IMPLEMENTATION OF FOUR-STEP SEARCH (4SS) ALGORITHM FOR MOTION ESTIMATION IN MATLAB

NORBAYAH BINTI YUSOP

This Report Is Submitted In Partial Fulfillment Of Requirements For The Bachelor Degree Of Electronic Engineering (Telecommunication Electronic) with honours

Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer
Universiti Teknikal Malaysia Melaka

APRIL 2007

Borang Pengesahan Status Laporan PSM II

UNIVERSITI TEKNIKAL MALAYSIA MELAKA
FAKULTI KEJURUTERAAN ELEKTRONIK DAN KEJURUTERAAN KOMPUTER

BORANG PENGESAHAN STATUS LAPORAN
PROJEK SARJANA MUDA II

Tajuk Projek : IMPLEMENTATION OF FOUR-STEP SEARCH (4SS) ALGORITHM FOR MOTION ESTIMATION IN MATLAB

Seisi Pengajian : 2006/2007

Saya NORBAYAH BT YUSOP

Mengaku membenarkan laporan Sarjana Muda ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. laporan adalah hak milik universiti teknikal Malaysia melaka
2. perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja
3. Perpustakaan dibenarkan membuat salinan laporan ini sebagai bahan pertukaran antara institusi tinggi.
4. Sila tandakan (√):

☐ **SULIT*** (mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

☐ **TERHAD*** (Mengandungi maklumat terhad yang telah ditentukan oleh organisasi/badan dimana penyelidikan dijalankan)

☑ **TIDAK TERHAD**

Disahkan Oleh:

(TANDATANGAN PENULIS)

Alamat Tetap: 327 (F) CHEMOMOI,
28310 TRIANG,PAHANG.

Tarikh : 27 APRIL 2007

(COP DAN TANDATANGAN PENYELIA)

REDZUAN B ABDUL MANAP
Pensyarah
Fakulti Kej Elektronik dan Kej Komputer (FKEKK),
Universiti Teknikal Malaysia Melaka (UTeM),
Karung Berkunci 1200,
Ayer Keroh, 75450 Melaka

Tarikh : 27 APRIL 2007

*CATATAN : Jika laporan ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

"I hereby declare that this report is the result of my own work except for quotes as cited in the references."

Signature : .....

Author : NORBAYAH BT YUSOP

Date : 27 APRIL 2007

"I hereby declare that I have read this report and in my opinion this report is sufficient in terms of scope and quality for the award of Bachelor of Electronic Engineering (Telecommunication Electronics) with honours."

Signature : 

Supervisor's Name : EN.REDZUAN B. ABD MANAP

Date : 27 APRIL 2007

ACKNOWLEDGEMENT

Alhamdulillah, thankful to Allah S.W.T for giving me a good health and an opportunity to complete my final year project, which titled an implementation of four step search (4SS) algorithm for motion estimation in MATLAB. I would also like to appreciate my supervisor, Mr. Redzuan Abd. Manap which has been supportive and helpful with my thesis.

Moreover, a grateful expression to my beloved family and friends on their unflinching courage and conviction that will always inspire me, and I hope to continue my ambition for engineering development in future.

Finally, sincerely acknowledgement from me to all the people that contributed to the development of my thesis.

ABSTRACT

Based on the real world image sequence's characteristic of center-biased motion vector distribution, a new four-step search (4SS) algorithm with center-biased checking point pattern for fast block motion estimation is proposed to be implemented thesis in this paper. Halfway-stop technique is employed in this algorithm with searching steps of 2 to 4 and the total number of checking points which is varied from 17 to 27. Simulation results show that the proposed 4SS performs better than the well-known three-step search (TSS), cross search (CS), cross diamond search (CDS) and has similar performance to the full search (FS) in terms of search points and PSNR. In addition, the 4SS also is more robust as compared with TSS and new three-step search (N3SS).

ABSTRAK

Berdasarkan karakter dan penyerakan *motion vector* pada imej sebenar, satu algoritma yang dipanggil *Four Step-Search (4SS)* yang mempunyai corak pemeriksaan *centre-biased* telah dicadangkan untuk projek ini. Teknik penghentian setengah digunakan dalam algoritma ini dengan melibatkan 2 hingga 4 langkah pencarian dan jumlah titik pemeriksaan adalah antara 17 hingga 27 titik. Keputusan simulasi menunjukkan prestasi algoritma ini adalah lebih baik jika dibandingkan dengan algoritma-algoritma lain seperti *Three-Step search (TSS)*, *Cross search (CS)*, *Cross-Diamond Search (CDS)* dan menghampiri prestasi algoritma *Full Search (FS)*.

