point that produced the success is called a temporary base point. If neither of the two points produces a success, the step size for that variable is reduced by half and the exploration is repeated.

The C source code is based on [18] and adapted from the Algol pseudocode found in [19]. The implementation also includes the improvements suggested in [3] and [50]. The

Hooke and Jeeves algorithm is supplied with: a) a subroutine that computes the sum-of-absolute-differences cost for a given set of motion parameters (which depends on the model order under consideration), b) an initial starting guess (where motion parameters propagate up from the finest level ($l = L$) to the coarsest ($l = 1$)) of the minimum point, and c) values for the algorithm convergence parameters. Then the program searches for a local minimum, beginning from the starting guess, using the direct search algorithm of Hooke and Jeeves [15]. The algorithm works by taking "steps" from one estimate of a minimum, to another (hopefully better) estimate. Taking big steps gets to the minimum more quickly, at the risk of "stepping right over" an excellent point. The stepsize is controlled by a user-supplied parameter called λ. At each iteration, the stepsize is multiplied by λ ($0 < \lambda < 1$), so the stepsize is successively reduced. Larger values of λ give greater probability of convergence, at a cost of more function evaluations. Smaller values of λ reduces the number of evaluations (and the program running time), but increases the risk of non-convergence. Experiments were conducted with a λ value of 0.3, which was found to be acceptable by empirical observation. The stepsize is reduced until it is equal to (or smaller than) η, which was set to 0.001. So the number of iterations performed by Hooke-Jeeves is determined by λ and η.

3.9 Motion-compensated prediction

The motion-compensated prediction error (MCPE) is quantized directly and entropy coded. Spatial decorrelation techniques such as the DCT are not employed due to the statistical properties of the MCPE [11][44]. It was shown in [8] that the MCPE cannot be

modeled by a first-order stationary Markov process. Since the inter-pixel correlation of the MCPE is low, the justification for using the DCT is questionable.

Table 3.1 compares typical values of energy and the correlations ρ_1 (first order correlation) and ρ_2 (second order correlation) of the original images and the motion compensated prediction error images [8]. The measurements were carried out on the video test sequences "Mobile Calendar", "Flower Garden", and "Table Tennis".

		Energy	ρ_1	ρ_2
Mobile Calendar	- Original	16201.56	0.96	0.94
	- MCPE	339.85	-0.11	-0.12
Table Tennis	- Original	15573.33	0.99	0.99
	- MCPE	28.37	0.18	-0.02
Flower Garden	- Original	36741.32	0.99	0.98
	- MCPE	227.06	0.34	-0.03

Table 3.1: Correlation results for MCPE of various sequences.

It is clear that conventional intraframe coding methods, such as transform techniques, which aim at decorrelating the pixels, perform poorly when applied to the MCPE. The results also verify that a high order statistical model of the MCPE is not required and our memoryless source model is sufficient to yield a reasonable approximation of the coding cost.

3.10 Lossless Bitstream Coding

The coding of quantized MCPE pixels is achieved using adaptive arithmetic coding [54], initialized using the Laplacian source model mentioned above. We note that no side information is required for updating the model. However, it is necessary to transmit a small amount of information to signify the motion model order used as well as the segmentation map for the quad-tree which is coded with an efficient variable length code

represented by a series of bits which indicate termination of a leaf with a “0” and branching into child nodes by a “1”. To avoid error propagation, the model is reset frequently. The same arithmetic coder, initialized with a uniform model, is used to code the motion parameters.

Chapter 4

Experimental Results

4.1 Introduction

In order to evaluate the performance of the algorithm, a simulation environment was developed to code and decode standard test sequences in CCIR 601 QCIF 176X144 format. In order to compare the performance of video coding schemes, the following comparisons can be made: the reconstructed image quality for a given bit-rate, or conversely the bit rate for a given reconstructed image quality. Ideally, the quality of the reconstructed sequences should be estimated by subjective quality tests. However, in practice, it is evaluated using the peak-signal-to-noise-ratio (PSNR) due to the difficulties associated with subjective tests. The PSNR is defined as

$$PSNR = 10\log_{10}\left(\frac{255^2}{\frac{1}{N}\sum_{i-1}^{N}(x_i - \tilde{x}_i)^2}\right). \tag{4.1}$$

where x_i are the samples of the original signal and $\tilde{x}_i$ are the samples of the reconstructed signal. It is key to point out that the PSNR is sometimes a poor measure of the visual

quality and it is widely used in video coding due to the lack of perceptually reliable visual quality measures. Recently, significant advancements have been made in the evaluation of subjective video quality [53]; however, evaluation tools are not widely available to the academic community.

Software simulation code was written in ANSI C and carried out on Sun Ultra workstations and more recently on an Apple Powerbook G4 personal notebook computer. The simulation environment was developed under a Unix based operating system (and Mac OS X) and compiled using the GNU gcc compiler.

4.2 Software Configuration

A 5-level quad-tree was used with the following block sizes:

- Level 1 – 176X144 (1 block)
- Level 2 – 88X72 (4 blocks)
- Level 3 – 44X36 (16 blocks)
- Level 4 – 22X18 (64 blocks)
- Level 5 – 11X9 (256 blocks)

The quantization step size ε for motion-compensated prediction error coding was selected at run time based on the desired target bit-rate. In essence, this is equivalent to fixing the distortion parameter and optimizing the MDL description length. This can also be viewed as a global bit-rate minimization subject to distortion constraints that are fixed by the quantization step size. All translational motion parameters were represented to 1/2 pixel resolution and rotational and scaling parameters (4 and 6 order models) are quantized with a step size of 1/16. The luminance component of the video sequence is coded

as per the description in Chapter 3, where motion estimation is performed on the previously reconstructed frame and the current original frame. The color components were omitted from the simulation to allow for more flexibility. This permits a straightforward process in making system changes with little detriment to the validity of key underlying principles being tested. Similarly, the color components are subtracted from any comparison simulations used to measure performance against the Quadtree-MDL algorithm.

4.3 Comparison Approach

The result of the optimal Quadtree-MDL algorithm is compared with that of the state-of-the-art international coding standard ITU-T H.263 [48]. The H.263 codec used for direct comparison was provided by Telenor Research and the most recent version available was used for bitstream encoding, which corresponds to Test Model 5 (see Section 2.2.7). Still images of the decoded frames of each codec are compared as well as video sequence movies were compiled to demonstrate the differences in quality versus bit-rate performance of each codec.

4.4 Simulation Results

4.4.1 “Mother-Daughter” Sequence

The luminance component of the ``Mother-Daughter'' sequence was coded at a frame rate of 10 frames/sec for a bit-rate of approximately 15 kb/s. This corresponds to a rate which is very compatible for two-visual visual communications using ordinary telephone lines, which is in fact the primary application of the H.263 codec used for comparison. The

"Mother-Daughter" sequence consists of 960 frames corresponding to approximately 30 seconds of full motion video. This sequence was chosen because of its complex motion and multiple moving objects and serves the basis of a detailed comparison used to evaluate the advantages of the Quadtree-MDL approach.

Figure 4.1a) and b) shows the bit-rate results at 15 kb/s for the "Mother-Daughter" sequence using Quadtree-MDL and H.263 showing the bit consumption of each frame as the algorithms progress through the entire sequence. Figure 4.2 shows the PSNR comparison curves for the Quadtree-MDL algorithm versus H.263 for all frames in the sequence. The mean PSNR for the sequence using the Quadtree-MDL algorithm was 36.94 dB versus 33.92 dB for H.263. This represents an average improvement of 3.02 dB.

