Technical University of Crete

Electronic & Computer Engineering Department

Digital Image & Signal Processing Laboratory

Diploma Thesis

Subject:

Adaptive Compression of Regions of Interest of Video in Real Time

By Omiros D. Kourakos

Committee: Prof. Dr. M. Zervakis (Thesis Advisor)
Prof. Dr. N. Sidiropoulos (Co-examiner)
Assoc. Prof. Dr. E. Petrakis (Co-examiner)

July 2005

Chania

Acknowledements

I would like to express my thanks, sincere gratitude and appreciation to my advisor Prof. M. Zervakis for his support, guidance and patience throughout the development of this thesis.

I am also grateful to all of my Professors and their assistants, for the knowledge they transmitted to me and to all the staff of the Technical University of Crete for their care and interest during the five years of my undergraduate studies.

Last but not least, I wish to thank my family for their support and their encouragement during my school and academic years.

"It is possible to fail in many ways... while to succeed is possible only in one way."
Aristotle, Nichomachean Ethics.

Dedicated to my family.

Table Of Contents

1. Introduction

1.1 What is adaptive Video Compression?

Video adaptation is an emerging field that includes a body of knowledge and techniques responding on various challenges of the video encoding process, namely the resource constraints (bandwidth, CPU speed, power, display capability etc.) and their huge variability.

Adaptive Video Compression is a very broad term and can be interprated in various ways, depending on the optical corner from which you see it. There is a wide variety of adaptation approaches; signal-level vs. structural level vs. semantic level, transcoding vs selection vs. summarization or bandwidth vs. power vs time constrained. A very good introduction to video adaptation, the basic concepts, technologies and open issues is [9].

The most common work on adaptive video compression focuses on adaptively determining some of the parameters of the compression process (quantization scales, quantization matrices, spatial resolution, temporal resolution (frame rate), group of pictures in MPEG-2 etc.) based on feedback from the network (bandwidth adaptation), for transmission of the video through the network. The target is to regulate the bit rate of the video stream in order to meet the available bandwidth and to avoid loss of information, network congestion etc.

Others (see Appeendix B, [10]) explore the Human Visual Systems (HVS) and establish a method that computes a metric called just noticeable distortion (JND) (signal level adaptation). Based on this metric they set a threshold on the distortion that can be introduced without affecting the subjective quality. Based on this threshold the optimal quantization matrices and quantization scales are chosen. However, most of these schemes are pretty complex and have been only implemented in image compression. Video compression schemes usually take only advantage of some characteristics of the HVS. For example, MPEG-2 takes advantage of the fact that the HVS cannot easily perceive the distortion introduced in areas with high activity and quantizes more heavily these regions of the image than regions with small activity,

Finally, there is also semantic video adaptation. Semantic video adaptation [11] [14] allows the encoding of video content with different viewing quality, depending on the relevance of the video content from the user's viewpoint. The biggest challenge in this case is to extract the contents of the video and to provide a description to the user. Video segmentation and semantic annotation is a very difficult task that hasn't been explored into great depths yet.

1.2 Target of this Thesis

Surveillance applications impose a different concept of quality to other applications, such as entertainment. Subjective degradation in images, video or sound is important only if it inhibits its use. In sequences acquired from fixed surveillance cameras only a part of the frame is important (for example people passing by), while the rest of the frame is of no or little importance (surroundings, backgroung). Improving the quality of important regions, which we call regions of interest (ROIs) and compressing more heavily the rest of the frame (non-ROIs) will improve the coding efficiency of this type of video without inhibiting its use. In addition, in security applications the encoding delay from the moment a camera records a certain incident till the recording reaches the surveillance centre is a very important factor and must be as low as possible. Thus, most of the surveillance applications require real time video compression.

Our target is to adaptively compress the regions of the frames of the video acquired from fixed surveillance cameras on a common PC, without using special hardware for video compression, and to see whether real time compression is possible.

First we classify the regions of each frame in Regions of Interest (ROIs: moving objects in our case) and non-ROIs. The classification of the regions in ROIs and non-ROIs is not the focus of this thesis and a simple scheme is used to discriminate between the regions. Then we compress ROIs more finely than non-ROIs. However, non-ROIs should't be compressed very heavily and they should maintain an acceptable quality or else the operator of the surveillance application would get tired (or even dizzy) from the very bad quality.

We introduce the most popular compression standads, which are JPEG, MPEG-2 and MPEG-4, and explore their capabilities. Finally, we implement one adaptive MPEG-2 encoder and two adaptive M-JPEG encoders and measure their performance in terms of quality and speed.

1.3 Organization of thesis

The organization of this thesis is as follows:

- In chapter 2 we present the JPEG still Picture Compression Standard.
- In chapter 3 the MPEG-2 video standard is described and
- In chapter 4 we make a brief introduction in the MPEG-4 strandard.
- In chapter 5 we present our implementions, the test results and some conlusions for each implementation.
- In chapter 6 we have the final conclusions from this thesis and
- Finally, in chapter 7 some suggestions for improvements.

2. The JPEG Still Picture Compression Standard

JPEG [1], [2] was designed to compress color or gray-scale continuous-tone images of real-world subjects: photographs, video stills, or any complex graphics that resemble natural subjects. Animations, ray tracing, line art, black-and-white documents, and typical vector graphics don't compress very well under JPEG and shouldn't be expected to. Although JPEG is used to provide motion video compression in industrial applications (M-JPEG) due to its simple structure that allows fast implementation and avoids delays, the standard makes no special provision for such an application.

The JPEG standard has four parts. Part 1 is the basic JPEG standard, which defines many options and alternatives for the coding of still images of photographic quality. Part 2 sets rules and checks for making sure software conforms to Part 1. Part 3 adds a set of extensions to improve the standard. Finally, Part 4 defines methods for registering some of the parameters used to extend JPEG.

2.1 Modes of Operation

JPEG is not a single algorithm. Instead, it may be thought of as a toolkit of image compression methods that may be altered to fit the needs of the user. Jpeg defines the following modes of operation:

1) Sequential encoding: each image component is encoded in a single left-to-right, top-to-bottom scan.
2) Progressive encoding: the image is encoded in multiple scans for applications in which transmission time is long, and the viewer prefers to watch the image build up in multiple coarse-to-clear passes (video on demand applications).
3) Lossless encoding: the image is encoded to guarantee exact recovery of every source image sample value.
4) Hierarchical encoding: the image is encoded at multiple resolutions so that lower-resolution versions may be accessed without first having to decompress the image at its full resolution.

Implementations are not required to provide all of these modes.

The most widely used mode of operation is the baseline sequential encoding, a rich and sophisticated compression method, which is sufficient for many applications. It provides a capability which is sufficient for many applications.

Baseline codecs, defined in JPEG-Part 1, allow only 8 bits sample precision and only Huffman entropy coding. In addition to the Baseline sequential codec, other DCT sequential codecs are defined to accommodate two different sample precisions (8 and 12 bits) and two different types of entropy coding methods (Huffman and arithmetic).

Progressive and hierarchical image buildup, improved compression ratios using arithmetic encoding and the lossless compression scheme are defined as extensions to the baseline specification for JPEG (JPEG-Part 1). These features are beyond the needs of most JPEG implementations and have therefore been defined as "not required to be supported" extensions to the JPEG standard. In any decoder using *extended DCT-based*

decoding processes, the baseline decoding process is required to be present in order to provide a default decoding capability.

Finally, the lossless coding process is not based upon the DCT and is provided to meet the needs of applications requiring lossless compression. These lossless encoding and decoding processes are used independently of any of the DCT-based processes.

2.2 Progressive & Hierarchical Processes

For the *progressive DCT-based mode*, 8x 8 blocks are typically encoded in multiple *scans* through the image. This is accomplished by adding an image-sized coefficient memory buffer between the quantizer and the entropy encoder. As each block is transformed by the forward DCT and quantized, its coefficients are stored in the buffer. The DCT coefficients in the buffer are then partially encoded in each of multiple scans. The typical sequence of image presentation at the output of the decoder for sequential versus progressive modes of operation is shown in Figure 2.1.

There are two procedures by which the quantized coefficients in the buffer may be partially encoded within a scan. First, only a specified band of coefficients from the zig-zag sequence need be encoded. This procedure is called *spectral selection*, because each band typically contains coefficients which occupy a lower or higher part of the frequency spectrum for that 8x8 block.

Secondly, the coefficients within the current band need not be encoded to their full (quantized) accuracy within each scan. Upon a coefficient's first encoding, a specified number of most significant bits are encoded first. In subsequent scans, the less significant bits are then encoded. This procedure is called *successive approximation*. Either procedure may be used separately, or they may be mixed in flexible combinations.

In *hierarchical mode*, an image is encoded as a sequence of *frames*. These frames provide reference reconstructed components which are usually needed for prediction in subsequent frames. Except for the first frame for a given component, differential frames encode the difference between source components and reference reconstructed components. The coding of the differences may be done using only DCT-based processes, only lossless processes, or DCT-based processes with a final lossless process for each component. *Downsampling* and *upsampling filters* may be used to provide a pyramid of spatial resolutions (fig. 2.2). Alternatively, the hierarchical mode can be used to improve the quality of the reconstructed components at a given spatial resolution.

Hierarchical mode offers a progressive presentation similar to the progressive DCT-based mode but is useful in environments which have multi-resolution requirements. Hierarchical mode also offers the capability of progressive coding to a final lossless stage.

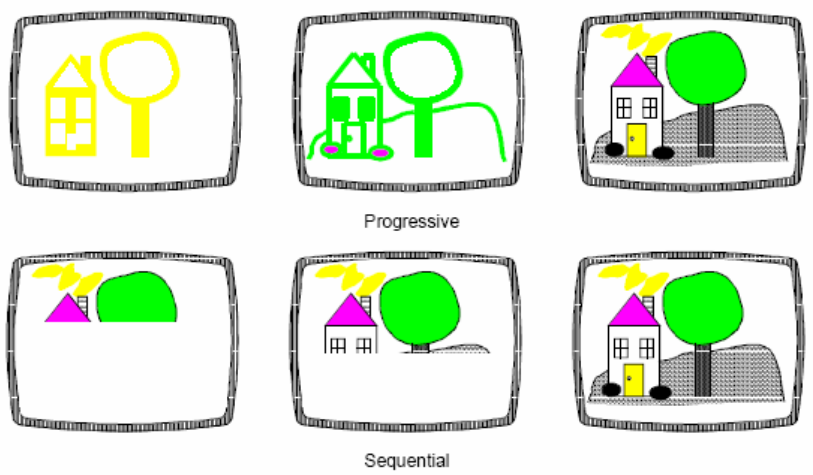


Figure 2.1: Sequential versus progressive

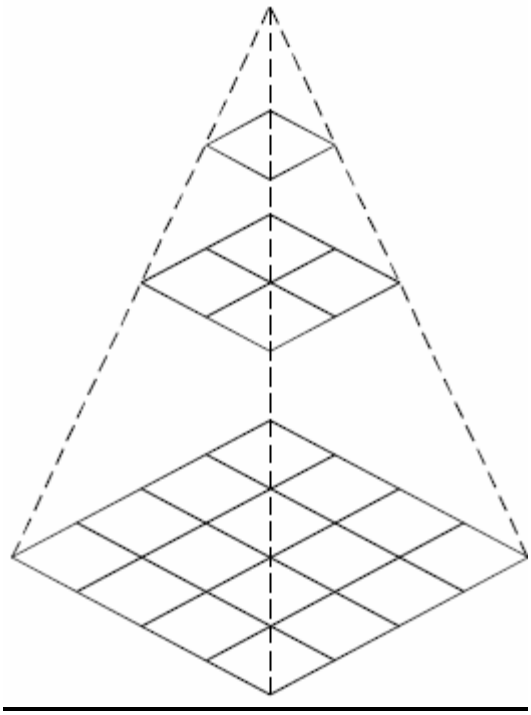


Figure 2.2: Hierarchical presentation

2.3 Baseline Sequential encoding Process

For the sequential DCT-based mode, 8x8 sample blocks are typically input block by block from left to right and block-row by block-row from top to bottom. After a block has been transformed by the forward DCT, quantized and prepared for entropy encoding (zig-zag), all 64 of its quantized DCT coefficients can be immediately entropy encoded and

output as part of the compressed image data, thereby minimizing coefficient storage requirements.
The compression scheme is divided into the following stages:

1. Transform the image into an optimal color space (not part of the standard).
2. Downsample chrominance components (not part of the standard).
3. Apply a Discrete Cosine Transform (DCT) to blocks of pixels, thus removing redundant image data.
4. Quantize each block of DCT coefficients using weighting functions optimized for the human eye.
5. Encode the resulting coefficients (image data) using a Huffman variable word-length algorithm to remove redundancies in the coefficients.

These stages are explained in detail in the following sections.

2.3.1 Transform the image

The JPEG algorithm is capable of encoding images that use any type of color space. JPEG itself encodes each component in a color model separately, and it is completely independent of any color-space model, such as RGB, HSI, or CMY. The best compression ratios result if a luminance/chrominance color space, such as YCbCr is used. Most of the visual information to which human eyes are most sensitive is found in the high-frequency, gray-scale, luminance component (Y) of the YCbCr color space. The other two chrominance components (Cb and Cr) contain high-frequency color information to which the human eye is less sensitive. Most of this information can therefore be discarded.

In comparison, the RGB, HSI, and CMY color models spread their useful visual image information evenly across each of their three color components, making the selective discarding of information very difficult. All three color components would need to be encoded at the highest quality, resulting in a poorer compression ratio. Gray-scale images do not have a color space as such and therefore do not require transforming.

2.3.2 Downsample chrominance components

When the uncompressed data is supplied in a conventional format (equal resolution for all channels), a JPEG compressor must reduce the resolution of the chrominance channels by *downsampling* or averaging together groups of pixels. The JPEG standard allows several different choices for the sampling ratios, or relative sizes, of the downsampled channels. The luminance channel is always left at full resolution (1:1 sampling). Typically both chrominance channels are downsampled 2:1 horizontally and either 1:1 or 2:1 vertically, meaning that a chrominance pixel covers the same area as either a 2x1 or a 2x2 block of luminance pixels. JPEG refers to these downsampling processes as 2h1v and 2h2v sampling, respectively.

Another notation commonly used is 4:2:2 sampling for 2h1v and 4:2:0 sampling for 2h2v. In 4:2:2 sampling the two Chrominance pictures (Cb,Cr) possess only half the “resolution” in the horizontal direction and full resolution in the vertical direction compared to the luminance (Y) picture. In 4:2:0 the two chrominace pictures (Cb, Cr) possess only half the "resolution" in both the horizontal and vertical direction as the luminance picture (Y). This notation derives from television customs (color

transformation and downsampling have been in use since the beginning of color TV transmission). 2h1v sampling is fairly common because it corresponds to National Television Standards Committee (NTSC) standard TV practice, but it offers less compression than 2h2v sampling, with hardly any gain in perceived quality.
Downsampling the chrominance components is not part of the standard.

2.3.3 Apply a Discrete Cosine Transform

The image data is divided up into 8x8 blocks of pixels, called data units. (From this point on, each color component is processed independently, so a "pixel" means a single value, even in a color image.) A DCT is applied to each 8x8 block. DCT converts the spatial image representation into a frequency map: the low-order or "DC" term represents the average value in the block, while successive higher-order ("AC") terms represent the strength of more and more rapid changes across the width or height of the block. The highest AC term represents the strength of a cosine wave alternating from maximum to minimum at adjacent pixels.

The DCT calculation is fairly complex; in fact, this is the most costly step in JPEG compression. The point of doing it is that all power is compacted in as few coefficients as possible and the high- and low-frequency information present in the image is separated out. We can discard high-frequency data easily without losing low-frequency information. The DCT step itself is lossless except for roundoff errors.

The NxN two-dimensional DCT is defined as:

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

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

eq.1: Forward 2-D DCT

The inverse DCT is defined as:

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

eq. 2: Inverse 2-D DCT

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

2.3.4 Quantize each block

To discard an appropriate amount of information, the compressor divides each DCT output value by a "quantization coefficient" and rounds the result to an integer (eq. 3). The larger the quantization coefficient, the more data is lost, because the actual DCT value is represented less and less accurately. Each of the 64 positions of the DCT output block has its own quantization coefficient, with the higher-frequency terms being quantized more heavily than the low-order terms. Furthermore, separate quantization tables are employed for luminance and chrominance data, with the chrominance data being quantized more heavily than the luminance data. This allows JPEG to exploit further the eye's differing sensitivity to luminance and chrominance.

It is this step that is controlled by the "quality" setting of most JPEG compressors. The compressor starts from a built-in table that is appropriate for a medium-quality setting and increases or decreases the value of each table entry in inverse proportion to the requested quality. The complete quantization tables actually used are recorded in the compressed file so that the decompressor will know how to reconstruct the DCT coefficients.

Selection of an appropriate quantization table is something of a black art. Most existing compressors start from a sample table developed by the ISO JPEG committee.

$$QC_{ij} = Round(\frac{C_{ij}}{W_{ij}})$$

eq. 3: JPEG quantization

In Part-3 of the JPEG specification the variable quantization enhancement is defined for the DCT-based processes (see section 2.5).

2.3.5 Encode the resulting coefficients

After quantization, the DC coefficient is treated separately from the AC coefficients. The DC coefficient is a measure of the average value of the 64 image samples. Because there is usually strong correlation between the DC coefficients of adjacent 8x8 blocks, the quantized DC coefficient is encoded as the difference from the DC term of the previous block in encoding order (differential coding). Finally, all of the quantized coefficients are ordered into the “zig-zag” sequence shown in fig 2.3. This ordering helps to facilitate entropy coding by placing low-frequency coefficients (which are more likely to be non zero) before high-frequency coefficients. Huffman coding is used for entropy coding.

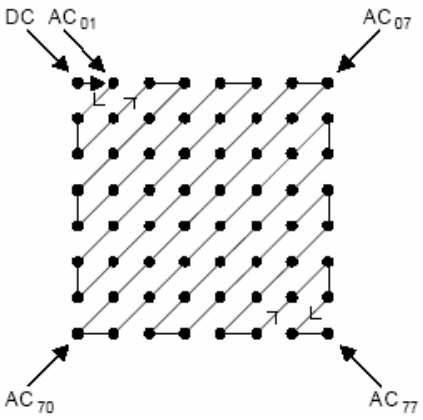


Fig. 2.3: zig-zag order

At this point, the JPEG data stream is ready to be transmitted across a communications channel or encapsulated inside an image file format.

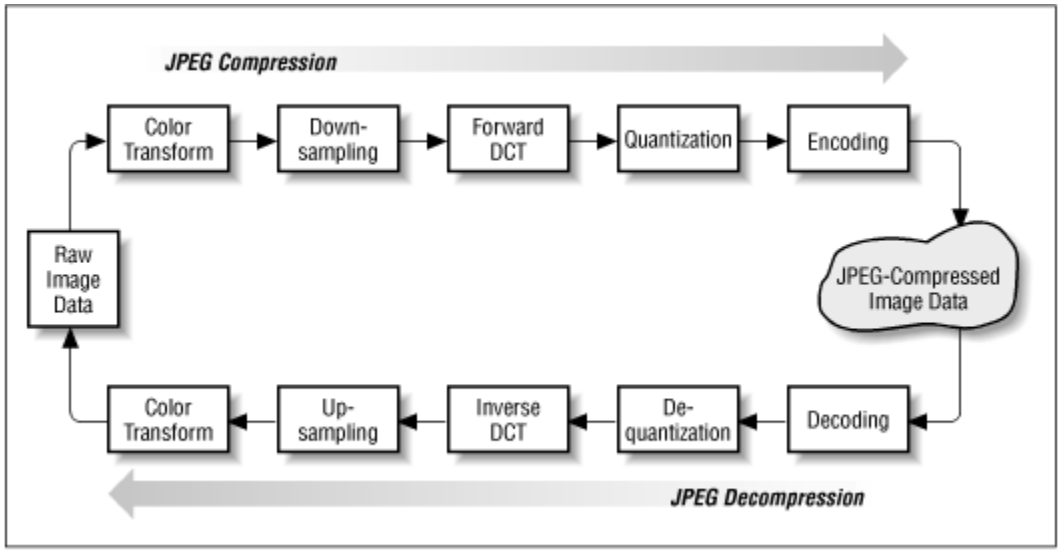


Fig. 2.4: JPEG diagram

2.4 JPEG Syntax

2.4.1 Structure of compressed data

Compressed image data are described by a uniform structure and set of *parameters* for both classes of encoding processes (lossy or lossless), and for all modes of operation (sequential, progressive, lossless, and hierarchical). The various parts of the compressed image data are identified by special two-byte codes called *markers*.

There are three compressed data formats:

a) The interchange format: In addition to certain required marker segments and the entropy-coded segments, the interchange format shall include the marker segments for

all quantization and entropy-coding table specifications needed by the decoding process. This guarantees that a compressed image can cross the boundary between application environments, regardless of how each environment internally associates tables with compressed image data.

b) The *abbreviated format* for compressed image data: The abbreviated format for compressed image data is identical to the interchange format, except that it does not include all tables required for decoding. (It may include some of them.) This format is intended for use within applications where alternative mechanisms are available for supplying some or all of the table-specification data needed for decoding.

c) The abbreviated format for table-specification data: This format contains only table-specification data. It is a means by which the application may install in the decoder the tables required to subsequently reconstruct one or more images.

The interchange format is the one most widely used and is described, in the context of the sequential coding process, in the following section.

2.4.2 Image, frame, and scan

Compressed image data consists of only one image. An image contains only one frame in the cases of sequential and progressive coding processes; an image contains multiple frames for the hierarchical mode. A frame contains one or more scans. Finally, a scan contains a complete encoding of one (non-interleaved) or more image components (interleaved).

Related to the concepts of multiple-component interleave is the *minimum coded unit* (MCU). If the compressed image data is non-interleaved, the MCU is defined to be one data unit (8x8 block for DCT-based processes). If the compressed data is interleaved, the MCU contains one or more data units from each component.

For example consider an image in Y, Cb, Cr format with 4:2:0 sampling as seen in Fig. 2.5. This image has three components. If the compressed data is interleaved, there would be only one scan, and the first and second MCU would be:

$$MCU_1 = Y_{00}Y_{01}Y_{10}Y_{11}Cb_{00}Cr_{00}$$
$$MCU_2 = Y_{02}Y_{03}Y_{12}Y_{13}Cb_{01}Cr_{01}$$

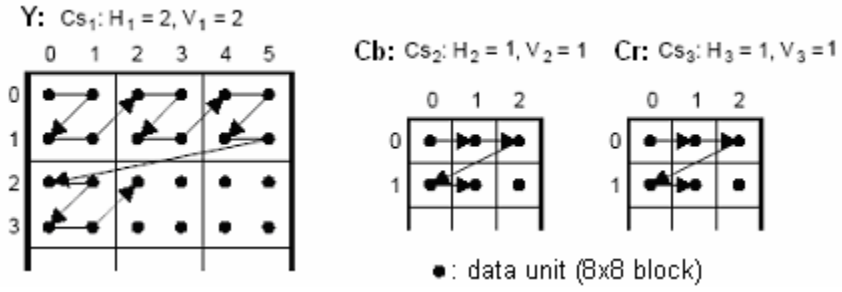


Figure 2.5: Interleaved Components

If the compressed data is non-interleaved, there would be three scans. The first scan would contain the coded luminance (Y), the second the coded Cb component and the last the Cr. For each scan the MCU is a data unit.

The MCU's in either case (interleaved, non-interleaved) are encode in raster scan order as seen in fig. 2.6.

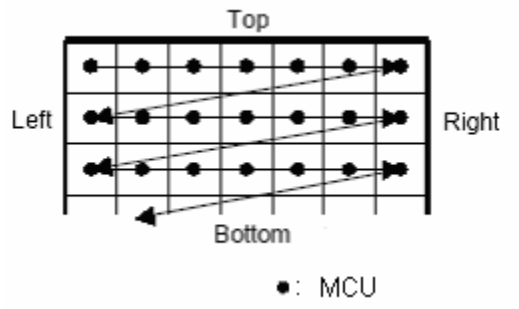


Figure 2.6: Coding order

2.4.3 Interchange Format

This interchange format syntax seen below applies to all coding processes for sequential DCT-based, progressive DCT-based, and lossless modes of operation.

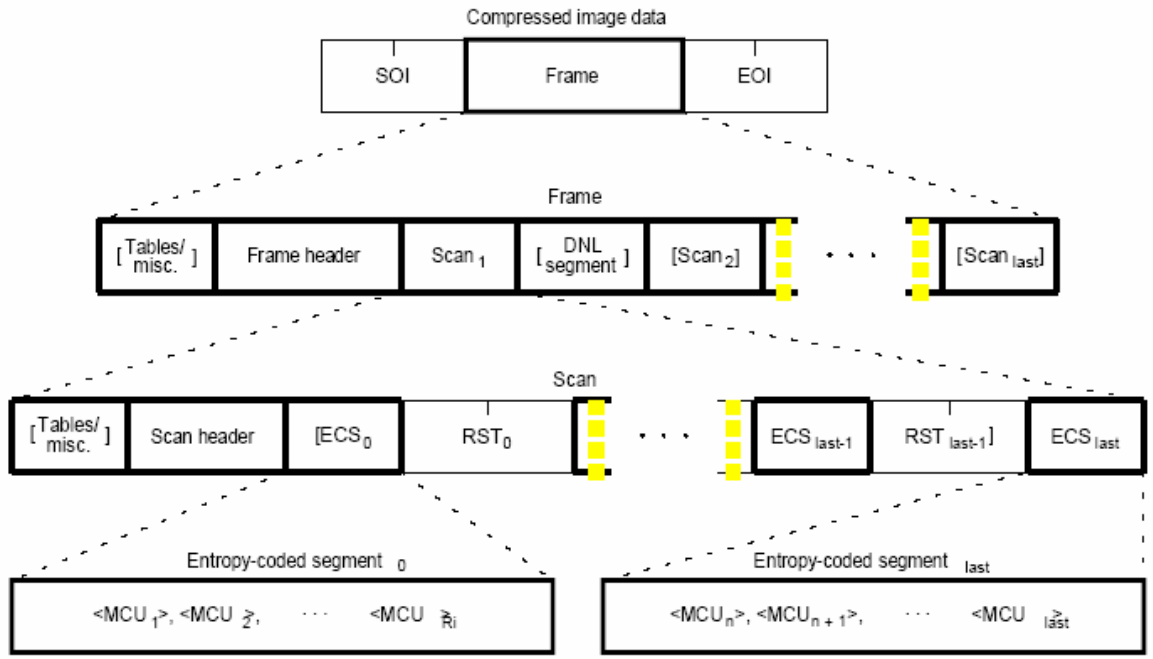


Fig. 2.7: JPEG High Level Syntax

The markers shown in Figure 2.7 are defined as follows:

SOI: Start of image marker – Marks the start of a compressed image represented in the interchange format or abbreviated format.

EOI: End of image marker – Marks the end of a compressed image represented in the interchange format or abbreviated format.

RSTm: Restart marker – A conditional marker which is placed between entropy-coded segments only if restart is enabled. There are 8 unique restart markers (m = 0 - 7) which repeat in sequence from 0 to 7, starting with zero for each scan, to provide a modulo 8 restart interval count. The encoder outputs the restart markers, intermixed with the entropy-coded data, at regular *restart intervals* of the source image data. Restart markers can be identified without having to decode the compressed data to find them. Because they can be independently decoded, they have application-specific uses, such as parallel encoding or decoding, isolation of data corruptions, and semi-random access of entropy-coded segments.

The top level of figure 2.7 specifies that the non-hierarchical interchange format shall begin with an SOI marker, shall contain one frame, and shall end with an EOI marker.

The second level of figure 2.7 specifies that a frame shall begin with a frame header and shall contain one or more scans. A frame header may be preceded by one or more table-specification or miscellaneous marker segments as described in the next section. In the table specification header are defined the quantization matrices. Up to 4 matrices may be specified, each used to quantize one component. For sequential DCT-based and lossless processes each scan shall contain from one to four image components. If two to four components are contained within a scan, they shall be interleaved within the scan.

The third level of figure 2.7 specifies that a scan shall begin with a scan header and shall contain one or more entropy coded data segments. Each scan header may be preceded by one or more table-specification or miscellaneous marker segments. If restart is not enabled, there shall be only one entropy-coded segment and no restart markers shall be present. If restart is enabled, the number of entropy-coded segments is defined by the size of the image and the defined restart interval. In this case, a restart marker shall follow each entropy-coded segment except the last one.

The fourth level of Figure 2.7 specifies that each entropy-coded segment is comprised of a sequence of entropy coded MCUs. If restart is enabled and the restart interval is defined to be Ri, each entropy-coded segment except the last one shall contain Ri MCUs. The last one shall contain whatever number of MCUs completes the scan.

The required table-specification data **must** be present at one or more of the allowed locations.

2.4.3.1 Frame header

The frame header which shall be present at the start of a frame specifies the source image characteristics (sample precision, dimensions), the components in the frame, and the sampling factors for each component, and specifies the destinations (see table specification syntax) from which the quantized tables to be used with each component are retrieved.

2.4.3.2 Scan header

The scan header which shall be present at the start of a scan specifies which component(s) are contained in the scan, specifies the destinations from which the entropy tables to be used with each component are retrieved.

If there is only one image component present in a scan, that component is, by definition, non-interleaved. If there is more than one image component present in a scan, the components present are, by definition, interleaved.

2.4.3.3 Table-specification and miscellaneous marker segment syntax

At the places indicated in Figure 2.7, any of the table-specification segments or miscellaneous marker segments specified may be present in any order and with no limit on the number of segments.

If any table specification for a particular destination occurs in the compressed image data, it shall replace any previous table specified for this destination, and shall be used whenever this destination is specified in the remaining scans in the frame or subsequent images represented in the abbreviated format for compressed image data. If a table specification for a given destination occurs more than once in the compressed image data, each specification shall replace the previous specification.

