Development and Analysis of Block-Based Motion Estimation Techniques for Efficient Video Compression

by

Jatinder S. Chana
BEng., University of Victoria, 1992

A Thesis Submitted in Partial Fulfillment of the
Requirement for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

We accept this thesis as conforming to the required standard

[redacted]
Dr. P. Agathoklis (Department of Electrical and Computer Engineering)

[redacted]
Dr. F. El-Guibaly (Department of Electrical and Computer Engineering)

[redacted]
Dr. P. Fisher (Department of Health Information Science)

[redacted]
Dr. G. Brauer (Department of Health Information Science)

Supervisor : Dr. P. Agathoklis

Abstract

Block-based Motion Estimation and Compensation (ME/MC) is a popular inter-frame coding technique and is used in video compression systems, e.g. the MPEG video coding standard, to exploit similarities between adjacent frames in typical video sequences. It provides additional compression and also improves fidelity of decoded video in low bit-rate applications. In a typical application, the frame being coded is partitioned into fixed sizes blocks and only motion information of these blocks, with respect to a reference frame, is coded. Search range is a parameter that directly controls the accuracy of estimated motion information and the computational effort required for motion estimation. Exhaustive ME using larger search range results in greater accuracy at the expense of greater computation. Although exhaustive ME is very accurate, it is also very demanding in terms of computation.

Efficient algorithms, collectively referred to as Block Matching Algorithms (BMAs), have been proposed as alternatives to exhaustive ME to provide computationally efficient ME. BMAs use efficient search algorithms to obtain motion information and require substantially less computation. A comprehensive evaluation of a select number of such BMAs is present in this thesis. The BMAs selected for evaluation are representative of the general approaches used in designing such algorithms and are evaluated based on their estimation accuracy and computational efficiency. The evaluation is performed using two real video sequences that contain motion present in typical video sequences. The evaluation results indicate that BMAs are effective only for ME involving small motion displacements.

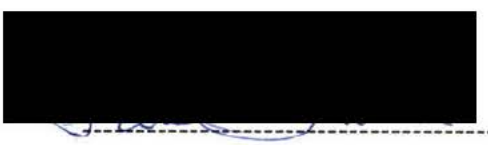
A predictive motion search method, the Adaptive Search Window Method (ASWM), is proposed to overcome this limitation. This method dynamically controls the location and search range for ME for individual blocks based on motion information of surrounding blocks. If correlation is detected in surrounding motion information, a prediction is made for the corresponding block and ME is performed around the predicted block using a reduced search range. The reduction in the search range results in accurate ME using BMAs or, alternatively, in less computation for exhaustive ME. The applica-

tion of BMAs incorporating ASWM for ME is evaluated and a significant improvement in estimation accuracy, compared to application of BMAs without ASWM, is observed for ME involving larger motion displacements. Two BMAs incorporating ASWM are also implemented in a software video coder and compression gains over respective BMAs without ASWM are demonstrated.

Examiners:

Dr. P Agathoklis (Supervisor)
Department of Electrical and Computer Engineering

Dr. Fayez El-Guibaly (Departmental Member)
Department of Electrical and Computer Engineering



Dr. Paul Fisher (External Member)
Department of Health Information Science

Dr. Gerhard Brauer (External Examiner)
Department of Health Information Science

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ASWM	Adaptive Search Window Method
BMA	Block Matching Algorithm
CCITT	International Committee on Telegraph and Telephones
CCIR	International Consultative Committee on Broadcasting
CIF	Common Intermediate Format
CSA	Cross Search Algorithm
DCT	Discrete Cosine Transform
DSP	Digital Signal Processor
GOP	Group of Pictures
HDTV	High Definition TeleVision
HVS	Human Visual System
ISO	International Standards Organization
JPEG	Joint Experts Picture Group
MAD	Mean Absolute Difference
ME	Motion Estimation
ME/MC	Motion Estimation and Motion Compensation
MPEG	Motion Pictures Expert Group
MSE	Mean Square Error
MTSS	Modified Three Step Search
OSS	Orthogonal Step Search
OTS	One Time Search
QCIF	Quarter-CIF
TDL	2-D Logarithmic Search
TSS	Three Step Search
VLC	Variable Length Coding
VLSI	Very Large Scale Integration

ACKNOWLEDGEMENTS

I would first like to acknowledge the financial support of both the University of Victoria and my supervisor, Dr. P. Agathoklis without which I would not have been able to undertake this work. I would like to thank my supervisory committee: Dr. P. Agathoklis, Dr. F. El-Guibaly and Dr. P. Fisher, for the guidance and consultation that they provided through out my research. I would also like to thank the computer and technical staff for the support that they extended towards my work. I would like to thank Al Keddy for making available the DEDIP image processing library which proved invaluable during my research. Finally, I would like to extend my heart felt gratitude towards my family who have always supported and encouraged me towards my goals, both academic and other.

Dedicated to my family

Chapter 1

Digital Video Background

1.1 Introduction

The old adage -"a picture is worth more than a thousand words"- inarguably embodies the overwhelming preference, by humans, for visual information. Visual information forms a vital part of human cognition and the use of images greatly enhances the process of communicating ideas. As a result, visual communication systems such as television/video/computers are increasingly being used for a wide variety of purposes. This increasing use of visual information in a growing spectrum of applications keeps perpetuating the need to build more efficient and economical visual communication systems. A fundamental requirement for developing such systems is the need for efficient image storage and transmission techniques.

Interest in efficient image transmission and storage techniques has steadily grown since the first successful laboratory demonstrations for television broadcast in the early 1920s. However, it wasn't until the last three decades that significant improvements were achieved. These achievements are attributed to two major factors: the discovery and refinement of new image coding methods, and the advent of relatively inexpensive, compact, high-speed, solid-state signal processing circuitry [37]. The related growth in capacity of digital storage mediums and digital transmission links has also spawned an interest in high quality digital image storage and transmission techniques.

Digital data representations provide a high degree of reconstruction and integration. They are resilient to distortion induced by noise present in practical transmission channels and, therefore, particularly suited for long distance transmission. Even in the presence of moderately high noise, digital data can be reconstructed exactly repeatedly over short distances and remain immune to dispersive channels. Also, an important long

term advantage of using digital representations is the fact that all signals, i.e. voice, television, fascismile and computer data become a stream of similar looking pulses and, as a consequence, not only relieve the need to engineer separate transmission channels for respective signals [38], but also provide a high degree of integration between these signals.

Transmission and storage of images in their digital form, however, is extremely demanding on digital mediums because of the immense amount of data bits required to represent these images. This is particularly true for digital video which requires an incredibly large number of bits for representation and, in many cases, this requirement surpasses the capacity of digital mediums. Efficient image coding[1] techniques are therefore essential to overcome limitations imposed by such mediums and also vital for designing economical digital visual communication systems. Implementing real-time digital video systems presents an extraordinary challenge to designers since, in addition to achieving high processing speeds associated with video systems, extremely efficient coding methods are required to provide a high level of compression without degrading decoded video fidelity.

This chapter introduces relevant concepts and terminology pertaining to digital images and provides a general background on digital video compression. Since digital video compression schemes generally extend upon still image compression schemes, a background on still image compression techniques and related performance measures are presented first. Finally, the motivation for the work undertaken, and an outline of the thesis is presented.

1.2 Digital Image Representation

Digital images are images that have been discretized both in spatial co-ordinates and in brightness [36], and are acquired from analog scenes by sampling in space and quantizing in brightness levels. The sampling step size is generally chosen to be small

1. The terms *coding* or *encoding* are also commonly used in electronic communication texts and literature to imply error-control coding which increases the number of bits to represent data. In this thesis presentation, however, the terms *coding*, *encoding* and *compression* are used interchangeably to imply the act of reducing the number of bits to represent associated data.

enough to rely on the integration abilities of the human eye while brightness levels are restricted to discrete values that can be recorded in digital form. In its canonical form, a digital image can be viewed as a matrix whose row and column indices determine a sampled point, or picture element, whose value in turn indicates its respective brightness level. Picture elements are also referred to in abbreviated terms as *pixels* or *pels*. An 8-bit 512x512 image, for instance, can be viewed as being composed of 512 rows and 512 columns of 8-bit values representing 256 unique brightness levels.

To represent color in images, a number of different color models have been proposed. Three hardware-oriented color models are RGB, used with color CRT monitors, YIQ, the broadcast TV color system, and CMY (cyan, magenta, yellow) for some color printing devices [39]. Each pixel of a color image displayed on CRT terminals is composed a red component, R(i,j), a green component, G(i,j), and a blue component, B(i,j). Similarly, color images in the YIQ color space are composed of a luminance component (Y) and two chrominance components, I and Q. The YIQ color space is used for broadcast television to provide compatibility between color and black/white television receivers. On black/white television sets, only the luminance Y component is displayed, while on color television sets, the additional IQ components are decoded to provide color information. The forward and inverse transformations from RGB space to the corresponding YIQ space are represented by the matrix operations of Eq.(1.1) and Eq.(1.2) respectively [35].

$$\begin{bmatrix} Y(i,j) \\ I(i,j) \\ Q(i,j) \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R(i,j) \\ G(i,j) \\ B(i,j) \end{bmatrix} \tag{1.1}$$

$$\begin{bmatrix} R(i,j) \\ G(i,j) \\ B(i,j) \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y(i,j) \\ I(i,j) \\ Q(i,j) \end{bmatrix} \tag{1.2}$$

A slight variation of the YIQ color model is the YCbCr color model which is used for image compression and will be discussed later.

Digital color images using any of these models can be represented with three matrices, each recording the level of individual spectral components. Digital video in its canonical representation is composed of a sequence of digital images each representing

individual frames of the video.

The immense number of bits associated with such canonical representations of digital images and video are very demanding on both transmission and storage resources. Consider, for example, the digital representation of a single 8-bit monochrome image of resolution 512x512 pixels. Such an image requires about $2x10^6$ bits of information in its canonical form. A color image of the same resolution requires three times this amount of bits. The situation is further compounded with digital video where a 30 minute color video sequence, displaying 30 such images every second, requires an astronomical amount of about $3x10^{11}$ bits for canonical representation. Thus, for both economical reasons and to overcome restrains of digital media, more efficient representations of digital images and digital video are desired. This need has spawned a number of sophisticated compression schemes that attempt to minimize the amount of bits required to represent digital images and video.

1.3 Image Compression Background

The foundations of signal compression date back to work done by Shannon [1], in the area of information theory, where he proposed a mathematical means of measuring information content or *entropy* of a data source. This information measure, acquired by appropriately modeling the data source, determined a minimum bound for compressing data sequences generated by the source without information loss, and provides an insight into the amount of statistical redundancy present in the source. The development of Rate Distortion Theory [34] extended Shannon's work by investigating information content of data sources subject to distortions as measured by specific distortion measures. Classical source coding techniques, developed around these information theory concepts, achieve compression by removing statistical redundancy present in data sources. Two such popular methods, huffman coding [2] and arithmetic coding [3], result in compression that comes arbitrarily close to Shannon's minimum bound for long data sequences. However, a fundamental disadvantage of classical source coding techniques is the need for suitable source models. In the case of digital audio and video, such models are difficult to obtain since the corresponding signals are generally non-gaussian, non-stationary and have an intractable power spectrum [10]. Nevertheless, Shannon's work and Rate Distortion The-

ory has provided a valuable basis for compression techniques applicable to speech, digital audio and digital video which, in essence, attempt to remove any form of redundancy present in corresponding data.

Natural image scenes inherently possess significantly high levels of *Spatial*, *Spectral* and *Temporal* redundancies. *Spatial Redundancy* is present in the form of high correlation between neighboring pixels, *Spectral Redundancy* exists as high correlation between spectral components, e.g. RGB components in color images, while considerable *Temporal Redundancy* is present in the form of high inter-frame correlation between neighboring frames in a video sequence. Image coding techniques achieve compression by removing these redundancies.

The first generation of image coding techniques relied heavily on information theory concepts and produced compression ratios at most of 10:1. These techniques have been classified into three categories: Spatial, Transform, and Hybrid methods [4]. Spatial image coding methods directly code spatial information and include techniques such as Pulse Code Modulation (PCM), Bit-plane coding and predictive methods such as Differential-PCM (DPCM) and Delta Modulation. Transform Image coding techniques, on the other hand, transform spatial information into a less correlated set of transform coefficients using suitable transforms and subsequently code the coefficients. Transform coded images are decoded using corresponding inverse-transforms. Hybrid coding incorporates both spatial coding (usually predictive techniques) and transform coding and thereby combine the advantages of these two approaches.

Second generation image coding methods exploit limitations of the Human Visual System (HVS) to remove 'visually redundant' information and consequently achieve significantly higher compression. Since for the most part, the end viewer in many visual communication systems is human, the loss of this information has no perceptual impact on the decoded image quality. Preliminary models of the HVS have been proposed and applied in recent coding schemes but research still continues in determining more accurate models of the HVS for designing optimal perceptual coders. A common component of second generation image coding schemes is the use of perceptually weighted quantization which exploits the fact that the HVS is less sensitive to loss of high spatial frequency components. These components are coarsely quantized for additional compression. More sophisticated techniques that decompose images into a number of spatial frequency bands to provide greater control for coding have also been proposed.

These techniques, which include Laplacian Pyramid coding [8], Sub-band coding [6]-[7] and Wavelet Coding [9], are constantly being refined for economic feasibility.

Lossy image compression techniques, implying loss of information, based on HVS responses provide an attractive means for building economical and efficient visual communication systems. In fact, the proposed ISO[1] standards for still image compression, JPEG [13], and digital video compression, MPEG [14], utilize lossy image compression techniques to achieve both high compression and high image fidelity. These standards also provide designers with the option of compromising between image fidelity and amount of compression. However, it should be pointed out that a small class of applications particularly in the field of medical imaging still require, for both diagnostic and legal reasons, absolutely lossless compression schemes[2].

1.4 Image Compression Performance

In general, the dimensions of performance of signal compression schemes, as presented in [10], include signal quality, bit-rate, coding complexity and coding delay. The large range of applications that utilize compression generally place different emphasis on these performance measures. In the case of image compression schemes, these performance measures refer to:

Image Quality - This is a subjective measure. Generally, the decoded image fidelity is viewed from a certain distance, under certain lighting conditions, and ranked on a scale of 1 to 10 or from EXCELLENT to UNACCEPTABLE.

Transmission/Bit Rate - This term is analogous to the term *bandwidth* as used in analog transmission links. The maximum transmission rate/throughput of digital links/ storage media, measured in bits per second (bps), places a fundamental limit on how quickly digital data can be correctly transmitted/retrieved. Another term of interest for still image compression is bits per pixels (bpp). This measure corresponds directly to the average number of bits used for each pixel in the coded image, thus providing an esti-

1. International Organization for Standardization.

2. Dr. P.D. Fisher, Department of Health Information Science, University of Victoria; 1994; personal communication.

mate of compression achieved.

Coding Complexity - This corresponds to the computational effort used by the coder/decoder to process corresponding data and is generally measured by arithmetic and memory requirements. Coding schemes are usually classified based on this measure as being either *symmetric* or *asymmetric*. Symmetric coding schemes incorporate equal complexity in both the coder and decoder and are suited for systems that incorporate equal number for coders and decoders. Asymmetric schemes, on the other hand, incorporate more complexity in either the coder (or decoder) and are suited for systems that utilize more decoders than coders (or vice versa) such as video broadcast systems or image databases.

Communication Delay - Communication delay or coding delay is closely linked with coding complexity and measures corresponding processing delays. This measure is important for real-time two-way communication schemes such as video conferencing or videotelephony systems. It is of particular importance in transport of video over networks [11].

1.5 Digital Video Compression Background

Digital video coding methods can be classified as either Intra-Frame or Inter-Frame coding methods. Intra-frame compression techniques do not attempt to remove temporal redundancy, and code each frame in the sequence as an individual image. Although, intra-frame techniques do not achieve high compression, the independence of neighboring frames in the coded bit stream makes random accessibility and editing of individual frames much easier. Inter-frame compression techniques, on the other hand, remove temporal redundancy and consequently achieve significantly higher compression. Inter frame video coding techniques generally extends upon still image compression in that, still image compression techniques are still used to remove both spatial and spectral redundancy, while sophisticated inter-frame coding techniques are used to remove temporal redundancy. Inter-frame coding techniques are essential for applications that demand both high compression and high video fidelity.

Three general approaches that have been applied for interframe coding include 3-D transform coding, Hybrid coding and Frame replenishment [37]. 3-D transform coding

extends upon 2-D transform coding used for still image compression and computes an addition 1-D transform in the temporal direction. Provided there is high temporal correlation in the data, 3-D transform coding produces high compression but at the cost of being computational more intensive and requiring storage of a number of frames, equal to the dimension of the temporal transform, both at the coder and decoder. This storage of extra frames is generally wasteful in practical systems. Hybrid transform/predictive coders incorporate, in addition to 2-D transform for spatial information, a predictive scheme for temporal data. Frame replenishment techniques, on the other hand, attempt to reduce temporal redundancy by coding and transmitting, with varying resolution, only changing parts between consecutive frames [37]. The parts of the frame that do not change are reconstructed from the previous frame. This approach is well suited for video telephony which involves stationary cameras and typically small areas of motion in front of a larger still background. However, for television broadcast programs, which generally contain substantial amount of motion, such a simple approach is not adequate.

When scenes contain objects moving more or less in translation, more redundancy exists than is taken advantage of by frame replenishment [37]. Sophisticated techniques, collectively referred to as Motion Estimation and Motion Compensation (ME/MC) techniques, that estimate and compensate for motion of objects within image sequences have been investigated and applied to remove this motion redundancy for additional compression. Two general approaches that have been applied for ME/MC include pel-recursive ME/MC and block based ME/MC [28]. Pel-recursive ME/MC operates with individual pels while block based ME/MC operates with image blocks. Although pel-recursive ME/MC uses a more realistic model, it is computationally intensive and works well only with sequences containing small amount of motion. Block based ME/MC, on the other hand, although based on a very simple motion model, has gained greater application as a result of its computational and implementation simplicity.

It has been demonstrated that ME/MC not only provides additional compression but also improves decoded video fidelity and is applied extensively in video coding techniques designed for low-bit rate applications [15]. Also, economical real-time digital video systems demand compression ratios that are achievable only with inter frame coding techniques, and ME/MC is becoming an increasingly important part of inter-frame coding methods A comprehensive evaluation of Block-Based ME/MC, which forms an essential part of emerging video coding standards, is the focus of the work under taken for this thesis.

1.6 Outline of Thesis

The work undertaken in this thesis is directed at Block Based Motion Estimation and Compensation. Motion Estimation and Compensation is an important inter-frame coding technique that provides addition video compression without degrading decoded image quality. It forms an integral part in many proposed video coding algorithms [16]-[19] and the use of Block-based ME/MC is predominant in these schemes. Accurate ME/MC, however, is computationally expensive. To alleviate much of the computational load for Motion Estimation, a number of efficient Block Matching Algorithms (BMAs) have been proposed [23]-[29]. These algorithms substantially reduce the computational complexity of block-based ME/MC. A comprehensive evaluation of BMAs for use in emerging video coding standards is essential as such an evaluation will prove invaluable to designers who wish to implement emerging video coding standards on general purpose computers or DSP chips which provide limited computational performance. Moreover, such an evaluation will also provide a stepping stone for developing more efficient BMAs

This thesis addresses this issue and undertakes a comprehensive evaluation of a number of existing Block Matching Algorithms. In addition, a predictive ME technique is proposed that, when incorporated for ME, substantially improves the performance of BMAs.

The thesis presentation is divided into 5 chapters:

In Chapter 2, an overview of the MPEG-1 video coding standard, an ISO video coding standard for multimedia applications at 1.15Mb/s, is presented. The two essential components of the algorithm, Block DCT image coding and Block Based Motion Estimation and Compensation, are introduced. The application of Block Based ME/MC in the MPEG algorithm to reduce temporal redundancy is discussed in detail and the need for efficient ME/MC reemphasized.

In Chapter 3, a select number of representative Block Matching Algorithms (BMAs) for efficient ME are evaluated. The algorithms are introduced and the evaluation criteria used to measure their performance are discussed. A series of simulation tests are performed using two video sequences as benchmarks. These two sequences are chosen because they represent cases of motion that appears in many video sequences. The results of these tests are presented and discussed.

In Chapter 4, a robust predictive ME technique, the Adaptive Search Window

Method (ASWM), is proposed. This technique can be incorporated with BMAs to substantially improves their accuracy. Simulation tests presented in Chapter 3 are repeated for BMAs incorporating the ASWM and corresponding results presented. In addition, two robust BMAs incorporating the ASWM are implemented in a software MPEG coder and corresponding compression gains over respective BMAs not incorporating ASWM reported.

Finally, Chapter 5 concludes the thesis by summarizing important results of the work undertaken.

Chapter 2

MPEG: An Emerging Video Coding Standard

2.1 Introduction

A concerted effort is being made to standardize video compression techniques so as to contain the proliferation of incompatible coders and decoders. Also, standardization of video coding to allow inter-operability between coders and decoders from different manufacturers will trigger large scale manufacturing of standard VLSI components and significantly reduce the cost of building these coders and decoders. Such a trend is clearly evident in the spectacular growth of the fax industry after standardization of the Group 3 fascismile compression algorithm by CCITT [14]. The International Standards Organization (ISO) has already produced the MPEG-I standard, named after the expert group that initiated the effort-Moving Pictures Expert Group, for coding video signals at 1.5 Mb/s and associated audio at 64, 128 and 192 kb/s [20]. MPEG activities are now being directed towards completing the MPEG II standard for coding of higher resolution video and associated audio. This second standard, MPEG II, has already been accepted by The HDTV Grand Alliance for broadcasting of HDTV in the U.S. [21].

Both of these video coding standards, MPEG I and MPEG II, rely on the Discrete Cosine Transform (DCT) to reduce spatial redundancy and block-based Motion Estimation and Motion Compensation (ME/MC) to reduce temporal redundancy. Transform properties of DCT have been favored extensively for compressing digitized continuous tone still images, and it (DCT) forms the basis for the ISO standard proposed by the Joint Pictures Expert Group (JPEG) for coding such images [13]. Block-based ME/MC is preferred to reduce temporal redundancy because of its implementation and computational

simplicity. Interframe coding removes temporal redundancy present in video sequences to allow for greater compression and block-based ME/MC forms an integral component of the interframe coding technique used in these standards.

This chapter provides an overview of the MPEG I video coding standard and introduces the two essential components of this coding standard: Block-based DCT image coding and Block-based ME/MC. The application of block-based ME/MC to exploit temporal redundancy is discussed in detail. Initially, however, digital video coding targets and design issues addressed by the MPEG committee in developing the MPEG-I algorithm are presented.

2.2 Digital Video Compression Targets

The astronomically high transmission rate associated with canonical digital video representation pose an extraordinary challenge for designing digital video systems. Even with the promise of larger bandwidths of optical links, there is still an essential need for efficient digital video compression techniques because of the continued and, in fact growing, use of band-limited media such as satellite links and bit-rate limited storage media such as CD-ROMs and solid-state memory [10]. This vital need for efficient and robust video coding techniques is underlined in the following sub-sections which compare transmission rates of some un-compressed standard digital video formats with limitations of existing communication links.

2.2.1 Video Rates

Table.(2.1) presents sampling rates of the luminance components of four recommended video transmission formats: Quarter-CIF, CIF[1], CCIR-601[2] and one of the proposed formats for HDTV[3]. The respective resolutions are presented in pels and number of

1. Common Intermediate Format: proposed for easier conversion with existing video formats
2. A format for digital coding of color television signal recommended by the International Consultative Committee for Broadcasting.
3. High Definition Television

frames/second.

Format	Resolution	Sampling Rates (MHz)
QCIF	180x144 @ 15hz	0.4
CIF	360x288 @ 30hz	3
CCIR 601	720x576 @ 30hz	12
HDTV	1280x720@ 60hz	60

Table 2.1. Bit rates for some visual communication formats [10]

For un-compressed video, the sampling rates correspond directly to transmission rates, where an N-bit pixel representation requires a transmission bit rate of (N x Sampling Rate) bps. Color versions of corresponding video formats require three times the sampling rates presented in Table (2.1).

