2003

Algorithms for motion estimation.

Songtao. Huang
University of Windsor

Algorithms for Motion Estimation

by

Songtao Huang

A Thesis

Submitted to the Faculty of Graduate Studies and Research through the Department of Electrical and Computer Engineering in partial fulfillment of the requirements for the Degree of Master of Applied Science at the University of Windsor.

Windsor, Ontario, Canada

2003

Your file Votre référence
ISBN: 0-612-82884-0
Our file Notre référence
ISBN: 0-612-82884-0

98208

Acknowledgements

I would like to acknowledge the persons that either directly or indirectly have supported me in these years work.

A special thank goes to my supervisor, Majid Ahmadi, for his profound knowledge and keen discernment. He provides guidance and endless encouragement during all the work. His great patience and consideration and his support in my life make me feel indebted frequently. His great personality also establishes an ideal model for me to follow.

I would like to express my special appreciation to Professor Chunhong Chen. His selfless help and precise study attitude educate me greatly. The member of the committee, Professor David Ting who accepted to be member of the examining board, with the additional work involved; he provided very useful comments and suggestions.

Herein I also express my appreciation to Professor William C. Miller who taught me three courses in my comparable short study career.

My appreciation should also be expressed to my wife for her unlimited help.

Abstract:

There has been tremendous growth in video technology during the last decade. New applications like video e-mail, third generation of mobile phone, video communications, videoconferencing, video streaming on the web continuously push for further evolution of research in digital video coding. At the same time with the development of HDTV and digital TV, better video compression technologies are imperative. Compression is mainly realized by exploiting the redundancy present in the image data. One can easily show the similarity between two successive frames in a sequence of images as an example. This observation is referred to as temporal redundancy. The development of a proper scheme to exploit the temporal redundancy completely changes the required steps needed when compression of still pictures is needed versus the sequence of images. It also represents the key for very high performances in image sequence coding when compared to still image coding. Motion estimation (ME) and compensation techniques have shown their efficiency in reducing the temporal redundancy in this respect. The principle is the following. The displacement of objects between successive frames is first estimated (motion estimation). The resulting motion information is then exploited for an efficient interframe coding (motion compensation). Consequently the motion information along with the prediction error is transmitted instead of the frame itself.

This thesis presents several block matching techniques which are utilized for motion estimation in video coding. Among these algorithms the full search algorithm is known for its superiority in the performance over the other matching techniques; this method however requires an enormous computation costs. There are reported techniques in the literature which yield sub-optimal solutions. These are discussed in this thesis while another three efficient motion estimation algorithms for different applications are developed and presented in this thesis. The main objective of these proposed algorithms is to reduce the computation cost

while maintaining the performance. Experimental results with a large data set prove the utility of proposed method.

Also this thesis presents an architecture for the implementation of one of the proposed ME algorithms. This architecture uses a scheme named dynamic voltage scaling (DVS) capable of changing the power supply according the workload demand of the circuit and enjoys a low power characteristic in CMOS technology. This thesis also presents a discussion on the implementation of DVS in motion estimation circuits.

Index

List of figures

List of tables

Chapter 1

Introduction

1.1: Statement of the problem.

Recent advances in video technology have led to new communication media where information plays a key role. Digital video telecommunications have enjoyed an exponential growth in recent years and their continued evolution have been pushed by new applications such as videoconferencing, net-meetings, video e-mail, third generation mobile phones and video streaming over personal digital assistants. When compared to audio or texts, video signals contain a huge amount of information. Despite of increasing disks storage capacity and the development of broadband networks, their storage and transmission require effective compression techniques. At the same time with the boom of HDTV technology, the improvement of the image quality without increasing data storage and transmission bandwidth becomes a vital topic. This is because of the majority of proposed analog and digital HDTV systems are working toward an increase of about 100% in the number of horizontal and vertical pixels, which means the information contained in an image is many times larger than that are contained using the present technology. As a result advanced image compression techniques are in urgent need. These techniques are based on two principles: the reduction of the statistical redundancies in the data and the considerations of the human visual system imperfections.

What characterizes video compression and makes this domain of research substantially different from that of still image compression is presence of a sequence of images. Sequence of images contain an element of redundancy, that is two successive images coming one after the other with very short time interval usually are very similar, which means part of the new frame is unchanged from the last frame. This simple concept is called *temporal redundancy*.

The temporal redundancy is utilized to compress efficiently a sequence of images. This also represents a key task for very high performance image sequence coding when compared to still image coding. Therefore it is unnecessary to store and transmit the redundant information. The main objectives in video compression are to compare and obtain the difference between two successive frames, which is called motion estimation (ME) and motion compensation (MC). Moving picture expert group (MPEG) has become the international standard for the moving picture compression in the last decade. Many researchers have studied the power consuming part of MPEG, ME and MC in recent years [4,7]. In this thesis we present new algorithms for low power ME applications.

Specially, this thesis deals with the development of a methodology to estimate the motion field between two frames for video coding applications. In video coding, motion estimation is generally realized through the block matching technique[7]. Therefore, block matching motion estimation can be considered as the fundamental engine of video coding. Due to the high computation costs of ME, attention has been focused by many researchers to develop a more computationally effective ME algorithm.

Our study focuses on the fast ME algorithms and low power ME chip design at the architectural level. With the boom of portable computing and high-density VLSI circuits, low power consumption has emerged as a principal design requirement. Since ME algorithm in MPEG circuit is the most energy consuming device, to ensure low power requirement for MPEG application ME circuit has to be minimized for power consumption. A new low power chip design methodology---Dynamic Voltage Scaling (DVS) algorithm is developed in this thesis. Implementation of the DVS algorithm is difficult because of existence of complex variance between frames. To overcome this problem Predictive Hierarchical Search (PHS) motion estimation algorithm is proposed.

In this thesis we will present three novel ME algorithms to meet different requirements. Our study especially focuses on hierarchical motion estimation because of its promising outcome and efficient performance.

1.2 Organization of the thesis

The thesis is organized as follows; chapter 2 presents a detailed treatment of MPEG, its family and standards. Later the application of Motion Estimation in MPEG circuits is demonstrated.

Chapter 3 presents a survey of different algorithms for ME. A comparative study of these techniques is also provided to demonstrate the utility of each technique.

Chapter 4 discusses the development of two new ME algorithms in this thesis. The first search algorithm is based on the characteristic of the video sequence and is named plus search algorithm. It is shown that this process can improve the accuracy of the ME without increasing the computation burden. The second algorithm uses a hierarchical ME search technique with the aim to reduce the computation cost drastically while maintaining the accuracy close to the optimal result.

Chapter 5 develops a new low power design methodology through dynamic voltage scaling (DVS) technique. In this way the voltage source for the ME chip is dynamically altered based on the anticipation of the future workload of the chip to reduce the power consumption.

Finally, a summary of the contribution, conclusion derived from the research is presented in chapter 6.

Chapter 2

Introduction of MPEG: a review.

2.1What is MPEG.

MPEG (Moving Pictures Experts Group) is a group of organizations, companies and individuals that meet under ISO (*International Organization for Standardization*) to generate standards for digital video (sequences of images in time) and audio compression. In particular, they define a compressed bit stream, which implicitly defines a decompressor. However, the compression algorithms are developed by the individual manufacturers, that is where proprietary advantage is obtained within the scope of a publicly available international standard.

2.2 MPEG family and standards:

Until now the MPEG family of standards includes MPEG-1, MPEG-2, MPEG-4, MPEG-7, MPEG-21.

MPEG-1 [1] is used for as Video CD and MP3.

MPEG-2 [2] is used for 2-60 Mbit/s (TV and HDTV).

MPEG-4 [3] –7 and –21 are used for teleconferencing, describing the video features or enable extended use of multimedia content.

The official name of the **MPEG-1** standard is: "*Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Megabits per second*". The MPEG-1 Standard, which was drafted in 1992, defines a bit stream of compressed audio and video data with a data rate of 1.5 Mbits/sec, suitable for CD-ROMs and VideoCD applications. It is possible to generate MPEG-1 streams with other data rates. The MPEG-1 Standard is formally described in ISO/IEC 11172.

The **MPEG-2** Standard was designed later *for digital transmission of broadcast quality video with data rates from 2 to 10 Mbits/sec.* It was written to be more "generic", that is to address a broader range of applications, and is the compression standard for DVD and various digital television systems. The MPEG-2 Standard is described in ISO/IEC 13818 documents.

MPEG-4, whose formal ISO/IEC designation was ISO/IEC 14496, was released in November 1998 and was regarded as International Standard in January 1999. MPEG-4 is building on the proven success of three fields: digital television, interactive graphics applications (synthetic content) and the World Wide Web (distribution of and access to content) and will provide the standardized technological elements enabling the integration of the production, distribution and content access paradigms of the three fields.

MPEG-7, formally named "Multimedia Content Description Inter-face," is the standard that describes multimedia content so users can search, browse, and retrieve that content more efficiently and effectively than they could using today's mainly text-based search engines. It's a standard for describing the features of multimedia content.

The vision for **MPEG-21** is to define a multimedia framework to enable transparent and augmented use of multimedia resources across a wide range of networks and devices used by different communities.

2.3 MPEG Compression Overview

The basic idea behind MPEG video compression is to remove spatial redundancy within a video frame as well as temporal redundancy between video frames. As in JPEG, the standard for still image compression, DCT-based (Discrete Cosine Transform) compression is used to reduce spatial redundancy. Motion compensation is used to exploit temporal redundancy. The images in a video stream usually do not change much within small time intervals. Therefore the motion between frames can be described by a limited number of motion parameters (i.e. by motion vectors for translatory motion of pels). In this way much higher compression ratio can be obtained than compressing the frames independently [4].

2.3.1 Video imaging.

Video is simply an electronic sequence of still images displayed or projected (quickly) in succession to one another. As a result, the human mind is tricked into believing that people or objects in the presented sequence are moving. As we know the images in successive frames are correlated and only part of the objects in an image are changed.

Frame rate is the number of frames that are displayed to a viewer each second. For example, in motion picture film in the United States it is common to display 24 frames each second. In color television for the US home (called NTSC) 29.97 frames a second are displayed. [German PAL is based on a frame rate of 24 frames each second]. Even though computers are not normally thought of in terms of *frame rates*, most computers "refresh" the screen by repainting every element of the screen as often as 72 times a second. Obviously the higher frame rate is, we can feel the video smoother and less flickering. With the development of video technology, the frame rate will be improved dramatically. Thereby the speed of the MPEG encoder and decoder has to be improved accordingly, which is a big challenge to the video technology.

Frame size or **number of picture elements** is the next component of video. This is measured horizontally and vertically in pixels. "Pixels" are picture elements -- the small dots that make up the displayed picture. Some common dimensions, or resolutions numbers, in the computer world include: 640 horizontal pixels x 480 vertical pixels, 1024 horizontal x 768 vertical, and 800 horizontal x 600 vertical pixels.

The numbers of colors which make up each picture or frame is a third component of video. As is the case with a painter's palette, a color can be described in terms of several "primary" colors. For instance, when playing with paints as a child, mixing equal parts of red, yellow and blue created black. By mixing these primary colors in different combinations, it is possible to produce any other color. Color mixing works a bit different with light than with paints, but we can still make any color from three primaries. In the video world, however, we substitute green for yellow in our "primary" color palette.

Color Spaces: In mixing colors of light, we vary the amount of red, green and blue light that makes up the color of a pixel. To make video practical, it is necessary to limit the number of differing shades of red, green, or blue that can be generated. This puts an upper limit on the total number of colors that video can recreate. Here is an example of a common digital color scheme. Each primary color (red, green, or blue) may have 256 different levels or shades. Since a color may be composed of the three primaries, this means we can generate 16.8 million different colors, or 256 levels of red times 256 levels of green times 256 levels of blue (16.8 million roughly equals 256x256x256). The color for a pixel is normally written as follows:

pixel_color = (red_level, blue_level, green_level).

The previous example has just described a 24 bits video data. The term 24 bits comes from the fact that 256 shades of the primaries may be represented as an 8 bits value. Since it takes three primaries to represent a single value it takes 8+8+8 or 24 bits to represent color for a single pixel:

8bits red 8bits green 8bits blue RRRRRRRR GGGGGGGG BBBBBBBB = 24 bits

As a final note about color and video, it is possible to choose different primaries or entirely different colorspaces/colorsystems. The color systems described in this thesis therefore are not unique and can be changed according to different application or equipments. For example the common colorspace for printing is **CMYK** (which is abbreviation of cyan, magenta, yellow, and black), Another colorspace is **YCrCb**, (which stands for luminance (shade intensity) and chrominance-red and chrominance-blue (chrominance components define the hue and value of the color)). This colorspace is commonly used in video, since it more closely resembles the colorspace of human eyes, where rods detect luminance components and cones detect the chrominance components of the color. The international standard CCIR-610-1 specifies eight-bit digital coding for video. Conversion of **CMYK** to **YUV**(Alias of **YCrCb)** is described by the following equation 2-1.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} +0.2990 & +0.5870 & +0.1140 \\ -0.1687 & -0.3313 & +0.5000 \\ +0.5000 & -0.4187 & -0.0813 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad 2\text{-}1$$

CCIR-610-1 Rec. calls for two-to-one horizontal sub-sampling of Cb and Cr, to achieve 2/3 of the data rate of RGB with virtually no perceptible penalty. This is denoted by 4:2:2. JPEG and MPEG normally sub-sample Cb and Cr two-to-one horizontally and also two-to-one vertically, to get 1/2 the data rate of RGB. To obtain better results using sub-sampling you should not just drop and replicate pixels, but implement proper decimation and interpolation filters.

We assume our video source is a digital video with called **ITU-T 601** (formerly known as **CCIR 601**) format. Then the video is represented in the following fashion:

1. Frame rate of 30 frames per second.
2. Picture size of 720x480 (NTSC).
3. Color and colorspace: YCrCb 4:2:2.

Luminance (Y) is sampled at full resolution; each **chrominance component** (Cr and Cb) is sub-sampled half of the full resolution. On the average then, it takes 16 bits to represent each pel. Using these values, it is easy to calculate the total disk space required to hold one second of uncompressed video in this format:

720 horizontal Pixels × 480 vertical Pixels = 345600 pixel per frame

345600 pixel per frame × 30 frames per second =10368000 pixels per second

10368000 pixels per second × 2 bytes per pixel =20,736,000 total bytes per second

Therefore a 20 Gigabyte hard drive could hold about 1000 seconds of uncompressed video. Clearly this is not practical for most applications. For the HDTV, the three elements (YUV) will be increased largely to obtain better video quality, therefore the video compression turns to be a more important research topic.

2.3.2 Intra frame compression:

The human vision system exhibits some characteristics that are exploited by MPEG video compression. Since human vision notices the large object at first sight before noticing the detailed characteristic within that object [1], therefore, low spatial frequency information is much more noticeable than high spatial frequency information. This permits us to eliminate some information contained in a frame without affecting the observation process greatly.

As a result in MPEG video compression, some high spatial frequency information which is less noticeable to the human's eyes is discarded. The first step in this process is to map a static picture into the frequency domain. The DCT performs this transformation. MPEG video compression uses several techniques to achieve high compression ratios with minimal impact on the perceived video quality as outlined below.

1. Discrete Cosine Transformation (DCT)
2. Quantization
3. Huffman / Arithmetic Encoding

2.3.2.1 DCT:

In general, neighboring pixels in the image tend to be highly correlated. This permits further compression as the correlation between the adjacent pixels can be used to eliminate redundant information. This would create a downsized image with neighboring pixels uncorrelated to each other. Taking a DCT of an image help us to compress the information by deleting its high frequency components. It should be noted the lower frequencies DCT have higher magnitude. Therefore it is possible to transform image into fewer and decorrelated parameters. The Discrete Cosine Transform (DCT) is an ideal candidate to performance above transformation. The DCT/IDCT transform operations are described with Equations 2-2 & 2-3 respectively:π

$$F(u,v)=\frac{1}{4}\ C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\left[\frac{(2x+1)u.\pi}{16}\right]\cos\left[\frac{(2y+1)v.\pi}{16}\right] \qquad 2\text{-}2$$

C(u)= $\frac{1}{\sqrt{2}}$ for u=0 C(u)=1 for u=1,2,....7

C(v)= $\frac{1}{\sqrt{2}}$ for v=0 C(v)=1 for v=1,2,....7``

$$f(x,y)=\frac{1}{4}\sum_{x=0}^{7}\sum_{y=0}^{7}C(u)C(v)F(u,v)\cos\left[\frac{(2x+1)u.\pi}{16}\right]\cos\left[\frac{(2y+1)v.\pi}{16}\right] \quad 2\text{-}3$$

f(x,y) is the intensity of the pixel in row x and column y;

F(u,v) is the DCT coefficient in row u and v of the DCT matrix

2.3.2.2 Quantization:

The lower frequency DCT coefficients toward the upper left-hand corner of the coefficient matrix correspond to smoother spatial contours, while the DC coefficient [Element of the (0,0) of the DCT matrix] corresponds to a solid luminance or color value for the entire block. Also, the higher frequency DCT coefficients toward the lower right-hand corner of the coefficient matrix correspond to finer spatial patterns, or even noise within the image. Since it is well known that the human visual sense is less sensitive to errors in high frequency coefficients than it is for lower frequencies, it is desired that the higher frequencies be more coarsely quantized in their representation.

The process of DCT coefficient quantization is described as follows. Each 12-bit coefficient is divided by a corresponding quantization matrix value that is supplied from an intra quantization matrix [1] as shown in Table 2-1. It should be mentioned that other quantization matrices have been used by other researchers based on the applications in hand.

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Table 2-1 The MPEG quantization matrix

Each value in the DCT matrix should be divided by the corresponding value in the quantization matrix shown in Table 2-1. The goal of this operation is to force the as many as possible DCT coefficients to zero, or near zero, in this way the bit rate can be reduced.

2.3.2.3 Zig-zag:

Since most of the non-zero DCT coefficients will typically be concentrated in the upper left-hand corner of the matrix, it is obvious that a zigzag scanning pattern will tend to maximize the probability of achieving long runs of consecutive zero coefficients. This zigzag scanning pattern is shown in Figure 2-1. The block of quantized DCT coefficients as presented in Figure 2-2 demonstrates the elimination of the DCT coefficients with this process. Scanning of the example coefficients in a zigzag pattern results in a sequence of numbers as follows: 8, 4, 2, (5 zeroes), 1, (7 zeroes). This sequence is then represented as a run-length (representing the number of consecutive zeroes) and an amplitude (coefficient value following a run of zeroes). These values are then looked up in a fixed table of variable length codes, where the most probable occurrence is given a relatively short code, and the least probable occurrence is given a relatively long one.

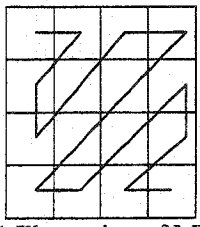


Figure 2-1 Illustration of MPEG Zigzag scanning pattern

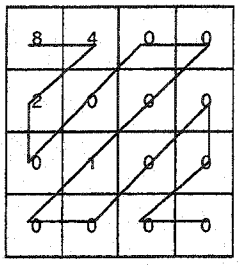


Figure 2-2 An example of DCT coefficients

2.3.2.4 Modified Huffman Coding

The idea behind Huffman coding is simply to use shorter bit patterns for more common characters [1], and longer bit patterns for less common characters.

To generate the Huffman code table, the procedures are as following:

Step 1 : Calculate the probability of occurrence of the data set which is to be encoded. Sort the symbols in decreasing order of probability

Step 2 : Combine two of the lowest probability symbol into an other symbol. The lowest probability in the list is assigned a bit value of one, the higher one is assigned a bit value of zero.

Step 3 : Repeat the step 2 recursively, until a tree form.

The Huffman encoding has following property:

1. No two code words may consist of identical sequence of code bits.

2. No information beyond the code itself is required to specify the beginning and end of any code value.

3. For a given number of symbols arranged in descending order of probability, the length of their associated code values will be such that the code associated with any given symbol will be less or equal to the length of the code assoiated with the next less probable symbol.

4. No more that two code words of the same length are alike in all but their final bits.

Modified Huffman coding uses fixed tables to perform Huffman coding. The DCT output is encoded using this technique to reduce the number of bits required.

2.3.3 Motion Compensation:

The previously discussed intra frame coding techniques were limited to processing the video signal in spatial domain, relative only to information within the current video frame. Considerably more compression efficiency can be obtained, however, if the inherent *temporal*, or time-based redundancies, are exploited as well. Successive video frames may contain the same objects (still or moving). Therefore it is possible to further compress the video data utilizing the correlating information between frames instead of compressing the frames independently. Motion estimation examines the movement of objects in an image sequence to try to obtain vectors representing the estimated motion.