TABLE OF CONTENTS

LIST OF ACRONYMS

4SS	-	Four-Step Search
BDM	-	Block Distortion Measure
BMA	-	Block Motion Algorithm
BMC	-	Block Motion Compensation
BME	-	Block Motion Estimation
CDS	-	Cross-Diamond Search
CIF	-	Common Intermediate Format
CS	-	Cross Search
DFD	-	Displaced Frame Difference
FS	-	Full Search
MAD	-	Measure Absolute Distortion
MAE	-	Measure Absolute Error
ME	-	Motion Estimation
MPEG	-	Moving Pictures Experts Group
MSD	-	Mean Squared Distance
MSE	-	Mean Squared Error
MV	-	Motion Vector
N3SS	-	New Three-Step Search
PDC	-	Pel Difference Classification
PSNR	-	Peak Signal To Noise Ratio
QCIF	-	Quarter Common Intermediate Format
SQCIF	-	Sub Quarter Common Intermediate Format
TSS	-	Three-Step Search

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Objectives

There are several techniques used in video coding to achieve high compression. Block Matching Algorithm (BMA) is one of common techniques that is being used. There are a lot of BMA fast algorithms that have been developed and it is so difficult to choose the best algorithm to be implemented.

The objective of this project is to investigate one of BMA fast algorithm known as Four- Step Search (4SS) Algorithm. For this project, 4SS Algorithm will be implemented by using MATLAB. Then, the performance is compared to other common BMA such as Full Search (FS), Three-Step Search (TSS), Cross Search (CS) and Cross-Diamond Search (CDS) algorithms in terms of peak signal-to-noise ratio (PSNR) and computational complexity.

1.2 Scope of Work

The scope of work for this project is divided into two parts. The first part is mainly based on the acquisition of information involving background theory on Motion Estimation (ME), BMA, MATLAB, 4SS Algorithm and video coding techniques. Basically, all the information is obtained from the internet, books and some journals.

The second part is the implementation of 4SS Algorithm using MATLAB and the performance comparison in terms of PSNR, the number of search points required and the computational complexity to other common BMA algorithms such as FS, TSS, CS and CDS.

CHAPTER II

BACKGROUND STUDY

2.1 Video Coding

Video coding in computer science deals with finding efficient coding formats for digital video. Video data usually not only contains visual information but also audio. Therefore, it is often referred to as multimedia. Modern video coding standards even include other multimedia data such as synthetic computer graphics, text and meta information for searching/browsing and digital rights management. They also often provide mechanisms for user interaction.

However, the most intensive parts of video data in terms of data size remain video and audio data. These parts have to be compressed. Unfortunately, this can hardly be done without loss of quality because of the enormous size of a lossless video stream. There are two special research areas that deal with multimedia compression: video compression and audio compression.

Video coding has two distinct goals: storing and transmission of video data. These two goals have much in common. Therefore, video file formats usually have the same structure as streaming video formats with just a little header information added. A computer program that encodes and decodes video data is called video codec.

Almost all successful techniques developed for video coding have been integrated in the MPEG standards. Therefore, the MPEG standards represent a comprehensive knowledge base for video coding. Most video coding standards and commercial formats are modifications of MPEG [1].

There are several image formats such as Common Intermediate Format (CIF) is used to standardize the horizontal and vertical resolutions in pixels of YCbCr sequences in video signals. It was designed to be easy to convert to PAL or NTSC standards. It was first proposed in the H.261 standard. It defines a video sequence with a resolution of 352×288, a framerate of 30000/1001 (roughly 29.97) fps, with colour encoded using YCbCr 4:2:0. Quarter Common Intermediate Format (QCIF) have one fourth of the area as *quarter* implies, height and width of the frame are halved. Terms also used are SQCIF , 4CIF (4× CIF) and 16CIF (16× CIF). The resolutions for all of these formats are summarized in the Table 2.1 . All xCIF formats result in an image of aspect ratio 4:3, hence the xCIF pixels are not square. On a computer screen with square pixels xCIF pictures have to be stretched for a resulting aspect ratio of 4:3. To correctly display CIF content on a computer screen, the CIF content must be stretched horizontally by ~9% to 384 x 288 [2].

Table 2.1: Resolution of several formats [2].

Format	Video Resolution
SQCIF	128 × 96
QCIF	176 × 144
CIF	352 × 288
4CIF	704 × 576
16CIF	1408 × 1152

2.2 Motion Estimation and Compensation

2.2.1 Motion Estimation

ME is the process of finding optimal or near-optimal motion vectors (MV). The amount of prediction error for a block is often measured using the mean squared error (MSE) or sum-of-absolute-differences (SAD) between the predicted and actual pixel values over all pixels of the motion-compensated region.