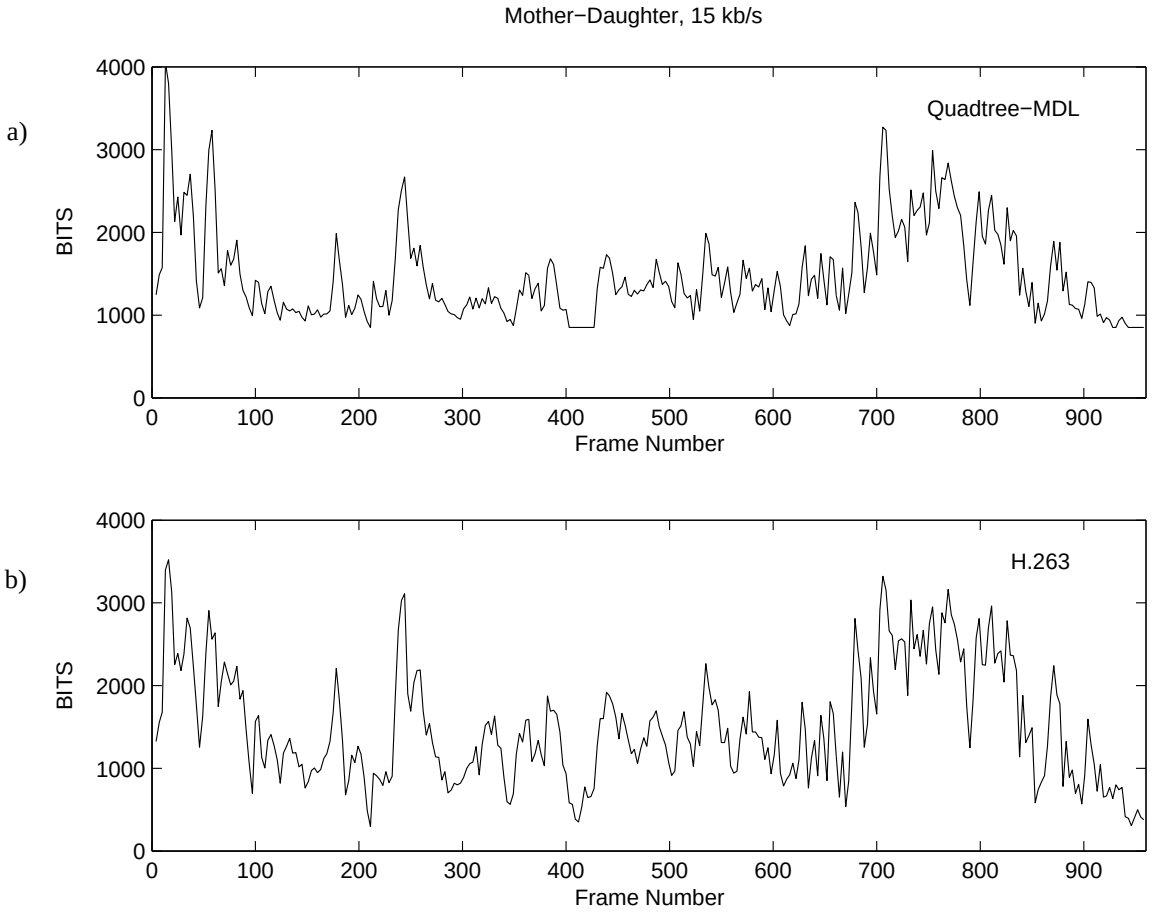


Figure 4.1: Bit-rate results for a) Quadtree-MDL and b) H.263 algorithms.

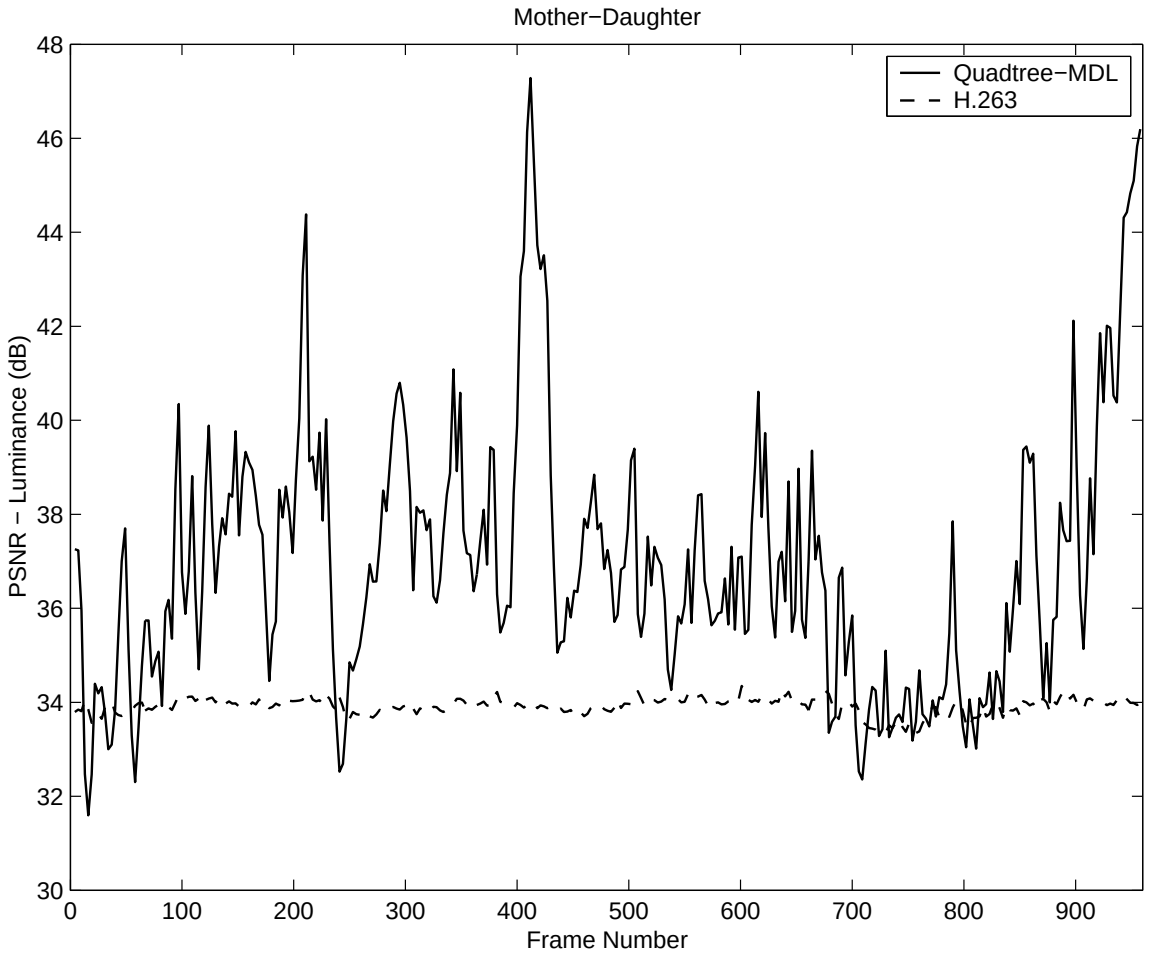


Figure 4.2: PSNR comparisons for “Mother-Daughter” using Quadtree-MDL and H.263.

Four separate frames of the “Mother-Daughter” sequence are used for comparison in order to give a good cross-section of results, as well as a sense for each algorithm’s strengths and weaknesses. The decoded images for frame 268 for the Quadtree-MDL and H.263 algorithms are shown in Figure 4.3. For frame 268, the decoded result using the Quadtree-MDL algorithms yields a PSNR of 36.94 dB versus 33.70 dB for H.263. Frame 268 represents the average performance of Quadtree-MDL. Note that the PSNR results correspond to the mean PSNR for the entire sequence of 36.94 dB.

a) Mother-Daughter, Frame 268
Original Image

b) Mother-Daughter, Frame 268
Decoded Image - Quadtree-MDL
15 kb/s, PSNR = 36.94 dB

c) Mother-Daughter, Frame 268
Decoded Image - H.263
15 kb/s, PSNR = 33.70 dB

Figure 4.3: Original and decoded images for frame 268 of "Mother-Daughter".

