# SCALABLE TRANSCODING FOR DYNAMIC VIDEO RATE ADAPTATION

A thesis submitted
to Kent State University
in partial fulfillment of the requirements
for the degree of Master of Science

by

Darshan Patel

May, 2003

Thesis written by

Patel, Darshan

M.S., Kent State University, 2003

Approved by

________________________________________, Advisor

________________________________________, Chair, Department of Computer Science

________________________________________, Dean, College of Arts and Sciences

**Table of Contents**

## List Of Figures

**List Of Tables**

*Number* *Page*

**Acknowledgements**

I would like to extend my sincere gratitude to my research advisor, Dr. Javed Khan. Without his constant guidance and encouragement; this research would have never been possible. Also, my special thanks to him, for providing me the opportunity and department facilities to carry out this thesis. Whenever I had any problems, complex technical issues or simple word documentation, he always had the solution ready for me. I consider myself fortunate to be his student.

I would also like to convey my thanks to my faculty advisor, Dr. Mikhail Nesterenko, especially for his moral support. I am grateful to him for guiding me plan my coursework from day one.

Thanks to Dr. Arvind Bansal and Dr.Hassan Peyravi for agreeing to be on my thesis committee and for reviewing this thesis.

Last but not the least, thanks to almighty, my family and friends, for always being there to help me take the right decisions.

# CHAPTER 1

## *INTRODUCTION*

### 1.1 Multimedia services

Recent advances in signal processing combined with an increase in network capacity are paving the way for users to enjoy multimedia services wherever they go and on a host of multimedia capable devices. There are various multimedia services such as Video on Demand, Teleconferencing, Digital TV, Distance Learning etc. in which video is the major multimedia component. Let us review instance of their applications with respect to certain scenarios in today's world. As an instance of application for Video on Demand (VOD), there are wireless applications, which allow movie trailers, cartoons, news and sports to be viewed on the cell phone screens. Teleconferencing is becoming very popular multimedia service for various kinds of applications these days. Voice and video conferencing are the popular applications of teleconferencing. Teleconferencing can also be used to deliver live events via satellite to geographically dispersed downlink sites. Distance learning is one other application of teleconferencing in which student and instructor are at distant places. With the help of multi-point video conferencing and other existing technologies, it's possible to provide distance learning with lot of manpower and time being saved. Digital TV is another area of focus, which will provide interactive viewing of TV besides better picture, sound and more compressed data. All of the above applications require video to be delivered over some kind of network. Video contains lot more data as compared to various other multimedia contents. Despite of the huge information associated with video, delivery of digital video through existing networks is becoming increasingly common.

## 1.2 Network and application constraints

In today's world, Internet has become very popular and significant. We can identify from the above discussion that the huge information has to be delivered also through Internet, while considering the video delivery through existing networks. Internet is a huge and heterogeneous network. Internet delivers the content with the help of various kinds of networks such as ATM (Asynchronous Transfer Mode), TCP/IP, WIRELESS, PSTN Public Switched Telephone Network) and etc. All of these networks can support different maximum bandwidths. Hence if the video has to be delivered through these networks, it has to be compressed to fit into the bandwidth requirement of each of these networks. This process of compressing the video to adjust to the given bandwidth of the network is called Bandwidth adaptation and the process of compressing the video is called Bit Rate Scaling.

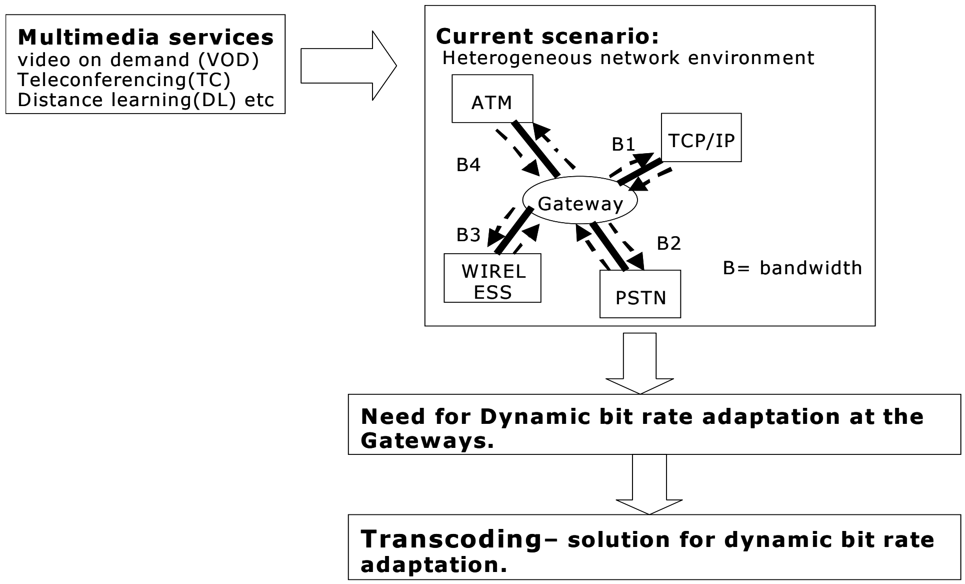



**Fig. 1, Heterogeneous Network environment**

### 1.3 Current Video Compression Standards:

Digital video comes in many different forms, each with its own use. Video formats can be split into three categories based on their output: TV, CD/Internet and streaming. Formats for TV, like DV and MPEG-2, have larger file sizes and processing requirements that limit their use. Internet/CD-ROM formats are designed to offer good quality, while also saving on file sizes. These movies are trimmed down from their full screen size to that of a fraction to make playing manageable. Streaming formats are designed to be small enough so that a movie can be downloaded as it is playing. Usually streaming compromises quality to ensure a complete video can be seen. By my definition, streaming has the capacity to broadcast a live video feed. You can also transfer a saved or "canned" movie using either a streaming or Internet/CD formats. Internet/CD format will offer higher quality, but will typically require a fast data connection and a period of pre-

downloading before the movie can start playing. The most commonly used video codecs are MPEG-2, MPEG-4, Sorenson Video, and DV. These formats cover a wide range of uses we can use.

**Table 1: Comparison between Compression standards**

| **Format** | **Output Rate** | **Comments on application** | **Output** |
|---|---|---|---|
| DV | 3.7 MB/sec | Consumer video used for desktop video, camcorders. Good quality, low processessing requirements | TV |
| MPEG-2 | 400-1300 KB/sec | DVD standard. High quality, high processessing requirements | TV |
| Sorenson Video | Variable | Small File sizes w/ good quality and low processing requirements | Net/CD |
| MPEG-4 | Variable | Small File sizes w/ good quality and low processing requirements. Standard offers compatibility for multiple platforms | Net/CD |
| Real Video | Variable | Mostly used for live streaming. | Net |
| Media Player | Variable | Microsoft's video format. Based on MPEG-4, but not fully compatible | Net/CD |

### *1.3.1 MPEG-2*

The thesis focuses on MPEG2. MPEG-2 is the standard format for DVD movies. DVD size can vary through the use of variable bit rate (VBR) encoding. MPEG-2 is capable of changing its compression to accommodate the current video image. Generally speaking, MPEG-2 compression for DVD runs between 400 KB/sec and 1.3 MB/sec. MPEG-2 can compress at higher and lower rates. MPEG-2 offers higher quality and smaller file sizes through use of intensive compression methods.

### *1.3.2 DV*

DV has become a popular video standard for consumers. DV is used in all digital video cameras and has become a primary method of bringing video into desktop computers. DV is transferred from computers and cameras via FireWire, or as it's officially known as IEEE 1394. FireWire is a high-speed serial bus capable of transferring data up to 50 MB/sec. DV includes a video and audio stream that is transmitted at about 3.7 MB/sec. Unlike MPEG2, DV is not capable of changing its compression to accommodate the current video image.

### *1.3.3 Sorenson Video*

Sorenson Video was first introduced in QuickTime 3. Sorenson Video manages to do three things at once, which is required for movies over the Internet. Sorenson Video allow movies to have relatively good quality with small file sizes and low processing requirements. Unlike MPEG-2, and to a lesser extent DV, Sorenson doesn't require major horsepower to both play and encode the video. For the most part Sorenson is exclusively used by Apple's QuickTime. The format is ideal for Internet and CD-ROM use.

### *1.3.4 MPEG-4*

MPEG-4 has carried quite a bit of hype since it was first introduced. The format promises to create a standard Internet video format. Like Sorenson Video, MPEG-4 offers good quality at smaller file sizes, while also not requiring a significantly large amount of processing power.

MPEG-4's strength is in its transportability, meaning video saved to MPEG-4 should be able to be viewed on any platform.

### *1.3.5 Realvideo, Quicktime for Streaming*

There are numerous formats for streaming content over the Internet. The essence of streaming is not to download, but rather play video in real-time. To do this, some type of streaming server is required, such as the QuickTime Streaming Server or one of RealNetworks' servers. Streaming is similar to saved files, but instead the video can be compressed and transmitted on the fly. Typically you wouldn't want to stream a video unless it's live.1.3.6 Comparison between MPEG-2 and DV.

*1.3.6.1 Playback vs. Encoding*

*As noted above, MPEG-2 and DV have specific processing requirements. This is an important area of video compression in that a computer needs to handle a lot of data in order to both play and encode video.*

When captured, video is encoded, and when played back it is decoded. The DV codec works in real-time so that no special processing is required. Computers do not include DV hardware codecs, but instead rely on software codecs. The software utilized the computer's CPU to processes the data. A computer is capable of playing a DV movie without much trouble in part because modern computers are fast enough, but also the DV format is not too intensive.

Encoding a movie to DV can be a different story as a significant amount of rendering will be required on the CPU's part. This is not true for DV camcorders and converters as they have hardware DV codecs that manage to both encode and decode in real time. For example, when recording a movie on a DV camcorder, the DV codec encodes the video image in real-time to tape. When using a computer, a period of processing is required. There are a couple DV compression cards available to help assist in DV encoding and effects, but for average users these may not really be necessary. 1.3.6.2 MPEG-2 better than DV at lower bit rates.

This is possible because of its high-powered encoding and decoding technology. The trick with digital video always has been is to create something that can viewed without extensive processing time. It wouldn't make much sense to bring home a DVD and need to render it for 5 hours before you could watch it. Set top DVD players have MPEG-2 decoders built in to allow viewing of movies in real-time. Computers on the other hand can now play DVD movies without a dedicated MPEG decoder. This is possible through newer, faster CPU and graphics cards. Like DV and QuickTime, a computer still relies heavily on the computer's graphics card and CPU for software decoding.

Encoding MPEG-2 is usually a very time consuming process. It's so intensive, expensive MPEG-2 encoding hardware is available to allow for faster encoding times. On a pure CPU level, MPEG-2 encoding can take seemingly forever. For example, a 90-minute DVD could take 45 to 75 hours to encode. Apple's DVD Studio Pro combined with multi-processor G4 PowerMacs can drastically cut this rendering time to around 10 hours, but still that's a lot of sitting around. This lengthy encoding time is because of MPEG-2 intensive compression.

The other Internet/CD-ROM compression formats offer much less CPU processing requirements. As illustrated above, playback usually is significantly less complicated than encoding. Any modern computer will be able to play any of the other supported movie formats without much problem. Encoding can be more time consuming, as this is a combination a movie's specifications, compression settings and computer system.

**1.4 Introduction to Video Transcoding:**

Several solutions have focused on adapting the multimedia content to various multimedia capable devices. The adaptation of multimedia content is provided in two basic ways.

I. The first is by storing, managing, selecting, and delivering different versions of the media objects (images, video, audio, graphics and text) that comprise the multimedia presentations.

II. The second is by manipulating the media objects on the fly, by using for example methods for text-to-speech translation, image and video transcoding, media conversion, and summarization. This allows the multimedia content delivery to adapt to the wide diversity of client device capabilities in communication, processing storage and display.

In both of the basic ways, the need for converting a compressed signal into another compressed signal of different format, bit rate or any other attribute occurs. The device that performs such an operation is called a transcoder. Such a device could be placed in the network to help relaying transmissions between these different bit rates or could be used as a pre-processing tool to create the various versions of the media objects that was mentioned ea.

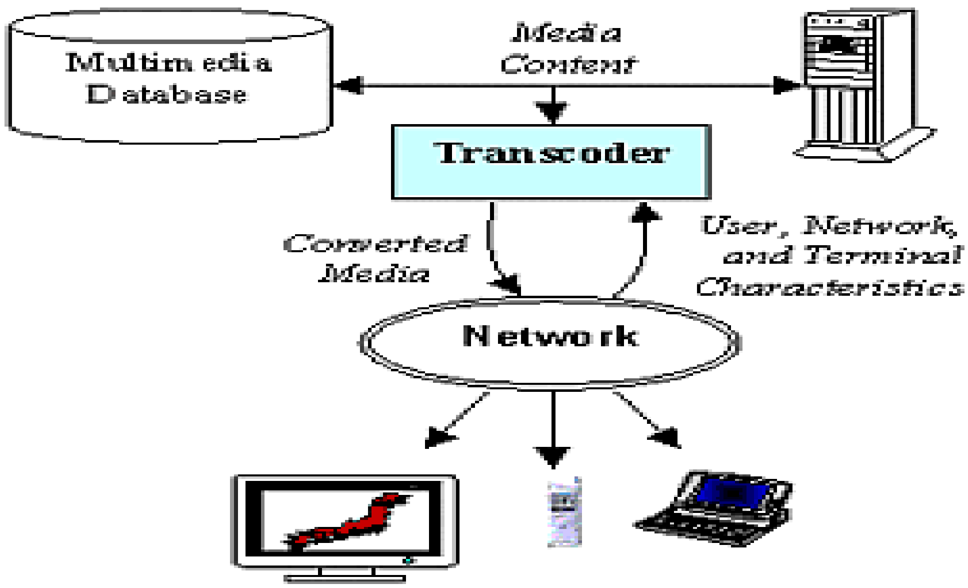



**Fig. 2, General Transcoding**

If the transcoding is used specifically for video it's called video transcoding. Hence video transcoding provides the solution for the dynamic and fine tune adjustments of the video stream during the transmission over the existing networks.

Some of the functions of video transcoding can be video manipulation, format conversion, rate conversion and modifying error resilience. We will discuss video transcoding in detail in later chapters.

**1.5 Statement of problem:**

***Higher bit rate adaptation required:***

The multimedia capable devices include personal computers, televisions and new classes of pervasive computing devices such as PDA, wireless modems, mobile phones, automative computing devices. Each of these terminals may support a variety of different formats. Furthermore, the networks that they are connected to are heterogeneous i.e characterized by different network bandwidth constraints, and the terminals themselves vary in display capabilities, processing power and memory capacity. Therefore, it is required that the digital video transmission should be flexible enough to dynamically adapt to the various terminal capabilities also besides bandwidth constraints. The figure below will help in clarifying the above discussion.

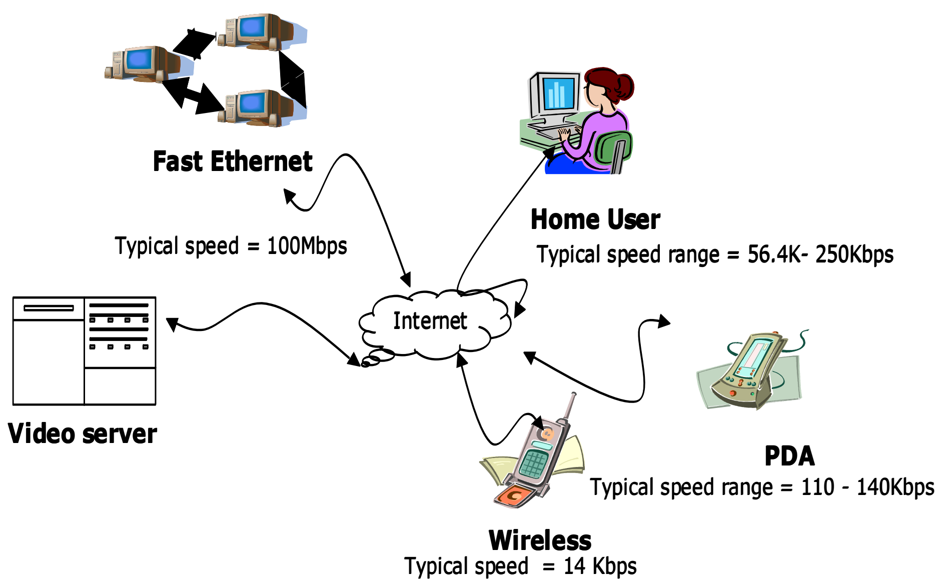



**Fig. 3, Users with different channel capacities**

As we see from the diagram that internet is the best example of a heterogeneous network environment and probably the largest. However, in recent years the asymmetry of Internet has grown enormously. It has been consistently observed that backbone speed is doubling every 9-12 months. While the LAN speed has increased somewhat slower every 3-4 years (about 10 times in 5 years). The slowest part of the network seems to have drastically slower linear rate of advancement (the modem speed have increased about 4 times in last 7 years). A customer connected via high-speed lab typically has 100 Mbps connection, while a home used have connection speed ranging from 56.4K- 250K). With thee advent of handheld and wireless integrated low power devices, interestingly the low range is expected to dip now. Consequently seamless video transmission will require at least 30-60 times rate adaptation ability, much above what is achievable by current popular video coding techniques.

***Need for optimizing the transcoder's speed:***

Transcoder is computationally very expensive. Hence it is required to reduce the computational complexity of the transcoder as much as possible so that its capable of running on the machine with minimum processing power.

Before moving onto the next chapter, which will provide the readers with more insight about the transcoding process, let us discuss some of the current video compression standards used for various kinds of applications.

### 1.6 Scope of thesis

Chapter 1 introduces to the need of bandwidth adaptation for various video based multimedia services that are emerging and identifies the problem of higher bandwidth adaptation. It also introduces to the concept of video transcoding and the various compression standards available today. The comparisons between these standards are made to identify the right kind of video codec for the problem addressed. Chapter 2 discusses the various types of transcoding and their applications. The issues and their probable solutions are also discussed. The major focus is given for the bit rate transcoding since it's the type of transcoding used for the particular kind of problem in focus. It also discusses and analyses some important works in the filed of bit rate transcoding. Chapter 3 gives a detail background regarding the MPEG-2 standard since the techniques investigated are implemented using MPEG-2 ISO/IEC 13818-2 complaint TM5 (Test Model 5). Chapter 4 discusses the issue of transcoder's speed and depicts the technique used to optimize it. It also discusses how the technique is made suitable for dynamic transcoding environment. Chapter 5 discusses the technique investigated and implemented for the higher bit rate adaptation problem. It discusses the need for a joint scheme which uses the traditional and the investigated technique in an optimum way and hence analyses the techniques in terms of error and entropy associated with a video frame at a given bit rate. It also discusses in brief the various transcoding modes possible due to the techniques. Chapter 6 presents the necessary results, which are briefly discussed and analyzed. Chapter 7 includes the conclusions made regarding the area and the techniques studied and the need for the future works. The necessary detailed implementation diagrams of the investigated techniques, the parameter tables and the related reports and papers are depicted in the appendices for the advanced reader who has the knowledge of the MPEG-2 application.

## C H A P T E R 2

## *BACKGROUND ON TRANSCODING*

### 2.1 General Transcoding:

#### *2.1.1 Definition*

The simplest and general definition of transcoding would be to convert the multimedia content with respect to the given parameter. Media contents could be video, audio, image etc. So the various parameters that we could consider for video transcoding would be format, color, bit rate, video dimensions and so on. Let's briefly discuss the architecture of the general transcoder.

#### *2.1.2 Architecture*

The simplest architecture of a video transcoder can be viewed from the diagram below:

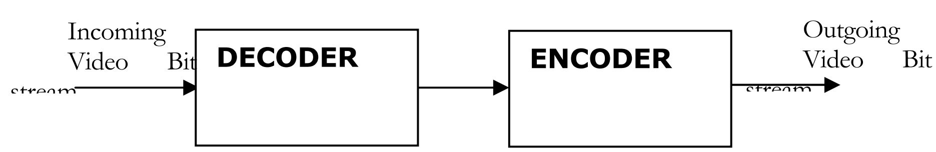



**Fig. 4, simplified transcoder**

Concatenation of a video decoder and a video encoder forms a codec. The codec used for transcoding can be one of the given compression standards discussed in chapter 1.The choice of a particular video compression standard depends on the type of application, the required QoS and many other factors.

