PERFORMANCE AND QUALITY IMPROVEMENT OF MPEG-4 VIDEO ENCODER

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree of

MASTER OF TECHNOLOGY

in

ELECTRONICS & COMMUNICATION ENGINEERING

(With Specialization in Communication Systems)

By

RAJNESH M.P.

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE -247 667 (INDIA)
September, 2007

CANDIDATE'S DECLARATION

I hereby declare that the work, which is presented in this dissertation report, entitled **"PERFORMANCE AND QUALITY IMPROVEMENT OF MPEG-4 VIDEO ENCODER"** being submitted in partial fulfillment of the requirements for the award of the degree of **MASTER OF TECHNOLOGY** with specialization in **COMMUNICATION SYSTEMS**, in the Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee is an authentic record of my own work carried out , under guidance and supervision of **Dr. S. K.VARMA**, Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee and **Dr. Rasmi Rekha Misra**, Program Manager, Sasken Communication Technologies Limited , Bangalore.
The results embodied in this dissertation have not submitted for the award of any other Degree or Diploma.

Date : 6 September, 2007
Place : Roorkee

[illegible]
RAJNESH M.P.

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my Knowledge and belief.

Date : 6 September, 2007
Place : Roorkee

[illegible]
Dr. S. K.VARMA
Professor, E&CE Department
Indian Institute of Technology, Roorkee
Roorkee – 247 667, (INDIA)
and

[illegible]

Dr. Rasmi Rekha Misra
Program Manager,
Sasken Communication Technologies LTD,
Bangalore-560071 (INDIA)

PeopleFirst

sasken

July 9, 2007

To Whomsoever It May Concern

This is to certify that project on "Performance and Quality Improvement of MPEG-4 Video Encoder" has been successfully completed with Sasken from January 6 2007 till July 6 2007 by "Rajnesh Pandarinath" student of "Indian Institute of Technology Roorkee (IIT Roorkee)"

The following tasks have been accomplished in the project work.

1. Implementation of fast motion estimation algorithms like diamond and Hexagonal search.
2. Implementation of motion estimation using multi resolution analysis using Haar-Wavelet.
3. Motion vector prediction using spatiotemporal information and combining with Fast motion estimation algorithms.
4. Motion vector prediction using spatiotemporal information and combining with multi-resolution analysis for better motion vector prediction.

And the code has been successfully ported on TI64XXDSK.

His conduct has been good during this period.

Yours Sincerely

For Sasken Technologies Ltd

[illegible] [illegible]

Pragya Shrimali
Executive HR

Dr. Rasmi Rekha Misra
Program Manager

Sasken Communication Technologies Limited
139/25 Domlur, Ring Road, Bangalore-560 071, India
T 91 80 66944000 F 91 80 66944329
www.sasken.com

ACKNOWLEDGEMENTS

It is my privilege and pleasure to express my profound sense of respect, gratitude and indebtedness to my guide, **Dr. S. K. Varma,** Professor, Department of Electronics and Computer Engineering , Indian Institute of Technology, Roorkee, for his inspiration, guidance and encouragement throughout this dissertation work.

I am thankful to **Sasken Communication Technologies-** Bangalore, Dr. Srinivas Rao Kandula (Director Sasken) and my cousin M.A. Raviender (General Manager Power Grid) for providing me the opportunity to work in Sasken. I would also like to thank **Dr. Rasmi Rekha Misra** (Program Manager), Bob (Team Leader) and Prashanth (Video specialist) for their advice and constructive criticisms during my dissertation work at Sasken.

I am greatly indebted to all my friends, who have graciously applied themselves to the task of helping me with ample morale support and valuable suggestions. Most of all, I would like thank my family. My parents provided me a perfect environment for my studies and supported me throughout. Finally, I would like to extend my gratitude to all those persons who directly or indirectly helped me in the process and contributed towards this work.

RAJNESH M.P.

ABSTRACT

Transmission of real-time video over wireless channels is becoming feasible, as high data rates are available with the advent of third generation mobile communication. The latest multimedia standard MPEG-4 Simple Profile (SP) is used in Mobile Multimedia Messaging (MMS), Video transmission and Packet Switched Streaming (PSS) applications. MMS and PSS are used for multimedia content delivery over wireless networks. MPEG-4 is ideally suited for wireless channels because of its high coding efficiency, error resilience and content-based functionalities. In mobile solutions, real time encoding efficiency without much computational cost and low-power consumption are essential. Due to the increased complexity of compression tools, MPEG-4 video encoding processes require significant amount of computation and processing power. With the ever-increasing growth in handheld and mobile devices, embedded system applications have become complex and RISC cores are increasingly being used. The RISC processors are used in hand held devices because of their low power consumption, high performance and small in silicon, which are important for mobile devices. Hence, RISC cores are being used to perform video encoding and decoding in mobile video communications. As a result, optimization of applications to the extent possible is essential.

Our work is aimed at performing efficient encoding of QCIF (4:2:0 format) video at 15 frames per second, which is sufficient for mobile video communications, on efficient RISC core (TMS320C64XX DSK). The video sequence cannot be transmitted as it is since it consumes lot of bandwidth and storage capacity. Motion estimation is implemented at the encoder side, which is a very essential block for efficient motion prediction thus reducing the amount of bandwidth and storage capacity that are required.

Full search block matching motion estimation is superior to any other Fast block matching motion estimation methods. It does not get trapped onto the local minima while estimating motion. But it consumes lot of computations which is not desired in mobile communication. Our aim in this work is to reduce the amount of computations at the same time not compromising on the quality of the video sequence obtained after motion compensation. We have experimented on motion estimation using Fast block motion estimation techniques, motion estimation using multi resolution analysis, motion estimation using spatio-temporal contextual information and combining multi resolution analysis with spatio-temporal prediction to obtain good motion vector prediction and at the same time reduce the computational cost.

The quality of the reconstructed video sequence is measured using peak signal to noise ratio (PSNR).The code is ported on to TMS320C64XX DSK and we tried to optimize the code.

CONTENTS

CHAPTER 1

Introduction

Digital Video-coding technology has developed into a mature field and products have been developed that are targeted for a wide range of emerging applications, such as video on demand, digital TV / HDTV (High Definition TV) broadcasting, and multimedia image / video data-base services. Video sequences contain a significant amount of statistical and subjective redundancy within and between frames. With the increasing demand of multimedia applications, considerable efforts are needed for efficient video compressing and encoding algorithms. Compression algorithm tries to exploit the temporal and spatial redundancy by using some form of motion compensation followed by transform coding. The key step in removing temporal redundancy is the motion estimation where a motion vector is predicted between the current frame and a reference frame. Motion estimation has proven to be an effective technique for exploiting the temporal redundancy in video sequences and is therefore an essential part of MPEG and H.26X compression standards.

Knowledge of the motion is not available from a video data and must be deduced using computationally intensive algorithms. The motion vector search process is done by evaluating candidates within a motion search window in the reference frame to find out the best-matched block that produces the minimum error according to a certain matching criterion, e.g., the widely used mean absolute difference (MAD) or equivalently sum of absolute difference error / mean square error (SAE / MSE). Computation is consumed on the calculation of MSE / SAE.

There has been many motion estimation techniques like Frequency domain technique, Pel-Recursive technique, Gradient method, Bayesian Method, Optical Flow equation based method and most importantly Block matching technique which has been recommended in all the MPEG standards.

Generally, the optimal full search (FS) block matching algorithm is the simplest block matching algorithm and results in the best performance with respect to the quality of decoded video sequences. However, such exhaustive search and full-matching error calculation at each check point yields FS extremely computational expensive, thus making real-time video applications impossible. It consumes up to 80% of the computation power of the encoder. Hence fast, yet accurate, motion-estimation algorithms are highly in need to significantly reduce the computational complexity while maintaining prediction accuracy. Several fast search algorithms have been developed and introduced in recent years including the diamond search (DS), the hexagon-based search (HEXBS) and the enhanced hexagonal based search. All the methods are proposed keeping any one or more directions aimed at

1. Reducing computational complexity.
2. Representing true motion (providing good quality)
3. Reducing bit rate (high compression ratio).

Since videos exist with variety of contents, no stand-alone fast block matching algorithm (FBMA) can efficiently remove the redundant data. The assumptions which all the FBMA make are:

1. Uni-modal error surface: The matching error increases monotonically as the search moves away from the position of the global minimum error.
2. Illumination is uniform along the motion trajectory.
3. Problems due to uncovered motions are neglected.

The implication of assumption one is that FBMA tend to get trapped into the local minima instead of the global minima which is desired for getting the correct motion information.

1.1 STATEMENT OF THE PROBLEM

In this dissertation we have tried to overcome the uni-modal error surface assumption disadvantage and tried to produce a good quality image after motion compensation. Peak signal to noise criteria is used as a quality measure. We have tired to implement various fast motion estimation techniques and use multi resolution analysis using Haar-wavelets for estimating motion. Some times this can lead to wrong motion vector prediction, to reduce this we have tried to use the spatio-temporal information which is available for predicting the motion vectors.

The main objectives of the work are:

1. To study the fast motion estimation algorithms.
2. To study motion estimation using multi resolution analysis using wavelets.
3. To study motion estimation using spatio-temporal contextual information.
4. To study motion estimation using multi resolution analysis combined with spatio-temporal contextual information.

1.2 Organization of the Report

Chapter 2 deals with the basics of video coding. In Chapter 3 we have introduced the encoder Block Diagram of MPEG standard, discussed the coding tools available for video coding in MPEG-4 standard and optimization techniques available in TMS320C64XX processor. In Chapter 4 we discuss Motion estimation algorithms in depth. In Chapter 5 we have discussed about simulation and the results are produced.

CHAPTER 2

Video Basics and Standards

There are different video coding standards like MPEG-1, MPEG-2, MPEG-4, MPEG-7, MPEG-21, H.261, H.264 (MPEG-4 Part-10). Inter-frame predictive coding is the main coding principle that is used in all standard video codecs.

2.1 2-D Motion Vs Apparent Motion [5]

2-D motion fields are projections on the 2D image plane of the 3D motion in the scene. Optical Flow field (apparent motion) is the field associated with spatio-temporal variation of intensity. In an ideal case, the optical flow corresponds to the 2D motion field. However, in practice this is not guaranteed. For instance, it may happen that a moving object gives rise to a constant brightness pattern, thus optical flow is zero even though motion exists in the scene. Conversely, in a still scene the optical flow may be nonzero due to illumination changes.

Motion formulations in terms of either instantaneous velocity or displacement are possible. The formulation in term of displacement is adapted, and the term motion vector is understood accordingly.

The goal is not to assess the motion present in the scene, but to model the changes in the spatio-temporal intensity and therefore to estimate the optical flow. Generally no distinction is made between global and local motions. The global motion is thus taken into account through local estimation of the motion. In MPEG-4 standard we have a feature known as global motion compensation which is used to take into account global motion in the frame. The hypothesis is that there is no change of illumination and that the variations in the spatio-temporal intensity are only due to the global and local motion.

2.2 Frame [1] [7]

A frame consists of three rectangular matrices of integers: a luminance matrix (Y), and two chrominance matrices (Cb and Cr). The relationship between these Y, Cb and Cr components and the primary (analogue)

Red, Green and Blue Signals (E'R, E'G and E'B), the chromaticity of these primaries and the transfer characteristics of the source frame may be specified in the bit stream (or specified by some other means).

Field [1] [7]

A field consists of every other line of samples in the three rectangular matrices of integers representing a frame. A frame is the union of a top field and a bottom field. The top field is the field that contains the top-most line of each of the three matrices. The bottom field is the other one.

2.3 Group Of Pictures (GOP) [16]

Due to the existence of several picture types, a group of pictures, called GOP, is the highest level of the hierarchy. A GOP is a series of one or more pictures to assist random access into the picture sequence. The first coded picture in the group is an I-picture. It is followed by an arrangement for P and B-pictures.

I-VOP – Intra-coded VOP, coded independently of all other VOPs.

P-VOP – Predictively coded VOP, coded based on previously coded VOP.

B-VOP –Bi-directionally predicted VOP, coded based on both previous and future coded VOPs.

D-pictures: - These are intra-VOP coded, where only the DC coefficients are retained. Hence the picture quality is poor and normally used for applications like fast forward. D-pictures are not part of the GOP; hence they are not present in a sequence containing any other picture type.

The GOP length is normally defined as the distance between I-pictures, which is represented by parameter *N* in the standard codec's. The distance between the anchor I / P to P-pictures is represented by *M.* The group of pictures may be of any length, but there should be at least one I-picture in each GOP. Applications requiring random access, fast forward play or fast and normal reverse play may use short GOPs. GOP may also start at scene cuts or other cases where motion compensation is not effective. The number of consecutive B-pictures is variable. Neither a P nor a B-picture needs to be present.

2.4 Video Object Plane (VOP) [7] [16]

A video object (VO) is an area of the video scene that may occupy an arbitrarily-shaped region and may exist for an arbitrary length of time. A video object in a scene is an entity that a user is allowed to access (seek, browse) and manipulate (cut and paste). The instances of video objects at a given time are called video object planes (VOPs). The encoding process generates a coded representation of VOP as well as composition information necessary for display. Further, at the decoder, a user may interact with and modify the composition process as needed. A sprite (S) VOP is a VOP for a sprite object or a VOP which is coded using prediction based on global motion compensation from a past reference VOP.

2.4.1 Formation of VOP [16]

The shape information is used to form a VOP. For maximum coding efficiency, the arbitrary shape VOP is encapsulated in a bounding rectangle such that the object contains the minimum number of macro blocks. To generate the bounding rectangle, the following steps are followed:

1. Generate the tightest rectangle around the object, as shown in Fig-2.1. Since the dimensions of the chrominance VOP are half of the luminance VOP (4:2:0), then the top left position of the rectangle should be an even numbered pixel.

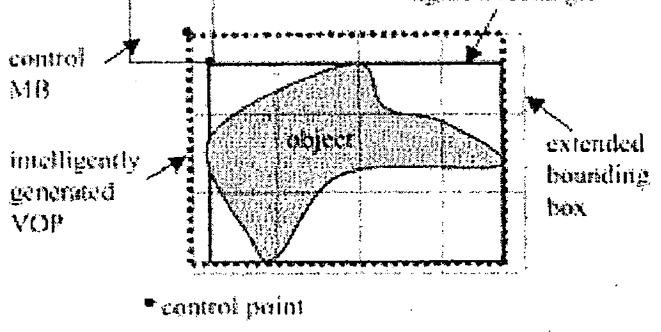


Fig-2.1. Intelligent VOP formation [16]

2. If the top left position of this rectangle is the origin of the frame, skip the formation procedure.

3. Form a control macro block at the top left corner of the tightest rectangle, as shown in Fig-2.3.

4. Count the number of macro blocks that completely contain the object, starting at each even numbered point of the control macro block. Details are as follows:

    i. Generate a bounding rectangle from the control point to the right bottom side of the object that consists of multiples of 16 × 16 pixel macro blocks.
    ii. Count the number of macro blocks in this rectangle that contain at least one object pixel.

5. Select that control point which results in the smallest number of macro blocks for the given object.

6. Extend the top left coordinate of the tightest rectangle to the selected control coordinate.

This will create a rectangle that completely contains the object but with the minimum number of macro blocks in it. The VOP horizontal and vertical spatial references are taken directly from the modified top left coordinate.

2.5 Image Format [1] [7]

0 – Represents Luminance sample

X- Represents Chrominance sample

2.5.1 (4:2:0) Format

In this format the Cb and Cr matrices shall be one half the size of the Y-matrix in both horizontal and vertical dimensions. The Y-matrix shall have an even number of lines and samples. If the matrices represent RGB colour primary matrices, this 4:2:0 formats shall not be applied.

When interlaced frames are coded as rectangular field VOPs, the VOP reconstructed from each of these field VOPs shall have a Y-matrix with half the number of lines of the corresponding frame. Thus the total number of lines in the Y-matrix of an entire frame shall be divisible by four. The luminance and chrominance samples are positioned as shown in the Fig-2.2.

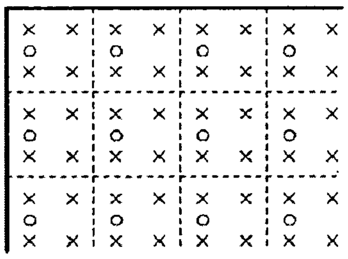


Fig - 2.2. The position of luminance and chrominance samples in 4:2:0 data [7]

2.5.2 (4:2:2) Format

In this format the Cb and Cr matrices shall be one half the size of the Y-matrix in the horizontal dimension and the same size as the Y-matrix in the vertical dimension.. Thus the total number of lines in the Y-matrix of an entire frame shall be divisible by two. The luminance and chrominance samples are positioned as shown in Fig-2.3.

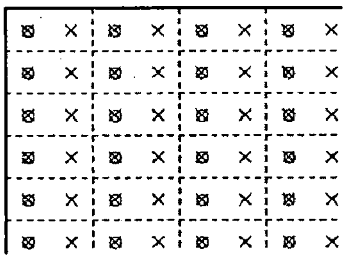


Fig-2.3. The position of luminance and chrominance samples. [7]

2.5.3 (4:4:4) Format

In this format the Cb and Cr matrices shall be the same size as the Y-matrix in the horizontal and the vertical dimensions. Thus the total number of lines in the Y-matrix of an entire frame shall be divisible by two. The luminance and chrominance samples are positioned as shown in the Fig-2.4.

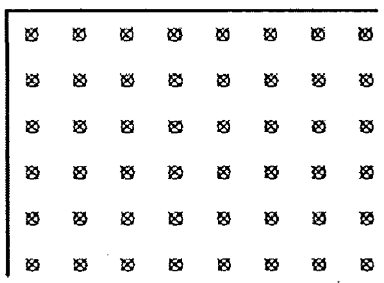
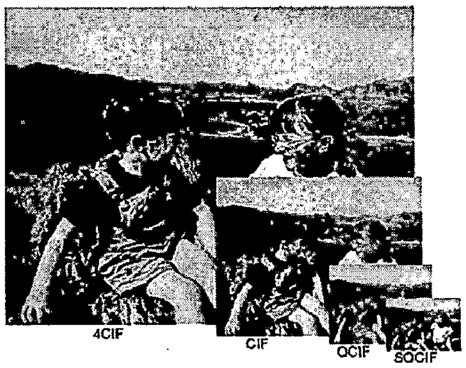


Fig-2.4. The position of luminance and chrominance samples. 4:4:4 data. [7]

2.6 Video Frame Formats [19]



Format	**Luminance resolution**
Sub-QCIF	**128 x 96**
Quarter CIF	**176 x 144**
CIF	**352 x 144**
4CIF	**704 x 144**

Fig-2.5. Picture formats [19]

2.7 Video Redundancy [16] [19]

1. Spatial redundancy reduction: To reduce spatial redundancy among the pixels within a picture (similarity of pixels, within the frames), by employing some data compressors, such as transform coding.
2. Temporal redundancy reduction: To remove similarities between the successive pictures, by coding their differences.
3. Entropy coding: To reduce the redundancy between the compressed data symbols, using variable length coding techniques

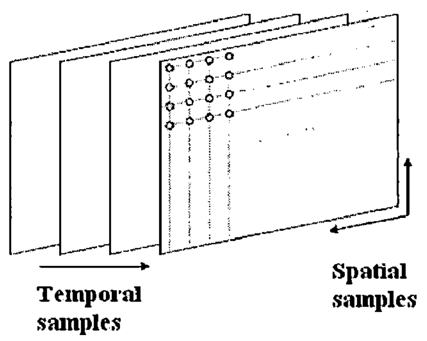


Fig-2.6.Spatial and temporal sampling of a video sequence [19]

2.8 Profiles and levels [5]

A "profile" is a defined subset of the entire bit stream syntax that is defined by this specification. Within the bounds imposed by the syntax of a given profile it is still possible to require a very large variation in the performance of encoders and decoders depending upon the values taken by parameters in the bit stream. For instance it is possible to specify frame sizes as large as 214 samples wide by 214 lines high. It is currently neither practical nor economical to implement a decoder capable of dealing with all possible frame sizes.

In order to deal with this problem "levels" are defined within each profile. A level is a defined as a set of constraints imposed on parameters in the bit stream. These constraints may be simple limits on numbers. Alternatively they may take the form of constraints on arithmetic combinations of the parameters (e.g. frame width multiplied by frame height multiplied by frame rate). Bit streams complying with this specifications use a common syntax. In order to achieve a subset of the complete syntax flags and the parameters are included in the bit stream that signal the presence or otherwise of syntactic elements that occur later in the bit stream. In order to specify constraints on the syntax (and hence define a profile) it is thus only necessary to constrain the values of these flags and parameters that specify the presence of later syntactic elements.

2.9 Video-Coding Standard [5]

Commercially, international standardization of video communication systems and protocols aims to serve two important purposes:

1. Interoperability
2. Economy of scale.

Networking between video communication equipment from different vendors is a desirable feature for users and equipment manufactures alike. One key factor for the success is the generic structure of the MPEG standards, supporting a wide range of applications and applications-specific parameters. To support the wide range of applications profiles, a diversity of input parameters including flexible picture size and frame rate can be specified by the user. Another important factor is the fact that the MPEG group only standardized the decoder structures and the bit-stream formats. This allows a large degree of freedom for manufactures to optimize the coding efficiency (or, in other words, the video quality at a given bit rate) by developing innovative encoder algorithms even after the standards were finalized.

The MPEG-1 Standard [5]

The video-compression technique developed by MPEG-1 covers many applications from interactive systems on CD-ROM to the delivery of video over telecommunications networks. However, MPEG-1 was primarily targeted for multimedia CD-ROM applications, requiring additional functionality supported by both encoder and decoder. Important features provided by MPEG- 1 include frame based random access of video, fast foward /fast reverse (FF / FR) searches through compressed bit streams, reverse playback of video and edit ability of the compressed bit stream.

The MPEG-2 Standard [5]

Basically, MPEG-2 can be seen as a superset of the MPEG-1 coding standard and was designed to be backward compatible to MPEG-l every MPEG-2- compatible decoder can decode a valid MPEG-1 bit stream. Many video-coding algorithms were integrated into a single syntax to meet the diverse applications requirements.

The MPEG-4 Standard [5]

Anticipating the rapid convergence of telecommunications, computer, and TV / film industries, the MPEG group officially initiated a new MPEG-4 standardization phase in 1994 with the mandate to standard algorithms and tools for coding and flexible representation of audio-visual data to meet the challenges of future multi-161. In particular, MPEG-4 addresses the need for:

1. Universal accessibility and robustness in error prone environments:
2. Coding of natural and synthetic data

3. Compression efficiency

For a simple example of an image scene containing a number of video objects, like the background, several items in foreground, and a text overlay. The attempt is to encode the sequence in a way that will allow the separate decoding and reconstruction of the objects for the user to assist the presentation and manipulation of the original scene in a flexible way.

1. **Simple profile:** This profile provides the simplest tool for low cost applications, such as video over mobile and Internet. It supports up to four rectangular objects in a scene within QCIF pictures. There are three levels in this profile to define bit rates from 64-384 Kbit/s (64, 128 and 384 Kbit/s for level-1, level-2 and level-3, respectively). The simple profile also supports most of the optionalities that are mainly useful for error resilience transmission. In addition to I and P-VOPs (video object planes to be defined shortly) they include: AC / DC prediction, four motion vectors, unrestricted motion vectors, quarter-pixel spatial accuracy, slice synchronization, data partitioning and reversible VLC. This profile can decode a bit stream generated by the core H.263.

2. **Advanced real time simple:** This adds error protection to the simple profile, through the introduction of the back channel. In response to a negative acknowledgement from the decoder, the encoder encodes the affected parts of the picture in intra mode. This profile improves the robustness of real time visual services over error prone channels such as videophone.