Absolute-error images are shown for each frame in Figure 4.4. The error images shown are revealing in that they show the differences in the nature of the distortion introduced by the Quadtree-MDL and H.263 algorithms. It is clear that the distortion related to Quadtree-MDL is concentrated along moving edges in the scene, while H.263 exhibits more evenly distributed distortion throughout the entire image.

a) Mother-Daughter, Frame 268
Error Image - Quadtree-MDL
15 kb/s, PSNR = 36.94 dB

b) Mother-Daughter, Frame 268
Error Image - H.263
15 kb/s, PSNR = 33.70 dB

Figure 4.4: Error Images for decoded frame 268 of "Mother-Daughter".

In Figure 4.5, Figure 4.6, Figure 4.7, Figure 4.8, Figure 4.9, and Figure 4.10, decoded frames from each algorithm are compared in a similar fashion. The figures correspond to frames 16, 412, and 736 respectively. Frame 16 represents worst-case results of the Quadtree-MDL algorithm in the sense that the PSNR is the lowest in the entire sequence, at 31.59 dB. Frame 412 represents the best-case results of the Quadtree-MDL algorithm where the PSNR is the highest in the sequence, at 47.28 dB. Finally, frame 736 was chosen because the PSNR of the Quadtree-MDL and H.263 algorithms are equal, 33.45 dB, and serves as a good comparison point for subjective quality differences. Table 4.1 shows a summary of the coding results for the selected frames of the "Mother-Daughter" sequence.

	Quadtree-MDL	**H.263**	**Condition**
Frame 268	36.94 dB	33.70 dB	Average
Frame 16	31.59 dB	33.85 dB	Worst-case
Frame 412	47.28 dB	33.96 dB	Best-case
Frame 736	33.45 dB	33.45 dB	Equal

Table 4.1: Coding results of selected frames from "Mother-Daughter" sequence, 15 kb/s.

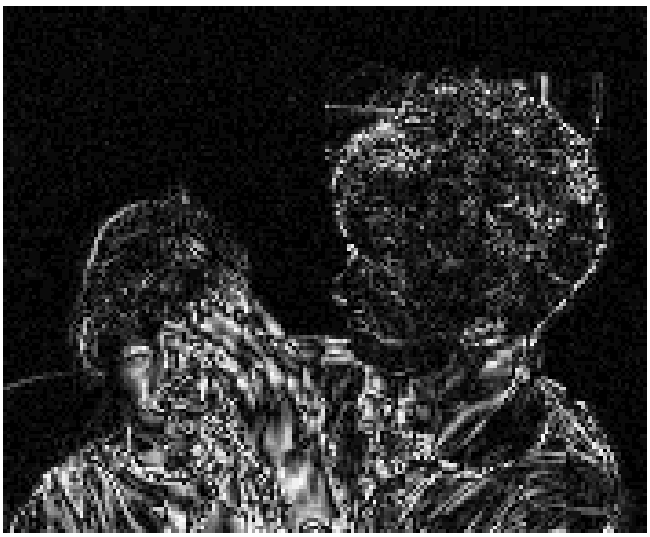
a) Mother-Daughter, Frame 16
Original Image

b) Mother-Daughter, Frame 16
Decoded Image - Quadtree-MDL
15 kb/s, PSNR = 31.59 dB

c) Mother-Daughter, Frame 16
Decoded Image - H.263
15 kb/s, PSNR = 33.85 dB

Figure 4.5: Original and decoded images for frame 16 of "Mother-Daughter".

a) Mother-Daughter, Frame 16
Error Image - Quadtree-MDL
15 kb/s, PSNR = 31.59 dB



b) Mother-Daughter, Frame 16
Error Image - H.263
15 kb/s, PSNR = 33.85 dB

Figure 4.6: Error Images for decoded frame 16 of "Mother-Daughter".

a) Mother-Daughter, Frame 412
Original Image

b) Mother-Daughter, Frame 412
Decoded Image - Quadtree-MDL
15 kb/s, PSNR = 47.28 dB

c) Mother-Daughter, Frame 412
Decoded Image - H.263
15 kb/s, PSNR = 33.96 dB

Figure 4.7: Original and decoded images for frame 412 of "Mother-Daughter".

a) Mother-Daughter, Frame 412
Error Image - Quadtree-MDL
15 kb/s, PSNR = 47.28 dB

b) Mother-Daughter, Frame 412
Error Image - H.263
15 kb/s, PSNR = 33.96 dB

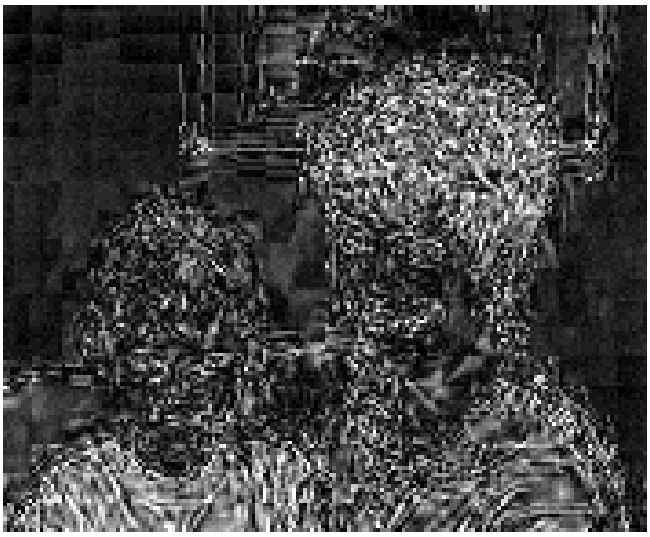


Figure 4.8: Error Images for decoded frame 412 of "Mother-Daughter".

a) Mother-Daughter, Frame 736
Original Image

b) Mother-Daughter, Frame 736
Decoded Image - Quadtree-MDL
15 kb/s, PSNR = 33.45 dB

c) Mother-Daughter, Frame 736
Decoded Image - H.263
15 kb/s, PSNR = 33.45 dB

Figure 4.9: Original and decoded images for frame 736 of "Mother-Daughter".

a) Mother-Daughter, Frame 736
Error Image - Quadtree-MDL
15 kb/s, PSNR = 33.45 dB

b) Mother-Daughter, Frame 736
Error Image - H.263
15 kb/s, PSNR = 33.45 dB

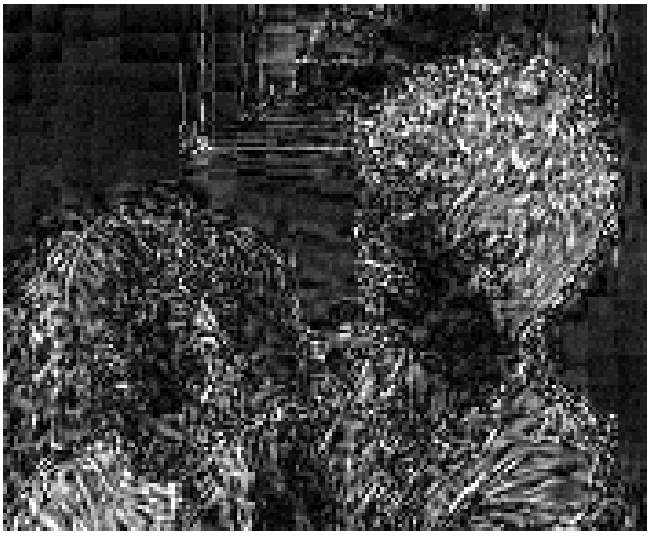


Figure 4.10: Error Images for decoded frame 736 of "Mother-Daughter".