ME is used also for other applications than video encoding, like image stabilization, motion segmentation, and image analysis. However, the requirements for these applications differ significantly for video encoding algorithms: MV have to reflect the real motion within the image sequence, otherwise the algorithms will not give the desired results. For video encoding the situation is different. MV are used to compensate the motion within the video sequence and only the remaining signal (prediction error) has to be encoded and transmitted. Therefore, the MV have to be selected in order to minimize the prediction error and the number of bits required to code the prediction error. Within this project, the investigations on ME algorithms are focused on video coding. However, similar concepts are applicable to other applications as well [3].

To find optimal MV, one basically has to calculate the block prediction error for each MV within a certain search range and pick the one that has the best compromise between the amount of error and the number of bits needed for MV data. The ME technique of simply exhaustively testing all possible motion representations to perform such an optimization is called FS. A faster method, which is sub-optimal with respect to rate-distortion, is to use a coarse search grid for a first approximation and to refine the grid in the surrounding of this approximation in further steps. One common method is the TSS, which uses search grids of 3×3 MV and 3 refinement steps to get an overall search range of 15×15 pixel [4].

2.2.2 Requirements for Motion Estimation and Compensation

ME creates a model of the current frame based on available data in one or more previously encoded frames known as reference frames. These reference frames may be *past* frames (i.e. earlier than the current frame in temporal order) or *future* frames (i.e. later in temporal order). The design goals for a ME algorithm are to model the current frame as accurately as possible whilst maintaining acceptable computational complexity. In Figure 2.1, the ME module creates a model by modifying one or more reference frames to match the current frame as closely as possible according to a matching criterion.

The current frame is motion compensated by subtracting the model from the frame to produce a motion-compensated residual frame. This is coded and transmitted, along with the information required for the decoder to recreate the model which is typically a set of MV. At the same time, the encoded residual is decoded and added to the model to reconstruct a decoded copy of the current frame which may not be identical to the original frame because of coding losses. This reconstructed frame is stored to be used as a reference frame for further predictions.

The residual frame is encoded and transmitted, together with any *side information* needed to recreate the model at the decoder. The *best* compression performance is achieved when the size of the coded distanced frame difference (DFD) and coded side information is minimized. The size of the coded DFD is related to the energy remaining in the DFD after motion compensation. It should be possible to reduce this energy and improve compression performance using ME and motion compensation [3].

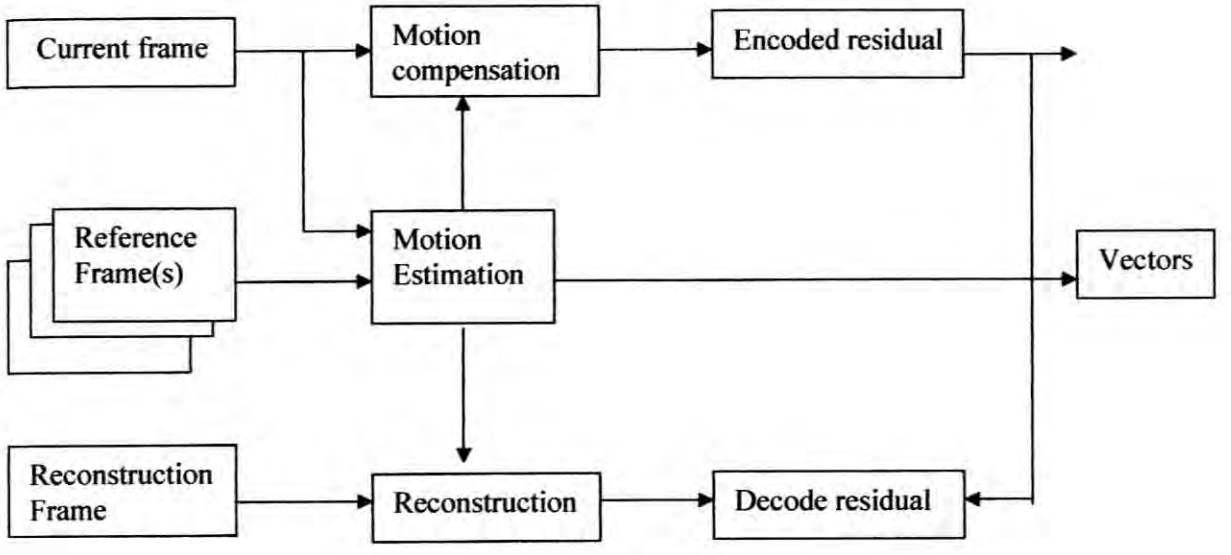

Figure 2.1: Motion estimation and compensation block diagram [3].

