PERFORMANCE ENHANCEMENT OF ANIMATED IMAGES COMPRESSION USING VIDEOCONFERENCING TECHNIQUES

By
Ammar Mohammed Kamel

Supervisor
Dr. Mohamed B. Al-Zoubi

Submitted in Partial Fulfillment of the Requirements for the Degree of Master Science in Computer Sciences

Faculty of Graduate Studies
University of Jordan

May 2003

This thesis was successfully defended and approved

on:…27-5-2003….

Examination Committee	Signature
Dr. Moh'd Belal Al-Zoubi	
Assist. Prof. of Graphics and Pattern Recognition	……………………………..
Dr. Ahmad Sharieh	
Assoc. Prof. of Parallel Processing	……………………………..
Dr. Sami Serhan	
Assist. Prof. of Compiler Design	……………………………..
Dr. Mahmoud A. Hassan	
Assoc. Prof. of Computer Networks	……………………………..
Prof. Dr. Saleh Oqeili	
Prof. of Architecture	……………………………..

DEDICATION

To my father, mother, and family.

To the coming partner of my life.

And before them to Iraq.

AKNOWLEDGMENT

I have to express my thank and grateful to my research supervisor, Dr. Moh'd Belal Al-Zoubi for his continuous support and significant advice as well as his brotherly help beyond academic assistance.

I would like to thank Dr. Sami Serhan for his patient, honest help. And for all the encouragement he gave me.

I wish to thank, Prof. Jubair J. Al- Ja'afer for his valuable advice.

My sincere appreciation for all the professors and administrative staff in KASIT, Jordan University for their important support along the last three years.

I would like to thank my friend Raheem Khalaf Al-Zubaidy for his help to accomplish this work.

Finally, I thank estrangement friends, companions of joy and sorrow, the Iraqi students.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ATM :A synchronous Transform Mode

3SS :Three-Step Search block-matching algorithm

B :Bi-directional Picture

BMA :Block Matching Algorithm

CCIR :Committee Consultative International for Radio Communication

CCITT :Committee Consultative International for Telephone and Telegraph

CIF :Common Intermediate Format

CODEC :Video compression/decompression technique

C_r :Compression ratio

DCT :Discrete Cosine Transformation

DPCM :Differential Pulse Coded Modulation

EDTV :Extended Definition Television

e_{RMS} :Root-mean-square error

FS :Full Search block-matching algorithm

GOBS :Group of Blocks

GOP :Group of Picture

H.261 :Video Coding Standard for Videoconferencing Applications

H.263 :Video Coding Standard for Videotelephony Applications

HDTV :High-Definition Television

HVS :Human Visual System

I :Intra-Picture

IDCT :Inverse Discrete Cosine Transform

ISDN :Integrated Service Digital Network

ISO :International Standard Organization

ITU :International Telecommunication Union

ITU-R :International Telecommunication Union Radio

JPEG :Joint Picture Expert Group

Kbps :Kilo bit per second

KLT :Karhunen Loeve Trasfrom

LZW :Lemple-Ziv-Walch

MAD :Mean-Absolute Difference

MPEG :Moving Picture Expert Group

MSD :Mean-Squared Difference

MV :Motion Vector

OES :OddEven Search block-matching algorithm

P :Predictive Picture

PSNR :Peak Signal-to-Noise Ratio

QCIF :Quarter Common Intermediate Format

RLC :Run-Length Coding

SIF :Source Input Format

SNR_{RMS} :Root-mean-square signal-to-noise ratio

VLC :Variable-Length Coding

PERFORMANCE ENHANCEMENT OF ANIMATED IMAGES COMPRESSION USING VIDEOCONFERENCING TECHNIQUES

By

Ammar Mohammed Kamel

Supervisor

Dr. Mohamed B. Al-Zoubi

ABSTRACT

A digital image is defined as visual representation of meaningful data within spatial domain. Arranging certain number of digital images successively is called animated images or video. Digital video is video information that is stored and transmitted in a digital form. Digital video has been used for a number of years, for example, in the television broadcasting industry. Most digital images contain high degree of redundancy, which means that an efficient compression technique can significantly reduce the amount of information needed to store or transmit them.

Special techniques work to reduce the degree of information redundancy, which can be found between single pixels, between lines, or between frames, when a scene is stationary or slightly moving. In this thesis, H.263 technique has been adopted. The structure of this technique and characteristics has been studied. Moreover, factors that limit the efficiency of this technique has been identified through implementing and analyzing the well-known motion search techniques (Full search, 2-D logarithmic search, Conjugate search, and Three-step search). In this thesis, two new techniques

have been developed; Thresholding Half-pixel and OddEven motion search to enhance H.263 technique efficiency. The first technique increases the image smoothness of video frames. The second decreases the elapsed time of video compression (encoding delay time).

The results of this thesis proved the effectiveness of the proposed technique in comparison with the well-known motion search techniques in terms of delay time, image accuracy and compression ratio. In addition, this thesis concluded that decreasing encoding delay time and increasing video compression ratio and maintaining image accuracy would enhance the effectiveness of H.263 technique.

1. INTRODUCTION

1.1 Overview

Analog video communications technology has impressed the lives of people everywhere in the world. The overwhelming majority of households in the developed world have at least one television set, which provides a major source of entertainment to the home.

Digital video offers many advantages over analog video for the overwhelming majority of applications. Therefore, as digital video technology becomes maturer, a gradual shift is expected to take place from analog to digital technology. The ease with which digital video can be integrated with computers, together with the introduction of global communication networks that support the integration of video with audio and conventional computer data make a whole range of new applications possible.

Moving pictures are usually called "Video", referring to sequence of frames (set of images) that are displayed in order at a specific time. These frames are combined to refer to selected subjects.

Digital video is video information represented in digital from. Digital representation has a number of advantages over traditional analog video and television. All information can be represented in digital form, so the same techniques and systems can be used to store, process, and transmit a wide range of different types of data (multiple media or "multimedia"). The fast growth in digital processing power means the complex processing and coding operations can be carried out on digital video data in real time.

The number of computers and systems connected by networks such as the Internet has grown considerably in a relatively short time. Also, the universal, networks can handle higher volumes of data and higher transmission rate. The current networking structure is loosely defined and “heterogeneous” (consisting of a range of interconnected networks with different technologies and capabilities).

Digital video has an inherently high bandwidth (i.e., a digitized video signal require a very high data rate for transmission). In order to store and transmit this information effectively, it is necessary to develop techniques for video data compression (i.e., encoding them into smaller number of bits), otherwise raw video may require large storage space and bandwidth. The establishment of international standard for encoding video data has enabled a wide range of applications of using digital video transmission and storage. Image coding provides means of compressing digitized photographic images by 10 to 20 times. Current video coding techniques enabled video data compression rate between 20 to 50 times.

Special techniques, for example (MPEG1, MPEG2, MPEG4, H.261 and H.263) which take the characteristics of the video into account, can compress the video with a high compression ratio. All these techniques work to reduce the degree of information redundancy, which can be found between single pixels, between lines, or between frames, when a scene is stationary or slightly moving.

The H.261 is video coding standard published by the ITU (International Telecom Union) in 1990. It is designed for data rates which are multiples of 64Kbit/s, and is sometimes called p x 64Kbit/s (p is in the range 1-30). H.261 is intended for conferencing applications with only small, controlled amounts of motion in a scene, and with rather limited views consisting mainly of head-and-shoulders views of people along with the background.

The H.263 technique is designed for very low-bit rate coding, which is concerned with real-time two-way communication. The coding algorithm of the H.263 is similar to that used by the H.261 technique, however few changes can improve performance and error recovering. Several important features that are different from H.261 including the following new option: motion compensation with half-pixel accuracy and bidirectionally-coded macrobloacks, 8x8 overlapped block motion compensation, unrestricted motion vector range at picture boundary, and arithmetic coding are also used in the H.263. These features make the H.263 as most popular and important technique for videoconferencing applications. Therefore, in this thesis, the H.263 technique will be focussed on and its characteristics will be further studied.

Conferencing applications are designed to replace face to face communication. In order to provide effective interactive communication it is necessary to limit the end-to-end delay.

Videoconferencing can be used effectively in a number of different application areas; usually with the aim of avoiding the inconvenience and/or cost of traveling.

The data rate, transmission delay, and error or loss probability (bit error rate, cell or packet loss probability, error patterns) are videoconferencing parameters, where can affect the quality and reliability of digital video communications. The available data rate has a significant effect on the quality (image accuracy) and resolution of coded video information that can be transmitted through the network. Different coding techniques (i.e. H.261/H.263) are appropriate for different data rates. Transmission delay is very important since real-time video applications are sensitive to changes in transmission delay. Decoding video frames must be presented to the viewer at a constant rate, and if a particular frame is delayed too long then it cannot be displayed and is "lost". Errors and losses have a particular sever effect on the quality of decoded video.

Encoding delay time and image accuracy are major problem that limit, H.263 efficiency. Encoding delay time greatly contributes to videoconferencing interactivity achievement, through decreasing the elapsed time in the video compression. On the other hand, increasing image accuracy directly affects video compression ratio, therefore, new techniques are required to achieve this increase.

1.2 Aim of the Thesis

In this thesis, the H.263 constraints (encoding delay time and image accuracy) will be concentrated on through presenting and developing the H.263 videoconferencing technique. This will be done through the development of new algorithms and techniques to enhance video frame and increase the ability to speed up the transmission rate through the communication process. This task represents the main goal (aim) of this research work.

1.3 Thesis Outline

In addition to Chapter one, the study includes the following chapters:

- Chapter two presents a historical review of the H.263 technique.
- Chapter three deals with the image and video theoretical background and their concepts. It provides this chapter a brief illustration about the image and video compression techniques.
- Chapter four presents a detailed view of image and video standard compression techniques, especially (JPEG –sequential DCT based mode, H.261, H.263, and MPEG). The common motion estimation and compensation methods will be discussed also in this chapter.
- Chapter five proposes two novel techniques as an improvement of the H.263 motion estimation-compensation. A comparison between the proposed and the well-known techniques results will be made in this chapter also. Finally, the results will be discussed.
- Chapter six presents the conclusion and recommendations for future work.

2. LITERATURE REVIEW

2.1 Introduction

In this chapter, several studies on the H.263 technique are presented. These studies apply different kinds of solutions to solve (transmission delay, data rate, encoding delay time, image accuracy and error or loss probability) limitations. The following section explains the strategies of these studies and their experimental results.

2.2 Literature Review

Digital video has emerged since 1990s as a technology that can provide a new "dimension" to electronic communications, hence, L. Hanzo and P. Cherriman (1996), developed a bit rate control algorithm in order to maintain a selectable near-constant video bit rate. Their endeavors are focussed on exploring the quality versus bit rate performance of both systems for various image resolutions, in order to provide the required video quality, bit rate, frame rate, image size and resolution on a demand basis in adaptive multi-mode transceivers.

L. Hanzo and P. Cherriman (1996), proposed “power-controlled H.263-based robust video transceiver scheme” with several algorithms. The combination of algorithms guarantee a robust videophone performance over wireless channels across the area of traffic cells.

C. Zhu. (1997), has developed the payload format for encapsulating an H.263 bit stream in the Real-Time Transport Protocol (RTP). He defined three modes for the H.263 payload header. An RTP packet can use one of the three modes for H.263 video streams depending on the designed network packet size and H.263 encoding options employed. The shortest H.263 payload header (mode A) supports fragmentation at Group of Block (GOB) boundaries. The long H.263 payload headers (mode B and C) support fragmentation at Macroblock (MB) boundaries.

D. Bangi, et al (1997), presented a post-processing algorithm for low-bit rate videoconferencing on ISDN lines. This algorithm applies motion estimation and compensation techniques to predicate a moving objects in a video frames. The post-processing algorithm is based on the assumption that it is better to have a few good images than much poor quality ones. Although, spatial degradation usually can not be restored, temporal resolution can often be increased by motion compensated interpolation. Several CIF video sequences were used to evaluate the post-processing. Experimentally, good spatial resolution images were obtained for 50 Hz displays at the expense of the coding processing complexity.

T. Keller et al (1998), dedicated a study titled “Orthogonal Frequency Division Multiplex transmission of the H.263 encoded video over highly frequency-selective wireless networks”. They evaluated a 2 Mbps Universal Mobile Telecommunication System (UMTS) concepts by using H.263 video codec, assigned by novel packetisation and packet acknowledgement scheme.

B. Andreas et al (2000), presented the first VLSI implementation of a real-time color video compression/decompression system, based on the threedimensional discrete cosine transform (3D-DCT). This system is compared to motion-estimation/compensation based algorithm, where no motion estimation is required, reducing the number of encoding/decoding operation per pixel. The complexity of the implementation is dependent on the compression ratio.

R. Injong, (2002), presented several retransmission-based error control schemes that could be used for interactive video applications. In particular, the schemes do not require any artificial extension of control time and play-out delay. By correcting errors in a reference frame caused by earlier packet loss, the schemes prevent error propagation.

C. Peter et al (2000), suggested a burst-by-burst adaptive transceiver technique in order to increase the system bits per symbol capacity and conversely, invoking a lower order modulation scheme when the channel exhibits inferior channel quality. The main advantage of the proposed burst-by-burst adaptive transceiver technique is that irrespective of the prevailing channel conditions, the transceiver achieves always the best possible source-signal representation quality, such as (video, speech, or audio quality) by automatically adjusting the achievable bit rate and the associated multimedia source-signal representation quality in order to match the channel quality experienced.

G. Justin et al (2003), studied the application of Unequal Loss Protection (ULP) to motion compensation video over lossy packet networks. In practice, they focused on streaming video application over the Internet. The original ULP framework applies unequal amount of forward error correction (FEC) to embedded data to provide graceful degradation of quality in the presence of increasing packet loss.

The main contribution of this work is in illustrating how the ULP framework can be applied to motion compensated video (in particular, H.263 video). They applied the ULP framework to baseline H.263, a non-embedded video compression standard, by investigating re-ordering of the bit stream. The re-ordering process makes the bit stream appear embedded, allowing a receiver to display high quality video even at the high loss rates encountered in wireless transmission and current Internet.

F. Nick and W. Susie, (1999), presented an MPEG-2 to H.263 transcoder that accepts an interlaced MPEG-2 bit stream as an input and produces a lower-bit rate progressive H.263 bit stream as the output. The proposed algorithm exploits the properties of the MPEG-2 and H.263 compression standards to perform interlaced to progressive (field to frame) conversion with spatial downsampling and frame-rate reduction in a CPU and memory efficient manner, while additionally minimizing picture quality degradation as measured by PSNR.

S. Olivieri et al (1999), proposed a spatio-temporal recursive estimator that combines coding efficiency with a high computational efficiency. Experimentally, the new algorithm proves to be comparable to full-search block matching when encoding typical videoconferencing sequences in presence of additive noise, even though the computational burden has been greatly reduced.

L. Zhijun and D. Nicolas (2002), developed a new approach to refine motion vectors adaptively according to the motion of every frame or every macroblock in a frame. The proposed approach can improve the video quality and reduce predicative residues of every frame, hence reduce the transmission bit rate. The experiment and comparison results show that this approach can be used as a pre-processing tool to create various size of video files that can be saved in the video on demand server.

C. Chok-kwan and P. Lai-man (1997), developed a novel algorithm-Hierarchical Partial Distortion Search (HPDS) which reduces the number of pixels considered for each motion block instead of reducing the number of search locations by using partial distortion measure. It is based on the assumption that if the full distortion between two blocks is the global minimum, then there has a high probability that its partial distortion is also the global minimum. The searching procedures are divided into three search levels. It uses a coarse to fine approach to refine the search for each higher level. Complex mathematical formula was applied to predict the block similarity and extracting the motion vectors in different levels.

M. Alexis et al (2000), presented a new motion estimation algorithm. The algorithm named as Predictive Diamond Search (PDS), which is actually based on the Diamond Search (DS) algorithm, adopted inside the MPEG-4. This algorithm added some predictive criteria to DS algorithm that can significantly improve performance. The simulation results show that the proposed algorithm manages to have similar complexity with the DS algorithm, while having robust quality, similar to that of the Full Search (FS).

L. Po, and W. Ma (1996), proposed a new four-step search (4SS) algorithm with center-biased checking point pattern for fast motion estimation. Half-way stop technique is employed in the new algorithm with searching steps of 2 to 4 and the total number of checking points is varied form 17 to 27. Simulation results show that the proposed 4SS perform better than well-known three-step search and has similar performance to the new three-step search (N3SS) in terms of motion compensation error.

In this thesis, we propose algorithms based on H.263 motion estimation-compensation techniques. The algorithms and the results are discusses in chapter five.

3. IMAGE AND VIDEO CODING FUNDAMENTALS

3.1 Introduction

Image and video data coding refer to a process in which the amount of data used to represent image and video is reduced to meet a bit rate requirements, while the quality of reconstructed image or video must satisfy a requirement of a certain applications.

From the above definition, image and video data compression involves several fundamental concepts. This chapter is concerned with the general concepts of image and video coding.

3.2 The Digital Image Features

In general perspective, the digital images are a way of recording and presenting information visually. This information is manipulated as a collective of lighted point called "Pixels" and sorted in a manner that gives some objectives and meaning. Sampling, Quantisation, and Color are most important features of digital images. Below is a brief illustration of these principles:

3.2.1 Image Sampling: Sampling is a process of measuring the value of the image function $f(x,y)$ at discrete intervals in space. Each sample corresponds to a small, square area of image known as a Pixel. A digital image is a two-dimensional array of these pixels. Pixels are indexed by x and y coordinates, with x and y taking integer values. In deciding whether a digital image has been sampled appropriately, we must consider the rapidity with which the value of $f(x, y)$ changes as we move across the image. This rate of change is measured by spatial frequency.

Video standard enforces a particular *sampling rate* for a video signal. An RS-170 signal, for instance, has 485 active lines and each frame must have an aspect ratio of 4:3 [Efford, N., 2000].

3.2.2 Image Quantization: is the process of reducing the image data by removing some of the detail information by mapping groups of data points to a single point. This can be done either to the pixel values themselves $f(x, y)$ or to the spatial coordinates (x, y). Operation on the pixel values is referred as "gray-level reduction", while operation on the spatial coordinates is called "spatial reduction" [Unbaugh, S. E., 1999].

3.2.3 Image Color: A colored image is represented in electronic form in terms of "component" signals. A color can be synthesized by combining the three primary colors, Red, Green, and Blue (RGB). The RGB color component system is one means of representing colored images. Each of these primary components contains information about brightness "luminance" and color "chrominance". Alternatively, the luminance and chrominance information can be represented separately [Martyn, J., 1997].

A luminance signal (Y) can be obtained by adding R, G and B together, not in equal amounts, but in a sum which is weighted by the relative response of the eye.

Thus:

$$Y = 0.3R + 0.59G + 0.11B \qquad \ldots(3.1)$$

As color pictures require three signals, it should be possible to send Y and two other signals that a color display could arithmetically convert back to R, G and B. There are two important factors that restrict the form, which the other two signals may take. One is to achieve reverse compatibility. If the source is a monochrome camera, it can only produce Y and the other two signals will be completely absent. A color display should be able to operate on the *Y* signal only and shows a monochrome picture. The other is the requirement to conserve bandwidth for economic reasons [Watkinson , J.,1997].

There are several color representations in addition to RGB color system such as (YC_bC_r, YUV, YIQ, CIELAB etc). It is common practice to convert one color representation to another color representation.

The Color YC_bC_r representation is used for most video coding standards in compliance with the CCIR601 (Committee Consultative International Radiocommication).

The *Y* component specifies the luminance information and the C_b and C_r components specify the color information [Yun, Q.,and Huifang, S., 2000].

The luminance component provides a grayscale version of the image, while the two-chrominance components give additional information to converts a grayscale image to a color image.

And, the forward and inverse transform between RGB and YC_bC_r could be given as follows.

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.439 \\ 0.439 & -0.368 & -0.071 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \quad \text{.....(a)}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.164 & 0.000 & 1.596 \\ 1.164 & -0.393 & -0.813 \\ 1.164 & 2.017 & 0.000 \end{bmatrix} \begin{bmatrix} Y-16 \\ C_b-128 \\ C_r-128 \end{bmatrix} \quad \text{.....(b)}$$

: The Color systems RGB and YC_bC_r, (a) represent image converting from RGB to YC_bC_r. And (b) represents reverse conversion from YC_bC_r to RGB.

Finally, the use of different color systems are essential for compatibility in both directions between color and monochrome, but it has a further advantage which follows from the way in which the eye works. There is evidence that the nervous system uses some form of color difference processing to make this possible. As a result the acuity of human eye is only available in monochrome. Differences in color cannot be resolved so well. A further factor is that the lens in the human eye is not achromatic and this means that the ends of the spectrum are not well focused. This is particularly noticeable on blue [Michael, O., and Peter, S., 1998].

3.3 Video Fundamentals

As we mentioned earlier, digital video represents a collection of images or frame that are displayed in order at a specific time. These frames are correlated to achieve main aspect of the digital video representation. There are several kinds of video fundamentals that are impact on video representation. These fundamentals are described bellow.

3.3.1 Sampling of Chrominance and Luminance Values

In digital video, for each pixel there is color information in the form of color component values (e.g., for Y, C_r, and C_b), which are defined. However, for some applications, like TV broadcasting, the color information for each pixel can be less accurate than the luminance information. In this case, it is possible to assign color information, for example, only to every second pixel. This method is described by the "colon notation," as for instance, 4:2:2. This notation basically describes the relations between the number of luminance and chrominance samples taken while digitizing video pictures on [Michael, O., and Peter, S., 1998]:

- **4:4:4 Sampling Ratio:** In this case, luminance (intensity) and chrominance (color) information and are present for every pixel. Figure (3.2) shows rows of pixels with Y,C_r, and C_b information assigned for each pixel[Michael, O., and Peter, S., 1998].

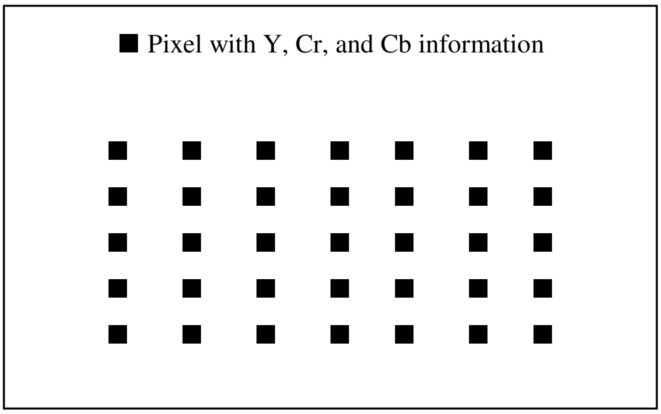


Figure (3.2): Y, C_r, and C_b distribution for 4:4:4

- **4:2:2 Sampling Ratio:** In this case, luminance (intensity) is present for every pixel and chrominance (color) information is present for every second pixel in the horizontal direction. This example is shown in figure (3.3) [Michael, O., and Peter, S., 1998].

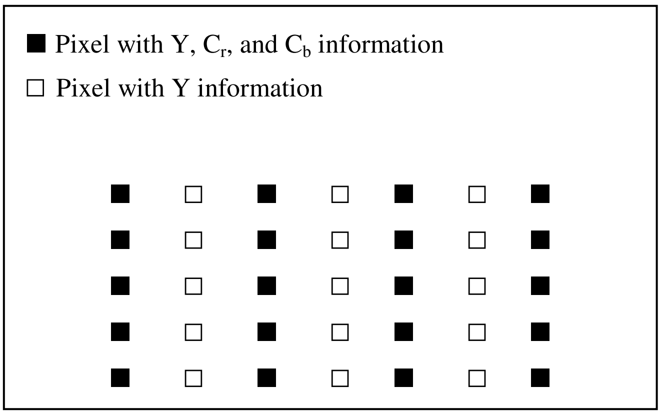


Figure (3.3): Y, C_r, and C_b distribution for 4:2:2

- **4:2:0 and 4:1:1 Sampling Ratio:** The 4:2:0 and 4:1:1, formats further reduce the number of chrominance samples. For 4:2:0 and 4:1:1 chrominance information is only available for every fourth pixel. The 4:2:0 format is a special case of 4:1:1, where the chrominance values are calculated and therefor represent a value that is offset from luminance samples. Figure (3.4) shows the color and intensity distribution for 4:2:0[Michael, O., and Peter, S., 1998].

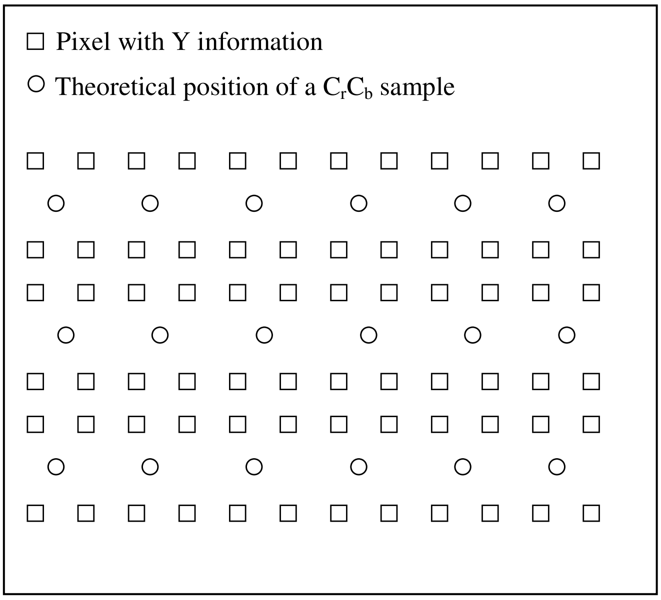


Figure (3.4): Y, C_r, and C_b distribution for 4:2:0

Spatial resolution is usually the same for all components of the RGB representation, while in the luminance-chrominance systems the chrominance components are mostly subsampled with respect to the luminance components. The 4:2:2 sampling scheme is used in high-fidelity application like contribution-quality television while 4:2:0 is very common in high-compression applications [Sangwine, S. J., and Horne, R. E. N., 1998].

3.3.2 Digital Video Formats

If video is represented digitally, there are quite a few options regarding the horizontal and vertical resolution of the picture, the number of pictures per second, and the number of bits used to store the color information. If different applications should handle digital video material, it is necessary that all of these applications have a common understanding of the above mentioned parameters. The most important digital video formats are therefore described below [Michael, O., and Peter, S., 1998].

3.3.2.1 Source Input Format (SIF) and Common Interchange Format (CIF)

The Source Input Format (SIF) and Common Interchange Format (CIF) are digital video format that are defined by the MPEG-1 and the ITU H.261 Videoconferencing Recommendation. The SIF format specifies the luminance resolution of a frame to be 360 × 242 pixels for 30 frames/per second systems. For 25 frames/per second system, SIF define a luminance resolution of 360 x 288 pixels. Table (3.1) summarizes the parameters for SIF and CIF [Michael, O., and Peter, S., 1998].

Table (3.1): SIF and CIF parameters

Horizontal/Vertical Resolutions	SIF (30 frames per second)	SIF (25 frames per second)	CIF(30 frames per second)
Y	360 x 242	360 x 288	352 x 288
C_r	180 x 121	180 x 144	176 x 144
C_b	180 x 121	180 x 144	176 x 144
Sampling Formats	4:2:0	4:2:0	4:2:0

In Addition, Table (3.2) illustrates several extended video formats that are related to H.261 and H.263 video codec [Keith, J., 2001].