A magnified section of “Mother-Daughter” is shown in Figure 4.11 showing more clearly the nature of distortion introduced by each algorithm for frame 736. Note the “salt and pepper” distortion in Figure 4.11(a) and the sharpness of the remaining regions. In Figure 4.11(b), the distortion it evenly distributed through the image causing an overall blurring effect as well as blocking artifacts.

a) Mother-Daughter, Frame 736
ZOOM - Quadtree-MDL
15 kb/s, PSNR = 33.45 dB

b) Mother-Daughter, Frame 736
ZOOM – H.263
15 kb/s, PSNR = 33.45 dB

Figure 4.11: Magnified section of "Mother-Daughter" frame 736

4.4.2 “Suzie” Sequence

The luminance component of the “Suzie” sequence was coded at a frame rate of 10 frames/sec yielding a bit-rate of approximately 10 kb/s. The “Suzie” sequence consists of 150 frames and contains a single moving object with very rapid motion. Figure 4.12

shows the PSNR comparison curves for the Quadtree-MDL algorithm versus H.263 for all frames in the sequence. The mean PSNR for the sequence using the Quadtree-MDL algorithm was 34.62 dB versus 32.73 dB for H.263. This represents an average improvement of 1.89 dB. The decoded images for frame 16 for the Quadtree-MDL and H.263 algorithms are shown in Figure 4.13. For frame 16, the decoded result using the Quadtree-MDL algorithms yields a PSNR of 34.65 dB versus 32.74 dB for H.263. Frame 16 represents the average performance of Quadtree-MDL. The PSNR results correspond closely to the mean PSNR for the entire sequence of 34.62 dB. Absolute error images for the "Suzie" sequence are shown in Figure 4.14.

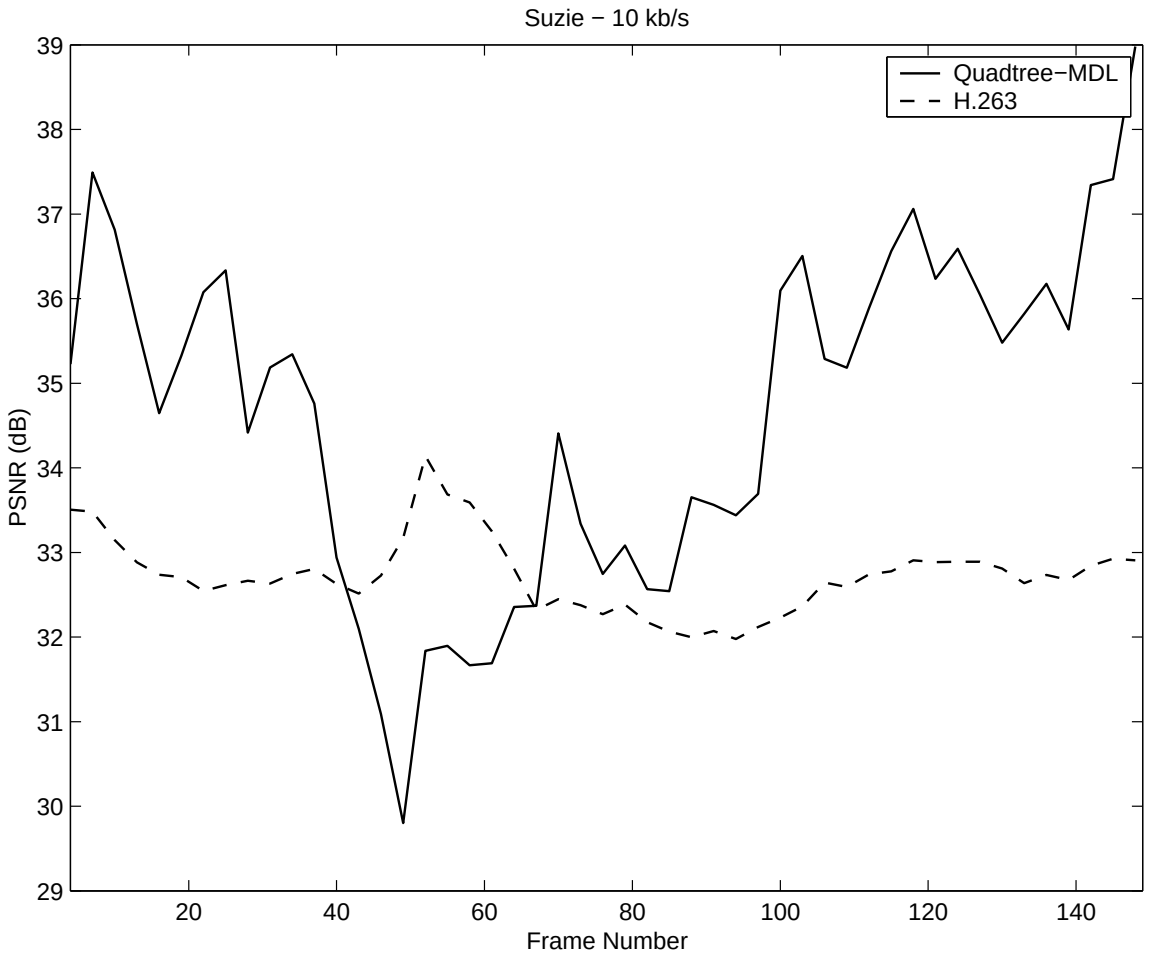


Figure 4.12: PSNR comparisons for "Suzie" using Quadtree-MDL and H.263.

a) Suzie, Frame 16
Original Image

b) Suzie, Frame 16
Decoded Image - Quadtree-MDL
10 kb/s, PSNR = 34.65 dB

c) Suzie, Frame 16
Decoded Image - H.263
10 kb/s, PSNR = 32.74 dB

Figure 4.13: Original and decoded images for frame 16 of "Suzie".

a) Suzie, Frame 16
Error Image - Quadtree-MDL
10 kb/s, PSNR = 34.65 dB

b) Suzie, Frame 16
Error Image - H.263
10 kb/s, PSNR = 32.74 dB

Figure 4.14: Error Images for decoded frame 16 of "Suzie".

4.4.3 “Miss America” Sequence

The luminance component of the “Miss America” sequence was coded at a frame rate of 10 frames/sec yielding a bit-rate of approximately 7.5 kb/s. The “Miss America” sequence consists of 150 frames and contains a single moving object with slow, simple motion. Figure 4.15 shows the PSNR comparison curves for the Quadtree-MDL algorithm versus H.263 for all frames in the sequence. The mean PSNR for the sequence

using the Quadtree-MDL algorithm was 39.56 dB versus 37.09 dB for H.263, for an average improvement of 2.47 dB. The decoded images for frame 43 for the Quadtree-MDL and H.263 algorithms are shown in Figure 4.16. For frame 43, the decoded result using the Quadtree-MDL algorithms yields a PSNR of 39.42 dB versus 37.24 dB for H.263. Frame 43 represents the average performance of Quadtree-MDL. The PSNR results correspond closely to the mean PSNR for the entire sequence of 39.56 dB. Absolute error images for the "Miss America" sequence are shown in Figure 4.17. Finally, a typical quadtree segmentation with selected motion model orders is shown in Figure 4.18