2.2.2 Media Limitations

The essential need for video compression is underscored in Table.(2.2) which compares bit-rates of available digital medias with un-compressed transmission rates of applicable video formats.

Channel Type	Channel Bandwidth
Telephone	48-112kb/s
N-ISDN	64-382kb/s
CD-ROM Media	1.5 Mb/s
NTSC Channel (6Mhz)	20Mb/s

Digital Video Format	Un-Compressed transmission rate
QCIF (Monochrome)	~ 3 Mb/s
CIF (color)	~ 74 Mb/s
CCIR (color)	~ 298 Mb/s
HDTV (color)	~ 1.3 Gb/s

Table 2.2. Bit-rates of digital media and corresponding un-compressed digital video transmission rates

As evident from Table (2.2), extremely high compression is required to match transmission rates with limitations of corresponding digital media. Also, if digital transmission is to replace analog transmission, the minimum visual quality of decoded video should at least match or surpass the fidelity of the analog counterpart with similar bandwidth limitations. Thus, robust digital video coding schemes that achieve high compression without compromising decoded image fidelity are essential for digital video systems. One such

technique, MPEG-I, proposed by the MPEG committee as an ISO standard for coding video and associated audio at 1.5Mb/s is discussed here.

2.3 MPEG-I Video Coding Standard Targets

The MPEG-I video standard was developed in consideration of the large range of applications. High quality video compression is essential for applications ranging from Multimedia, Video databases, Movies-on-demand to video telephony, but these applications impose disparate requirements on the coding process. For instance, while symmetric coding is preferable for video telephony, asymmetric coding works better for video database systems where much of the coding complexity can be placed in the encoder. Also, the need for random access of frames, from the coded bitstream, in some applications limits inter frame dependency, thereby limiting the amount of compression possible. Coding/decoding delay requirements in other applications, such as video conferencing, place severe restrictions on coding/decoding complexity. These and other important considerations addressed in developing the MPEG-I coding algorithms, discussed in greater detail in [14], are listed here:

- Random Access of particular frames
- Fast Forward.Reverse Searches
- Reverse Playback
- Audio/Visual Synchronization
- Robustness to Errors
- Coding/Decoding Delay
- Edit-ability
- Format Flexibility
- Cost Tradeoff

In consideration of these restrictions, the MPEG group proposed the MPEG-I standard for coding of moving pictures and associated audio for digital storage media at up to about 1.5Mb/s [20]. This standard incorporates extensive flexibility, allowing applications to appropriately configure coding parameters and streamline the coding process to satisfy requirements.

2.4 Overview of MPEG-I Standard

The MPEG video compression algorithm processes input video in YCbCr representation where each frame is represented by one luminance component (Y) containing brightness information and two sub-sampled chrominance components (Cb, Cr) containing color information. This color space is similar to, but not the same as, the YIQ color space used for broadcast television. The YCbCr color plane exhibits low cross-correlation between its spectral components and possesses less spectral redundancy, and it is for this reason that YCbCr color space is preferred in image compression schemes. Also, in YCbCr color space, most of the image energy is concentrated in the luminance Y component. This representation complements the human visual system (HVS) which is more sensitive to variations in the luminance component than in the chrominance components. Thus, during compression, the perceived image quality can be maintained by coding the Y-component more accurately while compressing both chrominance components to a high degree. This property of the HVS is exploited by the MPEG standard. The MPEG syntax requires the input video signal to be digitized and represented in YCbCr color space with both chrominance signals, Cb and Cr, subsampled with respect to the luminance signal by a ratio of 2:1 in both vertical and horizontal directions [20].

MPEG operates on individual frames by partitioning them into fixed sized 8x8 pel blocks. These blocks are grouped into *Macroblocks*, which in turn are grouped into *Slices* as illustrated in Fig.(2.1). Each macroblock is composed of four luminance blocks and two sub-sampled chrominance blocks and forms the basic unit of the MPEG syntax. Each slice contains an integral number of macroblocks ordered in a raster scan manner. The minimum number of slices in a frame is one, the maximum number is equal to the number of macroblocks in the frame. Slices are coded independently from each other and limit propagation of decoding error during decoding. The macroblocks in the frame are processed in a left-right and top-bottom raster scan manner.

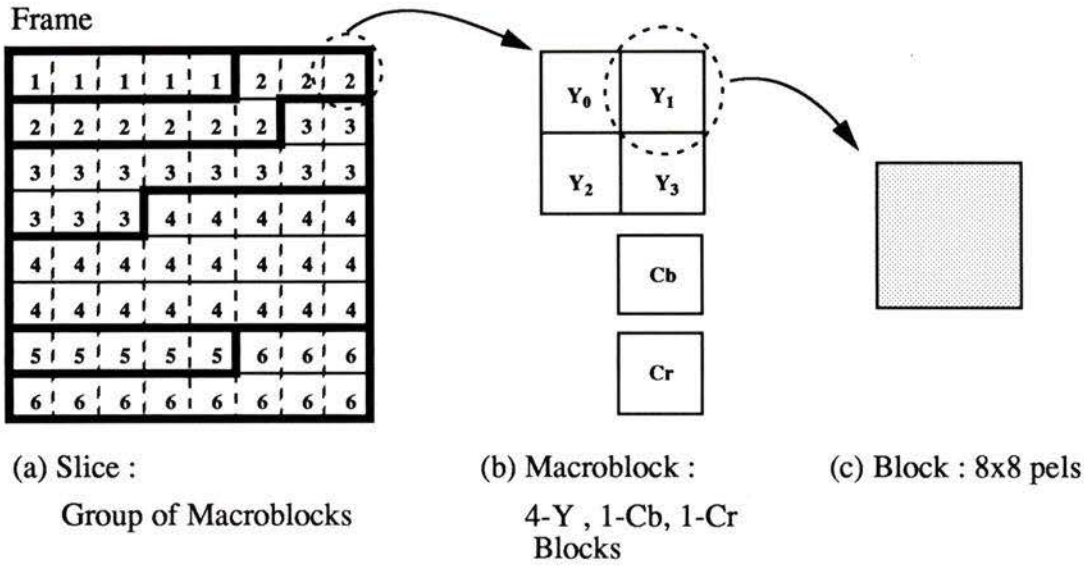


Figure 2.1 Frame representation in terms of Slices, Macroblocks and Blocks

Compression is achieved by first performing block-based ME/MC for the macroblocks to reduce temporal redundancy and then applying Block Discrete Cosine Transform (DCT) and entropy coding on the resulting data to reduce spatial redundancy as illustrated in Fig.(2.2). Motion estimation is performed only on the Y-component of the frame, assuming that motion in the Cb and Cr components is the same. The DCT coding steps used in MPEG are similar to those used in JPEG, the still image compression standard.

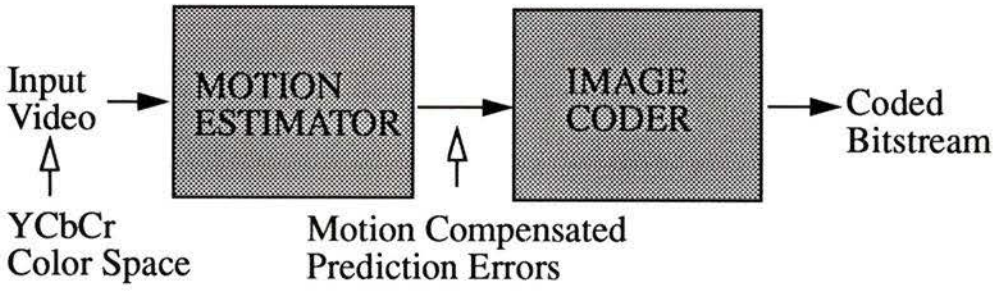


Figure 2.2 An oversimplified block diagram of the MPEG-I coder

The following sections present in detail the major components of the MPEG algorithm. The Block-based DCT image coder is presented first, followed by a detailed introduction to Block-based ME/MC and its use in the MPEG standard. In conclusion, more complete block diagrams of a MPEG coder and decoder are presented and the need for efficient motion estimation discussed.

2.5 Block-DCT based Image Coder

In general, an image coder compresses an image in three interrelated steps as illustrated by Fig.(2.3). The transformation step transforms, or maps, the input data into a more suitable domain for compression. The quantization step rounds the resulting information into a finite set of output levels. The last step, codeword assignment, assigns a codeword to each of these quantized levels. Input to the coder may be the image itself, individual spectral components of the image (e.g. RGB or YCbCr components), or frequency sub-bands of an image obtained via filter bank analysis. In the case of MPEG, motion-compensated prediction errors generated by ME/MC are input to the image coder. A brief description of each of these steps is included in the presentation of Block-based DCT coding as used in MPEG.

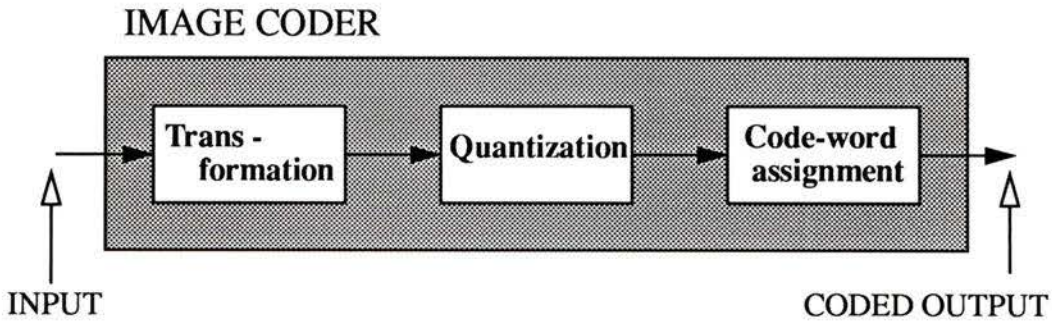


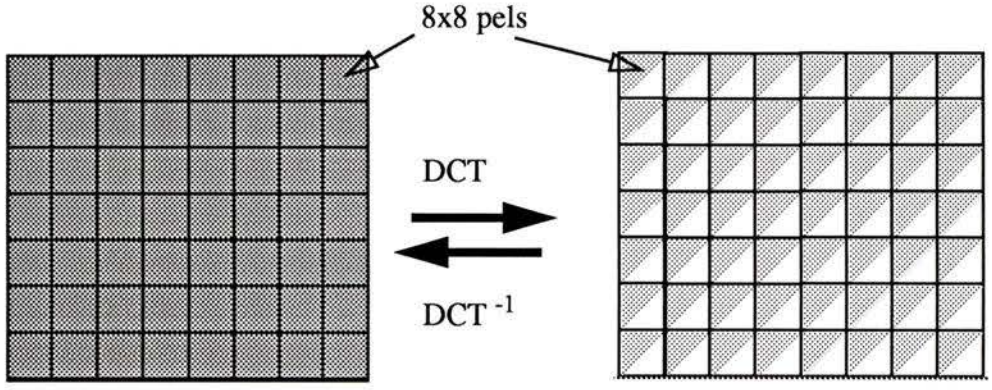
Figure 2.3 General structure of an image coder

2.5.1 Block DCT Transformation

DCT image coding is a member of the general class of transform coding techniques. Transform coding techniques removes redundancy in input data by transforming it into a less correlated set of transform coefficients that are then quantized and coded. The most

commonly used transformations in image coding applications are linear transformations implemented with fast algorithms for computational efficiency. The Karhunen-Loeve transform (KLT) is the best linear transform in the sense that it leads to un-correlated coefficients but has proved unsuited for practical applications because of its computational complexity [4]. Other fast transforms that have been investigated for image coding include Walsh-Haadmard Transform, Discrete Fourier Transform and Slant transform and are discussed in [40]. DCT has two significant advantages over these transforms. It needs fewer coefficients for a good approximation of a signal and efficient DCT implementations are available. These superior properties have made DCT the transform of choice for image compression. Its performance in de-correlating input data approaches that of the KLT and it has been selected as the basis of ISO standards for still Image compression (JPEG) and video compression (MPEG-I).

In the MPEG algorithm, the DCT is implemented to transform the input image block by block. The image is first partitioned into fixed size blocks of 8x8 pels, as illustrated in Fig(2.4), and each block is then transformed using the 2-D Forward DCT of Eq.(2.1). The corresponding Inverse DCT is presented in Eq.(2.2).



(a) Partitioned Image (b) Transformed Image

Figure 2.4 Block transformation of an image using 2-D DCT

$$\mathbf{F(u,v)} = \frac{1}{4}\mathbf{C(u)\,C(v)} \sum_{\mathbf{x}=0}^{7} \sum_{\mathbf{y}=0}^{7} \mathbf{f(x,y)} \cos\left(\frac{\pi(2\mathbf{x}+1)\mathbf{u}}{16}\right) \cos\left(\frac{\pi(2\mathbf{y}+1)\mathbf{v}}{16}\right) \quad (2.1)$$

$$\mathbf{f(x,y)} = \frac{1}{4}\sum_{x=0}^{7}\sum_{y=0}^{7} \mathbf{C(u)\,C(v)\,F(u,v)} \cos\left(\frac{\pi(2x+1)u}{16}\right)\cos\left(\frac{\pi(2y+1)v}{16}\right) \quad (2.2)$$

where $\mathbf{C(u), C(v)} = 1/\mathrm{sqrt}(2)$ for u,v=0
and $\mathbf{C(u), C(v)} = 1$ otherwise

The Block DCT transforms each 8x8 block of pixel values from the original image into an 8x8 block of transform coefficient. Once a block has been transformed, the respective transform coefficients are quantized appropriately.

2.5.2 Quantization

Transform coefficients obtained from the transformation stage are assigned a finite number of quantization or reconstruction levels. For instance, an 8-bit representation of output values generated by the transformer/mapper provides a finite set of 256 reconstruction levels, which can be distributed uniformly or non-uniformly over the dynamic range of these values. Optimum distribution of these reconstruction values can be determined by either minimizing the quantization error, i.e. difference between the actual and corresponding quantized value, or perceptually weighted to suit the HVS characteristics. Quantization of individual values independently is referred to as *scalar quantization*, while quantization of entire sub-blocks of data is referred to as *vector quantization*. According to Shannon's rate-distortion theory, a better performance is always achievable in theory by coding vectors instead of scalars, even if the data is memoryless [5]. For vector quantization, however, a *codebook* is required to record reconstruction levels for coded blocks and this codebook must be available at the decoder for decoding.

The MPEG algorithm uses 8-bit scalar quantization and allows users to specify appropriate quantization levels for individual transform coefficients. The respective quantization level for each coefficient in the 8x8 block of transform coefficient is specified with a 8x8 quantization matrix for that block. To exploit the HVS for additional compression, transform coefficients corresponding to low spatial frequencies are finely quantized while transform coefficients corresponding to high spatial frequencies are coarsely quantized.

2.5.3 Code-word assignment

Following the quantization process, efficient codewords are assigned to each quan-

tized output to minimize any statistical redundancy that may be present in the data. This step is often referred to as source coding and the assigned codewords are generally of variable length. Variable Length Codes (VLC) assign codewords whose lengths are determined by the occurrence probability of each respective symbol and are more efficient since the average number of bits required for coding comes arbitrarily close to the source entropy.

The MPEG algorithm first converts the block of quantized transform coefficients into a long string of data using a zig-zag scanning pattern as illustrated in Fig.(2.5). This string of quantized values is then coded using Run-Length coding which results in significant compression since the quantized DCT coefficients in the lower right portion of the block generally turn out to be zeros and the zig-zag scanning results in long continuous strings of zeros. Finally, Huffman coding is applied to the resulting data to reduce any statistical redundancy in the final coded bit stream.

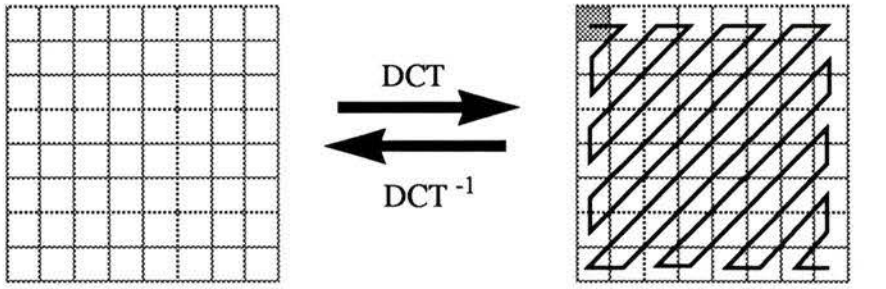


Figure 2.5 Zig-Zag scanning of DCT coefficients

2.6 Block Based Motion Estimation and Compensation

A significant amount of additional compression can be achieved for video sequences by coding individual frames based on information available from neighboring frames as they exhibit a high degree of similarity. One popular technique extensively used in interframe coding methods to exploits this similarity is block-based ME/MC. The basic idea of block based ME/MC is to code motion information of fixed size blocks[1] in

the frame being coded with respect to a nearby reference frame. This reference frame can either be a previous frame, resulting in forward prediction, or a future frame, resulting in backward prediction, and the respective reference frame has to be present at the decoder during decoding. In addition, bidirectional prediction is also possible, in which case a frame is coded using two reference frames, one in the future and one in the past. Only motion information that allows a frame to be reconstructed from its respective reference(s) frame is coded, and for sequences with moderate to high amount of motion, this results in a high level of compression.

Fig.(2.6) illustrates a typical forward motion compensation implementation where the current frame (frame 6) is coded based on the information in the corresponding reference frame (frame 3). The current frame is first partitioned into fixed size image blocks and if motion is detected for a block, motion estimation (ME) is performed to find a corresponding matching block in the reference frame. In the case of MPEG, only the Y luminance component is used in motion estimation. Fig.(2.7) illustrates an example showing blocks in the current frame and corresponding matching blocks in the reference frame. The translational difference between a block and its match is termed as the Motion Vector (MV).

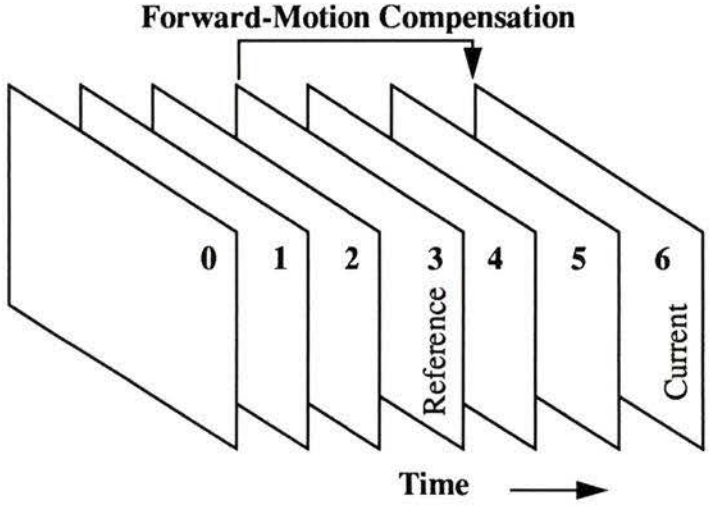
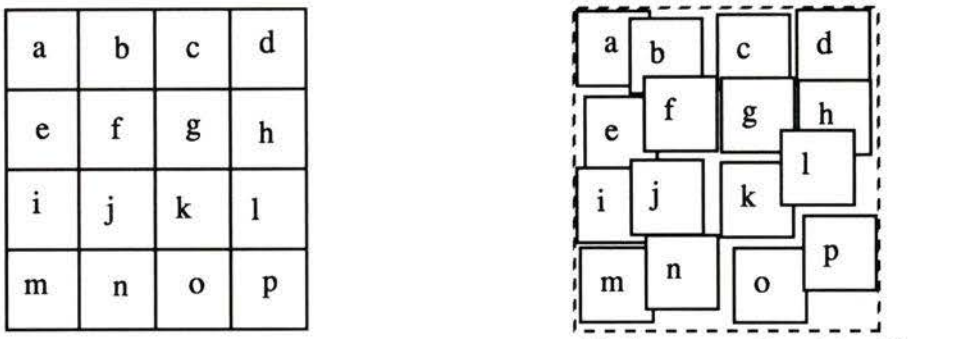


Figure 2.6 A typical forward motion compensation implementation for video sequence with frame #3 as the reference frame

1. While the MPEG standard defines a block to be 8x8 pels, this term, when used in the context of motion estimation, refers to blocks of any size.



(a) Partitioned blocks in Current Frame

(b) Corresponding Matching blocks in Reference Frame

Figure 2.7 Partitioned blocks in current frame with corresponding matching blocks in reference frame

The process of computing motion vectors for blocks, motion estimation, is illustrated in Fig.(2.8). A search area is first setup around the respective block's position in the reference frame and then, using a suitable matching criterion, the closest matching block is located. In general, the Mean Absolute Difference (MAD) between the two blocks, one in the current frame and the other in the reference frame, is preferred as a matching criterion. Although other matching criterion, for example Mean Square Error (MSE) and Cross-correlation, have been used, MAD provides equivalent performance at reduced computational complexity since it does not involve multiplications or divisions[25]. For two NxN pixel blocks, s_k and s_{k-1}, displaced by <dx,dy>, the MAD is computed as

$$MAD\,(dx, dy) = \frac{1}{N^2} \sum_{x=0}^{(N-1)} \sum_{y=0}^{(N-1)} \left| s_k(x, y) - s_{k-1}(x + dx, y + dy) \right| \qquad (2.3)$$

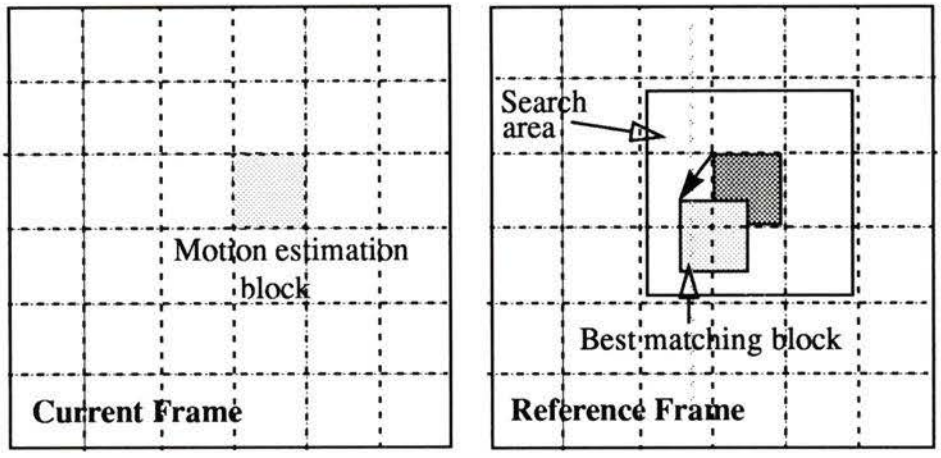


Figure 2.8 Motion Estimation of blocks

An exhaustive or full search of the search area locates the motion vector. An exhaustive search at *full pixel accuracy* implies that every block located at an integer pel position in the search area is compared with the block to be matched. Motion Estimation is also performed at *sub-pixel accuracy* where blocks located between integer pel positions are also compared. Such blocks are interpolated from surrounding blocks at integer pel positions.

Since Eq.(2.3) is computed every time any two blocks are compared, one can immediately senses the enormous amount of computation required for ME. Moreover, computational complexity of full searches increases quadratically with respect to the search area which is chosen in regard to motion displacement anticipated between the reference frame and the frame being coded. The choice of the reference frame is determined by the level of inter-frame dependence required. Video coding algorithms such as MPEG-I that incorporate frame interpolation techniques tend to use distant frame as opposed to immediate neighboring frames for reference.The further the reference frame is from the frame being coded, the larger the search area becomes.

Once the motion vector for a particular block has been computed, motion compensation is achieved by coding only the motion vector and corresponding prediction error terms i.e. residual difference between the block and its corresponding match. Still image compression techniques are generally applied to code the resulting prediction error while corresponding motion information is differentially coded to take advantage of correlation

between neighboring motion vectors. The motion information, thus, becomes part of the coded information.