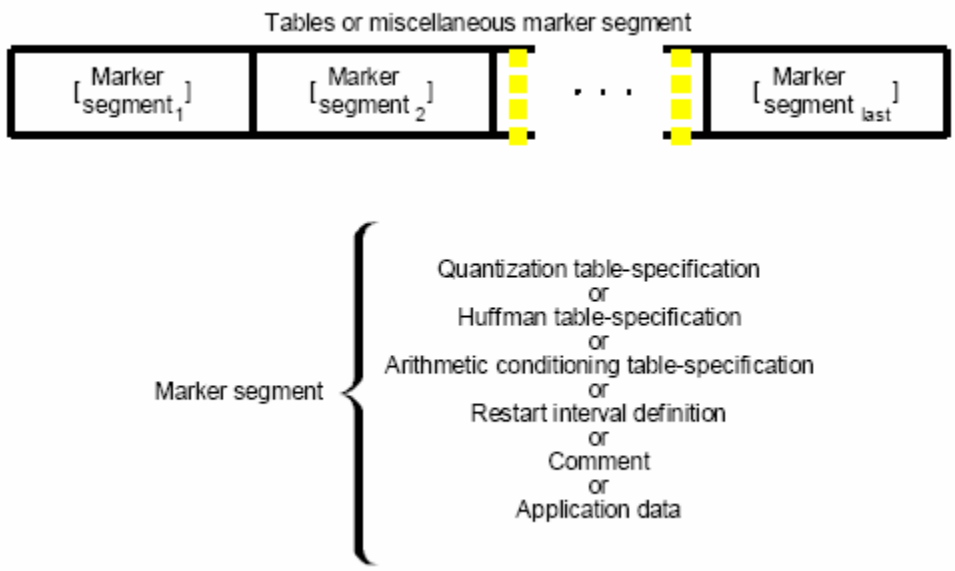


Fig. 2.8: Table Specification or misc. Marker Segment

The Huffman table-specification segment defines a Huffman table to be used for entropy coding.

The Arithmetic conditioning table-specification replaces the default arithmetic coding conditioning tables established by the SOI marker for arithmetic coding processes.

The Restart interval definition defines the restart interval.

The Comment segment allows for user comments.

The Application data are reserved for application use. Since these segments may be defined differently for different applications, they should be removed when the data are exchanged between application environments.

The quantization table-specification segment defines the quantization matrices and is described analytically in the next section.

Quantization table-specification syntax

Figure 2.9 specifies the marker segment which defines one or more quantization tables.

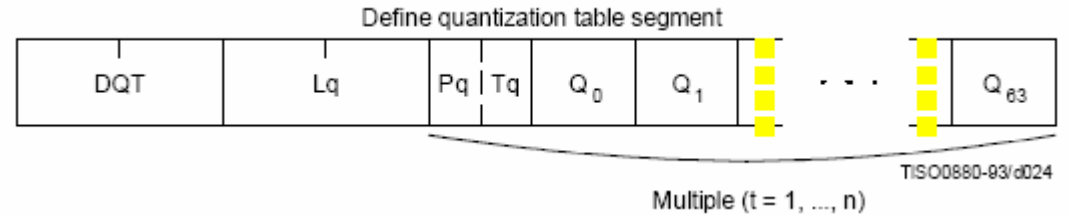


Fig. 2.9: Quantization table Specification

The marker and parameters shown in Figure 2.9 are defined below.

DQT: Define quantization table marker – Marks the beginning of quantization table-specification parameters.

Lq: Quantization table definition length – Specifies the length of all quantization table parameters shown in Figure 2.9.

Pq: Quantization table element precision – Specifies the precision of the Qk values. Value 0 indicates 8-bit Qk values; value 1 indicates 16-bit Qk values. Pq shall be zero for 8 bit sample precision P.

Tq: Quantization table destination identifier – Specifies one of four possible destinations at the decoder into which the quantization table shall be installed.

Qk: Quantization table element – Specifies the *k*th element out of 64 elements, where *k* is the index in the zigzag ordering of the DCT coefficients. The quantization elements shall be specified in zig-zag scan order.

The value n in figure 2.9 is the number of quantization tables specified in the DQT marker segment.

Once a quantization table has been defined for a particular destination, it replaces the previous tables stored in that destination and shall be used, when referenced, in the remaining scans of the current image and in subsequent images represented in the abbreviated format for compressed image data. If a table has never been defined for a particular destination, then when this destination is specified in a frame header, the results are unpredictable.

An 8-bit DCT-based process shall not use a 16-bit precision quantization table.

Application Data

Figure 2.10 specifies the marker segment structure for an application data segment

Fig. 2.10: Application Data Segment

APPn: Application data marker – Marks the beginning of an application data segment.

Lp: Application data segment length – Specifies the length of the application data segment.
Api: Application data byte – The interpretation is left to the application.

The APPn (Application) segments are reserved for application use. Since these segments may be defined differently for different applications, they should be removed when the data are exchanged between application environments.

2.5 Variable quantization (extension of JPEG-Part 3)

Variable quantization is an extension of JPEG defined in Part 3 of the specification. It is an enhancement available to the quantization procedure of DCT-based processes. This enhancement may be used with any of the DCT-based processes defined by JPEG**.**
While the quantization table W remains fixed throughout the image, the user may specify a multiplier QS for each MCU k. The quantized coefficient is then:

$$QC_{ijk} = Round(\frac{C_{ijk}}{W_{ij}QS_k})$$

Eq. 4: Variable Quantization

This scaling applies only to the AC coefficients; the DC quantization remains unchanged. The user initially chooses one of two prespecified tables (linear or non-linear) of 31 possible multiplier values. A 5 bit index identifies a particular table entry, and a single bit code is used to signal MCUs in which the multiplier changes. Thus there is a 6 bit cost for each change in multiplier. This is very much the same scheme as the one used in adaptive MPEG coding.

3. MPEG-2 Video Compression Standard

MPEG-2 [19] is an extension of the MPEG-1 international standard for digital compression of audio and video signals. MPEG-1 was designed to code progressively scanned video at bit rates up to about 1.5 Mbit/s. MPEG-2 is directed at broadcast formats at higher data rates; it provides extra algorithmic 'tools' for efficiently coding interlaced video, supports a wide range of bit rates and provides for multi-channel surround sound coding.

MPEG-2 volume consists of a total of 9 parts under ISO/IEC 13818. The full title is: "Information Technology--Generic Coding of Moving Pictures and Associated Audio." ISO/IEC 13818:

Part 1 Systems: Addresses the combining of one or more elementary streams of video and audio, as well as, other data into single or multiple streams which are suitable for storage or transmission. This is specified in two forms: the Program Stream and the Transport Stream. Each is optimised for a different set of applications.

Part 2 Video: describes syntax (header and bitstream elements) and semantics (algorithms telling what to do with the bits). Video breaks the image sequence into a series of nested layers, each containing a finer granularity of sample clusters (sequence, picture, slice, macroblock, block, sample/coefficient). At each layer, algorithms are made available which can be used in combination to achieve efficient compression. The syntax also provides a number of different means for assisting decoders in synchronization, random access, buffer regulation, and error recovery. The highest layer, sequence, defines the frame rate and picture pixel dimensions for the encoded image sequence. Part 2 of MPEG-2 builds on the powerful video compression capabilities of the MPEG-1 standard to offer a wide range of coding tools. These have been grouped in profiles to offer different functionalities.

Part 3 Audio: It is a backwards-compatible multichannel extension of the MPEG-1 Audio standard.

Part 4 Conformance: defines the meaning of MPEG-2 conformance for all three parts (Systems, Video, and Audio), and provides two sets of test guidelines for determining compliance in bitstreams and decoders.

Part 5 Software Simulation: Contains an example ANSI C language software encoder and compliant decoder for video and audio.

Part 6 Digital Storage Medium Command and Control (DSM-CC): provides a syntax for controlling VCR-style playback and random-access of bitstreams encoded onto digital storage mediums such as compact disc. Playback commands include Still frame, Fast Forward, Advance, Goto.

Part 7 Non-Backwards Compatible Audio (NBC): addresses the need for a new syntax to efficiently de-correlate discrete mutlichannel surround sound audio. MPEG-2 audio

attempts to code the surround channels as an ancillary data to the MPEG-1 backwards-compatible Left and Right channels. This allows existing MPEG-1 decoders to parse and decode only the two primary channels while ignoring the side channels. This is analogous to the Base Layer concept in MPEG-2 Scalable video ("decode the base layer, and hope the enhancement layer will be a fad that goes away.").

Part 8 10-bit video extension: Introduced in late 1994, this extension to the video part (13818-2) describes the syntax and semantics for coded representation of video with 10-bits of sample precision. The primary application is studio video (distribution, editing, archiving). Methods have been investigated by Kodak and Tektronix which employ Spatial scalablity, where the 8-bit signal becomes the Base Layer, and the 2-bit differential signal is coded as an Enhancement Layer. Part 8 has been withdrawn due to lack of interest by industry.

Part 9 Real-time Interface (RTI): defines a syntax for video on demand control signals between set-top boxes and head-end servers.

3.1 Video Compression in MPEG-2

The video image is separated into one luminance (Y) and two chrominance channels (also called color difference signals Cb and Cr). It is also divided into "macroblocks" (16x16 pixels), which are the basic unit of coding within a picture. Each macroblock is divided into four 8x8 luminance blocks. The number of 8x8 chrominance blocks per macroblock depends on the chrominance format of the source image. For example, in the common 4:2:0 format, there is one chrominance block per macroblock for each of the channels, making a total of six blocks per macroblock. The encoding order is the same as in JPEG. Macroblocks are encoded from left to right and from top to bottom (raster scanning).

The compression techniques are divided into two basic classes, intra (I, P, B-pictures) and non-intra (P-, B- pictures). Intra techniques compress a picture using information only from that picture; non-intra techniques also use information from one or two other pictures displaced in time.

For I-pictures (intra coding), the coding models are similar to those defined by JPEG. Each block is treated with an 8x8 discrete cosine transform. Since the variations in a block tend to be low, the transformation results in a more compact representation of the block. The energy of the block is packed in the lower frequencies and the DCT coefficients within the block are almost completely decorrelated.

The resulting DCT coefficients are then quantized. The quantizer step-size for each DCT coefficient is specified via a quantization matrix and a quantization scale factor. The quantization matrix takes advantage of the Human Visual Response and codes with less acuity higher spatial frequencies (perceptual coding). The quantizer scale factor provides spatial masking and adaptation to local picture characteristics.

Then, the quantized coefficients (except for the DC- coefficient) are re-ordered to maximize the probability of long runs of zeros, and run-length coded. Because of the correlation across wide areas of the picture, especially for the block averages (represented

by the DC coefficients) the DC coefficient is coded separately from the AC. The difference of the quantized DC coefficient in the 8x8 DCT block from its neighbouring block is coded, instead of the quantized DC coefficient itself. Finally, entropy coding is used.

For P-, B-pictures, macroblocks may be intra coded or non-intra coded. Intra coded macroblocks are handled as I-picture macroblocks.

Non-intra coded macroblocks are fed to both the subtractor and the motion estimator. The motion estimator compares each of these new macroblocks with macroblocks in a previously stored reference picture or pictures. It finds the macroblock in the reference picture that most closely matches the new macroblock. The motion estimator then calculates a motion vector (mv) which represents the horizontal and vertical displacement from the macroblock being encoded to the matching macroblock-sized area in the reference picture. Note that the motion vectors can have 1/2 pixel resolution achieved by linear interpolation between adjacent pixels.

The motion estimator also reads this matching macroblock (known as a predicted macroblock) out of the reference picture memory and sends it to the subtractor which subtracts it, on a pixel by pixel basis, from the new macroblock entering the encoder. This forms an error prediction or residual signal that represents the difference between the predicted macroblock and the actual macroblock being encoded. This residual is often very small. Then perceptual coding of the macroblock temporal prediction error is performed just like in the case of intra coded blocks (adaptive quantization and quantization of DCT transform coefficients but with different quantization matrix; the DC coefficient is treated like the AC coefficients). If the quantized block prediction error is equal to 0 then "No prediction error" for the block may be signalled and not code the prediction error for this block.

Moreover, macroblocks in P pictures with limited motion activity - which means that the motion vector is zero for both the horizontal and vertical vector components, and no quantized prediction error for the current macroblock is present- may be skipped. Skipped macroblocks are the most desirable element in the bitstream since they consume no bits, except for a slight increase in the bits of the next non-skipped macroblock.
From the above, we could divide the MPEG-2 encoding process to the following steps:

- Motion Compensation
  - Motion Estimation
  - Prediction
- DCT Transformation
- Quantization
- Stream Generation
- Inverse Quantization
- Inverse DCT Transformation
- Prediction Addition

The final three steps, are the reconstruction of the last encoded anchor frame that will be used to predict the next frame (if it is not an I frame).

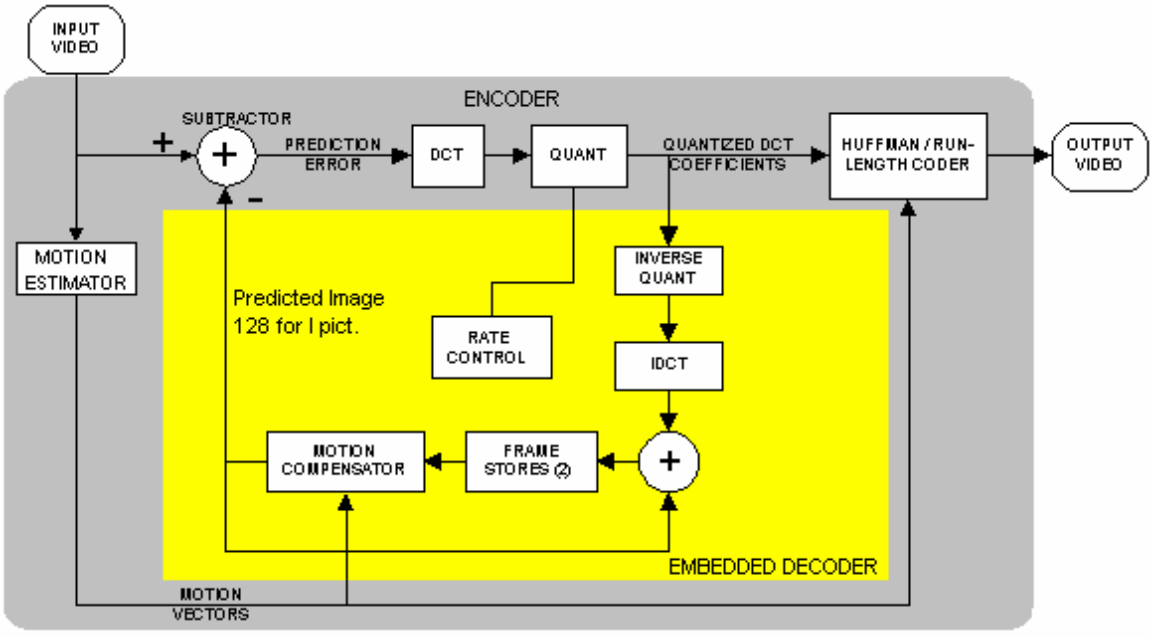


Figure 3.1: MPEG-2 Encoder Block Diagram

MPEG-2 takes advantage of:

1. **Spatial and temporal redundancy:** Pixel values are not independent, but are correlated with their neighbours both within the same frame and across frames. So, to some extent, the value of a pixel is predictable given the values of neighbouring pixels. Mpeg takes advantage of this redundancy by using DCT-transformation, Motion Compensation and Differential encoding of intra DC-coefficients.
2. **Psychovisual redundancy:** The human eye has a limited response to fine spatial detail and is less sensitive to detail near object edges or around shot-changes. Consequently, controlled impairments introduced into the decoded picture by the bit rate reduction process should not be visible to a human observer. Mpeg takes advantage of this redundancy by using the quantization matrix to quantize all macroblocks and adaptive quantization of each Macroblock according to its type (smooth, textured).

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Default Intra quantizer matrix

16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16

Default Non-intra quantizer matrix

Figure 3.2: MPEG-2 Quantization Matrices

There are 2 different available ranges for the quantizer scale factor QS that the encoder can choose from (Both ranges have 31 possible values). One is a linear range from 2 to 62. The other is a nonlinear range from 1 to 112. The chosen range is indicated by the quantizer_scale_type bit in the bitstream and the actual value of the quantizer scale factor QS is determined/indexed in the bit stream by the quantizer_ scale_code QP (fig. 3.3).

quantiser_scale_type='0' decimal (QS)	quantiser_scale_type='1' decimal (QS)	quantiser_scale_code(QP) decimal
forbidden	forbidden	0
2	1	1
4	2	2
6	3	3
8	4	4
10	5	5
12	6	6
14	7	7
16	8	8
18	10	9
20	12	10
22	14	11
24	16	12
26	18	13
28	20	14
30	22	15
32	24	16
34	28	17
36	32	18
38	36	19
40	40	20
42	44	21
44	48	22
46	52	23
48	56	24
50	64	25
52	72	26
54	80	27
56	88	28
58	96	29
60	104	30
62	112	31

Figure 3.3: Linear & non-Linear QS (=mquant)

3.2 Profiles & Levels

MPEG-2 aims to be a *generic* video coding system supporting a diverse range of applications. Different algorithmic 'tools', developed for many applications, have been integrated into the full standard. To implement all the features of the standard in all decoders is unnecessarily complex and a waste of bandwidth, so a small number of subsets of the complete MPEG-2 tool kit have been defined, known as profiles and levels. A profile is a subset of algorithmic tools (profiles limit syntax: MPEG video syntax provides an efficient way to represent image sequences in the form of more compact coded data. The language of the coded bits is the "syntax") and a level identifies a set of constraints on parameter values (such as picture size or bit rate). The profiles and levels fit together such that a higher profile or level is superset of a lower one. A decoder which

supports a particular profile and level is only required to support the corresponding subset of algorithmic tools and set of parameter constraints.

In a first time all the defined combinations of profiles and levels were structured in a hierarchical way. In this case a simpler profile happens to be a subset of every more complex profile and every parameter value of a lower level happens to be lower or equal to the corresponding value of every higher level. The consequence is the forward compatibility between different profiles and levels. The coded bitstreams must contain a ***profile_and_level_indication*** of the simplest decoder capable of successfully decoding the bitstream, but also every decoder more complex can decode correctly the same bistream. These are the hierarchical profiles described below

3.2.1 Hierarchical profiles

Details of non-scalable profiles

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

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

The *main profile* adds support for B-pictures and is the most widely used profile. Using B-pictures increases the picture quality, but adds about 120 ms to the coding delay to allow for the picture reordering. Main profile decoders will also decode MPEG-1 video. Coding is performed on a 4:2:0 sampled video signal.

Details of scalable profiles

The *SNR profile* adds support for enhancement layers of DCT coefficient refinement, using the 'signal to noise (SNR) ratio scalability' tool. The codec operates in a similar manner to the non-scalable codec, with the addition of an extra quantization stage. The coder quantizes the DCT coefficients to a given accuracy defined by the user; variable-length codes them and transmits them as the lower-level or 'base-layer' bitstream. The quantization error introduced by the first quantizer is itself quantized, variable-length coded and transmitted as the upper-level or 'enhancement-layer' bitstream. Side information required by the decoder, such as motion vectors, is transmitted only in the base layer. The base-layer bitstream can be decoded in the same way as the non-scalable case. To decode the combined base and enhancement layers, both layers must be received. The enhancement-layer coefficient refinements are added to the base-layer coefficient values following inverse quantisation. The resulting coefficients are then decoded in the same way as the non-scalable case. The SNR profile is suggested for digital terrestrial television as a way of providing graceful degradation.

The *spatial profile* adds support for enhancement layers carrying the coded image at different resolutions, using the 'spatial scalability' tool. Spatial scalability is characterised by the use of decoded pictures from a lower layer as a prediction in a higher layer. If the higher layer is carrying the image at a higher resolution, then the decoded pictures from the lower layer must be sample rate converted to the higher resolution by means of an 'up-converter'. As with SNR scalability, the lower-layer bitstream can be decoded in the same way as the non-scalable case. To decode the combined lower and upper layers, both layers must be received. The lower layer is decoded first and the 'up-converted' decoded pictures offered to the upper-layer decoder for possible use as a prediction. The upper-layer decoder selects between its own motion-compensated prediction and the 'up-converted' prediction from the lower layer, using a value for the weighting function, W, transmitted in the upper-layer bitstream. The spatial profile is suggested as a way to broadcast a high-definition TV service with a main-profile compatible standard-definition service.

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

3.2.2 Non-hierarchical profiles

In this case the profile, that is always a subset of the specification, has no relationship to other profiles in the sense that you cannot say if it is simpler than one or more complex than another. Also the levels of a non-hierarchical profile don't have necessarily any relationship to similarly named levels of other profiles. In this context compatibility, where it exists, is not a consequence of the hierarchy of profiles and levels, but exactly a choice of the definition of such a profile@level.

The 4:2:2@ML Profile aims to being suitable for the TV production environment applications. Actually it is also called, incorrectly, "Professional Profile" or "Studio Profile". It is based on MainProfile@MainLevel, of which it is intended to overcome the limitations concerning the professional use. The main removed limitation on the profile side is the *chroma_format*, **4:2:2**, that gives the name to the profile, is possible and it should allow chroma key effects for the decoded images. Besides also the Intra DC precision may be improved (till 11 bits). On the level side the most important increased parameter value is *Maximum Bit Rate*, **50 Mbps**, that allows a good picture quality even with only *Intra_pictures*, so that applications that used Motion-JPEG compression (non-standard) can now use an MPEG-2 standard profile.

Multiview Profile is a new Profile that, using the MPEG-2 toolkit, will be able to manage stereo images.

3.2.3 Details of levels

MPEG-2 defines four levels of coding parameter constraints. Table 2 shows the constraints on picture size, frame rate, bit rate and buffer size for each of the defined levels. Note that the constraints are upper limits and that the codecs may be operated below these limits.

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

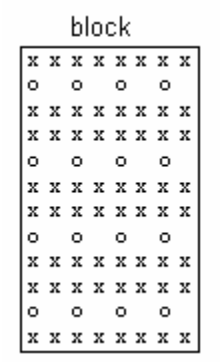
Figure 3.4: MPEG-2 Levels

	Simple	Main	SNR scalable	Spatial scalable	High	Multiview	4:2:2@ML
Low level		X	X				
Main level	X	X	X		X	X	
High-1440 level		X		X	X	X	
High level		X		X	X		

Figure 3.5: Profiles & Levels combinations

Only the combinations marked with an "X" are recognised by the standard.

We are interested in the Main Profile at the Main Level (abbreviated as MP@ML). In this format a macroblock consists of 4 Y blocks, 1 Cr block and 1 Cb block. The two chrominace pictures (Cb, Cr) possess only half the "resolution" in both the horizontal and vertical direction as the luminance picture (Y) and are aligned as seen in figure 3.6 (for progressive video).



x: Y sample
o: Both a Cr and a Cb samples

Figure 3.6: Chrominance Sampling

3.3 MPEG-2 Main Profile Syntax

The MPEG-encoded video bitstream syntax has a layered structure, comprised of blocks (8 x 8-pixel arrays), macroblocks (4 of luminance blocks, plus 2 chrominance blocks), slices, pictures, groups of pictures (GOP's).

A typical GOP in display order is:

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

The corresponding bitstream order is:

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

A regular GOP structure can be described with two parameters: *N*, which is the number of pictures in the GOP, and *M*, which is the spacing of P-pictures. The GOP given here is described as *N*=12 and *M*=3. A sequence may consist of almost any pattern of I, P, and B pictures (there are a few minor semantic restrictions on their placement). It is common in industrial practice to have a fixed pattern (e.g. IBBPBBPBBPBBPBB), however, more advanced encoders will attempt to optimize the placement of the three picture types according to local sequence characteristics in the context of more global characteristics.

Every MPEG video sequence starts with a sequence header and ends with a sequence_end_code. If the sequence header is not followed immediately by an extension_start_code, the syntax is governed by the MPEG-1 constrained parameters. If the sequence heard is followed by the sequence extension, then the syntax is governed by the MPEG-2 rules. After the required sequence extension, optional extension data and user data fields are allowed. Group of picture headers and picture(s) repeat until the sequence_end_code occurs. Note that the sequence header and sequence extension may be repeated. The GOP header is always optional and at least one picture follows each GOP header.

A picture header is always followed by the picture coding extension, other extensions and user data (optional), and picture data. Picture data is made up of slices and each slice consists of a slice header and macroblock data. Each macroblock has a header, followed by data for the coded DCT blocks. For the 4:2:0 chroma format up to six blocks follow the header (some blocks might be skipped). Finally is the block. If it is an intra coded block, the differential DC coefficient is coded first. Then, the rest of the coefficients are coded as runs and levels until the end_of_block terminates the variable length codes for that block.

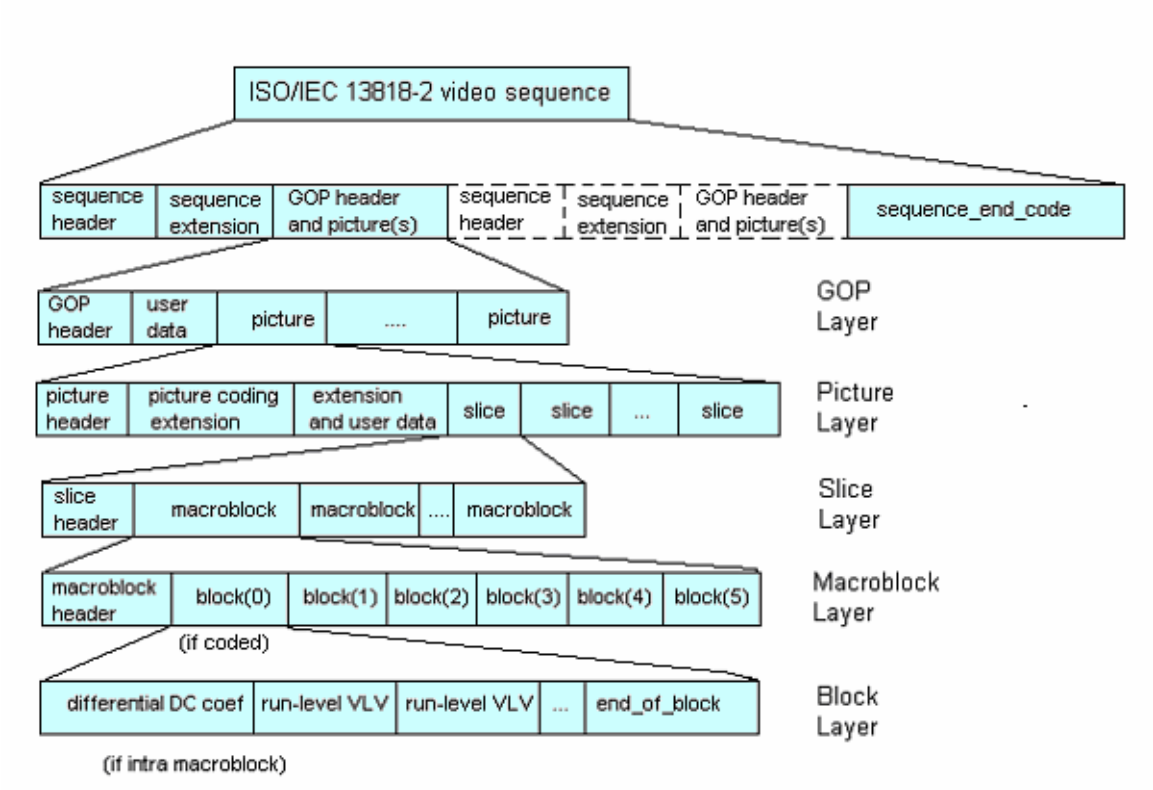


Figure 3.7: Main Profile Syntax

Sequence Layer: At least one picture must follow the sequence header. The first picture after a sequence header can be an I- or P-picture. The Sequence header at the sequence layer resets all the quantization matrices back to the default values. User defined quantization matrices can be downloaded in the sequence header (Fig. 3.7 and fig 3.8).

GOP layer: It is optional in MPEG-2.

Picture Layer: Among the various flags is the q_scale_type bit which determines how to interpret the quantiser_scale_code (linear or non-linear range). In this layer new quantization matrices (intra and non-intra) are allowed to be set (fig. 3.8).

Slice Layer: Each slice can only contain Macroblocks from the same row in sequential order. Here is defined the quantization_scale_code (quantization parameter QP) which applies for all macroblocks in the slice, unless another quantization_scale_code is explicitly defined in the Macroblock layer (fig. 3.8).
Macroblock Layer: If the macroblock_quant flag is 1 then the quantizer_scale_code is changed. In this layer are the motion vectors (fig. 3.8). Skipped macroblocks are identified by the field **"macroblock_address_increment"**. If this field has value larger than 1 then macroblocks are skipped. In this layer is also the field coded_block_pattern that indicates whether and which of the six blocks are coded.