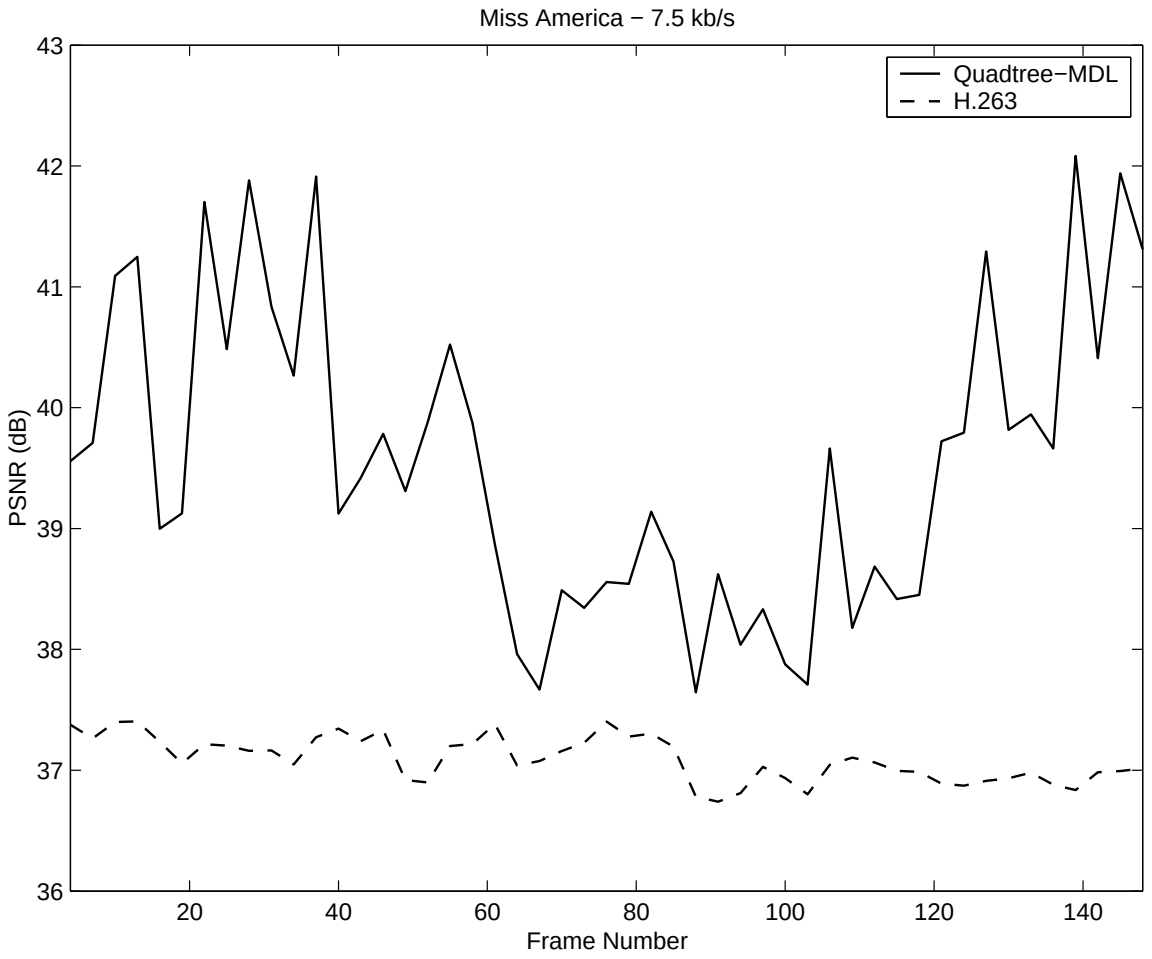


Figure 4.15: PSNR comparisons for "Miss America" using Quadtree-MDL and H.263.

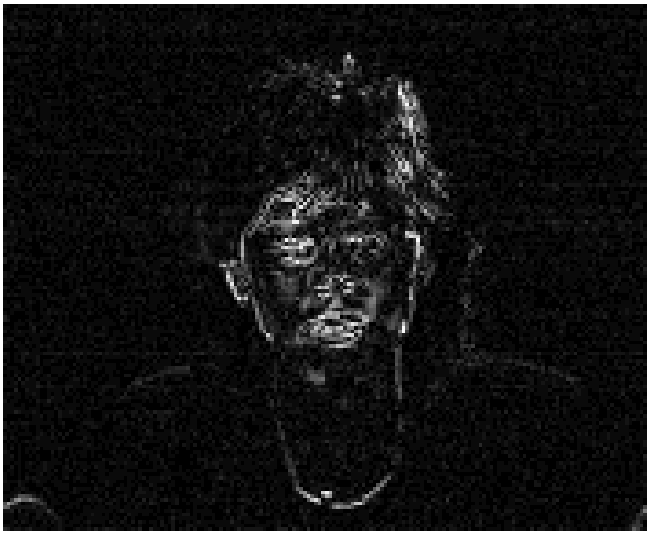
a) Miss America, Frame 43
Original Image

b) Miss America, Frame 43
Decoded Image - Quadtree-MDL
7.5 kb/s, PSNR = 39.42dB

c) Miss America, Frame 43
Decoded Image - H.263
7.5 kb/s, PSNR = 37.24 dB

Figure 4.16: Original and decoded images for frame 43 of "Miss America".

a) Miss America, Frame 43
Error Image - Quadtree-MDL
7.5 kb/s, PSNR = 39.42 dB



b) Miss America, Frame 43
Error Image – H.263
7.5 kb/s, PSNR = 37.24 dB

Figure 4.17: Error Images for decoded frame 43 of "Miss America".

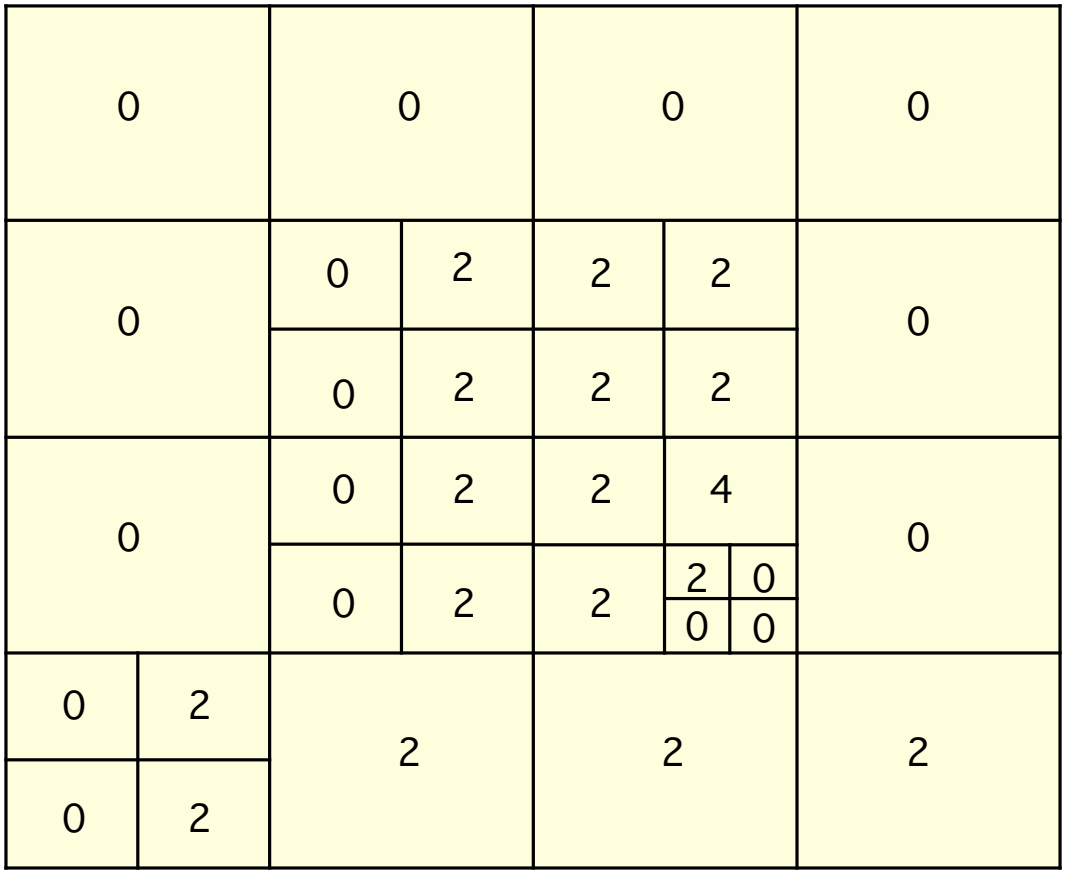


Figure 4.18: Typical quadtree decomposition and selected motion model orders for Miss America, Frame 73

4.5 Cost Function Validation

The accuracy of the MDL cost function plays an important role in ensuring the region segmentation procedure leads to an optimal solution. Therefore the result of equation (3.4) during the encoding process was compared with the actual coding length achieved using arithmetic coding at the bitstream level. The theoretical and actual coding lengths are shown in Figure 4.19 for each frame of the "Miss America" sequence. The MDL cost function tracks quite closely to the actual with excellent proportionality. The mean coding length for the MDL cost function was 633 bits per frame, while the actual coding

length had a mean of 613 bits per frame. The result of the MDL cost function is within 3.3%, on average, of the actual coding length obtained. Some differences can be expected due to the approximation of the Laplacian PDF used to model the MCPE as well as inaccuracies in estimating the variance of each region.