3. **Advanced simple profile:** This improves the compression efficiency of the simple profile, by supporting quarter-pixel resolution and global motion estimation in addition to B-VOPs.

2.10 Criteria for evaluation [16]

Subjective criteria: For video evaluation single stimulus continuous quality evaluation (SSCQE) is preferred, where the time-varying picture quality of the processed video without reference is evaluated by the subjects. In this method subjects are asked to continuously evaluate the video quality of a set of video scenes. The judgment criteria are the five scales as used in the (DSCQS) double stimulus continuous quality scale. Since video sequences are long, they are segmented into ten seconds shots, and for each video segment an MOS is calculated.

Objective Criteria: An alternative is objective measurements, or video quality metrics, which employ some mathematical models to mimic human visual systems behavior. Until any of the quality metrics become standards, it is customary to use the simplest form of objective measurement, which is the ratio of the peak-to-peak signal to the root-mean-squared processing noise. This is referred to as the peak-to-peak signal-to-noise ratio (PSNR) and defined as:

$$\mathrm{PSNR} = 10\log_{10}\left[\frac{255^2}{(1/N)\sum_i\sum_j\left(Y_{ref}(i,j) - Y_{pre}(i,j)\right)^2}\right]$$

where $Y_{ref}(i,j)$ and $Y_{pre}(i,j)$ are the pixel values of the reference and processed images, respectively, and N is the total number of pixels in the image. In this equation, the peak signal with an eight-bit resolution is 255, and the noise is the square of the pixel-to-pixel difference (error) between the reference image and the image under study. Although it has been claimed that in some cases the PSNR's accuracy is doubtful, its relative simplicity makes it a very popular choice.

CHAPTER 3

Video Codec

The MPEG group only standardized the decoder structures and the bit-stream formats. This allows a large degree of freedom for manufactures to optimize the coding efficiency (or, in other words, the video quality at a given bit rate) by developing innovative encoder algorithms even after the standards were finalized.

3. 1 Encoder

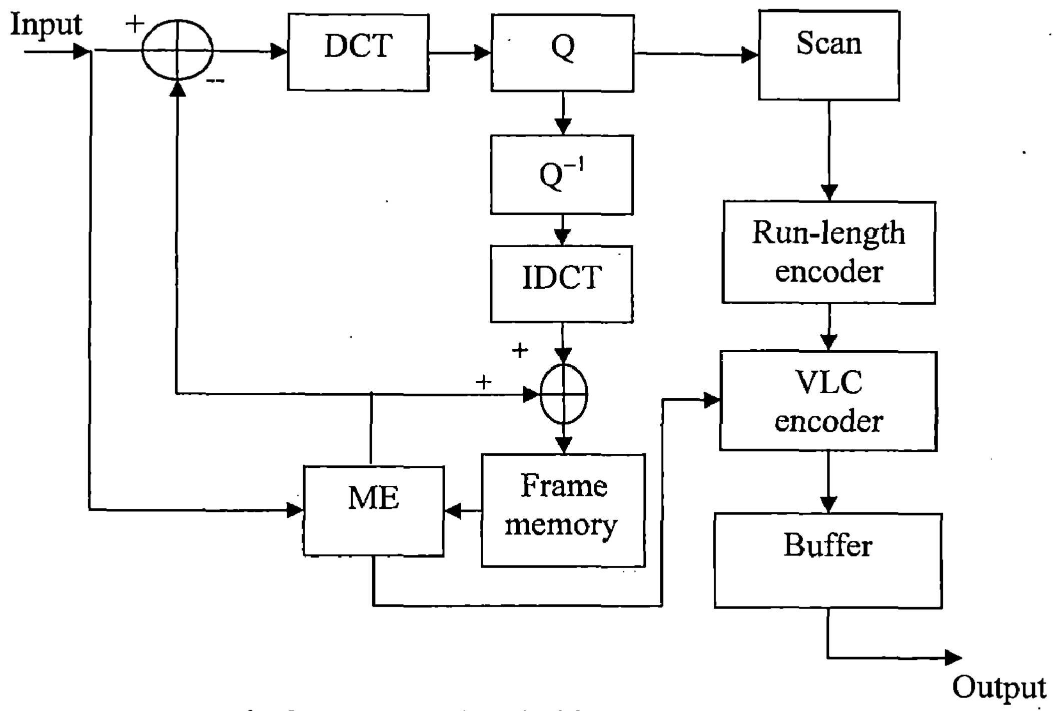


Fig-3.1. Conventional video encoder. [9]

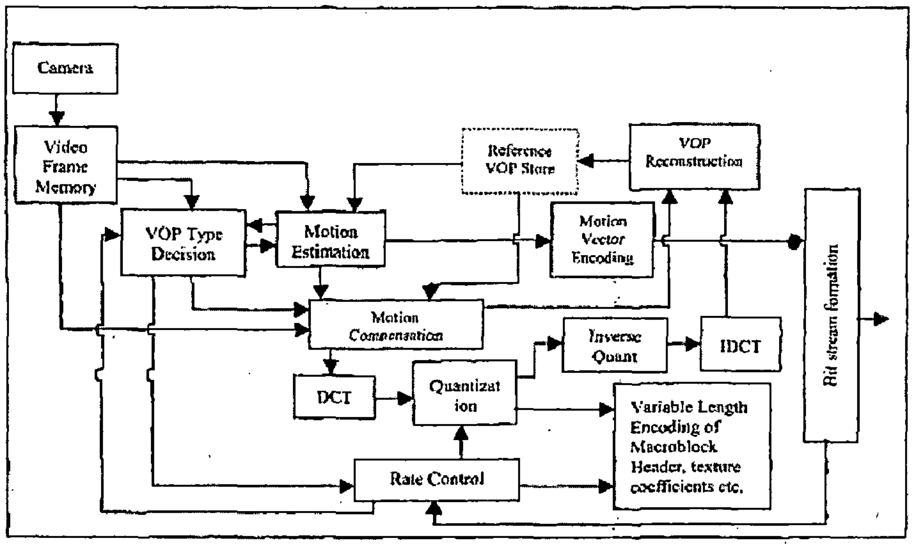


Fig-3.2. MPEG-4 Video Encoder Block Diagram [18]

From the below table we can see that motion estimation is the most computationally intensive task of all the blocks which are there in the encoder. The task of motion estimation is to reduce the temporal redundancy which is there in video sequence. Motion estimation is discussed in the next chapter here we will talk about important terms in MPEG-4 encoder and other blocks.

Module Name	**Memory cycles per sec**	**% of total computations**
ME	99.83	60.5%
DCT and Quantization	36.30	22.0%
MC	10.73	6.5%
IDCT and Inverse Quantization	5.78	3.5%
VLE	7.43	4.5%
VOP reconstruction	4.95	3.0%

ME - Motion Estimation.	MC- Motion Compensation
DCT- Discrete Cosine Transform	IDCT- Inverse DCT
VLE- Variable length encoding	VOP-Video Object Plane

3.2 Terms in motion estimation

Few important terms related to video encoding are discussed below.

3.2.1 Four motion vectors per macro block [7] [16]

Motion compensation tends to be more effective with smaller block sizes. The default block size for motion compensation is 16 × 16 samples (luma), 8 × 8 samples (chroma), resulting in one motion vector per macro block. This tool gives the encoder the option to choose a smaller motion compensation block size, 8 × 8 samples (luma) and 4 × 4 samples (chroma), giving four motion vectors per macro block. This mode can be more effective at minimizing the energy in the motion-compensated residual, particularly in areas of complex motion or near the boundaries of moving objects. There is an increased overhead in sending four motion vectors instead of one, and so the encoder may choose to send one or four motion vectors on a macro block by macro block basis.

3.2.2 Unrestricted motion vector [7]

In the default prediction mode of H.263, motion vectors are restricted so that all pixels referenced by them are within the coded picture area. In the optional unrestricted motion vector mode this restriction is removed and therefore motion vectors are allowed to point outside the picture. When a pixel referenced by a motion vector is outside of the coded picture area, an edge pixel is used instead. This edge pixel is found by limiting the motion vector to the last full pixel position inside the coded picture area. Limitation of the motion vector is performed on a pixel-by-pixel basis and separately for each component of the motion vector.

3.2.3 Short_video_header [7]

The short_video_header is an internal flag which is set to 1 when an abbreviated header format is used for the VOP. This indicates video data which begins with a short_video_start_marker rather than a longer start code such as visual_object_ start_code. The short header format is included herein to provide forward compatibility with video codecs designed using the earlier video coding specifications ITU-T recommendation H.263. All decoders which support video objects shall support both header formats (short_video_header equal to 0 or 1) for the subset of video tools that is expressible in either form. Short Header is not allowed in the base-layer.

3.2.4 Global motion compensation (GMC) [7] [14]

The use of global spatial transformation is to improve the efficiency of the prediction of the sample values. The prediction uses global spatial transformation to provide offsets into the past reference VOPs containing previously decoded sample values that are used to form the prediction error.

In MPEG-4, global motion compensation is based on the transmission of a static sprite. A static sprite is a (possibly large) still image, describing panoramic background. For each consecutive image in a sequence, only eight global motion parameters describing camera motion are coded to reconstruct the object. These parameters represent the appropriate perspective transform of the sprite transmitted in the first frame.

A sprite panorama is a still image that describes the content of the background over all frames in the sequence. The large panorama sprite image is transmitted to the receiver only once as the first frame of the sequence to describe the background. The sprite is stored in a sprite buffer. In each consecutive frame only the camera parameters relevant for the background are transmitted to the receiver. This allows the receiver to reconstruct the background image for each frame in the sequence based on the sprite. The moving foreground object is transmitted separately as an arbitrary-shape video object. The receiver composes both the foreground and background images to reconstruct each frame. For low delay applications it is possible to transmit the sprite in multiple smaller pieces over consecutive frames or to build up progressively the sprite at the decoder.

Global motion compensation can also be applied to the foreground objects. Here, for each VOP, a spatial transform, like the perspective transform, is used to estimate the transformation parameters. These are regarded as the global motion parameters that are used to compensate for the global motion of the foreground object. Globally motion compensated VOP is then coded by motion compensated texture coding, where it is once again motion compensated, but this time with a conventional block matching technique.

3.2.5 DC and AC prediction for intra macro blocks [1] [7]

This prediction process is only carried out for intra-macro blocks (I-MBs) and when short_video_header is "0". When short_video_header is "1" or the macro block is not an I-MB, this prediction process is not performed.

Low-frequency transform coefficients of neighboring intra-coded 8 × 8 blocks are often correlated. In this mode, the DC coefficient and (optionally) the first row and column of AC coefficients in an Intra-coded 8 × 8 block are predicted from neighboring coded blocks. The DC coefficients (top-left) are clearly similar but it is less obvious whether there is correlation between the first row and column of the AC coefficients in these blocks.

The DC coefficient of the current block (X in Fig-3.3) is predicted from the DC coefficient of the upper (C) or left (A) previously-coded 8 × 8 block. The rescaled DC Coefficient values of blocks A, B and C determine the method of DC prediction. If A, B or C are outside the VOP boundary or the boundary of the current video packet, or if they are not intra-coded, their DC coefficient value is assumed to be equal to 1024 (the DC coefficient of a mid-grey block of samples).

The direction of prediction is determined by:

```
if |DCA – DCB| < |DCB – DCC|
    Predict from block C
Else
    Predict from block A
```

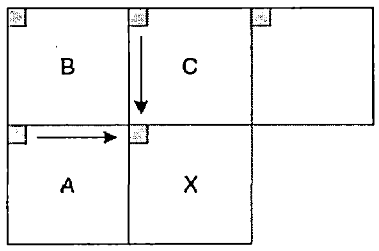


Fig-3.3: Prediction of DC coefficients [7]

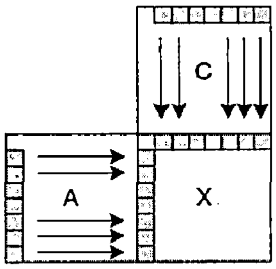


Fig-3.4: Prediction of AC coefficients [7]

The direction of the smallest DC gradient is chosen as the prediction direction for block X. The prediction, P_{DC}, is formed by dividing the DC coefficient of the chosen neighboring block by a scaling factor and P_{DC} is then subtracted from the actual quantized DC coefficient (QDCX) and the residual (PQDCX) is coded and transmitted. AC coefficient prediction is carried out in a similar way, with the first row or column of AC coefficients predicted in the direction determined for the DC coefficient (Fig-3.4). For example, if the prediction direction is from block A, the first column of AC coefficients in block X is predicted from the first column of block A. If the prediction direction is from block C, the first row of AC coefficients in X is predicted from the first row of C. The prediction is scaled depending on the quantizer step sizes of blocks X and A or C.

3.3 DCT and IDCT [16] [19] [22]

The purpose of transform coding is to de-correlate the intra or inter frame error image content and to encode transform coefficients rather than the original pels of the images, to aim this the input images are split into disjoint blocks of pels, b (i.e., of size N x N pels). The transformation can be represented as a matrix operation using an N x N transform matrix, A, to obtain the N x N transform coefficients, c, based on a linear, separable, and unitary forward transformation

$$c = AbA^T \tag{3.1}$$

Here, A^T denotes the transpose of the transformation matrix, A. Note that the transformation is reversible since the original N x N block of pels, b, can be reconstructed using a linear and separable inverse transformation. For a unitary transform the inverse matrix A^{-1} is identical with the transposed matrix A^T, that is $A^{-1} = A^T$ and

$$b = A^T cA \tag{3.2}$$

The purpose of the transform stage in a video CODEC is to convert motion-compensated residual data into another domain (the transform domain). The choice of transform depends on a number of criteria:

1. Data in the transform domain should be decorrelated (separated into components with minimal inter-dependence) and compact (most of the energy in the transformed data should be concentrated into a small number of values).

2. The transform should be reversible.

3. The transform should be computationally tractable (low memory requirement, achievable using limited-precision arithmetic, low number of arithmetic operations, etc .).

The Karhunen - Loeve Transform (KLT) is the best transform for achieving the above points. It transforms the basis set in the sample space into a new basis set such that the greatest energy is contained in the fewest number of transform coordinates in transform space is minimized. In other words, the greatest energy is contained in the earliest coefficients in the basis of the transform space. Formally, we say that the K-L transform should minimize the Mean Square Error in any truncated representation of the samples in the transform space.

DCT is an orthogonal transform; it allows conversion of the spatial representation of an 8x8 image to the frequency domain therefore reducing the number of data points. It makes many transform coefficients small enough so that they are insignificant and can be discarded in the quantization step.

Benefits of DCT

- DCT is proven to be optimal transform for large classes of images.
- DCT coefficients are easily quantized to achieve good compression
- DCT algorithm is efficient and easy to implement
- DCT is data independent transform

3.4 Quantization and Inverse Quantization [16] [19] [22]

The domain transformation of the pixels does not actually yield any compression. A block of 64 pixels is transformed into 64 coefficients. Due to the orthonormality of the transformation, the energy in both the pixel and the transform domains are equal, hence no compression is achieved. However, transformation causes the significant part of the image energy to be concentrated at the lower frequency components, with the majority of the coefficients having little energy. It is the quantization and variable length coding of the DCT coefficients that lead to bit rate reduction. Moreover, by exploiting the human eye's characteristics, which are less sensitive to picture distortions at higher frequencies, one can apply even coarser quantization at these frequencies, to give greater compression. Coarser quantization step sizes force more coefficients to zero and as a result more compression is gained, but of course the picture quality deteriorates accordingly.

3.4.1 MPEG-4 Quantizer Scale [7]

When using the MPEG-4 quantizer scale with the MPEG-4 non-intra quantization matrix, the quantization for INTER mode blocks AC/DC coefficients can be expressed as:

$$\mathrm{coeff}(u,v) = \mathrm{sign}(\mathrm{F}(u,v)) * \frac{(16 * |\mathrm{F}(u,v)|)//\mathrm{w}[u][v]}{2 * QP} \tag{3.3}$$

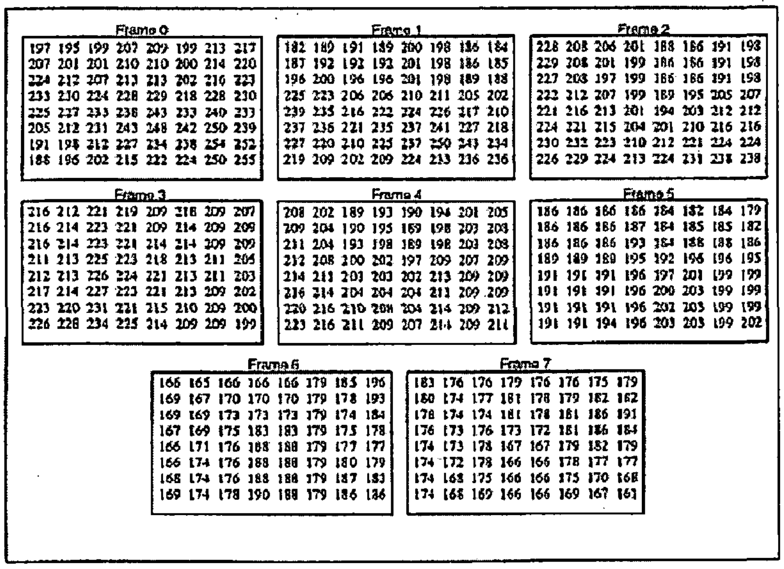
where w is the non-intra quantization matrix. In this case, to guarantee all coeff(u, v)=0, we have

$$|F(u,v)| < 2 * QP \tag{3.4}$$

From above equations we can guarantee that the DC component of the block would be zero if the following condition is satisfied:

$$\frac{1}{8}\text{SAE} < 2 * QP \qquad (3.5)$$
$$\text{SAE} < 16 * QP$$

The below figures are taken from [19] which shows: An example of encoding an 8x8x8 video cube (a) original pixels in video cube, encoding an 8x8x8 video cube. (b) DCT coefficients, after 3D DCT, and (c) quantized DCT coefficients.



Frame 0

197	195	199	207	209	199	213	217
207	201	201	210	210	200	214	220
224	212	207	213	213	202	216	223
233	230	224	228	229	218	228	230
225	237	233	238	243	233	240	233
205	212	231	243	248	242	250	239
191	198	212	227	234	238	254	252
188	196	202	215	222	224	250	255

Frame 1

182	189	191	189	200	198	186	184
187	192	192	192	201	198	186	185
196	200	196	196	201	198	189	188
225	223	206	206	210	211	205	202
239	235	216	222	224	226	217	210
237	236	221	235	237	241	227	218
227	220	210	225	237	250	243	234
219	209	202	209	224	233	236	236

Frame 2

228	208	206	201	188	186	191	198
229	208	201	199	186	186	191	198
227	203	197	199	186	186	191	198
222	212	207	199	189	195	205	207
221	216	213	201	194	203	212	212
224	221	215	204	201	210	216	216
230	232	223	210	212	221	224	224
226	229	224	213	224	231	238	238

Frame 3

216	212	221	219	209	218	209	207
216	214	223	221	209	214	209	209
216	214	223	221	214	214	209	209
211	213	225	223	218	213	211	205
212	213	226	224	221	213	211	203
217	214	227	223	221	213	209	202
223	220	231	221	215	210	209	200
226	228	234	225	214	209	209	199

Frame 4

208	202	189	193	190	194	201	205
209	204	190	195	169	198	203	208
211	204	193	198	189	198	203	208
212	208	200	202	197	209	207	209
214	211	201	203	202	213	209	209
216	214	204	204	204	211	209	209
220	216	210	208	204	214	209	212
223	216	211	209	207	214	209	211

Frame 5

186	186	186	186	184	182	184	179
186	186	186	187	184	185	185	182
186	186	186	193	184	188	188	186
189	189	189	195	192	196	196	195
191	191	191	196	197	201	199	199
191	191	191	196	200	203	199	199
191	191	191	196	202	203	199	199
191	191	194	196	203	203	199	202

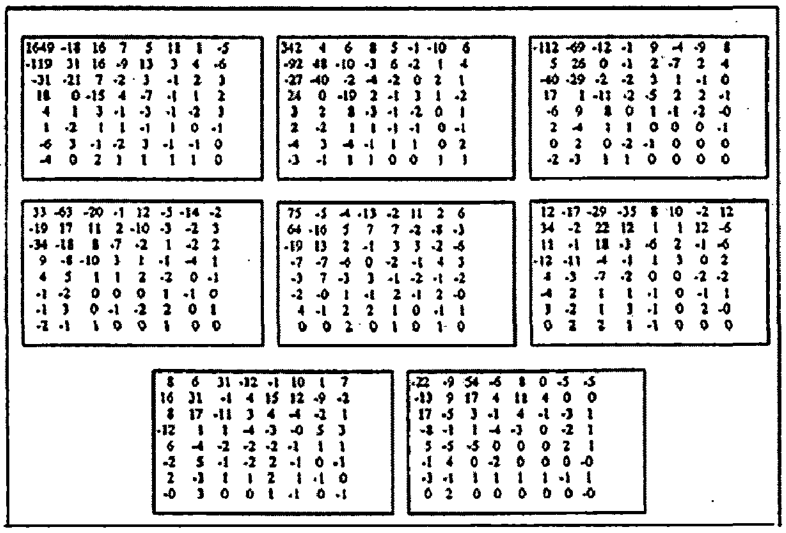
Frame 6

166	165	166	166	166	179	185	196
169	167	170	170	170	179	178	193
169	169	173	173	173	179	174	184
167	169	175	183	183	179	175	178
166	171	176	188	188	179	177	177
166	174	176	188	188	179	180	179
168	174	176	188	188	179	187	183
169	174	178	190	188	179	186	186

Frame 7

183	176	176	179	176	176	175	179
180	174	177	181	178	179	182	162
178	174	174	181	178	181	186	191
176	173	176	173	172	181	186	184
174	173	178	167	167	179	182	179
174	172	178	166	166	178	177	177
174	168	175	166	166	175	170	168
174	168	169	166	166	169	167	161

Fig-3.5: (a) An example of encoding - original pixels in video.



