MASTER

A real-time SNR scalable transcoder for MPEG-2 video streams

Al-khrayshah, M.

Award date:
2009

EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computer Science

A real-time SNR scalable transcoder for MPEG-2 video streams

by
Mohammad Al-khrayshah

Supervisors:

Prof. J.J. Lukkien
Eindhoven University of Technology
j.j.lukkien@tue.nl

Dr. P.D.V. van der Stok
Philips Research
peter.van.der.stok@philips.com

Dr. R.H. Mak
Eindhoven University of Technology
r.h.mak@tue.nl

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Philips is a global leader in healthcare, lifestyle, and technology. Information Processing Architecture (IPA) is member of Philips Research Eindhoven. This graduation project has been carried out for IPA. The key capability of IPA is the architecture of computer-based consumer terminals. This capability consists of defining, architecting, and designing resource constrained systems. A large part of its work is dedicated to realizing Philips' connected home.

Philips is now developing the concept of the connected home. It is an environment that places people at the center of technology. The connected home consists of multiple devices such as PCs, mobile phones, and laptops. The purpose of the connected home network is to connect the home devices together to improve daily activities. These devices are connected by a wireless home network. A broadband connection connects the wireless home network to the external world as shown in Figure 1.1.

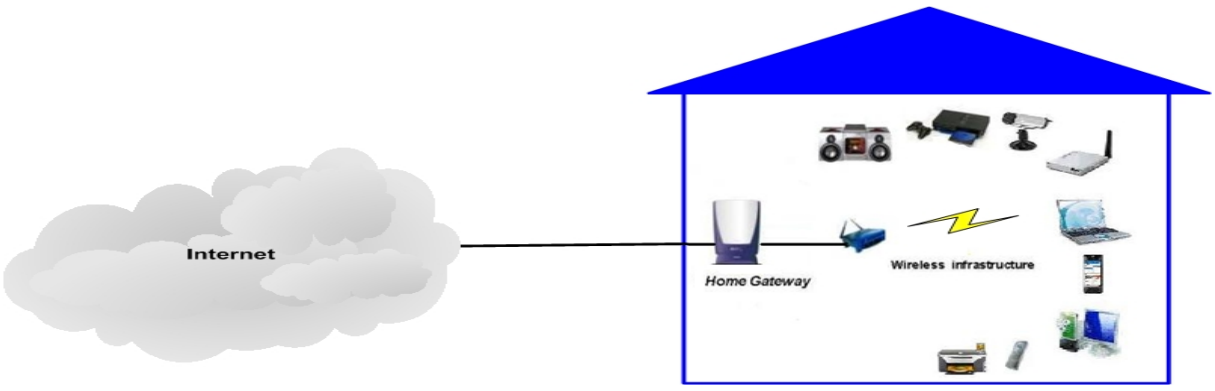


Figure 1.1: Connected home

Wireless networks are becoming more and more popular because they provide more features and benefits than traditional wire local area networks. These benefits are cost effectiveness, mobility, and ease of integration with another network and network component. Furthermore, wireless networks can increase the number of clients within existing equipment.

Quality of Service (QoS) refers to the capability of a network to provide better service to any network traffic over various technologies. In this project, high quality video is streamed over a wireless network within the available bandwidth. In other words there is QoS tradeoff between the available bandwidth provided by a wireless channel and the picture quality perceived by viewers. The main goal of the project is to deal with QoS, so high quality video is converted into the adapted video before sending over a network.

This graduation project discusses the research that I have performed to complete the requirement of my Master's degree in Computer Science at Eindhoven University of Technology (TU/e).

1.1 Challenges

Nowadays, the concept of video streaming becomes more popular because the Internet supports the concept of multimedia streaming, such as online TV. The concept of streaming is not the same as the concept of downloading. Streaming video can be played out as soon as it arrives at the end user. On the other hand, download video is played out after downloading. If streaming video is live broadcasting, then it is called "real-time video". Real-time video means that there is no a visible delay for end users. For example, a video conference is a real-time bidirectional video. the video conference requires a large bandwidth. At real-time, the challenge is to obtain high video quality within an available bandwidth. Therefore, networks can provide high video quality to the end users when providing the required bandwidth for streaming video. Nowadays,however, most networks do not have enough bandwidth due to network infrastructures and/or outdoor conditions.

Most of networks are often heterogeneous such as the Internet. The heterogeneous networks consist of wireless networks and wire networks. The wireless networks have some disadvantages such as multi-path fading. Furthermore, they suffer from both short-term fluctuation and long-term fluctuation within an available bandwidth. The problems in the present wireless networks are as follows:

- Several copies from the same signal arrive at the receiver at different times after traveling in different paths, after being reflected on different objects.
- Wireless networks often have a time-varying nature due to either the terminal mobility or a modification in environment along the transmit link.
- Interference is caused by other devices whose output signals occupy the same frequency band as transmitted signal.
- The wireless channel is not very reliable due to frequent transmission errors. This has a negative effect upon the available bandwidth.
- The available bandwidth can change very quickly when the distance changes between a sender (and an access point) and a receiver.

As a result, network packets get lost when a bandwidth fluctuates. This means that a wireless channel can degrade the perceived video quality. As mentioned before, QoS gives the description of network performance that is outside the administration of the end users. Consequently, the wireless networks must deal with these problems which make very difficult to provide high-level of QoS.

Besides that the problem of bandwidth, the end users use portable devices , such as mobiles, PDAs etc, that are introduce additional problems. These devices have limited computation complexities and display capabilities. Thus, the portable devices do not support high quality video. In this case, high quality video must be converted into a lower quality video for displaying on the portable devices. In addition, if a video source distributes the same video content to some clients through channels with different bandwidth, then high bit rate video must be converted into exact bit rates for each channel.

1.2 The Streaming Video

Bandwidth is the essential factor for realizing video quality. When the bandwidth is low, then the video quality is not high, and vice versa. A compressed video needs less bandwidth than an un-compressed video. Therefore, compressed video is used in many applications, such as DVD, CD, and TV broadcast etc, for to the following purposes:

- Compressed video saves available bandwidth.
- Compressed video can be transmitted faster in the same bandwidth.
- Compressed video can give better video quality in the same bandwidth.

Compressed video is an essential part of the current communication world. However, the bandwidth required by compressed video does not usually match the available bandwidth. Therefore, it is necessary to use some adaptive techniques that overcome a lower bandwidth.

Video multicasting is the distribution of a single video to a group of clients simultaneously over heterogeneous networks. The clients normally communicate with a video server over the Internet. A channel between any client and the video server has several hops. This means that each channel has different bandwidth or each client has different bandwidth. How can the same video be multicasted to different clients through different capacity channels in the same time?

The first technique is a multiple copies switching that is used to overcome a lower bandwidth. In this technique, the video sender has multiple copies of the same video. Each copy has different quality and bit rate. This means that each copy can match a specific bandwidth. However, this technique, which uses a large amount of storage on the sender, does not overcome the bandwidth fluctuations.

The second technique is a re-encoding that re-compresses the pre-encoded video. Figure 1.2.a shows the re-encoding life cycle that can summarize in the following steps:

1. The video source streams the compressed video to the decoder.

2. The decoder de-compresses the compressed video and stores the un-compressed video in the memory.

3. The encoder compresses the un-compressed video and streams the compressed video to the client.

Re-encoding requires a large computation complexity. Moreover, the re-encoded video can have different structure at each re-coding life cycle.

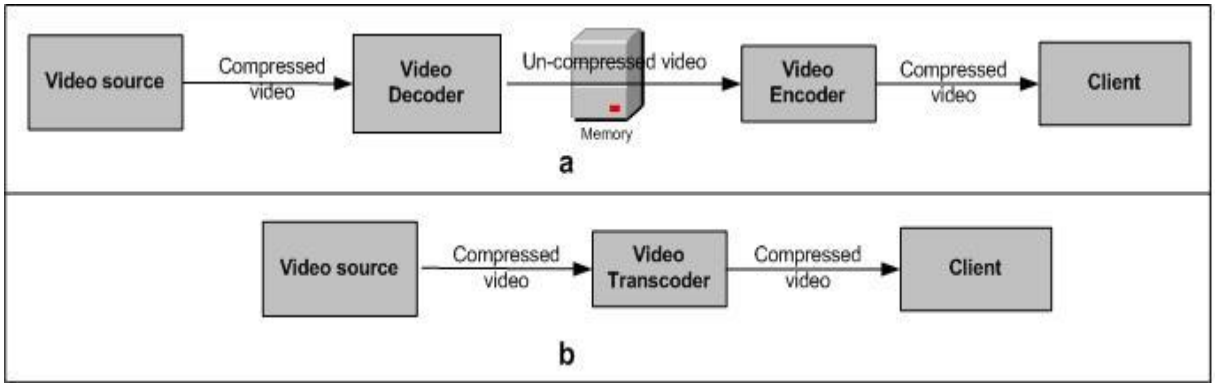


Figure 1.2: a) Re-encoding the compressed video , b) Transcoding the compressed video

The third technique is a scalable video transcoding method. A transcoder is different from an encoder and a decoder. Both the input and the output of the transcoder are a compressed video as shown in Figure 1.2b. The transcoder works partially in the decompressed video stream and carries out processing directly into the compressed domain.Therefore, transcoding needs less computation time than re-encoding. Furthermore, the transcoder can be used to multicast video. Finally, transcoding is more efficient than the re-encoding.

Compressed video is being used for transmission and storage by different media such as a satellite, LAN, and the Internet. In this project, we have been using the MPEG-2 standard. MPEG-2 video supports a wide range of video formats, including digital TV, DVD, and HDTV. MPEG-2 sequences consist of Groups of Pictures (GOP). Each GOP contains a number of frames. There are three kinds of frames: intra (I), predicted (P) and bi-directional (B). A GOP always starts with an I-frame which is coded as an independent frame. The P-frame is coded relative to the previous frame which could be either an I- or P-frame. The P-frame depends on the reference frame (I- or P-frame) for both encoding and decoding and is smaller in size than the I-frame. A B-frame is coded depending on both the previous and the next I-frame or P-frame. The B-frame is smaller in size than the P-frame, see Chapter 2 for more details.

1.3 Problem statement

In a real environment, streaming video suffers from a delay and a distortion that can appear to the end users when bandwidth fluctuates. This means that there is a QoS problem. The main goal of this project is to overcome the QoS problem such that high quality MPEG-2 can be streamed over a wireless channel. In this project, transcoding technique is used to overcome the QoS problem. A transcoder converts any pre-encoded video from one format to anther format. In this project, the transcoder converts high quality MPEG-2 video into scalable MPEG-2 video before being sent over a wireless channel.

High quality MPEG-2 is being streamed from the MPEG-2 source to the video transcoder gateway via WAN. A video transcoder gateway consists of a network sender and a MPEG-2 transcoder as shown in Figure 1.3. The MPEG-2 transcoder converts high quality MPEG-2 to scalable MPEG-2 by using transcoding parameters that come out of the sender. The transcoder divides the complete bit stream into two or more layers. A scalable video stream can be adapted by changing the layering configuration at run-time. The layering configuration consists of the number of layers and one target bit rate of each individual layer. This means that the number of layers and the target bit rate changes when the bandwidth fluctuates.

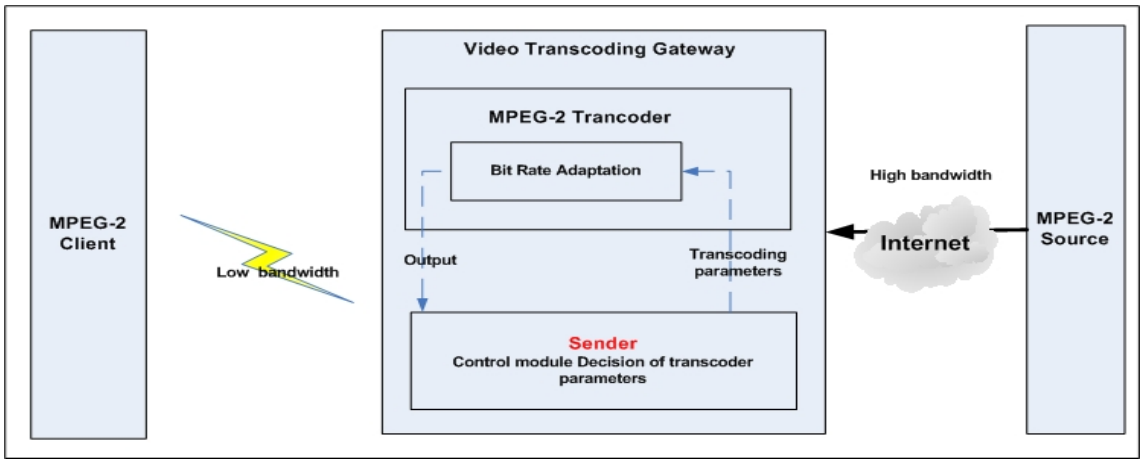


Figure 1.3: MPEG-2 transcoder in real environment

In this project, the transcoder converts high bit rate MPEG-2 into scalable MPEG-2. A scalable transcoder consists of a base layer and the enhancement layers. Each layer generates several bit rate. The base layer has different architectures that are: a closed-loop transcoder, a cascaded transcoder and an open-loop transcoder. Open-loop transcoding has a smaller computational complexity than closed-loop transcoding, but open-loop transcoding does not have better quality than the closed-loop one.

In previous work [17], an open-loop transcoder has implemented using the C programming language. This transcoder converts high bit rate MPEG-2 into a lower bit rate MPEG-2. TM-5 is an algorithm for bit rate control that is used widely [9]. In previous work, TM-5 was adapted to enable changing the target bit rate at run-time. The open-loop transcoder has the adaptive TM-5 algorithm that, which is implemented for the base layer only, controls the output streaming video.

Furthermore, in [17] a scalable transcoder has been introduced that consists of a closed-loop transcoder to generate a base layer, with the addition of generating enhancement layers. The closed-loop transcoder was selected as a compromise between the picture quality achievable by a cascaded transcoder and the performance by an open-loop transcoder. The enhancement layers are generated according to an adapted form of scalable video. As a result, the data of an enhancement layer can be easily dropped. In this project, the basic requirements for the transcoder are the following:

1. The number of video layers should be adjustable at run-time.
2. The preferred bit rate for each layer should be dynamically adjustable at runtime. The transcoder should react within the time of a frame period to the adjusted bit rate.

3. The transcoder should transcode a frame of the incoming video to frames for the video layers within the time of 1/f second, where f is the frame rate of the incoming video. We have used a Dell platform consisting of Pentium 4 2.4 GHz CPU, 512 MB memory, and running Windows XP.

4. The transcoder should be able to communicate with a control program as shown in Figure 1.3, that defines the preferred bit rate for each layer and the number of enhancement layers.

5. The transcoder should be able to connect to a network sender that will send the video layers over a network.

The basic tasks of transcoding are classified as follows: frame size conversion, bit rate reduction, frame rate conversion and coding syntax conversion. In this project, the bit rate reduction, that is used, converts an MPEG-2 video stream to a scalable MPEG-2 stream according to the channel conditions; such that the individual layers of the scalable video stream have a lower bit rate than the input stream.

In previous work, only the open-loop transcoder was implemented. In this project, we have researched, designed, and implemented enhancements to MPEG-2 video transcoding. We have investigated the proposed architecture, which consists of a closed-loop transcoder to produce the base layer. The closed-loop transcoder has been implemented using C programming language. This transcoder generates MPEG-2 that has better quality than the open-loop transcoder. The closed-loop transcoder improves QoS but does not perform at real-time. Therefore, we have investigated a new transcoder architecture which we have called a closed-loop in a P-frame transcoder. In this project, this transcoder must provide better QoS than the open-loop transcoder. Furthermore, the transcoder must work in real-time. Furthermore, the transcoder that must be more suitable for deployment for video streaming over wireless networks must satisfy some design constraints, such as CPU usage and picture quality. A closed-loop in a P-frame transcoder has implemented by using the C programming language. The closed-loop in a P-frame does not perform at real-time. Therefore, we have researched a new transcoder that can provide better video quality than the open-loop transcoder. Furthermore, the new transcoder must perform at real-time. Therefore, we have proposed a hybrid transcoder. In addition, the hybrid transcoder has been implemented. This transcoder performs at real-time, and generates MPEG-2 that has better quality than the open-loop transcoder. For more details see Chapter 3.

In this project, the use of scalable video has been investigated as a way to change adaptively the bit rate of a transcoded MPEG-2 video, such that this video fits the available bandwidth at any moment. Also the use of transcoder was investigated to deal with long-term bandwidth fluctuations, caused by e.g. bandwidth sharing between multiple streams.

Besides improving the quality of the QoS problem mentioned above, we have researched, designed and implemented a proposed TM-5 bit rate control algorithm that is called real-time TM-5. This algorithm is better than the adaptive TM-5 bit rate control algorithm that introduced in the previous work. Real-time TM-5 needs lees computational time than the adaptive TM-5 bit rate control algorithm. In addition, the adaptive TM-5 bit rate control algorithm has delay response time. The real-time TM-5 algorithm overcomes delay response time. For more details see Chapter 4.

1.4 Outline of the Thesis

Chapter 2 introduces the basic techniques specified in the MPEG-2 standard, which are required to understand the remaining chapters. Chapter 3 introduces transcoding principles. In this chapter, the different transcoding architectures are discussed. Chapter 4 describes the bit rate control algorithm, which is used to control the bit rate of video layers. Chapter 5 gives the conclusions and the future work. Chapter 6 presents the video multilayer transcoding principles. In this chapter, some SNR scalability transcoding architectures are discussed, they can target for the future work.

Chapter 2

MPEG-2 Video Standard

A data processing system is a system that processes data that has been created and stored by an information processing system. It uses data compression algorithms to minimize transmission time of data or to save broadcast bandwidth in a given communication channel. Furthermore, data compression algorithms are used to reduce the memory space required to store a given amount of data.

Video compression algorithms are data compression algorithms for digital video. Compressed videos are being used for transmission and storage by different media such as satellite, LAN, and the Internet. In this chapter, we introduce the MPEG-2 standard which is extensively used in the video industry for television broadcast, visual communications, and multimedia applications. Moving Picture Experts Group (MPEG) has successfully developed the MPEG-2 standard, which is formally referred to as ISO/IEC specification 13818, which is the second phase of the MPEG video solution for applications not originally covered by the MPEG-1 standard [15]. Specifically, MPEG-2 which can generally be seen as a superset of the MPEG-1 standard was developed to provide high video quality such as HDTV quality.

The process of compressing and uncompressing is often referred to as 'encoding' and 'decoding' respectively. MPEG specifies bitstream syntax and decoding process. Encoding algorithms (compression algorithms) are open to invention and proprietary techniques. In MPEG-2, the compression algorithm eliminates the redundant transmission of unchanging elements of the scene. For example, the remote video storage can send MPEG-2 through a communication channel with less bandwidth usage than an uncompressed video. This means that MPEG-2 requires less transmission time than uncompressed video in the same communication channel. Therefore, MPEG-2 is often used by applications requiring high video quality while a limited bandwidth is available that controls the quality level.

2.1 The MPEG-2 Standard Video

MPEG is a committee that was formed by ISO (International Organization for Standardization) [18]. The purpose of MPEG was to create the compressed video standards for multimedia and broadcast applications. MPEG files are often encoded by Huffman coding which is a compression method [18]. A frame contains the compressed information of the MPEG video. It contains a sequence of frames and it can be described by using the following variables:

- Frame rate: the number of frames displayed per second. For example, the frame rate of a television is 25 frames per second.
- Frame dimensions: the width and the height of the frame expressed in the number of pixels.
- Pixel depth: the number of bits per pixel.

Some of the current MPEG standards are:

1. MPEG-1
   (a) Medium Bandwidth (up to 1.5Mbps)

i. 1.25Mbps video 352 x 240 x 30Hz
ii. 250Kbps audio (two channels)

(b) Non-interlaced video

(c) works for applications such as CD-ROM

2. MPEG-2

(a) Higher Bandwidth (up to 40Mbps)

(b) Up to 5 audio channels

(c) Wider range of frame sizes (including HDTV)

(d) Can work with interlaced video

3. MPEG-3

(a) MPEG-3 was for HDTV application with dimensions up to 1920 x 1080 x 30Hz.

4. MPEG-4

(a) Very Low Bandwidth (64Kbps)

(b) 176 x 144 x 10Hz

(c) videophones

MPEG-2 is an extension of MPEG-1. An MPEG-2 video is using a bit stream to represent a video content. An MPEG-2 algorithm compresses video through converting a RGB color space(red, green, and blue) to another color space called YCbCr in order to achieve a high compression ratio. A luminance(Y) contains information of the brightness of every pixel. Chrominance components (Cb and Cr) contain color information.

The human eye is less sensitive to the color than to brightness. Thus, an MPEG-2 encoder is designed to compress the video color more than brightness. In other words, Y has more details than Cb and Cr. Therefore, the MPEG-2 video requires fewer number of bits to represent the color components.

A macroblock can be represented in several different manners when referring to the YCbCr color space. Figure 2.1 shows 3 formats known as 4:4:4, 4:2:2 and 4:2:0 video. For example, 4:2:0 means that the Cb and Cr matrices shall be one half the size of Y-matrix in both the horizontal and vertical dimensions, as shown in Figure 2.1. 4:2:0 allows a data reduction from 12 blocks per macroblock to 6 blocks per macroblock. 4:2:2 allows a data reduction from 12 blocks per macroblock to 8 blocks per macroblock.

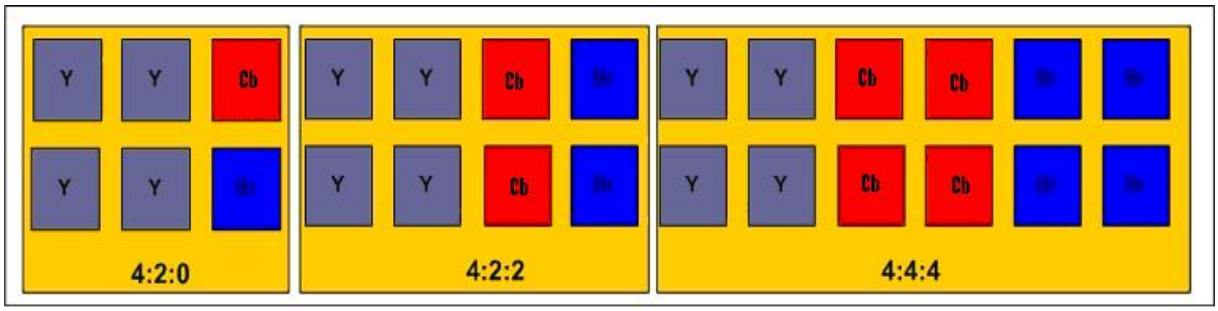


Figure 2.1: Macroblock structure

The MPEG-2 frames may be represented in two different methods: interlaced or progressive. The MPEG-2 defines the syntax for coding interlaced video. The interlacing method displays all odd lines in the frame first and then all even lines in the same frame, as shown in Figure 2.2 a. The interlacing method exploits half frames rather than full frames per second. The MPEG-2 video uses the advantages of the interlaced method to increase encoding efficiency and to improve the picture quality. The interlacing method supports bit rates up to 40 Mbps.

MPEG-2 also uses the progressive method that displays all lines sequentially from top to bottom, as shown in Figure 2.2 b. The progressive method is a method for representing frames that supports a wide area of applications such as DVD, digital TV.

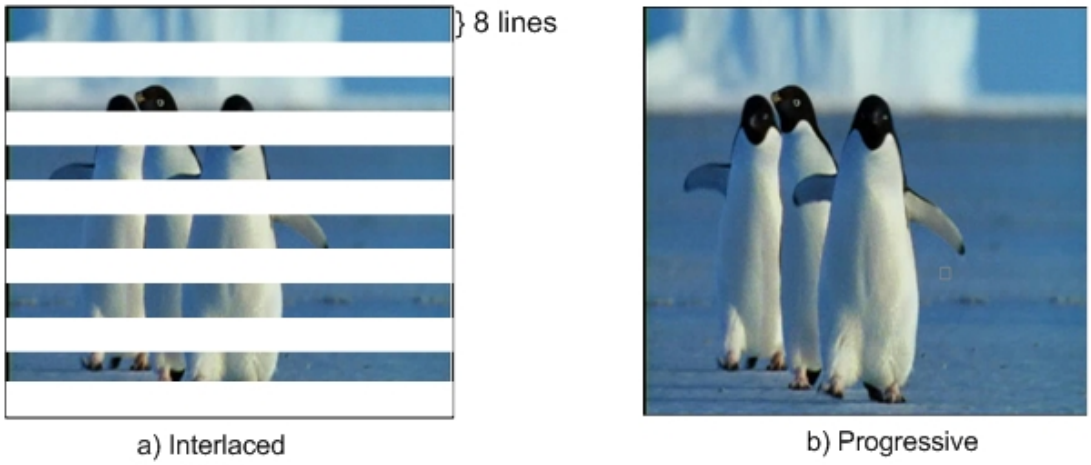


Figure 2.2: Frame representing a) Interlaced and b) Progressive

2.2 Structure of MPEG-2 Video

A video stream is a multilayer that contains video headers and video data. This means that the video bitstream has hierarchical structure consisting of six layers: a video sequence, GOP, Picture (frame), a slice, a macroblock, and a block. Each sequence, each GOP, each picture, and each slice has a header that is followed by the contents of the lower layer. Each layer includes some syntax as summarized in Table 2.1. The hierarchical structure of the video stream is shown in Figure 2.3.

Block	A block of 8x8 neighboring pixels. Intra frame coding techniques are applied on these blocks.
Macroblock	A macroblock contains a number of blocks depending on the chroma format that is used. For example, when the 4:2:0 chroma format is used, a macroblock consists of 4 luminance blocks and 2 chrominance blocks. Inter frame coding techniques are operations executed on a macroblock scale.
Slice	A slice consists of an arbitrary number of macroblocks. Slices offer a mechanism for synchronization and thus limit the propagation error.
Picture	A picture consists of an arbitrary number of slices.
Group of Pictures	Group of pictures is, as the name already suggests, a collection of pictures. A group of pictures generally starts with an I-picture. Between two succeeding I-pictures, an arbitrary number of P and B-pictures may occur in any order. The group of pictures mechanism provides a mechanism for achieving requirements like fast-forward, fast-reverse and instant play.
Sequence	A sequence consists of an arbitrary number of GOPs

Table 2.1: Entities inside a MPEG-2 video stream.

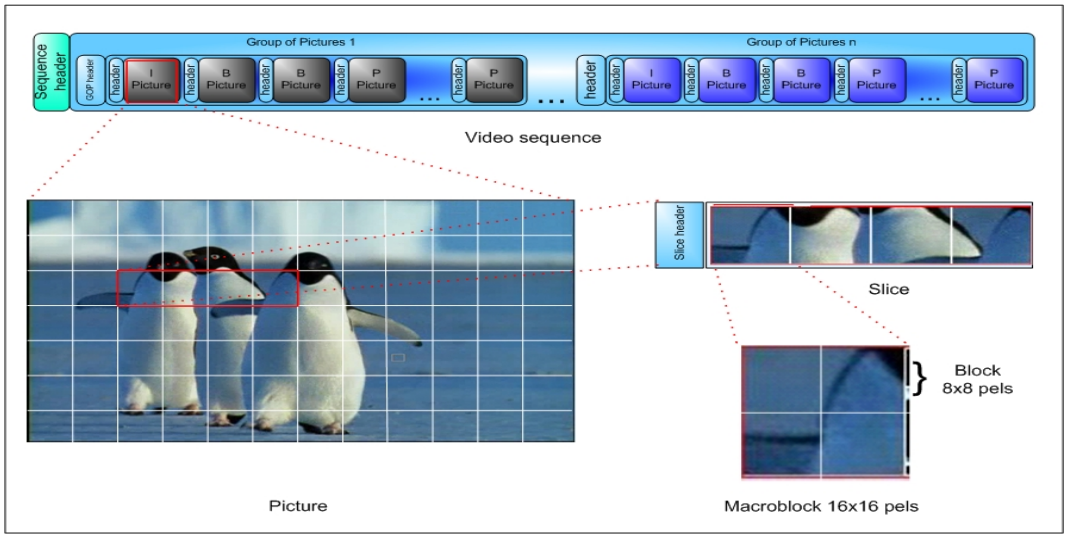


Figure 2.3: structure of the coded video bit stream

The MPEG-2 video stores three kinds of frames: intra (I) frame, predictive (P) frame, and bi-directional (B) frame. Reference frames are reconstructed frames that were coded in the form of an I-frame or a P-frame. Reference frames are used as a future reference and a past reference when P-pictures and B-pictures are decoded.

1. The I-frame is encoded completely independent and does not need any past references or any future reference frames. Therefore, the I-frame is reconstructed without any references to another frame. The I-frame usually occurs at the beginning of each GOP. The encoding scheme used here is similar to that of a JPEG compression [14].

2. P-frame is predicted from the last reference frame in the past. Thus, it is impossible to reconstruct this frame without any reference frames. A P-picture can get higher compression than any I-picture.

3. B-frame is predicted from the reference frame in the past, the reference frame in the future or both. It can get an even higher compression than any P-picture.

The number of frames for each GOP may be different in the same sequence. Therefore, standard MPEG-2 uses two parameters (N, M) that define the GOP structure as follows:

$$1 < M < 16 \tag{2.1}$$

$$1 < N \tag{2.2}$$

$$M\%N \ == 0 \tag{2.3}$$

M indicates the total number of frames in the GOP. N indicates the distance between two reference frames. The GOP is graphically displayed in Figure 2.4. The Figure shows the GOP that contains 9 frames and M is 3 in this GOP.

In a bitstream, the order of the coded frames are called coded order. A decoder uses the coded order to reconstruct the bitstream. As mentioned before, an I-frame is predicted independently of other reference frames. A P-frame is predicted relative to the past reference frame. The B-frame is predicted from reference frames. Thus, an encoder and a decoder already have processed and stored the reference frames (I-frame and P-frame), before a B-frame is encoded or decoded. Because the B-frame is predicted from the past reference frame or the future reference frame, the video frame is processed in the coded order. The order of the reconstructed frames at the output of the decoding process called the display order.