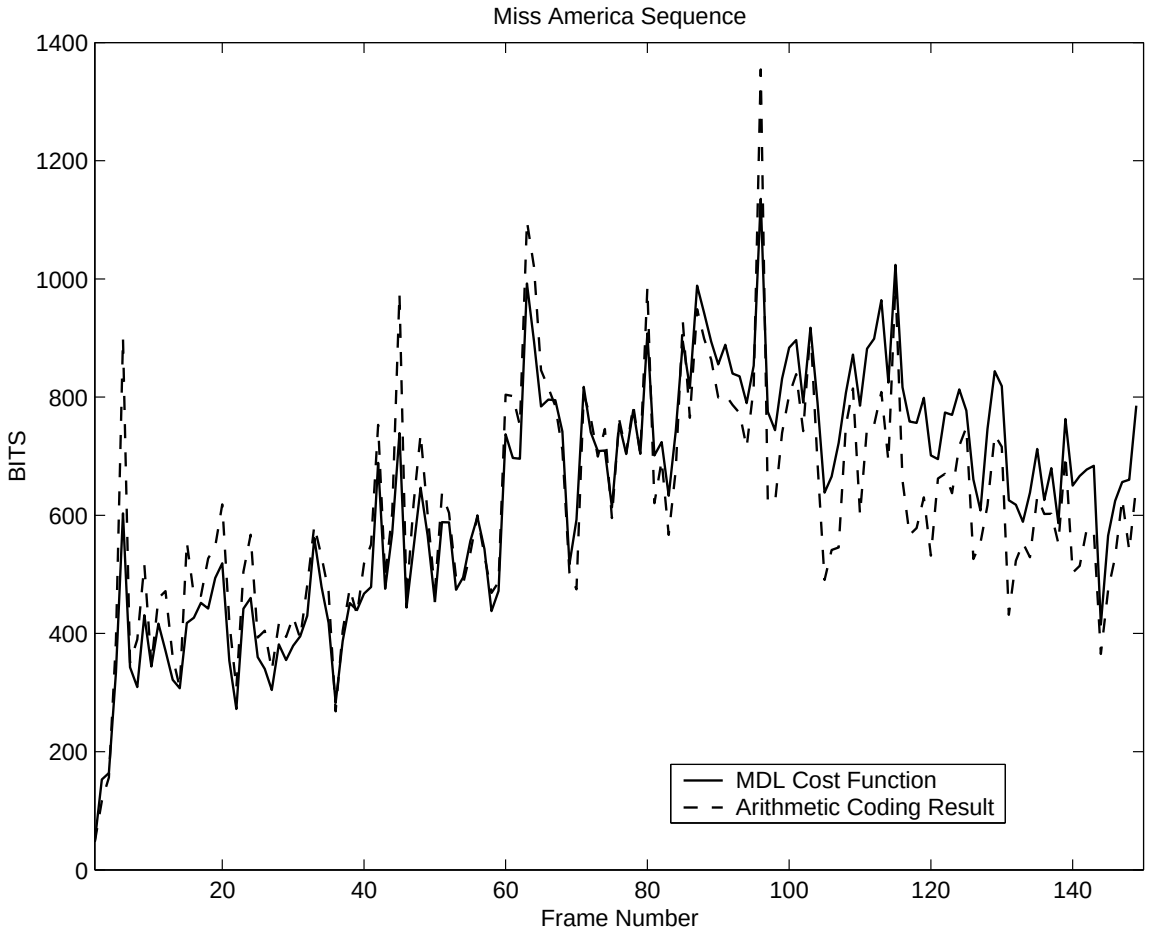


Figure 4.19: "Miss America" sequence; comparison of MDL cost function estimate versus actual coding costs from arithmetic coding.

4.6 Artifacts and Loop Filtering

It can be observed from Figure 4.11 that the nature of the distortion left by the Quadtree-MDL algorithm is quite different from that of H.263. Due to the DCT used in the H.263 algorithm, distortion permeates the entire scene and at low bit rates, the blocking artifacts and "mosquito" effects can be quite prominent [36][43]. The distortion related to Quadtree-MDL is primarily concentrated on moving edges resulting in "salt and pepper" distortion. The background regions and areas where there is little motion are reconstructed with excellent quality as compared to the original. Even when the PSNR is the same for each algorithm, there are marked differences in the visual appearance of each. The Quadtree-MDL results in sharper video, yielding more detail while H.263 has a blurred, blocky look. However, in some respects, the "salt and pepper" distortion of Quadtree-MDL can be more noticeable due to the rapid change in pixel value (high frequency spikes or impulse noise).

It is important to note that it is a relatively simple task to combat the impulse noise created by the Quadtree-MDL encoding by using the proper filtering technique. Because the filtering would be carried out during the encoding process, it becomes part of the structure of the encoder itself, and is therefore referred to as a *loop filter* [34][5]. In order to verify the potential of such a filter, experiments were conducted on the "Mother-Daughter" sequence. It was found that a median filter was most effective. The median filter is often used to remote "shot" noise, pixel dropouts and other spurious features of single pixel extent while preserving overall image quality [17][23]. In contrast, low pass filters would only blur the noise instead of removing it. Furthermore, a mask can be used to prevent filtering areas of the image unnecessarily. By using a mask, image sharpness

can be maintained for background areas and areas with low motion. An efficient mask can be constructed by using a threshold on the MCPE. This allows us to perform region-of-interest filtering in areas where the prediction error is large which corresponds well to areas where impulse noise is likely to occur. A 3X3 median filter was implemented and applied to a region-of-interest mask. The unfiltered image corresponding to frame 736 of "Mother-Daughter" sequence in shown in Figure 4.20(a). The region-of-interest mask is shown in Figure 4.20(b) that corresponds to areas with large prediction error. The result of the median filter is shown in Figure 4.20(c). A PSNR improvement of 0.27 dB was observed, but it is evident that the improvement in perceptual quality is far greater. The "salt and pepper" noise is much less noticeable while the rest of the image retains its sharpness.

a) Mother-Daughter, Frame 736
Decoded Image – Quadtree-MDL
15 kb/s, PSNR = 33.45 dB

b) Mother-Daughter, Frame 736
Region-Of-Interest Mask

c) Mother-Daughter, Frame 736
3X3 Median Filtered Image
15 kb/s, PSNR = 33.72 dB

Figure 4.20: Results of median filter on frame 736 of "Mother-Daughter".

4.7 Conclusions

The potential improvements using the concepts presented in this thesis were demonstrated by comparing the performance versus the state-of-the-art H.263 encoder. An average quality improvement in terms of PSNR of 2-3 dB can be obtained depending on the scene content. It was also shown that the image sharpness of the Quadtree-MDL algorithm is much improved for background and low motion areas. Impulse noise is typical with Quadtree-MDL at low bit rates, which can be quite noticeable. The impulse noise can be filtered using a median filter resulting in much improved perceptual quality.