Table (3.2): Picture Format for H.261 and H.263 Video Codecs

Picture Format	Luminance Pixels	Max Frame Rate [F/S]	Video Source Rate	Average Coded Bit Rate	H.261 CODEC	H.263 CODEC
SQCIF	128 x 96	30	1.3 Mb/s	26 Kb/s	Optional	Required
QCIF	176 x 144	30	9 Mb/s	64 Kb/s	Required	Required
4CIF	704 x 576	30	438 Mb/s	3-6 Mb/s	Not defined	Optional
16CIF	1408 x 1152	50	2.9 Gb/s	20-60 Mb/s	Not defined	Optional

3.3.2.2 Video Information Units

When the motion video is represented in digital form, it can be decomposed into a time dependent sequence of individual information units. For example, a motion video sequence can be divided into film, clip, frames, blocks, and pixels, as illustrated in Figure (3.5).

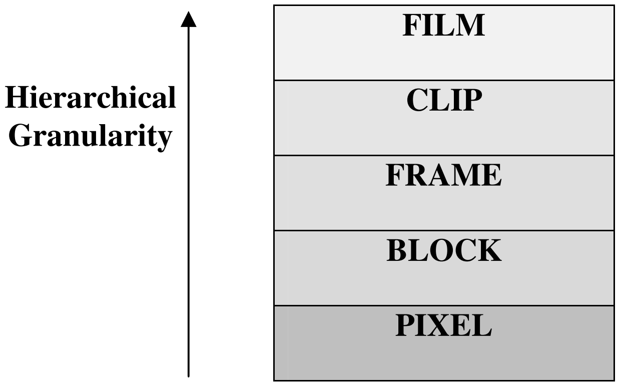


Figure (3.5): Motion video sequence divided into information units

A full motion video, or film, consists of a number of clips, which are characterized with a common thread (for example, a camera shot). Each clip consists of a number of frames. Each frame can be divided into blocks. Typical size of the blocks, which are used in video processing systems (such as compression, retrieval and indexing, motion estimation, etc.) are 8x8 and 16x16 pixels. Pixels are the smallest pieces of information, which consist of 8, 16, or 24 bits [Keith, J., 2001].

3.4 Feasibility of Image and Video Compression

Image and video compression is not only a necessity for the rapid growth of digital visual communications, but it is also feasible. Its feasibility rests with two types of redundancies, i.e., statistical redundancy and psychovisual redundancy. By eliminating these redundancies, we can achieve image and video compression.

3.4.1 Statistical Redundancy

Statistical Redundancy can be classified into two types: Interpixel redundancy and coding redundancy.

3.4.1.1 Interpixel redundancy: means that pixels of an image frame and pixels of a group of successive image or video frames are not statistically independent. On the contrary, they are correlated to various degrees. This type of interpixel correlation is referred to as interpixel redundancy. Interpixel redundancy can be divided into two categories, spatial redundancy and temporal redundancy.

- **Spatial redundancy**: spatial redundancy represents the statistical correlation between pixels within an image frame. Hence it is also called intraframe redundancy.
- **Temporal redundancy:** temporal redundancy is concerned with the statistical correlation between pixels from successive frames in a temporal image or video sequence. Therefore, it is called interframe redundancy. Removing a large amount of temporal redundancy leads to a great deal of data compression.

3.4.1.2 Coding redundancy: coding redundancy refers to the representation of information, i.e., coding itself instead of information redundancy. Coding redundancy occurs when the data used to represent the image are not utilized in an optimal manner.

3.4.2 Psychovisual Redundancy: refers to the fact that some information is more Important to the Human Visual System (HVS) than other types of information. It is known that the HVS perceives the outside world in a rather complicated way. Its response to visual stimuli is not a linear function of the strength of some physical attributes of the stimuli, such as intensity and color. In the HVS, visual information is not perceived equally; some information may be more important than other information. In this sense, we see that some visual information is psychovisually redundant. Eliminating this type of psychovisual redundancy leads to data compression [Yun, Q.,and Huifang, S., 2000].

3.5 Digital Image Compression Techniques

Generally speaking, the classification of image coding techniques can be grouped into two main classes: “still image coding“ and “motion image coding” techniques.

Still image coding exploits the spatial redundancy within images. Motion video coding takes into account temporal as well as spatial redundancy [Martyn, J., and Lain E., 1997].

3.5.1 Still Image Compression

A typical photographic-quality still image contains a large amount of spatial redundancy. The pixel values are often highly correlated. The redundancy can be removed to achieve compression of the image data [Martyn, J., and Lain E., 1997].

Image compression involves reducing the size of image data file, while retaining necessary information. The reduced file is called the "compressed file" and is used to reconstruct the image, resulting in the "decompressed image". The original image, before any compression is performed, is called the "uncompressed" image file [Unbaugh, S. E., 1998].

Removing the redundant information leads to represent the data in a more efficient form. Figure (3.6) shows the general procedure for compression image information.

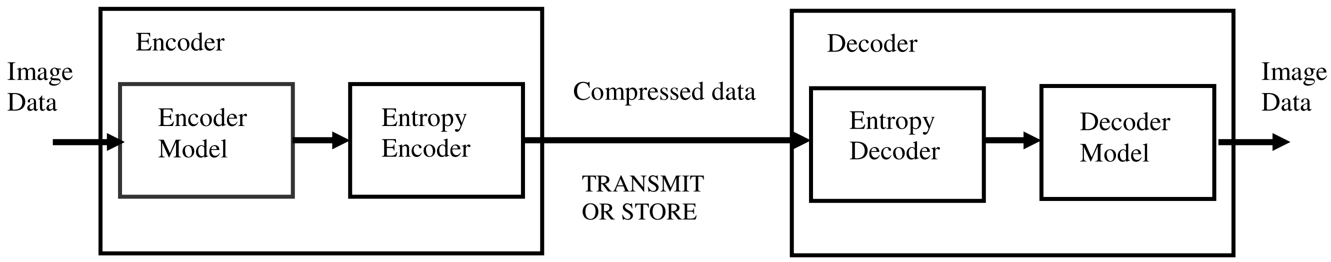


Figure (3.6): Image compression.

The "encoder model" models the image in some way to exploit its statistical properties and to remove redundancy. The encoder produces "symbols" that represent the information in the original image. These symbols are then "entropy encoded" (for example using Huffman encoding) to code them as efficiently as possible. The decoder carries out the reverse procedure to recreate a copy of the original image [Martyn, J., and Lain E., 1997].

The compression ratio is the main factor that describes the correlation between original uncompressed image file and compressed file. The compression ratio is denoted by:

$$\text{Compression Ratio} = \frac{\text{Uncompress File Size}}{\text{Compress File Size}} = \frac{SIZE_U}{SIZE_C} \quad \ldots (3.2)$$

Another way to state the compression is by using the terminology of “bit per pixel”. For and $N \times N$ image:

$$\text{Bits per Pixel} = \frac{\text{Number of Bits}}{\text{Number of Pixels}} = \frac{(8) \times (\text{Number of Byte})}{N \times N} \quad \ldots (3.3)$$

The reduction in file size is necessary to meet the bandwidth requirements for many transmission systems, as well as the storage requirements in computer database. The amount of data required for digital images is enormous. For example, a single 512 $\times$ 512, 8 bit image requires 2.097.152 bits for storage. If we want to transmit this image over the WWW, it would take about [Unbaugh, S. E., 1998]:

$$\frac{(512 \times 512 \text{ pixels})(24 \text{ bits/ pixel})}{(28.8 \times 1024 \text{ bits / second})} \approx 213 \text{ seconds} \approx 3.6 \text{ minutes}$$

Based on the mathematical methods used in digital image compression, still image compression techniques can be classified as “lossy compression techniques” and/or “lossless compression techniques”. A classification tree for digital image compression techniques is shown in Figure (3.7) and Figure (3.8). Lossy compression techniques can compress the image down to 50:1 ratio, where as lossless compression technique can compress the image only up to a ratio of 3:1 [Pennebaker, W.B., and et al, 1993].

Figure (3.7) and (3.8) show the most common classification of data coding techniques.

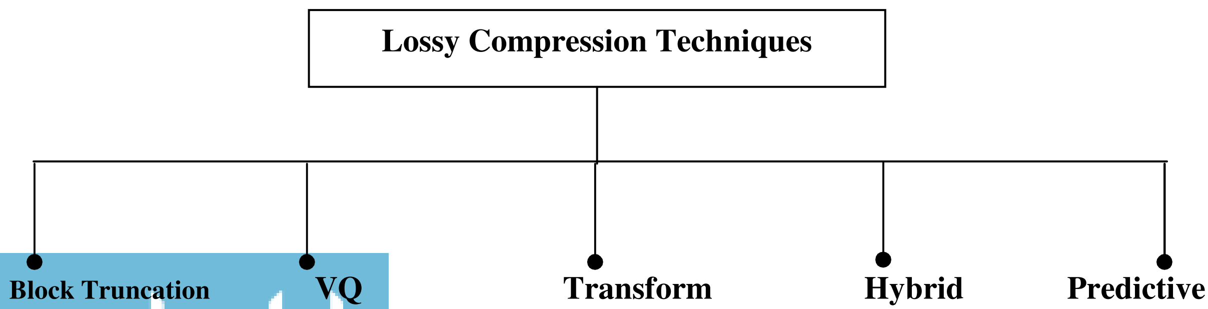


Figure (3.7): Lossy Compression techniques

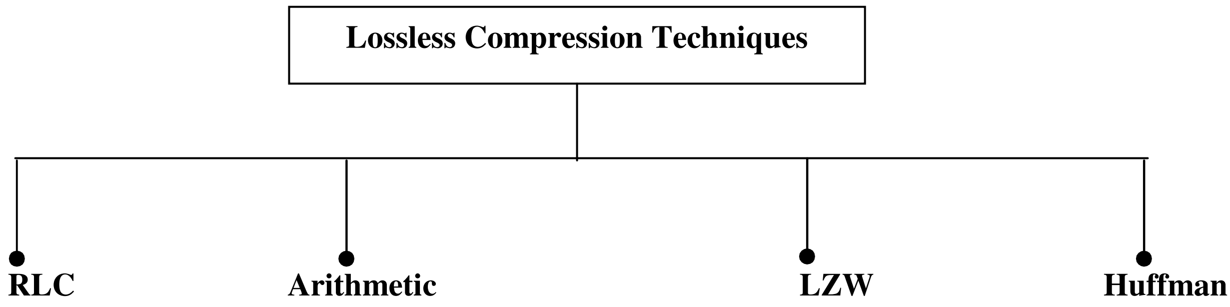


Figure (3.8): Lossless Compression techniques

3.4.1.1 Lossy techniques

Lossy compression schemes involve the loss of some information, and data that have been compressed using a lossy scheme. Generally, in this type of compression, the reconstructed image cannot be recovered exactly as the original. The discarded information is almost undetectable by the human visual system (HVS) and not effected on image clarity.

3.4.1.1.1 Block Truncation Coding

Block truncation coding works by dividing the image into small sub-images and then reducing the number of gray levels within each block. This reduction is performed by a quantizer that adapts to the local image statistics. The levels for the quantizer are chosen to minimize a specified error criterion, and then all the pixel values within each block are mapped to the quantized level. The necessary information to decompress the image is then encoded and stored [Unbaugh, S. E., 1998].

3.4.1.1.2 Vector Quantization

The main idea of vector quantization (VQ) is to quantize vectors formed from groups of image pixels into so called codevectors from a limited set called a codebook. The codebook is usually designed for a class of images. The basic technique is the algorithm proposed by Linde, Buzo and Gray (1980). This algorithm is in fast designed to iteratively improve a given initial codebook. [Sangwine, S. and Horne, R., 1998]

3.4.1.1.3 Transform Coding

Transform coding is a form of block coding done in the transform domain. The image is divided into blocks, or sub-image, and the transform is calculated for each block. Any of previously defined transforms can be used, frequency (e.g. Fourier) or sequence (e.g. Walsh), but it has been determined that the Discrete Cosine Transform (DCT) is the optimal for most image. After the transform has been calculated, the transform coefficients are quantized and coded [Unbaugh, S. E., 1998].

3.4.1.1.3.1 Karhunen-Loeve Transform (KLT)

It is the best (optimum) transform, which satisfies a number of criteria such as minimum mean square truncation error, uncorrected transform coefficients and minimum entropy. Jain (1976) proposed an approach implying some approximations to the fast KLT. However, the KLT has the disadvantage of being very computationally intensive and hence impractical in an image coding system [Jorj, L., 1997].

3.4.1.1.3.2 Discrete Cosine Transform (DCT)

The discrete cosine transform (DCT) is a popular alternative to the (KLT) for image coding [Ahmed, N., and et al, 1973]. For most continuos-tone photographic images, the DCT provides energy compaction that is close to the optimum. A number of fast algorithms exist for calculating the DCT of a block of pixel. Energy compaction has led to the wide spread use of the DCT for image and video compression systems. A fast and more direct DCT approach was reported by Chen et al. in 1977[Chen, W., and et al, 1977].

3.4.1.1.4 Hybrid Coding

This kind of methods uses both the spatial domain and the transform domain. For example, the original image (spatial domain) can be differentially mapped, and then this differential image can be transform coded. Alternately, a one-dimensional transform can be performed on the rows, and this transformed data can undergo differential predictive coding along the columns such methods are often used for compression of analog video signal [Unbaugh, S. E., 1998].

3.4.1.1.4.1 Wavelet Transform

Wavelet-based compression shows much promise for the next generation of image compression methods. Because wavelets localize information in both the spatial and frequency domain, these are included under the hybrid method category. The wavelet transform combined with vector quantization has led to the development of compression algorithms with high compression ratios [Unbaugh, S. E., 1998].

Wavelet transform breaks the signal into a number of wave pulses (Wavelet) that can be dilated and translated in two or more dimensions. Wavelet basis functions are orthonormal. Therefore, these transformations can be used to remove redundant signals from the original signal, this leads to compression of the original signal.

There are two types of Wavelet Transforms: "Continues Wavelet Transform" and "Discrete Wavelet Transform". The Continues Wavelet Transform was first presented by Grossmann and Morlet in 1984. Thereafter it was developed by others, including Holschneider (1988), Arneo'odo et al. (1989) and Forge (1992). Daubechies (1986 and 88) was one of the first to work on Discrete Wavelet Transform [Bevinakoppa, S.,1999].

3.4.1.1.5 Differential Predictive Coding

Differential predictive coding works by predicting the next pixel value based on the previous values and encoding the difference between the predicted value and the actual value (for analog signals, this is also called differential pulse code modulation or DPCM). This technique takes advantage of the fact that adjacent pixels are highly correlated, which means that the difference between adjacent pixel is typically small [Unbaugh, S. E., 1998].

DPCM was first invented and patented in 1952 by Culter [Dasavathy, B., V., 1995].

3.4.1.2 Lossless techniques

Lossless compression techniques are designed to remove the redundant data when data are stored or transmitted and then replace it when the image is reconstructed from those data. The reconstructed image is identical to the original, i.e., all of the information originally present in the image has been preserved by compression.

3.4.1.2.1 Run_Length Coding (RLC)

Run length coding (RLC) is an image compression method that works by counting the number of adjacent pixels with the same gray level value. This count, called the Run Length, is then coded and stored.

Basic RLC is used primarily for binary images, but can work with complex images that have been preprocessed by thresholding to reduce the number of gray levels to two.

The standard for RLC have been defined by International Telecommunication Union Radio (ITU-R) previously Committee Consultative International for Radio Communication (CCIR) [Unbaugh, S. E., 1998].

3.4.1.2.2 Arithmetic Coding

In arithmetic coding there is no direct correspondence between the code and the individual pixel values. Arithmetic coding transforms input data into a single floating point number between 0 and 1. As each input symbol (in this case, pixel value) is read, the precision required for this number becomes greater. Because images are very large and the precision of digital computers is finite, an entire image must be divided into small subimages to be encoded [Unbaugh, S. E., 1998].

3.4.1.2.3 Lemple - Ziv Walch Coding (LZW)

Lemple - Ziv (LZ) coding is a lossless technique first described by Lemple and Ziv (1977). It was extended by Walch (1984) to form the widely used LZW algorithm (Whitaker , 1998).

LZW coding algorithm works by coding strings of data. For images these strings of data correspond to sequences of pixels value. It works by creating a string table that contains the strings and their corresponding codes. The string table is updated with new codes whenever a new string from the file is encountered. If a string is encountered that is already in the table, the corresponding code for that string is put into the compressed file [Unbaugh, S. E., 1998].

3.4.1.2.4 Huffman Coding

The Huffman code, developed by D.Huffman in 1952, is a minimum length code. Thus when the statistical distribution of the gray levels (the histogram) is given, the Huffman algorithm will generate a code that is as close as possible to order entropy (minimum bound) of the distribution.

This method results in a variable length code, where the codewords are of unequal length. For complex images, Huffman coding alone will typically reduce the file by 10 to 50% (1.1:1 to 1.5:1), but this ratio can be improved to 2:1or 3:1 by preprocessing for the irrelevant information removal [Unbaugh, S. E., 1998].

3.4.2 Video Coding (Motion images compression)

Many different techniques for compressing (coding) digital video have been developed in recent years. All of these techniques exploit the inherent spatial and temporal redundancy of a video sequence in order to achieve compression [Martyn J., and Lain E., 1997].

Video signals exist in four dimensions: the magnitude of the sample, the horizontal and vertical spatial axes, and the time axis. Compression can be applied in any or all of these four dimensions. Video compression is generally divided into two basic categories. First, when individual pictures are compressed without reference to any other pictures, the time axis does not enter the process, which is therefor describe as "intra-coded" (intra = within) compression. It is an advantage of intra-coded video that there is no restriction to the editing which can be carried out on the picture sequence. As intra-coding treats each picture independently, it can employ certain techniques developed for the compression of still pictures, like (JPEG technique). Second, greater compression factors can be obtained by taking account of the redundancy from one picture to the next. This involves the time axis and the process is known as "inter-coded" (inter = between) compression [Watkinson , J.,1997].

The compression ratios of digital video compression techniques vary according to the subjective acceptable level of error. Table (3.3) summarizes video compression techniques, their compression ratios and their characteristics [Furth, B., 2000].

Table (3.3): Overview of Video Compression Techniques

COMPRESSION TECHNIQUE	TYPICAL COMPRESSION RATIO	CHARACTERISTICS
Intel RTV/Indeo	3:1	A 128x240 data stream is interpolated to 250x240. Color is subsampled 4:1. A simple 16bit codebook is used without error correction. Frame differencing is used.
Intel PLV	12:1	A native 256x240 stream is encoded using vector quantization and motion compensation. Compression requires specialized equipment.
IBM Photomotion	3:1	An optimal 8-bit color palette is determined, and run-length encoding and frame differencing are used.
Motion JPEG	10:1	Uses 2-D DCT to encode individual frames. Gives good real-time results with inexpensive but special-purpose equipment. This technique supports random-access since no frame differencing is used

Fractals	10:1	Fractals compress natural scenes well, but require tremendous computing power.
Wavelets	20:1	2-D and 3-D wavelets have been used in the compression of motion video. Wavelet compression is low enough in complexity to compress entire images, and therefore does not suffer from the boundary artifacts seen in DCT-based techniques.
H.261/H.263	50:1	Real-time compression and decompression algorithm for video telecommunications. It is based on 2-D DCT with simple motion estimation between frames.
MPEG	30:1	Uses 2-D DCT with motion estimation and interpolation between frames. The MPEG standard is difficult and expensive to compress, but plays back in real time with inexpensive equipment.

The following sections are concentrating on the techniques adopted within the international standards for video coding.

3.4.2.1 H.261: Motion Video Coding for Videoconferencing

The H.261 video coding standard was developed by International Telecommunication Union (ITU) during 1988 to 1993. It was designed for two-way video communication over Integrated Service Digital Network (ISDN). The H.261 refereed also to as the Px64 standard because it encodes the digital video signals at the bitrates of Px64 Kilo bit per second (Kbps), where P is an integer from 1 to 30 [Yun Q. and Huifang S., 2000,].

The coding algorithm is a hybrid of inter-picture prediction, transform coding, and motion compensation.

3.4.2.2 H.263: Low Bit Rate Video Coding

The H.263 video codec standard was specified in 1996 by the International Telecommunication Union (ITU) standard. H.263 was designed for very low bit rate coding application, such as particular video telecommunication. It is largely based on H.261, with a number of improvements that can provide higher quality video at low bit rates.

H.263 improvements include motion compensation with half-pixel accuracy and bidirectionally-coded macroblocks. 8x8 overlapped block motion compensation, unrestricted motion vector range at picture boundary, and arithmetic coding are also used in H.263 [Martyn J., and Lain E., 1997].

3.4.2.3 MPEG: Motion Video Coding for Entertainment and Broadcast

The Moving Picture Experts Group (MPEG) is part of the International Standards Organization working group ISO-IEC/JTC1/SC2/WG11 [Martyn J., and Lain E., 1997].

The MPEG committee developed many international standards, such as MPEG-1, MPEG-2, MPEG-4 and MPEG-7. The following sections illustrate these formats.

3.4.2.3.1 MPEG-1

MPEG-1 video compression standard is a layered, DCT-based video compression standard that results in VHS quality compression video stream.

The MPEG-1 supports coding of video and associated audio at a bit rate of about 1.5 Mbps. The bit rate of a coded video stream together with its associated audio matches the data transfer rate of about 1.4 Mbps provided by a CD-ROM system [Martyn J., and Lain E., 1997].

The audio portion of MPEG-1 is divided into three layers. Each layer provides successively better quality at the cost of a more complex implementation [Kientzle, T., 1998].

3.4.2.3.2 MPEG-2

It was quickly established that MPEG-1 would not be ideal for coding of video and audio information for television applications. "Television quality" video information produces a higher encoded bit rate that requires different coding techniques than those provided by MPEG-1. MPEG-2 extends the functions provided by MPEG-1 to enable efficient encoding of video and associated audio at a wide range of resolutions and bit rates [Martyn J., and Lain E., 1997].

Video sequence layers are similar to MPEG-1 the only improvements are field/frame motion compensation and DCT processing, scalability. Macroblocks in MPEG-2 has 2 additional chrominance blocks when 4:2:2 input format is used.

3.4.2.3.3 MPEG-4

The goal of the MPEG-4 standard is to provide the core technology that allows efficient content-based storage, transmission, and manipulation of video, graphics, audio, and other data with a multimedia environment [Yun Q. and Huifang S., 2000].

Visual part of the MPEG-4 standard describes methods for compression of images and video, compression of textures for texture mapping of 2-D and 3-D meshes, compression of implicit 2-D meshes, compression of time-varying geometry streams that animate meshes. Also, the MPEG-4 supports coding of video objects with spatial and temporal scalability [Ali, S., 1999].

3.4.2.3.4 MPEG-7

The aim of MPEG-7 is to specify a set of descriptors to describe various forms of multimedia. It will also standardize ways to define other descriptors as well as structures for the descriptors and their relationship. This information will be associated with the content to allow fast and efficient search. MPEG-7 will also standardize a language to specify description schemes [José, M.,1999].

3.5 Visual Quality Assessment

The quality of Image and video is an important factor in dealing with image and video compression. For instance, in evaluating two different compression methods we have to base the evaluation on some definite image and video quality. When both methods achieve the same quality of reconstructed image and video, the one that requires less data is considered to be superior to the other. There are two types of quality assessment. One is "objective assessment" (using electrical measurements), and the other is "subjective assessment" (using human observers) [Yun Q. and Huifang S., 2000].

3.5.1 Objective Quality Measurement

This kind of criteria was borrowed from digital signal processing and information theory provides equations that can be used to measure the amount of error in the reconstructed (decompressed) image. The objective criteria, although widely used, are not necessarily correlated with our perception of image quality. However, they are useful as a relative measure in comparing different versions of the same image.

Commonly used objective measures are the root-mean-square error e_{RMS}, the root-mean-square signal-to noise ratio SNR_{RMS}, and the peak signal-to noise ratio $SNR_{peak.}$ The error between the original (uncompressed) pixel value and the reconstructed (decompressed) pixel value can be defined as in equation (3.8) [Unbaugh, S. E., 1998]:

$$error(r,c) = I(r,c) - \hat{I}(r,c) \quad ,\ldots\ldots\ldots(3.8)$$

Where I(r,c) = the original image,

And $\hat{I}(r,c)$ = the decompressed image,

Next, we can define the total error in an N × M decompressed image as in equation (3.9):

$$\text{Total error} = \sum_{r=0}^{n-1}\sum_{c=0}^{m-1}\left| I(r,c) - \hat{I}(r,c) \right| \quad ,\ldots\ldots\ldots(3.9)$$

Where M and N are the dimensions of the image in the horizontal and vertical directions.

The "root-mean-square error" is found by taking the square root of the error squared divided by the total number of pixels in the image, as in equation (3.10)

$$e_{RMS} = \sqrt{\frac{1}{NxM}\sum_{r=0}^{N-1}\sum_{c=0}^{M-1}[I(r,c) - \hat{I}(r,c)]^2} \quad ,\ldots\ldots\ldots(3.10)$$

The smaller the value of the error metrics, the better the compressed image represents the original image. Alternately, with the signal-to-noise (SNR) metrics, a larger number implies a better image.

The SNR metrics considers the decompressed image Î(r,c) to be the "signal" and the error as the "noise". The "root-mean-square signal-to-noise ratio" can be defined as in equation (3.11):

$$SNR_{RMS} = \sqrt{\frac{\sum_{r=0}^{N-1}\sum_{c=0}^{M-1}\left[\hat{I}(r,c)\right]^2}{\sum_{r=0}^{N-1}\sum_{c=0}^{M-1}\left[\hat{I}(r,c)-I(r,c)\right]^2}} \quad ,\ldots\ldots\ldots(3.11)$$

Another related metric the peak signal-to-noise ratio (PSNR) is defined as in equation (3.12):

$$SNR_{PEAK} = 10\log_{10}\frac{(L-1)^2}{\frac{1}{NxM}\sum_{r=0}^{N-1}\sum_{c=0}^{M-1}[\hat{I}(r,c)-I(r,c)]^2} \quad ,\ldots\ldots\ldots(3.12)$$

Where L= the number of gray levels (e.g., for 8 bits L=256).

On one hand, we see that objective quality measurement does not always provide reliable picture quality assessment. On the other hand, however, its implementation is much faster and easier than that of the subjective quality measurement [Unbaugh, S. E., 1998].

3.5.2 Subjective Quality Measurement

It is natural that the visual quality of reconstructed image and video frames to be judged by human viewers if they are to be the ultimate receivers of the data. Therefore, the subjective visual quality measure plays an important role in visual communications [Yun Q. and Huifang S., 2000].

Subjective quality criteria require the definition of a qualitative scale to assess image quality. This scale then can be used by human test subjects to determine image quality.

Subjective quality measures can be classified into three categories. The first type are referred to as "impairment tests", where the test subject scores the images in terms of how bad they are. The second type are "quality tests", where the test subjects rate on the images in terms of how good they are. The third type are called "comparison tests", where the images are evaluated on a side-by-side basis [Unbaugh, S. E., 1998].

4. VIDEO STANDARDS: PRESENTATION AND COMPRESSION

4.1 Introduction

This chapter presents an overview of video compression techniques. Two standards and related techniques are described – Px64 Kbps (or H.261/H.263) standard for video-based communications and MPEG standard for intensive applications of full-motion video. Both standards use the combination of DCT-based intraframe compression and predictive interframe coding based on motion vector estimation. Moreover, JPEG standard is described as intraframe compression technique. The techniques of motion vector estimation (Block-matching techniques) are analyzed as well.