Hereby the frames in a sequence can be divided into 3 types. The first type is "I" or intra frame. This is simply a frame coded as a still image, without using any past history. The second type is "P" or predicted frame. This frame is predicted from the most recently reconstructed I or P frame. Each macroblock in a P frame can come from a close match in the last I or P, or it can just be "intra" coded (like I frames) if there was not a good match. Lastly, there are "B" or bidirectional frames. These are predicted from the closest two I or P frames, one in the past and one in the future. Our search is for matching blocks in those frames, and therefore we propose three different techniques in this thesis and demonstrate the utility of each technique.

Motion compensation uses the knowledge of object in motion to achieve data compression. In interframe coding, motion estimation and compensation have become powerful techniques to

eliminate the temporal redundancy due to high correlation between consecutive frames. In real video scenes, motion can be a complex combination of translation and rotation. Translational motion is easily estimated while the rotation motion is not due to present motion estimation technology [5].

Motion compensation is a technique for enhancing the compression of P- and B-frames by eliminating temporal redundancy. Motion compensation typically improves compression by a factor of three compared to intra-frame coding. Motion compensation algorithms work at the macroblock level. When a macroblock is compressed by motion compensation, the compressed file contains the following information:

The spatial vector between the reference macroblock(s) and the macroblock being coded (motion vectors). The contained differences between the reference macroblock(s) and the macroblock being coded (error terms). Not all information in a frame can be predicted from a previous frame. Consider a scene in which a door opens: The visual details of the room behind the door cannot be predicted from a previous frame for which the door was closed. When a case such as this arises--i.e., a macroblock in a P-picture cannot be efficiently represented by motion compensation--it is coded in the same way as a macroblock in an I-picture using transform coding techniques [1].

The difference between B- and P-picture motion compensation is that macroblocks in a P-picture use the previous reference (I- or P-picture) only, while macroblocks in a B-picture are coded using any combination of a previous or future reference picture.

I-frames (Intra coded frames)

I-frames use DCT encoding only to compress a single frame without reference to any other frame in the sequence. Typically I-frames are encoded with 2 bits per pixel. Since the initial data comprises 4 bytes of Y, 1 byte of U and 1 byte of V (total 6 bytes = 48 bits) per pixel, this gives a compression ratio of 24:1. For random MPEG video, the decoder must start decoding from an I-frame not a P-frame. I-frames are inserted every 12 to 15 frames and are used to start a sequence, allowing video to be played from random

positions and for fast forward/reverse. Decoding of video can start only at an I-frame.

P-frames (Predicted frames)

P-frames are coded as differences from the last I or P frame. The new P-frame is first predicted by taking the last I or P frame and 'predicting' the values of each new pixel. P-frames use Motion prediction and DCT encoding. As a result P-frames will give a compression ratio better than I-frames but depending on the amount of motion present in that frame. Then the differences between the predicted and actual values are encoded instead of the frame itself for the I frame. Most prediction errors will be small since pixel values do not have large changes within a small area. The error values will therefore compress better than the values themselves. Quantization of the prediction errors further reduces the information.

B-frames (Bidirectional frames)

B-frames are coded as differences from the last or next I or P frame. B-frames use motion prediction as for P-frames but for each block either the previous I or P frame is used or the next I or P frame. P-frames use motion estimation and DCT encoding. Because B-frames require both previous and subsequent frames for correct decoding, the order of MPEG frames as required is not the same as the displayed order. This gives improved compression compared with P-frames, because it is possible to choose for every macroblock whether the previous or next frame is taken for comparison. The main advantage of the usage of B frames is its coding efficiency. In most cases, B frames require less bits for coding purpose. Quality can also be improved in the case of moving objects that reveal hidden areas within a video sequence. Backward prediction in this case allows the encoder to make more intelligent decisions on how to encode the video within these areas. Also, since B frames are not used to predict future frames, errors generated will not be propagated further within the sequence. Four coding methods are therefore possible for each macroblock in a B-picture:

. Intra coding: no motion compensation

. Forward prediction: the previous reference picture is used as a reference

. Backward prediction: the next picture is used as a reference

. Bidirectional prediction: either the previous or the next picture is used as a reference

Let us consider a sequence of frames shown in Figure 2-4. The encoder starts by encoding a complete representation of the first frame (similar to a static JPEG image). This is known as an Intra-Frame (or I-Frame). I-frames are necessary to give the decoder a starting point. The encoder could choose to encode the fourth frame in the video as a Predicted frame (or P-frame). This requires scanning the first frame (the *reference* frame) and the fourth frame, looking for macroblock size areas of the picture that appear similar. If the video contains moving objects, the encoder detects this. For areas in the image which have not changed between the first and fourth frame, macroblocks are skipped. Skipped macroblocks do not consume any data in the video stream. The decoder simply copies the macroblock from the previous reference frame. For areas that have changed slightly compared to the reference it takes the pixel difference and encodes this using DCT and quantization techniques. For areas that the encoder can detect the movement of an object from one macroblock position to another it encodes a motion vector and difference information. The motion vector tells the decoder how far and in what direction the macroblock has moved. Where the encoder cannot find a similar macroblock in the reference frame, the macroblock is encoded as if it was an I-frame.

The other frames in the sequence (second, third, fifth and sixth) could be encoded as Bidirectional Predicted frames (B-frames). Considering the second frame, this has two reference frame; the previous reference frame is the first frame and the next reference frame is the fourth frame. A B-frame can use macroblocks from either the previous or next reference frames, or preferably a combination of both. Using forward and backward motion vectors allows interpolation of 2 images, reducing noise at low bit rates.

Using our example, video would be encoded using the frame sequence shown in Figure 2-3:

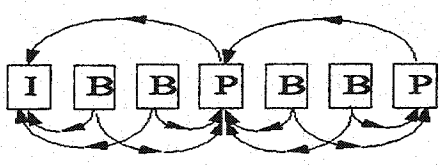


Figure 2-3 An example of encoded frame sequence

It should be noted that I frame can generally appear less regularly(i.e once every 12 frames). A more sophisticated encoder would dynamically detect which frames should be encoded using which frame types, e.g. a scene change would result in an I frame being inserted. Thus the sequence could end up looking more random, such as:

IBBBPBBIBBBPBPBBBBPBBBPBBBPBI...

By using information from previous and next pictures, substantial saving can be made in the bit requirements for P and B frames compared to I frames. Typically P-frames would require 30% to 50% the number of bits compared to I-frames, B-frames would require 15% to 25% the number of bits.

2.3.4 Conclusion:

In this chapter a brief review of MPEG is presented. The theory of MPEG compression, MPEG family members and their standards are also introduced. From above description, we can see that the most important part in the video compression is to compare the difference between the successive frames. Approximately 75% of the frames' compressions depend on the elimination of temporal redundancy of neighboring frames. Therefore, efficient motion estimation is very important for MPEG encoders. In the following chapters we will focus on the exploitation of low power and high efficient motion estimation algorithms and their architectures.

Chapter 3

Motion estimation

3.1 Introduction

There are different models for displacements of objects in two consecutive frames reported by many researchers[4,15-23]. These models are changing depending on the applications. Two of the main approaches for motion estimation are pel-recursive algorithm (PRA) and block-matching algorithm (BMA). PRAs predict recursively the displacement of each pixel from its neighboring pixels. BMAs assume that all the pixels within a block have the same motion activity. BMAs motions estimations is using rectangular blocks and produce one motion vector for each block. PRAs are more computationaly intensive and difficult to realize in hardware. In contrast, BMAs are more suitable for a simple hardware realization.

3.2 Pel-recursive

Pel recursive techniques rely on the hypothesis that the image luminance is invariant along motion trajectories. In these methods the spatiotemporal constraint is minimized recursively [7-9]. The recursion is usually carried out on a pel-by-pel basis, leading to a dense motion vector field. More generally, it can be performed on a block by block basis. These methods have been utilized for image sequence processing applications. The first pel-recursive algorithm has been proposed by Netravali and Robbins [4]. This algorithm is based on the minimization of the prediction error. They use spatial and temporal grey-level variations to estimate the instantaneous velocity at each pixel in the image. Assume $I_k(m,n)$ be the luminance value of a point at the spatial domain (m,n) of the kth image frame. Assume that the displacement is purely translated and the displacement between (k-1)th and kth frame is $\mathbf{D}=(u,v)^T$, then the following relationship holds in the moving area:

$I_k(m,n) = I_{k-1}(m-u,n-v)$ 3-1

The luminance error value between the same pels of the consecutive frame is defined as;

$FD(m,n) = I_k(m,n) - I_{k-1}(m,n)$ 3-2

This is also referred to as the frame difference (FD). Suppose that the initial estimate of displacement is $\mathbf{D}_{i-1}=(u_{i-1},v_{i-1})^T$,then the displaced frame difference(DFD) is also defined as;

$DFD(m,n, \boldsymbol{D}_{i-1}) = I_k(m,n) - I_{k-1}(m-u_{i-1},n-v_{i-1})$ 3-3

If we assume a spatially linear intensity image model, then the DFD is proportional to the error ($\mathbf{D}_{i-1}$-D) of the estimated displacement. Thus DFD is the prediction error for the estimated value D_{i-1}.The approach taken by PRA is to seek a local minimum of the function$(DFD)^2$ in the vicinity of (m,n). The displacement estimation process implements a recursive steepest descent algorithm given by;

$D_i = D_{i-1} - (\varepsilon \times DFD(m,n,D_{i-1}) \times \square I_{k-1}(m-u_{i-1},n-v_{i-1}))$ 3-4

Here $\square I_{k-1}$ is the spatial gradient of I_{k-1},. The convergence factor ε, which affects the performance significantly, is normally chosen as a positive fractional number. The above computation is required in the PRA and is performed recursively with each pel to obtain the final result. Improved algorithms based on the same principle have been proposed in Reference [7-10]. A more recent contribution for improving accuracy can be found in Reference [11]. When the update of the displacement vector is based only on previously transmitted data (causality), the decoder is able to estimate the same motion field than the encoder. Pel recursive algorithms suffer from two additional drawbacks. First, since the objective function contains many local minima, the iterative procedure may converge to a local minimum rather than the global one. In particular, these algorithms are very sensitive to noise. Second, large displacements and discontinuities in the motion field cannot be efficiently handled.

3.3 Block matching algorithms

3.3.1 Introduction:

Block matching algorithm is an alternative motion estimation techniques which enjoys a better computational efficiency than the pel-recursive algorithm. Therefore, block matching motion estimation is considered as generic engine of video coding with good compression performance. This however, requires an enormous computational burden. One can find a tradeoff between the accuracy of the prediction, and the number of operations needed. The fundamental problem is to understand how to realize a general model that should work for every sequence (static and videoconferencing , as well as more complex sport sequences) and format. In a typical use of this method, images are partitioned into non-overlapped rectangular blocks (Figure 3-1). Each block is viewed as an independent object, and it is assumed that the motion of picture elements (pels) within the same block are uniform. The block matching method is essentially an object recognition approach. Block matching is a correlation technique that searches for the best match between the current image block and candidates in a confined area of the previous frame. The displacement (motion) vector is the result where the new location of the object (block) is identified. The size of the block affects the performance of motion estimation. For the natural objects boundaries small block performs better than large block; it also provides good approximation to real motion, which is now approximated by piecewise translation movement. However, small block size produces a large amount of motion information, which may result in the increase of the number of transmission bits and compression complexity for condensation of the motion information. Small blocks also suffer from the object (block) ambiguity problem and the random noise problem. Large blocks, on the other hand, may produce less accurate motion vectors since a large block may likely contain pels moving at different speeds and in different directions. In entertainment and teleconferencing typical picture size ranges from 240 lines by 352 pels to 1080 lines by 1920 pels (HDTV). Block sizes of 8x8 or 16x16 are generally considered

adequate for these applications. The international video transmission standards, H.261, H.263, MPEG-1, MPEG-2, and MPEG-4 all adopt the block size of 16x16.

Block matching algorithms estimate number of motions on a block by block basis. For each block in the current frame, a block from the previous frame is found for matching based on a given criterion.

Figure 3-1 Illustration of how frames are divided into blocks

The basic operation of block matching is to pick up a candidate block, and calculate the matching function (usually a nonnegative function of the intensity difference) between the candidate and the current block. Theoretically, this operation should be repeated until all the candidates have been processed and the best match candidate is identified. The relative location of the best candidate gives the estimated displacement vector. Several parameters are involved in the searching process:

- The number of candidate blocks, search areas,
- The matching functions,
- Search order.

Each of these have impacts on the final results with respect to both accuracy and complexity. This motion vector's search range is chosen either empirically and imposed by hardware constraints, or is decided by the different applications and standards.

3.3.2 Objective functions for motion estimation.

In order to measure the similarity between the block of the current frame and a candidate block in the reference frame, various matching criteria such as mean square error (MSE), mean absolute error (MAE) and maximum matching pel count (MPC) are applied. It is necessary to choose a proper matching function in the process of searching for the optimal location. The selection of the matching function has a direct impact on computational complexity and coding efficiency. Assume the top left corner of the searching block of frame n is (k,l), the displacement of the matching block of frame n-1 is (u,v) and the block size equals to M×M, the MAE, MSE and MPC matching criteria is described below:

3.3.2.1: Mean absolute error (MAE)

The MAE is the most popular criterion for hardware implementations. Its hardware implementation complexity is much lower than that of the MSE criterion. To compute MAE, we add the absolute value of the errors between reference block and search block as shown below.

$$MAE_{(k,l)}(x,y)=\frac{1}{m^2}\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}|I_n(k+i,l+j)-I_{n-1}(k+x+i,l+y+j)| \qquad 3\text{-}5$$

This straightforward and succinct approach has naturally become the dominant criterion of motion estimation.

3.3.2.2: Mean square error (MSE)

MSE compute the summation of all of the square values of the errors between reference block and search block. The MSE can express the difference between successive frames better. However, in view of VLSI design, the multiplication should be the last arithmetic hardware to be considered because of its high complexity and great chip area requirement in the real world design.

$$MSE_{(k,l)}(x,y)=\frac{1}{m^2}\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}|I_n(k+i,l+j)-I_{n-1}(k+x+i,l+y+j)|^2 \qquad 3\text{-}6$$

3.3.2.3: Maximum matching pel count (MPC)

$$MPC(k,l;u,v)=\frac{1}{m^2}\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}|T(k+i,l+j;u,v)| \qquad 3\text{-}7$$

where

$$T(s,t;u,v)\begin{cases}1 & \text{if}|I_n(s,t)-I_{n-1}(s+u,t+v)|<=t\\ 0 & \text{Otherwise}\end{cases} \qquad 3\text{-}8$$

where t is a predetermined threshold. The best matching block is obtained using this matching criterion when MPC value is the maximum. Considering the extensive application of video compression, the determination of an uniform value t to match all of the cases is not easy.

3.4 Block matching motion estimation

As illustrated in Figure 3-2, a reference block taken from the current frame will be compared with the same size block around the corresponding coordinate in the searched frame. The block in the search area can be checked according to one of the three criteria explained above. According to the standards of MPEG the block size is usually chosen as 16×16, the search area is usually chosen as –8/+7 or –16/+15. With the development of HDTV, the search size of block and search area will be increased exponentially. The block size may be increased to 128×128. Therefore the development of more efficient motion estimation algorithm is the first priority because of its increasing computation cost.

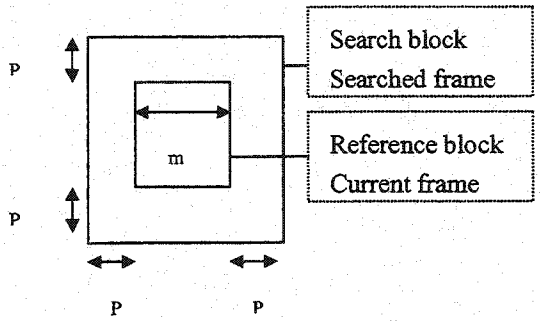


Figure 3-2. Block matching motion estimation

m is the length and width of reference block.

P is the maximum displacement of the reference block within the search area in the four spatial directions

It should be noted that there are $(2p+1)^2$ potential candidate locations to be compared with the reference block in the search area.

3.5 Full search motion estimation.

Full search motion estimation [13] is the straight forward and optimal method, which compares the difference between the blocks exhaustively inside the search area. Figure 3-3 demonstrates full search algorithm with the block size of 4×4 and the maximum motion displacements of –2/+1.

.

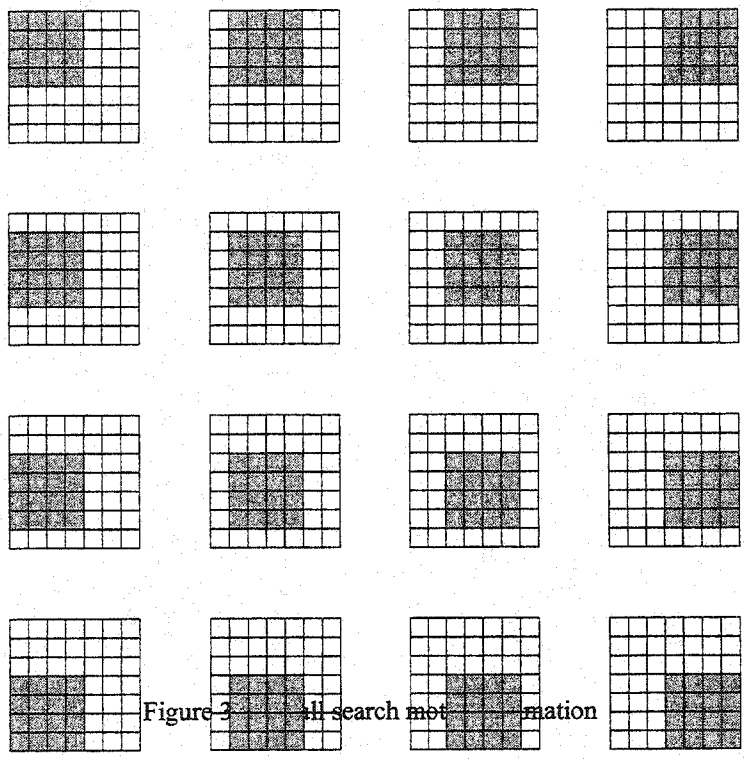


Figure 3 Full search motion estimation

All of blocks in the locations of the search area have to be compared with the reference block, to obtain Search block ons es Reference block er, considering the computation cost, this method is not used widely. For example to obtain the proper motion vector the block with the size of 16×16 pixels is searched in -7/+8 search area. Assuming that the frame rate is 30 frame/per second and the frame size is 352×288 pixels, there are 22×18 blocks in a frame, and

there are 225 locations for MAE values to be calculated for each block to perform full search algorithm. This requires a total of 684288000 times add and subtraction operations in a second. The computational requirements for real world application utilizing MPEG-2 is enormous for a VLSI implementation.

3.6 Performance measure parameters

3.6.1 Peak signal noise ratio (PSNR)

PSNR (peak signal to noise ratio) is used to evaluate the quality of a reconstructed video sequence of a video codec quantitatively. Before introducing the PSNR, we will describe SNR. Signal-to-noise ratio(SNR) measures are estimates of the quality of a reconstructed image compared with an original image. The basic idea is to compute a single number that reflects the quality of the reconstructed image as shown Equation 3-7:

$$\text{SNR}=\frac{\sum\left[f(i,j)^2\right]}{\sum[f(i,j)-F(i,j)]^2} \qquad 3\text{-}7$$

In fact, traditional SNR values do not equate with human subjective perception, though signal-to-noise measures are very popular because they are easier to compute. Bearing in mind that higher measures do not always mean better quality. The proposed measure computes the peak signal-to-reconstructed image's noise, which is called PSNR. Assuming that we are given a source image f(i,j) that contains N by N pixels and a reconstructed image F(i,j) where F is reconstructed by decoding the encoding f(i,j). Error are computed on the luminance signal only when the pixel values f(i,j) range between black (0) and white (255). First we compute the MSE of the reconstructed image as follows:

$$\text{MSE}=\frac{\sum\left[[f(i,j)-F(i,j)]^2\right]}{N^2} \qquad 3\text{-}8$$

The summation is over all pixels. The root mean squared error (RMSE) is the square root of MSE.PSNR in decibels (dB) is computed by using

$$PSNR=20\log_{10}\left(\frac{255}{RMSE}\right) \qquad 3\text{-}9$$

The value 255 is due to the peak value of an 8-bit quantized pixel. Typical PSNR values range between 20 and 40. They are usually reported in two decimal points (e.g., 25.47). Although the actual value is not meaningful, however the comparison between two values for different reconstructed images gives one measure of quality. The MPEG committee used an informal threshold of 0.5 dB PSNR.

Some definitions of PSNR use 255^2/MSE rather than 255/RMSE. Either formulation will work because we are interested in the relative comparison, not the absolute values.