2.2.3 Motion Compensation

In video compression, motion compensation describes a picture in terms of where each section of that picture came from, in a previous picture. This is often employed in video compression. It can be also used for the interlacing. A video sequence consists of a number of pictures - usually called frames. Subsequent frames are very similar, thus, containing a lot of redundancy. Removing this redundancy helps achieve the goal of better compression ratios.

A first approach would be to simply subtract a reference frame from a given frame. The difference is then called *residual* and usually contains less energy (or information) than the original frame. The residual can be encoded at a lower bit-rate with the same quality. The decoder can reconstruct the original frame by adding the reference frame again.

A more sophisticated approach is to approximate the motion of the whole scene and the objects of a video sequence. The motion is described by some parameters that have to be encoded in the bit-stream. The pixels of the predicted

frame are approximated by appropriately translated pixels of the reference frame. This gives much better residuals than a simple subtraction. However, the bit-rate occupied by the parameters of the motion model must not become too large [5].

Usually, the frames are processed in groups. One frame, usually the first, is encoded without motion compensation just as a normal image. This frame is called *I-frame* (intra-coded frame, MPEG terminology) or *I-picture*. The other frames are called *P-frames* or *P-pictures* and are predicted from the I-frame or P-frame that comes immediately before it. The prediction schemes are, for instance, described as IPPPP, meaning that a group consists of one I-frame followed by four P-frames.

Frames can also be predicted from future frames. The future frames then need to be encoded before the predicted frames and thus, the encoding order does not necessarily match the real frame order. Such frames are usually predicted from two directions, i.e. from the I- or P-frames that immediately precede or follow the predicted frame. These bidirectionally, predicted frames are called *B-frames* [6].

2.2.4 Block Motion Compensation

In block motion compensation (BMC), the frames are partitioned in blocks of pixels. Each block is predicted from a block of equal size in the reference frame. The blocks are not transformed in any way apart from being shifted to the position of the predicted block. This shift is represented by a MV. The MV are the parameters of this motion model and have to be encoded into the bit-stream. As the MV are not always independent (e.g. if two neighbouring blocks belong to the same moving object), they are usually encoded differentially to save bit-rate. This means that the difference of the MV and the neighbouring MV encoded before is encoded.

It is possible to shift blocks by non-integer vectors, which is called sub-pixel precision. This is done by interpolating the pixel's values. Usually, the precision of the MV is increased by one bit: half-pixel precision. Of course, the computational expense for sub-pixel precision is much higher since the interpolation functions can be quite consuming [5].

2.3 Block Matching

2.3.1 Introduction

Block matching is the most time consuming part of the encoding process. During block matching each target block of the current frame is compared with a past frame in order to find a matching block. When the current frame is reconstructed by the receiver this matching block is used as a substitute for the block from the current frame. Block matching takes place only on the luminance component of frames. The colour components of the blocks are included when coding the frame but they are not usually used when evaluating the appropriateness of potential substitutes or candidate blocks.

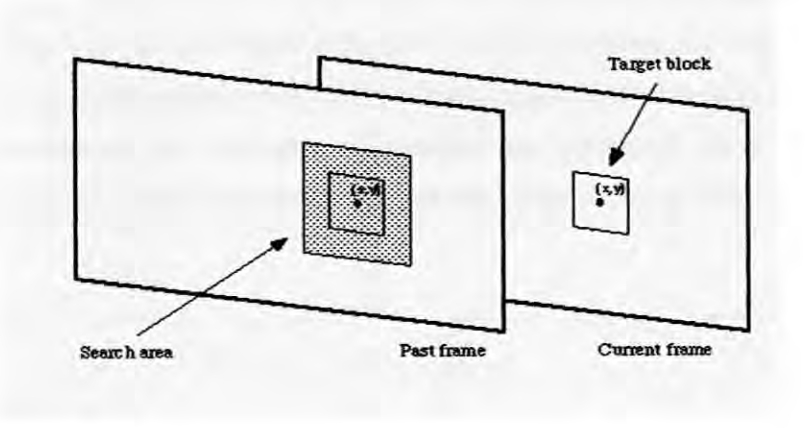

Figure 2.2: Target Block [7].

Corresponding blocks from a current and past frame, and the search area in the past frame. The search can be carried out on all of the past frame, but is usually restricted to a smaller *search area* centered around the position of the target block in the current frame (see above Figure 2.2). This practice places an upper limit, known as the *maximum displacement*, on how far objects can move between frames, if they are to be coded effectively. The maximum displacement is specified as the maximum

number of pixels in the horizontal and vertical directions that a candidate block can be from the position of the target block in the original frame. The quality of the match can often be improved by interpolating pixels in the search area, effectively increasing the resolution within the search area by allowing hypothetical candidate blocks with fractional displacement.

