DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND
APPLIED SCIENCES

MOTION ESTIMATION
FOR VIDEO SEQUENCES

by
Hasan ALIMLI

June, 2008
IZMIR

MOTION ESTIMATION FOR VIDEO SEQUENCES

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Electrical and Electronics Engineering, Electrical and Electronics
Engineering Program

by
Hasan ALIMLI

June, 2008
IZMIR

M.Sc. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"MOTION ESTIMATION FOR VIDEO SEQUENCES"** completed by **HASAN ALIMLI** under supervision of **ASS. PROF. DR. HALDUN SARNEL** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Ass. Prof. Dr. Haldun SARNEL

Supervisor

(Jury Member) (Jury Member)

Prof. Dr. Cahit HELVACI

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to thank my advisor Ass. Prof. Dr. Haldun Sarnel and The Scientific and Technological Research Council of Turkey (TUBITAK) for their valuable guidances and supports, and I am grateful to my colleagues İlker Arslan and Anıl İkizler.

Most of all, I would like to thank to my family and my dear friend Gökçe Dündar since I am deeply indebted to their sincere support and help.

Hasan ALIMLI

MOTION ESTIMATION FOR VIDEO SEQUENCES

ABSTRACT

Motion estimation is the processes which generates motion vectors that determine how an object moves from previous frame to current frame. They are in many applications, such as video technology, computer vision, tracking and industrial designs.

In this thesis, two main approaches to the motion estimation, spatial domain research and frequency domain research, have been investigated and several methods in these categories have been implemented. The methods include all well-known block matching algorithms, differential-gradient based techniques and phase correlation based methods. I have also developed and implemented a new hierarchical phase correlation method with adaptive threshold.

Twenty pairs of pictures have been used in order to evaluate all the implemented methods. Power Signal Noise Ratio (PSNR), computational load and entropy performances of each method are calculated and compared in detail.

The motion estimation techniques is also used in 100/120 Hz TV applications which double the input frame rate using motion based frame interpolation. I have designed and implemented an application for such frame rate conversion task. My application produces new intermediate frames after finding the motion vectors of blocks between successive frames. So there are twice as much frame and consequently this provides a judder-free and fluid motion in the video sequences.

Keywords: Motion Estimation, Motion Vector, Block Matching, Optic Flow, Phase Correlation, Motion Compensation

VİDEO DİZİLERİ İÇİN HAREKET KESTİRİMİ

ÖZ

Hareket kestirimi, bir resimden bir sonraki resme gecerken bir objenin nasıl hareket ettiğini gösteren hareket vektörlerini bulma işlemidir. Video teknolojisi, bilgisayar gösterimi, takip işlemleri ve endüstriyel tasarımlar gibi birçok uygulamalarda kullanılmaktadır.

Bu tezde hareket kestirimi için uzaysal alan araştırmaları ve frekans alan araştırmaları olmak üzere iki büyük yaklaşım incelenmiş ve bu kategorilerdeki bir çok methot uygulanmıştır. Bu methotlardan başlıcaları blok karşılaştırma algoritmaları, türevlenebilir-değişim ölçüsüne dayalı teknikler ve faz korelasyon yöntemleridir. Ayrıca adaptif limitlemeye dayalı yeni bir hiyerarşik faz korelasyonu methodu geliştirdim ve uyguladım.

Tüm methotları değerlendirmek için 20 çift resim kullanılmıştır. Her yöntemin PSNR, işlem hacmi ve entropi performansı ayrıntılı olarak incelenmiş ve karşılaştırılmıştır.

Hareket kestirim teknikleri, giriş resim sayısını harekete dayalı resim üreterek iki katına çıkartan 100/120 Hz televizyon uygulamalarında da kullanılmaktadır. Tez de bu tarz bir resim oranı artırma uygulaması yaptım. Uygulama, hareket vektörlerini bularak ve kullanarak yeni ara resimler oluşturuyor. Böylece birim zamanda iki kat resim elde ediliyor ve sonuç olarak video dizilerinde akıcı ve sürekli bir hareket sağlar.

Anahtar Sözcükler: Hareket Kestirimi, Hareket Vektörü, Blok Karşılaştırma, Optik Akış, Faz Korelasyonu, Hareket Kompansazyonu

CONTENTS

Page

CHAPTER ONE

INTRODUCTION

1.1 Introduction

Motion estimation is the processes which generates the motion vectors that determine how each motion compensated prediction frame is created from the previous frame. Motion estimation techniques have been explored by many researchers in the past 30 years. They are in many applications, such as computer vision, tracking and industrial monitoring.

Block Matching is the most common method of motion estimation. The basic idea of block matching is to divide the current frame into a matrix of blocks, then these macroblocks are compared with corresponding block and its adjacent neighbors in the next frame to create a vector which shows the movement of a macro block from one location to another in the previous frame. Nowadays there are many fast search algorithms to get the best performance with lower computational load.

There are many other approaches to motion estimation. One of them is differential techniques, also called as optical flow. Optical flow cannot be computed locally and it assumes that the apparent velocity of the brightness pattern varies smoothly almost everywhere in the image.

One of the other approaches to motion estimation uses phase correlation of successive frames. Phase correlation technique is a frequency domain motion estimation method that makes use of the shift property of the Fourier transform. According to this property, a shift in the temporal domain is equivalent to a phase shift in the frequency domain.

Nowadays, the motion estimation techniques is also used in 100/120 Hz LCD TV applications. The motion vectors, which show the paths of blocks, are used to produce intermediate frame between current frame and next frame. So there are twice as much frames and this provide a very clear and fluid motion.

1.2 Literature Review

Motion estimation techniques have been explored by many researchers in the past 30 years. They are in many applications, such as computer vision, tracking and industrial monitoring. In interframe coding, for example, motion estimation and compensation can reduce the bit rate significantly. Many motion estimation schemes have been developed. They can be classified, roughly, into two groups: spatial domain search and frequency domain search. The spatial domain search stands out with two subgroups as block matching method and the differential-gradient based method. The frequency domain search focuses on the phase correlation methods.

Block matching (BM) for motion estimation has been widely adopted by current video coding standards such as H.261, H.263, MPEG-1, MPEG-2, MPEG-4 and H.264 due to its effectiveness and simplicity for implementation. BM is a very popular method and it finds the best match between the current image block and certain selected candidates in the previous frame under the assumption that the motion of pixels within the same block is uniform. The most straightforward BM is the *full search*, which exhaustively searches for the best matching block within the search window. It is firstly applied to interframe coding (Jain & Jain, 1981). Fast search algorithms are highly desired to significantly speed up the process without sacrificing the distortion seriously. Many computationally efficient variants were developed, typically among which are three-step search (Koga, Iimuna, Hirano, Iijima & Ishiguro, 1981), a new three-step search (Li, Zeng & Liou, 1994), a simple and efficient search (Lu & Liou, 1997), a four-step search (Po & Ma, 1996), a diamond search (Zhu & Ma, 2000) and an adaptive rood search pattern search (Nie & Ma, 2002) and further experiments to obtain the motion vectors.

The differential approach is developed based on the assumption that the image intensity can be viewed as an analytic function in spatial and temporal domains. It was first proposed by Cafforio and Rocca (Cafforio & Rocca, 1976) and later Netravali and Robbins developed an iterative algorithm, the so-called pel-recursive method (Netravali & Robbins, 1979). The optical flow method (Horn & Schunck, 1981) (Lucas & Kanade, 1981) in computer vision is much like the pel-recursive scheme even though they were derived from different bases. There are also many refined versions of the pel-recursive however and optical flow methods (Paquin & Dubois, 1983), (Walker & Rao, 1984), (Biemond, Looijenga, Boekee & Plompen, 1987), (Yamaguchi, 1989).

Phase correlation (Kuglin & Hines, 1975) is a well-known method in this class that utilizes phase information of frequency components in estimating the motion vectors and then it was improved (Pearson, Hines, Goldman & Kluging, 1977). Thomas (Thomas, 1987) did a rather extensive study on phase correlation and also suggested a two-stage process and a weighting function to improve this method. There are also many key researches (Strobach, 1990), (Nicolas & Labit, 1992), (Jensen & Anastassiou, 1993), (Banham & Brailean, 1994), (Lee, 1995), (Schuster & Katsaggelos, 1998). Also the work of hierarchical search was a effective reference for this thesis (Argyriou & Vlachos, 2005).

1.3 Outline

This thesis is presented in eight chapters. Chapter 1 presents this introduction and literature review. Chapter 2 introduces researches in the past and JPEG and MPEG applications. Chapter 3 shows motion estimation techniques at the spatial domain as block matching and differential–gradient based search. Chapter 4 presents the motion estimation techniques at the frequency domain. Chapter 5 presents the details of MPEG and motion based frame interpolation techniques at the 100/120 Hz LCD TVs. There are the details of the application methods in chapter 6. Chapter 7 presents the application results and chapter 8 finishes this thesis with discussing conclusion and mentioning the future works.

CHAPTER TWO

BACKGROUND

2.1 Image And Video Compression

Compared to analog communication, digital communication has many advantages as easiness and flexibility of process, immunity of noise and less error during communication. Therefore, video signals are preferred to be processed in the digital domain. But, video signals need to be transferred at high speed and rate and they need a big storage memory. It is, therefore, an important subject to compress a video signal to use less storage memory and smaller bandwidth of communication channel.

A typical video signal is subject to much data redundancy and this redundant data must be removed with the help of compressing before sending or storing. MPEG is the process that compresses a video data without an important loss or with an acceptable loss. While JPEG is a compression method for still pictures, MPEG is an advanced compression method including motion data.

The Moving Picture Experts Group, commonly referred to as simply MPEG, is a working group of ISO (International Standards Organization) charged with the development of video and audio encoding standards. Its first meeting was in May of 1988 in Ottawa, Canada.

2.1.1 JPEG

Discrete cosine transform (DCT), quantization and variable length coding (VLC) are base features for JPEG and MPEG.

1) After 2D analog signal is converted to digital form, this digital signal is splitted to blocks of 8x8 pixels. Then, an offset value is subtracted from the signal. This is 128 for 8-bit video signal.

2) This digital block is converted into the frequency domain by using DCT.

$$F(u,v) = \frac{c(u).c(v)}{4} \sum_{m=0}^{7} \sum_{n=0}^{7} f(m,n).\cos\left(\frac{2m+1}{16} u\pi\right) \cos\left(\frac{2n+1}{16} v\pi\right) \qquad \text{Equation 2.1}$$

where f(m, n) is a digital signal

F(u, v) is the frequency spectrum of input

c(0)=0.707 and c(k)=1 k=1, 2,…

In that result, the upper leftmost value is referred as DC value and the others are referred as AC values.

3) The human eye is good at seeing small differences in brightness over a relatively large area, but not so good at distinguishing the exact strength of a high frequency brightness variation. This fact allows one to get away with greatly reducing the amount of information in the high frequency components. This is done by simply dividing each component in the frequency domain by a constant (also called as quantization table) and then rounding to the nearest integer.

4) Then, the quantizated matrix is converted to an array by zig-zag scan as shown in the figure 2.1. It groups similar frequencies together.

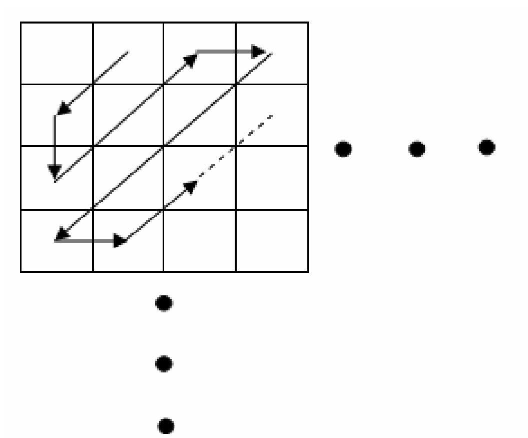


Figure 2.1 The zig-zag scan of quantizated matrix.

5) Differential pulse code modulation (DPCM) is used on the DC component apart from AC components. This strategy is adopted, because DC component is large and varied, but often close to previous value.

6) AC components go into a VLC. Huffman Coding is one of the most famous types of VLC. If the frequency of a number is high, it is coded by less bits or vise versa.

The general overview of JPEG can be easily seen in the figure 2.2.

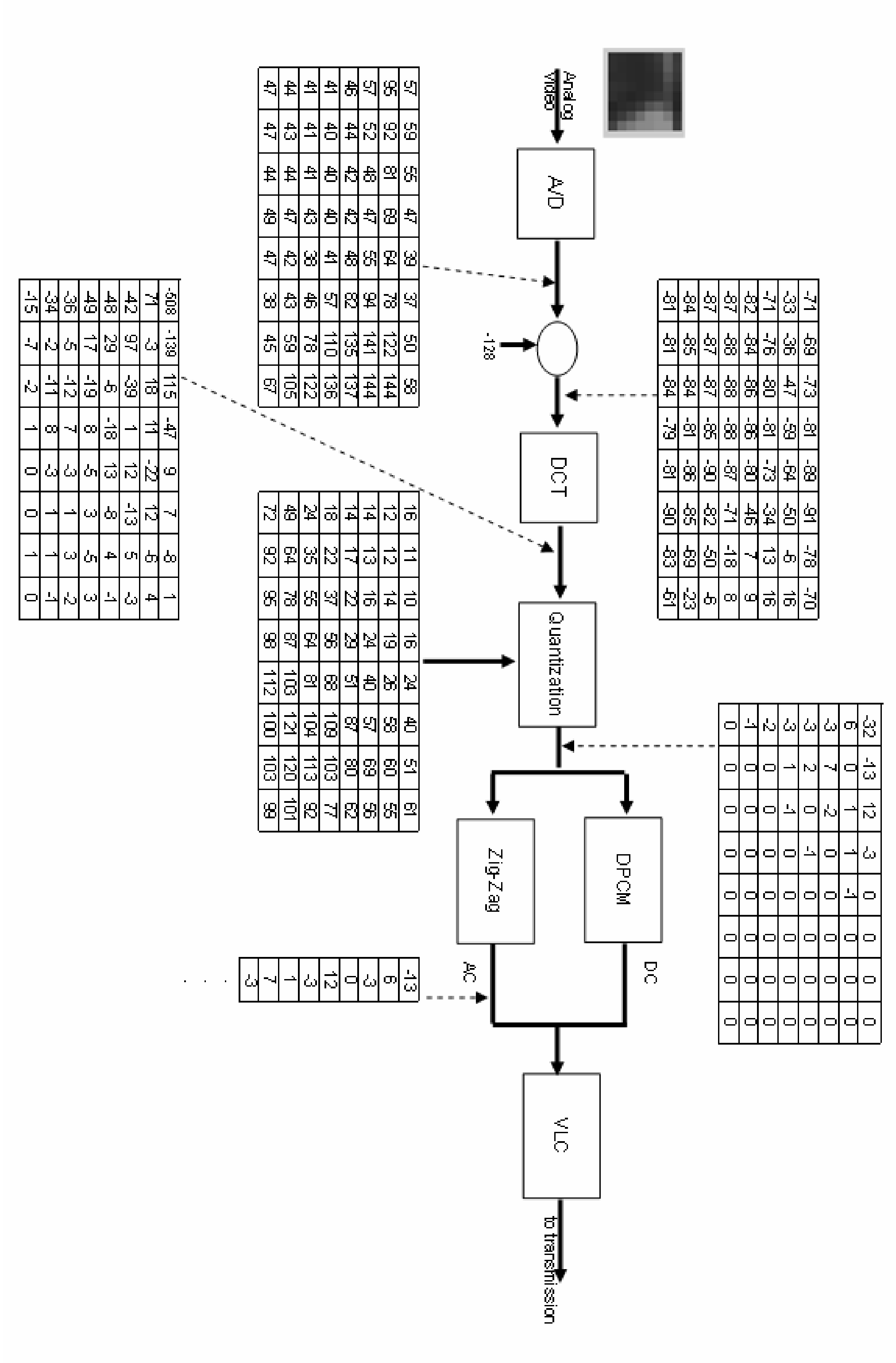


Figure 2.2 The base of Joint Photographic Expert Group (JPEG) with all steps and examples.

2.1.2 MPEG

Basically, MPEG is an advanced form of JPEG which includes motion data. MPEG decoder uses the main feature of JPEG, in addition, it includes the motion data (Figure 2.3).

Firstly, it applies a team of JPEG features (DCT, quantization, VLC…) to the first frame and sends it. Secondly, it makes inverse-JPEG processing (inverse DCT, inverse quantization…) and reproduces the first frame. Then it asks a question to send second frame: "Which block in the first frame moves 'where' to produce the second frame?" and the answer lies in the motion vectors. After finding motion vectors, it produces "the predicted second frame" by moving the blocks of first frame. Then, it takes the difference between the predicted second frame and real second frame and send it with motion vectors instead of whole second frame. As a result it processes less bits to send second frame.

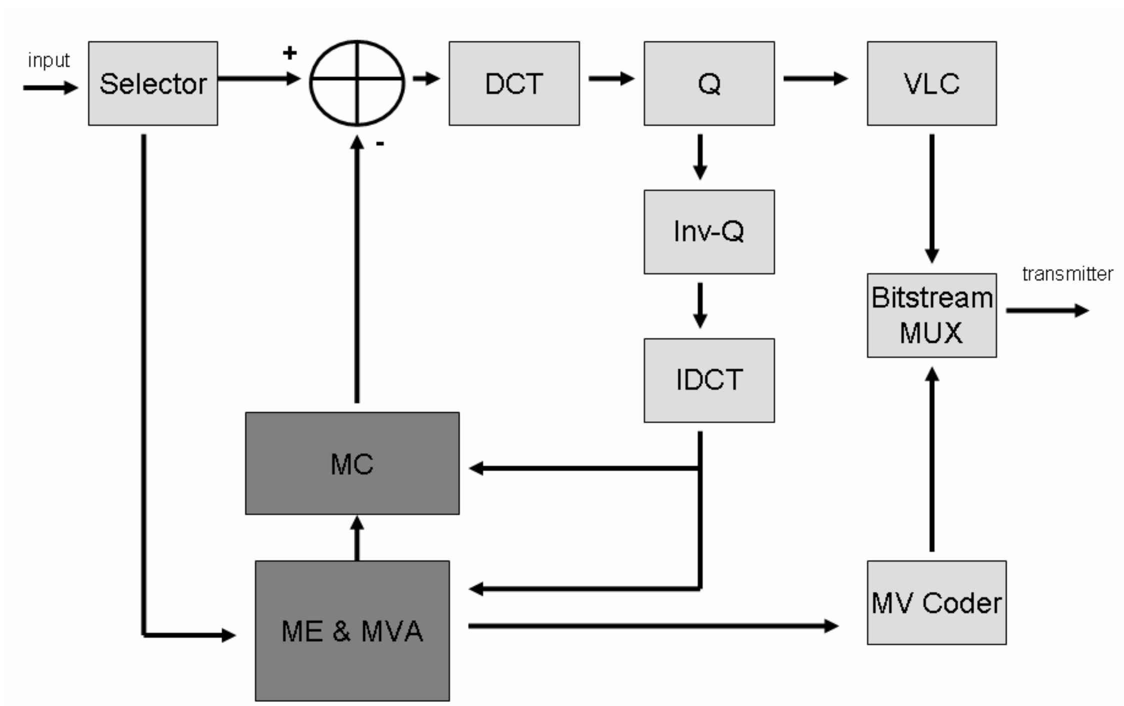


Figure 2.3 The base of Moving Picture Expert Group (MPEG) decoder with its main steps.

MPEG encoder takes the first frame and motion vectors (Figure 2.4). It reproduces the “predicted second frame” and add on it with the difference image and get the second frame.

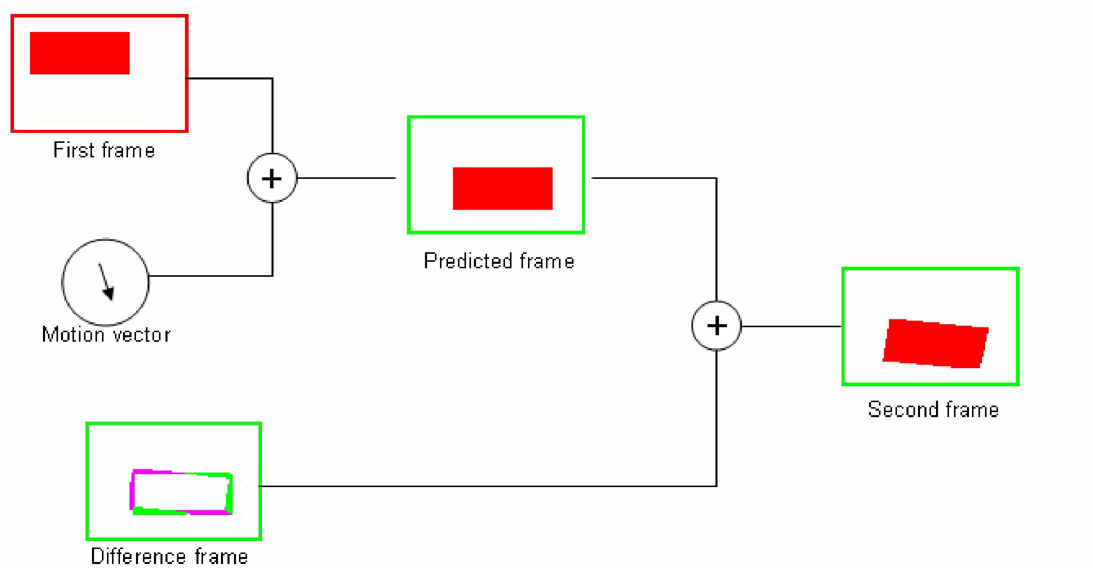


Figure 2.4 The base of Moving Picture Expert Group (MPEG) decoder with its main steps.