2.6.1 Forward Prediction Example

An illustrative forward prediction example using frames from a real video sequence is presented here. A forward prediction is performed for frame #2 in Fig.(2.10) using the previous frame, frame #1 in Fig.(2.9), as the reference frame. Fig.(2.11) shows the difference between the two frames without ME/MC. An exhaustive search with a search range of 16 pels around the original block and Mean Absolute Difference (MAD) as the matching criterion is used for ME. The size of the frames is 352x240 pels and are partitioned into 16x16 pel blocks for ME/MC. Fig.(2.12) and Fig.(2.13) show the prediction errors and corresponding motion vectors respectively. In a coding application, to code frame #2, only the prediction error and corresponding motion vectors would be coded. This information is enough to reconstruct frame #2 from frame #1 at the decoder.

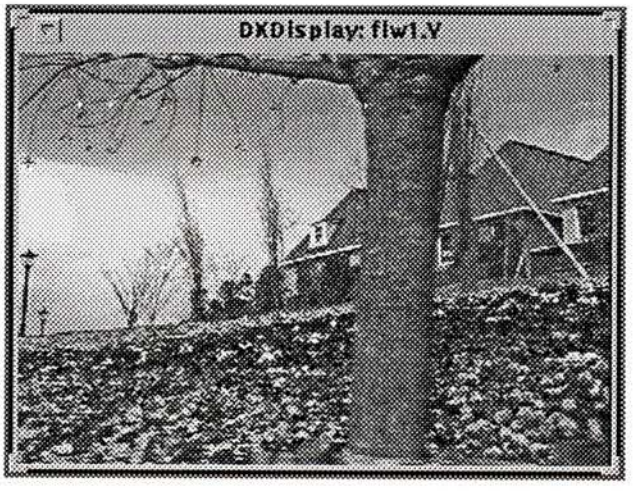


Figure 2.9 Frame #1 from the "flower-bed" video sequence

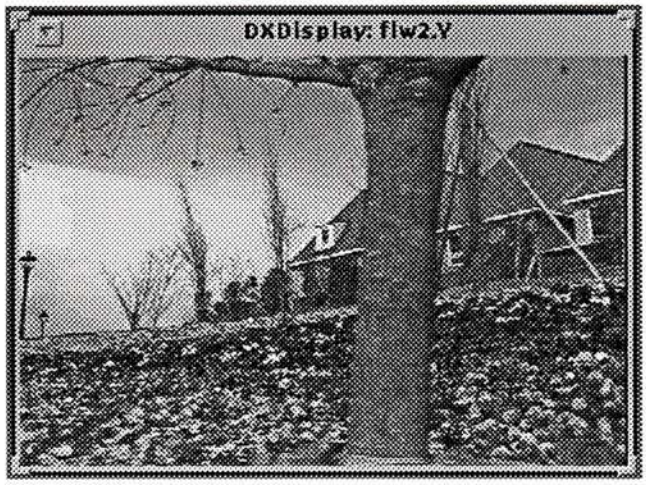


Figure 2.10 Frame #2 from "flower-bed' sequence

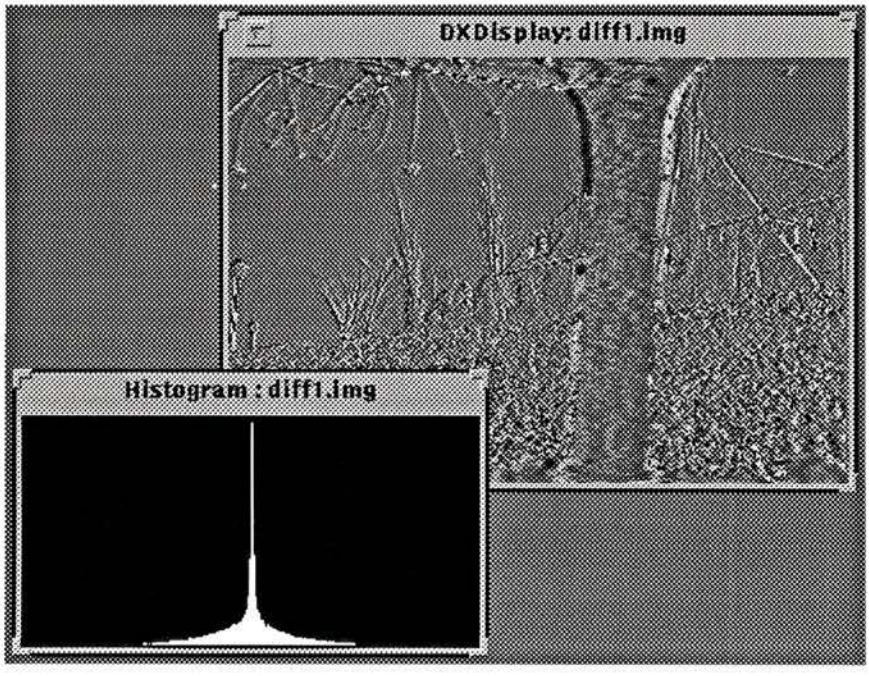


Figure 2.11 Differential error between frame #1 and frame #2.

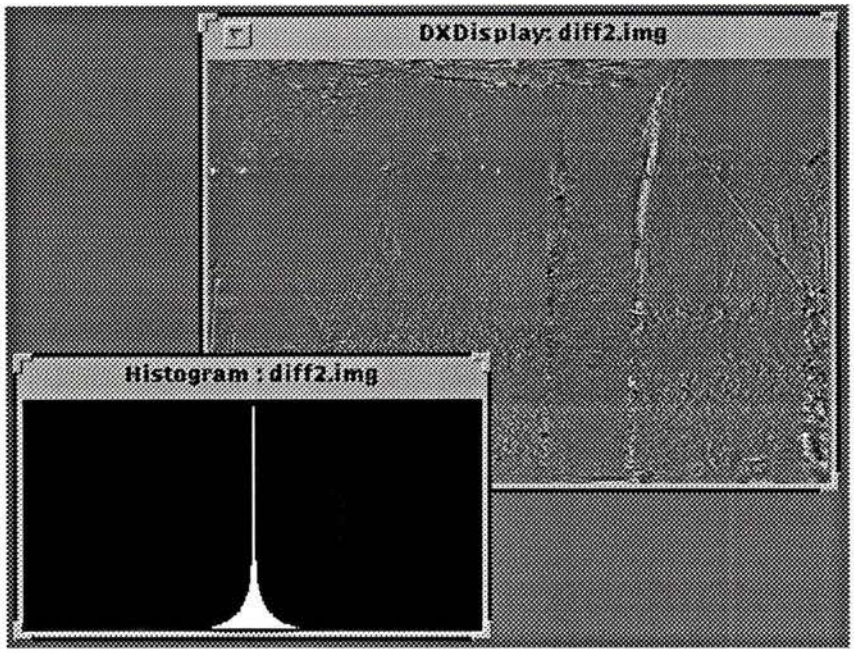


Figure 2.12 Motion compensated prediction error between frame#1 and frame #2.

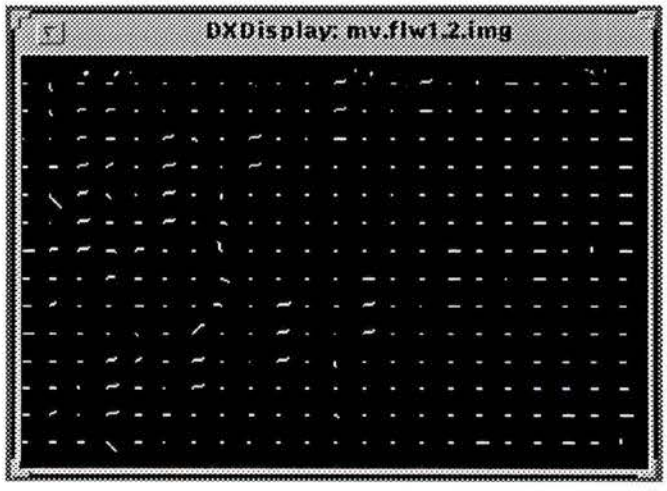


Figure 2.13 Motion Vectors for 16x16 pel blocks in frame #2 computed using exhaustive search

It is clear from Fig.(2.11) and Fig.(2.12) that motion compensated prediction error contains mainly high frequency changes between the two frames as compared to the simple frame difference. The fact that the human eye is less sensitive to high-frequency com-

ponents can be exploited by using coarse quantization for the motion compensated prediction error to achieve greater compression without sacrificing the visual quality of reconstructed data. Blocks that cannot be predicted efficiently from reference frames are coded as Intra-blocks, i.e. no ME/MC is performed. For image sequences with moderate to high motion, accurate ME/MC provides substantial coding gains without visually degrading the coded data [23].

Backward prediction is exactly the same as forward prediction, the only difference being that a future reference frame is used as reference. For instance, in this case, frame #2 could be used as the reference frame for backward prediction of frame #1.

In the case of bidirectional prediction, two reference frames, one in the future and one in the past, are used for prediction. A forward prediction is performed using the previous reference frame and backward prediction is performed using the future reference frame. This results in two sets of motion vectors and the coder decides, for each individual block, which motion vectors to use for better prediction. Thus, bidirectional prediction requires twice the computation of either forward or backward prediction.

2.7 Temporal Compression in MPEG-I

The MPEG-I and MPEG-II coding standards utilize Block-Based ME/MC to remove temporal redundancy. To achieve both high compression, possible only with inter-frame coding, and random accessibility of coded frames, best satisfied with intra-frame coding, the MPEG-I standard for video compression recognizes three different types of frames: Intra-frames (I), Predicted-frames(P) and Interpolated frames (B - Bidirectional Predicted frames). The I-frames are coded as still images and provide points for random accessibility. The P-frames are coded using forward-motion compensation with immediate previous I or P-frames as reference frames. The B-frames, on the other hand, are coded using bi-directional ME/MC with immediate surrounding I or P frames as reference frames, i.e. using both forward- and backward-motion compensation from a previous and a future frame respectively. ME is performed using macroblocks (16x16pels) over the luminance component of input video. The motion activity in the chrominance components is assumed to correspond directly with that in the luminance component of the video.

In a typical MPEG implementation, the input video is grouped into Group Of Pictures (GOP) as illustrated in Fig.(2.14). The user has the choice of selecting the ordering and type of frames within a GOP. Experimental results indicate that the use of B-frames provide the greatest amount of compression[20].

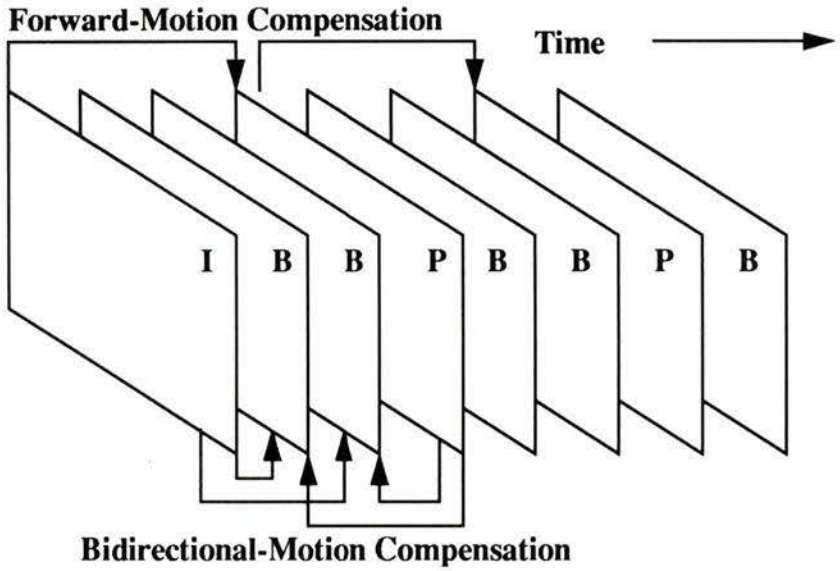


Figure 2.14 Types for frames specified by MPEG syntax

Every macroblock in I-frames is of type Intra, while macroblocks of types Intra and Forward-predicted make up P-frames. The B-Frames are composed of macroblocks of type Intra, Forward-predicted, Backward-predicted and Average. The Average macroblock uses the average of a forward and backward predicted macroblock as a prediction for the particular macroblock. Thus, motion information consists of one motion vector for forward-predicted and backward-predicted macroblocks, and two motion vectors for bi-directionally predicted macroblocks. After ME/MC, resulting motion vectors are coded differentially while corresponding prediction errors are coded as still images using the DCT-block based image coder presented in Sec.(2.5). Simulations indicate that ME/MC produces good result for reference frames which are at most 1/10 seconds away from the frame being coded [14].

The use of future frames as references in bi-directionally predicted B-frames necessitates the need to re-order frames within the coded sequence for efficient decoding and to

reduce de-coding delays. As an example, consider a thirteen frame sequence composed of frames 0 to 12 and a user specified GOP of IBBPBBPBBPBBI. Each frame is respectively coded as $I_0B_1B_2P_3B_4B_5P_6B_7B_8P_9B_{10}B_{11}I_{12}$, where the subscripts refer to the corresponding frame position in the sequence, and transmitted or stored in the following order: $I_0P_3B_1B_2P_6B_4B_5P_9B_7B_8I_{12}B_{10}B_{11}$. Thus, both reference frames for corresponding B-frames are decoded before respective B-frames are decoded.

2.8 MPEG Coder and Decoder Implementation

The MPEG-I standard specifies only the semantics and syntax of the MPEG bitstream and the signal processing in the decoder [20], and does not specify an encoding process. Thus, encoder implementations have the option to trade-off cost and speed against picture quality and coding efficiency. The main functional blocks of a typical MPEG coder and decoder implementation are shown in Fig.(2.15) and Fig.(2.16) respectively.

The frames are first re-ordered for prediction as is necessary for bidirectional prediction. Motion estimation is performed for each macroblock and the corresponding mode and motion vector(s) coded. The mode indicates the type of macroblock, i.e. Intra, Forward Predicted, Backward predicted or Average. The motion compensated error is then computed using the matching block from the reference frame that has been decoded and held in the Frame Store/Predictor block at the bottom of the Fig.(2.15). The decoding of the reference frame in the coder ensures that the reference frame used for prediction is identical to that available at the decoder. The motion compensated error, motion vectors and corresponding modes are then multiplexed and sent to the buffer. Since the resulting coded bitstream prior to buffering is of variable bit rate, as a result of Variable Length Coding, the buffer can be used to provide a constant bitrate at the output. Constant bitrates are desirable in transmission applications.

The decoding of the coded bitstream is simple and essentially reverses the coding procedure.

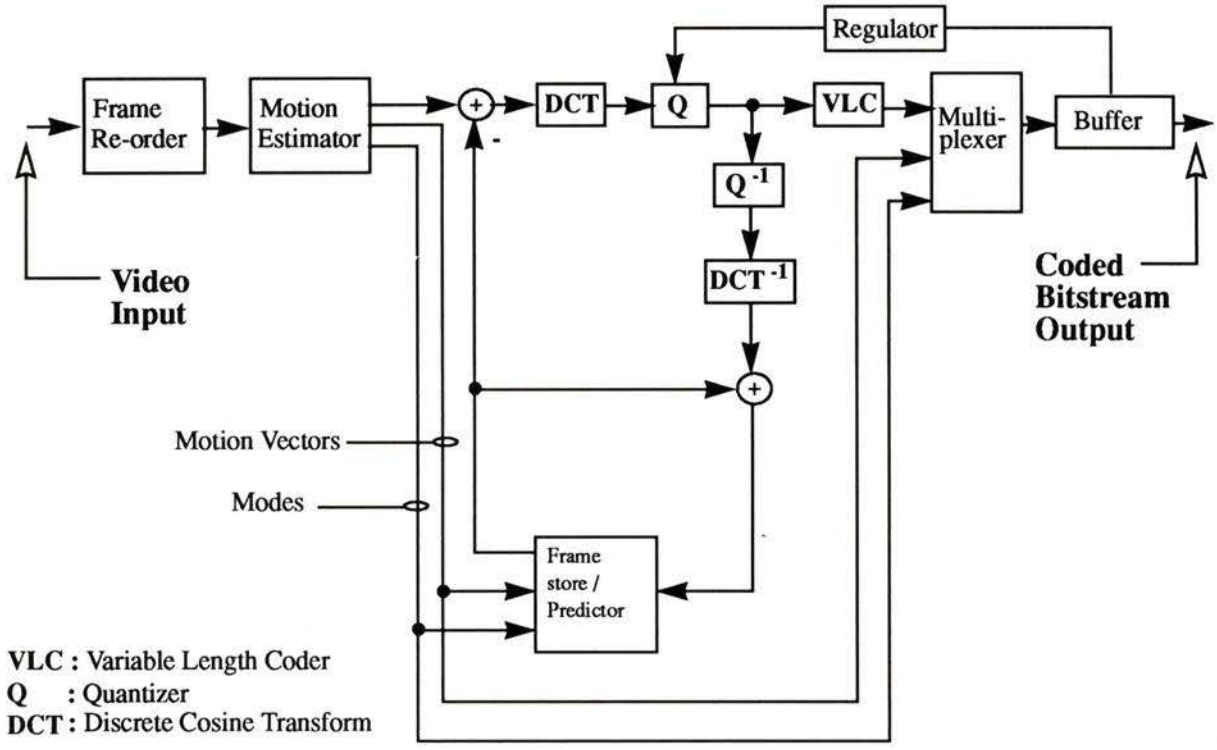


Figure 2.15 Block diagram of MPEG-I encoder [20]

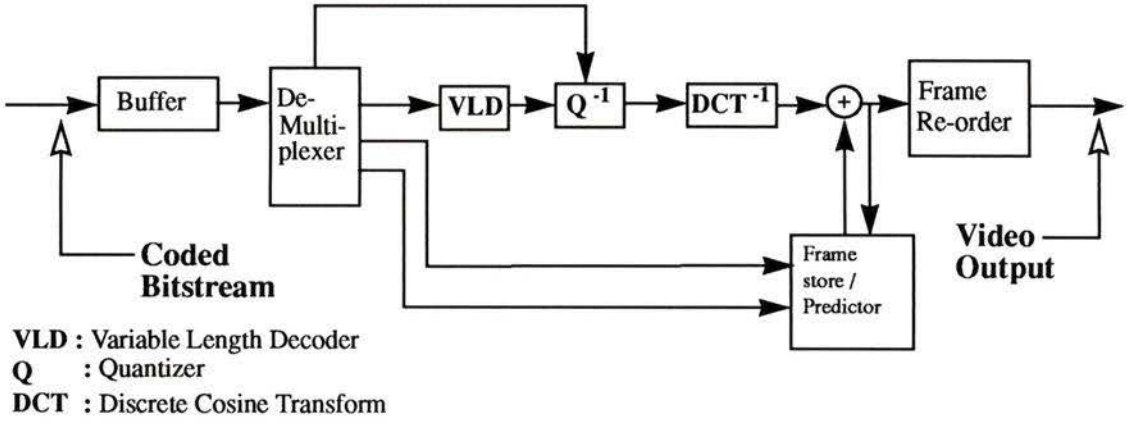


Figure 2.16 Block diagram of MPEG-I decoder [20]

2.9 The Need for Efficient Motion Estimation

Extensive use of ME/MC in the MPEG-I algorithm, outlined in this chapter, and other proposed video coding algorithms [16]-[19] has generated an interest in efficient motion estimation techniques. Real-time motion estimation using full-searches at video rates demands exceedingly high computation capability, achievable only with dedicated hardware. Hardware solutions such as [17] incorporate multiple processing units and are costly to implement. Also, these hardware solutions provide ME over limited search ranges.

An alternative approach for efficient ME estimation is to considerably reduce the number of search points considered during ME. Efficient search algorithms can be used instead of full searches to search the search area for matching blocks. A number of such search algorithms, collectively referred to as Block Matching Algorithms (BMAs), have been proposed and are the focus of this thesis. With the advent of powerful DSP chips, such as the MVP from Texas Instruments [22] and similar for other manufacturers, BMAs make it feasible to implement ME on such DSP chips and thereby reducing, considerably, the cost of building coders.

A comprehensive evaluation of the these BMAs in comparison to corresponding exhaustive searches, which provide the most accurate ME, is essential since motion estimation and compensation is effective only when it is accurate. Such an evaluation is presented in the next chapter which introduces some robust BMAs and evaluates them for use in motion estimation.

Chapter 3

Evaluation of some Block Matching Algorithms for Motion Estimation

3.1 Introduction

Motion estimation and compensation forms an integral part of coders that utilize inter-frame coding techniques to compress digital video. For image sequences with moderate to high motion, the use of motion estimation and motion compensation (ME/MC) provides substantial coding gains over simple frame difference coding [37]. From the two general classes of ME algorithms, pel-recursive and block matching algorithms (BMAs), the latter have gained greater popularity as a result of their computational and implementation simplicity. Emerging video coding standards such as CCITT recommendation for low-bit coding at pX64 kb/s [15] and the ISO standard for multimedia applications at 1.5Mb/s [20], both based on block-transform coding techniques, incorporate block-based motion estimation and compensation to reduce temporal redundancy.

Motion estimation, however, is computationally intensive and forms a bottle neck in the video-coder's throughput. Exhaustive searches to locate matching blocks in reference frames require an immense amount of computation. For real time performance at video rates, dedicated hardware with multiple processing units, as presented in [17], is required. To overcome this otherwise extensive computational load, efficient search algorithms, collectively referred to as Block Matching Algorithms (BMAs), have been proposed that attempt to provide efficient ME at a reduced computational load. Although BMAs generally provide sub-optimal results with respect to corresponding exhaustive searches, they are nonetheless essential for implementing ME/MC on general purpose computers or DSP chips. Efficient BMAs have been reviewed in current literature [23]

[29], however, substantive evaluation of their performance in terms of estimation accuracy, especially for frame prediction using distant reference frames and larger search ranges, has not been presented. Such an evaluation is invaluable for selecting between these BMAs for ME in implementation of emerging video coding standards.

In this chapter, a representative selection of such robust search algorithms are evaluated and corresponding results presented. The evaluation is targeted at the use of BMAs with the MPEG I and other related video coding standards. Simulation tests are devised to investigate and characterize the performance of each BMA in terms of estimation accuracy and computational complexity over different prediction settings using two video sequences as benchmarks. These two sequences are chosen because they represent cases of motion that appears in many video sequences. The particular algorithms tested include: Three Step Search [24], 2-D Logarithmic Search [23], One Time Search [25], Orthogonal Step Search [27], Cross Search [26] and Modified Three Step Search [29]. A brief description of each BMA is presented and the evaluation criterion applied are discussed. The simulation tests are also presented and corresponding results discussed. Finally, the performance of respective search algorithms are compared.

3.2 Block Matching Algorithms

Block matching algorithms use efficient search algorithms to locate matching blocks in reference frames and provide computationally efficient alternatives to exhaustive searches for motion estimation. The following subsections provide an overview of BMAs selected for evaluation. These algorithms are representative of the general approaches used in BMAs and include Three Step Search (TSS), 2-D Logarithmic Search (TDL), One Time Search (OTS), Orthogonal Step Search (OSS), Cross Search Algorithm (CSA)and Modified Three Step Search (MTSS). In presenting these algorithms, the following symbols are used repeatedly:

w = Search Range around current macroblock; This parameter determines the corresponding search area for ME

d = Step Size; This parameter controls the progression of respective search algorithms towards a match

The top-left corner of a macroblock is used to denote its position. In Fig.(3.1), the

search area for the macroblock is illustrated. The macroblock at location (i_0,j_0) in the current frame is compared with macroblocks in the shaded area in the reference frame.