Depending on the function of the transcoder, function of the decoder and the encoder changes. But decoder in general decodes the incoming video bitstream. The decoding process involves unwrapping the header formats and extracting the pixel-matrices of the Y,U and V components of the video.

The function of the encoder is to encode the bitstream according to the video coding algorithm used by the video coding standard used for the purpose. But during transcoding the various parameters for encoding can be reset according to the need of the application or network characteristics.

**2.2 Types of transcoding and their applications.**

Below are some of the sample operations of transcoding and the associated possible applications of each of it:

**Table 2: Transcoding operations and applications**

| Transcoding Operation | Applications |
|---|---|
| Bit Rate conversion | Multi-point conferecing<br>Stastical multiplexing<br>Congestion control<br>Adaptation to an available channel bandwidth |
| Format conversion | Conversion to a format suitable for user's decoder (MPEG-1,MPEG-2,MPEG-4,H.261, or H.263) |
| Modifying Error Resilience | modification to appropriate level of current channel condition<br>INTRA/INTER mode selection |
| Manipulation | editing,concatenation, segmentation, or adding data hiding function |

There are quite a few issues in transcoding. Transcoding is a highly computationally expensive process and in today's internet the speed of the delivery of the content is very critical. Transcoding is a lossy process and hence maintaining an acceptable quality becomes another issue. Some of the issues and their probable solutions are given in the following table.

## 2.3 Issues:

**Table 3: Transcoding Issues and possible solutions**

| Issues in transcoding | Possible solutions |
|---|---|
| Reducing Computational complexity or cost | Re-using Motion Vectors/modes.<br>Reducing number of IDCT.<br>Reducing Frame Memories |
| Maximizing quality | Rate Control (Re-quantization)<br>Maintaining accuracy while reducing complexity. |
| Avoiding drift Error during Format conversion | field motion vector→frame motion vector<br>Down Sampling<br>mode change for Error Resilience. |

## 2.4 Bit Rate Transcoding:

The major types of transcoding that are done as compared to other mentioned transcoding functions are:

I. Format conversion

II. Bit rate transcoding. (Bit rate conversion.)

The thesis focuses on the bit rate trancoding. Lets discuss bit rate transcoding in detail.

### *2.4.1 Architecture:*

The focus of this thesis is on bit rate transcoding i.e we used pre-encoded video stream to be supplied to the transcoder, which then produces an output stream which had been coded to the target bit rate required to adapt to that particular channel bandwidth in the network.

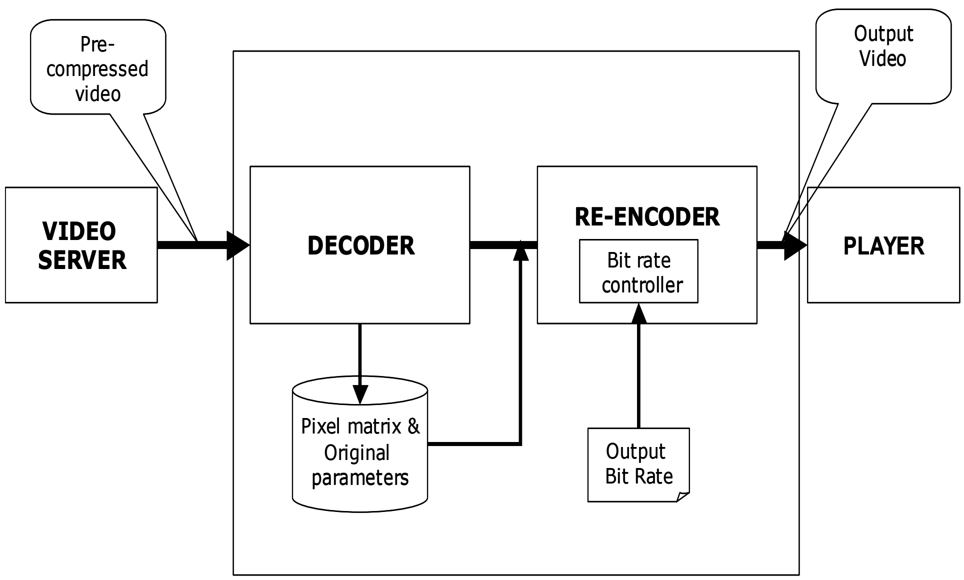



**Fig. 5, Bit rate transcoder**

### *2.4.2 Working:*

As we can see from the figure 5, the bit rate transcoder has a bit rate controller in the Re-encoder(encoder of the transcoder). The original pre-encoded video is served to the trancoder. The decoder first decodes the video to obtain the pixel matrices of Y, U and V components of the video. Since the transcoding is rate transcoding, the other parameters of the original video except the bit rate have to be recoded. These parameters are collected into either a file or a

TCP/IP socket according to the developer's preference. The re-encoder reads the parameters and the pixel matrices to make the settings for re-encoding the video at the given bit rate. As we can see that there is a bit rate controller mechanism in the encoder which will read the given output bit rate and accordingly encode the video. The output bit rate is either supplied by the user or a junction node like a Gateway, router.

These parameters can be looked into detail in the (APPENDIX B)

### *2.4.3 Examples*

Some of the examples of Bit rate transcoding are as follows:

*2.4.3.1.Multi-point video conferencing:*

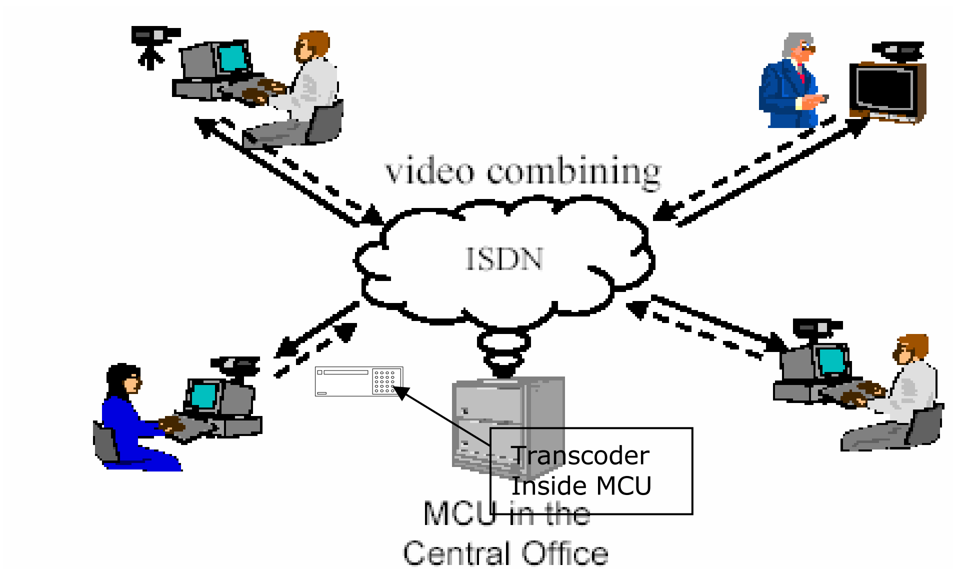



**Fig. 6, Multi-point Video conferencing**

With the rapid growth of video telephony, the need of multipoint video conferencing is also growing. A multipoint videoconference involves three or more conference participants. In continuous presence video conferencing, each conferee can see others in the same window simultaneously. Fig. 3 depicts an application scenario of multiple persons participating in a multipoint videoconference with a computer connected to the central server, referred to as the Multipoint Control Unit (MCU), which coordinates and distributes video and audio streams among multiple participants in a video conference according to the channel bandwidth requirement of each conferee. A video transcoder is included in the MCU to combine the multiple incoming encoded video streams from the various conferees into a single coded video stream and send the re-encoded bit-stream back to each participant over the same channel with the required bit rate and format for decoding and presentation.

*2.4.3.2. Statistical Multiplexing:*

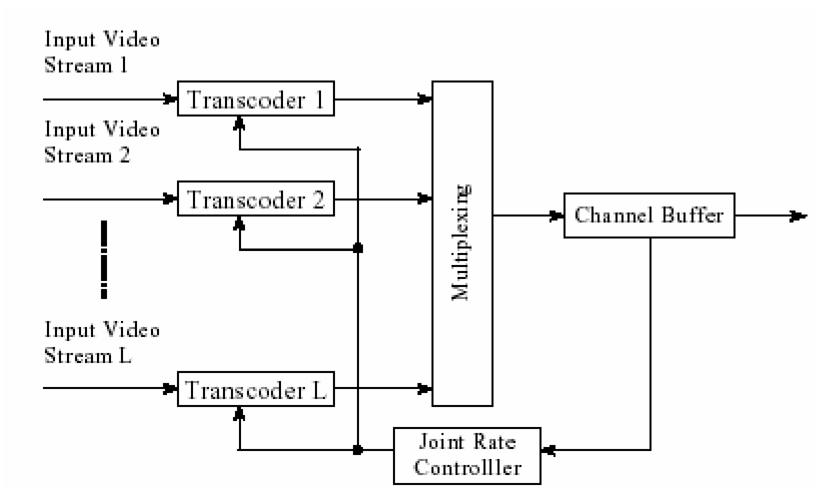



**Fig. 7, Joint transcoding of multiple pre-encoded MPEG video streams.**

Transporting multiple MPEG video streams over a Constant Bit-Rate (CBR) channel has many applications, including video broadcasting over Direct Broadcast Satellite (DBS) and over Digital Subscriber Line (DSL). In these applications, multiple pre-encoded video streams need to be transcoded in order to fit into the CBR channel. For transcoding of pre-encoded video streams, future super frame statistics can be extracted from the input video streams rather easily.

### *2.4.4 Area study*

We have discussed earlier the various operations of trancoding. As stated earlier, our focus is on the bit rate transcoding since we are more interested in the adaptation to the available channel bandwidth. Hence this section of the thesis will present overview of certain work done in the field of bit-rate trancoding and relate them to the objective of the thesis with respect to the bit rate adaptation.

The commonly used techniques used in bit-rate transcoding are:

- Frame dropping
- Discarding high-freq. DCT coeff
- Color suppression
- DCT coeff. Requantization
- Reducing spatial resolution of the video.

The graph below will also give u more insight on the amount of compression that can be achieved using these techniques.

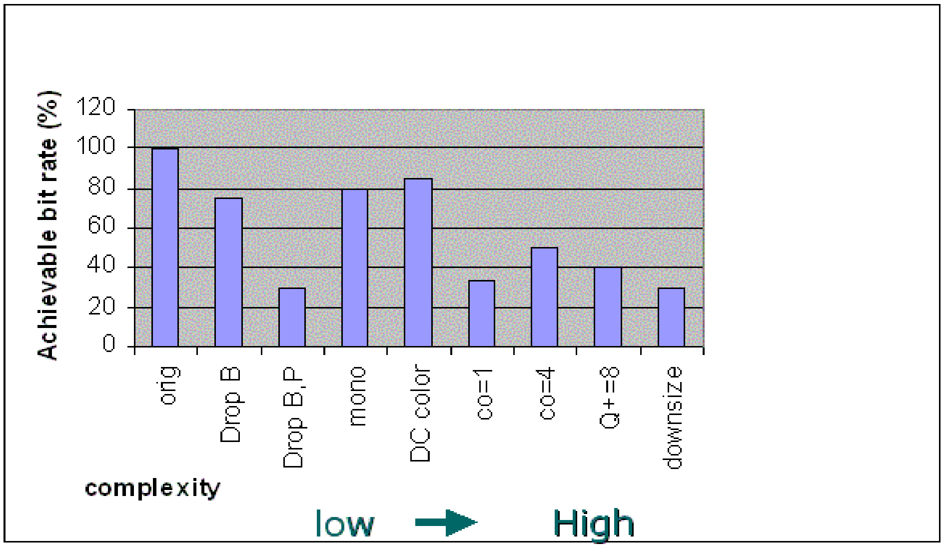



**Fig. 8, Graph for different bit rate reduction techniques**

Lets review some of the related papers.

**1.Video transcoding for universal multimedia access by Niklas Björk and Charilaos Christopoulos.**

The paper presents the basic video transcoding model as well as different ways of improving this model at low bit rates, in particular the H.263 standard. It mainly discusses the issue of adapting video streams to different type of terminals with different terminal capabilities such as screen size, amount of available memory, processing power and type of network access. Two different models for transcoding are examined, rate reduction and resolution reduction. Rate reduction is based on the DCT Requantization. Results will show that the computational complexity of the basic transcoding model can be reduced for each model by, on average, 39% and 23% without significant loss in quality.

**2. Video Transcoding By Reducing Spatial Resolution by *Peng Yin*, *Min Wu* and *Bede Liu* (Princeton Univ.).**

This paper presents a fast approach to derive from an MPEG stream a new MPEG stream with half the spatial resolution. For the downsized video, it's needed to first generate from the original compressed video an improved estimate of the motion vectors. Then the method uses a compressed domain approach with data hiding to produce DCT residues by an open-loop method. The computational complexity is significantly lower than a number of previous approaches. Simulation suggests that our approach produces reasonable image quality

**3. TSFD: Two-Stage Frame Dropping for Scalable Video Transmission over Data Networks by Bing Zheng and Mohammed Atiquzzaman**

Scalable video transmission is used to adjust the rate of video depending on the level of network congestion. Previous studies on scalable video transmission of MPEG over ATM ABR service required major changes in the network protocols, and did not provide methods to determine the ABR connection parameters. This paper presents a new scalable video transmission scheme, which does not require major changes in network protocols. In the proposed scheme, frames are dynamically dropped either by the source or the network depending on the level of network congestion. Scheme is based on encapsulating video frames with priority information, which is used to drop frames by the network during congestion. The scheme requires no major change of network protocols. The effect of MPEG video GOP on the client buffer size has been also analyzed. A general framework has have been developed to determine the client buffer size for no overflow at the client.

**4. Dynamic Frame Skipping in Video Transcoding by Jeng-Neng Hwang, Tzong-Der Wu and Chia-Wen Lin**

The paper investigates the dynamic frame skipping strategy in video transcoding. To speed up the operation, a video transcoder usually reuses the decoded motion vectors to re-encode the video sequences at a lower bit-rate. When frame skipping is used, those motion vectors are no longer used since the motion vectors of the current frame are no longer estimated from the past frame. To reduce the computational complexity of motion vector estimation, a bilinear interpolation approach is developed to overcome this problem.

**5. REAL-TIME TRANSCODING OF MPEG-2 VIDEO BIT STREAMS P N Tudor *and* O H Werner (BBC).**

This paper presents a method for real-time transcoding of MPEG-2 video bit streams that can be applied at different levels of complexity. The proposed method has been developed in the ACTS ATLANTIC project. It is based on the following elements:

- Reuse of motion vectors and coding mode decisions carried in the input bit stream.
- Modeling of the impairments already present in the input.
- Use of bit rate statistics from the input bit stream.

Experimental results confirm that high picture quality can be maintained. Furthermore, the proposed elements and transcoding algorithms are not limited to MPEG-2 and can be extended to a generic transcoding method suitable for the common standards JPEG, H.263, MPEG-1 and MPEG-2 alike.

### *2.4.5 Comparison and summary:*

Let's see the table below for comparison of various bit rate reduction techniques:

**Table 4: Bit rate reduction techniques**

| **Techniques** | **Comments** |
|---|---|
| • Frame dropping,<br>• Discarding high-freq. DCT coeff.<br>• Color suppression.<br>• DCT coeff. Requantization | All of them are Quantization based schemes, which limits the down-scalability to approx. 1:10.<br><br>Used for moderate bit rate reduction, since the quality degrades to unacceptable level at a very high compression.<br><br>Not applicable for video transfers to all kinds of digital devices.<br><br>Mostly used to reduce the computational complexity of the transcoding process. |
| Resolution Reduction | Used for compression factor of 2 and not more since high complexity is associated with the scheme. |
| Transcoding is computationally very expensive. | |

There are several other works in transcoding, but we have depicted some significant and related works above. We can see that all of the above schemes being investigated and implemented in the field of bit rate transcoding focus on the speed of the transcoder. There are schemes, which uses DCT-requantization with fast transcoder architectures. There are schemes, which use frame dropping and skipping. There are schemes, which use resolution reduction for bit rate transcoding.

None of the above works focus on how much bit rate adaptation is required in today's Internet. Most of the works focus on either increasing the speed of the transcoder or improving the image quality.

**C H A P T E R 3**

## *BACKGROUND ON MPEG-2 COMPRESSION*

### 3.1 Introduction to MPEG-2

Recent progress in digital technology has made the widespread use of compressed digital video signals practical. Standardization has been very important in the development of common compression methods to be used in the new services and products that are now possible. This allows the new services to interoperate with each other and encourages the investment needed in integrated circuits to make the technology cheap.

MPEG (Moving Picture Experts Group) was started in 1988 as a working group within ISO/IEC with the aim of defining standards for digital compression of audio-visual signals. MPEG's first project, MPEG-1, was published in 1993 as ISO/IEC 11172 [1]. It is a three-part standard defining audio and video compression coding methods and a multiplexing system for interleaving audio and video data so that they can be played back together. MPEG-1 principally supports video coding up to about 1.5 Mbit/s giving quality similar to VHS and stereo audio at 192 bit/s. It is used in the CD-I and Video-CD systems for storing video and audio on CD-ROM.

During 1990, MPEG recognized the need for a second, related standard for coding video for broadcast formats at higher data rates. The MPEG-2 standard [2] is capable of coding standard-definition television at bit rates from about 3-15 Mbit/s and high-definition television at 15-30 Mbit/s. MPEG-2 extends the stereo audio capabilities of MPEG-1 to multi-channel surround sound coding. MPEG-2 decoders will also decode MPEG-1 bit streams.

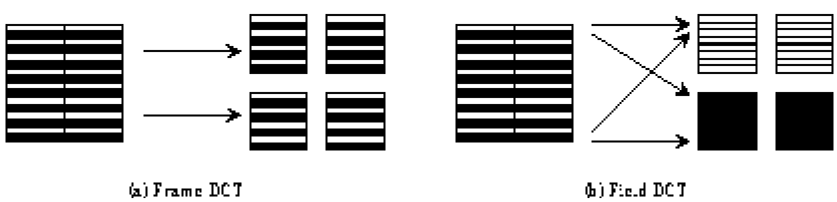
Drafts of the audio, video and systems specifications were completed in November 1993 and the ISO/IEC approval process was completed in November 1994. The final text was published in 1995. MPEG-2 aims to be a *generic* video coding system supporting a diverse range of applications. Different algorithmic 'tools', developed for many applications, have been integrated into the full standard. To implement all the features of the standard in all decoders is unnecessarily complex and a waste of bandwidth, so a small number of subsets of the full standard, known as *profiles* and *levels*, have been defined. A profile is a subset of algorithmic tools and a level identifies a set of constraints on parameter values (such as picture size and bit rate). A decoder, which supports a particular profile and level, is only required to support the corresponding subset of the full standard and set of parameter constraints.

## 3.2 VIDEO FUNDAMENTALS

### *3.2.1 Non-interlaced frames vs. Interlaced frames*

Television services in Europe currently broadcast video at a frame rate of 25 Hz. Each frame consists of two *interlaced* fields, giving a field rate of 50 Hz. The first field of each frame contains only the odd numbered lines of the frame (numbering the top frame line as line 1). The second field contains only the even numbered lines of the frame and is sampled in the video camera 20 ms after the first field. It is important to note that one interlaced frame contains fields from two instants in time. American television is similarly interlaced but with a frame rate of just under 30 Hz.

In video systems other than television, non-interlaced video is commonplace (for example, most computers output non-interlaced video). In non-interlaced video, all the lines of a frame are sampled at the same instant in time. Non-interlaced video is also termed 'progressively scanned' or 'sequentially scanned' video.