4.2 JPEG Standard: Sequential DCT-Based coding algorithm

The sequential DCT-Based coding algorithm is the baseline algorithm of the JPEG coding standard. The block diagram of the JPEG sequential encoder is shown in Figure (4.1).

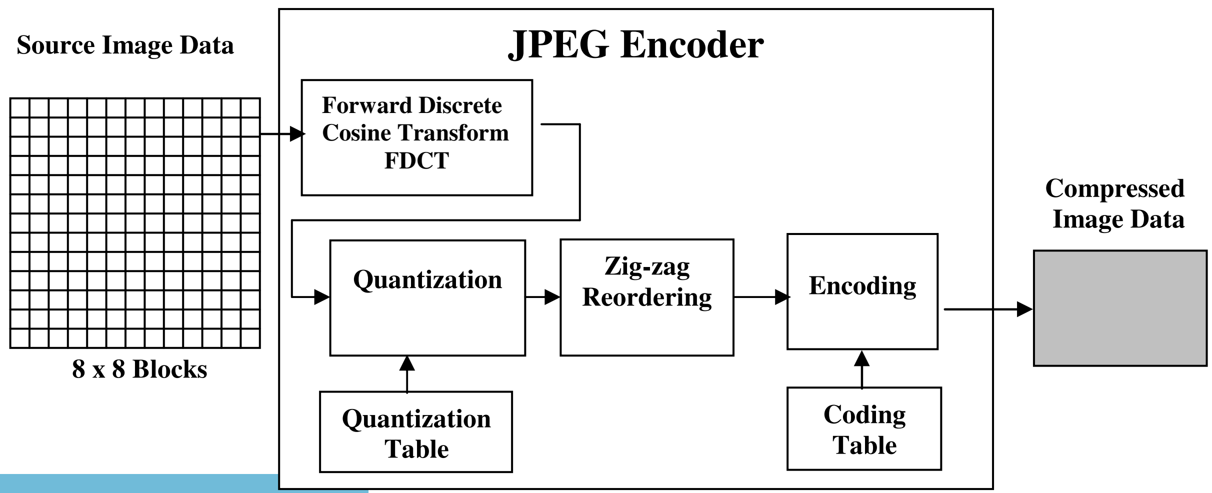


Figure (4.1): JPEG Encoder

As shown in Figure (4.1), the digitized image data are first partitioned into blocks of 8 x 8 samples (Pixels) [Yun, Q.,and Huifang, S., 2000]. Several encoding steps are then applied to each 8 x 8 blocks to accomplish the encoding process. The JPEG encoder arranges these steps in sequential mode.

4.2.1 JPEG encoder

- **DCT step**: After the block partition process is accomplished, each of the partitioned blocks is transformed by the Forward Discrete Cosine Transform (FDCT) into a set of DCT coefficients. An input sample is a number from the range 0 to 255, and it represents a shade of gray on the gray scale. However, JPEG uses a Zero-shift of the input samples so that the range [0,255] is shifted to the range [-128,+127]. (Generally, the range $[0,2^{P}-1]$ of P-bit numbers is shifted to the range $[-2^{P-1},+2^{P-1}-1]$, where P depends on the mode of operation.) Through this shift, the precision requirements in the calculation of DCT are reduced [Dorzdek, 2002].

The two-dimensional Forward DCT of 8 x 8 block is defined as in equation (4.1).

$$F(u,v)=\frac{C(u)}{2}\bullet\frac{C(v)}{2}\sum_{x=0}^{7}\sum_{y=0}^{7}f(x,y)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16} \qquad \ldots(4.1)$$

$$C_uC_v=\begin{cases}\frac{1}{\sqrt{2}} & \textit{For } u,v=0\\ 1 & \textit{Otherwise}\end{cases}$$

Where *f(x,y)* is the value of the pixel at position *(x,y)* in the block, and *F(u,v)* is the transformed *(u,v)* DCT coefficient.

The *F(0,0)* coefficient is called the “DC coefficient,” and the remaining 63 coefficients are called the “AC coefficients.”

- **Quantization step**: After the Forward DCT, quantization of the DCT coefficients is performed. Quantization in JPEG is the process of scaling each DCT coefficient by dividing it by a corresponding quantization value from the quantization table used by the quantizer. The quantized DCT coefficients are then rounded to the nearest integer, as in equation (4.2).

$$F_q(u,v) = Round\left\lfloor \frac{F(u,v)}{Q(u,v)} \right\rfloor \qquad \ldots(4.2)$$

In effect, F_q (u,v) is the normalization of F(u,v) by the quantizer step size Q(u,v) taken from a quantization table Q. In this way, some information is discarded whereby some compression is already accomplished-but this information is not crucial for adequate restoration of encoded image.

There are four quantization tables that may be used by the encoder, but there is no default quantization table specified by the standard. Two particular quantization tables are shown in Table (4.1).

Table (4.1): Two Examples of Quantization Tables (Q) Used by JPEG

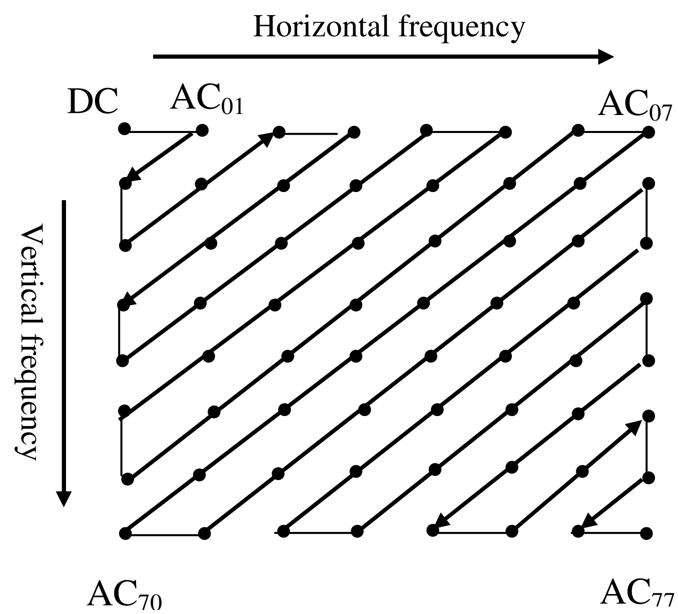
Luminance quantization table

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Chrominance quantization table

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

• **Zig-zag reordering Step**: After quantization, the 63 AC coefficients are ordered into the "Zig-zag" sequence and inserted into one-dimensional array ZZ, as shown in Figure (4.2). The zig-zag ordering will help to facilitate the next step, entropy encoding, by placing low-frequency coefficients, which are more likely to be nonzero, before high-frequency coefficients.



	DC	AC	AC	…	AC	AC	AC
Zig-zag Array (ZZ)	**ZZ[0]**	**ZZ[1]**	**ZZ[2]**	**…**	**ZZ[61]**	**ZZ[62]**	**ZZ[63]**
Quantized Array of DCT Coefficients (F_q)	**F_q [0,0]**	**F_q [0,1]**	**F_q [1,0]**	**…**	**F_q [6,7]**	**F_q [7,6]**	**F_q [7,7]**

Figure (4.2): Zig-zag ordering of AC coefficients.

• **Encoding Step**: This step consists of two sub-steps. First, The DC coefficient, which represent the average values of the 64 image samples, are coded using the predictive coding techniques, as illustrated in Figure (4.3). The reason for predictive coding of DC coefficient is that there is usually a strong correlation between the DC coefficients of adjacent 8x8 blocks. As a consequence, the compression ratio will be improved.

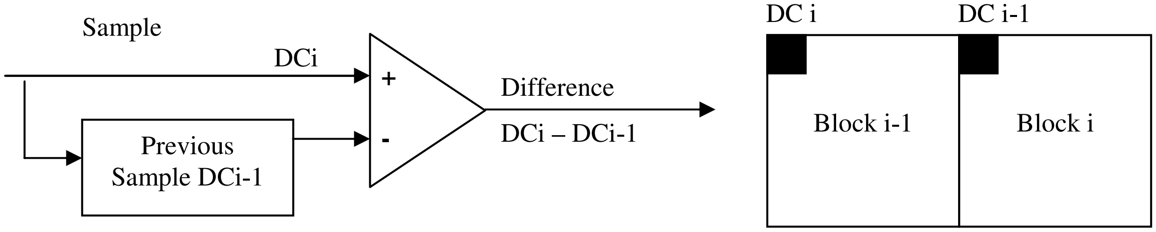


Figure (4.3): Predictive coding for DC coefficients

Finally, the entropy encoding is achieved, which provides additional compression by encoding the quantized DCT coefficients into more compact form. The JPEG standard specifies two entropy coding methods: Huffman coding and arithmetic coding. The baseline sequential JPEG encoder uses Huffman coding.

The Huffman coder converts the DCT coefficients after quantization into a compact binary sequence using two steps:

(1) Forming intermediate symbol sequence, and

(2) Converting intermediate symbol into binary sequence using Huffman tables.

In the intermediate symbol sequence, a pair of symbols represents each AC coefficient:

Symbol –1	Symbol-2
(RUNLENGTH, SIZE)	**(AMPLITUDE)**

Where:

RUNLENGTH is number of consecutive zero-lined AC coefficients preceding the nonzero AC coefficient. The value of RUNLENGTH is in the range 0 to 15, which requires 4 bits for its representation.

SIZE is the number of bits used to encode AMPLITUDE. The number of bits for AMPLITUDE is in the range of 0 to 10 bits, so there are 4 bits needed to code SIZE.

AMPLITUDE is the amplitude of the nonzero AC coefficient in the range of [+1024 to −1023], which requires 10 bits for its coding. For example, if the sequence of AC coefficients is

$$\underbrace{0,0,0,0,0,0}_{6},\ 476$$

The symbol representation of the AC coefficient 467 is:

(6,9) (476)

Where: RUNLEGTH = 6, SIZE = 9, AMPLITUDE = 467.

If RUNLEGTH is greater than 15, then Symbol-1 (15,0) is interpreted as the extension symbol with runlength = 16. These can be up to three consecutive (15,0) extensions. In the following example:

(15,0) (15,0) (7,4) (12)

RUNLEGTH is equal to 16+16+7 = 39, SIZE = 4, and AMPLITUDE = 12. The symbol (0,0) means ‘End Of Block’ (EOB) and terminates each 8x8 block.

For DC coefficients, the intermediate symbol representation consists of:

Symbol –1	**Symbol-2**
(SIZE)	**(AMPLITUDE)**

Because DC coefficients are differentially encoded, this range is double the range for AC coefficients, and is [-2048, +2047].

The second step in Huffman coding is converting the intermediate symbol sequence into binary sequence. In this phase, symbols are replaced with variable length codes, beginning with the DC coefficients, and continuing with AC coefficients.

Each Symbol-1 (both for DC and AC coefficients) is encoded with a Variable-Length Code (VLC), obtained from the Huffman table set specified for each image component. Symbol-2 is encoded using Variable-Length Integer (VLI), whose length in bits is given in Table (4.2) [Furth, B., 2000].

Table (4.2): Huffman Coding of Symbols-2

Size	Amplitude range
1	(-1,1)
2	(-3,-2) (2,3)
3	(-7..-4) (4..7)
4	(-15..-8) (8..15)
5	(-31..-16) (16..31)
6	(-63..-32) (32..63)
7	(-127..-64) (64..127)
8	(-255..-128) (128..255)
9	(-511..-256) (256..511)
10	(-1023..-512) (512..1023)

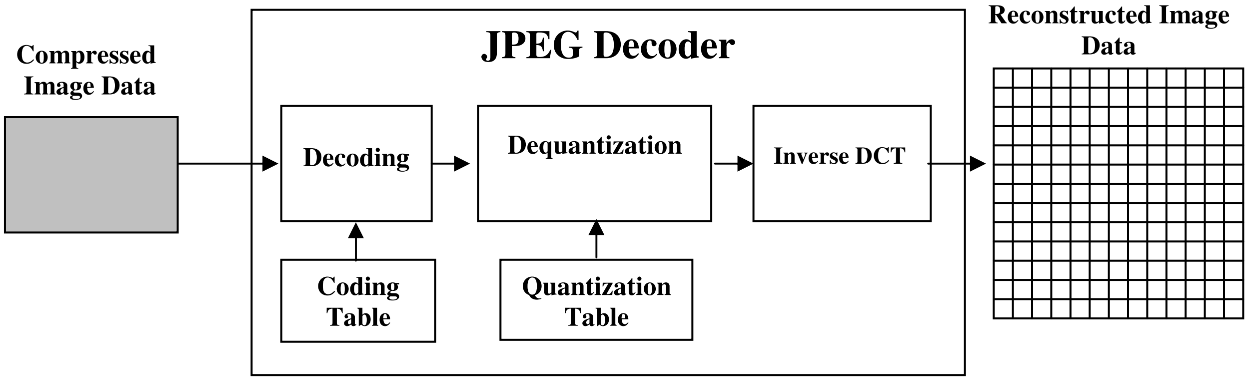


Figure (4.4): JPEG Decoder

4.2.2 JPEG decoder

In the JPEG sequential decoding, all the steps from the encoding process are inversed and implemented in a reverse order, as shown in Figure (4.4).

First, an entropy decoder (such as Huffman) is implemented on the compressed image data. The binary sequence is converted to a symbol sequence using Huffman table (VLC coefficients) and VLI decoding, and then the symbols are converted into DCT coefficients. Then, the dequantization is implemented using function (4.3)

$$F(u,v) = F_q(u,v) \times Q(u,v) \qquad \ldots(4.3)$$

The Inverse Discrete Cosine Transform (IDCT) is then implemented on dequantized coefficients in order to convert the image form frequency domain into spatial domain. The equation (4.4) is an (IDCT) equation.

$$F(x,y) = \frac{1}{4}\left[\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right] \qquad \ldots(4.4)$$

Where

$$C_u C_v = \begin{cases} \frac{1}{\sqrt{2}} & For\ u\ or\ v=0 \\ 1 & For\ u\ or\ v>0 \end{cases}$$

The last step consists of shifting back the decompressed samples in the range $[0, 2^P - 1]$ [Furth, B., 2000].

4.3 The MPEG video coding technique

The MPEG compression algorithm is a full-motion DCT and DPCM hybrid-coding algorithm [Yun, Q., and Huifang, S., 2000].

The compression method uses interframe compression and can achieve compression ratio of 200:1 through storing only the differences between successive frames. The MPEG approach is optimized for motion-intensive video applications, and its specification includes also an algorithm for the compression of audio data at ratios ranging from 5:1 to 10:1 [Furth, B., 2000].

The MPEG algorithm is intended for both asymmetric and symmetric application. Asymmetric applications are characterized by frequently use of the decompression process, while the compression process is performed once. Examples include movieson-demand, video serves, electronic publishing, and education and training. Symmetric applications require equal use of the compression and decompression processes. Examples include multimedia mail, and Videoconferencing [Steinmetz, R., 1994].

4.3.1 MPEG Frame Structure

In MPEG coding, the video sequence is divided into groups of pictures or frames (GOP) as shown in Figure (4.6). Each GOP may include three types of pictures or frames: Intra-coded (I) picture or frame, predictive-coded (P) picture or frame and bidirectionally predictive-coded (B) picture or frame [Yun, Q., and Huifang, S., 2000]. These types are coded using three different algorithms, as illustrated in Figure (4.5).

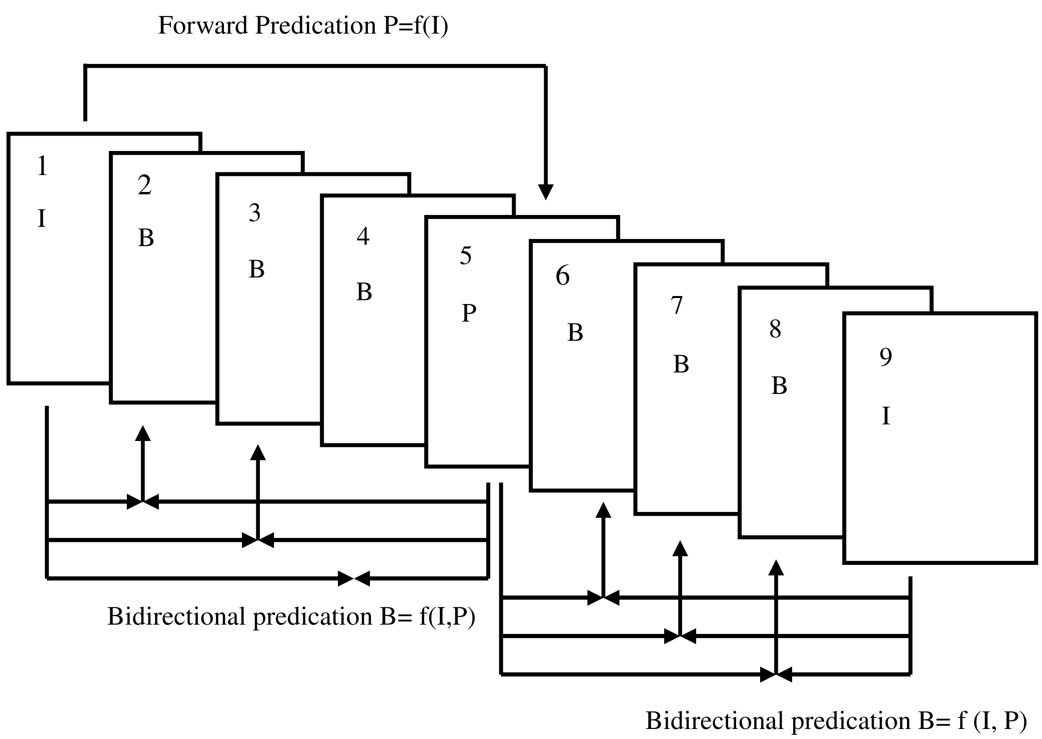


Figure (4.5): Types of frames in the MPEG standard

I-frames are self-contained and coded using a DCT-based techniques similar to JPEG. I-frames are used a random access points to MPEG streams, and they give the lowest compression rations within MPEG. Also, they are used as anchors for forward and/or backward prediction.

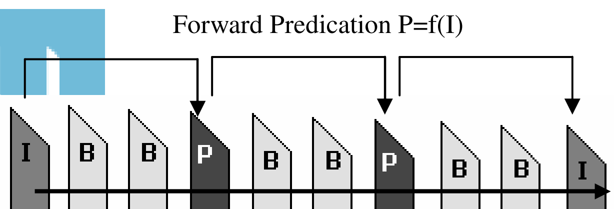
P-frames are coded using forward predictive coding, where the actual frame is coded with reference to a previous frame (I or P). The compression ratio of P-frames is significantly higher than of I-frames.

B-frames are coded using two reference frames, a past and a future frame (which can be I- or P- frames). Bidirectional coding provides the highest amount of compression.

Note that in Figure (4.5), the first three B- frames (2,3 and 4) are bidirectionally coded using the past frame I- (frame 1) and the future frame P- (frame 5). Therefore, the decoding order will differ from the encoding order. The P-frame 5 must be decoded before B-frames 2,3 and 4, and I-frame 9 before B-frames 6,7 and 8. If the MPEG sequence is transmitted over the network, the actual transmission order should be {1,5,2,3,4,9,6,7,8}.

The MPEG application determines a sequence of I-, P-, and B- frames. If there is a need for fast random access, the best resolution would be achieved by coding the whole sequence as I-frames. However, the highest compression ratio can be achieved by incorporating a large number of B-frames. The following sequence has been proven to be very effective for a number of practical application [Steinmetz, R., 1994]:

(I B B P B B P B B) (I B B P B B P B B) …



4.3.2 MPEG Video Encoder and Decoder

The block diagram of the MPEG encoder is given in Figure (4.7), while the MPEG decoder is shown in Figure (4.8).

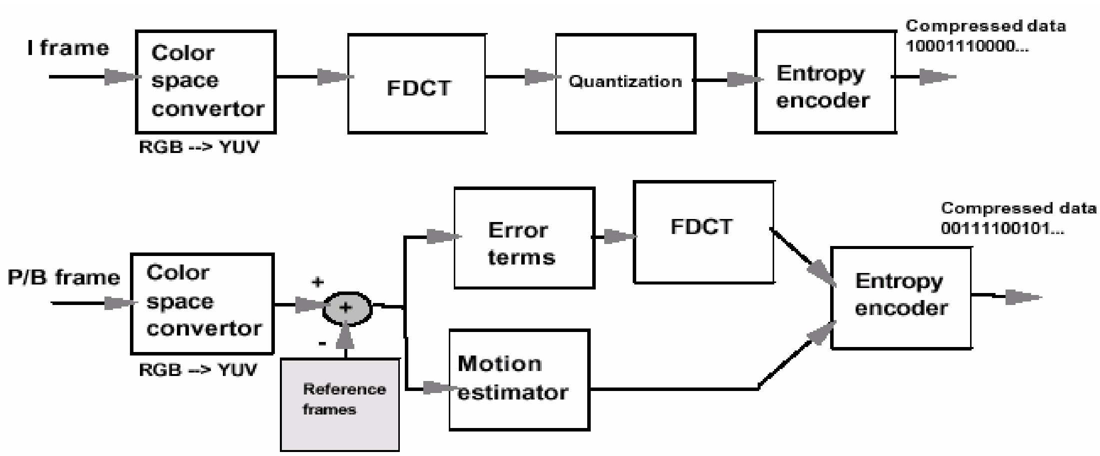


Figure (4.7): The block diagram of the MPEG encoder

I-frames are created similar to JPEG encoded pictures, while P- and B-frames are encoded in terms of previous and future frames. The motion vector is estimated, and the difference between the predicted and actual blocks (error terms) is calculated. The error term is then DCT encoded and the entropy encoder is used to produce the compact code [Furth, B., 2000].

It should be noted that in the encoding order the first frame in a GOP is always an I-picture. In the display order the first frame can be either an I-picture or the first B-picture of the consecutive series of B-pictures which immediately precedes the first I-picture, and the last picture in a GOP is an anchor picture, either an I- or P-picture. The first GOP always starts with an I-picture and, as a consequence, this GOP will have fewer B-pictures than the other GOPs [Yun, Q., and Huifang, S., 2000].

4.3.3 MPEG Data Stream

The MPEG specification defines a “video sequence” composed of a video sequence header and many GOP, as illustrated in Figure (4.9). The video sequence header defines the video format, picture dimensions, aspect ratio, frame rate, and delivered data rate.

As mentioned earlier, a GOP contains pictures that may be encoded into one of three supported compression formats. The GOP header contains a starting time for the group, and can therefor be used as a point of random access. Each frame within the GOP is numbered, and its number coupled with GOP start time and the playback frame rate determines its playback time. Each picture is subdivided into “slices” and then into "macroblocks." A macroblock is composed of four 8x8 blocks of luminance data, and typically two 8x8 blocks of chrominance data, one C_r and one C_b.

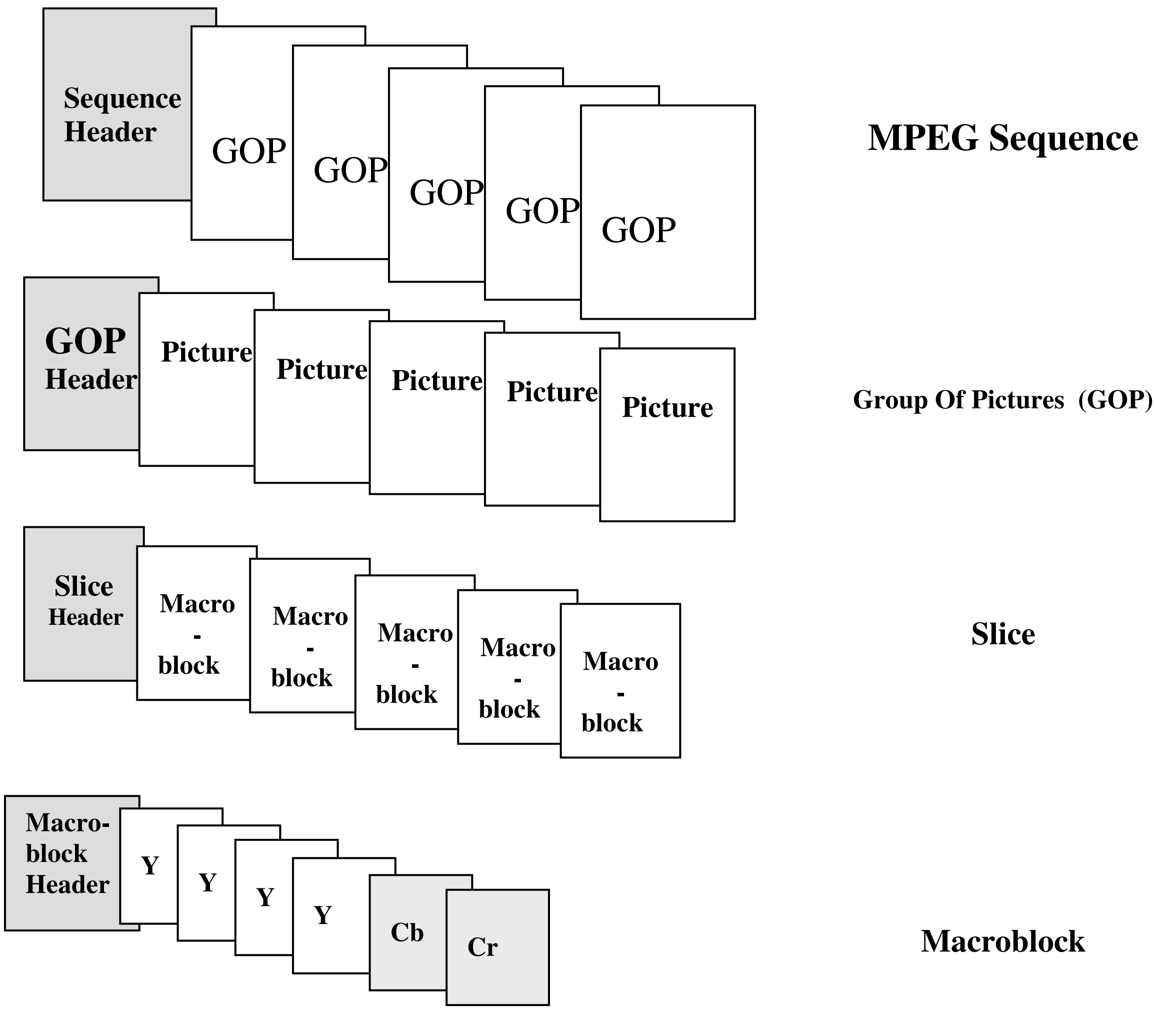


Figure (4.9): MPEG data stream

4.3.3.1 I Picture Format

The I- (Intraframe) picture format substantially corresponds to the JPEG format. These pictures are encoded by transformation into DCT space, quantization of the resultant coefficients, and entropy coding of the result. Transformation into DCT space is performed by an 8x8 DCT, Quantization is performed by reference to a user-loadable quantization table modified by a scale factor. This mechanism supports adaptive quantization at the cost of additional complexity-although 30% improvement in compression is claimed [Pennenbaker, W., and Mitchell, J., 1993].

After quantization, the resulting coefficients are reordered in zigzag order, run-length coded, variable-length coded, and entropy coded. The resulting data stream should roughly show JPEG levels of compression.

4.3.3.2 P Picture Format

The P- (Predicted) picture format introduces the concept of motion compression. Each macroblock is coded with a vector that predicts its value from an earlier I- or P-frame. The decoding process copies the contents of the macroblock-sized data at the address referenced by the vector into the macroblock of the P-frame currently being decoded. Five bits of resolution are reserved for the magnitude of the vector in each of the X and Y directions, meaning that 1024 possible data blocks may be referenced by the predicted macroblock. However, eight possible magnitude ranges may be assigned to those five bits, meaning as many as 8192 macroblocks might have to be evaluated to exhaustively determine the best vector. Each evaluation might require testing as many as 384 pixels, and a further complexity is seen in performing fractional interpolation of pixels (vector motions as small as ½ pixel is supported). Finally, the difference between the prediction and the macroblock to be compressed may be encoded in like fashion to I-frames encoding above.