1649	-18	16	7	5	11	1	-5
-119	31	16	-9	13	3	4	-6
-31	-21	7	-2	3	-1	2	3
18	0	-15	4	-7	-1	1	2
4	1	3	-1	-3	-1	-2	3
1	-2	1	1	-1	1	0	-1
-6	3	-1	-2	3	-1	-1	0
-4	0	2	1	1	1	1	0

342	4	6	8	5	-1	-10	6
-92	48	-10	-3	6	-2	1	4
-27	-40	-2	-4	-2	0	2	1
24	0	-19	2	-1	3	1	-2
3	2	8	-3	-1	-2	0	1
2	-2	1	1	-1	-1	0	-1
-4	3	-4	-1	1	1	0	2
-3	-1	1	1	0	0	1	1

-112	-69	-12	-1	9	-4	-9	8
5	26	0	-1	2	-7	2	4
-40	-29	-2	-2	3	1	-1	0
17	1	-13	-2	-5	2	2	-1
-6	9	8	0	1	-1	-2	-0
2	-4	1	1	0	0	0	-1
0	2	0	-2	-1	0	0	0
-2	-3	1	1	0	0	0	0

33	-63	-20	-1	12	-5	-14	-2
-19	17	11	2	-10	-3	-2	3
-34	-18	8	-7	-2	1	-2	2
9	-8	-10	3	1	-1	-4	1
4	5	1	1	2	-2	0	-1
-1	-2	0	0	0	1	-1	0
-1	3	0	-1	-2	2	0	1
-2	-1	1	0	0	1	0	0

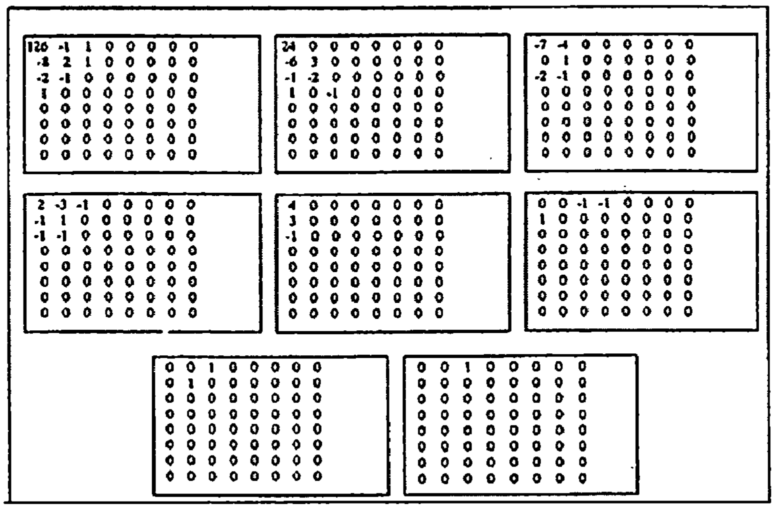
75	-5	-4	-13	-2	11	2	6
64	-16	5	7	7	-2	-8	-3
-19	13	2	-1	3	3	-2	-6
-7	-7	-6	0	-2	-1	4	3
-3	7	-3	3	-1	-2	-1	-2
-2	-0	1	-1	2	-1	2	-0
4	-1	2	2	1	0	-1	1
0	0	2	0	1	0	1	0

12	-17	-29	-35	8	10	-2	12
34	-2	22	12	1	1	12	-6
11	-1	18	-3	-6	2	-1	-6
-12	-11	-4	-1	1	3	0	2
4	-3	-7	-2	0	0	-2	-2
-4	2	1	1	-1	0	-1	1
3	-2	1	3	-1	0	2	-0
0	2	2	1	-1	0	0	0

8	6	31	-12	-1	10	1	7
16	31	-1	4	15	12	-9	-2
8	17	-11	3	4	-4	-3	1
-12	1	1	-4	-3	-0	5	3
6	-4	-2	-2	-2	-1	1	1
-2	5	-1	-2	2	-1	0	-1
2	-3	1	1	2	1	-1	0
-0	3	0	0	1	-1	-0	-1

-22	-9	54	-6	8	0	-5	-5
-13	9	17	4	18	4	0	0
17	-5	3	-1	4	-1	-3	1
-8	-1	1	-4	-3	0	-2	1
5	-5	-5	0	0	0	2	1
-1	4	0	-2	0	0	0	-0
-3	-1	1	1	1	1	-1	1
0	2	0	0	0	0	0	-0

Fig-3.5: (b) DCT coefficients



126	-1	1	0	0	0	0	0
-8	2	1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

24	0	0	0	0	0	0	0
-6	3	0	0	0	0	0	0
-1	-2	0	0	0	0	0	0
1	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

-7	-4	0	0	0	0	0	0
0	1	0	0	0	0	0	0
-2	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

2	-3	-1	0	0	0	0	0
-1	1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

4	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

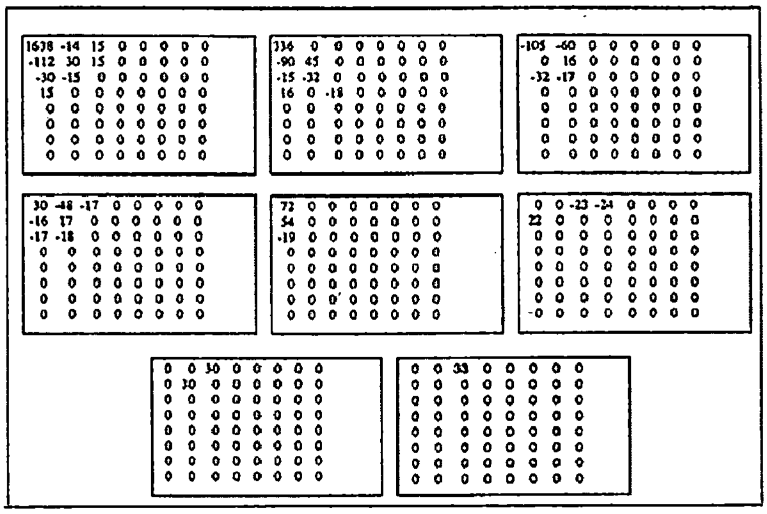
0	0	-1	-1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

0	0	1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Fig-3.5: (c) Quantized DCT coefficients

The below figure shows an example of decoding (d) DCT coefficients after dequantization, (e) Decompressed pixels after inverse DCT.



1638	-14	15	0	0	0	0	0
-112	30	15	0	0	0	0	0
-30	-15	0	0	0	0	0	0
15	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

336	0	0	0	0	0	0	0
-90	45	0	0	0	0	0	0
-15	-32	0	0	0	0	0	0
16	0	-18	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

-105	-60	0	0	0	0	0	0
0	16	0	0	0	0	0	0
-32	-17	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

30	-48	-17	0	0	0	0	0
-16	17	0	0	0	0	0	0
-17	-18	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

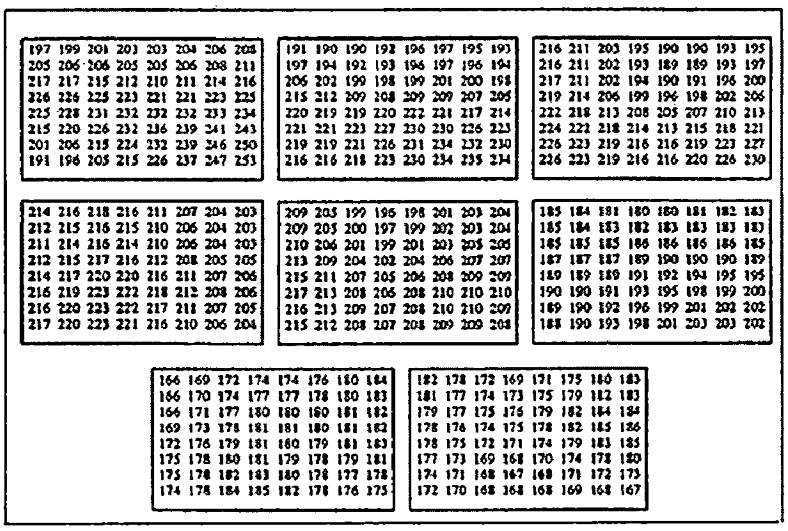
72	0	0	0	0	0	0	0
54	0	0	0	0	0	0	0
-19	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0′	0	0	0	0	0
0	0	0	0	0	0	0	0

0	0	-23	-24	0	0	0	0
22	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
-0	0	0	0	0	0	0	0

0	0	30	0	0	0	0	0
0	30	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

0	0	33	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Fig-3.5: (d) DCT coefficients after dequantization



197	199	201	203	203	204	206	208
205	206	206	205	205	206	208	211
217	217	215	212	210	211	214	216
226	226	225	223	221	221	223	225
225	228	231	232	232	232	233	234
215	220	226	232	236	239	241	243
201	206	215	224	232	239	246	250
191	196	205	215	226	237	247	253

191	190	190	192	196	197	195	193
197	194	192	193	196	197	196	194
206	202	199	198	199	201	200	198
215	212	209	208	209	209	207	205
220	219	219	220	222	221	217	214
221	221	223	227	230	230	226	223
219	219	221	226	231	234	232	230
216	216	218	223	230	234	235	234

216	211	203	195	190	190	193	195
216	211	202	193	189	189	193	197
217	211	202	194	190	191	196	200
219	214	206	199	196	198	202	206
222	218	213	208	205	207	210	213
224	222	218	214	213	215	218	221
226	223	219	216	216	219	223	227
226	223	219	216	216	220	226	230

214	216	218	216	211	207	204	203
212	215	216	215	210	206	204	203
211	214	216	214	210	206	204	203
212	215	217	216	212	208	205	205
214	217	220	220	216	211	207	206
216	219	223	222	218	212	208	206
216	220	223	222	217	211	207	205
217	220	223	221	216	210	206	204

209	205	199	196	198	201	203	204
209	205	200	197	199	202	203	204
210	206	201	199	201	203	205	206
213	209	204	202	204	206	207	207
215	211	207	205	206	208	209	209
217	213	208	206	208	210	210	210
216	213	209	207	208	210	210	209
215	212	208	207	208	209	209	208

185	184	181	180	180	181	182	183
185	184	183	182	183	183	183	183
185	185	185	186	186	186	186	185
187	187	187	189	190	190	190	189
189	189	189	191	192	194	195	195
190	190	191	193	195	198	199	200
189	190	192	196	199	201	202	202
188	190	193	198	201	203	203	202

166	169	172	174	174	176	180	184
166	170	174	177	177	178	180	183
166	171	177	180	180	180	181	182
169	173	178	181	181	180	181	182
172	176	179	181	180	179	181	183
175	178	180	181	179	178	179	181
175	178	182	183	180	178	177	178
174	178	184	185	182	178	176	175

182	178	172	169	171	175	180	183
181	177	174	173	175	179	182	183
179	177	175	176	179	182	184	184
178	176	174	175	178	182	185	186
178	175	172	171	174	179	183	185
177	173	169	168	170	174	178	180
174	171	168	167	168	171	172	173
172	170	168	168	168	169	168	167

Fig-3.5: (e) Decomposed pixels after inverse DCT

3.5 Scan [16]

After quantization, the DCT coefficients for a block are reordered to group together nonzero coefficients, enabling efficient representation of the remaining zero-valued quantized coefficients. The optimum reordering path (scan order) depends on the distribution of nonzero DCT coefficients. For a typical frame block, a suitable scan order is a zigzag starting from the DC (top-left) coefficient. Starting with the DC coefficient, each quantized coefficient is copied into a one-dimensional array in the order shown in Fig-3.6. Nonzero coefficients tend to be grouped together at the start of the reordered array, followed by long sequences of zeros.

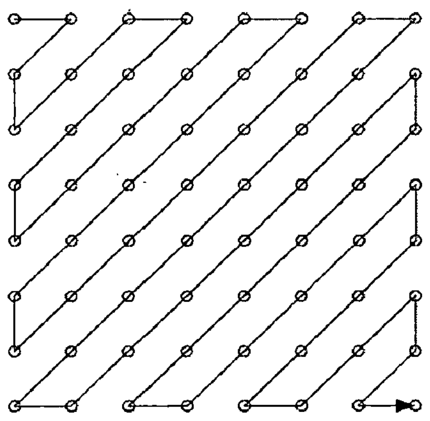


Fig-3.6: Zig-zag scan

3.6 Run-Level Encoding [16][19]

The output of the reordering process is an array that typically contains one or more clusters of nonzero coefficients near the start, followed by strings of zero coefficients. The large number of zero values may be encoded to represent them more compactly, for example by representing the array as a series of (run, level) pairs where run indicates the number of zeros preceding a nonzero coefficient and level indicates the magnitude of the nonzero coefficient.

Higher-frequency DCT coefficients are very often quantized to zero and so a reordered block will usually end in a run of zeros. A special case is required to indicate the final nonzero coefficient in a block. In 'Two-dimensional' run-level encoding is used, each run-level pair is encoded as above and a separate code symbol, 'last', indicates the end of the nonzero values.

3.7 Variable length coding [16][19]

For further bit rate reduction, the transform coefficients and the coordinates of the motion vectors are variable length coded (VLC). In VLC, short code words are assigned to the highly probable values and long code words to the less probable ones. The lengths of the codes should vary inversely with the probability of occurrences of the various symbols in VLC. The bit rate required to code these symbols is the inverse of the logarithm of probability, p, at base 2 (bits). Hence, the entropy of the symbols which is the minimum average bits required to code the symbols can be calculated as:

$$H(x) = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{3.6}$$

There are two types of VLC, which are employed in the standard video codecs. They are Huffman coding and arithmetic coding. Huffman coding is a simple VLC code, but its compression can never reach as low as the entropy due to the constraint that the assigned symbols must have an integral number of bits. However, arithmetic coding can approach the entropy since the symbols are not coded individually. Huffman coding is employed in all standard codecs to encode the quantized DCT coefficients as well as motion vectors. Arithmetic coding is used, for example, in JPEG, JPEG2000, H.263 and shape and still image coding of MPEG-4, where extra compression is demanded.

3.8 Buffer [16] [19]

The bit rate generated by an inter-frame coder is variable. This is because the bit rate is primarily a function of picture activity (motion of objects and their details). Therefore, to transmit coded video into fixed rate channels, the bit rate has to be regulated. Storing the coded data in a buffer and then emptying the buffer at the channel rate does this. However, if the picture activity is such that the buffer may overflow (violent motion) then a feedback from the buffer to the quantizer can regulate the bit rate. Here, as the buffer occupancy increases, the feedback forces the quantizer step size to be increased to reduce the bit rate. Similarly, if the picture activity is less, then the quantizer step size is reduced to improve the picture quality.

3.9 TI64XX – Optimization [23] [24]

Performance is a key issue for embedded systems developers. Profiling provides better insight into the system performance. It is the technique used to determine how long a processor spends in each section of a program. Profiling can be done for the entire project, for individual files as well as for a specific range. MIPS and memory optimization is the major concern. Both are inversely proportional. Different compiler options are provided in C6x for better optimization of the code.

Methods of optimization

- Compiler options
- C-level
- Intrinsics
- Linear assembly
- Scheduled assembly

3.9.1 Compiler Options

Options control the operation of the compiler. They can be set for the whole project or individual files. Some options are for speed (MIPS/Cycle) improvement and some are intended for memory reduction. Some options are obsolete or intended only for debugging and could potentially decrease performance and increase code size. It is the programmer's responsibility to decide optimal combination of options.

Four options are available for performance improvement:

(Listed in the increasing order of performance improvement)

- -o0 (Register) :

  Enables register-level optimizations

  Allocates variables to registers

  Performs loop rotation

  Eliminates unused code

  Performs control-flow-graph simplification

  Simplifies expressions and statements

  Expands calls to functions declared as inline

- -o1 (Local) :

  Performs all –o0 optimizations, and

  Enables local optimizations. Performs local copy/constant propagation

  Removes unused assignments

  Eliminates local common expressions

- -o2 (Function) :

  Performs all –o1 optimizations, and

  Enables function-level optimizations

  Performs software pipelining

  Performs loop optimizations such as loop unrolling

  Eliminates global common sub expressions

  Eliminates global unused assignments

  Converts array references in loops to incremented pointer form

- -o3 (File) :

  Performs all –o2 optimizations, and

  Represents the highest level of optimization available

  Enables file-level optimizations

  Removes all functions that are never called

  Simplifies functions with return values that are never used

  In lines calls to small functions

  Various loop optimizations are performed such as using SIMD instruction.

Four levels are available for code size reduction :

- Speed Most Critical (no ms)
- Speed More Critical (-ms0)
- Speed Critical (-ms1)
- Size Critical (-ms2)
- Size Most Critical (-ms3)

These options cause the compiler to increasingly sacrifice performance in favor of reducing code size. It is recommended that a code size flag not be used with the most performance-critical code. Using -ms0 or -ms1 is recommended for all but the most performance-critical code. Using -ms2 or -ms3 is recommended for seldom-executed code

-pm: With program-level optimization, all the source files are compiled into one intermediate file called a module.

The compiler can perform the following optimizations:
If a particular argument in a function always has the same value, the compiler replaces the argument with the value and passes the value instead of the argument. If a return value of a function is never used, the compiler deletes the return code in the function. If a function is not called directly or indirectly, the compiler removes the function.
Along with –pm the following options can be used:

- op3 (No External Variable Refs): Specifies that the module has functions that are called from other modules but does not have global variables that are modified in other module.
- op2 (No External Function / Variable Refs): Specifies that the module does not have functions that are called by other modules or global variables that are modified in other modules.
- op1 (No External Function Refs): Specifies that the module does not have functions that are called by other modules but has global variables that are modified in other modules.
- op0 (External Function/Variable Refs): Specifies that the module has functions that are called from other modules and global variables that are modified in other modules.

  These options improve variable analysis and allowed assumptions.
- -g (Full Symbolic Debug): Produces symbolic debugging directives that enable C source-level debugging and assembly source debugging.
- No debug: Disables any previously specified debug information option.

3.9.2 RTS Modifications:

-ol2 (Defines no RTS Functions): Defines no RTS Functions informs the optimizer that your file does not declare or alter library functions.
-ol1 (Contains RTS functions): Informs the optimizer that your file declares a standard library function.

-ol0 (Alters RTS Functions): Informs the optimizer that your file alters one or more standard library functions

Memory Models

• -ml/-ml0	:	Aggregate data (structures / arrays) default to far
• -ml1	:	All calls default to far
• -ml2	:	All aggregate data and calls default to far
• -ml3	:	All calls and all data default to far

If no level is specified, all data and functions default to near.

RTS (Run-time support) Calls

Use Memory Model disables any previously enabled run-time-support calls option.

- -mr0 (Are Near): Makes calls to run-time-support functions near.
- -mr1 (Are Far) : Makes calls to run-time-support functions far.

Aliasing

Default disables any aliasing options previously selected.

- -ma (Aliased Variables) : Assumes aliased variables.
- -mt (No Bad Alias Code): Allows the compiler to make certain assumptions about aliasing and loops.
- -oi (Auto Inlining Threshold) : Specifies the automatic in lining size (level 3 only) for the –oi size option.
- -min : Specifies an interrupt threshold value
- -mu : Turns off software pipelining
- -mo : Turns on function subsections so the linker places each function in a file in its own subsection

3.9.3 Assembly Coding

- Porting C code to specific processor.
- Time Consuming and Error prone.
- Maintenance is difficult.

C6000 code Generation tools

- Optimize C code rather than writing hand assembly.
- Compiler's task: instruction selection, parallelizing, pipelining, register selection.

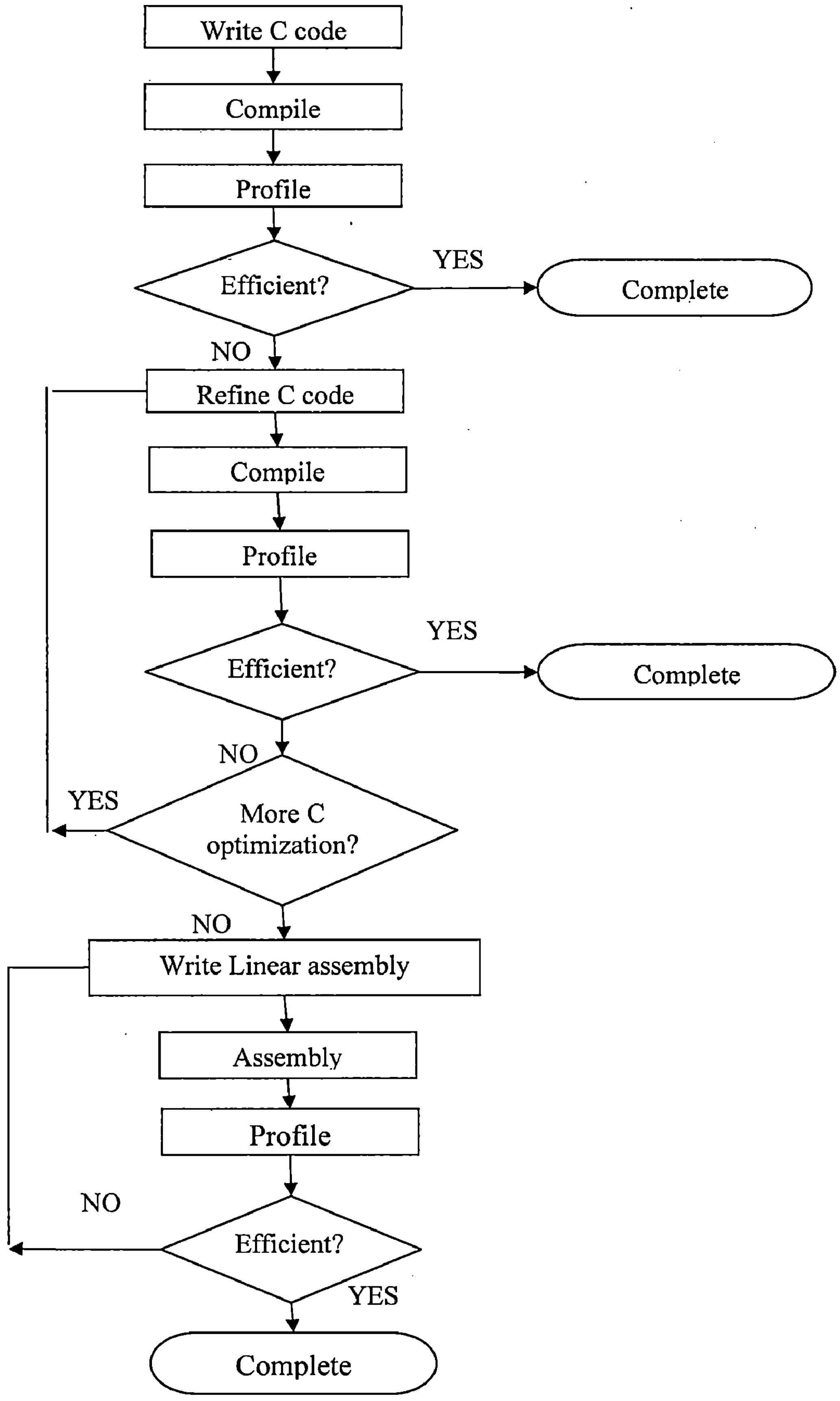


Fig- 3.7. Code Development Flow to Increase Performance

Provide the programmer feedback what it understood

- Programmer's task: Understand the C code properly.
- Provide compiler enough info to fully maximize its potential.

Phases of Code Development

- Phase 1: Compile and profile C code.
- Determine cycle consuming loops.
- Phase 2: Add restrict qualifier loop, loop iteration count, memory bank and data alignment info.
- Phase 3: Optimize C code using intrinsics..

3.9.4 Linear Assembly

Linear assembly is the input for the assembly optimizer. A linear assembly file must be specified with a .sa extension. All of the information that is specified in regular C6000 assembly code need not be mentioned here.

The following need not be specified:

Parallel instructions

Pipeline latency

Register usage

Functional unit being used

If above points are not mentioned, the assembly optimizer determines on the information that is not included, based on the information that it has about the code. Linear assembly code should include the .cproc and .endproc directives. .cproc is used at the beginning of the section and .endproc at the end of the section. The .reg directive allows to use descriptive names for values that will be stored in registers. The .trip directive specifies the value of the trip count. The trip count indicates how many times a loop will iterate. The .mdep directive specifies dependence between two or more memory references .The .no_ mdep directive tells the assembly optimizer that no memory dependences occur within that function

3.9.5 SCHEDULED ASSEMBLY - C64x

Scheduled assembly is an extension of linear assembly. It is a fully hand-coded assembly code. No further optimization by assembler. It is the final optimized code. Better performance than linear assembly.

Need to provide the following info.: (which is not provided in case of linear assembly)

Parallel instructions

Pipeline latency

Register usage

Which functional unit is being used (optional)

Writing kernel with optimal number of cycles is the programmer's greatest challenge. Kernel involves all the basic functionality of the loop. In other words, it is the "Heart"of the loop. Epilog is optional. If required, epilog always performs the last iteration of the loop.

- Two general-purpose register files (A and B).
- Eight functional units (L1, .L2, .S1, .S2, .M1, .M2, .D1, and .D2) resulting in 8 parallel instructions in single cycle. Four functional units on each side A and B.
- Two load-from-memory data paths (LD1 and LD2) resulting in 2 loads in parallel in single cycle but in different data paths.
- Two store-to-memory data paths (ST1 and ST2) resulting in 2 stores in parallel in single cycle but in different data paths.
- Two data address paths (DA1 and DA2)
- Two register file data cross paths (1X and 2X)
- Specific instructions executed in functional units :

L1 & L2	-	Logical instructions
S1 & S2	-	Shift Instructions
M1 & M2	-	Multiplication instructions
D1 & D2	-	Load/Store Instructions

- 32/64 accumulators (A0 to A15 / A31 & B0 to B15 / B31) in C62x/C64x respectively.