2.1.3 Compression Types In MPEG

Two of the most powerful techniques for compressing video are interframe and intraframe compression. Interframe compression uses one or more earlier or later frames in a sequence to compress the current frame, while intraframe compression uses only the current frame, which is effectively image compression.

With the interframe compression, if the frame contains areas where nothing has moved, the system simply issues a short command that copies that part of the previous frame, bit-for-bit, into the next one.

There are many parts of a frame whose gray levels or RGB values don’t change, like a wall or sky. MPEG splits the frame in an appropriate way and reduces the similar parts to a unique part, so it reduces the spatial differences. That is intraframe compression. The intraframe compression also uses discrete cosine transform (DCT) and store or transmit the frame in the frequency domain.

MPEG also uses the sensitivity of a human eye. Since an eye is more sensitive to luminance data of an object than its chrominance data, MPEG uses more bits to store or transmit luminance data compared to chrominance data.

There is one more issue about the eye. During motion, the human focuses on the motion or motion type of object more than detail of object. MPEG makes use of this property of human visual system and reduces the image details of a moving object in frames.

As you see, the motion analysis is one of the most important issue in the MPEG technology. It can reproduce the frame by using motion vectors.

CHAPTER THREE

SPATIAL DOMAIN MOTION ESTIMATION

3.1 Block Matching Search

The underlying assumption behind motion estimation is that the patterns corresponding to objects and background in a frame move within the frame to form corresponding objects on the following frame. The basic idea is to divide the current frame into a matrix of 'macro blocks', then these macroblocks are compared with corresponding block and its adjacent neighbors in the next frame to create a vector which shows the movement of a macro block from one location to another in the next frame. This movement is calculated for all the macro blocks. The search area for a good macro block match is limited to *k* pixels on all fours sides of the corresponding macro block in previous frame. This *k* is called as the search parameter. Larger motions require a larger *k* and it is obvious that the larger the search parameter means more computationally complex process. Usually the macro block is taken as a square of 8 pixels and the search parameter *k* is 7 pixels (Figure 3.1).

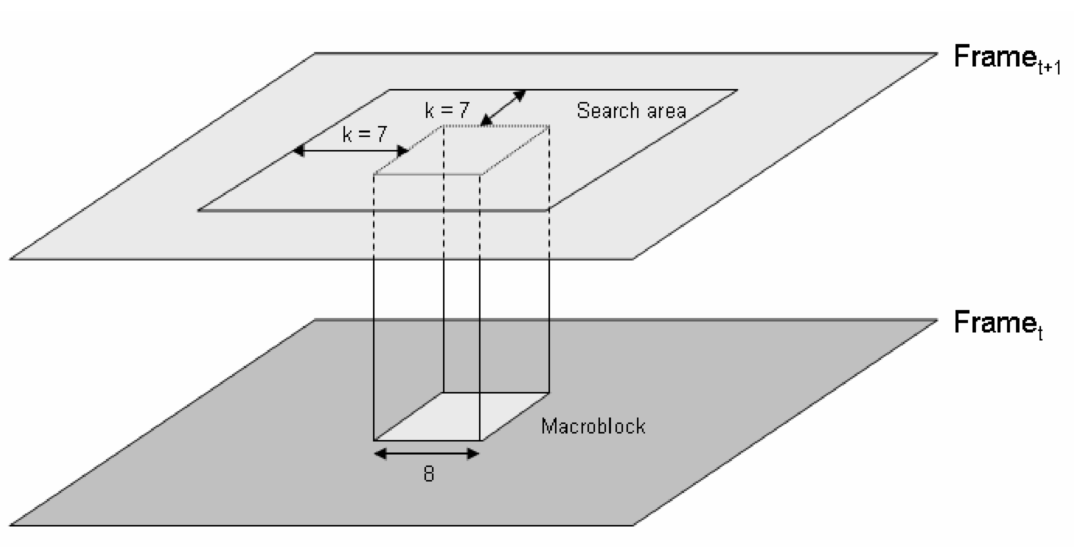


Figure 3.1 Block matching for a macroblock with 8 pixels and a search area with k=7 pixels.

The matching of one macro block with another is based on the output of a cost function. The macro block that results the least cost is the one that matches the closest to current block. There are various cost functions. The most popular and less computationally expensive is *mean absolute difference* (MAD) given by equation 3.1 and another cost function is *mean squared error* (MSE) given by equation 3.2.

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|K_{ij} - M_{ij}\right| \qquad \text{Equation 3.1}$$

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(K_{ij} - M_{ij}\right)^2 \qquad \text{Equation 3.2}$$

where N is the side of the macro bock, K and M are the pixels being compared in current macro block and reference macro block, respectively.

3.1.1 Full Search

This algorithm, also known as *exhaustive search* (ES), is the most computationally expensive block matching algorithm of all. This algorithm calculates the cost function at each possible location in the search window. As a result of which it finds the best possible match and gives the highest peak signal to noise ratio (PSNR) amongst all block matching algorithms. The obvious disadvantage of ES is that the larger the search window gets the more computations it requires. For an example in the figure 3.1, when k is 7 and the size of macroblock is 8x8, then each comparison needs 64 subtractions and 63 additions to calculate the MAD and for each macroblock it needs 15x15=225 comparisons to find the motion vector.

Other fast block matching algorithms try to achieve the same PSNR doing as less computation as possible.

3.1.2 Three Step Search

This is one of the earliest attempts at fast block matching algorithms and dates back to starts 1980s (Koga, Iimuna, Hirano, Iijima & Ishiguro, 1981). The general idea is represented in the figure 3.2.

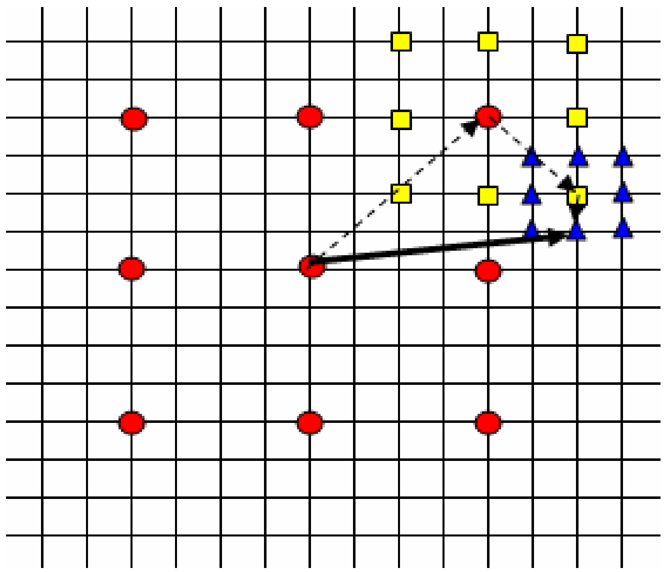


Figure 3.2 Three step search which results the motion vector with [6, 1].

Three step search (TSS) starts with a search location at the center of the search area and searches in search window with sides of S=4 for a usual search area (k=7). It then searches at eight locations +/- S pixels around location (0, 0). From these nine locations searched so far it picks the one giving least cost and makes it the new search origin. It then sets the new step size S=S/2 and repeats similar search for two more iterations until S=1.

3.1.3 New Three Step Search

The new three step search algorithm utilizes additional checking points and has provisions for half way stop to reduce computational cost to improve the performance of TSS (Li, Zeng & Liou, 1994). There are two basic search area with S=4 and S=1. If the lowest weight is at any one of the 8 locations at S=1, then we change the origin of the search to that point and check for weights adjacent to it, then there remains one more search with S=1. On the other hand if the lowest weight after the first step was one of the 8 locations at S=4, then we follow the normal TSS procedure as shown in the figure 3.3.

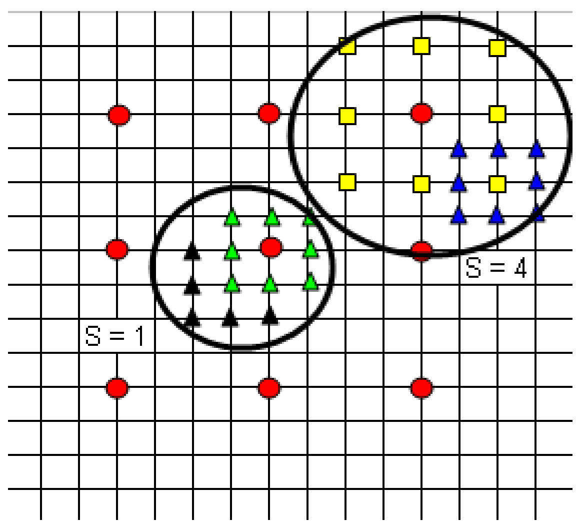


Figure 3.3 New three step search.

3.1.4 Four Step Search

Four step search employs the center biased property of the motion vectors similar to NTSS (Po & Ma, 1996). First, the search step size is set to 2 as shown in the figure 3.4.

Nine points are checked in the search window. If the best match occurs at the center of the window, the step size of neighbor search window reduced to one with eight checking points and the best match is the best predicted motion vector. If the best match in the first step occurs on the edges or corners of the search window, additional three or five points will be checked in the second step, respectively. If the current minimum occurs on the center of the search window, the step size will reduce to one. The algorithm stops while all the neighboring points are checked.

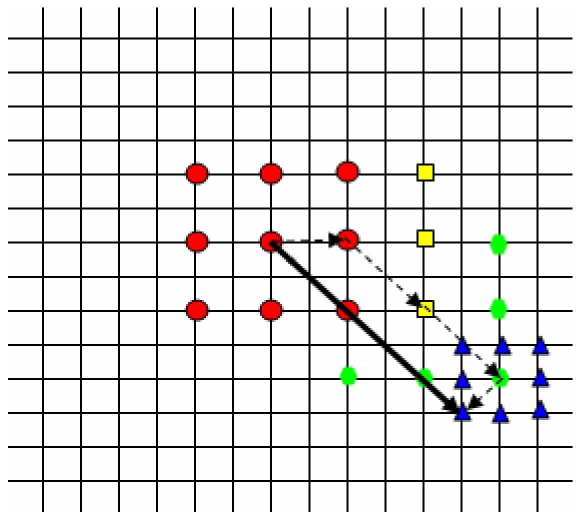


Figure 3.4 Four step search which results the motion vector with [5, -5].

3.1.5 Adaptive Rood Pattern Search

Adaptive Rood Pattern Search algorithm makes use of the fact that the general motion in a frame is usually coherent, i.e. if the macro blocks around the current macro block moved in a particular direction then there is a high probability that the current macro block will also have a similar motion vector (Nie & Ma, 2002). This algorithm uses the motion vector of the macro block to its ROS (Region of Search) to predict its own motion vector (Predicted Vector). An example is shown in the figure 3.5.

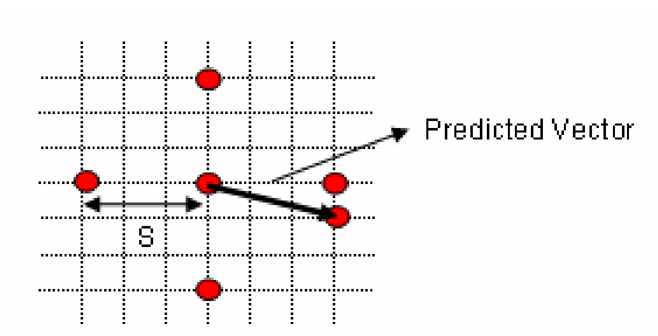


Figure 3.5 Adaptive rood pattern with the predicted motion vector is (3, -1) and the step size S=Max(|3|, |-1|)=3.

Question is how to find predicted vector. For reference predicted vector referred to upper leftmost macroblock must be found by a usual method mentioned before. The predicted vector of other macroblocks can be found by averaging or median filtering of neighboring vectors. ROS defines which neighbors can be noticed. There are many types of ROS as shown in the figure 3.6.

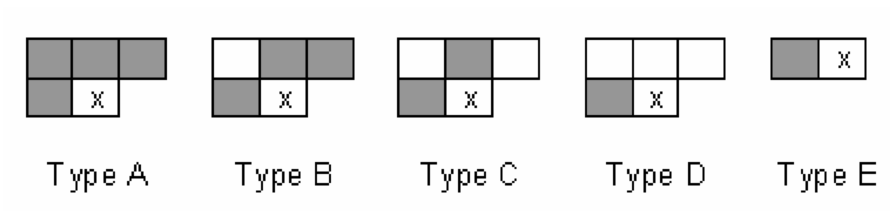


Figure 3.6 ROS types: ROS is depicted by the shaded blocks and the macroblock marked by "x" is the current block.

Type A covers all the four neighboring blocks and type B is the prediction ROS adopted in some international standards such as H.263. Type C is composed of two directly adjacent blocks and type D has only one block that situates at the immediate left to the current block. Type E is designed for upper macroblock except the leftmost one.

The algorithm checks the location pointed by the predicted motion vector, it also checks at a rood pattern distributed points, as shown in figure 3.5, where they are at a step size of S=Max (|X|, |Y|). It has been noticed that the MV distribution in horizontal and vertical directions are higher than in other directions since most of camera movements are in these directions.

This rood pattern search is always the first step. It directly puts the search in an area where there is a high probability of finding a good matching block. The point that has the least weight becomes the origin for following search steps and the search pattern is changed to window with the size of S=1. The procedure keeps on this until least weighted point is found to be at the center of the search window.

In many visual communication applications such as video telephony, there is little motion between the adjacent frames. Hence, a large percentage of zero-motion blocks are encountered in such type of video sequences. So a further small improvement in the algorithm can be to check for Zero Motion Prejudgment, using which the search is stopped half way if the least weighted point is already at the center of the rood pattern.

3.2 Differential - Gradient Based Search

3.2.1 Basic Principle Of Differential - Gradient Based Search

In the literature, mostly, differential-gradient based search is also called optical flow and optical flow is the distribution of apparent velocities of movement of brightness patterns in an image. Optical flow can arise from relative motion of objects and the viewer. Discontinuities in the optical flow can help in segmenting images into regions that correspond to different objects. The optical flow cannot be computed at a point in the image independently of neighboring points without introducing additional constraints.

To avoid variations in brightness due to shading effects it is initially assumed that the surface being imaged is flat. It is further assumed that the incident illumination is uniform across the surface. The brightness at a point in the image is then proportional to the reflectance of the surface at the corresponding point on the object. Also, it is assumed that reflectance varies smoothly and has no spatial discontinuities. This latter condition assures us that the image brightness is differential. It is excluded situations where objects occlude one another, in part, because discontinuities in reflectance are found at object boundaries (Horn & Schunck, 1981).

3.2.2 Constraints

3.2.2.1 Differential

We will derive an equation that relates the change in image brightness at a point to the motion of the brightness pattern. Let the image brightness at the point (x, y) in the image plane at time r be denoted by E(x, y, t). It is assumed that the brightness moves along the motion and the brightness of a point is equal to new point after ($\delta x, \delta y, \delta t$) variation.

$$E(x, y, t) = E(x + \delta x, y + \delta y, t + \delta t) \qquad \text{Equation 3.3}$$

If the right side of the equation 3.3 is focused on with leading of Taylor series;

$$E(x, y, t) = E(x, y, t) + \delta x \frac{\partial E}{\partial x} + \delta y \frac{\partial E}{\partial y} + \delta t \frac{\partial E}{\partial t} + \in \qquad \text{Equation 3.4}$$

where $\in$ refers to second and higher terms of series and it can be assumed to be equal to zero. After dividing both sides of equation 3.4 with δ t;

$$\frac{\delta x}{\delta t}\frac{\partial E}{\partial x} + \frac{\delta y}{\delta t}\frac{\partial E}{\partial y} + \frac{\delta t}{\delta t}\frac{\partial E}{\partial t} = 0 \qquad \text{Equation 3.5}$$

While $\delta t \rightarrow 0$, equation 3.5 becomes;

$$\frac{\partial E}{\partial x}\frac{dx}{dt}+\frac{\partial E}{\partial y}\frac{dy}{dt}+\frac{\partial E}{\partial t}=0$$ Equation 3.6

Assuming that $u=\frac{dx}{dt}$ and $v=\frac{dy}{dt}$, then equation 3.6 becomes;

$$E_X.u+E_Y.v+E_t=0$$ Equation 3.7

where E_x, E_y, and E_t, for the partial derivatives of image brightness with respect to x, y and t, respectively.

Equation 3.7 can be also written as;

$$\begin{bmatrix} E_x & E_y \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}=-E_t$$ Equation 3.8

3.2.2.2 Smoothness

Neighboring points on the objects have similar velocities and the velocity field of the brightness patterns in the image varies smoothly almost everywhere. Discontinuities in flow can be expected where one object occludes another. An algorithm based on a smoothness constraint is likely to have difficulties with occluding edges as a result. One way to express the additional constraint is to minimize the square of the magnitude of the gradient of the optical flow velocity as;

$$\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial u}{\partial y}\right)^2 \text{ and } \left(\frac{\partial v}{\partial x}\right)^2+\left(\frac{\partial v}{\partial y}\right)^2$$

Another measure of the smoothness of the optical flow field is the sum of the squares of the Laplacians of the x and y components of the flow. The Laplacians of u and v are defined as;

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial u^2} \text{ and } \nabla^2 v = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial u^2} \qquad \text{Equation 3.9}$$

3.2.3 Estimating The Partial Derivatives

We must estimate the derivatives of brightness from the discrete set of image brightness measurements available. It is important that the estimates of E_x, E_y and E_t be consistent. That is, they should all refer to the same point in the image at the same time. While there are many formulas for approximate differentiation, it is commonly used a set which gives us an estimate of E_x, E_y and E_t at a point in the center of a cube formed by eight measurements. The relationship in space and time between these measurements is shown in the figure 3.7.

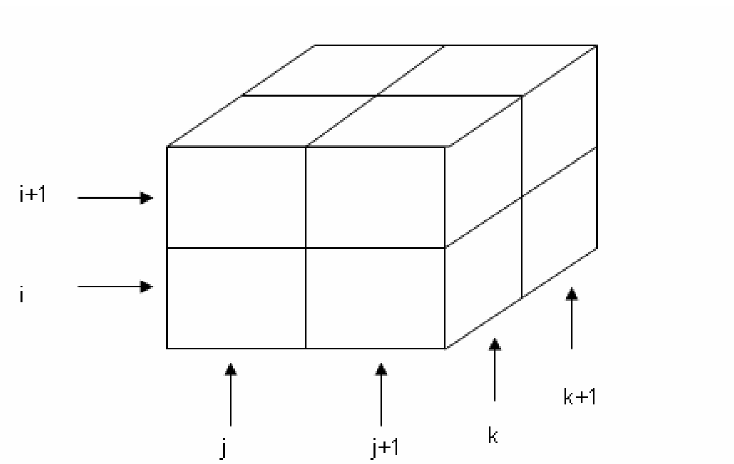


Figure 3.7 Here the column index j corresponds to the x direction in the image, the row index i to the y direction, k lies in the time direction.

The three partial derivatives of images brightness at the center of the cube are each estimated from the average of first differences along four parallel edges of the cube.

$$E_x \approx \frac{1}{4}\left[E_{i,j+1,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i+1,j,k+1}\right]$$

$$E_y \approx \frac{1}{4}\left[E_{i+1,j,k} - E_{i,j,k} + E_{i+1,j+1,k} - E_{i,j+1,k} + E_{i+1,j,k+1} - E_{i,j,k+1} + E_{i+1,j+1,k+1} - E_{i,j+1,k+1}\right]$$

$$E_t \approx \frac{1}{4}\left[E_{i,j,k+1} - E_{i,j,k} + E_{i+1,j,k+1} - E_{i+1,j,k} + E_{i,j+1,k+1} - E_{i,j+1,k} + E_{i+1,j+1,k+1} - E_{i+1,j+1,k}\right]$$

Equation 3.10

3.2.4 Estimating The Laplacian Of The Flow Velocities

We also need to approximate the Laplacians of and v. One convenient approximation takes the following form;

$$\nabla^2 u \approx k(\bar{u}_{i,j,k} - u_{i,j,k}) \text{ and } \nabla^2 v \approx k(\bar{v}_{i,j,k} - v_{i,j,k})$$ Equation 3.11

where the local averages $\bar{u}$ and $\bar{v}$ are defined as follows;

$$\bar{u} = u * \begin{bmatrix} 1/12 & 1/6 & 1/12 \\ 1/6 & 0 & 1/6 \\ 1/12 & 1/6 & 1/12 \end{bmatrix} \text{ and } \bar{v} = v * \begin{bmatrix} 1/12 & 1/6 & 1/12 \\ 1/6 & 0 & 1/6 \\ 1/12 & 1/6 & 1/12 \end{bmatrix}$$ Equation 3.12

So the Laplacian becomes as;

$$\nabla^2 u \cong u * \begin{bmatrix} 1/12 & 1/6 & 1/12 \\ 1/6 & -1 & 1/6 \\ 1/12 & 1/6 & 1/12 \end{bmatrix} \text{ and } \nabla^2 v \cong v * \begin{bmatrix} 1/12 & 1/6 & 1/12 \\ 1/6 & -1 & 1/6 \\ 1/12 & 1/6 & 1/12 \end{bmatrix}$$ Equation 3.13

3.2.5 Minimization

The problem then is to minimize the sum of the errors in the equation for the rate of change of image brightness as;

$$\varepsilon_b = E_x u + E_y v + E_t \qquad \text{Equation 3.14}$$

and the measure of the departure from smoothness in the velocity flow as;

$$\varepsilon_c^2 = \nabla^2 u + \nabla^2 v \qquad \text{Equation 3.15}$$

What should be the relative weight of these two factors? In practice the image brightness measurements will be corrupted by quantization error and noise so that it is cannot be expected that ε_b is identically zero. This quantity will tend to have that we cannot expect an error magnitude that is proportional to the noise in the measurement. This fact forces to choose a suitable weighting factor, denoted by α^2.This weighting factor plays a significant role only for areas where the brightness gradient is small, preventing haphazard adjustments to the estimated flow velocity occasioned by noise in the estimated derivatives.