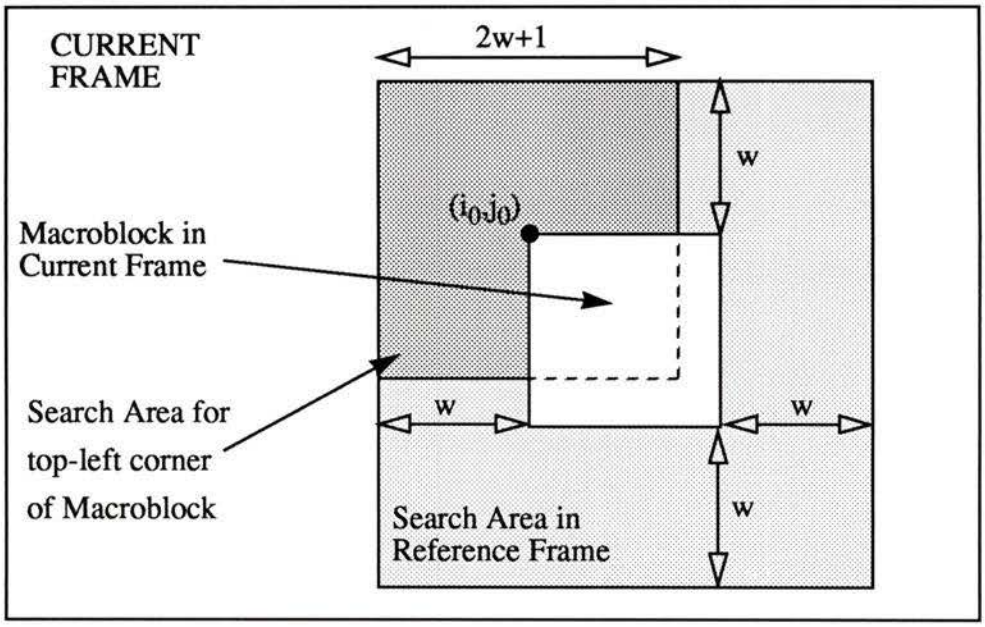


Figure 3.1 Search Area for macroblocks

3.2.1 2D Logarithmic (TDL)

Jain and Jain proposed the 2D logarithmic algorithm in 1981 and used Mean Square Error (MSE) as the matching criterion [23]. The following steps describe the implementation of a TDL search algorithm with Mean Absolute Difference (MAD), see Eq.(2.3), as the matching criterion:

1. Initialize Search: Initial Point ($i=i_0$, $j=j_0$), $d=d_0$, Search Range = w
2. Compute MAD at five surrounding points (i, j), ($i\pm d$, j), (i, $j\pm d$) and select (i_{min}, j_{min}) from the five points for which MAD is lowest.
3. If (i_{min}, j_{min}) is the centre point or at the boundary of the search area, reduce d: d=d-1. Set new initial point ($i=i_{min}$, $j=j_{min}$).
4. if d=0 stop search, otherwise go to Step 2.

Fig.(3.2) shows the progression of the TDL search algorithm, initiated with (i_0,j_0) at the origin, $d_0=2$, and w=6 pels, as it converges to a possible motion vector (5,6). The

motion vector corresponds to the translational difference between the original macroblock and its match which, in this case, is located 5 pels to the right and 6 pels on top of the original macroblock position. This motion vector is obtained after 6 iterations.

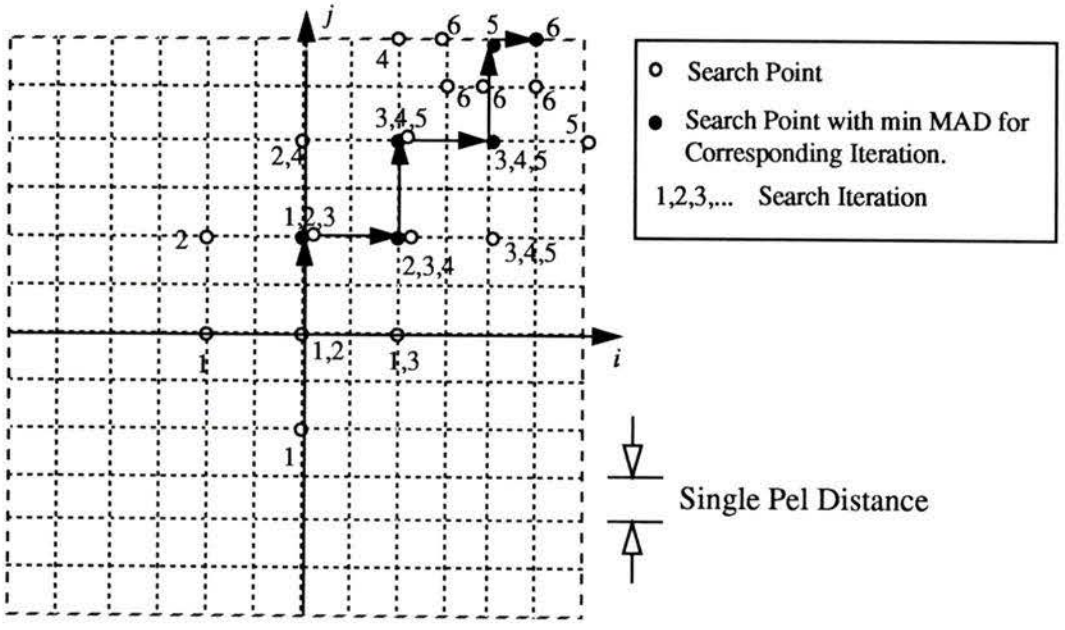


Figure 3.2 2-D Logarithmic Search Algorithm

3.2.2 Three Step Search (TSS)

Koga proposed the Three Step Search algorithm [24] at about the same time as Jain and Jain proposed their algorithm. The original proposal, incorporated in a video coder, used MAD as the matching criterion. TSS implementation steps are outlined below:

1. Initialize Search: Initial Point ($i=i_0$, $j=j_0$), d=3, Search Range = 6.
2. Compute MAD at nine surrounding points (i, j), (i±d , j), (i, j±d), (i±d , j±d) and select (i_{min},j_{min}) from the nine points for which MAD is lowest.
3. Reduce d: d=d-1. Set new initial point ($i=i_{min}$, $j=j_{min}$).
4. if d=0 stop search, otherwise go to Step 2

The original TSS algorithm, presented in the preceding steps, was implemented for search range of 6 pels with the best match being located after three such steps; hence the

name Three Step Search. Fig.(3.3) shows a possible converging path of the TSS algorithm.

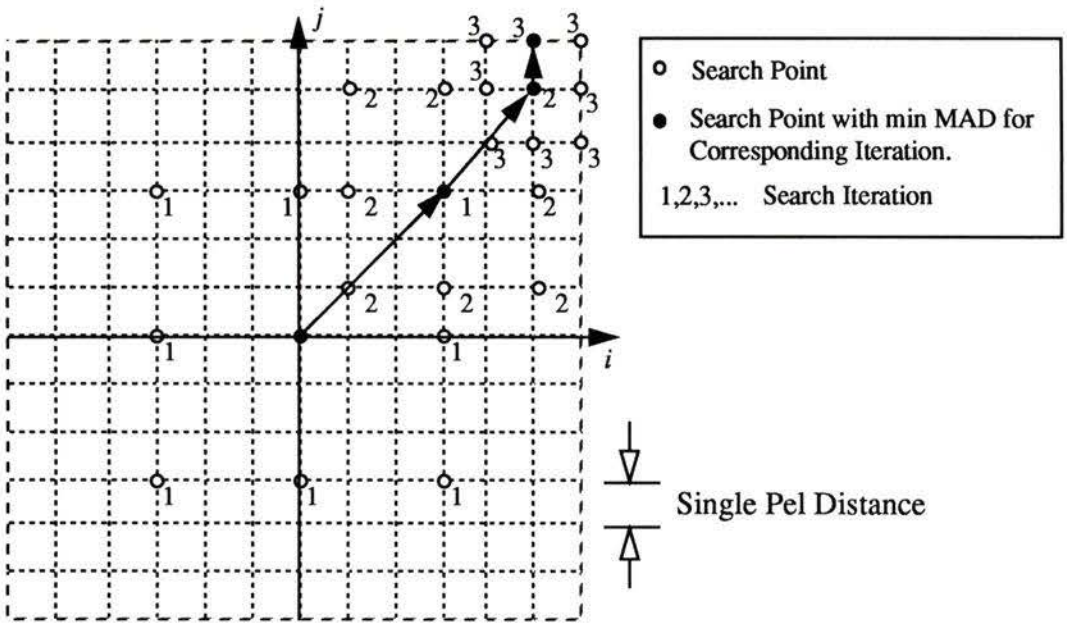


Figure 3.3 Three-Step Search Algorithm

A slightly different version of the above algorithm, the Modified Three Step Search (MTSS) [29], has also been proposed. It differs from the TSS in the way it reduces the step size between consecutive searches and works with arbitrary search ranges. MTSS implementation follows directly form the TSS implementation except that, the search is initiated with search range=w, d_0=w/2 and, in Step 3, d is reduced by half. i.e. d=d/2.

3.2.3 One Time Search (OTS)

The OneTime Search algorithm was proposed by Srinivas and Rao in 1985 as part of the Conjugate Direction Search Algorithm [25]. OTS first seeks a minimum in the horizontal direction and then follows to seek a minimum in the vertical direction. The implementation steps are outlined here:

1. Initialize Horizontal Search: Initial Point ($i=i_0$, $j=j_0$), d=1, Search Range = w.
2. Compute MAD at (i, j), (i+d, j), (i-d, j) and select (i_{min},j_{min}) for which MAD is lowest.

3. Set ($i=i_{min}$, $j=j_{min}$). If (i_{min},j_{min}) was not the center point, go to Step 2.
4. Compute MAD at (i, j), (i, j+d), (i, j-d) and select (i_{min},j_{min}) for which MAD is lowest.
5. Set ($i=i_{min}$, $j=j_{min}$). If (i_{min}, j_{min}) was not the center point, go to Step 4 otherwise terminate search.

3.2.4 Orthogonal Step Search (OSS)

The Orthogonal Step Search [27] algorithm was proposed by Puri, Hang and Schilling in 1987. Its implementation is outlined in the following steps:

1. Initialize Search: Initial Point ($i=i_0$, $j=j_0$), Search Range = w, d=w/2.
2. Horizontal Step. Compute MAD at (i, j), (i+d, j), (i-d, j) and select (i_{min},j_{min}) for which MAD is lowest. Set ($i=i_{min}$, $j=j_{min}$).
3. Vertical Step. Compute MAD at (i, j), (i, j+d), (i, j-d) and select (i_{min},j_{min}) for which MAD is lowest. Set ($i=i_{min}$, $j=j_{min}$).
4. If d>1 reduce d: d=d/2 and go to Step 2 otherwise stop search.

Fig.(3.4) shows the progression of OSS search, initiated with (i_0,j_0) at the origin and w=6, as it terminates at the search point (5,6).

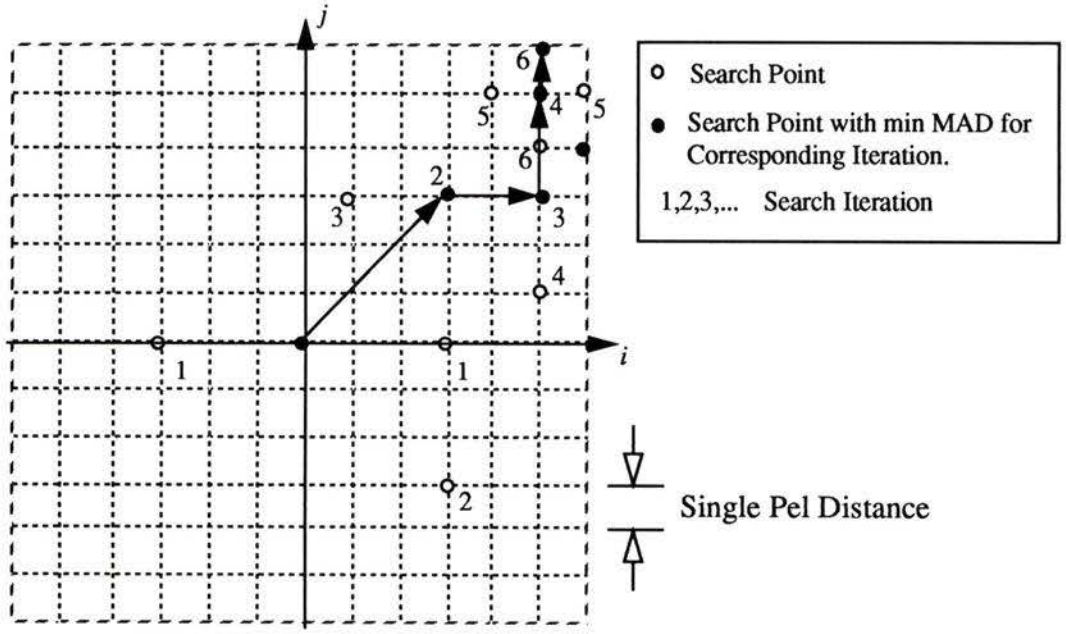


Figure 3.4 Orthogonal Search Algorithm

3.2.5 Cross Search Algorithm (CSA)

The Cross Search Algorithm [26] was proposed by Ghanbari in 1990. The CSA is similar to the MTSS algorithms except that it progresses by searching only diagonal search points. Its implementation is outlined here:

1. Initialize Search: Initial Point ($i=i_0, j=j_0$), Search Range = w, d=w/2.
2. Compute MAD at five surrounding points (i, j), (i+d, j±d d), (i-d, j±d d) and select (i_{min}, j_{min}) for which MAD is lowest. Set ($i=i_{min}$, $j=j_{min}$).
3. if d=1 go to Step 4 otherwise reduce d: d=d/2 and go to Step 2.
4. if (i_{min}, j_{min}) in Step 2 was either (i, j), (i-1,j-1) or (i+1,j+1) then go to Step 6.
5. ($m=i_{min}$, $n=j_{min}$); Search points (m, n), (m-1, n),(m+1, n), (m, n-1) and (m, n+1) and select minimum.
6. ($m=i_{min}$, $n=j_{min}$); Search points (m, n), (m-1,n-1), (m-1,n+1), (m+1,n-1) and (m+1,n+1) and select minimum.

Fig.(3.5) shows the progression of CSA search initiated with w=6.

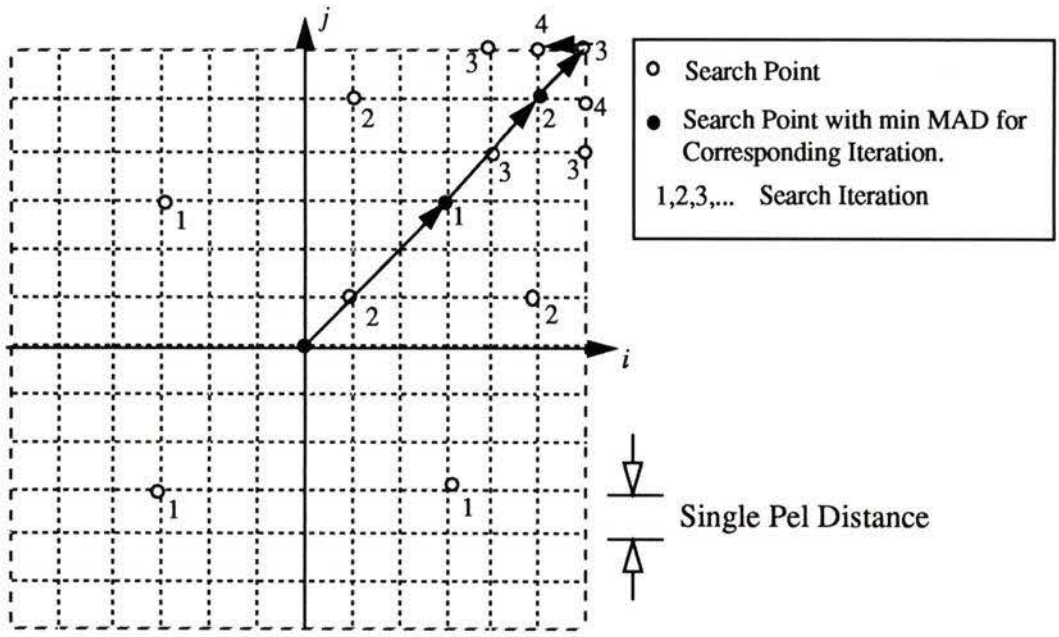


Figure 3.5 Cross Search Algorithm

3.2.6 Discussion

Block Matching Algorithms provide a substantial computational saving over exhaustive searches. As an example, consider a search range of 8 pels around a macroblock for motion estimation: an exhaustive search requires 289 block matches while the Three Step Search BMA requires only 27 block matches to locate a corresponding match. A computational saving of an order of magnitude. Such computational savings are even greater when larger search ranges are involved. BMAs have different computational complexities and some require a larger amount iterations than others to locate a final match. The number of these intermediate iterations are either fixed, i.e. independent of the intermediate search results, or variable and dependent on intermediate searches. Table (3.1) outlines the worst case computation complexity of these algorithms.

Algorithm	Max. No. of Search Points	Intermediate Iterations
Exhaustive Search	$(2w + 1)^2$	No Iterations
2-D Logarithmic	$2 + 7 \log_2 w$	Variable
Three Step Search	$1 + 8 \log_2 w$	Fixed
One Step Search	$3 + 2w$	Variable
Cross Search	$5 + 4 \log_2 w$	Fixed
Orthogonal Search	$1 + 4 \log_2 w$	Fixed

Table 3.1. Maximum number of Search Points checked by respective BMAs for Search range=w (adapted from [26])

As mentioned earlier, BMAs are not as accurate as exhaustive searches. The search algorithms used by BMAs are typically iterative in nature and assume the MAD between macroblocks to increase monotonically as the search moves away from the match [28]. This assumption, which is not always true, directly results in their sub optimal performance. Also, some BMAs converge quickly on local minimas while others are not able to reach every point within the search area. Inaccurate ME results in poor compression performance and causes motion artifacts in decoded video. In selecting between these BMAs for ME implementation, a comprehensive evaluation to determine their accuracy and computational complexity is vital. Such an evaluation is presented here. The criteria used in evaluating respective BMAs are discussed in the next section, after which, simulation tests and corresponding results are presented.

3.3 Evaluation Criteria

The objective of the simulations was to determine how well each search algorithm performed for different ME/MC prediction settings. This performance was measured using the following criteria: estimation accuracy and computational complexity.

Estimation accuracy measures the accuracy of matches located. It was measured by computing the first order entropy of prediction errors generated after ME/MC (see Sec.(2.6.1)). Lower entropy indicates accurate matches and results in higher compression. The first order entropy (H) of errors values, each with probability of occurrence p_i, is given by

$$H = -\sum_{i=1}^{N} p_i [Log_2 (p_i)] \tag{3.1}$$

where N bounds all possible error values. The histogram of prediction errors is used to estimate p_i. As evident from the histogram in Fig.(2.12), the binning of these prediction errors results in an inverse Laplacian distribution with the peak centered at zero. Since 8-bit values are used to represent the Y-frame, the prediction errors can range in value from -255 to +255. However, for the simulations, only a prediction error range of -127 to 128 was considered as it is more than sufficient, thus, making N=256.

Computational complexity, on the other hand, determines the level of computational effort required to undertake the search. It was measured by recording the number of block matches required for corresponding frame predictions and is presented in terms of Number of Block Matches/frame. Computationally efficient and accurate motion estimation is highly desired.

The simulation tests used to evaluate respective BMAs are presented after the following section which describes the two test video sequences used in these simulations.

3.4 Test Video Sequences

All BMAs were tested over two 352x240 pel resolution, 30 frames/sec, video sequences: "flower bed" and "tennis". Appendix I includes every fifth frame from these sequences. The "flower bed" sequence is essentially a view from a moving camera as it

passes a flower bed with a house in the background and a tree in the foreground. It contains a substantial amount of horizontal panning which constitutes large amounts of horizontal motion at different scales within the sequence. The "tennis" sequence records a table tennis playoff that contains very little motion for the first 20 frames, while the rest of the sequence contains large amount of motion as a result of fast zooming and panning of the camera. Both these sequences provide a representative amount of motion present in video scenes and are well suited for testing Motion Estimation Algorithms.

3.5 Simulation Tests

Two simulation tests were designed for the evaluation of the following:

1. The influence of search range on the estimation accuracy of respective BMAs
2. Estimation accuracy and corresponding computational complexity of respective BMAs for frame prediction over real video sequences

Only forward prediction is considered since the direction of frame prediction does not influence the performance of BMAs. Also, block size of 16x16 pels is always used for ME/MC and corresponds to the size of the macroblock in the MPEG standard. ME is performed at full pixel accuracy.

To take into account practical video coding algorithms, the simulation tests consider two different frame prediction settings.

- $\Delta n = 1$: This setting predicts frames at (n) from frames at (n-1).
- $\Delta n = 3$: This setting predicts frames at (n) from frames at (n-3).

The latter simulates frame predictions involving distant reference frames as is the case in predicting P frames in a MPEG system configured to code frames in the sequence IBBPBBPBBPBBPBBI... where P frames are predicted from immediate previous I or P frames using forward prediction. The two simulation tests are presented here in greater detail:

TEST 1 : This test measures the influence of search range on the estimation accuracy of respective BMAs. A singe frame is considered for forward prediction from each of the two test sequences and forward prediction is performed for $\Delta n=1$ and $\Delta n=3$. The forward prediction is repeated using different search ranges for ME and corresponding

estimation accuracy of respective BMAs recorded.

- For the "tennis" sequence with $\Delta n=1$, frame 30 is used as the reference frame to predict frame 31, and with $\Delta n=3$, frame 30 is used as the reference frames to predict frame 33. The search range for the $\Delta n=1$ prediction setting extends from 4 pels to 16 pels while the search range for the $\Delta n=3$ prediction setting extends from 4 pels to 32 pels.
- For the "flower bed" sequence with $\Delta n=1$, frame 1 is used as the reference frame to predict frame 2, and with $\Delta n=3$, frame 1 is used as the reference frame to predict frame 4. Again, the search range for the $\Delta n=1$ prediction setting extends from 4 pels to 16 pels while the search range for the $\Delta n=3$ prediction setting extends from 4 pels to 32 pels.

TEST 2 : This test measures the estimation accuracy and corresponding computational complexity of respective BMAs for forward prediction over real video sequences. The following parameters settings are used for the frame predictions[1]:

Setting	Block Size (pels)	Search Range (pels)
$\Delta n=1$	16x16	8
$\Delta n=3$	16x16	16

Table 3.2. Motion Estimation Settings for Frame Prediction

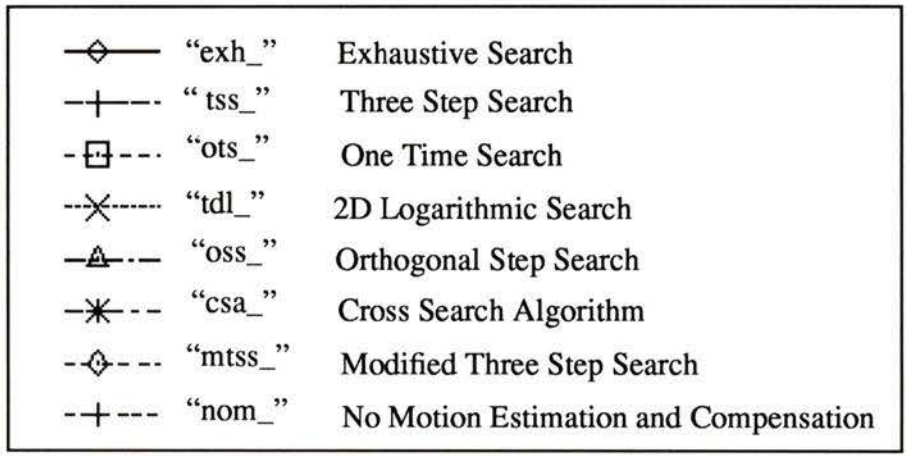
- For the "tennis" sequence, forward prediction is performed at full-pixel accuracy over the first 60 frames for both $\Delta n=1$ and $\Delta n=3$.
- For the "flower bed" sequence, forward prediction is performed at full-pixel accuracy over the first 30 frames for both $\Delta n=1$ and $\Delta n=3$.

In the case of $\Delta n=1$ prediction setting, frame 2 is predicted from frame 1, frame 3 is predicted from frame 2, frame 4 is predicted from frame 3 and so on. In the case of $\Delta n=3$ prediction setting, frame 4 is predicted from frame 1, frame 5 is predicted from frame 2, frame 6 is predicted from frame 3 and so on. The estimation accuracy and corresponding computational complexity of every frame prediction is recorded for respective BMAs. The computational complexity is measured by recording the number of block matches performed for respective frame predictions.

---

1. For the TSS algorithm, a fixed search range of w=6 pels is always used for ME as defined in the original proposal of [24].

These simulation tests were performed for all BMAs and the results are summarized in the following sections. Estimation accuracy of corresponding exhaustive searches provides the benchmark for comparing the estimation accuracy of respective BMAs. The following legend is used in the figures presenting the results of these simulations:



It should be noted that the TSS algorithm always performs ME over a fixed search range of 6 pels and is included here to make the following performance comparison more complete.