4.3.3.3 B Picture Format

The B- (Bidirectional prediction) picture format is calculated by two vectors. A backward vector references a macroblock-sized region in the previous I- or P- frame, the forward vector references a macroblock-sized region in the next I- or P- frame. For this reason, I- and P-frames are placed in the coded stream before any B-frames that reference them. The macroblock-sized regions referenced by the motion compensation vectors are averaged to produce the motion estimate for the macroblock being decoded. As with P-frames, the error between the prediction and the frame being encoded is compressed and placed in the bitstream. The error factor is decompressed and added to the prediction to form the B-frame macroblock [Furth, B., 2000].

4.4 Px64 Compression Algorithm for Video Telecommunications

The H.261/263 standard, commonly called Px64 Kbps, is optimized to achieve very high compression ratios for full-color, real-time motion video transmission. The Px64 compression algorithm combines intraframe and interframe coding to provide fast processing for on-the-fly video compression and decompression. Intraframe coding refers to the coding of individual frames, while interframe coding is the coding of a frame in reference to the previous or future frames.

The Px64 standard is optimized for applications such as video-based telecommunications. Because these applications are usually not motion-intensive, the algorithm uses limited motion search and estimation strategies to achieve higher compression ratios. For standard video communication image, compression ratios of 100:1 to over 2000:1 can be achieved.

The Px64 compression standard is intended to cover the entire ISDN channel capacity (p=1,2,…30). For p =1 to 2, due to limited available bandwidth, only desktop face-to-face visual communications (videophone) can be implemented using this compression algorithm. However, for p>6, more complex pictures are transmitted, and the algorithm is suitable for videoconferencing applications [Furth, B., 2000].

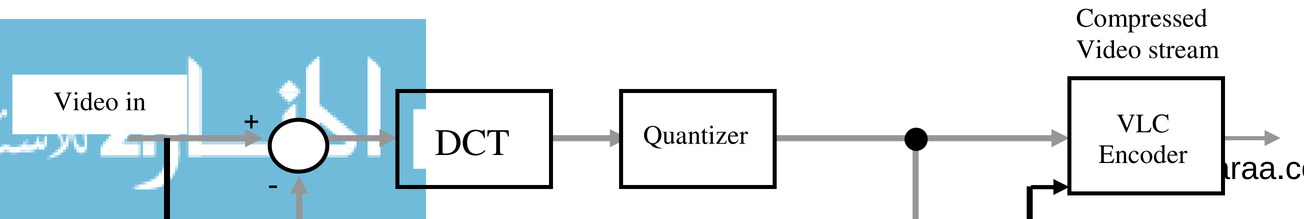
4.4.1 The H.261/H.263 Video Encoder

The coding algorithm of H.263 is similar to that used by H.261, which they combines intraframe and interframe coding to provide fast processing for on-the-fly video. The algorithm creates two types of frames:

- DCT-based intraframe compression, which is similar to JPEG, uses DCT, quantization, and entropy coding.
- Predicative interframe coding based on Differential Pulse Code Modulation (DPCM) and motion estimation.

The block diagram of the video encoder is presented in Figure (4.10).

The H.261/H.263 coding algorithm begins by coding an intraframe block and then sends it to the video multiplex coder. The same frame is then decompressed using the inverse quantizer (IQ) and inverse DCT (IDCT), and then stored in the frame memory (frame store) for interframe coding.



During the intraframe coding, the prediction based on the DPCM algorithm is used to compare every macro block of the actual frame with the available macro blocks of the previous frame, as illustrated in Figure (4.11). To reduce the encoding delay, only the closest previous frame is used for prediction. Then, the difference, created as error terms, is DCT-coded and quantized, and sent to the video multiplex coder with or without the motion vector.

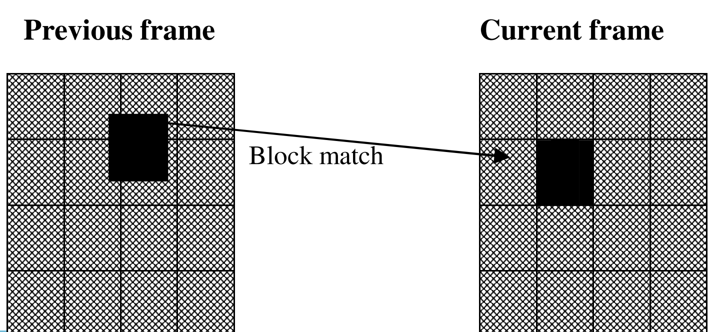


Figure (4.11): The principle of interframe coding in H.261/263 codec

For interframe coding, the frames are encoded using one of the following three techniques:

1-DPCM coding with no motion compensation (zero-motion vectors).

2-DPCM coding with non-zero vectors.

3-Blocks are filtered by an optional predefined filter to remove high-frequency noise.

At the final step, variable-length coding (VLC), such as Huffman encoder, is used to produce more compact code. An optional loop filter can be used to minimize the prediction error by smoothing the pixel in the previous frame. As least one in every 132 frames should be an intraframe [Furth, B., 2000].

4.4.2 The H.261/H.263 Video Decoder

The H.261/H.263 video decoder performs the inverse operations of the encoder. This decoder is shown in Figure (4.12). It consists of the receiver buffer, variable-length decoder (VLD), inverse quantizer (IQ), inverse DCT (IDCT), and the motion compensation, which includes frame memory and an optional loop filter [Liou, M., 1991].

The H.261/H.263 decoder enters the compressed bitstream to the decoder buffer and then is parsed by the VLD. The output of the VLD is applied to the IQ and IDCT where the data are converted to the values in the spatial domain. For the interframe-coding mode, the motion compensation is performed and the data from the macroblocks in the anchor frame are added to the current data to form the reconstructed data [Yun, Q.,and Huifang, S., 2000].

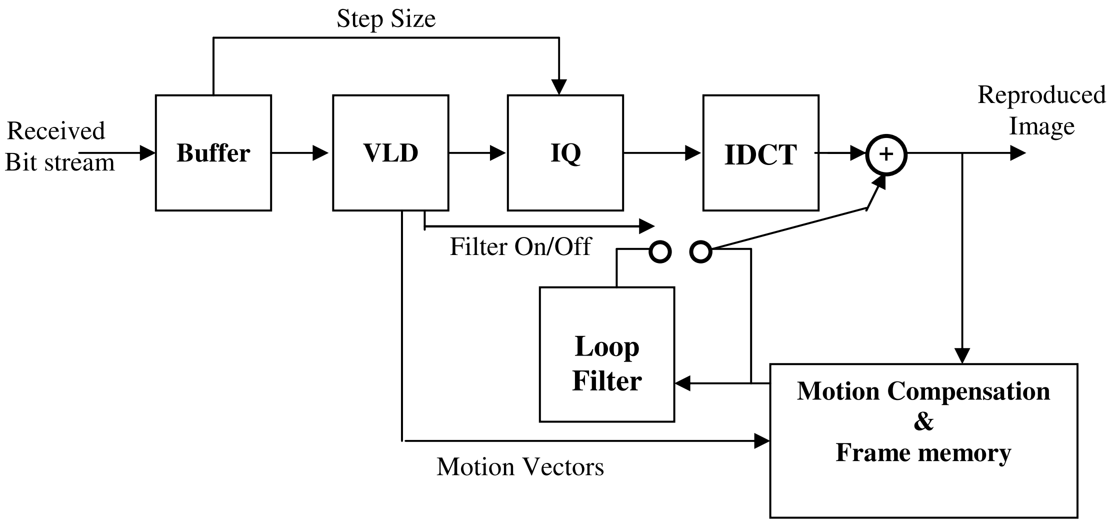


Figure (4.12): The principle of interframe coding in H.261/263 codec

In addition to the encoding and decoding of the video, the audio data must also be compressed and decompressed. Special buffering and multiplexing/demultiplexing circuitry is required to handle the complexities of combining the video and audio [Furth, B., 2000].

4.4.3 Video Data Structure

According to the H.261 standard, a data stream has a hierarchical structure consisting of a Picture, a Group of Block (GOB), Macro Blocks (MB), and Blocks [Aravind, R., and et al, 1993]. A Macro Block is composed of four (8 x 8) luminance (Y) blocks, and two (8 x 8) chrominance (C_r and C_b) blocks, as illustrated in Figure (4.13).

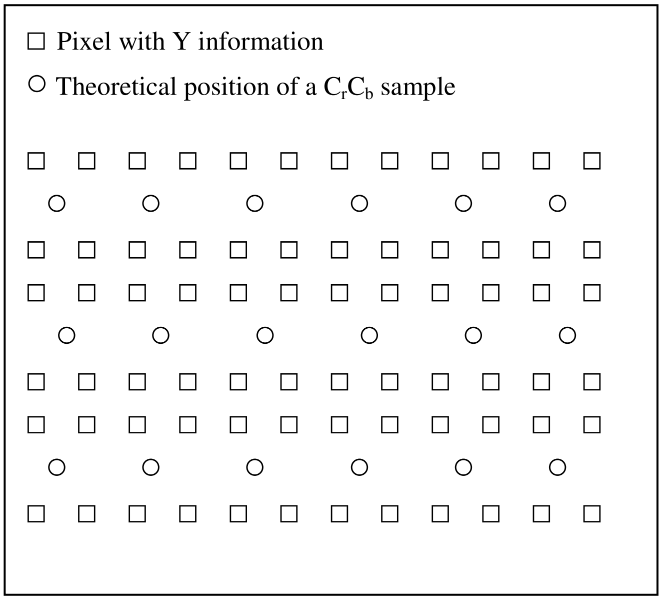
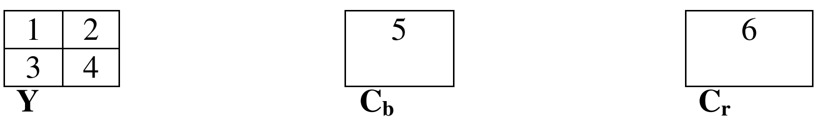


Figure (4.13): The composition of a Macroblock MB = 4Y + C_b + C_r

A Group of Blocks is composed of 3x11 MBs. A CIF Picture contains 12 GOBs, while a QCIF Picture consists of four GOBs. The hierarchical block structure is shown in Figure (4.14).

MB = 4Y + C_b + C_r



GOB = 3x 11 MBs

1	2	3	4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19	20	21	22
23	24	25	26	27	28	29	30	31	32	33

1	2
3	4
5	6
7	8
9	10
11	12

CIF = 12 GOBs

1
3
5

QCIF = 3 GOBs

Figure (4.14): Hierarchical block structure of the P x 64 data stream

Each of the layers contains headers, which carry information about the data that follows [Furth, B., 2000].

4.4.4 H.263 Technical Features Improvements

H.263 is largely based on H.261, with a number of improvements that can provide higher quality video at low-bit rates. These improvements include motion compensation with half-pixel prediction, Arithmetic coding rather than Huffman coding, advanced prediction mode, unrestricted motion vectors, and PB frames mode [Martyn J., and Lain E.,1997].

4.4.4.1 Half-pixel Accuracy

In H.263 video coding, half-pixel accuracy motion composition is used (as opposed to full-pixel prediction in H.261). The half-pixel values are found using bilinear interpolation as shown in Figure (4.15).

Note that H.263 uses sub-pixel accuracy for motion compensation instead of using loop filter to smooth the anchor frames as in H.261 [Yun, Q.,and Huifang, S., 2000].

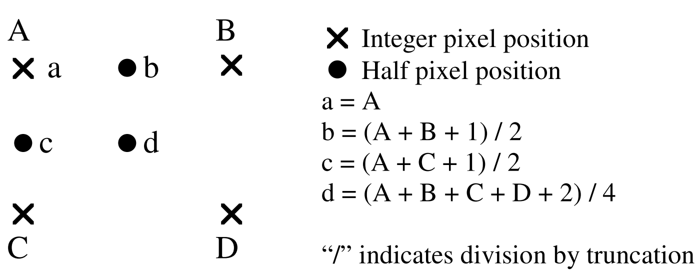


Figure (4.15): Half-pixel predication by bilinear interpolation

4.4.4.2 Unrestricted Motion Vector Mode (Optional)

Part of the macroblock indicated by a motion vector may be outside the picture. The pixel values outside the boundary are constructed by interpolating from the edge pixel, which can reduce the prediction error if movement occurs across the edge of the picture [Martyn J., and Lain E.,1997].

4.4.4.3 Advanced Prediction Mode (Optional)

Four motion vectors, each for one 8x8 luminance block within a macroblock, are used instead of one motion vector for the macroblock. The four vectors require more bits than a single vector. The prediction error is compared with the error for the equivalent single vector and if the error is significantly less, then the four motion vectors are used [Martyn J., and Lain E.,1997].

4.4.4.4 Arithmetic Coding rather than Huffman Coding (Optional)

As in other video-coding standards, H.263 uses VLC and variable-length decoding (VLC/VLD) to remove the redundancy in the video data.

The basic principle of VLC is to encode a symbol with a specific table based on the syntax of the coder. The symbol is mapped to an entry of the table in a table lookup operation, then the binary codeword specified by the entry is sent to a bitstream buffer for transmitting to the decoder. In the decoder, an inverse operation, VLD, is performed to reconstruct the symbol by the table lookup operation based on the same syntax of the coder. The tables in the decoder must be the same as the one used in the encoder for encoding the current symbol. To obtain better performance, the tables are generated in a statistically optimized way (such as a Huffman coder) with a large number of training sequences. This VLC/VLD process implies that each symbol be encoded into a fixed-integral number of bits. An optional feature of H.263 is to use arithmetic coding to remove the restriction of fixed-integral number bits for symbols. This syntax-based arithmetic coding mode may result in bit rate reductions [Yun, Q.,and Huifang, S., 2000].

4.4.4.5 PB Frames Mode (Optional)

The PB frame is a new feature of H.263 video coding. A PB frame consists of two pictures (one P-picture and one B-picture) being coded as one unit, as shown in Figure (4.16) since H.261 dose not have B-picture, the concept of a B-picture comes from the MPEG video coding standards. In a PB-frame, the P-picture is predicted from the previous decoded (I) or P-picture and the B-picture is bidirectionally predicted both from the previous decoded (I) or (P) picture and p-picture in the PB-frame unit which is currently being decoded [Yun, Q.,and Huifang, S., 2000].

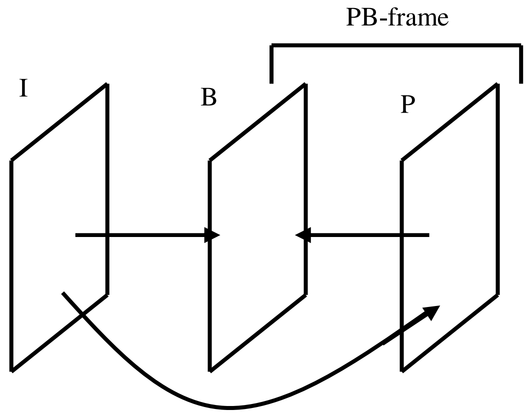


Figure (4.16): Prediction in PB –frames mode.

The use of some or all of these features can lead to a significant improvement in decoded quality over H.261, particularly at low-bit rates.

4.5 Motion Compensation and Motion Estimation

A key feature of most video compression techniques is motion compensation and motion estimation. In motion compensation, it is assumed that areas of current picture are translations of areas of another picture. Compression for a macroblock M_1 being encoded is accomplished by finding in a reference picture a macroblock M_2 that most closely match M_1 and transmitting the displacement of M_2 relative to the position of M_1. For example, a macroblock M_1 of a B-picture in Figure (4.17) is encoded by reference to a macroblock M_2 of a preceding picture and macroblock M_3 of future picture. Because the size of each macroblock is fixed, it is enough to refer to a macroblock location by the position of its upper left corner. If the position of M_1 is (x,y) and the position of M_2 is (x_{m2}, y_{m2}), then the Motion displacement Vector (MV):

$$MV = \text{position}(M_2) - \text{position}(M_1)$$
$$= (\text{horizontal displacement, vertical displacement})$$
$$= (x_{m2} - x,\ y_{m2} - y)$$

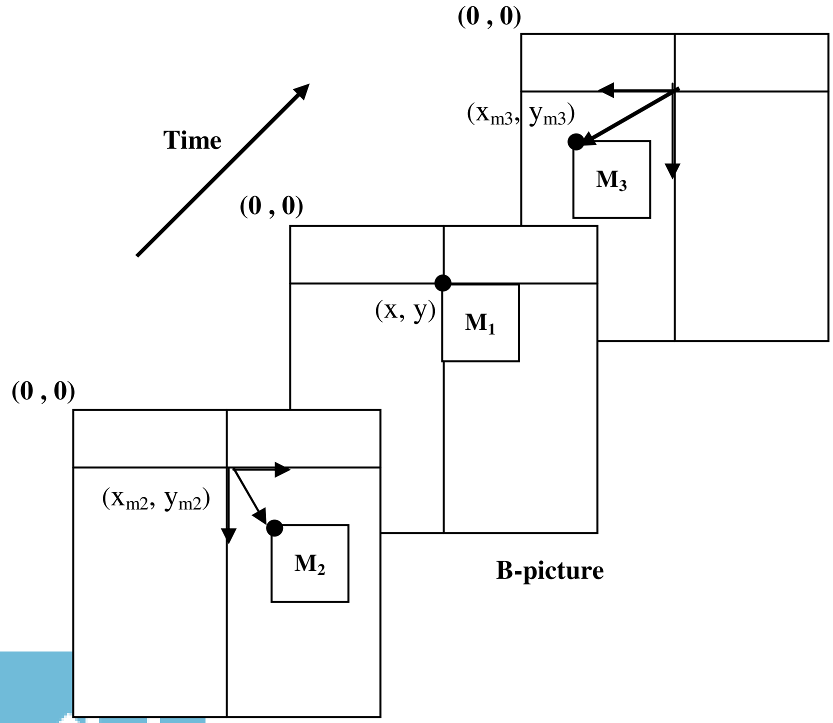


Figure (4.17): Motion displacement for a B-picture

The example in Figure (4.17) illustrates an interesting feature of B-picture than can use both forward and backward prediction for the same macroblock. The formula (4.5) is used to computes the M_1 value of B-picture.

$$M_1[i][j] = round\ (\ (\ M_2[i+dx_{M2}][j+dy_{M2}] + M_3\ [i+dx_{M3}][j+dy_{M3}])\ /\ 2) \qquad \ldots (4.5)$$

Where dx and dy are horizontal and vertical displacement.

For P-picture, only forward reference is used. If the current picture in Figure (4.17) is a P-picture, not a B-picture, then only the forward reference is taken into account. The formula (4.6) is used to find values of pixels in M_1.

$$M_1[i][j] = M_2[i+dx_{M2}][j+dy_{M2}] \qquad \ldots (4.6)$$

The process of finding and extracting the motion displacement vectors from the video sequence is called motion estimation. There are many techniques to determine these vectors that can be divided into two classes: Pixel-recursive and block matching. Pixel-recursive techniques are used in situations when the motion vectors changes form one pixel to another. In these techniques, the motion vectors of neighboring pixels are used to predict the current motion vector and then iteratively change this estimation to minimize the value of displaced frame difference. Pixel-recursive algorithms are computationally intensive, and although quite effective, they are not used very often. If the change of these vectors takes place from one area to another, then block matching techniques are more relevant.

A block matching technique tries to find for each macroblock M_1 a macroblock M_2 in a reference picture that is its closet match [Drozdek, A., 2002].

4.5.1 Block Matching

Block matching is a process, which is used to achieve the comparison between two correlated sized blocks in different successive frames and select the best match according to specific criteria. The following sections explain the most important criteria that affect the block matching process.

4.5.1.1 Block Size Selection

To avoid the kind of difficulties encountered in motion estimation and motion compensation with arbitrarily shaped blocks, the image is partitioned into a set of nonoverlapped, equally spaced, fixed size, small rectangular blocks; and the translation motion within each block is assumed to be uniform.

The block size needs to be chosen properly. In general, the smaller block size is the more accurate is the approximation. It is apparent, however the smaller block size leads to more motion vectors being estimated and encoded, which means an increase in both computation and side information. As a compromise, a size of 16×16 is considered to be a good choice. (This has been specified in international video coding standard such as H.261, H.263, and MPEG-1 and MPEG-2). Note that for accurate estimation a block size of 8×8 is sometimes used [Yun, Q.,and Huifang, S., 2000].

4.5.1.2 Matching Criteria

The choice of matching criteria is important since block matching might require the distortion function to be evaluated many thousands of times. Therefor, many matching functions (Cost functions) was proposed and used in several block matching techniques.

4.5.1.2.1 Cost Functions

The block matching techniques for motion estimation obtain the motion vector by minimizing a cost function. The following cost function have been proposed in the literature:

(a) The Mean-Absolute Difference (MAD), defined as:

$$MAD(dx, dy) = \frac{1}{mn} \sum_{i=-n/2}^{n/2} \sum_{j=-m/2}^{m/2} \left| F(i, j) - G(i + dx, j + dy) \right| \qquad \ldots (4.5)$$

Where

F(i,j) represents a (m x n) macroblock form the current frame,

G(i,j) represents the same macroblock form a reference frame (past or future),

(dx,dy) a vector representing the search location.

The search space is specified by dx = {-p, +p} and dy = {-p, +p}.

(b) The Mean-Squared Difference (MSD) cost function is defined as:

$$MSD(dx, dy) = \frac{1}{mn} \sum_{i=-n/2}^{n/2} \sum_{j=-m/2}^{m/2} \left[F(i, j) - G(i + dx, j + dy) \right]^2 \qquad \ldots (4.6)$$

(c) The Cross-Correlation Function (CCF) is defined as:

$$CCF(dx, dy) = \frac{\sum_{i=-n/2}^{n/2} \sum_{j=-m/2}^{m/2} F(i, j) G(i + dx, j + dy)}{\left(\sum_{i=-n/2}^{n/2} \sum_{j=-m/2}^{m/2} F(i, j) \right)^{\frac{1}{2}} \left(\sum_{i=-n/2}^{n/2} \sum_{j=-m/2}^{m/2} G(i + dx, j + dy) \right)^{\frac{1}{2}}} \qquad \ldots (4.7)$$

The mean absolute difference (MAD) cost function is considered a good candidate for video applications, because it is easy to implement in hardware. The other two cost functions, MSD, and CCF, can be more efficient, but are too complex for hardware implementations.

The Pixel Difference Classification (PDC) is an alternative criterion used to reduce the computational complexity of MAD, MSD, and CCF cost functions. The PDC criterion is defined as [Gharavi, H., and Mills, M., 1990]:

$$PDC(dx, dy) = \sum_{i=1}^{8} \sum_{j=1}^{8} T(dx, dy, i, j), \qquad \ldots(4.8)$$

$$for(dx, dy) = \{-p, +p\}$$

T(dx ,dy, i, j) is the binary representation of the pixel difference defined as:

$$T(dx, dy, i, j) = \begin{Bmatrix} 1, \left| F(i, j) - G(i + dx, j + dy) \le t \right| \\ 0, otherwize \end{Bmatrix}$$

Where t is a pre-defined threshold value.

In this way, each pixel in a macroblock is classified as either a matching pixel (T = 1) or a mismatching pixel (T = 0). The block that maximizes the PDC function is selected as the best matched block [Furth, B., 2000].

4.5.1.3 Motion Vector Estimation Algorithms

In block-matching techniques, the goal is to estimate the motion of a block of size (m × n) in the present frame in relation to the pixels of previous or the future frames. The block is compared with the corresponding block within a search area of size (m + 2p)×(n + 2p) in previous (or the future) frame. Figure (4.18) shows the search window allocation.

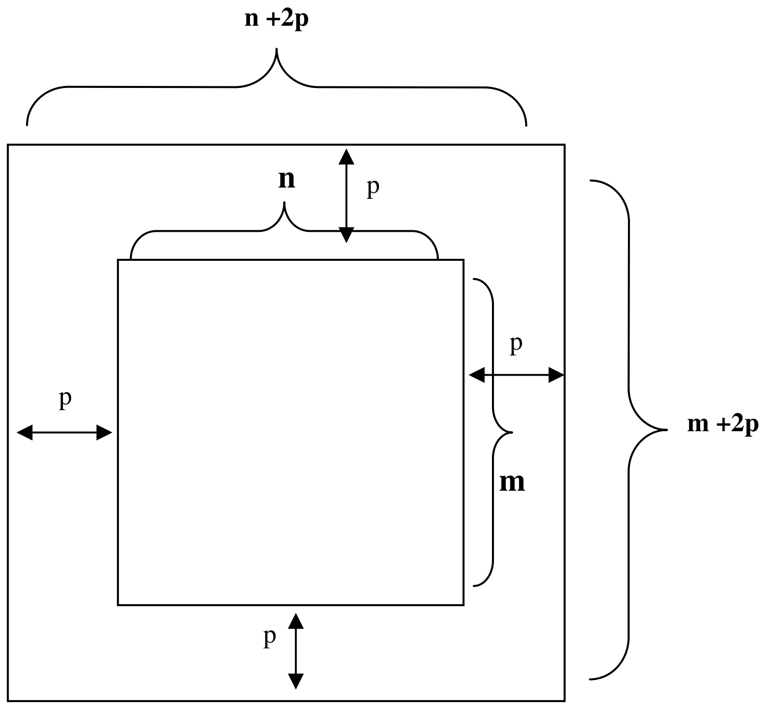


Figure (4.18): The search window allocation

In a typical MPEG 16 ×16 pixels (n = m = 16) and the parameter p = 6, are as illustrated in Figure (4.18).

Many block-matching techniques for motion vector estimation have been developed and evaluated in the literature. They are introduced below.

4.5.1.3.1 The Full Search Algorithm

The full (exhaustive) search algorithm is the simplest but computationally most intensive search method that evaluates the cost function at every location in the search area. If MSD cost function is used for estimating the motion vector, it would be necessary to evaluate (2p+1) X (2p+1) MSE functions. For p = 6, it gives 169 iterations for each macroblock [Furth, B., 2000].

Apparently, this full search procedure is a brute force in nature. While the full search delivers good accuracy in searching for the best match (thus, good accuracy in motion estimation), a large amount of computation is involved.

4.5.1.3.2 The Three-step Search Algorithm

The three-step search algorithm proposed by Koga et al. [Koga, J., and et al, 1981] and implemented by Lee et al. [Lee, W., and et al, 1994] first calculated the cost function at the center and eight surrounding locations in the search area. The location that produces the smallest cost function (typically MSD function is used) becomes the center location for the next step, and the search range is reduced by half. A three-step motion vector estimation algorithm for p = 6 is shown in Figure (4.19) [Furth, B., 2000]. The following steps illustrate the three-step searching strategy.