Let the total error to be minimized be;

$$\varepsilon = \iint \left(\alpha^2 . \varepsilon_c^2 + \varepsilon_b^2\right) dx.dy \qquad \text{Equation 3.16}$$

After studying on equation 3.16, the result is;

$$E_x^2 u + E_x E_y v = \alpha^2 \nabla^2 u - E_x E_t$$
$$E_x^2 v + E_x E_y u = \alpha^2 \nabla^2 v - E_y E_t \qquad \text{Equation 3.17}$$

If the Laplacian is embedded into the equation 3.17;

$$(E_x^2+\alpha^2)u+E_xE_yv=\alpha^2\bar{u}-E_xE_t$$
$$E_xE_yu+(\alpha^2+E_x^2)v=\alpha^2\bar{v}-E_yE_t \qquad \text{Equation 3.18}$$

The determinant of the coefficient matrix equals $\alpha^2(\alpha^2+E_x^2+E_y^2)$. Solving for u and v, we find that;

$$\alpha^2(\alpha^2+E_x^2+E_y^2)u=\left(\alpha^2+E_y^2\right)\bar{u}-E_xE_y\bar{v}-E_xE_t$$
$$\alpha^2(\alpha^2+E_x^2+E_y^2)v=\left(\alpha^2+E_x^2\right)\bar{v}-E_xE_y\bar{u}-E_yE_t \qquad \text{Equation 3.19}$$

These equations can be written in the alternate form as;

$$\left(\alpha^2+E_x^2+E_y^2\right)\left(u-\bar{u}\right)=-E_x\left[E_x\bar{u}+E_y\bar{v}+E_t\right]$$
$$\left(\alpha^2+E_x^2+E_y^2\right)\left(v-\bar{v}\right)=-E_y\left[E_x\bar{u}+E_y\bar{v}+E_t\right] \qquad \text{Equation 3.20}$$

3.2.6 Iterative Solution

To get less error, an iterative method needs to be used and Berthold K.P. Horn and Brian G. Rhunck suggest the Gauss-Seidel method cause of less computational load. Anew set of velocity can be computed a new set of velocity (u^{n+1}, v^{n+1}) from the estimated derivatives and the average of the previous velocity estimates (u^n, v^n) by;

$$u^{n+1}=\bar{u}^n-E_x\left(\frac{E_x\bar{u}^n+E_y\bar{v}^n+E_t}{\alpha^2+E_x^2+E_y^2}\right) \quad v^{n+1}=\bar{v}^n-E_y\left(\frac{E_x\bar{u}^n+E_y\bar{v}^n+E_t}{\alpha^2+E_x^2+E_y^2}\right) \qquad \text{Equation 3.21}$$

CHAPTER FOUR

FREQUENCY DOMAIN MOTION ESTIMATION

4.1 Fourier Transform

Fourier analysis, named after Joseph Fourier's introduction of the Fourier series, is the decomposition of a function in terms of sinusoidal functions (called basis functions) of different frequencies that can be recombined to obtain the original function. The recombination process is called Fourier synthesis (in which case, Fourier analysis refers specifically to the decomposition process).

Mathematical expression is varied whether the signal is periodic or non-periodic. The analog form of Fourier transform is shown in the table 4.1.

Table 4.1 The analog form of Fourier transform

	Periodic	Non-Periodic
Analysis	$a_k = \frac{1}{T}\int_T x(t)e^{-jkw_o t}.dt$	$X(jw) = \int_{-\infty}^{\infty} x(t)e^{-jwt}dt$
Synthesis	$x(t) = \sum_{-\infty}^{\infty} a_k e^{jkw_o t}$	$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(jw)e^{jwt}.dw$

In the above table, a_k means complex coefficients in the frequency domain and T means period.

Some Fourier Transform Properties

Linearity

$$ax_1(t)+bx_2(t) \leftrightarrow aX_1(jw)+bX_2(jw)$$ Equation 4.1

Multiplication

$$x(t).p(t) \leftrightarrow \frac{1}{2\pi}X(jw).P(jw)$$ Equation 4.2

Modulation

$$e^{jw_0t}.x(t) \leftrightarrow X\left[j(w-w_0)\right]$$ Equation 4.3

Convolution

$$h(t)*x(t) \leftrightarrow H(jw).X(jw)$$ Equation 4.4

Integration

$$\int_{-\infty}^{t} x(t).d(t) \leftrightarrow \frac{1}{jw}X(jw)+\pi.X(0).\delta(w)$$ Equation 4.5

Differentation

$$\frac{dx(t)}{dt} = jwX(jw)$$ Equation 4.6

Conjugation

$$x^*(t) \leftrightarrow X^*(-jw)$$ Equation 4.7

Scaling

$$x(at) \leftrightarrow \frac{1}{|a|} X\left(\frac{jw}{a}\right)$$ Equation 4.8

Time shift

$$x(t-t_0) \leftrightarrow e^{-jwt_0} X(jw)$$ Equation 4.9

Parseval's theorem

$$\int_{-\infty}^{\infty} |x(t)|^2 .dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(jw)|^2 .dw$$ Equation 4.10

4.2 Basic Phase Correlation

Phase correlation technique is a frequency domain motion estimation method that makes use of the shift property of the Fourier transform. According to this property, a shift in the temporal domain is equivalent to a phase shift in the frequency domain.

Assuming a translational shift between the two frames;

$$s(n_1, n_2, k) = s(n_1 + d_1, n_2 + d_2, k+1)$$ Equation 4.11

Their 2-D Fourier transforms are;

$$S_k(f_1, f_2) = S_{k+1}(f_1, f_2).e^{-j2\pi(d_1.f_1 + d_2.f_2)}$$ Equation 4.12

Therefore, a shift in the spatial-domain is reflected as a phase change in the spectrum domain. The cross-correlation between the two frames is

$$c_{k,k+1}(n_1,n_2) = s(n_1,n_2,k+1) * s(n_1,n_2,k) \qquad \text{Equation 4.13}$$

whose Fourier transform is;

$$C_{k,k+1}(f_1,f_2) = S_{k+1}(f_1,f_2).S^*(f_1,f_2) \qquad \text{Equation 4.14}$$

In order to get rid of the luminance variation influence during our phase analysis, we normalize the cross-power spectrum by its magnitude and obtain its phase as;

$$\phi\left[C_{k,k+1}(f_1,f_2)\right] = \frac{S_{k+1}(f_1,f_2).S^*(f_1,f_2)}{\left|S_{k+1}(f_1,f_2).S^*(f_1,f_2)\right|} \qquad \text{Equation 4.15}$$

By equation 4.11 and 4.15, we have,

$$\phi\left[C_{k,k+1}(f_1,f_2)\right] = e^{-j2\pi(d_1.f_1+d_2.f_2)} \qquad \text{Equation 4.16}$$

whose 2-D inverse transform is given by;

$$c_{k,k+1}(n_1,n_2) = \delta(n_1 - d_1, n_2 - d_2) \qquad \text{Equation 4.17}$$

As a result, by finding the location of the pulse in the equation 4.17 we are able to tell the displacement, which is the motion vector. In practice, the motion is not pure translational, and we will get the phase correlation similar to what is depicted in the figure 4.1. In this case, we locate the pulse by finding the highest peak or a few candidates.

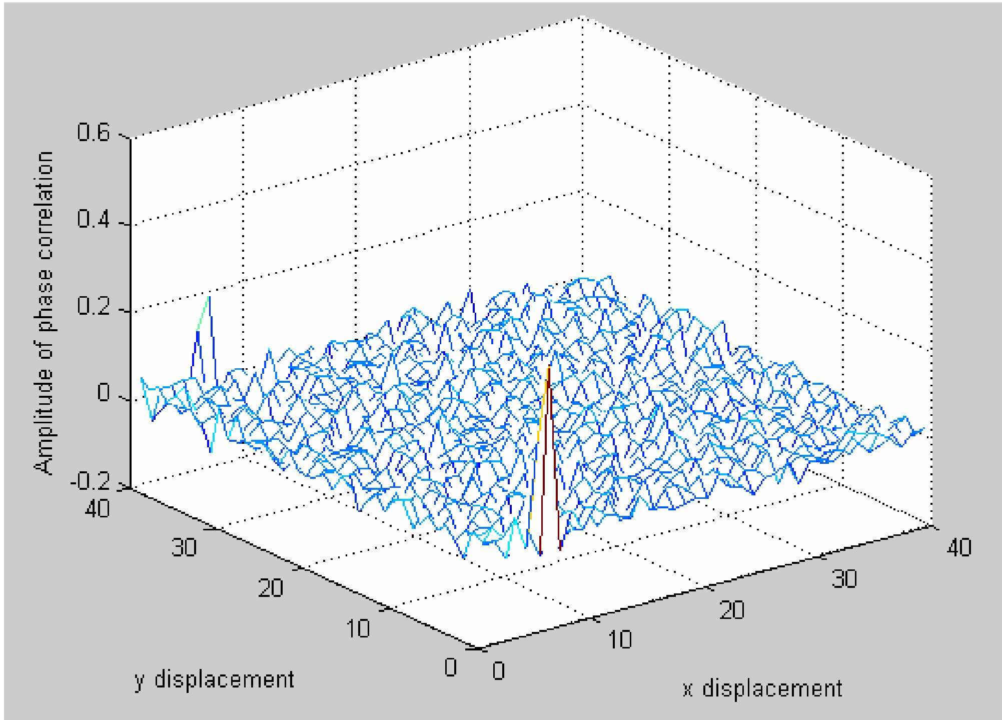
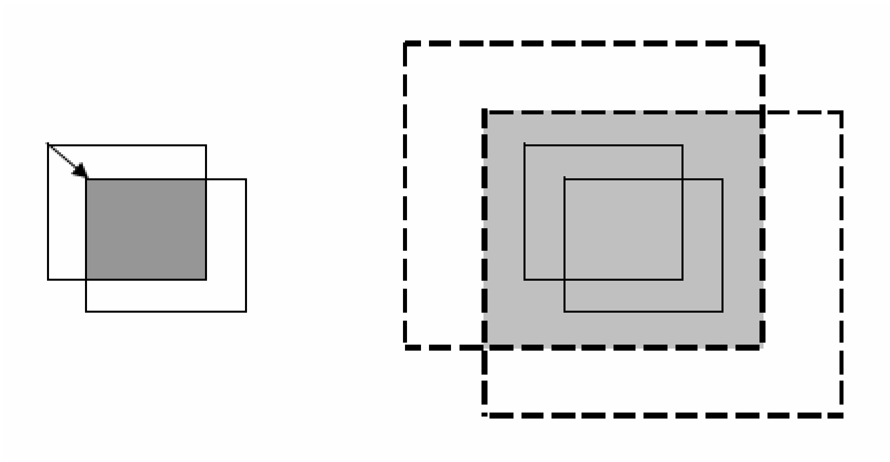


Figure 4.1 The phase correlation of two successive frames and the peaks give possible motions.

In implementation, the current frame is divided into blocks of 16x16 pixels and the phase correlation calculation is performed for each block. In order to correctly estimate the cross correlation of corresponding blocks in respective frames, we take the extended subimage of 32x32 pixels in size, including the predefined block of 16x16 pixels at the center, and then calculate the phase correlation. If we do the phase correlations only over a subimage of 16x16 pixels, the correlation might be too low for a particular motion due to the small overlapping area, as shown in the figure 4.2.a. Once the subimage size is extended to 32x32, the overlapping area is increased for better correlation estimation, as is shown in the figure 4.2.b. The resulting motion vector is assigned to the associated block of 16x16.



(a) (b)

Figure 4.2 Overlapping areas for phase correlation. a presents the small and inefficient overlapping area and b presents the right one.

Although the phase correlation peak gives us some idea on the displacement between the blocks, it doesn't tell us whereabouts within the block the movement takes place. In addition to this, there may be more than one peak which should be focused on. In this case, several candidate peaks are selected instead of just one highest peak and then we decide on which peak best represents the displacement vector for the object block. Once the candidates are selected, we examine them one by one using image correlations. For each candidate motion vector, the current block of 16x16 pixels in the current frame can be placed in the corresponding search window of 32x32 pixels in the previous frame to measure the extent of correlation. The candidate resulting in the highest image correlation is the one we searched for and its displacement is the right motion vector for the object block.

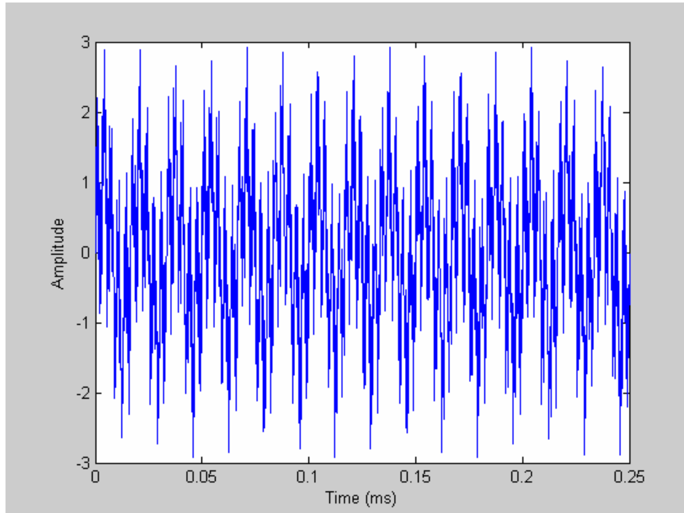
The image correlations used here is in fact a matching procedure, which is similar to the BM method, except that an image correlation is performed after the displacement vectors are already found. Therefore the computation time is greatly reduced by not trying to search the whole area.

4.3 Phase Correlation With Windowing

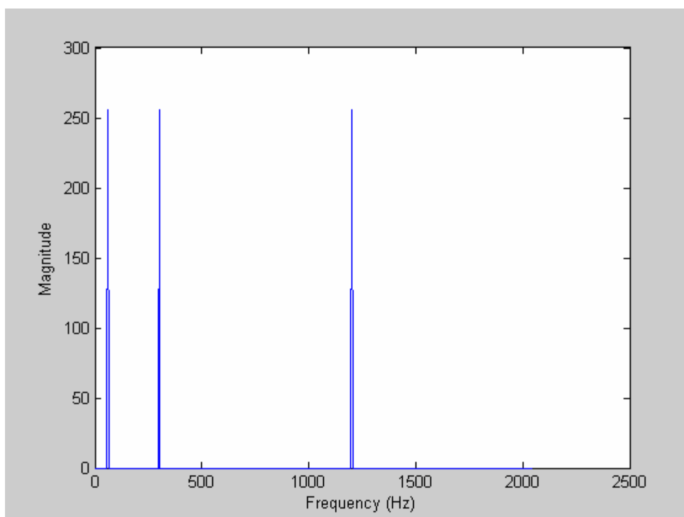
Fourier transform theory assumes that the signal being analyzed has been "on" forever. If a signal, that appears to be a clean repeating sine wave, suddenly turns off or on during an FFT analysis, the power spectrum no longer provides a simple answer for the emitter frequency. This sudden turning on or off during transmission defines a signal "edge" (Edge effect). The power spectrum of an edge no longer yields a peak for a single frequency. It is called *leakage effect.* If the signal is multiplied with a window in time domain, this problem can be reduced.

There are many types of window function such as Rectangular, Hamming, Hanning, Cosine, Lanczos, Bartlett, Triangular, Gauss, Bartlett–Hann, Blackman, and Kaiser windows. The main characteristic features of window functions are the bandwidths of the mainlobe and sidelobe, and their stopband attenuations.

In order to focus on the advantage of window function, consider a signal with;
*The length of sample period is 0.25 second.
*Sample frequency is 4096Hz.
*FFT point number is 1024, so FFT generates 512 harmonics.
*Signal is sin(2.pi.60.t)+sin(2.pi.300.t)+sin(2.pi.1200.t)
The signal and its frequency spectrum are shown in the figure 4.3, respectively.



(a)

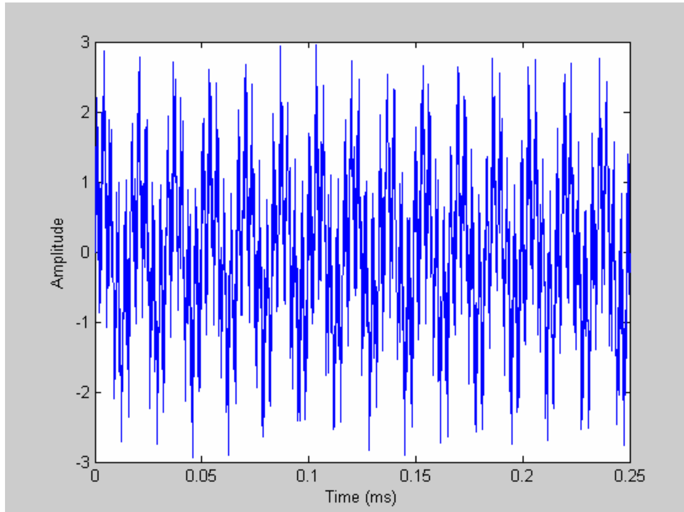


(b)

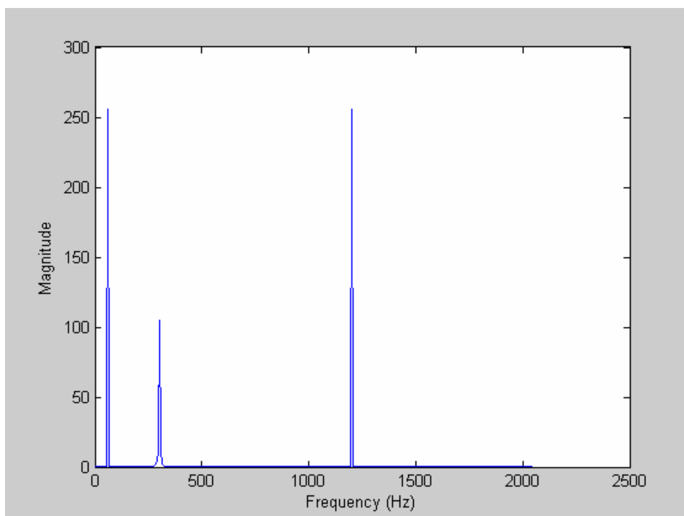
Figure 4.3 (a) The signal, (b) Its frequency spectrum

The length of the sample period is 0.25 seconds, so the first harmonic (f1) is 4 Hz. The spectrum looks nice and clean because all of the tones are multiples of 4 Hz (f1).

What about if the signal isn't multiples of 4Hz as in(2.pi.60.t)+sin(2.pi.**302**.t)+sin(2.pi.1200.t) ? The signal and its frequency spectrum are shown in the figure 4.4, respectively.



(a)



(b)

Figure 4.4 (a) The signal and (b) its frequency spectrum. Focus around 300 Hz.

As it is easy to see around 300 Hz, there occurs leakage and to prevent leakage as shown in the figure 4.5, a window function is used.

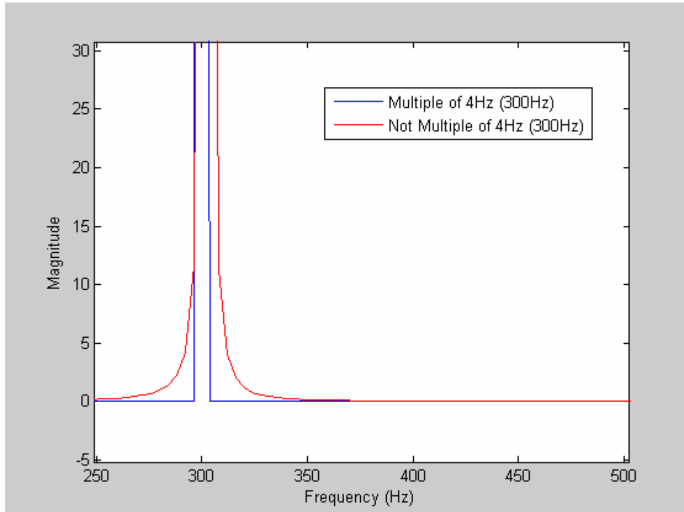
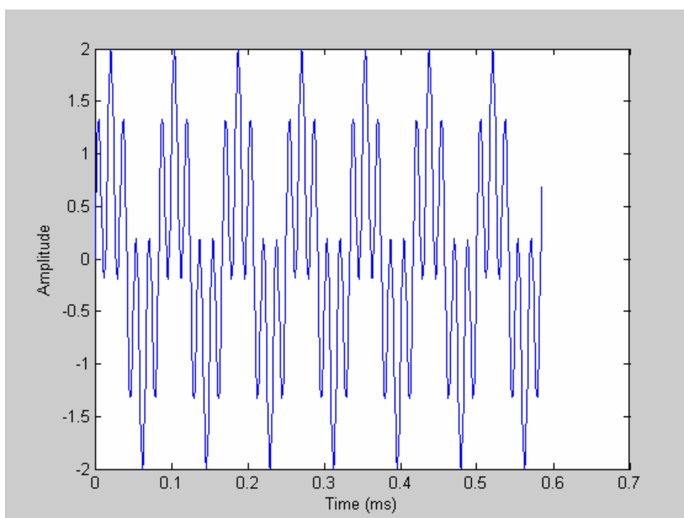
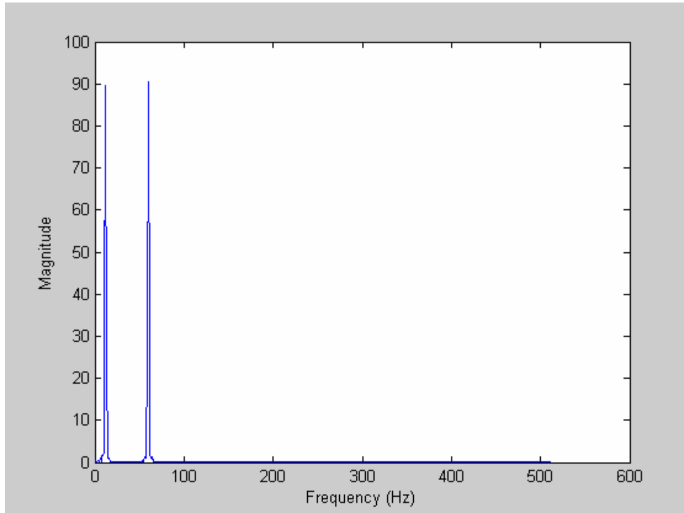


Figure 4.5 The leakage occurs around 300 Hz as shown by using red line.