3.6 Results for Prediction Setting of Δn=1

This sections presents the results for the prediction setting of Δn=1 i.e. frames at (n) are predicted from frames at (n-1). The results for the "tennis" sequence are presented first, followed by results for the "flower bed" sequence. Also, for the purpose of comparing computational gains of respective BMAs, it should be noted that an exhaustive search for the corresponding search range, w=8 pels as presented in Table(3.2), requires 84480 block matches/frame.

3.6.1 Results for "Tennis" Sequence (Δn=1)

The results for the two simulation tests: TEST 1 and TEST 2 for the "tennis" sequence are respectively presented in Figs.(3.6)-(3.8). For TEST 2, which measured estimation accuracy, results for the first 20 frames are omitted as all BMAs exhibited the same estimation accuracy and did not provide any coding gain over simple frame differences (i.e. no ME/MC). This is due to the fact that there is very little motion in the first 20

frames.

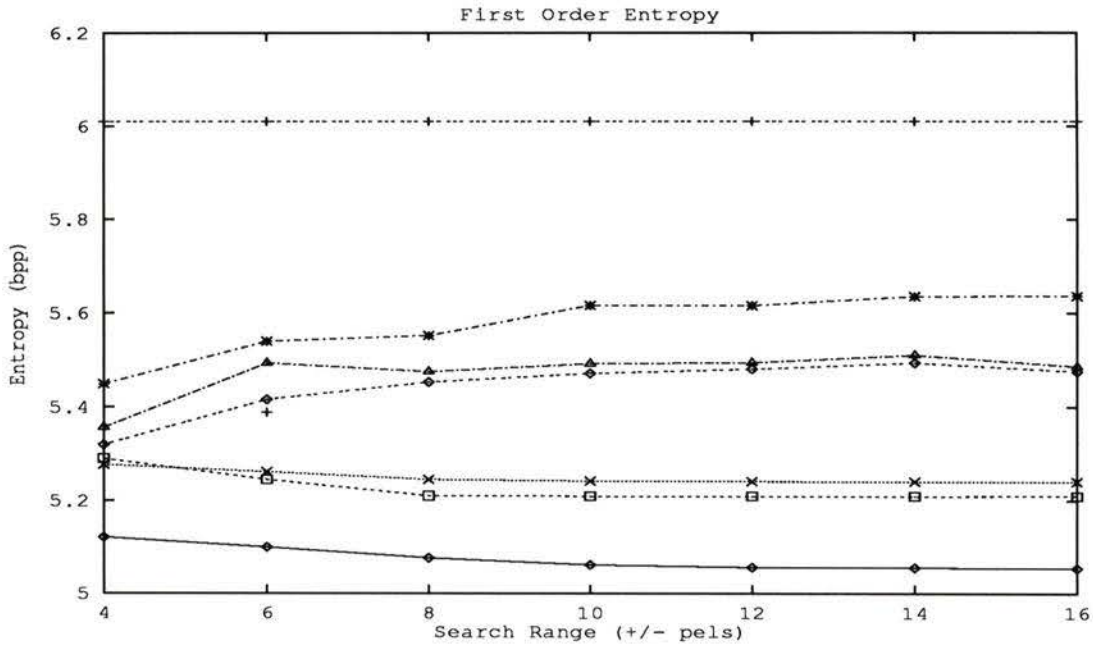


Figure 3.6 TEST 1 : Estimation accuracy v/s Search Range for "tennis" sequence with frame 31 predicted from frame 30.

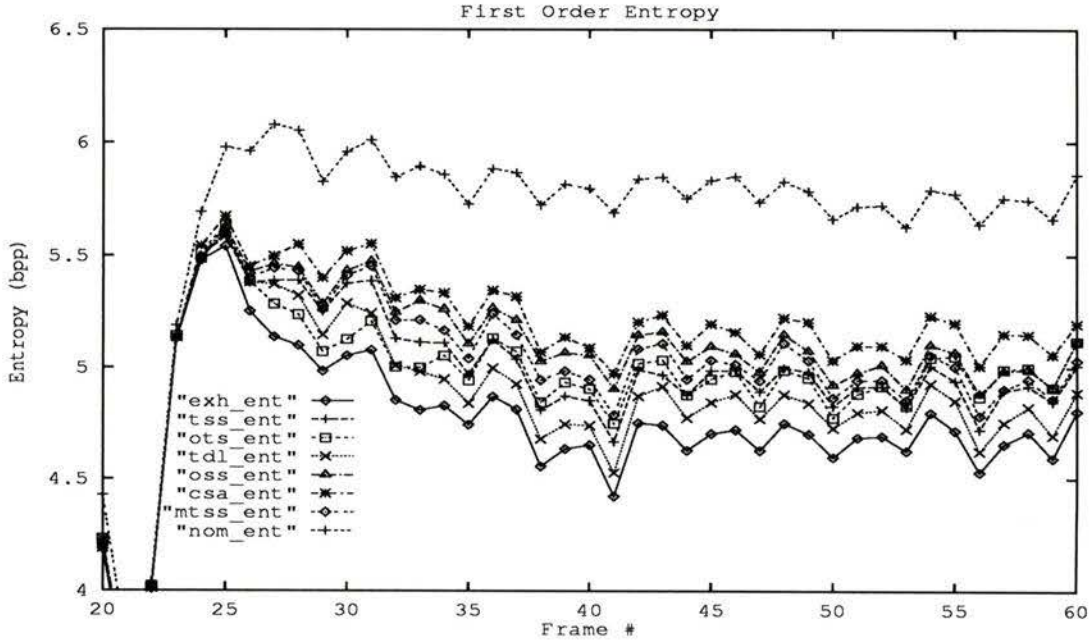


Figure 3.7 TEST 2 : Estimation accuracy for "tennis" sequence over frames 20-60 with Reference Frames at (n-1) and Search Range=8 pels.

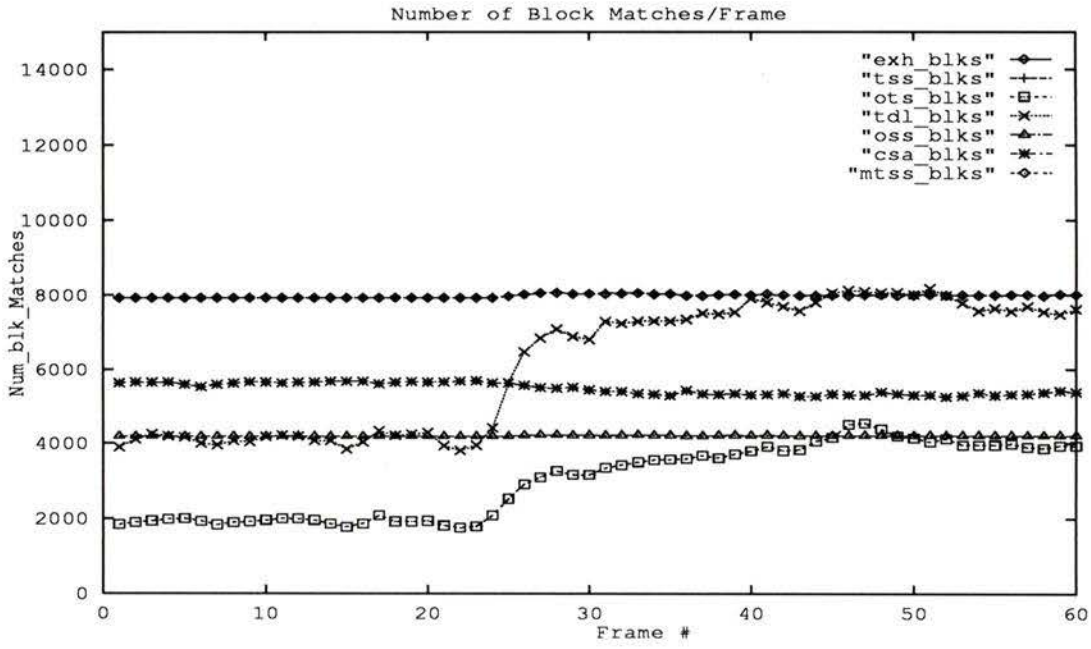


Figure 3.8 TEST 2 : Number of Block Matches/Frame for "tennis" sequence with Reference Frames at (n-1) and Search Range=8.

3.6.2 Results for "Flower Bed" Sequence ($\Delta n=1$)

The results for the two simulation tests: TEST 1 and TEST 2 for the "flower bed" sequence are respectively presented in Figs.(3.9)-(3.11).

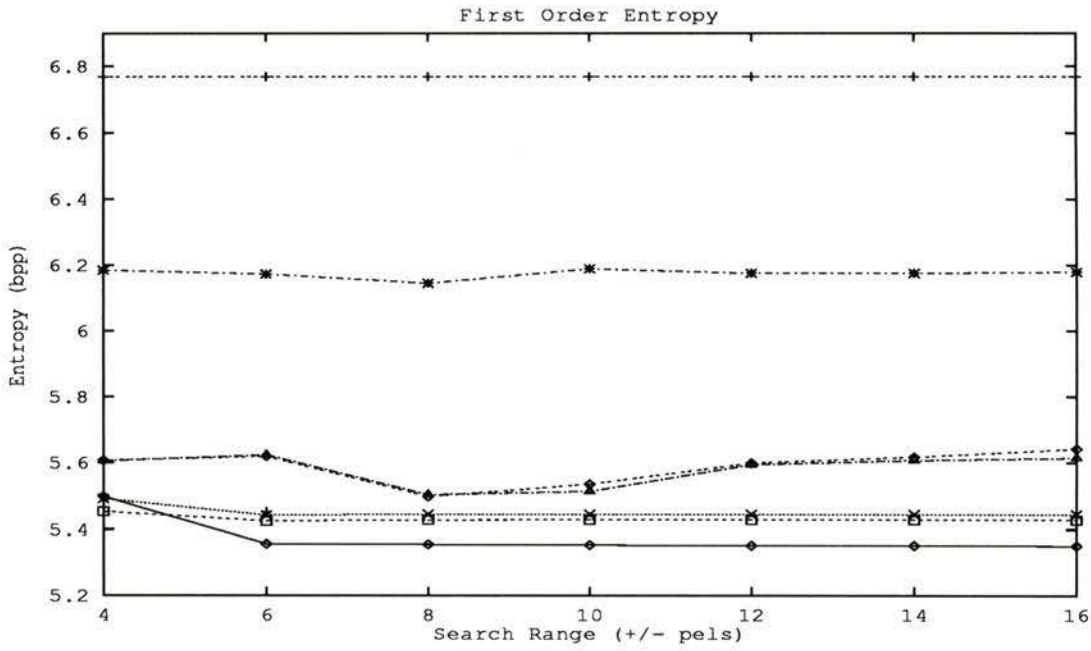


Figure 3.9 TEST 1 : Estimation accuracy v/s Search Range for "flower bed" sequence with frame 2 predicted from frame 1.

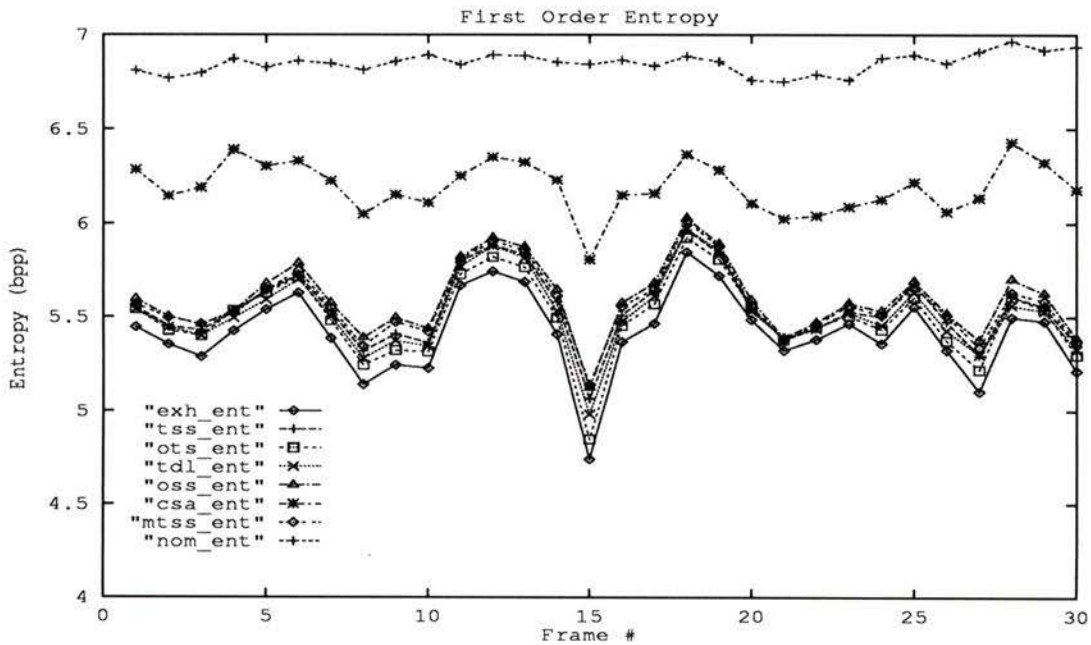


Figure 3.10 TEST 2 : Estimation accuracy for "flower bed" sequence over frames 1-30 with Reference Frames at (n-1) and Search Range=8 pels.

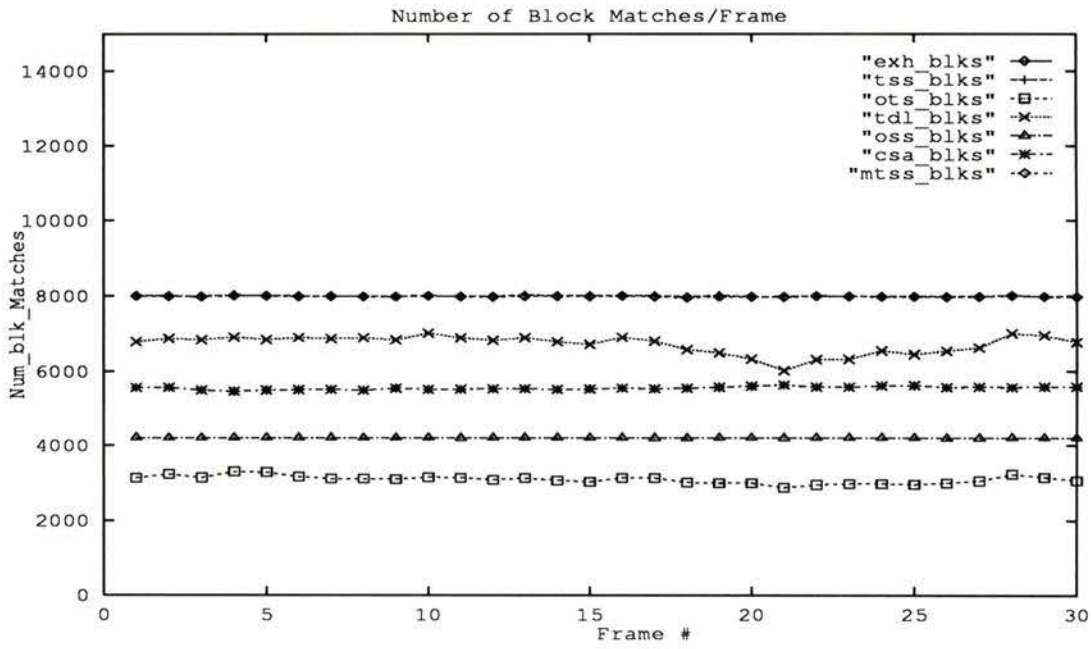


Figure 3.11 TEST 2 : Number of Block Matches/Frame for "flower bed" sequence with Reference Frames at (n-1) and Search Range=8.

3.6.3 General Discussion

An important observation related to results from TEST 1, see Figs.(3.6) and (3.9), is that the estimation accuracy of respective BMA accuracy does not necessarily increase for larger search ranges. For the "tennis" sequence, which mainly contains zooming motion, the estimation accuracy of the MTSS, CSA, and OSS algorithms actually decreases for larger search ranges. For the "flower bed" sequence, which mainly contains horizontal panning, the estimation accuracy of these BMAs is optimal for the search range of 8 pels and any increase of decrease in this range results in lower estimation accuracy. A common aspect of these algorithms, MTSS, CSA and OSS, is that the initial step size d (see Sec.(3.2)) is computed based on the search range. For larger search ranges, this step size (d) gets large and consequently results in poor accuracy.

From simulation results of TEST 2 measuring estimation accuracy, see Figs.(3.7) and (3.10), it is evident that the estimation accuracy of BMAs differs, relative to each other, over the two sequences. For the "flower bed" sequences, all BMAs with the exception of the CSA provide equivalent estimation accuracy. However, for the "tennis"

sequence, the difference in their estimation accuracy is more evident. The discrepancy in the estimation accuracy of the CSA over the "flower bed" sequence, with respect to other BMA, can be attributed to the fact that the CSA search progresses using diagonally displaced search points and subsequently cannot estimate horizontal displacement well. It is more suited to estimate diagonal motion or zooming scenes than horizontal or vertical motion.

From simulation results of TEST 2 measuring computational complexity, see Figs.(3.8) and (3.11), the computational efficiency of all BMAs over the exhaustive search is clear with the OTS algorithm being the most efficient. Also, from Fig.(3.8), it is evident the computational complexity of both the TDL and OTS algorithms is related directly to the amount of motion present in the sequence. In the "tennis" sequence, as the camera quickly zooms out at around frame 20, there is a jump in the number of block matches performed by these two algorithms while the other algorithms maintain a constant computational load throughout the sequence.

3.7 Results for Prediction Setting of $\Delta n=3$

This sections presents the results for the prediction setting of $\Delta n=3$, i.e. frames at (n) are predicted from frames at (n-3). Once again, the results for the "tennis" sequence are presented first, followed by results for the "flower bed" sequence. Also, for the purpose of comparing computational gains of respective BMAs, it should be noted that an exhaustive search for the corresponding search range, w=16 as presented in Table(3.2), requires 337920 block matches/frame.

3.7.1 Results for "Tennis" Sequence ($\Delta n=3$)

The results for the two simulation tests: TEST 1 and TEST 2 for the "tennis" sequence are respectively presented in Figs.(3.12)-(3.14). Again, estimation accuracy results for the first 20 frames for this sequences are omitted for the same reason as in Sec.(3.6.1).

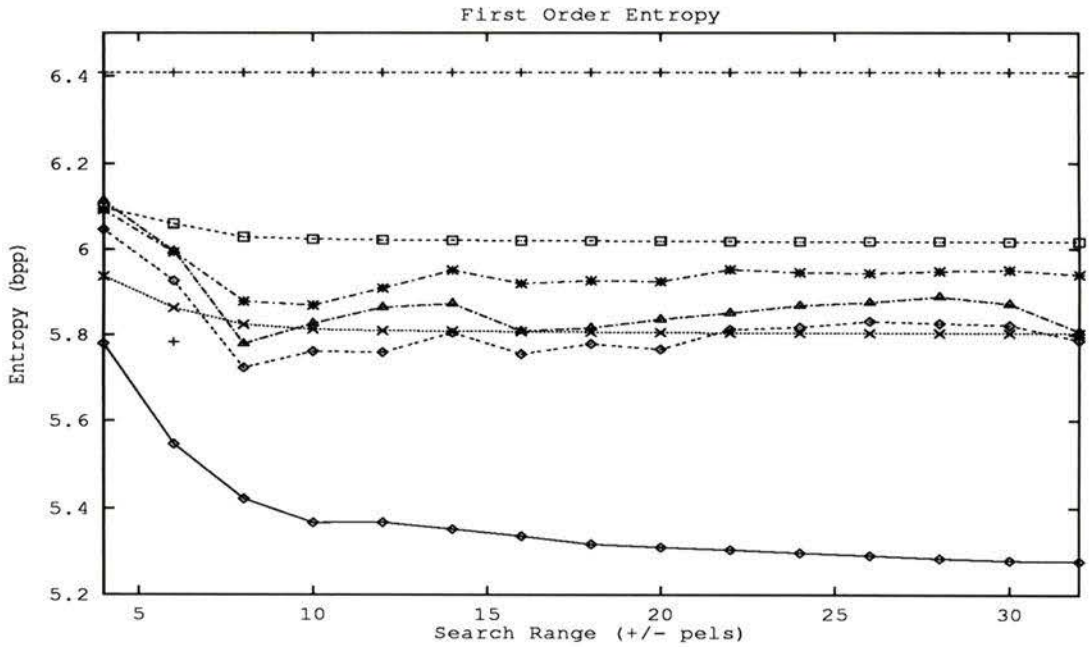


Figure 3.12 TEST 1 : Estimation accuracy v/s Search Range for "tennis" sequence with frame 33 predicted from frame 30.

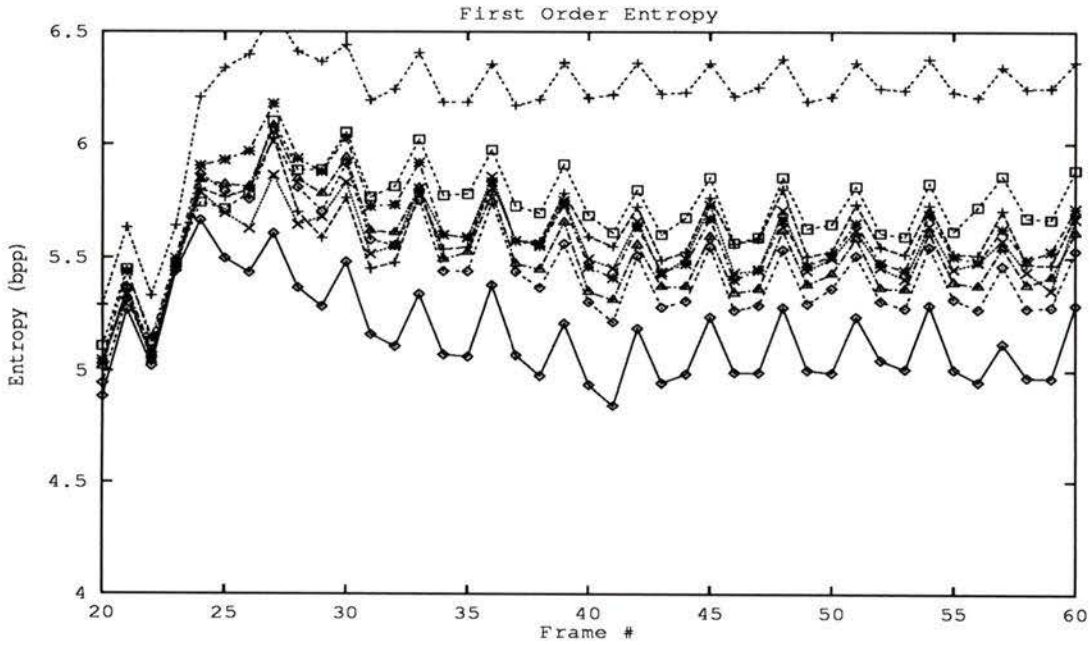


Figure 3.13 TEST 2 : Estimation accuracy for "tennis" sequence over frames 20-60 with Reference Frames at (n-3) and Search Range=16 pels.