**Frame-DCT** = Non-Interlaced , **Field-DCT** = Interlaced

**Fig. 9, Progressive vs. Interlaced video**

### *3.2.2 RGB TO YUV*

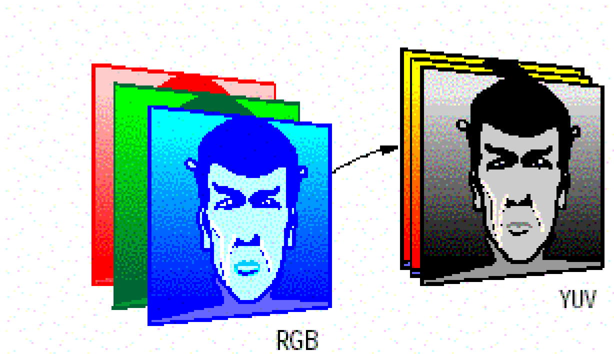



**Fig. 10, RGB to YUV**

RGB to YUV - Conventional floating-point equations

Y = 0.299R 0.587G + 0.114B

U =-0.146 R - 0.288 G + 0.434 B

V = 0.617 R - 0.517 G - 0.100 G

The red, green and blue (RGB) signals coming from a color television camera can be equivalently expressed as luminance (Y) and chrominance (UV) components. The chrominance bandwidth may be reduced relative to the luminance without significantly affecting the picture quality. For standard definition video, CCIR recommendation 601 [3] defines how the component (YUV) video signals can be sampled and digitized to form discrete *pixels*. The terms *4:2:2* and *4:2:0* are often used to describe the sampling structure of the digital picture. 4:2:2 means the chrominance is horizontally sub sampled by a factor of two relative to the luminance; 4:2:0 means the chrominance is horizontally and vertically sub sampled by a factor of two relative to the luminance.

The active region of a digital television frame, sampled according to CCIR recommendation 601, is 720 pixels by 576 lines for a frame rate of 25 Hz. Using 8 bits for each Y, U or V pixel, the uncompressed bit rates for 4:2:2 and 4:2:0 signals are therefore:

4:2:2:720 x 576 x 25 x 8 + 360 x 576 x 25 x (8 + 8) = 166 Mbit/s

4:2:0:720 x 576 x 25 x 8 + 360 x 288 x 25 x (8 + 8) = 124 Mbit/s

MPEG-2 is capable of compressing the bit rate of standard-definition 4:2:0 video down to about 3-15 Mbit/s. At the lower bit rates in this range, the impairments introduced by the MPEG-2 coding and decoding process become increasingly objectionable. For digital terrestrial television broadcasting of standard-definition video, a bit rate of around 6 Mbit/s is thought to be a good compromise between picture quality and transmission bandwidth efficiency.

## 3.3 BIT RATE REDUCTION PRINCIPLES

A bit rate reduction system operates by removing redundant information from the signal at the coder prior to transmission and re-inserting it at the decoder. A coder and decoder pair is referred to as a 'codec'. In video signals, two distinct kinds of redundancy can be identified.3.3.1 Spatial and temporal redundancy:

Pixel values are not independent, but are correlated with their neighbors both within the same frame and across frames. So, to some extent, the value of a pixel is predictable given the values of neighboring pixels.

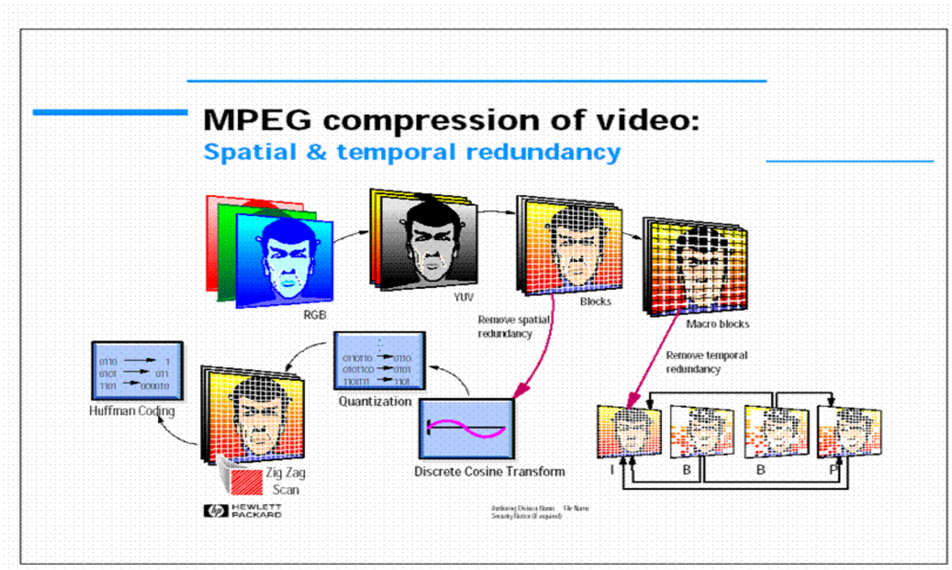



**Fig. 11, MPEG –Spatial and Temporal Redundancy**

Two key techniques employed in an MPEG codec are intra-frame Discrete Cosine Transform (DCT) coding and motion-compensated inter-frame prediction. These techniques have beensuccessfully applied to video bit rate reduction prior to MPEG, notably for 625-line video contribution standards at 34 Mbit/s [5] and video conference systems at bit rates below 2 Mbit/s [6]. Video compression relies on the eye's inability to resolve High Frequency color changes, and the fact that theres a lot of redundancy within each frame and between frames. The Discrete Cosine Transform is used, along with quantization and Huffmann coding; to predict a pixel value from all adjacent pixel values, and minimize the overall bit rate. This generates the Intra-frames (I-frame s). Prediction & motion compensation, predicts the value of pixels in a frame, from the information in adjacent frames. Audio compression makes use of the fact that, high power tones tend to blot out lower power adjacent tones. So if you can't hear it, don't transmit it.

### *3.3.2 Intra-frame DCT coding*

**DCT** [7]**:** A two-dimensional DCT is performed on small blocks (8 pixels by 8 lines) of each component of the picture to produce blocks of DCT coefficients (Fig. 1). The magnitude of each DCT coefficient indicates the contribution of a particular combination of horizontal and vertical spatial frequencies to the original picture block. The coefficient corresponding to zero horizontal and vertical frequency is called the DC coefficient.

The *NxN* two-dimensional DCT is defined as:

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

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

The inverse DCT (IDCT) is defined as:

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

where *x,y* are spatial co-ordinates in the image block
*u,v* are co-ordinates in the DCT coefficient block

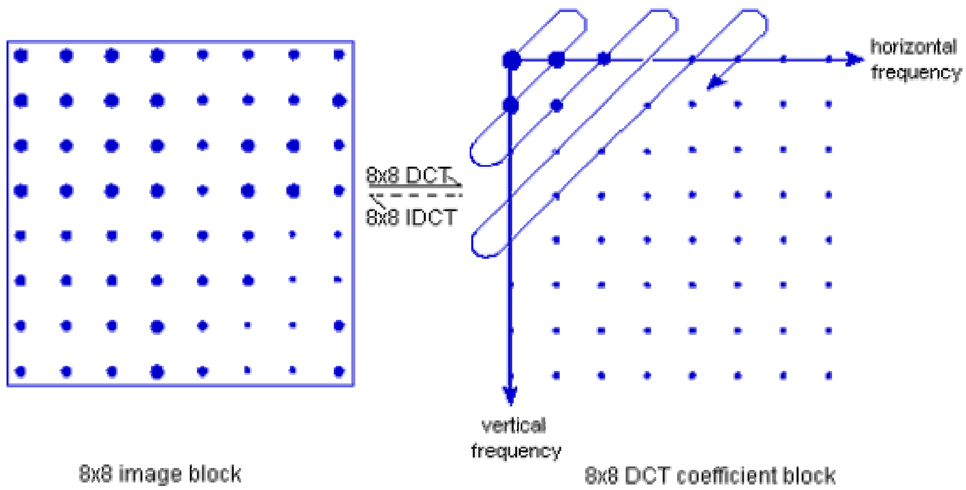



**Fig. 12, The discrete cosine transform (DCT).**

Pixel value and DCT coefficient magnitude are represented by dot size.

The DCT doesn't directly reduce the number of bits required to represent the block. In fact for an 8x8 block of 8 bit pixels, the DCT produces an 8x8 block of 11 bit coefficients (the range of coefficient values is larger than the range of pixel values.) The reduction in the number of bits follows from the observation that, for typical blocks from natural images, the distribution of coefficients is non-uniform. The transform tends to concentrate the energy into the low-frequency coefficients and many of the other coefficients are near zero. The bit rate reduction is achieved by not transmitting the near-zero coefficients and by quantizing and coding the remaining coefficients as described below. The non-uniform coefficient distribution is a result of the spatial redundancy present in the original image block.

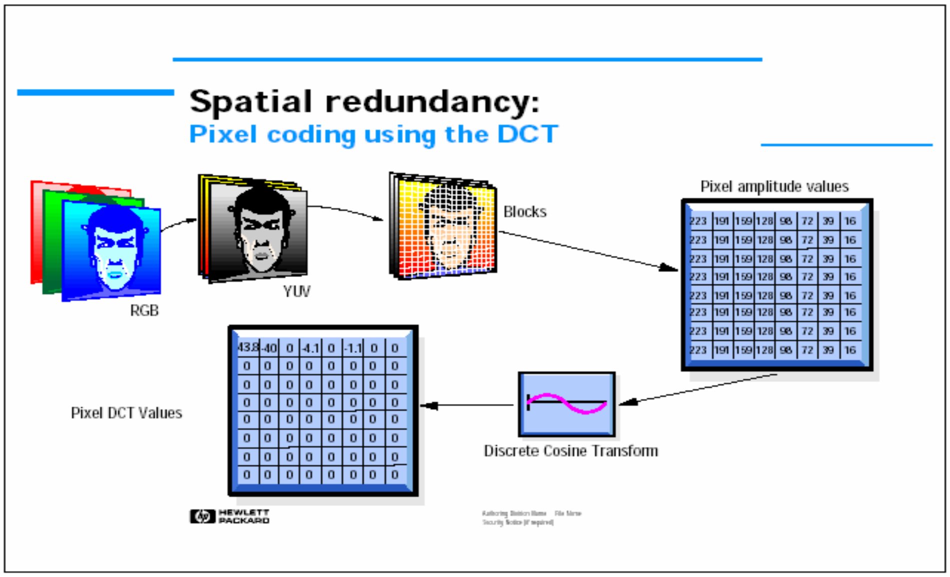



**Fig. 13, Pixel coding using DCT in achieving spatial redundancy**

The first stage is to create an I-frame; subsequent frames in a group of frames will be predicted from this frame.

As the eye is insensitive to HF color changes, we convert the R,G,B signal into a luminance (how bright the picture is) and two color difference signals. We can remove more U, V information than Y.

Each pixel DCT is calculated from all other pixel values, so taking 8x8 blocks reduces the processing time. The top left pixel in a block is taken as the dc datum for the block. DCT's to the right of the datum are increasingly higher horizontal spatial freqs. DCT's below are higher vertical spatial frequencies. Using an Inverse DCT we could reconstruct each pixel's value in the 8x8 block. The DCT is a lossless and reversible process. Its the next stage which introduces compression. Note that the smaller the difference between one pixel and its adjacent pixels, the smaller its DCT value. In the example shown, a greyscales 8x8 pixel values are reduced to one row of DCT's. With all other values going to zero

### *3.3.3 Quantization:*

The function of the coder is to transmit the DCT block to the decoder, in a bit rate efficient manner, so that it can perform the inverse transform to reconstruct the image. It has been observed that the numerical precision of the DCT coefficients may be reduced while still maintaining good image quality at the decoder. Quantization is used to reduce the number of possible values to be transmitted, reducing the required number of bits

The degree of quantization applied to each coefficient is weighted according to the visibility of the resulting quantization noise to a human observer. In practice, this results in the high-frequency coefficients being more coarsely quantized than the low-frequency coefficients. Note that the quantization noise introduced by the coder is not reversible in the decoder, making the coding and decoding process 'lossy'.

### *3.3.4 Coding:*

The serialization and coding of the quantized DCT coefficients exploits the likely clustering of energy into the low-frequency coefficients and the frequent occurrence of zero-value coefficients. The block is scanned in a diagonal **zigzag pattern** starting at the DC coefficient to produce a list of quantized coefficient values, ordered according to the scan pattern.

The list of values produced by scanning is entropy coded using a **variable-length code (VLC).** Each VLC code word denotes a run of zeros followed by a non-zero coefficient of a particular level. VLC coding recognizes that short runs of zeros are more likely than long ones and small coefficients are more likely than large ones. The VLC allocates code words, which have different lengths depending upon the probability with which they are expected to occur. To enable the decoder to distinguish where one code ends and the next begins; the VLC has the property that no complete code is a prefix of any other.

Figure 12. shows the zigzag scanning process, using the scan pattern common to both MPEG-1 and MPEG-2. MPEG-2 has an additional 'alternate' scan pattern intended for scanning the quantized coefficients resulting from interlaced source pictures.

To illustrate the variable-length coding process, consider the following example list of values produced by scanning the quantized coefficients from a transformed block:

12, 6, 6, 0, 4, 3, 0, 0, 0...0

The first step is to group the values into runs of (zero or more) zeros followed by a non-zero value. Additionally, the final run of zeros is replaced with an end of block (EOB) marker. Using parentheses to show the groups, this gives:

(12), (6), (6), (0, 4), (3) EOB

The second step is to generate the variable length code words corresponding to each group (a run of zeros followed by a non-zero value) and the EOB marker. Table 5 shows an extract of the DCT coefficient VLC table common to both MPEG-1 and MPEG-2. MPEG-2 has an additional 'intra' VLC optimized for coding intra blocks. Using the variable length code from Table 1 and adding spaces and commas for readability, the final coded representation of the example block is:

0000 0000 1101 00, 0010 0001 0, 0010 0001 0, 0000 0011 000, 0010 10, 10

**Table 5: Extract from the MPEG-2 DCT coefficient VLC table.**

| Length of run of zeros | Value of non-zero coefficient | Variable-length codeword |
|---|---|---|
| 0 | 12 | 0000 0000 1101 00 |
| 0 | 6 | 0010 0001 0 |
| 1 | 4 | 0000 0011 000 |
| 0 | 3 | 0010 10 |
| EOB | - | 10 |

The higher the DCT frequency, the higher the Quant Matrix value its divided by. This makes many coeficientss go to zero. The fixed value scale factor reduces even more of the DCT's to

zero. The next stage is to increase the number of zero's in the run of bits into the entropy coder. This is done by zig-zag scanning the 8x8 pixel block DCT values and helps the entropy coder do its job. Entropy coding essentially sizes coeficients by how often they occur. The more a coeficient occurs, the smaller a binary value its given. Since in any frame your going to get a large number of identical 8x8 blocks, your reducing the overall binary data rate. To summarize then, quantization makes many higher frequency DCT values go to zero. Entropy coding removes duplication of DCT's, assigning each DCT position with a pointer to its value. This all has a cost. Thats shown in the pictures above: the upper picture is unquantized, the lower one quantized. The process described can be summarized with the help of the figure below:

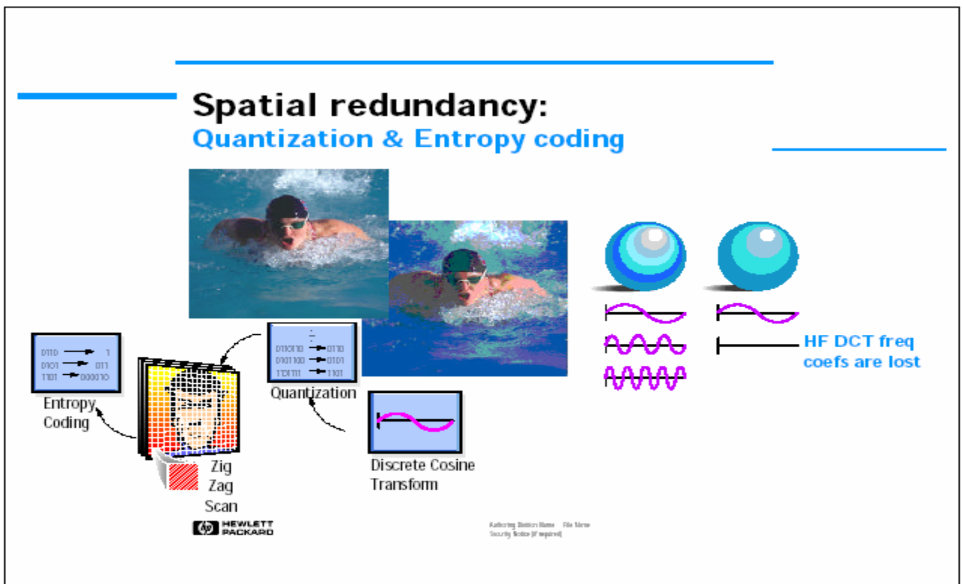



**Fig. 14, Quantization and Entropy coding for spatial redundancy**

### *3.3.5 Motion-compensated inter-frame prediction*

This technique exploits temporal redundancy by attempting to predict the frame to be coded from a previous 'reference' frame. The prediction cannot be based on a source picture because the prediction has to be repeatable in the decoder, where the source pictures are not available (the decoded pictures are not identical to the source pictures because the bit rate reduction process introduces small distortions into the decoded picture.) Consequently, the coder contains a local decoder, which reconstructs pictures exactly as they would be in the decoder, from which predictions can be formed.

The simplest inter-frame prediction of the block being coded is that which takes the co-sited (i.e. the same spatial position) block from the reference picture. Naturally this makes a good prediction for stationary regions of the image, but is poor in moving areas. A more sophisticated method, known as motion-compensated inter-frame prediction, is to offset any translational motion which has occurred between the block being coded and the reference frame and to use a shifted block from the reference frame as the prediction.

One method of determining the motion that has occurred between the block being coded and the reference frame is a 'block-matching' search in which a large number of trial offsets are tested by the coder using the luminance component of the picture. The 'best' offset is selected on the basis of minimum error between the block being coded and the prediction.

The bit rate overhead of using motion-compensated prediction is the need to convey the motion vectors required to predict each block to the decoder. For example, using MPEG-2 to compress standard-definition video to 6 Mbit/s, the motion vector overhead could account for about 2 Mbit/s during a picture making heavy use of motion-compensated prediction.

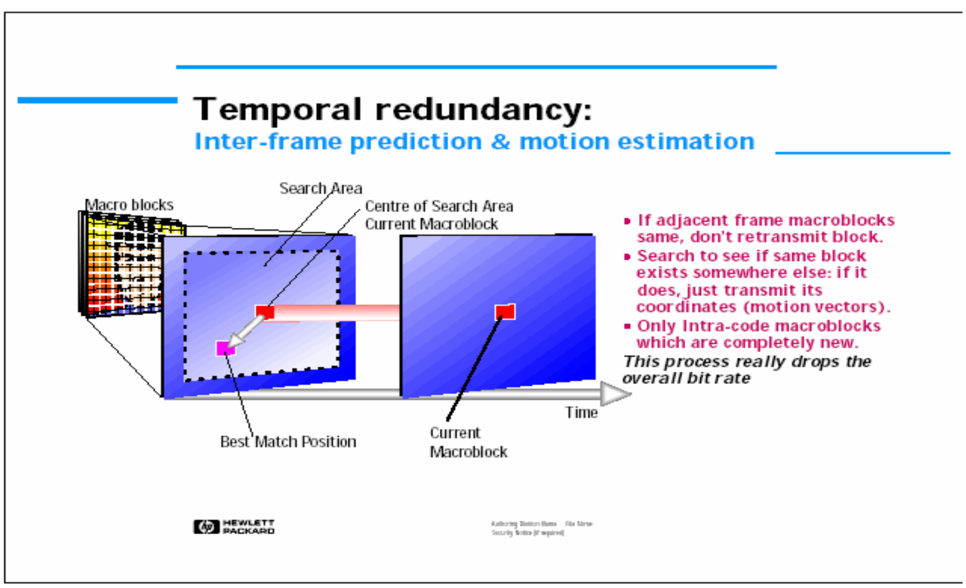



**Fig. 15, Inter-frame Prediction and motion estimation**

This is the real bit rate reduction kicks in. As we'll cover in the next slide, there are three different frame types. By just doing spatial redundancy on a frame you create an I frame. This has all the information necessary to decode the picture. The next stage is to look at the next frame to this and see how similar it is. You can do three things to minimize this frames bit rate. Firstly, look to see if the macroblock in the same position in the next frame hasn't change. If it hasn't, don't do any coding, just transmit that its the same. The next stage is to search around in the I-frame and see if this macro-block exists, but its in a different place. If so transmit motion vectors for its old location. Only if its completely new, do you go for the complete intra-coding process. This really reduces the overall bit rate from frame to frame. But note if you kept predicting each frame from the last, it would only take a little error, and the whole process would fast start to unravel. That's why there are three different frame types, and a specific frame transmit process.