The following is an example of frames taken from the beginning of a video sequence. In this example there are two B-frames between P-frames and also two B-frames between I- and P-frames. Frame '1I'

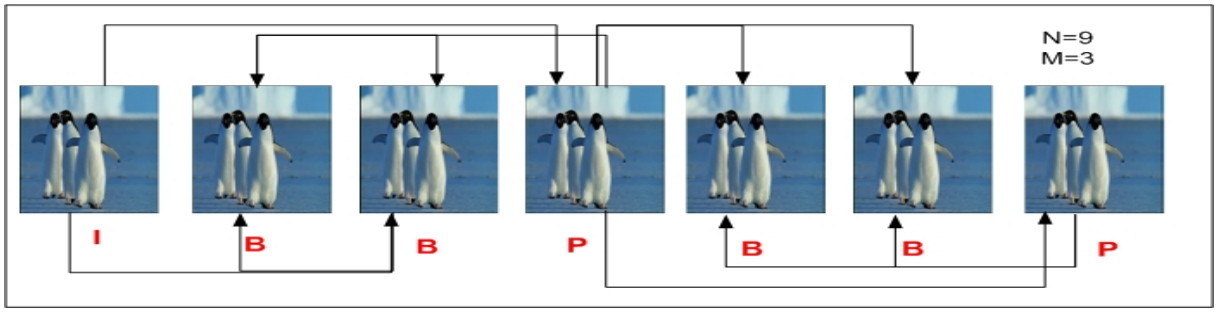


Figure 2.4: A group of picture of video sequence in display order.

is used to form a prediction for frame '4P'. Frames '4P' and '1I' are both used to form predictions for frames '2B' and '3B'. Therefore the order of coded frames in the coded sequence shall be '1I', '4P', '2B', '3B'. However, the decoder shall display them in the order '1I', '2B', '3B', '4P'. At the encoder input,

1	2	3	4	5	6	7	8	9	10	11	12	13
I	B	B	P	B	B	P	B	B	I	B	B	P

At the encoder output, in the coded bitstream, and at the decoder input,

1	4	2	3	7	5	6	10	8	9	13	11	12
I	P	B	B	P	B	B	I	B	B	P	B	B

The major advantages of the coded order that is used by an encoder and a decoder are the following:

1. The buffer size is reduced. The buffer size of an MPEG-2 decoder or an MPEG-2 encoder is only two frames (one for the future reference and one for the previous reference).

2. Low Computational Complexity.

2.3 Intra-Frame Coding

In Intra-frame coding, a coded frame only uses information from itself. Intra-frame coding uses the spatial redundancy that exists between adjacent pixels of a picture such as the sea, sky...etc. Intra-frame coding is used within a single picture if there is redundancy in the same picture. For example, if the picture includes an area such as a sea, a sky or a wall, the pixels in this area will have approximately the same color. MPEG-2 uses this spatial redundancy by dividing a picture into blocks and dividing this block. If some pixel point in the same block, it have nearly the same color. Picture coded using intra frame coding only, are called I-pictures. Intra-frame coding is based on three steps as follows:

1. DCT.

2. Quantization.

3. VLC.

2.3.1 Discrete Cosine Transform (DCT)

Modern compressing video formats such as MPEG, JMPEG...etc often uses Discrete Cosine Transforms(DCT). In a video sequence, any frame often shows a certain level of correlation with neighboring pixels in the same frame. There is a correlation between values of pixels in the same frame. This correlation can be used to forecast the value of a pixel from its respective neighbors. The process that converts a pixel block into a frequency coefficient matrix is called a transformation. *The major benefit of the transformation process is that it manages video information in a way that is easy to compress and manipulate.* This leads to de-correlated DCT coefficients that can be encoded independently and keeping compression efficiency at the same time.

One-Dimensional DCT/IDCT

The definition for one-dimensional sequence with N elements, $C\,[u]$, is given by equation 2.4:

$$C[u] = \alpha(u) \sum_{x=0}^{N-1} f[x] \cos\left(\frac{\pi(2x+1)u}{2N}\right),\ for\ x = 0, 1, 2, ..., N-1. for\ u = 0, 1, 2, ..., N-1. \quad (2.4)$$

Where $f\,[x]$ is the element number x in the sequence. Similarly, the inverse Discrete Cosine Transform, $f[x]$, is given by the equation 2.5:

$$f\,[x] = \sum_{u=0}^{N-1} \alpha(u) C\,[u] \cos\left(\frac{\pi(2x+1)u}{2N}\right) \quad (2.5)$$

In both equations 2.4 and 2.5 $\alpha(u)$is defined as in the equation:

$$\alpha(u) = \begin{cases} \frac{1}{\sqrt{N}} & u = 0 \\ \sqrt{\frac{2}{N}} & u \neq 0 \end{cases}$$

It is clear from (2.4) that

$$C[0] = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} f[x] \quad (2.6)$$

Thus, the first transform coefficient is the average value of the sample sequence. $C[0]$ value is called a DC Coefficient. All other transform coefficients are called AC coefficients. The compressed video uses the two-dimensional DCT that uses the basic operation of one-dimensional DCT. The two-dimensional DCT is introduced in next subsection.

Two-Dimensional DCT/IDCT

The two-dimensional DCT (2D-DCT) equation converts a pixel block into a frequency coefficient matrix because *the matrix in the frequency domain is more efficient to compress and manipulate.* Each picture can be coded after being partitioned into blocks (8x8). Then, each block is transformed into a frequency domain by applying 2D-DCT. The block is coded the DCT coefficients by using the same number of bits for each block. Therefore, DCT technique does not result in any bit reduction.
The benefits of DCT uses as follows:

1. DCT coefficients become more suitable for bit-rate reduction. Some DCT coefficients values become zero or near zero that can be used in the next step. It is shown in Figure 2.5.
2. DCT separates out the high frequency coefficients and the low-frequency coefficients that present the block. Figure 2.5 shows that.
3. The frequency domain is more concerning compression and manipulation.

In MPEG-2, the block, $X[8][8]$, is transformed into matrix $Y[8][8]$ by equation 2.7

$$y[m,n] = \frac{1}{4}\alpha(m)\alpha(n) \sum_{i=0}^{7}\sum_{j=0}^{7} x[i,j] \cos(\frac{(2i+1)m*\pi}{16}) \cos(\frac{(2j+1)n*\pi}{16}) \quad (2.7)$$

Where $\alpha(x) = \begin{cases} 1 & x > 0 \\ 1/\sqrt{2} & x = 0 \end{cases}$

Transformation is shown in Figure 2.5. In the figure, the low-frequency DCT coefficients store the detail of the picture. The high frequency DCT coefficients enhance the level of detail of the picture. Finally, DCT is theoretically lossless. Nevertheless, in practice, DCT is lossy due to round off errors.

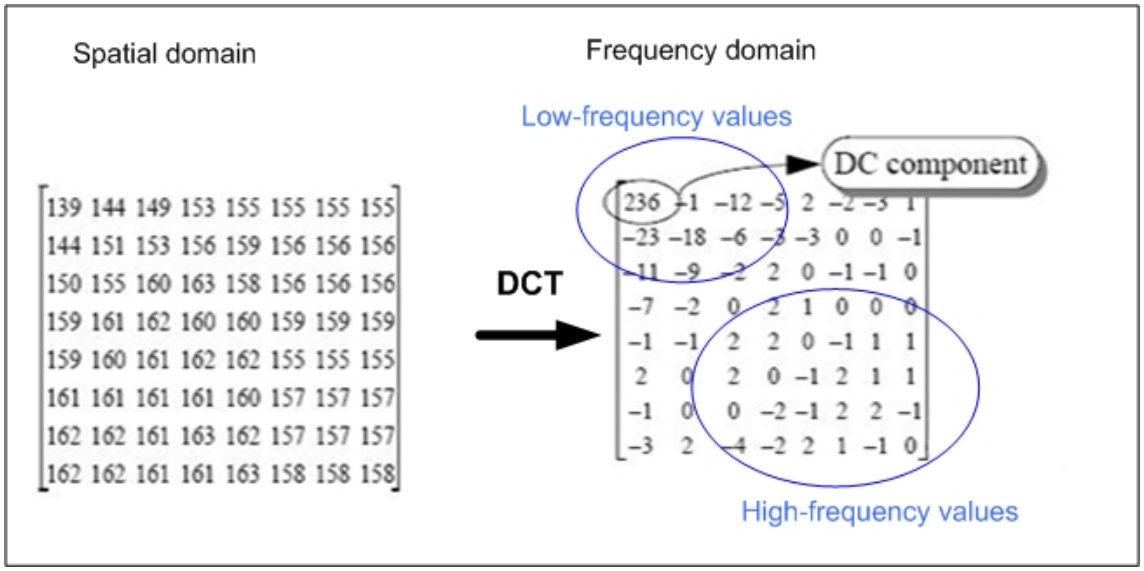


Figure 2.5: Pixels block is transformed into the frequency domain by using 2D-DCT.

2.3.2 Quantization

Quantization allows to remove some information in the high frequency coefficients. This is done by dividing each coefficient in the frequency domain by a quantization step, and then rounding to the nearest integer. Therefore, some data can be lost. The important characteristic of quantization is that low-frequency DCT coefficients become zeros after quantizing. MPEG-2 uses two quantization methods. These methods are as follows:

1. Quantization matrix approach.
2. Quantizer scale approach.

During the process of quantization, an encoder and a decoder can use one quantization matrix. MPEG-2 has two quantization matrixes:

1. Intra quantization matrix.
2. Non-intra quantization matrix.

The purpose of these matrixes is to quantize the high-frequencies coefficients with coarser quantization steps that will suppress high-frequencies with no subjective degradation, thus taking advantage of human visual perception characteristics [15]. A quantizer scale value may be changed per macroblock. A bit rate can modify a quantizer scale value in a way that keeps picture quality high for a given bit rate.

In this project, we use a quantizer scale value that allows to control the quality of the output video based on a given bit rate. Furthermore, the quantizer scale value must provide a variable quantization to keep high compression ratio. Therefore, the amounts of output data can be decreased or increase by raising or lowering value of the quantizer.

For example, an intra-macroblock is, a DC value, which is ac(0,0), is 11 bits before quantizing. This DC will reduce to 8, 9, or 10 bits as follows:

$$8-bit:\ QDC = int(dc/8) \tag{2.8}$$

$$9-bit:\ QDC = int(dc/4) \tag{2.9}$$

$$10-bit:\ QDC = int(dc/2) \tag{2.10}$$

$$ac'(i,j) = int((16 * ac(i,j))/W_I(i,j)) \quad i \neq 0\ and\ j \neq 0 \tag{2.11}$$

$$QAC(i,j) = (ac'(i,j) + sign(ac'(i,j)) * ((3 * quantizer\_scale)/4))/(2 * quantizer\_scale) \tag{2.12}$$

Where: QDC is the quantized DC value, and $W_I(i,j)$ is the element at (i,j) index in the intra quantization matrix. *quantizer_scale* is calculated for each macroblock by the control bit rate algorithm.

2.3.3 Variable Length Coding (VLC)

Zigzag ordering is used to convert a two-dimensional array into a one-dimensional sequence. This order collects the low-coefficients and mid-coefficients together respectively. DCT coefficients are read from the block, these coefficients are ordered as sequential strings of a number. This sequence (output stream) can contain strings of zeros s due to the compression of data in the low-frequency DCT coefficients and the quantization. Coefficients of high-frequency are expected to be zero. For this reason, the coefficients generally have a zigzag order (see the left side of Figure 2.6). MPEG-2 also offers an alternative scan order (see the right side of Figure 2.6).

For example, the below block elements are scanned by the zigzag order and the alternative order. The zigzag order and the alternative order converts a two-dimensional block to a one-dimensional sequence, as shown in Figure 2.7.

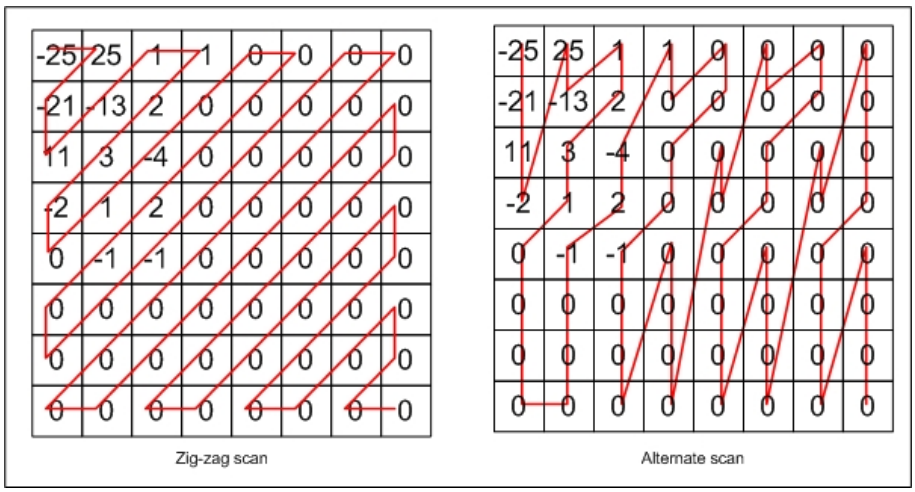


Figure 2.6: scan order

index	0	1	2	3	4	5	6	7	8	9	10	11	12	....	17	18	19	...63
Zigzag order	-25	25	-21	11	-13	1	1	2	3	-2	0	1	-4	0	0	2	-1	0
Alternative order	-25	-21	11	-2	25	-13	1	2	3	1	0	0	0	0	2	-4	1	0

Figure 2.7: Zigzag and Alternative scan converts a two-dimensional array

In VLC, two different operations are applied on DCT blocks:

1. Run-length coding does not code DCT coefficients that have the value zero. After ordering, an algorithm that counts the number of occurring zeros prior to a non-zero coefficient is applied. The pairs of numbers, which consists of a number of zeros prior to the non-zero coefficient and the coefficient, are mapped into a code word using a VLC table. The number of zeros is also known as the run-length (or simply run) and the procedure is therefore referred to as run-length coding.

2. Entropy coding replaces coding each DCT coefficient in a block by a code with the fixed length of only a few bits, that represents the same value as the original quantized DCT coefficient [15].

The final step of coding an intra block is to apply VLC process to the quantized DCT coefficients to reduce the number of bits. VLC is a process of coding and gives short code words to frequently repeated numbers and longer code words to less repeated numbers.

2.4 Inter-Frame Coding

The previously discussed intra-frame coding technique is limited to processing the video contact within one video frame only. However, that is too expensive and an encoder is not able to achieve a high compression ratio. Consequently, MPEG-2 standard includes an inter-frame coding technique in order to improve coding efficiency. This technique uses information both from itself and from macroblocks in others pictures. They have been encoded in past. This technique uses similarities between pictures to reduce a video size. The number of pictures that a video sequence consists of contains a lot of redundancy. Removing this redundancy leads to better compression ratios, resulting in the video taking less memory capacity and a lower for transmission.

In inter-frame coding technique, the picture is not sent as a whole, but rather the difference between the current picture and the reference only is sent. In MPEG-2, predictions are formed on a macroblock basis, where a macroblock is defined as a block of 16x16 pixels.

A MPEG-2 encoder uses Motion Estimation (ME) that is the process of estimating Motion Vectors (MVs) between the current picture and the reference picture during the encoding process. MV is a two-dimensional vector used in Motion Compensation (MC) that gives an offset from the coordinate position in the current picture to the coordinates in a reference frame. Motion Compensation (MC) uses MVs to improve the efficiency of the prediction of sample values. The prediction uses MVs to provide offsets into the past and/or future reference frames containing previously decoded sample values that are used to form the prediction error.

ME compares each macroblock to some macroblocks in the reference frame using some errors measure such as mean squared error (MSE) or sum-of-absolute-differences (SAD), and the best matching macroblock is selected. The search is conducted over a fixed window. The location of the macroblock in the reference frame might be different from the location of the macroblock in the current frame. MVs provide an offset from the coordinate position in the current picture to the coordinates in reference picture. MVs have to be encoded in a bitstream. The B-frames and the P-frames can refer to the I-frames as reference frames. For the P-frames, only the earlier frame is searched for matching macroblocks. In the B-frames, the earlier and future frames are searched for matching.

For example, the MPEG-2 encoder contains ME, which measures the direction and distance of any motion between macroblocks of the current frame and the macroblocks of the reference frame in the form of motion vectors. When most of the current frame content is transmitted to the reference frame, the encoder only sends the motion vectors. When a decoder receives the motion vectors, it uses them to shift data in a reference to resemble the current frame more closely. The disadvantages of Motion Estimation are:

1. An increase in computational complexity.
2. Dependency of optimal motion vectors on neighboring macroblocks.

MC describes any frame in terms of the locations where each macroblock of the current frame appearing in reference frames. MC uses MVs to improve the efficiency of prediction. The prediction uses MVs to provide offsets into the reference frames containing previously decoded frames. Matching macroblocks are employed by using MVs to build a prediction frame that is compared to the current frame in the same way.

In MPEG-2, predictions are formed on a macroblock basis, where a macroblock is defined as a block of 16x16 pixels. Inter-frame coding is based on three predictions:

1. A forward predicted: A forward predicted macroblock depends on decoded pixels from the previous reference picture. Forward prediction can be used to code macroblocks in P-pictures and B-pictures. This is shown in Figure 2.8a. For a P-picture, both I-pictures and P-pictures can serve as a reference picture.
2. Backward Prediction: A backward predicted macroblock depends on decoded pixels from the immediately next reference picture. A backward prediction can only be used to code macroblocks in B pictures.

3. Bi-directional Prediction: a bi-directional predicted macroblock depends on decoded pixels from the reference pictures immediately next and immediately previous. Bi-directional Prediction can only be used to code macroblocks in B pictures. Inter-frame coding for B-pictures is graphically depicted in Figure 2.8b.

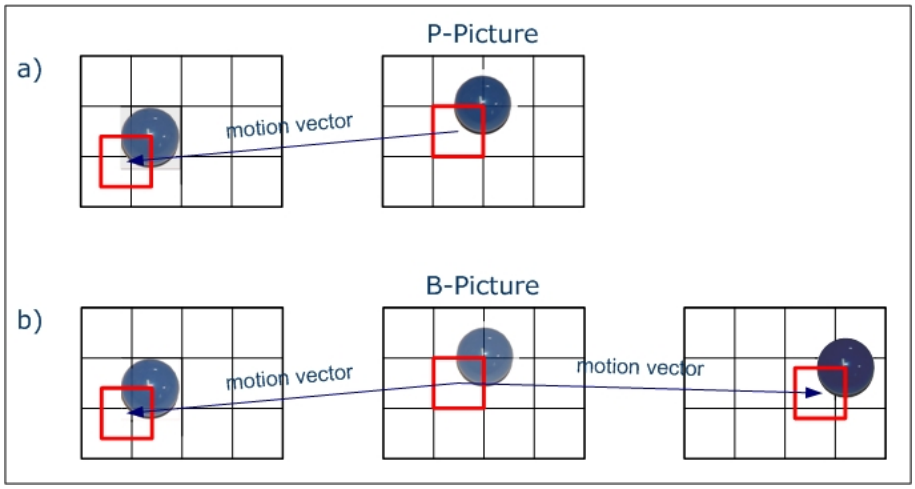


Figure 2.8: a) inter-frame coding for P-picture. b) inter-frame coding for B-picture.

In this project, we will use MVs and DCT coefficients that are produced by the MPEG-2 encoder. Therefore, computational requirements are reduced by eliminating the process of extra decompressing.

2.5 Decoder Architecture

In order to decode a bitstream it is generated from the MPEG-2 encoder. The MPEG-2 decoder is required to reverse the order of the process of encoding. Decoding is the process defined in the MPEG-2 standard specification that reads an input coded bitstream and produces decoded pictures. The functional diagram of the decoder is shown in Figure 2.9. Firstly intra-macroblock, VLD produced the quantized DCT coefficients block after bitstream has arrived. After that DCT coefficients block is produced by Inverse Quantization (IQ). Finally, IDCT transforms this block into the special domain.

Secondly non-intra-macroblock, the non-intra frame decoder was reconstructed upon the essential structure blocks of the intra-macroblock decoder. The non intra-macroblock decoder contains the same core structure as the intra-macroblock decoder with the addition of motion compensation support. Again, support for intra-macroblock decoding is inherent in the structure, enabling I-frame, P-frame, and B-frame decoding is possible. Finally, I-pictures and P-Pictures are stored in two frame buffers in the decoder and they form the basis for the prediction of P-pictures and B-pictures.

2.6 Encoder Architecture

Encoding processes read a stream of input pictures and generate a valid coded bitstream as defined in the MPEG-2. However, the design and implementation of MPEG-2 encoder is open. Therefore, the encoder should generate a MPEG-2 video bit stream that can be used by the MPEG-2 decoder to meet the requirements of the MPEG-2 standard. This section gives an overview of the MPEG-2 encoder.

The MPEG encoder architecture is shown in Figure 2.10. Firstly, the DCT coefficients were calculated for each macroblock in the input picture. Furthermore, they are quantized. If the picture is I- or P- frame it is stored in the frame memory at the buttom of the figure for future reference. ME uses these pictures for calculating MVs within the current picture. ME sends MVs to MC unit so that a motion-compensated prediction block can be supplied to the subtracter at the top of the Figure. ME and MC are based on macroblocks (16x16), whereas DCT, Quantizer and VLC are process (8x8) DCT blocks.

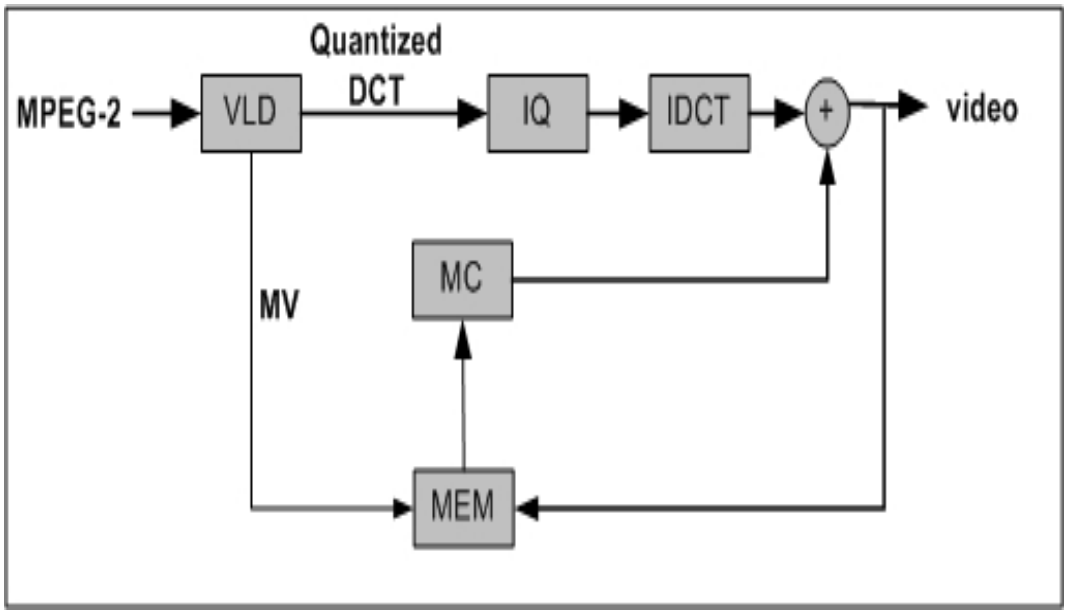


Figure 2.9: The MPEG-2 decoder architecture.

The MPEG-2 encoder contains a prediction error loop that performs an internal decoding, thus Inverse Quantization (IQ) and Inverse DCT (IDCT) are needed to make the reference frames in the special domain.
Any input GOP starts with an I-picture and it is processed in coded order as follows:

1. Intra-frame encoding of the I-Picture and after rebuilding, is stored in the reference picture memory as a past reference.
2. Inter-frame encoding of the P-picture and after being rebuilt, is stored in the reference picture memory as a future reference.
3. Sequential inter-frame encoding of the B-picture, which is between the two references frames, is directly supplied to the output and is not stored.
4. If the current frame is a P-picture, the future reference picture of Step 2 becomes the past reference picture, and returns to Step 2.
5. If the GOP ends, restart with Step 1.

A bit rate control algorithm (RC) controls the size of compressed video. This means that the RC maintains the bit rate of compressed video close to the average target bit rate. Therefore, RC is important for encoding. RC produce a quantizer that determines the target bit rate. For more details about the rate control algorithm see Chapter 4. In this project, we focus on the MPEG-2 transcoder architecture. The most simplified transcoder architecture consists of decoder and encoder. Therefore, they are introduced in this chapter.

2.7 MPEG Scalability

The model of MPEG-2 can be divided into two main types: non-scalable methods are structured like a super set of the model defined in ISO/IEC 11172-2, the main property being the additional compression tool for interlaced video [14]. The second type is the scalable methods. The purposes of this is to provide a coded video at multiple levels, *which are pre-defined at the time of encoding*, and a multilayer transmission for increased signal robustness. The encoder divided the complete bitstream into two or more layers. And then the decoder that rebuilds the complete bitstream starts from a base layer and adds a number of enhancement layers. Thus, the scalable methods allow rebuilding a useful video from

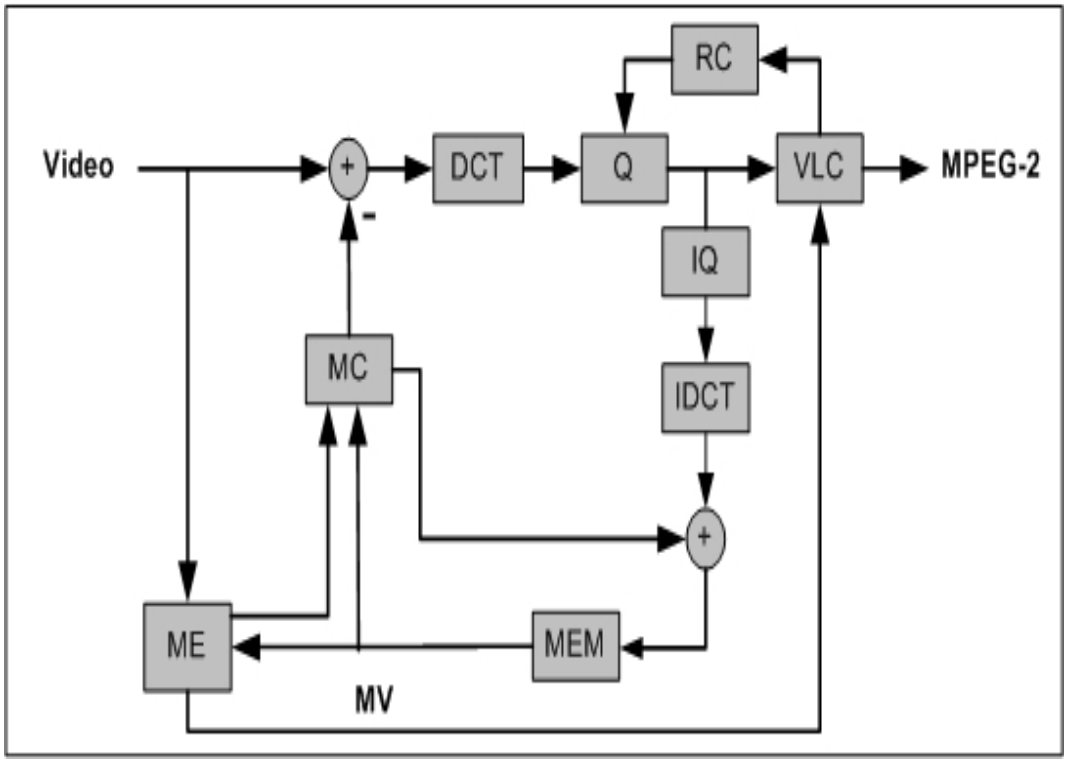


Figure 2.10: The MPEG-2 video encoder architecture.

the parts of entire bitstream [1]. Furthermore, the decoder generates more than one quality from single bit stream. The disadvantage of the scalability is that the effectiveness of coding is lost because of the additional overheads. The scalability has several scalable methods as follows:

1. Data partitioning.
2. SNR (signal-to-noise ratio) scalability.
3. Spatial scalability.
4. Temporal scalability.

We will briefly introduce the basic notions of the features of scalability below.

1. *Data partitioning* separates a bitstream between two layers into more important parts of the bitstream such as headers, motion vectors, and low-frequency DCT coefficients. They are coded in the base layer with the better error performance as shown in Figure 2.11. Less important data such that high-frequency DCT coefficients are coded in the enhancement layer with lower error performance. Thus, the errors of a bitstream are minimized since the important parts of the bitstream are better protected.

2. *SNR scalability* includes generating two video layers of same spatial resolution. Each layer has different video quality. These layers have the same spatial (frame size) and temporal (frame rate) resolution. As mentioned before, the base layer is coded by itself to provide the basic video quality and the enhancement layer is coded to enhance the base layer. In SNR scalability, the DCT coefficients are coded with high quantization step size for the base layer where they have high bit rate. Also the DCT coefficients are coded with low quantization step for the enhancement layer where it have low bit rate as show in Figure 2.12.

3. *Spatial scalability* uses different spatial resolution video layers from a single video source resulting in the base layer being coded by itself. This provides the basic spatial resolution and the enhancement layer containing the update for achieving full resolution as in Figure 2.13.

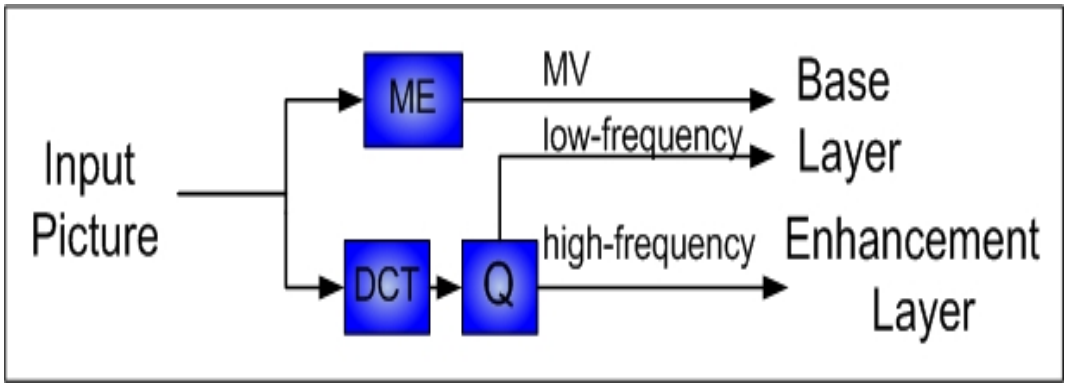


Figure 2.11: Principle of data partitioning.