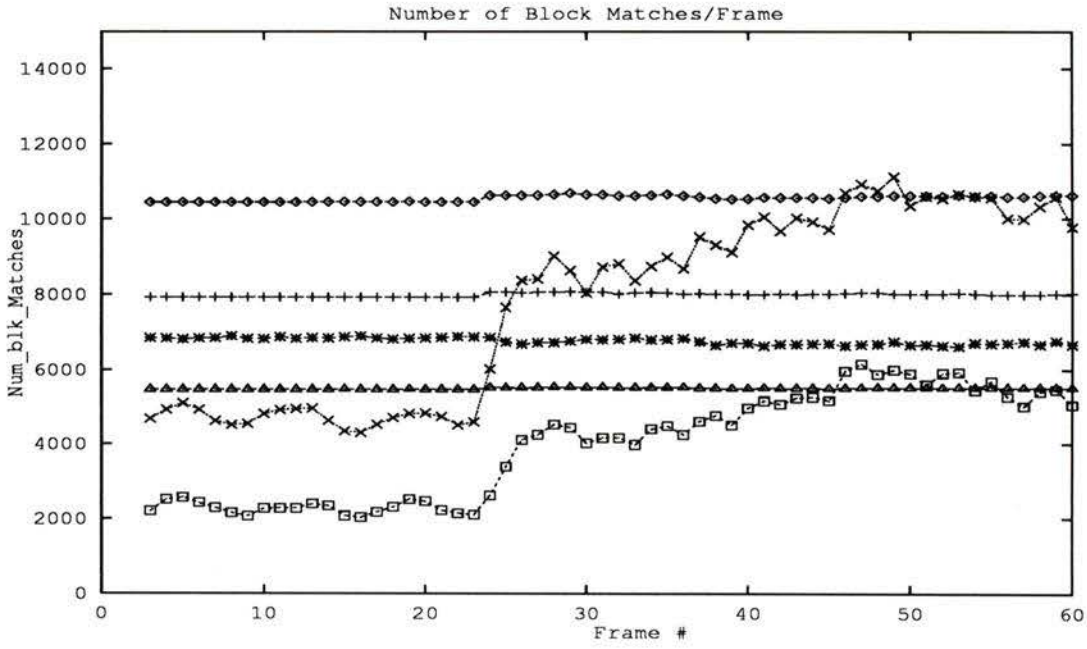


Figure 3.14 TEST 2 : Number of Block Matches/Frame for "tennis" sequence with Reference Frames at (n-3) and Search Range=16.

3.7.2 Results for "Flower Bed" Sequence ($\Delta n=3$)

The results for the two simulation tests: TEST 1 and TEST 2 for the "flower bed" sequence are respectively presented in Figs.(3.15)-(3.17).

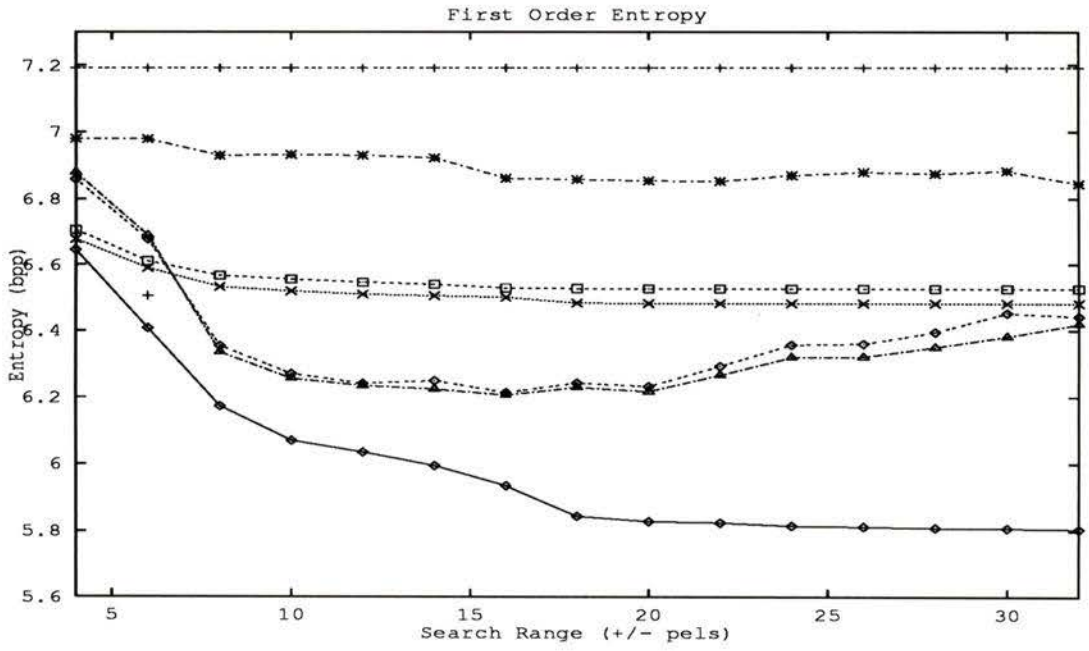


Figure 3.15 TEST 1 : Estimation accuracy v/s Search Range for "flower bed" sequence with frame 4 predicted from frame 1.

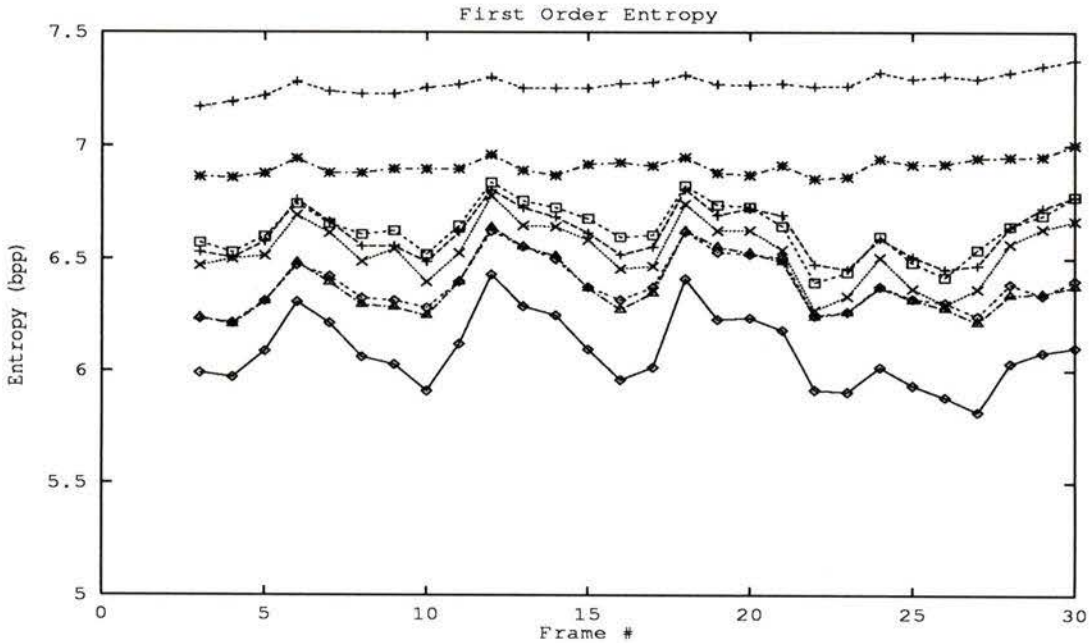


Figure 3.16 TEST 2 : Estimation accuracy for "flower bed" sequence over frames 1-30 with Reference Frames at (n-3) and Search Range=16 pels.

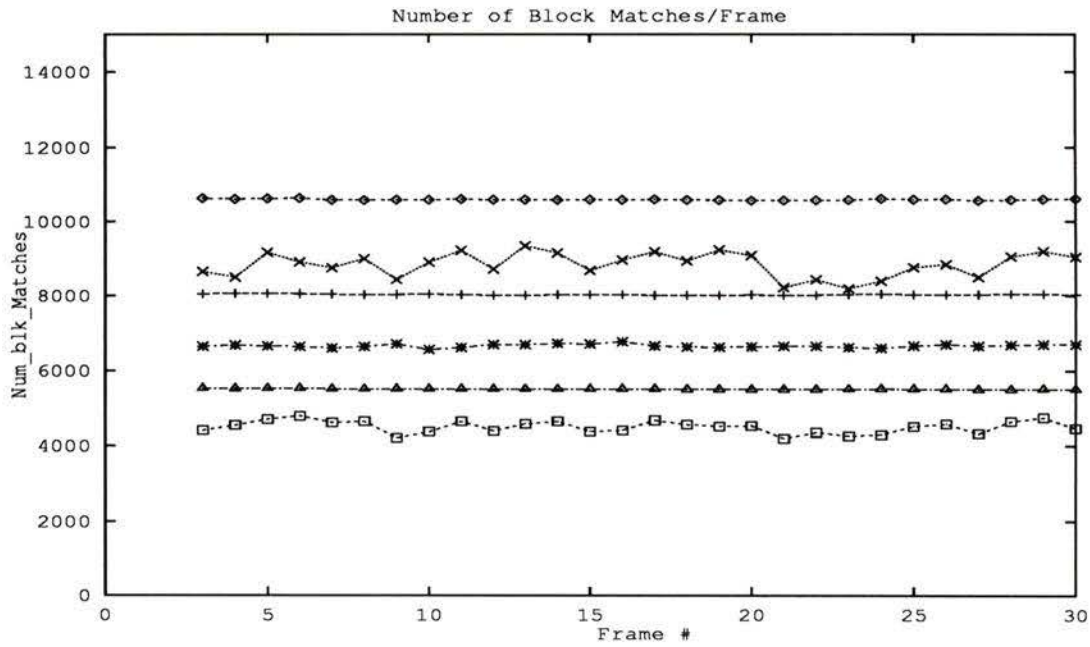


Figure 3.17 TEST 2 : Number of Block Matches/Frame for "flower bed" sequence with Reference Frames at (n-3) and Search Range=16.

3.7.3 General Discussion

From these results for prediction setting of $\Delta n=3$, similar observations are made as those for prediction setting of $\Delta n=1$. Again, estimation accuracy of BMAs does not increase with larger search ranges and their performance, with respect to each other, is related to the type of motion present in the test sequences. The computational savings of BMA over corresponding exhaustive search is even more significant in this case.

However, for this prediction setting, $\Delta n=3$, the estimation accuracy of all BMAs over both test sequences is significantly lower than that of corresponding exhaustive searches. This is evident from Figs.(3.13)and (3.16).

3.8 Conclusions

Although none of the BMAs match the estimation accuracy of corresponding exhaustive searches, the computational savings, as measured by Number of Block Matches/frame, is significant. For the search range of 8 pels, the computational saving are almost 10-fold while for the larger search range of 16 pels, the savings are over 30-

fold for the worst BMA. Such computational savings make ME using BMAs attractive for DSP implementations. Other more specific observations are presented here.

An important observation made from the simulation results is that the estimation accuracy of BMAs, with respect to corresponding exhaustive searches, does not improve with increased search ranges. For BMAs that initialize their initial step size according to the search range, such as OSS, CSA and MTSS algorithms, their estimation accuracy is optimum at a certain search range and any increase or decrease in this range results in degradation in estimation accuracy. For BMAs that always use a fixed initial step size, such as OTS and TDL algorithms, their estimation accuracy levels off after a certain search range and any further increase in the search range does not improve their accuracy. For larger search ranges, the estimation accuracy of all BMAs becomes significantly poorer as compared to that of corresponding exhaustive searches

Also, it is evident from the simulation results that the estimation accuracy of respective BMAs is related to both the distance of the reference frame from the frame being predicted and the type of motion present in the video sequence. The distance of the reference frame directly determines the suitable search range for ME, which for distant reference frames is larger to ensure accurate motion estimation. This has a direct influence in the performance of respective BMAs. For instance, the One Time Search algorithm, which has the lowest computational complexity, performs well for ME involving small motion displacements (in the case of immediate reference frames), but is not suited for ME involving larger motion displacements (in the case of distant reference frames). Another factor determining the performance of respective BMAs is the type of motion present in the sequence. This is clear from the estimation accuracy recorded for CSA. It results in exceptionally poor estimation accuracy, as compared to other BMAs, over a sequence which contains horizontal panning. For a similar prediction setting over a video sequence containing zooming, the estimation accuracy of CSA is comparable to other BMAs, thus, indicating that CSA is unsuited for ME involving panning motion.

In general, for frame prediction involving distant reference frames, the estimation accuracy of all BMAs is more discrepant from corresponding exhaustive searches. This is attributed to the poor accuracy of BMAs for ME involving large search ranges. It can therefore be concluded that BMAs are not as well suited for ME involving large motion displacements, as is the case in the use of distant reference frames, as they are for ME involving smaller motion displacements in the case for using immediate reference frames.

Chapter 4

Efficient Motion Estimation using Predictive Motion Searches

4.1 Introduction

The evaluation of BMAs presented in Chapter 3 clearly indicates that BMAs are only effective for ME involving smaller motion displacements. They do not provide the estimation accuracy of corresponding exhaustive searches and, moreover, their estimation accuracy does not improve with larger search ranges. These sub-optimal results are a direct result of the assumption, applied in all BMAs, that the Mean Absolute Difference between blocks increases monotonically as the search moves away from the match. This assumption, although valid around the vicinity of the match, does not hold true for larger search ranges. The computational savings achieved with BMAs, however, still make them attractive for efficient ME implementation.

A predictive motion search technique, the Adaptive Search Window Method (ASWM), that can be incorporated with BMAs to provide computationally efficient and accurate ME involving larger motion displacements is presented in this chapter. This technique dynamically controls the location and corresponding search range for ME for individual blocks in the frame. For each block, a prediction of the general location of the match is made based on motion information of surrounding blocks, and if a good prediction is possible, ME is performed around the predicted block using a smaller search range. ME can be performed using any BMA or even an exhaustive search. The reduction in the search range ensures accurate ME using BMAs and less computation in the case of exhaustive searches. ASWM also provides integration with video coding standards while keeping the overhead computation and memory requirements minimal. Previ-

ous work done on predictive motion estimation is presented in [30]-[31]. These proposed predictive techniques, however, are not adaptive and thus lead to satisfactory results only for scenes involving large homogeneous motion such as camera panning as reported in [30]. The ASWM works for both zooming and panning motion scenes.

In this chapter, the detailed implementation of ASWM is presented and corresponding simulation results for ASWM incorporating respective BMAs are reported. The simulation tests conducted are exactly the same as those presented in Chapter 3 and only simulation results for the prediction setting of $\Delta n=3$, which expounded the poor BMA performance for ME involving large search ranges, are reported. A substantial enhancement in estimation accuracy is achieved with the ASWM using BMAs over tradition application of respective BMAs as presented in Chapter 3 for ME. In addition, two robust BMAs incorporating ASWM are implemented for ME in a MPEG-I coder and corresponding coding gains over independent BMAs reported. It is evident from these results that substantial improvement in terms of estimation accuracy is achieved for all BMA after incorporating this predictive search technique.

4.2 Predictive Motion Search

As mentioned earlier, the assumption that the matching error increases monotonically as the search algorithms moves away for the match holds true only within close vicinity of the match. This is clearly evident in Fig.(4.1) which displays the inverse of MAD error over a search range of 16 pels centered around a block. The location of the best match coincides with the sharp peak.

To locate the match in this case, it is essential to initiate the search algorithm within close vicinity of the peak since the error surface is highly non-linear with many local minimas (small peaks) on which BMAs would likely converge.

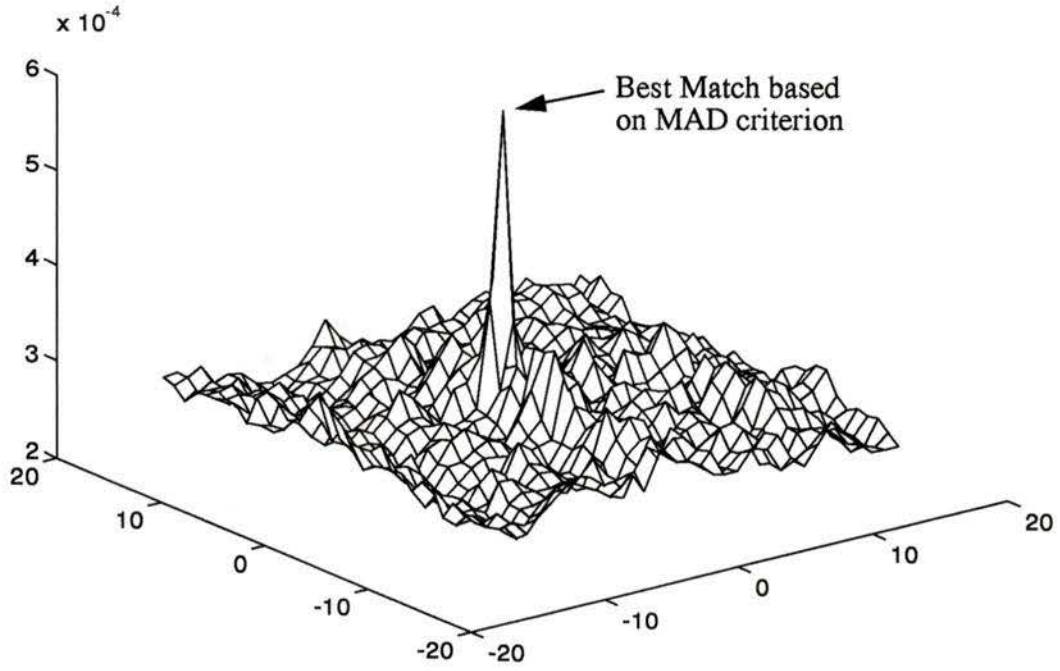


Figure 4.1 Measurement of inverse MAD over a search range of 16 pels

Thus, a scheme that can use BMAs for ME over a smaller search range, in close vicinity of corresponding matches, can be used effectively to provide computationally efficient and accurate ME involving large motion displacements. To accomplish such a task, however, the scheme has to provide good predictions of the possible matches for corresponding macroblocks beforehand. It is generally assumed that motion information of neighboring blocks is correlated and can be used to provide suitable predictions. Preliminary ideas for such predictions have been presented in the literature [30]-[31] and are outlined in the following subsection. The limitations of the techniques presented in [30]-[31] will be discussed and a new technique will be proposed to overcome these limitations.

4.2.1 Previous Work

The two approaches that have been investigated for predicting appropriate search area for ME include inter-frame and inter-block predictive methods [30]-[31]. The Inter-frame predictive method proposed in [31] uses motion vector information from previously coded frames to compute initial predictions for corresponding blocks in the current frame. Since this prediction technique requires motion vector information of previous

frames, it imposes significant overhead for storage facilities, thus, making it unsuitable for use in practical video coder implementations. The Inter-block predictive method proposed in [30], on the other hand, uses motion vector information of neighboring blocks as predictions. Since both the predictive technique proposed in this chapter and the inter-block predictive method proposed in [30] exploit correlation present in motion information of neighboring blocks, it is appropriate to present the inter-block prediction approach of [30] in some detail for comparison.

The method proposed in [30] first groups neighboring macroblocks into superblocks, e.g. 4 macroblocks can be grouped to form a superblock, and then performs accurate ME on the first macroblock, in every superblock, using an exhaustive search and a large search range. The computed motion vectors are then used as predictions for the rest of the macroblocks in corresponding superblocks. For each macroblock with a prediction, ME is performed around the predicted macroblocks using a reduced search area to compute the corresponding final motion vector. Predicted macroblocks are macroblocks pointed to by the predicted motion vector, which in this case is same as the motion vector of the first macroblock of the corresponding superblock. The authors acknowledge the limitations of this approach and report that the technique works well only for scenes containing large areas moving in the same direction such as panning scenes. It is, thus, not suitable for use with sequences containing zooming motion.

4.2.2 Search Area Prediction

The difficulty in designing predictive inter-block motion estimation lies in the fact that high correlation between motion information cannot always be guaranteed, especially for zooming scenes. This is due to the fact that motion estimation using exhaustive searches or BMAs does not always yield motion vectors representative of the true motion present in the scene [32]. This is exemplified in Fig.(4.2) which shows the motion field generated using exhaustive search for a zooming scene. The motion vectors generated do not conform well with the motion in the scene and the resulting motion field does not possess a high degree of correlation, thus making the simple prediction approach applied in [30] ineffective. However, there is still enough local correlation between neighboring motion vectors that can be exploited.

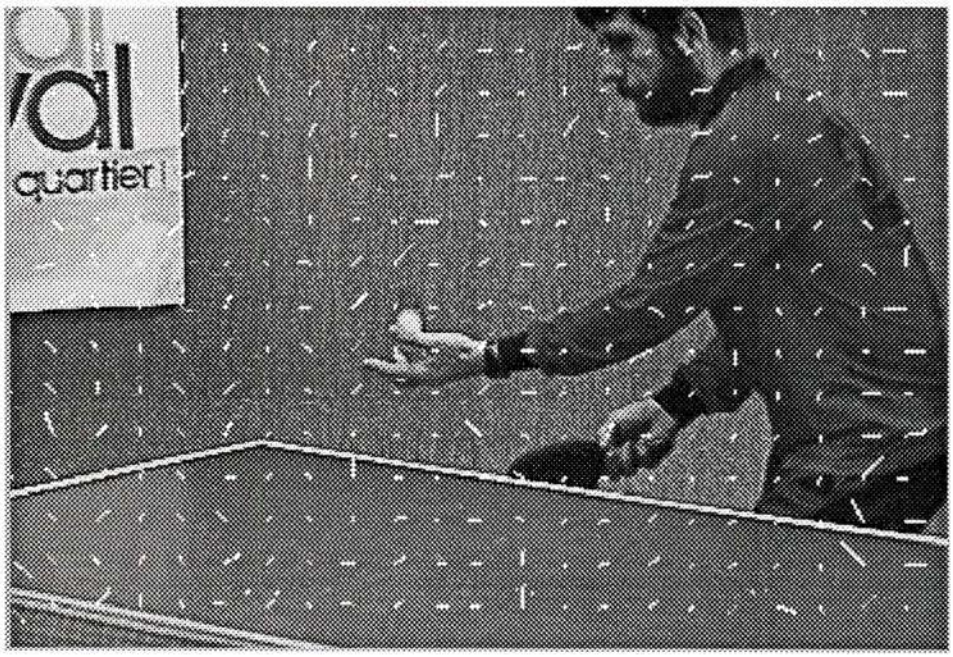


Figure 4.2 Motion vectors generated using exhaustive search to predict frame#56 from frame #54

Since the predominant type of motion present in most video sequences is a combination of panning and zooming, it is imperative that any predictive scheme designed for use with block matching algorithms be robust enough to yield good predictions for both panning and zooming scenes. One such method is proposed in the following section.

4.3 Adaptive Search Window Method

The Adaptive Search Window Method (ASWM) incorporates an inter-block predictive scheme that is computationally efficient and robust. It exploits the local correlation that exists between neighboring motion vectors to obtain motion vector prediction for current blocks. An important observation made from analyzing motion fields generated for zooming scenes, such as Fig. (4.2), is that although there is no general correlation over the entire motion field, some correlation exists between neighboring motion vectors. It can be seen from Fig. (4.2) that each motion vector is similar to at least one of its surrounding motion vectors. For panning scenes, there is even greater correlation between these

motion vectors. Based on this observation, the Adaptive Search Window Method is proposed. Other important issues considered in designing this scheme include: amount of overhead computation, memory requirements for storing previous motion information, suitability for use in emerging video coding standards and, to a certain degree, suitability for hardware implementation.

4.3.1 Implementation

In essence, the ASWM method uses motion information of three surrounding macroblocks blocks to obtain motion vector predictions. If a suitable prediction is found, then ME is preformed around the predicted block using a reduced search range (w_p) to locate the final match. If a suitable prediction is not found, which may occur if a corresponding block contains two objects moving in opposite directions or new information is being introduced into the scene as a result of covered areas being uncovered, then ME is performed around the original block using a larger search range (w_o).

The exact implementation of the ASWM scheme introduced here is outlined by the following steps:

For each block in the frame being processed

1. Compute MAD between the current block and the block at the same location in the reference frame, i.e. zero motion vector, and record it as MAD_0.
2. Use the three motion vectors corresponding to blocks located at; the top (MV_1), the top-left (MV_2) and the left (MV_3) and compute respectively MAD_1, MAD_2, and MAD_3 (see Fig. (4.3b))
3. Compare all MAD values, (MAD_0, MAD_1, MAD_2, and MAD_3), and select as predicted the motion vector corresponding to the lowest MAD.
4. If the final prediction is not the zero motion vector, then perform ME around predicted block using a reduced search range w_p (Fig(4.3b)), otherwise perform ME over the original block using a larger search range w_0 (Fig.(4.3a))

This simple prediction procedure is designed to work efficiently as blocks are processed in a raster scan either row by row or column by column in the frame. Step (4) ensures that proper ME is performed for blocks whose motion cannot be predicted from surrounding information by reverting back to a large search range.