It is important to note that the H.263 encoder undergoes constant improvement as the standardization body engages in a continuous improvement program. The comparisons in this thesis are based on the most recent software code available for academic use, TMN5, whereas the most recent test model is TMN11 [49], and the results obtained from TMN11 are sure to be significantly improved. However, it is equally important to note that H.263 contains a large number of small optimizations based on the experiences of those participating in the standardization process. The work presented in this thesis represents the validation of theoretical concepts as opposed to an attempt to implement a commercially viable encoder. Therefore, it can be expected that as the Quadtree-MDL algorithm matures similar incremental improvements can be achieved.

Chapter 5

Summary and Conclusions

This chapter summarizes the material in the previous chapters. This research proposes a novel video coding algorithm which can be viewed as an extension of state-of-the-art video coding standards. The objective was to design and test a bitstream codec in order to verify the feasibility of the proposed approach. Based on extensive simulation, we are able to draw conclusions as summarized in this chapter. Also, suggestions for further research are discussed.

5.1 Summary of Contributions

1. A novel approach is proposed for finding sub-optimal image segmentations achieving bit-rate minimization applied to efficient coding of digital video sequences. In this work, a computationally efficient codec algorithm has been designed that demonstrates the bit-stream performance of a region-oriented approach. The quadtree approach has allowed a global minimum to be achieved with respect to image segmentation while minimizing the number of cost function evaluations required. This results in an improvement over existing video coding

standards by avoiding overly simplistic region segmentation, while maintaining practicality.

2. Several possible extensions of first generation video coding standards are proposed such as variable region segmentation, higher order motion models, and bit-rate minimization. This leads to improved video quality at low bit rates. It was shown that at least a 2-3 dB improvement in PSNR is possible compared to the H.263 video coding standard under TMN5 [48]. A video codec structure is presented and justified by theoretical analysis. A cost function formulation is shown along with a minimization procedure to achieve bit-rate optimization.

3. A software based video codec was designed to evaluate the performance of the new codec and serve as a basis for future experimentation. The performance of the codec is evaluated against modern videoconferencing standards, namely H.263. Tests were performed on three separate video sequences to demonstrate the performance under varying conditions. The performance was verified at 15, 10 and 7.5 kb/s which correspond to bit-rates which are in the range required for videoconferencing using standard telephone lines. The nature of distortion introduced with each algorithm was evaluated. An approach to dealing with the distortion related to Quadtree-MDL was presented and verified, where a median filter resulted in a 0.27 dB improvement in PSNR with even greater improvement based on subjective observation.

4. The software implementation used had a computation time of approximately 20 seconds per frame on an Apple Powerbook G4 notebook computer. Due to the

separability of the MDL cost function, the region-oriented video coding approach taken in this thesis is shown to be highly parallelizable, potentially leading to fast hardware implementation, such as the approach taken in [51].

5.2 Suggestions for Further Research

Based on the results presented in this thesis, we recommend the following areas for further research:

1. In this thesis, an independent software codec was developed without regard to bit-stream compatibility with existing video coding standards. Further work would be required in analyzing the bitstream syntax of standards such as H.263 compared to that of Quadtree-MDL to determine if some level of compatibility can be achieved with the fewest possible changes to existing standards.

2. A more direct comparison of particular features of the Quadtree-MDL algorithm with H.263 is needed in order to achieve fair comparisons of individual features that would lead to a series of practical optimizations. Furthermore, a current implementation of the latest test model (TMN11) [49] should be used to evaluate the performance of each algorithm.

3. An analysis should be performed on the computational aspects of the Quadtree-MDL algorithm with a view towards accelerating the software computation time. The motion estimation algorithm currently uses a direct search method, which is effective but too slow for real-time implementation. A more efficient method

needs to be found to minimize the mean-absolute-error metric for multiple motion model orders.

4. Different tree structures require experimentation to implement smaller block sizes to allow for finer region segmentation. Deviation from the strict quadtree structure may be explored in order to more closely match the region segmentation with actual objects in the scene without a large in increase in the overhead requirement.

5. Motion vector initialization can be improved through temporal linking and deterministic relaxation.

6. Perceptually weighted quantization represents on interesting area of research that can be used to improve the performance of the Quadtree-MDL algorithm. Noise shaping can be used which fits the reconstruction noise specturn to the frequency characteristics of the human visual system such that larger quantization errors are permitted without visible distortion [12].

7. The quadtree structure may lend itself to unequal error protection schemes that could be used to modify the existing work to allow for wireless channel transmission. Further investigation is required to evaluate this potential.

8. Hardware implementation methods should be explored, such as those presented in [51] which outlines the design of processing elements that can be implemented on an application specific integrated circuit for real-time implementation for applications such as videophones and wireless terminals.

9. Extensions of the MDL principle should be examined within the context of video coding. One approach is to consider how the MDL principle might apply to the case when the requirement for lossless coding is relaxed (lossy compression). Recent work in [21][22] has considered the connection between the MDL principle and rate-distortion theory, and applications to VQ- based image coding. Further research is required to apply the lossy-MDL principal to moving video sequences.

Bibliography

[1] N. Ahmed, T. Natarajan, and K.R. Rao, "Discrete cosine transform" *IEEE Trans. On Computers*, C-23:90-3, 1974.

[2] K. Aizawa and T.S. Huang, "Model-based image coding: Advanced video coding techniques for very low bit-rate applications" *Proceedings of the IEEE*, Vol. 83, pp. 259-271, Feb 1995.

[3] M. Bell and M.C. Pike, "Remark on Algorithm 178" Communications of the ACM, Vol. 9, pp. 685-686, Sept. 1966

[4] ITU-R Recommendation BT.601-4: "Encoding parameters of digital; television for studios".

[5] CCITT SG XV WP/1/Q4 Specialist Group on Coding for Visual Telephony, "Improvement of Reference Model 5 by a noise reduction filter" Document 376, 1988.

[6] P. Cicconi and H. Nicolas, "Efficient Region-Based Motion Estimation and Symmetry Oriented Segmentation for Image Sequence Coding" *IEEE Trans. On Circuits and Systems for Video Technology*, Vol. 4, No. 3, June 1994.

[7] N. Diehl, "Object-oriented motion estimation and segmentation in image sequences" *Signal Processing: Image Communications 3*, pp 23-56, 1991.

[8] F. Dufaux, “Multigrid Block Matching Motion Estimation for Generic Video Coding” PhD Dissertation, Swiss Federal Institute of Technology, Lausanne, Switzerland, 1994

[9] F. Dufaux, and F. Moscheni, “Motion Estimation Techniques for Digital TV: A Review and a New Contribution” *Proceedings of the IEEE*, Vol. 83, No. 6, pp. 858-875, June 1995.

[10] T. Ebrahimi, “New technique for motion field segmentation and coding for very low bitrate video coding applications”, *1994 International Conference on Image Processing*, 1994.

[11] M. Gilge, “A high quality videophone coder using hierarchical motion estimation and structure coding of the prediction error” *SPIE Proc. Visual Communications and Image Processing ’88*, Vol. 1001, pp. 864-874, Nov. 1988.

[12] B. Girod, H. Almer, L. Bengtsson, B. Christensson, and P. Weiss, "A Subjective Evaluation of Noise Shaping Quantization for Adaptive Intra-/Interframe DPCM Coding of Color Television Signals" *IEEE Transactions on Communications,* vol. 36, no. 3, pp. 332-346, March 1988.

[13] ITU-T Recommendation H.261: “Video codec for audiovisual services at p x 64 kbit/s” Geneva, 1990, revised at Helsinki, March 1993.

[14] ITU-T Recommendation H.263: “Video coding for low bitrate communication” March 1996.