3.6.2 Mean Absolute Error (MAE)

The MAE can represent the motion compensation more directly and clearly. As it is known for the P frames the motion compensation is searching the most similar block in the search area and compute the difference between reference and search blocks the same as carried out for the I frame. Smaller MAE means less information contained in the DCT transform, therefore higher compression ratio can be obtained. Of course one can judge the effects of different algorithms directly by comparing its mean MAE values for a sequence of images. Therefore, in this thesis MAE is used as the criterion for comparing different algorithms.

3.7 Motion Estimation Algorithm Complexity

Optimization of motion estimation algorithms has been studied extensively for the last two decades [12-23] because of its fundamental impact on compression efficiency and its high requirements in both energy consumption and data bandwidth. In this section the complexity analysis of motion estimation is reviewed. At the same time the advantages/disadvantages of the different technologies, both in terms of compression quality and hardware implementation efficiency are characterized. In Figure 3-2, we described the motion estimation process for the current macroblock of size MxM within a search range of (+/-p, +/- p) in the search frame.

Each time we generate a new candidate motion vector we need to evaluate the fitness of this candidate by matching the two macroblocks. This process is repeated until we find a final motion vector with minimal cost. The complexity cost of performing motion estimation for a macroblock is given by the equation

$$CX_{MB}=CP\times[(M\times M)\times CX_P]+CX_{Control} \quad 3\text{-}10$$

This equation shows that the complexity of CX_{MB} is proportional to the number of checking points CP, the number of pixels used to perform the matching (MxM), and the complexity to evaluate one pixel match CX_P. Most of the selected location algorithms focus on the reduction of CP while assuring the accuracy.

$CX_{Control}$ expresses the additional complexity for motion vectors generation to control the dataflow or select the minimum value in an array of obtained MAE value. The complexity of $CX_{Control}$ depends on different algorithms. The $CX_{Control}$ only occupies a small portion of the whole calculation.

In motion estimation CX_P judges the absolute value of the difference of two pixels and add these values together. This process therefore requires an adder and a subtraction and a comparator. The full search algorithm computation cost is:

$CX_{MBfull} = (2p+1)^2\times [(M\times M)\times CX_P]+CX_{Control}$.

All of the other algorithms focus on the reduction of CP or CX_P to reduce the final computation cost [12-23].

3.8 Motion estimation algorithms:

3.8.1 Three Step Search (TSS)

This algorithm was introduced by Koga et al in 1981 [12]. It became very popular because of its simplicity, and near optimal performance. It searches for the best motion vectors in a coarse to fine search pattern. The algorithm may be described as:

Step 1: An initial step size is picked, which should be equal to or slightly larger than half of the maximum search range. The central point of the square search area and eight search points located on the search area boundaries are chosen to calculate the MAE values.

Step 2: The step size is reduced by a factor of two. The center is moved to the point with the minimum distortion.

Steps 1 and 2 are repeated until the step size becomes smaller than 1. A particular path for the convergence of this algorithm is shown in Figure 3-4:

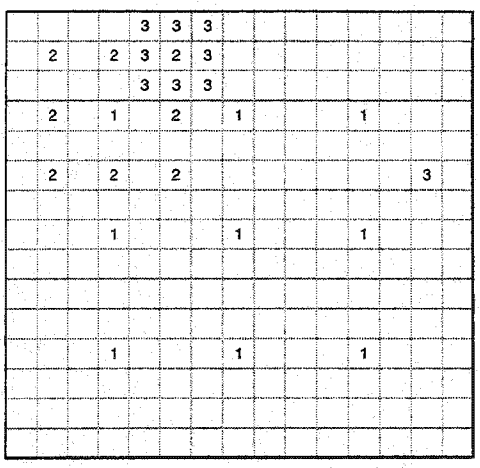


Figure 3-4 : Example path for convergence of Three Step Search

Locations marked 1 means where the first step search performed

Locations marked 2 means where the second step search performed

Locations marked 3 means where the third step search performed

One problem that occurs with the Three Step Search is that it uses a uniformly distributed

checking point patterns in the first step, while the step size is large in comparison with other algorithms. As a result the TSS becomes inefficient for small motion vector.

-Advantages:

1. Low complexity in terms of selected candidate locations.

2. Good regularity in terms of motion vector generation (two folded loops).

-Disadvantages

1. Complexity factor slightly increasing with search area.

2. All the data in the search area must be accessed (high data bandwidth).

3. The result of simulation is not very good in comparison with other algorithms which have similar computation costs as demonstrated later in this chapter.

3.8.2 Two Dimensional Logarithmic Search (TDL)

This algorithm was introduced by Jain & Jain [13]. Although this algorithm requires more steps than the Three Step Search, it is more accurate, especially when the search window is large. The algorithm is described as:

Step 1: Select an initial step size. Look at the block at the Center of the search area and the four blocks at a distance of s from this center on the X and Y axis (The five positions form a plus sign).

Step 2 : If the position of best match is at the center, reduce the step size by half. In case one of the four points is the best match, then this point becomes the center and step 1 is repeated.

Step 3: When the step size becomes 1, all the nine blocks around the center are chosen for the search and the best among them is picked as the required block.

A particular path for the convergence of the algorithm is shown in Figure 3-5:

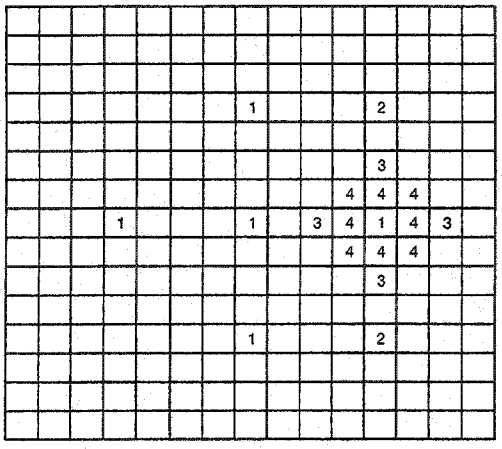


Figure 3-5: Example path for convergence of Two Dimensional Logarithmic Search

Locations marked 1 means where the first step search performed

Locations marked 2 means where the second step search performed

Locations marked 3 means where the third step search performed

Locations marked 4 means where the fourth step search performed

Many variations of this algorithm exist and they differ mainly in the way in which the step size is changed [13]. One can argue the step size should be reduced by two at every stage; while one can reduce the step size by a factor of two if and only if the edge of the search area is reached.

-Advantage:

It is suitable for sequences with fast motion.

-Disadvantage

If the motion vector is at an angle to X-Y axis, the motion estimation process is not straight forward and this may affect the accuracy of the algorithm.

3.8.3 Binary Search (BS)

This is also one of the algorithms that are used for motion estimation by MPEG-Tool [14]. The basic idea behind this algorithm is to divide the search window into a number of regions

and do a full search only in one of these regions. The algorithm is described by the following steps :

Step 1 : The MAE is evaluated on a grid of 9 pixels that include the center, the four corners of the search window and four pels at the boundaries. The search window is divided into regions based on these points.

Step 2: A full search is performed in the region corresponding to the point with the smallest MAE.

The convergence of the algorithm may be viewed in Figure 3-6. The pels that lie between the dashed lines are never considered. Hence, although the Binary search requires fewer comparisons (the worst case scenario for this search window is 33 comparisons), its performance is not acceptable since there is a zone where its pixels are never considered in the ME algorithm.

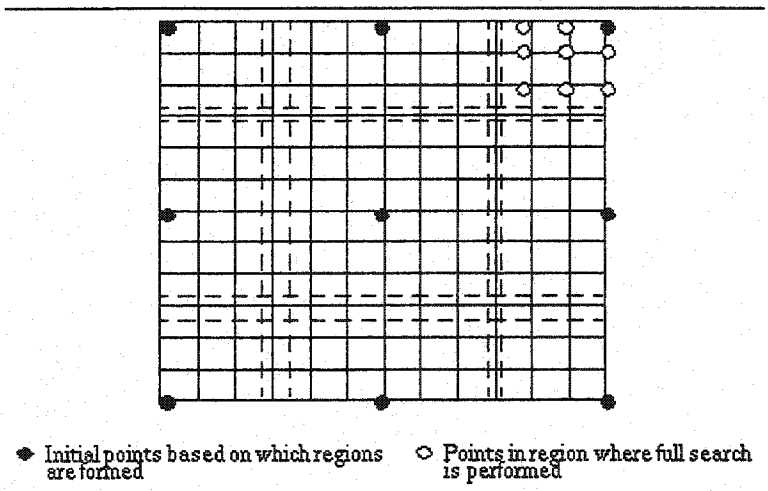


Figure 3-6 : Example path for convergence of Binary Search

-Advantage:

It is more accurate than three step search.

-Disadvantages

The computation cost is high, because the algorithm conducts full search in part of the ME process.

Some locations in the area are not checked, which contributes to the lack of accuracy of the algorithm.

If in the first step search wrong zone is selected, obviously the algorithm fails to obtain the right result.

3.8.4 Four Step Search (FSS)

This algorithm was proposed in 1996 by L.M. Po and W.C. Ma [15]. This algorithm is based on this hypothesis that the block motion field of real world image sequence is usually gentle, smooth, and varies slowly, the global optimum motion vector distribution is highly biased at the central area. The algorithm starts with a nine points comparison and then the other points for comparison are selected based on the following algorithm:

Step 1: Start with a step size of 2. Pick nine points around the search window center. Calculate the MAE value and find the point with the smallest MAE value. If this point is found to be the center of the searching area, go to step 4, otherwise go to step 2.

Step 2: Move the center to the point with the smallest MAE. The step size is maintained at 2. The search pattern, however, depends on the position of the previous point with minimum MAE.

a) If the previous minimum point is located at the corner of the previous search area, five points are picked (as shown in the Figure 3-7).

b) If the previous minimum MAE point is located at the middle of the horizontal or vertical axis of the previous search window, three additional checking points are picked. (as shown in the Figure 3-7).

Locate the point with the minimum MAE. If this is at the center, go to step 4 otherwise go to step 3.

Step 3 : Repeat step 2 once more.

Step 4: The step size is reduced to 1 and all nine points around the center of the search are examined.

The computational complexity of the Four Step search is less than that of the Three Step search, while the performance in terms of quality is comparable. It also performs better than the Three Step search when it encounters image sequences with complex movements like camera zooming and fast motion. Hence it is regarded as a very attractive strategy for motion estimation.

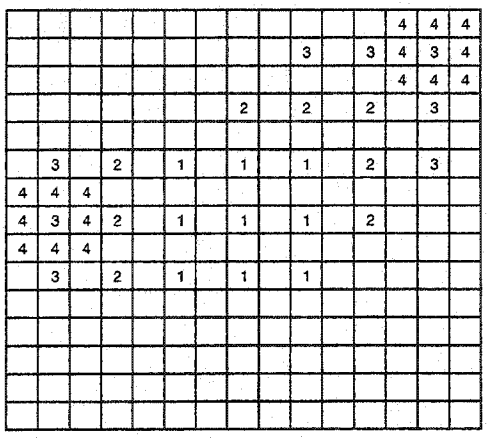


Figure 3-7: Example path for convergence of Four Step Search.

---Advantages:

-Very low complexity in terms of the selected candidate locations.

-More accurate than 3SS.

-Only portion of search area pixel data should be accessed (memory bandwidth saving).

-For small motion, fewer step calculations are needed.

---Disadvantage:

-Risk of local minima if motion is far away from (0,0).

3.8.5 Orthogonal Search Algorithm (OSA)

This algorithm was introduced by Puri in 1987 [16] and it is a mixture of the Three Step Search and the Two Dimensional Logarithmic Search. It has a vertical stage followed by a horizontal stage for the search for the optimal block. That algorithm is described as follows:

Step 1 : Pick a step size (usually half the maximum displacement in the search window). Take two points at a distance of step size in the horizontal direction from the center of the search window and locate (among these) the point of minimum MAE. Move the center to this point.

Step 2 : Take two points at a step size distance from the center in the vertical direction and find the point with the minimum MAE.

Step 3: Halve the step size, if it is greater than one, or else halt.

A particular path for the convergence of the algorithm may be shown in the Figure 3-8.

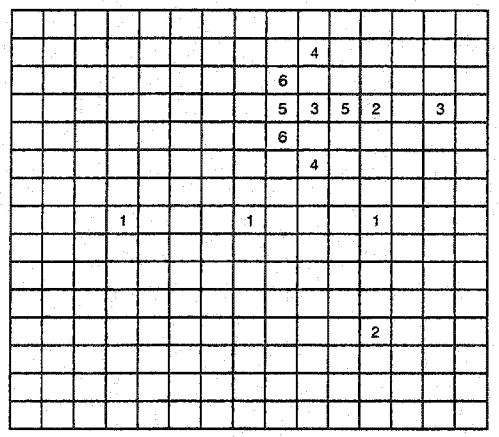


Figure 3-8 : Example path for convergence of Orthogonal Search

Locations marked 1 means where the first step search performed
Locations marked 2 means where the second step search performed
Locations marked 3 means where the third step search performed
Locations marked 4 means where the fourth step search performed
Locations marked 5 means where the fifth step search performed
Locations marked 6 means where the sixth step search performed

---Advantage:

For small motion, this algorithm can get the result quickly.

---Disadvantage:

It can easily be trapped in local minima.

3.8.6 One at a Time Algorithm (OTA)

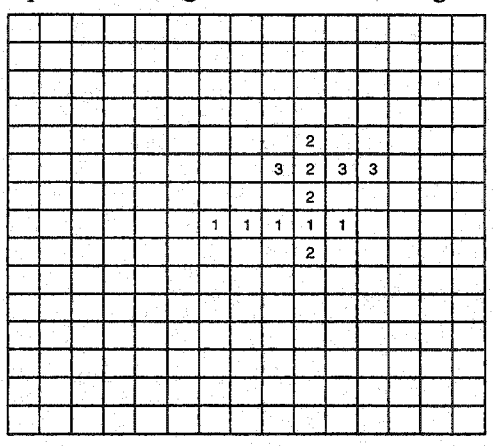
This is a simple, but effective way of trying to find a point with the optimal block [17]. During the horizontal stage, the point on the horizontal direction with the minimum MAE is found. Then, starting with this point, the minimum MAE in the vertical direction is found. The algorithm is described as follows :

Step 1 : Pick three points about the center of the search window in horizontal direction.

Step 2 : If the smallest MAE is in the center point, start the vertical stage, otherwise look at the next point in the horizontal direction closer to the point with the smallest MAE (from the previous stage). Continue looking in that direction till you find the point with the smallest MAE.

Step 3: Repeat the above steps in the vertical direction for the point which has the smallest MAE in the horizontal direction.

One particular search path for the algorithm is shown in Figure 3-9.



Locations marked 1 means where the first step search performed

Locations marked 2 means where the second step search performed

Locations marked 3 means where the third step search performed

Figure 3-9: Example path for convergence of One at a Time Algorithm

Advantage:

The procedure can be stopped at any step to ensure low computational costs.

Disadvantage:

The algorithm can easily be trapped in local minima.

3.8.7 Cross Search Algorithm (CSA)

This algorithm was introduced by M. Ghanbari in 1990 [18]. This algorithm is based on a logarithmic step search, however, the main difference between this algorithm and the logarithmic search method presented before is the locations of the search points which are at the end points of a × shape search rather than the + shape search presented in the previous method. The algorithm is described as follows:

Step 1 : The center block is compared with the current block and if the MAE is less than a given threshold, the algorithm stops.

Step 2 : Pick the first set of points in the shape of a "x" around the center. (The step size picked is usually half the maximum displacement). Move the center to the point of minimum MAE.

Step 3 : If the step size is larger than 1, divide it by two and repeat step 2, otherwise go to step 4.

Step 4 : If in the final stage the point of minimum MAE is the bottom left or the top right point in the search area, then evaluate MAE at 4 more points around the location with minimum MAE with a search area of a "+" shape. If, however, the point of minimum MAE is at the top left or bottom right point, evaluate the MAE at 4 more points around it in the shape of a "x".

A probable search path for the algorithm is shown in Figure 3-10. The cross search algorithm requires 5 + 4 log2w comparisons, where w is the largest allowed displacement. The algorithm has a low computational complexity. It is, however, not the best in terms of the result it produces for motion compensation.

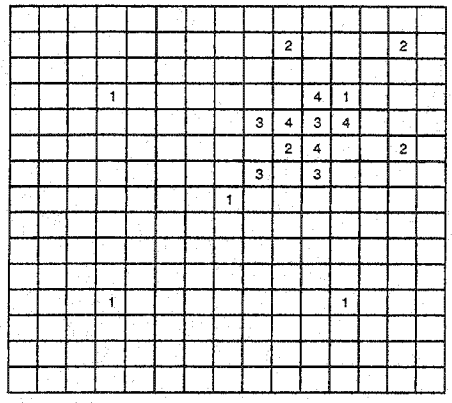


Figure 3-10: Example path for convergence of Cross Search Algorithm.

Locations marked 1: Initial set of locations *Locations marked 2: Second step locations*

Locations marked 3: Third step locations *Locations marked 4: Fourth step locations*

---Advantage:

Comparable low computation cost.

---Disadvantage:

Performance is worse than TSS.

3.8.8 Spiral Search (SS)

The spiral search algorithm was proposed by Zahariadis and Kalivas in 1995 [19]. It combines the method of the Three Step search and the binary search. By doing so, it tends to not only speed up the computation, but also removes the problem of the Binary search, where there is a zone of pixels that is never evaluated. The algorithm may be described as follows:

Step 1 : The step size is picked to half of the maximum displacement in the search window. The point of minimum MAE is found from among the nine points selected in the following manner. Five points are selected in the shape of a "+" around the center of the search window (at a distance of step size in the vertical and horizontal directions). The remaining four points are picked at the corners of the search window.

Step 2 : The step size is reduced and a search is performed around the point with the smallest MAE. This is repeated till the step size falls to 1.

A probable search path for the algorithm is shown in Figure 3-11.

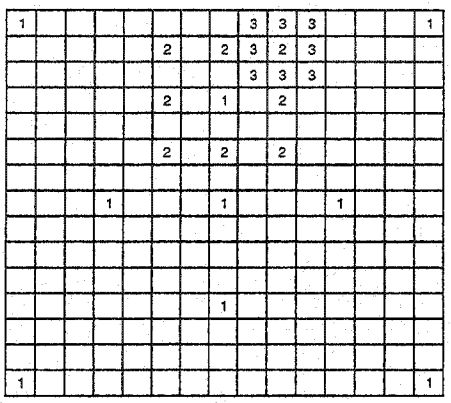


Figure 3-11: Example path for convergence of Spiral Search

Locations marked 1: Initial set of locations

Locations marked 2: Second step locations

Locations marked 3: Third step locations

---Advantage:

This search gives performance (in terms of quality) comparable to the Three Step search, with less computational complexity.

---Disadvantage:

The spiral search does not outperform the four step search.

3.8.9 New Three Step Search

The new three-step search algorithm (NTSS) is described in [20]. It is a modified version of the Three Step search algorithm to improve the quality for small motion video sequences occurring in video conferencing applications. For these video sequences, the motion vector distributions are highly center biased. Therefore, eight additional checking points, centered on the origin are searched in the first step of NTSS. If the minimum is indeed center biased, the average number of searched locations becomes (17 +(3 or 5)) = 20 or 22. If the minimum is not located from the eight additional points, the algorithm behaves exactly the same as the

Three Step search. Shown in Figure 3-12

. Advantages:

-Low complexity in terms of selected candidate locations.

-Obtain high quality at very low processing power for center biased motion scene.

Disadvantages:

-Candidate vector generation slightly more sophisticated than TSS.

-Complexity factor increases with search area.

-Extra computations are needed if the motion vector is not in the 3×3 window around the center in comparison with other TSS.

-Risk of local minima if motion not centered around (0,0).

-All the search area memory data should be accessed (high data bandwidth).

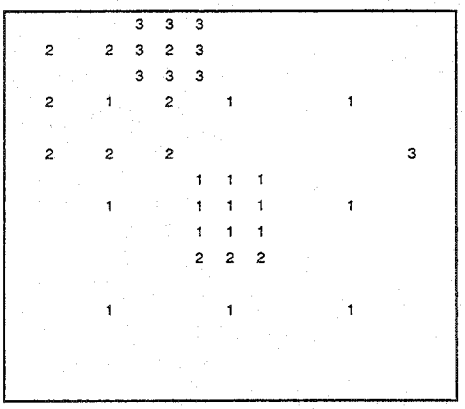


Figure 3-12 : Example path for convergence of Three Step Search

Locations marked 1 means where the first step search performed

Locations marked 2 means where the second step search performed

Locations marked 3 means where the third step search performed

3.8.10 Hierarchical Search Block Matching Algorithms

To reduce the complexity of the motion search algorithms, coarse-to-fine hierarchical searching schemes have been suggested [21]. This reduction in the computation is due to the reduced image size at higher levels .For hierarchical algorithm the motion vector search is

divided into several levels in a way that lower levels use partial distortions with higher decimation ratios. For each level, a number of candidate motion vectors with minimal partial distortions are selected to enter the next level. In this way this method reduces the probability of being trapped in the local minimum value while reducing the computational costs. Therefore, the CX_P in Equation 3-9 can be reduced, and as a result the total computation cost can be decreased proportionally, while avoiding local minima. An example of hierarchical ME search algorithm is shown as Figure 3-13.