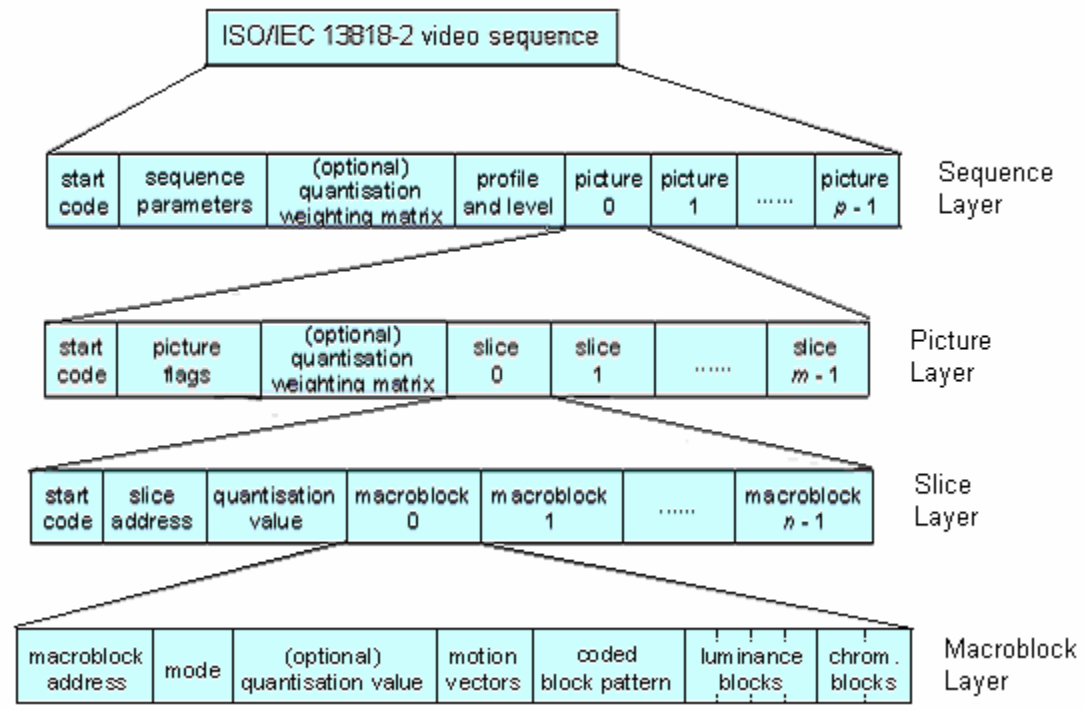


Figure 3.8: Main Profile Syntax

From Fig. 3.8 it is clear that there are two possibilities for adaptive quantization.

The first possibility is to adaptively modify the **quantization matrices** (intra and non intra quantization matrices). New matrices can be defined either in the sequence or the picture layer and are valid until new quantization matrices are defined. Each sequence header resets these matrices to their default values. If no matrices are defined, then the default matrices (Fig. 3.2) are used.

For more accurate control over the quantization process of MPEG-2 it is possible to choose the **quantization scale (QS)** in the macroblock layer. Each macroblock in a picture can have a different quantization scale value as described in 3.1.

The final actual quantization value is the product of the quantization scale QS and the appropriate value in the quantization matrix.

3.4 Rate Control

The purpose of rate control is to control the bitrate of the encoded video sequence in order to meet specific requirements, while improving the quality of the video [20].

The rate control algorithm controls all aspects of the encoding process such as the GOP structure, the bit allocation between I- , P-, B-pictures, the quantization scale for each macroblock, the macroblock type selection in non intra coded pictures etc.

The MPEG standards do not specify how to perform rate control. Various methods for rate control have been considered for different applications. In general the rate control algorithms could be classified into two major classes: Constant Bit Rate (CBR) control algorithms and Variable Bit Rate (VBR) algorithms. The purpose of CBR algorithms is to keep the variation of the bit rate from picture to picture as small as possible. VBR

algorithms regulate the compressed bitstream according to channel conditions and enhance video quality under various buffer and channel constraints.

The most important aspect of MPEG-2 that allows the regulation of the bit-rate of the compressed sequence as well as its quality (trade off relation) is adaptive quantization. The selection of the quantization scale (QS) for each macroblock in order to meet the bit rate requirements while maximizing the video quality has been studied thoroughly. Some of the approaches to solve this problem are described in the next paragraphs.

One method to get the optimal solution is to apply the rate-distortion theory (See annex A). The rate distortion curve for the set of quantizers that minimize a distortion metric (MSE usually) is found. To find the actual R-D curve and solve the bit-allocation problem for a set of arbitrary quantizers the Lagrange multiplier method or dynamic programming can be used. These methods deterministically ensure optimal bit allocation. However, the computational complexity of these methods is very high because they require the measuring of the R-D characteristics of current and feature frames before the bit allocation step. So these methods are inappropriate for use in applications requiring low end-to-end delay and low complexity; they are primarily applicable for nonreal-time and/or low resolution videoconferencing applications.

Another approach is to approximate the rate-distortion characteristic of the blocks to be quantized by using a R-D model. In [2] the blocks to be quantized are considered random Gaussian variables and the theoretical rate-distortion function for Gaussian source with squared error distortion is applied to build a rate quantization model.

In addition, the rate control in MPEG-4 is based on the following quadratic model:

$R = \frac{X_1 MAD}{QS} + \frac{X_2 MAD}{QS^2}$, where $X_{1,2}$ are the model parameters and MAD is the mean absolute difference of the current frame after motion compensation.

When using parametric models, the parameters can be acquired in two ways. Either by a pre-processing step, that gathers statistical information from the video sequence, or by estimating the parameters from previous encoded frames. The first solution is not suitable for real-time applications. The second method suffers in performance when large variations occur between frames because the coding parameters are determined entirely based upon the coding history.

In the following sections we present some aspects of some rate control algorithms proposed in the literature. More particularly, we present how the bit allocation among the pictures types is done and how they choose the quantization scale QS. Motion compensation and motion estimation is done as described in previous section.

In section 3.4.1 the rate control algorithm used in MPEG-2 is presented and in section 3.4.2 an adaptive model driven rate control algorithm, which we used in our M-JPEG implementations.

3.4.1 TM5 Rate Control

The MPEG standards do not specify how to perform rate control. However, to allow testing and experimentation using a common set of encoder routines, MPEG created a series of test models. Test model 5 outlines the rate control strategy for MPEG-2 [4].

To smooth out possibly large fluctuations in encoded bit rate from picture to picture, a buffer is typically used between the encoder and the channel. MPEG bit-allocation (rate control) regulates DCT quantization, so as to maintain constant quality from frame to frame in the decoded video while limiting variations in the encoded bit rate.

The MPEG-2 TM5 is a feedback rate control approach. This approach adjusts coding parameters to be used in the encoding process according to the currently available information such as the observable short-term buffer fullness, local scene activity, and average QP and bit amount generated from previous frame.

MPEG2 TM5 bit-allocation is accomplished in the context of the layered MPEG structure. First (at the GOP layer), a target bit-budget is calculated for each GOP. Within a GOP (at the picture level), bits are allocated between the different picture types (I, P, B) according to their relative coding efficiencies. Next, within a picture, *a reference quantization parameter (Q_{REF})* is set for each macroblock using a formula involving virtual buffer fullness and an empirical "reaction parameter." TM5 further modulates this Q_{REF} by the local variance of the video signal (for reasons related to HVS properties). In addition to these bit-allocation calculations, TM5 also monitors buffer fullness after encoding each macroblock in order to ensure that the target bit-rate for the encoded sequence is being met.

The algorithm works in three-steps:

1. Target bit allocation: In this step, the complexity of the current picture is estimated based upon the encoding of previous pictures to allocate a number of bits to code the picture.
2. Rate control: A reference quantization scale is determined using a virtual buffer in a feedback loop to regulate the coding rate so as to achieve the target bit allocation.
3. Adaptive quantization: The reference quantization scale is modulated according to the spatial activity in each macroblock to determine the actual quantization scale with which to code the macroblock.

A. Bit Allocation

Complexity estimation

The number of bits to be allocated to a picture depends upon its type: I,P or B. For each picture type, there is complexity model that attempts to relate the number of bits that would result from coding a picture of a given type to the quantization scale used. The complexity models are of the form:

$S_{pt} = \frac{X_{pt}}{Q_{pt}}$, where pt =I, P or B

where S_{pt} are the number of bits generated by encoding this picture, X_{pt} is a global complexity measure and Q_{pt} is the average quantization scale computed by averaging the actual quantization values QS used during the encoding of all the macroblocks of the previous frame of the same picture type, including the skipped macroblocks.

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

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

Initially the complexity values are set to:

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

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

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

bit_rate is measured in bits/s and is defined by the user.

Picture Target Setting

Bit allocation is performed with the goal that the average bit rate is achieved at the GOP layer. A corresponding number of bits is assigned to code each GOP. Bits are allocated to each picture in a GOP based upon the complexity models, the number of bits available to code the remaining pictures in the GOP and the number of remaining I, P and B pictures in the GOP.

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

$$T_I = \max\left[\frac{\mathrm{Re}\, m}{\left(1+\frac{N_P X_P}{X_I K_P}+\frac{N_B X_B}{K_B X_I}\right)}, \frac{bit\_rate}{8 * picture\_rate}\right]$$

$$T_P = \max\left[\frac{\mathrm{Re}\,m}{\left(N_P + \frac{N_B X_B K_P}{K_B X_P}\right)}, \frac{bit\_rate}{8 * picture\_rate}\right]$$

$$T_B = \max\left[\frac{\mathrm{Re}\,m}{\left(N_B + \frac{N_P X_P K_B}{K_P X_B}\right)}, \frac{bit\_rate}{8 * picture\_rate}\right]$$

Where:

Kp and Kb are "universal" constants dependent on the quantization matrices. These constants express the intention of the TM5 bit-allocation process to keep the ratio of quantization scales between different picture-coding types constant For the matrices specified in fig. 3.2 Kp = 1.0 and Kb = 1.4.

Rem is the remaining number of bits assigned to the GROUP OF PICTURES. Rem is updated as follows:

After encoding a picture, Rem = Rem - Spt

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

After encoding all pictures in a GOP, any difference between the target and actual bit allocation is carried over to the next GOP as shown below.

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

Rem = G + Rem

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

Where G is the total bits assigned to a GOP, N is the number of pictures in the GROUP OF PICTURES.

At the start of the sequence Rem = 0.

Np and Nb are the number of P-pictures and B-pictures remaining in the current GROUP OF PICTURES in the encoding order. For example in fig. 2.9, we have a GOP with N=13 (total frames in GOP). If we are just to encode the seventh frame in encoding order (bold), then Np=3 and Nb=4

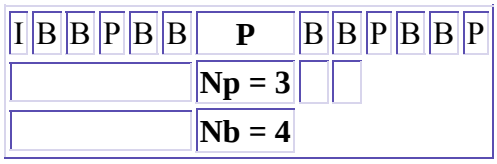


Fig. 3.9: Example of Remaining pictures in GOP at frame 7

B. Rate Control

Video coders often have to operate within fixed bandwidth limitations. Since MPEG-2 standard results in a variable bit rate, some form of bit rate control is required for operation on bandwidth-limited channels. The MPEG-2 rate control regulates DCT quantization, so as to maintain constant quality in the decoded video while limiting variations in encoded bit rate. A separate virtual buffer is maintained for each picture type.

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

$$b\_f_j^{pt} = b\_f_0^{pt} + B_{j-1} - \left(\frac{T_{pt} * (j-1)}{MB\_cnt} \right)$$

depending on the picture type.

where

$b\_f_0^{pt}$ are initial fullnesses of virtual buffers - one for each picture type.

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

MB_cnt is the number of macroblocks in the picture.

$b\_f_j^{pt}$ are the fullnesses of virtual buffers at macroblock j- one for each picture type.

The final fullness of the virtual buffer ($b\_f_j^{pt}$: j = MB_cnt) is used as $b\_f_0^{pt}$ for encoding the next picture of the same type.

Next the reference quantization scale QS^j_{REF} for macroblock j is computed as follows:

$$QS_{REF}^{j} = \frac{b\_f_j * 31}{r}$$

where the "reaction parameter" r is given by

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

and b_fj is the fullness of the appropriate virtual buffer.

The initial value for the virtual buffer fullness is:

$$b\_f_0^I = 10 * \frac{r}{31}$$
$$b\_f_0^P = K_P * b\_f_0^I$$
$$b\_f_0^B = K_B * b\_f_0^I$$

C. Adaptive Quantization

The rate control step provides a reference quantization parameter to code each macroblock. The reference quantization scale is modulated with an activity factor that is determined from a measure of the spatial activity of the macroblock. The rationale is that a macroblock that has little spatial activity, such as a smooth region, should be quantized more finely than a macroblock with high spatial activity,such as textured region, since quantization error is typically more noticeable in smooth regions than in textured regions. This approach attempts to equalize perceptual quality for a given quantization scale.

First, a spatial activity measure for the macroblock j from the four luminance frame-organised sub-blocks (n=1...4) and the four luminance field-organised sub-blocks (n=5…8, only for interlaced video) using the intra (i.e. original) pixel values is computed:

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

where

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

and

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

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

Then actj is normalized:

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

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

Finally QSj is obtained as:

$$QS_j = QS^j_{REF} * N\_actj$$

where QS^j_{REF} is the reference quantization parameter obtained in step 2. Finally, QS_j is clipped to the allowed values (depending on the quantization type, linear-nonlinear) and the quantization scale code QPj for this QSj is computed.

3.4.2 Adaptive Model-Driven Bit Allocation for MPEG Video Coding

An adaptive model-driven bit-allocation algorithm for video sequence coding is introduced in [3]. The algorithm is based on a parametric rate-distortion model, and facilitates both picture- and macroblock level bit allocation. A region classification scheme is incorporated, which exploits characteristics of human visual perception to efficiently allocate bits according to a region's visual importance.

A. Block-Level Parametric Rate-Quantization Model

Applying the theoretical rate-distortion function for Gaussian random variable (appendix A) with squared error distortion, we have the following expressions for rate and distortions in a block (N=1-64):

$$R_b = \sum_{i=1}^{N} \frac{1}{2} \log_2 \frac{\sigma_i^2}{d_i} \quad (1)$$

And

$$D_b = \sum_{i=1}^{N} d_i \quad (2)$$

where σ^2 is the variance of X_i, and d_i is the power of the distortion induced by quantization of X_i.

For relatively high bit-rates, the noise induced by quantization of X_i is approximately uniformly distributed over

$$\left(\frac{-QS * W_i}{2}, \frac{QS * W_i}{2} \right] \quad (3)$$

and the power of the quantization noise is $d_i=(QS^2* (W_i/12))$

At lower bit rates, the uniform distribution assumption is not valid. Therefore, for any bit rate, low or high, we can use $d_i=\gamma_i*QS^2$, and then have the following expressions for rate and distortion in a block:

$$R_b = \sum_{i=1}^{N} \frac{1}{2} \log_2 \frac{\sigma_i^2}{\gamma_i QS^2} \quad (4)$$

And

$$D_b = \sum_{i=1}^{N} \gamma_i QS^2 = QS^2 \sum_{i=1}^{N} \gamma_i = \Gamma * QS^2 \quad (5)$$

For the special case of intracoded blocks it is:

$$R_b = r_1 + \sum_{i=2}^{N} \frac{1}{2} \log_2 \frac{\sigma_i^2}{\gamma_i QS^2} \quad (6)$$

And

$$D_b = d_1 + \sum_{i=2}^{N} \gamma_i QS^2 \quad (7)$$

Where r_1 is the bit count for encoding X_1, and d_1 is the distortion induced by quantization of X_1 (relatively quite small and therefore neglected).

We define

$$\kappa = \frac{\sigma^2}{\sqrt[N]{\prod_{i=1}^{N} \sigma_i^2 / \gamma_i}}$$

For non-intra coded blocks and

$$\kappa = \frac{\sigma^2}{\sqrt[N-1]{\prod_{i=2}^{N} \sigma_i^2 / \gamma_i}}$$

For intra coded blocks
Using (4), (5) and the above equations we form a rate-quantization block model as follows:

$$R_b = \frac{\hat{N}}{2} \log_2 \frac{\sigma^2}{\kappa * QS^2} \quad (8)$$

where σ^2 is the variance of the pixel data in the block and $\check{N}$=N-1 for intracoded blocks and $\check{N}$=N otherwise.

B. Macroblock-Level HVS-Based Bit allocation

A macroblock-level perceptual classification algorithm classifies each macroblock in a video sequence in a perceptual class with index p, $1 \le p \le P$. Associated with each perceptual class is a perceptual noise sensitivity β_p.
The bit allocation algorithm allocates bits to each macroblock, such that each macroblock in the same perceptual class has the same quantization noise power
$MD_1:MD_2:\ldots:MD_p=\beta_1:\beta_2:....:\beta_p$ (9)

Where MD_p is the quantization noise in a macroblock with perceptual index p. From (5) we have:
$QS_1^2/\beta_1=QS_2^2/\beta_2=...=QS_p^2/\beta_p$ (10)
Where QS_p is used to encode all macroblocks in perceptual class p.

C. Macroblock-Level QS Selection

Suppose a total of M blocks are to be quantized in a picture, including all color channels. Let the coding parameter α be defined such that

$$\sum_{j=1}^{M} \frac{\hat{N}_j}{2} \log_2 a = \sum_{j=1}^{M} \frac{\hat{N}_j}{2} \log_2 \kappa_j \quad (11)$$

Then using (8), the picture-wise rate quantization function is given by:

$$R_p = \sum_{j=1}^{M} \frac{\hat{N}_j}{2} \log_2 \frac{\sigma_j^2}{\alpha * QS_{p_j}^2} \quad (12)$$

Where p_j is the perceptual class index for block j, QS_{pj} is the QS used for blocks in perceptual class p_j, and σ_j^2 is block j's pixel variance. We choose the QS_ps for a picture by first using (12) to determine an optimal QS_p for any one particular p, and then applying (10) to solve for the QS_p's for all other perceptual classes.
For example, suppose that QS_1 is chosen first. Then bit allocation first requires minimization of QS_1, subject to

$$\sum_{j=1}^{M} \frac{\hat{N}_j}{2} \log_2 \frac{\sigma_j^2}{\alpha * \beta_{p_j} * QS_1^2 / \beta_1} \le T_p \quad (13)$$

Where T_p denotes the target bit-budget for a picture. Since, in MPEG coding, QS is limited to a finite set of values, and since the bit rate is a monotonic function of QS, this minimization can be readily performed using an efficient search procedure.

In (13), the parameter α could be determined by performing a precoding pass to generate an (R, D) pair. However, in video compression, we can take advantage of the temporal stationarity in the video sequence to calculate an estimate for α. The block-pixel variance σ_j^2 is easy to compute, and in fact has been commonly used as an activity measure.

4. MPEG-4 Video Compression Standard

In the past the Moving Pictures Experts Group had focused on limited application fields. The MPEG-1 and MPEG-2 standards, were only concerned about audio and video compression; MPEG-1 for example targeted towards consumer-oriented applications like CD-I etc. MPEG-2 forms the core of digital TV and digital broadcasting and was extended for professional studio applications like post-production, non-linear editing etc.

MPEG-4 is meant to become the universal language between broadcasting, movie and multimedia applications. It will provide additional functionality over simple media compression, like bitrate scalability, object-based representation, intellectual property management & protection etc., and is based on a rich tool set starting at bitrates as low as 2 kbit/s for a single audio channel.

MPEG-4 achieves these goals by providing standardized ways to:

1. Represent units of aural, visual or audiovisual content, called "media objects". These media objects can be of natural or synthetic origin; this means they could be recorded with a camera or microphone, or generated with a computer;
2. Describe the composition of these objects to create compound media objects that form audiovisual scenes;
3. Multiplex and synchronize the data associated with media objects, so that they can be transported over network channels providing a QoS appropriate for the nature of the specific media objects; and
4. Interact with the audiovisual scene generated at the receiver's end.

The MPEG-4 standard consists of a total of 20 parts under ISO/IEC 14496. In brief these parts are:

Part 1 Systems: Describes synchronization and multiplexing of video and audio.

Part 2 Visual: A compression codec for visual data (video, still textures, synthetic images, etc.). One of the many "profiles" in Part 2 is the Advanced Simple Profile (ASP).

Part 3 Audio: set of compression codecs for perceptual coding of audio signals, including some variations of Advanced Audio (AAC) Coding as well as other audio/speech coding tools.

Part 4 Conformance: Describes procedures for testing conformance to other parts of the standard.

Part 5: Provides software for demonstrating and clarifying the other parts of the standard.

Part 6 DMIF (Delivery Multimedia Integration Framework): Defines an interface between the application and network/storage.

Part 7 Optimized Reference Software: Provides examples of how to make improved implementations (e.g., in relation to Part 5).

Part 8 Carriage on IP networks: Specifies a method to carry MPEG-4 content on IP networks.

Part 9 Reference Hardware: Provides hardware designs for demonstrating how to implement the other parts of the standard.

Part 10 Advanced Video Coding: A codec for video signals which is also called *AVC* and is technically identical to the ITU-T H.264 standard.

Part 12 ISO Base Media File Format: A file format for storing media content.

Part 13 Intellectual Property Management and Protection (IPMP) Extensions.

Part 14 MPEG-4 File Format: The designated container file format for MPEG-4 content, which is based on Part 12

Part 15 AVC File Format: For storage of Part 10 video based on Part 12

Part 16: Animation Framework eXtension (AFX).

Part 18: Font Compression and Streaming (for OpenType fonts).

Part 19: Synthesized Texture Stream.

Part 20: Lightweight Scene Representation (LASeR) (not yet finished - reached "FCD" stage in January 2005).
Part 21: MPEG-J Extension for Rendering (not yet finished - reached "CD" stage in January 2005).

4.1 MPEG-4 Video Compression

4.1.1 Structure and Syntax

The central concept defined by the MPEG-4 standard is the audio-visual object, which forms the foundation of the object-based representation. Such a representation is well suited for interactive applications and gives direct access to the scene contents. A video object may consist of one or more layers to support scalable coding. The scalable syntax allows the reconstruction of video in a layered fashion starting from a standalone base layer, and adding a number of enhancement layers. This allows applications to generate a single MPEG-4 video bitstream for a variety of bandwidth and/or computational complexity requirements. A special case where a high degree of scalability is needed is when static image data is mapped onto two or three dimensional objects. To address this functionality, MPEG-4 provides a special mode for encoding static textures using a wavelet transform.

An MPEG-4 visual scene may consist of one or more video objects. Each video object is characterized by temporal and spatial information in the form of shape, motion, and texture. For certain applications video objects may not be desirable, because of either the associated overhead or the difficulty of generating video objects. For those applications, MPEG-4 video allows coding of rectangular frames which represent a degenerate case of an arbitrarily shaped object.

An MPEG-4 visual bitstream provides a hierarchical description of a visual. Each level of the hierarchy can be accessed in the bitstream by special code values called start codes. The hierarchical levels that describe the scene most directly are:

- ***Visual Object Sequence (VS):*** The complete MPEG-4 scene which may contain any 2-D or 3-D natural or synthetic objects and their enhancement layers.
- ***Video Object (VO):*** A video object corresponds to a particular (2-D) object in the scene. In the simplest case this can be a rectangular frame, or it can be an arbitrarily shaped object corresponding to an object or background of the scene.
- ***Video Object Layer (VOL)***: Each video object can be encoded in scalable (multilayer) or non-scalable form (single layer), depending on the application, represented by the video object layer (VOL). The VOL provides support for scalable coding. A video object can be encoded using spatial or temporal scalability, going from coarse to fine resolution. Depending on parameters such as available bandwidth, computational power, and user preferences, the desired resolution can be made available to the decoder.

Each video object is sampled in time, each time sample of a video object is a video object plane. Video object planes can be grouped together to form a group of video object planes:

- ***Video Object Plane (VOP):*** A VOP is a time sample of a video object. VOPs can be encoded independently of each other or dependent on each other by using motion compensation. A conventional video frame can be represented by a VOP with rectangular shape.
- ***Group of Video Object Planes (GOV):*** The GOV groups together video object planes. GOVs can provide points in the bitstream where video object planes are 8 encoded independently from each other, and can thus provide random access points into the bitstream. GOVs are optional.

4.1.2 Coding

A video object plane can be used in several different ways. In the most common way the VOP contains the encoded video data of a time sample of a video object. In that case it contains motion parameters, shape information and texture data. These are encoded using macroblocks. A macroblock contains a section of the luminance component and the spatially subsampled chrominance components. In the MPEG-4 visual standard there is support for only one chrominance format for a macroblock, the 4:2:0 format. In this format, each macroblock contains 4 luminance blocks, and 2 chrominance blocks. Each block contains 8x8 pixels, and is encoded using the DCT transform. A macroblock carries the shape information, motion information, and texture information. Each video object is coded separately. For reasons of efficiency and backward compatibility, video objects are coded via their corresponding video object planes in a hybrid coding scheme somewhat similar to previous MPEG standards. There are I-VOPs, P-VOPs and B-VOPs. I-VOPS are code independently from other VOPs, while P and B- VOPs can be predicted from previous or future VOPs, just like I-, P- and B- pictures in MPEG-2. Then, after DCT transformation, the resulting coefficients are quantized just like in MPEG-2 by the quantization matrix and the quantization scale. Finally, run length coding is used to encode the quantized coefficients.

The coding efficiency in MPEG-4 is improved by allowing the prediction of the quantized coefficients from the block above, left or above left.

4.1.3 Video Profiles, Video Object types

The MPEG-4 Visual standard defines (by October 2001) 18 visual object types and 19 visual profiles. Nine visual profiles have been defined in MPEG-4 Visual Version 1: Simple, Simple Scalable, Core, Main, N-bit, Scaleable Texture, Simple Face Animation, Basic Animated Texture, and Hybrid.

Six additional visual profiles have been defined in MPEG-4 Visual Version 2: Core Scalable, Advanced Core, Advanced Coding Efficiency, Advanced Real Time Simple, Advanced Scaleable Texture, and Simple FBA.

Moreover 2 additional profiles have been defined in the 1st Extension to the 2nd Edition of the MPEG-4 Visual standard: Simple Studio and Core Studio. And 2 profiles in the 2nd Extension to the 2nd Edition of the MPEG-4 Visual standard: Advanced Simple and Fine Granularity Scalability.

MPEG-4 video profiles are defined in terms of video object types. In the next table, the object types supported by some of the most used profiles are shown.

MPEG-4 video object types	MPEG-4 video profiles					
	Simple	Core	Main	Simple Scalable	N-Bit	Scalable Texture
Simple	●	●	●	●	●	
Core		●	●		●	
Main			●			
Simple Scaleable				●		
N-Bit					●	
Scalable Still Texture			●			●

Table 3.1 MPEG-4 Video Profiles

Each MPEG-4 video object type can support a specific set of tools as seen in the following table:

MPEG-4 video tools	MPEG-4 video object types					
	Simple	Core	Main	Simple Scalable	N-bit	Still Scalable Texture
Basic(I and P-VOP, coefficient prediction, 4-MV, unrestricted MV)	●	●	●	●	●	
Error resilience	●	●	●	●	●	
Short Header	●	●	●		●	
B-VOP		●	●	●	●	
P-VOP with OBMC (Texture)						
P-VOP based temporal scalability		●	●		●	
Binary Shape		●	●		●	
Grey Shape			●			
Interlace			●			
Temporal Scalability (Rectangular)				●		
Spatial Scalability (Rectangular)				●		
N-Bit					●	
Scalable Still Texture						●

Table 4.2: MPEG-4 Video Object Types

From table 4.2, we can notice the following attributes of the Video Objects Types:

- Simple VO: Codes rectangular video and uses intra (I) and predicted (P) video object planes (VOPs, the MPEG-4 term for *frames*).
- Core VO: Codes arbitrarily shaped video, uses a tool superset of Simple, adds bidirectional (B) video object planes (VOPs), binary shape coding, and supports temporal scalability based on sending extra P-VOPs.
- Main VO: Codes arbitrarily shaped video, adds to Core the coding of grayscale shapes and interlaced coding.
- Scalable Texture: Codes arbitrarily shaped still image with wavelet compression and incremental download.

4.1.4 Levels

A level within a profile defines constraints on parameters in the bitstream that relate to the tools of that profile. Currently there are 11 natural video profile and level definitions that each constrains about 15 parameters that are defined for each level. To provide some insight into the levels, for the three most important profiles, core, simple, and main, a subset of level constraints is given in Tab. 3.3. The macroblock memory size is the bound on the memory (in macroblock units) which can be used by the (Video reference Memory Verifier) VMV algorithm. This algorithm models the pixel memory needed by the entire visual decoding process.

Profile and Level		Typical scene size	Bitrate (bit/sec)	Maximum number of objects	Total mblk memory (mblk units)
Simple Profile	L1	QCIF	64 k	4	198
	L2	CIF	128 k	4	792
	L3	CIF	384 k	4	792
Core Profile	L1	QCIF	384 k	4	594
	L2	CIF	2 M	16	2376
Main Profile	L2	CIF	2 M	16	2376
	L3	ITU-R 601	15 M	32	9720
	L4	1920x1088	38.4 M	32	48960

Table 4.3: MPEG-4 Profiles & Levels

4.2 Rate Control for Simple Profile (SVO)

The MPEG committee has adopted a quadratic rate-quantizer model for the single video object rate-control algorithm [15].

The number of target bits per frame is initially set to a weighted sum of the number of bits used for coding the previous frame and the average number of the remaining bits per frame, and then to prevent buffer underflow and overflow, the target is scaled by a proportional factor based on the current buffer occupancy.