Block matching is the most computational intensive part of encoding a video sequence, thus takes the longest time. In block matching the target block in the current frame is compared to the last frame trying to find the area of resembling size that is the most similar, this is often done by comparing block distortion. The block with the least distortion is known as *the best match.*

When comparing, only the luminous component of the pixel is considered [8]. This procedure is repeated for all blocks in the current frame. In order to decrease the computational load of the block matching, often a smaller area than the whole frame is searched. This smaller area is often called search area or search window, see Figure 2.3. It is usually quadratic with a fixed size but there are some algorithms with search windows of varying shapes and sizes. To further decrease the load some of the elements in the search window are not examined. Algorithms have been developed to make sure that as few elements as possible are examined while maintaining acceptable quality. These are called block matching algorithms, or BMA's [9].

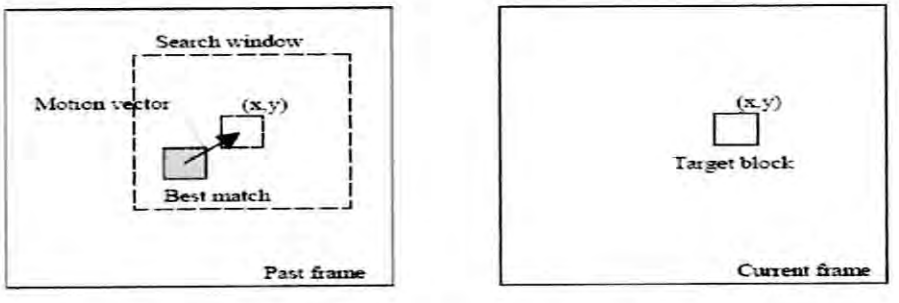

Figure 2.3: The block matching algorithm searches the search window in order to find the best match for the target blocks [9].

In the popular video coding standards (H.261, H.263, MPEG-2 and MPEG-4), ME and motion compensation are carried out on 8 x 8 or 16 x 16 blocks in the current frame. ME of complete blocks is known as block matching.

For each block of luminance samples (say 16 x 16) in the current frame, the ME algorithm searches a neighbouring area of the reference frame for a *matching* 16 x 16 area. The best match is the one that minimizes the energy of the difference between the current 16 x 16 block and the matching 16 x 16 area. The area in which the search is carried out may be centered around the position of the current 16 x 16 block, because (a) there is likely to be a good match in the immediate area of the current block due to the high similarity (correlation) between subsequent frames and (b) it would be computationally intensive to search the whole of the reference frame [3].

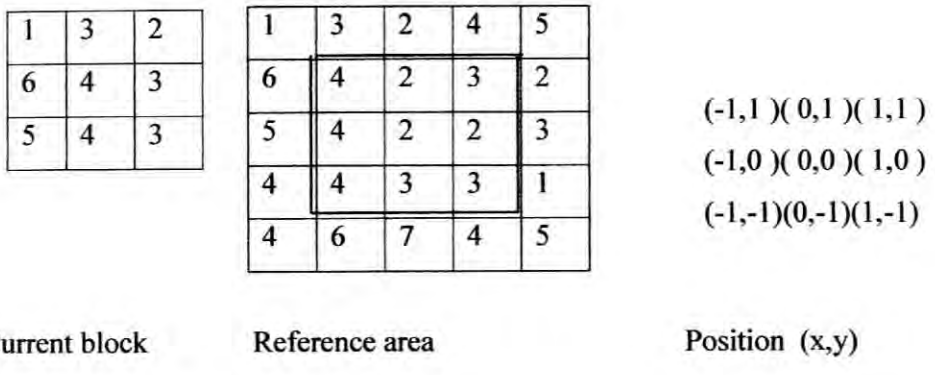

Figure 2.4 : Current 3 x 3 block and 5 x 5 reference area [3].

Figure 2.4 illustrates the block matching process. The current *block* (in this case, 3 x 3 pixels) is shown on the left and this block is compared with the same position in the reference frame (shown by the thick line in the centre) and the immediate neighbouring position (+/- 1 pixel in each direction). The MSE between the current block and the same position in the reference frame (position (0,0)) is given by

$$\{ (1-4)^2 + (3-2)^2 + (2-3)^2 + (6-4)^2 + (4-2)^2 + (3-2)^2 + (5-4)^2 + (4-3)^2 + (3-3)^2 \} / 9 = 2.44$$