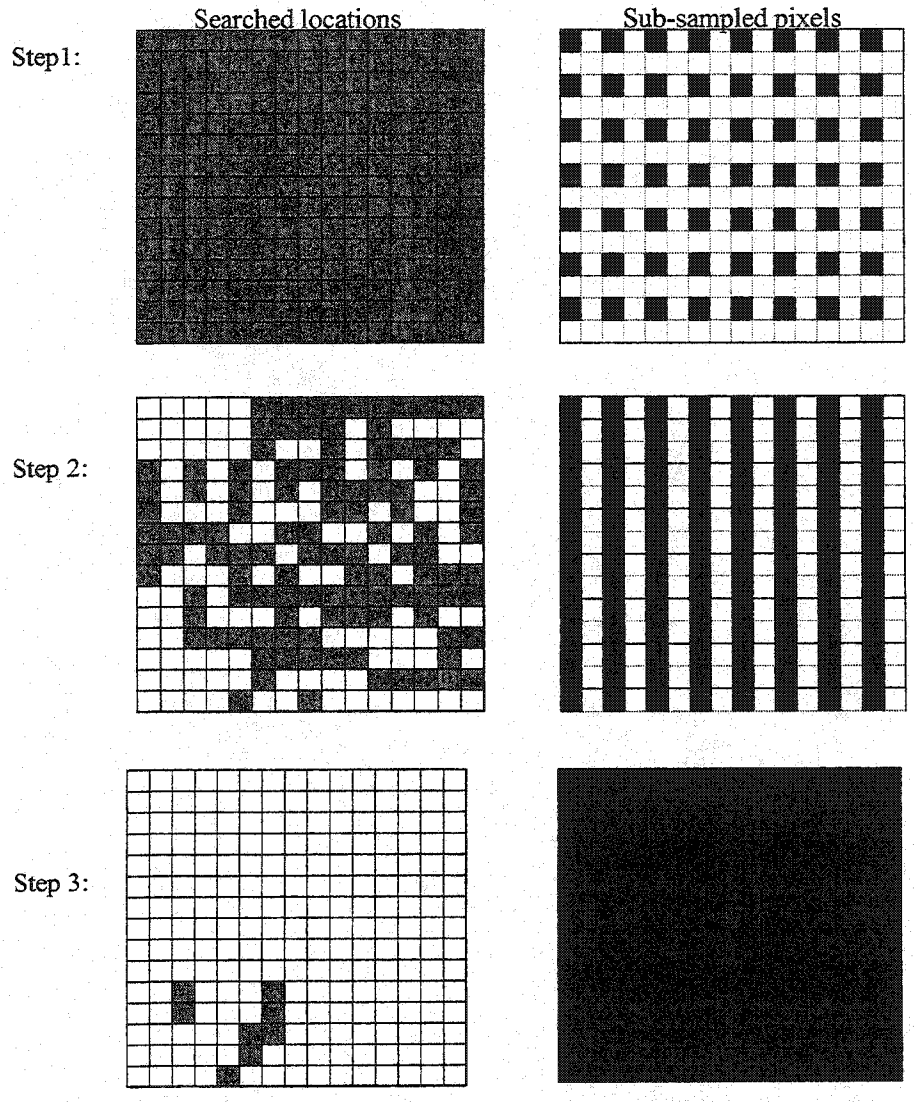


Figure 3-13 An example of hierarchical motion estimation algorithm

□Advantages:

-All macroblock positions in the searching area are likely to be covered.

-Reduced computation cost.

-High flexibility, because the sub-sampling factor can be chosen according to the actual requirement.

□Disadvantages:

- Memory bandwidth can be increased because of subsampling and pre-stage filtering.

- Low accuracy because of the high probability of local minima when the scene contains a lot of spatial details (depends on subsampling factors).

3.8.11 Spatially Dependent Algorithms

Spatial dependency checks the correlation between the motion vectors of neighboring blocks to provide a prediction to the previous matching algorithms. Frequently the prediction is formed by taking a weighted average of the neighboring motion vectors. A typical block has eight immediate neighbors. Depending on the order in which target blocks are matched, the motion vectors for some of these neighbors may not be available when a block is being matched. In Reference [22], it was suggested to calculate the spatially dependent motion vector from neighboring motion vectors. As an example let MV1, MV2 and MV3 be the motion vectors of the neighboring blocks. To estimate motion vector of the current block MVc, the motion vectors of the neighboring Block (MV1 MV2 MV3) are examined to find a proper group out of five groups given in Fig. 3-14. Then, the corresponding shaded block motion vectors are averaged and scaled down to obtain an estimate of MVC, which is the third candidate beside the other candidates that were driven using the previous algorithm.

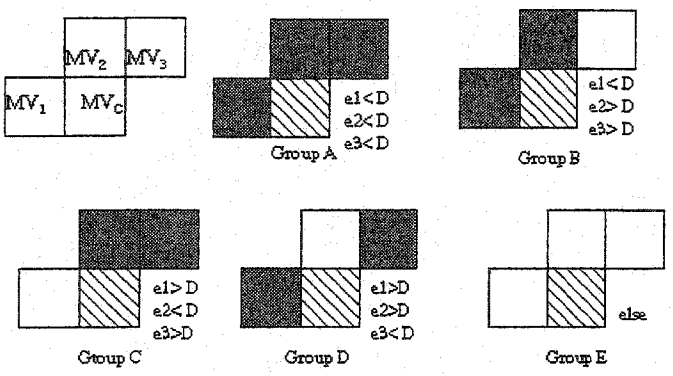


Figure. 3-14 Grouping of similar neighboring motion vectors (MVs) to predict (MVc)

Here, e1=|| MV1 - MV2 ||, e2=|| MV2 - MV3||, and e3=||MV3-MV1||, and D is threshold value to examine similarity between the two MVs.

3.9 Simulation results and conclusion:

In this chapter, the state-of-the art techniques to reduce temporal redundancies in encoding image sequences have been discussed. Advantages and disadvantages of the motion estimation algorithms have also been addressed. Here eight motion estimation algorithms, Full Search (FS), Three Step Search (TSS), Two Dimensional Logarithmic Search (TDL), Four Step Search (FSS), Orthogonal Search Algorithm (OSA), Cross search, binary search and New Three Step search (NTSS) are simulated, using the luminance components of the first 20 frames of the "Paris", "Bowing", "Deadline" and "Mad900" sequences. Each frame of 352×228 pixels in size is uniformly quantized to 8-bits per pixel. The block size is 16×16 pixels. The maximum motion displacement is ±7 in both horizontal and vertical directions.

Algorithm	Computation cost (Unit *1)	"Paris" (Unit *2)	"Bowing" (Unit *2)	"Deadline" (Unit *2)	"Mad900" (Unit *2)	"Student" (Unit *2)	"Pamphlet" (Unit *2)
Full search	225	680.18	350.40	458.66	633.55	321.48	241.57
Four step search	17--27	718.69	381.25	465.25	640.65	322.30	241.77
Three step search	25	719.62	384.72	468.82	648.61	322.83	241.77
TDL	17-19	721.50	387.13	476.66	653.84	323.65	241.77
Binary search	17-33	708.48	381.95	483.46	694.49	323.65	241.75
Orthogonal Search	13	868.95	531.69	543.88	795.92	338.66	242.30
Cross Search	17	732.04	397.17	483.31	699.65	327.65	241.91
New three step search	17-33	687.34	352.41	469.54	645.51	321.96	241.77

Table 3-1 Simulation results of motion estimation algorithms

**1 Number of locations needed to be calculated in 15*15 search area*

**2 Mean MAE values of 20 frames of sequences, the smaller values mean better result, according to our above analysis.*

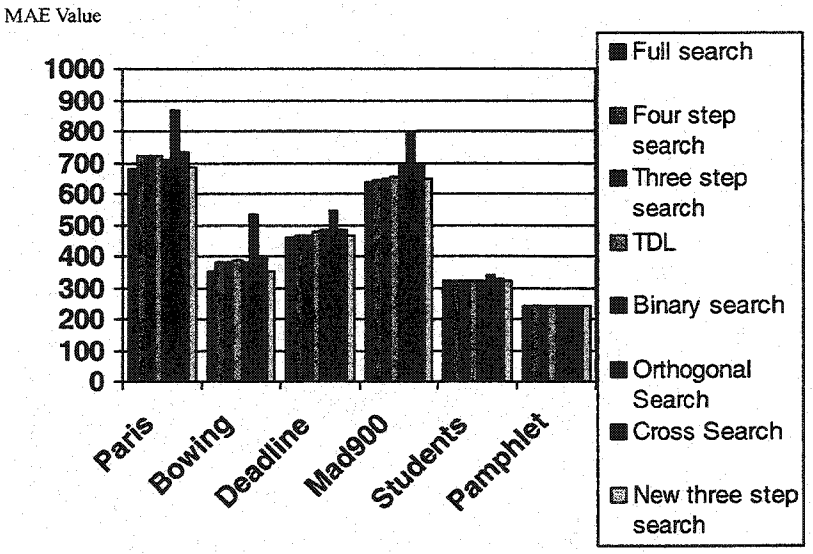


Figure 3-15 Comparison of fast ME algorithms

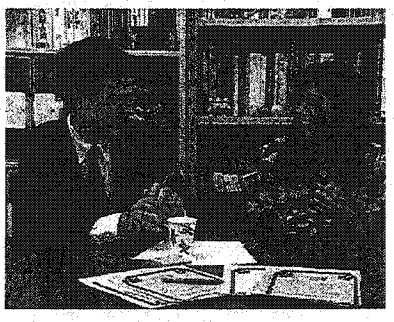
Algorithm	Computation costs * "Paris"	Computation costs * "Bowing"	Computation costs * "Deadline"	Computation costs * "Mad900	Computation costs * "Students"	Computation costs * "Pamphlet"
Full search	225	225	225	225	225	225
Four step search	17.63	18.48	17.57	18.14	17.07	17.16
Three step search	25	25	25	25	25	25
TDL	17	17	17	17	17	17
Binary search	27.71	26.94	27.73	27.16	27.96	27.95
Orthogonal Search	13	13	13	13	13	13
Cross Search	17	17	17	17	17	17
New three step search	18.08	19.16	18.10	19.52	17.24	17.38

Table 3-2 Computation costs of motion estimation algorithms

**Unit: Number of locations needed to be calculated.*

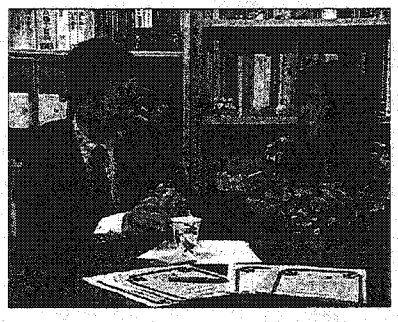
From above results (Table 3-1, Table 3-2, Figure 3-15) we can see that there is a tradeoff between the computation cost and performance of the algorithms. Their reconstructed images are shown in Figures following the chapter (Figure 3-16---Figure 3-21). In general, in terms of computation cost, the OTA is the best one. On the other hand, its quality is the worst among these algorithms. The New Three Step search and Four Step search have identical performance and computation cost. How to optimize the tradeoff to guarantee the better performance with lower computation cost is the major subject of the thesis. In the following chapters three motion estimation algorithms will be proposed and they will be compared with the Four Step search.

Full search

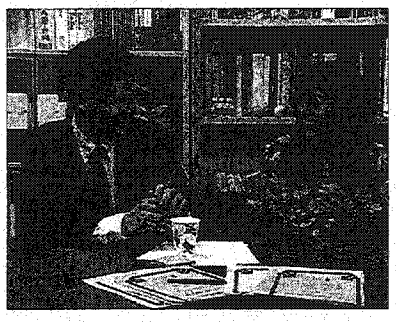


Four step search

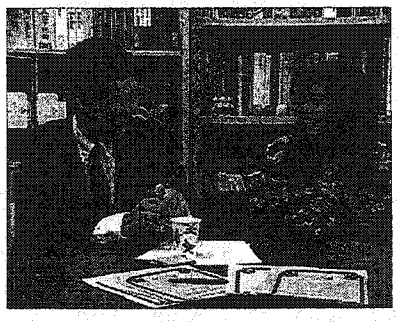
Three step search



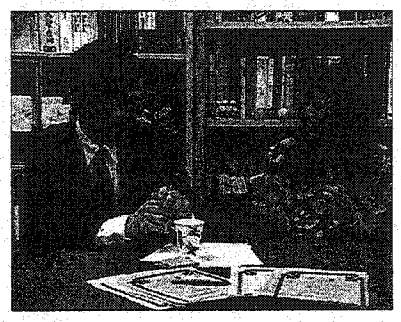
TDL search



Binary search



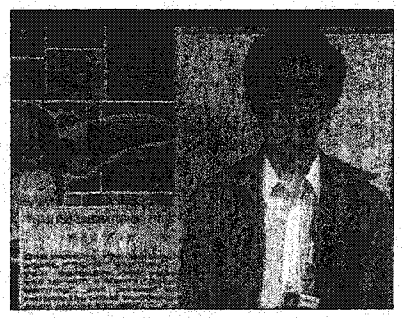
Orthogonal search



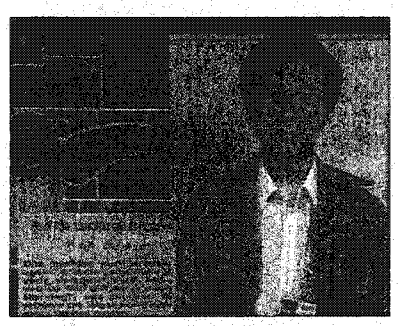
Cross Search

New three step search

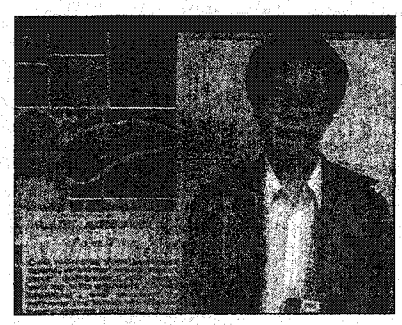
Figure 3-16 Comparison of reconstructed 'Paris' image sequence



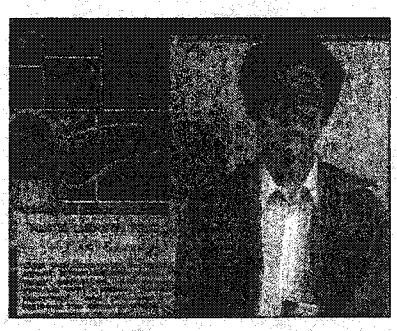
Full search



Four step search



Three step search



TDL search

Binary search

Orthogonal search

Cross Search

New three step search

Figure 3-17 Comparison of reconstructed 'Bowing' image sequence

Full search

Four step search

Three step search

TDL search

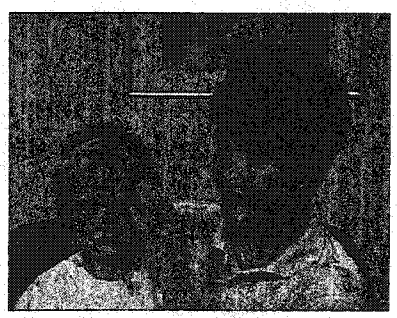
Binary search

Orthogonal search

Cross Search

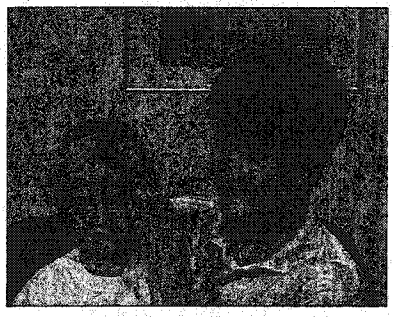
New three step search

Figure 3-18 Comparison of reconstructed 'Deadline' image sequence

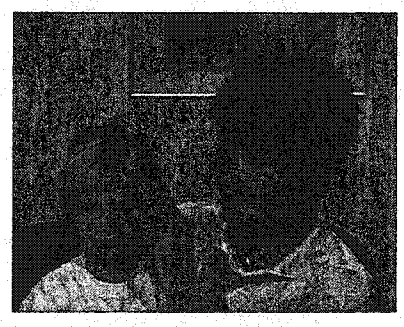


Full search

Four step search



Three step search

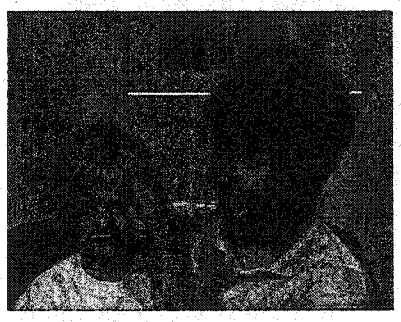


TDL search

Binary search

Orthogonal search

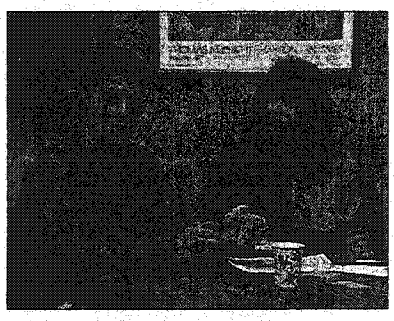
Cross Search



New three step search

Figure 3-19 Comparison of reconstructed 'Mad900' image sequence

Full search

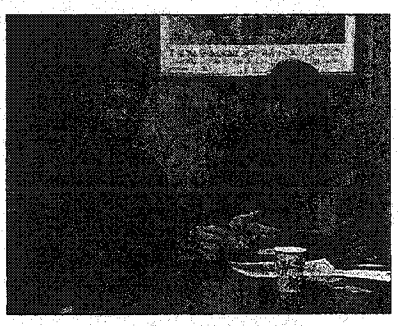


Four step search

Three step search

TDL search

Binary search

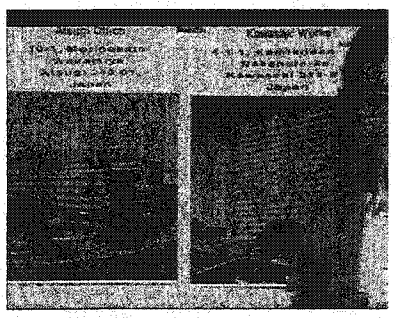


Orthogonal search

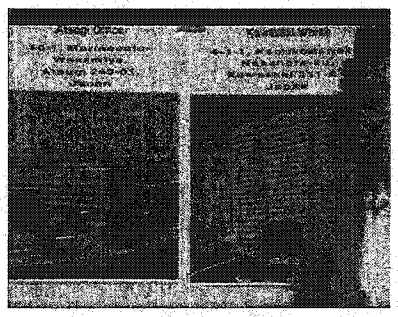
Cross Search

New three step search

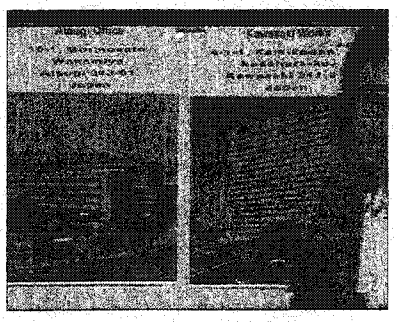
Figure 3-20 Comparison of reconstructed "Students' image sequence



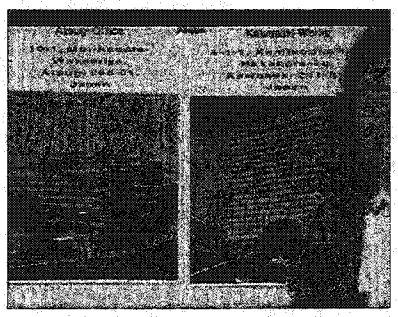
Full search



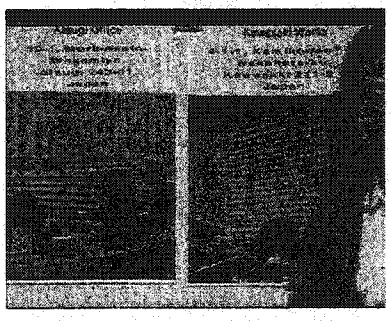
Four step search



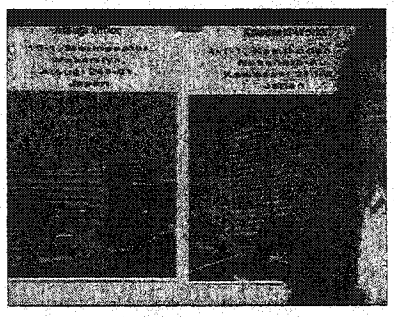
Three step search



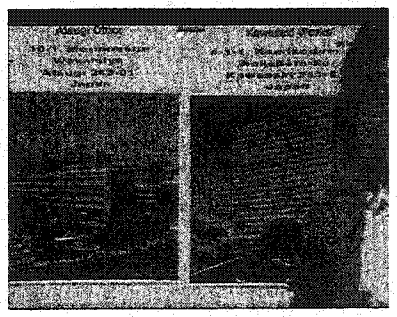
TDL search



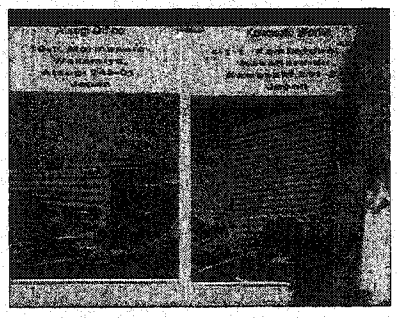
Binary search



Orthogonal search



Cross Search



New three step search

Figure 3-21 Comparison of reconstructed "Pamphlet' image sequence

Chapter 4

New algorithms for search motion estimation

4.1 Introduction

In this chapter two novel algorithms for ME are presented. It is shown that these proposed techniques perform better than Four Step Search (FSS) algorithm [15] even though their computation costs are identical. These algorithms are named, plus search method and hierarchical search ME. In the following sections these proposed techniques are explained and compared with various existing techniques.