## 3.4 MPEG-2 DETAILS

### *3.4.1 Codec structure*

In an MPEG-2 system, the DCT and motion-compensated interframe prediction are combined, as shown in Fig. 2. The coder subtracts the motion-compensated prediction from the source picture to form a 'prediction error' picture. The prediction error is transformed with the DCT, the coefficients are quantized and these quantized values coded using a VLC. The coded luminance and chrominance prediction error is combined with 'side information' required by the decoder, such as motion vectors and synchronizing information, and formed a bitstream for transmission. Fig. 3 shows an outline of the MPEG-2 video bitstream

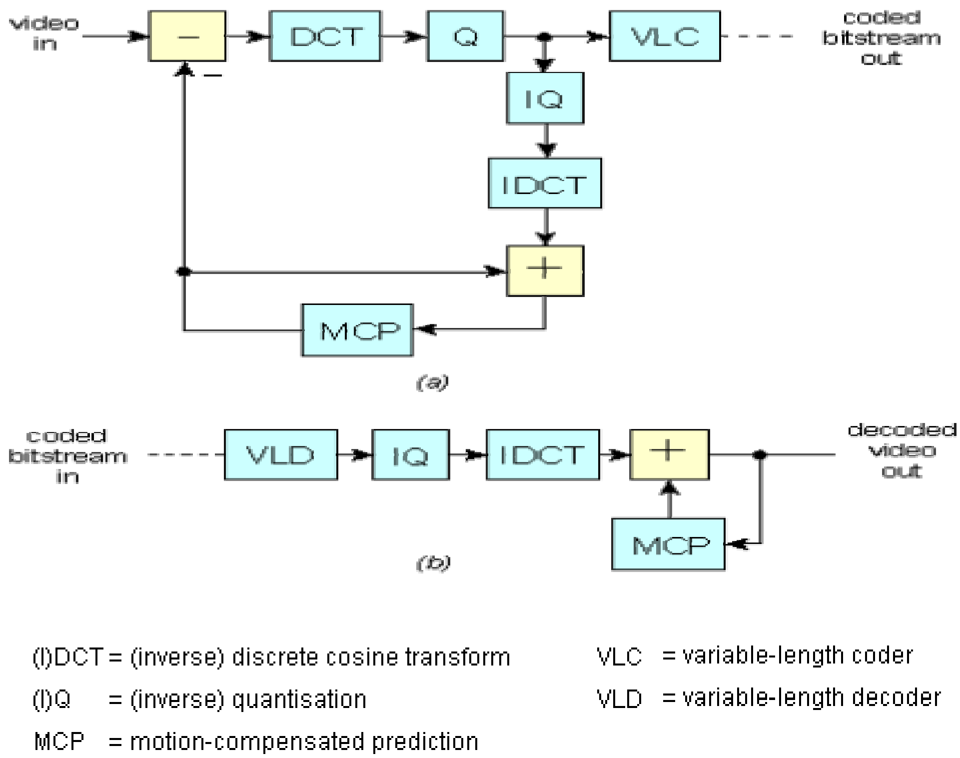



**Fig. 16, (a) Motion-compensated DCT coder; (b) motion compensated DCT decoder.**

In the decoder, the quantized DCT coefficients are reconstructed and inverse transformed to produce the prediction error. This is added to the motion-compensated prediction generated from previously decoded pictures to produce the decoded output.

In an MPEG-2 codec, the motion-compensated predictor shown in Fig. 2 supports many methods for generating a prediction. For example, the block may be 'forward predicted' from a previous picture, 'backward predicted' from a future picture, or 'bidirectionally predicted' by averaging a forward and backward prediction. The method used to predict the block may change from one block to the next. Additionally, the two fields within a block may be predicted separately with their own motion vector, or together using a common motion vector. Another option is to make a zero-value prediction, such that the source image block rather than the prediction error block is DCT coded. For each block to be coded, the coder chooses between these prediction modes, trying to maximize the decoded picture quality within the constraints of the bit rate. The choice of prediction mode is transmitted to the decoder, with the prediction error, so that it may regenerate the correct prediction.

Before moving on to the types of picture, lets look at the layered MPEG-2 structure in terms of buiding an elementary MPEG video stream.

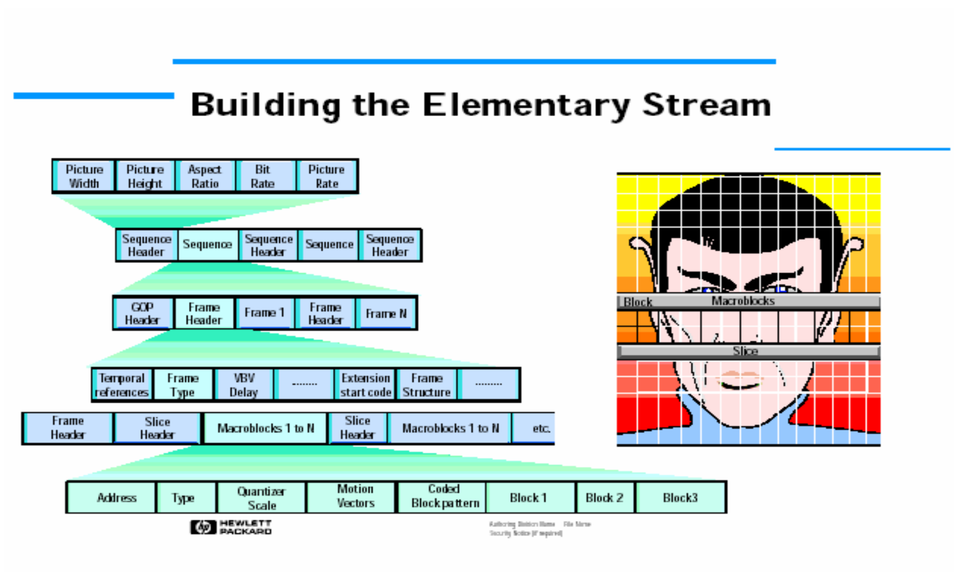



**Fig. 17, Outline of MPEG-2 video bitstream structure (shown bottom up)**

The figure 17 shows how the actual blocks, slices, frames etc. are all put together to form the elementary stream. Along with the actual picture data, header information is required to reconstruct the I, B, P frames. This header structure is shown. Each slice will contain a header detailing its contents & location. Each frame will have a header, and each group of I, B, P frames, known as a Group Of Pictures (GOP) will have a header. The next stage is to take this ES and convert it into something that can be transmitted and decoded at the other end. At this stage, the elementary stream is a continual stream of encoded video frames. Though all the data required to reconstuct frames exists here. No timing information or systems data is contained Thats the job of the MPEG-2 multiplexer First a few words on what we do with the audio signal associated with the video .

### *3.4.2 Picture types*

In MPEG-2, three 'picture types' are defined. The picture type defines which prediction modes may be used to code each block.

'Intra' pictures (I-pictures) are coded without reference to other pictures. Moderate compression is achieved by reducing spatial redundancy, but not temporal redundancy. They can be used periodically to provide access points in the bitstream where decoding can begin.

'Predictive' pictures (P-pictures) can use the previous I- or P-picture for motion compensation and may be used as a reference for further prediction. Each block in a P-picture can either be predicted or intra-coded. By reducing spatial and temporal redundancy, P-pictures offer increased compression compared to I-pictures.

'Bidirectionally-predictive' pictures (B-pictures) can use the previous and next I- or P-pictures for motion-compensation, and offer the highest degree of compression. Each block in a B-picture can be forward, backward or bidirectionally predicted or intra-coded. To enable backward prediction from a future frame, the coder reorders the pictures from natural 'display' order to 'bitstream' order so that the B-picture is transmitted after the previous and next pictures it references. This introduces a reordering delay dependent on the number of consecutive B-pictures.

The different picture types typically occur in a repeating sequence, termed a 'Group of Pictures' or GOP. A typical GOP in display order is:

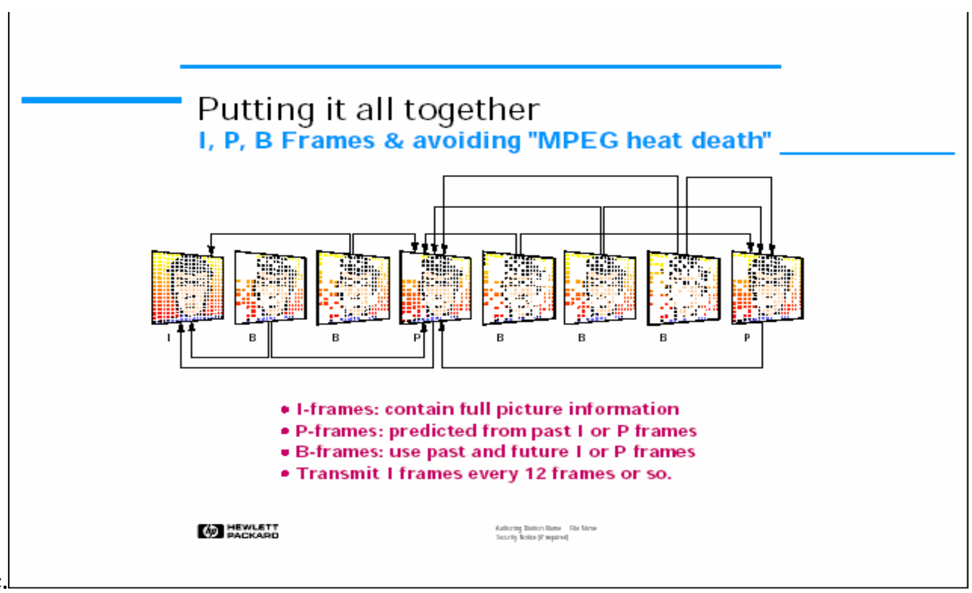
$B_1$ $B_2$ $I_3$ $B_4$ $B_5$ $P_6$ $B_7$ $B_8$ $P_9$ $B_{10}$ $B_{11}$ $P_{12}$

The corresponding bitstream order is:

$I_3$ $B_1$ $B_2$ $P_6$ $B_4$ $B_5$ $P_9$ $B_7$ $B_8$ $P_{12}$ $B_{10}$ $B_{11}$

A regular GOP structure can be described with two parameters: *N*, which is the number of pictures in the GOP, and *M*, which is the spacing of P-pictures. The GOP given here is described as *N*=12 and *M*=3. MPEG-2 does not insist on a regular GOP structure. For example, a P-picture following a shot-change may be badly predicted since the reference picture for prediction is completely different from the picture being predicted. Thus, it may be beneficial to code it as an I-picture instead.

For a given decoded picture quality, coding using each picture type produces a different number of bits. In a typical example sequence, a coded I-picture was three times larger than a coded P-picture, which was itself 50% larger than a coded B-




picture.

**Fig. 18, (I, P and B pictures) in MPEG to avoid heat death**

The Intra Frames contain full picture information. These are your lifeline, if errors occur, or the decoder loses a frame. Without periodic transmission of these the whole process falls apart. But the I-frames are the least compressed. Predicted (P) Frames are predicted from past I, or P frames, Bi-directional predicted frames offer the greatest compression and use past and future I & P frames for motion compensation. But they are the most sensitive to errors. The encoder will cycle through each frame and decide whether to do I, P, or B coding. The order will depend on the application. But roughly every twelve frames, an I-frame is created. If the encoder didn't do this, any small errors would build up and the MPEG compressor would rapidly descend into an electronic form of Entropic "heat death".

### *3.4.3 Buffer control*

By removing much of the redundancy from the source images, the coder outputs a variable bit rate. The bit rate depends on the complexity and predictability of the source picture and the effectiveness of the motion-compensated prediction.

For many applications, the bitstream must be carried in a fixed bit rate channel. In these cases, a buffer store is placed between the coder and the channel. The buffer is filled at a variable rate by the coder, and emptied at a constant rate by the channel. To prevent the buffer from under- or overflowing, a feedback mechanism acts to adjust the average coded bit rate as a function of the buffer fullness. For example, increasing the degree of quantizationapplied to the DCT coefficients may lower the average coded bit rate. This reduces the number of bits generated by the variable-length coding, but increases distortion in the decoded image. The decoder must also have a buffer between the channel and the variable rate input to the decoding process. The size of the buffers in the coder and decoder must be the same.

MPEG-2 defines the maximum decoder (and hence coder) buffer size, although the coder may choose to use only part of this. The delay through the coder and decoder buffer is equal to the buffer size divided by the channel bit rate. For example, an MPEG-2 coder operating at

6 Mbit/s with a buffer size of 1.8 Mbits would have a total delay through the coder and decoder buffers of around 300 ms. Reducing the buffer size will reduce the delay, but may affect picture quality if the buffer becomes too small to accommodate the variation in bit rate from the coder VLC.

### 3.5 Rate Control and Adaptive Quantization:

This section describes the procedure for controlling the bit-rate of the Test Model by adapting the macroblock quantization parameter ***quantizer_scale***. The algorithm works in three-steps

**1 Target bit allocation**: this step estimates the number of bits available to code the next picture. It is performed before coding the picture.

**2 Rate control**: by means of a "virtual buffer", this step sets the reference value of the quantization parameter for each macroblock.

**3 Adaptive quantization**: this step modulates the reference value of the quantization parameter according to the spatial activity in the macroblock to derive the value of the quantization parameter, mquant that is used to quantize the macroblock.

***Step 1 - Bit Allocation***

Complexity estimation

After a picture of a certain type (I, P, or B) is encoded, the respective "global complexity measure" (Xi, Xp, or Xb) is updated as:

$$X_i = S_i Q_i,$$
$$X_p = S_p Q_p,$$
$$X_b = S_b Q_b,$$

where Si, Sp, Sb are the number of bits generated by encoding this picture and Qi, Qp and Qb are the average quantization parameter computed by averaging the actual quantization values used during the encoding of the all the macroblocks, including the skipped macroblocks.

Initial values

Xi = (160 * **bit_rate**) / 115

Xp = (60 * **bit_rate)** / 115

Xb = (42 * **bit_rate**) / 115

**bit_rate** is measured in bits/s.

Picture Target Setting

The target number of bits for the next picture in the Group of pictures (Ti, Tp, or Tb) is computed as:

$$T_i = \max\left\{ \frac{R}{\left(1 + \frac{N_p X_p}{X_i X_p} + \frac{N_b X_b}{X_i K_b}\right)}, \frac{bit\_rate}{8 \times picture\_rate} \right\}$$

$$T_p = \max\left\{\frac{R}{\left(N_p + \frac{N_b K_p X_b}{K_b X_p}\right)}, \frac{bit\_rate}{8 \times picture\_rate}\right\}$$

$$T_b = \max\left\{\frac{R}{\left(N_b + \frac{N_p K_b X_p}{K_p X_b}\right)}, \frac{bit\_rate}{8 \times picture\_rate}\right\}$$

Where:

Kp and Kb are "universal" constants dependent on the quantization matrices.R is the remaining number of bits assigned to the GROUP OF PICTURES. R is updated as follows:

After encoding a picture , R = R - Si,p,b

Where is Si,p,b is the number of bits generated in the picture just encoded (picture type is I, P or B).

Before encoding the first picture in a GROUP OF PICTURES (an I-picture):

R = G + R

G = **bit_rate** * N / **picture_rate**

N is the number of pictures in the GROUP OF PICTURES.

At the start of the sequence R = 0.

Np and Nb are the number of P-pictures and B-pictures remaining in the current GROUP OF PICTURES in the encoding order.

**Table 6. - Example of Remaining pictures in GOP at frame 7**

| I | B | B | P | B | B | P | B | B | P | B | B | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | R-bits | | | | | | |
| | | | | | | Np = 3 | | | | | | |
| | | | | | | Nb = 4 | | | | | | |

### *Step 2 - Rate Control*

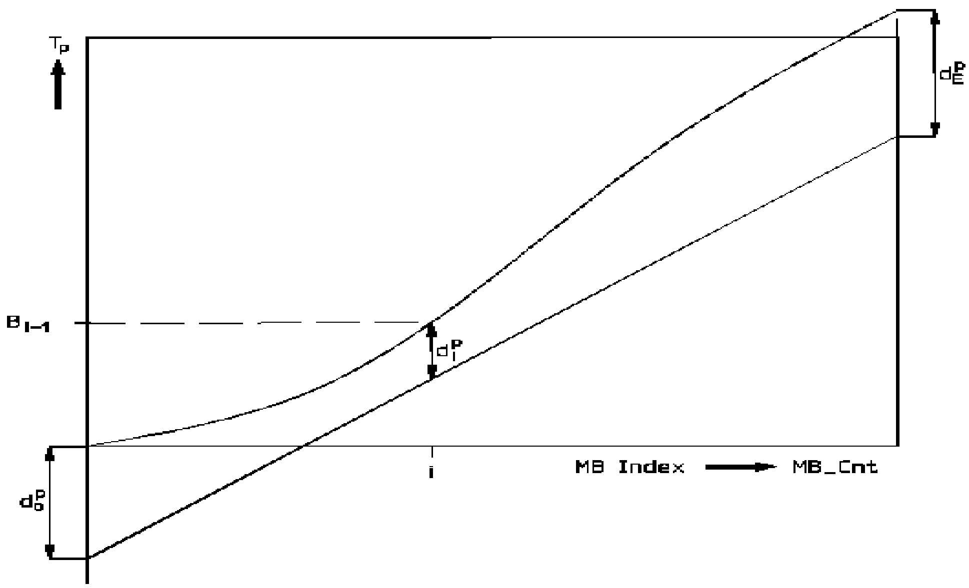



**Fig. 19, Rate Control for P-pictures**

Before encoding macroblock j (j >= 1), compute the fullness of the appropriate virtual buffer:

$$d_j^i = d_0^i + B_{j-1} - \left( \frac{T_i \times (j-1)}{MB\_cnt} \right)$$

or

$$d_j^p = d_0^p + B_{j-1} - \left(\frac{T_p \times (j-1)}{MB\_cnt}\right)$$

or

$$d_j^b = d_0^b + B_{j-1} - \left(\frac{T_b \times (j-1)}{MB\_cnt}\right)$$

…depending on the picture type.

Where

$d_0^i, d_0^p, d_0^p$ are initial full nesses of virtual buffers - one for each picture type.

Bj is the number of bits generated by encoding all macroblocks in the picture up to and including j.

MB_cnt is the number of macroblocks in the picture.

$d_j^i, d_j^p, d_j^p$ are the fullnesses of virtual buffers at macroblock j- one for each picture type.The final fullness of the virtual buffer ($d_j^i, d_j^p, d_j^p$ : j = MB_cnt) is used as $d_0^i, d_0^p, d_0^p$ for encoding the next picture of the same type.

Next compute the reference quantization parameter Qj for macroblock j as follows:

$$Q_j = \left(\frac{d_j \times 31}{r}\right)$$

where the “reaction parameter” r is given by

$$r = 2 \times \frac{bit\_rate}{picture\_rate}$$

$$d_0^i = 10 \times \frac{r}{31}$$
$$d_0^p = K_p \times d_0^i$$
$$d_0^b = Kb_b \times d_0^i$$

and dj is the fullness of the appropriate virtual buffer.The initial value for the virtual buffer fullness is:

Step 3 - Adaptive Quantization

Compute a spatial activity measure for the macroblock j from the four luminance frame-organized sub-blocks (n=1..4) <u>and</u> the four luminance field-organized sub-blocks (n=5..8) using the intra (i.e. original) pixel values:

$$act_j = 1 + \min(vblk_1, vblk_2, \ldots, vblk_8)$$

where $vblk_n = \frac{1}{64} \times \sum_{k=1}^{64} \left(P_k^n - P\_mean_n\right)^2$

and

$$P\_mean_n = \frac{1}{64} \times \sum_{k=1}^{64} P_k^n$$

and Pk are the sample values in the n-th original 8*8 block.