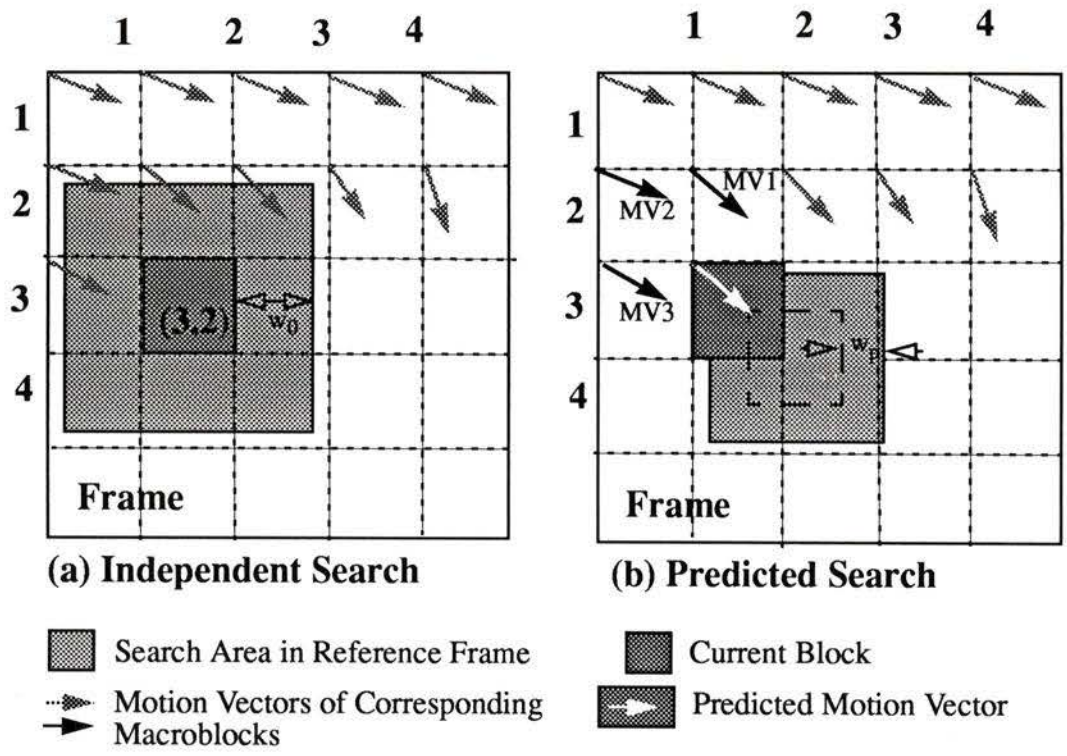


Figure 4.3 Reduction of ME search area for block (3,2) using motion vector information of blocks located at (3,1), (2,1) and (2,2)

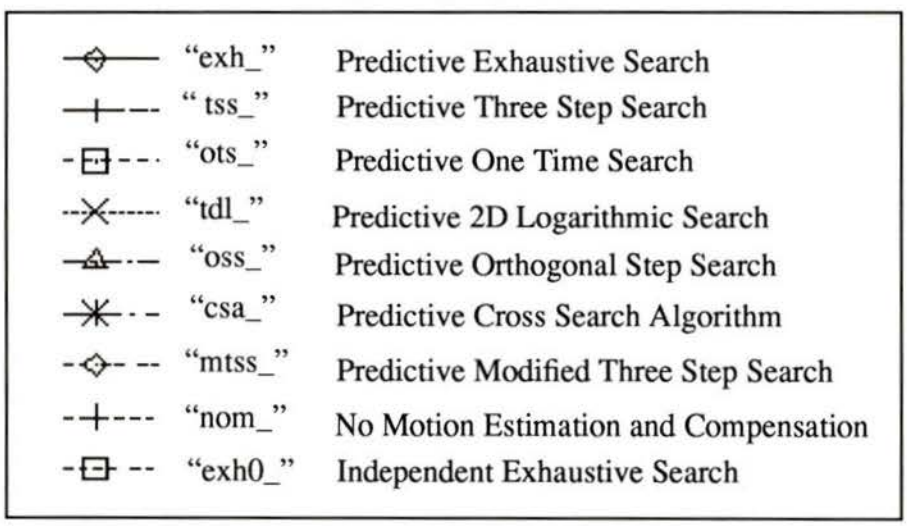
The robustness of this technique lies in fact that, unlike in [30], a new prediction is made for every block in the frame and, also, a larger subset of motion vectors is used in computing this prediction. The use of extra vectors for prediction not only eliminate propagation of prediction errors, but also compensates for blocks that may have been coded as Intra-blocks without any associated motion information. In addition, the computational overhead and storage requirements are minimal. The ASWM requires an overhead of four extra MAD computations per block and storage of motion information for the previous row or column of blocks depending on the direction of the raster scan. Also, the regularity of the algorithm makes it attractive for hardware implementation.

4.4 Simulation Results

The simulation tests presented in Chapter 3, TEST 1 and TEST 2, were repeated for all BMA incorporating ASWM, and results for the prediction setting of $\Delta n=3$, which

expounded the poor BMA performance for ME involving larger motion displacements, are reported here. All the related parameters for the respective simulation tests are identical to those presented in Sec.(3.5) of Chapter 3. In the implementation of ASWM algorithm implementation, the search range for predicted blocks, in Step (4), is reduced to half of that used for un-predicted blocks. For instance, in TEST 2 with $\Delta n=3$, the search range for predicted blocks was reduced to 8 pels from 16 pels for valid predictions. The three simulation tests were conducted over the two video sequences, "tennis" and "flower bed", and the corresponding results are summarized in the following subsections.

Once again, the results for the "tennis" sequence are presented first followed by results for the "flower bed" sequence. The computational load for the ASWM using exhaustive searches is also reported. The following legend is used in the figures presenting the results of the simulations.



The Independent Exhaustive Search, "exh0_", refers to the exhaustive search results that were used as benchmarks in Chapter 3.

4.4.1 "Tennis" Sequence

The results for the two simulation tests: TEST 1 and TEST2 for the "tennis" sequence are respectively presented in Figs.(4.4)-(4.7). Again, estimation accuracy results for the first 20 frames for this sequences are omitted (see Sec.(3.6.1)).

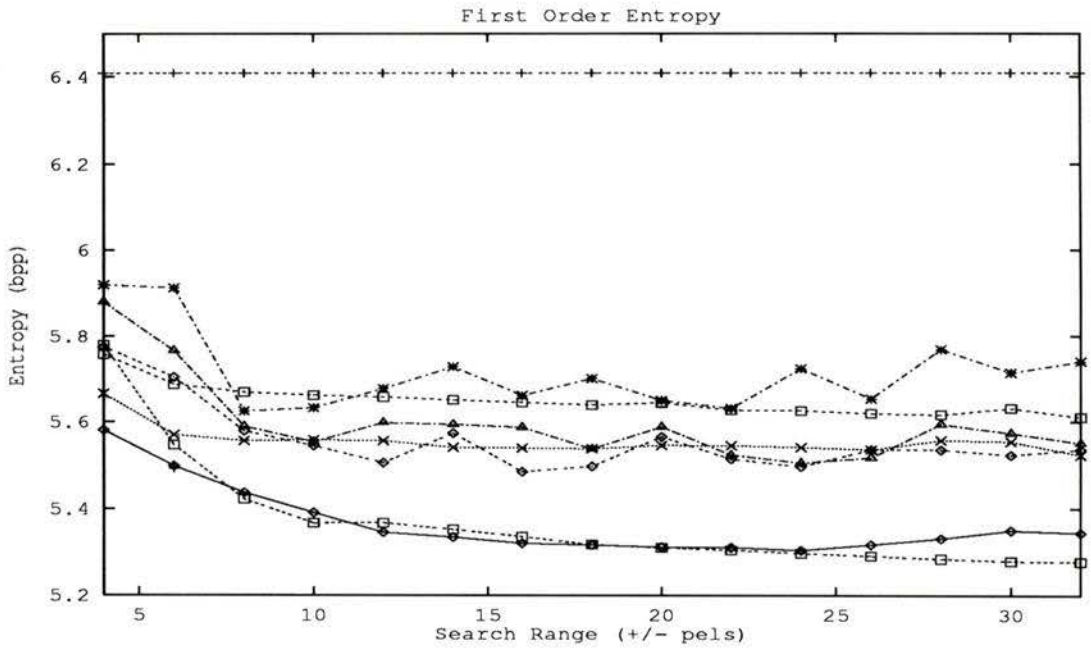


Figure 4.4 TEST 1 : Estimation accuracy v/s Search Range using ASWM for "tennis" sequence with frame 33 predicted from frame 30.

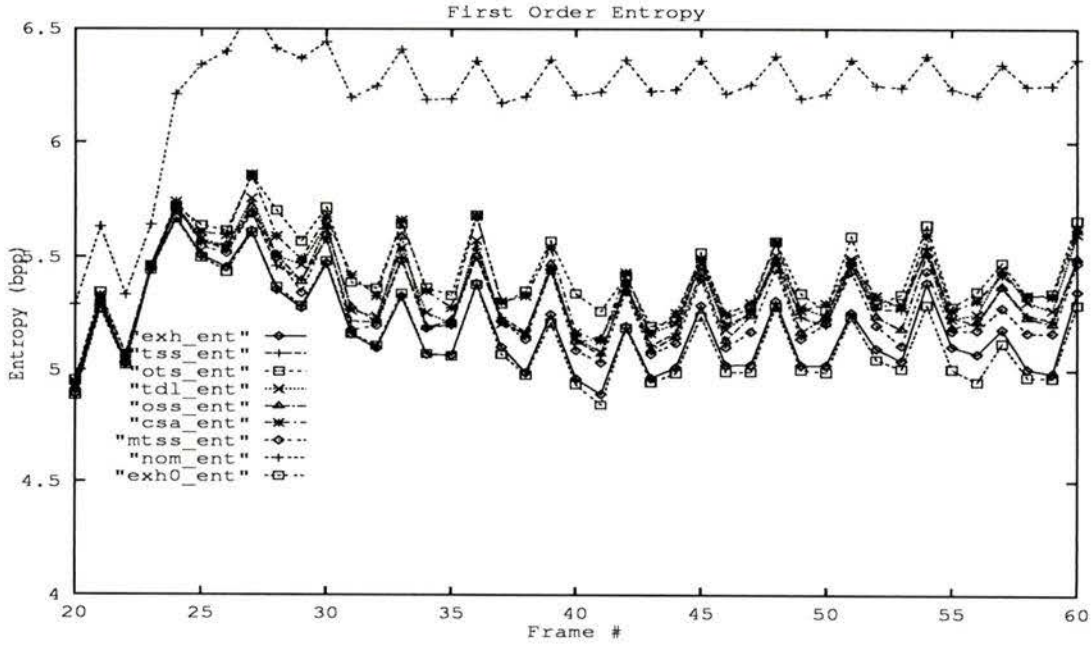


Figure 4.5 TEST 2 : Estimation accuracy using ASWM for "tennis" sequence over frames 20-60 with Reference Frames at (n-3), Initial Search Range=16 pels and Reduced Search Range = 8 pels.

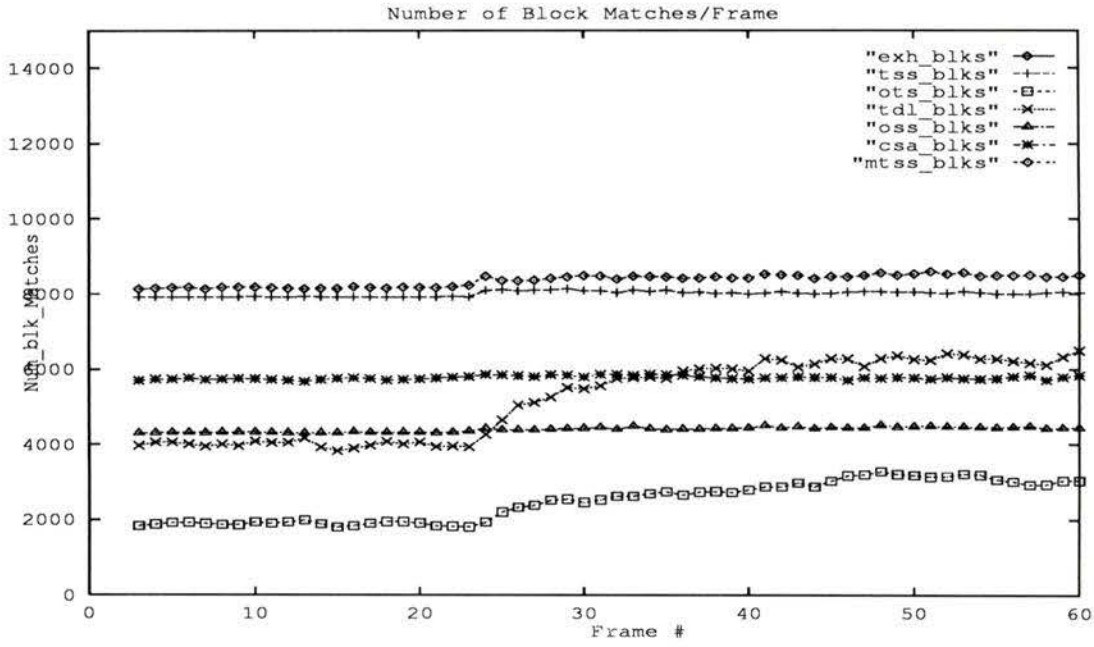


Figure 4.6 TEST 2 : Number of Block Matches/Frame for "tennis" sequence with Reference Frames at (n-3), Initial Search Range=16 pels and Reduced Search Range = 8 pels.

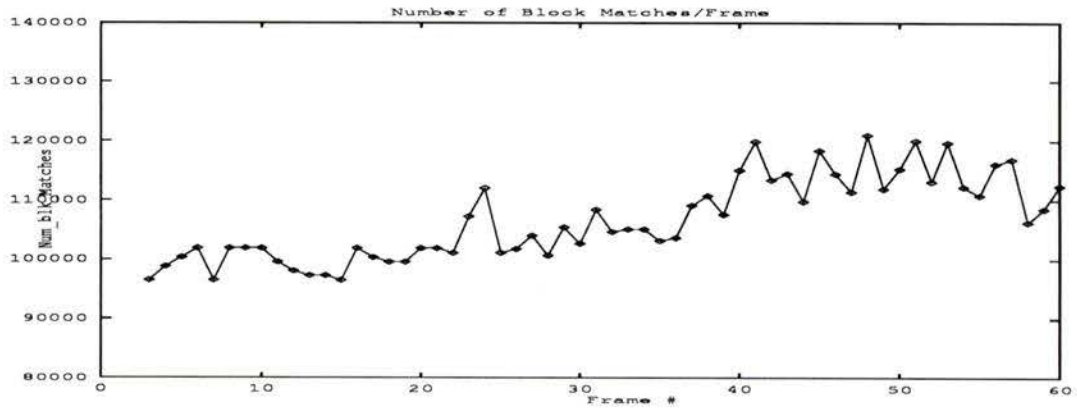


Figure 4.7 Number of Block Matches/Frame for exhaustive search using ASWM for "tennis" sequence with Reference frame at (n-3), Initial Search Range=16 pels and Reduced Search Range = 8 pels.

4.4.2 "Flower Bed" Sequence

The results for the two simulation tests: TEST 1 and TEST2 for the "flower bed" sequence are respectively presented in Figs.(4.8)-(4.11).

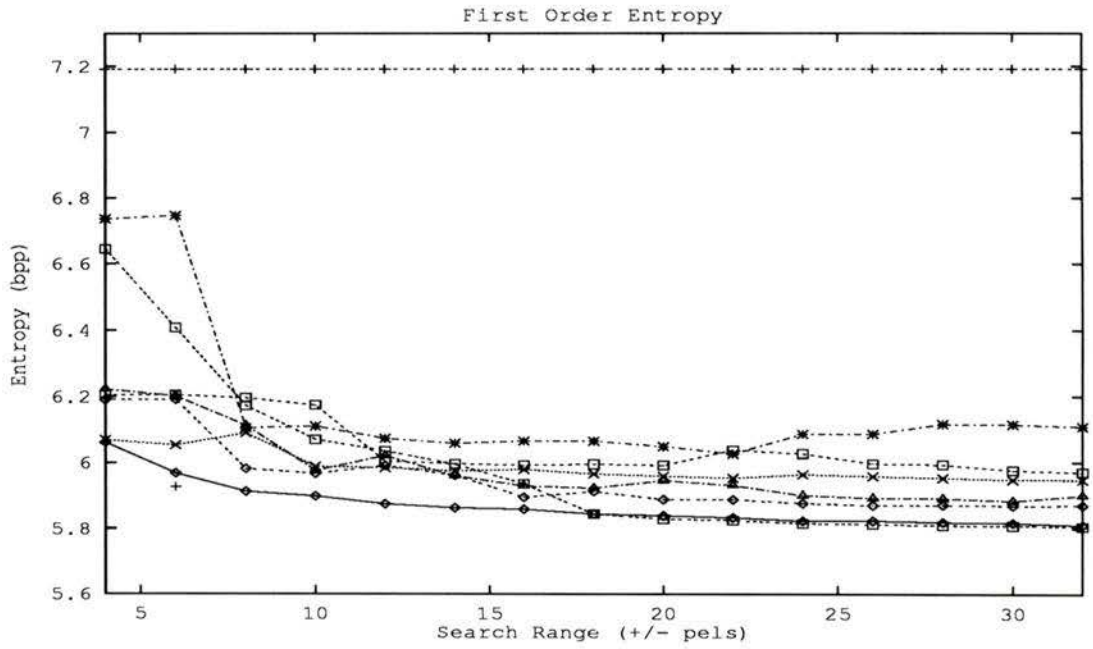


Figure 4.8 TEST 1 : Estimation accuracy v/s Search Range using ASWM for "flower bed" sequence with frame 4 predicted from frame 1.

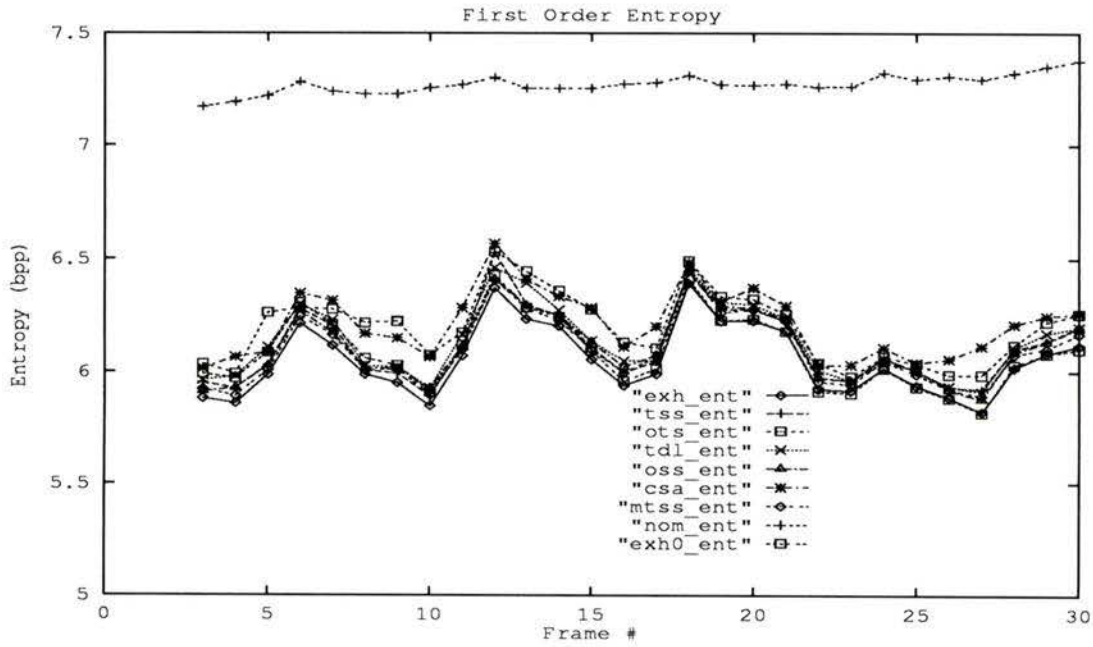


Figure 4.9 TEST 2 : Estimation accuracy using ASWM for "flower bed" sequence over frames 20-60 with Reference Frames at (n-3), Initial Search Range=16 pels and Reduced Search Range = 8 pels.

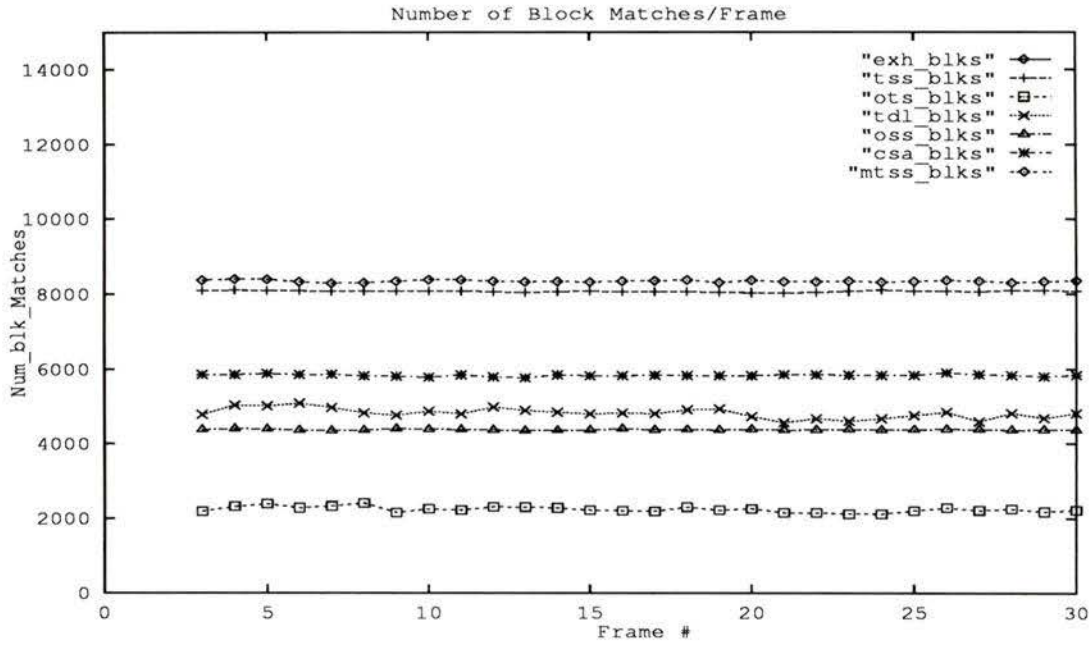


Figure 4.10 TEST 2 : Number of Block Matches/Frame for "flower bed" sequence with Reference Frames at (n-3), Initial Search Range=16 pels and Reduced Search Range = 8 pels.

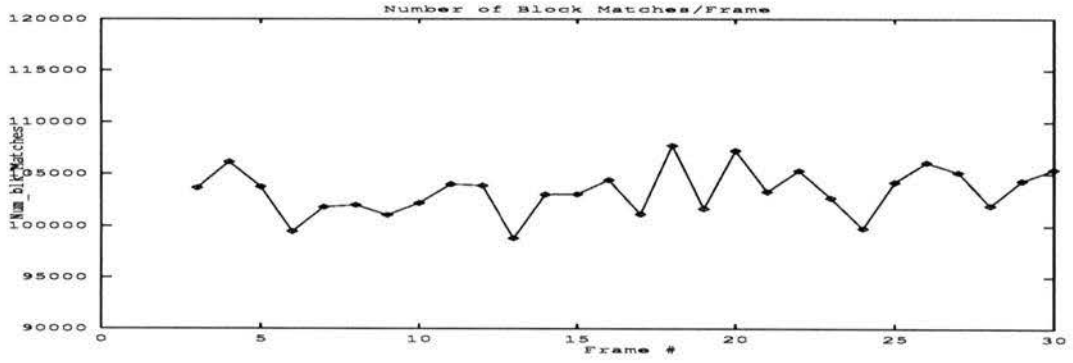


Figure 4.11 Number of Block Matches/Frame for exhaustive search using ASWM for "flower bed" sequence with Reference frame at (n-3), Initial Search Range=16 pels and Reduced Search Range = 8 pels

4.4.3 General Discussion

Comparing these results with corresponding results presented in Chapter 3, It is clear that for all BMAs, an improvement in estimation accuracy is achieved with predictive searches. From results of TEST 1, which measures estimation accuracy v/s search

range, it is observed from comparing Fig.(4.4) with Fig.(3.12) and Fig.(4.8) with Fig.(3.15), that BMAs incorporating ASWM are greatly effective over large search ranges. From results of TEST 2 measuring the estimation accuracy, it is evident from comparing Fig.(4.5) with Fig.(3.13) and Fig.(4.9) with Fig.(3.16) that the incorporation of ASWM with respective BMAs substantially improves their estimation accuracy.