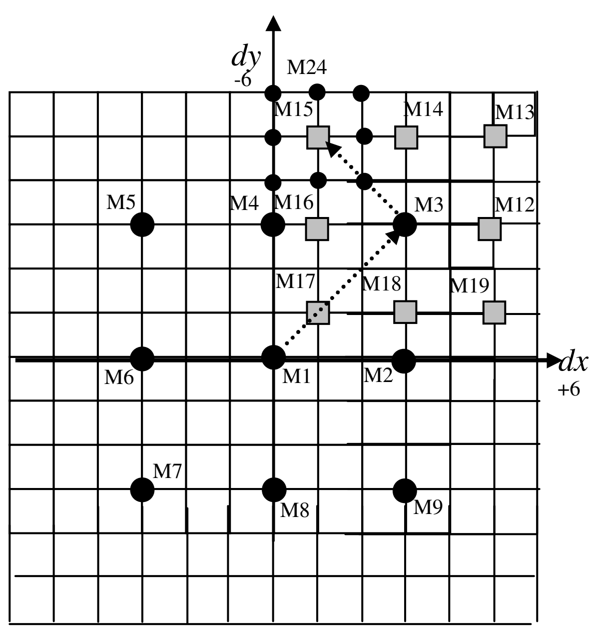


Figure (4.19): The three-step motion vector estimation algorithm

Step 1

In the first step, nine values for the cost function MAD (for simplification purposes denoted as M) are calculated: M_1 = M (0, 0), M_2 = M (3, 0), M_3 = M (3, 3), M_4 = M (0, 3), M_5 = M (-3,3), M_6 = M (-3, 0), M_7 = M (-3, -3), M_8 = (0, -3), M_9 = M (3, -3), as illustrated in Figure (4.19). Assuming that M_3 gives the smallest cost function, it becomes the center location for the next step.

Step 2

Nine new cost functions are calculated, for M_3 and eight surrounding locations, using a smaller step equal to 2. These nine points are denoted in Figure (4.19) as M_{11}, M_{12}, M_{13}…M_{19}.

Step 3

In the last step, the location with the smallest cost function is selected as a new center location (in the example in Figure (4.19), this is M_{15}), and nine new cost functions are calculated surrounding this location: M_{21}, M_{22}, M_{23},…M_{29}. The smallest value is the final estimate of the motion vector. In the example in Figure (4.19), it is M_{24}, which gives the motion vector {dx , dy} equal to {1, 6}.

Note that the total number of computations of the cost function is: 9 x 3 - 2 = 25, which is much better than l69 in the full search algorithm.

4.5.1.3.3 The 2-D logarithmic Search Algorithm

Jain and Jain [Jain, J., and Jain, A., 1981] developed a 2-D logarithmic searching procedure. Based on 1-D logarithmic search procedure [Knuth, 1973], the 2-D procedure successively reduces the search area, thus reducing the computational burden [Yun, Q.,and Huifang, S., 2000].

The modified version of the algorithm, described by Srinivasan and Rao [Srinivasan, R., and Rao, K.,1985], uses the MAD cost function, and can be described using the following steps, as illustrated in Figure (4.20) [Furth, B., 2000].

Step 1

The MAD function is calculated for dx = dy = 0, M (0, 0) and compared to the threshold (e.g., the value is 4 out of 255): M (0, 0) < T. If this is satisfied, the tested block is unchanged and the search is complete.

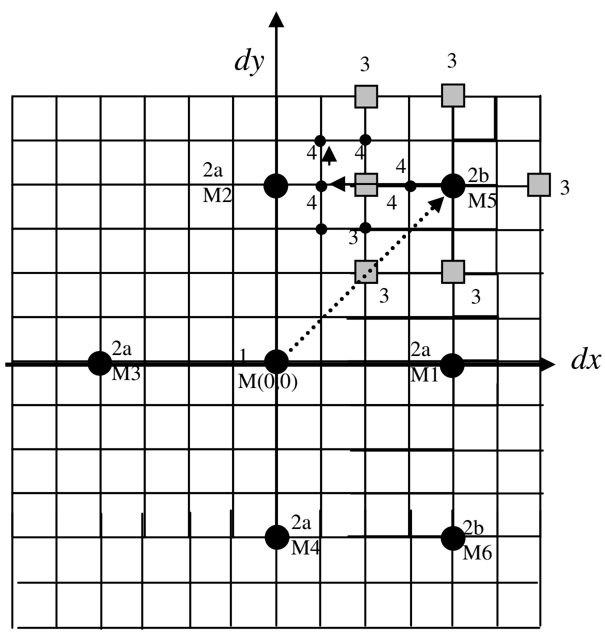


Figure (4.20): The modified 2-D logarithmic search algorithm

Step 2a

The next four cost functions are calculated, M_1 (4, 0), M_2 (0, 4), M_3 (-4, 0), and M_4 (0, -4), and their minimum is found and compared to M (0, 0):

$$M' = \min (M_1, M_2, M_3, M_4) < M(0,0)$$

If the minimum M' > M (0, 0), go to step 3; otherwise, this value is compared against the threshold, T. If M' < T, the value M' is the minimum and the search ends. Otherwise, the algorithm continues with step 2b.

Step 2b

Assuming in the previous step 2a that the minimum M' = M_1 (4, 0), then the next two surrounding positions are calculated: M_5 (4, 4), and M_6(4, -4), as indicated in Figure(4.20). The tests for minimum and threshold are performed again and, if the minimum is found, the procedure is complete. Otherwise, step 3 continues.

Step 3

Assuming that the new minimum location is M_5(4, 4), a similar search procedure (steps 2a and 2b) is continued, except the step is divided by 2. In Figure (4.20), the new minimum becomes M(2, 4).

Step 4

The step is further reduced by 2, and the final search (steps 2a and 2b) is performed. The minimum (dx, dy) is found. In Figure (4.20) it is (1, 5).

For p = 6, this algorithm requires maximum 19 cost function calculations, as shown in Figure (4.20).

4.5.1.3.4 The Conjugate Direction Search Algorithm

This algorithm for motion vector estimation, proposed by Srinivasan and Rao [Srinivasan, R., and Rao, K.,1985], is an adaptation of the traditional iterative conjugate direction search method. This method can be implemented as one-at-a-time search method, as illustrated in Figure (4.21). In Figure (4.21), direction of search is parallel to one of coordinate axes, and each variable is adjusted while the other is fixed. This method has been adapted for motion vector estimation [Srinivasan, R., and Rao, K.,1985], as illustrated in Figure (4.21). The algorithm consists of the following three steps:

Step 1:

Values of the cost function MAD in the dx direction are calculated until the minimum is found. The calculation is as follows: (a) M(0, 0), M(1, 0), and M(-1, 0). (b) If M(1, 0) is the minimum, M(2, 0) is computed and evaluated, and so on. This step is complete when a minimum in the dx direction is found [in Figure (4.21), the minimum is M(2, 0)].

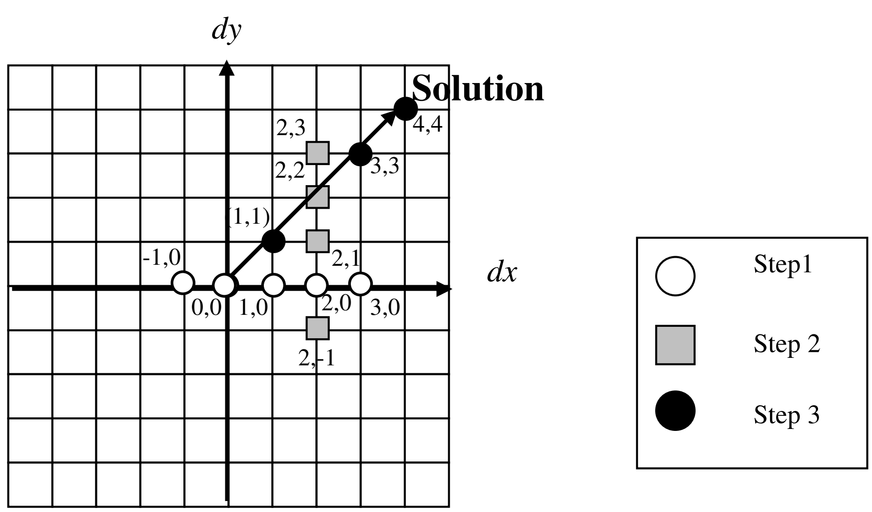


Figure (4.21): The Conjugate direction search method for motion vector estimation

Step 2:

The search now continues in the dy direction by calculating cost functions M(2, -1) and M(2, 1). A minimum in the dy direction is then found at M(2, 2) in Figure (4.21).

Step 3:

The direction of search is now the vector connecting the starting point (0, 0) and the obtained minimum (2, 2). The following cost functions are calculated and evaluated next: M(1, 1) and M(3, 3), and so on, until a minimum in this direction is found. In the example in Figure (4.21), the minimum is M(4, 4), and the obtained motion vector is dx = 4 and dy = 4.

It may happen that the dx and dy vectors, obtained in steps 2 and 3, do not constitute a square as given in Figure (4.21). In that case, the nearest grid points on the direction joining (0, 0) and the obtained minimum point are selected.

5. H.263-MOTION ESTIMATION-COMPENSATION: PROPOSED IMPROVEMENTS

5.1 Introduction

In the previous chapter, four well-known block matching search techniques were reviewed as well as the strategy used in each of them. These techniques differ from each other. Each one has advantages and drawbacks affecting directly the effectiveness of H.263 CODEC.

In this chapter, two considerably improved techniques are developed to enhance the efficiency of H.263 technique. This is done through enhancing video image quality and reducing encoding delay time limitations.

The first technique (called Thresholding Half-pixel technique) greatly intensifies the performance of the traditional half-pixel technique by adding threshold value controls clarity and smoothness for each of the pixel groups included from which video frames are composed.

The second technique (called OddEven search technique) employs a new search strategy to match similar blocks from video successive frames, and exclude the similar blocks. This technique achieves an efficient compression process. These techniques are explained below.

5.2 Thresholding Half-Pixel Accuracy Improvement

As we mentioned earlier, H.263 uses sub-pixel accuracy for motion compensation instead of using a loop filter to smooth the anchor (reference) frame as in H.261. The half-pixel technique achieves the image smoothing by considering a pixel and its neighbors, eliminating any extreme values (high frequency-noise) in this group. The values of half-pixel are found using bilinear interpolation [Yun, Q.,and Huifang, S., 2000].

Half-pixel technique does not take into account the color distance relationship among pixels directly, but rather it produces values representing half the distance among these pixels.

In the current study, we improved the technique by adding an equilibrium factor to achieve high level of smoothness among anchor frame pixels through computing threshold value.

Threshold value stands for the average difference between two pairs of pixels which reduces the noisy pixel's values (not correlated pixels) resulted from half-pixel technique. This makes pixel values of each group closer, hence leading to enhance the video anchor frames.

The improvement of half-pixel accuracy is explaining in Figure (5.1) and equation (5.1):

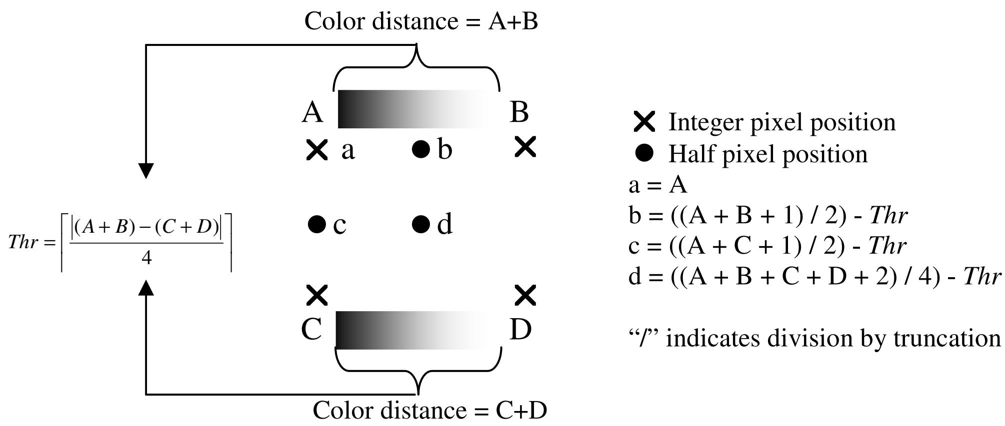


Figure (5.1): Thresholding Half-pixel technique

$$Thr = \left\lceil \frac{|(A+B)-(C+D)|}{4} \right\rceil \quad \ldots(5.1)$$

$$a = A$$
$$b = ((A+B+1)/2) - Thr$$
$$c = ((A+C+1)/2) - Thr$$
$$d = ((A+B+C+D+2)/4) - Thr$$

Where *A, B, C*, and *D* are grouped pixels and *Thr* is an additional value used to reduce the extreme half-pixel values. According to equation (5.1), threshold value will decrease color distance for group (a, b, c, and d) and produces a new group of pixels with suitable color approximation.

5.3 OddEven Search Technique (OES)

The main idea of the proposed OddEven (OES) search algorithm starts by computing the cost function (Typically, MAD function is used) for the odd blocks of x-axis and for the even blocks of y-axis finding the best match.

The matching candidate block contains the minimum value of the computed cost function values in the search windows of the previous frame. This block represents the most similar for the selected block in the current frame of the video sequence.

The matching process starts by calculating the cost function of the odd locations of x-axis and even locations of y-axis in the search window. The location that produces the smallest cost function becomes the center location for the next step, and the search range is reduced by half.

The matching process of the OES algorithm will continue until achieving the condition that the minimum computed cost function value is equal or less than the threshold value previously determined to achieve the matching process.

Threshold value is determined, depending on the requirements of the applications that use the proposed algorithm. The threshold value controls the speed and efficiency of the block matching process. This effectively affects the H.263 bit rate and encoding delay time.

Figure (5.2a) shows the block matching process on the y-axis. Figure (5.2b) shows the block matching process on the x-axis.

5.3.1 OddEven Control Parameters

This technique uses two control values to produce the best and fastest block matching process. These control values are:

- **Information Density Control Value**: A value is determined representing number of bits from which the maximum color value of the matched block is composed. This value represents the amount of color information distributed of the matched block. The following formula is used to compute this value:

$$MADThreshold = \lceil Log_2(Max(SBlk)) \rceil$$

Where, SBlk represents the current frame candidate block for matching process.

The computed value of MADThreshold is compared with the minimum MAD computed values. If the selected MAD minimum value is less or equal to MADThreshold value, the matching process will be terminated. The block of the minimum MAD value is the approximate match of the SBlk.

- **Loop Step Size Control Value**: Three values are used (0.5, 0.7, or 0.9) to reduce the matching process comparison on searching window blocks to find the best match. Where:

  - 0.5 value, divides the loop iteration by 2, this makes the OddEven technique compare half of searching window blocks.
  - With 0.7 value, OddEven technique compare one third of searching window blocks.
  - With 0.9 value, OddEven technique compare all of searching window blocks.

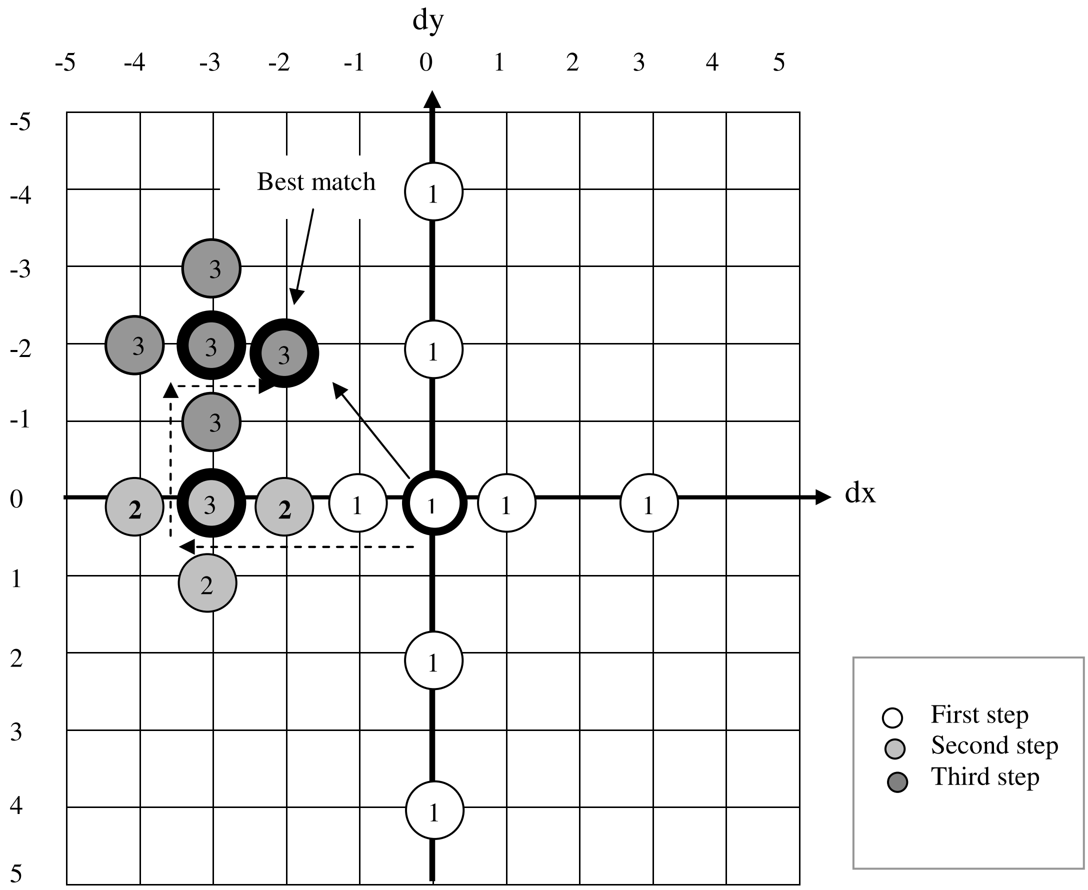


Figure (5.2a): A OddEven search procedure. Points at (0,-3), (-2,-3), and (-2,-2) are found to give the minimum dissimilarity in steps 1,2, and 3 respectively.

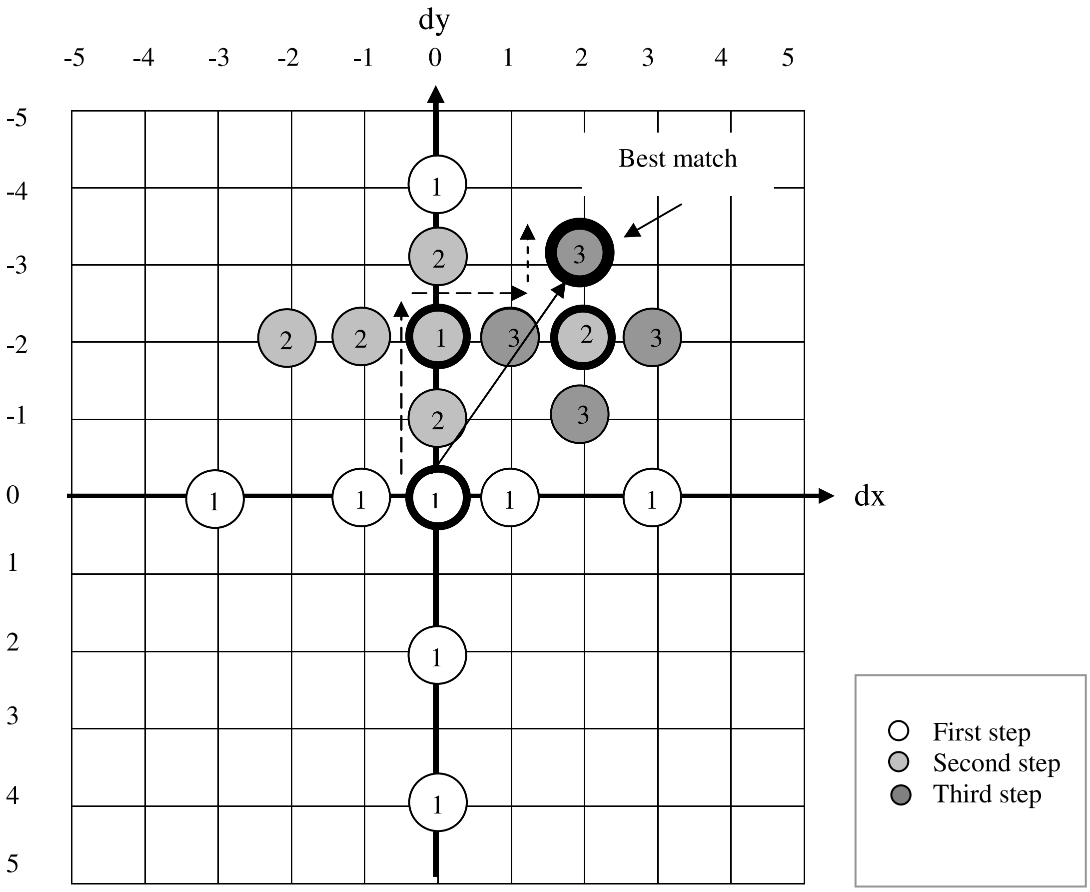


Figure (5.2b): A OddEven search procedure. . Points at (-2,0), (-2,2), and (-3,2) are found to give the minimum dissimilarity in steps 1,2, and 3 respectively.

OddEven Search Algorithm

INPUT

Swin The searching window.

SBlk : Any block of the current frame, Frame $_i$.

$StepSize_{Threshold}$ A value used as a loop step size. This value takes (0.5, 0.7 , or 0.9)

S_r , S_c : Stands for width and height of search window (Swin).

OUTPUT:

MV : Resulted motion vector prediction value.

STEP 1:

Compute the starting search location of Swin,

$$Y_{cnter} = S_r \text{ div } 2,$$

$$X_{cnter} = S_c \text{ div } 2,$$

Where, Y_{cnter} and X_{cnter} represent the centers of the search window.

Initializing jump variables,

$$Jump_i = \text{size (SBlk)} * 2,$$

$$Jump_j = Jump_i,$$

Where, $Jump_i$ and $Jump_j$ represent Y-axis and X-axis jump variables.

Initializing MAD variable,

$$Min_{MAD} = 0.$$

Allocating fixed array size that holds the computed MAD values,

$$\text{MAD\_array} = \text{array of real}.$$

Compute an accuracy value used to control the search process.

$$\text{MAD Threshold} = \lceil \text{Log}_2(Max(SBlk)) \rceil$$

Compute the MAD of the center of Swin,

$$Min_{MAD} = MAD\ (Y_{cnter}, X_{cnter}, Swin, SBlk),$$

$$MAD_{bits} = \lceil Log_2(Min_{MAD} \times 255) \rceil$$

If MAD $_{Threshold}$ >= MAD_{bits} Then

Extract motion vector ,

MV = position of Min_{MAD},

Go to STEP 5,

EndIf

STEP 2:

Initializing the MAD counter: Let Count = 1.

Compute the MAD of the Y-axis odd blocks,

For i = (Y_{cnter} div 2) To (S_c – $Jump_i$) * $StepSize_{Threshold}$ Step $Jump_i$

MAD_array (Count) = MAD (Y_{cnter}, i, Swin, SBlk),

Increment Count by 1

EndFor

STEP 3:

Compute the MAD of the X-axis even blocks,

For j = (X_{cnter} div 2) To (S_r – $Jump_j$) * $StepSize_{Threshold}$ Step $Jump_j$

MAD_array (Count) = MAD (j, X_{cnter}, Swin, SBlk),

Increment Count by 1,

EndFor.

STEP 4:

Find the minimum MAD value form MAD array (MAD_array),

$$Min_{MAD} = Min\ (MAD_{\_array}).$$

Comparing the Min_{MAD} with MAD $_{Threshold}$ and find the motion vector,

$$MAD_{bits} = \lceil Log_2(Min_{MAD} \times 255) \rceil$$

If MAD $_{Threshold}$ >= MAD_{bits} Then

Extract motion vector ,

MV = position of Min_{MAD},

Go to STEP 5,

Else

Changing the location of X_{cnter} and Y_{cnter} according to Min_{MAD} location.

Y_{cnter} = The location Y of Min_{MAD},

X_{cnter} = The location X of Min_{MAD},

Detecting the new values of the X_{cnter} and Y_{cnter} if they are equal or exceed the search window boundaries,

If ((X_{cnter}+$Jump_i$) >= S_c) OR ((X_{cnter}<=1)) Then

X_{cnter} =the Left or Right boundary value,

$Jump_i$ = $Jump_i$ div 2,

EndIf

If ((Y_{cntr}+$Jump_j$) >= S_r) OR ((Y_{cntr}<=1) Then

Y_{cnter} = the Top or Bottom boundary value;

$Jump_j$ = $Jump_j$ div 2;

EndIf.

Repeat the steps 2, 3, and 4 until finding appropriate motion vector,

EndIf.

STEP 5:

END

5.4 Experimental Results

In this section, four well-known techniques are evaluated and compared with the proposed OddEven search (OES) technique: Full Search (FS), Conjugate search, 2-D logarithmic search, and Three-step search (3SS). Several results are constructed and analyzed according to total delay time, average of PSNR, and total of compression ratio. In addition, another proposed Thresholding Half-pixel technique is compared with classical Half-pixel technique also.

The proposed and the four well-known techniques were reviewed in this thesis. The obtained results has been developed using MATLAB 6.5 and a computer system with a PIII processor and 128 MB memory. The appendix shows part of the code.

5.4.1 Image Accuracy Evaluation

The proposed Thresholding Half-pixel technique was simulated and compared with traditional Half-pixel technique. PSNR and e_{RMS} are used as two measures utilized to evaluate the image quality. Results illustrated in Table (5.1) indicate that the proposed technique has achieved clear increase in PSNR compared with the PSNR of traditional technique. Increasing PSNR means that enhanced decompressed image maintain color properties close to that of original image.

Figure (5.3a) shows the original image before compression, while Figure (5.3b) shows the color distribution (image histogram).

Figure (5.4a) represents the same original image of figure (5.3a) after compression and decompression using JPEG technique, while Figure (5.4b) represents the histogram of decompressed image.

Figure (5.5a) denotes the decompressed image of Figure (5.4a) after applying traditional Half-pixel technique. Clear enhancement is shown on decompressed image through Figure (5.5b). Produced image through applying the traditional Half-pixel is smoother than the image of Figure (5.4a).

Figure (5.6a) represents the decompressed image shown in Figure (5.4a) after applying the proposed Thresholding Half-pixel technique. Figure (5.6b) shows that color distribution of the image is better than that of Figure (5.5b).

Figure (5.7) illustrates the difference in PSNR of the proposed Thresholding Half-pixel and traditional Half-pixel techniques. It shows the efficiency of the proposed technique which achieves high image quality.

5.4.2 The Evaluation of Block Matching Search Techniques

The proposed OddEven Search (OES) algorithm is simulated using the luminance component of first 10 QCIF frames of (Foreman, Miss America, Employer, Ball, and Women model) video sequence with the IPPP… scheme (first frame is an I frame, all others are P frames). These sequences consist of a large amount of information and different scenes movement. The Mean Absolute Difference (MAD) is used as the cost function to calculate the similarity between the matched blocks. The maximum displacement in the search area is (±7) pixels in both horizontal and vertical directions for 16 ×16 block size.

All images of video sequences are composed using [0..1] scale, rather than [0..255] scale. And this will affect the PSNR constructed values.

Choosing 10 video sequence frames is based on:

- Each video clip is composed of 30 frames per second, where each 10 frames represents one of the video playback stages. The movement and speed of the moving objects of this stage are predicted. Thuse, the movement of these objects is approximately the same in the rest 20 frames.

- For test purposes, using many frames does not greatly affect the efficiency of searching techniques in video compression.

The performance comparison of OES, FS, Conjugate search, 2-D logarithmic, and 3SS in terms of total delay time are shown in Table (5.2), (5.5), (5.8), (5.11), and (5.14).

Tables (5.8), (5.14) show that the proposed OES algorithm that used three different threshold values (0.5, 0.7, and 0.9) achieved good results. This was obtained through decreasing the elapsed time for matching process in comparison with the well-known techniques in particular 3SS algorithm which had the second less delay time. Tables (5.2), (5.5), and (5.11) show that the proposed OES algorithm that used two different threshold values (0.5, 0.7) achieved the best results in comparison with 3SS algorithm.