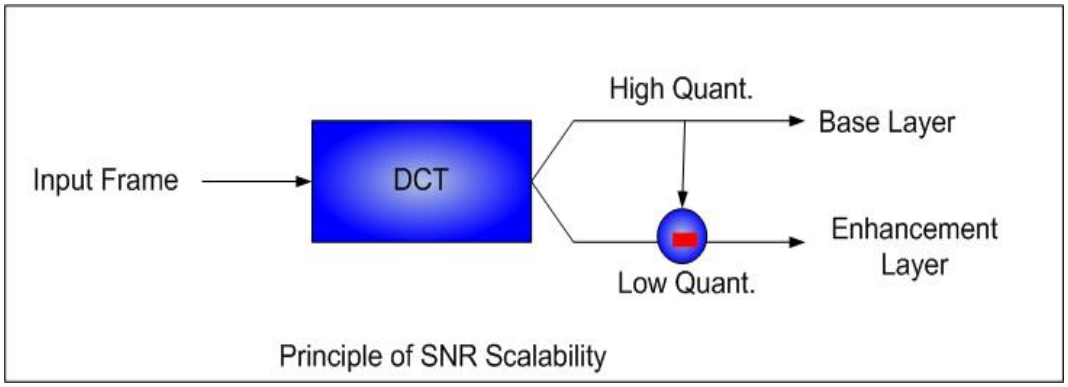


Figure 2.12: Principle of SNR scalability

4. *Temporal scalability* involves partitioning of video frames into layers. Each layer provides different frame rates. A base layer controls the frames for a certain frame rate, and the enhancement layer controls the frames that are needed to increase the frame rate as in Figure 2.14. The base layer and the enhancement layer use the same spatial resolution.

The type of scalability used in this project, it is an adapted version of SNR scalability. The SNR transcoder should be able to produce a maximum of two enhancement layers in this project. Furthermore, the enhancement layers coded by the adapted version have the following characteristics:

1. Frames of an enhancement layer have no reference to other frames from the same enhancement layer.

2. Frames of an enhancement layer have no reference to frames from previous layers.

3. The first frame in an enhancement layer is an I-picture. The last frame is a P-picture. All the pictures in between are B-pictures.

These characteristics enable a decoding of an enhancement layer independently from other layers, using a legacy MPEG-2 decoder. Furthermore, it allows for the dropping of frames from an enhancement layer arbitrarily, without resulting in artifacts in the perceived video. This makes it possible to adapt to fluctuations in the available bandwidth of the wireless network. After being received, the decoded frames from the enhancement layers are added to the base layer to enhance the quality.

The base layer of this scalable video stream is generated by IQ the DCT coefficients such that the target bit rate for the base layer is satisfied. The IQ errors of a previous layer are stored in the enhancement layers.

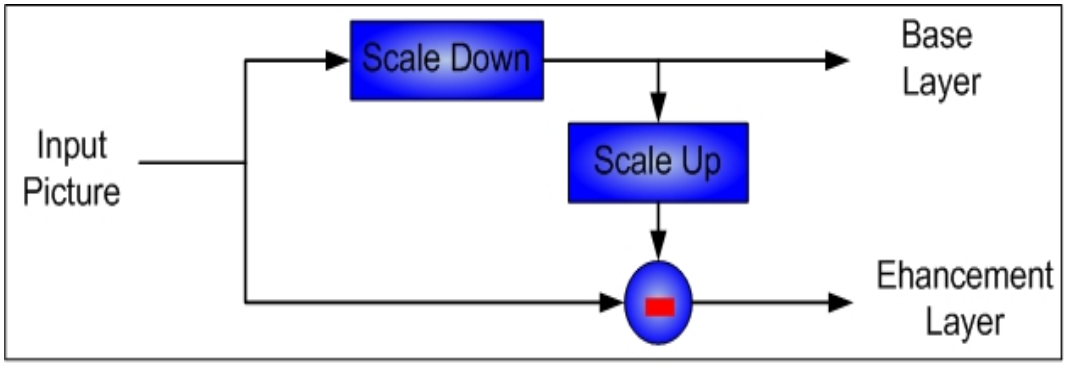


Figure 2.13: Principle of Spatial scalability

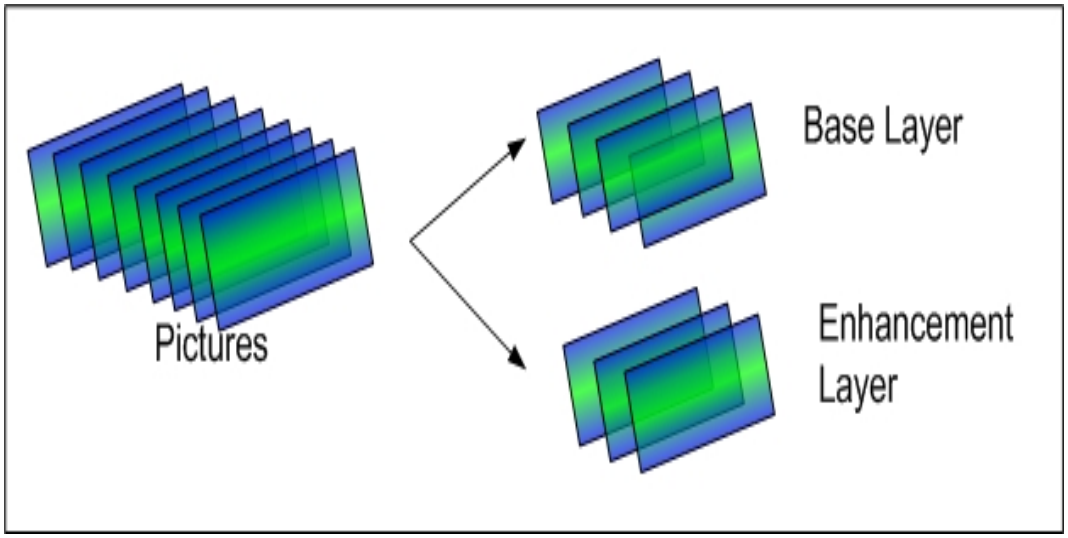


Figure 2.14: Principle of temporal scalability

2.8 Conclusion

The MPEG-2 video efficiently uses the compression video algorithm that takes full advantage of the correlation between times and frame. Thus, MPEG-2 does not contain a data redundancy,even though it can be between an input frame and reference frames by using motion compensation (MC). Moreover, the MPEG-2 algorithm removes a special redundancy within the frame by using the Discrete Cosine Transform (DCT).

The MPEG standards are called hybrid video coding, because temporal and spatial redundancy of video frames are achieved. The MPEG-2 encoder compresses video accordance to the standard MPEG-2. The MPEG-2 decoder decompresses the MPEG-2 video to create high video quality that is displayed at a video monitor.

Chapter 3

Video Transcoder System

Video transcoding is the operation of converting a compressed video from one format into another one with different properties, such as different frame rate, bit rate, frame size, or compression standard. For example, MPEG-2 is transmitted to client over a heterogeneous network. Capacities and characteristics of this network are often unknown when this video is compressed. Hence, the compressed video is usually not able to meet an available bandwidth. Therefore, this video needs to convert into a lower bit rate to meet the available bandwidth. Video transcoding is one of the most important techniques that apply a bit rate reduction in any compressed video to meet the available bandwidth over heterogeneous networks.

In this project, a MPEG-2 video stream is transcoded into a SNR scalable MPEG-2 stream. This means that a high bit rate video must be converted into scalable MPEG-2 when the MPEG-2 video stream is required. In project, the encoder embedded in the transcoder is only compressing at a lower bit-rate than the incoming bit rate, so there is no other reformatting involved, such as re-sampling. Therefore, the transcoded MPEG-2 video has the same GOP structure as the pre-encoded MPEG-2 video. In this chapter, the transcoding methods discussed. The chapter explains techniques of bit rate reduction that are used to create SNR scalable MPEG-2. Furthermore, this chapter includes one section that contains some video transcoder architectures. They are used to apply the techniques of the bit rate reduction.

3.1 Video Transcoder

Video transcoding is used in video servers and a number of applications, for example mobiles and PDAs. These applications often use a compressed video to save the available bandwidth and to receive high quality picture. Some techniques can work the same as a transcoder. For example, a video server could store multiple copies of the same video that were encoded at different bit rates, resolutions or formats. However, this technique has disadvantages such as any video server that offered limited choices and expansive solutions because the server requires large memory storage. Therefore, the transcoder is the best technique for streaming video over a heterogonous network for following reasons:

- The transcoder is used to increase the robustness of video streaming against packet loss in networks.
- The transcoder provides a scalable video content over heterogeneous networks in order to adapt with the available bandwidth.
- The transcoder is used to distribute the same video content to a group of devices simultaneously over heterogeneous networks.

As mentioned in Chapter one, a number of devices used in the home network such as laptops, mobiles, PDAs $\cdots$ etc. most of end users prefer to use the hand devices. These devices have limited computing and displaying capabilities that are not suitable for high quality video decoding and displaying. Therefore, a compressed video must be converted into a lower quality video for displaying on mobiles devices. Moreover, when the compressed video is streamed to a number of end users through channels with various capacities, this video must be converted into a specified bit rate to meet the available bandwidth for each channel. Therefore, a scalable transcoder can be installed on a video server where the compressed

video is stored. Furthermore, the scalable transcoder can be installed on an intermediate node between video servers and clients.

The basic tasks of the transcoding are classified as follows:

- Frame size conversion: when the compressed video has a big frame size, it needs to be transcoded to small frame size. The transcoder converts the frame size according to the end user device ė.g. most portable devices can only display small size frame.
- Bit rate reduction: convert the video bit rate according to the available bandwidth. The compressed video is often encoded at high bit rate. For low bandwidth channel, the video bit rate needs to be transcoded into a lower bit rate
- Frame rate conversion: changes because the some portable devices can only play video at low frame rate, such as 15 or 20 fps (frame per second). This means that the high video frame rate needs to be transcoded into a lower frame rate. Furthermore, frame rate conversion can also reduce the bit rate.
- Coding syntax conversion: converts the compression standards, such as from MPEG-1 to H.261, $\cdots$ etc.

In this project, the bit rate reduction, which is used, converts an MPEG-2 video steam to a SNR scalable MPEG-2 stream according to the channel conditions; such that the individual layers of the scalable video stream have a lower bit rate than the input stream.

3.2 Bit Rate Reduction

As mentioned before, the bit rate reduction converts the pre-coded video of high bit rate (R1) into the video of a lower bit rate (R2). The purpose of the bit rate reduction is to reduce the bit rate while keeping low complexity and achieve the highest quality possible. The applications need this transcoding such as the internet streaming and the television broadcast $\cdots$ etc.

An encoder can not measure the channel capacities during compressing. The problem of video streaming obtains a compressed according to the channel conditions. In order to overcome this problem, the bit rate reduction technique is used to convert high bit rate vide to a lower bit rate. Therefore, the bit rate reduction is an important technique to stream the pre-encoded video over a heterogeneous network.

Figure 3.1 shows the complete life cycle of a video stream to illustrate the problem of bit rate reduction. The encoder compresses the input video at a bit rate R1 Mbps, then the transcoder converts this compressed video into the compressed video of a lower bit rate R2 Mbps, and finally the decoder decompressed this transcoded video.

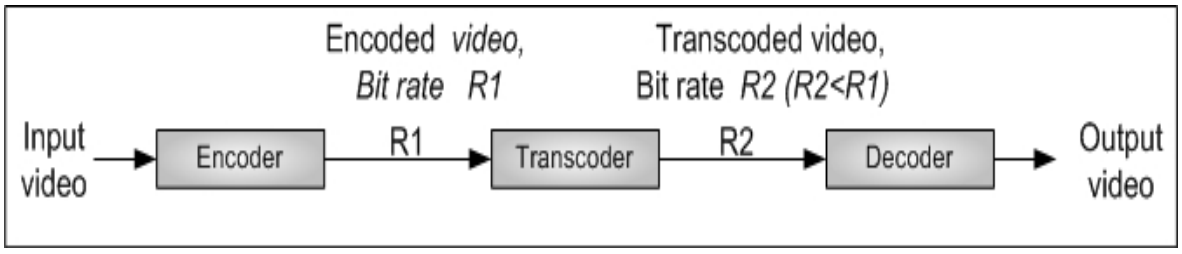


Figure 3.1: The complete life cycle of a video stream.

As mentioned before, the bit rate reduction is a solution to obtain better video quality according to channel conditions. Popular methods can be used to achieve bit rate reduction on streaming video as follows:

- Frame dropping: dynamically drops frames that are less importance for a pre-encoded video [3]. Dropping frame transcoder knows frame types, such as I-frame, P-frame and B-frame.

- Spatial resolution reduction: reduces the number of the blocks for each macroblock [3]. The transcoder converts high spatial resolution to a lower spatial resolution. For example, each macroblock (16 X 16) in the input video is reduced into one block (8X8) Therefore, the spatial resolution reduction reduces a bit rate.
- Cutting high frequencies: is a frequency filter [3]. Low DCT coefficients are transcoded into the output video stream. Other DCT coefficients, most likely high-frequency coefficients, are discarded.
- Re-quantizing DCT coefficients: de-quantizes the DCT coefficients of a pre-encoded video [3]. The transcoder re-quantizes DCT coefficients using a larger quantizer step. As a result, a small DCT coefficient value of the pre-encoded video becomes zero after that.

Table 3.1 gives the architectures of the bit rate reduction transcoders. Each architecture is described its own hardware complexity that saves over full decoding and re-encoding. In this project, re-quantizing DCT coefficients method is used to generate a lower bit rate MPEG-2 video.

Architecture of transcoder	Complexity saving
Cutting high frequencies	No decoding loop, no DCT/IDCT, no frame store memory, no encoding loop, quantizer and inverse quantizer, no motion compensation, no VLC, simplified rate control
Re-quantization	No decoding loop, no DCT/IDCT, no frame store memory, no encoding loop, no motion compensation, simplified rate control.
Reducing the spatial resolution	No motion estimation, no macroblock coding decisions
Dropping frame	No motion estimation

Table 3.1: Transcoders architectures

3.3 Transcoder Architectures

This section describes four transcoding architectures frequently occurring in the previous work [17]. The transcoding architectures will not show the rate control algorithm. They can all be used to generate the base layer of the scalable video stream in this project. This section describes the behavior, the advantages and the disadvantages of each of them.

3.3.1 The Full Cascaded Transcoder

The situation with transcoding is as shown in Figure 3.1. An encoder compresses the incoming video at a bit-rate of R1 , then this compressed signal is converted into a compressed format of a lower bit-rate R2 and finally a decoder decompresses the incoming and displays the resulting video. The most straightforward approach of transcoding a MPEG-2 video stream is to fully decode the stream and then encode the reconstructed video to the desired bit rate. This situation is depicted in Figure 3.2. The decoder and encoder depicted in Figure 3.2 can be implemented as described in section 2.5 and section 2.6 respectively.

The main advantage of using this architecture is because of its straightforward approach and simplicity. Furthermore, this approach can be easily used to transcode a video into a different video standard. The main disadvantage of this method is the high computational complexity that is required for performing all operations.

3.3.2 The Cascaded Transcoder

The cascaded transcoder consists of a decoder and an encoder as shown in Figure 3.3. Cascaded transcoding decodes the pre-encoded MPEG-2 video and then the cascaded transcoder encodes this video to meet the available bandwidth. The cascaded transcoder can achieve complexity saving via re-using information contained in a pre-encoded significant while still maintaining acceptable quality. The cascaded transcoder uses MVs of the pre-encoded MPEG-2 video, as alternative for calculating new MVs. This

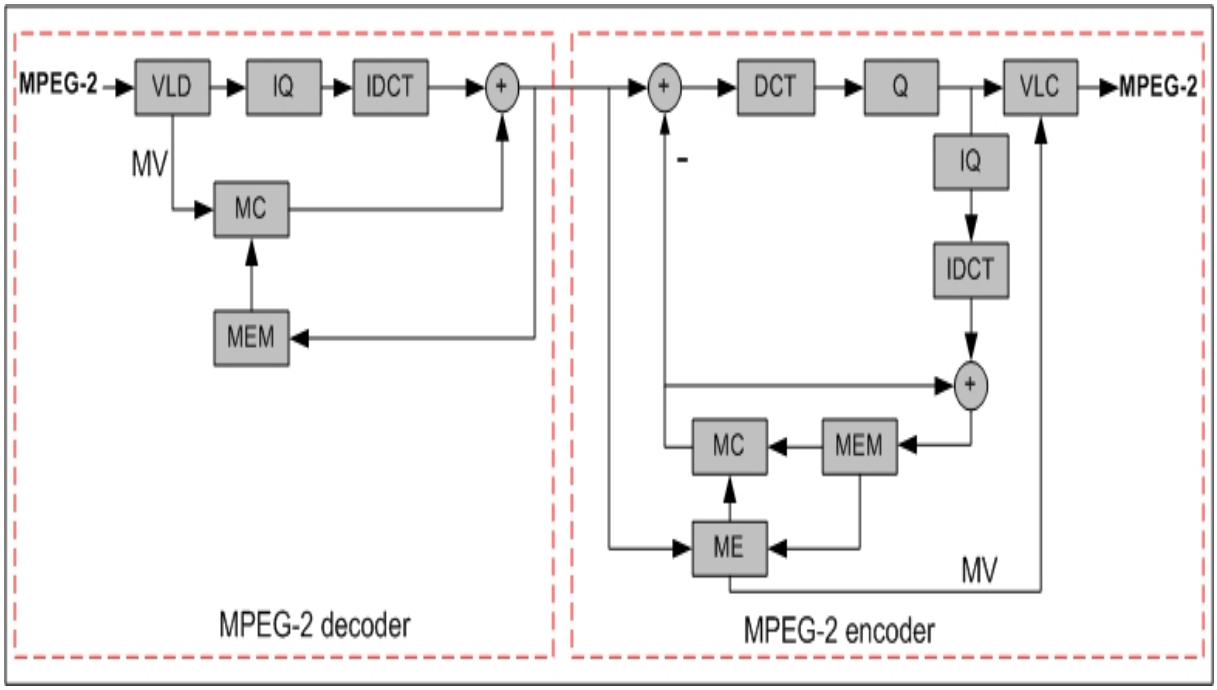


Figure 3.2: The full cascaded transcoder.

means that this transcoder does not include ME [2]. Therefore, the cascaded transcoder is not a complete encoder and a decoder.

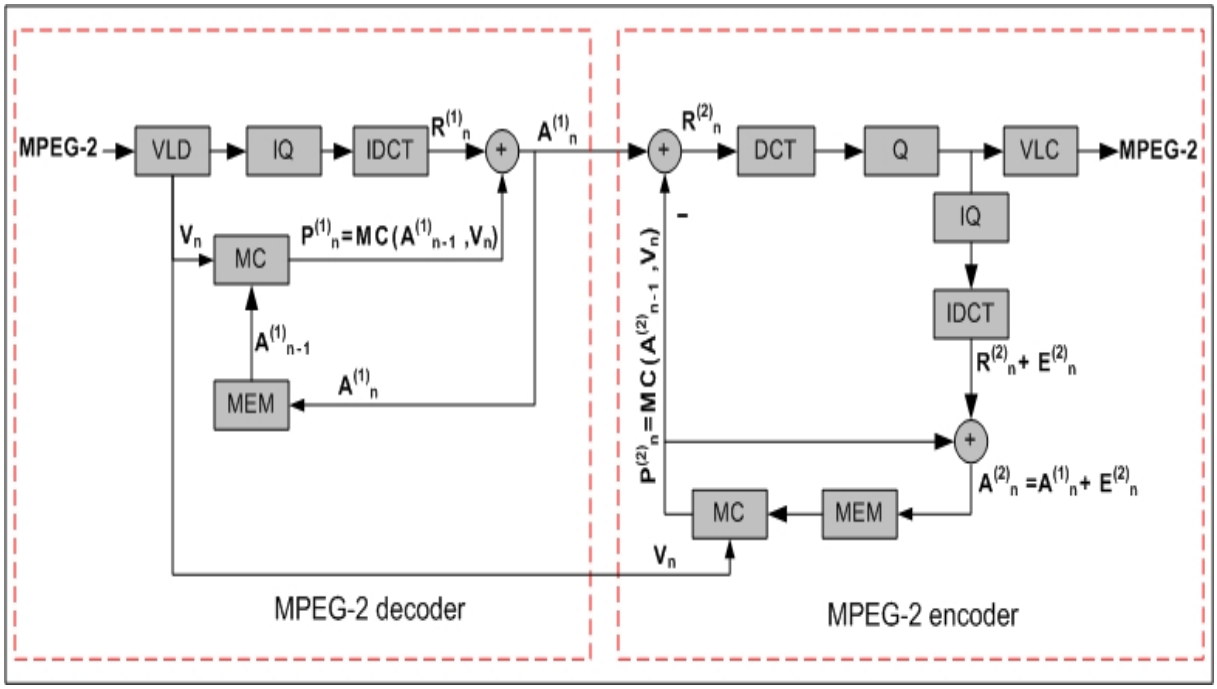


Figure 3.3: The cascaded transcoder.

Figure 3.3 shows the decoder part of the cascaded transcoder. VLD that decodes the pre-encoded video produces quantized DCT coefficients, than IQ de-quantizes these quantized coefficients, than IDCT gives the residual information ($R_n^{(1)}$) of the picture ($A_n^{(1)}$). $A_n^{(1)}$ is rebuilt by adding the prediction ($P_n^{(1)}$). $P_n^{(1)}$

results from the previously reference picture ($A^{(1)}_{n-1}$) by applying MC.

In the encoder part of the cascaded transcoder, the predication ($P^{(2)}_2$) is subtracted from $A^{(1)}_n$ that gives the residual information ($R^{(2)}_n$) of $A^{(1)}_n$. Than, DCT converts $R^{(2)}_n$ from the special domain to the frequency domain. The DCT coefficients are quantized by Q. IQ de-quantizes the quantized DCT coefficients, than IDCT rebuilds the residual information ($R^{(2)}_n + E^{(2)}_n$), where $E^{(2)}_n$ denotes the quantization error in the transcoder. The picture $A^{(1)}_n$ is rebuilt by adding $P^{(2)}_n$. $MC()$ gives $P^{(2)}_n$ using the previously reference picture ($A^{(1)}_{n-1}$). However, the picture contains quantization error $E^{(2)}_n$.

In the coded order, a cascaded transcoder receives the pre-encoded video. The decoder of this transcoder generates an output video stream in the coded order. Therefore, the cascaded transcoder does not require reorder buffers. This coded order fits for any transcoder. The encoder of this transcoder directly codes this streaming video to meet channel bandwidth. Finally, the transcoded MPEG-2 video has the same GOP structure as the pre-encoded MPEG-2 video.

The main advantages of the cascaded transcoder are:

- The cascaded transcoder can be used to convert any pre-encoded video standard into anther video standard.
- The cascaded transcoder is a free re-quantization errors (drift errors) [2]. This transcoder can generate high quality video because of decoding and re-encoding the video stream.

Drift errors give a description for reducing in the frame quality when the inter-frame will be decoded. The drift errors, which are caused by the loss of high frequency data, create a difference between an original reference frame and a reference frame.

Cascaded video transcoding does not usually perform at real-time. In this project, therefore, we prefer to have a transcoder with a lower complexity. Moreover, the new transcoder architecture has tradeoffs between a complexity and a video quality.

3.3.3 The Closed-loop Transcoder

As mentioned before, a cascaded transcoder consists of a decoder and an encoder. The cascaded transcoder complexity can be reduced. A closed-loop transcoder is less complexity than the cascaded transcoder. The closed-loop transcoder is shown in Figure 3.4. This means that the closed-loop transcoder converts MPEG-2 of a given bit rate into MPEG-2 of a lower bit rate, if MPEG-2 needs a conversion. The closed-loop transcoder works the same as the cascaded transcoder, such as the cascaded transcoder and the closed-loop transcoder use the motion vectors of a pre-encoded video.

A closed-loop transcoder is less complex than the cascaded transcoder. It has two motion compensation loops, one of that loops is removed in the closed-loop transcoder. This improves the efficiency and reduces the computational complexity.

The purpose of a closed-loop transcoder compensates an error of re-quantization (drift error). During transcoding, the re-quantization operation of the transcoder introduces drift errors. A feedback loop performs a motion compensation for the drift errors. The motion compensation depends on the drift errors that stored in a buffer(MEM).

In Figure 3.4, VLD decodes an MPEG-2 video stream, than VLD generates the quantized DCT coefficients, than the inverse quantizer (IQ) that de-quantizes these coefficients provides DCT coefficients. Finally, to reduce an input bit rate, the transcoder includes a quantizer (Q) for re-quantizing the DCT coefficients.

During transcoding, drift errors that Q is introduced is overcome by using the feedback loop in the transcoder. Therefore, IQ of the feedback loop de-quantizes any block was quantized by Q. IQ uses the same quantization step as Q. After giving the DCT coefficients, the subtracter provides the different between the first IQ and the second IQ. The output of the subtracter is drift errors.

In the transcoder, re-quantization errors (drift errors) are evaluated in the frequency domain but MC is not defined in the frequency domain. Therefore, the closed-loop compensation must be converted

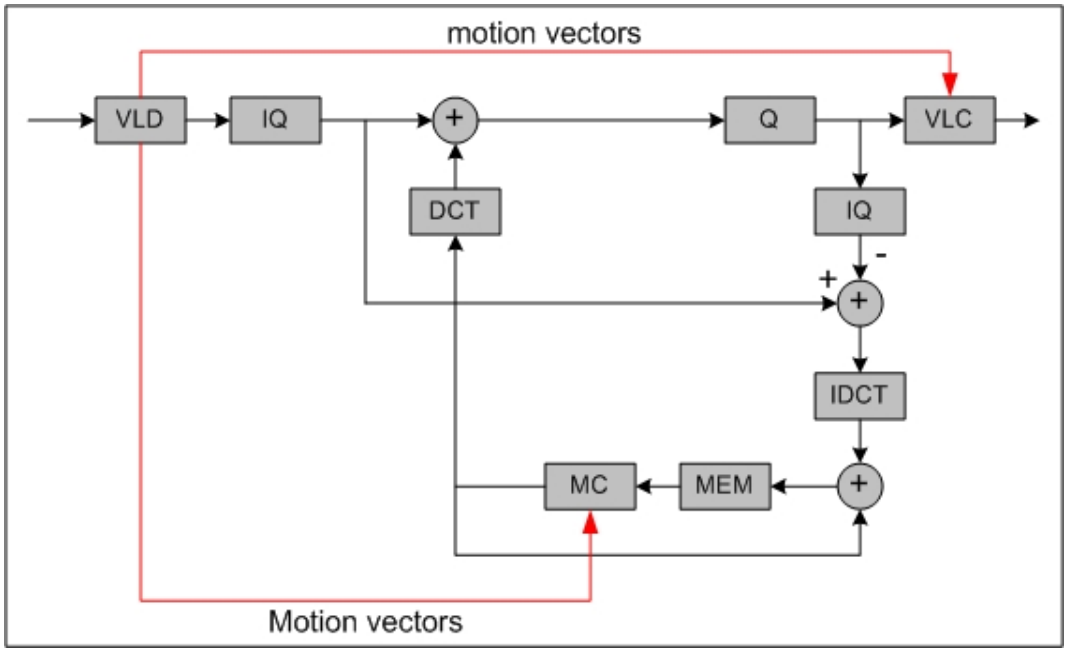


Figure 3.4: the closed-loop transcoder block diagram

between the special domain and the frequency domain. The re-quantization errors must be converted from the frequency domain to the special domain before this re-quantization errors are used in MC. Thus, this closed-loop transcoder architecture requires one IDCT that converts the re-quantization errors from the frequency domain to the special domain. The re-quantization errors are produced from MC in special domain, but a video frame is in the frequency domain. Thus, these errors have to be converted to the frequency domain, and then the errors can be compensated. Thus, the closed-loop transcoder requires one DCT. The closed-loop transcoder could not work entirely in the special domain.

A macroblock of an I-frame is encoded without a reference frame by using an Intra-frame method. A macroblock of a P-frame may be encoded according to an earlier reference frame (I-frame or P-frame). A macroblock of a B-frame may be encoded according to the previous reference frame (I-frame or P-frame) and to the next reference frame (I-frame or P-frame). The P-frames and the B-frames are transcoded according to the reference frames. Quantized DCT coefficients of the reference frames may contain drift errors. This means that every frame is used as a reference that has the drift errors after quantizing. Moreover, the drift errors will propagate if they do not compensate.

The cascaded transcoder complexity can be reduced that can produce the closed-loop transcoder. The cascaded transcoder generates the same frames types as the pre-encoded video. For example, an I-frame is transcoded as an I-frame, a P-frame is transcoded as a P-frame and a B-frame is transcoded as a B-frame. This means that the cascaded transcoder does not include a frame coding decision; therefore, its complexity can be reduced.

The cascaded transcoder complexity because of two factors, they are as follows:

1. A pre-encoded video does not require reordering in the cascaded transcoder.
2. Most of pre-encoded information can be used in the cascaded transcoder, such as motion vectors.

In the cascaded transcoder, the decoded references pictures are stored in the pictures buffers as shown in Figure 3.3. In this transcoder, the pictures buffer of the encoder contains the same reference pictures as that of the decoder. This means that reference pictures can be copied from the decoder to the encoder. The transcoder can be simplified by removing one of the two pictures buffers.

The transcoder can be simplified using fact MC is linear as follows:

$$MC(R_n^{(2)} - E_n^{(2)}, Vn) = MC(R_n^{(2)}, Vn) - MC(E_n^{(2)}, Vn) \tag{3.1}$$

In the cascaded transcoder, the output of the decoding part is the input of the encoding part as shown in the Figure. The output of the decoding part is given by

$$A_n^{(1)} = R_n^{(1)} + MC(A_{n-1}^{(1)}, V_n). \tag{3.2}$$

The residual $R_n^{(2)}$ in the encoding part is given by

$$R_n^{(2)} = A_n^{(1)} - MC(A_{n-1}^{(2)}, V_n) \tag{3.3}$$
$$R_n^{(2)} = A_n^{(1)} - MC(A_{n-1}^{(1)} - E_{n-1}^{(2)}, V_n) \tag{3.4}$$
$$R_n^{(2)} = A_n^{(1)} - MC(A_{n-1}^{(1)}, V_n) + MC(E_{n-1}^{(2)}, V_n) \tag{3.5}$$

Substitution of 3.2 into 3.5 gives

$$R_n^{(2)} = R_n^{(1)} + MC(E_{n-1}^{(2)}, V_n). \tag{3.6}$$

According to equation 3.6, this residual $R_n^{(2)}$ can be obtained directly by adding the residual $R_n^{(1)}$ from $MC(E_{n-1}^{(2)}, V_n)$. Thus, $E_n^{(2)}$ must be stored. However, the buffer used for $A_n^{(2)}$ can be remove as shown Figure 3.5.