According to the VM18 verification model, a quadratic rate-quantizer (R-Q) for a single frame at $t=t_n$ is given by:

$$R(t_n) = S_n(\frac{X_{1,n}}{Q_n} + \frac{X_{2,n}}{Q_n^2}) \quad (51)$$

where S_n is the encoding complexity, often substituted by the sum or mean of absolute differences of the residual component, Q_n denotes the QP, and $X_{i,n}$ denotes the model parameters that are updated by linear regression method from previous coded parameters. The initial target is determined according to the following expression:

$$T_{1,n} = Max(\frac{bitrate}{30}, 0.95\frac{R_r}{N_r} + 0.05R_{c,n-1}) \quad (52)$$

where T_1,nis the target bit rate to be used for current frame, R_r is the number of bits remaining for encoding this sequence, N_r is the number of P frames remaining to be encoded, and $R_{c,n-1}$ is the actual bits used to encoding previous frame.
Once the initial target has been set, it is scaled according to:

$$T_{2,n} = T_{1,n}\frac{B_n + 2(B_s - B_n)}{2B_n + (B_s - B_n)} \quad (53)$$

where T_2,n is the adjusted target bit rate to achieve the middle level and to reduce any buffer overflow or underflow, B_s is the buffer size, and B_n is the current buffer level which is expressed as:

$$B_n = B_{n-1} + R_{c,n-1} - R_{drain} \quad (54)$$

With a safety margin of 0.9, the final target estimate is described by:

$$T_n = \begin{cases} \text{Max}\left(\frac{R_s}{30}, 0.9B_s - B_n\right), & \text{if } B_n + T_{2,n} > 0.9B_s \\ R_{\text{drain}} - B_n + 0.1 \times B_s, & \text{if } B_n - R_{\text{drain}} + T_{2,n} < 0.1B_s \\ T_{2,n}, & \text{otherwise} \end{cases} \quad (55)$$

After the target bit rate is computed, the QP (Q_n) is solved based on (51), and is clipped between 1 and 31. Q_n is limited to vary within 25% of the previous to maintain a stable quality. After encoding the current frame, the next frame is skipped if the current buffer status is above 80%.

4.3 Rate control for multiple video objects (MVO)

The rate control algorithm for multiple objects was proposed in [17] by A. Vetro, H. Sun and Yao Wang and was adopted by the MPEG committee.

It is an extension of the SVO algorithm. The same method is used to allocate target bits to a frame, and the total target bits of this frame are distributed proportional to the relative size, motion and variance of each object within this frame.{Vetro and Sun; MPEG-4 rate control for multiple video objects}.

It is obvious that, once the distribution of the bits among the objects is done, then we can simply use (51) to obtain the quantization parameter for each object.

4.3.1 MPEG-4 rate control for multiple video objects

A. Overview of the MVO Algorithm

Initialization: The initialization process is not very different from the SVO process described before. Most of the notation is unchanged, but many of the variables are extended to vector quantities so that each object can maintain its own set of parameters.

Initial Target Bit-Rate Estimation: To estimate an initial total target bit rate, the solution given by (2) can be used. Alternatively, the target can be made object based by allocating the bit rate for the ith object proportional to the bit rate used for the ith object of the previous frame $R_{p,}i$

$T_1 = \sum_{i \in M} T_i$ with

$$T_i = \max\left\{ \frac{R_s}{mF_s}, (1-w_p)\frac{T_r}{mN_r} + w_p R_{p,i} \right\} \quad (56)$$

In the above equation, M={0,1,…,m}is the set of video object (VO) id's. An increase in the value of w_p will skew the individual targets more proportional to $R_{p,i}$. It should be noted that the initial estimate does not need to be very accurate, and either of the above two methods can be used.

Joint Buffer Control: For the MVO algorithm, the scaling procedure and the overflow/underflow adjustments can be performed in the same way.

Quantization Level Calculation: Given the values of X_{1i}, X_{2i}, MAD_i and $T_{texture,i}$, the appropriate values of Q_i can easily be found. The target number of bits for the texture of the ith object is defined as:

$$T_{texture,i} = T_i - T_{hdr,i} \quad (57)$$

where $T_{hdr,i}$ represents the amount of shape, motion, and header bits used for the ith object of the previous frame. Then the Q_i is calculated from (52) by replacing $R(T_n)$ with $T_{texture,i}$. Notice that each object has its own model parameters $X_{i,n}$.

Shape-Coding Parameter (AlphaTH) Calculation: This block is used to determine AlphaTH, the parameter that controls shape distortion in MPEG4. The adjustment of this parameter can provide a trade off between texture and shape coding accuracy. For now, we assume that it is fixed to zero, which leads to lossless shape coding.

B. Target Distribution Among Objects

An object-based coder attempts to code each object with a different quantization parameter. This is done to exploit the fact that each object need not be coded with the same precision to achieve comparable quality.

For a given target, the target for object is given by:

$T_i = T\ (w_s SIZE_i + w_m MOT_i + w_u VAR_i)$

where $SIZE_i$, MOT_i and VAR_i are the size, motion, and MAD^2 of object i normalized by the total SIZE, MOT, and VAR of all objects, respectively. Here, the motion magnitude of the ith object, MOT_i is the sum of the absolute values of each motion vector component within objects i and the size of the object $SIZE_i$ is simply the number of macroblocks or partial macroblocks within the object. The weights $\{w_s, w_m, w_v\} \epsilon [0,1]$ and satisfy $w_s+w_m+w_v=1$.

4.3.2 Rate Control and Bit Allocation for MPEG-4

Another approach proposed in [18] relies on the modelization of the source and the optimization of a cost criterion based on signal quality parameters. Algorithms are introduced to minimize the average distortion of the objects to guarantee desired qualities to the most relevant ones and to keep constant ratios among the object qualities.

A. Some Basic Cost Functions

1) Cost Function for Weighted Distortion (WD) Control: A natural choice for the control objective is the minimization of the weighted average of the distortion of the different VO's. If a different positive weighting factor is assigned to each object, the control problem can be stated as the minimization of the total weighted distortion:

$$D^{(n)} = \sum_{i=1}^{N} \alpha_i p_i^{(n)} d_i^{(n)} \quad (58)$$

where α_i the are weighting factors. By assigning larger factors to the semantically more important VO's, the encoder is indirectly instructed to be more careful with them than with other less important objects.

2) Cost Function for Priority-Based (PB) Control: A limitation of the previous approach is the lack of direct control on the distortion with which each VO is coded. A way to express priorities by specifying precise quality targets is to establish an ordered list of objectives and distribute the bits in such a way that as many objectives as possible, in order of priority, are achieved.
Formally, control objectives can be specified by an ordered list of the form

$$L_P = \{(i_1, \overline{d}_1), (i_2, \overline{d}_2), \ldots, (i_L, \overline{d}_L)\}$$

where the i_j are VO numbers and the d_j represent *target distortions*. The semantics of a specification of this form is the following. Each pair (i_j,d_j) in LP specifies the objective of coding VO number i_j with a distortion equal to or lower than d_j. The list establishes the order in which these objectives should be fulfilled. Nothing prevents that the same object number appears in as many list items as desired as long as the associated target distortions are listed in decreasing order.

A practical difficulty associated with this approach is that nothing prevents the user from establishing a list of objectives impossible for the given bit rate and which might produce an assignment of all the available bits to a single object. A careful design of the objective list is therefore an important issue.

3) Cost Function for Constant Distortion Ratios (CDR) Control: We introduce now an alternative that also provides a better control of the results than WD, but with a simpler and more object-property-independent specification than the PB technique. The aim now is to achieve ratios of distortions between pairs of VOP's of the sequence as close as possible to given constants.

Since in practice exact ratios cannot be achieved, due to the discrete nature of the control parameter set, it is more realistic to consider a compromise between optimization of the global quality and achievement of the target distortion ratios. This leads to an alternative formulation of the objectives in terms of the minimization of the weighted sum of two terms

$$D^{(n)} = k_1 \sum_{i=1}^{N} p_i^{(n)} d_i^{(n)} + k_2 \sum_{i=1, i\neq s}^{N} |d_i^{(n)} - \beta_i d_s^{(n)}|$$

where s is the number of a VO arbitrarily selected as reference and k_1 and k_2 are nonnegative constants that allow one to tune the compromise between the two cost terms, a larger k_1/k_2 ratio meaning that more importance is given to the first objective than to the second.

B. Quantizer Parameter Computation

This phase receives as input the total bit budget target R(n) for the current VOP's and returns the coding parameters to employ in their coding. These parameters are computed in order to minimize (maximize) the adopted cost (quality) measure for the set of current VOP's.

Formally, the problem is to compute the control parameters $u_i^{(n)}$ that minimize

$D(\tilde{d}_1^{(n)}(u_1^{(n)}), \ldots, \tilde{d}_N^{(n)}(u_N^{(n)}), p_1^{(n)}, \ldots, p_N^{(n)})$ under the restriction

$\Sigma_{i=1}^{N} \tilde{r}_i^{(n)}(u_i) \leq \tilde{R}^{(n)}$

where $d_i^{(n)}$ ($u_i^{(n)}$) and $r_i^{(n)}$ ($u_i^{(n)}$) represent, respectively, the distortion and the number of bits that the model estimates for the coding of the current VOP of object when QP $u_i^{(n)}$ is employed.

1) Algorithm for WD Cost Optimization: For the first of the considered cost functions, the coding parameters for each VOP which minimize the global weighted distortion (58) can be obtained by minimizing the following expression with respect to the u_i:

$$D = \sum_{i=1}^{N} \alpha_i p_i \tilde{d}_i(u_i)$$ under the restriction $\Sigma_{i=1}^{N} \tilde{r}_i(u_i) \leq \tilde{R}$

where Ř is the target number of bits to distribute among the VOP's. This optimization problem is just an instance of the well-known *optimal quantizer assignment problem* and is solved employing the discrete version of the Lagrange multiplier technique.

2) Algorithm for PB Cost Optimization: Now we provide a procedure to achieve a rate distribution respecting a list of priority objectives. The algorithm operates by successively assigning bits so that the objectives of the list are achieved in order of priority. More precisely, the following steps are performed.

1) Assign each current VOP the coarsest quantizer and estimate the corresponding total number of bits. If it is greater than Ř finish.
2) Otherwise, for each item (i,d) in the list, while there still remain bits to allocate, obtain the minimum number of bits necessary to code the current VOP of object i with a distortion equal or lower than d.
3) If the resulting increase in the number of bits assigned to the VOP is affordable, update this assignment as well as the amount of remaining bits. Otherwise, assign this VOP the remaining bits and finish the assignment.

As in the previous case, no assumption is made on the characteristics of the model functions $d_i(u)$ and $r_i(u)$.

3) Algorithm for CDR Cost Optimization: In this case we want to minimize with respect to the u_i

$$k_1 \sum_{i=1}^{N} p_i \tilde{d}_i(u_i) + k_2 \sum_{i=1, i\neq s}^{N} |\tilde{d}_i(u_i) - \beta_i \tilde{d}_s(u_s)|$$ subject to $\Sigma_{i=1}^{N}\ \tilde{r}_i(u_i) \leq \tilde{R}$

The Lagrange multiplier technique can also be applied to this problem, resulting in the associated parametric unrestricted problem of minimizing:

$$k_1 \sum_{i=1}^{N} p_i \tilde{d}_i(u_i) + k_2 \sum_{i=1, i\neq s}^{N} |\tilde{d}_i(u_i) - \beta_i \tilde{d}_s(u_s)| + \lambda \sum_{i=1}^{N} \tilde{r}_i(u_i).$$

However, the λ -optimal values $u_i(\lambda)$ can no longer be obtained independently as in the WD case because of the particular role played by the parameter for the reference VOP(u_s). The difficulty can be overcome by embedding the quantizer assignment algorithm within an exhaustive search in u_s. In spite of the increase in complexity, the computational cost remains easily affordable.

Regarding the expected behavior of this algorithm, observe that a small value of the k_1/k_2 factor can result in an underuse of the available rate, constituting a price paid for a more exact achievement of the distortion ratios. This can be solved by increasing this ratio if this rate underuse results in buffer underflow.

5. Implementations

5.1 Modified MPEG-2 Encoder

The modified MPEG-2 encoder is an enhanced version of the original *MPEG-2 Encoder / Decoder, Version 1.2, July 19, 1996* of the *MPEG Software Simulation Group*, offered as open source code. The code can be found at http://www.mpeg.org.

The modifications result in faster encoding process and better compression. The most important modifications are presented in brief in the next paragraph.

First, the structure of the implementation is altered in order to speedup the encoding process. Then, in order to improve the coding efficiency of intra coded macroblocks (MBs) (P- or I-frames), we predict each intra coded MB from previous MBs in the same frame; a process we call "intra frame" prediction.

Finally, the quantization matrix of a picture is chosen based on the mean absolute difference (MAD) from the previous frame, while the quantization scale (QS) of each MB based on whether it belongs to a region of interest (ROI) or not. ROIs are detected by computing the total absolute difference (TAD) of a MB from the MB in the same position in the previous original frame.

In the following section all the changes are presented analytically.

5.1.1 Improved implementation of the encoder

In general the encoding process of MPEG-2 can be described by the following steps:

- Motion estimation
- Prediction
- Transformation
- Quantization
- Stream Generation
- Inverse quantization
- Inverse Transformation
- Prediction Addition

The final three steps, are the reconstruction of the last encoded anchor frame that will be used to predict the next frame (if it is not an I frame).

In the original implementation, **for each of these steps**, the encoder would loop through all the MBs of the frame. In our implementation, we loop only twice through all the MBs- once to do the motion estimation step and once for all the other steps

Except for the obvious speed up of the encoding process, this modification also gives us the capability to predict a MB from the reconstructed MBs in the same frame (as will be explained later).

5.1.2 Motion Estimation

During the motion estimation step the original encoder decides whether to intra code a MB and whether to use or not Motion Compensation (MC). First it finds the region from the reference picture that best predicts the current MB. This is achieved by doing a full

search on the original reference picture, within the user defined search limits, and by choosing the region with the smallest Total Absolute Difference (TAD) from the current MB. Then, the motion vectors are generated and the Total Squared Difference (TSD) of the prediction from the current MB is computed. If the computed TSD is greater than the Variance (VAR) of the current MB multiplied by 256 and greater than 9*256, then the MB is intra coded. Otherwise, if the TSD of the current MB from the MB in the original reference picture at the same position is smaller from 1.25 times the TSD of the current MB from the prediction, then no Motion Compensation is used (i.e slightly biased towards No-MC to avoid coding the motion vectors). We should note that the decision of whether a MB is skipped or not is taken in the stream generation step. The conditions to skip a MB are:

1) Must be a No-MC MB.
2) The values of the quantized coefficients must be all zero.
3) Should not be the first or last MB in a slice.

In order to further speed up the encoder we also altered the decision process for skipped and no-MC MBs.

In our modified encoder, for each MB we first compute the TAD from the MB at the same position in the previous frame (reference frame). The TAD is computed on the original values of the MBs and not the reconstructed. We chose TAD for MBs because it gives us more flexibility in setting the thresholds for skipped MBs (T3), no-MC MBs (T2) and ROIs (T1).

In our implementation, if the TAD is lower than a threshold T3 (skipped MB decision threshold) then the MB is automatically considered skipped. If the TAD is higher than T3 but lower than T2 (No-MC decision MB) then no motion compensation is used for the MB. By defining these two thresholds (T2, T3) we don't have to search for a better prediction like the Original encoder does and improve the encoding delay. By selecting carefully these thresholds we will also not loose in coding efficiency as our decisions for the type of the MBs (skipped, No-MC) will actually be the same as the decisions of the Original encoder. Notice that the motion estimation step is the most time consuming step in the whole MPEG-2 encoding process.

If the TAD is greater than T2, then the encoding process followed is identical to the original described above. Finally, a MB is considered a ROI if the TAD is greater than a threshold T1.

The equations of the variance (VAR), total absolute difference (TAD), total squared difference (TSD) and mean absolute difference (MAD) can be found in table 5.1 in page 58.

5.1.3 Prediction of Intra Coded MBs

For intra coded MBs (I-frames or P-frames), we use "intra frame" prediction. The prediction can be formed from either the MB above (C) or the MB to the left (A) as illustrated if fig. 5.1. MBs A and C contain the reconstructed values and not the original.

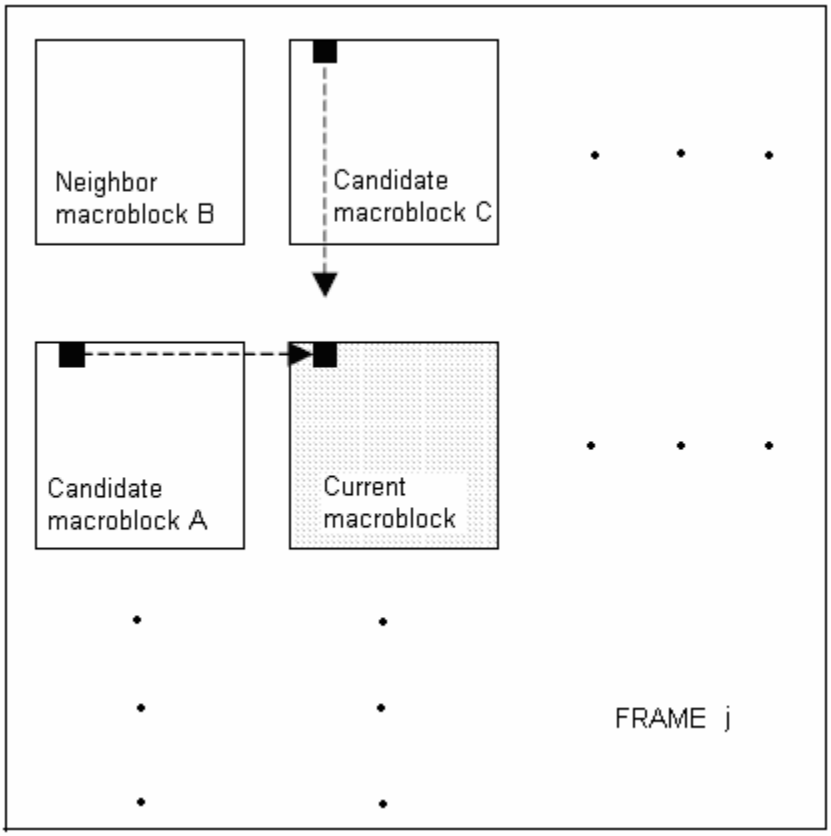


Figure 5.1

The decision is based on the total squared difference (TSD, table 5.1) of the Y component of the current MB from MB A and MB C. In the original encoder, all intra coded MBs are predicted from a virtual MB with pixel values 128 in all positions. We also compute the TSD of the current MB from the virtual MB. The MB with the smallest TSD is used to predict the current MB.

We chose TSD because it gave better results in the tests. MB B in fig. 5.1 was not taken into account because it didn't improve much the coding efficiency, and raised the complexity of the implementation.

We recognize the type of "intra frame" prediction from the value of the quantization scale code QP. The following formula was adopted:

$$QP=\begin{cases}1+m*3, & \text{If prediction from MB A}\\ 2+m*3, & \text{If prediction from MB C} \qquad m=0,\ 1,\ldots,9\\ 3+m*3, & \text{If prediction from Virtual MB}\end{cases}$$

5.1.4 Computation of Quantization Matrices

The MAD (table 5.1) of the current frame from the previous is used to scale the quantization matrices used to code the frame. The higher the MAD, the more movement is present in the frame. As a result, the encoding complexity raises and in order to avoid undesirable spikes in the bit-rate higher quantization values are used.

To compute the mean absolute difference (MAD) of the current frame from the previous (reference) we add all the TADs (computed in the motion estimation step) of all MBs in the current frame from the reference frame and divide by the total number of MBs in a frame.

We use the default quantization matrices as reference and acquire the new quantization matrices with the following equation:

$$\text{Intra\_matrix}[i] = \text{default\_intra\_matrix}[i] * (1 + MAD * 0.2),\ i=0, .., 63$$

$$\text{Inter\_matrix}[i] = \text{default\_inter\_matrix}[i] * (1 + MAD * 0.2),\ i=0, .., 63$$

New quantization matrices are used only if the change of MAD is greater than 2. Note that the downloading of new quantization matrices was not implemented in the open source version of the encoder, thus we had to implement it.

5.1.5 Rate Control of Modified MPEG-2 Encoder

In the Modified MPEG-2 implementation it is considered that every frame is made up from two sub- frames; the sub-frame of the ROI MBs and the sub-frame of the rest MBs. For each sub-frame the TM-5 rate control algorithm, used in the original encoder, is applied separately.

We keep the bit allocation and complexity estimation steps as in the original encoder. After the initial bit allocation step for a picture, we allocate the bits between the two sub-frames. The bit allocation between the frames is based on the number of MBs classified as ROIs in the picture. Then, we duplicate the rate control step by defining two virtual buffers, one for each sub frame. Quantization scales are then computed independently for each sub frame, just like in the original rate control algorithm.

The new rate control is described in detail in the following section in correspondence with the TM-5 algorithm description in section 3.4.1.

The only new notation introduced is the index m which takes the value r (ROI) for an attribute that belongs to a region of interest or n (non-ROIs) if it doesn't.

A. Bit Allocation

Complexity estimation

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

$$X_i = (S_n + S_r) * Q_i$$

$$X_p = (S_n + S_r) * Q_p$$

Where Sn, Sr are the total bits used to encode all the MBs of type n,r respectively in the frame. Note that the sum of Sn and Sr is equal to S used in the original TM-5 algorithm.

Picture Target Setting

As in the original TM5 algorithm, first a target bit setting is calculated for the whole frame. Then, the target bits are distributed to the two sub frames.
The new equations are:

$$T_I = \max\left[\frac{\mathrm{Re}\,m}{\left(1+\frac{N_P X_P}{X_I K_P}\right)}, \frac{bit\_rate}{8 * picture\_rate}\right]$$

$$T_P = \max\left[\frac{\mathrm{Re}\,m}{N_P}, \frac{bit\_rate}{8 * picture\_rate}\right]$$

After the bit target for a picture is defined, we define the target bit number for each sub-frame Tr (ROI) and Tn (non-ROI) as follows:

$$Tr = \left(scale * (\frac{T}{total\_MBs})\right) * nROI$$

$T_n = T - T_r$

Where T is T_i or T_p depending on the picture type, total_MBs is the total number of MBs in a frame, nROI is the number of ROIs in the frame and scale is a scaling factor. In order to achieve higher quality of ROI MBs the scaling factor has to be greater than 1.This means that we allocate more bits per MB for ROIs which we "steal" from MBs that don't belong to ROIs in order to maintain the total *Picture Target Setting* and ensure low bitrate variability.

In order to achieve constant quality of ROIs, in terms of SNR, the scaling factor scale should be constant for all frames in a sequence (assign the same number of bits for every MB). Unfortunately, the number of MBs that are classified as ROIs is not constant. In addition, in order to achieve the high quality of ROIs the scaling factor scale must take a high value (greater than 2). As a result the scaling factor depends on the total number of MBs classified as ROIs (nROI). In our implementation the scaling factor for P-pictures is:

$$scale = \frac{total\_MBs * safety\_margin}{nROI'}$$

where safety_margin takes the value 0.9 and ensures that not all the bits of a frame are allocated to the ROI sub-frame and nROI' is given by the following equation:

$$nROI' = \left(\left\lfloor \frac{nROI}{15} \right\rfloor + 1 \right) * 15$$

For I-pictures the value of the scaling factor is constant and equal to 1.5 and the safety margin is set to 1. I pictures play an important role in the coding of successive frames, since predictions are made from them. Thus, we wish to maintain the quality of the whole I-picture in order to improve the coding of the following P-pictures.

After encoding all pictures in a GOP, any difference between the target and actual bit allocation is carried over to the next GOP as shown below.

After encoding a picture, Rem = Rem – (Sn+Sr)

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

Rem = G + Rem

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

respectively.

Where G is the total bits assigned to a GOP, N is the number of pictures in the GROUP OF PICTURES.

At the start of the sequence Rem = 0.

B. Rate Control

We define a virtual buffer for each sub-frame.
Before encoding macroblock j (j >= 1), the fullness of the appropriate virtual buffer is computed:

$$b\_f_j^{pt,m} = b\_f_0^{pt,m} + B_{j-1}^m - \left(\frac{T_{pt}^m * (j-1)}{MB\_cnt^m} \right)$$

depending on the picture type pt and whether the MB is ROI or non-ROI, m (m= r,n).

where

$b\_f_0^{pt,m}$ are initial fullnesses of virtual buffers - one for each picture type and MB type m (m = r : ROI or n : nonROI).

$B^m j$ is the number of bits generated by encoding all macroblocks of type m in the picture up to and including j.

MB_cnt^m is the number of macroblocks of type m in the picture. It is equal with nROI if m=r and with total_MBs-nROI if m=n.

$b\_f_j^{pt,m}$ are the fullnesses of virtual buffers at macroblock j- one for each picture type and MB type.

The final fullness of the virtual buffer ($b\_f_j^{pt,m}$: $j = MB\_cnt^m$) is used as $b\_f_0^{pt,m}$ for encoding the next picture of the same type.

Next the reference quantization scale QS^j_{REF} for macroblock j is computed. The modified encoder uses different equations for each type of quantization scales (linear and non-linear).

If the linear quantization scales are used (I-frames), then:

$$QS_{REF}^{j} = \frac{b\_f_j^{pt,m} * 31}{r}$$

else

$$QS_{REF}^{j} = \frac{b\_f_j^{pt,m} * 56}{r}$$

where r is the "reaction parameter" as defined in TM5.

The initial values for the virtual buffer fullness are:

$$b\_f_0^{I,r} = \frac{r}{31} \quad b\_f_0^{I,n} = 9 * \frac{r}{31}$$

$$b\_f_0^{P,r} = K_P * b\_f_0^{I,r} \quad b\_f_0^{P,n} = 9 * K_P * b\_f_0^{I,n}$$

The rate control step provides a reference quantization parameter to code each macroblock. Contrary to the TM5 algorithm, the reference quantization scale is not modulated with an activity factor that is determined from a measure of the spatial activity of the macroblock. This reference quantization scale is clipped to the allowed values and is actually used to quantize the inter MB.

If this is an intra coded MB, then the quantization scale code QP that corresponds to the QS that resulted is modified by the following equation:

$QP_{final} = \lfloor QP/3 \rfloor * 3 + pred$, where pred=0, 1or 2 depending on the MB selected for the intra frame prediction (above, left or virtual)

$TAD_{MB} = \sum_{i=0}^{255} \lvert cur[i] - prev[i] \rvert$
$MAD_{frame} = \frac{\sum_{i=0}^{\#MBs-1} TAD_i}{\#MBs}$
$TSD_{MB} = \sum_{i=0}^{255} (cur[i] - prev[i])^2$
$VAR_{MB} = \frac{\sum_{i=0}^{i=255} cur[i]^2}{256} - \left(\frac{\sum_{i=0}^{i=255} cur[i]}{256} \right)^2$

Table 5.1

5.1.2 Test Results

In these tests, the threshold T1 takes the value 1000, the threshold T2 the value 700 and T3 the value 20. Other important parameters that are common in both the modified and the original encoder are the q_scale_type which is set to linear for I-frames and non-linear for P-frames and the search window in the motion estimation step which is set to 4 MBs in both directions (horizontal and vertical).

Although I- frames cannot offer the coding efficiency of P-frames, it is important to have them in order to facilitate some functions such as random access or fast forward. In the test results both the Original and the Modified encoder use only I and P picture types and the distance of two consecutive I-frames is 25 (M=1, N=25). The values of the most important parameters of the encoding process are shown in table 5.2.

T1 : ROI decision threshold	1000
T2 : No-MC decision threshold	700
T3 : Skipped MB decision threshold	20
Sxf : Horizontal Search Window	4
Syf : Vertical Search Window	4
N : # of pictures in GOP	25
M : distance between P frames	1 (no B-frames allowed)

Table 5.2

The video sequences used in our tests are from the EC Funded CAVIAR project/IST 2001 37540 and can be found at http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/.

These sequences include indoor shopping centre observations of individuals and small groups of people walking, browsing, window shopping, meeting, entering and exiting

shops, etc. The resolution of all the test sequences is 384x288 and the frame rate is 25 frames/sec.

For the tests we used 4 video sequences with different activity. We measure activity with the average number of MBs per frame classified as ROIs. In our tests, we include one sequence with average activity (16 MBs per frame), one sequence with zero activity (0 MBs per frame), one sequence with low activity (12 MBs per frame), and one with high activity (25 MBs per frame).

In the next section, 5.1.2.1, we present the improvement achieved by the Modified MPEG-2 encoder in terms of quality and efficiency over the Original MPEG-2 encoder. First, we present the results from encoding at 370kbps with the Modified and the Original encoder and exhibit the quality improvement introduced by the Modified encoder. Then we encode using the Modified encoder at a lower bit rate that results to the same ROI quality as the frame quality of the modified encoder at 370kbps to exhibit the improvement in coding efficiency of ROIs. Finally we encode the test sequences at 180kbps and 300kbps with the Modified encoder to exhibit its stable behaviour.

Quality is measured in SNR (db) and is given by:

$$SNR_{db} = 10\log(\frac{VAR}{MSE})$$

where

VAR is given in table 5.1 in page 55 and

$$MSE_{MB} = \frac{\sum_{i=0}^{i=255}(org[i] - rec[i])^2}{256}$$

In section 5.1.2.2 we present the improvement in the speed of the encoding process and finally in section 5.1.2.3 some conclusions.

5.1.2.1 Quality and Coding Efficiency

EnterExitCrossingPaths1cor.mpeg

This is a test sequence with average activity. The average number of MBs classified as ROIs is 16.0 MBs per frame. The total number of frames in the sequence is 383 and the size of the uncompressed video sequence is 63535104 bits.

The encoding results of both the modified and the original MPEG-2 encoder at a bit-rate of 370kbps are shown in table 5.3.

	Total Bits	Comp. ratio	Average SNR (db)				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	5668559	1:11.2	20.97	26.5	6.48	10.1	242.14
Original Enc.	5669724	1:11.2	18.04	18.6	6.34	9.35	204.12

Table 5.3: Encoding results at 370Kbps

From table 5.3 we notice that the Modified encoder has higher average quality, in terms of SNR than the Original encoder. The SNR of the Y component of the whole frame is higher by 16.25% and the SNR of the Y component of the MBs classified as ROIs by 42.4%. The quality improvement for the other two components (Cb, Cr) is 2% and 8% respectively.

Moreover, the quality of ROIs of the modified encoder is 26.4% higher than the quality of the whole frame.

Finally, the average number of skipped MBs per frame in the Modified MPEG-2 encoder is higher than the original encoder by 18.6%. Since we allocate fewer bits per MB to non-ROIs than ROIs higher quantization scales (QS) are chosen for non-ROI MBs. In combination with the fact that non-ROIs MBs most likely have small difference from the MBs at the same position in the previous frame, it is obvious that the quantization of the prediction error will probably have value 0, which will lead to the skipping of more MBs than in the case of the Original encoder, where more bits per MB are allocated and thus lower QSs are chosen.