Figures (5.16), (5.22) represent the results of Tables (5.8), (5.14). These figures indicate that the proposed OES algorithm achieved satisfactory results through using three different threshold values (0.5, 0.7, and 0.9). In Figures (5.10), (5.13), and (5.19), threshold value of (0.9) increased the elapsed time for block matching process in comparison with 3SS elapsed time.

Tables (5.3), (5.6), (5.9), (5.12), and (5.15) give some PSNR statistical comparisons between the proposed and each of the four well-known techniques.

Tables (5.3), (5.15) illustrate that the proposed OES algorithm which used (0.9) threshold value, achieved higher PSNR comparison to the PSNR of FS algorithm. Tables (5.6), (5.9) illustrate that the proposed OES algorithm which used the three different threshold values (0.5, 0.7, and 0.9) resulted in the highest PSNR in comparison with FS. Table (5.12) illustrate that threshold value of (0.5), which was used in the algorithm, achieved the highest PSNR in comparison with FS.

Tables (5.4), (5.7), (5.10), (5.13), (5.16) show another comparison according to compression ratio (C_r). Tables (5.13), (5.16) show that the proposed OES algorithm, which used the three different threshold values (0.5, 0.7, and 0.9), achieved the highest compression ratio in comparison with all of the well-known algorithms. Tables (5.4), (5.7), and (5.10) show that the proposed OES algorithm, which used the three threshold values (0.5, 0.7, and 0.9), has come near to the value of 3SS.

Different degree of reconstructed video frame clarity and compression ratio are shown in Figures (5.9), (5.12), (5.15), (5.18), and (5.21).

5.5 Discussion

The idea discussed in this thesis implies developing several improvements on the parameters that hinder the H.263 CODEC efficiency. Particularly image accuracy and encoding delay time.

These improvements included Thresholding Half-pixel technique which develops the traditional Half-pixel technique to enhance video frame accuracy through computing values indicate color distance relationship of a group of pixels. This considerably improves the performance of H.263 motion estimation/compensation.

The improvement results indicate:

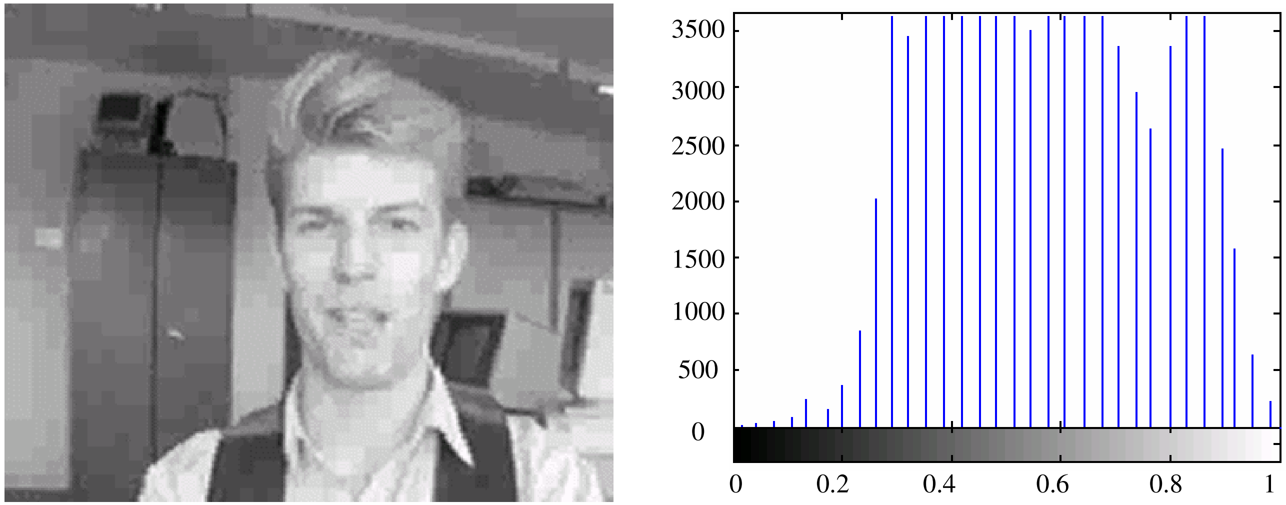
1. When computing PSNR of the decompressed image applying both the proposed Thresholding Half-pixel technique and the traditional Half-pixel technique, it appeared to be 81.1708 for the proposed technique and 79.9991 for the traditional one. The higher PSNR of the proposed technique indicates remarkable image accuracy and clarity. The image is smoother unlike the traditional Half-pixel.
2. When testing a decompressed image and applying the proposed technique, e_{RMS} appeared to be 0.0223 compared with 0.0255 for the traditional Half-pixel technique. The nearer e_{RMS} value to zero, the closer decompressed image to the original one.

The other improvement dealt with block matching search time. Using the proposed OES of one of threshold values (0.5, 0.7, or 0.9) gave higher results in comparison with the well-known four search methods. The elapsed time for block matching search and determining the motion vectors (MV) has been decreased considerably, this in turn will decrease lessen encoding delay time and increase H.263 CODEC efficiency.

The results of testing the four well-known techniques (FS, Conjugate, 2-D logarithmic, and 3SS) and the proposed (OES) indicate a reverse relationship between PSNR and compression ratio (Cr). Maintaining equilibrium between PSNR and Cr and increasing both of them result in great improvement in H.263. It can be noticed that:

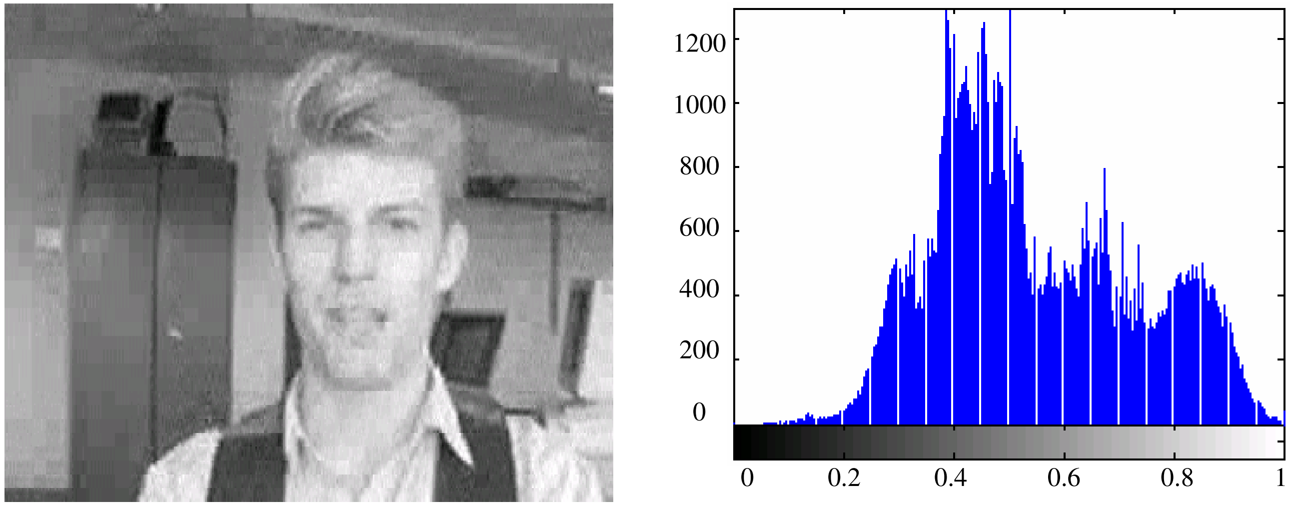
1. Using both the proposed OES and Thresholding Half-pixel resulted in noticeable increase in PSNR compared with PSNR of the traditional FS technique which is characterized with a high PSNR value.
2. With increasing PSNR using the two proposed techniques (Thresholding Half-pixel, OES). A high compression ratio (C_r) was maintained in comparison with 2-D logarithmic and 3SS techniques. Figure (5.18) and Table (5.13) shows this clearly.

5.6 Experimental Figures and Tables



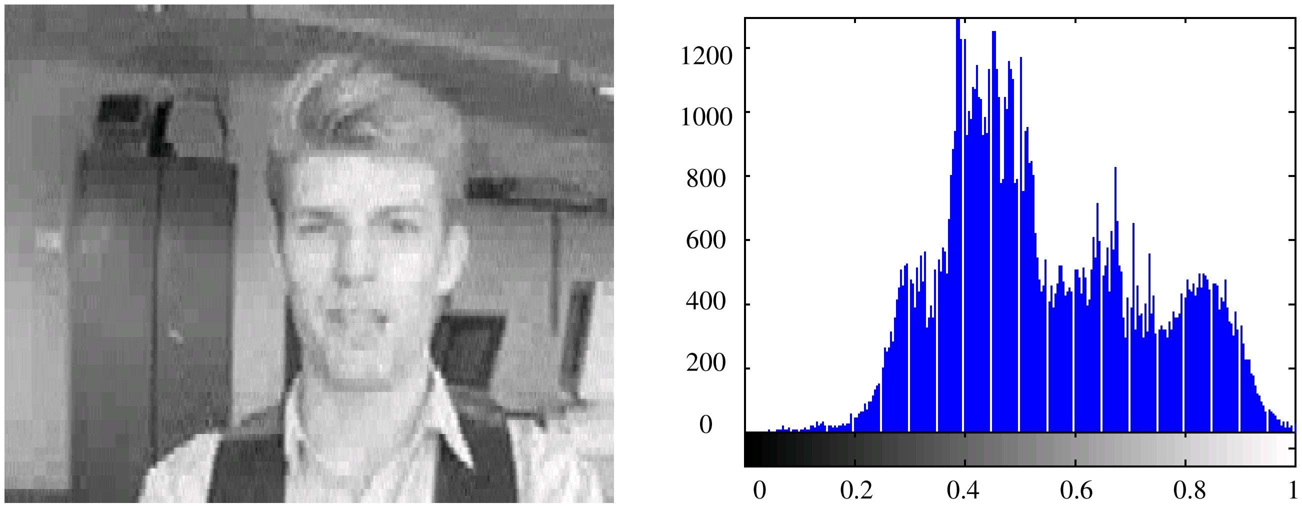
(a) (b)

Figure (5.3): (a) Original image without compression. (b) Image histogram.



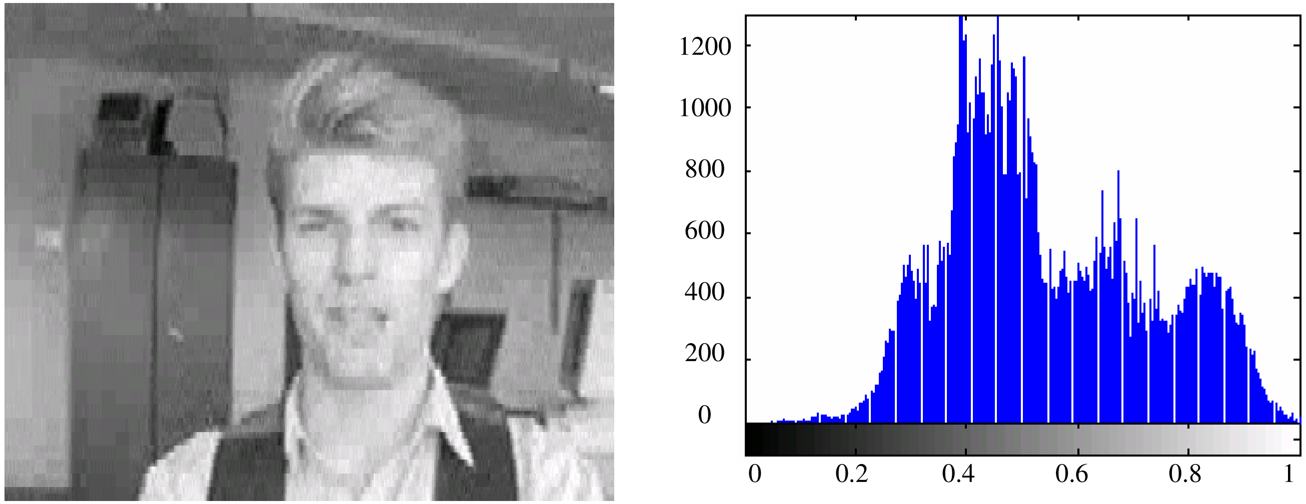
(a) (b)

Figure (5.4): (a) Decompressed image using JPEG technique. (b) Image histogram.



(a) (b)

Figure (5.5): (a) Applying traditional Half-Pixel technique on a decompressed image. (b) Image histogram.



(a) (b)

Figure (5.6): (a) Applying proposed Thresholding Half-Pixel technique on a decompressed Image. (b) Image histogram.

Table (5.1): PSNR and e_{RMS} comparison criteria between the proposed Thresholding Half-pixel technique and well-known Half-pixel technique

Image Technique	PSNR in dB	e_{RMS}
Proposed Thresholding Half-pixel	81.1708	0.0223
Half-pixel	79.9991	0.0255

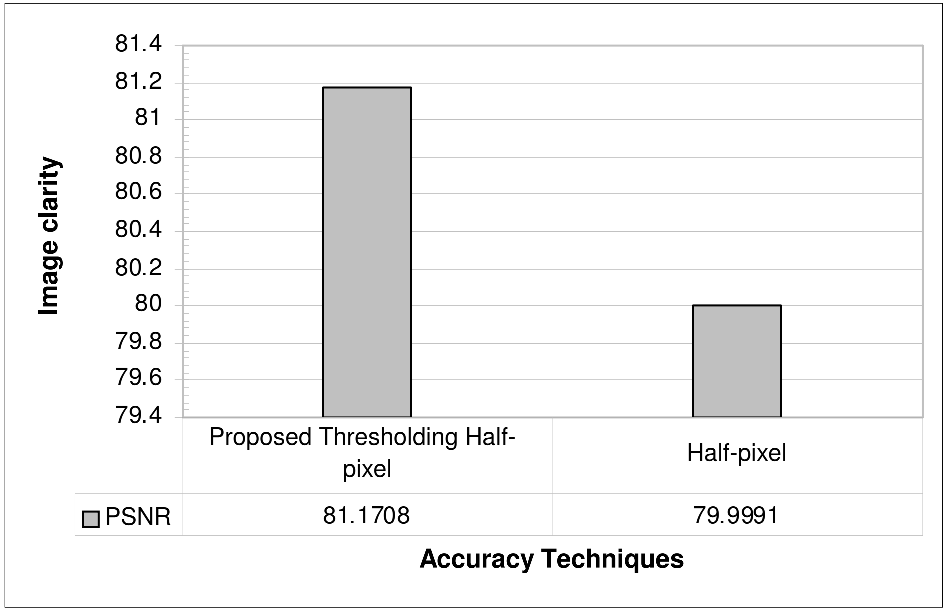


Figure (5.7): The measurement of image clarity by applying the proposed Thresholding Half-pixel and traditional Half-pixel techniques

(a) (b) (c)

Figure (5.8): Samples of Foreman video frames. (a) Current frame. (b) Previous frame. (c) The difference blocks between current and previous frames using the proposed OddEven (OES) search technique.

Table (5.2): Elapsed delay time comparison between four well-known techniques and the proposed OddEven technique for Forman video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
1.	I	11.257 ms	11.257 ms	11.257 ms	11.257 ms	11.257 ms	11.257 ms	11.257 ms
2.	P	31.826 ms	6.469 ms	7.251 ms	6.489 ms	6.158 ms	6.329 ms	6.499 ms
3.	P	30.404 ms	6.470 ms	7.250 ms	6.449 ms	6.149 ms	6.249 ms	6.479 ms
4.	P	30.363 ms	6.479 ms	7.261 ms	6.429 ms	6.169 ms	6.229 ms	6.489 ms
5.	P	30.493 ms	6.469 ms	7.210 ms	6.419 ms	6.159 ms	6.259 ms	6.499 ms
6.	P	30.364 ms	6.440 ms	7.190 ms	6.419 ms	6.149 ms	6.239 ms	6.500 ms
7.	P	30.304 ms	6.479 ms	7.201 ms	6.439 ms	6.139 ms	6.229 ms	6.469 ms
8.	P	30.714 ms	6.480 ms	7.200 ms	6.439 ms	6.168 ms	6.239 ms	6.500 ms
9.	P	30.494 ms	6.459 ms	7.210 ms	6.500 ms	6.149 ms	6.229 ms	6.490 ms
10.	P	31.335 ms	6.479 ms	7.231 ms	6.429 ms	6.178 ms	6.239 ms	6.479 ms
Total Delay Time (ms)		287.554 ms	69.481 ms	76.261 ms	69.369 ms	66.675 ms	67.498 ms	69.661 ms

Table (5.3): PSNR comparison between four well-known techniques and the proposed OddEven technique for Forman video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
PSNR	I	80.50	80.50	80.50	80.50	80.50	80.50	80.50
PSNR	P	80.614	74.395	78.695	75.499	79.233	79.912	79.438
Average of PSNR (dB)		80.557	77.445	79.597	77.999	79.866	80.206	79.969

Table (5.4): Compression ratio (C_r) comparison between four well-known techniques and the proposed OddEven technique for Forman video sequence

Frame Compression ratio (C_r)		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
C_r	I	8.264	8.264	8.264	8.264	8.264	8.264	8.264
C_r	P	37.263	41.558	51.429	56.256	53.850	54.124	54.601

Total C_r	45.527	49.822	59.693	64.5205	62.114	64.388	62.865

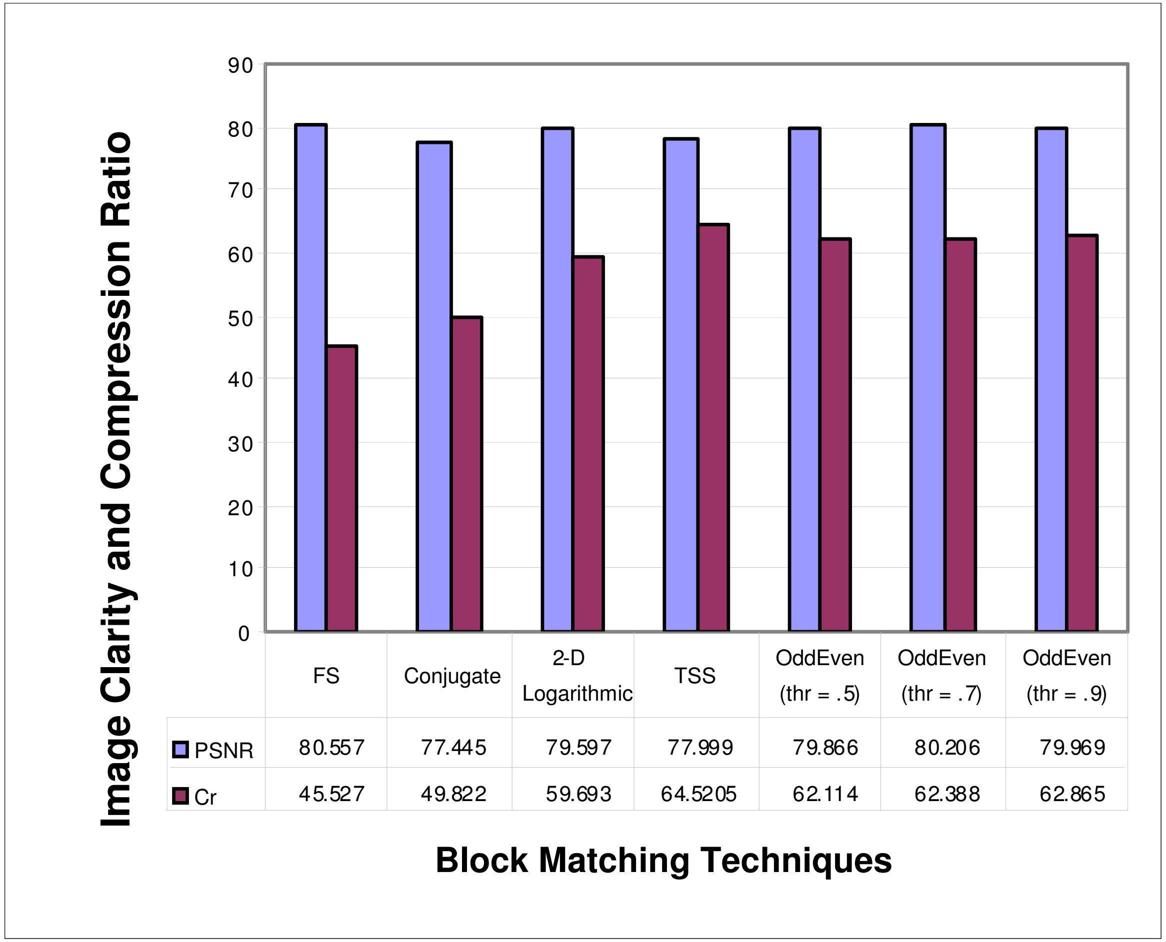


Figure (5.9): Image clarity and compression ratio distribution values of the reconstructed Forman video frames

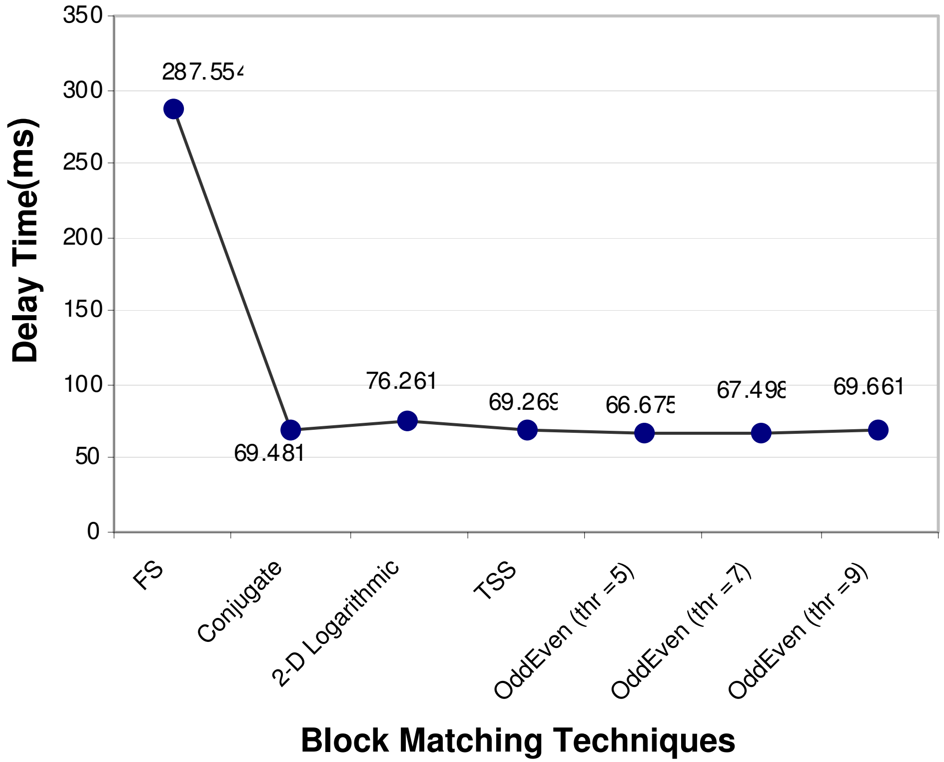


Figure (5.10): Encoder delay time distribution values of the reconstructed Forman video frames

(a) (b) (c)

Figure (5.11): Samples of Miss America video frames. (a) Current frame. (b) Previous frame. (c) The difference blocks between current and previous frame using Three-step technique.