Consider the signal as y=sin(2.pi.12.t)+sin(2.pi.60.t) and its frequency spectrum is shown in the figure 4.6.



(a)



(b)

Figure 4.6. (a) The signal without window and (b) its frequency spectrum.

If the signal is multiplied by the Hanning window as shown in the figure 4.7, then it and its frequency spectrum becomes as shown in the figure 4.8.

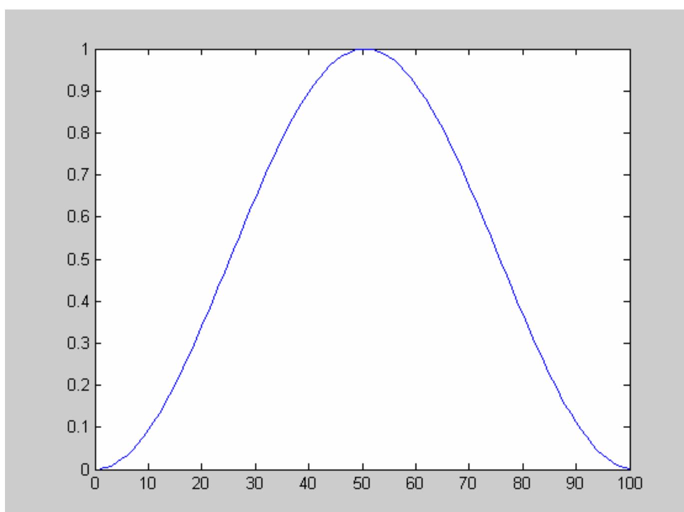
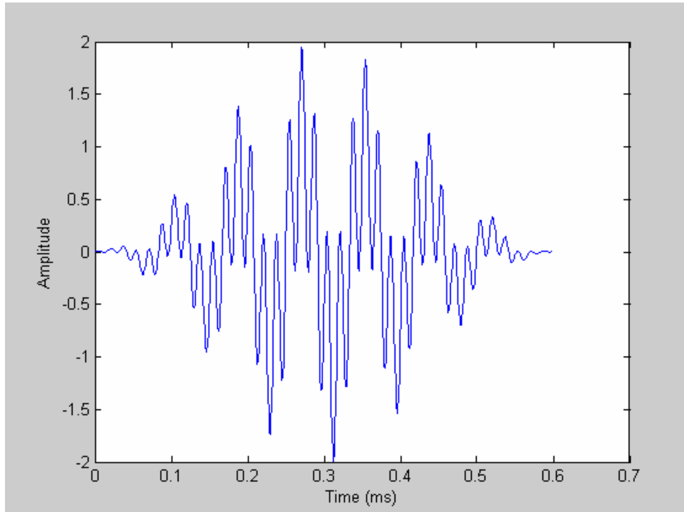
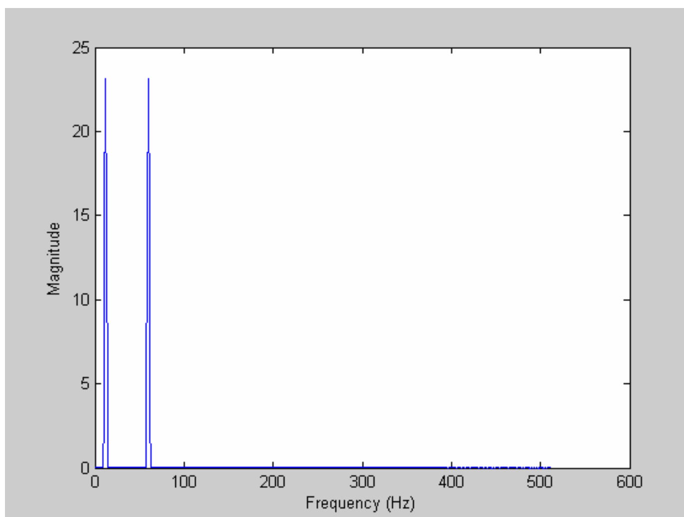


Figure 4.7. A Hanning window in 1-D.



(a)



(b)

Figure 4.8. (a) The signal with window and (b) its frequency spectrum.

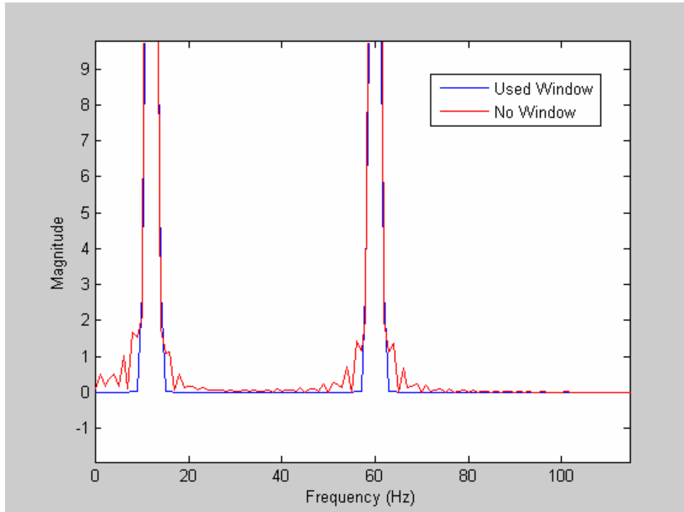


Figure 4.9 The leakage effects decreased by window function.

As a result of multiplying with a window, there occurs very less leakage as shown in the figure 4.9.

A two-dimensional window is applied to each current block of 32x32 pixels to give more weight to our formerly defined 16x16 region, to which a motion vector will be assigned.

4.4 Phase Correlation With Half Pixel Accuracy

Although digital video is represented by pixels, the motion is not necessarily limited to integer number of pixel offset. Fractional motion vector representation gives sub-pixel accuracy to motion compensation. Half-pixel improves the ME performance since it also reduces noise as it averages and interpolates the pixels.

In the phase-correlation method, we estimate possible half-pixel offset from the correlation map after the integer motion vector has been found. We interpolate the correlation peak at half-pixel offsets by examining three neighboring correlation samples adjacent to the found peak using cubic spline interpolation. If the interpolated correlation supersedes the previously found highest integer value, the motion vector is updated to the new offset, which is not an integer.

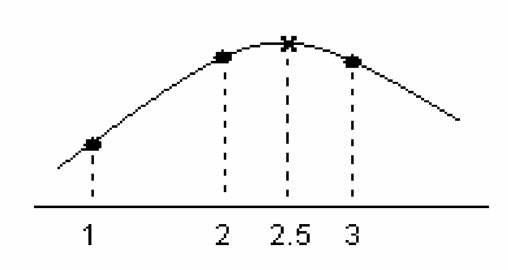


Figure 4.10 Half pixel accuracy for 1-D case

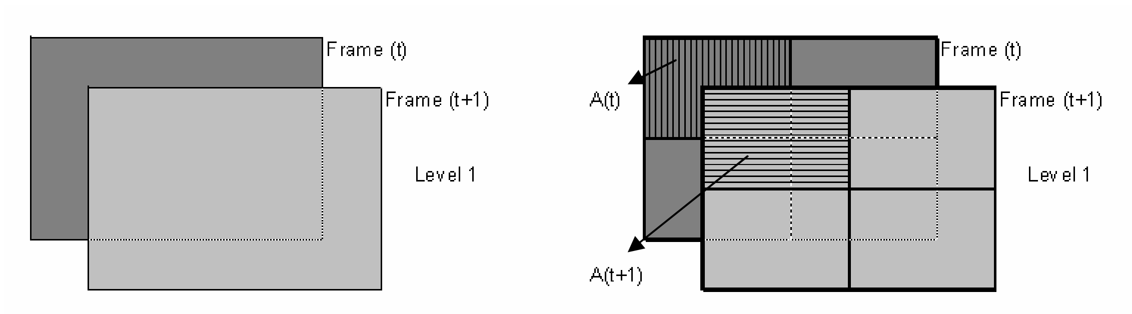
Figure 4.10 depicts this procedure for 1-D case. Samples represented by 'x' are interpolated from their neighboring samples with integer offsets, which are represented by '●'. The previously found offset is zero, but the correlation interpolated at 0.5 is even higher now. As a result, displacement zero is updated to the non-integer value 0.5.

4.5 Hierarchical Phase Correlation

Hierarchical is based on phase correlation and uses key features of the phase correlation surface to control the partition of a parent block to four children quadrants. The partition criterion is applied iteratively until no more than a single motion component per block can be identified. By this method, bigger motion can be found in contrast to fixed-block size phase correlation which does not notice the motion vector bigger than block size (Argyriou & Vlachos, 2005).

4.5.1 Splitting

The starting point for a hierarchical method is a phase correlation operation between two frames, f_t and f_{t+1}. The result may have one peak of a single motion, or more peaks, of which the highest 2-3 can indicate the significant motion components of a compound motion between the frames. At this point, the height of each of these peaks is a good measure of reliability of the corresponding motion component. In light of the foregoing, a splitting criteria can be formulated that if the ratio of the strength of the highest peak to that of the second peak is bigger than a predefined threshold, it is assumed that there is one dominant motion and that block is not split. If the ratio is lower, there are two or more motions included in the block and it should be further split. Otherwise, the motion vector corresponding to the highest peak is assigned to the entire block. When a block is splitted into quadrants, the phase correlations are performed between the two children quadrants from the two parent blocks. As shown in the figure 4.11, for example, if the ratio of peaks for A_T and A_{T+1} is lower than a threshold, it is assumed that there is two or more different motions. So, quadrants A_T and A_{T+1} are splitted into four and so on.



(a) (b)

Figure 4.11 (a) If the ratio is lower, then the frames are divided by four as shown in (b). If the phase correlation ratio between A_T and A_{T+1} is lower, then it is divided by four again and so on.

4.5.2 Overlapping Areas

As shown in the figure 4.2, the block size is extended to get bigger overlapping area for better correlation estimation. Suppose that, after any number of splitting operations, there remains two blocks, B_T and B_{T+1}. During processing of phase correlation, the original blocks B_T and B_{T+1} are extended to a bigger size as shown in the figure 4.12. Here, a window function is also advised to be used.

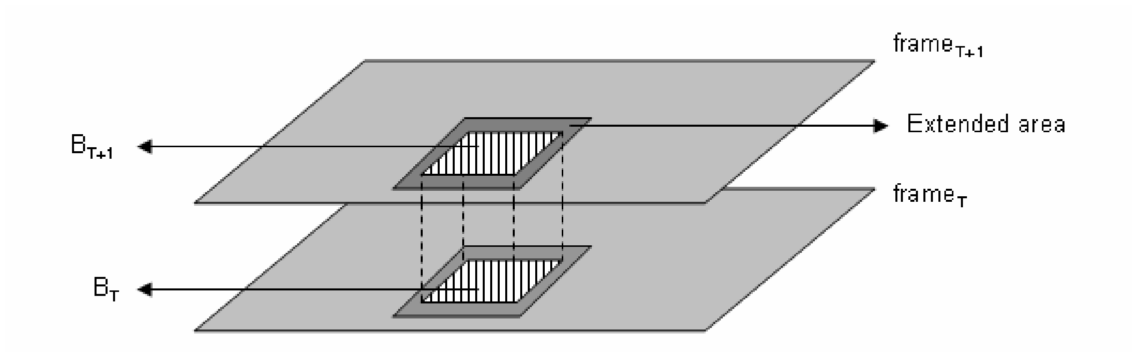


Figure 4.12 To get better motion estimation, the original areas (B_T and B_{T+1}) are extended.

4.5.4 Automatically Splitting

Alternatively, the threshold can be calculated automatically in an adaptive way. I developed a formula experimentally. Assume that L is the length between the center of gravity and the coordinate of the highest peak of the phase correlation result. If L is small enough, it means that the most of the energy of the phase correlation is concentrated at the peak point. So it can be assumed that there is one single dominant motion. The threshold can be devised as;

$$Threshold = \left| \left(\frac{\min imum\_of\_size\_of\_block - L}{\min imum\_of\_size\_of\_block} \right)^{0.4} \right| \qquad \text{Equation 4.18}$$

4.5.4 Iterations

The algorithm processes in an iterative fashion to determine whether or not quadrant image will be partitioned any further or there can be an iteration formula which is automatically calculated from the size of image.

$$Iteration = floor(\log_2(\min[row\_size, col\_size]) - 4) \qquad \text{Equation 4.19}$$

This formula prevents to get smaller blocks less than around 16x16 pixels which is a preferred size of smallest block.

CHAPTER FIVE

MOTION COMPENSATION

5.1 Basic Terms

The terminology of “Motion Compensation” is used in many areas related with video. Two of most important and famous ones are MPEG and 100/120 Hz LCD TV application. It refers to increase the field rate in a video sequence and the term of “Motion Based Frame Interpolation” is used mostly.

5.1.1 Motion Compensation In MPEG

There are four types of coded frames in the MPEG. I (intra) frames are frames coded as a stand-alone still image. They allow random access points within the video stream. As such, I frames should occur about two times a second. I frames should also be used where scene cuts occur.

P (predicted) frames are coded relative to the nearest previous I or P frame, resulting in forward prediction processing, as shown in the figure 5.1. P frames provide more compression than I frames, through the use of motion compensation and are also a reference for B frames and future P frames.

B (bi-directional) frames use the closest past and future I or P frame as a reference, resulting in bi-directional prediction, as shown in the figure 5.1. B frames provide the most compression and typically, there are two B frames separating I or P frames.

D (DC) frames are frames coded as a stand-alone still image, using only the DC component of the DCTs. D frames may not be in a sequence containing any other frame types and are rarely used.

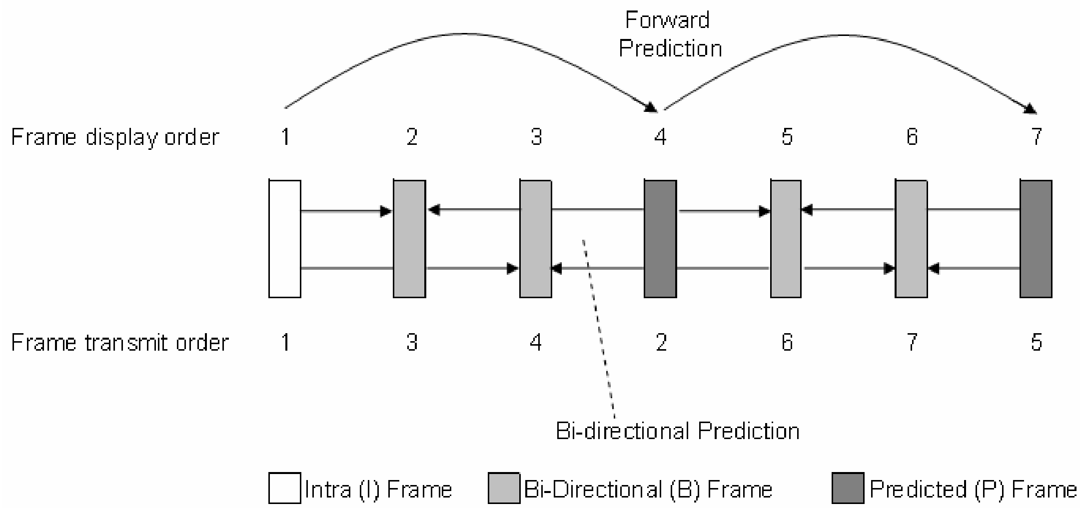


Figure 5.1 The base of MPEG. I frame is a reference frame, P frame is produced by forward prediction and B frame is produced by bi-directional prediction.

A group of pictures (GOP) is a series of one or more coded frames intended to assist in random accessing and editing. In the coded bitstream, a GOP must start with an I frame and may be followed by any number of I, P, or B frames in any order. In display order, a GOP must start with an I or B frame and end with an I or P frame. Thus, the smallest GOP size is a single I frame, with the largest size unlimited.

Motion compensation produces the future frame after I frame and improves compression of P and B frames by removing temporal redundancies between frames. The technique relies on the fact that within a short sequence of the same general image, most objects remain in the same location, while others move only a short distance. The motion is described as a two-dimensional motion vector that specifies where to retrieve a macroblock from a previously decoded frame to predict the sample values of the current macroblock.

After a macroblock has been compressed using motion compensation, it contains both the spatial difference (motion vectors) and content difference (error terms) between the reference macroblock and macroblock being coded.

Note that there are cases where information in a scene cannot be predicted from the previous scene, such as when a door opens. The previous scene doesn't contain the details of the area behind the door. In cases such as this, when a macroblock in a P frame cannot be represented by motion compensation, it is coded the same way as a macroblock in an I frame (using intra-picture coding). Macroblocks in B frames are coded using either the closest previous or future I or P frames as a reference, resulting in four possible codings:

- intra coding

  no motion compensation
- forward prediction

  closest previous I or P frame is the
  reference
- backward prediction

  closest future I or P frame is the
  reference
- bi-directional prediction

  two frames are used as the reference:
  the closest previous I or P frame and
  the closest future I or P frame

5.1.2 Motion Based Frame Interpolation In 100/120 Hz LCD TVs

There are three types of broadcast systems: PAL, SECAM and NTSC. PAL and SECAM have a vertical frequency of 50 Hz whereas NTSC has 60 Hz. There are also other main differences such as luminance bandwidth, chrominance bandwidth, resolution, etc.

The question is where these numbers come from? Film cameras work at a rate of 25 frames/second. This format is converted to 50 / 60 Hz according to the broadcast system. The reason of number 50 / 60 is power transmission system.

Problems of 50 / 60 Hz are motion judder (panning camera movements), discontinuity motion and smoother picture during motion. The third one is also related with MPEG itself and the solution of all these problems lie on the 100 / 120 Hz systems.

Motion judder is a shaking, wobbling or jerky effect in a video image (Figure 5.2). Discontinuity motion can be solved by studying on motion vectors and calculating the interval frame with the help of motion judder reduction (Figure 5.2). To get rid of motion blur, calculating based on motion vectors steps in the processing too.

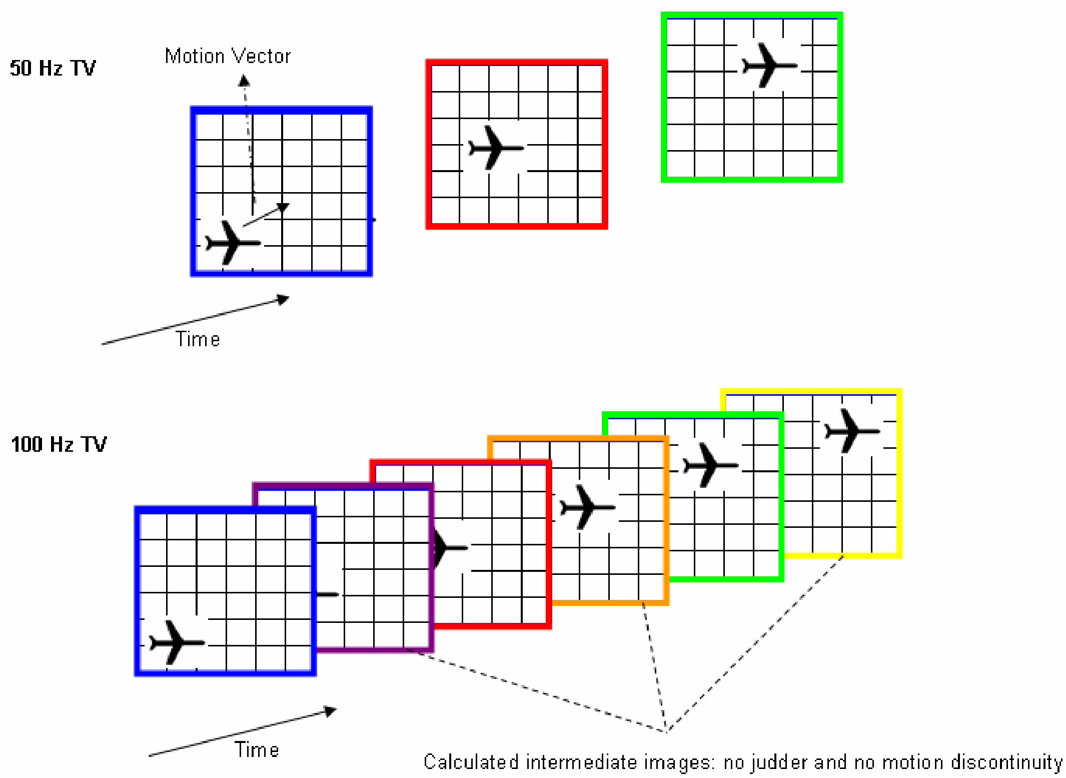


Figure 5.2 Calculated intermediate frames enhances the motion performance in a video sequence.