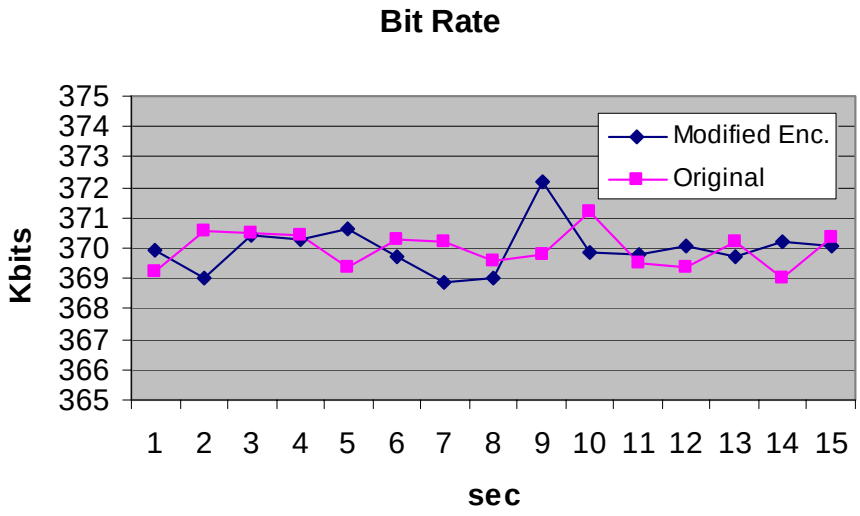


Figure 5.2: Bitrate at 370kbps

As the rate control algorithm is an extension of the existing rate control algorithm used in the original MPEG-2 encoder we expect that the bitrate of the Modified encoder will be similar to the bitrate of the original encoder. However, because of the the way we scale the quantization matrices, as explained in 5.1.4, we see improvement in the bit rate variability. This improvement is more visible in the next tests. From figure 5.2, we can see that the Modified encoder has slightly higher bit-rate variability.

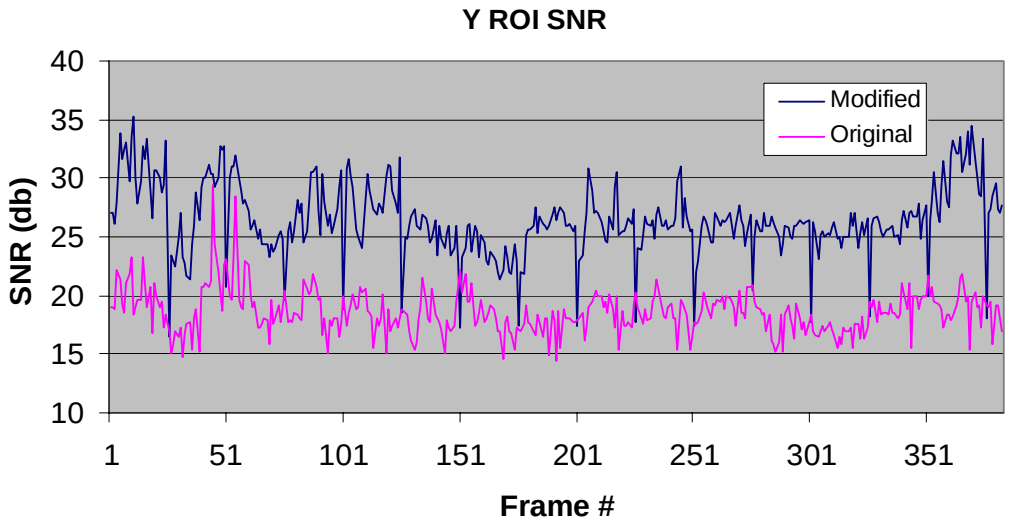


Figure 5.3: Signal to Noise ratio of Y component of ROIs at 370kbps

As seen in figure 5.3 some frames have a very low ROI-SNR compared to the rest frames. These are the anchor frames (I-frames). In these frames all the MBs are intra-coded and as a result are not coded as efficiently as MBs in P-frames. In addition, we don't want to allocate more bits to ROIs because the quality of non-ROI MBs will drop significantly and the quality of the next frames will be affected.

The variability in the quality is mainly due to the method used to detect the ROIs. The number of MBs that belong to a ROI in successive frames is expected to be the same. However, the TAD is not a very reliable measure to detect ROIs and two successive frames might have a very different number of ROI MBs.

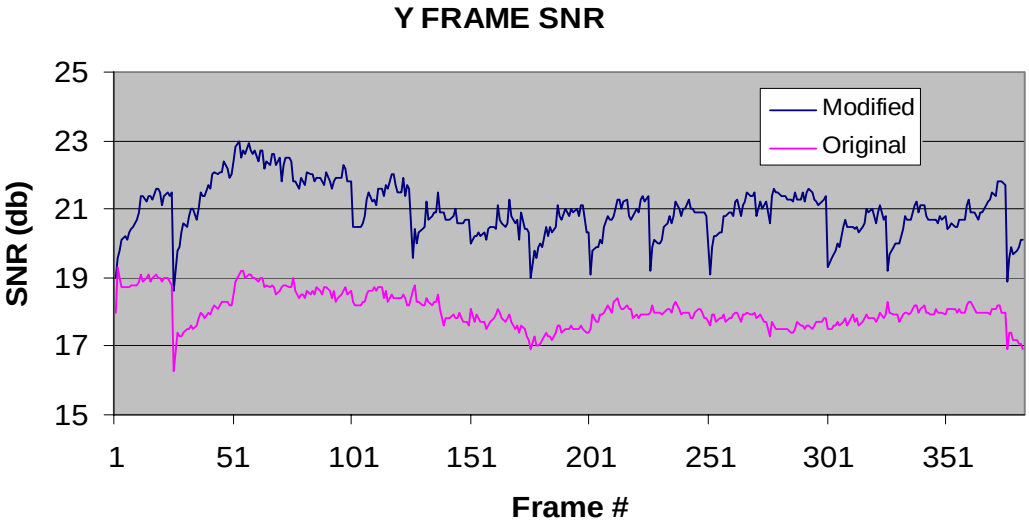


Figure 5.4: Signal to Noise ratio of Y component at 370kbps

At a bitrate of 160kbps the modified encoder can achieve almost the same quality of the ROIs as the original encoder at 370kbps, as seen in the next table (5.4).

	Total Bits	Comp. ratio	Average SNR (db)				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	2474749	1:25.67	17.23	19.05	5.4	8.2	342.18
Original Enc.	5669724	1:11.2	18.03	18.60	6.34	9.35	204.12

Table 5.4: Encoding results with same ROI quality

The quality of the Y component of the whole frame, achieved by the Modified encoder, is 4% lower than the quality achieved by the Original encoder, while for the Cb, Cr components lower by 14% and 12% respectively. We can achieve approximately the same quality of ROIs with the frame quality of the Original encoder with 56.35% bits less than the original encoder.

Moreover, the quality of ROIs of the Modified encoder is 10.56% higher than the quality of the whole frame.

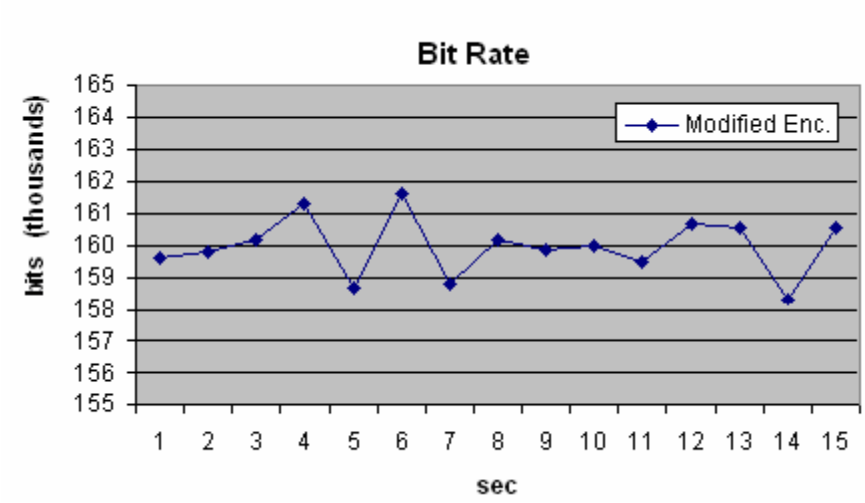


Figure 5.5: bitrate of Modified encoder at 160kbps

The bit rate variability of the modified encoder remains good (fig 5.5).

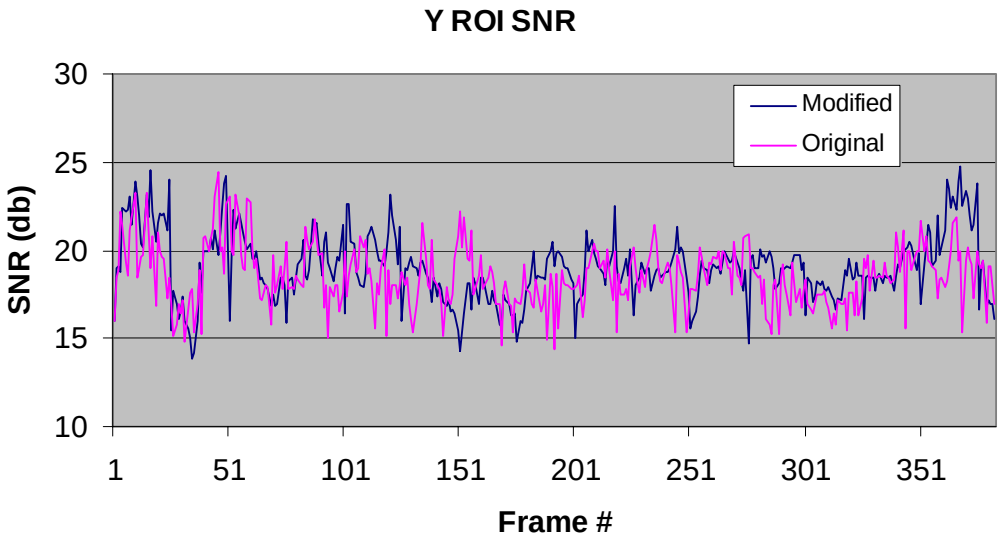


Figure 5.6: SNR of Y component of ROIs. Modified enc. at 160kbps, Original Enc. at 370kbps.

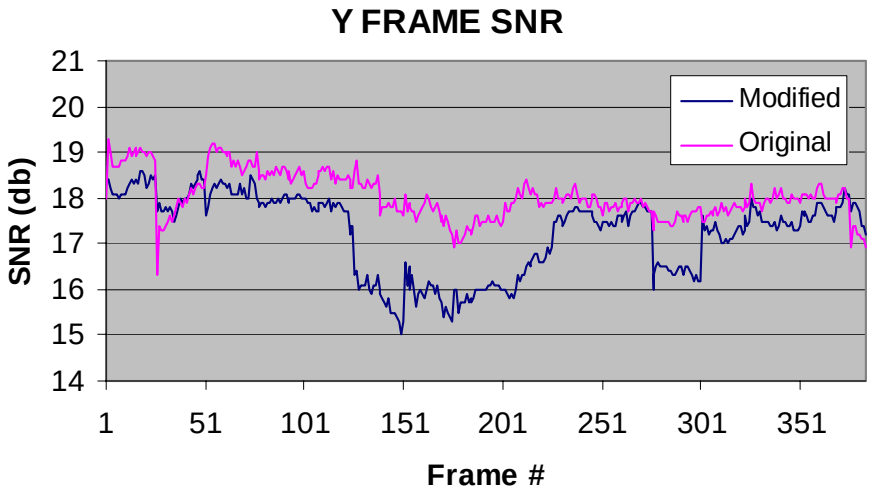


Figure 5.7: SNR of Y component. Modified enc. at 160kbps, Original Enc. at 370kbps.

The low frame quality of the Modified encoder from frame 121 till 241 and 281 till 310 is a result of the many MBs classified as ROIs compared to the rest frames. Since the bit rate of the Modified encoder is very low the deterioration in the quality of the frame is more visible than in higher bit rates.

In the next section we show the encoding results at 300kbps and 180kbps.

	Total Bits	Comp. ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
300kbps	4596268	1:13.8	19.99	24.6	6.13	9.53	271.96
180kbps	2776361	1:22:88	17.65	20.1	5.45	8.32	333.76

Table 5.5: Encoding results of Modified Encoder at 300Kbps and 180kbps

At 300kbps the quality of ROIs is higher by 23% from the quality of the frame, while at 180kbps higher by 13.88%.

From the next figures (5.8, 5.9, 5.10, 5.11) it is clear that the Modified encoder maintains a stable behavior.

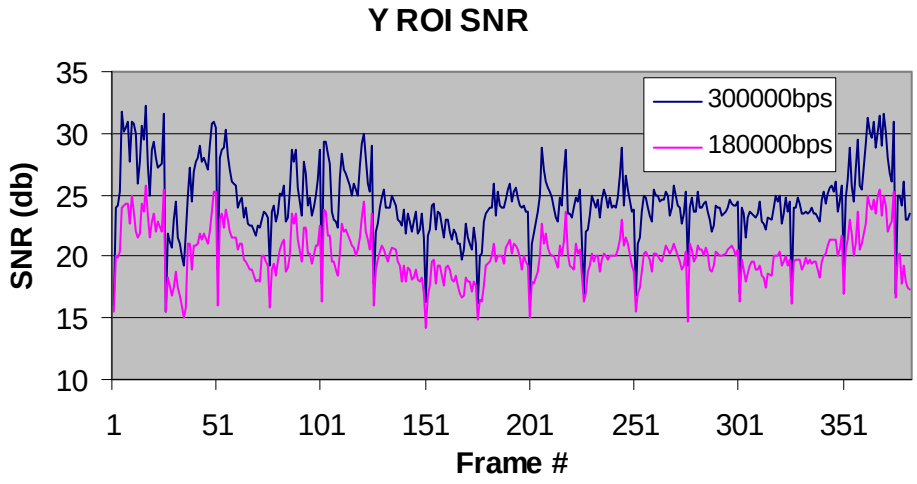


Figure 5.8: SNR ratio of Y component ROIs. Modified enc. at 180kbps and 300kbps.

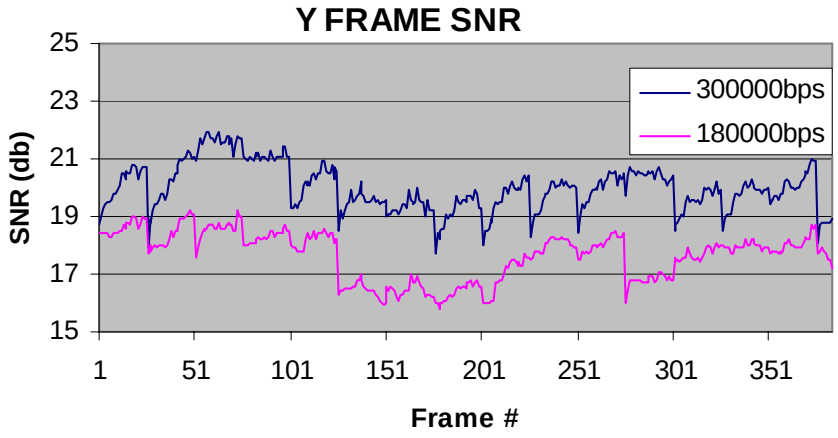


Figure 5.9: SNR of Y component. Modified enc. at 160kbps, and 300kbps.

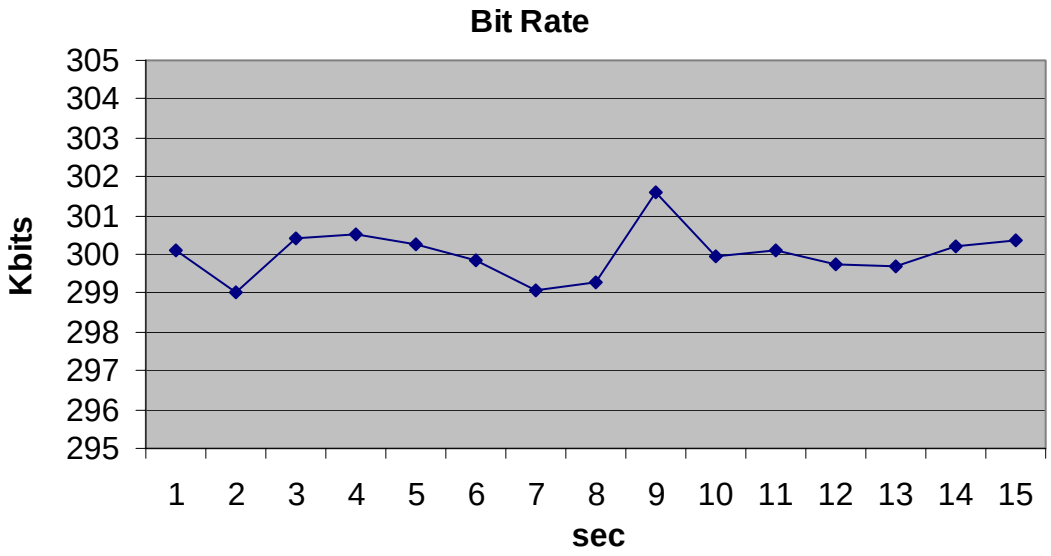


Figure 5.10: Bitrate of Modified Encoder at 300kbps

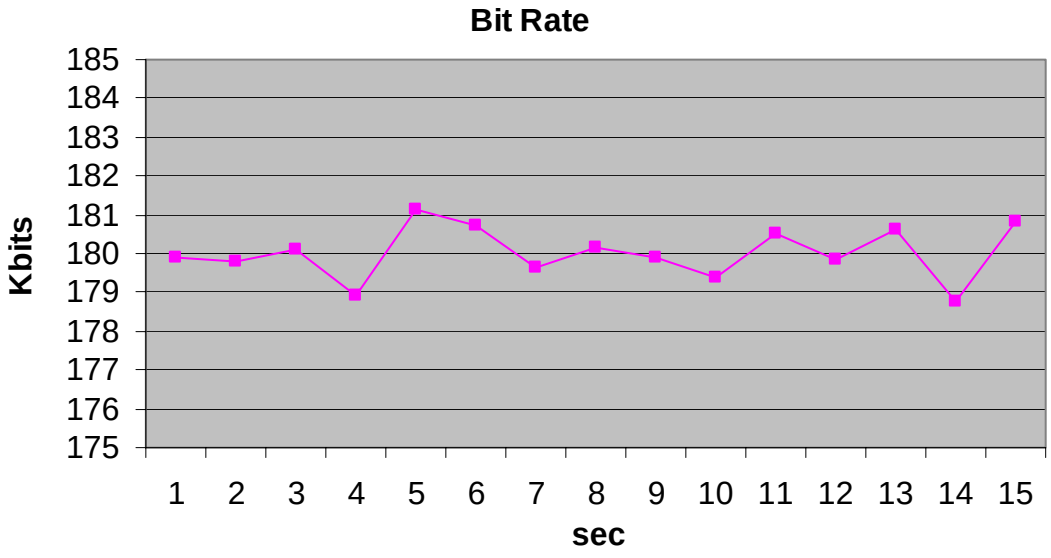


Figure 5.11: Bitrate of Modified Encoder at 180kbps

We can see that the encoder exhibits a stable behavior both in terms of quality and in terms of bitrate variability.

ThreePastShop2front.mpeg

This is a test sequence with no activity. The average number of MBs classified as ROIs is 0.0 MBs per frame. The total number of frames in the sequence is 500 and the size of the uncompressed video sequence is 82944000 bits.

The encoding results of both the modified and the original MPEG-2 encoder at a bit-rate of 370kbps are shown in table 5.6.

	Total Bits	Comp. ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	7399691	1:11.2	22.1	-	11.3	14.9	169.76
Original Enc.	7400065	1:11.2	19.1	-	10.2	13.4	213.78

Table 5.6: Encoding results at 370kbps

The SNR of the Y component of the whole frame, achieved by the Modified encoder, is 15.7% higher from the quality of the whole frame achieve by the Original encoder. The quality improvement for the chrominance components is 10.8% and 11.2% respectively.

Notice that the average number of skipped MBs per frame of the Modified encoder is 20% **less** than the average skipped MBs per frame of the Original encoder. This happens because there are no ROIs and the Modified encoder is much more efficient in compressing the sequence. Since there are no ROIs all the bits of a frame are allocated to the non-ROIs sub frame. In addition, thanks to the higher compression efficiency of the Modified MPEG-2 encoder lower quantization scales (QS) are chosen. On the other hand the Original encoder chooses high QSs. For some MBs, that have a high prediction error, the quantization with the QS chosen by the Original encoder will lead to all zero coefficients (skipped MB), something that will not happen with the lower QSs chosen by the Modified encoder.

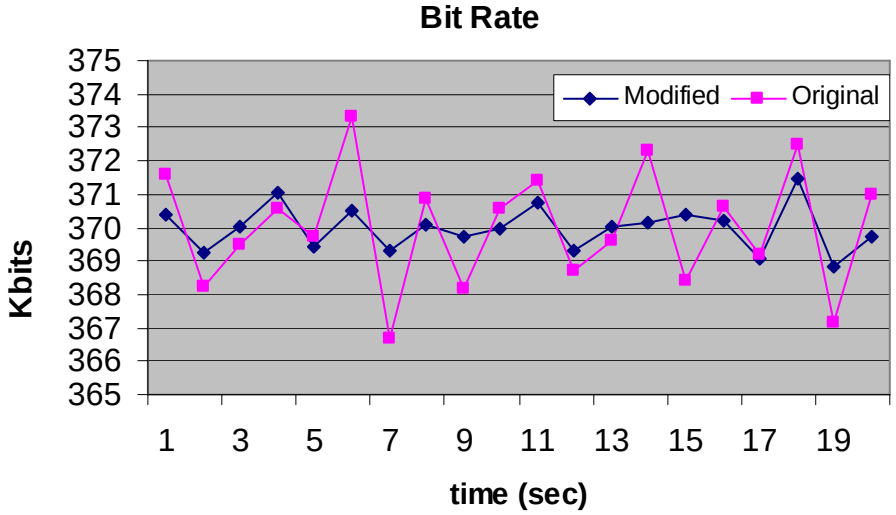


Figure 5.12: Bitrate at 370kbps

From the above figure (5.12), we can see that the modified encoder has much better bit-rate variability.

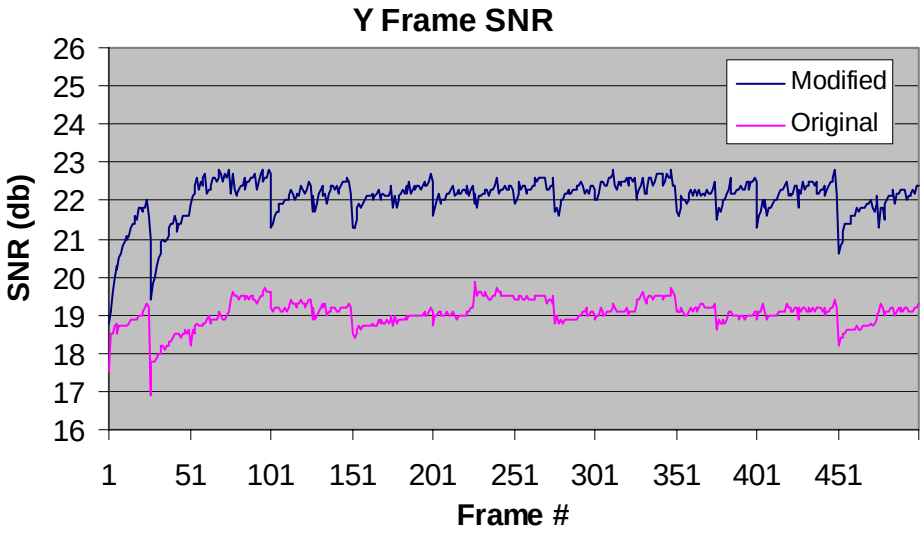


Figure 5.13: SNR of Y component at 370kbps

At a bitrate of 170kbps the modified encoder can achieve almost the same quality as the original encoder at 370kbps.

	Total Bits	Comp. ratio	Average SNR (db)				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	3400305	1:24.39	19.2	-	9.52	12.9	313.93
Original Enc.	7400065	1:11.2	19.1	-	10.2	13.4	213.78

Table 5.7: Encoding results at same quality

The quality of the Y component of the whole frame of the Modified encoder is 0.05% higher than that of the Original encoder, while for the chrominance components lower by 6.6% and 3.7% respectively. We save approximately 54% bits.

In this case, the bitrate of the Modified encoder is much lower than that of the Original encoder. Thus, it is natural for the Modified Encoder to skip much more MBs than the Original Enoder, since high quantization scales are selected.

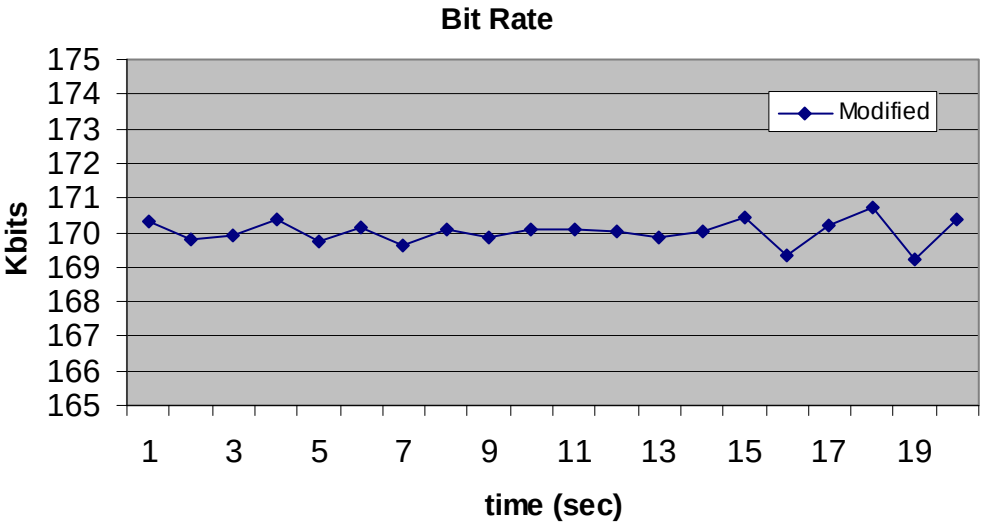


Figure 5.14: Bitrate of modified encoder at 170kbps

The bit rate variability of the Modified encoder is very low (fig. 5.14),

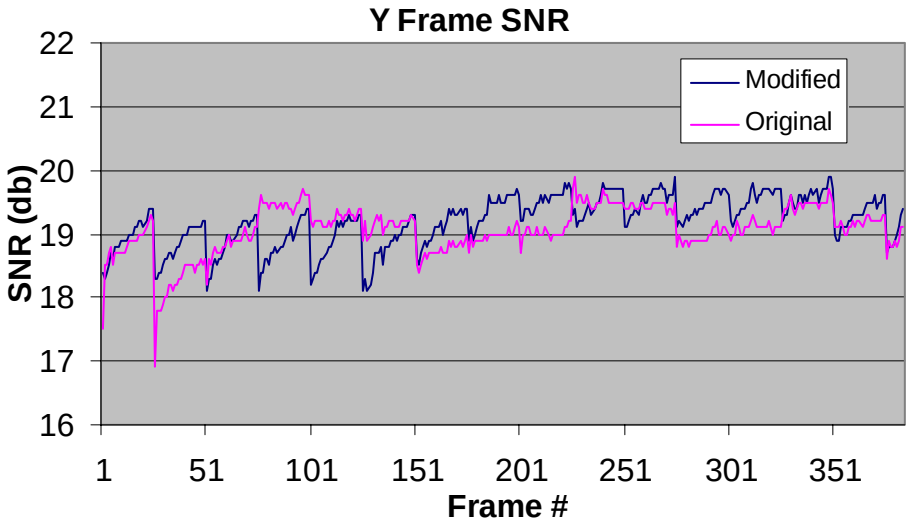


Figure 5.15: SNR of Y component. Modified enc. at 170kbps, Original Enc. at 370kbps.

In the next table we show the encoding results at 300kbps and 180kbps.

	Total Bits	Compression ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
300kbps	5999687	1:13.8	21.3	-	10.8	14.4	214.96
180kbps	3600141	1:23.03	19.4	-	9.6	13.1	305.63

Table 5.8: Encoding results at 300kbps and 180kbps

Again from figures 5.16, 5.17, 5.18 we see that the Modified encoder presents a stable behavior in terms of quality and bitrate variability.

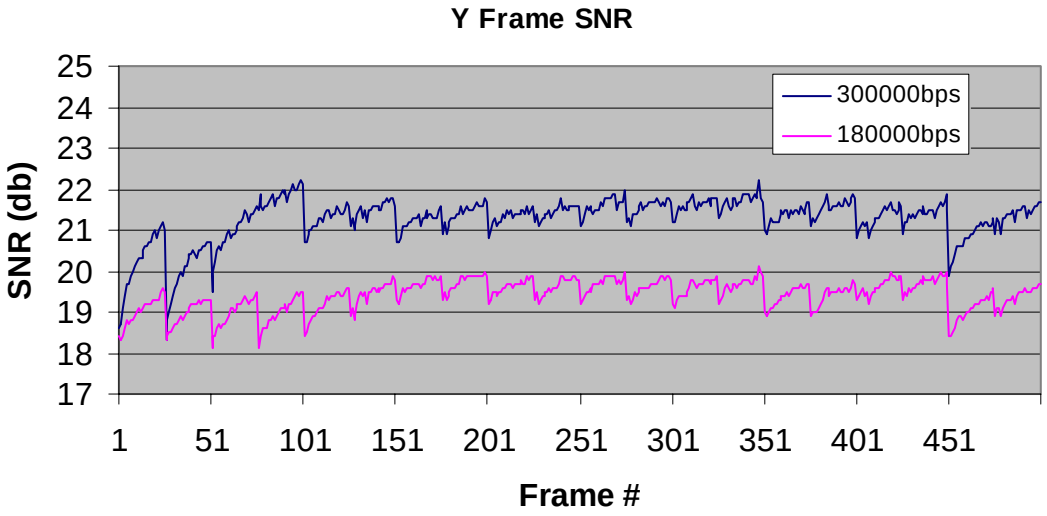


Figure 5.16: SNR of Y component. Modified enc. at 180kbps and 300kbps.