Table (5.5): Elapsed delay time comparison between four well-known techniques and the proposed OddEven technique for Miss America video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
1.	I	10.756 ms	10.756 ms	10.756 ms	10.756 ms	10.756 ms	10.756 ms	10.756 ms
2.	P	28.691 ms	6.459 ms	7.211 ms	6.549 ms	6.099 ms	6.309 ms	6.599 ms
3.	P	29.323 ms	6.309 ms	7.831 ms	6.890 ms	6.349 ms	6.249 ms	6.610 ms
4.	P	29.022 ms	6.309 ms	7.110 ms	6.309 ms	6.209 ms	6.269 ms	6.600 ms
5.	P	28.571 ms	6.309 ms	7.110 ms	6.339 ms	6.109 ms	6.259 ms	6.610 ms
6.	P	29.242 ms	6.319 ms	7.131 ms	6.339 ms	6.098 ms	6.249 ms	6.589 ms
7.	P	29.443 ms	6.319 ms	7.110 ms	6.349 ms	6.098 ms	6.279 ms	6.600 ms
8.	P	29.242 ms	6.329 ms	7.131 ms	7.040 ms	6.088 ms	6.259 ms	6.609 ms
9.	P	29.032 ms	6.329 ms	8.051 ms	6.339 ms	6.119 ms	6.239 ms	6.609 ms
10.	P	29.101 ms	6.299 ms	7.121 ms	6.339 ms	6.129 ms	6.229 ms	6.630 ms
Total Delay Time (ms)		272.423 ms	67.737 ms	76.562 ms	69.249 ms	66.054 ms	67.097 ms	70.212 ms

Table (5.6): PSNR comparison between four well-known techniques and the proposed OddEven technique for Miss America video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
PSNR	I	84.583	84.583	84.583	84.583	84.583	84.583	84.583
PSNR	P	77.776	78.045	81.992	81.047	83.491	82.899	82.662
Average of PSNR (dB)		81.179	81.314	83.287	82.815	84.037	83.741	83.622

Table (5.7): Compression ratio (C_r) comparison between four well-known techniques and the proposed OddEven technique for Miss America video sequence

Frame Compression ratio (C_r)	Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9

C_r	I	13.661	13.661	13.661	13.661	13.661	13.661	13.661
C_r	P	54.067	54.558	72.712	76.245	73.720	76.009	75.859
Total C_r		67.728	68.219	86.373	89.906	87.381	89.670	89.520

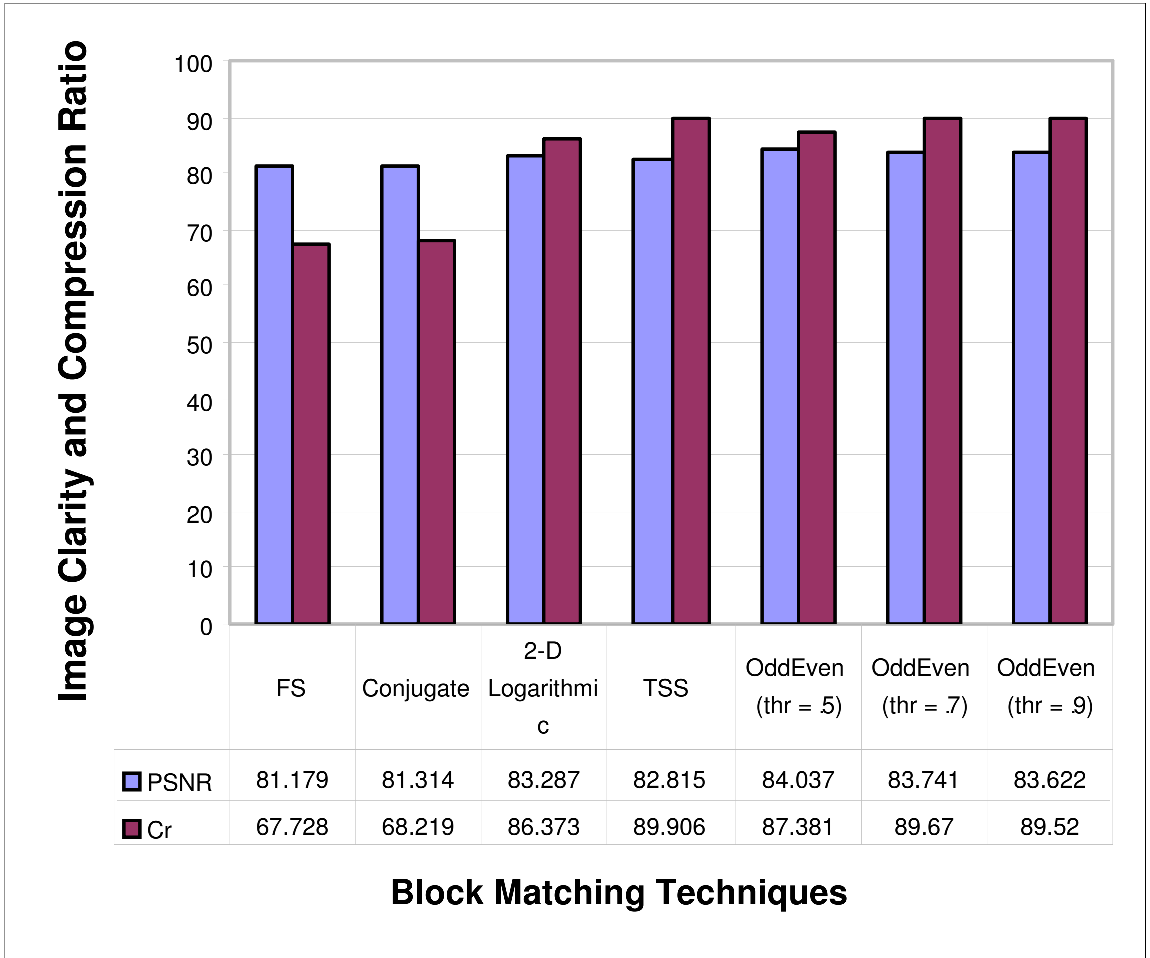


Figure (5.12): Image clarity and compression ratio distribution values of the reconstructed Miss America video frames

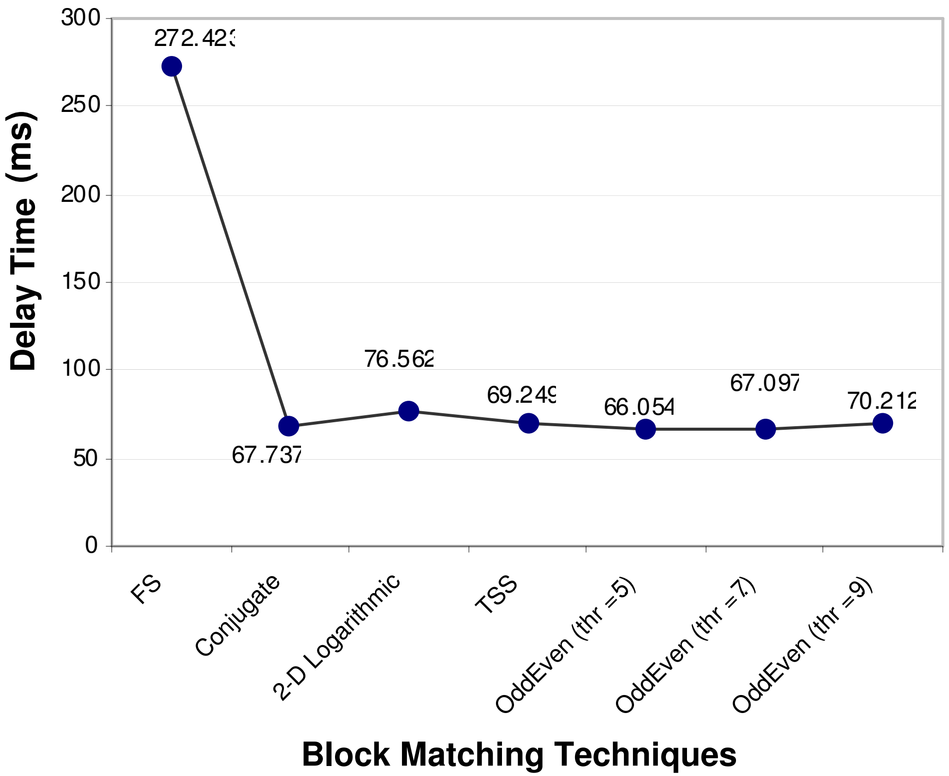


Figure (5.13): Encoder delay time distribution values of the reconstructed Miss America video frames

(a)

(b)

(c)

Figure (5.14): Samples of Employer video frames. (a) Current frame. (b) Previous frame. (c) The difference blocks between current and previous frame using 2-D logarithmic technique.

Table (5.8): Elapsed delay time comparison between four well-known techniques and the proposed OddEven technique for Employer video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
1.	I	11.076 ms	11.076 ms	11.076 ms	11.076 ms	11.076 ms	11.076 ms	11.076 ms
2.	P	31.315 ms	6.719 ms	7.621 ms	6.609 ms	6.179 ms	6.269 ms	6.559 ms
3.	P	31.225 ms	6.559 ms	7.451 ms	6.530 ms	6.179 ms	6.279 ms	6.519 ms
4.	P	31.546 ms	6.549 ms	7.470 ms	7.421 ms	6.189 ms	6.299 ms	6.509 ms
5.	P	31.736 ms	6.559 ms	7.451 ms	6.509 ms	6.148 ms	6.309 ms	6.549 ms
6.	P	31.455 ms	6.559 ms	7.441 ms	6.560 ms	6.168 ms	6.299 ms	6.540 ms
7.	P	31.625 ms	6.559 ms	7.430 ms	7.380 ms	6.159 ms	6.289 ms	6.559 ms
8.	P	31.465 ms	6.520 ms	7.441 ms	6.579 ms	6.159 ms	6.289 ms	6.569 ms
9.	P	31.034 ms	6.550 ms	8.332 ms	6.559 ms	6.169 ms	6.289 ms	6.559 ms
10.	P	31.796 ms	6.550 ms	7.430 ms	7.401 ms	6.139 ms	6.309 ms	6.539 ms
Total Delay Time (ms)		294.273 ms	70.200 ms	79. 143 ms	72.624 ms	66.565 ms	67.707 ms	69.978 ms

Table (5.9): PSNR comparison between four well-known techniques and the proposed OddEven technique for Employer video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
PSNR	I	81.274	81.274	81.274	81.274	81.274	81.274	81.274
PSNR	P	80.532	75.587	80.323	79.180	80.732	80.894	80.975
Average of PSNR (dB)		80.903	78.430	80.798	80.227	81.003	81.084	81.124

Table (5.10): Compression ratio (C_r) comparison between four well-known techniques and the proposed OddEven technique for Employer video sequence

Frame Compression ratio (C_r)	Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9

C_r	I	21.296	21.296	21.296	21.296	21.296	21.296	21.296
C_r	P	36.464	32.114	53.592	56.454	55.294	54.960	54.668
Total C_r		57.760	53.410	74.888	77.750	76.590	76.256	75.964

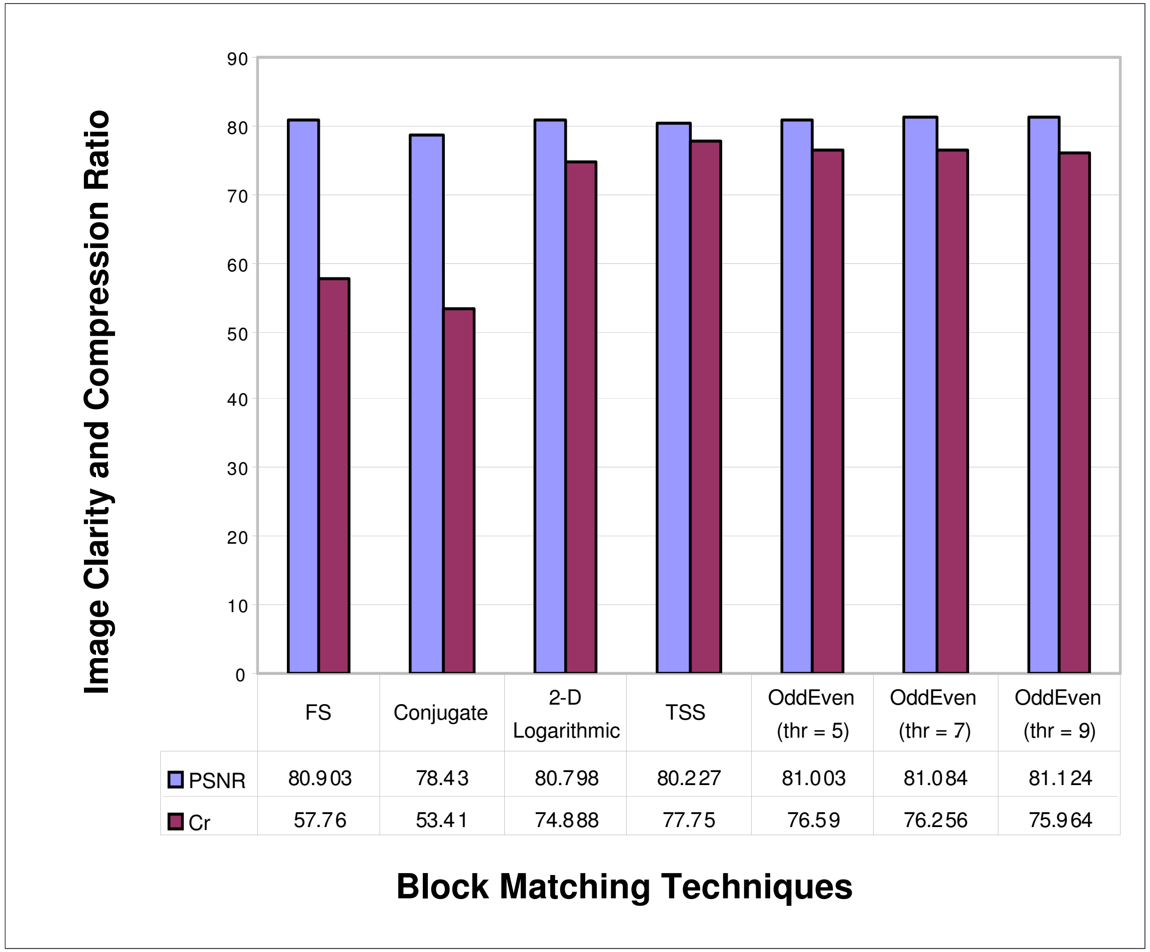


Figure (5.15): Image clarity and compression ratio distribution values of the reconstructed Employer video frames

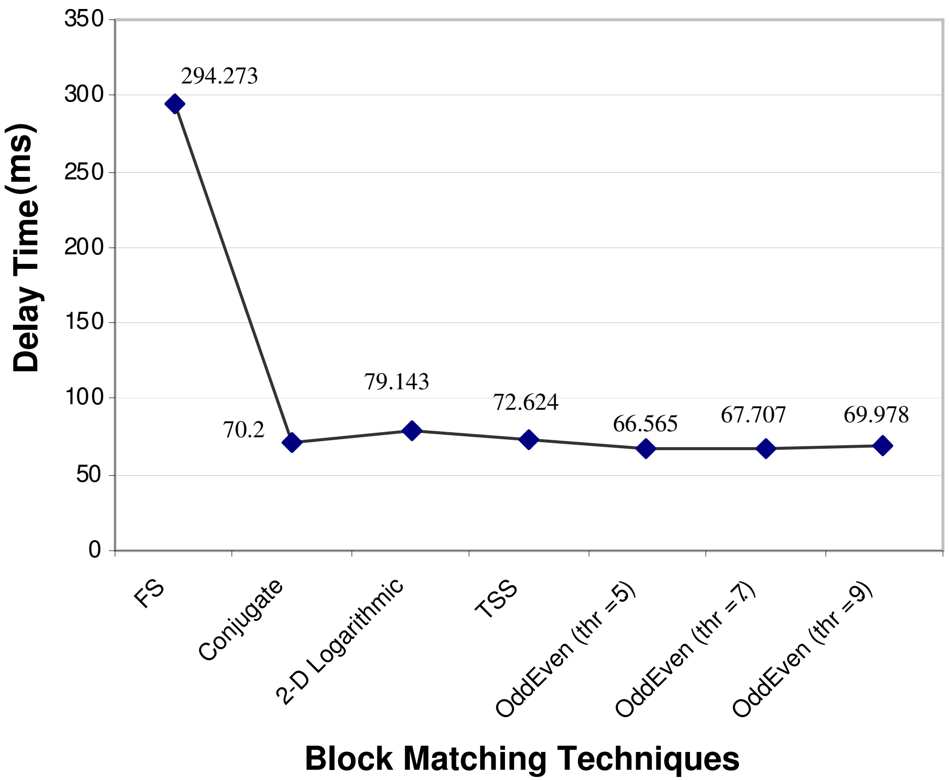
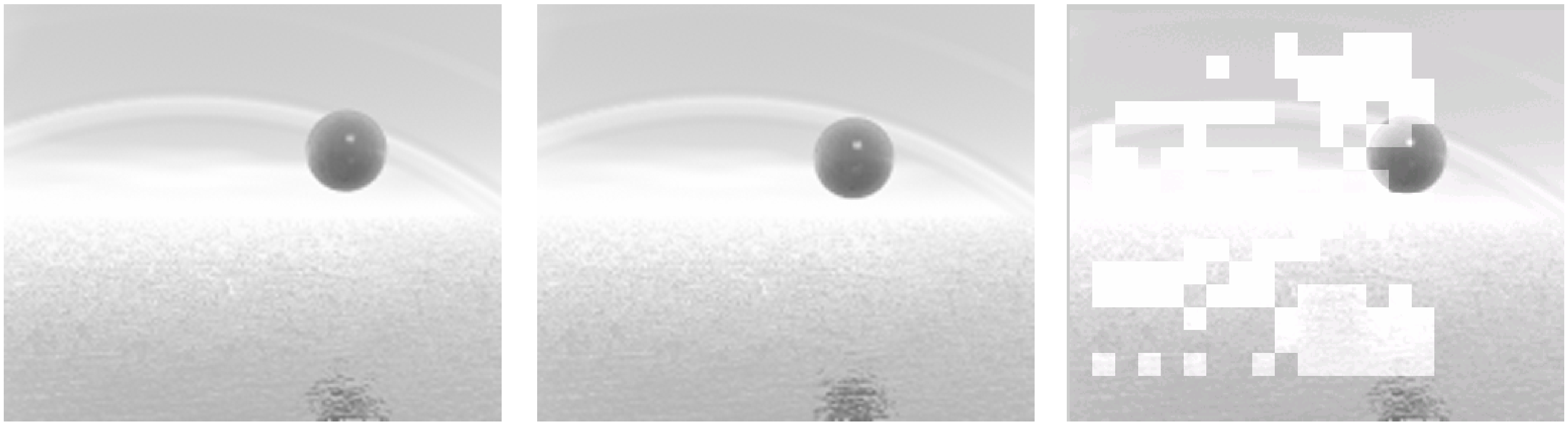


Figure (5.16): Encoder delay time distribution values of the reconstructed Employer video frames



(a) (b) (c)

Figure (5.17): Samples of Ball video frames. (a) Current frame. (b) Previous frame. (c) The difference blocks between current and previous frame using Conjugate technique.

Table (5.11): Elapsed delay time comparison between four well-known techniques and the proposed OddEven technique for Ball video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
1.	I	10.976 ms	10.976 ms	10.976 ms	10.976 ms	10.976 ms	10.976 ms	10.976 ms
2.	P	32.557 ms	6.540 ms	7.370 ms	6.520 ms	6.189 ms	6.359 ms	6.599 ms
3.	P	31.164 ms	6.560 ms	7.360 ms	6.469 ms	6.109 ms	6.349 ms	6.589 ms
4.	P	30.935 ms	6.560 ms	7.371 ms	6.469 ms	6.108 ms	6.359 ms	6.570 ms
5.	P	31.295 ms	6.549 ms	7.361 ms	6.490 ms	6.129 ms	6.340 ms	6.590 ms
6.	P	31.355 ms	6.559 ms	7.330 ms	6.490 ms	6.108 ms	6.339 ms	6.609 ms
7.	P	31.185 ms	6.569 ms	7.351 ms	6.479 ms	6.309 ms	6.300 ms	6.560 ms
8.	P	31.325 ms	6.579 ms	7.351 ms	6.469 ms	6.179 ms	6.279 ms	6.569 ms
9.	P	31.575 ms	6.559 ms	7.340 ms	6.489 ms	6.159 ms	6.449 ms	6.550 ms
10.	P	31.426 ms	6.549 ms	7.351 ms	6.469 ms	6.119 ms	6.329 ms	6.579 ms
Total Delay Time (ms)		293.793 ms	70.000 ms	77.161 ms	69.320 ms	66.385 ms	68.079 ms	70.191 ms

Table (5.12): PSNR comparison between four well-known techniques and the proposed OddEven technique for Ball video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
PSNR	I	85.135	85.135	85.135	85.135	85.135	85.135	85.135
PSNR	P	80.171	75.337	78.763	78.296	80.803	78.917	79.898
Average of PSNR (dB)		82.653	80.236	81.949	81.7155	82.969	82.026	82.516

Table (5.13): Compression ratio (C_r) comparison between four well-known techniques and the proposed OddEven technique for Ball video sequence

Frame Compression ratio (C_r)	Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9

C_r	I	13.869	13.869	13.869	13.869	13.869	13.869	13.869
C_r	P	35.600	42.417	71.915	73.329	75.607	74.667	73.674
Total C_r		49.469	56.286	85.784	87.198	89.476	88.536	87.543

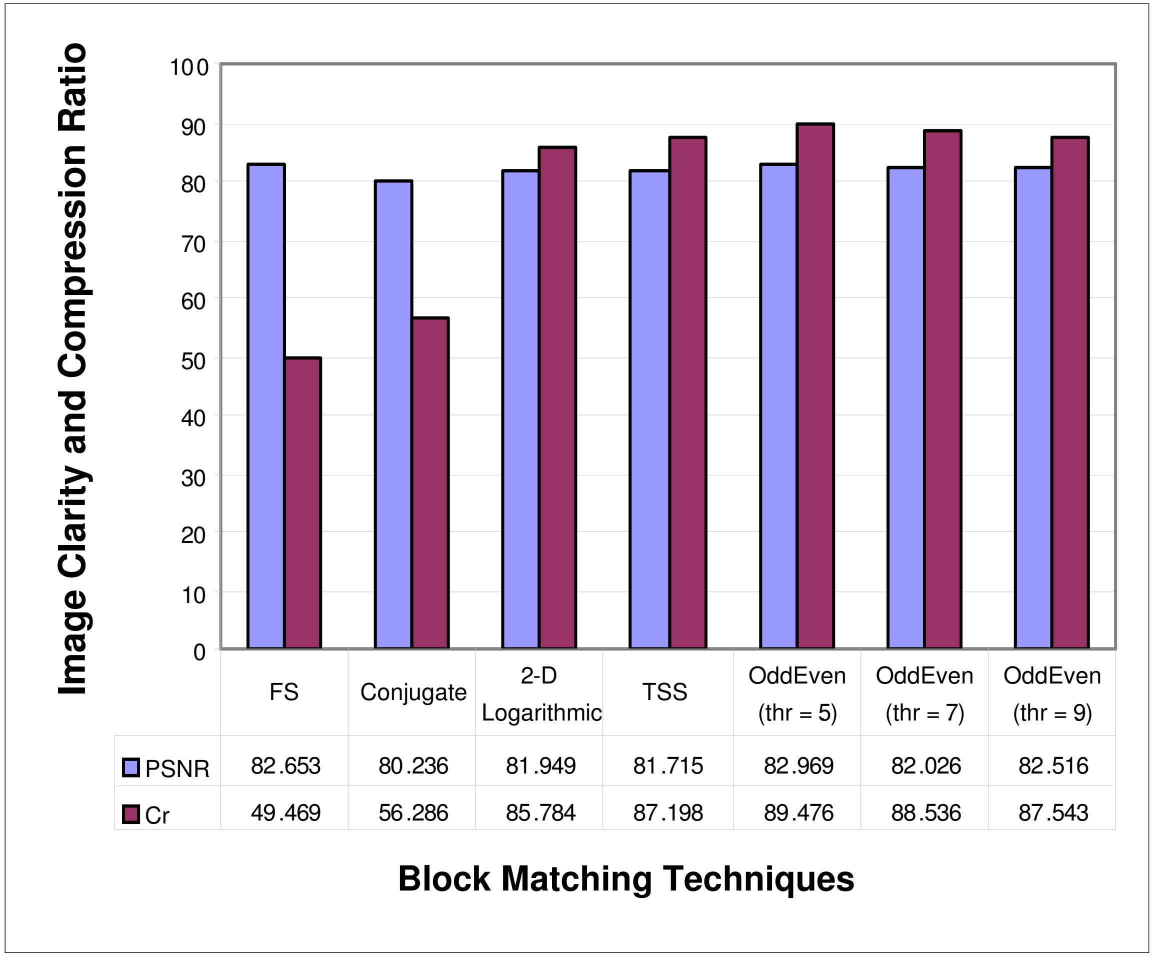


Figure (5.18): Image clarity and compression ratio distribution values of the reconstructed Ball video frames

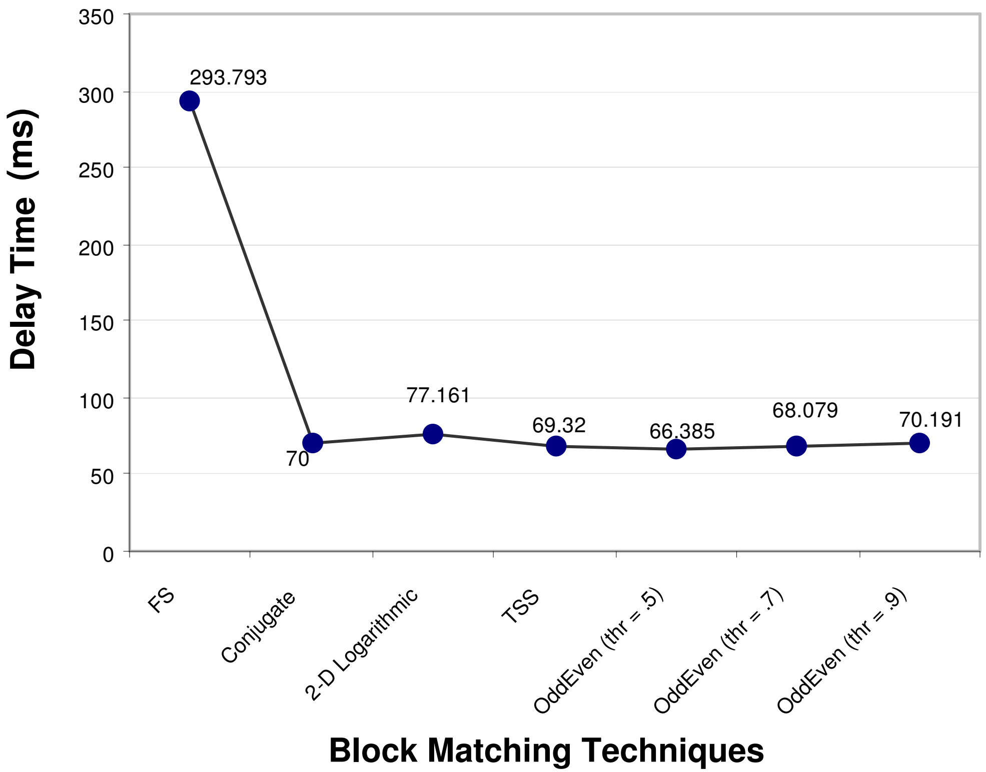


Figure (5.19): Encoder delay time distribution values of the reconstructed Ball video frames

(a) (b) (c)

Figure (5.20): Samples of Women model video frames. (a) Current frame. (b) Previous frame. (c) The difference blocks between current and previous frame using Full technique.