Normalize actj:

$$N\_act_j = \frac{\left(2 \times act_j\right) + avg\_act}{act_j + \left(2 \times avg\_act\right)}$$

avg_act is the average value of actj the last picture to be encoded. On the first picture, avg_act = 400.

Obtain mquantj as:

$$mquant_j = Q_j \times N\_act_j$$

where Qj is the reference quantization parameter obtained in step 2. The final value of mquantj is clipped to the range [1..31] and is used and coded as described in either the slice or macroblock layer.

***Known Limitations***

- Step 1 does not handle scene changes efficiently.
- A wrong value of avg_act is used in step 3 after a scene change.
- VBV compliance is not guaranteed.

**3.6 Profiles and levels**

MPEG-2 video is an extension of MPEG-1 video. MPEG-1 was targeted at coding progressively scanned video at bit rates up to about 1.5 Mbit/s. MPEG-2 provides extra algorithmic 'tools' for efficiently coding interlaced video and supports a wide range of bit rates. MPEG-2 also provides tools for 'scalable' coding where useful video can be reconstructed from pieces of the total bitstream. The total bitstream may be structured in layers, starting with a base layer (that can be decoded by itself) and adding refinement layers to reduce quantization distortion or improve resolution.

A small number of subsets of the complete MPEG-2 tool kit have been defined, known as profiles and levels. A profile is a subset of algorithmic tools and a level identifies a set of constraints on parameter values (such as picture size or bit rate). The profiles and levels defined to date fit together such that a higher profile or level is superset of a lower one. A decoder, which supports a particular profile and level, is only required to support the corresponding subset of algorithmic tools and set of parameter constraints.

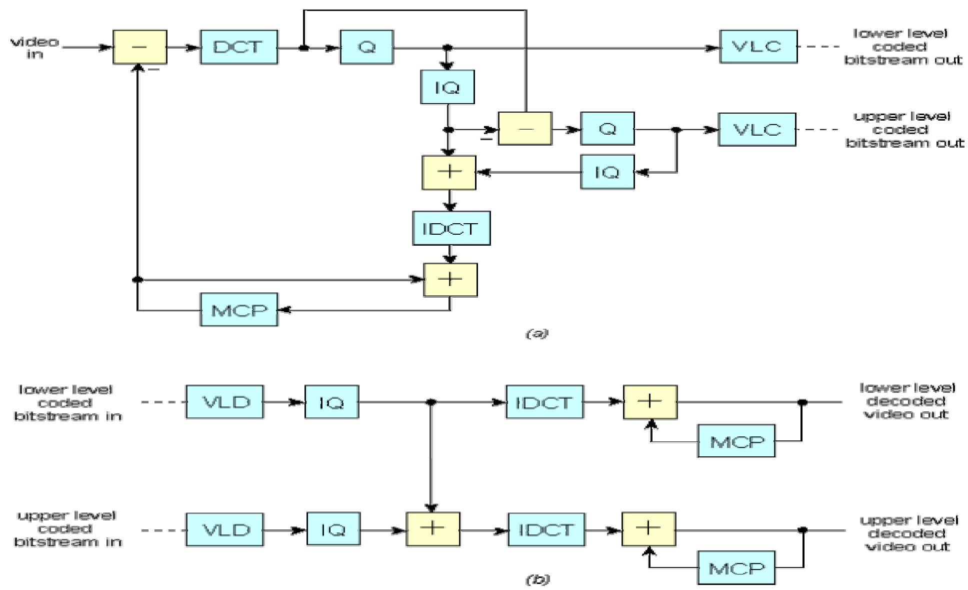
### *3.6.1 Details of non-scalable profiles:*

Two non-scalable profiles are defined by the MPEG-2 specification.

The *simple profile* uses no B-frames, and hence no backward or interpolated prediction. Consequently, no picture reordering is required (picture reordering would add about 120 ms to the coding delay). With a small coder buffer, this profile is suitable for low-delay applications such as video conferencing where the overall delay is around 100 ms. Coding is performed on a 4:2:0 video signal.

The *main profile* adds support for B-pictures and is the most widely used profile. Using B-pictures increases the picture quality, but adds about 120 ms to the coding delay to allow for the picture reordering. Main profile decoders will also decode MPEG-1 video. Currently, most MPEG-2 video decoder chip-sets support the main profile at main level.3.6.2 Details of scalable profiles:

The *SNR profile* adds support for enhancement layers of DCT coefficient refinement, using the 'signal to noise (SNR) ratio scalability' tool. Fig. 4 shows an example SNR-scalable coder and decoder.




| | | | |
|---|---|---|---|
| (I)DCT | = (inverse) discrete cosine transform | VLC | = variable-length coder |
| (I)Q | = (inverse) quantisation | VLD | = variable-length decoder |
| MCP | = motion-compensated prediction | | |

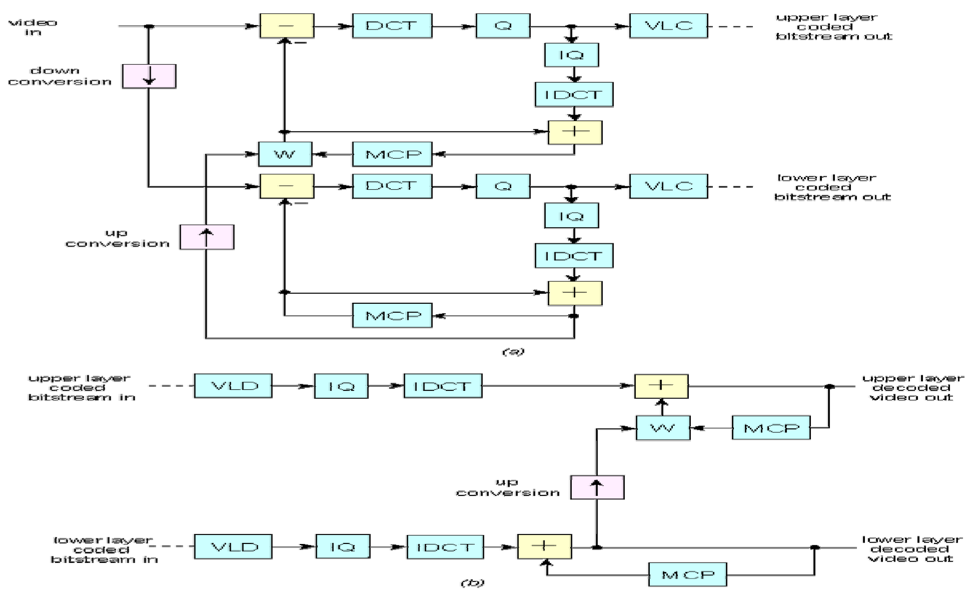
**Fig. 20, (a) SNR-scalable video coder; (b) SNR-scalable video decoder.**

The codec operates in a similar manner to the non-scalable codec shown in Fig. 2, with the addition of an extra quantization stage. The coder quantizes the DCT coefficients to a given accuracy; variable-length codes them and transmits them as the lower-level or 'base-layer' bitstream. The quantization error introduced by the first quantizer is itself quantized, variable-length coded and transmitted as the upper-level or 'enhancement-layer' bitstream. Side information required by the decoder, such as motion vectors, is transmitted only in the base layer.

The base-layer bitstream can be decoded in the same way as the non-scalable case shown in Fig. 2(b). To decode the combined base and enhancement layers, both layers must be received, as shown in Fig. 4(b). The enhancement-layer coefficient refinements are added to the base-layer coefficient values following inverse quantization. The resulting coefficients are then decoded in the same way as the non-scalable case.

The SNR profile is suggested for digital terrestrial television as a way of providing graceful degradation.

The *spatial profile* adds support for enhancement layers carrying the coded image at different resolutions, using the 'spatial scalability' tool. Fig. 5 shows an example spatial-scalable coder and decoder.




(I)DCT = (inverse) discrete cosine transform VLC = variable-length coder
(I)Q = (inverse) quantisation VLD = variable-length decoder
MCP = motion-compensated prediction W = adaptive weighting function

**Fig. 21,(a) Spatial-scalable video coder; (b) spatial-scalable video decoder.**

Spatial scalability is characterized by the use of decoded pictures from a lower layer as a prediction in a higher layer. If the higher layer is carrying the image at a higher resolution, then the decoded pictures from the lower layer must be sample rate converted to the higher resolution by means of an 'up-converter'.

In the coder shown in Fig. 5(a), two coder loops operate at different picture resolutions to produce the base and enhancement layers. The base-layer coder produces a bitstream, which may be decoded in the same way as the non-scalable case. The enhancement-layer coder is offered the 'up-converted' locally decoded pictures from the base layer, as a prediction for the upper-layer block. This prediction is in addition to the prediction from the upper-layer's motion-compensated predictor. The adaptive weighting function, W in Fig. 5(a), selects between the prediction from the upper and lower layers.

As with SNR scalability, the lower-layer bitstream can be decoded in the same way as the non-scalable case. To decode the combined lower and upper layers, both layers must be received, as shown in Fig. 5(b). The lower layer is decoded first and the 'up-converted' decoded pictures offered to the upper-layer decoder for possible use as a prediction. The upper-layer decoder selects between its own motion-compensated prediction and the 'up-converted' prediction from the lower layer, using a value for the weighting function, W, transmitted in the upper-layer bitstream.

The spatial profile is suggested as a way to broadcast a high-definition TV service with a main-profile compatible standard-definition service.

The *high profile* adds support for coding a 4:2:2 video signal and includes the scalability tools of the SNR and spatial profile.

### *3.6.3 Details of levels:*

MPEG-2 defines four levels of coding parameter constraints. Table 7 shows the constraints on picture size, frame rate, bit rate and buffer size for each of the defined levels. Note that the constraints are upper limits and that the codecs may be operated below these limits (e.g. a high-1440 decoder will decode a 720 pixels by 576 lines picture).

In broadcasting terms, standard-definition TV requires main level and high-definition TV requires high-1440 level. The bit rate required to achieve a particular level of picture quality approximately scales with resolution.

**Table 7: MPEG-2 levels: Picture size, frame-rate and bit rate constraints.**

| Level | Max. frame, width, pixels | Max. frame, height, lines | Max. frame, rate, Hz | Max. bit rate, Mbit/s | Buffer size, bits |
|---|---|---|---|---|---|
| Low | 352 | 288 | 30 | 4 | 475136 |
| Main | 720 | 576 | 30 | 15 | 1835008 |
| High-1440 | 1440 | 1152 | 60 | 60 | 7340032 |
| High | 1920 | 1152 | 60 | 80 | 9781248 |

### 3.7 CONCLUSION:

MPEG-2 has been very successful in defining a specification to serve a range of applications, bit rates, qualities and services.

Currently, the major interest is in the main profile at main level (MP@ML) for applications such as digital television broadcasting (terrestrial, satellite and cable), video-on-demand services and desktop video systems. Several manufacturers have announced MP@ML single-chip decoders and multichip encoders. Prototype equipment supporting the SNR and spatial profiles has also been constructed for use in broadcasting field trials.

The specification only defines the bitstream syntax and decoding process. Generally, this means that any decoders which conform to the specification should produce near identical output pictures. However, decoders may differ in how they respond to errors introduced in the transmission channel. For example, an advanced decoder might attempt to conceal faults in the decoded picture if it detects errors in the bitstream.

For a coder to conform to the specification, it only has to produce a valid bitstream. This condition alone has no bearing on the picture quality through the codec, and there is likely to be a variation in coding performance between different coder designs. For example, the coding performance may vary depending on the quality of the motion-vector measurement, the techniques for controlling the bit rate, the methods used to choose between the different prediction modes, the degree of picture preprocessing and the way in which the quantizer is adapted according to the picture content.

The picture quality through an MPEG-2 codec depends on the complexity and predictability of the source pictures. Real-time coders and decoders have demonstrated generally good quality standard-definition pictures at bit rates around 6 Mbit/s. As experience of MPEG-2 coding increases, the same picture quality may be achievable at lower bit rates.

# C H A P T E R 4

## *DYNAMIC MOTION VECTOR BYPASS*

### 4.1 Computational Complexity:

One of the issues under focus in this thesis, besides the amount of bandwidth adaptation required for today's internet is, the speed of the transcoder. Transcoding is computationally very expensive process. It is now that powerful machines have come, as PCs in the market, but in earlier days computers weren't that powerful as compared to today's computers. The speed of the transcoding process was the important issue in focus. Hence considerable amount of work exists in the field of optimizing the transcoder in terms of speed. Some of the works are cited and explained in brief during chapter 2, in section 2.4.4. As summarized, most of the works concentrate on optimizing the transcoder in terms of computational complexity. But increasing the speed of the trancoder comes with a trade-off that it also reduces the quality of the video. Hence proper amount of computational complexity is to be reduced to preserve the quality of the video at the acceptable level for the human eye.

Most of the transcoding schemes are DCT based and employ both motion-estimation and motion-compensation. Around 60% of the encoding time is spent in motion estimation and 11% of the time is spent in motion compensation. Both these processes are the part of the temporal prediction logic explained in chapter 3, section 3.3.5.Hence to achieve the transcoding speed, one needs to focus on reducing the computational complexity of these processes. To better understand the technique implemented to achieve the speed of the transcoder, lets discuss the processes in little more detail as compared to previous sections of the thesis.

### 4.2 Overview of Motion compensation

Motion-compensation is the important part of the MPEG-2 or any DCT based predictive coding scheme. The prediction logic is explained in brief in chapter 3, section 3.3.5. Lets discuss it in more detail.

According to the temporal prediction logic, to exploit the temporal redundancy associated between the video frames, the co-ordinates of the redundant macroblocks along with the pixel difference of each macroblock are sent instead of the macroblocks itself. The co-ordinates are called "Motion Vectors" and the pixel difference of each macroblock is called Prediction-error.

For exploiting the redundancy, search algorithms are implemented to find the best match for the current macroblock of current frame with respect to the past or future frame. The motion-estimation routine employs the search algorithm to find the match and estimate the motion vectors for that macroblock.

### 4.3 Motion-Vector Bypass Scheme:

In most of the transcoding architectures, decoding is performed to extract the information regarding the original bitstream which was coded at a different quantization level and bit rate. Some of this information can be re-used at the time of re-encoding with proper optimization. Motion vector information and macroblock coding type information are such kind of information. This macroblock information will be shortly overviewed in the coming sections. To see how much of this information can be re-used; we performed some experiments to see how much of the information is redundant.

#### *4.3.1 Experiment Setup:*

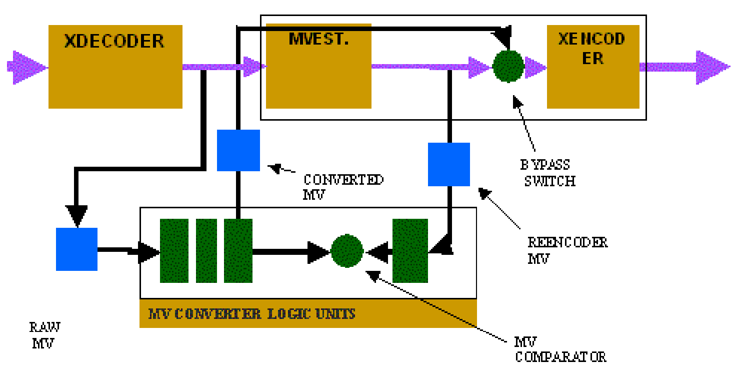



**Fig. 22, Experimental setup for MV Bypass scheme.**

As u see from the figure 22, there are two units that are being implemented into the original transcoding system, the "COMPARATOR" and the "CONVERTER". The Motion-vectors of the decoder are named as RAW MVs. There are MVs which are estimated and predicted at the re-encoder and are called Re-encoder MVs in the diagram above. Both these are fed into COMPARATOR, which will compare the RAW MVs with the Re-encoder MVs. Both the set of MVs were studied and compared for quite a few videos. Around 70% of the macroblock attributes match exactly and the rest of them had impairments. Hence these impairments needed to be refined which led to the development of the CONVERTER logic. To better understand the CONVERTER logic, let's look at the macroblock information associated with the MV information.

### *4.3.2 Macroblock information*

**Table 8: Macroblock attributes and their possible values**

| **Attributes** | **Types** |
|---|---|
| Mb_type | [Intra/forward/backward/interpolated] |
| motion_type | [Frame/field/16x8/dual_prime] |
| Skipped | [Skipped macroblock] |
| MV [2][2][2] | [Motion vectors] |
| mv_field_sel[2][2] | [Motion vertical field select – to select top or bottom field] |
| Picture_type | I/P/B/D |
| Picture_structure | Top field/bottom field/frame |

As we know from chapter 3, that video frames can be coded as frame or field type. The frame type is called Progressive frame and the field type is called the interlaced frame.

I. As we can see the MV parameter can take 8 possible values. It can have forward or backward or both forward and backward motion vectors. Forward motion vectors are those, which are estimated from the previous frame, and backward motion vectors are those, which are estimated from the future frame.

II. Mb_type indicates if the macroblock can be intra or inter. The inter types can be any of the forward, backward or interpolated. That means the MB can have forward, backward or both kinds of motion vectors.

III. Motion_type indicates if the macroblock has to be coded as any of the above-mentioned types. Though it allows 4 modes, the mostly used types are frame and field.

IV. Skipped macroblock is the one, which does not have to be estimated and predicted, but to be sent as it is.

V. Picture_type indicates if the frame is I, P or B frame. As we know there are I-Intra, P-Predicted and B- Bi-interpolated frames in MPEG-2.

VI. Picture_structure will indicate if the frame is to be coded as frame or field type.

### *4.3.3 Problems and impairments:*

- The passed MV's, coming from the decoder, which were computed at the front encoder for quantization factor Q1, may be not suitable due to different quantization in re-encoder. (Q1 is not equal to Q2).
- Macroblocks can have field motion vectors, though the motion_type indicates it as frame type due to the impairments caused during the loss.
- Macroblocks might not have information as been indicated by the macroblock_type attribute and the second case possible is that the attribute has an invalid value such as zero.
- MB's might be coded in the wrong mode. For instance, a MB that should be coded in a SKIPPED mode at re-encoder due to larger quantization making all coefficients zero could be coded as an INTER MB since it was coded as an INTER MB at the transmitter.
- Prediction error Drift problem.

Before we go on to look at the "CONVERTER " logic, lets discuss the drift problem described as the last point in the problems list.

Drift problem:

As discussed earlier in the prediction logic, that instead of the macroblock itself from (prediction error matrix + motion vectors) are coded for P and B frames if present. Prediction error is given by

$$Ec(x, y) = \text{prediction error} = Ic(x,y) - Ir[(x,y)+Mrc]$$

Where Ic(x,y) = value of the pixel at the co-ordinate (x,y) in the current frame.
Ir(x,y) = value of the pixel at the co-ordintate (x,y) in the reference frame.
Mrc = Motion vector for the frame c relative to frame r.

Since the quantization step Q1 at the transmitter is different than the quantization step Q2 at the transmitter, the prediction error matrix for the same macroblock during transmitting and re-encoding would be different. This effect is called "Drift in the transcoder". It is given as follows:

The MPEG-2 TM5 model takes care of the drift problem by re-calculating the prediction error for each macroblock at the re-encoder.

Drift = Ec(x,y)Q1 – Ec(x,y)Q2.

### *4.3.4 CONVERTER LOGIC:*

The table below refines all the impairments present in the bitstream that was decoded and also prevents the possible impairments by putting the checks for the various macroblock attributes. Below is the table, which gives the procedure, employed in the CONVERTER logic.