Finally, by comparing results of TEST 2 measuring the respective computational complexity, Fig.(4.6) with Fig.(3.14) and Fig.(4.10) with Fig.(3.17), a general reduction in computational effort is observed for all BMAs. For BMAs that use a fixed number of iterations to locate a match (determined from the corresponding search range), such as MTSS, CSA and OSA algorithms, the reduction in their computation load is a result of the adaptive reduction of corresponding search ranges for ME using ASWM. It should be noted, however, that for BMAs such as TDL and OTS algorithms, a reduction in the search range does not necessarily reduce the computational effort required in locating a match.

4.5 Simulation Results with MPEG-I Video Coder

Further analysis of ME utilizing the ASWM was performed by incorporating it in a MPEG-I coder. A software implementation of MPEG-I coding standard[1] was modified to incorporate two BMA algorithms, MTSS and TDL, for Motion Estimation (see Chapter 2). These two BMAs are chosen because of their superior performance over other BMAs in the simulation tests of Chapter 3. Both independent versions of BMAs and those incorporating ASWM were implemented. Also, a predictive exhaustive search and a hybrid ME scheme were implemented. The hybrid ME method utilized predictive exhaustive search for forward prediction, i.e. P-frames, and predictive MTSS search for bidirectional prediction, i.e. B-frames.

Ten frame from the "tennis" sequence, frames 30-39, were coded and corresponding computational measures and compression ratios reported for respective ME algorithms. In addition, the coded bitstreams generated were played on a MPEG-I decoder for visual analysis. The frames used are essentially composed of zooming out motion. The MPEG

1. The software MPEG coder implementation used in these simulations was the Berkeley MPEG encoder, mpeg_encode v1.6 (c) 1993, developed at University of California.

coding parameters used for generating the MPEG-I bitstreams are listed below:

Frames: [30-39]
GOP Sequence: IBBPBBPBBP
Search Range: 16 pels
ME Accuracy: Full Pixel

Table (4.1) summarizes corresponding results.

ME Algorithms	Compression Ratio	Total No. of bits in coded bitstream	Total No. of Block Matches performed
Exhaustive	87:1	28923	4,230,412
Exhaustive with ASWM	86:1	29223	1,704,728
MTSS	64:1	39401	162,101
MTSS with ASWM	77:1	32500	141,147
TDL	69:1	36679	161,461
TDL with ASWM	77:1	32756	173,821
Hybrid (pr_exh + pr_mtss)	84:1	29848	450,960

Table 4.1. Summary of MPEG-I coding results for respective Motion Estimation Algorithms

From visual analysis of decoded video, it was confirmed that for all ME algorithms tested, none of the decoded frames contained any visual motion artifacts that would have otherwise resulted in case of sub-optimal Motion Estimation. The visual quality of the decoded frames was equivalent for all ME algorithms.

From the results presented in Table (4.1), the improvement in compression ratios for the two BMAs after incorporating ASWM is clear. An almost 20% improvement is achieved for the MTSS BMA while, at the same time, reducing the amount of corresponding computation. Also, the predictive exhaustive search provides an equivalent compression ratio as the independent exhaustive search but at a fraction of the computational effort. The hybrid ME scheme, on the other hand, provides a good compromise between compression and computation. Further computational saving are possible using efficient block matching techniques presented in [33].

4.6 Discussion

The Adaptive Search Window Method when incorporated with BMAs provides computationally efficient and accurate ME. It is particularly attractive for ME involving

large motion displacements in the case of frame predictions involving distant reference frames. Also, simulation results indicate that ASWM using exhaustive search yields results almost equivalent to independent exhaustive search which requires greater computation as a result of the larger fixed search range used for ME. Since in its implementation, ASWM uses surrounding motion vectors, it is necessary to store the previous row or column of motion vectors depending on how the blocks are being processed, and an addition of four MAD computations per block are required to compute appropriate predictions. This additional computation is eclipsed by corresponding reduction in computation achieved by the adaptive reduction of search ranges using ASWM. It is also anticipated that the use ASWM for ME yields smoother motion fields since surrounding motion vectors information is directly applied in locating block matches. Motion field smoothness increases coding efficiency of coders such as MPEG that differentially code motion vectors in corresponding motion fields. The ASWM implementation outlined in this chapter is simple and compatible for use in existing and proposed video coding standards.

Chapter 5

Conclusions

In this thesis, the performance of a selected number of Block Matching Algorithms (BMAs) was evaluated for Motion Estimation, and, in addition, a predictive motion search technique was proposed to enhance their performance. The relevant important results obtained from this evaluation, and the application of predictive search for ME are respectively summarized in the following two sections.

5.1 BMA Evaluation

Although none of the BMA algorithms matched the estimation accuracy of corresponding exhaustive searches, the computational savings over exhaustive searches are significant. This computational efficiency of BMAs makes their use feasible for real-time ME implementation on fast DSP chips that are now becoming available commercially. BMAs, however, provide good alternatives to exhaustive searches only for ME involving small motion displacements. This is the case when immediate reference frames are used for frame predictions. They are not suited for ME involving large motion displacements as required when using distant reference frames for frame prediction. Another evident characteristic of BMAs is that their performance, both in terms of estimation accuracy and computational complexity, is linked directly with the type of motion present in the video sequences. i.e. some BMAs are better suited for ME involving zooming scenes while other perform better over panning scenes. In conclusion, BMAs provide computationally efficient alternatives to exhaustive searches only for ME involving small motion displacements.

5.2 The Adaptive Search Window Method

The Adaptive Search Window Method (ASWM) proposed in Chapter 4, when incorporated with Block Matching Algorithm (BMAs), provide computationally efficient and accurate ME even for ME involving larger motion displacements. A significant enhancement, in terms of the estimation accuracy, is achieved for ME involving large motion displacements over the independent application of BMAs while maintaining the computational efficiency of respective BMAs. In addition, the ASWM using exhaustive searches provides equivalent compression as independent exhaustive searches at a fraction of the computation effort. The computational overhead and memory requirements in incorporating the ASWM are minimal, with every block requiring a maximum of four additional Mean Absolute Difference (MAD) computations. For some BMAs, this additional computation is eclipsed by the overall computational gains achieved by the dynamic reduction of search range by ASWM for ME. When incorporated in the MPEG-I coder, a substantial gain in compression ratio is achieved for the ASWM using BMAs over independent application of respective BMAs. In conclusion, the combined application of the ASWM with BMAs, or exhaustive searches, provides an excellent avenue for computationally efficient and accurate motion estimation.

5.3 Future Work

The preliminary application of the Adaptive Search Window Method proposed in this thesis produced the two desirable qualities required for Motion Estimation: accuracy and computational efficiency. Further simulations results with the ASWM using different and longer video sequences can ultimately lead to further refinements of the method. Moreover, such simulations can also lead to better prediction techniques for ME.

Another important issue relating to the use of BMAs for ME in coders is their feasibility for real-time ME at video rates using DSP chips. With the advent of powerful commercially available DSP chips, such a study is now possible. Real-time coding of video sequences is the ultimate comparison study for evaluating various BMAs

Some other important issues related to ME/MC and the general area of motion video coding that still need to be addressed are listed here:

- Investigate better methods to estimate and compensate for zooming and rotation motion in video sequences. Current ME/MC techniques, as presented in this thesis, only compensate for translation motion and in the case of fast zooming motion, such translational approximations do not provide substantial coding gains.
- Investigate motion rendition by Human Visual System and its application in motion video coding
- Investigate video coding techniques for low-bit rate applications at around 4-64kb/s as required for video transmission over telephone lines. Radically different coding methods are required at such low bit rates as the current approaches using DCT with ME/MC are inadequate.

Bibliography

1. Shannon, C.E, "A mathematical theory of communications", Bell System Technical Journal, Vol.27, 1948. p379-423
2. Huffman D.A., "A method for the construction of minimum redundancy codes", Proc. IRE, 40, 1952. p1098-1101.
3. Langdon, G. G. Jr., "An Introduction to Arithmetic Coding", IBM Journal of Research and Development, Vol. 28, No.2, March 1984. p135-149
4. Kunt, M., Ikonomopoulis, A., and Kocher, M., "Second Generation Image-Coding Techniques", Proceedings of the IEEE, Vol.73, No.4, April 1985. p549-573
5. Nasrabadi, N.M., and King, R., "Image Coding Using Vector Quantization: A Review", IEEE Trans. Comm., Vol.36, No.8, August 1988. p957-971
6. Woods, J.W. and O'Neil, S.D., "Subband Coding of Images", IEEE Trans. Acoustics, Speech, and Signal Processing, Vol. ASSP-34, No.5, October 1986. p1278-1288
7. Gharavi, H., and Tabatabai, A., "Sub-Band Coding of Monochrome and Color Images", IEEE Trans. Circuits and Systems, Vol.35, No.2,February 1988. p207-214
8. Burt, P.J., and Adelson, E.H., "The Laplacian Pyramid as a Compact Image Code", IEEE Trans. Comm., Vol.COM-31, No.4, April 1983. p532-540
9. Antonini, M., Mathieu, P., and Daubechies, I., "Image Coding Using Wavelet Transform", IEEE Trans. Image Processing, Vol.1, No.2, April 1992. p205-220
10. Jayant N., Johnston J., and Safranek R., "Signal Compression Based on Models of Human Perception", Proc. IEEE, Vol.81, No.10, October 1993. p1385-1421
11. Wakeman, I., "Packetized Video - Options for Interaction between the User, the Network and the Codec", The Computer Journal, Vol.36, No.1, 1993. p55-67
12. Anastassiou D., "Digital Television", Proc. IEEE Vol.82, No.4, April 1994. p510-519
13. Wallace, G.K., "The JPEG Still Picture Compression Standard", Comm. of the ACM, Vol.34, No.4, April 1991. p31-44
14. Le Gall, D., "MPEG: A Video Compression Standard for Multimedia Applications", Comm. of the ACM, Vol.34, No.4, April 1991. p47-63
15. Liou M., "Overview of the px64 kbit/s Video Coding Standard", Comm. of the ACM, Vol.34, No.4. April 1991. p59-63

16. Petajan, E., "Digital Video Coding Techniques for US High-Definition TV", IEEE Micro Magazine, October 1992. p13-21

17. Ruetz P. A., Tong P., Bailey D., Luthi A., and Ang P.H., "A High-Performance Full-Motion Video Compression Chip Set", IEEE Trans. for Video Technology, Vol.2, No.2, June 1992. p111-121

18. Uz K.M., Vetterli M, and LeGall D.J., "Interpolative Multiresolution Coding of Advanced Television with Compatible Subchannels", IEEE Trans. Circuits and Systems for Video Tech., Vol 1, No.1. March 1991. p86-99

19. Zhang Y., and Zafar S., "Motion-Compensated Wavelet Transform Coding for Color Compression", IEEE Trans. Circuits & Systems for Video Technology, Vol.2, No.3, September 1992. p285-292

20. International Organization for Standardization, *Information technology-coding of moving pictures and associtated audio for digital storage media at up to about 1.5 Mbit/s, 1st ed.*, Copyright ISO/IEC 11172-2 1993.

21. Sun M.T., "MPEG-2 Systems and the transport over ATM", ISCAS '1994 Tutorials. Chapter 3.14. p138-146

22. Gutlag K.M., "Multimedia Powerhouse", Byte Magazine, June 1994. p57-64

23. Jain J. R., and Jain A. K., "Displacement measurement and its application in interframe image coding", IEEE Trans. Comm., Vol. COM-29, December 1981. p1799-1806

24. Koga T.," Motion compensated interframe coding for video conferencing", Proc. National Telecommunications Conference, New Orleans, Nov. 29-Dec. 3, 1981. G5.3.1-G5.3.5

25. Srinivasan R. and Rao K.R., "Predictive Coding Based on Efficient Motion Estimation", IEEE Trans. Comm., Vol.com-33, No.8, August 1985. p888-896

26. Ghanbari M., "The Cross-Search Algorithm for Motion Estimation", IEEE Trans. Comm., Vol.38, No.7, July 1990. p950-953

27. Puri A., Hang H.M. and Schilling D.L., "An efficient block-matching algorithm for Motion-Compensated coding", Proc. IEEE ISASSP 1987. p25.4.1-24.4.4

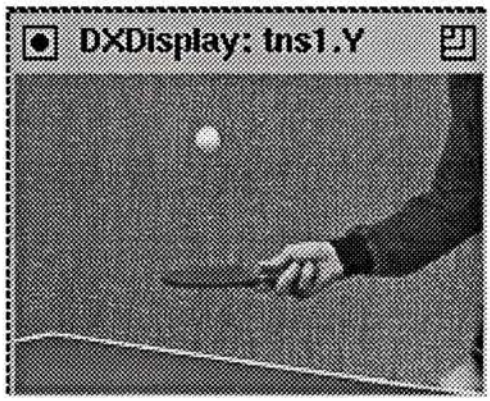
28. Musmann H.G., Pirsch P. and Grallert, H.J., "Advances in Picture Coding", Proc. IEEE, Nol.73, No.4, April 1985. p523-548

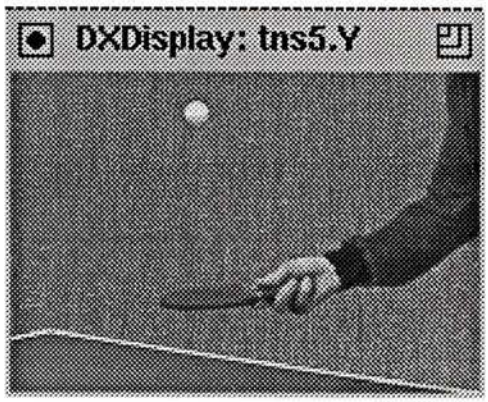
29. Kappagantula S. and Rao K.R., "Motion Compensated Interframe Image Prediction", IEEE Trans. Comm.,Vol.com-33, No.9, September 1985. p1011-1014

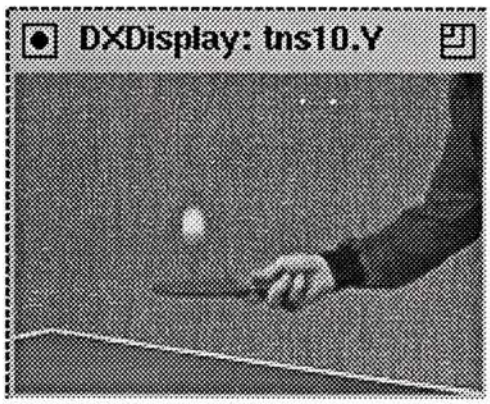
30. Zafar S., Zhang Y., and Baras J.S., " Predictive Block Matching Motion Estimation for TV Coding -- Part I: Inter-Block Prediction", IEEE Trans. Broadcasting, Vol.37, No.3. September 1991. p97-101

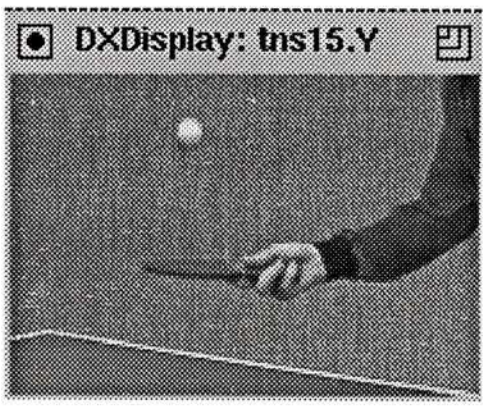
31. Zhang, and Zafar S., " Predictive Block Matching Motion Estimation for TV Coding -- PartII: Inter-Frame Prediction", IEEE Trans. Broadcasting, Vol.37, No.3. September 1991. p102-105
32. Bierling M., "Displacement estimation by hierarchical blockmatching", SPIE Vol.1001 Visual Comm. and Image Processing. 1988. p942-951
33. Liu B., and Zaccarin A., "New Fast Algorithms for the Estimation of Block Motion Vectors", IEEE Trans. Circuits & Systems for Video Technology, Vol.3, No.2, April 1993. p148-157
34. Berger, T., *Rate Distortion Theory: A Mathematical Basis for Data Compression*, Prentice Hall, Inc. N.J., Copyright 1971. p22
35. Rabani, M., *Digital Image Compression Techniques*, SPIE Optical Engineering Press, Bellingham, Wash.,. Copyright 1991. ISBN 0-8194-0648-1
36. Gonzales R, and Wintz, P, *Digital Image Processing 2nd ed.*, Addison-Wesley Publishing Company Inc., Copyright 1987.
37. Pratt, W. K (Editor)., *Image Transmission Methods*, Academic Press, N.Y., Copyright 1979.
38. Tugal D.A, and Tugal O., *Data Transmission (2nd ed)*., Mc Graw Hill, Inc. Copyright 1989. ISBN 0-07-065447-6.
39. Foley J.D., vanDam A., Feiner S.K., and Hughes J.F., *Computer Graphics Principles and Practice*, Addison-Wesley Publishing Company Inc., Copyright 1990. p584
40. Clarke R.J., *Transform Coding of Images*, Academic Press Inc. (London) Ltd. Copyright 1985.

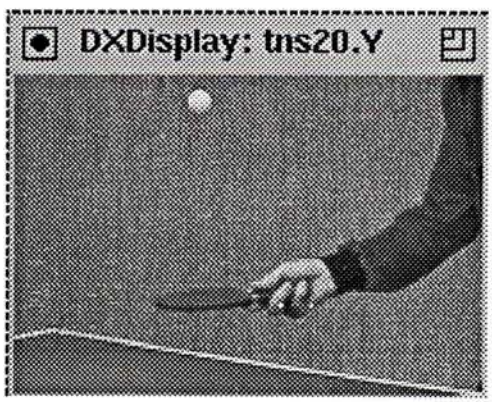
Appendix I: Selected Frames From Test Sequences

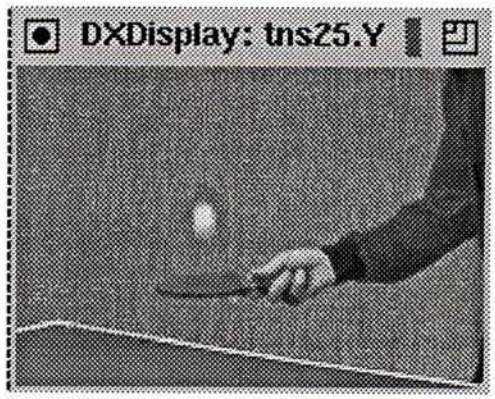


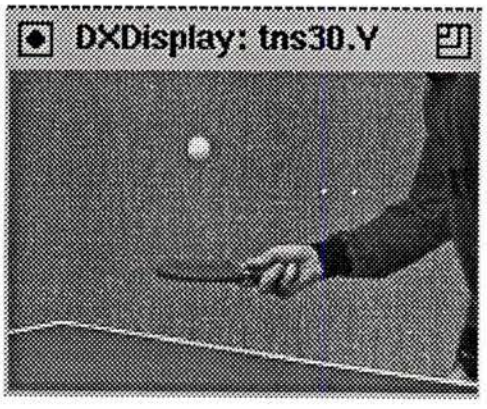


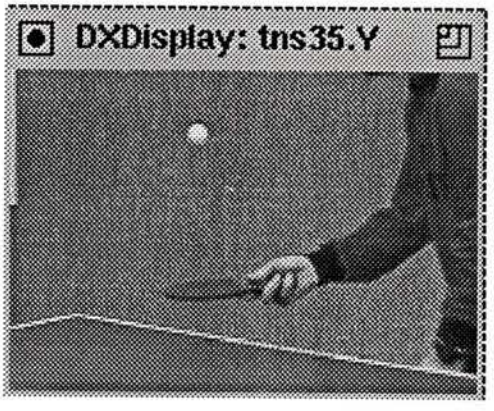


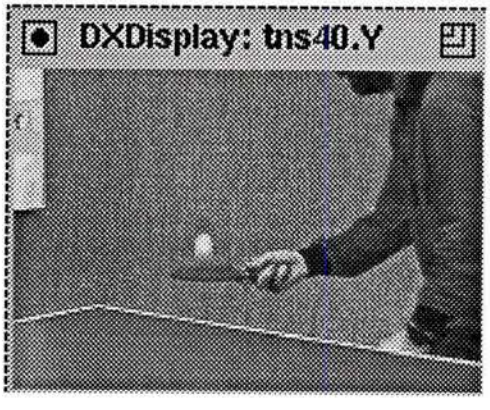


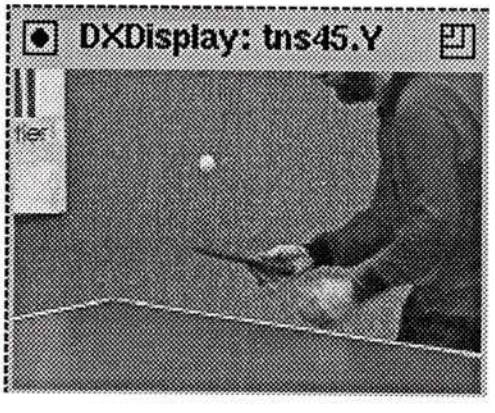


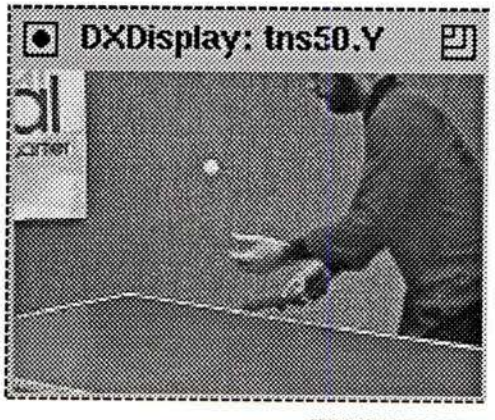



DXDisplay: tns30.Y


DXDisplay: tns35.Y


DXDisplay: tns40.Y


DXDisplay: tns45.Y


DXDisplay: tns50.Y

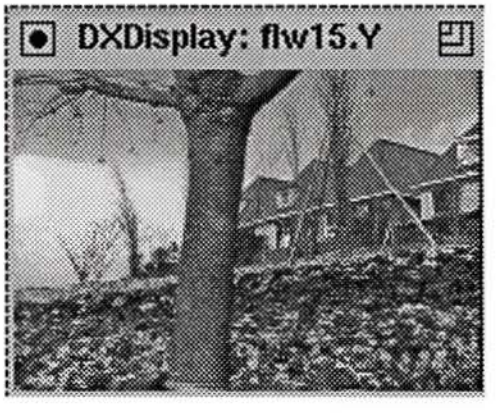

DXDisplay: tns55.Y


DXDisplay: tns60.Y


DXDisplay: flw1.Y


DXDisplay: flw5.Y


DXDisplay: flw10.Y


DXDisplay: flw15.Y


DXDisplay: flw20.Y


DXDisplay: flw25.Y


DXDisplay: flw30.Y

VITA

Surname:	*Chana*	Given Names:	*Jatinder Singh*
Place of Birth:	*Nairobi, Kenya*	Date of Birth:	*22nd, November, 1967*

Educational Institutions Attended:

University of Victoria	1987-1994

Degrees Awarded:

B.Eng. (First Class)	University of Victoria	1992

Honours and Awards:

University of Victoria Fellowship	1992-1994
BC ASI Graduate Student Scholarship Award	1992
Charles Humphrey Memorial Scholarship in Engineering	1990, 1991
University of Victoria Faculty Scholarship	1991
Woods Trust Scholarship	1991
James R. Bullock Memorial Scholarship	1991
Howard Petch Scholarship	1991

PARTIAL COPYRIGHT LICENCE

I hereby grant the right to lend my thesis to users of the University of Victoria Library, and to make single copies only for such users or in response to a request from the library of any other university, or similar institution, on its behalf or for one of its users. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by me or a member of the University designated by me. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Title of Thesis: Development and Analysis of Block-Based Motion Estimation Techniques for Efficient Video Compression

Author 

Jatinder Singh Chana

27th, September, 1994