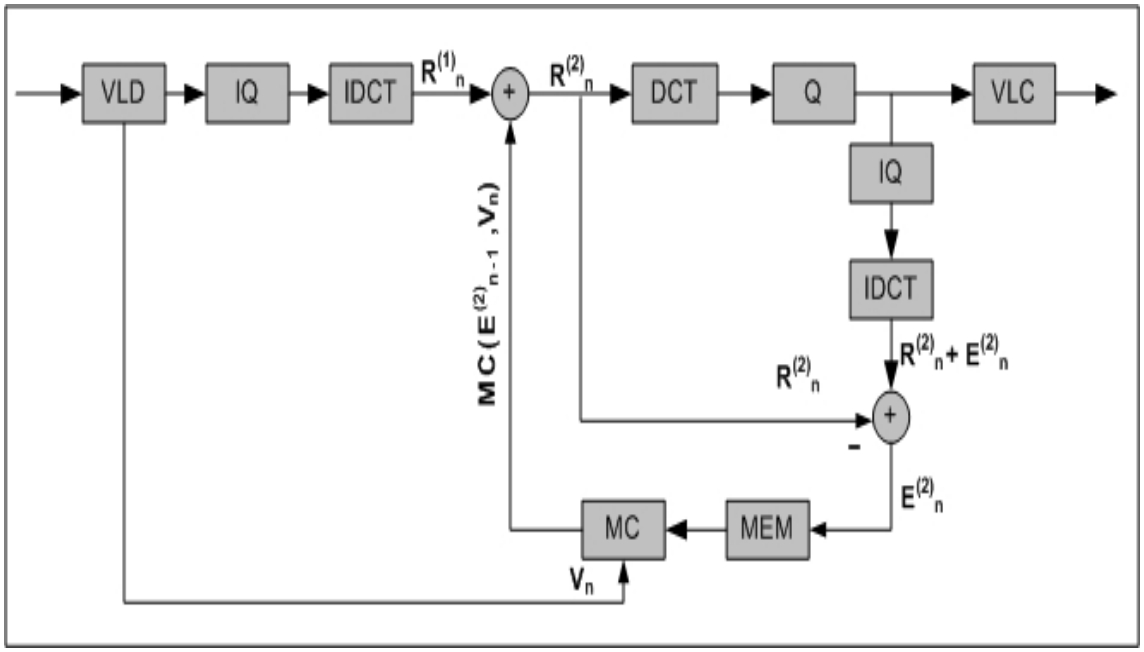


Figure 3.5: Simplified cascaded transcoder.

$$E_n^{(2)} = R_n^{(2)} - (R_n^{(2)} - E_n^{(2)}). \tag{3.7}$$
$$E_n^{(2)} = R_n^{(1)} + MC(E_{n-1}^{(2)}, V_n) - (R_n^{(2)} - E_n^{(2)}). \tag{3.8}$$

The transcoder of the Figure has two IDCT and one DCT. This transcoder can be simplified by using the linearity properties of the DCT. These properties are as follows:

$\chi, \omega \in$ *spatial domain*
$\chi = IDCT(DCT(\chi))$
$\chi = DCT(IDCT(\chi))$
$DCT(\chi + \omega) = DCT(\chi) + DCT(\omega)$

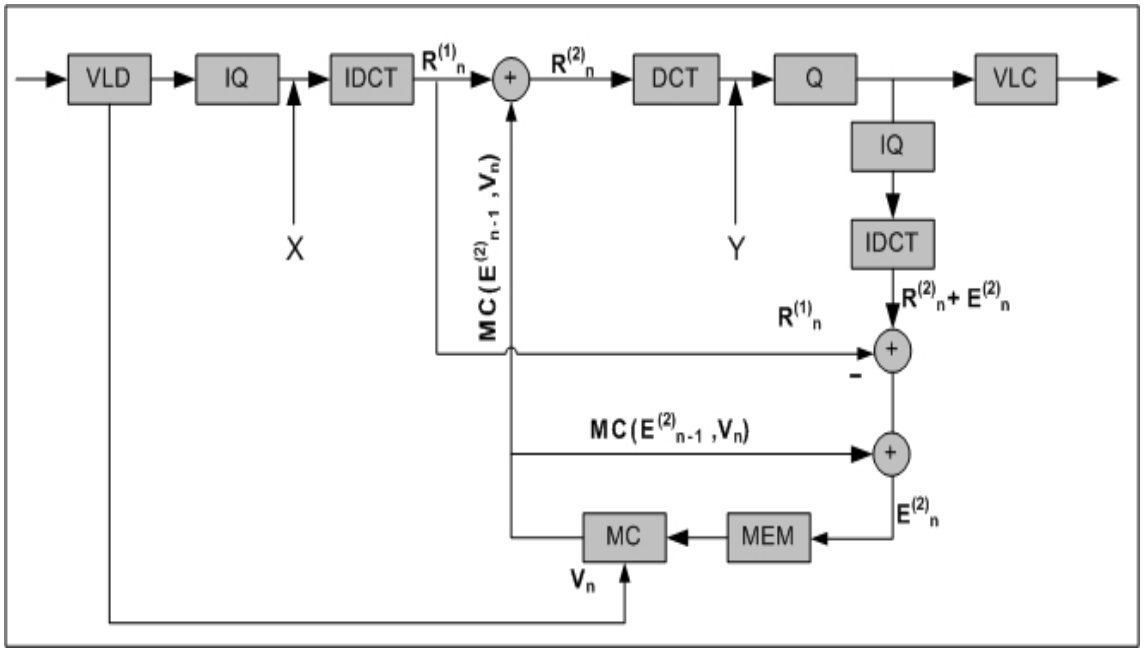


Figure 3.6: Simplified cascaded transcoder.

The transcoder can be simplified as follows:

$$Y = DCT(R_n^{(2)}) \tag{3.9}$$

$$Y = DCT(R_n^{(1)} + MC(E_{n-1}^{(2)}, V_n)) \tag{3.10}$$

$$Y = DCT(R_n^{(1)}) + DCT(MC(E_{n-1}^{(2)}, V_n)) \tag{3.11}$$

$$Y = DCT(IDCT(X)) + DCT(MC(E_{n-1}^{(2)}, V_n)) \tag{3.12}$$

$$Y = X + DCT(MC(E_{n-1}^{(2)}, V_n)) \tag{3.13}$$

According to equation 3.13, the transcoder in Figure 3.5 becomes as shown in Figure 3.4. This transcoder is called a closed-loop transcoder. The closed-loop transcoder has a lower complexity than the cascaded transcoder. Table 3.3.3 shows a comparison between the closed-loop transcoder and the cascaded transcoder.

Name	Closed-loop Transcoder	Cascaded Transcoder
DCT/IDCT	2	3
Reference buffer	1	2
Adds	2	3
motion compensation	1	2

3.3.4 The Closed-Loop Transcoder in a P-frame

The closed-loop transcoder significantly decreases the complexity of transcoding method. The closed-loop transcoder always attempts to compensate drifting errors. Clearly, that would increase the complexity of the re-quantization method.

Generally, the drift-free method in the closed-loop transcoder consists of two processes: an errors accumulation and errors compensation. Since B-frames in MPEG-2 are not used as a reference, the drift errors in the B-frames cannot be propagated to other frames. Therefore, there is no errors accumulation method for B-frames in this transcoder. Moreover, if the accumulated errors are not always presented into the transcoding loop in B-frames, the accumulated errors will not result in severe quality loss. Thus, in this transcoder, the B-frames are not updated with the accumulated errors. In fact, the structure for transcoding B-frames are same as the open-loop one. Since both I-frames and P-frames will be used as

reference frames for other frames, the reference memory is necessary to update with the accumulated errors. However, because I-frame and intra blocks in P-frame are coded without any prediction from other pictures, no drift errors compensation is needed for them. At last, inter blocks in P-frames will be dealt with the following transcoding method.

Consequently, the closed-loop transcoder in P-frame is shown in Figure 3.7. The motion compensated scheme shows MC that can be omitted for B-frames. They do not present any cumulative errors because the B-frames are not used as a reference frame. The simple re-quantization algorithm can be applied to these frames.

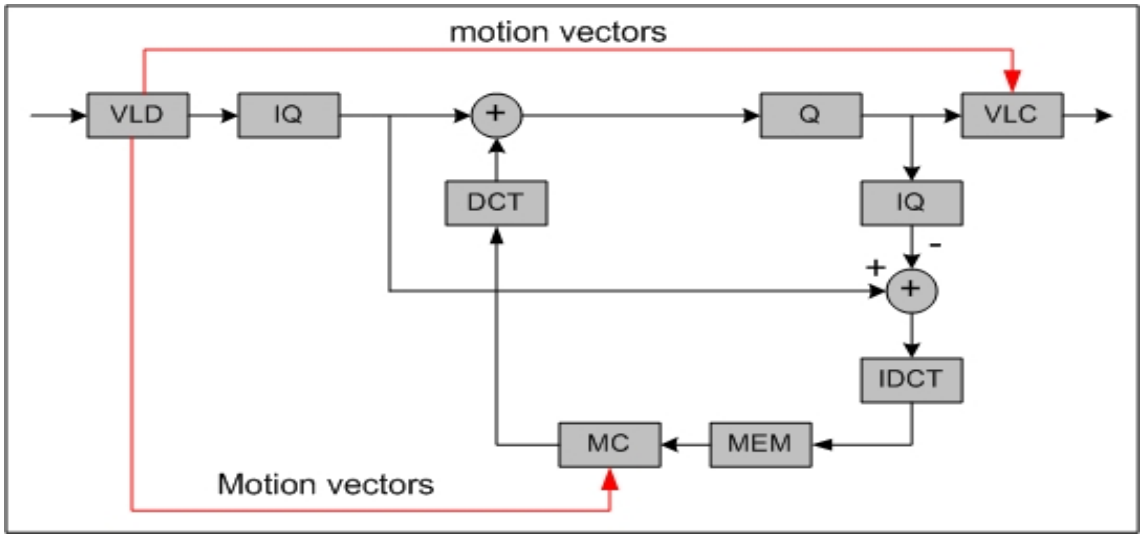


Figure 3.7: Closed-loop transcoder in P-frame block diagram.

3.3.5 Open-loop Transcoder

The open-loop transcoder partially decodes a pre-encoded MPEG-2 video to generate DCT coefficients, and then this transcoder re-quantizes the DCT coefficients as shown in Figure 3.8 [5]. VLD decodes an input MPEG-2 video stream to generate quantized DCT coefficients, MVs, and headers. Then the quantized DCT coefficients are de-quantized and then re-quantized to satisfy a target bit rate. Finally, the re-quantized coefficients are sent VLC and MVs and the headers are copies to VLC. MVs and the headers do not change during transcoding. The open-loop video transcoder only needs the quantization operations that are much less complexity than the cascaded transcoder. Furthermore, the open-loop transcoder is simple because it does not require a frame memory and it does no need DCT, IDCT and MC. In terms of quality, open-loop transcoding is, however, subject to drift errors.

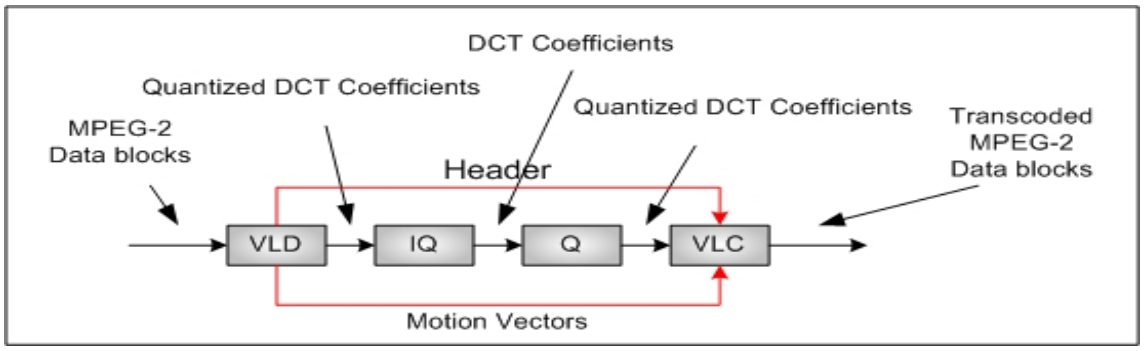


Figure 3.8: Re-quantization bit rate transcoding block diagram.

An open-loop transcoder is faster then a closed-loop transcoder. The open-loop transcoder generates low video quality that is caused by two reasons:

1. The output video stream from the transcoder suffers from drift errors (continuous degradation of picture quality in a GOP).
2. The assumption that motion vectors do not change during the transcoding process is proven to be false [17].

First reason, inter-frames are coded by using reference frames. During transcoding, the qualities of the reference frames are reduced because of drift errors in frames. In order to understand how the drift errors arise consider the design of an open-loop transcoder that is shown in Figure 3.9. The open-loop transcoded frames analyze as follows:

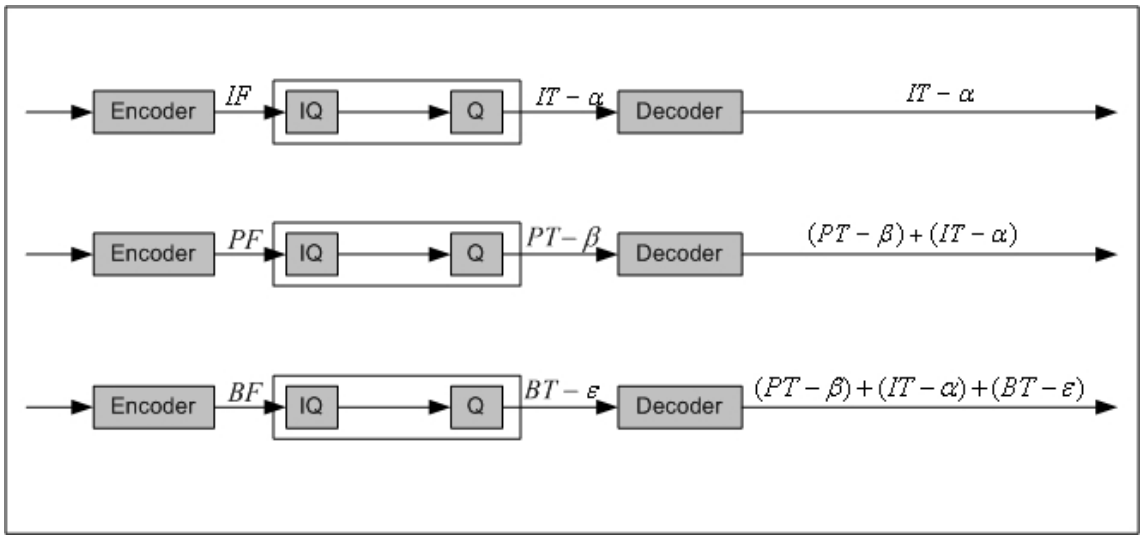


Figure 3.9: Life cycle of an I-frame, a P-frame and a B-frame when open-loop transcoding.

I-Frame

Transcoding begins with an I-frame(IF). IF is re-quantized by transcoding to reach the target bit rate. The transcoded I-frame (IT) is given by

$$IT = Q(IQ(IF)) \tag{3.14}$$

$IQ()$ is an inverse quantization function and $Q()$ is a quantization function. $Q()$ can produce a quantization error(α). α appears because the integer truncation is used during transcoding. Thus, an I-frame has α during transcoding. The transcoded I-frame is given by

$$\alpha = IQ(IF) - IQ(Q(IQ(IF))) \tag{3.15}$$

$$TIF = Q(IQ(IF)) - \alpha = IT - \alpha \tag{3.16}$$

The encoder generated I-frame (IF) as shown in Figure 3.9. Then the transcoder converts IT into a lower I-frame size (TIF). TIF has a lower quality because of α. When the decoder receives TIF, the decoder decodes and stores TIF with reduced quality in a decoder buffer. TIF is used as a reference for decoding other frames.

P-Frame

Figure 3.9 shows, the encoder generates the P-frame (PF), then the transcoder uses PF as input to provide the transcoded P-frame ($PT - \beta$).

$$\beta = IQ(IF) - IQ(Q(IQ(IF))) \tag{3.17}$$

β is a re-quantization error. Then the decoder uses $IT - \alpha$ as a reference picture that will be used to decode $PT - \beta$. Finally, the decoder generates the P-picture $(IT - \alpha) + (PT - \beta)$. As a result, this picture has a re-quantization error $(\alpha + \beta)$. The mismatch in the re-quantization grows because a P-frame is used as reference frame for future frame. As time continues, the mismatch is shown before growing slowly until the end of the GOP.

B-Frame

In MPEG-2, GOP contains B-frames. An encoder generates a B-frame (BF) by using I-frame and P-frame as reference frame. Figure 3.9 shows BF is used as input for transcoding. The transcoder provides the transcoded B-frame $(BT - \epsilon)$. $IT - \alpha$ and $PT - \beta$ were being stored in the decoder buffer when the decoder received $BT - \epsilon$. Then the decoder decodes $BT - \epsilon$. Finally, the decoder applies motion compensation to build the B-picture $(BT - \epsilon)+(PT - \beta)+(IT - \alpha)$. As a result, the re-quantization error $(\epsilon + \beta + \alpha)$ is visible in the B-frame.

3.4 Experimental Results

In this project, the transcoder must converts non-scalable MPEG-2 to SNR scalable MPEG-2 at real-time. This chapter introduces some transcoders architectures, which are implemented using C programming language. These transcoders are able to transcode MPEG-2 to scalable MPEG-2 at an arbitrary bit rate by using a bit rate control algorithm, which is described in chapter four. In addition, the transcoders must be able to satisfy the requirement of the real-time performance. For example, 60 seconds of the pre-encoded video must be transcoded to a lower bit rate within at most 60 seconds.

As mentioned before, the transcoders have to be able to work at real-time. The architectures that are described in this chapter have different computational complexities. For the experiments purpose, we have used some MPEG-2 video sequences that were encoded at an arbitrary bit rate as shown in Table 3.2. The table gives the video sequence time, bit rates for pre-encoded video, frame rate, and sequence size. During the experiments, the transcoders are configured to generate an MPEG-2 bit stream at 3 Mbps. In all the experiments, the SNR scalability results are compared transcoding of the same video sequence using a single-layer transcoder (base layer).

Sequence name	Sequence time	Encoded bit rate	Frame rate	Sequence size
Flower	9 second	9 Mbps	25 f/s	210 Frames
Tank 90	40 second	9 Mbps	25 f/s	1000 Frames
Tank T80UK	42 second	10 Mbps	25 f/s	1050 Frames
Penguin	60 second	5 Mbps	25 f/s	1500 Frames

Table 3.2: The parameters of the video sequences that can be used in the transcoding algorithms.

Table 3.3 shows a comparison between the transcoders. The table is based on evaluations executed on a Dell platform. It is Pentium 4 2.4 GHz CPU, 512 MB memory, and running Windows XP. The transcoder has been built and has been tested using the Microsoft Visual Studio dot Net with Microsoft C++ compiler. Table 3.3 summarizes the performance comparison of three transcoders architectures. The table shows transcoding time that is significantly more than the real-time.

sequence name	Penguin	Tank 90	Tank T80UK	Flower
Closed-loop transcoder	235 seconds	140 seconds	149 seconds	31 seconds
Closed-loop in P frame transcoder	92 seconds	54 seconds	59 seconds	13 seconds
Open-loop transcoder	22 seconds	19 seconds	22 seconds	4 seconds

Table 3.3: The performance comparison of three transcoders

3.5 Hybrid Transcoder

In this project, the PC does not support any high complexity algorithms to work in real-time such as a closed-loop transcoder, which provides high quality pictures. Therefore, we have to return to open-loop transcoding that performances at real-time. However, this transcoding does not generate high quality pictures and does not use all available recourses. Therefore, a fast closed-loop video transcoder with limited drifting error was introduced in [16]. The fast closed-loop video transcoder works similar to the closed-loop transcoder.

The fast closed-loop transcoder still accumulates the re-quantization errors of each I-frame or each P-frame. However, the accumulated errors are not always introduced into the transcoding loop in every block. The fast closed-loop video transcoder has one switch that is proposed to intentionally control whether or not to update a block with the accumulated errors. If the accumulated error in a block is more than the threshold, it is introduced into the transcoding loop as in the traditional closed-loop transcoder; otherwise input non-zero levels will simply go through the re-quantization process (open-loop).

We was proposed a transcoding algorithm that uses the given resources to provide better pictures quality. Within the available recourses, the hyper transcoder must provide high quality pictures and must performance at real-time. The hybrid transcoder has the same efficiency as the fast closed-loop but the hybrid transcoder has some changes in the original design of the fast closed-loop transcoder. This mean that, hybrid transcoding allows to find the best compromise between a low complexity algorithm with low quality and high complex algorithm with high quality while taking advantage of all recourses available within consider PC.

Most of the end users are always concentrated on a single focus area. For example, when there is more than one object on the video, an end user may focus his attention on one object position. Thus, we divide the screen area in two areas: firstly the important area and secondly the unimportant area. In any picture, the object position is the important area for the users. This means that the pixels of the important area must be received more bits than the remaining area does. In other words, the important area has a better picture quality than the unimportant area.

The important area can be transcoded by using closed-loop transcoding, which is of a high computational complexity. The unimportant area can be transcoded by using open-loop transcoding, which has a lower computation complexity. In this project, the hybrid transcoder switches between two transcoding algorithms (open-loop and closed-loop) in the same picture. Therefore, a macroblock quality depends on its position at the picture.

End users normally concentrate at the center area in a screen that is the important area for the end users. However, most of the end users do not concentrate on the border area in a screen such as the first slice and the last slice in the frame. Therefore, when the border area in a screen does not have high quality picture, end users cannot notice that. Due to this fact, the border area reduces the consuming time. The center area in a screen must have a better quality than the border area. Consequently, time consumptions are only at the center of the screen.

The hybrid transcoder consists of one switch in the motion compensated loop as described in Figure 3.10. The switch applies independently to any macroblock, open at the border area, and close at the center area. In other words, strategies can be adapted with available resources by selecting turn on or turn off at the macroblock level. Hence, this part provides a solution to find the correct switch selection strategy to optimize the resulting picture quality.

3.5.1 Experimental Results

In this section, we introduce a comparative study of the quality and complexity of the transcoding architecture. For evaluation purpose, the transcoders that are introduced in this chapter are as following: the closed-loop transcoder (CLT), the open-loop transcoder (PLT), the closed-loop in P-frame transcoder (CLPT) and the hybrid transcoder (HLT). They are used to generate the output bit stream. In this project, the transcoders have two important requirements: they have to generate excellent picture quality and they have to be able to operate in real-time. We describe our experiments based on a number of

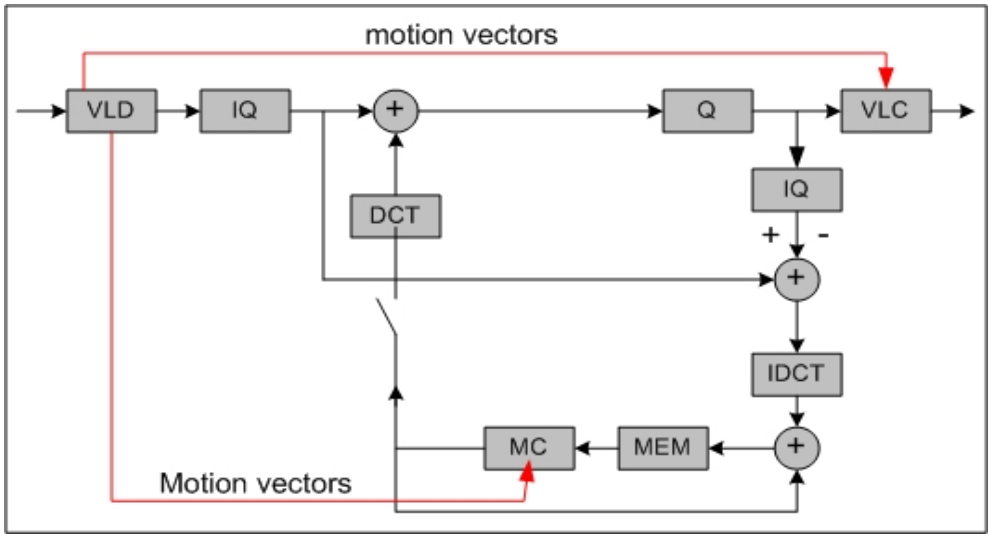


Figure 3.10: hybrid transcoder architecture.

test video sequences that are introduced in Table 3.2. All video sequences have a spatial resolution that is 720 x 576 pixels. A color sampling ratio of these video sequences is 4:2:0 and. The test sequences are transcoded at 3 Mb/s bit rates.

This chapter is introduced some transcoders architectures that suffer from high computational complexity. By using a hybrid transcoding algorithm, the transcoder is able to achieve significant performance improvements. Pre-encoded MPEG-2 is transcoded into the base layer. This base layer can be simplified by skipping the B-pictures of the pre-encoded video stream resulting in a lower complexity and lower delay transcoding with a lower resolution. The total transcoding time was reduced by using hybrid algorithm over 20% making the real-time transcoding possibility. In the hybrid algorithm, an unimportant area is 600 macroblocks. Table 3.4 summarizes the performance comparison of four transcoders.

sequence name	Penguin	Tank 90	Tank T80UK	Flower
Closed-loop transcoder	235 seconds	140 seconds	149 seconds	31 seconds
Closed-loop in P frame transcoder	92 seconds	54 seconds	59 seconds	13 seconds
Open-loop transcoder	22 seconds	19 seconds	22 seconds	4 seconds
hybrid transcoder	59 seconds	39 seconds	41 seconds	8 seconds

Table 3.4: Transcoding time of different transcoders.

The transcoders are compared with each other to select the best transcoder. It will be used to generate a transcoded video stream in this project. Thus, the perceived quality pictures are compared when using these transcoders. Table 3.5 gives the average Peak Signal-to-Noise (PSNRs) value of the PSNRs of the frames within the same sequence. Theses transcoders generate quite different visual quality. The closed-loop transcoder generally shows better picture quality than other transcoders in the experiments. The average PSNRs of closed-loop transcoder is 0.2 dB higher than the hybrid transcoder and the average PSNRs of the hybrid transcoder is 0.07 dB higher than the open-loop transcoder as shown in table. Figure 4.9 shows the frame-based peak signal-to-noise ratio (PSNR) plots comparing the picture quality of the open-loop transcoder, the closed-loop transcoder and the hybrid transcoder. Since the open-loop transcoder causes severe drift error, it is shown in the plot. The quality of the frames from the transcoded video continuously degrades until the next I-frame; the I-frame is indicated by the vertical grid lines. From the plots, we observe that the quality produced by the closed-loop transcoder is comparable to the hybrid transcoder. The key point to note is that it achieves this quality with much lower complexity.

Sequence name	average PSNRs	Algorithm
Flower	35.268	CLT
	35.138	CLPT
	34.993	HLT
	34.918	OLT
Tank 90	33.857	CLT
	33.752	CLPT
	33.663	HLT
	33.607	OLT
Tank T80UK	33.901	CLT
	33.799	CLPT
	33.707	HLT
	33.665	OLT
Penguin	41.298	CLT
	41.277	CLPT
	41.189	HLT
	41.162	OLT

Table 3.5: The average PSNRs of the video sequences.

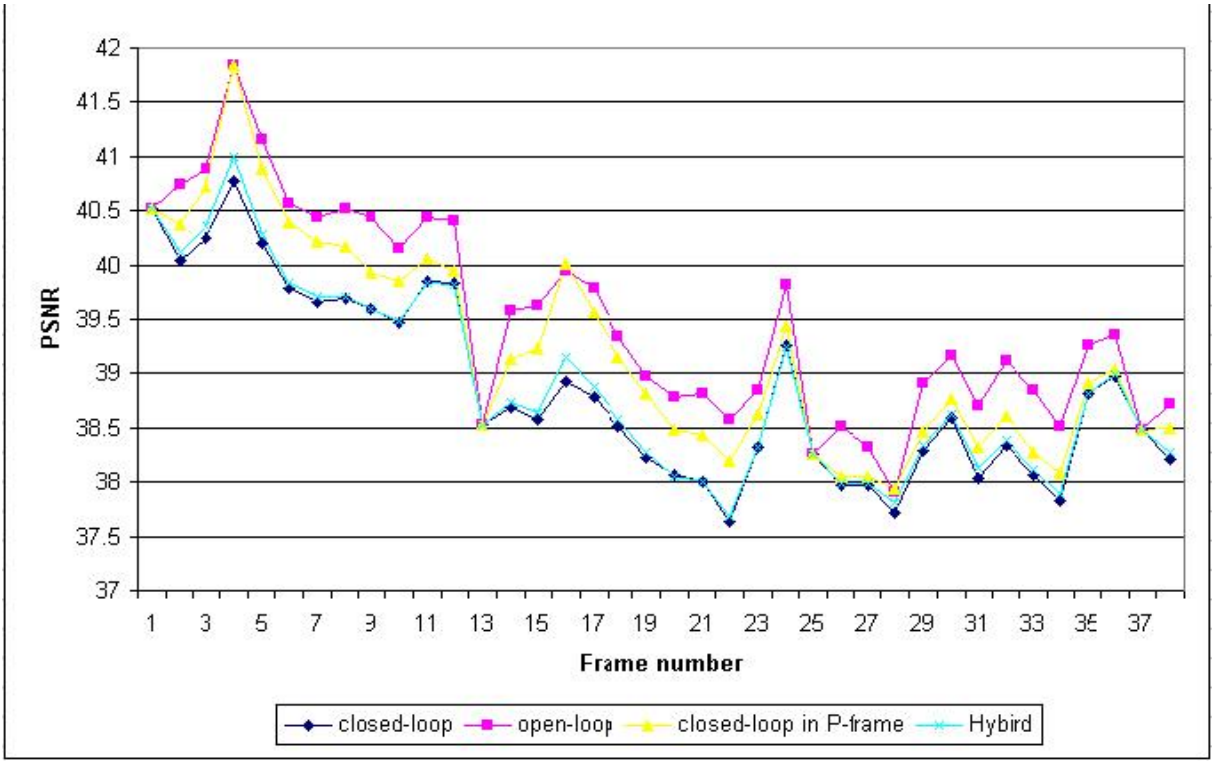


Figure 3.11: PSNR comparison for 200 frames; CLT, CLPT, HLT and OLT for Penguin.

Chapter 4

Bit Rate Control

A bit rate control algorithm is important for transcoding because it determines the size of a transcoded video stream. In this project, the transcoder, which has both a base layer and enhancement layers, generates a SNR scalable video stream. In general, bit rate control of the base layer is the same as bit rate control of a non-scalable encoder. The base layer uses the bit rate algorithm to produce re-quantized DCT coefficients. Re-quantization errors of the base layer are input of the first enhancement layer. Re-quantized errors of the first enhancement layer are input of the second enhancement layer. The bit rate control algorithm maintains the bit rate of each layer that determines the target bit rate of a transcoded video stream.