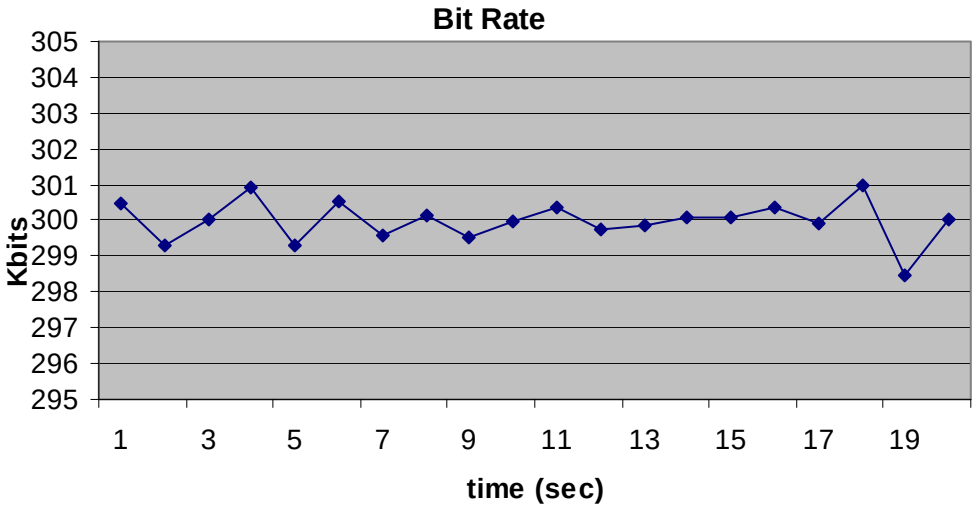


Figure 5.17: Bitrate of modified encoder at 300kbps

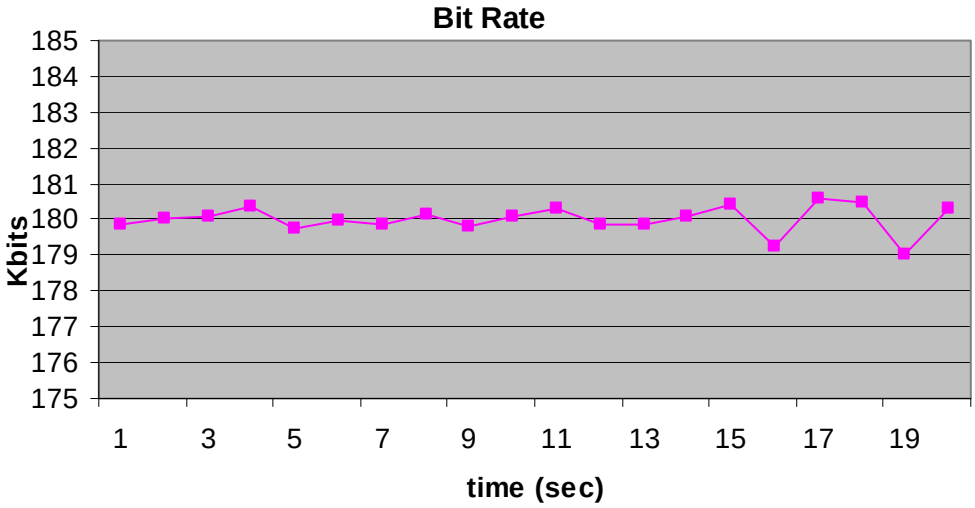


Figure 5.18: Bitrate of modified encoder at 180kbps

TwoEnterShop2front.mpeg

This is a test sequence with low activity. The average number of MBs classified as ROIs is 12.0 MBs per frame. The total number of frames in the sequence is 375 and the size of the uncompressed video sequence is 62208000 bits.

The encoding results of both the modified and the original MPEG-2 encoder at a bit-rate of 370kbps are shown in table 5.9.

	Total Bits	Comp. ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	5550933	1:11.2	20.3	27.2	9.99	13.7	250.98
Original Enc.	5550586	1:11.2	17.8	17.2	9.54	12.8	215.03

Table 5.9: Encoding results at 370kbps

From the above table we notice that the Modified encoder has higher average quality, in terms of SNR. The SNR of the Y component of the frame is higher by 14.04% and the SNR of the Y component of the MBs classified as ROIs by 58.1%. The quality improvement for the other two components (Cb, Cr) is 4.7% and 7% respectively.

Moreover, the quality of ROIs of the Modified encoder is 33.9% higher than the quality of the whole frame.

Notice that in this sequence, that has ROIs, the average number of skipped MBs per frame of the Modified encoder is higher by 16% from the average number of skipped MBs per frame of the Original encoder at the same bit rate.

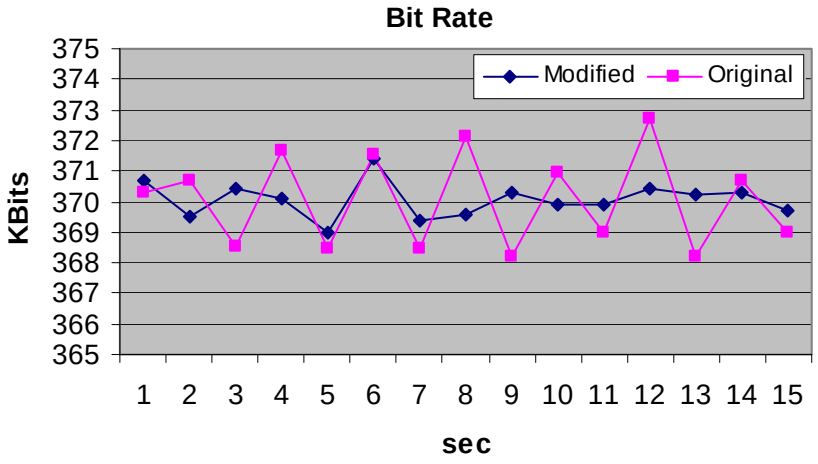


Figure 5.19: Bitrate at 370kbps

From the above figure (5.19), we can see that the modified encoder has better bit-rate variability. This improvement is mainly due to the scaling of the quantization matrices. By computing the MAD of the current frame from the previous, we get a measure of the change in the complexity of the current frame from the previous. The higher the MAD, the higher the complexity change and the more likely it is that the rate control algorithm will fail. By using larger quantization tables we avoid undesirable spikes in the bitrate.

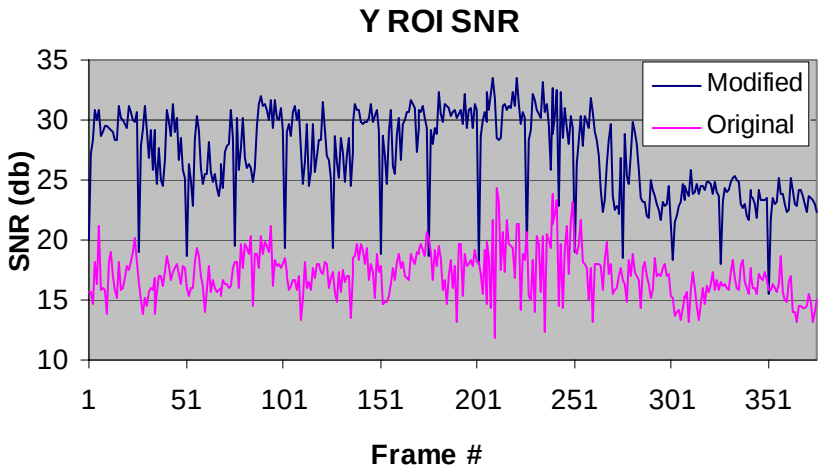


Figure 5.20: SNR of Y component Regions of Interest at 370kbps

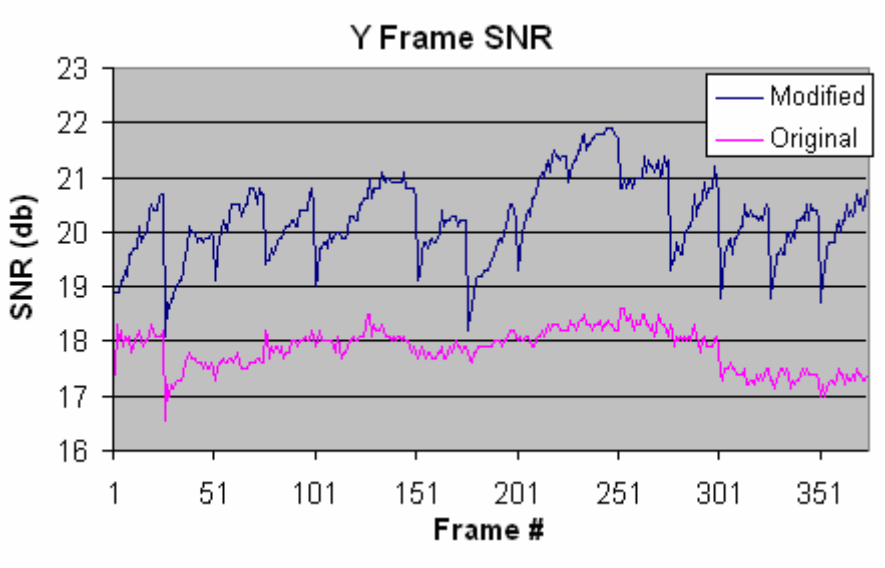


Figure 5.21: SNR of Y component at 370kbps

The variability in the quality of the hole frame we see in figure 5.21 is a result of the variable number of ROIs detected from frame to frame.

At a bitrate of 150kbps the modified encoder can achieve almost the same quality of the ROIs as the original encoder at 370kbps.

	Total Bits	Comp. ratio	Average SNR (db)				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modfied Enc.	2248823	1:27.66	17.7	17.9	8.69	12.3	359.83
Original Enc.	5550586	1:11.2	17.8	17.2	9.54	12.8	215.03

Table 5.10: Encoding results with same quality

The quality of the Y component of the whole frame of the Modified encoder is lower by 0.05% than that of the Original encoder, while for the Cb, Cr components lower by 8.9% and 3.9% respectively.

We can achieve the same quality of ROIs with 59.48% bits less than the original encoder, without significant deterioration in the quality of the whole frame.

Moreover, the quality of ROIs of the modified encoder is only 1.1% higher than the quality of the whole frame. The Modified encoder doesn't achieve much higher quality of ROIs than the whole frame, but at least it manages to keep the quality of ROIs higher, something that the Original encoder fails to do.

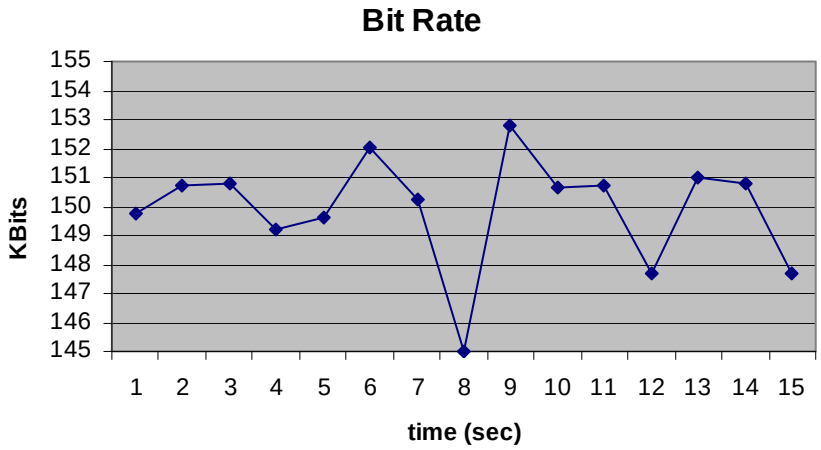


Figure 5.22: Bitrate of Modified encoder at 150kbps

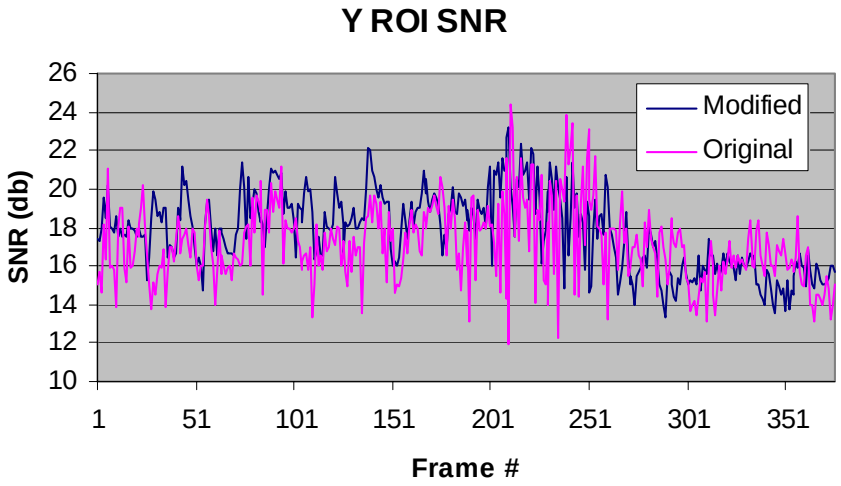


Figure 5.23: SNR of Y component of ROIs. Modified enc. at 150kbps, Original Enc. at 370kbps.

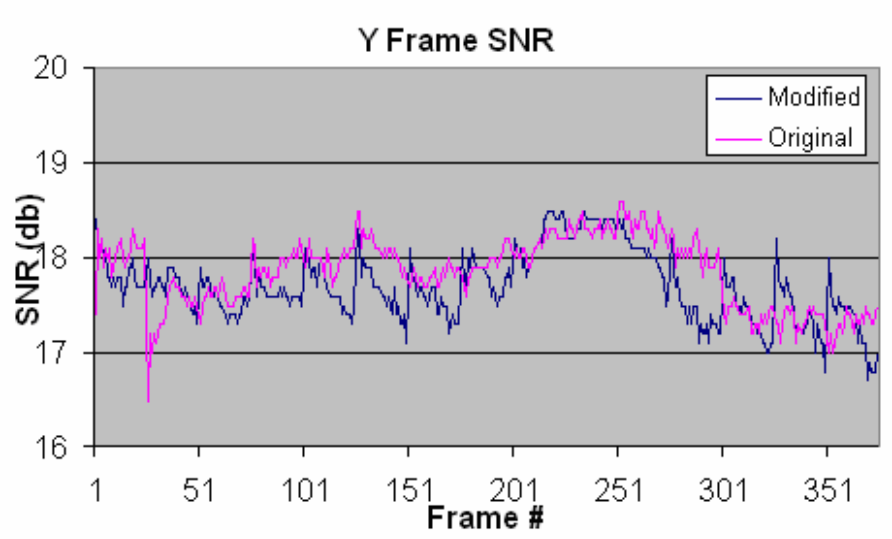


Figure 5.24: SNR of Y component. Modified enc. at 150kbps, Original Enc. at 370kbps.

In the next table we show the encoding results at 300kbps and 180kbps.

	Total Bits	Comp. ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
300kbps	4501141	1:13.8	19.4	25.2	9.47	13	279.36
180kbps	2700404	1:23.03	18.2	19.8	8.81	12.4	342.21

Table 5.11: Encoding results at 300kbps and 180kbps

At 300kbps the quality of ROIs is higher by 29.8% from the quality of the whole frame, while at 180kbps higher by 8.8%.

Figures 5.25 to 5.28 exhibit the stable behavior of the Modified encoder at 300 and 180kbps.

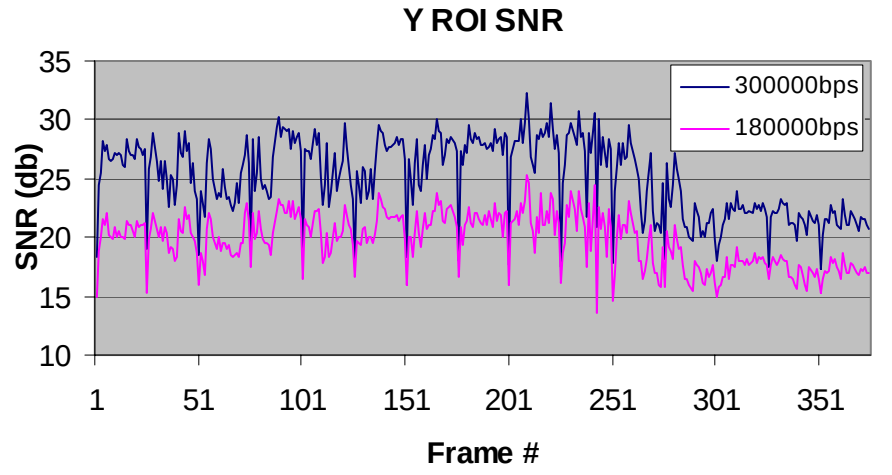


Figure 5.25: SNR of Y component of ROIs. Modified enc. at 180kbps and 300kbps.

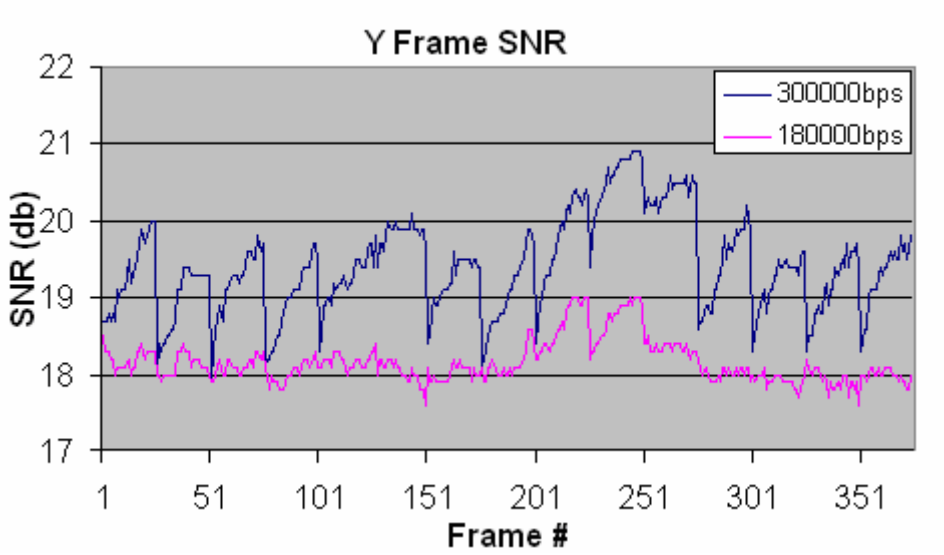


Figure 5.26: SNR of Y component. Modified enc. at 180kbps, and 300kbps.

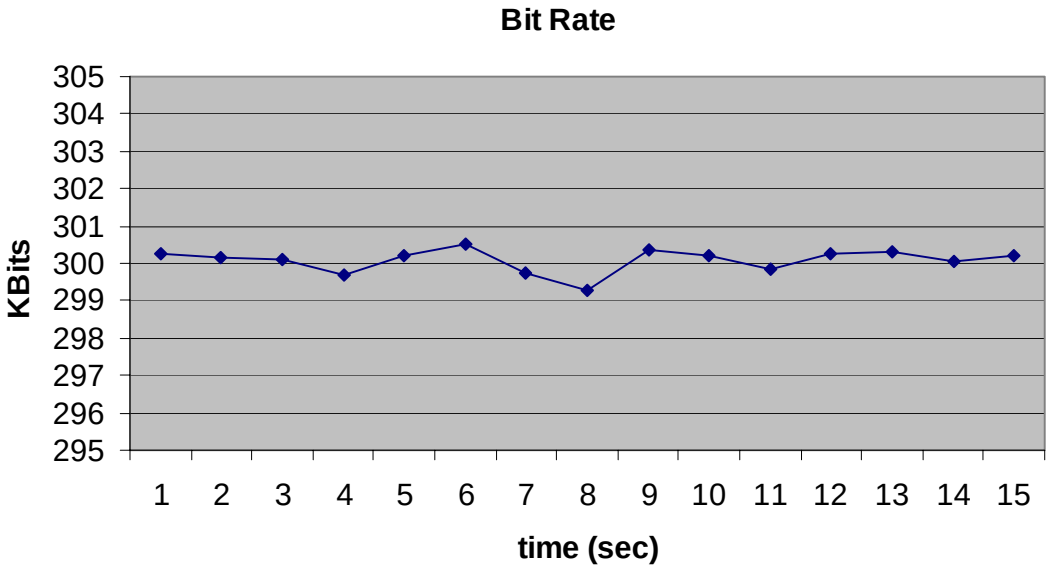


Figure 5.27: Bitrate of Modified encoder at 300kbps

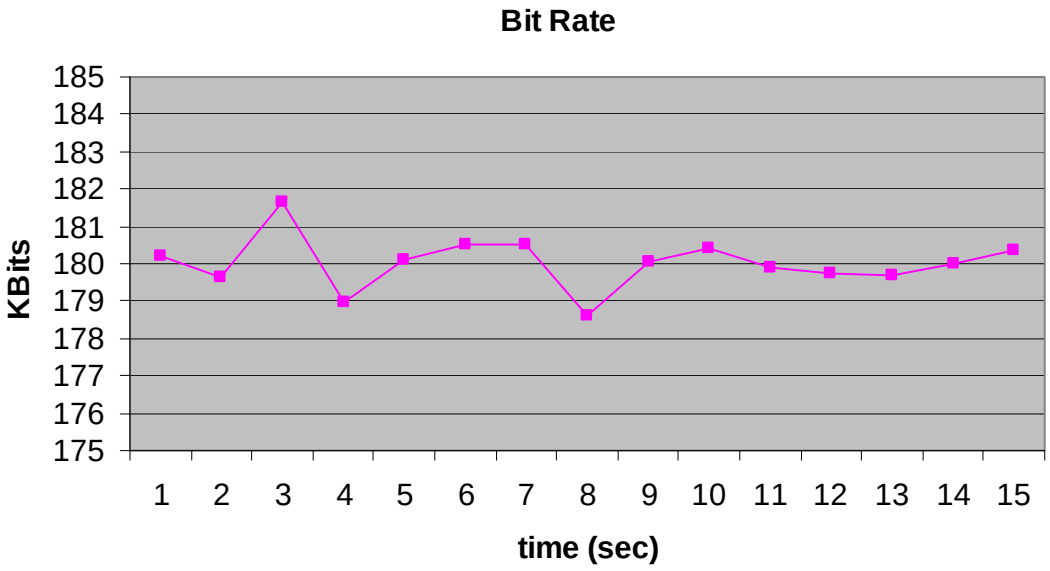


Figure 5.28: Bitrate of Modified encoder at 180kbps

TwoEnterShop1cor.mpeg

This is a test sequence with high activity. The average number of MBs classified as ROIs is 25.0 MBs per frame. The total number of frames in the sequence is 625 and the size of the uncompressd video sequence is 103680000 bits.

The encoding results of both the modified and the original MPEG-2 encoder at a bit-rate of 370kbps are shown in table 5.12.

	Total Bits	Comp. ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	9249404	1:11.2	20.8	25.6	6.68	9.82	246.13
Original Enc.	9250401	1:11.2	18.7	18.6	6.62	9.3	193.38

Table 5.12: Test results at 370kbps

From table 5.12 we notice that the Modified encoder has higher average quality, in terms of SNR. The SNR of the Y component of the frame is higher by 11.2% and the SNR of the Y component of the MBs classified as ROIs by 37.6%. The quality improvement for the other two components (Cb, Cr) is 0.09% and 5.6% respectively.

Moreover, the quality of ROIs of the Modified encoder is 23.1% higher than the quality of the whole frame.

Finally, the average number of skipped MBs per frame of the Modified encoder is 27% higher than the average number of skipped MBs per frame of the Original encoder.

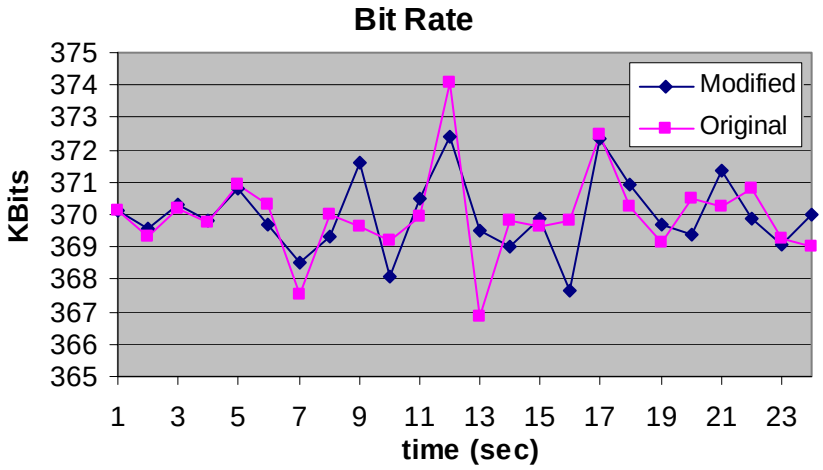


Figure 5.29: Bitrate at 370kbps

From the above figure (5.29), we can see that the Modified encoder has better bitrate than the Original encoder, as it presents lower spikes. The important is to avoid exceeding the desired input (370kbps) by many bits, especially for applications that involve transmission of the video through channels with limited bandwidth.

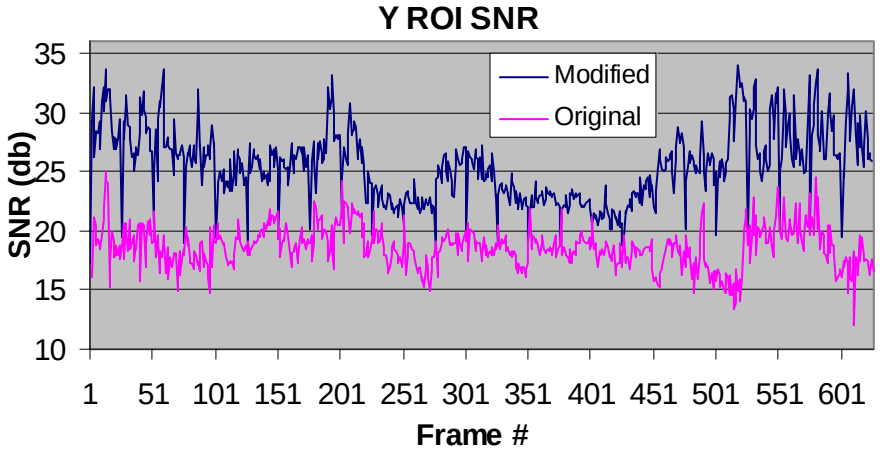


Figure 5.30: Signal to Noise ratio of Y component of ROIs at 370kbps

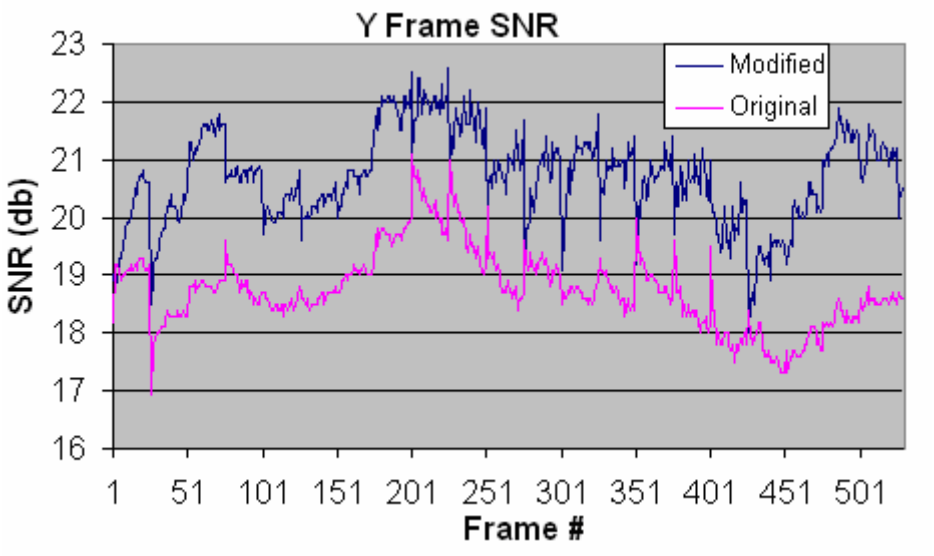


Figure 5.31: Signal to Noise ratio of Y component at 370kbps

At a bitrate of 170kbps the modified encoder can achieve almost the same quality of the ROIs as the original encoder at 370kbps.

	Total Bits	Comp. ratio	Average SNR (db)				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
Modified Enc.	4249424	1:24.39	17.3	19.2	5.72	8.2	334.57
Original Enc.	9250401	1:11.2	18.7	18.6	6.62	9.3	193.381

Table 5.13: Encoding results at same quality

The quality of the Y component of the whole frame of the Modified encoder is lower by 7.4% from that of the Original encoder, while for the Cb, Cr components lower by 13.6% and 11.8% respectively. We can achieve the same quality of ROIs with 54.1% fewer bits than the Original encoder.

Moreover, the quality of ROIs of the modified encoder is 10.9% higher than the quality of the whole frame.