**Table 9: MV – Refining Table.**

| **At Q1 (Front encoding)** | | **Refining procedure** | **At Q2 (Transcoding)** | |
|---|---|---|---|---|
| Macroblock_Type | Intra | No | Macroblock_Type | Intra |
| | Skipped | No | | Skipped |
| Picture_type | Field/ Frame | Need to check the MVs for second field. If exists, make them zero. | Picture_type | Frame |
| Macroblock_Type | Inter | Check all the attributes of the macroblocks, | | |
| | | If they are zero | Macroblock_Type | Skipped |
| | | If they are not zero | Macroblock_Type | To be recalculated. |
| Macroblock-type | Zero | Evaluate the other attributes and set the attribute to the appropriate value | Macroblock_Type | Appropriate value. |
| Motion_type | | Check to see if it is right by checking the dependant attributes. | Motion_type | Appropriate value. |

The drift problem is take care of by plugging the CONVERTER logic or MV-Refiner before the prediction error is calculated in the Prediction part of the motion-compensation routine in re-encoding as shown in the diagram. Since all the impairments were taken care off, it was possible to fully bypass the motion-estimation routine from the re-encoder, which would lead to lot of timesavings. Hence the technique was named " Motion-Vector Bypass ". Lets see the transcoder architecture after the implementation of the scheme.

*4.3.5 Architecture:*

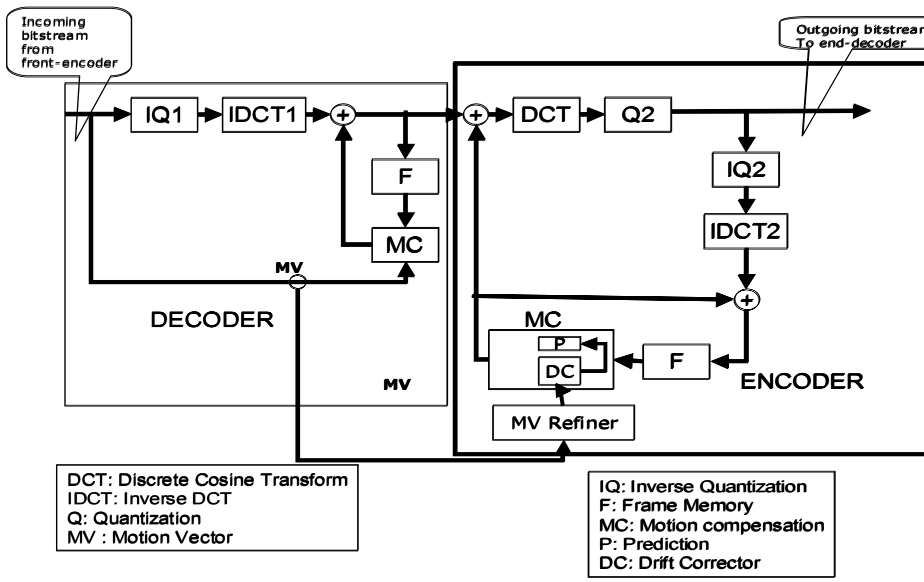



**Fig. 23, Block diagram of transcoder architecture with MV bypass implemented.**

As we can see from the figure above that MV from the decoder are directly been supplied to the MV – Refiner unit that employs the CONVERTER logic and removes the impairments present and possible during re-encoding. As u can also see that the MV-Refiner unit is implemented before the Drift Corrector unit. The detailed implementation the scheme can be seen diagrammatically in the (APPENDIX A)

**4.4 Dynamic Motion-Vector Bypass:**

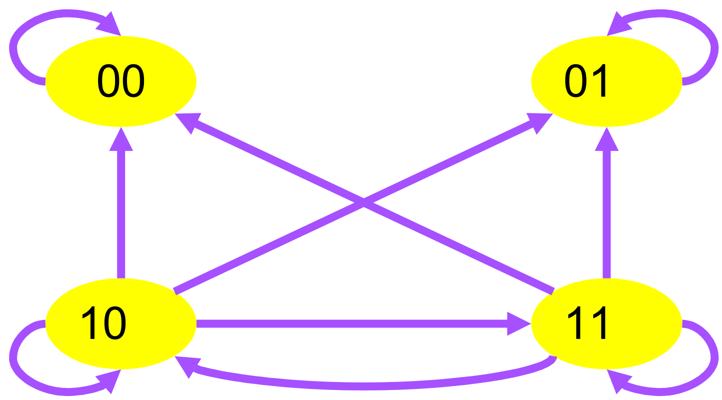



**Fig. 24, State diagram for dynamic MV bypass in transcoder.**

As seen from the state diagram above, the transcoder can be in any of these four states at a given time. There are files named bypass.par for entire re-encoding and Framedecinfo.txt for each frame to store the decoded MVs inside it. The bypass.par contains the flag for implementing bypass, which can be set by the user.

The transcoder can operate in 4 states. In addition to motion vector bypass / full logic mode, it can also choose between a dynamic and non-dynamic states. Once, the transcoder moves into non-dynamic state, its mode becomes final and cannot be changed. This transcoding system is intended for processing stream inside a programmable network such as active network.

# CHAPTER 5

## *BIT RATE SCALABILITY*

### 5.1 Tiling

As mentioned earlier in chapters 1 and 2, that the focus of this thesis is on the speed of the transcoder and also amount of bit rate adaptation required. The amount of bit rate adaptation is also called bit rate scalability. In this chapter, we will be discussing the proposed technique in detail for higher bit rate scalability. The proposed technique is called "TILING". Tiling is a resolution-reduction scheme. There have been few works cited in the literature regarding the resolution reduction based technique. In those works, the technique is used mostly to reduce the computational complexity of the transcoder.

In this chapter we will be discussing, how the technique is used for bit rate control in the transcoder as an alternative to the traditional quantization based rate control described in chapter 3, section 3.5. The chapter also describes how the traditional quantization based scheme and the tiling scheme can be used jointly to achieve the best of both of the schemes. It will also describe the method and analysis of the optimum operation point of the joint scheme.

### *5.1.1 Tiling process:*

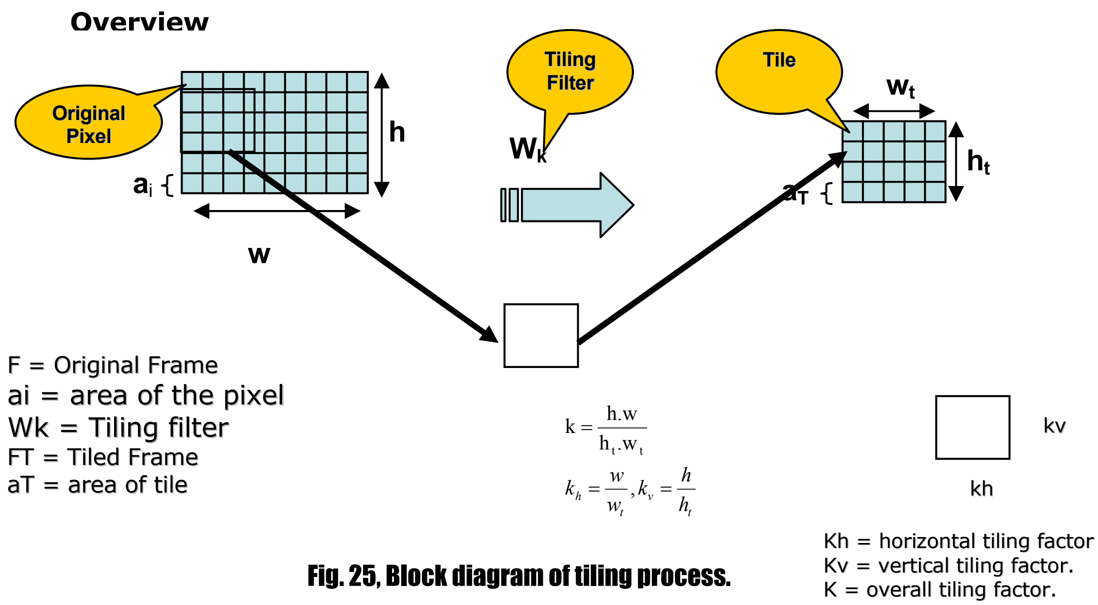



**Fig. 25, Block diagram of tiling process.**

Tiling (T) acts as a second means for rate adaptation in the proposed transcoder. The tiling operation can be explained from the diagram above. The frame of the video is viewed as a matrix of pixels. Each pixel can be viewed as rectangular entity in that frame. In tiling generally k spatially adjacent pixels from the base frame (F) are merged into one sample called *tile*. The *tile size* is denoted by k. The tiling operation can be represented as following where the base frame F of size with height (h) and width (w) is converted into a tiled frame ($F_T$) with height ($h_t$) and width ($w_t$) of smaller dimension using tile filter $W_k$:

$$\overline{F}_T = W_k \cdot F \qquad .. (9)$$

And $k = \frac{h.w}{h_t.w_t}$

Since we are using Tiling as a means for Rate Control, it is required that we choose the filter with minimal error and high flexibility. In other words, using the filter it should be possible to reduce the size of the original to any needed reduced size.

### *5.1.2 LSAT*

Several techniques for tiling have been investigated under image scaling techniques. Such as *digital differential analyzer* (DDA).

For our case we used a *linear surface approximation* (LSAT) based 2D DDA tiling process. It is a weighted average based on the surface area overlap. Each tile and pixel has rectangular area coverage. If $a_i$ is the area covered by a pixel $x_i$ and $a_T$ is the area covered by the tile T, then the value of the tile T is determined by:

$$\bar{x}_T = \frac{\sum_{i \in F} (a_i \cap a_T) \cdot x_i}{\sum_{i \in F} (a_i \cap a_T)} \qquad ..(11)$$

LSAT tiling process enables reduction of video frame size beyond what can be achieved by quantization. The LSAT mechanism shown here creates zero error for linearly distributed pixels. However, for curve surfaces it creates a gradual smoothing effect and just like the traditional quantization, it creates gradual loss of image quality and correspondingly yields higher compression.

**Implementation of LSAT algorithm**:

We will explain the algorithm in brief using the diagram below:

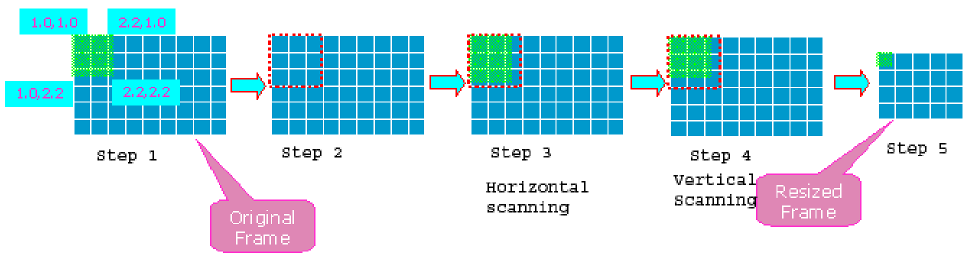



**Fig. 26, Brief step-wise representation of the Tiling algorithm.**

The procedure below is for one iteration of tile determination. The procedure is put in the loop to achieve tiles for all the original pixels.

Step 1: To calculate the co-ordinates for the current tile so as to exactly know which pixels will form the tile and how much area of the pixels to be included.

Step 2: To create a window to include all the pixels necessary for the creation of the tile.

Step 3: To scan the pixels of the window horizontally so as to find the pixels which are non-aligned horizontally. To calculate the area of the pixel that would be included in the tile horizontally.

Step 4: To scan the pixels of the window which are vertically non-aligned and then calculate the area of the tile again, which will give the total area of the tile.

Step 5: To find the value of the tile by dividing the value of the pixel in the area of the tile with the area of the tile.

### ***5.1.3 Implementation of Tiling:***

*5.1.3.1 Transcoder architecture after tiling:*

The diagram below will help one to understand the implementation of tiling in the transcoder.

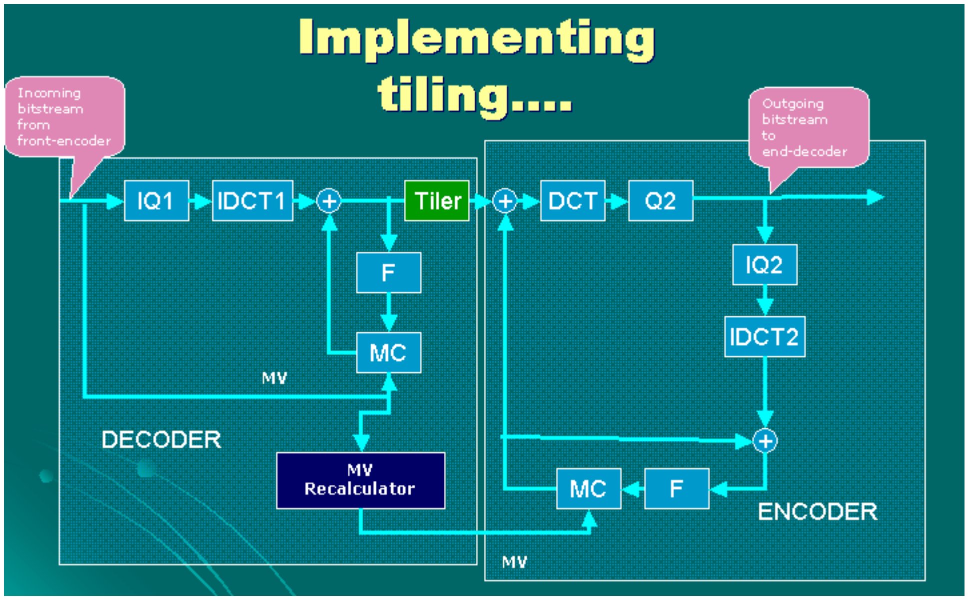



**Fig. 27, Transcoder architecture with Tiling implemented.**

We have used the Berkeley's TM-5 model for transcoding the MPEG-2 bitstreams. The codec structure of the transcoder is similar to the codec structure of MPEG-2 explained in chapter 3, section 3.4.1.

As we can see from the figure 27, that the tiling filter is being implemented at the stage of the transcoder when the input bit stream is decoded and being stored into the Y, U and V components identified by the block name "TILER" in the diagram. The pixel matrices of all the above components within the input bitstream are passed to the "TILER" and been converted to the pixel matrices of the reduced size according to the tiling factor supplied to the tiling filter. We will see in the next section, how the reduction of the input video is achieved according to the output bit rate.

*5.1.3.2 Motion-Vector Re-calculation:*

Another important modification that u can see in the transcoding architecture is implementation of MV-Recalculator implying motion-vector re-calculator. The concept of Motion vectors is studied in chapter 3 during the explanation of temporal prediction used to remove temporal redundancy.

The motion-vectors of the original bit stream cannot be re-used for the re-encoding purposes since the number of macroblocks at the time of re-encoding will be reduced. Since the video is not clipped but reduced, the macroblocks of the reduced video frame are not the same as the macroblocks of the original video frame but rather are calculated from the original macroblocks.

The algorithm for the motion-vector re-calculation is developed and implemented on the same lines as that of the LSAT tiling filter with a change that, it is applied on the matrix of macroblocks and the entities involved are forward and backward motion vectors of each macroblock if present. The source-code of the algorithm can be viewed from the APPENDIX C.

### ***5.1.4 Rate Control by Tiling:***

Algorithm:

Step 1: Determine the size of the tile (k) from the input and output bit rates from the followingequation:

K = Input Bit rate/Output bit rate

Step 2: To obtain the horizontal and vertical tiling factors by taking the square root of the K obtained in the first step.

Kv=kh= sqrt(K)

In the figure 27, its explained that, kv = h/ht and kh = w/wt, where kv an kh are vertical and horizontal tiling factors. Vertical tiling factor gives the number of pixels to be merged vertically and horizontal tiling factor gives number of pixels to be merged horizontally.

Step 3: To obtain the required reduced height and width of the video.

Resized width = Original_width/Kh
Resized height = Original_height/Kv

**5.2 Joint Error Analysis**

Now we will provide a joint analysis of the distortion created by the combined use of the tiling and quantization processes that can be used by the rate transcoder.

When $f_{init}(x)$ is the distribution of the original samples, the error created by the first stage quantization is given by:

$$\sigma_{Q1}^2 = \int_{\theta n} (x - r_n)^2 \cdot f_{init}(x) \cdot dx \qquad ..(13)$$

The error created by tiling is given by the variance of the distortion created by the process. Here $f_{tile}(x)$ is the probability distribution function of the tiled values, and can be determined from the TM5 operations for certain distributions. In LSAT interpolation the loss occurs during the initial linear approximation of the surface defining the pixel values in the tile area. Thus the error created by tiling can be considered to be bounded by the variance of the distortion created by the piece-wise linear approximation in the tiling operation and is given by the following:

$$\sigma_T^2 = \int_T (x_i - \hat{x}_{T \ni i})^2 \cdot f_{tile}(x) \cdot dx \qquad ..(14)$$

The error added by the second stage quantization of the tiles is given by:

$$\sigma_{Q2}^2 = \sum_{all \ \theta_m} \int_{\theta_m} (\bar{x}_t - r_m)^2 \cdot f_{tile}(\bar{x}_t) \cdot d\bar{x} \qquad ..(15)$$

Let $r_{m,T}$ is the quantization decision level of the effective quantizer corresponding to the tile that includes pixel $x_i$. The joint error is given by:

$$\sigma_{Q1+T+Q2}^2 = \int_{\theta_m} (x - r_{m,T \ni i})^2 \cdot f(x) \cdot dx \qquad ..(16)$$

The distortion can be computed in a logarithmic format by means of PSNR. In image processing the *peak-signal-to-noise-ratio* (PSNR) is defined as:

$$PSNR = 10 \log_{10}\left(\frac{255^2}{\sigma^2}\right) \qquad ..(17)$$

The value 255 refers to the maximum possible value of the pixel value and σ refers to the current pixel value.

**5.3 Joint Entropy Analysis**

To understand this part of the thesis it is necessary to know the meanings of the various parameters that will be seen in the forthcoming discussion. To understand the meanings and the values of these parameters, refer to APPENDIX B (optimum tiling parameters.)

The bit rate resulting from the re-quantization and tiling process can be estimated based on entropy analysis. For requantization the probability that x is in the m-th tiling interval is given by:

$$p_m = \int_{\theta_m} f(\bar{x}) \cdot d\bar{x} \quad ..(18)$$

These intervals are uneven however with bound 1. The entropy of each symbol is then given by:

$$H = -\sum_m p_m \cdot \log_2 p_m \quad \text{bits/symbol} \quad ..(19)$$

Let us assume that F is the initial samples/frame, k is the tiling factor (samples/tile), B is the number of tiles in each encoding block (Macro Block for MPEG) [number of encoding units in a frame i.e. macro-blocks?] and H is the amortized coding block overhead. Then the bit requirement for each frame due to quantization is given by:

$$\Pi = \frac{B \cdot H}{k} - \frac{\alpha \cdot F}{k} \sum_m p_m \cdot \log_2 p \quad \text{bits/frame} \quad ..(20)$$

Here α (<1) is the compression factor due to non-quantization techniques used (such as variable length decoding].

## 5.4 Transcoding Modes:

After implementing tiling, the transcoder can at least work in two modes.

The first mode is called mute mode, in which only quantization based scheme is used. That means, no tiling is used for rate control. In the second mode, only tiling is used for the rate control and is called the aggressive mode.

However to achieve least distortion, that is to get the best of both the schemes, we have also investigated the third mode called, Optimum Hybrid. Lets discuss each of them in brief.

### *5.4.1 Mute- Quantization only Reduction*

If we use only quantization as a means of reduction, then this Q-only schemes quantization factor for any given outgoing data rate R can be estimated from equation (20) with k=1.

$$R = \frac{B \cdot H}{1} - \frac{\alpha \cdot F}{1} \log_2 \frac{1}{m}$$ bits/fra ..(20)

me

**Thus:**

$$m_{Q-only} = 2^{\frac{R-BH}{\alpha \cdot F}}$$ ..(20)

### *5.4.2 Aggressive- Tiling only Reduction*

In opposite, extreme a tiling only (T-only) reduction can reduce samples without directly reducing the quantization factor. Since, tiling proportionately reduces the header as well as samples, the tiling factor can be computed simply as a ratio of incoming and outgoing bit rate.

$$k_{opt} = \frac{R_{init}}{R} \qquad ..(26)$$

### *5.4.3 Optimum - Hybrid Reduction*

The least distortion best results can be obtained if we use a Hybrid scheme, which prudently uses both, tiling as well as quantization. Now we demonstrate the optimum choice of the tiling factor and the number of quantization levels. The optimality is defined in the sense where for a given bit rate the overall distortion can be minimized. We show the solution for *uniform* sample distribution and *linear sample surface* approximation. We will track the bounds on the maximum distortion that can be generated in each of the processes.