5.2 Motion Based Frame Interpolation

The motion vectors found previously are used to produce P&B frames in the MPEG and intermediate frames in 100/120 Hz LCD TV application for the purpose of increasing the frame rate, also called frame rate conversion (FRC). The main idea is very similar; move the macroblocks as motion vectors specify and reproduce the intermediate frame. But there are also slight differences and we are going to focus on 100/120 Hz LCD TV application, because I worked on it experimentally.

There are now quite a few motion compensated products on the market and all of them use similar ways. The basic principle of motion compensation is quite simple. In the case of a moving object, it appears in different places in successive source fields. Motion compensation computes where the object will be in an intermediate target field and then shifts the object to that position in each of the source fields (Figure 5.3).

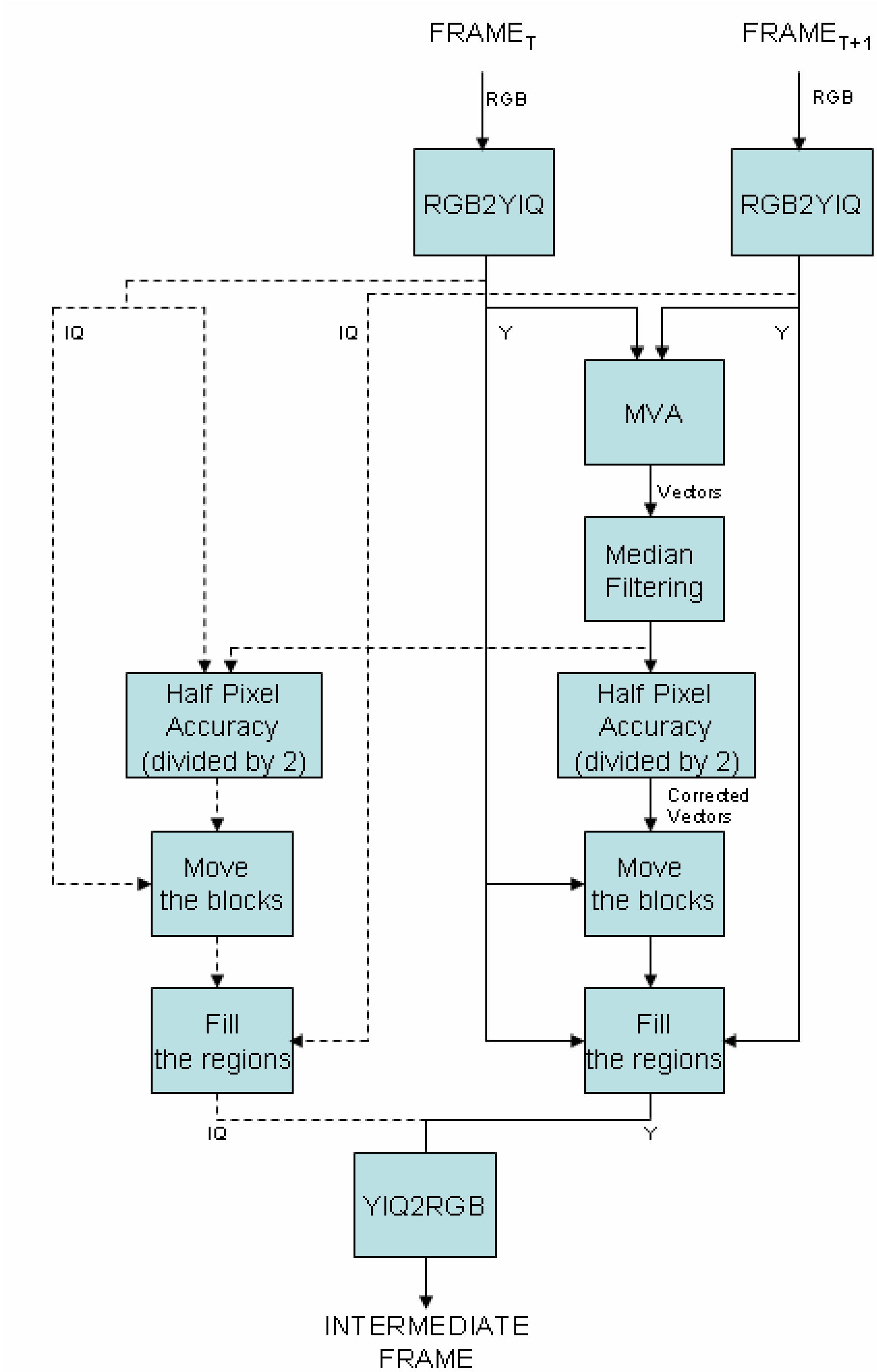


Figure 5.3 The base and common steps of motion based frame interpolation in the 100/120 Hz TV applications. MVA refers to motion vector analysis.

5.2.1 Converting RGB To YUV Form

First of all, the video is converted to YIQ form. If it is already in the YIQ form, it is OK. Then, we use on the Y (Luminance) data of the video. The I&Q (Chrominance) data is adjusted or used according the luminance motion data.

5.2.2 Motion Vector Analysis

By using any of previous motion estimation algorithms, the motion vectors are found by using blocks of 8x8 pixels as shown in the figure 5.4.

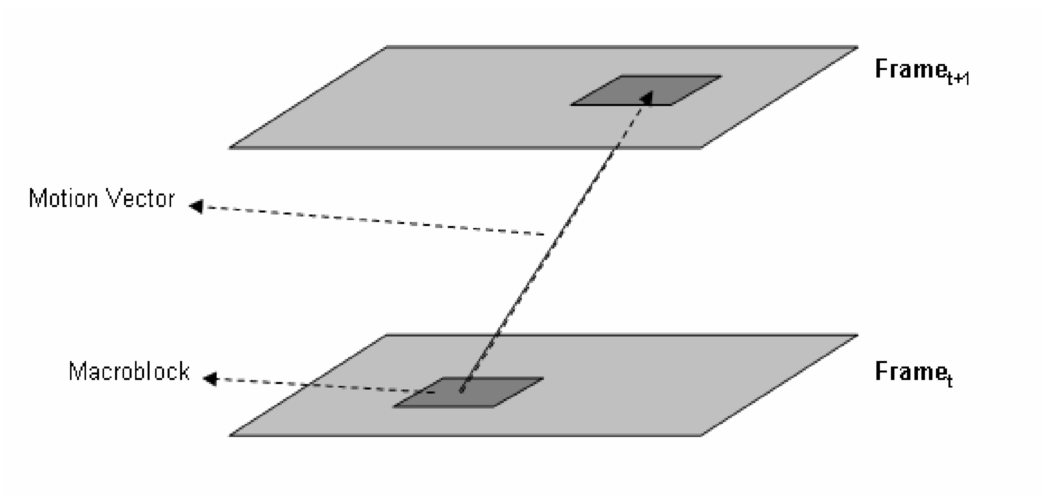


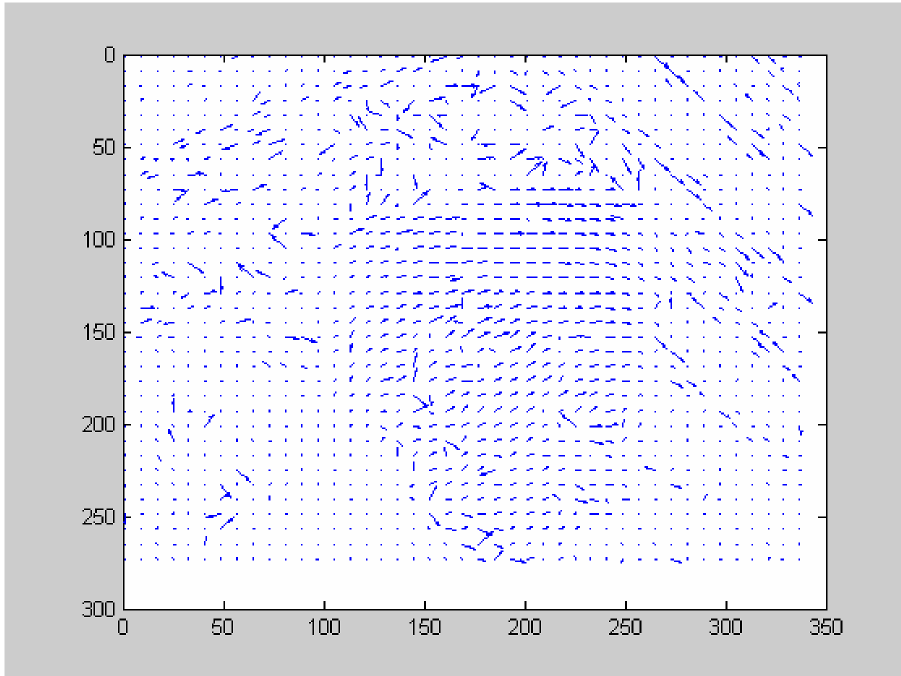
Figure 5.4 The motion vector which is determined by using a motion estimation algorithms

5.2.3 Median Filtering

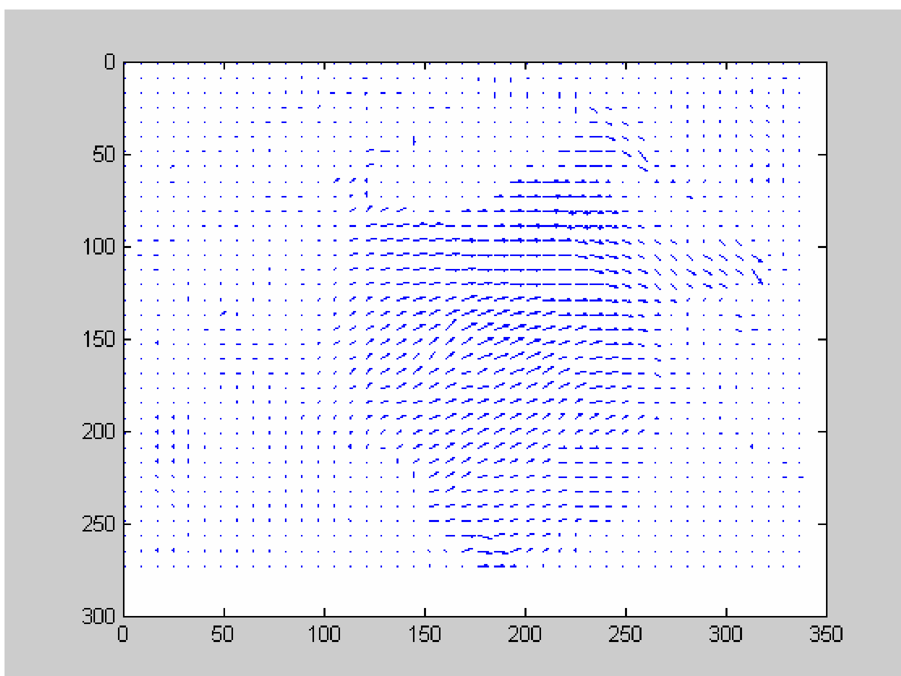
There is an assumption about the motion of an object: The motion vector of a macroblock is very similar to the motion vector of neighbor macroblocks which are on the same object. But, there may be some extreme, undesired and unexpected vectors due to noise or smooth transitions. So the median filter for each direction (x and y) is applied. (Figure 5.5)

(a)

(b)



(c)



(d)

Figure 5.5 (a) The first frame, (b) the next frame, (c) traditional motion estimation results, (d) after applying median filter.

The following steps are also valid for chrominance data.

5.2.4 Half Pixel Accuracy& Move The Blocks

The corrected motion vectors are divided by 2 to move the macroblock to build the intermediate frame. Basically, the macroblocks are moved as half-motion vector and put on an empty frame as shown in the figure 5.6.

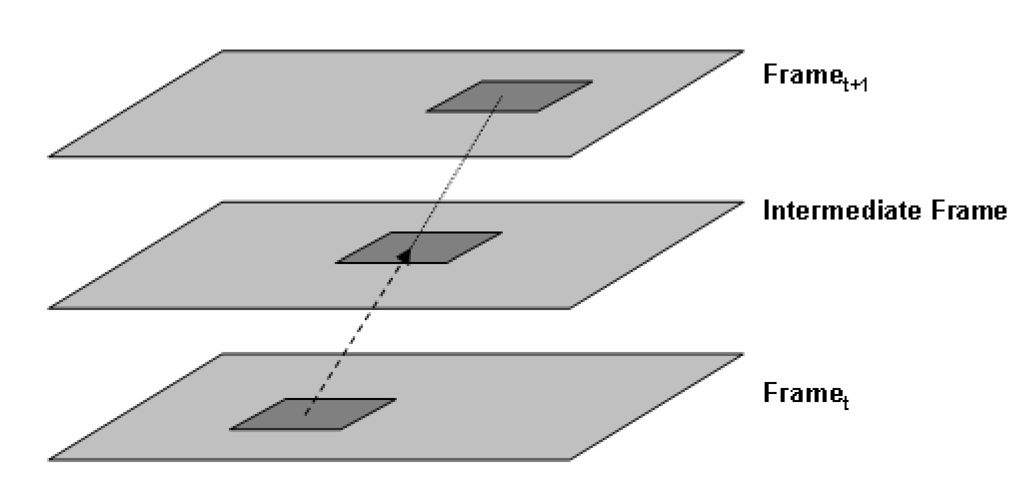


Figure 5.6 Producing the intermediate frame by moving the block as half-motion vector.

But what happens if the motion vector is an odd number? Dividing it by 2 gives a fractional number as 7/2=3.5. And the macroblock can't be moved as fractional number of pixels cause of integer matrix system. Rounding process can create some troubles. So we should use interpolation techniques, called half pixel accuracy.

At half pixel accuracy processing, the first image is interpolated by 2 in each dimensions. Then the original vectors (not divided vectors) are used to move the macroblocks into the appropriate locations. After that, the builded frame is downsampled by 0.5.

5.2.5 Fill The Region

During motion compensation, most probably there occurs some problems called as *uncovered region* and *lost region*.

5.2.5.1 Uncovered Region

If there are two neighboring macroblocks which separates from each other or move opposite directions, there occurs uncovered region.

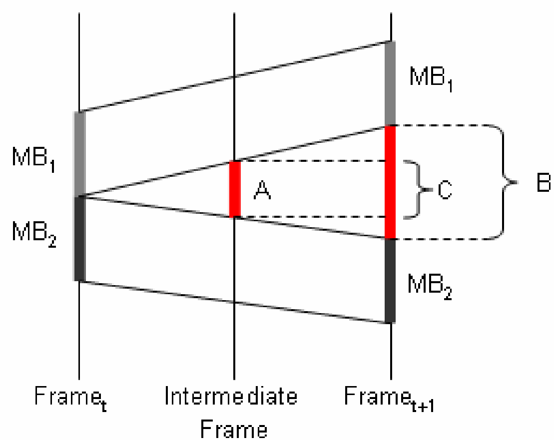


Figure 5.7 A is uncovered an uncovered region.

In the above figure 5.7, A is an uncovered region. To fill the region A, there is one basic and more preferred method cause of low computational load, directly copy region C to region A. Another advanced but not a favorable method is to downsample the region B and then paste on the region A. Mostly, region A creates a **Halo Effect**.

If region A is an unique line, it can be filled up by interpolating using the neighbor pixels as shown in the figure 5.8. The value of pixel Y can be filled up by average of its neighbors X. This reduces the computational load and doesn't create as much artifacts.

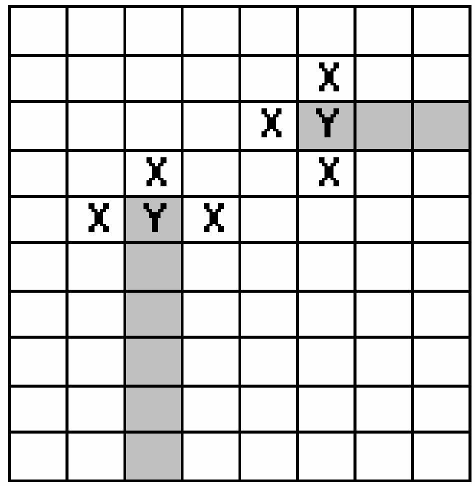


Figure 5.8. The white pixels refer the pixels already filled by a motion vectors and the shaded pixels refers the region A.

5.2.5.2 Lost Region

If two separate macroblocks moves and come together side by side in the frame$_{t+1}$, there occurs lost region.

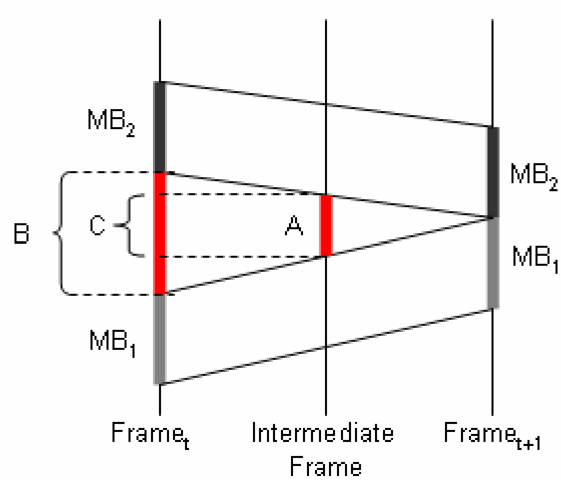


Figure 5.9 A is a lost region cause of two frames come together side by side.

In the figure 5.9, A is a lost region. To fill the region A, there is one basic and more favorable method cause of low computational load, directly copy region C to region A. Another advanced but not a favorable method is to downsample region B and then paste on the region A.

Owing to above steps, an intermediate frame can be produced. So, a broadcast or a video with 50Hz/60Hz can be upconverted to 100/120 Hz, respectively. This technology removes the motion judder and the motion discontinuity, providing a fluid motion in a video sequence.

CHAPTER SIX

APPLICATION

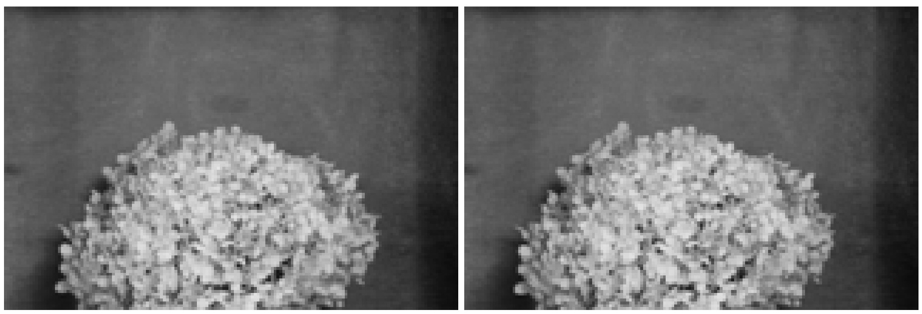
5.2 Application Methods

There are twenty pairs of picture which are focused in order to process block matching motion estimation, differential-gradient based motion estimation and phase correlation motion estimation, as shown in the figure 6.1. These pictures have many types of motion as global, local, diagonal, vertical, horizontal, zooming etc. So we will have tested the performance of all algorithms according to the different motion types with my own codes.