Instructions with delay slots

• Load	-	4 Delay slots
• Branch	-	5 Delay slots
• Multiply	-	1 Delay slot

Advantages

- Best possible optimization.
- Good utilization of the available resources.
- Good optimization of the loops. In other words, better way of handling the MIPS intensive loops.
- Good performance numbers (MIPS) of the overall algorithm.
- Best way of handling the delay slots.

CHAPTER 4

Motion Estimation

The different types of motion estimation algorithms which are available in literature [5] are:

1. Frequency-Domain Method.
2. Gradient Method.
3. Pel-Recursive Method.
4. Block Matching Method.

Before discussing about the Block Matching Motion estimation techniques which is the main focus of this dissertation work .Lets briefly discuss about the other techniques.

4.1 Frequency- Domain Techniques

It consists of three methods:

1. DCT-based Method
2. Wavelet Based Method
3. Phase Correlation Method.

Some advantages [17] of the frequency approach include:

1. It is robust to global illumination changes.
2. Inaccuracies in motion estimation near the object boundaries are avoided, since the estimation is not based on spatially local inter-frame luminance differences rather on global intensity distribution.
3. Efficient algorithms are available for FFT computation.
4. There is no need for the commonly used separate stages of feature detection and matching as in feature based spatial method.

Disadvantages [17] with this method:

1. The main difficulty they encounter is called “localization problem”, they are able to detect and estimations, but they do not directly relate each estimate to the associated frame pixels, thus not providing object localization.
2. It is computationally intensive.

4.1.1 Phase Correlation Method for multiple motion [17]

Let each frame consist of M moving objects i , $1 \le i \le M$, with luminance $s_i(\bar{r})$ at pixel $\bar{r}$ and displacement $\bar{b}_i(n)$ at frame n. The FT of object *i* is $S_i(\bar{\omega}) = M_i(\bar{\omega})e^{j\Phi_i(\omega)}$ where $\bar{\omega} = [2\pi m/N_1, 2\pi m/N_2]^T, m,n \in Z$ is the 2-D frequency, N1 ×N2 the image size, $M_i(\bar{\omega})$ the FT magnitude, and $\Phi_i(\omega)$ the FT phase. The FT of frame k is $A(\bar{\omega},k)$ $1 \le k \le N$, the measurement noise is $V_{noise,k}(\bar{\omega})$, and the background area hidden by the moving objects is denoted by $V_{bck,k}(\bar{\omega})$. Then the FT of frame k is

$$A(\bar{\omega},k) = S_b(\bar{\omega}) + \sum_{i=1}^{M} S_i(\bar{\omega})e^{-j\bar{\omega}^T\bar{b}_i(k)} - V_{bck,k}(\bar{\omega}) + V_{noise,k}(\bar{\omega}) \tag{4.1}$$

Stacking the FT's of the N frames, we get the over-determined system $A = HS + V_{noise} - V_{bck}$ where A is an N × 1 data vector, with the values of each frame's FT, and H is an $N \times (M + 1)$ matrix, containing the motion information, with row k $H_k(\bar{\omega}) = [1, e^{-j\bar{\omega}^T\bar{r}_1(k)}, \cdots, e^{-j\bar{\omega}^T\bar{r}_M(k)}]$. V_{noise} is the N × 1 vector of measurement noise, and $V_{bck,k}$ represents the occluded background areas.

For the translation between frames 1 and k, we consider the ratio $\Phi_{1,k}$ of their F.Ts:

$$\Phi_{1,k}(\bar{\omega}) = \frac{A(\bar{\omega},k)}{A(\bar{\omega},1)} = \gamma_b(\bar{\omega}) + \sum_{i=1}^{M} \gamma_i(\bar{\omega})e^{-j\bar{\omega}^T\bar{b}_i(k)} + n_k(\bar{\omega}). \tag{4.2}$$

$$\gamma_b(\bar{\omega}) = S_b(\bar{\omega}) \Big/ \left(S_b(\bar{\omega}) + \sum_{i=1}^{M} S_i(\bar{\omega}) + V(\bar{\omega},1)\right), \tag{4.3}$$

$$\gamma_i(\bar{\omega}) = S_i(\bar{\omega}) \Big/ \left(S_b(\bar{\omega}) + \sum_{i=1}^{M} S_i(\bar{\omega}) + V(\bar{\omega},1)\right), \tag{4.4}$$

$$and\, n_k(\bar{\omega}) = \frac{\left(V_{noise,k}(\bar{\omega}) - V_{bck,k}(\bar{\omega})\right)}{\left(S_b(\bar{\omega}) + \sum_{i=1}^{M} S_i(\bar{\omega}) + V(\bar{\omega},k)\right)} \tag{4.5}$$

$$V(\bar{\omega},k) = V_{noise,k}(\bar{\omega}) - V_{bck,k}(\bar{\omega}) \tag{4.6}$$

The inverse FT $\Phi_{1,k}(\bar{r})$ is a weighted sum of delta functions, whose peaks give the M displacements between frames 1 and k.

$$\phi_{1,k}(\bar{r}) = a_b(\bar{r}) + \sum_{i=1}^{M} a_i(\bar{r})\delta(\bar{b} - \bar{b}_i(k)) + n_k(\bar{r}) \tag{4.7}$$

This yields the 2D trajectory $\bar{b}_i(k)$ of each object. Displacements between frames 1 and k can be estimated by simply dividing the motion estimates $\bar{b}_i$ by k − 1.

4.1.2 DCT-based method [9]

In the conventional DCT-based video coder, the feedback loop has the following functional blocks: DCT, Inverse DCT (IDCT), quantizer, inverse quantizer, and spatial domain motion estimation and compensation. However, if motion estimation and compensation are performed entirely in the DCT domain, IDCT can be removed from the feedback loop. Therefore, the feedback loop contains the reduced functional blocks: quantizer, inverse quantizer, and transform domain motion estimation and compensation. However, high computational complexity is still the main drawback of the DCT-based motion estimation and compensation approach.

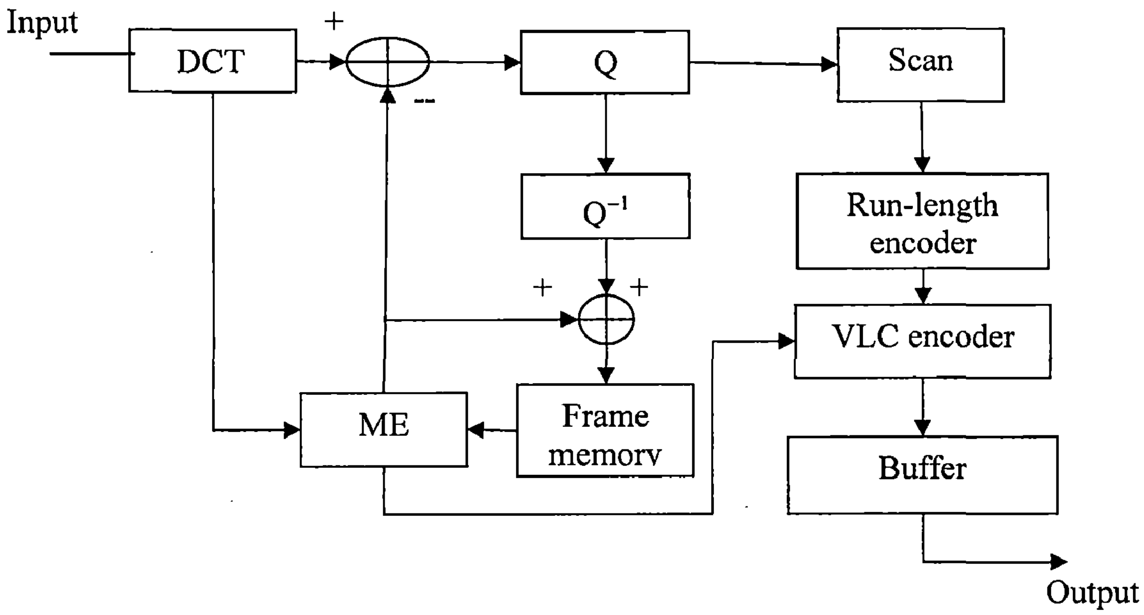


Fig 4.1. Simplified video encoder with motion estimation and compensation in the transform domain [9]

The DCT $N \times N$ matrix $T = \{t(k,n)\}$, where $t(k,n)$ represents the matrix entry in the kth row and the nth column, is given by

$$H_0 = D^t_{8-\Delta y}, H_1 = D_{\Delta y}, H_2 = D^t_{8-\Delta y}, H_3 = D^t_{\Delta x}$$

The one-dimensional DCT of a sequence $x(n), 0 \le n \le N-1$ is given by

$$y(k) = \sum_{n=0}^{N-1} t(k,n)x(n), \qquad 0 \le k \le N-1 \tag{4.8}$$

We will concentrate on the case of N=8, since image and video compression standards use 8x8 block DCT. The shifting matrix-based technique is to perform integer-pixel accurate motion estimation and compensation by multiplications of shifting matrices and previous block matrices in the DCT domain. It shows that better visual quality can be obtained by transform domain motion estimation and compensation method. The following is the basic idea of the shifting matrix-based algorithm. A predicted block can be represented as a summation of horizontal and vertical shifted versions of the four surrounding blocks $f_{0,}f_1, f_2$ and f_3 which can be calculated by applying vertical shifting matrix and horizontal shifting matrix. The best estimate for the current block and the corresponding motion vector are indicated in Fig-3.2. The predicted block can be obtained by

$$f_{pred} = \sum_{i=0}^{3} V_i f_i H_i. \tag{4.9}$$

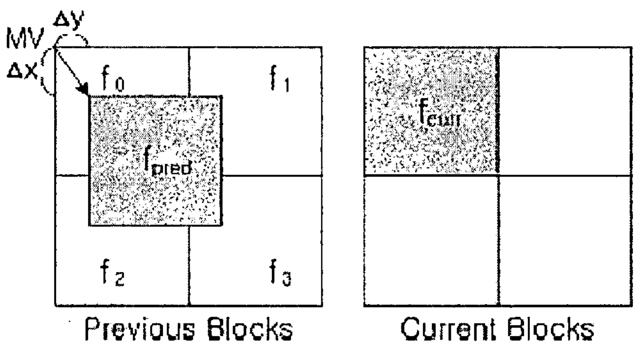


Fig 4.2 . Motion estimation in transform domain

The vertical shifting matrix V_i and the horizontal shifting matrix H_i are given by

$$V_0 = D_{8-\Delta x}, V_1 = D_{8-\Delta x}, V_2 = D^t_{\Delta x}, V_3 = D^t_{\Delta x}$$

$$H_0 = D^t_{8-\Delta y}, H_1 = D_{\Delta y}, H_2 = D^t_{8-\Delta y}, H_3 = D_{\Delta y}$$

displacement matrix D_n is defined as

$$D_n = \left[\begin{array}{c|c} 0 & \mathrm{I}_n \\ \hline 0 & 0 \end{array}\right]$$

where $A_i = \dfrac{a_i + b_i + c_i + d_i}{4}$ I_n is $n \times n$ identity matrix.

Therefore, the DCT coefficients of the predicted block f_{pred} can be obtained using the DCT coefficients of the previous surrounding blocks $f_{0,}f_1, f_2, f_3$ and the pre-calculated DCT coefficients of the horizontal shifting matrix H_i and the vertical shifting matrix V_i , as

$$\begin{aligned} f_{pred} &= DCT(f_{pred}) \\ &= DCT\left(\sum_{i=0}^{3} V_i f_i H_i\right) \\ &= \sum_{i=0}^{3} (TV_iT^t)(Tf_iT^t)(TH_iT^t) \\ &= \sum_{i=0}^{3} \hat{V}_i \hat{f}_i \hat{H}_i \end{aligned} \qquad (4.10)$$

$$\hat{V}_i = TV_iT^t,\ \hat{f}_i = Tf_it^t,\ \hat{H}_i = TH_iT^t.$$

In motion estimation and compensation, the location of the best estimate DCT coefficients block $\hat{f}_{pred}$ for the current DCT coefficients block $\hat{f}_{curr}$ provides the corresponding motion vector and the differences between $\hat{f}_{curr}$ and $\hat{f}_{pred}$ are encoded.

4.1.3 Wavelet-based Method [6] [8]

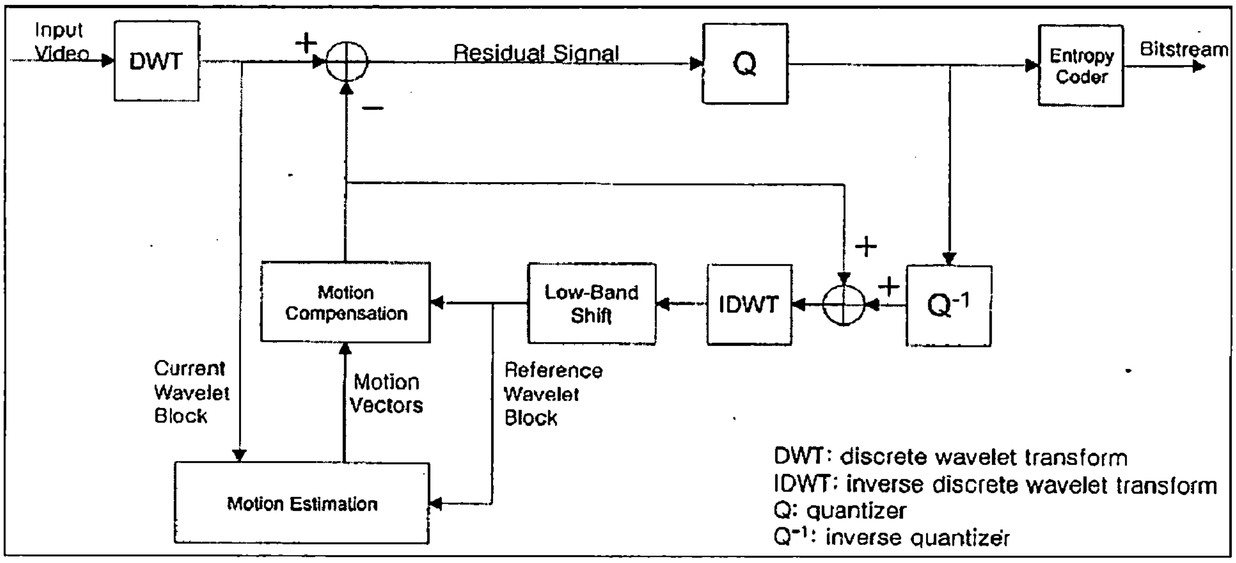


Fig. 4.3. Block diagram of the motion estimation and compensation scheme for wavelet-based video coding. [6]

WAVELET theory provides a unified framework for a number of techniques that have been developed independently for various applications such as multi resolution signal processing for computer vision, sub-band coding for speech and image compression, and wavelet series expansions in applied mathematics. As an alternative to the classical Short-Time Fourier Transform (STFT) or Gabor Transform (GT), the Wavelet Transform (WT) provides an elegant solution for the analysis of non stationary signals. In contrast to the STFT or GT, both of which use only a single analysis window, the WT uses short windows at high frequencies and long windows at low frequencies. This is in the spirit of so-called constant relative bandwidth frequency or "constant-Q" frequency analysis. This tradeoff in the time/space frequency resolution is very useful for the analysis of image/video signals, which are typically non stationary. In image compression, the conventional transform coding techniques, such as those using the DCT, decompose images into a representation in which each coefficient corresponds to a fixed-size spatial area and a fixed frequency bandwidth, where the bandwidth and spatial area are essentially the same for all of the coefficients in the representation. As a consequence, an image coder that fully exploits the multi resolution representation, such as the embedded zero tree wavelet (EZW) coder or the set partitioning in hierarchical trees (SPIHT) coder is able to achieve a better performance than that of a DCT-based coder.

In order to choose the wavelets from the point of view of comparing their performance for MRME of video sequences, there are two characteristics of the wavelets that should be taken into consideration. First, the wavelet should provide a good performance for the coding of still images, since the reference frame for MRME is encoded in the intra mode. Second, the wavelet should have a reasonably small number of coefficients so that the computational load required for the wavelet decomposition of a video frame is not excessively large.

Motion compensation in the wavelet domain is highly dependent on the alignment of the signal and the discrete grid chosen for the analysis. There exist very large differences between the wavelet coefficients of the original image and the one-pixel-shifted image. This shift-variant property happens frequently around the image edges, so motion compensation of the wavelet coefficients can be difficult.

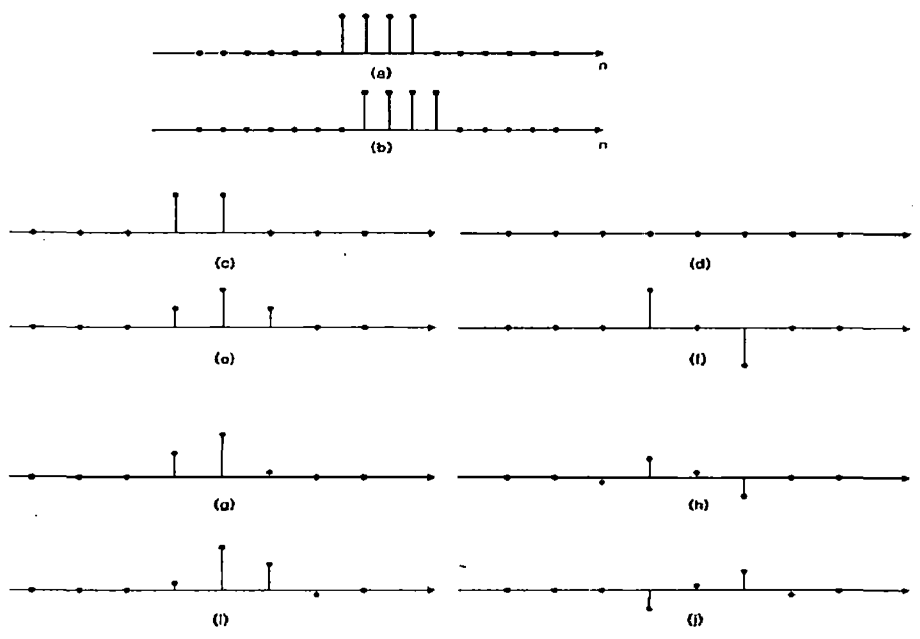


Fig.4.4.Examples of the DWT coefficients from the Haar and the D (9, 7) filters for a 1-D signal s (n) and 1-pixel-shifted 1-D signal s (n – 1) [6]

When the low-band signal is smooth and the difference in the low-band coefficients between the original and the shifted signal is small. Thus, it is possible to estimate the low-band coefficients of the shifted signal from those of the original signal with small error. However, there is a big difference between the high-band coefficients of the shifted signal and those of the original signal. Such phenomena will happen frequently around the edges in the image. The high-band signal difference around the image edges generates large errors in predicting the motion vector in the wavelet domain when using the conventional block-matching method.

4.2 Bayesian Method [5]

We first introduce the notation used in the following sections. Let I(x, y, t) be the image intensity at time instant t at location r = (x, y) and d = (dx, dy) is displacement during time interval t. All techniques rely on the assumption that change in image intensity is only due to the displacement d [4], i.e.

$$I(r,t) = I(r-d, t-\Delta t) \tag{4.11}$$

Gradient Techniques

The first assumption in gradient techniques is that image luminance is invariant during motions. Taylor's series expansion of right hand side of would give

$$I(r-d,t-\Delta t)=I(r,t)-d.\nabla I(r,t)-\Delta t\frac{\partial I(r,t)}{\partial t}+\text{higher order terms}$$

where $\nabla=[(\partial/\partial x),(\partial/\partial y)]$ is the gradient operator and by assuming $\Delta t\to 0$ neglecting higher order terms, and defining the motion vector as v = (vx, vy) $=d/\Delta t$ we obtain

$$v.\nabla I(r,t)+\frac{\partial I(r,t)}{\partial t}=0 \tag{4.12}$$

which is known as spatio- temporal constraint. Since the motion vector has two components, the motion field can be solved only by introducing an additional constraint. Additional constraint known as smoothing constraint is introduced in that minimizes optical flow gradient magnitude. The motion field is obtained by minimizing the following error term defined as

$$\iint\left\{(v.\nabla I+\frac{\partial I}{\partial t})^2+\alpha^2\left[\left(\frac{\partial v_x}{\partial x}\right)^2+\left(\frac{\partial v_x}{\partial y}\right)^2+\left(\frac{\partial v_y}{\partial x}\right)^2+\left(\frac{\partial v_y}{\partial y}\right)^2\right]\right\}$$

where α^2 is a minimization factor. This optimization problem can be solved by variational calculus. Many variations of the above algorithm are proposed in literature .From coding perspective, these motion estimation methods suffer from two main drawbacks. First, the prediction error has high energy due to smoothness constraint, and second, the motion field requires high motion overhead.

4.3 Pel-recursive method [5]

These methods rely on recursive reduction of predictive error or displacement frame difference (DFD). The DFD or frame dissimilarity measure is denoted by

$$DFD(r,t,d)=I(r,t)-I(r-d,t-\Delta t) \tag{4.13}$$

These methods are among the very first algorithms designed for video coding with the goal of having low hardware complexity. The first pel-recursive algorithm was proposed to minimize DFD2 by applying steepest descent technique. The displacement d at (k + 1) iteration is given by

$$\vec{d}^{(k+1)} = \vec{d}^{(k)} - \frac{\varepsilon}{2}\nabla_{\vec{d}}\mathrm{DFD}^2(\vec{r},t,\vec{d}^{(k)}) \tag{4.14}$$

where ε is a constant gain, k is the iteration index, and rd is the gradient vector with respect to the displacement d. Substituting DFD, i.e. , in above formula, we obtain

$$\nabla_{\vec{d}}\mathrm{DFD}^2(\vec{r},t,\vec{d}) = 2\mathrm{DFD}(\vec{r},t,\vec{d}).\nabla_{\vec{r}}I(\vec{r}-\vec{d},t-\Delta t). \tag{4.15}$$

The displacement field update is obtained as follows

$$\vec{d}^{(k+1)} = \vec{d}^{(k)} - \varepsilon\mathrm{DFD}(\vec{r},t,\vec{d}^{(k)}).\nabla_{\vec{r}}I(\vec{r}-\vec{d}^{(k)},t-\Delta t). \tag{4.16}$$

The performance of pel-recursive algorithms strongly depends on the way the above equation is updated. Various research works in literature have focused on proposing efficient methods for computation of above formula. Causality constraints reduce the predictive capability of these algorithms comparing to non-casual methods. High computational complexity is another drawback of pel-recursive algorithms. Furthermore, the error function to be minimized has generally many local minima. These algorithms are also very sensitive to noise and large displacements and discontinuities in the motion field which cannot be efficiently handled.

4.4 Block Matching Method [12] [26]

Fast block –matching algorithms are designed mainly in two lines, i.e., reducing the number of checking points and lowering computational complexity at each of the checking points.

The fast search algorithms can be classified into following categories.