Let s is the average *slope-factor* of the sample surface. The average distortion generated by the tiling process is given by:

$$E = |E_{Q1}| + |E_S| + |E_{Q2}| \qquad ..(21)$$

**Under the given assumptions the error is bounded by:**

$$E = \left|\frac{256}{4n}\right| + \left|\frac{s.k}{4}\right| + \left|\frac{256}{4m}\right| \qquad ..(22)$$

**Let R is the choice of bit-allocation per frame (MPEG-2 TM5 model), then:**

$$E = \left|\frac{256}{4n}\right| + \left|\frac{256}{4m}\right| + \frac{s}{4R} \cdot \left(B \cdot H + \alpha \cdot F \log_2 m\right) \qquad ..(23)$$

The number of first stage quatization levels (n) is not selectable at transcoding stage. We can solve for minimum E by:

$$\left.\frac{dE}{dm}\right|_{m=m_{opt}} = 0 \qquad ..(24)$$

**This gives the optimum quantization steps:**

$$m_{opt} = \frac{256R}{\alpha \cdot s \cdot F} \cdot \log_e 2 \qquad ..(25)$$

**And the optimum tiling factor:**

$$k_{opt} = \frac{1}{R}\left[B \cdot H + \alpha \cdot F \cdot \log_2\left(\frac{256R}{\alpha \cdot s \cdot F}\log_e 2\right)\right] \qquad ..(26)$$

# CHAPTER 6

## *RESULTS*

## 6.1 Motion Vector Bypass:

### *6.1.1 Speed-up Analysis and Results:*

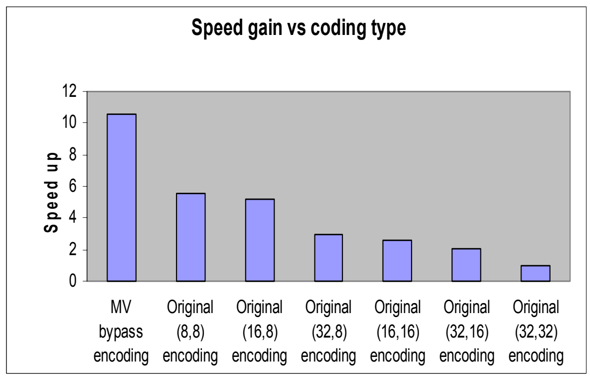



**Fig. 28, Speed gain for coding with different Motion Vector search area.**

The first graph shows the speed gain by implementing the motion-vector bypass scheme in the transcoder for different search area. It shows that with decreasing the search area for the matching macroblock in the past or future frame in the process of motion-estimation as explained in the earlier chapters, the maximum speed up gained is 3 times. But with the bypass scheme being implemented with the maximum search area of (32,32), the speed gain is approximately 10 times than the original encoding with the same search area.

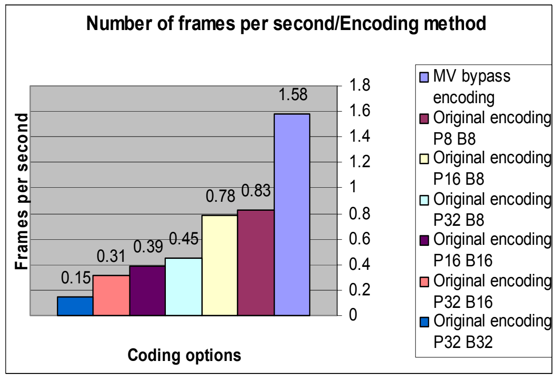



**Fig. 29, Number of frames per second for coding with different MV-search area.**

### *6.1.2 Quality analysis and results:*

As mentioned earlier, that any speed up in the transcoder comes with a tradeoff in loosing quality of the video. We have measured the quality of video in terms of SNR

(Signal to noise ratio). The formula for PSNR calculation is shown in chapter 5, section 5.2.

The graph below reveals that for different video sizes (704x480, 352x240, 176x120) at different output bit rates the quality of the video goes down by approximately 0.9 DB (unit for measuring SNR)

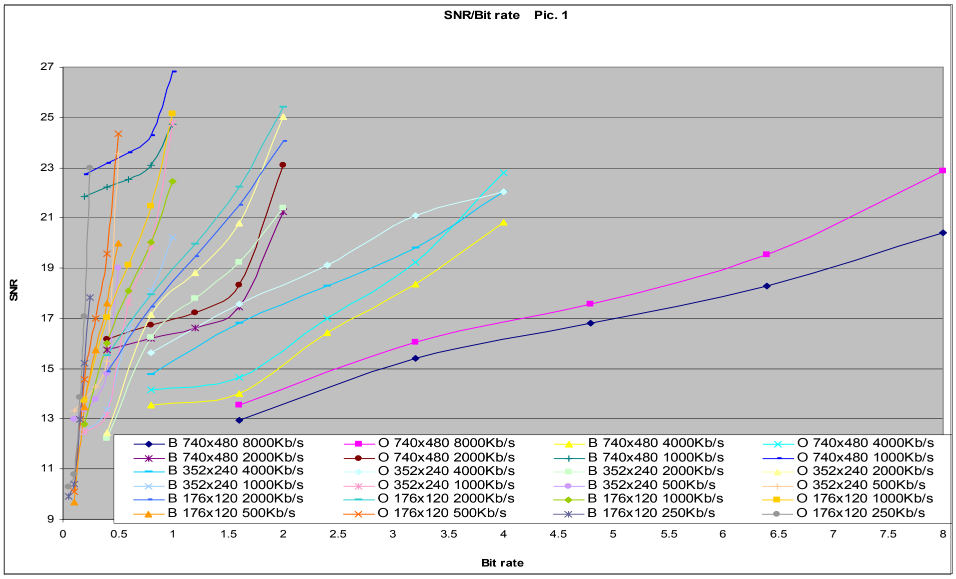



**Fig. 30, PSNR graph for various videos with MV bypass and without it.**

## 6.2 Various Transcoding Modes:

### *6.2.1 Analysis of Optimum tiling and Quantization factor*

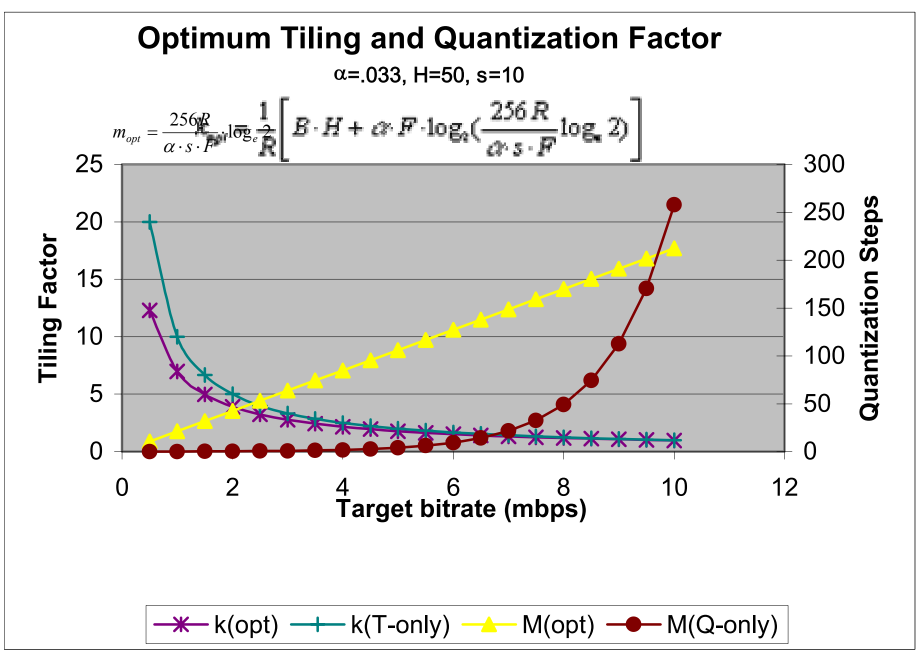


**Fig. 31, Graph for optimum tiling factor and quantization factor.**

As we can see from the figure 31, that analysis of the optimum tiling factor(k-opt), aggressive tiling factor(K-Tonly) , optimum quantization factor(M-opt) and Mute-only quantization step size(M-Qonly) have been depicted. The quantization factor is taken to be the inverse of the quantization step size used .

As we can see that M(Q-only ) decreases exponentially with the decrease in the output bit rate and gradually flatens out at around 2 Mbps. Hence it tells that compression after around 2 Mbps is not achieved.

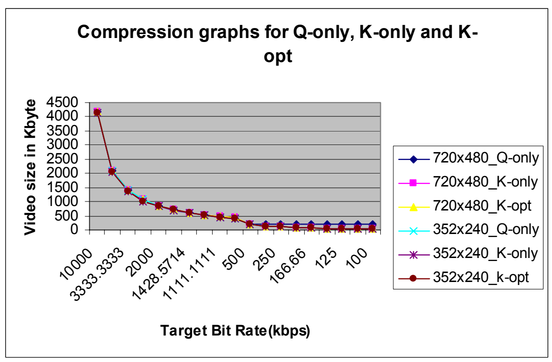
The curve for the M-opt decreases linearly with the linear decrease in the bit rate. Hence linear compression is achieved , which is the optimum solution achievable.

The curve for the K(T-only) is more steeper at the end bit rates as compared to the K-opt curve. Hence compression by tiling is controlled by optimally distributing with the quantization factor.

In Figure 31, we show the predicted optimum tiling factor ($k_{opt}$) and quantization steps for various target reduction rates ($m_{opt}$). Let d is the sample density. For 4:4:4, 4:2:2, and 4:2:0 video d is respectively 3, 2 and 1.5 samples/pixel. Thus for a frame size of X x Y pixels, F= X.Y/d samples/frame. We started with an initial 256 step quantizes 740x480 sized 10 Mbps encoded high quality 4:2:0 MPEG-2 video. The plot shows both the predicted optimum m and k for various reductions (encoding $\alpha$=0.033, H=50 bits, and s=10).

### *6.2.2 Compression analysis and results:*

The graph below shows the amount of compression achieved in all the three-transcoding modes (Mute, Aggressive, Optimum) indicated by (Q-only, K-only, K-opt) respectively. Videos of two sizes (704x480, 352x240) were chosen for data collection. The output bit rate ranges from 10 Mbps – 0.1 Mbps. The Y-axis shows the size of the video in Kbytes (Kilobytes) and the X- axis shows the output bit rate in Mbps (megabits per second). It is noticeable from the graph that there is a difference in compression at lower bit rates in the Mute transcoding mode as compared to the other two modes. The compression seems more in the other two modes.




**Error!**

**Fig. 32,Video sizes at different bit rates for each type of trancoding mode.**

Lets look at the lower bit rate region in the above graph more closely from the graph below. The difference in compression achieved by the mute mode as compared to the other two modes is quite clear. The aggressive and optimum modes achieve higher compression than the mute mode. Also the compression achieved by the aggressive and the optimum mode is almost the same.

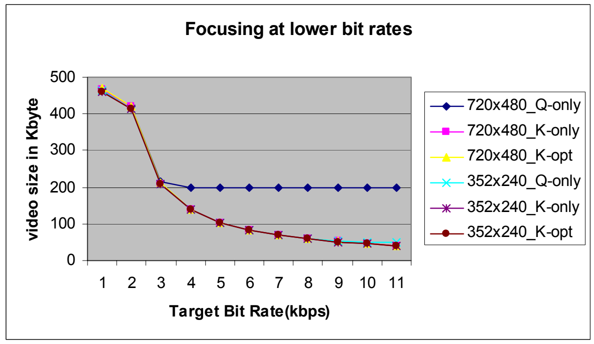



**Fig. 33, Lower bit rates analysis graph.**

### *6.2.3 Quality analysis and results:*

The graph below shows, how the quality of the original video is affected in all the three-transcoding modes (Mute, Aggressive, Optimum) indicated by (Q-only, K-only, K-opt) respectively. Videos of two sizes (704x480, 352x240) were chosen for data collection. The output bit rate ranges from 10 Mbps – 0.1 Mbps. The Y-axis shows the quality of the video in terms of PSNR (Peak Signal to Noise Ratio) and the X- axis shows the output bit rate in terms of Mbps (Megabits per second).

As seen from the graph below, the quality of the video is better in the other two modes as compared to the mute-mode.

Between the optimum and aggressive tiling, the optimum performs better in most of the lower bit rates as compared to the aggressive. It is also clear that the optimum mode is more effective with the video of  (352x240) size as compared to the other video size (704x480).

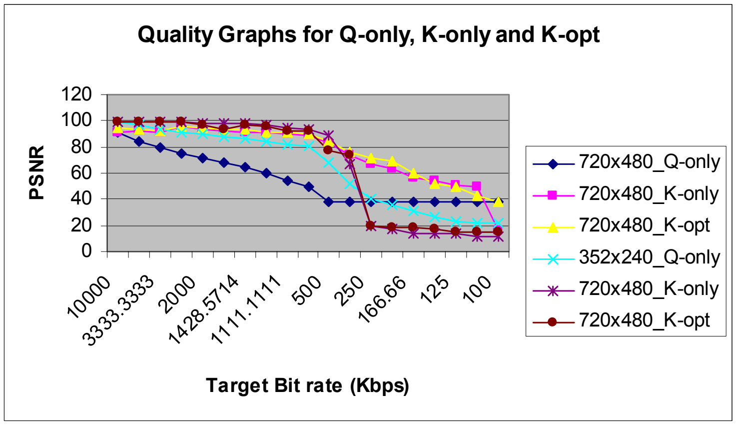



**Fig. 34, Quality - PSNR of various transcoding modes with given (bit rate range)**

## Chapter 7

## *CONCLUSIONS AND FUTURE WORK*

The main objective of this research was to investigate techniques for higher bit rate adaptation, adaption for terminals of different processing power and display capabilities. In this chapter we will summarize the results in brief and give future directions regarding the area studied.

### *7.1 Conclusions*

As studied and analysed in the chapter of results , we can conclude that the trancoder scheme being investigated has been able to achieve a speed-up of around 10 times using the Motion-Vector Bypass scheme.

### *7.2 Future Works*

- Transcoder can also be used dynamically inside a network or active network.
- Transcoder could also be used as a pre-processing tool on a video server.
- Optimum tiling parameter are at present specified by the user, but it can be automated by using an optimization algorithm.

# BIBLIOGRAPHY


1. [SuZd96] W. K. Sun and J. W. Zdepski, Architecture for MPEG compressed bitstream scaling, IEEE Transactions on Circuit Systems Video tech, 6(2), August 1996, pp.191-199.
2. [Kafa98] Kou-Sou Kan and Kua-Chin Fan, Video Transcoding Architecture with Minimum Buffer Requirement for Compresses MPEG-2 bitstream, Journal of Signal Processing, 67 (1998), pp.223-235.
3. [ChKi96] U. Chong and S. P. Kim, Wavelet Trancoding of block DCT-based images through block transform domain processing, SPIE Vol. 2825, 1996, pp901-908.
    a. Dean Clark, A 2-D DDA Algorithm for Fast Image Scaling, Dr. Dobb's Jounral, April 1997.
4. [YoSL99] J. Youn, M.T. Sun, and C.W. Lin, "Motion Vector Refinement for High Performance Transcoding," IEEE, Transactions on Multimedia, Vol. 1, No. 1, pp.30-40, March 1999.
5. [KHHH96] G. Keesman, R. Hellinghuizen, F. Hoeksema, & G. Heideman, "Transcoding of MPEG Bitstreams," Signal Processing Image Comm., vol. 8, pp. 481-500, 1996.
    a. P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded video for transmission at lower bit rates," ICASSP'96, vol. 4, pp. 1998-2001, May 1996.
6. [AsGh98] P. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit rate reduction of MPEG-2 bit streams," Trans. On Circuits Syst. Video Technol., vol. 8, no. 8, pp. 953-967, 1998.
7. [KaFC98] Kan, Kou-Sou; Fan, Kuo-Chin, Video transcoding architecture with minimum buffer requirement for compressed MPEG-2 bitstream Signal Processing, Volume: 67, Issue: 2, , pp. 223-235, June 18, 1998
8. [SLKK98] Seo, Kwang-Deok; Lee, Sang-Hee; Kim, Jae-Kyoon; Koh, Jong S., Efficient rate-control algorithm for fast transcoding, ,SPIE Vol. 3528, 1998.
9. [BjCh00] Niklas Björk and Charilaos Christopoulos, Video transcoding for universal multimedia acces; Proceedings on ACM multimedia 2000 workshops, 2000, Pages 75 - 79
10. [DoSK00] Dogan, S.; Sadka, A. H.; Kondoz, A. M, Video transmission over mobile satellite systems International Journal of Satellite Communications Volume: 18, Issue: 3, May/June 2000, pp. 185 – 205.

11. [YoSu00] J. Youn, M.T. Sun, Video Transcoding with H.263 Bit-Streams, Journal of Visual Communication and Image Representation Volume: 11, Issue: 4, December 2000, pp. 385 – 403.
12. L. Chiariglione, "The development of an integrated audiovisual coding standard: MPEG," *Proc. IEEE*, vol. 83, pp. 151–157, Feb. 1995.
13. G. Kerr, "BT's video on demand technology," in *5th IEE Conf. Telecommun.*,Brighton, U.K., Mar. 1995, pp. 318–322.
14. H. J. Stuttgen, "Network evolution and multimedia communication," *IEEE Multimedia*, vol. 2, pp. 42–59, Fall 1995.
15. C. Kaplan, "Interactive video services," in *5th IEE Conf. Telecommun.*,Brighton, U.K., Mar. 1995, pp. 323–326.
16. J. Sandvoss and T. Schuttt, "Ecole: An ATM based environment fordistributed learning," in *IEEE Int. Conf. Commun.*, Seattle, WA, June1995, vol. 1, pp. 585–590.
17. M. R. Civanlar and G. L. Cash, "An experimental system for MPEG-2 video transmission over ATM networks," in *6th Int. Workshop Packet Video*, Portland, OR, Sept. 1994, pp. C2.1–C2.4.
18. M. Ghanbari, "Two-layer coding of video signals for VBR networks," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 771–781, June 1989.
19. ISO/IEC 13818-2, "Information technology—Generic coding of moving pictures and associated audio information—Part 2: Video," 1995.
20. H. Sun, W. Kwok, and J. W. Zdepski, "Architectures for MPEG compressed bitstream scaling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 191–199, Apr. 1996.
21. Eleftheriadis and D. Anastassiou, "Meeting arbitrary QoS constraints using dynamic rate shaping of coded digital video," in *5th Int. Workshop Network and Operating Syst. for Digital Audio and Video*, Durham, NC, Apr. 1995, pp. 95–106.
22. Y. Nakajima, H. Hori, and T. Kanoh, "Rate conversion of MPEG codedvideo by re-quantization process," in *IEEE Int. Conf. Image Processing*,Washington, DC, Oct. 1995, vol. 3, pp. 408–411.

23. P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded videofor transmission at lower bit rates," in *IEEE Int. Conf. Acoust., Speech,Signal Processing, ICASSP'96*, Atlanta, GA, May 1996, vol. 4, pp.1998–2001.

24. D. G. Morrison, M. E. Nilsson, and M. Ghanbari, "Reduction of bitrate of compressed video while in its coded form," in *6th Int. Workshop Packet Video*, Portland, OR, Sept. 1994, pp. 392–406.

25. G. Keesman, R. Hellinghuizen, F. Hoeksema, and G. Heideman, "Transcoding of MPEG bitstreams," *Signal Processing: Image Commun.*, vol. 8, pp. 481–500, Sept. 1996.

26. P. Assuncao and M. Ghanbari, "Transcoding of MPEG-2 video in the frequency domain," in *IEEE Int. Conf. Acoust., Speech, Signal Processing, ICASSP'97*, Munich, Germany, Apr. 1997, vol. 4, pp.2633–2636.