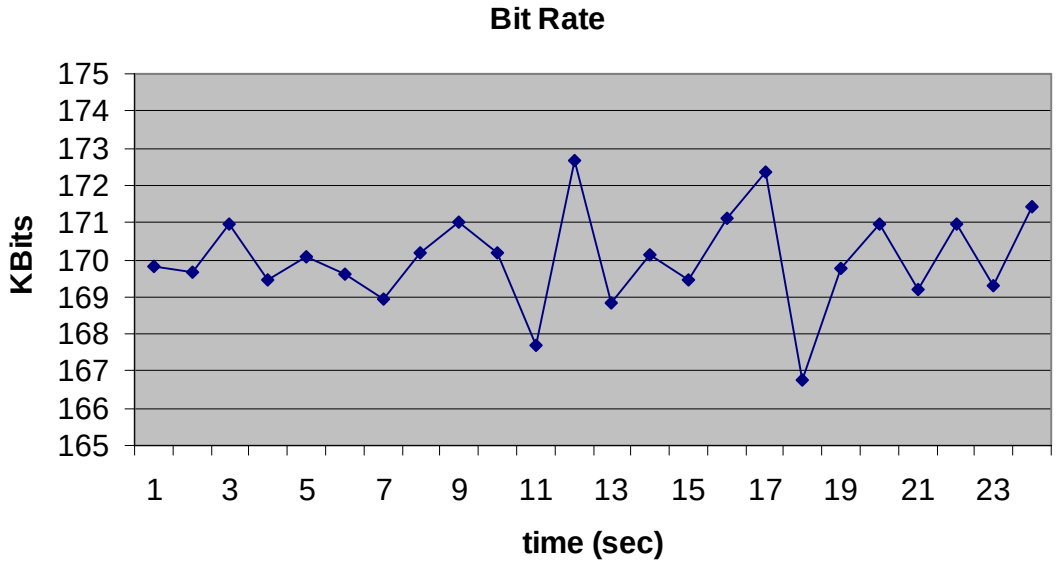


Figure 4.32: Bitrate of modified encoder at 170kbps

The bit rate of the Modified encoder presents high bit rate variability. This variability is a result of the characteristics of the video, which presents sudden changes from periods with low activity to periods with high activity. However, the Modified encoder still does better than the original encoder, just like in the case of 370kbps (fiure 5.29).

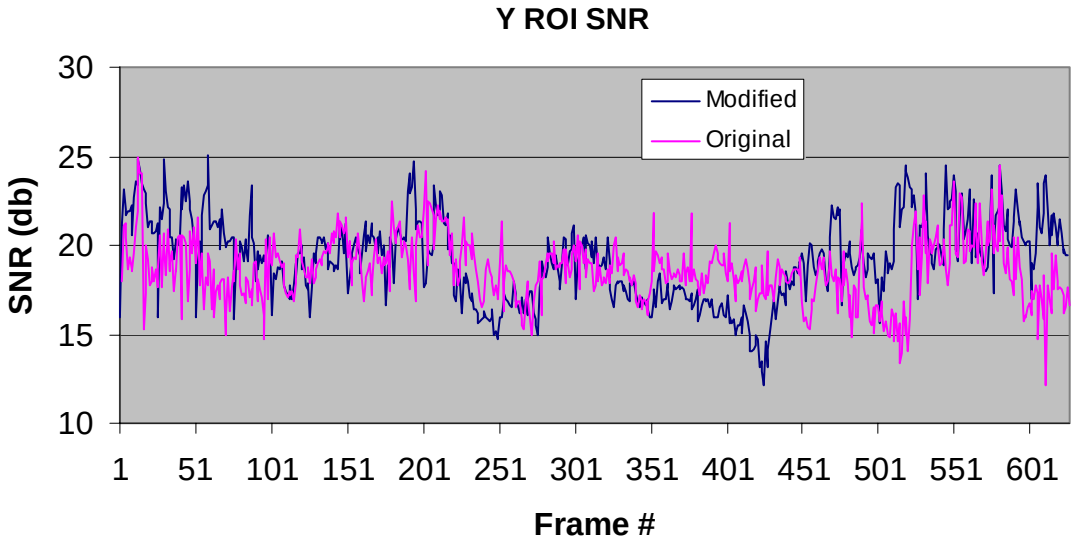


Figure 5.33: SNR of Y component of ROIs. Modified enc. at 170kbps, Original enc. at 370kbps.

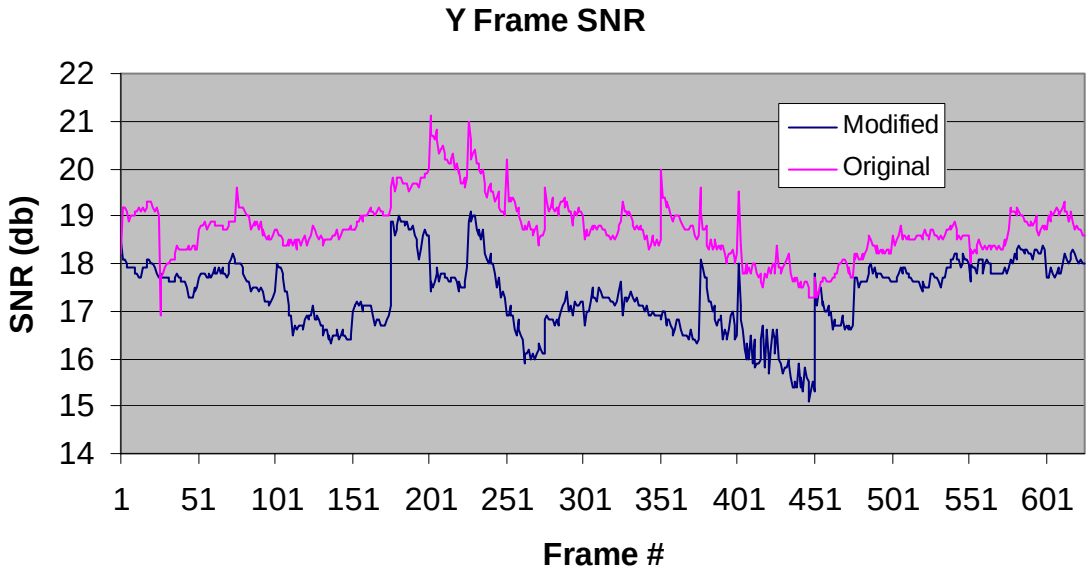


Figure 5.34: SNR of Y component. Modified enc. at 170kbps, Original Enc. at 370kbps.

In the next section we show the encoding results at 300kbps and 180kbps.

	Total Bits	Comp. ratio	Average SNR				Average Skipped MBs
			Y	Y- ROI	Cb	Cr	
300kbps	7499728	1:13.8	19.9	23.9	6.38	9.34	273.77
180kbps	4499651	1:23.04	17.5	19.8	5.74	8.24	328.72

Table 5.14: Encoding results at 300kbps and 180kbps

At 300kbps the quality of ROIs is higher by 20.1% from the quality of the whole frame, while at 180kbps higher by 13.1%.

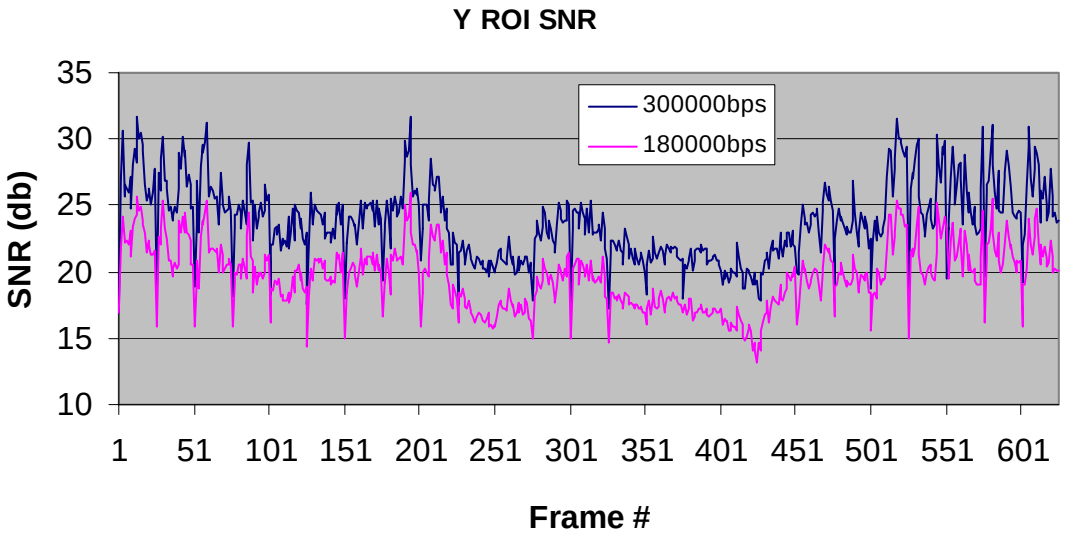


Figure 5.35: SNR of Y component of ROIs. Modified enc. at 180kbps and 300kbps.

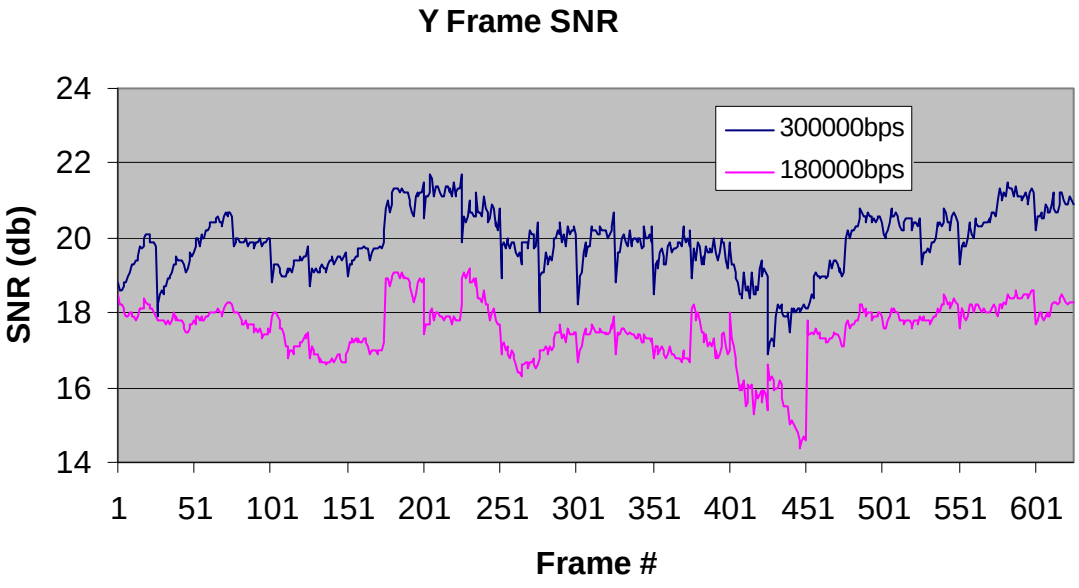


Figure 5.36: SNR of Y component. Modified enc. at 180kbps and 300kbps.

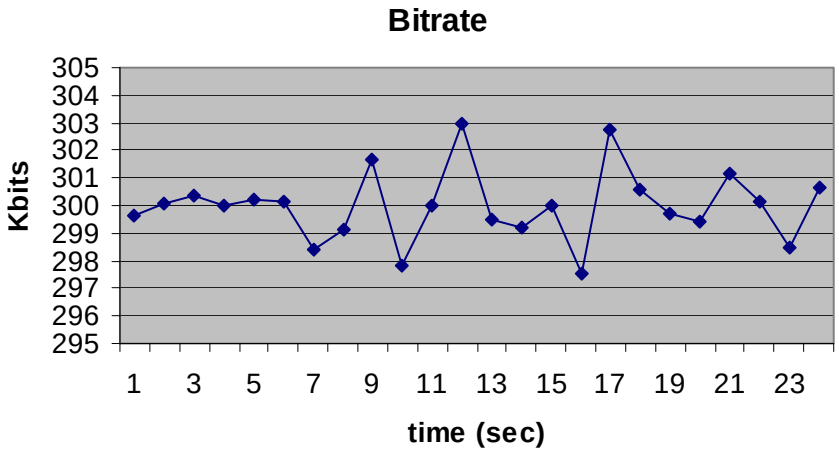


Figure 5.37: Bitrate of Modified encoder at 300kbps

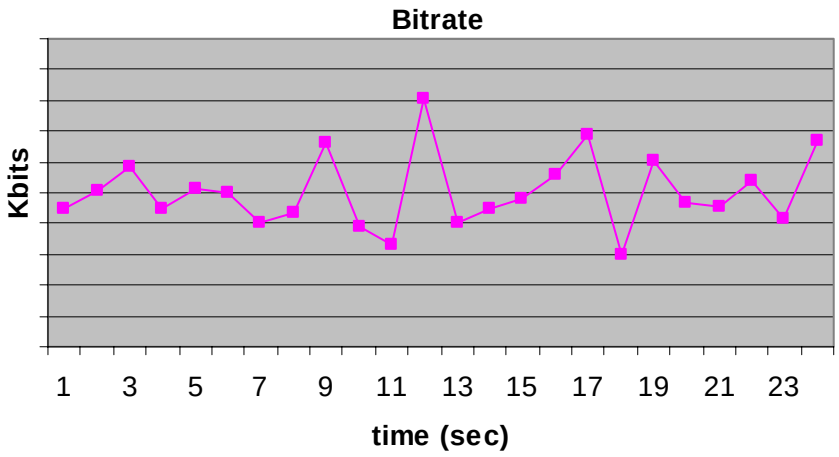


Figure 5.38: Bitrate of Modified encoder at 180kbps

5.1.2.3 Encoding Speed

For the tests of the encoding speed we used the same parameters as described at the beginning of section 5.1.2 (table 5.2). We compressed three of the test sequences used in the previous section with both encoders several times and kept the fastest time. We measured actual time to encode the test sequences; the time to read the various parameters required for the encoding process is not included. The time to read the input video from the hard disk and to store the compressed sequence back to the hard disk is included in our measurements.

The tests where performed on a general purpose desktop PC with an Intel Penitum4 at 2.53GHz processor with no special hardware for video compression.
The results are presented in the next table

Test sequence	#frames	Modified Enc.	Original Enc.
EnterExitCrossingPaths1cor.mpeg	383	39386 msec	102529 msec
TwoEnterShop2front.mpeg	375	39397 msec	101548 msec
TwoEnterShop1cor.mpeg	625	80351 msec	169626 msec

Table 5.15: Encoding delay results

In order to encode a sequence with frame rate 25 frames/sec in real time, an encoder should be able to encode a frame in maximum 40msec. The original encoder requires, in average, 269msec/frame while the modified encoder 110msec/frame.

The modified encoder results to a 59% speedup of the encoding process. However, both encoders fail to encode a video of 25fps at 384x288 in real time.

5.1.2.4 Conclusions

From the above tests we conclude that the Modified encoder is much more efficient than the Original encoder, both in terms of coding efficiency and encoding delay. At the same bit rate it can achieve higher ROI and frame quality than the Original encoder, while with approximately 56% fewer bits it can achieve ROI quality comparable to the frame quality of the Original encoder. At the same time it keeps an acceptable frame quality. Moreover, the Modified encoder achieves higher quality of ROIs than the whole frame in all cases, while the Original encoder not.

Finally, the modified encoder has lower bit rate variability but higher quality variability than the original encoder. The variability in the quality is mainly a result of the variable number of ROIs detected from frame to frame.

Regarding the encoding delay, both encoders fail to encode a sequence of 25fps frame rate and 384x288 resolution in real time on a common PC. However, the Modified encoder offers 56% speedup of the encoding process over the Original encoder.

In summary, we reach the following conclusions:

- At the same compression ratio (1:11.2, 370kbps), the Modified MPEG-2 encoder can achieve in average **14.25% better frame quality** in terms of SNR and **46% better quality of regions of interest** than the original encoder.
- The modified encoder can achieve the same quality with the Original encoder with approximately **55.95% less bits** at compression ratios between 1:24 and 1:26.
- At 300kbps the Modified MPEG-2 encoder achieves approximately **24.3% higher quality of ROIs** than the quality of the whole frame.
- At 370kbps the Modified MPEG-2 encoder achieves approximately **27.8% higher quality of ROIs** than the quality of the whole frame.

- At bit rates lower than 200kbps the Modified MPEG-2 encoder achieves approximately **12.8% higher quality of ROIs** than the quality of the whole frame.
- The higher the activity (average number of MBs classified as ROIs) of the sequence the less the improvement that the Modified encoder can achieve.
- The higher the activity (average number of MBs classified as ROIs) the more MBs are skipped in average per frame by the Modified encoder.
- The modified encoder offers a **59% speedup**.
- The TAD is not a very good measure to detect ROIs and results in variability in the quality of regions of interest (ROIs).
- The **bitrate variability** of the modified encoder is as good, if not better, as the original encoder.
- I-frames can not code as efficiently as P-frames but are needed in order to facilitate some functions such as random access and fast forward.

5.2 M-JPEG Encoder

We implemented two different adaptive M-JPEG encoders. One is the "M-JPEG encoder with prediction". In this implementation MCUs of the frame currently being encoded are predicted from the reconstructed MCUs in the same position in the previous frame. The second implementation is the "M-JPEG encoder without prediction". In this case no predictions are allowed. Both implementations are based on the open source JPEG library offered by the IJG (independent JPEG) group (`release 6b of 27-Mar-1998`). The code can be found at http://www.ijg.org.

Note: In MPEG-2 a "unit" is a MB. In JPEG a "unit" is a MCU. Just like in the MPEG-2 encoder the input in the M-JPEG encoder is downsampled (at 4:2:0) YCbCr data. Thus a MCU consists of 4 Y blocks one Cb and one Cr, just like a MB in MPEG-2.

Just like in the Modified MPEG-2 encoder implementation, in both M-JPEG encoders ROIs are detected by computing the total absolute difference (TAD) of a MCU from the MCU in the same position in the previous original frame. In the "MJPEG encoder with prediction" the decision to predict is based on the TAD (total absolute difference) of the current original MCU from the previous reconstructed MCU in the same position.

Finally, an efficient rate control algorithm that chooses the quantization matrices and the variable quantization scales of ROIs and non-ROIs in order to achieve a distortion of these regions as near as it can to a user defined distortion is introduced. The rate control algorithm in the two implementations is very similar.

In the next section 5.2.1 we present briefly the JPEG library we used and the alterations we did in order to add support for variable quantization.

In sections 5.2.2-5.2.4 we present the most important aspects of our M-JPEG encoders, namely the scheme we use to detect ROIs and predicted MCUs and the rate control algorithm.

5.2.1 IJG Group JPEG library

The library is an implementation of the JPEG image compression and decompression standard in C. It implements JPEG baseline, sequential and progressive compression processes. The hierarchical and lossless processes defined in the standard are not supported.

In addition, in order to support file conversion, viewing software and other typical JPEG applications, considerable functionality beyond the bare JPEG coding/decoding capability has been included. These functions pre-process the image before JPEG compression or post-process it after decompression. They include colorspace conversion, downsampling/ upsampling and color quantization.

The library does also not support variable quantization (extension of JPEG, PART-3). As a result we had to modify the library to add this feature. We added a function called jpeg_add_adaptive_quant. This function can be called by the application using the library, right before the actual compression of the data begins. It stores the quantization scales for each MCU in an array defined in the jpeg compression object. When the jpeg library reaches the quantization step, it reads these quantization scales and uses them to quantize the DCT coefficients. Variable quantization is allowed only for YCbCr input colorspace. It is the applications responsibility to choose the quantization scales (QS) for each MCU before the actual encoding process starts. If this function is not called, then the array is set to NULL and all the quantization scales are set to 1. The quantization scales have values that are integers and belong to the interval [1 31]. A 5 bit index is used to code a particular scale value and a single bit code is used to signal MCUs in which the multiplier changes. Thus there is a cost of 6 bits for each change in the multiplier. This is a scheme similar to MPEG-2. Quantization scale codes are stored right before the actual variable length encoded MCU starts.

Moreover, when decompressing data the variable quantization scales codes are read from the bit stream, translated to the actual scale values and used to dequantize the DCT coefficients.

5.2.2 Prediction & Regions of Interest Detection

For the detection of Regions of Interest and the decision whether to predict or not a MCU we use a simple scheme.

Just like in the Modified MPEG-2 encoder, we define a threshold T1 (ROI decision threshold). If the TAD of a MCU in the current original frame from the MCU in the same position in the previous original frame is greater than the threshold T1, then the MCU is considered a region of interest (ROI).

In the case of the "M-JPEG encoder with prediction" we also have to decide whether to predict or not a MCU. We define the "anchor frame period" N, where anchor frames, are frames without predictions (like an I frame in MPEG-2). If the TAD of a MCU in the current original frame from the MCU in the same position in the previous **reconstructed frame** is less than the threshold T2 and the MCU doesn't belong to an anchor frame it is predicted from the MCU in the previous reconstructed frame and we only need to encode their difference.

In the decoding process a predicted MCU is identified by its quantization scale QS. The rate control algorithm assigns the same quantization scale for all MCUs that belong to the same type of region. In our case, there are two region types, ROI and non-ROI, and thus only two quantization scale values are used throughout a frame. For predicted MCUs we use the quantization scale value assigned by the rate control algorithm increased by 1.

In order to let the decoder know which of the two QSs' have been assigned by the rate control algorithm, we emit an application marker (see section 1.3) that contains these values at the header of the frame. Thus, when the decoder finds a MCU with QS different from the ones found in the application marker it knows that this is a predicted MCU.

The overhead from this marker is 4 bytes to put the marker code, 3 bytes to write a string that lets us know that the contents of this marker are written by our application and 2 bytes for each QS; a total of 9 bytes per frame.

5.2.3 Rate Control of M-JPEG Encoders

The user initially defines a desired bit rate (bit_rate) and the desired distortion for each region (βr: roi, βn: non-roi) as well as the two thresholds T1, T2 (only when prediction is supported). The quantization scales and the quantization matrices are chosen so that the actual distortion of the regions is as close as possible to the user defined distortions **without exceeding the bit rate**. Actually we are not interested in the distortion of each region, but the ratio between the distortions of the two different types of regions.

Moreover, in the case of the "M-JPEG encoder with prediction" the user defines the frame period N every which an anchor frame appears.

A. Bit Allocation

In our implementation we use abbreviated data formats (see section 2.4.1). No tables are written in the data stream for all frames, except for the first frame in the sequence and the frames where the quantization tables change.

The bit overhead for all frames is at least 616 bits (no tables written). These bits include the necessary bits that are written with every frame (frame header scan header, SOI & EOI markers etc.). If the quantization and Huffman tables are emitted to the data stream then the overhead is increased by 1160 bits.

The bit allocation for the two different implementations is different and is described below.

MJPEG with prediction

If it is an anchor frame the bits allocated to the frame to are:

$$T_{pict} = \frac{bit\_rate}{frame\_rate} * (1 + \frac{average\_predicted\_MCUs}{X_1}) - overhead\_bits + Tov$$

where average_predicted_MCUs is the average number of MCUs that are predicted per frame and is found by adding the actual number of predicted MCUs in each frame up to the current and dividing by the number of the current frame, overhead_bits is the overhead introduced by the header as explained above, Tov is the difference of the bit estimation and the actual bits used for the previous frame and X1 is an empirically determined parameter set to 300.

Anchor frames play a critical role in the whole encoding process. The better their quality, the more and better predictions are made in the following frames. As a result, our target is to give anchor frames, which have no predicted MCUs, enough bits to maintain high quality.

If it is not an anchor frame:

$$T_{pict} = \frac{bit\_rate - T_{anch}}{frame\_rate - 1} - overhead\_bits + T_{ov}$$

where Tov is the difference of the actual bits to encode the previous frame from the estimated bits of the previous frame, frame_rate is the frame rate of the sequence and T_{anch} is the number of bits assigned for the last anchor frame.

MJPEG without prediction

In this case the bits allocated to the frame are simply:

$$T_{pict} = \frac{bit\_rate}{frame\_rate} - overhead\_bits + T_{ov}$$

B. Computation of Quantization Scales

The rate control is the same in both encoders and is based on the adaptive model-driven bit-allocation algorithm introduced in section 2.4.2.

We define two perceptual classes, the regions of interest perceptual class with noise sensitivity (mean square error) βr and the non-ROI class with noise sensitivity βn. The noise sensitivities are user defined.

The picture-wise rate quantization function used is:

$$R_p = \sum_{j=1}^{M} \frac{N_j}{2} \log_2 \frac{\sigma_j^2}{\alpha * QS_{p_j}^2}$$

Where p_j is the perceptual class index for block j, QS_{pj} is the QS used for blocks in perceptual class p_j, and σ_j^2 is block j's pixel variance. M is not the number of MCUs in the picture but the actual number of blocks in the frame (including Cr and Cb blocks).

We choose the QS_ps for a picture by first using:

$$\sum_{j=1}^{M} \frac{N_j}{2} \log_2 \frac{\sigma_j^2}{\alpha * \beta_{p_j} * QS_1^2 / \beta_1} \le T_p$$

to determine an optimal QS_p for any one particular p (we first find the optimal QS for non-roi regions), and then applying

$QSr^2/\beta r = QSn^2/\beta n$

we solve for the QS_p's for the ROI perceptual class.

The search procedure takes 3 iterations in average.

In the case of the "MJPEG encoder with prediction" we define two 'α' parameters; one for each type of frame (anchor frame α_i, and predicted α_p), while for the "MJPEG encoder without prediction" only one, α_i, since all the frames are of the same type.

The initial parameters α_i and a_p are not determined by performing a pre-coding pass to generate an (R,D) pair due to encoding delay reasons but have a constant initial value. This might result to bad estimations in the first frames until the rate control algorithm stabilizes.

If the MCU is going to be predicted then the actual QS used is QS=QS+1.

5.2.4 Computation of Quantization Matrices

The quantization scales are used to control the bit-rate while achieving the desired quality. However, by using a constant quantization matrix for every frame type (anchor frame or not) it is possible that the rate control algorithm will not be able to meet the desired bit-rate because of the limited rang of the quantization scales QS (1-31). Thus, we scale the quantization matrices in order to be able to meet a large range of input bit-rates. The decision to scale the quantization matrices is based on the actual values of the quantization scales QSs and the value of the parameter T_{ov}.

If the quantization scale that is found by the rate control algorithm for the non-roi class is to the bounds of the available range of values (1-31) and $|T_{ov}|$ is larger than a value, which in our test is set to 5000bits, the quantization matrix is scaled and emitted to the next frame.

The JPEG library has a function (jpeg_set_linear_quality) that generates quantization tables. The tables generated are the ones given in the JPEG specification [1] section K.1

multiplied by the input parameter scale_factor (which is expressed as a percentage; thus scale_factor 100 reproduces the spec's tables). The tables are given in figure 5.39;

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

Luminance Quantization Table

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

Chrominance Quantization Table

Figure 5.39: JPEG quantization matrices

M-JPEG encoder with prediction

In this case, depending on the frame type (anchor frame or not), we use different scales. Initially the scale_factor for anchor frames is set to 40 and for the non anchor frames to 50.

If the QS of the non-roi class found by the rate control algorithm is 30 and $|T_{ov}|>5000$ then the scale_factor is increased by 5 and the function jpeg_set_linear_quality is called before compressing the next frame of the same type. If the QS is 1 and $|T_{ov}|>5000$ the scale_factor is decreased by 5 and the function jpeg_set_linear_quality is called before compressing the next frame of the same type.

M-JPEG encoder without prediction

At the start of the sequence the scale_factor is set to 30.
Again, if the QS of the non-roi class found by the rate control algorithm is 31 and $|T_{ov}|>5000$ then the scale_factor is increased by 5 and the function jpeg_set_linear_quality is called before compressing the next frame of the same type. If the QS is 1 and $|T_{ov}|>5000$ the scale_factor is decreased by 5 and the function jpeg_set_linear_quality is called before compressing the next frame of the same type.

5.2.5 Test Results

In these tests, the threshold T1 takes the value 1000 for both implementations. In the case of the "M-JPEG encoder with predictions" the threshold T2 takes the value 900 and the anchor frame period N the value 25. In the next table we present the values of the most important parameters of the encoding process.

T1 : ROI decision threshold	1000
T2 : Predicted MCU decision threshold	700
β_r : ROIs distortion	10
β_n : non ROIs distortion	70
N : period of anchor frames	25

Table 5.16: M-JPEG encoding parameters

The video sequences used in our tests are the same as the ones used for the test in the case of MPEG-2 (4 test sequences). The resolution of all the test sequences is 384x288 and the frame rate is 25 frames/sec.

In the next section, 5.2.5.1 we present the performance of the coding process of the two encoders in terms of quality and coding efficiency. We encode each sequence at three different bit rates (1100, 1300, 1500 kbps) with both adaptive M-JPEG encoders (with and without prediction) and compare them. Since there is no official M-JPEG standard, there is no M-JPEG implementation that can be used as a reference to compare the results of our M-JPEG encoders with (in the case of the MPEG-2 encoder the implementation with the TM-5 rate control algorithm is used as a common base of comparison of all the improved implementations) and to draw some safe conclusions. However, in order to exhibit the improvement in the quality of ROIs we also encode the final sequence with our "M-JPEG encoder without prediction" but without adaptivity; that is the **"reference M-JPEG encoder"**. This is achieved by setting equal values in the desired distortions of the two region types βr and βn.

In section 5.2.5.2 we present the performance of both encoders in terms of encoding delay and finally in section 5.2.5.3 some conclusions.

5.2.5.1 Quality and Coding Efficiency

EnterExitCrossingPaths1cor.mpeg

This is a test sequence with average activity. The averageinumber of MBs classified as ROIs is 16.0 MCUs per frame. The total number of frames in the sequence is 383 and the size of the uncompressed video sequence is 63535104 bits.

The encoding results of both encoders at a bit-rate of 1100kbps are shown in table 5.17.