A transcoder should contain bit rate control to be able to meet the requirements of this project. In this chapter, the widely used standard Test Model 5 (TM-5) that is bit rate control is described in Section 4.2 [18]. TM-5 has been developed in connection with standard MPEG-2. The standard TM-5 algorithm requires low computational complexity. In this project, the standard TM-5 algorithm can be used as a basis for developing a bit rate control algorithm for the base layer. TM-5 is designed to work in the spatial domain. Encoders work in the spatial domain, but transcoders do not work in the spatial domain. The transcoders work in the frequency domain. Therefore, an encoder can use TM-5 but a transcoder can not use TM-5. Furthermore, the standard TM-5 algorithm does not dynamically adjustable at run time when the available bandwidth fluctuates. Section 4.3 introduces an adaptive TM-5 algorithm that the Philips research has developed. The adaptive TM-5 algorithm (ADTM) that works in the frequency domain is dynamically adjustable at run time. Further, this algorithm responds within the time of a GOP period to the adjusted bit rate. In this project, the transcoder should respond within the time of a frame period to the adjusted bit rate. Therefore, we introduce two bit rate algorithms: a real-time TM-5 algorithm (RTTM) in Section 4.4 and a scaled TM-5 algorithm (SCTM) in Section 4.5. These algorithms change the output bit rate within the time of a frame period at run-time when the available bandwidth fluctuates. These algorithms require low computational complexity. Finally, we introduce experimental results in Section 4.6.

4.1 Bit Rate Control

A bit rate control algorithm is a mechanism that controls the bit rate of the transcoded video. The bit rate control algorithm determines the suitable scale such that the bit rate of the transcoded video stream meets the target bit rate. The bit rate control algorithm achieves a pre-defined average bit rate by controlling bit allocation. The bit rate control algorithm allocates the bit budget to all macroblocks in a frame. Therefore, all macroblocks in a frame can generate the same number of bits after coding.

A bit rate control algorithm has to optimize the perceived quality for each frame within a given bit budget which depends on the available bandwidth. A transcoder gets the available bandwidth from the sender as shown in Figure 4.1. The transcoder gives the available bandwidth for the bit rate control algorithm that uses the available bandwidth to determine a quantizer. This means that there is a strong relation between the quantizer and the bit rate control algorithm. Figure 4.2 shows rate control determines a quantizer for each macroblock. In generally, a bit rate control algorithms are classified as follows:

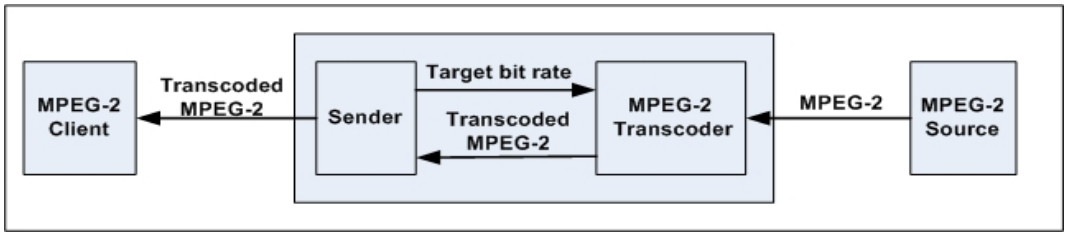


Figure 4.1: Overview of the transcode system.

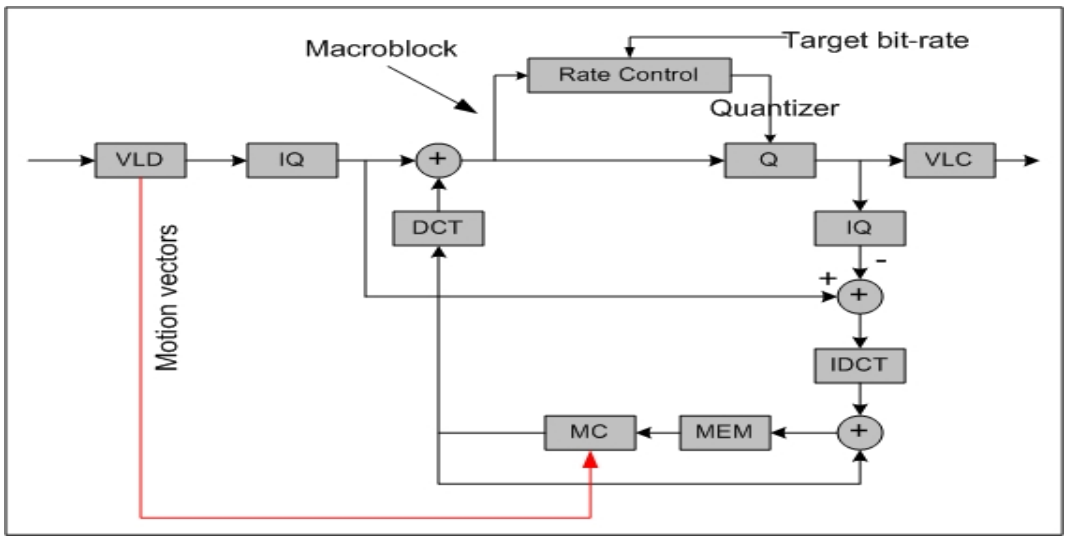


Figure 4.2: The bit rate control of the closed-loop transcoder.

1. Constant Bit Rate (CBR) coding, in which the number of bits transmitted per second is constant. In CBR, picture quality may differ depending on its content,since the encoder output rate generally varies depending on the picture content.

2. Variable Bit Rate (VBR) coding, in which the number of bits transmitted per second may vary under some restrictions. VBR tries to provide constant picture quality under a given average target bit rate.

This chapter introduces four bit rate control algorithms. Each algorithm has some properties which are summarized in Table 4.1. Bit rate control works in a hierarchical structure that consists of a GOP, a frame and a macroblock. Each GOP has the bit budget that a bit rate control algorithm allocates this bit budget between frames in the same GOP. In the same frame, the target bit budget is allocated between macroblocks. In CBR, each GOP in a video sequence has the same bit budget if all GOPs have the numbers of frames. CBR usually has five steps:

1. A GOP bit allocation assigns a target number of bits for each GOP, which is based on a frame rate($frame\_rate$), a GOP size(N) and a target bit rate($output\_bit\_rate$).
   Input: $frame\_rate$, $output\_bit\_rate$, and N.
   Output: a GOP bit budget(R).
   For example, a video sequence has a number GOPs, each GOP has 12 frames. This video sequence will be transcoded at bit rate 3 Mb/s with 25 frames/s. The GOP bit budget computes as follows:
   $R = N * (\text{output\_bit\_rate} / \text{frame\_rate})$

2. A frame bit allocation step assigns a target number of bits for each frame based on the frame type and the remaining number of frames in the same GOP.

	TM-5	ADTM	SCTM	RTTM
when to observed changed bandwidth	once per sequence	once per GOP	once per frame	once per frame
unit of budget	GOP	GOP	frame	frame
bit rate control type per GOP	CBR	CBR	*	*
bit rate control type per frame	VBR	VBR	VBR	VBR
number of steps to transcoded one frame	5 steps	5 steps	3 steps	4 steps
virtual buffer required	yes	yes	no	yes
domain	special domain	frequency domain	frequency domain	frequency domain

Table 4.1: comparison of four bit rate control algorithms

Input: the current GOP bit budget, the number of frames.
Output: a frame bit budget.
The frame bit budget for I-frame is
$Ti = R/N$.
The frame bit budget for P-frame is
$Tp = (R - Ti)/(N - 1)$.
The frame bit budget for jth B-frame is
$Tb = (R - (Ti + Tp + (j - 1) * Tb))/(N - (j + 2))$.

3. A rate control step attempts to adjust bit allocation if there is difference between the target bits for each macroblock and the actual transcoded bits for each macroblock. TM-5 has a virtual buffer is used to simulate the fullness of the decoder buffer. Size of this buffer is one frame.
   Input: a frame bit budget(T), the number of macroblocks per frame ($total\_MB$), the actual transcoded bits($bitcount - old\_bitcount$) in the current frame and virtual buffer fullness(d).
   Output: a quantization parameter (Qj).
   the rate control is computed by the following equation:
   $Qj = d + (bitcount - old\_bitcount) - (j) * (T/total\_MB)$.
   for example, the rate control of macroblock 0 is $Q0 = d$.
   the rate control of macroblock 1 is $Q1 = d + (bitcount - old\_bitcount) - (T/total\_MB)$.
   the rate control of macroblock 3 is $Q3 = d + (bitcount - old\_bitcount) - 3 * (T/total\_MB)$.

4. Adaptive quantization step controls the target bit rate. Therefore, the quantization parameter is adjusted at each macroblock according to activity of block against the normalized activity of the frame. The normalized activity measured in the pre-analysis stage of video transcoding, and the target buffer size status, you have a fairly good idea of how many bits can be spared for the target macroblock, for instance.In this step, variance for each block($variance(block)$) is computed.
   Input: macroblock data, number blocks per macroblock($block\_count$) and a quantization parameter.
   Output: a quantization step($mquant$).
   $Mini\_var = Min(variance(block(1)), variance(block(2)), .variance(block(block\_count)))$
   $mquant = Mini\_var * Qj$;

5. A update step gives the new values for the variables that will be used to transcode the next frame.
   Input: the virtual buffer fullness, a GOP budget, a frame size(S), and the quantization step.
   Output: the virtual buffer fullness and $old\_bitcount$.
   $S = bitcount - old\_bitcount$.
   $d = d + S - T$.
   $old\_bitcount = bitcount$.

The GOP bit allocation works at the GOP level. The frame bit allocation and the update work in the frame level. Rate control and adaptive quantization work in the macroblock level.

4.2 Standard Test Model 5 Algorithm

A TM-5 algorithm consists of five steps in order to adapt a video content. Before transcoding any frame, the standard algorithm estimates the number of bits needed to transcode the frame. This estimation depends on several factors such as the frame type. The standard TM-5 algorithm can be used as a basis for developing bit rate control for the base layer. Details of TM-5 can be found in [18]. This section introduces the standard TM-5 algorithm that forms the basis for understanding our algorithms.

Standard TM-5 is CBR per GOP but standard TM-5 is not CBR per frame. This means that standard TM-5 is VBR per frame. The frame bit budget usually depends on the frame content. TM-5 gives more bit budget for transcoding the more complex regions of a frame and less bit budget for transcoding the less complex regions of a frame. In other words, in the same frame, each macroblock may have its own bit budget to reach uniform quality. The numbers of generated bits depend on the complexity of a transcoded macroblock.

When VBR needs to be transmitted over a fixed-rate channel, a channel buffer is usually used to smooth output bit stream. TM-5 uses a buffer to prevent the channel buffer overflow and underflow. Therefore, TM-5 uses a feedback mechanism that controls the transcoder parameters. In Figure 4.3, a flowchart shows the rate controlling process according to TM-5. The pseudo code of the standard TM-5 algorithm is described in detail next.

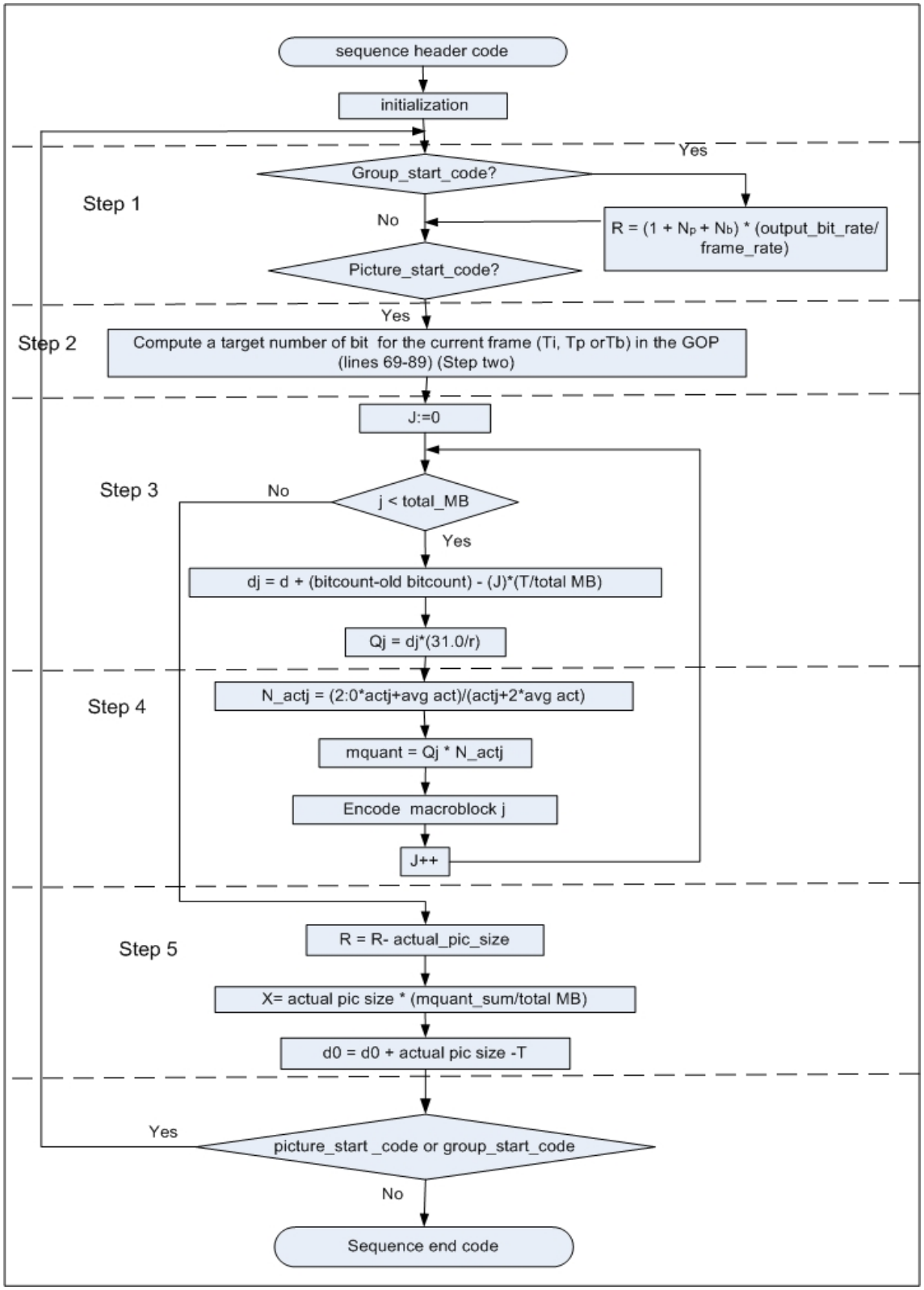


Figure 4.3: The process of the rate control according to TM-5 of MPEG-2

```
next_start_code() /* removes any zero bit and zero byte stuffing and locates the next start code.*/
nextbits()  /*permits comparison of a bit string with the next bits to be decoded in the bitstream. */
/* picture coding type */
 01    #define  I_TYPE                1
 02    #define  P_TYPE                2
 03    #define  B_TYPE                3
 04    #define  sequence_header_code  ...
 05    #define  sequence_end_code     ...
 06    #define  group_start_code      ...
 07    #define  picture_start_code    ...
 08    #define  output_bit_rate       ...     /* bit rate of encoded video */
 09    #define  input_bit_rate        ...      /* input bit rate of video */
 10    #define  frame_rate            ...     /* frame rate of video to be encoded*/
 11    #define  total_MB              ...     /* number of macroblocks per frame */
 12    #define  frame_size            ...     /* average number of bits per frame */
 13    #define  block_count           ...     /* number blocks per macroblock */
 14    int      bitcount= 0,                  /* counter for bits that will used for encoding video file*/
 15             R=0,                          /* remaining number of bits in GOP */
 16             X_i, X_p, X_b,                /* complexity estimate of current(I,P,B) frame */
 17             N_p,                          /* remaining number of P frames in current GOP */
 18             N_b,                          /* remaining number of P frames in current GOP */
 19             d0i, d0p, d0b;                /* initialize virtual buffer fullness of current(I,P,B) frame*/
 20    double   avg_act,                      /* is the average value of actj the previous frame to be encoded */
 21             actsum;                       /* is the sum value of actj the current frame to be encoded*/
 22    struct   GOP {
 23             Frame *F;                     /* sequence of frames */
 24             intN_p, N_b;
 25             ...
 26             };
 27    unsigned struct GOP          *Buf;    /* buffer with video*/
 28   void ComputeBitRate()                   /* This the main function of the standard TM-5*/
 29   {
 30     int       mquant_sum,                /* sum quantization per frame */
 31               MB_size,                   /* number of bits are used to encode current macroblock */
 32               old_bitcount,              /* number of bits had been used before encoding the current frame */
 33               T,                         /* target number of bits for frame */
 34               Tmin,                      /* target number of bits for frame */
 35               r,                         /* reaction parameter (constant)*/
 36               mquant,                    /* is used to re-quantize macroblock */
 37               GOP_count=0,               /* counter for GOPs */
 38               j                          /* counter for macroblocks within frame */
 39               i                          /* counter for blocks within macroblock */
 40               picture_coding_type;       /* identifies whether a frame is an I-frame, P-frame or B-frame */
 41     double    mini_var,                  /* hold minimum variances of the blocks in the current macroblock */
 42               actj,                      /* take minimum spatial activity measure of macroblock j_th */
 43               temp;
 44               ...
 45     next_start_code();     /*Search the bit stream and find the next header by looking for the start code.*/
 46     if(nextbits()==sequence_header_code)
 47     {
 48        ...
 49         next_start_code();
 50     }
```

```
51  do {    /* while ( nextbits() != sequence_end_code )*/
52          /* Initialize the TM-5 algorithm */
53          r = (2.0*output_bit_rate/frame_rate);
54          Initialize X_i, X_b, X_p; /* using some given algorithm */
55          Initialize d0i,d0p,d0b; /* using some given algorithm */
56      do { /*while ((nextbits() == picture_start_code)||(nextbits() ==group_start_code))*/
57          if (nextbits() == group_start_code)
58          {
59              ...
60              next_start_code();
61              N_p = buf[GOP_count].N_p;
62              N_b = buf[GOP_count].N_b;
63              R = (1 + N_p + N_b) * (output_bit_rate/frame_rate);
64              GOP_counter++;
65          }/* end of (nextbits() == group_start_code)*/
66          if(nextbits() == picture_start_code)
67          {
68              ...
69              switch (picture_coding_type)
70              {
71                  case I_TYPE:
72                  T = R/(1.0 + N_p * X_p/(X_i * 1.0) + N_b * X_b/(X_i * 1.4)) ;
73                  d = d0i;
74                  break;
75                  case P_TYPE:
76                  T = R/(N_p + N_b * 1.0 * X_b/(1.4 * X_p));
77                  d = d0p;
78                  break;
79                  case B_TYPE:
80                  T = R/(N_b + Np * 1.4 * X_p/(1.0 * X_b));
81                  d = d0b;
82                  break;
83              }
84              Tmin = output_bit_rate/(8.0*frame_rate);
85              if (T<Tmin)
86              T = Tmin;
87              mquant_sum=0;
88              actsum=0;
89              next_start_code();
90              ...
91              old_bitcount = bitcount; /* the amounts of bits that are used before encoding the current picture* /
92          }/* end of if(nextbits() == picture_start_code)*/
93          for(j=0;j<total_MB;j++)
94          {
95              dj = d + (bitcount-old_bitcount) - (j)*(T/total_MB);
96              Qj = dj*(31.0/r);
97              ...
98              mini_var = block_data(j,0);
99              for (i=1;i<block_count;i++)
100             {
101                 temp =block_data(j,i);
102                 if(mini_var>temp)
103                     mini_var=temp;
104             }
105             actj = 1 + mini_var;
```

```
106             N_actj = (2.0*actj+avg_act)/(actj+2*avg_act);   /* compute normalized activity */
107             mquant = Qj * N_actj;
108             ... /* use mquant valuse in order to re-quantize the current macroblock */
109             actsum += actj;
110             bitcount+= MB_size;
111             mquant_sum = mquant_sum + mquant;
112          } /* End of for loop */
113          ...
114          update_pict(old_bitcount,bitcount,R,total_MB,mquant_sum,picture_coding_type,T,actsum);
115          next_start_code();
116       } while ((nextbits()==picture_start_code)||(nextbits()==group_start_code))
117      }while (nextbits()!=sequence_end_code)
118      ...
119    }/*End of the main function "ComputeBitRate()"*/
120 /**************Functions************************/
121 double block_data(int j,int num)
122 {
123  int i;
124  short block[j][num][64], mean=0, sum=0, var=0;
125  ...        /* copies coefficients of the current block into block[j][num][64] */
126  for(i=0;i<64;i++)
127       sum = sum + block[j][num][i];
128  mean= sum/64;          /* computes mean per block*/
129  for(i=0;i<64;i++)
130      var = var + pow(block[j][num][i]- mean,2); /* computes variance per block */
131  return (var/64);
132 }
133 void update_pict(old_bitcount,bitcount,R,total_MB,mquant_sum,picture_coding_type,T,actsum)
134 {
133    double X;
134    int actual_pic_size;
135    actual_pic_size = bitcount - old_bitcount; /* total # of bits in picture */
136    R-= actual_pic_size; /* remaining # of bits in GOP */
137    avg_act = actsum/total_MB;
138    switch (picture_coding_type)
139    {
140       case I_TYPE:
141       X_i = actual_pic_size * (mquant_sum/total_MB);
142       d0i = d0i + actual_pic_size - T;
143       break;
144       case P_TYPE:
145       X_p = actual_pic_size * (mquant_sum/total_MB);
146       d0p= d0p + actual_pic_size - T;
147       N_p--;
148       break;
149       case B_TYPE:
150       X_b = actual_pic_size * (mquant_sum/total_MB);
151       d0b = d0b + actual_pic_size - T ;
152       N_b--;
153       break;
154    }
155 }
```

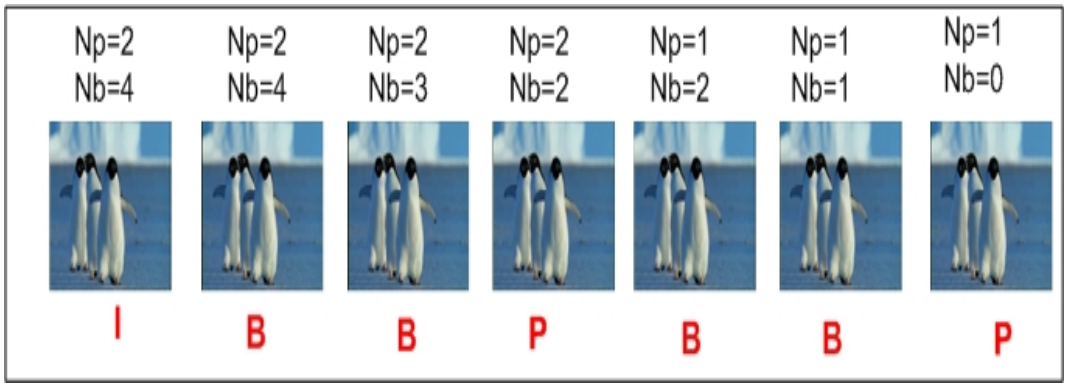


Figure 4.4: Np and Nb of the GOP before encoding

Before transoding a video sequence, TM-5 gives the initial complexity measure values (Xi, Xp, and Xb) as shown in line 54. The complexity measure for a frame is defined as a product of an average adaptive quantization step for all the macroblocks used during transcoding the frame and the target number of bits was generated to transcode the frame($S * (mquant/total\_MB)$). TM-5 uses a complexity measure to compute the target bit budget for each frame. The initial values of the virtual buffer fullness are set according to the frame type as shown in line 55.

Step 1: Bit Allocation for a GOP (lines 57-65)

The bit allocation algorithm is used to assign the number of bits for each frame (I-frame, P-frame, and B-frame) in the GOP. Standard TM-5 allocates the target bit budget,R, for the current GOP in line 63. Np and Nb are respectively the number of P-frames and B-frames of the current GOP that have yet to transcoded. Figure 4.4 shows Np and Nb of the GOP before encoding.

Step 2: Bit Allocation for each Frame in the GOP(lines 66-92)

The target bit budget for each frame is computed after TM-5 computes the target bit budget for the GOP. The bit allocation algorithm for each frame takes into account the prior knowledge of GOP parameters (e.g. GOP size) and coding types (e.g. frames types) to allocate target bits for each frame in the GOP.

Each frame type of I-frame, P-frame, and B-frame are assigned a relative X according to a complexity measure. X_i, X_p, and X_b values determine the target bit budget for each frame. The bit allocation algorithm computes the target bit budget for each frame based on the frame types (I, P, and B) and the remaining number of frames (N_p and N_b) and the remaining number of bits (R) in the GOP. The target number of bits for the current frame is computed as shown in lines 69-86. Any frame has the bit budget less than the minimum bit budget($Tmin$) as shown in line 84, this frame can cause the sender buffer underflow. Therefore, TM-5 prevents the buffer underflow by giving $Tmin$ as a bit budget for any frame that has a bit budget less than $Tmin$ bits as shown in lines 85 and 86.

Step 3: Rate Control (lines 95 and 96)

The rate control produces the quantization step for each macroblock. The target bit budget for each macroblock is determined by the quantization step. Furthermore, the rate control attempts to adjust bit allocation if there is a significant difference between the target bit budget and the actual encoded bits for the macroblock. The average macroblock bit budget is determined by equation 4.1.

$$Avge\_macroblock := (T/total\_MB) \tag{4.1}$$

The quantization step depends on the virtual buffer capacity and is updated for each macroblock as shown at line 95. Therefore, the rate control step changes the quantization step according to the sender virtual buffer fullness. The increase of the sender virtual buffer makes the quantization step increase, which results in decrease of bits used for a macroblock, and vice versa.

Before encoding the $j^{th}(j \geq 0)$ macroblock, the sender virtual buffer fullness is computed as shown:

$$d_j = d + (bitcount - old\_bitcount) - (j) * (T/total\_MB) \tag{4.2}$$

d is the initial fullness of the virtual buffer for each frame type. The final fullness of the virtual buffer dj ($j = total\_MB - 1$, the last macroblock in the current frame) is used as d for encoding the next frame of the same type. (bitcount-old_bitcount) is the number of bits generated by encoding all macroblocks in the frame up to and including j^{th} macroblock. The rate control step uses dj to regulate the output bit rate. In line 96, dj determines the quantization step(Qj) for each macroblock j.

Figure **??** shows an example of the rate control of P-frame. d is the initial value of the virtual buffer fullness, T_p is the target bits for the P-frame. The black line denotes the value of $bitcount - old\_bitcount$ which is the number of bits used to encode all previous macroblocks in the frame up to and including j^{th} macroblock. $total\_MB$ is the number of macroblocks in the frame. The dashed line in the Figure acts as a lower bound for the virtual buffer. Equation 4.3 is the dashed line equation.

$$Tp = (j) * (Tp/total\_MB) \tag{4.3}$$

Before encoding the j^{th} macroblock the virtual buffer fullness is adjusted during the encoding according to the following equation 4.2 for the P-frame:

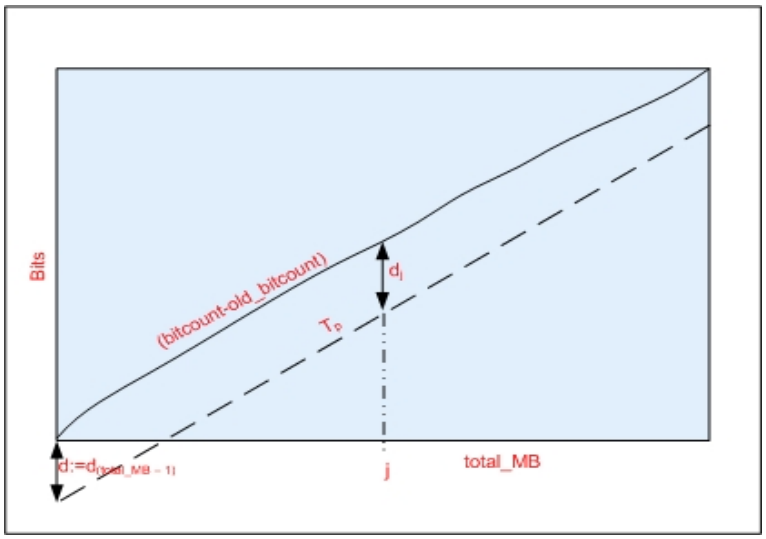


Figure 4.5: An example of rate control for an P-picture.

Step 4: Adaptive Quantization (lines 98-107)

After the quantization step is computed, an adapted quantization step will be computed. The increase of the adapted quantization step makes the output bit rate decrease, and vice versa. In line 98, the $block\_data\,(j, 0)$ function computes the variances in the first block of the current macroblock. In lines 99 - 104, the for-loop computes the variance for each block of the current macroblock. $mini\_var$ also represents the value of the minimum variance in the current macroblock. In line 109, $actsum$ represents the sum of minimum variance per block for the encoded macroblocks. A spatial activity factor ($actj$) is computed as shown in line 105.

The human eyes are not so sensitive to the quantization noise but the smooth areas are more sensitive to the quantization noise. Thus, encoder uses more bits in the smooth areas and fewer bits for the active areas. Based on areas, Qj obtaining from the rate control (step 3) is adjusted by N_actj to increase $mquant$ for the active areas and to reduce $mquant$ for the smooth areas.

Step 5: Update (line 114)

The updating step determines the target budget for the next frame to be transcoded. This step is carried out before transcoding the next frame. The updating step has the following activates:

- Update global complexity measure(X_i,X_p, or X_b).
- Update buffer fullness.

4.3 Adaptive TM-5 Algorithm

In this project, the sender of the video stream uses a wireless network to provide the video content. The bandwidth of the wireless network fluctuates due to outdoor conditions (microwave, mobile). Unfortunately, the standard TM-5 algorithm can not change its bit rate during run-time. C.J.J. Brouwers proposes the adaptive algorithm. It gives the solution for the standard TM-5 algorithm problems, they will introduce in this section.