4.2 The proposed plus search algorithm.

The plus search algorithm is a mixture of Two Dimensional Logarithmic Search algorithm [13] and the New Three Step Search algorithm [20].

4.2.1 Search strategy of plus search algorithm.

For the maximum motion displacements of ±7, the Plus Search algorithm utilizes a center-biased search pattern with 17 checking points on a plus shape search area in the first step. The center of the search window is then shifted to the point with minimum MAE. The search window size in the next step depends on the location of the minimum MAE points. If the minimum MAE point is found at the center of the search window, the search will stop. Otherwise, a search window of size 7×7 or 3×3 is chosen for next step search. In the final step, the search window is reduced to 3×3 and the search stops. The plus search algorithm is summarized as follows:

Step 1:MAE values of all of the 17 location marked with 1 should be calculated and compared to obtain the location with minimum MAE value to get a central location for next step (As

shown in Figure 4-1). If the minimum value is at the point [0,0], then the search stops. In this step 17 locations' MAE values should be calculated

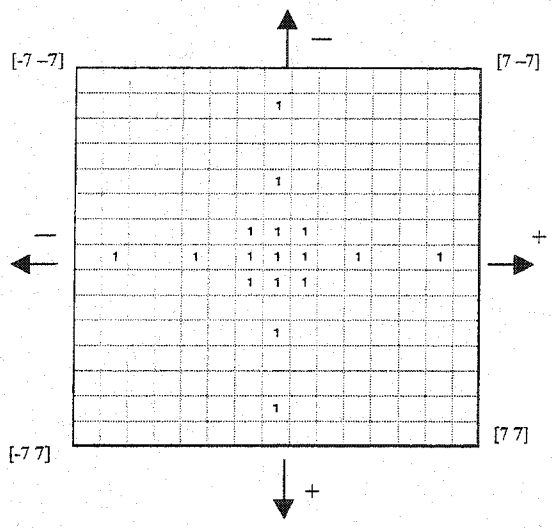


Figure 4-1 The first step search of plus search

Step 2-1: If the location with minimum MAE value is on the boundary of the 3×3 window around [0,0], extra 3×3 search window are needed to obtain the minimum point. For example, as shown in Figure 4-2, if the obtained motion vector is at the corner of the 3×3 window such as the [1,1], then 5 more unchecked locations are chosen and their respective MAE values are calculated. Also, if the obtained motion vector is located at the border of the previous search

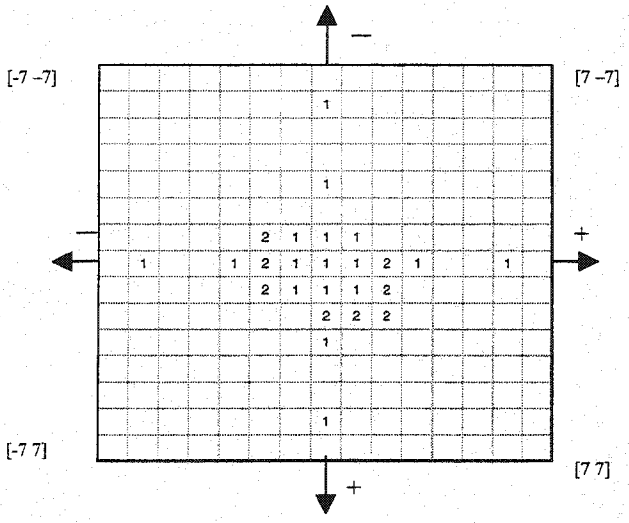


Figure 4-2 The second step search of Plus Search:case 1

window such as [-1,0], then 3 more unchecked locations are chosen and their respective MAE are calculated. The location with minimum MAE value can be regarded as the final result. Up to this step 20 or 22 MAE values should have been calculated.

Step 2-2: If the location with minimum MAE value is not within the 3×3 window around [0,0], a 7×7 search window is needed to obtain the location with minimum MAE. For example, as shown in Figure 4-3, if the obtained motion vector is at the [3,0] or [6,0] , the four locations marked 2 are selected and their MAE value are calculated . At this step 21 points are chosen and their MAE values should be calculated

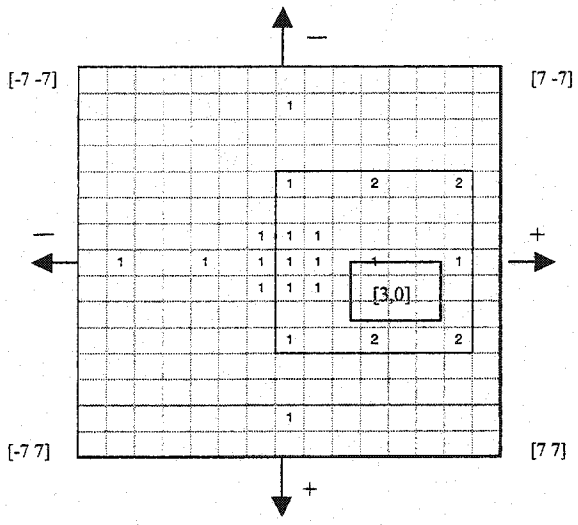


Figure 4-3 The second step search of Plus Search: case 2

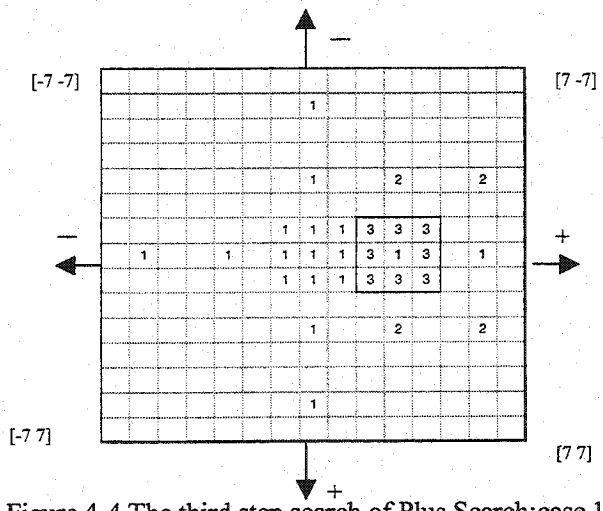


Figure 4-4 The third step search of Plus Search:case 1

Step 3-1: If the location with minimum MAE value is still the same as last step, as in Figure 4-4, then one 3×3 search window will be utilized to obtain the final result. At this step 29 MAE values are calculated

Step 3-2: If the location with minimum MAE value is at one of the locations marked 2, as shown in Figure 4-5, then another 7×7 search window will be used to obtain the candidate location that contains the minimum MAE value for next step search, which means 2 more locations [The locations marked 3] need to be checked.

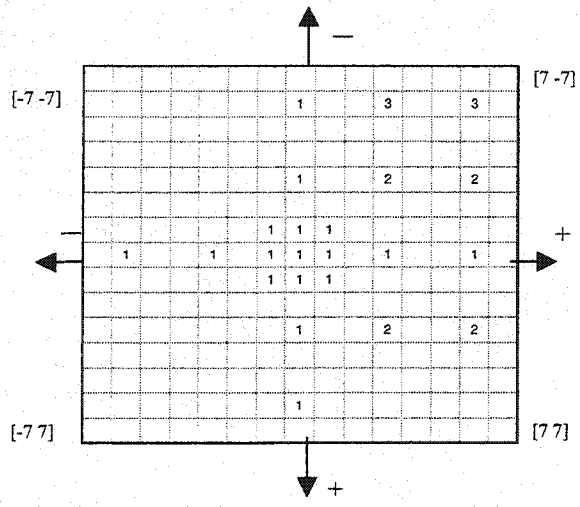


Figure 4-5 The third step search of Plus Search:case 2

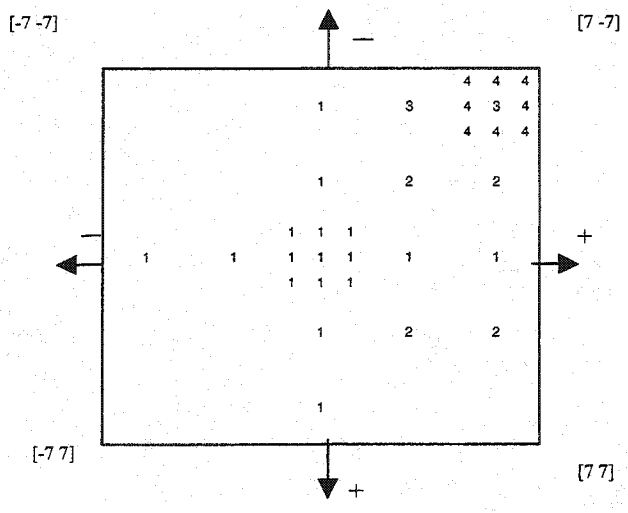


Figure 4-6 The fourth step search of Plus Search

Step 4: A 3×3 search window around the candidate location will be utilized to obtain the final result of the whole search. In this case 31 MAE values should be calculated .As shown in Figure 4-6:

4.2.2 Accuracy of Plus Search

The proposed algorithm is simulated using the luminance components of the first 100 frames of the "Pamphlet", "Paris", "students", "Bowing", "Deadline" and "Mad900" sequences as shown in appendix 1. Each frame of 352×228 pixels in size is uniformly quantized to 8-bits per pixel. The block size is 16×16 pixels. The maximum motion displacement is ±7 in both

Table 4-1 Comparison of NPS MAE value and FSS MAE value

Algorithm	Computation needed *1	Accuracy: (MAE) *2	Accuracy: (MAE) *2	Accuracy (MAE) *2	Accuracy: (MAE) *2	Accuracy: (MAE) *2	Accuracy (MAE) *2
		"Pamphlet"	"Paris"	"Students"	"Bowing"	"Deadline"	"Mad900"
Full search	225	457. 42	585. 91	376. 92	579. 42	400. 12	592. 77
Four step search	From 17--27	475. 44	592. 92	380. 13	613. 93	405. 56	601. 47
Plus Search	From 17-30	465. 86	589. 29	377. 51	589. 03	401. 57	596. 02

**1 Number of locations needed to be calculated.*

**2 Mean MAE value of 100 frames of sequences, the smaller value means better result, according to our above analysis.*

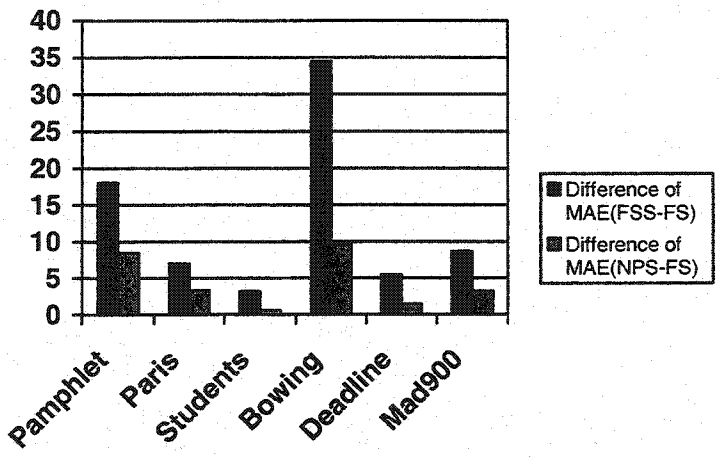


Figure 4-7 Diagram of comparison of FSS's MAE value and NPS's MAE value

horizontal and vertical directions. The proposed algorithm is simulated and compared with FS and 4SS. From the results shown in Table 4-1 and Figure 4-7, it is seen the Plus Search yields better result than FSS.

The major advantage of the plus search algorithm is that it cover larger area than other algorithms in the first step search, and has a better chance of avoiding a local minimum than other methods.

4.2.3 Computation cost of NPS

Since the proposed algorithm can reach its objectives at any of the steps which leads to the termination of the algorithm. The computation cost is from 17 to 31 MAE calculations.

1. If the final result is obtained at the first step search, 17 locations for checking are needed.
2. If the final result is obtained at the second step search, 20 or 22 locations for checking are needed.
3. If the final result is obtained at the third step search, 29 locations for checking are needed.
4. If the final result is obtained at the fourth step search, 31 locations for checking are needed.

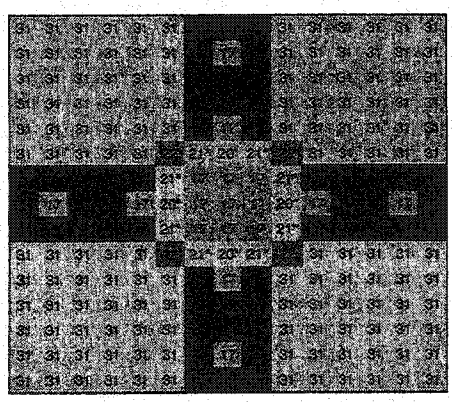


Figure 4-8 Various calculations needed in different locations for NPS.

** Calculation cost for the locations marked * varies from 20 or 22 to 29.*

As shown in Figure 4-8, for different motion vector various numbers of locations for checking are needed. The simulation result show that the computation cost is very close to Four Step

Search (As shown in Table 4-2).

Algorithm	Computation costs * "Paris"	Computation costs * "Bowing"	Computation costs * "Deadline"	Computation costs * "Mad900	Computation costs * "Students"	Computation costs * "Pamphlet"
Full search	225	225	225	225	225	225
Four step search	17.63	18.48	17.57	18.14	17.41	18.24
Plus search	18.13	19.09	18.01	19.5	17.92	18.95

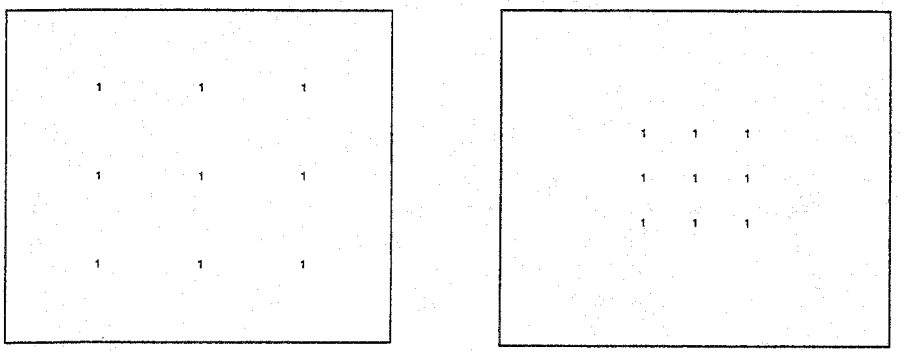
Table 4-2 Comparison of computation cost of Plus search, FSS and FS

**Unit: Number of the locations should be checked.*

4-3 Novel hierarchical search motion estimation.

Among those selected location algorithms such as Three Step Search [12], Two Dimensional Logarithmic Search [13], New Three Step Search [20], Four Step Search [15] and Binary Search [14], some algorithms perform better in searching large motions while others perform better in searching small motions. The performances of these algorithms depend on the step size, especially in the first step search.

In Figure 4-9, let us check the locations in the first step of TSS and FSS. If there is large motion, the TSS can obtain better result because its step size is bigger than the FSS. So large size window is suitable for large motion, but for small motion it is difficult to obtain exact result because of the longer distance between selected locations. Obviously if the number of searched locations in the first step is increased, the accuracy can be increased because of the larger coverage, but the computation cost is also increased. For example if we increase the window from 5×5 to 7×7 while the step size is kept 2, the number of locations for checking will be increased from 9 to 25, which is almost 3 times more than the original one. Therefore, how to select the locations efficiently is of main concern in ME field of research.



Selected locations in the first step of TSS Selected locations in the first step of FSS

Figure 4-9 Comparison of different step size.

Most of the full search matching for selected locations inside the search area are assuming that the mean absolute error (MAE) decreases monotonically as the motion vector gets closer to the global minimum. The assumption is used in methods such as 2D-logarithmic search, cross search, three-step search, four step search. The MAE between the reference block and search block using a full search algorithm with regular characteristic is presented in Figure 4-10. It is easy to see that this figure contains one global minimum. The derivation of optimal motion vector is therefore easy.

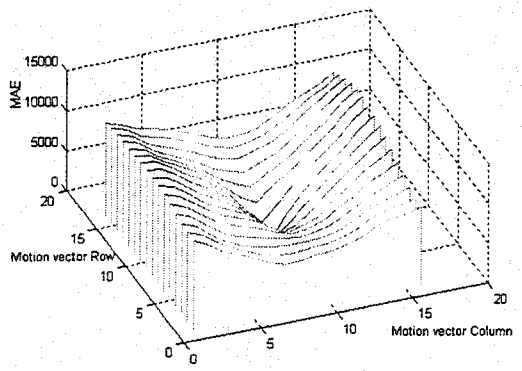


Figure 4-10. MAE between the reference block and search block with regular characteristic

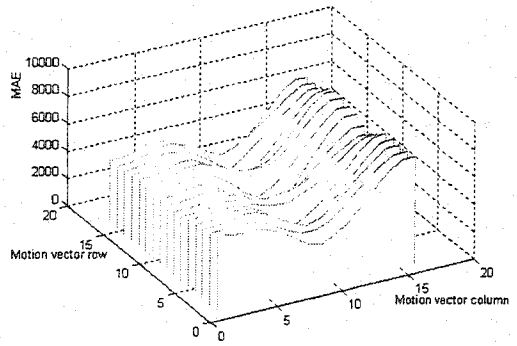


Figure 4-11. MAE between the reference block and search block with irregular characteristic

Unfortunately as is shown in Figure 4-11 this assumption is not always true and as a result obtaining a global minimum MAE is not normally feasible. The MAE surface shown in Figure 4-11 has a significant impact on the performance of the search algorithms as it contains many local minima. Since all conventional algorithms have explicitly or implicitly assumed that the error surface is unimodal over the search window, it is unlikely that they could converge to the global minimum. For the surface in Figure 4-11, the possibility of finding the global minimum position is extremely reduced since any of the local positions will correspond to a satisfactory prediction block, as the prediction error is uniformly small. Increasing the number of candidates for each steps of MAE calculation may alleviate this problem. Obviously the computation cost is therefore increased with the addition of the number of candidate locations.

The major idea of hierarchical motion estimation is to utilize the sub-sampling ratio in the earlier steps, then choose the locations with smaller MAE values as candidates for the next step search. As a result all of the candidates are chosen intentionally, at the same time all of the locations in the search area have been checked to avoid local minima. But there are still some drawbacks with conventional hierarchical algorithms.

1. All of the locations in the search area need to be checked in the first step search. This requires all of the pixel data should be made available before the first step search begins. The wide bandwidth and high frequency are needed to meet the requirements of real time

operation.

2. The accuracy of the conventional hierarchical motion estimation search algorithms depends on the sub-sampling ratio. Higher sub-sampling ratio results in high accuracy with high computation cost, vice versa. Keeping the high accuracy while reducing the computation cost is the main task of this Hierarchical Search ME algorithm. In this proposed algorithm we combine a reduced search location algorithm with a hierarchical ME algorithm to guarantee low computation cost and high accuracy simultaneously.

4.3.1 The proposed Novel Hierarchical Search algorithm (NHS).

1.In the proposed technique the image block size of 16×16 pixels for search purpose is selected with the search area of ±7 pixels in horizontal and vertical direction. Therefore the search block size is 30×30 pixels. All of the pixels in the reference block are divided into 9 clusters marked from 1 to 9 as shown in Figure 4-12, while all of the pixels in the search block are divided into 9 clusters marked from 1 to 9 as shown in Figure 4-13.