1) Partial-Search-Set Techniques: - In the first category, the fast search algorithms seek for a way to select a subset of the candidate MB in to reduce the computational time. The efficiency of these techniques depends on the number of the selected search positions, while the resulted minimum matching error depends on how the search positions are selected. Consequently, the design issue of these techniques is how to find the motion vector with small minimum matching error

by examining as smaller number of search positions as possible. The gradient descent techniques, based on the uni-modal error surface assumption, belong to this category. Starting from the initial value, the motion vector with the local minimum matching error, SAD, can be found by using pixel-by-pixel search along the conjugate directions or simply the horizontal and vertical directions. The number of the search positions examined depends on the distance between the initial position and the "valley" position with the local minimum SAD. The minimum SAD obtained is sometimes larger than the global minimum SAD because the search process may be trapped at a local minimum. Some other gradient descent techniques use the multi resolution search space of motion vectors for the coarse-to-fine search. These techniques divide the search process into several search steps. Starting from the origin position, SADs of several coarsely-spaced search positions (i.e., in coarse resolution) are calculated and the one with the minimum SAD is selected as the new starting position of the next step. This procedure is repeated several times with smaller and smaller spacing between the search positions (i.e., in finer resolution) until the search positions with spacing of one pixel are examined. The final search position with the minimum SAD is selected as the search result. Well-known examples of this kind of techniques include the three-step search (TSS) algorithm, the two-dimensional logarithmic search algorithm, and the cross-search algorithm. The number of the search positions examined is roughly constant. Thus the computational cost of motion estimation is also roughly fixed. This is a good feature especially for on-line video compression applications, for example, video conference, which prefer static video compression speed. Another important technique in this category is to restrict the search region in a smaller region determined by the predicted motion vector. The value of the motion vector can be predicted from the motion vector of the spatially and temporally adjacent blocks together with the hierarchical related blocks because the motion vectors of these blocks are statistically coherent.

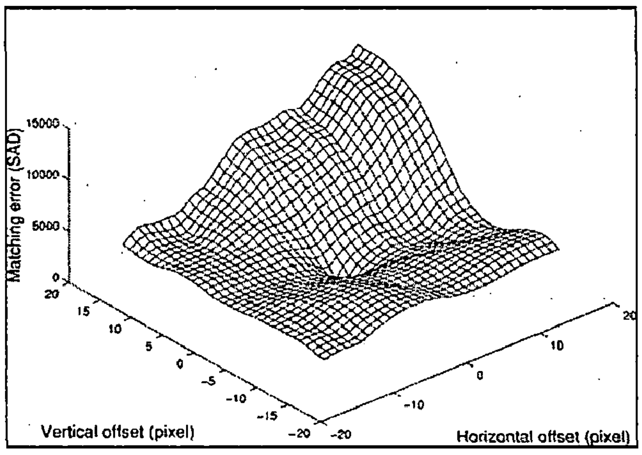


Fig 4.5. Uni-modal error surface [25]

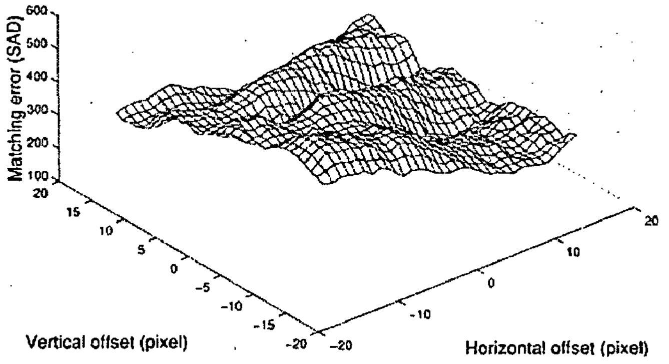


Fig.4.6.Non-uni-modal error surface [25]

2) Algorithms in this category use a reduced complexity distortion measure to save computation.

a) Algorithms using pixel decimation / sub-sampling techniques where only a fraction of pixels at each searching location are used.

b) Partial distortion search algorithms using halfway-stop techniques, where the complexity of the image area is recently used to determine the partial distortion calculation order such as pixel decimation and partial distortion (PDS) techniques. The pixel decimation technique sub-samples the pixels in the target MB and the candidate MBs. Hence, the computation for each SAD can be reduced. The PDS reduces the computation complexity by terminating the SAD calculation early when it finds that a partial SAD is already greater than the minimum SAD encountered so far in the searching. In general, the PDS is regarded as a fast full search algorithm because it has identical prediction quality as that of the FSA.

c) Algorithms in the third category make use of mathematical inequalities to reduce the computational load; algorithms utilizing mathematical inequality based on sum norms, in which some more efficient criterions are established to identify and reject all non best candidates before calculating matching errors, this includes the successive elimination algorithm (SEA).

d) The fourth category is to use a combination of the above techniques to further improve the coding efficiency. In practice, the approaches in the two lines can be effectively combined to achieve the fastest motion estimation. For the algorithms in line with the first line of reducing the number of search points, computation reduction is achieved by sacrificing the prediction performance.

Also, the algorithms in subgroup 1) under the second line normally introduce matching errors. Some applications may suffer from inappropriate motion vectors estimated by such lossy fast search algorithms. Therefore, the only way to accomplish the fast motion estimation with optimality guarantee is to perform full search in combination with the methods in aforementioned subgroup b) and c) in reducing the computations.

Block matching is widely used for stereo vision, vision tracking, and video compression. Video coding standards such as MPEG-1, MPEG-2, MPEG-4, H.261, H.263 and H.264 use block based motion estimation algorithms due to their effectiveness and simplicity for hardware implementation. The main idea behind block matching estimation is the partitioning of the target (predicted) frame into square blocks of pixels and finding the best match for these blocks in a current (anchor) frame. To find the best match, a search inside a previously coded frame is performed and the matching criterion is utilized on the candidate matching blocks. The displacement between the block in the predictor frame and the best match in the anchor frame defines a motion vector. In the encoder, it is only necessary to send the motion vector and a residue block, defined as the difference between the current block and the predictor block.

The matching criterion is typically the mean of absolute errors (MAE) or the mean of square errors (MSE), given respectively by:

$$MAE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} | C_{ij} - R_{ij} |^2$$
$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} (C_{ij} - R_{ij})^2 \tag{4.16}$$

where N × N is the size of each block, C_{ij} and R_{ij} are respectively the pixel values in the current block and the reference block. Peak signal to noise ratio (PSNR) characterizes the motion compensated image created by predicted motion vectors and blocks from the reference frame.

$$PSNR = 10\log_{10}\left[\frac{(\text{peak to peak value of the original signal})^2}{\text{MSE}}\right]$$
$$= 10\log_{10}\left[\frac{255\times 255}{\text{MSE}}\right] \tag{4.17}$$

4.5 Fast Block Matching Motion Estimation Algorithms

4.5.1 Full Search [16] [19]

The best predicted representative of the current block is searched by computing the matching criterion between the current block and all blocks in the search area. If the algorithm applies MSE as matching criterion, then for checking each point with block size of 16×16, it requires 256 subtractions, 256 multiplications and 255 additions to calculate the MSE. The size of the search area is given by

$$\text{Search area} = (2p+1)\times(2p+1)$$

where p is the search parameter . The illustration of search area is shown in Figure When p = 7, the size of the search area will be 225 and hence 225 points must be checked in full search (FS) algorithm which is very intensive from computational standpoint.

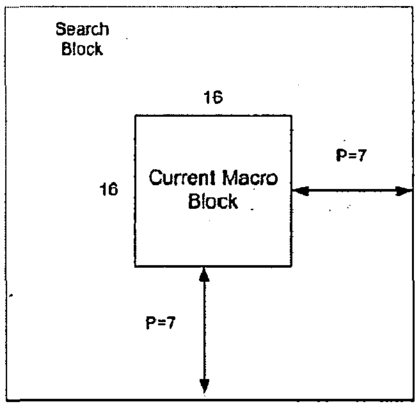


Fig. - 4.7. Search range [16]

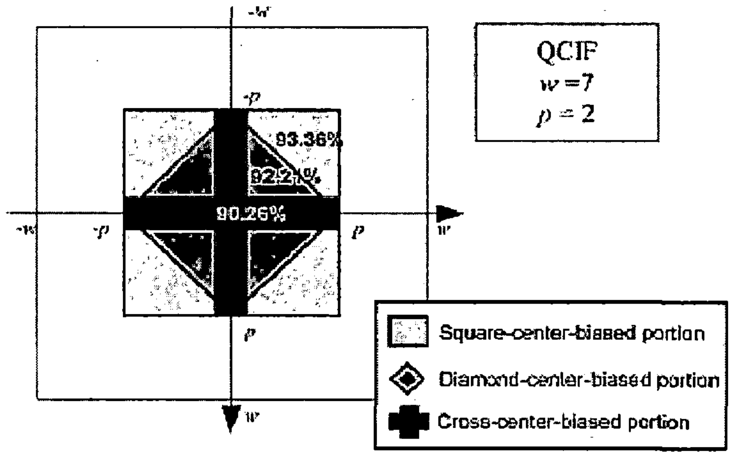

-w
QCIF
w=7
p=2
-p
93.36%
92.21%
90.26%
-w
-p
p
w
p
w
Square-center-biased portion
Diamond-center-biased portion
Cross-center-biased portion

Fig.- 4.8. Cross-center-biased property [4]

In Fig.4.8, over 93% of motion vectors are found within the central 5X 5 area when using QCIF sequences. This is a square-center-biased (SCB) distribution indicating most of real-world sequences move gently, smoothly and slowly, and can be regarded as quasistationary. Within this square region, about 90% and 92% motion vectors are found located in the cross-center-biased (CCB) and diamond-center-biased (DCB) portions. Amongst these three shaped-distributions, we can see CCB distribution is the most dominant. Moreover such cross-biased behavior maintains over 93% (and 83% for CIF / SIF) at boundaries of search area. Similar center-biased properties distributed as a cross, diamond and square shape are found in CIF / SIF and CCIR601 formats.

4.5.2 Cross Diamond Search [10]

It is based on centralized biased property. Basically, DS performs block-matching just like 4SS. It rotates the square-shaped search pattern by 45 to form a diamond-shaped one and with its size kept unchanged throughout the search before the new minimum block distortion measure (BDM) reaches the center of the diamond. The merits that DS yields faster searching speed can be regarded as: 1) the diamond-shaped pattern, which tries to behave as an ideal circle-shaped coverage for considering all possible directions of an investigating motion vector and 2) fewer checking points in the final converging step (only four instead of eight, as compared to square-shaped pattern BMA like New three step search and four step search).

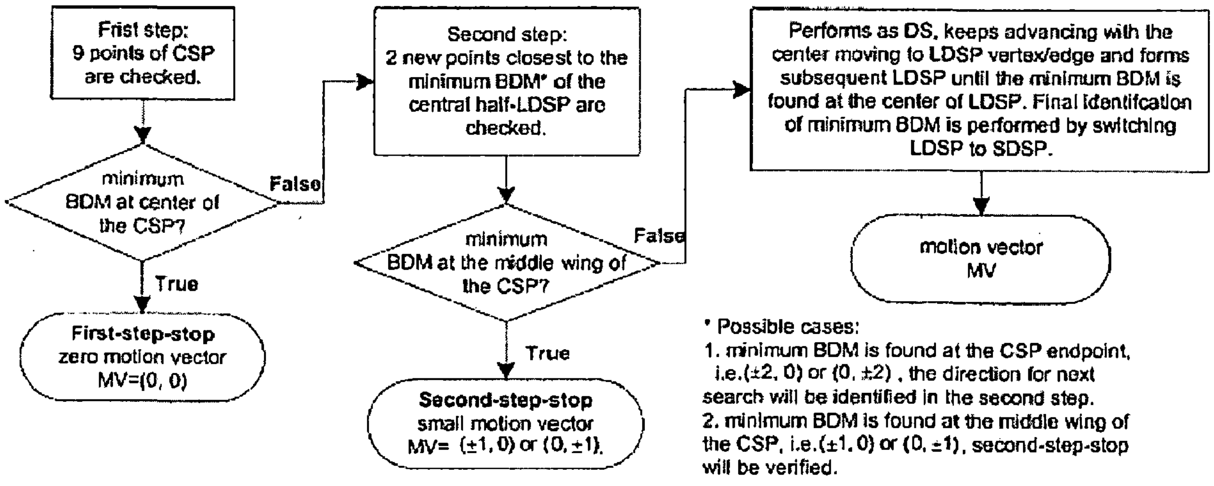

Frist step:
9 points of CSP are checked.
minimum BDM at center of the CSP?
False
True
First-step-stop
zero motion vector
MV=(0, 0)
Second step:
2 new points closest to the minimum BDM* of the central half-LDSP are checked.
minimum BDM at the middle wing of the CSP?
False
True
Second-step-stop
small motion vector
MV= (±1, 0) or (0, ±1).
Performs as DS, keeps advancing with the center moving to LDSP vertex/edge and forms subsequent LDSP until the minimum BDM is found at the center of LDSP. Final identifcation of minimum BDM is performed by switching LDSP to SDSP.
motion vector
MV
* Possible cases:
1. minimum BDM is found at the CSP endpoint, i.e.(±2, 0) or (0, ±2), the direction for next search will be identified in the second step.
2. minimum BDM is found at the middle wing of the CSP, i.e.(±1, 0) or (0, ±1), second-step-stop will be verified.

Fig.-4.9 Flowchart of the CDS algorithm. [10]

The reason behind the disadvantages above-mentioned for DS is that the diamond shape is not approximate enough to a circle, which is just 90 rotation of a square. Ideally, a circle-shaped search pattern with a uniform distribution of a minimum number of search points is desirable to achieve the fastest search speed uniformly. Practically, a more circle-approximated search pattern in the motion field is attainable in which a minimum number of search points are distributed uniformly. Each search point can be equally utilized with maximum efficiency, where the redundancy among search points should be removed maximally.

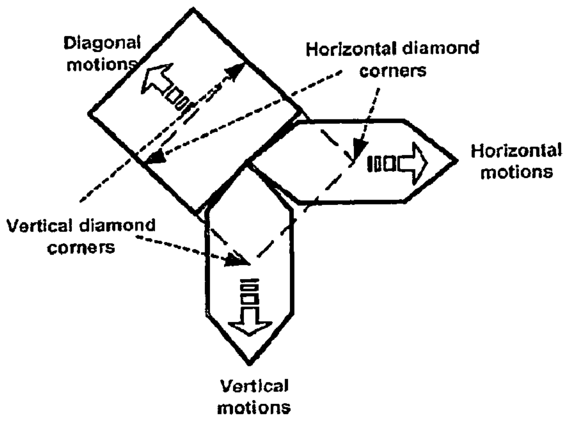


Fig-4.10. Search patterns switched for different directions. [4]

4.5.3 Hexagonal search [2]

It is based on cross centralized biased property. As shown in the Fig.4.10 its tries to cover motion in horizontal, vertical and diagonal direction.

Step 1) The large hexagon with seven checking points is centered at (0,0) the center of a predefined search window in the motion field. If the MBD point is found to be at the center of the hexagon, proceed to Step 3) (Ending); otherwise, proceed to Step 2) (Searching).

Step 2) With the MBD point in the previous search step as the center, a new large hexagon is formed. Three new candidate points are checked, and the MBD point is again identified. If the MBD point is still the center point of the newly formed hexagon, then go to Step 3) (Ending); otherwise, repeat this step continuously.

Step 3) Switch the search pattern from the large to the small size of the hexagon. The four points covered by the small hexagon are evaluated to compare with the current MBD point. The new MBD point is the final solution of the motion vector.

4.5.4 Enhanced HEXBS Algorithm [3]

On top of the original hexagonal algorithm HEXBS, an enhanced HEXBS can be developed by incorporating the above elaborated 6-side-based fast inner search scheme. Moreover, the predictive way of finding a good starting point using the neighboring motion vectors can also be employed in the enhanced HEXBS to reduce the number of search points in the coarse search before the inner search. Therefore, the reduction of number of search points for the enhanced HEXBS algorithm is expected to be from two aspects, i.e., one from the prediction for a good starting point using the predictive HEXBS, and the other from the proposed fast inner search.

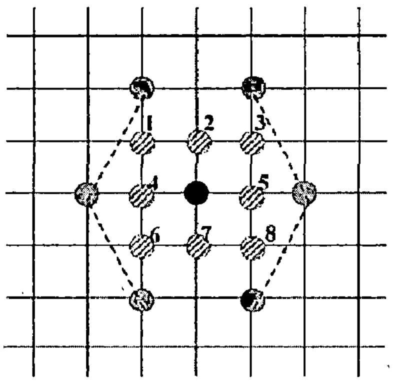


Fig-4.11 Inner points in the hexagonal search pattern. [3]

When the size of the fixed search pattern does not match the magnitude of the actual motion, over search or under search will be incurred which can cause certain search deficiency and inaccuracy. For example, in DS, LDSP will be too large for searching a small MV with the length less than 2 pixels away from the search center, thus causing unnecessary searches (i.e., over search). On the other hand, in the case of large and complex motion, the characteristic of center-biased MV distribution is very weak, and the uni-modal error surface assumption is no longer valid Even LDSP could be too small for searching large MV (i.e., under search) and leads to either a long search path (causing unnecessary intermediate searches) or being trapped into a local minimum matching error point (yielding large matching errors and degrading video quality).

A small search pattern made up by compactly spaced search points is more suitable than a large search pattern containing sparsely spaced search points in detecting small motions, because only a small number of positions around the search window center are necessary to be checked. However, when searching for a large MV, the small pattern tends to be trapped into local minimum along the search path and leads to wrong estimation. On the contrary, the large search pattern has the advantage of quickly detecting large motions, but it will incur unnecessary search for small MVs. In summary, the speed and accuracy of pattern-based search algorithms intimately depend on the size of the search pattern and the magnitude of the target MV. Therefore, it is highly desirable to use different search patterns according to the estimated motion behavior (in terms of the magnitude of motion) for the current block. This boils down to two issues required to be addressed:

1) How to pre-determine the motion behavior of the current block for performing efficient ME?
2) What are the most suitable size and shape of the search pattern?

4.5.5 Cross-Diamond-Hexagonal Search Algorithms (CDHS) [4]

The CDHS algorithms differ from DS, HEXBS, and CDS by performing a highly cross-center-biased search with SCSP in the first step. In addition, the search may involve up to two different patterns: diamond-shaped

LDSP and hexagonal pair LHSP. The common strategy amongst them is employing a halfway-stop technique.

The thick ones advance the search with bigger steps and result in faster search speed than the flat ones, but make sacrifice for quality. Conversely, the flat ones give better quality, as they require more steps and thus more points. From the orientation point of view, horizontal ones endeavor to trigger sequences with relatively higher horizontal motion content. In short, it is a priori problem that we cannot know the nature of a sequence in advance. Therefore, a diamond-shaped pattern, i.e., LDSP, plays an important role in the CDHSs. A pair of LHSPs will be consistently used throughout the search.

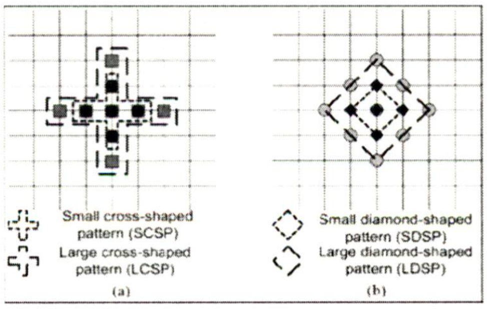

Small cross-shaped pattern (SCSP)
Large cross-shaped pattern (LCSP)
(a)
Small diamond-shaped pattern (SDSP)
Large diamond-shaped pattern (LDSP)
(b)

Fig-4.12. Search patterns used in CDHS algorithms.[4]

4.6 Hierarchical and Multi-resolution Fast Block Matching Algorithm [6] [8]

One family of fast block motion estimation algorithms relies on the idea of predicting an approximate large-scale MV in a coarse-resolution video and refining the predicted MV in a multi-resolution fashion to obtain the MV in the finer resolution. They are called the hierarchical or the multi-resolution methods .The hierarchical methods use the same image size but different block sizes at each level. The underlying assumption is that the MV obtained from a larger block size provides a good initial estimate for MV's associated with smaller blocks which are contained by the larger block. This assumption is often not true and the estimate can be very poor. Furthermore, a larger block size implies a higher computational cost in performing block matching. The multi-resolution methods use different image resolutions with a smaller image size at a coarser level (i.e., of a pyramid form). They can be further divided into two groups: constant block size and variable block size. Thus, a block at the coarser level represents a larger region than that at the finer level so that a smaller search area can be used at coarser levels. If the image size is reduced by half as the level becomes coarser, one block at a coarser level covers four corresponding blocks at the next finer level. Then, the MV of the coarser-level block is either directly used as the initial estimate for the four corresponding finer-level blocks or interpolated to obtain four MV's of the finer level. Different block sizes are employed at each level to maintain a one-to one correspondence between blocks in different levels. Then, the MV of each block is directly used as an initial estimate for the corresponding block at the finer level. Methods in this category work relatively well and provide fast computation. However, they only use the information from coarser levels for the MV refinement in finer levels without considering other useful information such as spatial and temporal correlations among MV's at the same level. Furthermore, the refinement process is performed by using a full search algorithm with a reduced search area which nevertheless requires a considerable amount of computation.

In the variable block size MRME scheme of based on the pyramidal data structure resulting from the wavelet analysis, a block size of $(p.2^{M-m})\times(p.2^{M-m})$ is used for the m^{th} level subbands, where M is the analysis level. Thus, the lower the resolution level, the smaller the block size. The constant p is the size of the block used at the lowest resolution. With this structure, the number of motion blocks for all of the subbands is the same and the blocks at the same position across all of the resolution levels corresponds to the same global position and the same object. This variable-block-size MRME technique has been used extensively as a baseline MRME scheme because of its simple architecture and good performance.

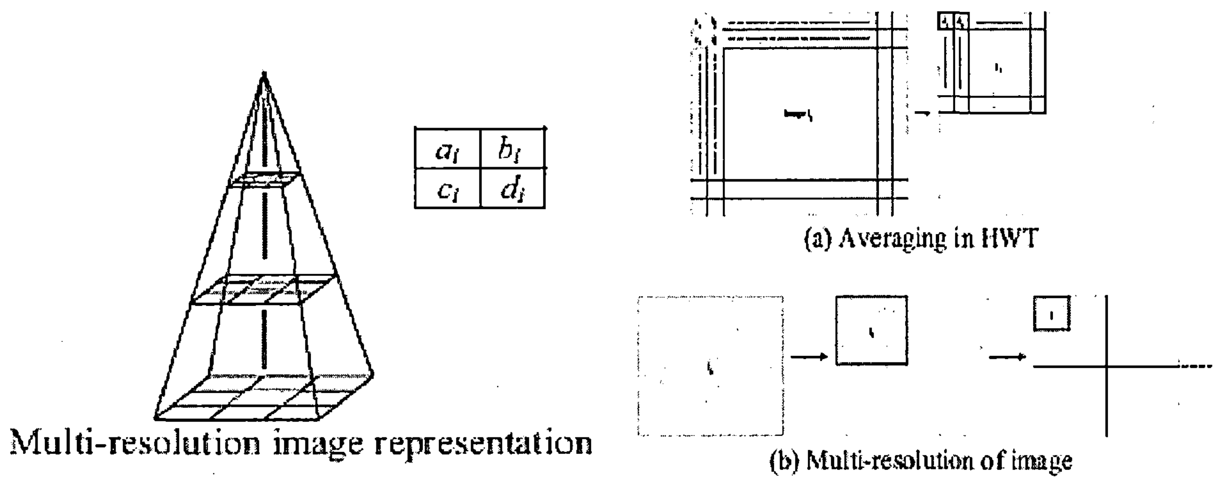
In order to obtain the prediction MV in a bandpass subband at a given resolution level, the corresponding bandpass MV at the lowest resolution level is multiplied by an appropriate power of two. A refinement factor is then added to this prediction after carrying out a motion search around this motion prediction. The MV prediction equation can be written as

$$V_{i,j}(x,y) = 2^{i-1}V_{1,j}(x,y) + \delta_{i,j}(x,y), for\, i = 2,3\, and\, j = 1,2,3 \qquad (4.18)$$

where $V_{i,j}$ is the MV at the resolution is level i with orientation j and $\delta_{i,j}$ is a refinement factor to be determined by a motion search process carried out around the predicted value $2^{i-1}V_{i,j}$ to estimate the MV $V_{i,j}$.