(a) Dimetrodon

(b) Street

(c) Tree

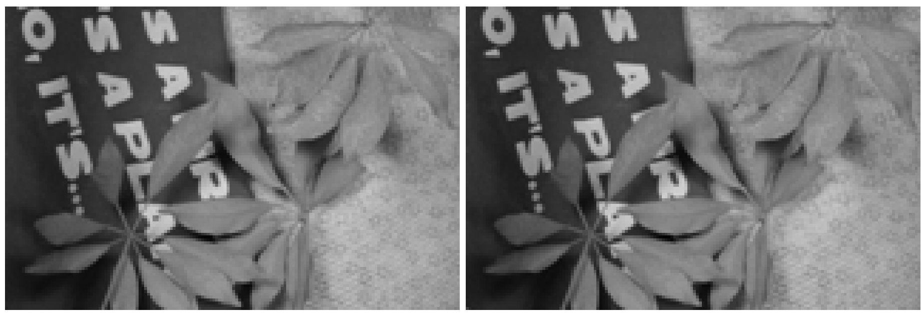


(d) Hydrangea

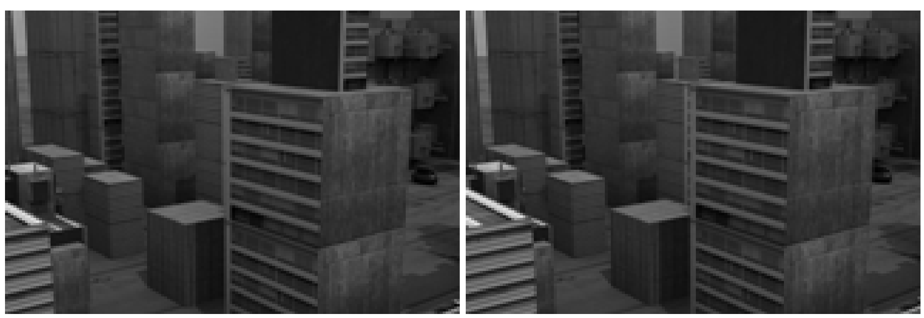
(e) Mequon

(f) MiniCooper

(g) RubberWhale



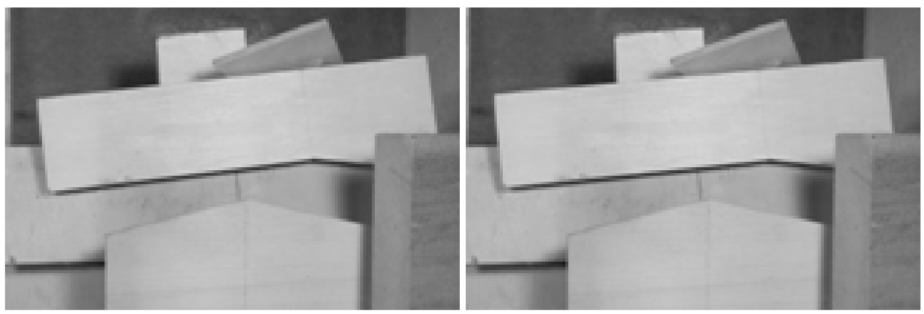
(h) Schefflera



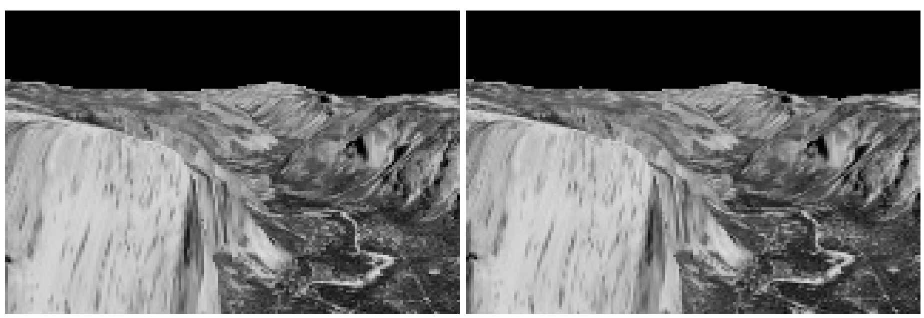
(i) Urban

(j) Venus

(k) Walking



(l) Wooden



(m) Yosemite

(n) Buildingman

(o)Village

(p) Window

(r) Future

(s) Snow

(t) Tractor

(u) Town

Figure 6.1 The application picture sequences. The sequences includes many types of motion.

I used ;

- a macroblock with 8x8 pixels and search area with 7 pixels for block matching algorithms.
- two macroblocks with 24x24 pixels and 32x32 pixels for phase correlation algorithms except hierarchical phase correlation, but each macroblock slides 8 pixels at each step to compare with block matching algorithms.
- a macroblock with 16x16 pixels for differential-gradient based algorithms, but each macroblock slides 8 pixels at each step to compare with the previous algorithms.

Peak-Signal-to-Noise-Ratio (PSNR) given by equation 6.1 characterizes the motion compensated image that is created by using motion vectors and macro clocks from the reference frame. It is computed between predicted second frame, which is created by moving macroblocks with their motion vectors from first frame, and original second frame.

$$PSNR = 10.\log_{10}\left[\frac{(peak\_to\_peak\_value\_of\_original\_data)^2}{MSE}\right]dB \qquad \text{Equation 6.1}$$

One of the biggest effects, which help to decide whether the algorithm is better or not, is computational load. You will see computational load of block matching algorithms as the number of comparison. It is also available to compare the other algorithms as duration of time. I used a desktop PC with Microsoft Windows XP, 2.8 GHz CPU, 768 Mb RAM and Matlab 7.0.4, and all programs, except Matlab, are switched off.

As we see before, the motion vectors are also sent according to the VLC. This means if there are many types of motion vectors, we need more bits to send or storage the video. This leads us to the term of entropy. A system has an entropy as high as its disorder. We can measure the entropy by using histogram (Figure 6.2) of motion vectors for each x and y components and then sum of their entropy result (Equation 6.2) gives the last entropy result.

$$E = -sum(p * \log(p)) \qquad \text{Equation 6.2}$$

In the equation 6.2, E is a scalar value and p contains the possibility of vectors returned from the figure 6.2.

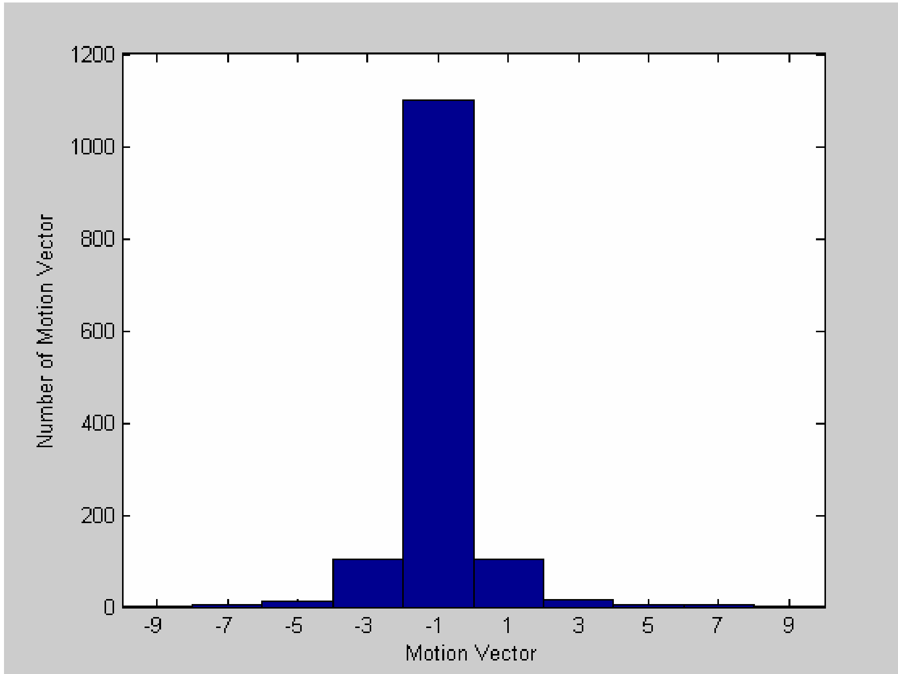


Figure 6.2. An example of histogram of motion vectors used to calculate the entropy.

Motion based frame interpolation is also worked by using some video within the form of avi. To show a video 100 frame/second is only available by using a panel with 100 Hz and also it takes a long time to compute by using an usual computer. So I studied on a video with 25 fps and get the output video with 50 fps. This way also can show the similar results and it is also enough to make a comment. As a result, the motion based frame interpolated video may be assumed as 100 Hz and the original video may be assumed as 50 Hz. It is important to show the output video that the display device and the output of display adapters must be set to 50 Hz not to see additional motion judder. I am going to use a LCD TV which is concordant to both 50 and 60 Hz and a display adapter which has DVI output. DVI output of a display adapter is not like its VGA output. DVI output doesn’t change the frame rate of video and show however it is. On the other hand, the full search block matching algorithm is used to find the motion vectors.

CHAPTER SEVEN

RESULTS

7.1 Block Matching Results

To avoid using up too much space, I only give the visual results of two sequences, B*uildingman* and *Hydrangea.*

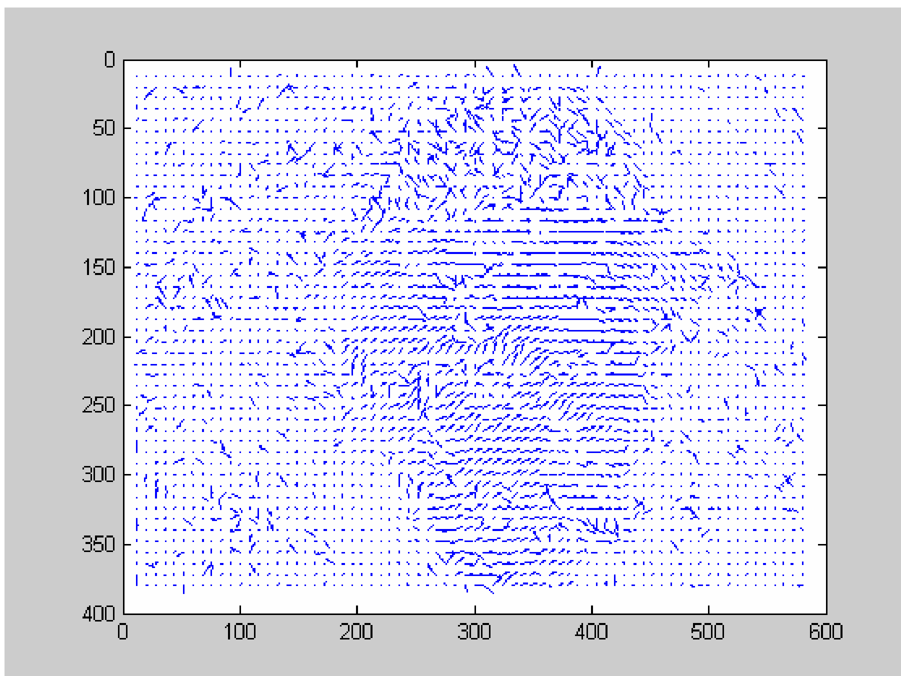


Figure 7.1 Full search result and motion vectors for Buildingman sequence.

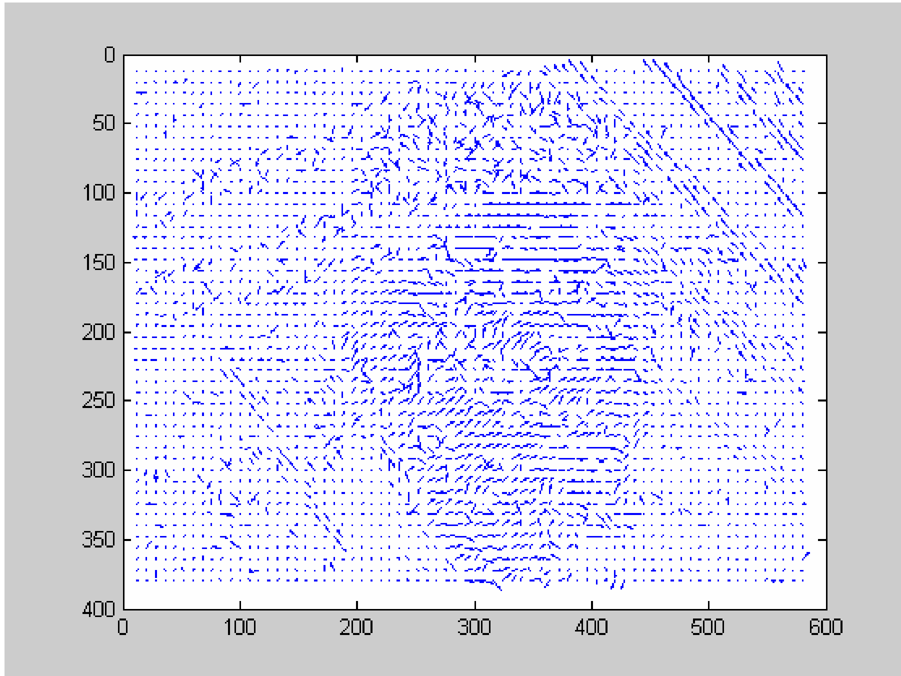


Figure 7.2 Three step search result and motion vectors for Buildingman sequence.

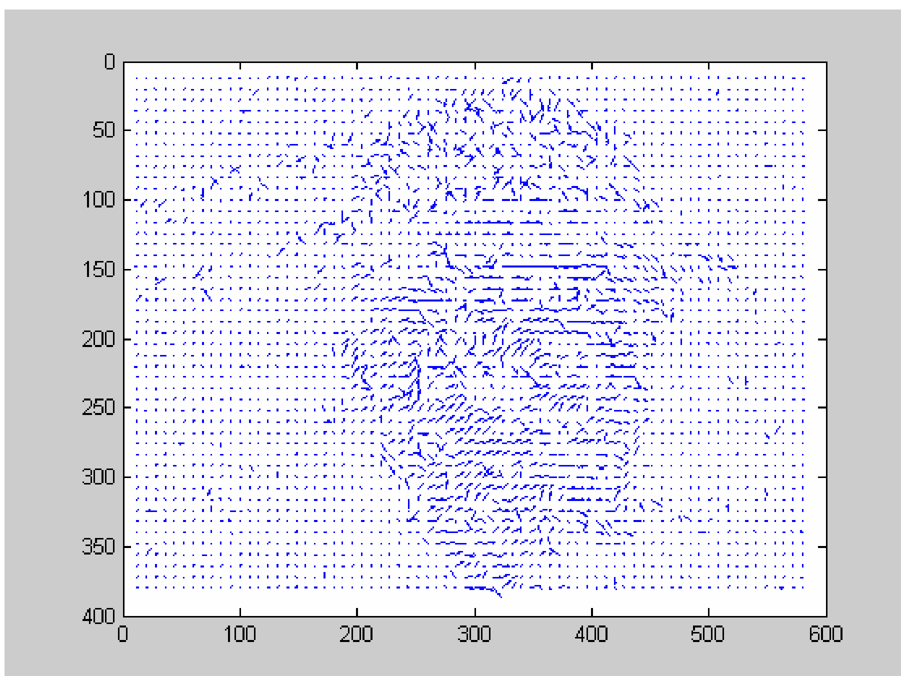


Figure 7.3 New three step search result and motion vectors for Buildingman sequence.

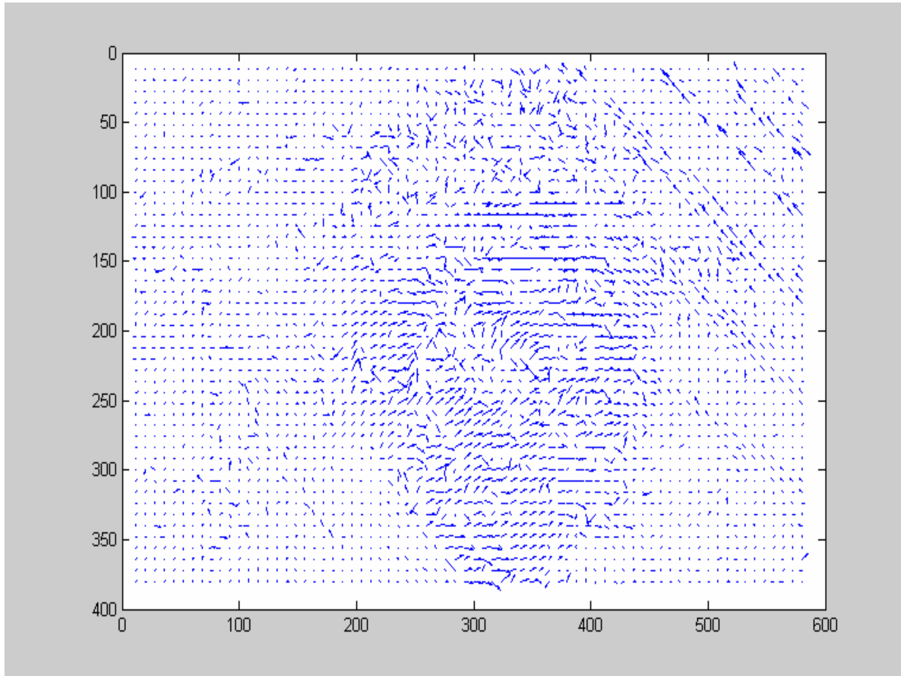


Figure 7.4 Four step search result and motion vectors for Buildingman sequence.

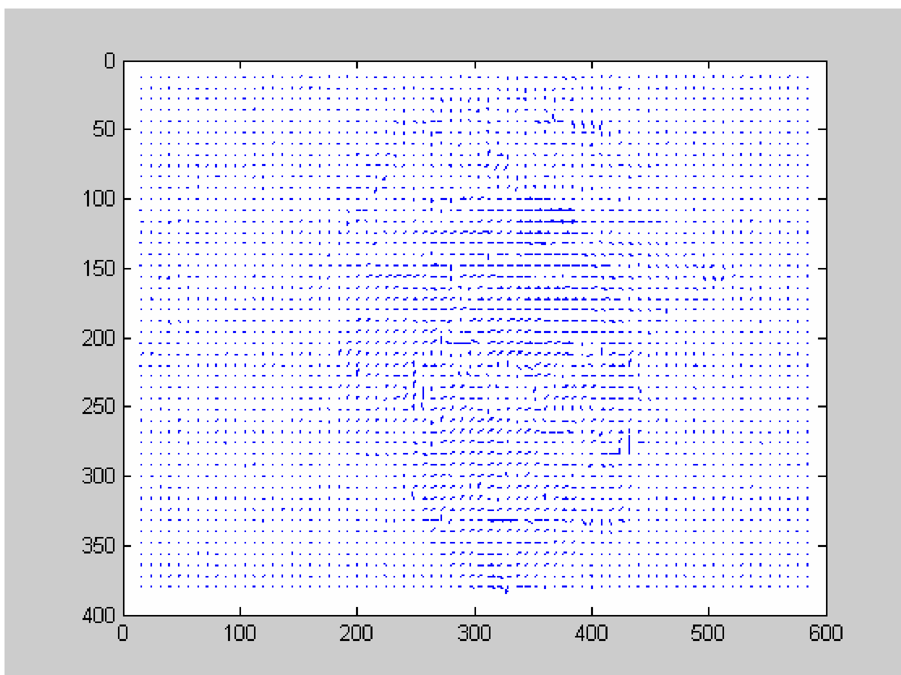


Figure 7.5 Adaptive Rood Pattern Search result and motion vectors for Buildingman sequence.

Table 7.1 and figure 7.6 represent the PSNR results of pairs of picture for each block matching method.

Table 7.1 PSNR results of pairs of picture for each block matching method.

#	Images	FS	TSS	NTSS	FSS	ARPS
1	Dimetrodon	31,034	25,808	26,997	27,659	29,884
2	Street	21,871	20,050	20,135	20,235	21,126
3	Tree	21,708	19,668	19,571	20,048	21,048
4	Hydrangea	23,825	21,537	21,540	21,935	22,323
5	Mequon	23,421	21,157	21,007	20,727	21,479
6	MiniCooper	21,848	18,371	18,355	18,904	19,885
7	RubberWhale	30,264	26,691	28,046	28,106	28,223
8	Schefflera	25,231	22,368	22,490	22,679	23,040
9	Urban	20,962	19,083	19,923	20,188	20,586
10	Venus	21,686	19,946	19,875	20,015	20,551
11	Walking	26,947	23,741	23,977	24,276	25,243
12	Wooden	26,560	23,627	23,559	24,591	25,351
13	Yosemite	25,628	23,047	23,762	23,987	24,567
14	Buildingman	29,572	26,981	27,554	28,253	28,736
15	Village	16,099	14,657	14,600	15,163	15,631
16	Window	20,721	19,234	19,441	19,976	20,348
17	Future	28,386	26,266	26,205	26,760	27,573
18	Snow	15,498	13,844	13,712	14,201	14,850
19	Tractor	29,969	27,716	27,909	29,467	29,718
20	Town	22,598	22,136	21,997	21,880	22,239

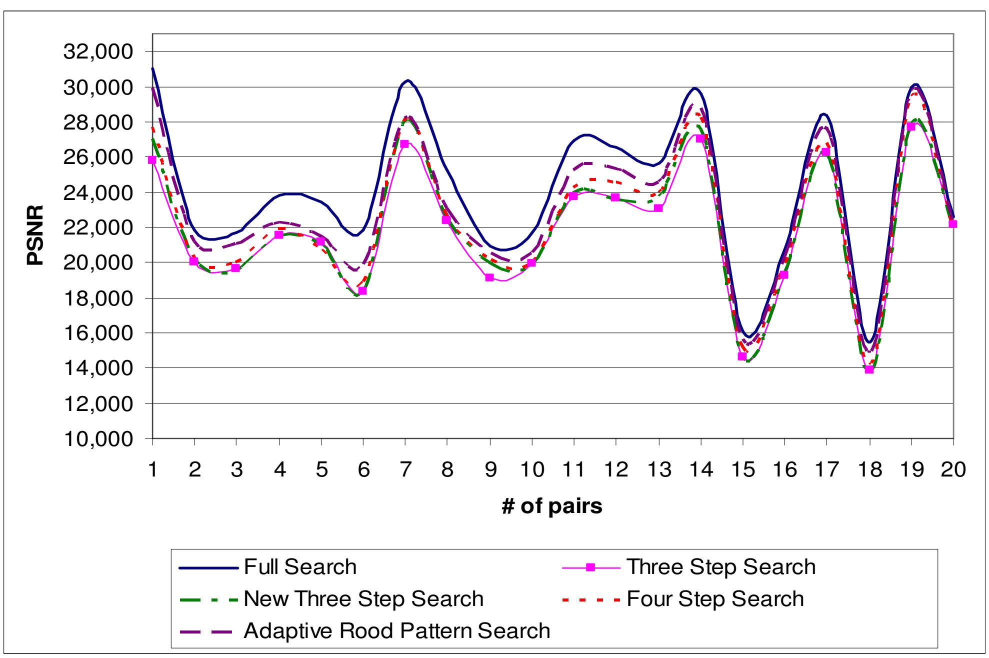


Figure 7.6 The PSNR results of each block matching method for twenty pairs of pictures.

Table 7.2 and figure 7.7 represents the computational load of each block matching method. The method for measuring computational load is the calculating the number of comparison.

Table 7.2 Computational load results of pairs for each block matching results.