1	4	7	1	4	7	1	4	7	1	4	7	1	4	7	1
2	5	8	2	5	8	2	5	8	2	5	8	2	5	8	2
3	6	9	3	6	9	3	6	9	3	6	9	3	6	9	3
1	4	7	1	4	7	1	4	7	1	4	7	1	4	7	1
2	5	8	2	5	8	2	5	8	2	5	8	2	5	8	2
3	6	9	3	6	9	3	6	9	3	6	9	3	6	9	3
1	4	7	1	4	7	1	4	7	1	4	7	1	4	7	1
2	5	8	2	5	8	2	5	8	2	5	8	2	5	8	2
3	6	9	3	6	9	3	6	9	3	6	9	3	6	9	3
1	4	7	1	4	7	1	4	7	1	4	7	1	4	7	1
2	5	8	2	5	8	2	5	8	2	5	8	2	5	8	2
3	6	9	3	6	9	3	6	9	3	6	9	3	6	9	3
1	4	7	1	4	7	1	4	7	1	4	7	1	4	7	1
2	5	8	2	5	8	2	5	8	2	5	8	2	5	8	2
3	6	9	3	6	9	3	6	9	3	6	9	3	6	9	3
1	4	7	1	4	7	1	4	7	1	4	7	1	4	7	1

Figure 4-12 Sub-sampled pixels in reference block

2. Step 1:

The proposed algorithm uses the partial distortion of the MAE as its matching criterion. This process begins with comparing the decimated block of present frame and the corresponding pixels of the block of previous frame. Assume the intensity of pixel (i, j) of the present frame is

$In(i,j)$ and the coordinate of the upper-left corner of a 16×16 searching block is (k, l). The MAE between the block (k, l) of current frame and the block $(k+x, l+y)$ of the previous frame is given by equation 4-2:

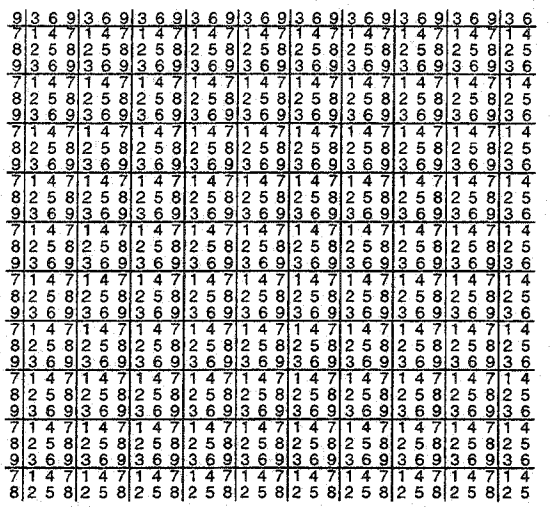


Figure 4-13 Sub-sampled pixels in search block

$$MAE_{(k,l)}(x,y)=\frac{1}{36}\sum_{i=0}^{5}\sum_{j=0}^{5}\left|I_n(k+3\times i,l+3\times j)-I_{n-1}(k+3\times x+3\times i,l+3\times y+3\times j)\right| \qquad 4\text{-}2$$

The 25 locations inside the search area should be processed using a 256:36 decimation factor as shown in Figure 4-14 instead of 225 locations for the conventional hierarchical algorithm. We compute the MAE with all of the pixels marked 1 with the corresponding pixels in the search and reference blocks as shown in Figures 4-12 and 4-13. The 25 locations and their neighboring points can cover every location in the whole searching area. Because the number of search locations is much smaller than conventional hierarchical algorithms, the sub-sampling factor can be set large to improve the accuracy, while the total computation cost can be decreased. Then we choose the 4 locations with minimum MAE values as the candidates for next step search. For this hierarchical search algorithm, when all of the 36 pixels marked 1 in the reference block and 100 pixels marked 1 in the search block are available, the calculation can begin, in this way the waiting time can be reduced.

3. Step 2:

This algorithm is based on this assumption that MAE values are smooth and continuous; therefore, there is better chance for the location with minimum MAE value being focused around the 4 locations with smaller MAE values. With the same sub-sampling factor, we calculate the 32 points' MAE values (Figure 4-15) around the 4 locations shown with * and choose 9

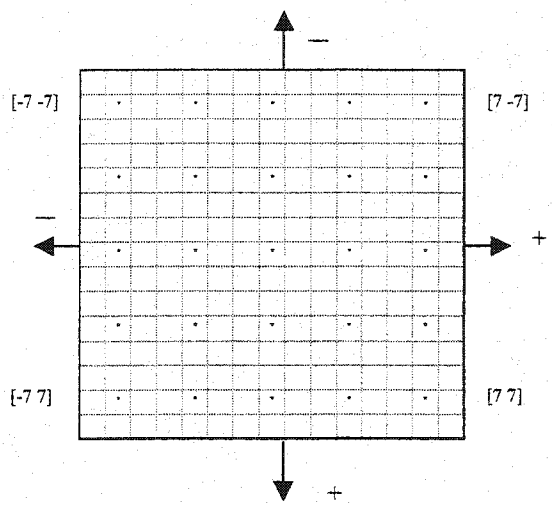


Figure 4-14 Searched locations in first step search

locations with minimum MAE values (as shown in Figure 4-16) as candidates for next step search.

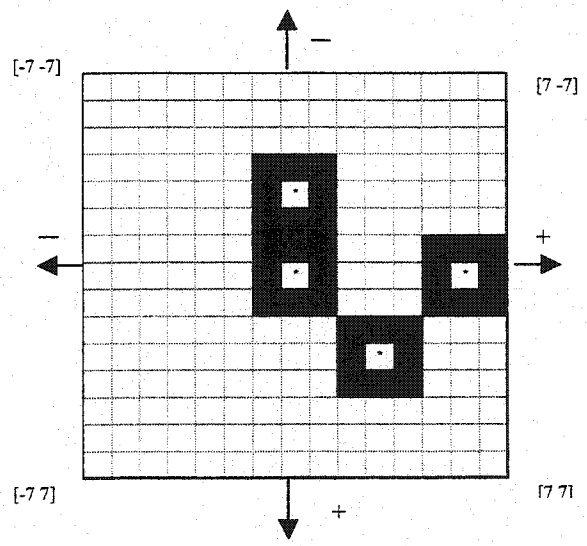


Figure 4-15 The searched locations in the second step search

3. Calculate the full MAE values in the 9 locations (Figure 4-16) that are obtained in the above step. The location with the minimum value is the motion vector.

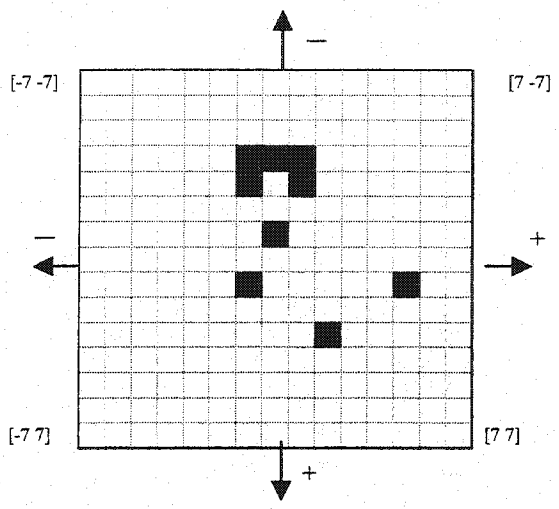


Figure 4-16 The searched locations in the third step search

4.3.2 Analysis of proposed hierarchical search

Differing from conventional hierarchical motion estimation algorithms, the proposed hierarchical search algorithm only checks part of the locations in the search area, since the 25 locations and their neighbors can cover the whole search area. In this way not only the computation cost is reduced but also the period of waiting for input of the data is reduced dramatically. For conventional hierarchical ME algorithm, only after 100 pixels data in the reference block and 900 pixels data in the search block are obtained the search can start; while for the proposed hierarchical search algorithm, after the input of the 36 pixels of the reference block and 100 pixels of the search block are made available the computation can start, in this way the bottleneck of data input can be alleviated. This is because in the first step search the sub-sampling ratio of pixels and sub-sampling ratio of location are the same. As shown in Figure 4-15, when the 4 locations in the first step search have been selected, one may find the final motion vector from the four 3×3 windows around the 4 locations, therefore the selected search area shown in Figure 4-17 are enough to accomplish the process instead of all of the

pixels data in the search area.

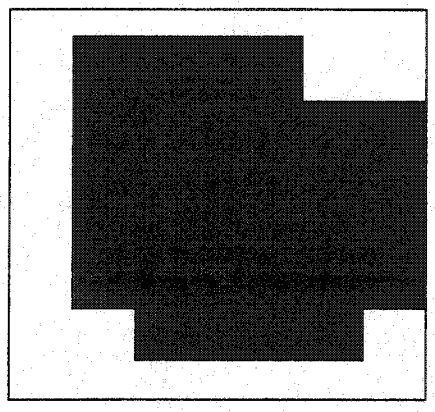


Figure 4-17 Selected pixels in search area

4.3.3 Simulation result:

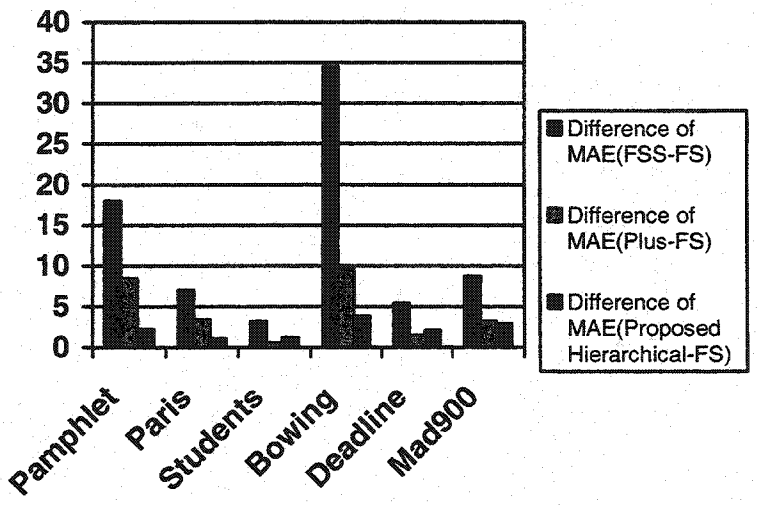


Figure 4-18 Comparison of FSS, Plus Search, New Hierarchical Search.

Algorithm	Computation needed *1	Accuracy: (MAE) *2	Accuracy: (MAE) *2	Accuracy: (MAE) *2	Accuracy: (MAE) *2	Accuracy: (MAE) *2	Accuracy: (MAE) *2
		"Pamphlet"	"Paris"	"Students"	"Bowing"	"Deadline"	"Mad900
Full search	225	457. 42	585. 91	376. 92	579. 42	400. 12	592. 77
Four step search	From 17–27	475. 44	592. 92	380. 13	613. 93	405. 56	601. 47
Plus search	From 17-30	465. 86	589. 29	377. 51	589. 03	401. 57	596. 02
Hierarchical Search	17	459. 62	586. 98	378. 11	583. 28	402. 20	595. 65

Table 4-3 Comparison of MAE value with different algorithms

**1 Number of locations needed to be calculated.*

**2 Mean MAE value of 100 frames of sequences, the smaller value means better result, according to our above analysis.*

The proposed algorithm is simulated using the luminance components of the first 100 frames of the "Pamphlet", "Paris", "students", "Bowing", "Deadline" and "Mad900" sequences. Their reconstructed images are shown in figures following the chapter (Figure 4-19—Figure 4-24). Each frame of 352×228 pixels in size is uniformly quantized to 8-bits per pixel. The block size is 16×16 pixels. The maximum motion displacement is ±7 in both horizontal and vertical directions. From the Table 4-3 and Figure 4-18 it is seen that Hierarchical Search yield results comparable with Full search technique while its computation cost is lowest.

4.3.4 Computation cost of NHS

STEP	Number of Searched locations	Sub-sampling ratio	Computation cost*
First	25	36/256	25*36/256
Sceond	32	36/256	32*36/256
Third	9	1	9
Total			17

Table 4-4 Computation cost in different step search in hierarchical search

**Unit: Number of locations to be checked Assume the size of the reference block is 16×16, then the calculation of each location's MAE value includes 256*

additions and subtractions and comparisons.

From the Table 4-4 one can derive the computational burden of each step of the NHS technique. For the conventional hierarchical ME, with the sub-sampling ratio of 36:256, 9 candidates are needed in the first step search. In the second step search, the 9 locations' MAE values will be calculated and compared without sub-sampling to obtain the motion vector. Then the computation cost can be calculated as below:

STEP	Number of Searched locations	Sub-sampling ratio	Computation cost*
First	225	36/256	225*36/256
Sceond	9	1	9
Total			40. 6

Table 4-5 Computation cost in different step search in conventional hierarchical search

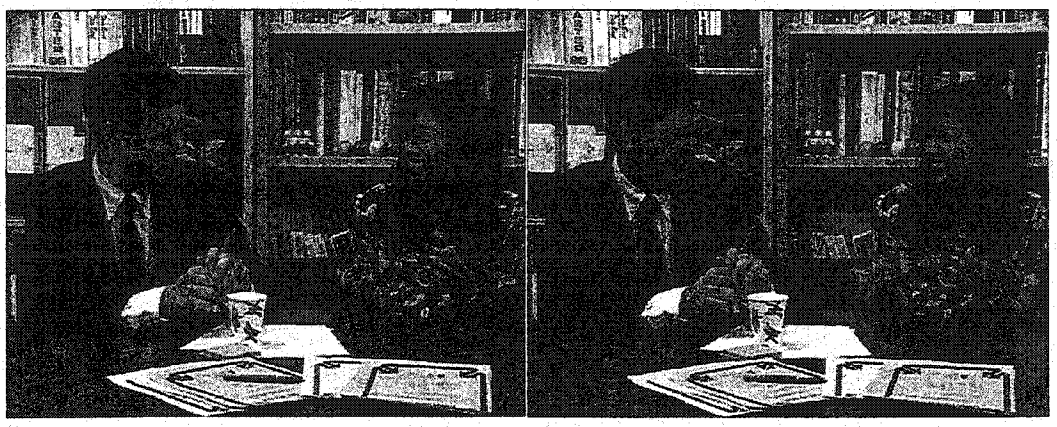
** Unit: Number of locations to be checked Assume the size of the reference block is 16×16, then the calculation of each location's MAE value includes 256 additions and subtractions and comparisons.*

We can see the conventional hierarchical ME search algorithm's computation cost is much larger than the proposed hierarchical ME search algorithm.

4.4 Conclusion

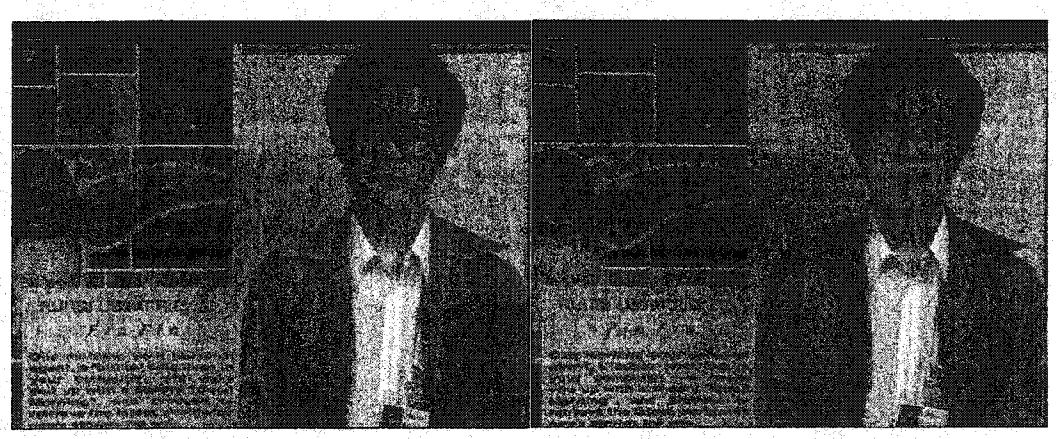
In this chapter two new ME algorithms are proposed. The simulation results show the algorithms have better performance in comparison with other algorithms with similar computation cost. To further reduce the computation costs, a new hierarchical ME algorithm will be introduced in the next chapter. In this algorithm a low power CMOS design methodology and Hierarchical ME algorithm are combined successfully to further reduce the energy consumption.

Full search | Four step search

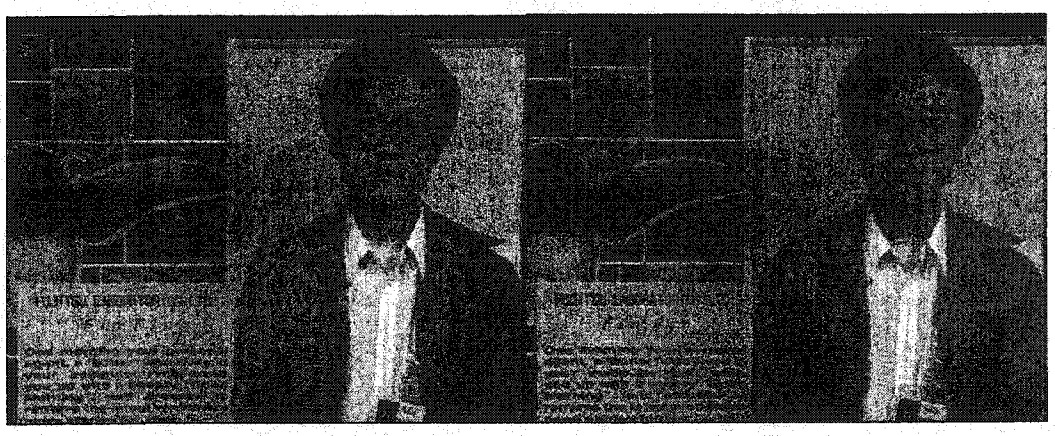


Plus search | Hierarchical search

Figure 4-19 Comparison of reconstructed 'Paris' image sequence

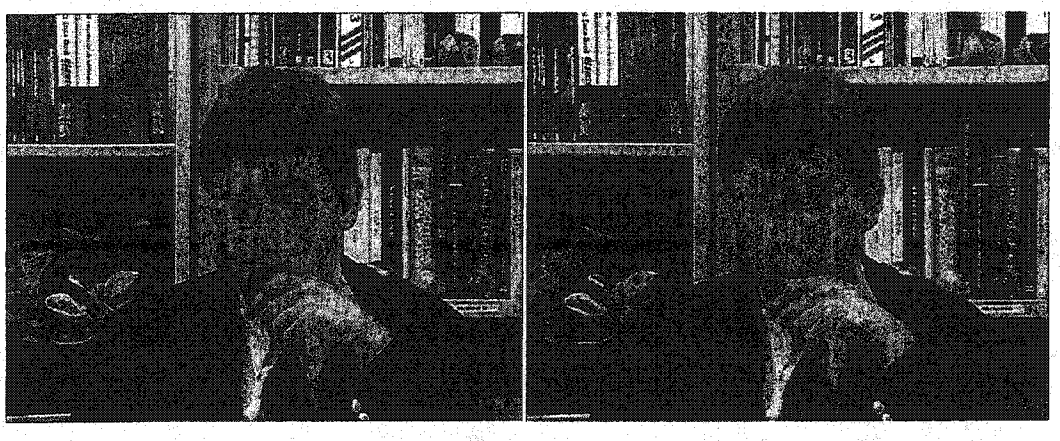


Full search | Four step search



Plus search | Hierarchical search

Figure 4-20 Comparison of reconstructed 'Bowing' image sequence

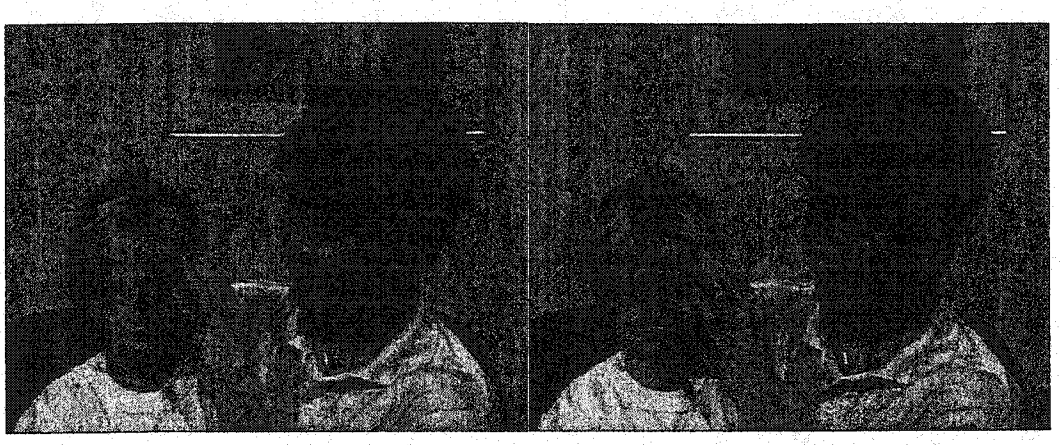


Full search Four step search

Plus search Hierarchical search

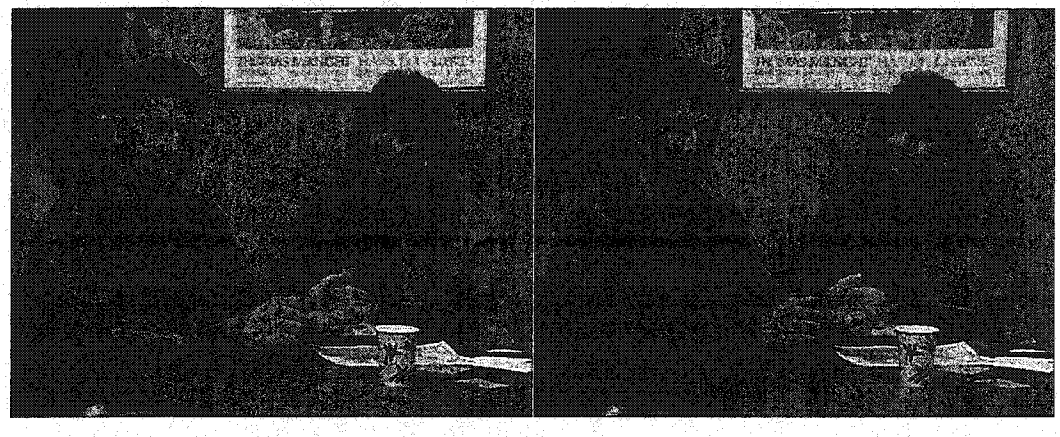
Figure 4-21 Comparison of reconstructed 'Deadline' image sequence