The adaptive TM-5 algorithm differs from the standard TM-5 algorithm in steps 1 and 4. At step two, the adaptive TM-5 algorithm determines the target bit budget for the current frame based on the current GOP structure and the current complexities (Xi , Xp and Xb). Step three computes the quantization step. Step five estimates the number of bits available for transcoding the next frame. In these steps, the adaptive TM-5 algorithm works in the same way as the standard TM-5 algorithm. Steps one and four are in the adaptive algorithm different than the standard algorithm. These steps do not meet the requirements for this project. Figure 4.6 shows the flowchart of the adaptive algorithm and the pseudo code is described in details next.

```
1   int input_Q_j              /*the quantization parameter of the MB retrieved from the input stream.*/
2   int input_Q_avg            /* the average quantization parameter of the current frame */
3   ...
4   int output_bit_rate        /* In standard TM-5, output_bit_rate is constant */
5   void ComputeBitRate()            /* This the main function of the standard TM-5*/
6   {
7       ...
8       do {/* while ( nextbits() != sequence_end_code )*/
9           .../* Initialize the TM-5 algorithm */
10          do { /*while ((nextbits() == picture_start_code)||(nextbits() ==group_start_code))*/
11              if (nextbits() == group_start_code)
12              {
12                ...
13                if (buf[GOP_count].closed_GOP)  /*read the current GOP header*/
14                {
15                    N = 15;
16                    N_p= 4;
17                    N_b= 10;
18                }
19                else
20                {
21                    N=13;
22                    N_p=4;
23                    N_b=8;
24                }
25                R = N * ( output_bit_rate / frame_rate );
26                GOP_counter++;
27              }/* end of (nextbits() == group_start_code)*/
28              ... /* The target number of bits for the current picture (T) is computed */
29              for(j=0;j<total_MB;j++)
30              {
```

```
31                  dj = d + (bitcount − old_bitcount) − j ∗ (T/total_MB);
32                  Qj = dj ∗ (31.0/r);
33                  ...
34                  mquant = Qj ∗ (input_Qj / input_Q_avg);
35                  .../* use mquant valuse in order to re-quantize the current macroblock */
36                }/* End of for loop */
37                ...
38                update_pict(old_bitcount,bitcount,R,total_MB,mquant_sum,picture_coding_type,T,actsum);
39                  next_start_code();
40                } while ((nextbits()==picture_start_code)||(nextbits()==group_start_code))
41            } while (nextbits()!=sequence_end_code)
42            ...
43  }/*End of the main function "ComputeBitRate()"*/
```

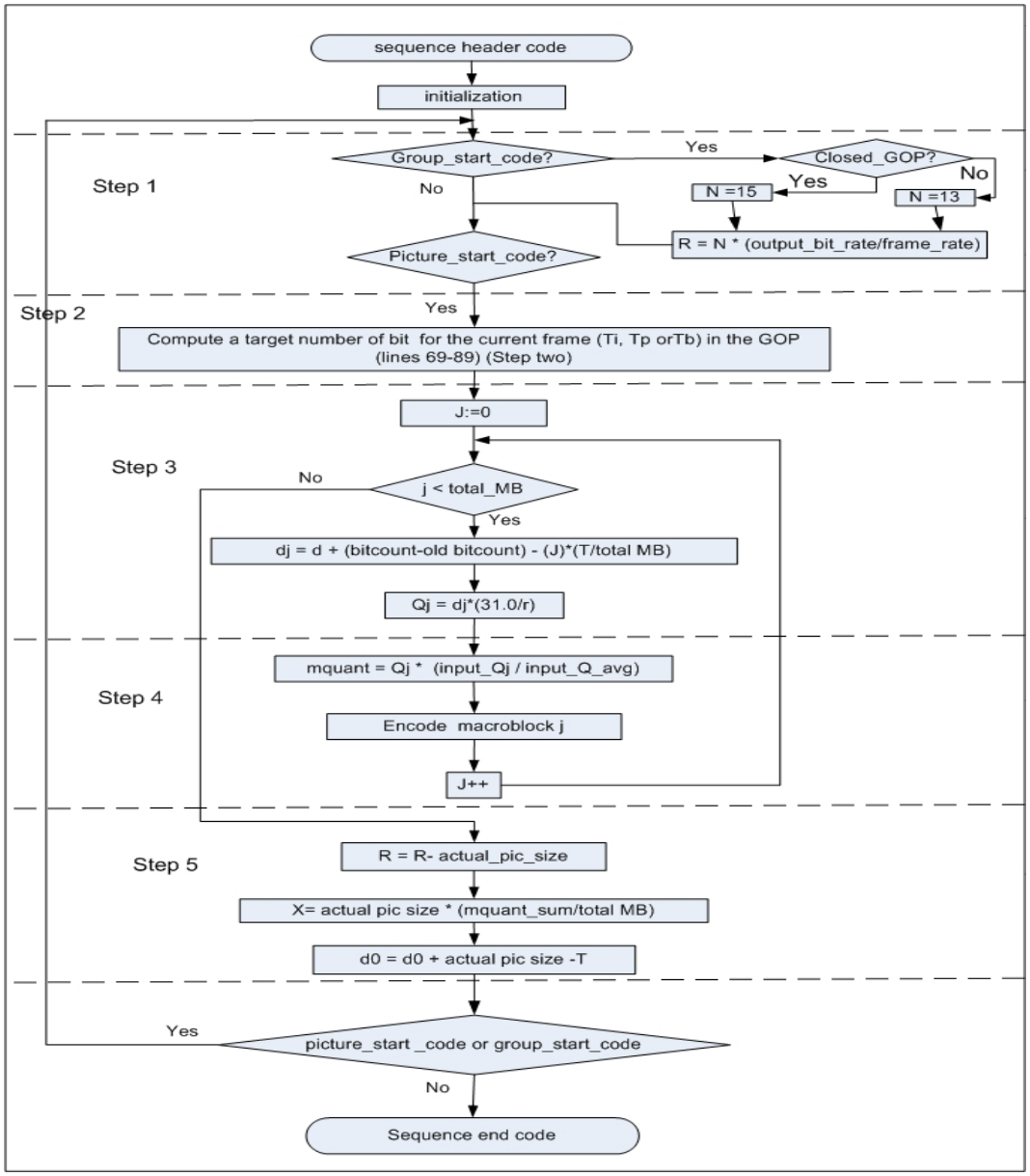


Figure 4.6: The process of the rate control according to adaptive TM-5 of MPEG-2

The problems of the standard version describe as follows:

1. P1: the standard TM-5 algorithm does not change the target bit rate during run-time.
2. P2: the standard TM-5 algorithm requires one GOP delay.
3. P3: the standard TM-5 algorithm calculates the spatial activity of a macroblock in the spatial domain.

P1 and P2 are in step1 and P3 is in step 4.

Step 1: Bit Allocation for GOP(lines 13-26)

P1, one of the important requirements for the transcoder is to change the target bit rate at run-time for every frame. For example, if the bandwidth of network changes, the TM-5 algorithm must modify the output bit rate of the transcoded video. However, standard TM-5 is developed to generate video at a fixed pre-defined bit rate. In such a case, the standard TM-5 algorithm will not generate the right bit rate when the bandwidth fluctuates. Therefore, the standard TM-5 algorithm is not suitable in real-time transcoding when the video content will be transmitted over a wireless link as required in this project.

Standard TM-5 determines R from step one before encoding the first picture within the GOP. R computes according to *output_bit_rate* and N, they are pre-defined. Furthermore, *output_bit_rate* does not change when the wireless network bandwidth is affected by outdoor conditions (microwave, mobile). That means *output_bit_rate* is constant and its value does not change during run-time. However, if *output_bit_rate* is defined as a variable, *output_bit_rate* value can be changed during run-time. In line 4, the adaptive TM-5 algorithm defined *output_bit_rate* as a variable that can obtain an available bandwidth when the wireless network bandwidth fluctuates during transcoding. For example, table 4.2 shows, *output_bit_rate* of the adaptive TM-5 algorithm has the same value as the wireless network bandwidth.

name		the standard TM-5 algorithm		the adaptive TM-5 algorithm	
GOP	bandwidth	R	*output_bit_rate*	R	*output_bit_rate*
1	3	1.8	3	1.8	3
2	3.5	1.8	3	2.1	3.5
3	4	1.8	3	2.4	4
4	3	1.8	3	1.8	3
5	2	1.8	3	1.2	2

Table 4.2: The target bit rate comparison of two bit rate control algorithms, the GOP size is 15 frames and frame rate is 25 frames per second.

P2, a GOP structure is considered as an essential unit for rate control. Furthermore, standard TM-5 usually gives a certain number of bits for the current GOP after knowing the GOP structure. Therefore, the target bit budget for the GOP can only be determined when Np and Nb are known. The GOP structure is often known in advance when it is constant. However, the GOP structure may be a variable structure. In this case, the transcoder needs to read the entire GOP to know its structure before transcoding. Therefore, standard TM-5 requires a delay of one GOP to determine Np and Nb. In this project, such a delay is not acceptable.

In general, the GOP structure is fixed when a simple encoder is used. However, a more advanced encoder can decide the GOP structure. Therefore, for most video streams to be transcoded, the GOP structure is not fixed [17].

Stream can consist of GOPs that have a different structure. In real-time transcoding, such changes of the GOP structure cannot be foreseen. Thus the GOP structure is learned from examined picture GOP header that contains a *closed_gop* parameter as shown in line 13. *closed_gop* is set to '1' to indicate B-pictures have been encoded using only backward prediction or intra coding that is called closed GOP. The adaptive algorithm assumes that closed GOP contains 15 frames; otherwise, it contains 13 frames.

Step 4: Adaptive Quantization (lines 29-36)

Thirdly, the adaptive quantization step computes the normalized activity factor N_actj for each macroblock. N_actj are computed in the spatial domain but transcoding codes the bits stream in the frequency domain. Therefore, transcoding needs a new method to compute the normalized activity factor.

Nakajima proposed a technique that can be used as a solution for P3 (step four) [19]. The technique computes the spatial activity of a macroblock from the ratio between the original adaptive quantization step of the macroblock retrieved from the pre-encoded video and the adaptive quantization step of the current picture. According to this technique, step 4 of standard TM-5 can therefore be replaced by:

$$mquant = Q_j \cdot \left(\frac{input\_Q_j}{input\_Q\_avg} \right) \tag{4.4}$$

This solution can be used in both spatial domain and frequency domain transcoders [19]. The equation 4.4 also decreases the computation time requires in step 4 in standard TM-5.

Conclusion, adaptive TM-5 can change the target bit rate during run-time by obtaining *output_bit_rate* from sender per each GOP. This algorithm that solves P2. gives the GOP size(N) fixed value. Therefore, the adaptive algorithm does not require one GOP delay. Finally, the adaptive quantization step is computed in the frequency domain by using equation 4.4. In this way, the adaptive TM-5 algorithm overcomes P1, P2 and P3.

4.4 Real-time TM-5 algorithm

The adaptive TM-5 algorithm is designed to overcome the problems that associated with the standard TM-5, such as change in GOP size. To solve the GOP size problem, the adaptive TM-5 algorithm gives the GOP size fixed value. The limitations of the adaptive TM-5 algorithm appear when applying in real-time transcoding as follows:

1. GOP size.

2. Response delay.

Firstly, the adaptive TM-5 algorithm assumes the GOP structure to solve P2. Therefore, adaptive TM-5 assumes N is fixed as showing in step one (lines 13-24). However, N is not fixed in reality. For example, GOP size can be 9 frames and this GOP can be a closed GOP. Adaptive TM-5 assumes size for this GOP is 15 frames but is not 9 frames. Therefore, the assumption of N does not work correctly with every GOP. Furthermore, because N is fixed, R become fixed. For example, the bit budget for the GOP consisted of six frames same as the bit budget for the GOP consisted of nine frames. Therefore, there is also problem in the bit budget of the GOP.

Secondly, one of the most important requirements for this project is to provide a solution when the bandwidth fluctuates. As mentioned before, the bandwidth of the streaming video is affected by outdoor conditions (mobile, microwave). Adaptive TM-5 solves the bandwidth fluctuations. However, the adaptive algorithm has a delay of response when bandwidth fluctuates. The algorithm checks the output bit rate per GOP at run-time but the algorithm does not checks the output rate per frame. Therefore, the adaptive algorithm has a delay of some frames.

Therefore, we proposed the Real-time TM-5 algorithm that overcomes the limitation of adaptive TM-5. Real-time TM-5 has the following properties:

- The algorithm determines the bit allocation per frame. Therefore, it does not need to know the GOP size N and does not need the bit allocation for each GOP.

- The algorithm changes the output bit rate at run-time for every frame

- The algorithm must have an average bit rate the same as an average target bit rate.

In MPEG-2, the target number of bits assigned to a frame depends on the complexity of the frame. We assume that X represents a complexity measure for the previous frame. The complexity measure for the frame is defined as a product of an average adaptive quantization step for all the macroblocks used during transcoding the frame and the target number of bits was generated to transcode the frame. Logically, the target number of bits should be proportional to the frame's complexity measure. It is computed as shown below:

$$T = X * C \tag{4.5}$$

We assume a linear relationship between the average frame target and the average complexity, with constant of proportionality C. Question how do we determine the value of C, argument values on T must be equal to the average frame size. C denotes a constant. Here, we have assumed a linear model between T and X. Furthermore, we assume that all the frames have the same complexity measure.

$$Y * frame\_size = Y * T \quad for\ all\ frame\ types\ Y frame\_size = T \tag{4.6}$$

Where $frame\_size$ is the number of bits used to transcode a frame and Y is the number of frame type's. In MPEG-2, Y is 3 {I-frame, P-frame and B-frame}. Equation 4.6 will become as follows:

$$3 * frame\_size = 3 * X * C \tag{4.7}$$
$$3 * frame\_size = (X_i + X_p + X_b) * C \tag{4.8}$$

We need to know $frame\_size$. It is used to compute the number of bits that is assigned for a transcoded frame:

$$frame\_size = \frac{output\_bit\_rate}{frame\_rate} \tag{4.9}$$

After that we need to know C in equation 4.5. From equations 4.8 and 4.9, we have

$$3 * \left(\frac{output\_bit\_rate}{frame\_rate}\right) = (X_i + X_p + X_b) * C \tag{4.10}$$

Equation 4.10 becomes

$$C = \left(\frac{output\_bit\_rate}{frame\_rate}\right) * \left(\frac{3}{X_i + X_p + X_b}\right) \tag{4.11}$$

MPEG-2 requires three complexity measures, one for each frame type. The target bit budget for each frame type is computed as follows:

$$T_i = X_i * C \tag{4.12}$$
$$T_p = X_p * C \tag{4.13}$$
$$T_b = X_b * C \tag{4.14}$$

From equations 4.11, 4.12, 4.12, and 4.12, we have:

$$T_i := \left(\frac{output\_bit\_rate}{frame\_rate}\right) * \left(\frac{3 * X_i}{X_i + X_p + X_b}\right) \tag{4.15}$$
$$T_p := \left(\frac{output\_bit\_rate}{frame\_rate}\right) * \left(\frac{3 * X_p}{X_i + X_p + X_b}\right) \tag{4.16}$$
$$T_b := \left(\frac{output\_bit\_rate}{frame\_rate}\right) * \left(\frac{3 * X_b}{X_i + X_p + X_b}\right) \tag{4.17}$$

We must notice that the bit budget for each MPEG-2 frame depends on the frame type in practice. Therefore, the complexity measures depend on the frame type as following $X_i \geq X_p \geq X_b$. This means that,

$$\left(\frac{3 * X_i}{X_i + X_p + X_b}\right) \geq \left(\frac{3 * X_p}{X_i + X_p + X_b}\right) \geq \left(\frac{3 * X_b}{X_i + X_p + X_b}\right) \tag{4.18}$$

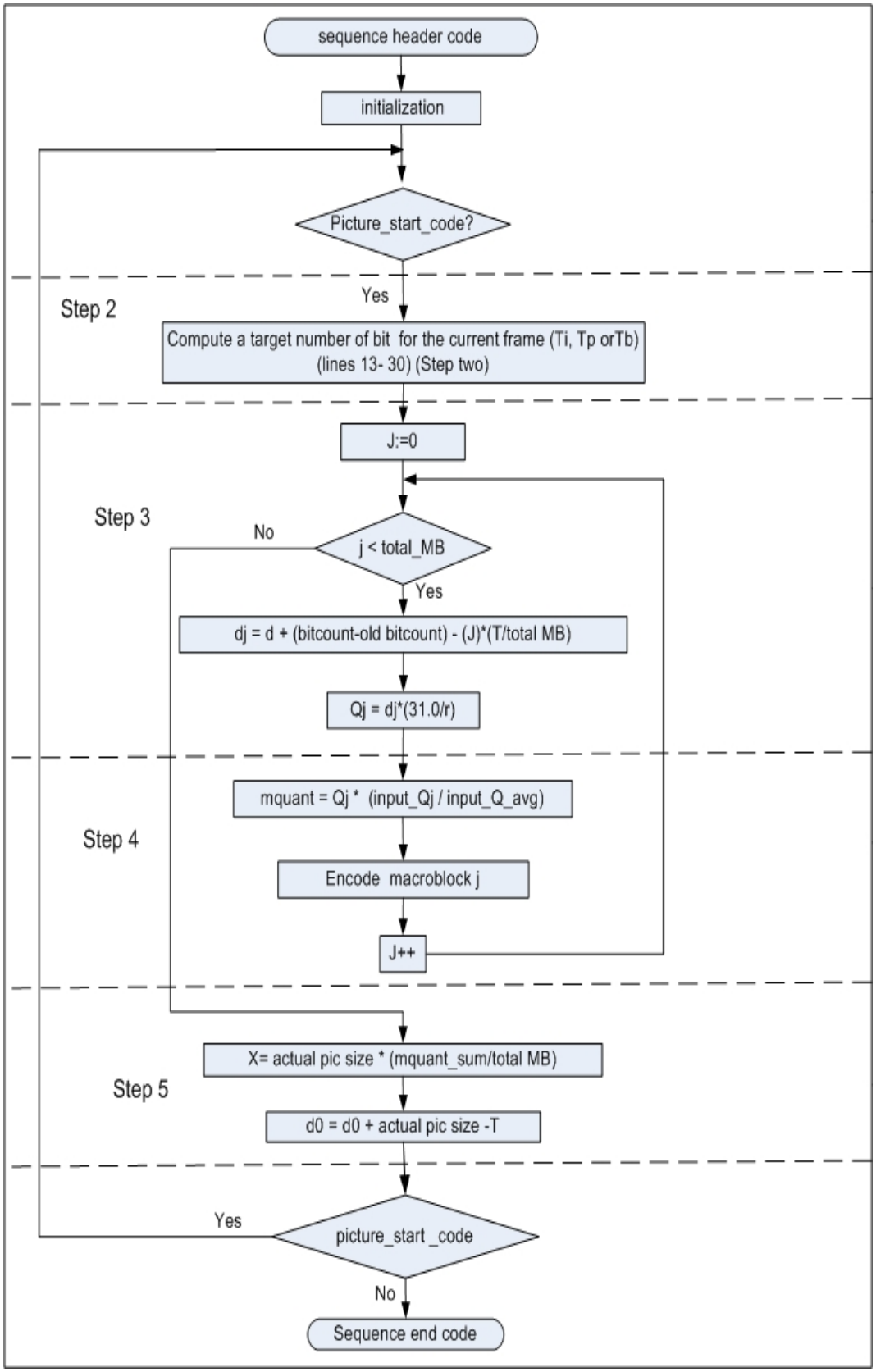


Figure 4.7: the process of the rate control according to real-time TM-5 of MPEG-2

Finally, the complexity of the previous frames decides the bit budget for the next frame. Therefore, high complexity frame obtains more budget then low complexity frame. The real-time TM-5 algorithm does not need step one. Step three, step four, and step five are similar to these steps in the adaptive TM-5 algorithm. The pseudo code of the real-time TM-5 algorithm is described as follows:

```
1   int output_bit_rate          /* output bit rate of video to be transcoded */
2   int T                        /* target number of bits for frame */
3   int input_Q_j                /* the quantization parameter of the MB retrieved from the input stream.*/
4   int input_Q_avg              /* the average quantization parameter of the current frame */
5   ...
6   void ComputeBitRate()        /* This the main function of the standard TM-5*/
7   {
8       ...
9       do {/* while ( nextbits() != sequence_end_code )*/
10          .../* Initialize the TM-5 algorithm */
11          do { /*while (nextbits() == picture_start_code)*/
12                      ...
13                       switch (picture_coding_type)
14                      {
15                          case I_TYPE:
16                          T = (output_bit_rate/frame_rate) * (3*Xi/(Xi+Xp+Xb)) ;
17                          d = d0i;
18                          break;
19                          case P_TYPE:
20                          T = (output_bit_rate/frame_rate) * (3*Xp/(Xi+Xp+Xb));
21                          d = d0p;
22                          break;
23                          case B_TYPE:
24                          T = (output_bit_rate/frame_rate) * (3*Xb/(Xi+Xp+Xb));
25                          d = d0b;
26                          break;
27                      }
28                  Tmin = output_bit_rate/(8.0*frame_rate);
29                  if (T<Tmin)
30                  T = Tmin;
31              /* The target number of bits for the current picture (T) is computed */
32              for(j=0;j<total_MB;j++)
33              {
34                dj = d + (bitcount - old_bitcount) - j * (T/total_MB);
35                Qj = dj * (31.0/r);
36                ...
37                mquant = Qj * (input_Q_j/input_Q_avg) ;
38                .../* use mquant valuse in order to re-quantize the current macroblock */
39              }/* End of for loop */
40              ...
41              update_pict(old_bitcount,bitcount,R,total_MB,mquant_sum,picture_coding_type,T,actsum);
42              next_start_code();
43          } while (nextbits()==picture_start_code)
44      } while (nextbits()!=sequence_end_code)
45      ...
46  }/*End of the main function "ComputeBitRate()"*/
```

4.5 Scaled TM-5 Algorithm

The scaled TM-5 algorithm changes the output bit rate within the time of a frame period at run-time when the available bandwidth fluctuates. Scaled TM-5 requires low computational complexity because it does not require the following steps:

1. Bit allocation for the GOP (step one).
2. Update (step five).

This algorithm does not need to compute the GOP budget and the frame complexity. Figure 4.8 shows a flowchart of the scaled TM-5 process. The pseudo code of the scaled TM-5 algorithm is described in detail next.

```
1   #define                 input_bit_rate ... /* input bit rate of video to be transcoded*/
2   int output_bit_rate     /* output bit rate of video to be transcoded */
3   int S                   /* the size in bits of the incoming frame*/
4   int T                    /* target number of bits for frame */
5   int input_Q_j           /* the quantization parameter of the MB retrieved from the input stream.*/
6   int input_Q_avg         /* the average quantization parameter of the current frame */
7   ...
8   void ComputeBitRate()          /* This the main function of the standard TM-5*/
9   {
10      ...
11      do {/* while ( nextbits() != sequence_end_code )*/
12            .../* Initialize the TM-5 algorithm */
13            do { /*while (nextbits() == picture_start_code)*/
14                  T=S * ( output_bit_rate / input_bit_rate );
15                  /* The target number of bits for the current picture (T) is computed */
16                  for(j=0;j<total_MB;j++)
17                  {
18                    dj = d + (bitcount - old_bitcount) - j * ( T / total_MB );
19                    Qj = dj * ( 31.0 / r );
20                    ...
21                    mquant = Qj * ( input_Q_j / input_Q_avg ) ;
22                    .../* use mquant valuse in order to re-quantize the current macroblock */
23                  }/* End of for loop */
24                  ...
25                  next_start_code();
26               } while (nextbits()==picture_start_code)
27          } while (nextbits()!=sequence_end_code)
28          ...
29  }/*End of the main function "ComputeBitRate()"*/
```

Step 2: Bit Budget for Each Frame

At step 2, the scaled TM-5 algorithm determines the target bit budget for every frame without needing to know the GOP size. Scaled TM-5 scales each frame according to formula 4.19.

$$\frac{output\_bit\_rate}{input\_bit\_rate} \tag{4.19}$$

Furthermore, this algorithm does not have a delay response when bandwidth fluctuates, because the output bit rate is used to scale each frame. In line 14, this step computes the target bit size for each frame. In lines 18 and 19, this algorithm computes the quantization step. In line 21, the algorithm computes the adaptive quantization step that is used to re-quantize a macroblock. In scaled TM-5, these steps work similar to these steps in adaptive TM-5.

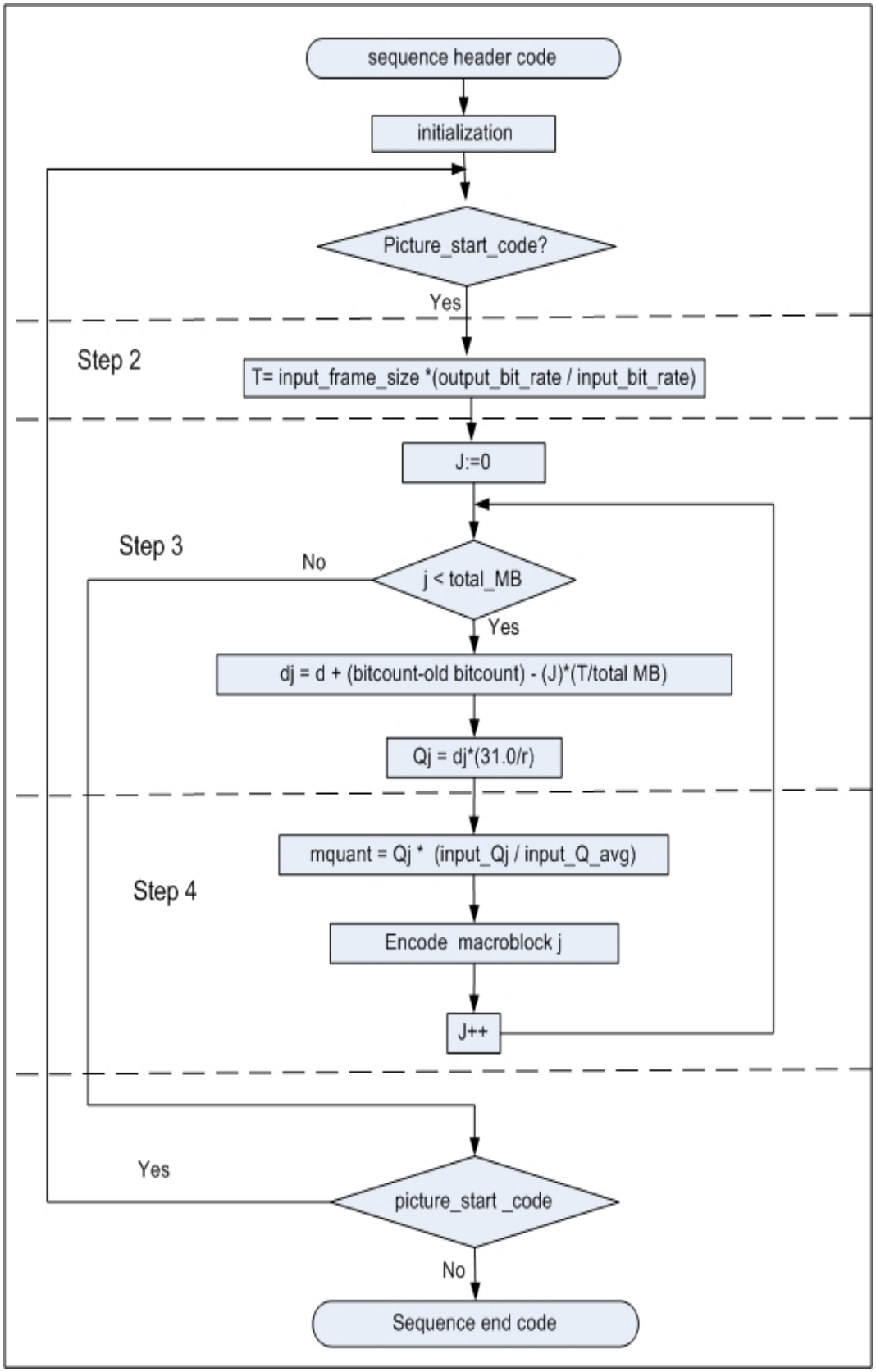


Figure 4.8: the process of the rate control according to scaled TM-5 of MPEG-2

4.6 Experimental Results

In this section, our experiments based on a number of test video sequences. All the video sequences have a spatial resolution of 720 x 576 pixels and a color sampling ratio of 4:2:0. The test sequences are coded using MPEG-2 at various bit rates. Table 4.3 gives the parameters may be used to control the output bit stream.

Sequence name	Encoded bit rate	Frame rate	Sequence size
Flower	9 Mbps	25 f/s	210 Frames
Tank 90	9 Mbps	25 f/s	1000 Frames
Tank T80UK	10 Mbps	25 f/s	1050 Frames
Penguin	5 Mbps	25 f/s	1500 Frames
Football	5.12 Mbps	29.97 f/s	97 Frames

Table 4.3: The parameters of the video sequences that are used in the bit rate control algorithms.

The algorithms that are introduced in the previous sections of this chapter have been implemented in the C programming language. The algorithms are as following: the adaptive TM-5 algorithm (ADTM), the scaled TM-5 algorithm (SCTM), and the real-time TM-5 algorithm (RTTM). They are used to control the output bit stream. The size of the output bit stream can increase or decrease the average bit rate.

The control rate algorithms are compared with each other to select the best algorithm. It will be used to control the output video stream in this project. Thus, the perceived quality picture is compared when using these rate control algorithms. Table 4.4 gives the performance of each algorithm in the average Peak Signal-to-Noise (PSNRs) that is the averaged value of the PSNRs of the frames within the same sequence. Three algorithms provide quite different visual quality. ADTM generally shows better picture quality than other algorithm in the experiments. The average PSNRs of ADTM are 0.074 dB higher than RTTM and the average PSNRs of RTTM are 0.116 dB higher than SCTM. Figure 4.9 shows PSNR with respect to frame number for sequence using RTTM, SCTM and ADTM.

Sequence name	Declared bit rate	Sequence size	average PSNRs	Algorithm
Flower	2.75 Mbps	210 Frames	31.11	RTTM
			31.15	ADTM
			30.58	SCTM
Tank 90	3 Mbps	300 Frames	30.26	RTTM
			30.38	ADTM
			30.20	SCTM
Tank T80UK	3.2 Mbps	300 Frames	30.12	RTTM
			30.22	ADTM
			30.09	SCTM
Football	3 Mbps	97 Frames	30.67	RTTM
			30.77	ADTM
			30.74	SCTM
Penguin	3 Mbps	300 Frames	29.93	RTTM
			29.94	ADTM
			29.84	SCTM

Table 4.4: The average PSNRs of the video sequences that can be used in the bit rate control algorithms.