4.6.1 Haar Wavelet Transformation [15]

Using wavelet transformation, an image VOP can be represented at different level of resolutions, such as finest level (original VOP) to a coarse level (Fig-4.14). In such a representation, the structural property of the image remains same in lower level, except the details, which is reduced in some extent than its higher levels. This property is very useful in motion estimation techniques, particularly to reduce the computational complexity. Applying the multi-resolution concept, the motion vectors can be predicted at a lower resolution level and then this prediction can be refined at each step proceeding towards the higher resolution levels. The computational cost involved at a lower resolution level is less than its higher resolution level. Recently, several wavelet transformations are known such as discrete wavelet transformation (DWT), Haar wavelet transformation (HWT) etc. Among the wavelet transformation domain, HWT is the most simplest and can be easily implemented in hardware. In the present work, HWT is used to represent an image at different level of resolutions. Our approach of multi-resolution is based on the concept of averaging in HWT. Suppose there is an image VOP of size $N \times N$ pixels. Assume that entire frame is divided into a number of sub-blocks each of having size 2×2. A sub-block, say i is therefore a quadruple.



a_i
b_i
c_i
d_i
Multi-resolution image representation
(a) Averaging in HWT
(b) Multi-resolution of image

Fig-4.14 .Motion estimation using Haar wavelet [15]

We define the averaging for this quadruple as $A_i = \frac{a_i + b_i + c_i + d_i}{4}$. This averaging can be carried out for each quadruple in the VOP$\{e, ne, n, nw\}$ I_0 of size $N \times N$ and store the result in the first quadrant of another matrix I_1 of size $\frac{N \times N}{2 \times 2}$. This corresponds to the image at the next lower resolution level. The above transformation can be repeated for I_1 to get the image I_2 at the next resolution level and so on. The size of an image at a lower resolution level is reduced by half than its just preceding level.

F_t and F_{t-1} denote the current and reference imageVOP at time t and t-1.

F_t^1 denotes the image VOP at an instant t with the resolution level 1.
Compute $SAD(b_c^{i,j}, b_r^{u,y}, N)$ is a usual routine to calculate the SAD value between two blocks $b_c^{i,j}$ and $b_r^{u,v}$, each of having size N.

Input: Current VOP F_t and reference VOP F_{t-1}.

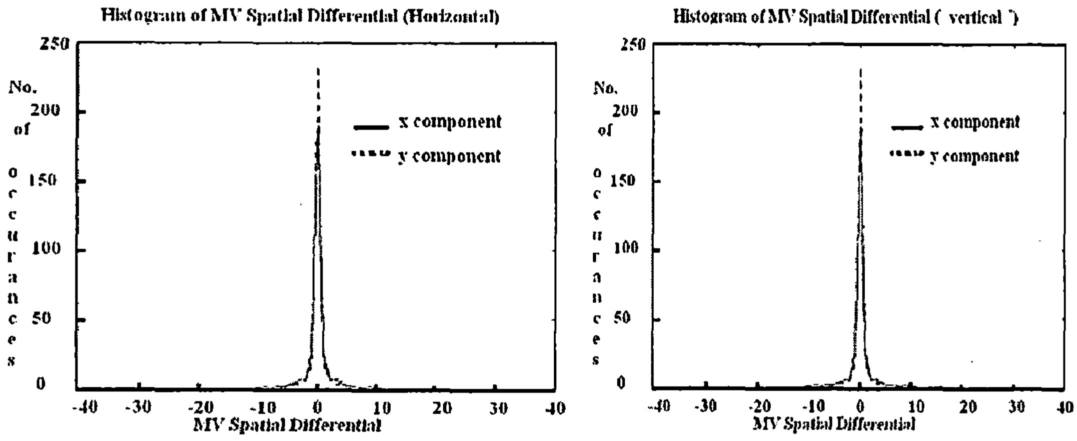
Output: Motion vector of each block $b^{i,j} \varepsilon F_t$.

```
1. Perform Haar wavelet transformation on the VOP $F_t$ and $F_{t-1}$ at resolution level
   2 and 3; original VOP are at the resolution level 1. Let $F_t^2$ and $F_t^3$ are the
   transformed forms of the VOP $F_t$ at resolution levels 2 and 3, respectively.
   Similarly $F_{t-1}^2$ and $F_{t-1}^3$ are the transformed forms of the VOP $F_{t-1}$ at resolution
   levels 2 and 3, respectively.

2. For each block $b^{i,j} \varepsilon F_t$ do
      2.1) For all candidate block $b_r^{x,y} \varepsilon F_{t-1}^3$ of the block $b_c^{i/4,j/4} \varepsilon F_t^3$ do
                      $SAD(b_c^{i/4,j/4}, b_r^{x,y}, 4)$
           Select the candidate block with the minimum SAD value.
           Let the winner block be $b_r^{u,v} \varepsilon F_{t-1}^3$

      2.2) For all neighboring candidate block $b_r^{x,y} \varepsilon F_{t-1}^2$ of the block $b_r^{2u,2v} \varepsilon F_{t-1}^2$
```

```
        do
        Compute $SAD(b_c^{i/2,j/2}, b_r^{x,y}, 8)$. Select the candidate block with the
        minimum SAD value. Let the winner block be $b_r^{u,v} \varepsilon F_{t-1}^2$.
    2.3) For all neighboring candidate block $b_r^{x,y} \in F_{t-1}$ of the block $b_r^{2u,2v} \varepsilon F_{t-1}$
        do
        Compute $SAD(b_c^{i,j}, b_r^{x,y}, 16)$ Select the candidate block with the
        minimum SAD value. Let the winner block be $b_r^{u,v} \varepsilon F_{t-1}$.
3. Calculate the motion vector between $b^{i,j}$ and $b_r^{x,y}$.
```

4.7 Fast Block Matching with Spatial Correlation [11] [21] [25] [27]

The motion vectors are highly correlated in both horizontal and vertical direction (Fig.-4.15). Adjacent MBs belong to the same moving object have the similar motions. Therefore, the current block's motion behavior can be reasonably predicted by referring to its neighboring blocks MVs in the spatial and or temporal domains.



(a) (b)

• Fig-4.15. Example of Histograms of the averaged number of MV spatial differentials of x and y-components along the (a) horizontal and (b) vertical direction [11]

The objective is to find a good starting point for the remaining local search so as to avoid unnecessary intermediate search and reduce the risk of being trapped into local minimum in the case of long search path. The new starting point identified is hopefully as close to the global minimum as possible. A small, compact, and fixed-size search pattern would be able to complete the remaining local search quickly. Note also that this small search pattern will be repeatedly and unrestrictedly used in the refined local search until the final MV is found.

4.7.1 Prediction of the Target MV [25]

In order to obtain an accurate MV prediction of the current block, two factors need to be considered:

1) Choice of the region of support (ROS) that consists of the neighboring blocks whose MVs will be used to calculate the predicted MV.

2) Algorithm used for computing the predicted MV.

The motion-vector predictor can be obtained from calculating the medium value of motion vectors of the three neighboring macro blocks.

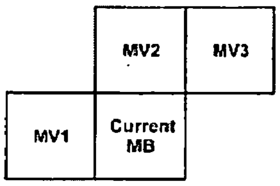


Fig-4.16. Motion-vector prediction: the predictor for the current macro block is the medium of MV1, MV2, and MV3.

The basic principle for motion-vector prediction is that the motion field of nature video is gentle, smooth, and varies slowly. Therefore, the correlation between motion vectors of neighboring macro blocks is very strong.

The Predictive Line Search (PLS) algorithm, with simplicity and regular search pattern in mind is summarized as follows:

Step 1) Search three consecutive lines of candidates centered at the motion-vector predictor. If the motion-vector predictor locates in line p, then all points in line p-1, line p, and line p+1 are tested. If the best-matching point calculated is located in line p+1, go to Step 2), if the best-matching point is in line p-1, go to Step 3), otherwise, go to Step 4).

Step 2) Let p=p+1, then test all points in line p+1. If the best matching point is in line p, go to Step 4), otherwise repeat the current step.

Step 3) Let p=p-1, then test all points in line p-1. If the best-matching point is in line p, go to Step 4), otherwise repeat the current step.

Step 4) Report the best-matching point as the position of the motion vector.

In short, this method starts from searching three lines around the motion-vector predictor, and then searches additional lines in the direction of descending distortion, and stops when the best matching point is not on the boundary of searched lines.

4.8 Fast Block Matching with Spatial/Temporal Correlations [13] [25] [11]

Another direction for fast MV estimation approach is to exploit information from adjacent blocks by using spatial and temporal correlations of MV's. The main idea is to select a set of initial MV candidates from spatially and/or temporally neighboring blocks and choose the best one (according to a certain rule) as the initial estimate for further refinement.

We consider nine blocks from the spatio-temporal neighborhood of the current image block. These nine blocks can be termed as the spatio-temporal context $C^t_{n(i,j)}$ for a block $b^t_{n(i,j)}$ at location (i, j) in the current VOP (n). This context $(C^t_{n(i,j)})$ consists of five blocks from the previous VOP (n-1) and four blocks from the current VOP (n).

$$C^t_{n(i,j)} = b^{t-1}_{n(i,j)}, b^{t-1}_{n(i,j-1)}, b^{t-1}_{n(i-1,j-1)}, b^{t-1}_{n(i-1,j)}, b^{t-1}_{n(i-1,j+1)}, b^{t}_{n(i,j+1)}, b^{t-1}_{n(i+1,j+1)}, b^{t}_{n(i+1,j)}, b^{t-1}_{n(i+1,j-1)} \tag{4.19}$$

The selection of specific blocks for spatio-temporal context is based on the following factors: proximity, scanning order, and completeness. The blocks are closest possible ones to the current block $b^t_{n(i,j)}$ at location (i, j). The four blocks in the current VOP would be scanned before the current block, following the raster scanning order. The context consists of at least one block in each direction $\{s, n, e, ne, se, sw, nw\}$ and one block at location (i, j) in the previous

VOP. Thus, the spatio-temporal context completely describes the immediate neighborhood of the block under consideration and provides an intuitively optimal solution. While additional contextual information may result in better motion estimates, it also increases the computational complexity

This scheme can be embedded in any fast block-based motion estimation algorithm as a preprocessing step.

```
For each block
  Begin
  • If the block belongs to the first VOP: Define the spatial context consisting of
    $\{w, sw, s, se\}$ .
  • If the block belongs to the subsequent VOPs: Define the spatio- temporal context
    consisting of $\{e, ne, n, nw\}$ from the previous VOP and $\{w, sw, s, se\}$ from the
    current VOP.
  • Consider the image blocks in the spatio-temporal context as illustrated in
    Fig.4.17.
  • Compute the SAD value between the current block and the blocks in the
    spatio-temporal context.
  • Choose the block that is closest to the current image block using the distance
    criterion.
  • Make the first pixel in block as the starting point for the search algorithm.
  • Use the standard search algorithm to find the MV for the current block starting
    from the location.
End
End for
```

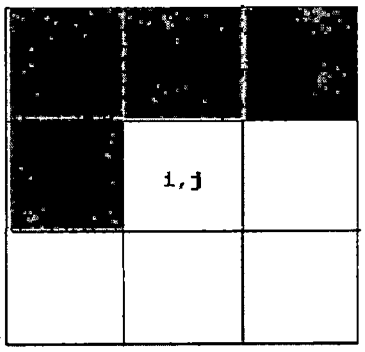


Fig- 4.17. Spatio-temporal neighborhood at location (i; j) in the current VOP.

Motion estimation algorithms based on multi-resolution and spatio-temporal correlations are based on the hypothesis that the motion field varies slowly both spatially and temporally, and hence it is highly probable that the macro blocks (MBs) closer to the current macro block, in time and space, may show the same motion. With this assumption, instead of calculating all the possible motion MVs as the full search does, the previously calculated MVs for the MBs that are spatially and temporally contiguous, can be used as the candidate predictors for the current block.

The MV information from the coarse level is used; we can only have the MV of an even number of length as the initial candidate. On the other hand, if only the spatio-temporal information is used, there are some blocks that still require a full search. The MV obtained from full search in the coarsest level provides a better set of candidates so that the full search mode is not needed at finer levels. So the algorithms motion estimation by multi-resolution analysis using Haar wavelet can be combined with spatio-temporal correlation to produce better results.

CHAPTER 5

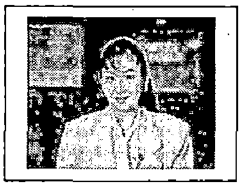
SIMULATION

This chapter presents the details of the methods used to evaluate the performance of MPEG-4 Video encoder. The chapter includes the step by step approach to simulation.

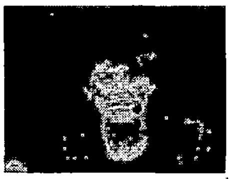
The input test sequences are standard MPEG-4 test sequences. The input test sequences are in. Y U V format.

Input Test Sequences :

1. Akiyo frame



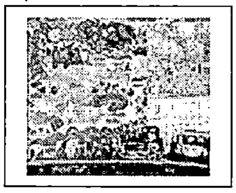
2. Miss-America frame



3. Mother-daughter frame

4. Salesman frame

5. Mobile frame



I have used the reference source code of Sasken for my dissertation. The reference code had three step method for motion estimation.

Specifications of the Encoder

Profile	:	Simple
Level	:	Level 1
Video Format	:	4:2:0

Image resolution	:	176*144
GOP	:	9
Quantization scale	:	5
Motion estimation	:	Only for Luminance component.
No. of frames	:	150
Frame rate	:	15 fps

The first two sequences (Akiyo and Miss-America) have very little motion, test sequence three and four (Mother-daughter and Salesman) have medium motion, and the last test sequence (Mobile) has high motion.

I have implemented all the motion estimation algorithms diamond search, hexagonal search, predictive line search, motion estimation using multi resolution analysis, predictive line search and motion estimation using spatio-temporal contextual information as mentioned in the IEEE transactions. I have combined motion estimation using multi-resolution analysis and motion estimation using spatio-temporal contextual information, which is our proposed algorithm (algorithm obtained by combining reference [13] and [15]).After estimation I have implemented the motion compensation block, since the motion compensated VOP is subtracted from the current VOP for (B , P VOP) and is inter-coded, so we need to have the reconstructed VOP in the encoder itself. For implementation for motion compensation block I have used the procedure mentioned in MPEG-4 standard. I have also computed the bounding box after obtaining the VOP after segmentation as mentioned below.

VOP reconstruction:-

The luminance and chrominance values of a VOP from the decoded texture and motion information are reconstructed as follows:

1. In case of intra macro blocks, the luminance and chrominance values f[y][x] from the decoded texture data form the luminance and chrominance values of the VOP: d[y][x] = f[y][x].

2. In case of inter macro blocks, first the prediction values p[y][x] are calculated using the decoded motion vector information and the texture information of the respective reference VOPs. Then, the decoded texture data f[y][x] is added to the

prediction values, resulting in the final luminance and chrominance values of the VOP:

$$d[y][x] = p[y][x] + f[y][x]$$

3. Finally, the calculated luminance and chrominance values of the reconstructed VOP are saturated so that

$$\boxed{d[y][x] = \begin{cases} 2^{bits\_per\_pixel} - 1; & d[y][x] > 2^{bits\_per\_pixel} - 1 \\ d[y][x] & ; \quad 0 \le d[y][x] \le 2^{bits\_per\_pixel} - 1 \\ 0 & ; \quad d[y][x] < 0 \end{cases}}$$

First I have developed the C code on Microsoft VC++, which has many features similar to code composer studio and it is easier and faster to develop the code on VC++ than on code composer studio. After compiling the code on VC++, I have run the same code on code composer simulator version. Then I integrated the motion estimation block, motion compensation block and bounding box computation block along with the reference MPEG-4 software which is available at Sasken.

Then I have ported the code on TMS320C64XX board and run the code. The code composer studio itself provides number of tools for optimization which I have mentioned in Chapter-3. I have used -02 level, Speed most critical, software pipelining which are available in CCS for optimization. The CCS generated the assembly code after we run the code. I have written assembly code for all motion estimation (the codes are being withheld by Sasken) and tried to optimize the code in terms of the number of cycles it consumes.

Results:-

	FS	DS	HS	MRHS	ST	PL	ST+MR
Akiyo	1000,000,000	970,000,000	978,000,000	980,000,000	940,000,000	950,000,000	983,000,000
Miss-America	1000,010,000	970,000,500	978,000,200	980,000,150	950,000,000	950,001,000	984,000,000
Mother-daughter	1010,000,000	1000,000,000	990,000,000	1100,000,000	960,000,000	970,000,000	1001,000,000
Salesman	1200,000,000	1080,000,000	1100,000,000	1250,000,000	980,000,000	990,000,000	1253,000,000
Mobile	1500,000,000	1100,000,000	1250,000,000	1400,000,000	1000,000,000	1010,000,000	1420,000,000

Table 5.1 Run – Time Profile: In terms of cycles consumed

FS - Full search. DS – Diamond search. HS – Hexagonal search. PL – Predictive line search ST – Spatio-Temporal MRHS – Multi resolution search ST+MR – Multi resolution + Spatio-Temporal search. (Proposed algorithm).

The above table gives information of the number of cycles consumed for different motion estimation algorithms. This information is observed from CCS, in which we have options to see the number of cycles a particular block consumes.

After we obtain the segmented image I have done the bounding box computation for the VOP. This is required since our VOP should be having blocks in multiples of 16 since it is helpful in motion estimation. For performing the motion estimation we need to know the level, profile, image size, format and progressive or interlaced information. The level and profile information are available in the code. For determining the image size and format I have used YUV sequence analyzer and atomic browser. Then I have separated the luminance from the chrominance component depending on the image format since for motion estimation we need only the luminance component. The image is progressive or interlaced information is obtained by observing the output of the encoder which will be in .bit format. I have used Nikon MPEG-4 software tool for observing the output streams.MPEG-4 video encoder does not support interlaced formats. Only the real video encoder (from helix community) supports both interlaced and progressive sequences.

After motion estimation we need to motion compensate the VOP. For doing motion compensation we need to have the motion vectors, shape and texture information. The shape and texture information I obtained from already existing code, the motion vectors information I obtained from the algorithms I developed. We take the macro block from the reference VOP and dump the macro block coefficents as pointed out by the motion vectors in the reconstructed (B and P VOPs).We also need to take care of unrestricted motion vectors, this is very important in motion compensation since the macro block which the unrestricted motion vectors point to are not available in the current VOP. For motion compensation I have used the standard procedure mentioned in the standard.

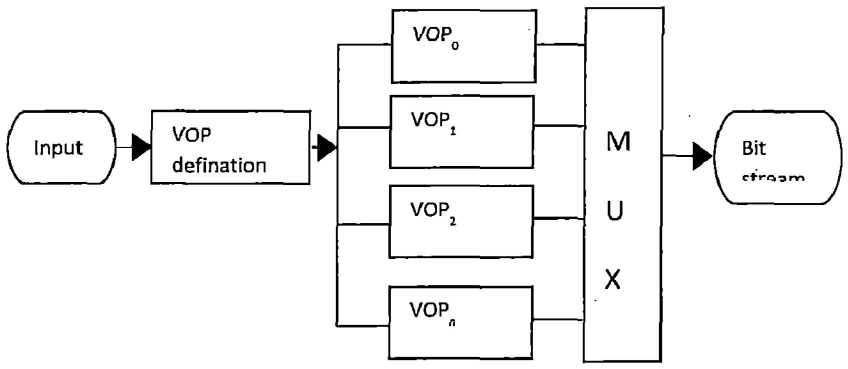


Fig:- 5.1 Encoder 1

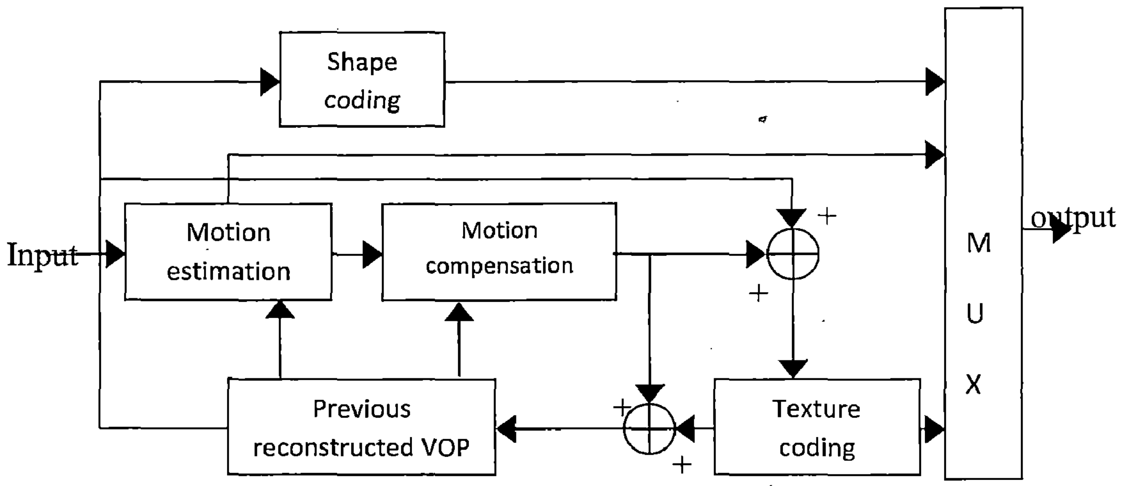


Fig:- 5.2 Encoder 2

The N x N two dimensional DCT is defined as

$$F(u,v) = \frac{2}{N} C(u)C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\frac{(2x+1)u\pi}{2N} \cos\frac{(2y+1)v\pi}{2N}$$

with u, v, x, y = 0, 1, 2, N-1

where x, y are spatial coordinates in the sample domain

u, v are coordinates in the transform domain

$$C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} \text{ for } u, v = 0 \\ 1 \quad \text{otherwise} \end{cases}$$

The inverse DCT (IDCT) is defined as:

$$f(x,y)=\frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1}C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{2N}\cos\frac{(2y+1)v\pi}{2N}$$

If each pixel is represented by n bits per pixel, the input to the forward transform and output from the inverse transform is represented with (n+1) bits. The coefficients are represented in (n+4) bits. The dynamic range of the DCT coefficients is $[-2^{n+3} : +2^{n+3} - 1]$

Quantization:

The default matrix for intra blocks (both luminance and chrominance) is:

8	17	18	19	21	23	25	27
17	18	19	21	23	25	27	28
20	21	22	23	24	26	28	30
21	22	23	24	26	28	30	32
22	23	24	26	28	30	32	35
23	24	26	28	30	32	35	38
25	26	28	30	32	35	38	41
27	28	30	32	35	38	41	45

The default matrix for non-intra blocks (both luminance and chrominance) is:

16	17	18	19	20	21	22	23
17	18	19	20	21	22	23	24
18	19	20	21	22	23	24	25
19	20	21	22	23	24	26	27
20	21	22	23	25	26	27	28
21	22	23	24	26	27	28	30
22	23	24	26	27	28	30	31
23	24	25	27	28	30	31	33

First Inverse Quantization method:

For intra DC coefficient:

In intra blocks F[0][0] shall be obtained by multiplying QF[0][0] by a constant multiplier.

The reconstructed DC values are computed as follows.