Full search | Four step search



Plus search | Hierarchical search

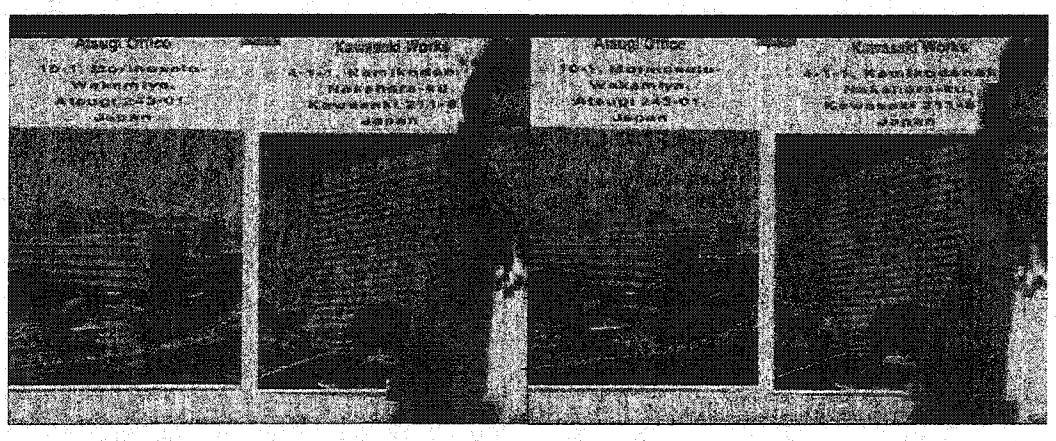
Figure 4-22 Comparison of reconstructed 'Mad900' image sequence



Full search | Four step search

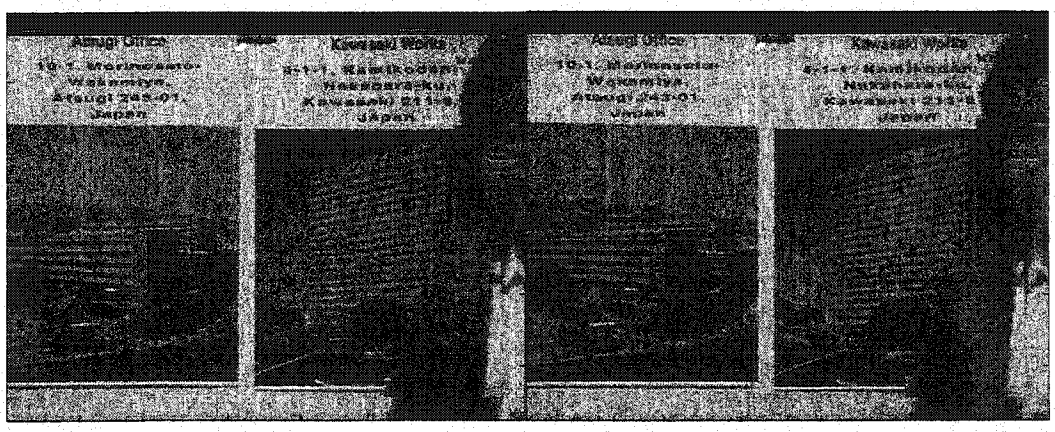
Plus search | Hierarchical search

Figure 4-23 Comparison of reconstructed 'Students' image sequence



Full search

Four step search



Plus search

Hierarchical search

Figure 4-24 Comparison of reconstructed 'Pamphlet' image sequence

Chapter 5

Dynamic Voltage Scaling Motion Estimation

5-1 Introduction of Dynamic Voltage Scaling.

Reduction of the power consumption of a motion estimation algorithm is an interesting topic. With the development of portable appliances and the boom of the third generation cellular phone, the requirements of applying digital video to portable devices will increase dramatically. However, the difficult problem to solve is to overcome the large energy consumption of ME algorithm and limited battery life for the MPEG applications. As was mentioned in the previous chapters the motion estimation algorithms will require more than 50 percent of the whole energy consumption of the MPEG encoder. As a result the energy reduction can come from the ME circuit without reducing the quality of the video substantially. This chapter presents a method for reduction of power consumption in ME circuit through DVS technique. A variety of power minimization methods have been applied at layout, logic, behavior, and architectural levels for a VLSI circuit realization. The dominant source of power dissipation in digital CMOS circuits is the dynamic power dissipation [24], defined as below:

$$P_{dynamic} = \sum C_L \cdot f \cdot V_{dd}^2 \qquad 5\text{-}1$$

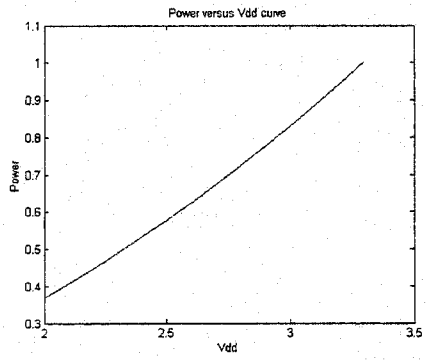


Figure 5-1 Relationship of dynamic power consumption and Vdd.

where C_L is the load capacitance, f is the working frequency, and V_{DD} is the supply voltage. While reducing voltage is the most effective way to achieve low power[Shown in Figure 5-1], this however may cause circuit delay to increase and the system throughput be reduced. The circuit delay can be estimated by the following formula [24], as shown in Figure 5-2:

$$d = C \cdot V_{DD} / (V_g - V_t)^2 \qquad 5\text{-}2$$

where d if the delay of the circuit, Vt is the threshold voltage, Vg is the voltage of the input gate, and C is a coefficient which is fixed for a specific circuit.

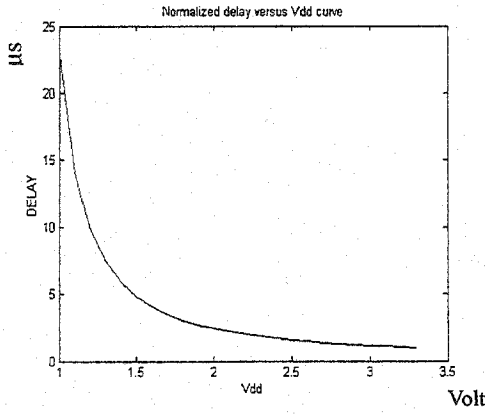


Figure 5-2 Relationship of delay and V_{dd}.

Normally it is difficult to reduce the supply voltage and guarantee the throughput of the whole circuit at the same time. There are some other low voltage design methodologies, such as the parallelism technique [24] and pipeline technique [24] which have been used widely. To compensate for the loss of the throughput because of the reduction of voltage, the parallelism methodology can be used to deal with the different serial tasks as shown in Figure 5-3, where the throughput of the whole circuit can be kept unchanged with reduced power supply, since the delay requirement to each task is increased in comparison to Figure 5-4. Unfortunately this requires the increase chip area. However the chip area is one of the most important constraints in the chip design, therefore the parallelism technique can not be utilized in many cases [24].

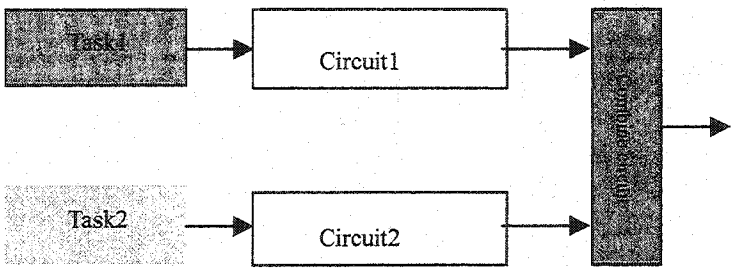


Figure 5-3 Low voltage design----Parallelism technique

Figure 5-4 Normal chip architecture

There is an alternative to apply lower voltage design without increasing the chip area, through pipelining. As shown in Figure 5-5, there are extra registers inserted between different stages to create fewer idle cycles as the register caches keep the balance of the dataflow between the two stages. In this way the working frequency can be reduced, which means the original supply voltage can be reduced. There is however extra chip area is needed for the registers [24].

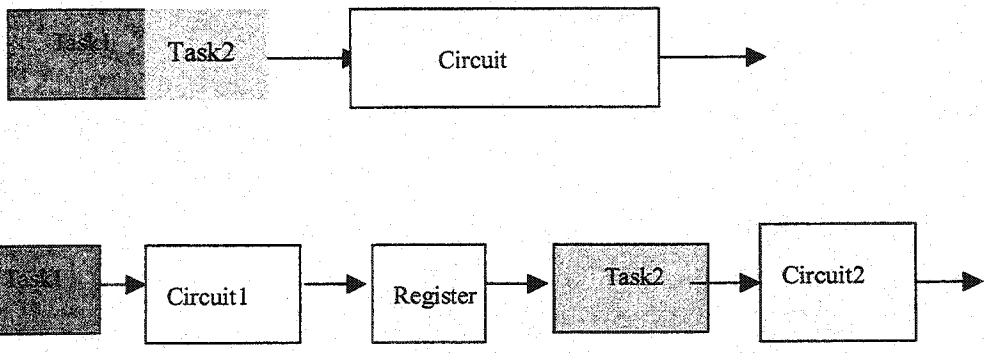


Figure 5-5 Low voltage design ----- Pipeline technique

In this thesis, we introduce a new low voltage design methodology named dynamic voltage scaling(DVS). DVS [25-28] is a new technique that allows devices to dynamically change

their working frequencies and voltages for power reduction according to its workload while maintaining the total throughput of the circuits. DVS tries to address the tradeoff between performance and battery life by taking into account two important characteristics of most current computer systems: (1) the peak computing rate needed is much higher than the average throughput that must be sustained; and (2) the processors are based on CMOS logic. The first characteristic effectively means that high performance is needed only for a small fraction of the time, while for the rest of the time, a low-performance, low-power processor would suffice. We can achieve the low performance by simply lowering the operating frequency of the processor when the full speed is not needed. DVS goes beyond this and scales the operating voltage of the processor along with the frequency. This is possible because static CMOS logic, used in the vast majority of chips today, has a voltage-dependent maximum operating frequency, so when used at a reduced frequency, the processor can operate at a lower supply voltage. Since the energy dissipated per cycle with CMOS circuitry scales quadratically to the supply voltage [28], DVS can potentially provide a very large net energy savings through frequency and voltage scaling.

The DVS seems a promising method to solve the MPEG chip energy consumption without affecting the performance, however when and how to adjust the voltage are the major problems with the DVS algorithm. A voltage scheduler analyses the state of the system and determines the optimal target voltage. One possible technique for the voltage scheduler is to use task deadlines, a concept common in real-time operating system, to determine the extent with which a task can be lengthened, so that the scheduler can decide the reduced frequency and voltage. Obviously the voltage should be adjusted before the coming tasks, if the workload of the coming task can be anticipated according to the past and present workload. But according to some other' studies [28], there are many problems in implementing DVS in the MPEG field. The variability of MPEG workload makes it difficult to anticipate future workload: the algorithm achieves a maximum 24% reduction in MPEG appliances, while in the case of audio field maximum 82% energy can be saved[28]. Here we would like to

propose a DVS hierarchical motion estimation algorithm, in this algorithm the implementation of DVS will affect the performance of the whole circuit in a minor way. Considering the continuity of video sequence, it is possible to anticipate the coming motion when there are small motions in a frame. So our major idea in this chapter is to estimate when the objects in a frame are still or moving slowly.

5-2. Estimation of slow motion.

If less than 10% the last frame's motion vectors are not in the 3×3 window around the center motion vector [0,0], we would regard this frame as an inert frame, then DVS algorithm can be implemented to reduce the consumption of energy further. For example in a video sequence, the mode of the video is Common Intermediate Format (CIF) format which means the pixels in a frame is 352×288. Then a frame can be divided into 22×16=352 blocks with size of 16×16 pixels, which will be used in block matching motion estimation. There is a motion vector to each block. If the motion vector is in the 3×3 window around original location [0,0], the block is named an inert block. If after the procedure of last frame's motion estimation more than 90 percent of blocks in the last frame are inert blocks as shown in Figure 5-6, the frame can be defined as an inert frame, then we have much more chance to deduce the motion vector from the last frame's motion vectors as shown in Figure 5-7. In this way we can reduce the size of the search window for the first step search, then the number of image data that should be inputted is reduced. Thereby the waiting data period is reduced. With the reduction of number of checked locations the period of whole calculation will be reduced, consequently, the total calculations will be decreased drastically. In another words, the working frequency for the whole circuit can be reduced properly without affecting the throughput, which means the working voltage can be reduced, because the deadline of the task of motion estimation for each frame is constant, since the frame rate will not change for a specific format. Here we will propose an algorithm named predictive hierarchical search (PHS) motion estimation demonstrated in the following paragraphs.

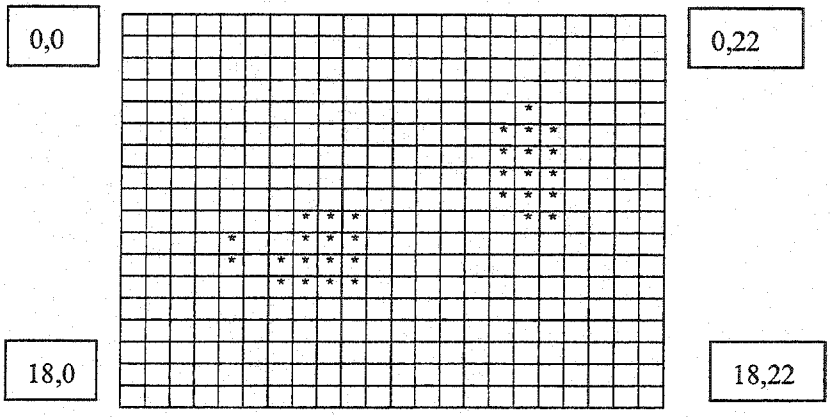


Figure 5-6 The 22×18 pieces of 16×16 blocks in a CIF frame.

*The blocks marked * means it is not inert block.*

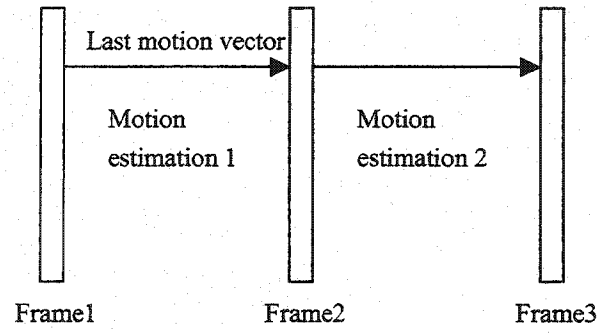


Figure 5-7 Deduce the motion vector from result of the last frame

5-3. Procedure of predictive search hierarchical motion estimation.

Step 1:

The same as proposed hierarchical search ME in last chapter, in this proposed technique the image block size of 16×16 pixels for search purpose is selected with the search area of ±7 pixels in horizontal and vertical direction. Therefore the search block size is 30×30 pixels. Here each pixel in the reference block is divided into 9 clusters marked from 1 to 9 as shown in Figure 4-12,while the search block is divided into 9 clusters marked from 1 to 9 as shown

in Figure 4-13. As shown in above Figure 5-7, we first judge which of the 25, 3×3 windows contains last motion vector, then the window will be expanded to 7×7 and the first step search of the present frame motion estimation will be processed in the window. Therefore instead of searching all 25 locations in the first step search, only 9 locations will be checked using a 256:36 decimation factor as shown in Figure 5-8. We compute the MAE values of all of the pixels marked 1 with the corresponding pixels in the search block. In the worst case the number of required input data drops from 900 to 576 as shown in Figure 5-10. Three motion vectors with minimum MAE value are obtained as candidates for next step search.

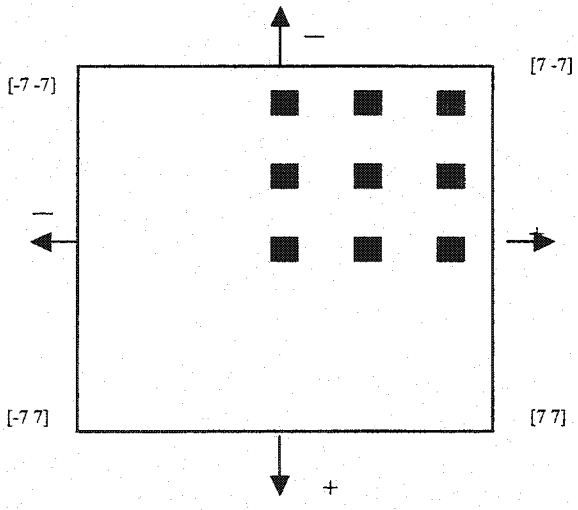


Figure 5-8 The searched locations in the first step search

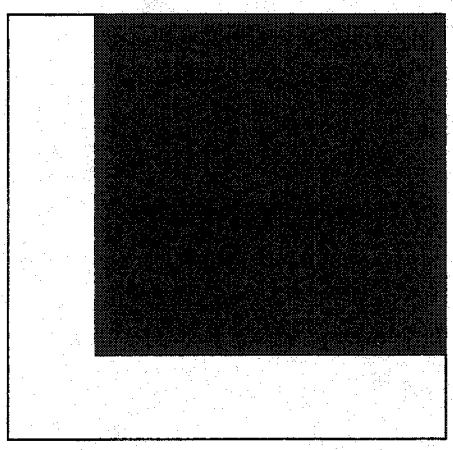


Figure 5-9 In above case the pixel data in the search area should be obtained

The following steps are similar to the proposed hierarchical search, except fewer locations are checked in each step, since the search area is reduced.

Step 2:

The 24 locations around the 3 candidate locations obtained in last step will be checked (Figure 5-10) and 6 locations with minimum MAE values are chosen as candidates for next step search, while the sub-sampling ratio is kept the same as the last step search.

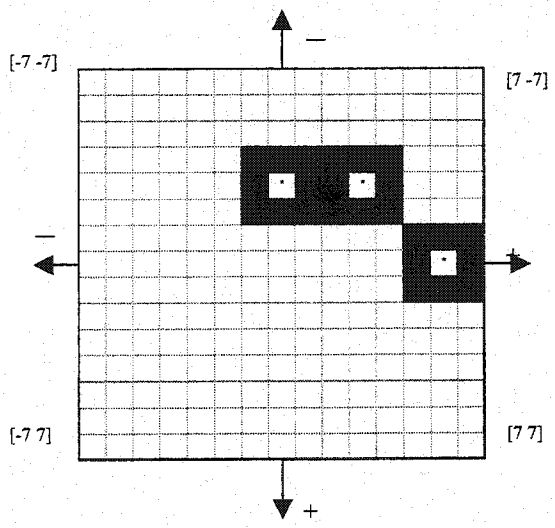


Figure 5-10 The searched locations in the second step search

Step 3:

Calculate the full MAE values of the 6 locations we obtain in above step (Figure 5-11). The location with the minimum value is the motion vector.