#	Images	FS	TSS	NTSS	FSS	ARPS
1	Dimetrodon	761400	91368	73350	62106	41625
2	Street	761400	91368	65016	59228	42700
3	Tree	761400	91368	74286	69317	48720
4	Hydrangea	761400	91368	112086	87168	43900
5	Mequon	761400	91368	105660	85810	58500
6	MiniCooper	761400	91368	69516	58460	44460
7	RubberWhale	761400	91368	63288	55726	40370
8	Schefflera	761400	91368	94176	85179	50872
9	Urban	761400	91368	90432	74178	49002
10	Venus	761400	91368	108126	86848	55215
11	Walking	761400	91368	78066	73033	50376
12	Wooden	761400	91368	83862	66795	50025
13	Yosemite	761400	91368	78876	77318	45745
14	Buildingman	761400	91368	79704	68972	53775
15	Village	761400	91368	113292	84111	59254
16	Window	761400	91368	88974	80261	55116
17	Future	761400	91368	75096	73779	50405
18	Snow	761400	91368	105704	80124	58407
19	Tractor	761400	91368	77436	71238	47445
20	Town	761400	91368	114354	68774	47887

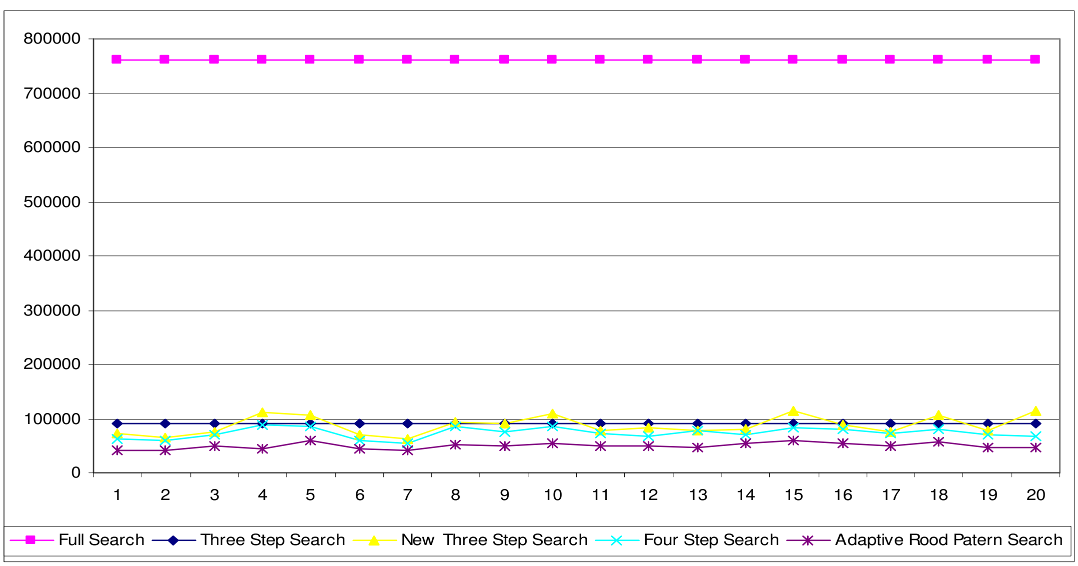


Figure 7.7 The computational load results of each block matching method for picture sequences.

Table 7.3 and figure 7.8 represents the entropy results of each block matching method.

Table 7.3 The entropy results of pairs for each block matching results.

#	Images	FS	TSS	NTSS	FSS	ARPS
1	Dimetrodon	3,233	5,217	3,092	3,945	2,076
2	Street	1,471	1,988	1,051	1,467	0,562
3	Tree	3,174	3,332	2,714	3,191	1,722
4	Hydrangea	2,775	2,951	2,681	3,630	2,611
5	Mequon	5,359	5,564	5,213	5,492	3,604
6	MiniCooper	2,581	3,046	2,247	2,437	1,998
7	RubberWhale	2,526	3,870	1,909	2,696	3,010
8	Schefflera	5,376	5,706	4,991	5,301	4,010
9	Urban	5,349	5,761	5,053	5,483	2,974
10	Venus	4,575	5,166	4,942	5,122	4,900
11	Walking	5,528	5,370	4,048	4,883	2,965
12	Wooden	5,142	5,469	4,428	4,893	3,335
13	Yosemite	4,523	5,248	3,565	4,241	3,311
14	Buildingman	4,860	5,074	3,864	4,740	2,814
15	Village	5,494	6,121	6,060	6,273	5,251
16	Window	5,542	5,614	4,965	5,500	3,243
17	Future	4,298	4,345	3,320	3,900	3,374
18	Snow	5,220	5,665	5,435	5,777	4,122
19	Tractor	4,771	5,092	3,666	4,735	3,660
20	Town	4,273	4,155	3,992	4,276	2,791

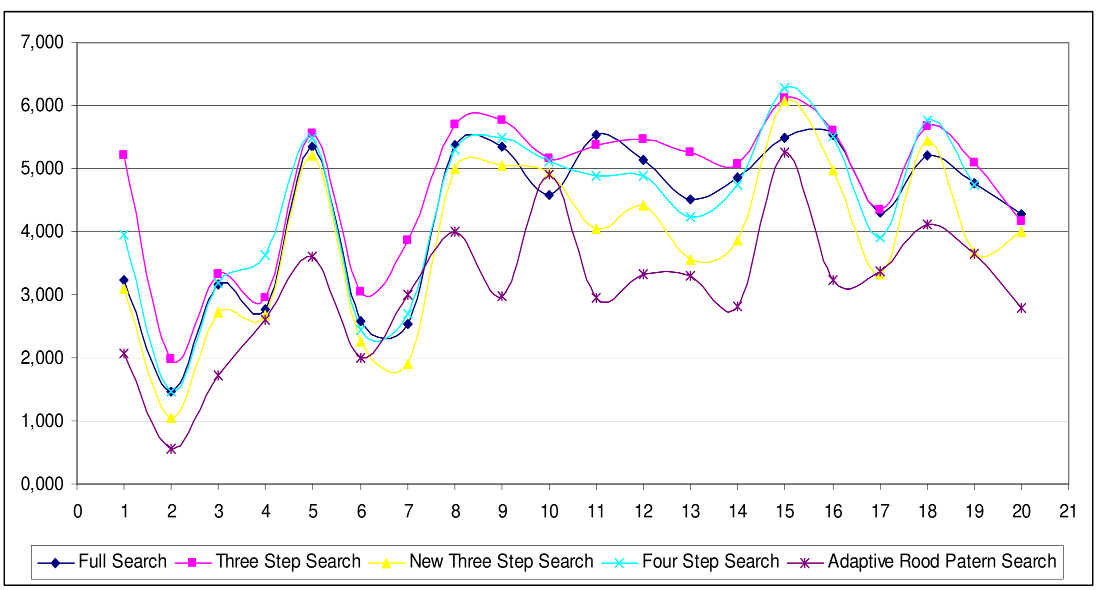


Figure 7.8 The entropy results of pairs for each block matching results.

7.2. Phase Correlation Results

Figure 7.9 shows the phase correlation results with block of 8x8 pixels, Kaiser window and half pixel accuracy for Hydrangea sequence. Figure 7.10 and figure 7.11 represents the hierarchical phase correlation results with a fixed threshold of 0.5 and adaptive hierarchical phase correlation results for Hydrangea sequence, respectively.

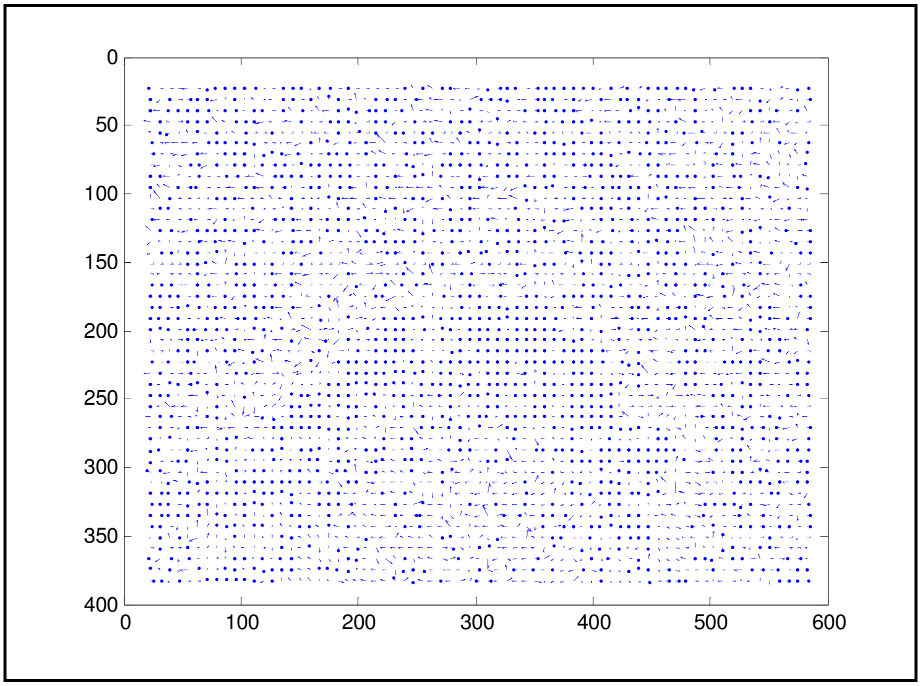


Figure 7.9 Phase correlation results with 8x8 block, Kaiser window and half pixel accuracy for Hydrangea sequence.

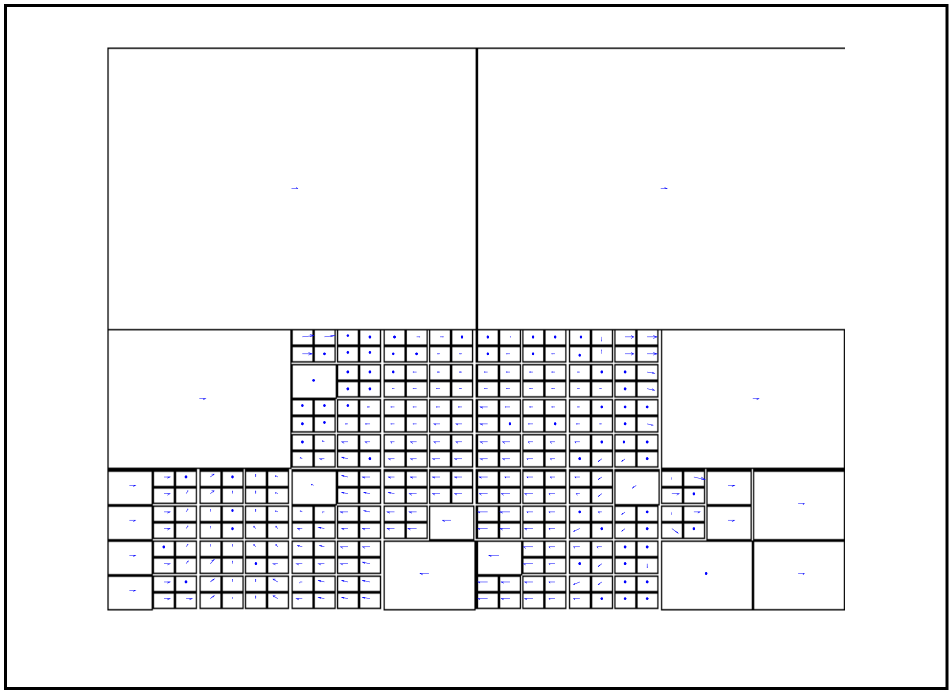


Figure 7.10 Hierarchical phase correlation results with a fixed threshold of 0.5 for Hydrangea sequence.

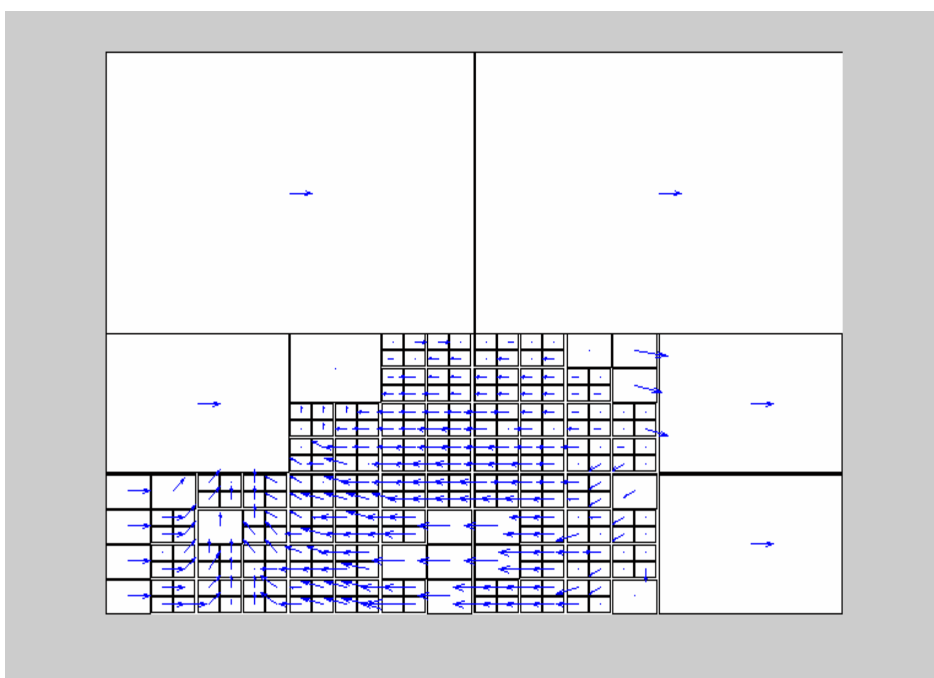


Figure 7.11 Adaptive hierarchical phase correlation result for the Hydrangea sequence.

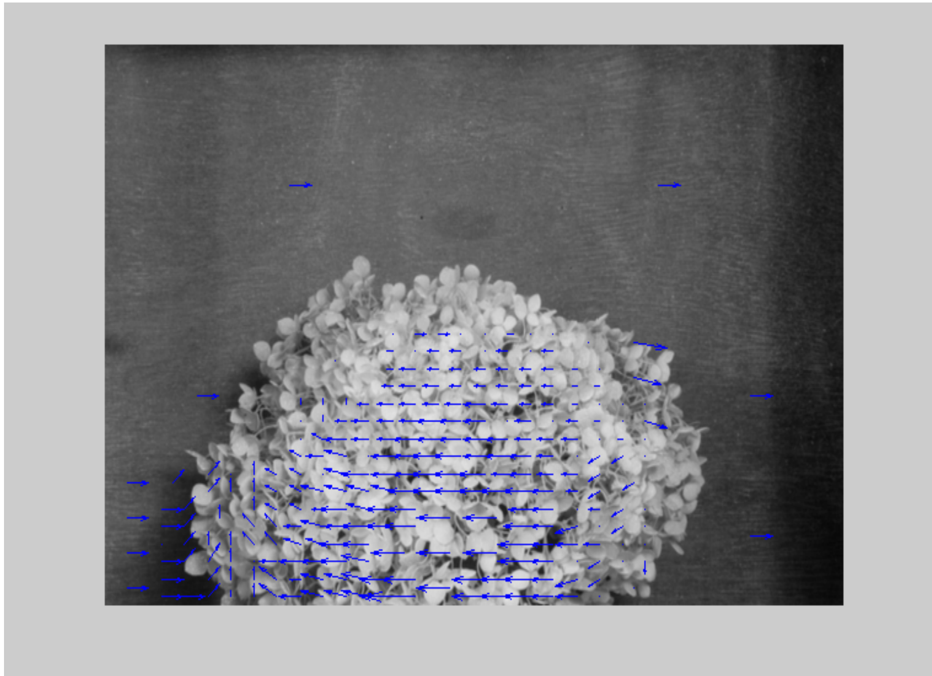


Figure 7.12 Adaptive hierarchical phase correlation result for the Hydrangea sequence.

Table 7.4 represents the PSNR performance of many types of phase correlation methods.

Table 7.4 The PSNR results of pairs for different phase correlation methods. PC means “phase correlation”, FPC means “phase correlation with fixed block size” and ME means “motion estimation”.

#	Images	FPC		8x8 FPC + Window		8x8 FPC Window ME	Hierarchical PC	
		8x8	16x8	Triangular	Kaiser		Thr=0.5	Adaptive Thr.
1	Dimetrodon	24.380	25.056	24.452	24.827	25.521	28.913	28,959
2	Street	19.154	19.652	19.239	19.296	19.633	19.899	19,975
3	Tree	18.429	18.144	18.603	18.442	19.123	19.417	19,505
4	Hydrangea	19.091	19.144	19.091	19.370	19.431	21.559	21,367
5	Mequon	18.090	18.266	18.140	18.414	19.036	19.979	20,723
6	MiniCooper	14.870	16.815	15.028	15.135	15.708	16.016	16,542
7	RubberWhale	24.217	24.149	24.381	24.716	25.357	25.489	25,735
8	Schefflera	18.996	19.783	19.111	19.000	19.462	20.069	20,908
9	Urban	18.136	18.548	18.241	18.657	19.309	19.169	19,173
10	Venus	17.508	17.867	17.640	17.855	18.179	19.340	20,521
11	Walking	20.027	20.512	20.066	20.513	20.669	21.206	21,301
12	Wooden	22.350	22.910	22.468	22.427	23.072	21.078	23,379
13	Yosemite	19.427	20.660	19.519	19.493	20.119	21.830	21,870
14	Buildingman	24.586	24.979	24.782	25.282	25.533	26.062	25,007
15	Village	14.137	14.581	14.193	14.750	14.935	15.531	15,643
16	Window	18.257	18.660	18.418	18.655	18.744	17.613	18,032
17	Future	24.077	24.416	24.147	24.469	25.075	24.947	24,965
18	Snow	11.350	11.728	11.540	11.776	11.980	11.887	11,923
19	Tractor	24.186	24.582	24.217	24.728	25.325	25.882	25,902
20	Town	18.636	18.525	18.710	19.113	19.560	19.240	19,347

Table 7.5 and figure 7.13 show number of vector results of some phase correlation results. In these table and figure, less number of motion vector means less correlation results and less computational load.

Table 7.5 The number of motion vectors of pairs for some phase correlation results.

#	Images	Simple PC	Hierarchical Thrs.=0.5	Adaptive Hierarchical
1	Dimetrodon	3266	157	103
2	Street	3266	58	49
3	Tree	3266	4	4
4	Hydrangea	3266	295	284
5	Mequon	3266	817	676
6	MiniCooper	3266	4	4
7	RubberWhale	3266	532	523
8	Schefflera	3266	220	208
9	Urban	3266	808	763
10	Venus	3266	514	506
11	Walking	3266	454	427
12	Wooden	3266	505	502
13	Yosemite	3266	604	245
14	Buildingman	3266	226	47
15	Village	3266	937	747
16	Window	3266	301	208
17	Future	3266	373	256
18	Snow	3266	613	568
19	Tractor	3266	697	268
20	Town	3266	793	757

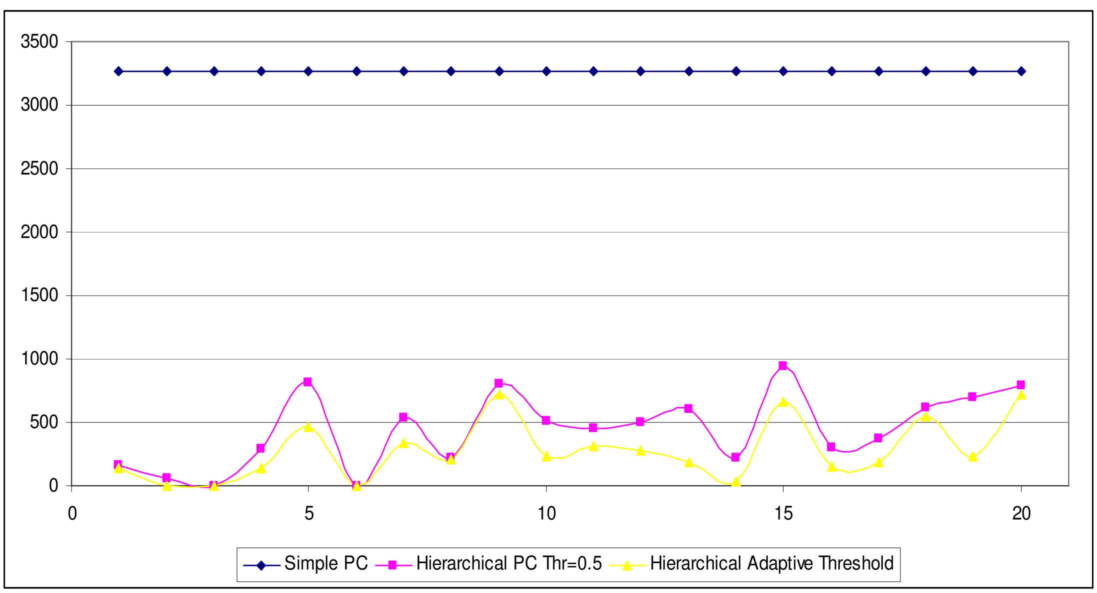


Figure 7.13 The number of motion vectors of pairs for some phase correlation vectors. This figure also gives information about calculation numbers.

Table 7.6 and figure 7.14 present the entropy results of pairs between phase correlation and block matching results. Smaller entropy means less necessary bit for motion vectors during transmission.

Table 7.6 The entropy results of pairs between phase correlation and block matching methods.