F[0][0] = dc_scaler* QF[0][0]

When short_video_header is 1, dc_scaler is 8; otherwise dc_scaler is defined in Table 5.5

Component:Type	dc_scaler for quantiser_scale range			
	1 through 4	5 through 8	9 through 24	>= 25
Luminance: Type1	8	2x quantiser_scale	quantiser_scale +8	2 x quantiser_scale -16
Chrominance: Type2	8	(quantiser_scale +13)/2		quantiser_scale -6

Table: 5.2 Quantization scale

For Other Coefficients:

$$F'[v][u] = \begin{cases} 0, & if\ QF[v][u] = 0 \\ ((2 \times QF[v][u] + k) \times W[w][v][u] \times quantiser\_scale)/16, & if\ QF[v][u] \neq 0 \end{cases}$$

where:

$$k = \begin{cases} 0 & \text{intra blocks} \\ Sign(QF[v][u] & \text{non-intra blocks} \end{cases}$$

For calculation of average peak signal to noise ratio we need to have the reconstructed sequence (output of video encoder) and the input sequence . From this we calculate the average PSNR.

Peak-to-Peak Signal-to-Noise ratio (PSNR) is defined as

$$\text{PSNR} = 10\log_{10}\left[\frac{255^2}{(1/N)\sum_i\sum_j\left(Y_{ref}(i,j) - Y_{pre}(i,j)\right)^2}\right]$$

where $Y_{ref}(i,j)$ and $Y_{pre}(i,j)$ are the pixel values of the reference and processed images, respectively, and N is the total number of pixels in the image. In this equation, the peak signal with an eight-bit resolution is 255, and the noise is the square of the pixel-to-pixel difference (error) between the reference image and the image under study. Although it has been claimed that in some cases the PSNR's accuracy is doubtful, its relative simplicity makes it a very popular choice.

Average PSNR							
	FS	DS	HS	MRHS	ST	PL	ST+MR
Akiyo	35.2	33.8	34.4	30.8	34.3	34	34.5
Miss-America	33.1	31.8	32.62	30.6	32.61	32.6	32.5
Mother-daughter	33.3	32.4	32	30.3	32.4	32.1	32.9
Salesman	29.4	25.8	26.5	25	28	27.9	28.7
Mobile	28.3	26.2	25.8	25.4	27.3	27.3	27.9

Table: - 5.3 Average PSNR

From the above results we make the following observations:-

FS: It produces best results in terms of the PSNR but it is computationally very intensive. As the motion in the video sequence increases the average PSNR decreases and the no of cycles consumed increases.

DS: This algorithm is recommended in MPEG-4 reference software. As the motion in sequence increases the difference in average PSNR between the proposed algorithm and DS increases.

HS: It produces almost similar performance as diamond search. For foreman or coast guard sequence we can probably find the improved performance of HS compared to DS.

MRHS: Its performance in terms of average PSNR and cycles consumed are bad compared to DS.

ST and PL: Both these algorithms have similar results for the above test sequence which is expected since both algorithms reply on spatial / temporal or both to get information for predicting the motion vector. The PL algorithm consumes little more cycles than the ST algorithm since it has more search points.

ST + MR: This algorithm is our proposed algorithm. It consumes slightly higher cycles than other algorithms. But this algorithm is simple to implement and computationally less intensive than the algorithm [11] which was proposed previous which combined multi resolution with spatio-temporal information for better motion vector prediction. Its performance in terms of average PSNR is better compared to DS.

```
temporal_reference: 0
split_screen_indicator: 0  document_camera_indicator: 0
source_format: 2
vop_quant: 5
```

```
Start Address: 00000000
00000000: 00 00 80 02 08 05 1E 5E - A4 B8 0C B8 C0 18 20 5A   .......^...... Z
00000010: CB 94 8B 87 E0 7F AE 4B - 3F 06 3E 1F AA 12 55 78   .......K?.>...Ux
00000020: 7E 5D F5 2A 95 CD 94 9F - FE FA FA E5 E5 73 CC 01   ~].*.........s..
00000030: 82 05 8C B8 C1 49 71 3A - 77 D5 A8 9A F5 60 60 7E   .....Iq:w....``~
00000040: 21 2C 80 92 0C 07 70 20 - 04 0F 81 E1 2C 4B 91 5A   !,....p ....,K.Z
00000050: C7 5F 3C 5C 5F F9 B2 1F - 79 7A 95 44 E5 FA A9 77   ._<\_...yz.D...w
00000060: 2A 83 03 EF 09 54 7C 24 - CF DC 3B 2D E0 60 39 C1   *....T|$..;-.`9.
00000070: 80 10 57 F9 E8 25 CD 57 - 72 72 B7 48 1E A8 90 5F   ..W..%.Wrr.H..._
00000080: 9B 43 2A 96 E7 AB 4F 29 - 38 08 17 63 CA 9F 60 66   .C*...O)8..c..`f
00000090: 54 54 56 56 FB 02 9C A8 - BA A9 30 54 52 E7 E7 FF   TTVV......0TR...
000000A0: BC B8 1C 2F E2 D0 70 BF - 8B CB C4 B5 06 0E 68 38   .../..p.......h8
000000B0: 1E D2 EA C1 80 E9 04 05 - 5F BE AA FB 2E 5E 3D E8   ........_....^=.
000000C0: 10 BE 4C F2 D3 A6 94 18 - 1F 9D E1 A3 B0 60 7D 43   ..L..........`}C
000000D0: 33 C0 60 7D 43 39 C0 60 - 79 01 80 0F 06 17 F2 81   3.`}C9.`y.......
000000E0: 07 BC 83 02 0E AC 99 E5 - FE 8A C4 53 65 98 19 9D   ...........Se...
000000F0: 83 03 EA 19 9D 83 03 EC - F9 D0 60 78 C1 80 0F 05   ..........`x....
00000100: 3F C0 83 DE 41 81 07 56 - 4C F2 CC 0C CB 70 33 3A   ?...A..VL....p3:
00000110: 06 07 DD E6 DE 08 6F 9D - 06 07 90 10 18 FF C1 C9   ......o.........
00000120: 02 7A 05 07 C9 9E 56 C9 - D2 88 DC 4B 06 25 92 A9   .z....V....K.%..
00000130: 3A 53 1B 89 60 C6 05 52 - B8 5E AB C5 FE 6B F3 61   :S..`..R.^...k.a
00000140: 0F FD 6E B8 00 02 10 AE - 97 80 38 1C 30 35 05 F1   ..n.......8.05..
00000150: F7 CD AB 1F 8D 06 00 3C - 4A 06 04 20 03 87 C3 ED   .......<J.. ....
00000160: D1 E7 B1 4B 5B D3 EF 63 - FA A4 BF F5 77 BD 74 03   ...K[..c....w.t.
00000170: FE 71 DB E3 FA 7D 98 10 - 4D 3F 8C F4 41 80 E8 E8   .q...}..M?..A...
00000180: 30 1D A1 05 2F A0 2A 9C - C4 01 80 C1 82 C2 17 BA   0.../.*.........
00000190: 2A 1F 9F 63 12 81 83 FC - 24 7C 73 40 42 07 0F F7   *..c....$|s@B...
000001A0: 84 21 03 07 EC 24 03 0F - F2 5C 0C 5F F3 97 44 B0   .!...$...\._..D.
000001B0: 60 C1 3E 05 1F 53 A0 0E - 68 33 2A 3B 00 C1 18 33   `.>..S..h3*;...3
000001C0: 2B 7E FF D9 59 81 78 FC - B8 7D E5 25 F7 3D B9 19   +~..Y.x..}.%.=..
000001D0: B6 AD E9 3B 6A 67 17 5A - A3 7F 79 21 B3 22 E2 EA   ...;jg.Z..y!."..
000001E0: AC 18 2F A2 F9 DB C2 E5 - 56 F3 DE 2D ED 65 EF D0   ../.....V..-.e..
000001F0: 2B 37 F1 78 43 12 70 14 - 05 DE 2E F5 11 44 9D 53   +7.xC.p......D.S
00000200: 91 90 3D 1A D6 35 E7 0A - E0 30 1D C5 DD F0 30 21   ..=..5...0....0!
00000210: 22 5C 03 1F F7 CB AD 49 - FF 5B 83 42 69 6B 2A B2   "\.....I.[.Bik*.
00000220: 2E 7C 7C 7B 36 4F D9 2B - 12 99 7E B0 9E 05 E9 34   .||{6O.+..~....4
00000230: 20 8F 7E 3C 2F 55 F1 20 - 4B 64 B9 5E 7E 25 F2 BF    .~</U. Kd.^~%..
00000240: 0E BF 10 3D 3D 4C DB DE - B8 A8 20 80 71 70 93 EF   ...==L.... .qp..
00000250: C5 1D FB 33 96 58 D5 D6 - CE 2F 7C 4A 2F 08 5F 1F   ...3.X.../|J/._.
00000260: CF B7 D5 1F AC C6 1F 78 - 0C 07 48 30 02 A0 C0 84   .......x..H0....
00000270: 03 05 F0 3E D0 42 56 AF - 15 CB DA 5D FF DE 5D 03   ...>.BV....]..].
00000280: 14 64 DE 24 17 A9 AA 2B - 6C 12 BB 4D 4F 21 79 A2   .d.$...+l..MO!y.
00000290: 48 FD AD C5 07 E0 95 FC - 21 FA 17 55 3C 2F BC 48   H.......!..U</.H
000002A0: 7F 38 18 11 61 FD E8 21 - AA D6 22 EC 9C 60 BE 00   .8..a..!.."..`..
000002B0: 85 29 C9 8F 7E 08 7D F8 - F7 F7 3D 64 F4 86 5A C1   .)..~.}...=d..Z.
000002C0: 00 49 56 5E 01 E5 FB AA - 27 B9 1F B0 AC 18 01 02   .IV^....'.......
000002D0: F2 E5 43 F5 7B 14 0F BD - 3F F4 FB 79 21 3B 23 D2   ..C.{...?..y!;#.
000002E0: CF A9 03 03 E8 5D 3F 8A - 4E AA 8F FC 07 FD 61 3B   .....]?.N.....a;
000002F0: 51 70 30 02 58 5E A5 49 - 9D E1 F0 21 74 7B F5 0B   Qp0.X^.I...!t{..
00000300: 48 E6 A5 6A 4C 3C C8 4A - FA 95 5A 60 A6 36 8E 0C   H..jL<.J..Z`.6..
00000310: 4D 42 02 A0 3C 3E 8E 29 - 8D C4 B0 63 0E 10 38 10   MB..<>.)...c..8.
00000320: 84 A5 81 0B C3 07 FA DD - 23 00 00 88 2A 46 08 71   ........#...*F.q
00000330: 6C FB 0B 04 0B 49 1C FE - D1 8C 4B 54 07 FC 79 C9   l....I....KT..y.
00000340: 50 30 01 AA 26 0F 77 BA - 4C C0 BF 2E 10 41 80 15   P0..&.w.L....A..
00000350: 00 C0 82 5F 4B CB FD F5 - 3F 11 F3 EA 35 87 3B AA   ..._K...?...5.;.
```

```
temporal_reference: 0
split_screen_indicator: 0  document_camera_indicator: 0
source_format: 2
vop_quant: 5
```

```
Start Address: 00000000
00000000: 00 00 80 02 08 05 18 8B - 9E 60 5A 5E F2 43 0A F6   .........`Z^.C..
00000010: 66 46 46 06 2F 1B 57 7C - 41 D6 66 86 86 46 6E FA   fFF./.W|A.f..Fn.
00000020: 4B 14 75 99 A9 A9 A1 AB - B6 D5 DF 16 75 99 A9 B1   K.u.........u...
00000030: A9 AB AE D2 78 B3 A8 A6 - A6 C6 A6 C7 1D 76 AE F8   ....x........v..
00000040: C6 43 63 63 89 FF 7F BE - A7 CD 5D 36 AE F8 C3 AC   .Ccc......]6....
00000050: CD CD CD 4D 5D B6 AE F8 - C3 AC CD CD CD 8D 5D 76   ...M].........]v
00000060: AE F8 B3 AC CD CD CD 8D - 5D 88 62 8E A3 36 36 35   ........].b..665
00000070: 35 76 EA EF 89 00 00 84 - 29 52 D7 97 96 BC BD E6   5v......)R......
00000080: 15 F3 2F 31 30 30 79 86 - 99 91 99 89 93 C4 44 CC   ../100y.......D.
00000090: CC CC CC DE 22 4E 65 87 - CC 99 E7 92 DA 17 97 C3   ...."Ne.........
000000A0: 45 A4 4F 10 7E 0C 9B 1A - 5F 36 DF D3 38 EF 15 AB   E.O.~..._6..8...
000000B0: D9 01 C2 03 2F 93 28 C4 - AD E0 02 60 81 C1 03 79   ..../.(....`...y
000000C0: BC 55 83 A2 63 A5 75 46 - 2D FB 8E 2C AE 3B 00 D9   .U..c.uF-..,.;..
000000D0: A2 57 FB 8B 26 7B B6 EB - E1 3F E3 C8 86 A6 88 01   .W..&{...?......
000000E0: 01 5A 95 59 6E DF 0E 36 - 3C D1 D5 F1 53 35 35 34   .Z.Yn..6<...S554
000000F0: 34 76 88 99 A1 A1 A1 A3 - B4 44 CD 0D 0D 0D 1D E1   4v.......D......
00000100: C0 00 22 0A 44 5A F2 F2 - D2 F7 98 47 CC C0 C0 BC   ..".DZ.....G....
00000110: C1 E2 16 43 23 33 17 99 - 3C 43 97 33 32 90 F9 91   ...C#3..<C.32...
00000120: 68 64 EF 0D 21 DE 58 60 - BD C5 8F 2F A3 07 97 F7   hd..!.X`.../....
00000130: 25 C5 87 0A 84 B8 3F B5 - 45 8A 1A 6C D9 98 ED 75   %.....?.E..l...u
00000140: C0 21 E0 F4 08 04 BC B9 - E7 25 E0 77 D7 C9 F2 C5   .!.......%.w....
00000150: 9E 66 23 C2 54 70 43 F8 - 28 60 18 B6 97 17 A6 57   .f#.TpC.(`.....W
00000160: 00 D7 D1 9A 76 90 65 07 - FA 33 54 88 25 78 0F 28   ....v.e..3T.%x.(
00000170: E3 19 53 59 CC 79 A2 97 - C1 0C 7C 10 6A B9 B8 AE   ..SY.y....|.j...
00000180: 8E E2 E3 75 91 93 9C AA - 57 3F 38 42 E7 F8 F8 A9   ...u....W?8B....
00000190: 9A 1A 1A 1A 3B C4 4C D0 - D0 CC CD DE 1E 66 66 86   ....;.L......ff.
000001A0: 66 6E F0 A0 00 11 85 22 - 2D 79 79 69 7B CC 23 E6   fn....."-yyi{.#.
000001B0: 60 60 60 62 F1 0D 21 91 - 91 93 CC DD A2 18 CC CB   ```b..!.........
000001C0: 1C 66 56 F0 F8 40 0A 79 - 65 B3 D2 DA D4 87 DA 84   .fV..@.ye.......
000001D0: B0 60 3A C4 B1 2D 59 79 - 70 EC 79 FA 3A 54 3E C9   .`:..-Yyp.y.:T>.
000001E0: 71 3A B4 E4 49 00 C0 75 - 04 10 3C 5C 0C 08 3E D8   q:..I..u..<\..>.
000001F0: 08 62 3D 80 A6 15 B5 89 - 05 CA B6 5F 02 00 F6 51   .b=........_...Q
00000200: D2 9B FB 2F 79 77 09 DD - 15 2A DE D0 11 16 25 FF   .../yw...*....%.
00000210: 7E 3C 95 A3 6F 67 12 81 - 04 7C 10 0B 87 85 E5 FE   ~<..og...|......
00000220: 12 BE DD 92 D2 FF CD 11 - 3D 44 55 78 DF F9 23 04   ........=DUx..#.
00000230: EC 30 18 0E 59 FF E2 A5 - 54 75 B9 69 97 30 60 3C   .0..Y...Tu.i.0`<
00000240: 42 12 A9 EA AF C2 54 54 - 05 3A D2 72 78 20 60 05   B.....TT.:.rx `.
00000250: AA A1 24 7F 79 39 41 C4 - FE 6A C9 69 E9 6B FF CB   ..$.y9A..j.i.k..
00000260: 87 DE 78 01 C2 5F 84 B2 - EF 2B F3 2D 91 9E 2B 12   ..x.._...+.-..+.
00000270: A2 95 7F 68 7A DA 56 1E - 9A 25 00 69 76 7E AA 2F   ...hz.V..%.iv~./
00000280: 2E 8A BD FA AB 9C DF CB - 97 8D D3 A9 EA C1 80 EC   ................
00000290: 55 E2 FA A9 5D F2 C9 88 - 5C 70 77 54 5E 11 C5 8F   U...]...\pwT^...
000002A0: 64 F4 52 7D E7 B1 41 78 - E9 58 8E D0 DE 15 B8 ED   d.R}..Ax.X......
000002B0: 57 87 9F FC B8 3B BB 2A - E3 72 73 95 54 7C 25 C5   W....;.*.rs.T|%.
000002C0: 5F 97 DF 66 5F 30 48 68 - 3C FA 9E DE 31 85 24 AE   _..f_0Hh<...1.$.
000002D0: 76 B6 54 2C 86 9D 82 F9 - 0D 0C CC 5E 66 ED 0F 33   v.T,.......^f..3
000002E0: 33 33 33 27 68 79 99 19 - 19 19 3B 42 80 00 48 14   333'hy....;B..H.
000002F0: 90 B0 B4 B1 E5 CF 30 6F - 99 79 79 79 83 C4 2C CC   ......0o.yyy..,.
00000300: 0C 0C 0C 5D E1 DD 98 97 - 5F E7 AF EF 30 D9 7B 8B   ...]...._...0.{.
00000310: 7F 07 AA BF 00 CC B0 6A - F2 F8 37 3D 80 18 0E 65   .......j..7=...e
00000320: 70 BE AA 56 AA EC 1E 66 - 90 C6 40 60 3A D5 8F BF   p..V...f..@`:...
00000330: 0B 95 D5 63 A1 1A AA E6 - 40 4D 7D 06 0B E8 10 C4   ...c....@M}.....
00000340: 90 80 3F DB 4B B5 B0 2D - F1 5C C8 07 84 21 EA 54   ..?.K..-.\...!.T
00000350: C7 1C 14 72 13 4D 97 AB - FC 57 69 97 C7 03 03 E4   ...r.M...Wi.....
```

```
temporal_reference: 0
split_screen_indicator: 0  document_camera_indicator: 0
source_format: 2
vop_quant: 5
```

```
Start Address: 00000000
00000000: 00 00 80 02 08 05 1E 40 - 0C 07 7D B0 18 00 E0 0F  .......@..}.....
00000010: B2 D9 41 0A F6 5C B1 25 - 73 78 21 89 60 C2 FC 89  ..A..\.%sx!.`...
00000020: 20 C4 FD 0F C1 CD FE 22 - 06 03 B4 18 31 00 0D 06   ......."....1...
00000030: 1C 1B E2 E7 D0 09 76 1F - 35 48 E0 5E B7 00 60 7E  ......v.5H.^..`~
00000040: 44 B0 61 7E 44 90 62 7E - 87 E0 E6 FF 38 83 03 F0  D.a~D.b~....8...
00000050: 25 83 0B F2 24 83 13 F4 - 3F 07 37 FB FC FC 11 F6  %...$...?.7.....
00000060: E3 D0 6A 1C C1 81 F7 12 - C1 85 F9 12 41 89 FA 2F  ..j.........A../
00000070: 07 37 F9 C4 B8 21 89 76 - AB 06 0B E4 49 06 0C 14  .7...!.v....I...
00000080: 7F 41 83 FA 2F 62 03 8F - FD 06 F8 24 01 B5 60 C3  .A../b.....$..`.
00000090: 83 89 60 E5 81 B7 1E 84 - 71 FE 0C 0F B0 43 06 17  ..`.....q....C..
000000A0: E0 49 06 27 E8 7E 0E 6F - F3 60 06 0F C4 B5 6A 95  .I.'.~.o.`....j.
000000B0: E8 92 24 F3 E0 C2 FD 17 - AC 0E 3F F4 53 88 30 01  ..$.......?.S.0.
000000C0: C0 68 BC 0A C1 83 87 DA - 7C 14 36 76 0D 1F 83 0B  .h......|.6v....
000000D0: F4 24 83 13 F4 5E 0E 6F - F3 20 F5 5F 01 82 F9 12  .$...^.o. ._....
000000E0: 5B 06 17 E4 BC 1C DF E6 - 3F FD 43 9B 75 CE EF E7  [.......?.C‾u...
000000F0: 9E D5 4B C7 EA 54 83 03 - F2 24 F1 80 50 97 E0 39  ..K..T...$..P..9
00000100: 7F CE 5F 12 44 9F D0 60 - FE 44 90 62 7E 87 E0 E6  .._.D..`.D.b~...
00000110: FF 38 FE CB 38 6D D4 02 - 1D 4F C0 D7 36 73 04 31  .8‾..8m...O..6s.1
00000120: 24 18 5F 91 24 18 9F A1 - F8 39 BF CE 20 86 10 C1  $._.$....9.. ...
00000130: 85 F9 12 41 89 FA 2F 07 - 37 F9 C4 F3 93 A1 F8 33  ..‾A../.7......3
00000140: CF 69 ED EF FF FF D5 FF - 6F EA 3B 4E B8 89 39 0B  .i......o.;N..9.
00000150: 95 D5 F0 FB 6C F9 C7 20 - 86 0C 18 2F D1 0C 5D 61  ....l.. .../..]a
00000160: B8 23 D8 88 3E FD F0 60 - C2 02 12 7A 3F A0 E5 7F  .#..>..`...z?...
00000170: 45 BF 17 83 06 0E 01 C0 - 50 BE 83 15 F8 F6 D8 69  E.......P......i
00000180: C3 43 43 C3 36 43 CE 16 - 12 16 11 B2 1C 00 02 10  .CC.6C..........
00000190: AE 88 C1 80 EC 06 0C 40 - 10 41 87 06 F8 BA 04 88  .......@.A......
000001A0: 18 0E C0 60 C4 01 04 18 - 70 60 CD FC F3 A8 94 D5  ...`....p`......
000001B0: 23 82 7C E0 A0 FF E0 5B - 21 2F 08 F8 10 81 83 FC  #.|....[!/......
000001C0: A0 C3 82 89 60 C5 81 BE - 09 E4 21 03 07 F6 0C 40  ....`.....!....@
000001D0: 28 96 0C 58 23 DB 61 1F - 13 1C E4 08 15 42 A2 F0  (..X#.a......B..
000001E0: 2A E8 83 F0 C3 DF C6 AD - 84 4D FF 9A 72 0D CF 68  *........M..r..h
000001F0: 1F D9 78 E7 07 C2 82 1A - B0 53 FC 6B 0C 08 65 E0  ..x......S.k..e.
00000200: A7 7B A0 1E A4 FF E0 12 - F7 18 E6 E8 E8 70 43 1F  .{...........pC.
00000210: 82 9D F0 C0 86 3F 05 3B - DC C0 F5 27 81 07 07 F5  .....?.;...'....
00000220: ED D5 CD E7 A0 C1 0C 7E - 0A 77 BE 8F A8 FC 7E AE  .......~.w....~.
00000230: 2A 03 F5 01 F7 41 E5 27 - 81 1E B9 ED F3 DB 79 29  *....A.'......y)
00000240: 02 E8 30 21 8A 8B D5 64 - F7 FA 22 DA CA 41 6A 80  ..0!...d.."..Aj.
00000250: 90 10 01 41 04 7F 8F ED - CF EE 65 72 3F F5 5F 25  ...A......er?._%
00000260: FF 92 E4 06 3B F9 EE 70 - 4F 0F FF 9F 5A E1 70 30  ....;..pO...Z.‾p0
00000270: 02 5E 4C A7 44 12 55 92 - F0 60 04 22 81 E0 F0 46  .^L.D.U..`."...F
00000280: 6B BA 1E 9D 44 CB 25 DF - 97 AB 5A 55 05 13 61 22  k...D.%...ZU..a"
00000290: 1C 99 36 B4 C6 F6 FB E9 - 49 9C 80 26 29 53 78 41  ..6.....I..&)SxA
000002A0: 06 00 41 59 74 8A CB 94 - AB 03 3E 51 DF 27 92 37  ..AYt.....>Q.'.7
000002B0: E2 D7 42 A1 FF D5 3E D2 - 66 20 0F 06 00 34 0F DF  ..B...>.f ...4..
000002C0: AB 55 AA 62 FC 1E A4 1C - BD BE 3E 26 48 48 38 2C  .U.b......>&HH8,
000002D0: CB 65 E5 86 20 5B 61 C0 - 00 22 0A D4 8C 18 0E C0  .e.. [a.."......
000002E0: 60 C4 01 04 18 70 6F 8B - BF C8 C1 80 EC 06 0C 40  `....po........@
000002F0: 10 41 87 06 F8 BA 05 D4 - 4A 6A 91 C1 3E 70 50 70  .A......Jj..>pPp
00000300: 50 6D 90 91 C2 3F 8F 95 - 81 E5 6A 72 D1 D7 FD C6  Pm...?....jr....
00000310: C0 EC 51 CD 39 09 0A D9 - 0B 47 F0 60 82 96 08 31  ..Q.9....G.`...1
00000320: 18 FA 8C 5F C1 82 0A 06 - 1B DC 1C A0 48 58 66 D8  ..._........HXf.
00000330: 6A 7F E0 07 01 90 86 0C - 48 1A A2 87 BF 28 06 00  j.......H....(..
00000340: 34 18 5F C0 83 01 88 05 - F5 07 33 F6 1A 1D B6 1A  4._.......3.....
00000350: 90 75 50 90 22 04 15 78 - 05 C0 3B FA 58 10 E0 C2  .u‾P."..x..;.X...
```

```
temporal_reference: 0
split_screen_indicator: 0  document_camera_indicator: 0
source_format: 2
vop_quant: 5
```

```
Start Address: 00000000
00000000: 00 00 80 02 08 05 0E 68 - 5F 92 F3 38 23 F3 FB F0   .......h_..8#...
00000010: 29 67 38 64 C9 54 54 A0 - 70 2F 36 FB 2A 91 F4 B6   )g8d.TT.p/6.*...
00000020: 6A 4A 6C DB E2 47 21 77 - BF 55 5A B2 66 51 3D D1   jJl..G!w.UZ.fQ=.
00000030: FE 7B D5 CF 36 1E F8 4B - F2 90 85 F5 5F F6 F4 44   .{..6..K...._..D
00000040: B0 09 53 C5 CA BF A9 AA - DB 10 CC 30 6C A8 20 45   ..S........0l. E
00000050: 53 6A E0 77 98 A5 7B C6 - 80 C9 E2 DC F6 CA 9B 21   Sj.w..{........!
00000060: 13 B1 C8 2F C8 63 25 65 - 85 AF 31 8A D5 7C 46 56   .../.c%e..1..|FV
00000070: AB EC F3 FE A3 07 78 87 - 65 65 01 91 92 B1 FA A2   ......x.ee......
00000080: E0 37 F9 E2 33 00 60 7E - 47 C5 CA 81 0D 59 7F A1   .7..3.`~G....Y..
00000090: 77 00 EB 03 B3 4E E9 8C - C3 A7 30 F2 B1 2C 47 12   w....N....0..,G.
000000A0: 81 87 FC 06 30 08 33 32 - 12 3C A0 B8 BA 2C 19 1A   ....0.32.<...,..
000000B0: AB 00 F1 28 BF E5 DE 12 - 84 B5 4A FF 15 8E FD 2F   ...(......J..../
000000C0: 67 B2 C6 92 9E 3B 12 04 - 9F 01 E5 50 BE E7 E4 9A   g....;.....P....
000000D0: D2 DA 69 D7 2C 9D 98 31 - 87 92 FE D5 07 5E 59 5E   ..i.,..1.....^Y^
000000E0: 5E 7C D7 E3 F1 F7 72 9F - 35 F5 D6 83 C9 45 EE C7   ^|....r.5....E..
000000F0: 21 A9 72 B9 DB 0E 54 B5 - E6 06 0F 32 76 8A B9 89   !.r...T....2v...
00000100: 91 93 43 38 6D D6 29 E4 - E2 25 22 3B 2E FF BD E9   ..C8m.)..%";....
00000110: CE E3 89 84 4A 91 E6 AA - 07 4A 3D 84 6F 0B 91 C3   ....J....J=.o...
00000120: 99 39 CE D0 3F EF 81 FE - 7F 7C 3C 11 17 EF 39 F1   .9..?....|<...9.
00000130: B9 A3 72 F2 F2 E5 5B F5 - 79 91 55 B6 AD E5 D1 1E   ..r...[.y.U.....
00000140: 31 BE 93 55 7B 37 04 76 - 5A E9 E3 F5 7E 8A EE 0F   1..U{7.vZ...~...
00000150: FD 15 F9 95 5F 9B 2E 55 - 99 22 77 83 26 7A 82 A0   ...._..U."w.&z..
00000160: 0D 12 02 0D BE 12 8B 95 - 8F 7D AA 0B D5 C8 3C BD   .........}....<.
00000170: EF 54 F1 AD 3A 84 10 15 - 82 18 F9 57 6F FF 35 4D   .T..:......Wo.5M
00000180: 02 9C DE 93 AA 09 5F 2F - 52 25 89 14 21 01 EF AB   ......_/R%..!...
00000190: 2F 53 0B 95 7B DE 2E 08 - 41 0B 47 5A A2 74 BD 5C   /S..{...A.GZ.t.\
000001A0: 96 DD ED 53 2B CE 02 10 - 1C C8 A3 20 FF 6A D1 44   ...S+...... .j.D
000001B0: 4D 86 5C FF 8E 81 38 00 - 01 08 57 9A B7 F6 54 CE   M.\...8...W...T.
000001C0: B6 16 18 97 AA F1 7D BD - F8 8C D4 B9 DD EA 42 53   ......}.......BS
000001D0: 29 6F A4 F0 EE 71 AA 52 - 7C D3 92 A9 A0 A5 E6 4D   )o...q.R|......M
000001E0: 11 D8 FE B5 1A EC 27 73 - 7B F9 11 CB 3D 77 DC FA   ......'s{...=w..
000001F0: 9B DD 4D 76 0E 9A EB 66 - 0B 64 BF 66 49 12 EA 57   ..Mv...f.d.fI..W
00000200: 18 AB F2 2B 9F 51 DA 23 - F3 FC 4B 23 CB 4B A8 F5   ...+.Q.#..K#.K..
00000210: 54 02 FC A8 81 C3 FE 77 - 81 79 CC 00 36 28 08 43   T......w.y..6(.C
00000220: FF 82 A2 96 F0 5E CA 5D - 80 86 A8 47 1D 79 5F 7C   .....^.]...G.y_|
00000230: D8 30 5F 2A C1 0B 89 D4 - 17 EC 22 3D 06 03 AB E0   .0_*......"=....
00000240: 78 21 67 B4 76 C6 AD 7D - F0 63 00 9E B0 0C 00 80   x!g.v..}.c......
00000250: 91 4B C4 B5 7F E2 82 EF - 89 1A 90 B8 77 62 6B D8   .K..........wbk.
00000260: 7D E2 18 CB D2 00 3F BF - AD E0 F7 92 B0 3C F7 50   }.....?......<.P
00000270: 9B 66 56 10 44 BF 89 5F - 1F 7C BE FC B8 7B 91 55   .fV.D.._.|...{.U
00000280: 9E C8 A1 5C 93 07 5B C2 - 35 21 2C 4B 54 07 FC AB   ...\..[.5!,KT...
00000290: B7 F5 5E 72 08 AA D5 AA - 54 0C 6F F0 D1 D1 48 1C   ..^r....T.o...H.
000002A0: 4A 5F 2F 40 BA B0 09 70 - 04 0C 03 EA EF F3 09 60   J_/@...p.......`
000002B0: 42 1D 80 78 BB DE A4 6F - 5C 06 00 3E 03 05 EC 8A   B..x...o\..>....
000002C0: 03 97 FA BA 24 89 23 F8 - 0C 0F 9C 94 18 30 5A 24   ....$.#......0Z$
000002D0: 01 E1 D7 B6 6E 6B 1D 21 - 5C 06 00 43 F0 18 1F 90   ....nk.!\..C....
000002E0: 24 3E 2D FD 7B 20 43 1F - FE 03 03 DF DC 1E 51 21   $>-.{ C.......Q!
000002F0: 43 5C DC 3E D6 10 FF EF - FF 67 67 0D 44 09 4A 46   C\.>.....gg.D.JF
00000300: 2F 60 56 3F 1F 8F 44 B9 - B4 18 30 AF 17 C1 D4 03   /`V?..D...0.....
00000310: 7C 02 CD A6 19 26 04 28 - 25 AB 06 00 3A 7E 7F E0   |....&.(%...:~..
00000320: 7E 8F C2 13 43 AF A8 1E - 89 43 DD E3 40 7B 75 A3   ~...C....C..@{u.
00000330: EB 62 57 A8 43 06 03 CF - E2 50 F7 07 A1 00 BA DC   .bW.C....P......
00000340: CD 8D 6F E1 41 29 75 05 - 00 40 55 AD 7A 66 12 BC   ..o.A)u..@U.zf..
00000350: BD FF F4 9D E6 9F 54 AF - EC FF 3C 3F 9F 03 1C 85   ......T...<?....
```

```
temporal_reference: 0
split_screen_indicator: 0  document_camera_indicator: 0
source_format: 2
vop_quant: 5
```

```
Start Address: 00000000
00000000: 00 00 80 02 08 05 1E 8D - EA AA DA 5C AA FA AB F0   ...........\....
00000010: 1B 55 EB BF F6 B4 A7 D3 - B1 36 98 4C A5 E2 48 30   .U.......6.L..H0
00000020: 3F 0A 01 81 09 9F 06 00 - 34 14 03 D0 60 7D 15 0F   ?.......4...`}..
00000030: E2 BF D1 2E 5E 89 50 21 - 7A 29 BC E8 F6 5E D3 8E   ....^.P!z)...^..
00000040: 1E 12 42 00 30 1C 40 A0 - 12 00 38 18 0E B0 0F 55   ..B.0.@...8....U
00000050: 07 DF D0 60 3C C4 8A 25 - 78 03 84 90 60 3B 84 A4   ...`<..%x...`;..
00000060: D5 56 7E E0 8D F8 5D F1 - E6 5B BE 69 ED 43 E1 2C   .V~...]..[.i.C.,
00000070: 03 7E 0C 08 39 75 1F 09 - 60 C0 06 83 03 F0 25 83   .~..9u..`.....%.
00000080: 01 DE 24 09 43 E8 5D 7C - 0C 08 69 70 94 25 48 0C   ..$.C.]|..ip.%H.
00000090: 00 6A 9C 1D 2A 1D F8 7B - BF A3 BB FB D7 B9 29 40   .j..*..{......)@
000000A0: BC E9 FA C0 60 42 14 73 - 9B 99 7D FD F0 F1 3B 1A   ....`B.s..}...;.
000000B0: D6 AD 23 67 5E B8 5C 0C - 07 58 20 09 62 4A A0 81   ..#g^.\..X .bJ..
000000C0: EF 00 6C 2F 2F B6 17 55 - 60 19 F8 25 7E FB B4 18   ..l//..U`..%~...
000000D0: 2F 99 FF AA F3 54 74 AD - 47 F4 1C 40 24 B2 FA 07   /....Tt.G..@$...
000000E0: A5 57 F9 72 26 CD AC 7A - 27 A2 B6 92 E1 23 EA A0   .W.r&..z'....#..
000000F0: 30 3D E3 F2 EA 01 80 C0 - 76 84 31 22 97 7C 03 95   0=......v.1".|..
00000100: E1 78 91 E0 0F 11 95 82 - 0C 8A AA 95 05 F0 0E FC   .x..............
00000110: BA FD 44 E6 28 FD 7B CA - B0 60 7A FD 4B 95 0F A0   ..D.(.{..`z.K...
00000120: 30 01 E1 0F D7 E0 A3 2F - F8 30 3E E3 EB E8 CF C1   0....../.0>.....
00000130: 41 4B D5 F9 18 94 AE FD - 80 08 6A 2E 9F C5 2A FD   AK........j...*.
00000140: FC D1 13 DA 72 50 03 81 - 80 18 54 10 6C 12 F0 7F   ....rP....T.l...
00000150: E0 81 EF FA 59 3C A4 10 - EE 81 D9 EB B3 3C C2 9E   ....Y<.......<..
00000160: B9 ED 42 5C 06 07 E4 18 - 3F AC AD A8 B2 48 AC B8   ..B\....?....H..
00000170: 18 70 52 FC 1E AA BE D1 - D5 95 29 F9 F5 60 C0 0B   .pR.......)..`..
00000180: 04 10 81 F5 12 17 83 01 - E6 0C 00 8A B1 2C 7C 5E   .............,|^
00000190: 24 89 01 0A E0 41 FF 95 - 7F 96 0F 81 85 FA 9E B7   $....A..........
000001A0: C0 7C 7D AA 67 1A 34 D8 - 5E 07 C7 94 0F 02 17 84   .|}.g.4.^.......
000001B0: BC E7 C7 E0 7E 8F 94 81 - 75 5E B1 5C 88 EE 37 4F   ....~...u^.\..7O
000001C0: C5 04 00 60 3A 3C 5C 10 - C4 B0 84 0C 07 32 A0 60   ...`:<\......2.`
000001D0: 43 3F EC 03 C3 F0 85 E5 - 6A 15 7C 18 6F B8 07 7F   C?......j.|.o...
000001E0: 55 2B 99 F1 F0 96 5D C2 - 76 50 0F 2E 08 55 44 57   U+....].vP...UDW
000001F0: 3E 5D 6C CF 2B F4 E2 4A - 18 B1 82 0A A0 0C 1F 17   >]l.+..J........
00000200: 79 48 94 AD 54 F7 81 44 - 08 45 DB 98 D4 C8 DE 2C   yH..T..D.E.....,
00000210: 6D EE 80 C0 8E 89 02 48 - 07 AA 00 DF 78 20 83 00   m......H....x ..
00000220: 28 10 84 85 43 EF 03 00 - 1E 01 C3 E1 E1 70 07 89   (...C........p..
00000230: 55 57 55 8F C4 B0 60 FE - FC A8 10 FF 68 F4 7D D5   UWU...`.....h.}.
00000240: 65 FD 05 0D 30 C1 01 80 - 1A 08 05 D0 BD 41 75 12   e...0.........Au.
00000250: CB 87 E2 4F 82 18 21 51 - D0 28 3E AB D7 3D E1 E2   ...O..!Q.(>..=..
00000260: 8B F5 4A 7F FF CE F8 79 - C9 21 C6 71 F8 30 1B E0   ..J....y.!.q.0..
00000270: C0 0C 0F 95 29 06 07 CE - 04 1D E0 F0 03 BC AC 4B   ....)..........K
00000280: 1F A5 FE C2 E9 E4 BF FF - 5E A8 01 C0 C0 74 03 00   ........^....t..
00000290: 26 10 64 1E 00 6A A5 5F - C0 34 10 F4 21 2B 06 2B   &.d..j._.4..!+.+
000002A0: F4 75 51 00 5B 9F 95 09 - 0A 94 4E 2B 50 06 6E E9   .uQ.[.....N+P.n.
000002B0: 1C EA A0 0F 06 03 C5 50 - FE 89 3F 1F 0F 4B B4 18   .......P..?..K..
000002C0: 30 4F AA 03 96 24 11 E5 - C3 2F 86 06 03 AE 51 F4   0O...$.../....Q.
000002D0: F8 20 78 B8 B8 B8 4A B6 - D5 53 F0 0E 8F BF CA A4   . x...J..S......
000002E0: 7D 3F 54 B3 62 B1 1E E5 - 69 F2 1E 1F 78 BC 4B 04   }?T.b...i...x.K.
000002F0: 05 3A 3D A1 00 BC 18 30 - 5C 05 18 1C 80 77 C9 FD   .:=....0\....w..
00000300: 83 B5 53 14 CE FB 2E B9 - C8 18 0F 11 28 18 1F 60   ..S.........(..`
00000310: 0D 12 F0 49 00 C1 F9 77 - D5 AA F8 FA 83 02 10 3F   ...I...w.......?
00000320: 12 47 90 BF DD 91 5C 08 - 2A 87 C5 E0 6A A9 B3 3D   .G....\.*...j..=
00000330: 39 01 C3 FE 81 05 02 B5 - 4A 15 97 09 20 81 54 67   9.......J... .Tg
00000340: CB AD 00 E0 0F 12 BF 27 - D9 F2 B2 E2 F2 F8 A8 0B   .......'........
00000350: 8E 89 9E ED 6C 75 9D AD - 27 35 07 B1 58 1D E7 94   ....lu..'5..X...
```