Table 4.5 shows the average bit rate for each sequence. These sequences are transcoded when bandwidth does not fluctuate. The table gives the average declared bit rate (target bit rate) and the average transcoded bit rate (actual bit rate). For every video sequence, the Table gives the average transcoded bit rate results at different the average declared bit rate. When SCTM is used to control the output stream, the average transcoded bit rate is far from the average declared bit rate. In most cases, the average transcoded bit rate of SCTM has from 190 Kbps to 550 Kbps more than the average declared bit rate. For the same sequences, Table 4.5 gives the average transcoded bit rate by using ADTM. When

Sequence name	Declared bit rate	Max. transcoded bit rate(Mbps)	Avg. transcoded bit rate(Mbps)	Min. transcoding bit rate(Mbps)	Algorithm
Flower	6	5.788	4.496	3.679	RTTM
		6.424	5.626	4.653	ADTM
		6.424	5.595	4.262	SCTM
	3.5	5.623	2.847	1.107	RTTM
		5.785	3.696	2.221	ADTM
		5.777	3.688	2.155	SCTM
Tank 90	4	4.330	3.917	3.592	RTTM
		4.594	4.418	4.111	ADTM
		4.646	4.406	4.144	SCTM
	3	3.576	3.078	2.515	RTTM
		3.695	3.433	3.052	ADTM
		3.83	3.549	3.365	SCTM
Tank T80UK	6	6.202	5.290	3.689	RTTM
		6.616	6.228	5.705	ADTM
		6.721	6.199	5.668	SCTM
	4	4.714	3.730	3.026	RTTM
		4.779	4.352	3.982	ADTM
		4.842	4.421	4.034	SCTM
	3.5	4.18	3.373	2.429	RTTM
		4.421	3.880	3.524	ADTM
		4.535	3.987	3.575	SCTM
Football	4	4.191	3.764	3.561	RTTM
		4.408	4.047	3.902	ADTM
		4.721	4.358	4.183	SCTM
	3	3.782	3.097	2.809	RTTM
		3.755	3.398	3.208	ADTM
		4.203	3.656	3.383	SCTM
Penguin	4	4.177	3.760	1.402	RTTM
		4.467	4.105	1.625	ADTM
		4.612	4.138	1.548	SCTM
	3	3.434	2.960	1.278	RTTM
		3.381	3.180	1.363	ADTM
		3.625	3.306	1.269	SCTM

Table 4.5: The average bit rate comparison of three bit rate control algorithm.

Bandwidth	Response delay	Algorithm
2.5 Mbps	9	ADTM
	0	RTTM
3.0 Mbps	8	ADTM
	0	RTTM
3.5 Mbps	2	ADTM
	0	RTTM
4.0 Mbps	11	ADTM
	0	RTTM
3.0 Mbps	5	ADTM
	0	RTTM
5.0 Mbps	6	ADTM
	0	RTTM

Table 4.6: ADTM has some frames delay when using to control Tank T80UK

ADTM controls the output stream, the average transcoded bit rate has from 100 Kbps to 400 Kbps more than the average declared bit rate. When the video sequences are controlled by using the proposed rate control (RTTM), the average transcoded bit rate is much closer to the declared bit rate, as compared to ADTM. Clearly, in term of average target bit rate, RTTM controls the output bit stream better than ADTM and SCTM.

One of the important requirements for this project is to change a target bit rate for every frame. The three algorithms discussed above, they allow changing a target bit rate at run-time. RTTM and ADTM use the same parameter that can be changed its value at run-time when bandwidth fluctuates. The parameter called the average frame bit budget is computed by equation 4.20.

$$avg = \frac{output\_bit\_rate}{frame\_rate} \tag{4.20}$$

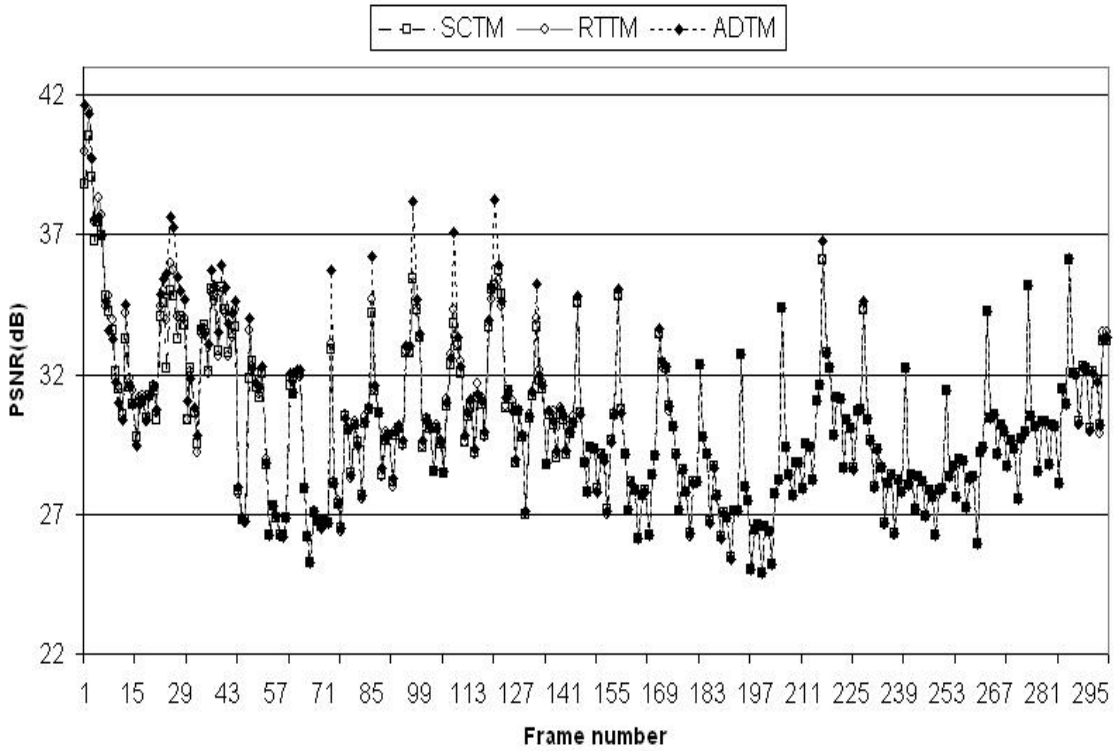
The pseudo code of RTTM and ADTM show this parameter in line 16, line 20 and line 24, and in line 25 respectively. The results of experiments are shown in Figure 4.11 for RTTM and ADTM. The bandwidth fluctuated during these experiments as shown in Figure 4.11(c). The 'Tank T80UK' video encoded at a bit rate of 10 Mbps is transcoded to 5 Mbps. However, during transcoding, bandwidth fluctuates 6 times as follows:

1. At frame 14 a bit rate changes to 2.5 Mbps.
2. At frame 39 a bit rate changes to 3.0 Mbps
3. At frame 70 a bit rate changes to 3.5 Mbps.
4. At frame 85 a bit rate changes to 4.0 Mbps.
5. At frame 115 a bit rate changes back to 3.0 Mbps.
6. At frame 126 a bit rate changes back to 5.0 Mbps.

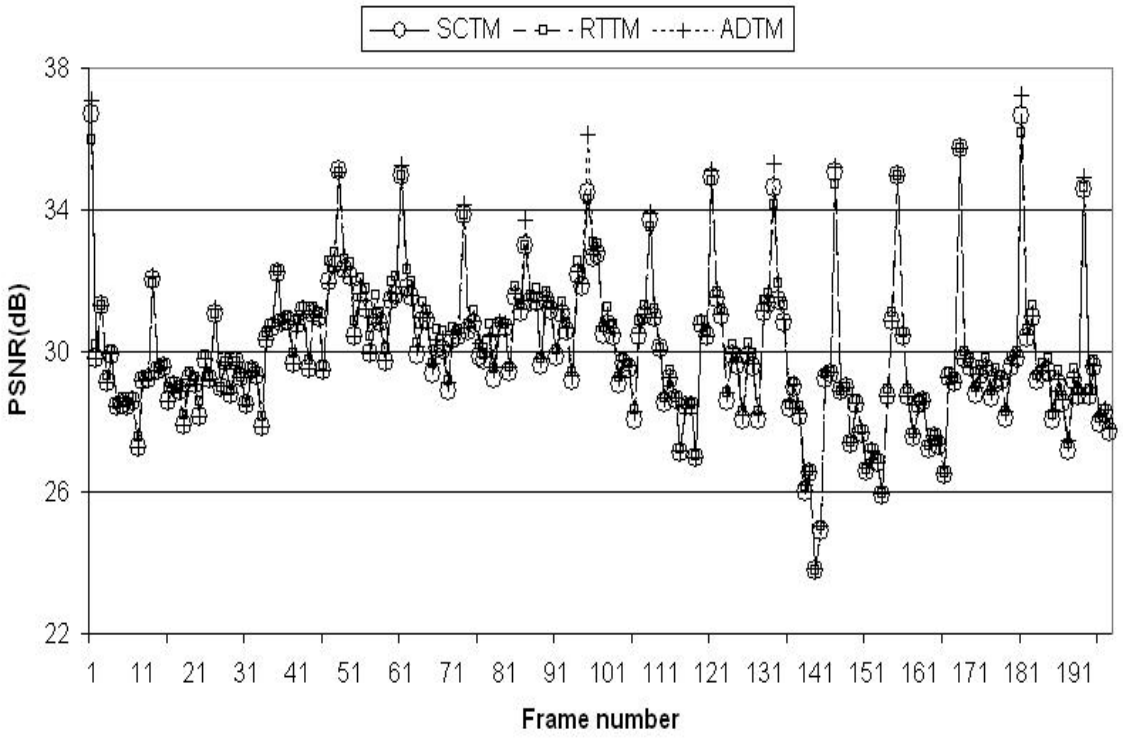
The 'Tank 90' video has the same fluctuation as shown in Figure 4.11(a). In Figures 4.11(a) and 4.11(b), RTTM changes immediately the average frame bit budget when bandwidth fluctuates. Furthermore, the same Figures show ADTM has some frames response delay because ADTM changes the target bit rate for every GOP when bandwidth fluctuates. This means that if the current GOP has a bandwidth fluctuates, ADTM will change the target bit rate in the next GOP. The GOP size in 'Tank 90' video is 12 frames but the GOP size in 'Tank 90' video is variable. Figure 4.11(b) is summary in Table 4.6. The Table gives a response delay for 'Tank T80UK' when ADTM is used to control the output stream.

As mentioned in the scaled TM-5 algorithm, the frame bit budget (T) is computed by equation 4.21.

$$T = S * ratio \tag{4.21}$$

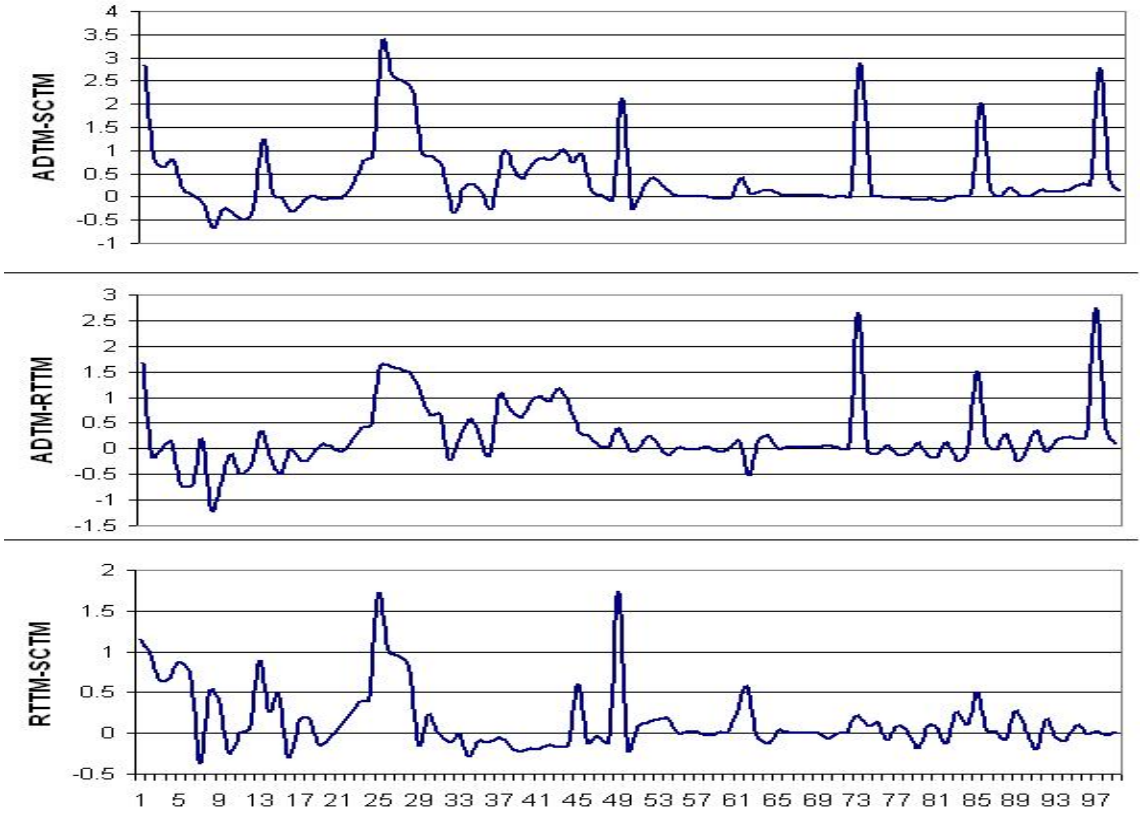


(a) PSNR comparison for 300 frames; RTMT, ADTM and SCTM for Tank 90.

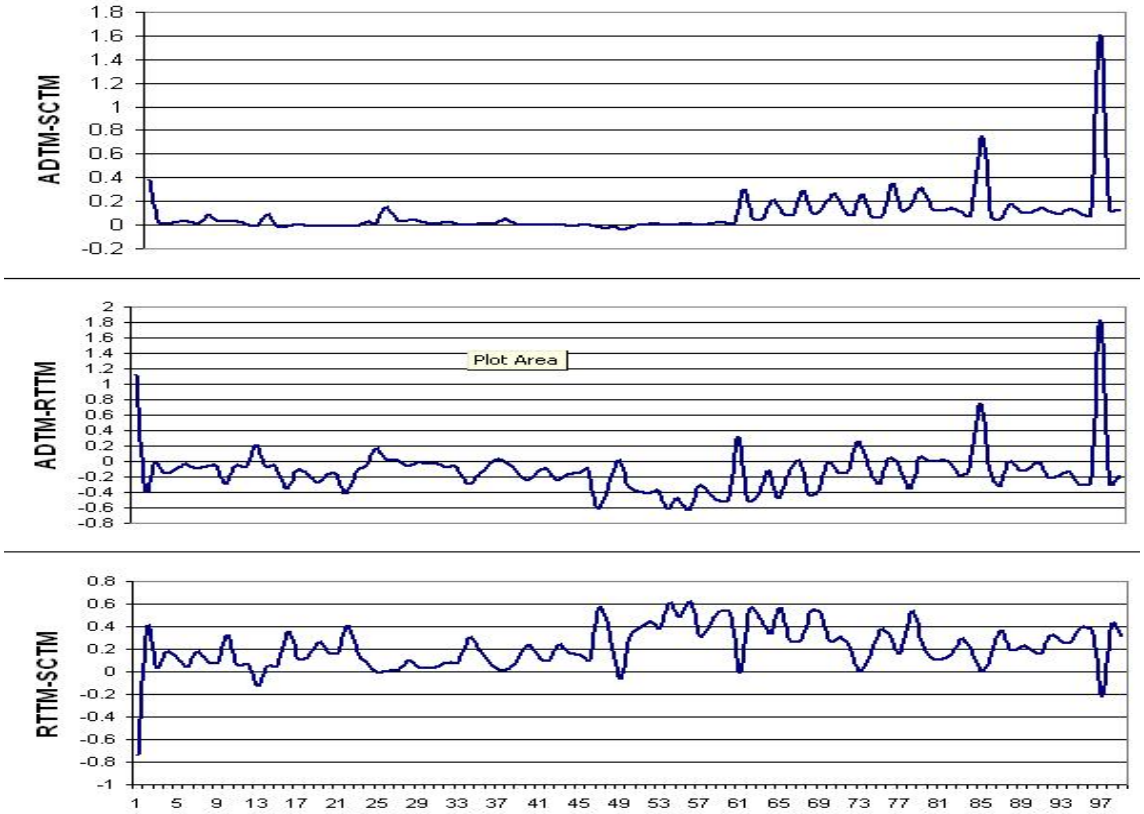


(b) PSNR comparison for 200 frames; RTMT, ADTM and SCTM for Tank T80UK.

Figure 4.9: PSNR comparison

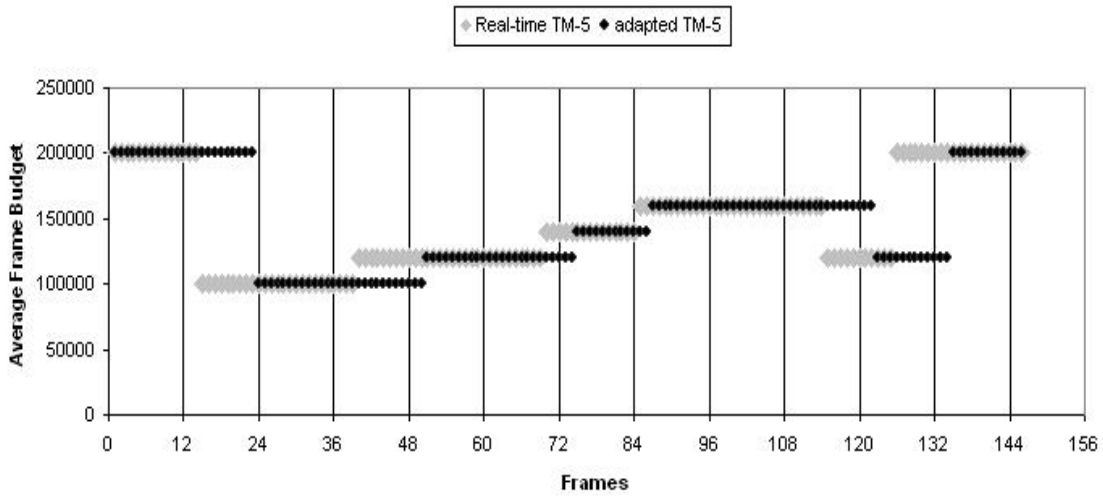


(a) different PSNR for 100 frames when RTMT, ADTM and SCTM are used ; ADTM - SCTM, ADTM - RTMT, and RTTM - SCTM for Tank 90

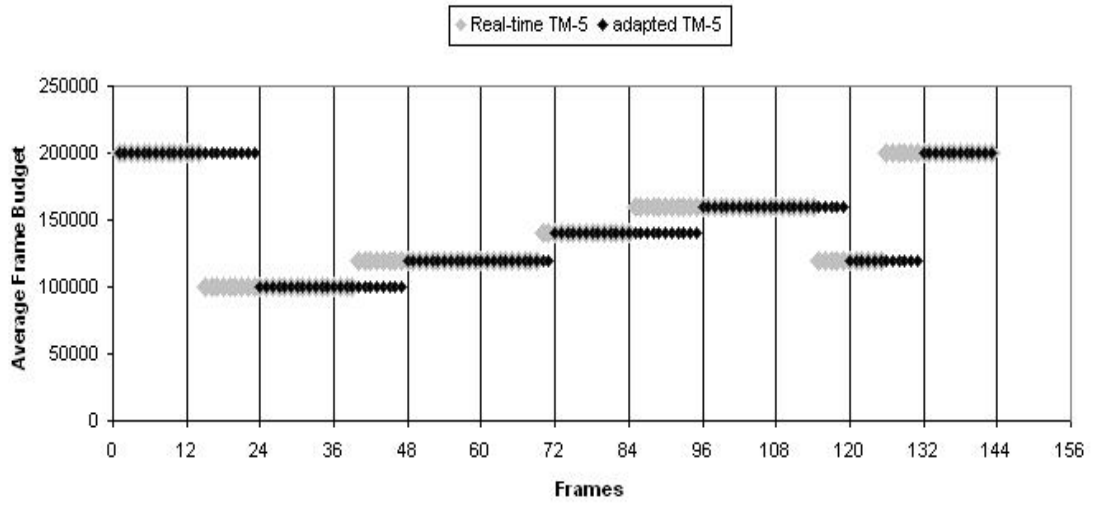


(b) different PSNR for 100 frames when RTMT, ADTM and SCTM are used ; ADTM - SCTM, ADTM - RTMT, and RTTM - SCTM for Tank 80

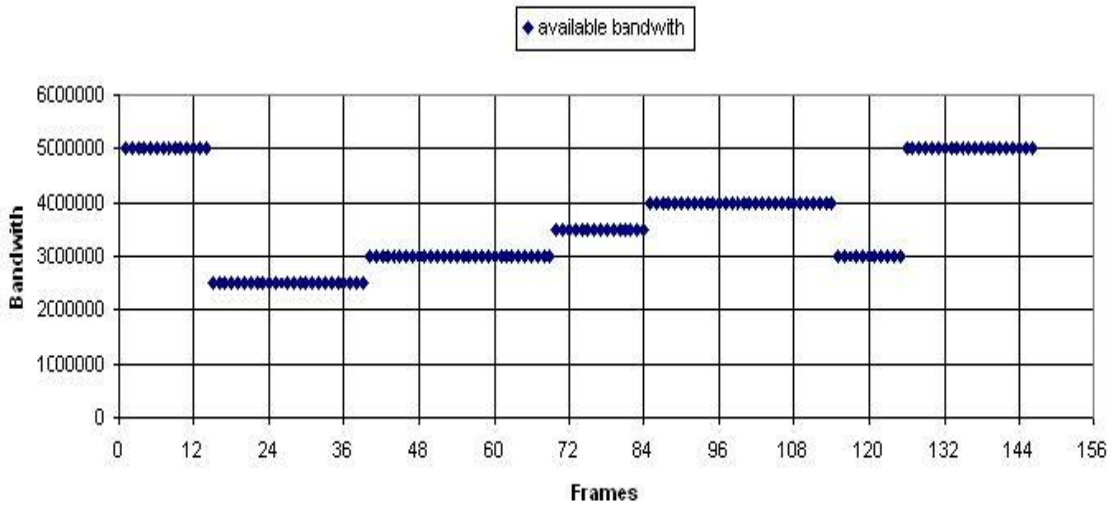
Figure 4.10: PSNR comparison



(a) The average bit budget is changed at runtime for the sequence Tank 90



(b) The average bit budget is changed at runtime for the sequence Tank T80UK



(c) The bandwidth fluctuates

Figure 4.11: RTTM and ADTM chnages an average bit budget when bandwidth fluctuates

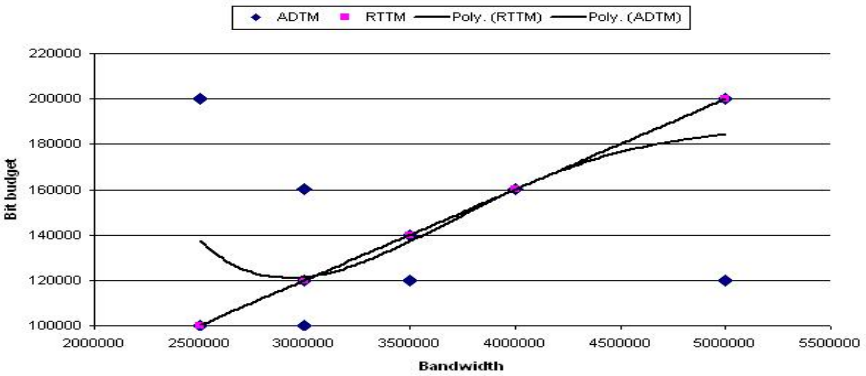


Figure 4.12: RTTM and ATTM correlated

	RTTM	ADTM	SCTM
average bit rate	1	2	3
the perceived quality picture	2	1	3
response delay	1	3	1
computation complexity	2	3	1

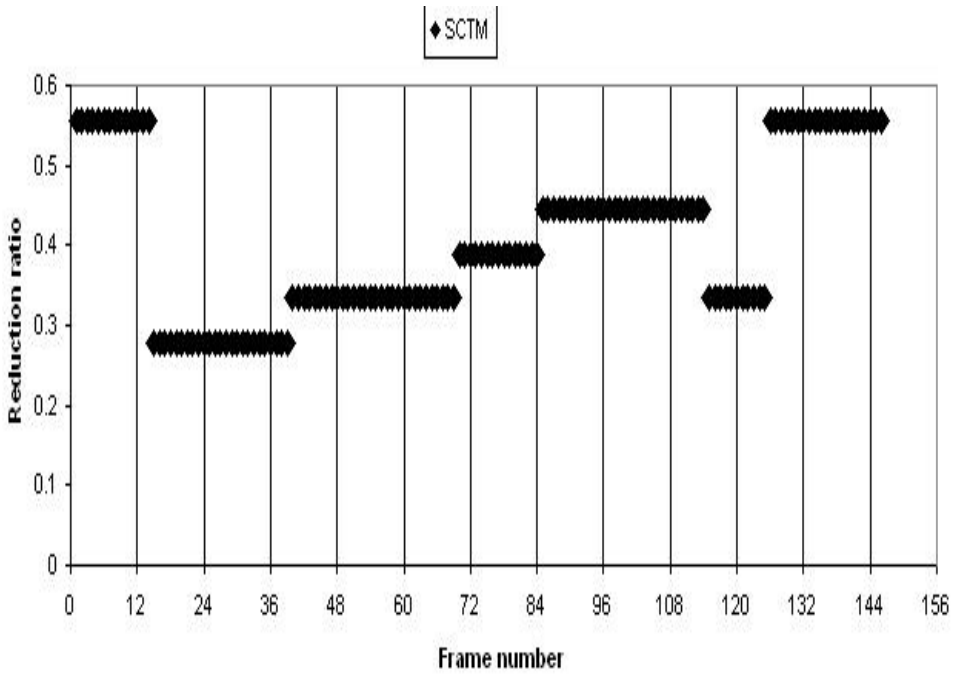
Where S denotes the size in bits of the input frame and $ratio = \frac{output\_bit\_rate}{input\_bit\_rate}$. $ratio$ is a reduction ratio achieved per frame. The equation shows that T is only dependant on the value of $ratio$. $ratio$ is changed as shown in Figure 4.13 when bandwidth fluctuates. This means that each frame is transcoded according to the current value of $ratio$ as shown in the Figure. When a new target bit rate is received by the transcoder, SCTM changes the current value of $ratio$ accordingly. Therefore SCTM does not have a response delay.

4.7 Bit rate control for the enhancement layers

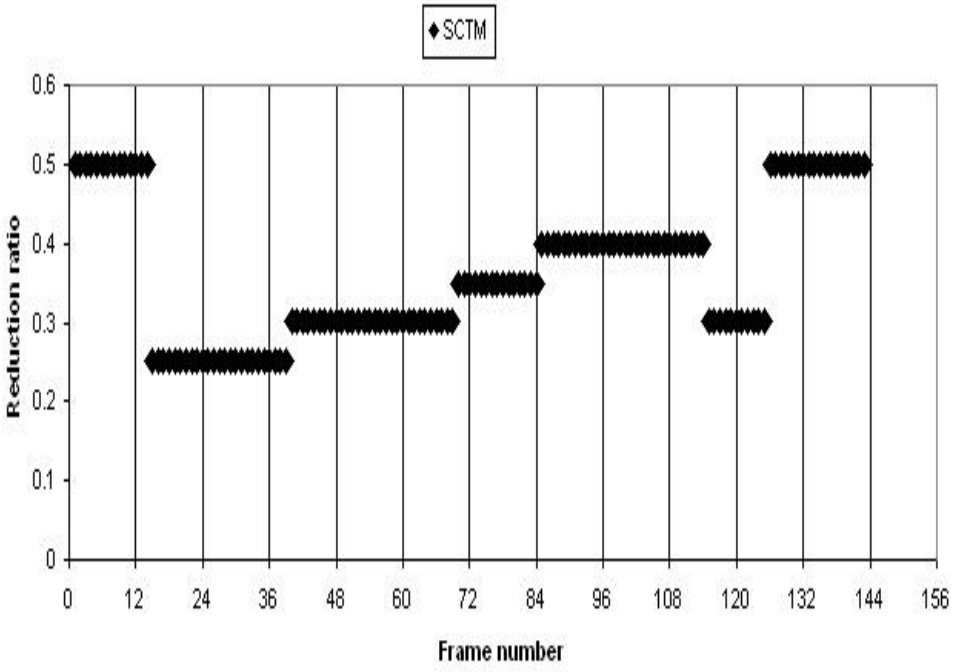
In the previous section it is discussed how to adapt TM-5, such that the real-time requirements concerning the base layer are met by the transcoder. Unfortunately, due to the lack of time, thorough research on how to adapt TM-5 to create a bit rate control algorithm for the enhancement layers could not be completed. However, it is expected that a bit rate control algorithm for the enhancement layers can be developed using the same approach of rate control as used in TM-5 by means of step three of the algorithm. Therefore, it is expected that most effort concerning the development of the bit rate control is required to invent a proper way of determining the bit allocation for the pictures in the enhancement layer.

sequence name	Penguin	Tank T80UK
ADTM	0.035 second	0.030 second
RTTM	0.025 second	0.023 second
SCTM	0.020 second	0.018 second

Table 4.7: The performance comparison of three bit rate control algorithms



(a) The reduction ratio achieved per frame is changed at runtime for the sequence Tank 90



(b) The reduction ratio achieved per frame is changed at runtime for the sequence Tank T80UK

Figure 4.13: SCTM changes the reduction ratio achieved when bandwidth fluctuates

Chapter 5

Conclusions and Future Work

In this project, we have researched, designed, and implemented enhancements to MPEG-2 video transcoding. Some transcoders are implemented by using C programming language, these transcoders are as follows:

1. The closed-loop transcoder.
2. The closed-loop in P-frame transcoder.
3. The hybrid transcoder.

These transcoder are evaluated by using two factors as follows:

1. The real-time performance.
2. The transcoded picture quality.

These transcoder generates MPEG-2 that has better quality than the open-loop transcoder. The closed-loop transcoder improve QoS. Unfortunately, the closed-loop transcoder does not work in real-time. To increase the performance of the closed-loop transcoder, we have investigated a new transcoder architecture that is a closed-loop in a P-frame transcoder. It was chosen as the best candidate. This transcoder must provide better QoS than the open-loop transcoder. Furthermore, the transcoder must work in real-time. Furthermore, the transcoder that must be more suitable for deployment for video streaming over wireless networks must be satisfied some design constraints, such as CPU usage and picture quality. The closed-loop in a P-frame does not perform at real-time.

Therefore, we have designed and implemented a novel transcoding architecture that is a hybrid transcoder. It mixes between the open-loop transcoder and the closed-loop transcoder. The hybrid transcoder has better quality than the open-loop transcoder. This transcoder performs these project requirements. A computational complexity of its transcoding can be changed dynamically

In chapter five, we introduced and proposed some single-to-multilayer transcoders as follows:

1. The open-loop multilayer transcoder.
2. The closed-loop multilayer transcoder.
3. The hybrid multilayer transcoder.

These transcoders may be used to multicast the pre-encoded video over the heterogeneous networks. The proposed transcoder consists of the hybrid transcoder to generate the base layer, with the addition of generating the enhancement layers. The enhancement layers are produced according to SNR scalability. These enhancement layers can be easily dropped according to the available bandwidth.