[15] R. Hooke, T.A. Jeeves, “Direct Search Solution of Numerical and Statistical Problems” *Journal of the ACM*, Vol. 8, No. 2, pp. 212-229, April 1961.

[16] M. Hötter, “Object-oriented analysis-synthesis coding based on moving two-dimensional objects” *Signal Processing: Image Communications 2*, pp 409-428, 1990.

[17] T.S. Huang, ed., “Two-Dimensional Digital Signal Processing II, Transforms and Median Filters” *Topics in Applies Physics*, Vol. 43, Spinger-Verlag, Berlin, 1980.

[18] M. G. Johnson, “Nonlinear Optimization using the algorithm of Hooke and Jeeves”, personal e-mail correspondence, February 1996

[19] A. F. Kaupe, “Algorithm 178: Direct Search” Communications of the ACM, Vol 6, p. 313, June 1963

[20] R. Koenen, “Overview of the MPEG-4 standard” International Standards Organization, Stockholm meeting, ISO/IEC JTC1/SC29/WG11 N1730, July 1997.

[21] I. Kontoyiannis, "Model Selection via Rate-Distortion Theory" *2000 Conference on Information Sciences and Systems,* Princeton University, March 2000.

[22] I. Kontoyiannis, J. Zhang, M. Harrison, and A. Dembo, "MDL ideas in lossy data compression" *MSRI Workshop on Information Theory,* Berkeley, CA, February 2001.

[23] M. Kopp, and W. Purgathofer, “Efficient 3x3 median filter computations” Institute of Computer Graphics and Algorithms, Vienna University of Technology, Technical Report TR-186-2-94-18, Dec. 1994.

[24] M. Kunt, "Second-generation image coding techniques" *Proc. IEEE,* vol. 73, pp. 549-574, April 1985.

[25] Y.G. Leclerc, “Constructing simple stable description for image partitioning” *International Journal of Computer Vision*, Vol. 3, pp. 73-102, 1989.

[26] J. Lee, "Optimal quadtree for varible block size motion estimation" *1995 International Conference on Image Processing,* 1995.

[27] C.W. Lin, Y.J. Chang, E. Fei, and Y.C. Chen, "Efficient Video Coding with R-D Constrained Quadtree Segmentation" *Picture Coding Symposium 1999,* March 1999.

[28] H. Musmann, M. Hötter, and J. Ostermann, “Object-oriented analysis-synthesis coding of moving images” *Signal Processing*, Vol. 1, pp. 117-138, Oct. 1989.

[29] A.N. Netravali, and J.D. Robbins, “Motion-compensated television coding: Part I” *Bell Systems Technical Journals*, Vol 58, No. 3, pp. 631-670, March 1979.

[30] A.N. Netravali, B.G. Haskell, *Digital Pictures*, Plenum Press, 1988, 1st ed.

[31] A.N. Netravali, and B.G. Haskell, *Digital Pictures*, Plenum Press, 1995, 2nd ed.

[32] H. Nicolas, S. Pateux, and D. Le Guen, “Minimum description length criterion and segmentation map coding for region-based video compression” *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, No. 2, pp. 184-198, Feb. 2001.

[33] M.T. Orchard, and G.J. Sullivan, “Overlapped block motion compensation - an estimation-theoretic approach” *IEEE Transactions on Image Processing*, Vol. 3, No. 5, pp. 693-699, Sept 1994.

[34] R. Plompen, *Motion video coding for visual telephony*, PTT Research Neher Laboratories, Leidschendam, the Netherlands, 1989.

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

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

[37] J. Rissanen, "Modeling by shortest data description" *Automatica*, Vol. 14, pp. 465-471, 1978.

[38] J. Rissanen, "Universal coding, information, prediction, and estimation" *IEEE Transactions on Information Theory*, Vol. IT-30, No. 4, 1984.

[39] J. Rissanen, "Minimum-description-length Principle," *Encyclopedia of Statistical Sciences*, Vol. 5, pp. 523-527, 1987.

[40] Description of Reference Model 8 (RM8), CCITT Study Group XV, Specialist Group on Coding for Visual Telephony. Doc. No. 525, June 1989.

[41] P. Salembier, L. Torres, F. Meyer, and C. Gu., "Region-based video coding using mathematical morphology" *Proceedings of IEEE*, Vol. 83, No. 6, pp. 843-857, June 1995.

[42] G.M. Schuster and A.K. Katsaggelos, "A very low bit-rate video codec with optimal trade-off among DVF, DFD, and segmentation" *European Signal Processing Conference, 1996.*

[43] R.L. Stevenson, "Reduction of coding artifacts in transform image coding", *IEEE ISCASSP Proceedings*, vol. V, pp. 401-404, 1993.

[44] P. Strobach, “Tree-structured scene adaptive coder” *IEEE Trans. Communications*, Vol. COM-38, No. 4, pp. 477-486, April 1990.

[45] K.W. Stuhlmüller, A. Salai, and B. Girod, “Rate constrained contour-representation for region-based motion compensation” *SPIE Symposium on Visual Communications and Image Processing*, March 1995.

[46] G.J. Sullivan and R.L. Baker, "Efficient quadtree coding of images and video" *1991 International Conference on Acoustics, Speech, and Signal Processing,* 1991.

[47] T. Cover, and J. Thomas, *Elements of Information Theory*, A Wiley-Interscience publication, John Wiley & Sons, 1991

[48] TMN H.263 Codec, Digital Video Coding at Telenor Research, Norway

[49] ITU-T SG15, H.263+ Ad Hoc Group, “Video test model, TMN11,” LBC-96-141, October 1999

[50] F.K. Tomlin, and L.B. Smith, “Remark on Algorithm 178” Communications of the ACM, Vol 12, no. 11, pp. 637-638, November 1969

[51] A. Tyagi and M.A. Bayoumi, “Image segmentation on a 2-D array by a directed split and merge procedure” *IEEE Trans. On Signal Processing*, Vol. 40, No. 11, Nov. 1992.

[52] K. Waters, “A muscle model for animating three-dimensional facial expression” *Computer Graphics*, Vol. 21, pp. 17-24, July 1987.

[53] S. Winkler, "Visual Fidelity and Perceived Quality: Towards Comprehensive Metrics" *SPIE Human Vision and Electronic Imaging Conference*, vol. 4299, pp. 114-125, San Jose, California, January, 2001

[54] I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic Coding for Data Compression" *Communications of the ACM*, Vol. 30, pp. 520-540, June 1987.

[55] R. Wood, "On Optimum Quantization" *IEEE Transactions on Information Theory*, Vol. IT-15, pp. 248-252, March 1969.

[56] Y. Yang, and S. Hemami, "Generalized Rate-Distortion Optimization for Motion Compensated Video Coders" *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No. 6, pp. 942-955, Sept. 2000.

[57] H. Zheng and S.D. Blostein, "Motion-Based Object Segmentation and Estimation Using the MDL Principle" *IEEE Transactions on Image Processing*, Vol. 4, pp. 1223-1235, Sept. 1995.

Vita

Paul C. Wareham

EDUCATION

M.Sc. (2002), Electrical and Computer Engineering, Queen's University

B.Eng. (1993), Electrical Engineering, Lakehead University

Dip. Tech. (1991), Electronics Engineering Technology, University College of Cape Breton

EXPERIENCE

Electronics Engineering Technology Instructor (1997-2002), University College of Cape Breton

Research Assistant (1995-1997), Electrical & Computer Engineering, Queen's University

Teaching Assistant (1996), Electrical & Computer Engineering, Queen's University

ASIC Design Engineer (1994), ATI Technologies Inc.

PUBLICATIONS

Paul C. Wareham and Steven D. Blostein, "Region-Oriented Video Coding Using the MDL Principle and Quad-Tree Optimization", *1996 International Conference on Image Processing*, September 1996.