Conclusion

In this dissertation work we have experimented on various types of fast motion estimation algorithms. We tried to reduce the computational complexity without compromising on the quality of the motion compensated image which is very important for transmission of video over mobile. The code is ported on to TMS32064XX processor and we tried to optimize the code in terms of no. of cycles consumed. In our work we have used simple profile of the MPEG-4 standard.

The problem with Fast motion estimation algorithms like Hexagonal and Diamond based search is that they tend to get stuck in the local minima which results in wrong motion vector. We tried to reduce this disadvantage by combining both multi-resolution analysis and spatio-temporal information for obtaining better motion vector prediction.

FUTURE WORK

This dissertation work can be further extended by incorporating half and quarter pixel motion estimation. The same motion estimation algorithms can be used tested on higher profiles of MPEG-4 standard and there quality needs to be tested.

REFERENCES

[1] "Advanced video coding for generic audiovisual services", SERIES H: "AUDIOVISUAL AND MULTIMEDIA SYSTEMS", Infrastructure of audiovisual services – Coding of moving video ITU-T Recommendation H.264, March 2005.

[2] Ce Zhu, Xiao Lin, and Lap-Pui Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation", IEEE Trans. on Circuits and systems for Video Technology, vol. 12, no. 5, pp.349-355, May 2002.

[3] Ce Zhu , Xiao Lin, Lappui Chau, and Lai-Man Po, "Enhanced Hexagonal Search for Fast Block Motion Estimation", IEEE Trans. on Circuits and Systems for Video Technology, vol. 14, no. 10, pp. 1210-1214, October 2004.

[4]. Chun-Ho Cheung and Lai-Man Po, "Novel Cross-Diamond-Hexagonal Search Algorithms for Fast Block Motion Estimation", IEEE Trans. on Multimedia , vol. 7,no. 1, pp. 16-22, Februrary 2005.

[5] Freseric Dufaux and Fabrice Moscheni, "Motion Estimation Techniques for Digital TV: A Review and a New Contribution", Proceedings of the IEEE, vol.83, no. 6, pp. 858-875, June 1995.

[6] Hyun-Wook Park and Hyung-Sun Kim, "Motion Estimation Using Low-Band-Shift Method for Wavelet-Based Moving-Picture Coding", IEEE Transactions on Image Processing, vol. 9, no. 4, pp. 577-587, April 2000.

[7] Information technology — "Coding of audio-visual objects" — Part 2: "Visual International Standard", ISO/IEC 14496-2, Third Edition, June 2004.

[8]. Jinwen Zan, M. Omair Ahmad, and M. N. S. Swamy, "Comparison of Wavelets for Multiresolution Motion Estimation", IEEE Trans. on Circuits and Systems for Video Technology, vol. 16, no. 3, pp. 439-446, March 2006.

[9] Jooheung Lee, N. Vijaykrishnan, Mary Jane Irwin, and Wayne Wolf, "An Efficient Architecture for Motion Estimation and Compensation in the Transform Domain", IEEE Trans. on Circuits and Systems for Video Technology, vol. 16 , no.2, pp. 191-201, February 2006.

[10] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, "A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation", IEEE Trans. on Circuits and Systems for Video Technology, vol. 8, no. 4, pp. 369-377, August 1998.

[11] Junavit Chalidabhongse and C.C. Jay Kuo, "Fast Motion Vector Estimation Using Multiresolution-Spatio-Temporal Correlations", IEEE Trans. on Circuits and Systems for Video Technology , vol. 7, no. 3, pp. 477-488, June 1997.

[12] Ko-Cheung Hui, Wan-Chi Siu, and Yui-Lam Chan, "New Adaptive Partial Distortion Search Using Clustered Pixel Matching Error Characteristic" IEEE Trans. on Image Processing, vol. 14, no. 5, pp. 597-601, May 2005.

[13] K.R.Namuduri,"Motion Estimation Using Spatio-Temporal Contextual Information", IEEE Trans. on Circuits and Systems for Video Technology, vol. 14, no. 8, pp. 1111-1115, August 2004.

[14} Luis Louro, Paulo Santos, Nuno Rodrigues, Vitor Silvat and Sergio Faria, "DSP performance evaluation for motion estimation", Seventh International Symposium On Signal Processing and its Applications, vol. 2, pp. 137 - 140 , July 2003.

[15] Monalisa Sarma, Debasis Samanta and Anindya Sundar Dhar, "Motion Estimation using Multi-Resolution based on Haar Wavelet Transformation" TENCON 2004. 2004 IEEE Region 10 Conference, vol. 1, pp. 247-250, November 2004.

[16] Mohammed Ghanbari, "Standard Codecs: Image Compression to Advanced Video Coding", The Institution of Electrical Engineers, London, United Kingdom, 2003.

[17] Narendra Ahuja and Alexia Briassouli, "Joint Spatial and Frequency Domain Motion Analysis", Processings of the 7th International Conference on Face and Gesture Recognition, pp. 203 – 208, April 2006.

[18] Prasad RSV and Ramkishore Korada, "Efficient Implementation of MPEG-4 Video Encoder on RISC core", IEEE Trans. on Consumer Electronics, vol. 49, no. 1, pp. 204-209, February 2003.

[19] Raymond Westwater and Borko Furht, "Real-Time Video Compression-Techniques and Algorithms", Kluwer Academic Publishers, 1997.

[20] S. Mallat, “A Theory for Multi-resolution Signal Decomposition: The Wavelet Representation”, IEEE Trans. PAMI, vol. 11, no. 7, pp. 674-693, December 1989.

[21] Seong-Ju Kim, Jong-Hak Ahn, Changhoon Yim, “Adaptive Motion Estimation Algorithms for MPEG-4 Video Coding”, Seventh International Symposium on Signal Processing and its Applications , vol. 2, pp. 141 - 144 , July 2003.

[22] Thomas Sikora, “MPEG Digital Video-Coding Standards”, IEEE Signal Processing Magazine, pp. 82-100, September 1997.

[23] “TMS320C6000 Programmer’s Guide” Texas Instruments, Literature Number: SPRU198G, August 2002.

[24] “TMS320C6000 Optimizing Compiler User’s Guide” Texas Instruments, Literature Number: SPRU187L, May 2004.

[25] Yao Nie and Kai-Kuang Ma, “Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation”, IEEE Trans. on Image Processing, vol. 11, no. 12, pp. 1442-1449, December 2002.

[26] Yong-Sheng Chen, Yi-Ping Hungand Chiou-Shann Fuh, “Fast Block Matching Algorithm Based on the Winner-Update Strategy” , IEEE Trans. on Image Processing, vol. 10, no. 8, pp. 1212-1222, August 2001.

[27] Yu-Wen Huang, Shyh-Yih Ma, Chun-Fu Shen, and Liang-Gee Chen, “Predictive Line Search: An Efficient Motion Estimation Algorithm for MPEG-4 Encoding Systems on Multimedia Processors”, IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, pp. 111-117, no. 1, January 2003.