A bit rate control algorithm is important for transcoding, as it directly determines the size of a transcoded video stream. Some bit rate control algorithms have been investigated. We select TM-5 algorithm because it requires low computational complexity. Unfortunately, these algorithms could not satisfy the requirements of this project. Consequently, TM-5 is modified to allow the change of bit rate control algorithms and to satisfy the reaction time per frame. In chapter four, we proposed two bit rate control algorithms:

1. Scaled TM-5 algorithm.
2. Real-time TM-5 algorithm.

These algorithms are faster than other algorithms. These algorithms do not have a response delay. These algorithms are developed to adjust the bit rate of streaming video in the sender side in accordance to a wireless network situation to avoid packet loosing.

5.1 Future Work

Our transcoders already supports the enhancement layers. Base on the experiments result, the closed-loop transcoder does not work in real-time. Furthermore, we expect that the performance of our transcoders is insufficient to generate a scalable SNR video in real-time using PC. It is Pentium 4 2.4 GHz CPU, 512 MB memory, and running Windows XP. This means that SNR transcoder will not be able to meet these project requirements. Therefore, the performance of our transcoder needs to be increased. This can be achieved as follows:

- Further optimization of the code and compiling it using different optimization flags. For example, MMX optimizations and Intel Compiler can be used to improve the performed.
- Introducing concurrency in the implementation such that data processing and independent operations are carried out simultaneously. These operations include operations like quantization, DCT, IDCT, and variable length coding.
- Pipelining the dataflow in the transcoder. For example, parsing and decoding the next frame can be achieved while re-quantizing the DCT coefficients and generating the output stream.
- Improve motion compensation performances.
- Updating the PC platform to today's standards (e.g. faster CPU).

The actions above expectantly result in a scalable transcoder fast enough to satisfy the corresponding real-time requirements. After that, we recommend to examine the transcoder performance, the transcoder can generate scalable video in real-time. After, the bit rate control for the enhancement layers should be implemented. Next, bit rate control for the enhancement layers should be implemented. Chapter four describes the bit rate control algorithms, which are used to control the base layer, can be employed for the enhancement layers as well. When the implementation of the bit rate control is insufficient, a new algorithm of bit rate control should be carried out.

In the entire transcoding algorithm in this project, significant computation complexity is reduced by assuming that the motion vectors computed at the incoming bit rate are simply reused in the low rate bit stream. Therefore, reusing the motion vectors in this way leads to non-optimal transcoding results due to the mismatch between prediction and residual components. We recommend some works in the motion vectors to improve picture quality.

Chapter 6

Future Work

The available bandwidth is the essential factor in video quality. High bandwidth can provide better video quality than the lower bandwidth. Video scalability provides different quality levels. SNR scalability provides multiple video qualities based on a single video input. SNR scalability is used to increase the robustness of streaming video against packet loss in networks [1].

In this chapter, section 6.1 introduces the challenge of a streaming single video for group of devices over a heterogeneous network. This chapter introduces SNR transcoders that convert MPEG-2 video to scalable MPEG-2 video. A SNR transcoder contains multiple layers. The first layer is 'base layer' and other layers are called 'enhancement layers'. The layers are used to come over heterogeneous bandwidth. Section 6.2 introduces the multilayer cascaded transcoder, Section 6.3 introduces the open-loop multilayer transcoder and section 6.4 introduces the closed-loop multilayer transcoder. We propos a new transcoding architecture in section 6.5 that may be subject for future work.

6.1 Challenge

Video multicasting distributes simultaneously a single video stream to group of devices over a heterogeneous network. One example of a heterogeneous multicasting network is the Multicast Backbone (MBone)[1]. As mentioned in chapter one, the home network may contain different devices, such as PDA, a mobile phone, or a laptop computer. The devices have different bandwidth. How can a single video stream multicast to different devices through different capacity channels ?

Based on the above discussion, the solution is a single-to-multilayer video transcoder that can distribute the pre-encoded video to multilayer video. The multilayer video transcoder provides scalable video content over heterogeneous networks in order to adapt with the available bandwidth. SNR scalable video transcoding can provide multi-quality video layers. For example, MPEG-2 scalable transcoding has a single base layer and one or more enhancement layers. Multilayer transcoding can provide a suitable number of video layers in a way matching with the available bandwidth. This means that SNR scalable transcoder has to provide the best quality under the limitation of the network. Figure 6.1 shows the multilayer transcoder that streams the input video to multiple receivers by using multicast.

6.2 The Multilayer Cascaded Transcoder

In this project, multilayer cascaded transcoding converts MPEG-2 video to scalable MPEG-2 video as shown in Figure 6.2. Scalable MPEG-2 video consists of three quality layers. Each enhancement layer uses a different channel. An important advantage of this transcoding prevents the drift error in the enhancement layers.

In this transcoder, the input of the first enhancement layer is the drifty error of the base layer as shown in Figure 6.2. The input of that is approximately zero. When enhancement layers are missed during transcoding, they will not produce a large drift error in the transcoded video. Finally, multilayer cascaded transcoding requires a large amount of a computational complexity. Therefore, this transcoder is less interesting to use in the future work than other transcoders.

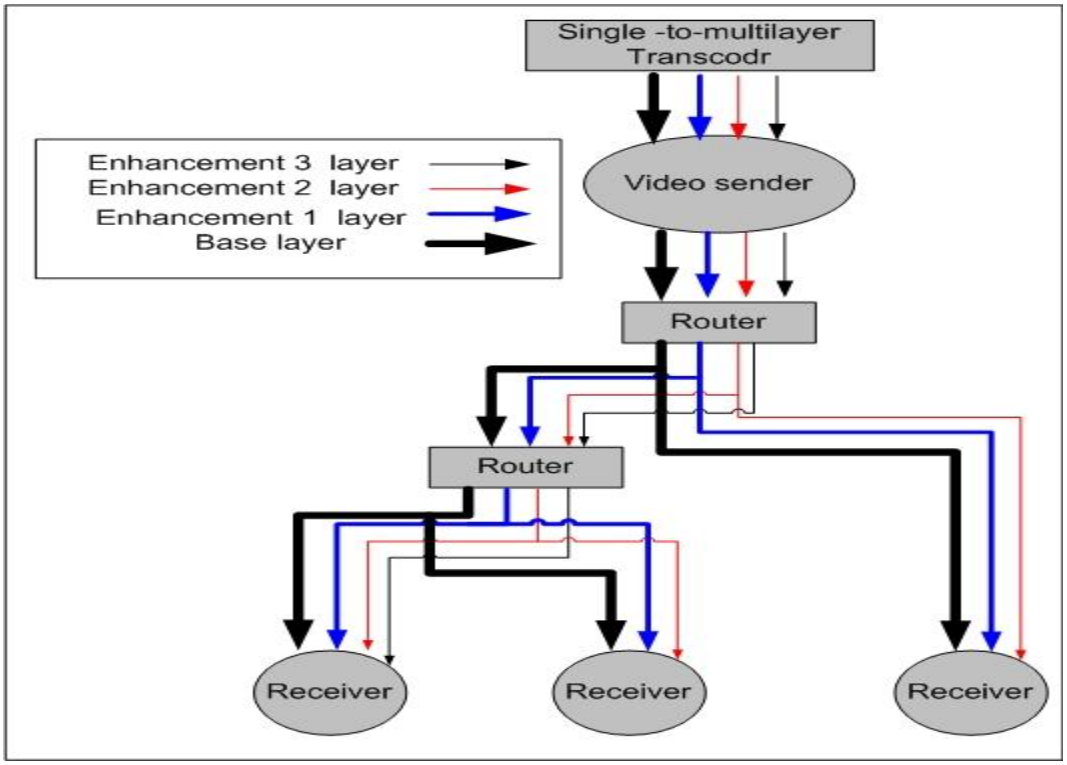


Figure 6.1: Multilayer video transcoding for multicasting of single-layered video.

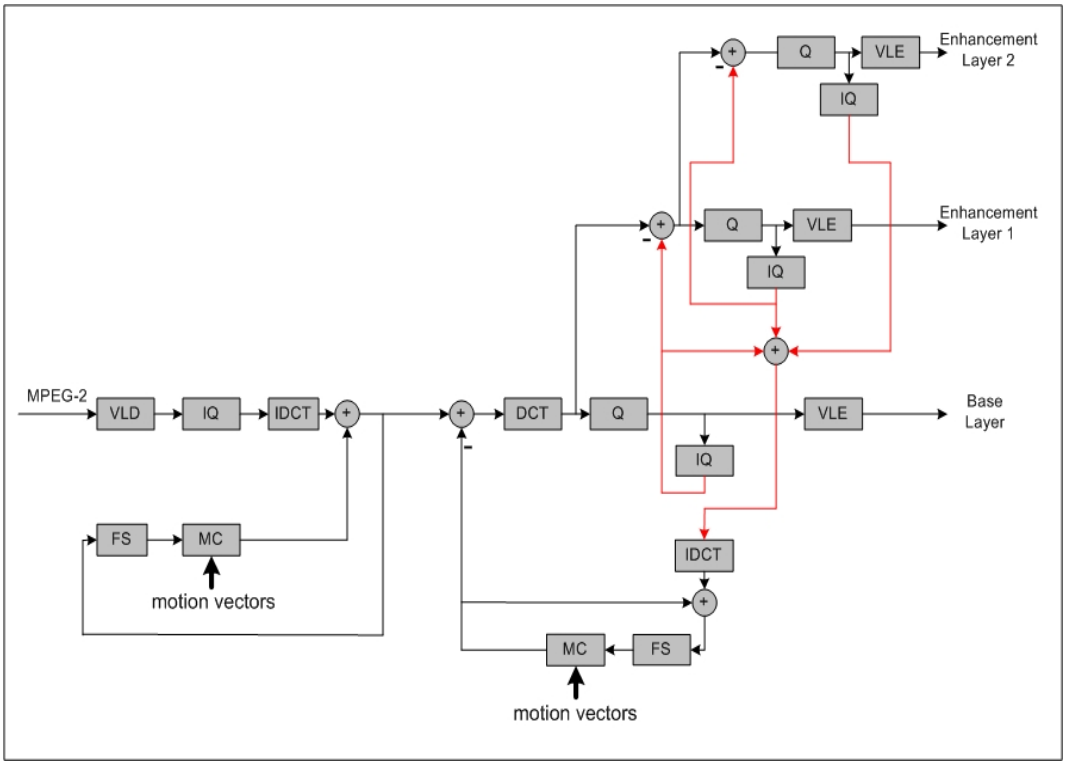


Figure 6.2: A cascaded multilayer transcoder on the multi-loop SNR scalability

6.3 The Open-Loop Multilayer Transcoder

The base layer of the open-loop multilayer transcoder is the same as the open-loop transcoder that is described in chapter 3. In addition, the architecture of the open-loop multilayer transcoder has two enhancement layers as shown in Figure 6.3.

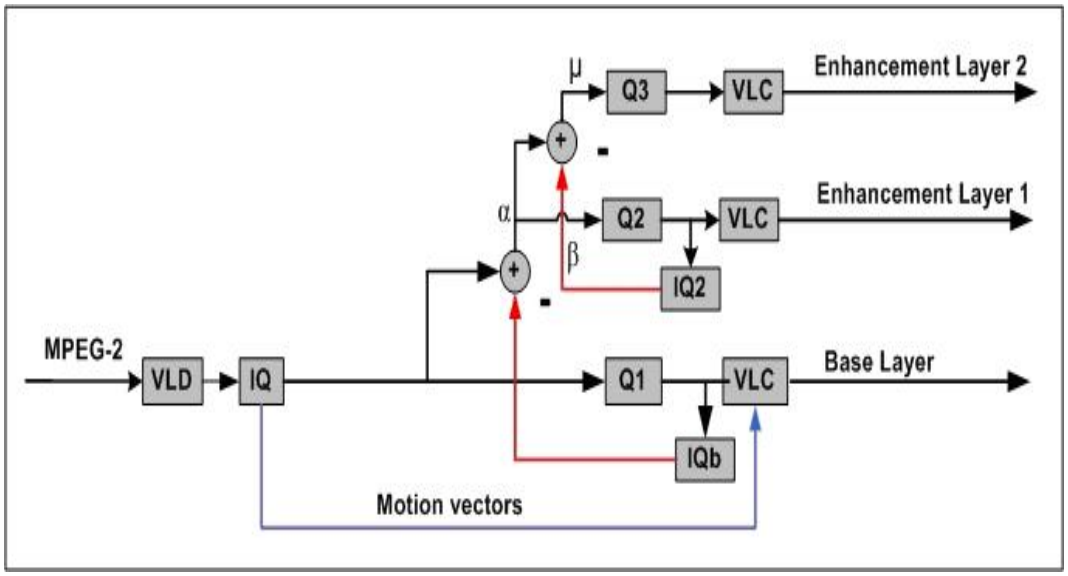


Figure 6.3: The open-loop multilayer transcoder on the multi-loop SNR scalability

In the open-loop multilayer transcoder, VLD of the base layer decodes the input video stream to extract the quantized DCT coefficients, the motion vectors, and the headers. After that, IQ de-quantizes the quantized DCT coefficients to produce DCT coefficients. Then $Q1$ re-quantizes the DCT coefficients to generate a scalable MPEG-2 video stream at a lower rate. Then the re-quantized coefficients are sent to VLC of the base layer that copies the motion vectors, and the headers.

The first enhancement layer gets its input(α) from a difference between IQ and IQ_b . Then, α is quantized by $Q2$ before VLC receives the quantized DCT coefficients. The input of the second enhancement layer is got from the drifty error in a previous layer. $IQ2$ de-quantizes $Q2$ to obtain the originally DCT coefficients (β). This means that the input of second enhancement layer is $\mu = \alpha - \beta$. Than $Q3$ quantizes μ. Then the re-quantized coefficients are sent to VLC of second enhancement layer.

The open-loop transcoder needs only quantize operations that are much less complex than other transcoders. Each layer of this transcoder does not require DCT, IDCT, MC, and memory. This transcoder can provide a scalable video, which may be subject to a large drift error, if GOP is large. A lower quality of an I-frame leads to cumulative errors in later predicted frames (B-frames and P-frames). However, this transcoder is less interesting to use in the future work due to the cumulative errors in later predicted frames (B-frames and P-frames).

6.4 The Closed-Loop Multilayer Transcoder

The closed-loop multilayer transcoder is proposed and verified in [1]. This architecture is presented as a solution to prevent a drift in enhancement layers. In this transcoder, each layer is a free drift. Therefore, each layer of a closed-loop multilayer transcoder should have its own feedback Loop, as shown in Figure 6.4.

In chapter 3, the closed-loop transcoder is introduced in a detail. Each layer in the closed-loop multilayer transcoder works the same as the closed-loop transcoder. We will introduce only the new features in this transcoder.

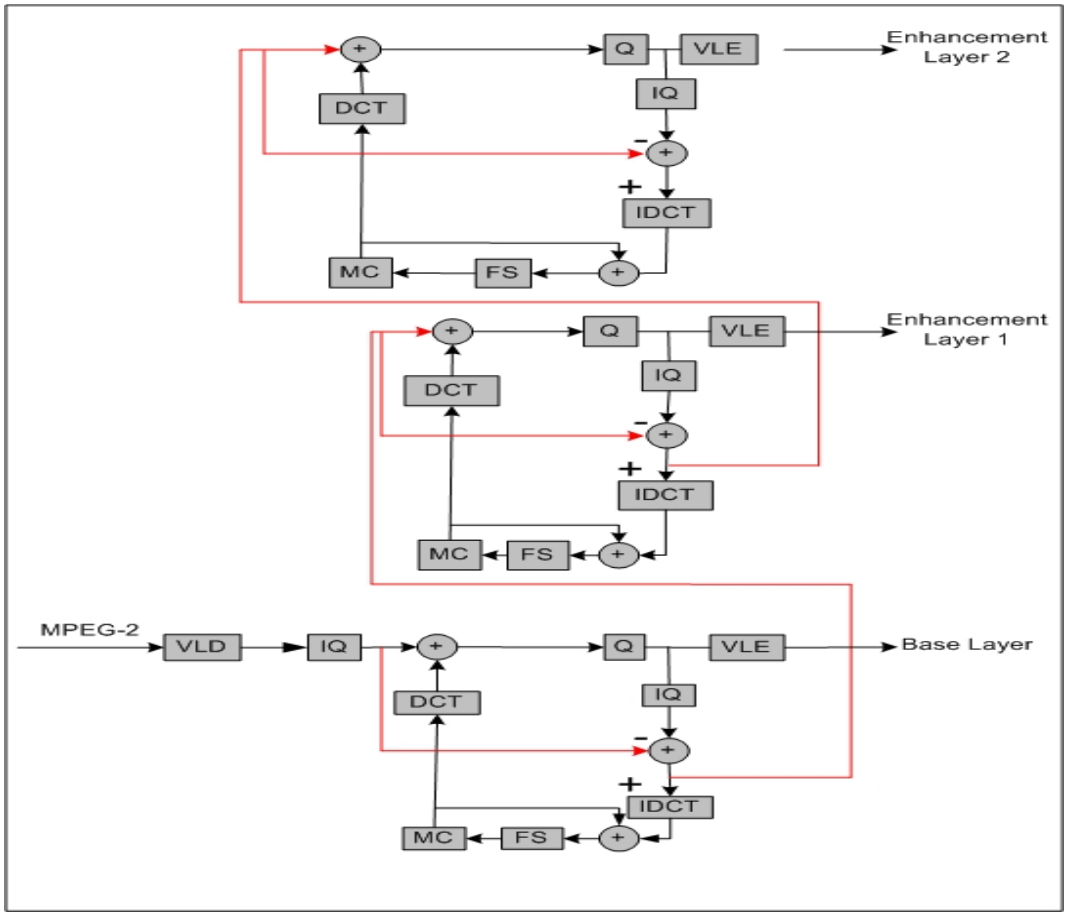


Figure 6.4: Drift compensated multilayer transcoder based on the multi-loop SNR scalability

The base layer provides a variable bit rate video stream. Enhancement layer input represents a re-quantized error during transcoding at the base layer. The input and the output of enhancement layer bit stream is a variable bit rate video stream. The inputs of the enhancement layers depend on the drifty errors in the previous layer. Therefore, their inputs are a variable bit rate video stream. The base layer and the enhancement layer generate the variable bit rate of the transcoded video.

Finally, each layer in closed-loop multilayer transcoding works as follows: the drift errors are accumulated in a feedback loop that uses the motion vectors. The feedback loop gets the motion vectors from the MPEG-2 video stream. Then drifty errors have been obtained from a subtracter, they are converted from the frequency domain to the spatial domain. Then the drift errors are stored at the reference frames memories. The drifty errors are converted to the frequency domain before being added as a correction to the future frames by motion compensation.

6.5 The Hybrid Multilayer Transcoder

In this section, new transcoder architecture is proposed for transcoding video with limited drifting error. The proposed architecture consists of the hybrid closed-loop transcoder to generate the base layer, with the addition of generating the enhancement layers. The hybrid closed-loop transcoder is selected as a compromise between the picture quality of closed-loop transcoding and the performance of open-loop transcoding. The proposed architecture is shown in Figure 6.5. In this transcoder, the enhancement layers are generated according to the available bandwidth. As a result, information of an enhancement layers can be easily dropped.

The hybrid transcoder has the same efficiency as the closed-loop but the hybrid transcoder has some

changes in the original design of the closed-loop transcoder. Closed-loop transcoding adds the re-quantization error for macroblocks in reference frame as introduced in chapter 3. Hybrid transcoding adds the re-quantization error for macroblocks in the reference frame when the sum of the re-quantization error is more than threshold. Hybrid multilayer transcoding can provide high quality video and can reduce a required time for some operations that are not very important. In other words, we may control the process that consumes time. The hybrid multilayer can be faster than the closed-loop transcoding because some processes are removed. The hybrid multilayer transcoder uses an algorithm to control the

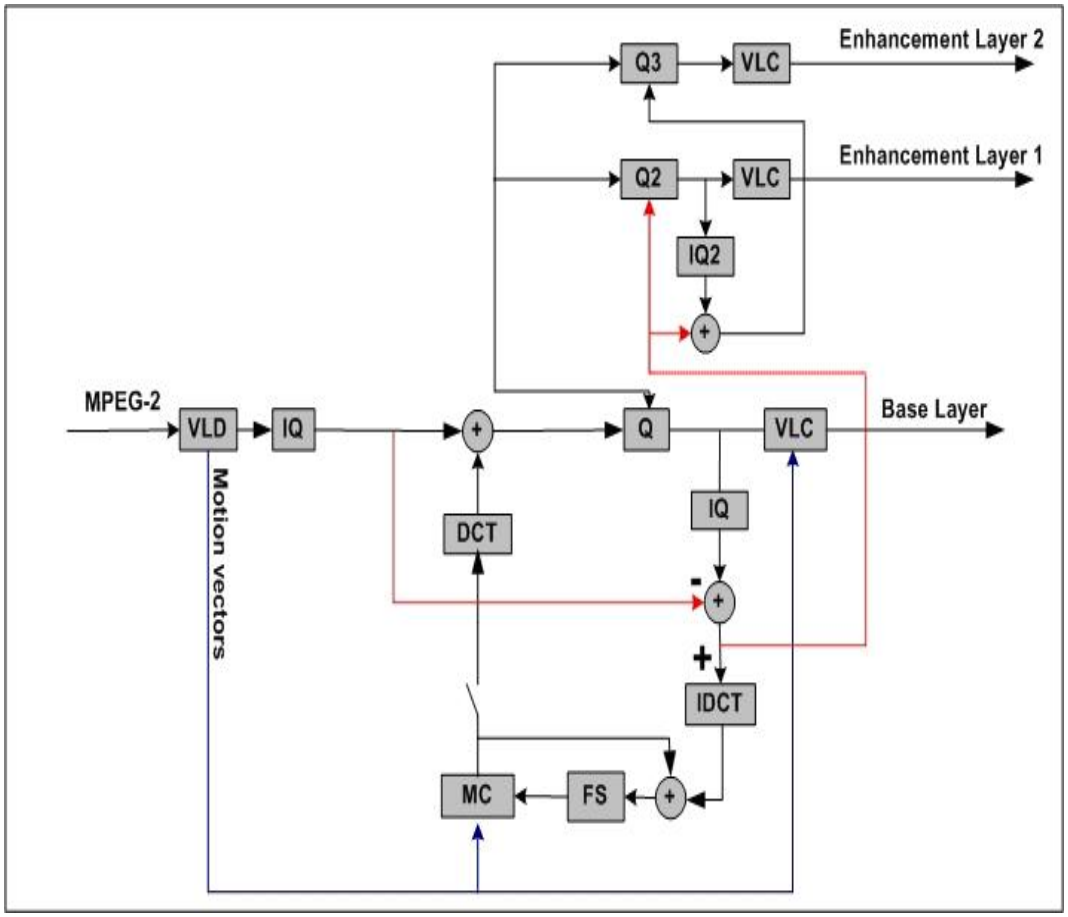


Figure 6.5: The hybrid multilayer transcoder on the multi-loop SNR scalability

re-quantization error at block level. The pseudo code of the hybrid multilayer transcoder algorithm is as follows:

$sum = \sum_{i=0}^{7} \sum_{j=0}^{7} block(i,j)$
$if(block = interblock)$
$\{$
$if(sum > threshold)$
$drifting\ error\ compensation;$
$else$
$no\ drifing\ error\ compesation;$
$\}$

In hybrid multilayer, if the error is more than a certain threshold in a block, it can be added into transcoding loop similar to closed-loop transcoding; otherwise, input block will only go through the re-quantization method.

In closed-loop transcoding, the re-quantization DCT coefficient is computed by equation 6.1.

$$QDCT = (GDCT + error)/step \tag{6.1}$$

Where $QDCT$ is the DCT coefficient, *error* is the re-quantization error, and *step* is the quantization step. In the equation 6.1, the division is integer arithmetic. *error* can increase $QDCT$ when error is

$$error \geq (step - GDCT \ mod \ step) \tag{6.2}$$

In closed-loop transcoding, *error* is always added into transcoding loop. We can observe from equation 6.2 that in some cases *error* does not improve the video quality. In this case, closed-loop transcoding works the same as open-loop transcoding. As a result, closed-loop transcoding improves the video quality when *error* is more than a certain threshold. The enhancement layers are generated according to an adapted form of SNR scalability. As a result, frames of an enhancement layer can be dropped easily.

The multilayer video transcoding requires a multilayer decoder. Therefore, an MPEG-2 standard is introduced the SNR decoder as illustrated in Figure 6.5. The de-quantized DCT coefficients of the enhancement layers are added to the de-quantized DCT coefficients of the base layer, than IDCT transforms the de-quantized DCT coefficients. The reference fame is stored in the frame memory. For more information see the decoder section in chapter 2.

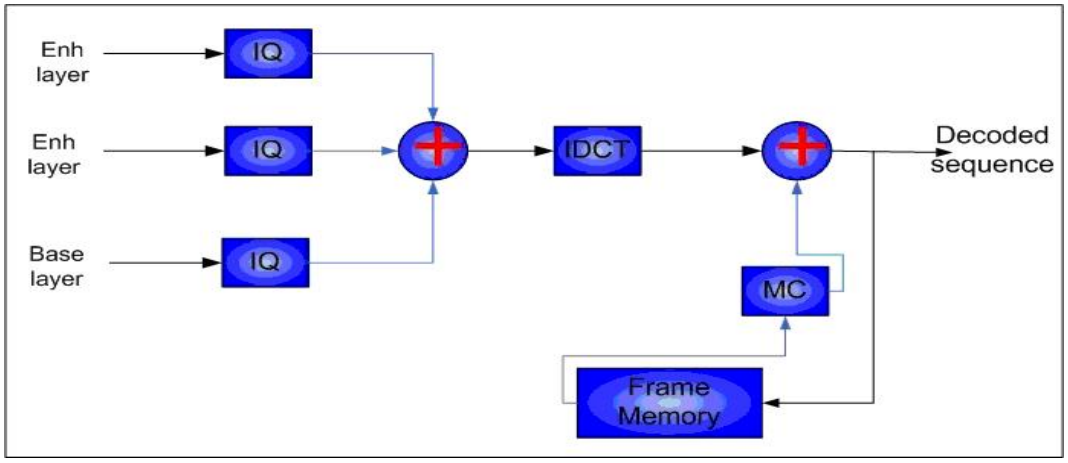


Figure 6.6: Multilayer MPEG-2 SNR decoder

Bibliography

[1] T. Shanableh and M. Ghanabari*," Multilayer Transcoding With Format Portability for Multicasting of Single-Layered Vide,*" IEEE Transactions, Vol. 7, No. 1, pp.1 – 15Feb. 2005.

[2] P. A. A. Assuncao and M. Ghanbari, *"A Frequency-Domain Video Transcoder for Dynamic Bit-Rate Reduction of MPEG-2 Bit Streams*," Circuits and Systems for Video Technology, IEEE Transactions, Vol. 8, No. 8, pp.953 – 967,Dec 1998.

[3] H. Sun, W. Kwok, and J. W. Zdepski, *"Architectures for MPEG Compressed Bit stream Scaling."* IEEE Transactions On Circuits And Systems For Video Technology, Vol. 6, No. 2, April 1996.

[4] A. Vetro, C. Christopoulos, and H. Sun, *" Video Transcoding Architectures and Techniques: An Overview,"* Signal Processing Magazine, IEEE, Vol. 20, No. 2, pp. 18-29 Mar 2003.

[5] J. XIN, C. –N. LIN, and M. -G. SUN, *"Digital Video Transcoding,"* Proceedings of the IEEE, Vol. 93, No. 1, pp. 84-97, Jan 2005.

[6] T. Shanableh, *"Embedded SNR multilayer video transcoding with MPEG-2 compliancy,"* Electronics Letters 3rd, Vol. 41 No. 5, March 2005.

[7] S. -F. Chang and D. G. Messerschmitt, *"Manipulation and Compositing of MC-DCT Compressed Video."* Selected Areas in Communications, IEEE Journal, Vol. 13, No. 1, pp. 1-11, Jan 1995.

[8] A. Leventer and M. Porat*, "On Bit Allocation in Video Coding and Transcoding"* Visual Information Engineering, International Conference, pp. 254-257 July 2003.

[9] D. Qiao and Y. F. Zheng, "*Dynamic Bit-Rate Estimation and Control for Constant-Quality Communication of Video,"* Intelligent Control and Automation, 2000. Proceedings of the 3rd World Congress, Vol. 4, pp. 2506-2511, 2000.

[10] M. Domaski and placeS Makowiak, *"MPEG2 – Based Video Coding with Three-Layer Mixed Scalability," Lecture* Notes In Computer Science, Vol. 2124, pp.110 - 117, 2001

[11] A. Vetro, J. Cai, and C. W. Chen, *"Rate-reduction transcoding design for wireless video Streaming" Journal of Wireless Communications and placeMobile Computing*, Sept. 2002.

[12] W. X. Guo, L. J. Lin, and I. Ahmad*, " MPEG-2 Profile transcoding in MC-DCT domain,"* Multimedia and Expo, ICME 2001, IEEE International Conference, pp. 1155-1158 Aug. 2001.

[13] D. Jarnikov, P. van der Stok, J. Lukkien, *" Wireless streaming based on a scalability scheme using legacy MPEG-2 decoders*" Ninth IASTED, International Conference on Internet & Multimedia Systems & Applications.

[14] John Watkinson, *"The MPEG handbook MPEG-1, MPEG-2, MPEG-4,"* 2ed edition, Elsevier, 2004.

[15] Yun Q. Shi, Huifang Sun, *"Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms, and Standards,"* CRC Press, 2000.

[16] L.Yuan, F. Wn, Q. Chen, S. Li, and W. Gao, *" The Fast Close-Loop Video Transcoder with Limited Drifting Error,"* Circuits and Systems, ISCAS 2006, Proceedings. 2006 IEEE International Symposium, pp. 4,May 2006.

[17] C.J.J. Brouwers, *"A SNR Scalable Transcoder for MPEG-2 Video Stream,"* master thesis, 2004.

[18] ISO/IEC JTC1/SC29 WG11, 1993. Test Model 5. http://www.mpeg.org/MSSG/tm5/Overview.html.

[19] Yasuyuki Nakajima, Hironao Hori and Tamotsu Kanoh ,1995. *"Rate conversion of MPEG coded video by re-quantization process"*.IEEE, 1995.