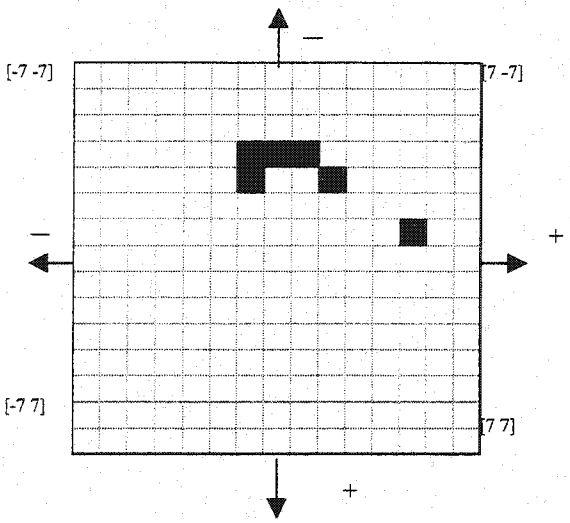


Figure 5-11 The searched locations in the third step search

5.4 Result:

The major purpose of the predictive hierarchical search (PHS) is to reduce the computation cost without affecting performance of the circuit drastically. For the PHS, the computation cost is shown in Table 5-1:

STEP	Number of Searched locations	Sub-sampling ratio	Computation cost*
First	9	36/256	9*36/256
Sceond	24	36/256	24*36/256
Third	6	1	6
Total			10.6

Table 5-1 Computation cost in different step search in Predictive Hierarchical Search

**Unit: The number of locations to be calculated. Assume the size of the reference block is 16×16, then the calculation of each location's MAE value includes 256 additions and subtractions and comparisons.*

The proposed algorithm is simulated using the luminance components of the first 100 frames of the "Pamphlet", "Paris", "students", "Bowing", "Deadline" and "Mad900" sequences the same as above two algorithms. In the following Table 5-2 we can see the computation cost of

PHS is only about half of the four step search, while the result is kept much better than four step search as shown in Figure 5-12. Their reconstructed images are shown in figures following the chapter (Figure 5-13---5-18).

Algorithm	Computation needed *1	Accuracy: (MAE) *2 "Pamphlet"	Accuracy: (MAE) *2 "Paris"	Accuracy (MAE) *2 "Students"	Accuracy: (MAE) *2 "Bowing"	Accuracy:(MAE) *2 "Deadline"	Accuracy (MAE) *2 "Mad900"
Full search	225	457. 42	585. 91	376. 92	579. 42	400. 12	592. 77
Four step search	From 17--27	475. 44	592. 92	380. 13	613. 93	405. 56	601. 47
NHS	17	459. 62	586. 98	378. 11	583. 28	402. 20	595. 65
NPS	10.6 or 17	461. 26	587. 60	378. 73	585. 73	403. 15	596. 66

Table 5-2 Comparison of PHS with other algorithms

**1 Number of locations needed to be calculated.*

**2 Mean MAE value of 100 frames of sequences,the smaller value means better result*

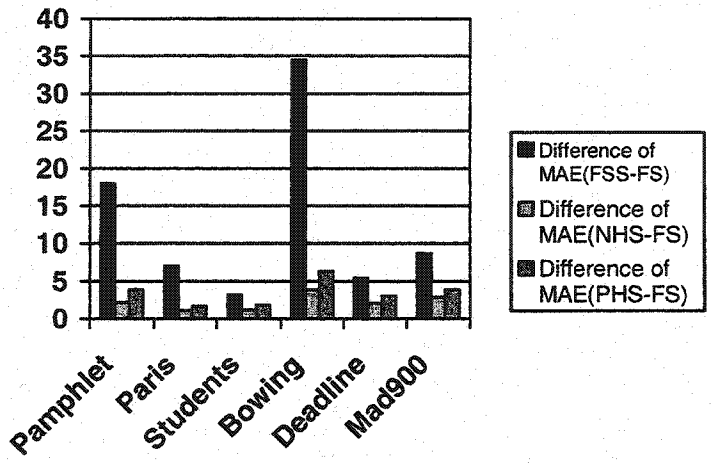


Figure 5-12 Diagram of comparison of PHS with other algorithms

The implementation of Predictive Hierarchical Search depends on the individual circuits. Here we only study the feasibility of implementation of DVS in motion estimation chip design. Since the locations are reduced as shown in Table 5-3.

At the same time as we have discussed in Figure 5-10, since the search area is decreased , the

obtained pixels in the search block is also reduced from 900 to 576, therefore, the waiting time is also reduced greatly. As a result the required working frequency can be reduced by 40%,

Algorithms *	Step1	Step2	Step3	Total
Predictive Hierarchical Search	9	24	6	39
Hierarchical Search	25	32	9	66

Table 5-3 Comparison of computation cost of predictive hierarchical search and another proposed hierarchical search

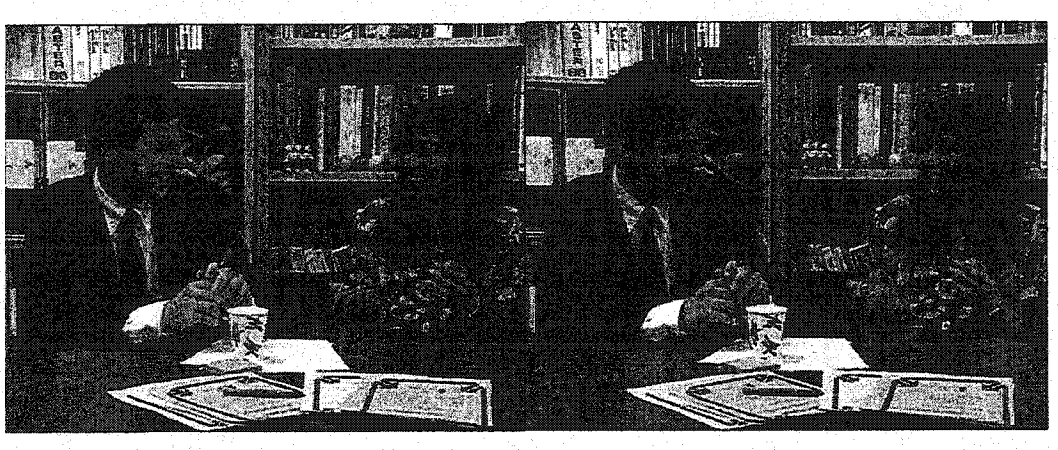
- *The unit of computation cost is the number of the locations that should be checked*

It should be mentioned that how much the voltage can be reduced depends on the architecture of the circuit.

5.5 Conclusion:

In this chapter an aggressive low power ME algorithm is proposed and discussed; the simulation results show that there is not much penalty for the performance, while the number of checked locations is reduced almost by two and it is possible to reduce the supply voltage dynamically. This algorithm can be utilized in energy-thirsty applications such as portable video camera, where the requirement to the quality of the video is not very high. It is shown that the performance of the algorithm is still better than Four Step Search [15] and some other conventional algorithms [13,16-20].

Full search | Four step search



Hierarchical search | Predictive hierarchical search

Figure 5-13 Comparison of reconstructed 'Paris' image sequence

Full search | Four step search

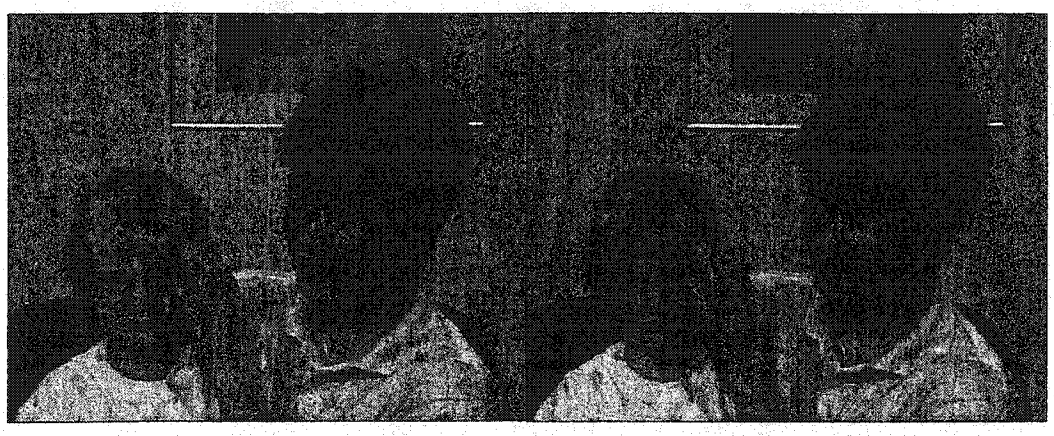
Hierarchical search | Predictive hierarchical search

Figure 5-14 Comparison of reconstructed 'Bowing' image sequence

Full search | Four step search

Hierarchical search | Predictive hierarchical search

Figure 5-15 Comparison of reconstructed 'Deadline' image sequence



Full search Four step search

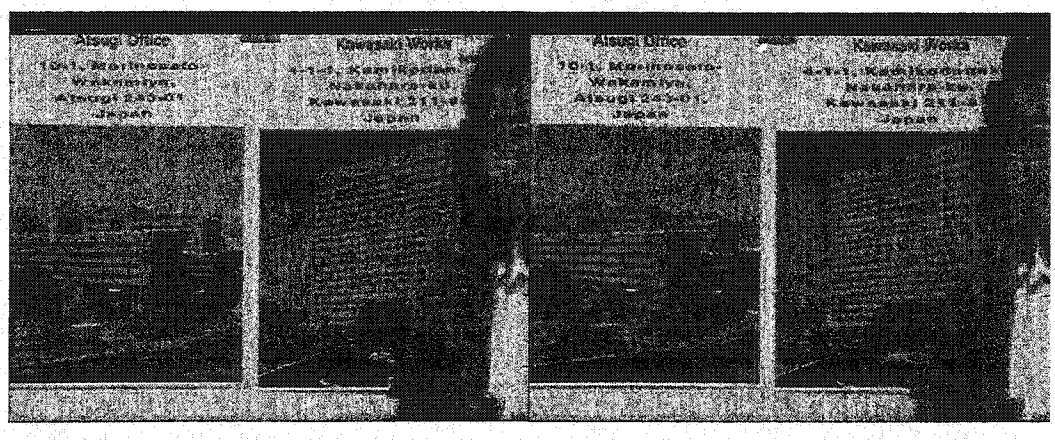
Hierarchical search Predictive hierarchical search

Figure 5-16 Comparison of reconstructed 'Mad900' image sequence

Full search Four step search

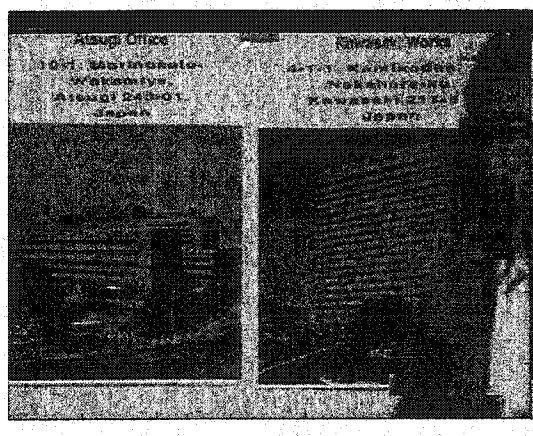
Hierarchical search Predictive hierarchical search

Figure 5-17 Comparison of reconstructed 'Students' image sequence

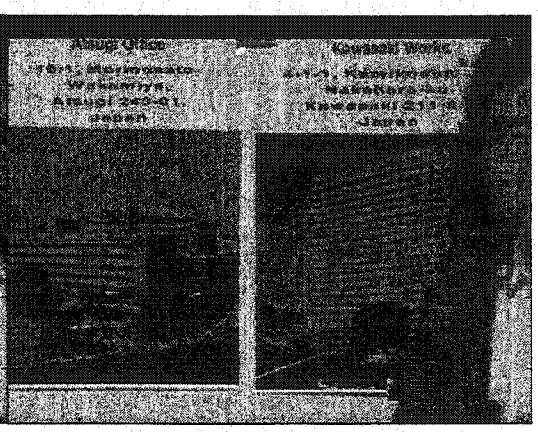


Full search

Four step search



Hierarchical search



Predictive hierarchical search

Figure 5-18 Comparison of reconstructed 'Pamphlet' image sequence

Chapter 6

Conclusion

Motion estimation and motion compensation algorithms are essential for the compression of video images. Tremendous progress has been made in last decade towards the development and realization of efficient, high performance, low power ME and MC algorithms and physical realization. This thesis therefore, presented a survey of various ME algorithms. In this survey computational burden, performance, quality of reconstructed images and their power consumption are fully studied. These techniques include Full Search (FS) [4], Three Step Search (TSS) [12], Two Dimensional Logarithmic Search (TDL) [13], Binary Search [14], Four Step Search (FSS) [15], Orthogonal Search Algorithm (OSA) [16], One at a Time Algorithm [17], Cross Search Algorithm [18], Spiral Search [19], New Three Step search(NTSS) [20] and Hierarchical Search Block Matching Algorithm [21]. Through extensive experimentation, it was proven that the full search algorithm provides the best quality of reconstructed images while Four Step Search provides the efficient solution. Through our study we are to select a single algorithm that provides the best performance, the most efficient in term of computation cost. As an example, we have found that if the motion is small, around the center of the image, such as broadcast news program in the TV, NTSS algorithm performance extremely well. However when the situation requires motions which are fast and can be at any part of image screen, such as broadcast of football, the four Step Search performs the best with excessive computation cost.

In this thesis, our attempt was to develop the ME algorithm with a tradeoff between computation cost, low power requirement by giving up slightly the quality of the reconstructed images. In this thesis we have proposed three new ME algorithms. These algorithms are based on the block search algorithm. In these algorithms the choices of the search area are delicately selected, so that the program can cover a vast area of image screen, while no potential motions are undetected. The step sizes chosen are capable of yielding fast

convergence and exit from the program at any given time once the motion is detected. These algorithms are named: Plus Search Algorithm, Novel Hierarchical Search Algorithm, Predictive Hierarchical Search Algorithm.

The experiment results (Table 4-3 and Table 5-1) show the three algorithms have much better performance than other proposed sub-optimal algorithms, such as Four Step Search [15], Three Step Search [12], etc., while the computation cost is identical to them. Additionally the accuracy of the Novel Hierarchical Search Algorithm is very close to the full search with less than one fifth computation cost. The Predictive Hierarchical Search is the modified Novel Hierarchical Search, in this way the computation cost can be reduced further with a minor change in the performance. In this algorithm Dynamic Voltage Scaling technique is used to reduce the power consumption. Considering the continuity of video sequence, there is more chance for the present frame's motion vector being the same or near the previous motion vector, especially when there are fewer motions contained in a frame. In this case the search area can be reduced. The experiment results also prove that there is little change to the performance of the algorithm after the reduction of the computation.

References:

[1] ISO/IEC 11172-2 (MPEG-1 Video), "*Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s: Video*," 1993.

[2] ISO/IEC 13818-2 | ITU-T H.262 (MPEG-2 Video),"*Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Video*," 1995.

[3] "*The MPEG-4 Video Standard Verification Model*", IEEE Trans. CSVT, Vol.7, No.1, Feb.1997

[4] A. Netravali and J.D.Robbins, *Motion compensated television coding part I.* Bell Syst. Tech. Journal, April 1979. vol. 58(no. 3): p. 629-668.

[5] P. Milanfar, "*Projection-based, frequency-domain estimation of superimposed translational motions*," *J. Opt. Soc. Amer. A,* vol. 133, pp. 2151–2161, Nov. 1996.

[6] J. Biemond,L. Looijenga, and D. Boekee, "*A pel-redursive wiener-based displacement estimation*," Signal Processing, vol.13,pp. 399-412 ,87.

[7] S. Sabri, *Movement-compensated interframe prediction for NTSC colour TV signals.* Sept. 1982, Bell Northern Res. Ltd.: Montreal Canada.

[8] C. Cafforio and F. Rocca, *The differential method for motion estimation*, in *Image Sequence Processing and Dynamic Scene Analysis*, InT.S.H. editor, Editor. 1983,Springer-Verlag: New-York. p. 104-124.

[9] D. R. Walker and K.R. Rao, *Improved pel-recursive motion compensation.* IEEE Trans. Comm., 1984. vol. COM-32, no. 10: p. 1128-1134.

[10] J. Biemond, et al., *A Wiener-based displacement estimation algorithm.* Signal Processing, Dec. 1987. vol. 13, no. 4: p. 399-412.

[11] P. Csillag and L. Boroczky *New methods to improve the accuracy of the pelrecursive Wiener-based motion estimation algorithm.* In *ICIP 1999*. Tokyo.

[12] T.Koga, et al. *Motion compensated intraframe coding for video conferencing.* In *Proc.*

NTC 81. 1981. New Orleans.

[13] J.R. Jain and A.K.Jain, *Displacement measurement and its application in interframe image coding.* IEEE Trans. Comm., 1981. COM-29: p. 1799-1808.

[14] Th. Zahariadis, D. Kalivas "*A Spiral Search Algorithm For Fast Estimation Of Block Motion Vectors" Signal Processing VIII, theories and applications*". Proceedings of the EUSIPCO 96. Eighth European Signal Processing Conference p.3 vol. lxiii + 2144, 1079-82,vol. 2.

[15] L.M.Po and W.C. Ma, *A novel four-step search algorithm for fast block motion estimation.* IEEE Trans. Circuits Syst. Video Technol., 1996. vol .6(June): p. 313-317.

[16] A. Puri, H. M. Hang, and D.L. Schilling. *An efficient motion compensated algorithm for motion compensated coding.* in *IEEE Int. Conf. Acoust., Speech, Signal Processing.* April 1987. Dallas, Texas.

[17] R.Srinivasan and K.R.Rao, *Predictive coding based on efficient motion estimation.* IEEE Trans. Comm., Sept. 1985. vol. COM-33: p. 888-896.

[18] M.Ghanbari *"The cross search algorithm for Motion Estimation*" IEEE Transactions on Communications, vol. 38, no. 7 pp 950-3, July 1990.

[19] Th. Zahariadis, D. Kalivas "*A Spiral Search Algorithm For Fast Estimation Of Block Motion Vectors*" Signal Processing VIII, theories and applications. Proceedings of the EUSIPCO 96. Eighth European Signal Processing Conference p.3 vol. lxiii + 2144, 1079-82,vol. 2.

[20] R. Li, B. Zeng, and M.L. Liou, *A new three step search algorithm for block motion estimation.* IEEE Trans. Circuits Syst. Video Technol., 1994. vol. 4 (Aug.):p. 438-442.

[21] Kwon Moon Nam, Joon-Seek Kim, Rae-Hong Park, Young Serk Shim, *A fast hierarchical motion vector estimation algorithm using mean pyramid*, IEEE Transactions on Circuits and Systems on Video Technology, Vol.5, No.4,Aug.1995, pp344-351

[22] Kyoung Won Lim, Jong Boem Ra "*Improved Hierarchical Search Block Matching Algorithm by Using Multiple Motion Vector Candidates*" Electroonic Letters, vol. 33, no.21,

pp 1771- 1772, October, 1997.

[23] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, "*A novel unrestricted center-biased diamond search algorithm for block motion estimation*", IEEE Trans. Circuits Syst. Video Technol., Vol. 8, No. 4, pp. 369-377, August 1998.

[24] T. Ishihara and H. Yasuura "*Voltage scheduling problem for dynamically variable voltage processor*". IEEE International Symposium on Low Power Electronics and Design, California, USA, 1998, pages 197-202.

[25] K. Govil, E. Chan, and H. Wassermann "*Comparing algorithms for dynamic speed-setting of a low-power CPU*". The 1st Conference on Mobile Computing and Networking (MOBICOM'95), 1995, pages 13-25.

[26] C. M. Krishna, and Y.H. Lee "*Voltage-clock-scaling techniques for low power in hard real-time systems*". IEEE Real-Time Technology and Applications Symposium, Washington D.C., USA, 2000, pages 156-165.

[27] T. Pering and R. Brodersen "*Energy efficient voltage scheduling for real-time operating systems*". The 4th IEEE Real-Time Technology and Applications Symposium RTAS'98, Work in Progress Session, Denver, USA, 1998.

[28] T. Pering and R. Brodersen " *The simulation and evaluation of dynamic voltage scaling algorithms*". IEEE International Symposium on Low Power Electronics and Design, California, USA, 1998, pages 76-81.

VITA

Name: Songtao Huang

Place of Birth: Qinglong town, HeBei province, P.R. China

Year of Birth: 14, Feb, 1973

Education:

--WuHan University. Department of Electrical Engineering. B.Sc.

P.R. of China 1989-1993