27. ISO/IEC, "Test Model 5, ISO/IEC JTC1/ SC29/ WG11/ N0400, MPEG93/457," Apr. 1993.

28. N. Merhav and V. Bhaskaran, "A fast algorithm for DCT-domain inverse motion compensation," in *IEEE Int. Conf. Acoust., Speech,Signal Processing*, Atlanta, GA, May 1996, vol. 4, pp. 2307–2310.

29. R. Plompen, B. F. Schuurink, and J. Biemond, "A new motioncompensated

30. transform coding scheme," in *IEEE Int. Conf. Acoust. Speech, Signal Processing, ICASSP'85*, Tampa, FL, Mar. 1985, vol. 1, pp. 371–374.

31. S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 1–11, Jan. 1995.

32. Y. Arai, T. Agui, and M. Nakajima, "A fast DCT-SQ scheme for images," *Trans. IEICE*, vol. E71, pp. 1095–1097, Nov. 1988.

33. N. Merhav and V. Bhaskaran, "Fast algorithms for DCT-domain image down-sampling and for inverse motion compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 468–476, June 1997.

34. W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Data Compression Standard.* New York: Van Nostrand Reinhold, 1993.

35. Y. Shoham and A. Gersho, “Efficient bit allocation for an arbitrary set of quantizers,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1445–1453, Sept. 1988.

36. K. Ramchandran and M. Vetterli, “Best wavelet packet bases in a, ratedistrotion sense,” *IEEE Trans. Image Processing*, vol. 2, pp. 160–175,Apr. 1993.

37. W. Y. Lee and J. B. Ra, “A fast algorithm for optimal bit allocation,” in *SPIE, Visual Commun. Image Processing, VCIP'97*, San Jose, CA, Feb. 1997, vol. 3024, pp. 167–175.

38. W. Coene and G. Keesman, “A fast route for application of ratedistortion optimal quantization in an MPEG video encoder,” in *IEEE Int.Conf. Image Processing, ICIP'96*, Lausanne, Switzerland, Sept. 1996, vol. 2, pp. 825–828.

39. Manipulating Temporal Dependencies in Compressed Video Data with Applications to Compressed-Domain Processing of MPEG Video”, S. Wee, ICASSP 1999.

40. “Compressed-Domain Reverse Play of MPEG Video Streams”, S. Wee, B. Vasudev, SPIE Inter. Sym. On Voice, Video, and Data Communications 1998.

41. “Field-to-frame Transcoding with Spatial and Temporal Downsampling”, S. Wee, J. Apostolopoulos, N. Feamster, ICIP 1999.

42. Bo Shen, Ishwar K. Sethi, Vasudev Bhaskaran, “Adaptive motion vector resampling for compressed video downscaling,” *ICIP'97.*

43. Nicholas Yeadon, Francisco Garcia, David Hutchison and Doug Shepherd, “Continuous Media Filters for Heterogeneous Internetworking,” *Proceedings of SPIE – Multimedia Computing and Networking (MMCN'96)*, 1996.

*44.* Neri Merhav and Vasudev Bhaskaran, “A transform domain approach to spatial domain image scaling,” *ICASSP'96*

45. Michael T. Orchard, Gary J. Sullivan, "Overlapped Block Motion Compensation: An estimation-theoretic approach," *IEEE Transaction on Image Processing*, vol. 3, NO. 5, 1994.

46. Bede Liu, King-Wai Chow and Andre Zaccarin, "Simple method to segment motion £eld for video coding," *Proceeding of SPIE*, vol. 1818, pt 2, 1992.

47. Min Wu, Heather Yu and Alex Gelman, "Multi-level Data Hiding for Digital Image and Video", *SPIE Photonics East '99*

48. Neri Merhav and Vasudev Bhaskaran, "A fast algorithm for DCT-domain inverse motion compensation," *ICASSP'96*.

49. J. Moura, R. Jasinschi, H. Shiojiri-H. and C. Lin, "Scalable video coding over heterogeneous networks," in Proc. of the SPIE - The International Society for Optical Engineering. vol. 2602, pp. 294-306. 1996.

50. N. Chaddha, "A software only scalable video delivery system for multimedia applications over heterogeneous networks," in Proc. Int. Conf. on Image Processing, Washington, DC, Oct. 1995.

51. M. Ghanbari, "Two-layer coding of video signals for VBR networks," IEEE J. Select. Areas Commum., ~01.7, pp.771-781, June 1989.

52. ITU-T Rec. H.263, "Video Codec for Low Bit Rate Communication," May 1996.

53. ISO/IEEE 13818-2, "Information technology-Generic coding of moving pictures and associated audio information-Part 2: Video,", 1995.

54. Eleftheriadis and D. Anastassiou, "Constrained and general dynamic rate shaping of compressed digital video," in Proc. IEEE Int. Conf. Image Processing, Washington, DC., Oct. 1995.

55. G Keesman, et al., "Transcoding of MPEG Bitstreams," Signal Processing Image Comm., vol. 8, pp. 481-500, 1996.

56. P. Assuncao and M. Ghanbari, "Post-processing of MPEG2 coded video for transmission at lower bit rates," ICASSP'96, vol. 4, pp. 1998-2001, May 1996.

57. P. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit rate reduction of MPEG-2 bit streams," Trans. On Circuits Syst. Video Technol., vol. 8, no. 8, pp. 953-967, 1998.

58. H. Sun, W. Kwok, and J.W. Zdepski, "Architectures for MPEG compressed bitstream scaling, ", IEEE Trans. Circuits Syst. Video Technol., vol. 6, pp. 191-199, Apr.1996.

59. S.F.Chang and D.G.Messerschmitt, "Manipulation and Cornpositing of MC-DCT Compressed Video," IEEE Journal of Selected Areas in Communications, pp. l- 11, Jan, 1995.

60. J. Youn and M-T Sun, "Motion Estimation for High Performance Transcoding," IEEE Int. Conf. on Consumer Electronics, Los Angeles, June 1998

61. M. T. Sun and I-Ming Pao, "Statistical Computation of Discrete Cosine Transform in Video Encoders," Journal of Visual Communications and Image Representation, Vol. 9, No. 2, pp, 163-170, June, 1998

62. P. A. A. Assuncao and Mohammed Ghanbari, Post processing of MPEG2 Coded Video for Transmission at lower bit rates, Proc. IEEE ICASSP'96, vol. 4, pp. 1998-2001, May 1996.

63. V. Bhaskaran and K. Konstantinides, Image and Video Compression Standards, Algorithms and architectures, second edition, Kluwer Academic Publishers, 1997.

64. Björk N. and Christopoulos C., "Transcoder Architectures for video coding", *IEEE Transactions on Consumer Electronics*, Vol. 44, No. 1, pp. 88-98, February 1998.

65. S.F. Chang and D. Messerschmitt, "Manipulation and composition of MC-DCT compressed video", IEEE JSAC Vol. 13, No. 1, January 1995, pp. 1-11.

66. ITU-T Recommendation H.263, "Video coding for low bitrate communication", 11/95.

67. ITU - T, Study Group 16, Draft Text of recommendation H.263 version 2 (H.263+), September 1997.

68. G. Keesman, R. Hellinghuizen, Fokke Hoeksema and Geert Heideman, Transcoding of MPEG bitstreams, Signal Processing: Image Communication, vol. 8, pp. 481-500, September 1996.

69. L.-K. Liu, and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding", IEEE Trans. CAS, Vol. 6, No. 4, pp. 419-422, August 1996.

70. B. K. Natarajan and B. Vasudev, "A fast approximate algorithm for scaling down digital images in the DCT domain", Proc. ICIP 95, Vol. II, pp. 241-243, Oct. 1995.

71. H. Sun, W. Kwok and J. W. Zdepski, Architectures for MPEG compressed bitstream scaling, IEEE Trans. on Circuits and Systems for Video Technology, vol. 6, No. 2, pp. 191-199, April 1996.

72. "MPEG-7 Applications document v.8.1", ISO/IEC JTC1/SC29/WG11/M4839, July 1999, Vancouver, Canada.

73. "MPEG-7 Multimedia Description schemes XM (version 3.0), ISO/IEC JTC1/SC 29/Wg 11/N3410, May 2000, Geneva.

74. *Peng Yin, Min Wu, and Bede Liu,*Department of Electrical Engineering Princeton University, Princeton, NJ 08544, U.S.A.

**75.** http://www.merl.com/projects/MPEG-transcoding

76. ISO/IEC 11172: 'Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s'.

77. ISO/IEC 13818: 'Generic coding of moving pictures and associated audio (MPEG-2)'.

78. 'Encoding parameters of digital television for studios', CCIR Recommendation 601-1 XVIth Plenary Assembly Dubrovnik 1986, Vol. XI, Part 1, pp. 319-328.

79. JAIN, A.K.: 'Fundamentals of digital image processing' (Prentice Hall, 1989).

80. WELLS, N.D.: 'Component codec standard for high-quality digital television',*Electronics & Communication Engineering Journal*, August 1992, **4**, (4), pp. 195-202.

81. CARR, M.D.: 'New video coding standard for the 1990s', *Electronics & Communication Engineering Journal*, June 1990, **2**, (3), pp. 119-124.

82. RAO, K.R. and YIP, P.: 'Discrete cosine transform: algorithms, advantages, applications' (Academic Press, 1990).

83. Department of Electrical Engineering ,Princeton University, Princeton, NJ 08544, U.S.A.

# APPENDIX A

## *SYSTEM DESIGN AND IMPLEMENTATION*

### A.1 Detailed diagram for MV bypass implementation

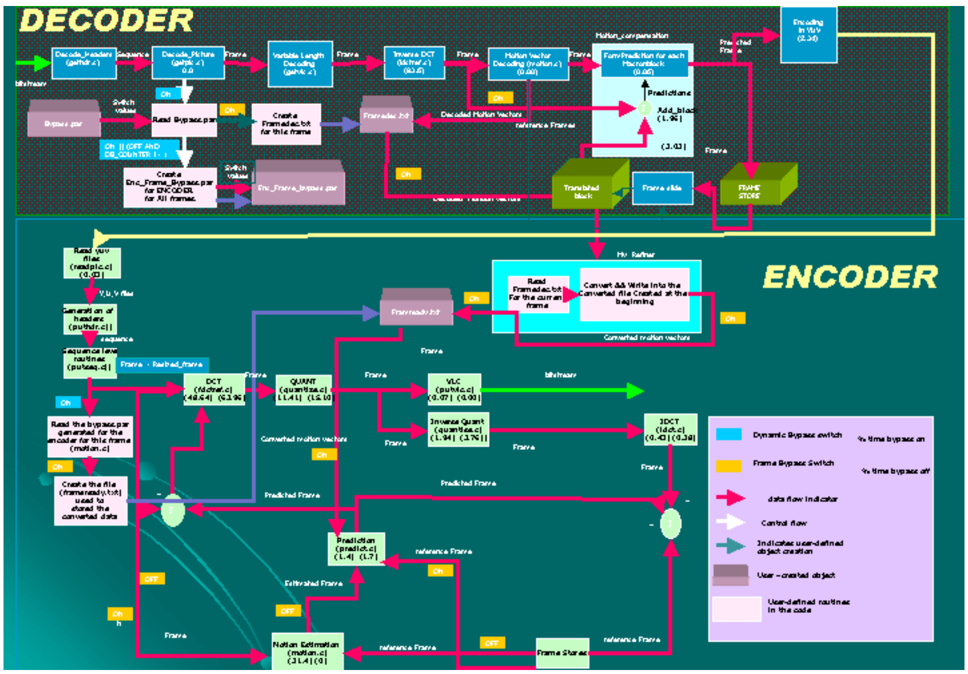



**Fig. 35, Detailed implementation of Motion Vector Bypass scheme.**

## A.2 Detailed diagram for MV bypass implementation and Tiling

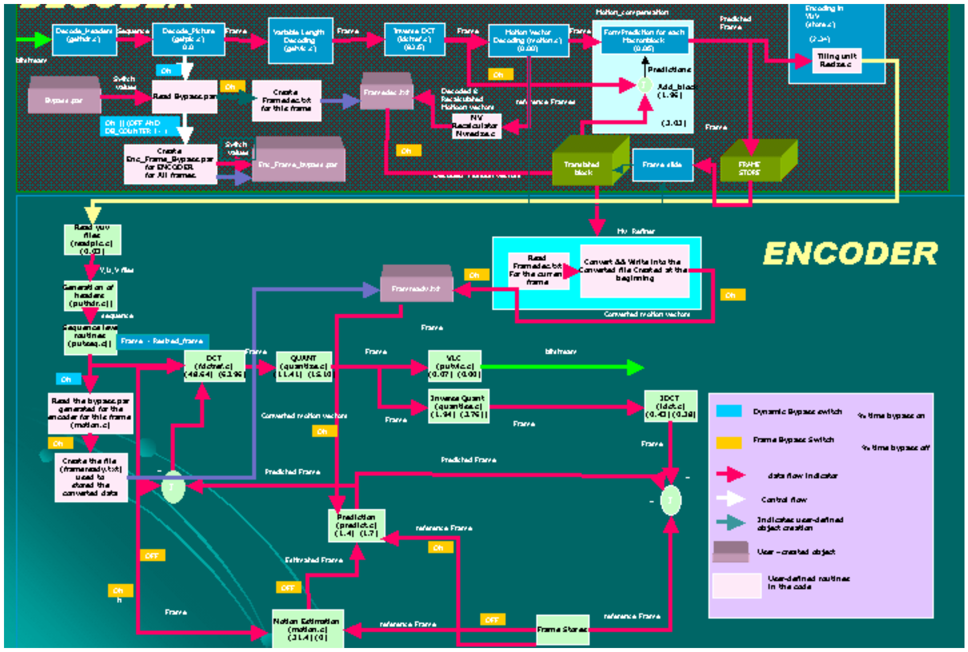


**Fig. 36, Detailed Implementation of MV Bypass and Tiling together**

## A P P E N D I X B

### *SYSTEM PARAMETERS*

### B.1 Decoded bitstream parameters

Below are the decoded parameters that are sent to the the re-encoder by storing them in the file « test.par » which is accessed by the re-encoder.

**Table 10 : Decoded bit stream parameters**

| **test%d** | **nature of source files** |
|---|---|
| - | Name of reconstructed images ("-": don't store) |
| - | Name of intra quant matrix file ("-": default matrix) |
| - | Name of non intra quant matrix file ("-": default matrix) |
| Stat. out | Name of statistics file ("-": stdout ) |
| 0 | Input picture file format: 0=*.Y,*.U,*.V, 1=*.yuv, 2=*.ppm |
| 20 | Number of frames |
| 0 | number of first frame |
| 00:00:00:00 | time code of first frame |
| 15 | N (# of frames in GOP) |

| | |
|---|---|
| 3 | M (I P frame distance) |
| 0 | ISO IEC 11172-2 stream |
| 0 | 0:frame pictures, 1:field pictures |
| 320 | horizontal_size |
| 230 | vertical_size |
| 2 | aspect_ratio_information 8=CCIR601 625 line, 9=CCIR601 525 line |
| 5 | sec. |
| 3000000.0 | bit_rate (bits/sec ) |
| 112 | Vbv_buffer_size (in multiples of 16 kbit) |
| 0 | Low_delay |
| 0 | Constrained_parameters_flag |
| 4 | Profile ID: Simple = 5, Main = 4, SNR = 3, Spatial = 2, High = 1 |
| 8 | Level ID: Low = 10, Main = 8, High 1440 = 6, High = 4 |
| 0 | progressive_sequence |
| 1 | chroma_format: 1=4:2:0, 2=4:2:2, 3=4:4:4 |
| 1 | video_format: 0=comp., 1=PAL, 2=NTSC, 3=SECAM, 4=MAC, 5=unspec. |
| 5 | color_primaries |

| | |
|---|---|
| 5 | transfer_characteristics |
| 4 | matrix_coefficients |
| 320 | display_horizontal_size |
| 230 | display_vertical_size |
| 0 | intra_dc_precision (0: 8 bit, 1: 9 bit, 2: 10 bit, 3: 11 bit ) |
| 1 | * top_field_first * |
| 0 0 0 | * frame_pred_frame_dct (I P B) * |
| 0 0 0 | * concealment_motion_vectors (I P B) * |
| 1 1 1 | * q_scale_type (I P B) * |
| 1 0 0 | * intra_vlc_format (I P B)* |
| 0 0 0 | * alternate_scan (I P B) * |
| 0 | * repeat_first_field * |
| 0 | * progressive_frame * |
| 0 | * P distance between complete intra slice refresh * |
| 0 | * rate control: r (reaction parameter) * |
| 0 | * rate control: avg_act (initial average activity) * |
| 0 | * rate control: Xi (initial I frame global complexity measure) * |
| 0 | * rate control: Xp (initial P frame global complexity measure) * |
| 0 | * rate control: Xb (initial B frame global complexity measure) * |
| 0 | * rate control: d0i (initial I frame virtual buffer fullness) * |

| | |
|---|---|
| 0 | * rate control: d0p (initial P frame virtual buffer fullness) * |
| 0 | * rate control: d0b (initial B frame virtual buffer fullness) * |
| 2 2 11 11 | * P: forw_hor_f_code forw_vert_f_code search_width |
| 1 1 3 3 | * B1: forw_hor_f_code forw_vert_f_code search_width |
| 1 1 7 7 | * B1: back_hor_f_code back_vert_f_code search_width |
| 1 1 7 7 | * B2: forw_hor_f_code forw_vert_f_code search_width |
| 1 1 3 3 | * B2: back_hor_f_code back_vert_f_code search_width |

**B.2 Optimum tiling factor parameters:**

$$k_{opt} = \frac{1}{R}\left[ B \cdot H + \alpha \cdot F \cdot \log_2\left(\frac{256R}{\alpha \cdot s \cdot F} \log_e 2\right) \right]$$

X = width of the video

Y = height of the video

Rout = Outgoing Bit Rate

FRAME-RATE used = 30 frames per second

B = Number of Macroblocks

H = MB overhead

R = bits per frame = Rout/ FRAME-RATE

P = pixels per frame = X*Y

F = samples per frame] {here F=p*1.5 for 4:2:0, 2 for 4:2:2}

S = slope of the samples

ALPHA [a] = compression due to non-quantization techniques.

e =2.7182

# APPENDIX C

## *TECHNICAL PAPER , REPORTS AND SOURCE CODES*

### C.1 Technical Papers:

- Javed I. Khan, Seung Su Yang, Qiong Gu, Darsan Patel, Patrick Mail, Oleg Komogortsev, Wansik Oh, and Zhong Guo Resource Adaptive Netcentric Systems: A case Study with SONET- a Self-Organizing Network Embedded Transcoder, Proceedings of the ACM Multimedia 2001, October 2001, Ottawa, Canada, pp617-620

- Javed I. Khan, Seung S. Yang, Darsan Patel, et. al, Resource Adaptive Netcentric Systems on Active Network: A Self-Organizing Video Stream that Auto Morphs Itself while in Transit via a Quasi-Active Network, Proceedings of the DARPA Active Networks Conference and Exposition, DANCE 2002, San Jose, CA May 21-24, 2002, IEEE Computer Society Press, [available at URL http://medianet.kent.edu/technicalreports.html, also mirrored at http://bristi.facnet.mcs.kent.edu/~javed/medianet/technicalreports.html] (accepted as full paper).

## C.2 Technical Reports

- TR2000-10-01 A Performance Profile of MPEG-2 Transcoding with Motion Vector Reuse, October, 2000 [Patel, Oh]
- TR2001-01-01 Architectural Overview of Motion Vector Reuse Mechanism in MPEG-2 Transcoding, January, 2001 [Khan, Patel, ALL]
- TR2000-08-01 Architectural Overview of Medianet Multiprocessor Active Video Transcoder, August 2000.
- TR2001-02-02 Impact on Stream Compression and Video Quality Motion Vector Reuse Transcoding [Khan, Oleg]

## C.3 Source Code

Transcoder with MV bypass and all the transcoding modes implemented is available here.

http://niihau.medianet.kent.edu/dpatel/thesis_working/thesis_resize/s-mvX-p