#	Images	Full Search	Adaptive Rood Pattern Search	Simple PC	Adaptive Hierarchical PC
1	Dimetrodon	3.233	2.076	2.042	1.062
2	Street	1.471	0.562	0.262	0.162
3	Tree	3.174	1.722	1.489	1.358
4	Hydrangea	2.775	2.611	2.238	2.163
5	Mequon	5.359	3.604	3.386	3.202
6	MiniCooper	2.581	1.998	1.071	0.569
7	RubberWhale	2.526	3.010	1.340	1.240
8	Schefflera	5.376	4.010	3.633	2.988
9	Urban	5.349	2.974	2.575	1.232
10	Venus	4.575	4.900	3.674	3.367
11	Walking	5.528	2.965	1.692	0.912
12	Wooden	5.142	3.335	3.240	1.826
13	Yosemite	4.523	3.311	1.880	1.022
14	Buildingman	4.860	2.814	1.681	0.181
15	Village	5.494	5.251	4.682	3.417
16	Window	5.542	3.243	3.092	3.050
17	Future	4.298	3.374	1.225	0.201
18	Snow	5.220	4.122	3.406	2.307
19	Tractor	4.771	3.660	1.740	0.504
20	Town	4.273	2.791	2.572	2.275

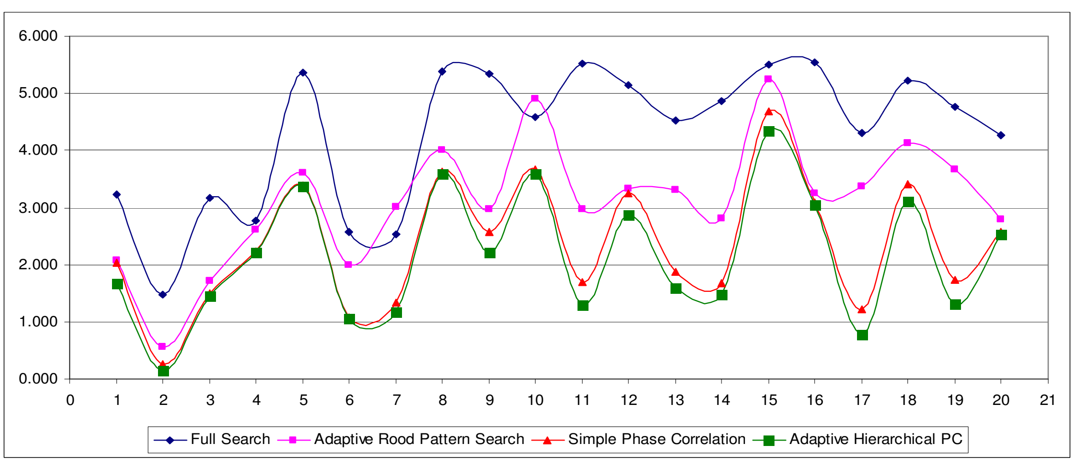


Figure 7.14 The entropy results of pairs between phase correlation and block matching methods.

7.3 Differential - Gradient Based Method Results

Table 7.7 and table 7.8 presents the PSNR performance of differential-gradient based method with fixed alpha value & increasing iteration and fixed iteration value & increasing alpha value, respectively.

Table 7.7 PSNR results of pairs with fixed alpha value and increasing iteration number.

#	Images	Alpha = 15		
		Iteration=10	Iteration=100	Iteration=190
1	Dimetrodon	25,191	26,138	26,234
2	Street	20,283	20,491	20,537
3	Tree	18,606	18,956	18,977
4	Hydrangea	18,619	18,888	18,897
5	Mequon	18,449	18,785	18,821
6	MiniCooper	16,962	16,977	16,965
7	RubberWhale	26,332	28,716	28,842
8	Schefflera	20,666	21,104	21,149
9	Urban	18,835	19,100	19,097
10	Venus	16,900	17,109	17,133
11	Walking	21,535	22,280	22,322
12	Wooden	21,981	22,409	22,445
13	Yosemite	21,487	22,155	22,221
14	Buildingman	26,366	27,186	27,414
15	Village	14,153	14,310	14,323
16	Window	18,612	19,187	19,184
17	Future	26,502	27,488	27,624
18	Snow	12,705	12,866	12,900
19	Tractor	26,568	29,096	29,610
20	Town	19,671	20,385	20,544

Table 7.8 PSNR results of pairs with fixed iteration and increasing alpha value.

#	Images	Iteration = 10				
		Alpha = 2	Alpha = 5	Alpha = 15	Alpha = 50	Alpha = 100
1	Dimetrodon	24.633	24.874	25.191	24.227	23.069
2	Street	19.794	19.939	20.283	20.382	20.283
3	Tree	18.353	18.519	18.606	18.156	17.632
4	Hydrangea	18.139	18.301	18.619	18.285	17.796
5	Mequon	17.768	18.042	18.449	18.095	17.402
6	MiniCooper	16.423	16.652	16.962	17.322	16.705
7	RubberWhale	27.067	27.177	26.332	24.921	24.398
8	Schefflera	20.115	20.425	20.666	20.253	19.576
9	Urban	18.662	18.703	18.835	18.605	18.267
10	Venus	16.372	16.515	16.900	17.019	16.573
11	Walking	20.977	21.118	21.535	21.505	20.554
12	Wooden	21.580	21.750	21.981	22.011	21.100
13	Yosemite	20.626	21.088	21.487	20.431	19.401
14	Buildingman	25.840	26.357	26.366	25.441	25.101
15	Village	13.993	14.083	14.153	13.922	13.642
16	Window	17.931	18.265	18.612	18.038	17.444
17	Future	25.963	26.349	26.502	25.257	24.429
18	Snow	12.139	12.415	12.705	12.725	12.288
19	Tractor	26.914	27.247	26.568	24.918	24.346
20	Town	18.691	18.994	19.671	18.407	17.140

Figure 7.15 presents the visual results of the best of gradient based method with alpha=15 and iteration=190 for Buildingman sequence.

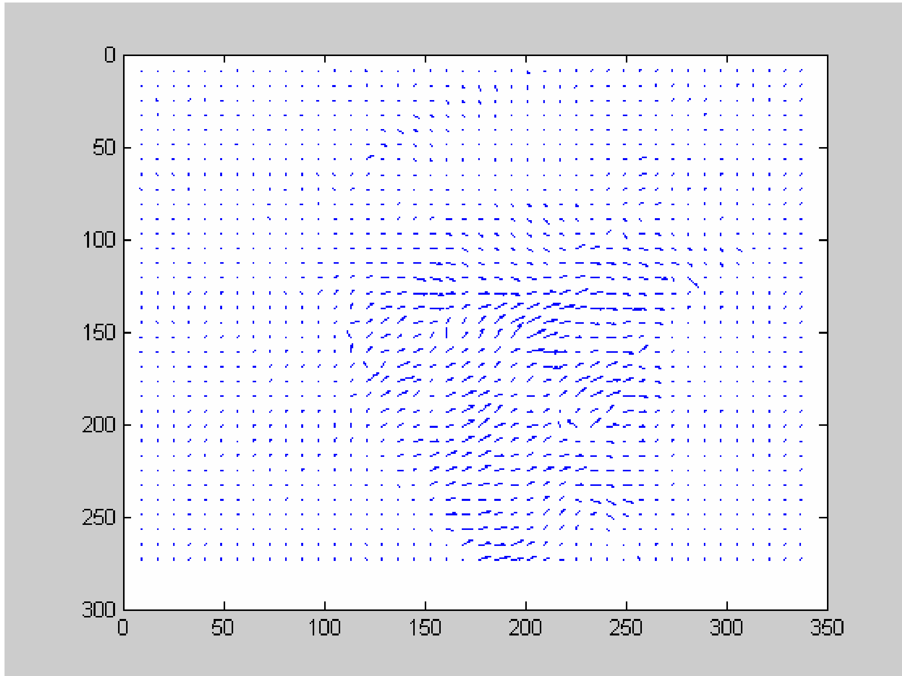


Figure 7.15 The best of gradient base method with alpha=15 and iteration=190 for Buildingman sequence.

Table 7.9 shows the PSNR performance of main or best methods mentioned before and table 7.10 shows the processing time for each method and for each sequences. It is measured as explained in the application chapter.

Table 7.9 The PSNR results of main or best methods.

Spatial Domain				Frequency Domain		
Block Matching		Gradient Based		Phase Correlation		
Full S.	A.R.P.S.	α=15 l=100	α=15 l=190	Simple PC 8x8	PC+W+ME	Adaptive Hierarchical
31,034	29,884	26,138	26,234	24,380	24,530	28,959
21,871	21,126	20,491	20,537	19,154	19,979	19,975
21,708	21,048	18,956	18,977	18,429	18,914	19,505
23,825	22,323	18,888	18,897	19,091	19,764	21,367
23,421	21,479	18,785	18,821	18,090	18,577	20,723
21,848	19,885	16,977	16,965	14,870	15,606	16,542
30,264	28,223	28,716	28,842	24,217	24,721	25,735
25,231	23,040	21,104	21,149	18,996	19,621	20,908
20,962	20,586	19,100	19,097	18,136	18,438	19,113
21,686	20,551	17,109	17,133	17,508	17,984	20,521
26,947	25,243	22,280	22,322	20,027	21,012	21,301
26,560	25,351	22,409	22,445	22,350	23,266	23,379
25,628	24,567	22,155	22,221	19,427	20,501	21,870
29,572	28,736	27,186	27,414	24,586	24,841	25,007
16,099	15,631	14,310	14,323	14,137	14,308	15,643
20,721	20,348	19,187	19,184	18,257	19,017	18,032
28,386	27,573	27,488	27,624	24,077	25,057	24,965
15,498	14,850	12,866	12,900	11,350	11,734	11,923
29,969	29,718	29,096	29,610	24,186	25,336	25,902
22,598	22,239	20,385	20,544	18,636	19,713	19,347

Table 7.10 The answer of question how long main or best methods take.

Spatial Domain				Frequency Domain		
Block Matching		Gradient Based		Phase Correlation		
Full S.	A.R.P.S.	α=15 I=100	α=15 I=190	Simple PC	PC+W+ME	Adaptive Hierarchical
12,984	1,141	9,109	17,625	10,922	14,234	3,384
12,313	1,031	9,188	17,625	10,484	13,578	1,572
12,828	1,031	9,078	17,625	10,641	13,578	1,584
12,938	1,016	9,109	17,656	10,547	13,500	1,629
12,938	1,031	9,172	17,578	10,578	13,719	3,056
11,984	1,031	9,094	17,594	10,391	13,422	1,678
12,891	1,031	9,094	17,547	10,656	13,516	2,183
12,547	1,031	9,188	17,609	10,672	13,672	1,615
12,813	1,016	9,094	17,609	10,672	13,594	2,490
12,953	1,016	9,125	17,625	10,656	13,578	1,615
12,844	1,031	9,188	17,609	10,625	13,594	2,602
12,938	1,016	9,094	17,656	10,641	13,531	2,387
12,937	1,031	9,063	17,594	10,614	13,521	1,670
12,938	1,031	9,203	17,641	10,781	13,672	1,614
12,969	1,047	9,109	17,609	10,594	13,609	2,178
12,969	1,031	9,094	17,625	10,516	13,531	2,199
12,984	1,047	9,188	17,641	10,609	13,609	1,708
12,938	1,031	9,109	17,609	10,609	13,578	2,312
12,969	1,047	9,141	17,578	10,656	13,563	1,685
12,781	1,016	9,188	17,641	10,781	13,641	6,701

CHAPTER EIGHT

CONCLUSION

8.1 Conclusion

Motion estimation is the processes which generates motion vectors that determine how an object moves from previous frame to current frame. They are in many applications, such as video technology, computer vision, tracking and industrial designs.

In this thesis, two main approaches to the motion estimation, spatial domain research and frequency domain research, have been investigated and several methods in these categories have been implemented. The methods include all well-known block matching algorithms, differential-gradient based techniques and phase correlation based methods. I have also developed and implemented a new hierarchical phase correlation method with adaptive threshold.

Twenty pairs of pictures have been used in order to evaluate all the implemented methods. Power Signal Noise Ratio (PSNR), computational load and entropy performances of each method are calculated and compared in detail.

The motion estimation techniques is also used in 100/120 Hz TV applications which double the input frame rate using motion based frame interpolation. I have designed and implemented an application for such frame rate conversion task. My application produces new intermediate frames after finding the motion vectors of blocks between successive frames. So there are twice as much frame and consequently this provides a judder-free and fluid motion in the video sequences.

In the application part of my project work, all of the motion estimation algorithms mentioned in this thesis have been implemented and we now come to a conclusion.

As shown by Table.1, four step search (FSS) is better than new three step search (NTSS) which is better than three step search. Adaptive rood pattern search (ARPS) comes pretty close to the PSNR results of full search (FS) and its PSNR performance is better than the previous three ones. According to the Table.2, except FS, all the other algorithms provide less and less computational load and ARPS is the best one when the computational load is concerned.

From calculated motion field entropy as shown in Table.3 and the motion vector field plots, we can see that the motion field given by ARPS is much smoother with much lower entropy. Except ARPS, the other fast algorithms don't give a big advantage about entropy. Their entropy performances are relatively the same as that of the FS. Less entropy means less number of bits are required for vector data during MPEG encoding. In this sense, ARPS is again the best block matching algorithm with its minimum vector data entropy, as well as with its best PSNR results and lowest computational load.

As to a frequency domain method, the Phase Correlation (PC), according to the Table.4, applying a window function provides better PSNR owing to less spectral leakage. On the other hand, its computational load is higher. Additionally, half pixel motion estimation gives better performance, but at the cost of more computational load. It is also found that the block size affects the PSNR performance. The blocks should be large enough to group pixels with similar motion and shouldn't be too large as to eliminate the different motions. In addition, the blocks should be small enough to separate pixels with different motion and multiple-object movement and shouldn't be too small as to result in inaccurate phase correlation calculations. So, the answers lies in the hierarchical phase correlation (HPC) with adaptive threshold (AHPC) which is the best one in the PC based methods. AHPC also removes the worries about choosing a suitable threshold value.

As shown by Table.5, AHPC provides the less computational load compared to other PC methods. According to Table.6, PC with fixed block size gives less entropy than the best block matching method, ARPS. As a result, AHPC is the best one in the phase correlation (PC) methods with the highest PSNR, the least computational load and the least number of motion vectors. Its performance can also be increased by using additional half pixel motion estimation.

The differential-gradient based method is the next. As shown by Table.6 with fixed alpha value and increasing iteration value, the PSNR performance increases nonlinearly by the larger number of iterations. For example, as the iteration number increases from 10 to 100, the PSNR value goes up approximately 1.2-3%. But, as the iteration number increases from 100 to 190, the PSNR value goes up less as it increases by 0.2-0.5%. The PSNR value becomes saturated for higher iteration numbers, suggesting that increasing the iteration number furthermore doesn't provide an advantage. According to Table.7, alpha=15 gives the best PSNR performance. Alpha shouldn't be too large as to dominate the motions of neighbors and noise and shouldn't be too small as to eliminate the motions of neighbors and noise.

As shown by Table.8, the PSNR values of FS and ARPS are better than simple PC algorithms. The best of the PC algorithms is AHPC and its PSNR performance is pretty close to that of ARPS. The PSNR performance of differential-gradient based method is better than that of the PC with fixed block size. According to Table.9, ARPS is the fastest method. The AHPC is also fast and its speed is very close to that of ARPS.

It is obvious that block matching methods are better for small motion between sequences. But if the motion is bigger than search area, it cannot detect it. For larger motion vectors the HPC becomes the best solution. PC methods are a little bit poorer for small motion, but generally better for large motion, especially for global motion.

Variation at DC value can cause some problem to detect motion. On the other hand, the immunity of the phase correlation algorithms against brightness and contrast changes between the sequences is very high. This property makes them more favorable than the block matching methods if sudden changes are expected in the brightness and contrast levels of video sequences.

At the future, a hybrid method can be devised by including both a block matching method and a phase correlation based method. If there is a global motion, the phase correlation based method can be used, otherwise the block matching method can be used.

The techniques of motion compensated interpolation for producing intermediate frame at 100/120 Hz video is becoming the most important issue of customer requirements in the TV markets. Because it provides very clear and fluid motion. But the nowadays technology needs to be improved more and more. It still uses frame repeat method for faster motion which is not easily compensated, and additionally there has been no exact solution for halo effect till now. To display the halo effect less, it is generally preferred to smooth the picture. Recently, instead of investigating only current and previous frames, the newer technology is focusing on the next frame additionally to get the whole answer of where the object comes from (previous frame), where it is right now (current frame) and where it is going (next frame). So the motion estimation for blocks is expected to be more accurate.

REFERENCES

Jain, J. R. & Jain, A. K. (1981). Displacement measurement and its application in interframe image coding. IEEE Trans. Commun. 29,1799–1808.

Koga, K., Iinuma, K., Hirano, A., Iijima, Y., & Ishiguro, T. (1981). Motion compensated interframe coding for video conferencing. Proc. NTC 81, C9.6.1-9.6.5.

Li, R., Zeng, B., & Liou, R. (August 1994). A new three step search algorithm for block motion estimation. IEEE Trans. Circuits And Systems For Video Technology, vol 4., no. 4, pp. 438-442.

Lu, J., & Liou, M. (April 1997). A simple and efficent search algorithm for block matching motion estimation. IEEE Trans.Circuits And Systems For Video Technology, vol 7, no. 2, pp. 429-433.

Po, L., & Ma, W. (June 1996). A novel four step search algorithm for fast block motion estimation. IEEE Trans. Circuits And Systems For Video Technology, vol 6, no. 3, pp. 313-317.

Zhu, S., & Ma, K. (February 2000). A new diamond search algorithm for fast block-matching motion estimation. IEEE Trans. Image Processing, vol 9, no. 2, pp. 287-290.

Nie, Y., & Ma, K. (December 2002). Adaptive rood pattern search for fast block matching motion estimation. IEEE Trans. Image Processing, vol 11, no. 12, pp. 1442-1448.

Cafforio, C., & Rocca, F. (September 1976). Methods for measuring small displacements of television images. IEEE Trans. Inform. Theory 22, 573–579.

Netravali, A. N., & Robbins, J. D. (1979). Motion-compensated television coding. Bell System Tech. J. 58, 631–670.

Horn, B. K. P., & Schunck, B. G. (1981). Determining optical flow. Artificial Intelligence. 17, 185–204.

Lucas, B. D., & Kanade, T. (1981) .An iterative image registration technique with an application to stereo vision. Proceedings of Imaging Understanding Workshop. 121-130.

Paquin, R., & Dubois, E. (1983). A spatio-temporal gradient method for estimating the displacement field in time-varying imagery. Computision Graphics Image Process. 21, 205–221.

Walker, D. R., & Rao, K. R. (1984). Improved pel-recursive motion estimation. IEEE Trans. Commun. 32, 1128–1134.

Biemond, J., Looijenga, L., Boekee, D. E., & Plompen, R. H. J. M. (1987). A pel-recursive wiener-based displacement estimation algorithm. Signal Process. 13, 399–412.

Yamaguchi, H. (1989). Iterative method of movement estimation for television signals. IEEE Trans. Commun. 37, 1350–1358.

Kuglin, C. D., & Hines, D. C. (1975). The phase correlation image alignment method in proceedings. IEEE International Conference on Cybernetics and Society, San Francisco, pp. 163–165.

Pearson, J. J., Hines, D. C., Goldman, S., & Kluging, C. D. (1977). Video rate image correlation processor. Proc. SPIE, Vol. 119, Application of Digital Image Processing.

Thomas, G. A. (1987). Television motion measurement for DATV and other applications. British Broadcasting Corporation Research Department Tech. Rep.

Thomas, G. A. (1987). HDTV bandwidth reduction by adaptive subsampling and motion-compensated DATV techniques. Soc. Motion Picture Television Eng. J., 460–465.

Strobach, P. (1990). Tree structured scene adaptive coder. IEEE Trans. Comm., vol. 38.

Nicolas, H., & Labit, C. (1992). Region-based motion estimation using deterministic relaxation schemes for image sequence coding. Proc. ICASSP, vol. 3, pp. 265-268.

Jensen, K., & Anastassiou, D. (April 1993). Digitally assisted deinterlacing for EDTV. IEEE Trans. Circ. Syst. Video Tech.,vol. 3, no. 2, pp. 99-106.

Banham, M. R., Brailean, J. C., Chan, C. L., & Katsaggelos, A. K. (September 1994). Low bit rate video coding using robust motion vector regeneration in the decoder. IEEE Trans. Image Proc., vol. 3, pp. 652-665.

Lee, J. (October 1995). Optimal quadtree for variable block size motion estimation. Proc. ICIP, vol. 3, pp. 480-483.

Schuster, G. M., & Katsaggelos, A. K. (November 1998). An optimal quadtree based motion estimation and motioncompensated interpolation scheme for video compression. IEEE Trans.Image Proc., vol. 7, no. 11, pp. 1505-1523.

Argyriou, V., & Vlachos, T. (2005). Motion estimation using quad-tree phase correlation. IEEE Trans. Commun.

Scharstein, D. (2008) Optical Flow. from http://vision.middlebury.edu/flow/

Kangas, M. (March 2002). MPEG-1 & MPEG-2 Compression.from http://mia.ece.uic.edu/~papers/WWW/MultimediaStandards/

Gilvarry, J. (September 1999). Calculation of Motion Using Motion Vectors Extracted from an MPEG Stream, from http://www.cdvp.dcu.ie/Papers/MVector2.pdf

APPENDIXS

List Of Figures

Page

List Of Tables

Page