Table (5.14): Elapsed delay time comparison between four well-known techniques and the proposed OddEven technique for Women model video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
1.	I	11.136 ms	11.136 ms	11.136 ms	11.136 ms	11.136 ms	11.136 ms	11.136 ms
2.	P	31.506 ms	6.570 ms	7.361 ms	6.469 ms	6.149 ms	6.610 ms	6.529 ms
3.	P	31.566 ms	6.540 ms	7.330 ms	6.469 ms	6.069 ms	6.299 ms	6.510 ms
4.	P	31.586 ms	6.620 ms	7.340 ms	6.479 ms	6.089 ms	6.279 ms	6.519 ms
5.	P	31.335 ms	6.650 ms	7.341 ms	6.469 ms	6.078 ms	6.359 ms	6.510 ms
6.	P	31.465 ms	6.540 ms	7.321 ms	6.480 ms	6.089 ms	6.309 ms	6.510 ms
7.	P	31.495 ms	6.559 ms	7.331 ms	6.489 ms	6.098 ms	6.349 ms	6.499 ms
8.	P	31.476 ms	6.560 ms	7.351 ms	7.160 ms	6.139 ms	6.279 ms	6.509 ms
9.	P	31.223 ms	6.550 ms	7.340 ms	6.569 ms	6.099 ms	6.289 ms	6.519 ms
10.	P	31.065 ms	6.539 ms	7.360 ms	6.449 ms	6.149 ms	6.299 ms	6.530 ms
Total Delay Time (ms)		293.853 ms	70.264 ms	77.211 ms	70.169 ms	66.095 ms	68.208 ms	69.771 ms

Table (5.15): PSNR comparison between four well-known techniques and the proposed OddEven technique for Women model video sequence

Frame		Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search threshold = 0.5	Proposed (OddEven) search threshold = 0.7	Proposed (OddEven) search threshold = 0.9
PSNR	I	82.764	82.764	82.764	82.764	82.764	82.764	82.764
PSNR	P	82.245	76.796	81.886	80.694	81.072	82.557	81.976
Average of PSNR (dB)		82.504	79.780	82.325	81.729	81.918	82.660	82.370

Table (5.16): Compression ratio (C_r) comparison between four well-known techniques and the proposed OddEven technique for Women model video sequence

Frame Compression ratio	Full search FS	Conjugate search	2-D logarithmic search	Three-Step search 3SS	Proposed (OddEven) search	Proposed (OddEven) search	Proposed (OddEven) search

(C_r)						threshold = 0.5	threshold = 0.7	threshold = 0.9
C_r	I	13.175	13.175	13.175	13.175	13.175	13.175	13.175
C_r	P	49.492	40.278	66.536	66.663	67.979	67.738	67.862
Total C_r		62.667	53.453	79.711	79.838	81.154	80.913	81.037

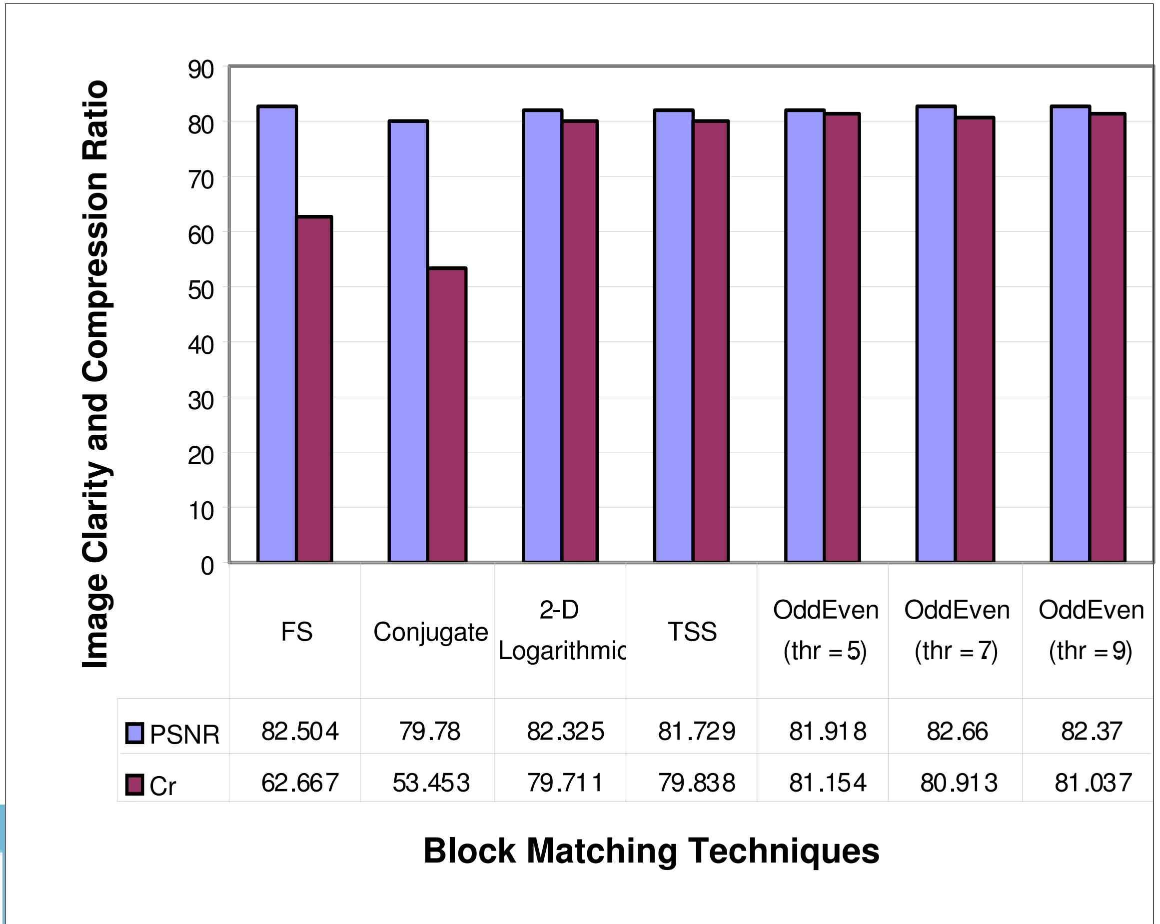


Figure (5.21): Image clarity and compression ratio distribution values of the reconstructed Women model video frames

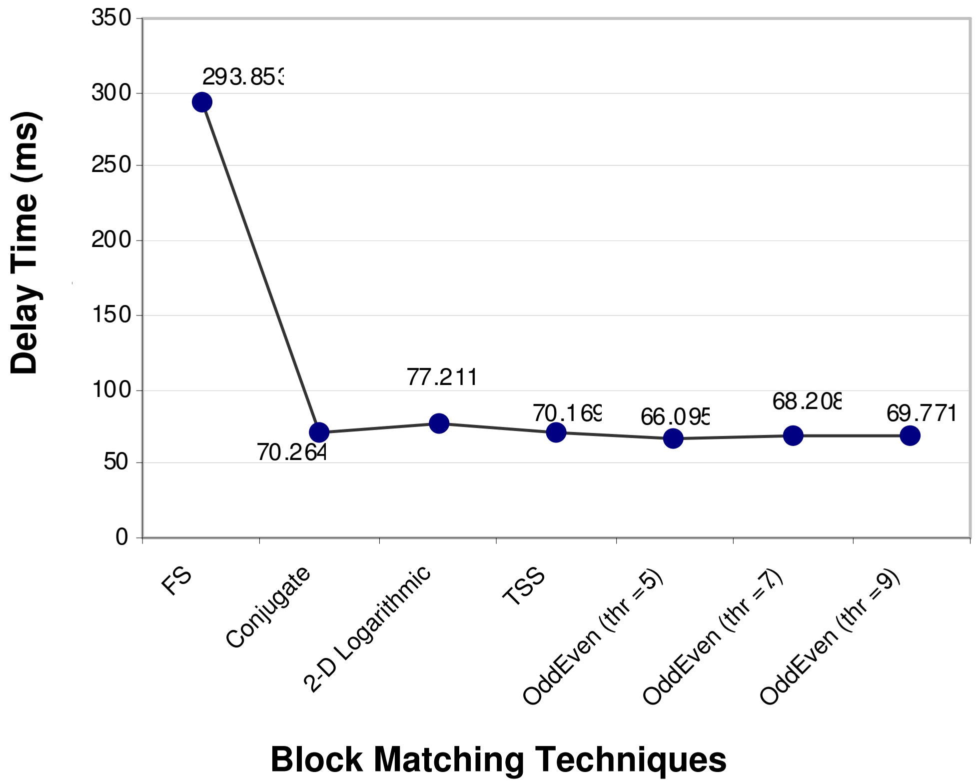


Figure (5.22): Encoder delay time distribution values of the reconstructed Woman model video frames

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this thesis we have discussed the problem of H. 263 video compression technique. Several techniques have been studied and discussed.

Two algorithms have been developed: Thresholding half-pixel technique and OddEven technique. These algorithms have enhanced video compression. The idea of the first algorithm is to increase the image clarity and accuracy, while the second one is concerned with decreasing the elapsed time of video compression and increasing the compression ratio.

Motion estimation/compensation is the core of H.263 technique, it depends on several block matching techniques. The four block matching techniques which have been studied and applied gave results of various accuracy degrees. They impose some limitation on the H.263 technique sometimes. The limitations are encoding delay time and image accuracy. This is why the two techniques were developed.

Testing these two techniques gave good results, improving significantly the H.263 technique performance. The following can be concluded:

1. Applying the proposed image enhancement technique (Thresholding Half-pixel) resulted in high accuracy and clarity in decompressed images compared with the traditional Half-pixel technique.
2. The proposed OES technique has remarkably decreased elapsed time of video compression in comparison with the four well-known techniques (FS, Conjugate,2-D logarithmic and 3SS). Using three different threshold values by OES resulted in fast block matching process, this decreased encoding delay time clearly.

3. Using both the two proposed techniques increased compression ratio (C_r) and maintained stable PSNR which is adopted as image clarity measurement for video frames. The first technique (Thresholding Half-pixel) gives high image accuracy, which makes the second technique OES perform better to find matched blocks. This in turn increase compression ratio.
4. Applying the proposed OES technique using small threshold value (i.e. 0.5) decrease elapsed delay time significantly. This threshold value make OES process half the odd and even locations, this in turn decreases the encoding delay time.
5. Choosing simple scenes with few moving objects which is called Head and Shoulder, increases PSNR, C_r and decrease the video compression delay time. Each motionless object is considered similar to an object in the same location in the next frame. These blocks can be substituted without computing motion vector (MV) for each block.

6.2 Future Work

The following are suggested future studies to complete what has been developed in this thesis.

1. Developing a new motion compensated temporal filter depending on distance pixel displacement between successive video frames.
2. Using dithering technique to enhance H.263 frames effectiveness.
3. Establishing a technique to eliminate the noise produced through H.263 bitstream transmission.
4. Developing block matching technique to perform specific comparison operations to choose the best search position. This will be done through computing color density for a randomly chosen group of video frame blocks before performing the traditional block matching search process.
5. Developing a new technique to eliminate the video transmission error.

7. REFERENCES

1- Andreas, B. and Roni, K. 2000. A 3D-DCT Real-Time Video Compression System for Low Complexity Single-Chip VLSI Implementation. *Proc. of the MoMuC2000.* P.1B-5-1. Tokyo.

2- Ahmed, N. Natarajan, T. and Rao, K. 1974. Discrete Cosine Transform. *IEEE Trans. On computers*, 1:90-93.

3- Ali, S., 1999, "Video Compression MPEG-4 and Beyond " www.cis.ohio-state.edu/~jain/cis788-99/compression/index.html.

4- Amir-Masud, E., Jamshid, S., and Fariborz, M. 2002. A New Image Retrieval Scheme Based On The Ic-Vq Compressed Image. *Proc. Of the 2002 International Arab Conference on Information Technology (ACIT' 2002).,* 1: Computer Center- University of Qatar. Qatar.

5- ALexis, M., Guobin, S., Ming, L., Oscar, C., Ishfaq, A.2000. A New Predictive Diamond Search Algorithm For Block Based Motion Estimation. *Proceeding of Visual Communication and Image Processing 2000 (VCIP-2000).* Perth. Australia.

6- Aravind, R., Cash, G., Duttweller, D., Hang, H-M., and Puri, A. 1993. Image and Video Coding Standard. *AT&T Technical Journal.* 72:67-88.

7- Bevinakoppa, S. 1999. *Still Image Compression on Parallel Computer Architectures.* 1st Ed., Kluwer Academic publisher, Dordrecht.

8- Bagni, D., de Haan, G., and Riva, V. 1997. Motion Compensated Post-Processing For Low Bit Rate Videoconferencing On Isdn Lines. *Proc. Of the (ICCE'97).* 1:28-29.

9- Cherriman , P. and Hanzo ,L. 1996. Power-Controlled H.263-Based Wireless

Videophone Performance in Interference- Limited Scemarios. *Proc. Of IEEE International Symposium Personal. Indoor. and Mobile Radio Communication.* 1:158-162,15-18.

10- Chen, W., Smith, G. and Fralick, S. 1977. A Fast Computational Algorithm for the Discrete Cosine Transform. *IEEE Trans. Communication.* 25:1004-1009.

11- Cherriman, P., Keller, T. 1998. Orthogonal Frequency Division Multiplex transmission of H.263 encoded video over highly frequency-selective wireless networks. To appear in *IEEE Trans. on CSVT*.

12- Cherriman, P., Choong H. W. 2000. Turbo- and BCH-Coded Wide-Band Burst-by-Burst Adaptive H.263-Assisted Wireless Video Telephony. *IEEE Circuits and Systems for Video Technology*.

13- Cherriman, P., and Hanzo, L. 1998. Error-rate Based Power-controlled Multimode H.263-Assisted Video Telephony. *IEEE trans. on Transactions of Vehicular Technology*.

14- Cheung, C., and Po, L. 1997. A Hierarchical Block Matching Algorithm Using Partial Distortion Measure. *Proceeding of IEEE International Conference of Image Processing.* 3: 606-609. Santa Barbara. CA. USA.

15- Drozdek, A. 2002. *Elements of Data Compression*. Wadsworth Group. Brooks/Cole, USA.

16- Dasavathy, B., V. 1995. *Image Data Compression Block Truncation Coding*. IEEE Computer Society Press, USA.

17- Efford N. 2000. *Digital Image Processing, a practical introduction using JAVA* . 1st Ed, Addison Wesley, England.

18- Furht, B., Smolior, S. W., and Zhang, H. 1996. *Video and Image in Multimedia* *system.* 2nd Ed., Kluwer Academic publisher, Dordrecht.

19- Furth, B., 2000. *Handbook of Multimedia Computing*. CRC Press, USA.

20- Furht, B., Greenberg, J., and Westwater, R. 1997. *Motion Estimation Algorithm for Video Compression*. Kluwer Academic Publishers, Norwel, MA.

21- Gharavi, H., and Mills, M. 1990. Block Matching Motion Estimation Algorithm – New Results. *IEEE Transactions on Circuits System*. 37: 649-651.

22- Hanzo, L. and Cherriman ,P. 1996. H.261 and H.263-Based Programmable Video Transceivers. *To appear in IEEE Tr. On CSVT*.

23- Hatem, E., Ismail, F., and Mohamed, H. 2002. Intelligent Motion Detection And Tracking System. *Proc. Of the 2002 International Arab Conference on Information Technology (ACIT' 2002).* 1: Computer Center- University of Qatar. Qatar.

24- Injong, R. 1998. Retransmission-Based Error Control for Interactive Video Application over the Internet. *IEEE Computer Society DL*. ICMCS. 1:118-127.

25- Jorj, L. A. 1997. *New Coding Methods for Compressing Remotely Second Images*. Ph.D. Thesis, College of Science, University of Baghdad, Iraq.

26- Jain, J., and Jian, A. 1981. Displacement Measurement and Its Application in Interframe Image Coding. *IEEE Transaction and Communication.* 29:1799-1808.

27- Jeffay, K., and Zhang, H. 2002. *Readings in Multimedia Computing and Networking*. Academic Press, USA.

28- José M. M. 1999. MPEG-7 Overview V.8. Available at (http://mpeg.telecomitalialab.com/standards/mpeg-7/mpeg-7.htm).

29- Jan, B., Keith, H. 2002. MPEG-21 Overview V.4. Available at (http://mpeg.telecomitalialab.com/standards/mpeg-21/mpeg-21.htm).

30- Justin, G., Alexander, E., Richard, E., and Eve, A. 2003. Unequal Loss Protection for H.263 Compressed Video. *Proceedings of the 2003 Data Compression Conference.* 1:73-82.

31- Khalid, S. 2000. *Introduction to Data Compression.* Academic Press, USA.

32- Kientzle, T. 1998. *A Programmer's Guide to Sound.* Addison-Wesley, USA.

33- Koga, J., Linuma, K., Hirani, Y., and Ishiguro, T. 1981. Motion Compensated Interframe Coding for Video Conferencing. *Proceedings of the National Telecommunications Conference.* 1: G5.3.1-5.3.5.

34- Keith, J. 2001. *Video Demystified A Handbook for the Digital Engineer.* LLH Technology Publishing, USA.

35- Lee ,C. S., Keller ,T., and Hanzo ,L. 2000. Turbo-Coded Terrestrial Mobile Video Broadcasting. *Proceeding of ICT.* 1:359-363.

36- Liou, M. 1991. Overview of the Px64 Kbits/s Video Coding Standard. *Communication of the ACM*, 4(34): 59-63.

37- Lee, W., Kim, Y., Gove, R., and Read, C. 1994. Media Station 5000:Intergating Video and Audio. *IEEE Multimedia*, 2(1): 50-61.

38- Martyn J., and Lain E.1997. *Digital Video Communication.* 1st Ed, Artech House, London.

39- Michael, O., and Peter, S. 1998. *ATM & MPEG-2 Integrated Digital Video into Broadband Networks*. Hewlett-Packard Company, USA.

40- Nick, F., and Susie,W. 1999. An MPEG-2 to H.263 Transcoder. *SPIE Voice, Video, and Data communication conference*, Boston, MA.

41- Olsen, T., and Strand, J., S. 2002. *An improved image processing chain for mobile terminals*. M.Sc. Thesis, Agder University College, Norway.

42 Olivieri, S., Gerard, H., and Luigi, A. 1999. Noise-robust Recursive Motion Estimation for H.263-based Videoconferencing System. *Proc. Int. Workshop on Multimedia Signal Processing*, Copenhagen, 1: 345-350.

43- Po, L., and Ma, W. 1996. A Novel Four-Step Search Algorithm for Fast Block Motion Estimation. *IEEE Trans., on Circuits and System for Video Technology*, 3(6): 313-317.

44- Pennabaker, W.B., and Mitchell, J.L. 1993. *JPEG Image Data Compression Standard*. Van Nostrand Reinhold, New York.

45- Sulaiman, Y., Ali, M. and Arif, S. 2002. An Algorithm For Scene Cut Detection From MPEG-1 Video Streams. *Proc. Of the 2002 International Arab Conference on Information Technology (ACIT' 2002)*. 1: Computer Center- University of Qatar. Qatar.

46- Steinmetz, R. 1994. Data Compression In Multimedia Computing-Standards And Systems. Part I and II. *Journal of Multimedia Systems*. 1:166-172 and 187-204.

47- Srinivasan, R., and Rao, K. 1985. Predictive Coding Based on Efficient Motion Estimation. *IEEE Transaction and Communication*. 33: 888-896.

48- Sangwine, S. J., and Horne, R. E. 1998. *The Color Image Processing*

Handbook. CHAPMAN & HALL, UK.

49- Tanenbaum, A. S. 1996. *Computer Networks.* Prentice-Hall Inc., USA.

50- Unbaugh, S. E., 1998. *Computer Vision and Image Processing: A practical Approach using CVIP Tools.* Prentice Hall, USA.

51- Wayner, P. 1997. *Digital Copyright Protection*. Academic Press. UK.

52- Watkinson J. 1997. *Compression in Video and Audio*. 2nd Ed. Focal press, Britain.

53- Yun Q. and Huifang S. 2000. *Image and Video Compression for Multimedia Engineering*. 1st Ed. CRC Press, USA.

54- Zhn, C. 1997. RTP Payload Format for H.263 video stream. *Intel Corporation*, New York, USA.

55- Zhijun, L., and Nicolas, D. 2002. H.263 Video Transcoding For Spatial Resolution Downscaling. *Proceedings of IEEE International Conference on Information Technology: Coding and Computing (ITCC) 2002.* Canada.

Appendix
MATLAB Source Code of Proposed Techniques

Conjugate Search Algorithm

```
function [Conj,y,x] = Conjucate(SWind,Sblk,BSize)
[Sr,Sc] = size(SWind);
Ycntr = fix(Sr /2);  %Mid of Searching Window (Y-axis)
Xcntr = fix(Sc /2);  %Mid of Searching Window (X-axis)
Blk    = BSize;       %Block Size
x = 0;
y = 0;
ii = 1;
hCost = zeros(Sc/Blk,2);
for i=1:Blk:Sc
    hCost(ii,1)=MADe(Ycntr,i,Blk,Sblk,SWind);
    hCost(ii,2)=i;
    ii=ii+1;
end
M = min(hCost(:,1));
Xcntr = find(hCost(:,1)==M);
x = hCost(Xcntr(1),2);
vCost = zeros(Sr/Blk,2);
ii=1;
for j=1:Blk:Sr
    vCost(ii,1)=MADe(j,x,Blk,Sblk,SWind);
    vCost(ii,2)=j;
    ii = ii+1;
end
N = min(vCost(:,1));
y = find(vCost(:,1)==N);
y = vCost(y(1),2);
%Conj = getblk(SWind,x,y,Blk);
Conj = N;

return

function made=MADe(dy,dx,BlkS,F,G)
S=0;
for i=1:BlkS
    for j=1:BlkS
        S=S+abs(F(i,j)-G(i+dy-1,j+dx-1));
    end
end
made = S/(BlkS^2);
return
```

The Full Search Algorithm

```
function [Ex,y,x] = ExhustiveS(SWind,Sblk,p,q)
[Sr,Sc]=size(SWind);
ii=1;
x   = 0;
y   = 0;
for i=1:q:Sr
    for j=1:p:Sc
        Ex(ii,1)=MADe(i,j,p,q,Sblk,SWind);
        Ex(ii,2)=i;
        Ex(ii,3)=j;
        ii=ii+1;
    end
end;
M = min(min(Ex(:,1)));
Comp = find(Ex(:,1)==M);
y = Ex(Comp(1),2);
x = Ex(Comp(1),3);
Ex= M;
return

function made=MADe(dx,dy,m,n,F,G)
S=0;
for i=1:n
    for j=1:m
        S=S+abs(F(i,j)-G(i+dx-1,j+dy-1));
    end
end
made = S/(m*n);
return
```

The 2-D logarithmic Search Algorithm

```
function [D,y,x] = Two_DLog2(SWind,Sblk,th)
[Sr,Sc] = size(SWind);
Ycntr    = fix(Sr/2);
Xcntr    = fix(Sc/2);
BlkS = size(Sblk,1);

IncP     = -2*BlkS;
Dcs      = 2*BlkS;
Dir = zeros(6,3);
Duc = 0;
for Lop=1:3
    ii = 1;
    for i=1:3
```

```
        Dir(ii,1) =
MADe(Ycntr+IncP,Xcntr,BlkS,Sblk,SWind);         %Center
        Dir(ii,2) = Ycntr+IncP; Dir(ii,3) = Xcntr;
        ii = ii + 1;
        IncP = IncP + Dcs;
    end
    IncP    = -2*BlkS+Duc;
    for j =1:3
        Dir(ii,1) =
MADe(Ycntr,Xcntr+IncP,BlkS,Sblk,SWind);         %Center
        Dir(ii,2) = Ycntr; Dir(ii,3) = Xcntr+IncP;
        ii = ii + 1;
        IncP = IncP + Dcs;
    end
    IncP    = -2*BlkS+Duc;
    D = min(Dir(:,1));
    Comp = find(Dir(:,1)==D);
    Ycntr = Dir(Comp(1),2);
    Xcntr = Dir(Comp(1),3);
    x     = Xcntr;
    y     = Ycntr;
    if abs(Xcntr+IncP)<=16
        if abs(Xcntr+IncP)<16
            Xcntr = 17;
        end
        Dcs = BlkS;
        IncP= -BlkS;
        Duc = Dcs;
    end
    if abs(Ycntr+IncP)<=16
        if abs(Ycntr+IncP)<16
            Ycntr = 17;
        end
        Dcs = BlkS;
        IncP= -BlkS;
        Duc = Dcs;
    end
    if abs(Ycntr-IncP)>=(Sr-16)
        if abs(Ycntr-IncP)==Sr
            Ycntr = Ycntr - (BlkS*2);
        end
        Dcs = BlkS;
        IncP= -BlkS;
        Duc = Dcs;
    end
    if abs(Xcntr-IncP)>=(Sc-24)
        if abs(Xcntr - IncP)==Sc
            Xcntr = Xcntr - (BlkS*2);
        end
        Dcs = BlkS;
        IncP= -BlkS;
```

```
        Duc = Dcs;
    end
end
return

function made=MADe(dy,dx,BlkS,F,G)
S=0;
for i=1:BlkS
    for j=1:BlkS
        S=S+abs(F(i,j)-G(i+dy-1,j+dx-1));
    end
end
made = S/(BlkS^2);
return
```

The Three-step Search Algorithm

```
function [Exx,Ycntr,Xcntr] = ThreeSteps(SWind,Sblk)
[Sr,Sc] = size(SWind);
Ycntr   = fix(Sr/2);
Xcntr   = fix(Sc/2);
BlkS = size(Sblk,1);

IncPi    = -4*BlkS;
IncPj    = -4*BlkS;
inij     = IncPj;
Dcs      = 4*BlkS;
Dir = zeros(8,3);
Comp = 0;
ii = 1;
for Lop=1:3
    for i=1:3
        for j=1:3
Dir(ii,1) =MADe(Ycntr+IncPi,Xcntr+IncPj,BlkS,Sblk,SWind);
%Center
      Dir(ii,2) = Ycntr+IncPi; Dir(ii,3) = Xcntr+IncPj;
            ii = ii + 1;
            IncPj = IncPj + Dcs;
        end
        IncPj    = inij;
        IncPi = IncPi + Dcs;
    end
    Dcs = Dcs - 2*BlkS;
    IncPi = -4;
    IncPj = -4;
    IncPi = (IncPi + (2^Lop))*BlkS;
    IncPj = (IncPj + (2^Lop))*BlkS;
    inij  = IncPj;
```

```
    M = min(Dir(:,1));
    Comp = find(Dir(:,1)==M);
    Ycntr = Dir(Comp(1),2);
    Xcntr = Dir(Comp(1),3);
    Exx = M;
    if Dcs ==0
        IncPi = -BlkS;
        IncPj = IncPi;
        inij  = IncPj;
        Dcs   = BlkS;
    end
    ii = 1;
end
return

function made=MADe(dy,dx,BlkS,F,G)
S=0;
for i=1:BlkS
    for j=1:BlkS
        S=S+abs(F(i,j)-G(i+dy-1,j+dx-1));
    end
end
made = S/(BlkS^2);
return
```

The Thresholding Half-pixel Technique

```
function [a,b,c,d]= Half_Pixel(A,B,C,D,th)
thr = 0;
if th == 1
    thr =round(abs((A+B)-(C+D))/4);
end
a=A;
b=floor(((A+B+1)/2)-thr);
c=floor(((A+C+1)/2)-thr);
d=floor(((A+B+C+D+2)/4)-thr);
return

function fimg = HalfImage(Img,th)
[r,cc]=size(Img);
Lop = (r*cc)/4;
ImP = Img;
if max(max(Img))<=1
    ImP = fix(Img * 255);
end
fimg = zeros(r*cc,1);
ImP = ImP(:);
```

```
jump = 4;
for i=1:Lop
    S = (jump*(i-1))+1;
    E = i*4;
    part = ImP(S:E);
    [a,b,c,d] =
Half_Pixel(part(1),part(2),part(3),part(4),th);
    fimg(S:E) = [a,b,c,d];
end
fig = reshape(fimg,r,cc);
fimg = fig;
```

تحسين فعالية ضغط الصور المتحركة باستخدام تقنيات التخاطب المباشر

أعداد

عمار محمد كامل الجبوري

أشراف

الدكتور محمد بلال الزعبي

الملخــــــص

يمكن تعريف الصور الرقمية على أنها التمثيل المرئي لبيانات رقمية ذات معنى ضمن حـــدود البعــد المكاني.و يطلق على ترتيب عدد محدد من الصور الرقمية بصورة متسلسلة صــور متحركـــة (Animated Images) او مقطع فيديوي (Video). يعرف الفيديو الرقمي على انه كل المعلومات الـــصورية التـــي تخزن وتتناقل في شكل رقمي، على سبيل الذكر، مؤسسات البث التلفزيوني الرقمية. أن أكثر الصور الرقميـــة تحتوي على نسبة عالية من وفرة البيانات وتكرارها حيث يشكل تكرار تلك البيانات عائقا كبيرا إذا ما تم تناقلها عبر وسائل تناقل البيانات المختلفة أمثال الشبكات الموسعة و الإنترنت. لذا اقتضت الحاجة اختزال ما تكـــرر من تلك البيانات بحيث لا يؤثر ذلك على هيئتها العامة في حالة استرجاع ما اختزل منها باستخدام مجموعة من تقنيات الضغط الصوري الفديوي.

تم العمل في هذه الدراسة على تحسين فعالية تقنية H.263 للضغط الفديوي والتي تختص بتطبيقات التخاطـــب المباشر بعيد المدى عبر الإنترنت. تمت دراسة خصائص هذه التقنية وتم التعرف على هيكلها العام، إضافة إلى ذلك تم تحديد العوامل التي تؤثر على كفاءة هذه التقنية مثل عاملي (وقت المستغرق في عملية الضغط، وكفاءة ووضوح الصورة الرقمية) من خلال بناء وتحليل مجموعة من خوارزميات توقع الحركة الصورية مثل (Full search, 2-D logarithmic search, Conjugate search, Three-step search) التي تستخدم في عملية الضغط الفديوي. تقترح هذه الدراسة تطوير تقنيتين جديـــدتين تحـــسن مـــن كفـــاءة تقنيـــة H.263 وهمـــا (Thresholding Half-pixel, OddEven search algorithm)، حيث خصصت الأولى للعمل على تحسين الخصائص الصورية للأطر الفديوية، أما التقنية الثانية فأنها تقلل من الوقت المـــستغرق فـــي عمليـــة الـــضغط الفديوي.

أوضحت نتائج هذه الدراسة بشكل مفصل كفاءة التقنيتين المقتـرحتين مقارنـة بـالطرق المعروفـة والموضحة سابقاً من حيث عامل الوقت، الكفاءة الصورية و نسبة الضغط. استنتجت هذه الدراسـة أن تقليـل عامل الوقت وزيادة نسبة الضغط مع المحافظة على الخصائص الصورية ووضوح الصورة الرقميـة للفيـديو الرقمي يزيد من كفاءة تقنية H.263.