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	16699392	1:3.8	16.44	20.1	5.35	7.5	243
No Pred.	16425482	1:3.86	14.91	18.26	4.61	6.18	0

Table 5.17: Encoding results at 1100kbps

From the above table we notice that the "M-JPEG encoder with prediction" has higher average quality, in terms of SNR, than the "M-JPEG encoder without prediction". The SNR of the Y component of the frame is higher by 10.26% and the SNR of the Y component of the MCUs classified as ROIs by 10.1%. The quality improvement for the other two components (Cb, Cr) is 16.1% and 21.3% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with prediction" is 22.2% higher than the quality of the whole frame, while for the "M-JPEG encoder without prediction" higher by 22.46%.

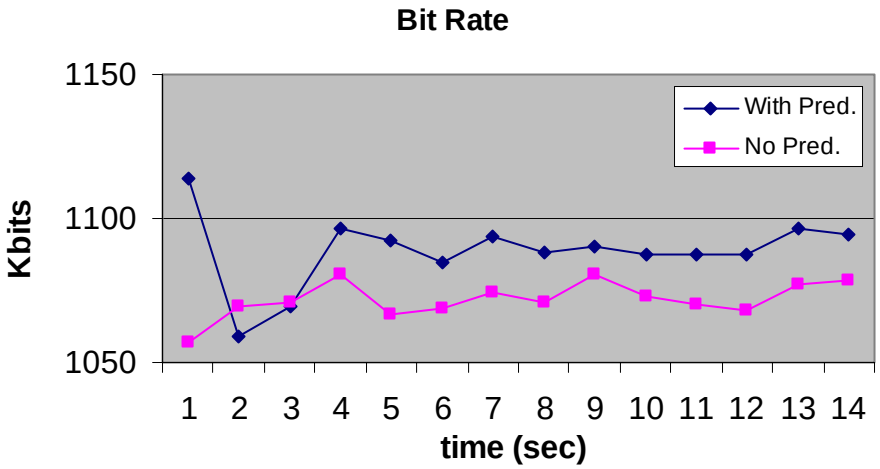


Figure 5.40: Bitrate at 1100kbps

From figure 5.40, we can see that the "M-JPEG encoder without prediction" has good bit-rate variability and never exceeds the 1100kbps.

On the other hand the rate control algorithm of the "M-JPEG encoder with prediction" fails to keep the bit-rate under 1100kbps at the beginning of the sequence. This means that it makes some bad estimations which are mainly a result of the high variability of the number of predicted MCUs and the time it takes to stabilize.

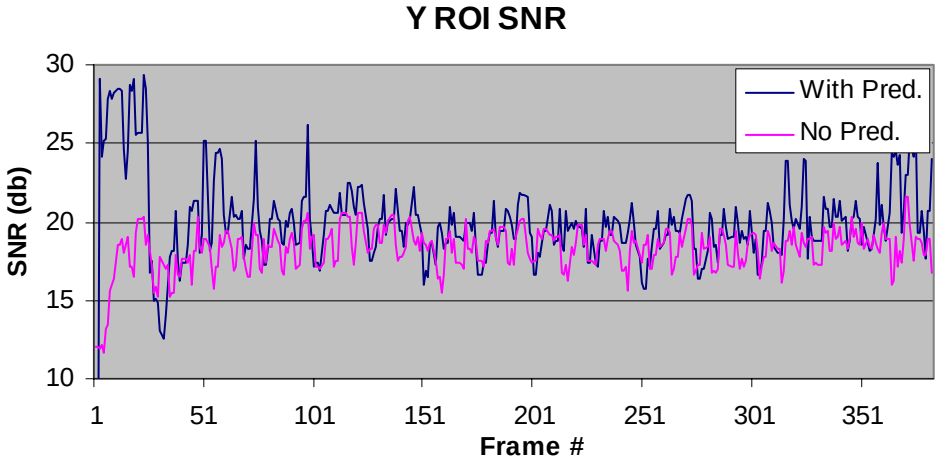


Figure 5.41: Signal to Noise ratio of Y component Regions of Interest at 1100kbps

The spikes of the quality of ROIs (fig. 5.41) and frame (fig. 5.42) in the first 25 frames are due to the large number of predicted MCUs (over 400) compared to the rest frames (approximately 250) and the time required for the rate control algorithm to stabilize.

As seen in figure 5.41 the quality of ROIs of the "M-JPEG encoder without prediction" is comparable to that of the "M-JPEG encoder with predictions". The use of prediction doesn't seem to bring much improvement. This happens because DC coefficients are encoded differently from AC coefficients using differential coding. By predicting a number of MCUs results in lower spatial correlation between the DC coefficient and as a result more bits are required to encode the DC terms. At higher bit rates, where the predicted MCUs are more the improvement is clear.

The low frame quality of the "M-JPEG encoder without prediction" in the first 25 frames is again because of the bad estimations of the rate control algorithm at the beginning of the encoding process. The bad estimations are due to the wrong initial values of the the parametesrs α_i, α_p of the rate control algorithm, as well as the large number of predicted MCUs in the first frames compared to the rest.

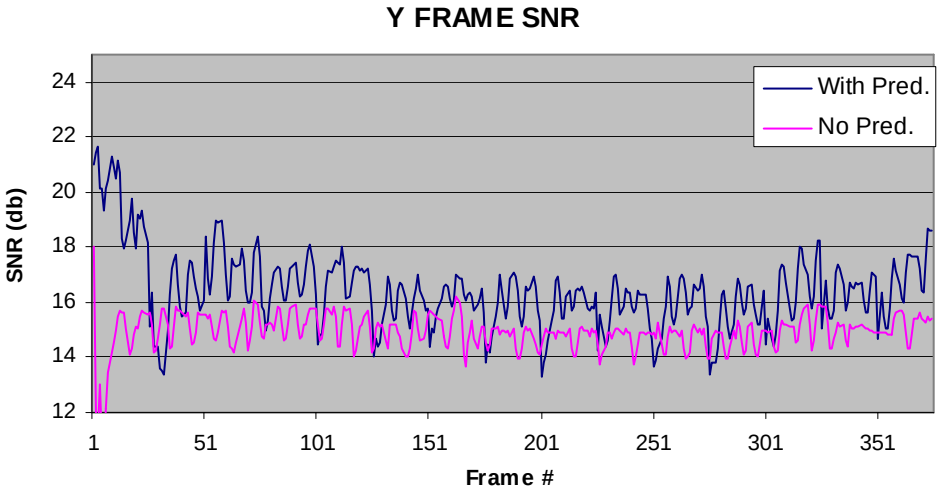


Figure 5.42: Signal to Noise ratio of Y component at 1100kbps

The encoding results at a bit-trate of 1300kbps are presented in table 5.18

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Pred.	19593192	1:3.24	21.22	24.67	6.7	10.3	345
No Pred.	19383104	1:3.27	15.50	18.81	5.07	6.8	0

Table 5.18: Encoding results at 1300kbps

The quality of the Y component of the whole frame of the "M-JPEG encoder with prediction" is higher by 36.9%, while for the Y component of ROIs by 31%. The quality of the chrominance components is higher by 32% and 22% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with prediction" is 16.2% higher than the quality of the whole frame, while for "M-JPEG encoder without prediction" 21.3%.

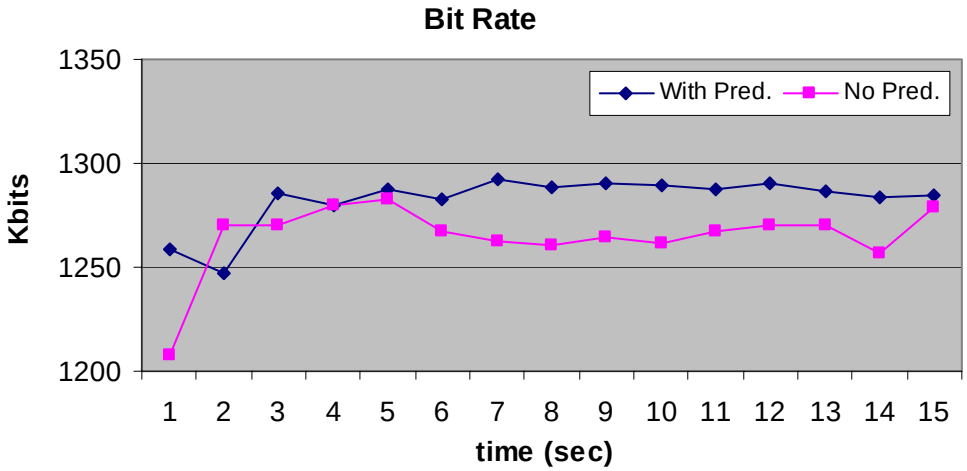


Figure 5.43: Bitrate at 1300kbps

Both encoders present good bit-rate variability, without exceeding the desired bit-rate. The low bit rate at the beginning of the sequence is a result of the rate control algorithm that makes bad estimations and allocates few bits to the first frames. After it has stabilized the estimations are very good and the bit rate remains stable.

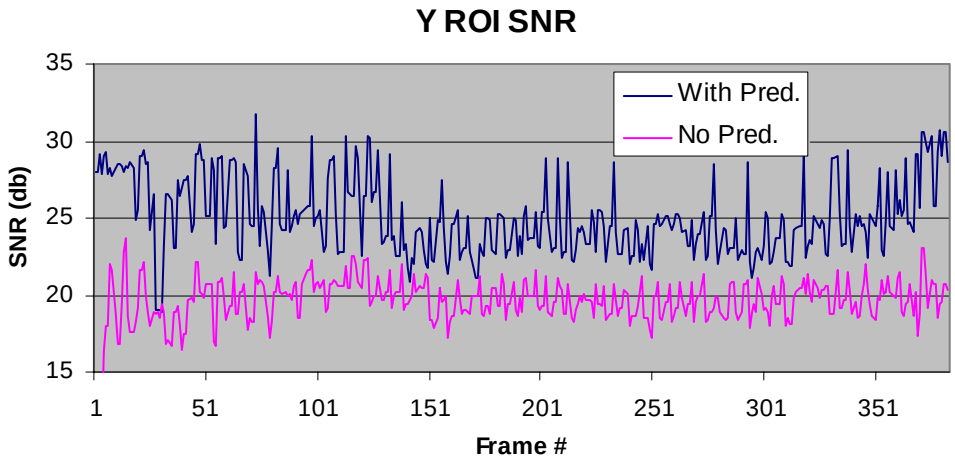


Figure 5.44: SNR of Y component of ROIs at 1300kbps.

In figure 5.44 we notice that the quality of frames 139-320 has low variability. The number of predicted MCUs from frame to frame is almost stable and this leads to good estimations from the rate control algorithm. In addition the spikes in the quality of anchor frames are not very large. The bit rate and the number of predicted MCUs per frame are high enough and result in high quality of non- anchor frames comparable to the quality of anchor frames (fig. 5.44, 5.45).

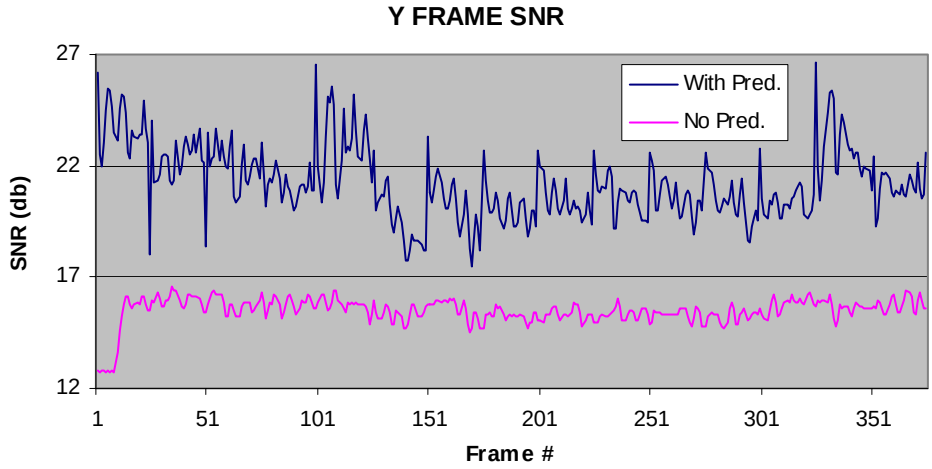


Figure 5.45: SNR of Y component at 1300kbps.

The encoding results at a bit rate of 1500kbps are presented in table 5.18

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With pre.	22774408	1:2.79	24.27	27.94	7.4	11.6	376.3
No pred.	22419456	1:2.83	17.53	21.2	5.5	8.2	0

Table 5.19: Encoding results at 1500kbps

The quality of the Y component of the whole frame of the "M-JPEG encoder with prediction" is higher by 38.4%, while for the Y component of ROIs by 31.7%. The quality of the Cb, Cr components is higher by 34.5% and 41.4% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with prediction" is 15.1% higher than the quality of the whole frame, while for "M-JPEG encoder without prediction" 20.9%.

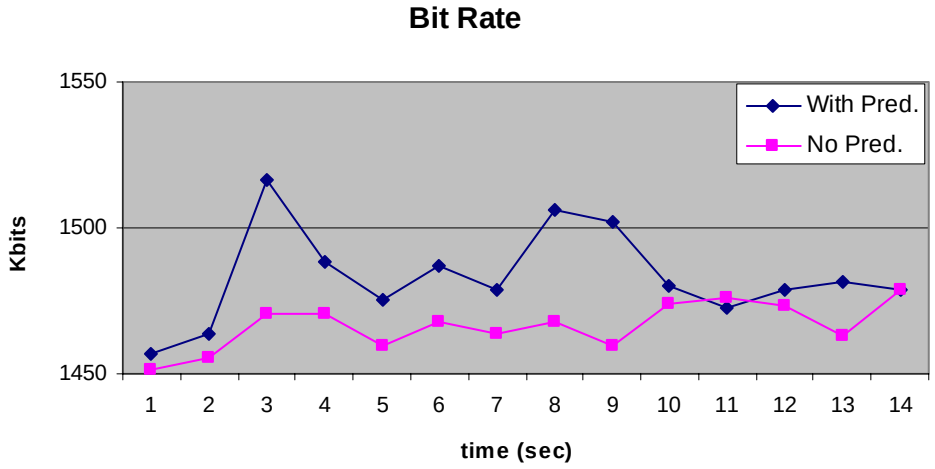


Figure 5.46: Bitrate at 1500kbps

The bit rate of the "M-JPEG encoder without prediction" remains good- it has small variability and it never exceeds the desired bit rate (fig 5.46).

On the other hand, the "M-JPEG encoder with prediction" presents high bit rate variability and exceeds the desired bitrate. This failure is a result of the variable number of MCUs predicted from frame to frame. Since our scheme to decide whether to predict or not a MCU is not very good, as discussed in the conclusions section, in one frame many MCUs might be predicted while in the next much fewer. This affects the rate control algorithm that makes bad estmaitions.

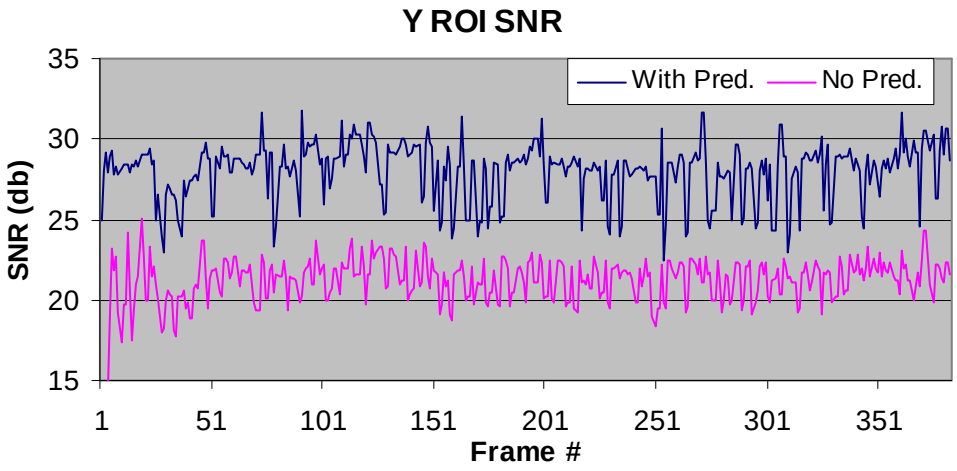


Figure 5.47: SNR of Y component of ROIs at 1500kbps.

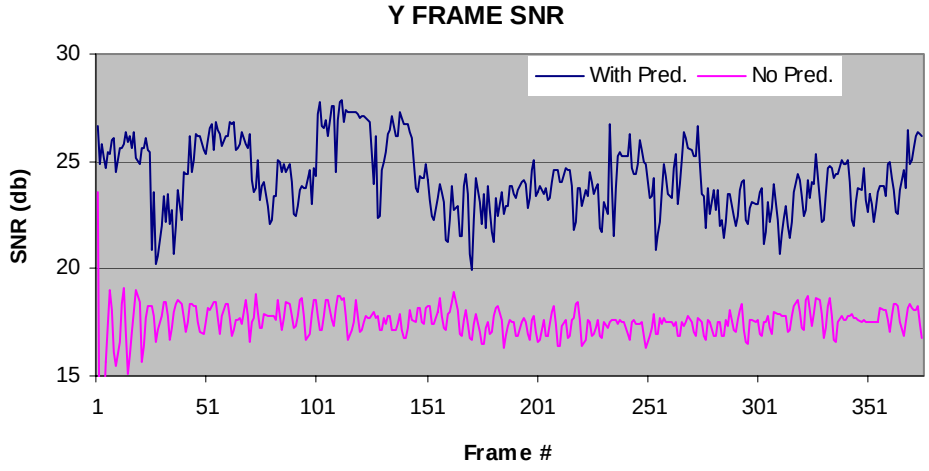


Figure 5.48: SNR of Y component at 1500kbps.

Notice that the quality achieved by the"M-JPEG encoder without prediction" at 1500kbps (table 5.19) is comparable to the quality achieved by the "M-JPEG encoder with prediction" at 1100kbps (table 5.17). The results are reproduced in the next table:

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Pred.	16699392	1:3.8	16.44	20.1	5.35	7.5	243
No Pred.	22419456	1:2.83	17.53	21.2	5.5	8.2	0

Table 5.20: Encoding results at same quality

From table 5.20 we reach the conclusion that the "M-JPEG encoder with prediction" can achieve approximately the same quality of ROIs as the frame quality in the "M-JPEG encoder without prediction" with 25.5% less bits.

ThreePastShop2front.mpeg

This is a test sequence with no activity. The average number of MCUs classified as ROIs is 0.0 MCUs per frame. The total number of frames in the sequence is 500 and the size of the uncompressed video sequence is 82944000 bits.

The encoding results of both encoders at a bit-rate of 1100kbps are shown in table 5.21. In this test sequence no ROIs appear.

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	16610712	1:3.84	14.18	-	6.6	9.31	91.47
No Pred.	16449912	1:3.88	12.89	-	5.85	8.68	0

Table 5.21: Encoding results at 1100kbps

From the above table we notice that the "M-JPEG encoder with prediction" has higher average quality, in terms of SNR. The SNR of the Y component of the frame is higher by 10%. The quality improvement for the other two components (Cb, Cr) is 12.8% and 8.31% respectively.

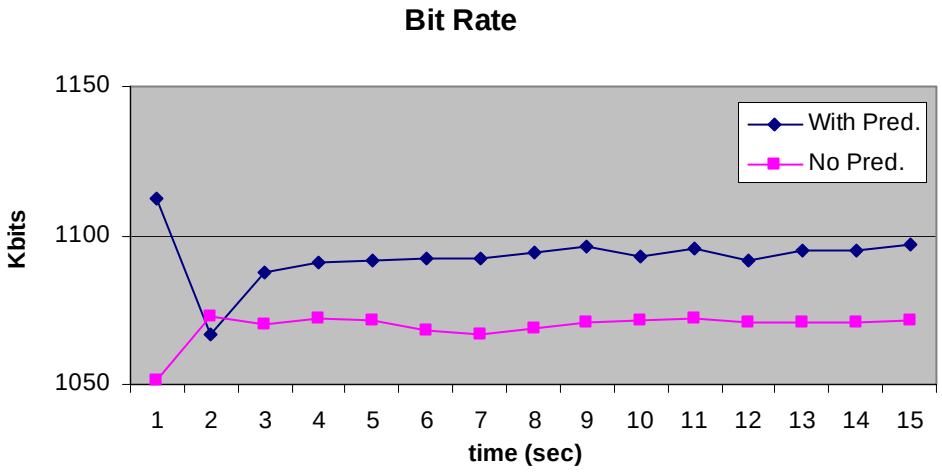


Figure 5.49: Bitrate at 1100kbps

In fig. 5.49, we can see that both the "M-JPEG encoder without prediction" and the M-JPEG encoder with prediction" have good bit-rate variability.

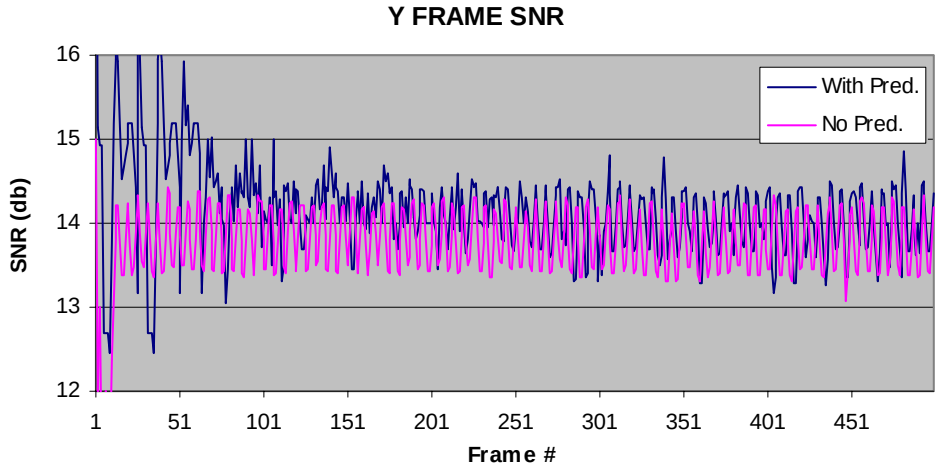


Figure 5.50: Signal to Noise ratio of Y component at 1100kbps

The "M-JPEG encoder with prediction" is more unstable. The spikes in the first 25 frames are due to the large number of predicted MCUs (over 400) compared to the rest frames (approximately 100) and the time required for the rate control to stabilize.

The low frame quality of the "M-JPEG encoder without prediction" in the first 25 frames is again because of the bad estimations of the rate control algorithm at the beginning of the encoding process.

The encoding results at a bitrate of 1300kbps are presented in table 5.22

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Pred.	19709432	1:3.22	16.59	-	7.88	10.7	174.1
No Pred.	19496872	1:3.26	14.50	-	6.74	9.47	0

Table 5.22: Encoding results at 1300kbps

The quality of the Y component of the whole frame of the "M-JPEG encoder with prediction" is higher by 14.4% from that of the "M-JPEG encoder without prediction". The quality of the Cb, Cr chrominance components is higher by 15.7% and 12.9% respectively.

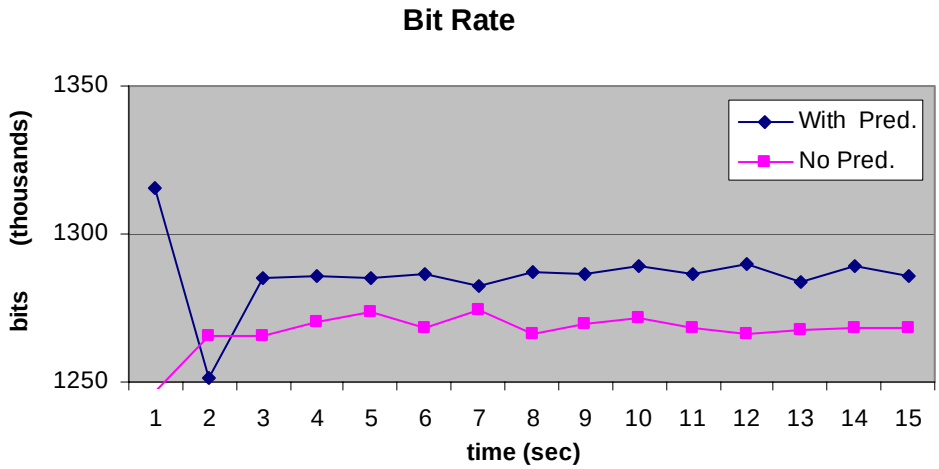


Figure 5.51: Bitrate at 1300kbps

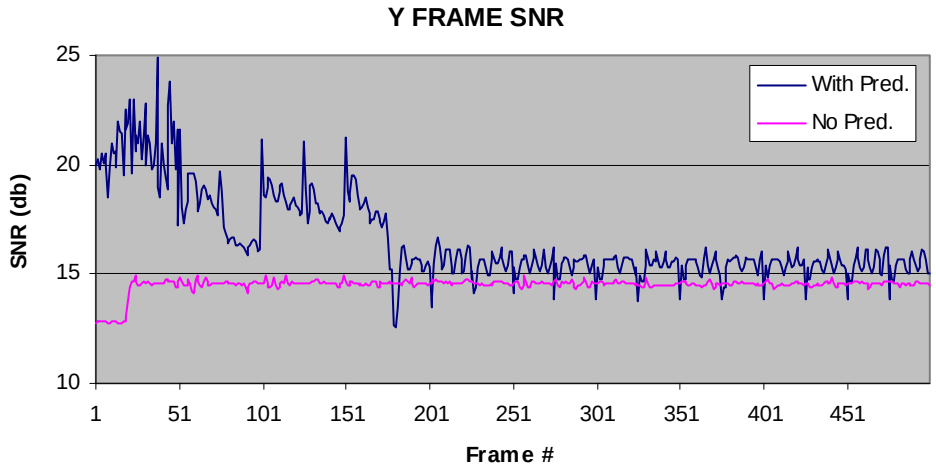


Figure 5.52: SNR of Y component at 1300kbps.

The high frame quality of the "M-JPEG encoder with prediction" at the beginning of the sequence (1-170) is due to the large number of MCUs predicted compared to the number of MCUs predicted in the rest frames.

The encoding results at a bit rate of 1500kbps are presented in table 5.23

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With pre.	28876336	1:2.87	23.87	-	10.5	14.1	371.2
No pred.	29077128	1:2.85	16.25	-	7.5	10.3	0

Table 5.23: Encoding results at 1500kbps

The quality of the Y component of the whole frame of the "M-JPEG encoder with prediction" is higher by 46.8%. The quality of the Cb and Cr components is higher by 40% and 34.6% respectively.

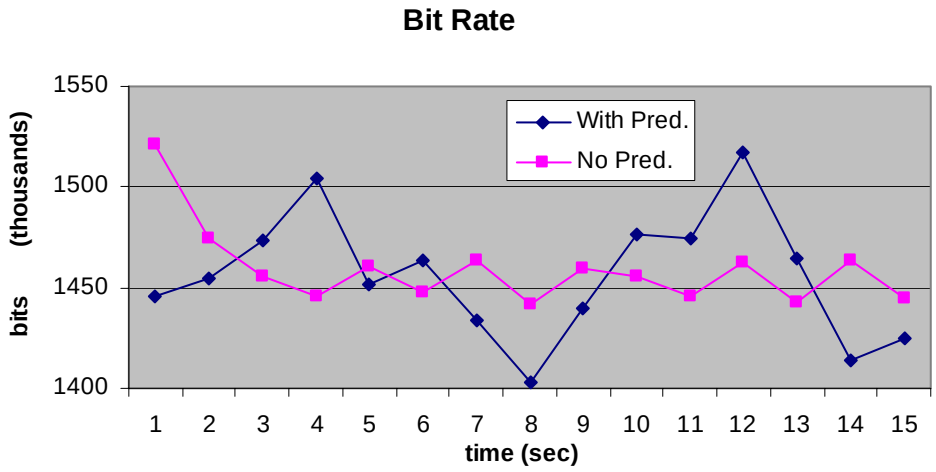


Figure 5.53: Bitrate at 1500kbps

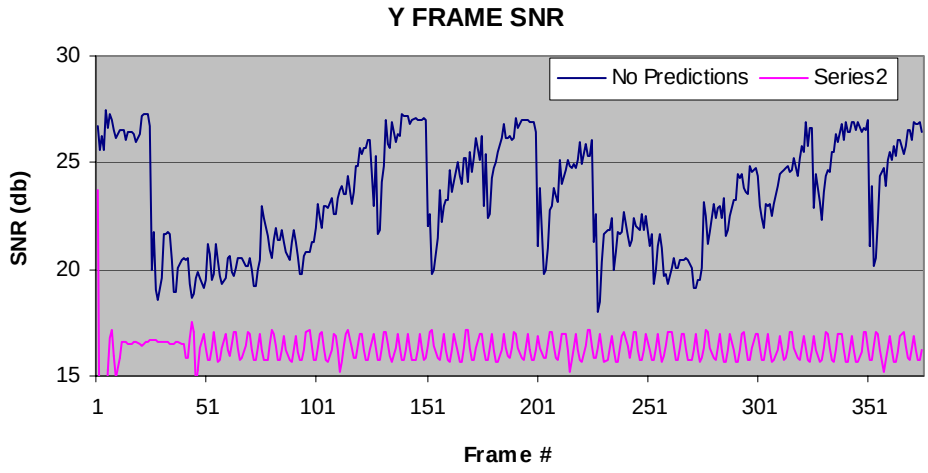


Figure 5.54: SNR of Y component at 1500kbps.

In figure 5.54 we can see that the quality of anchor frames can not achieve the quality of the rest frames, even though we allocate more bits to anchor frames. The number of predicted MCUs in every frame is high (approximately 90% of the total MCUs) which results in very high quality.

Notice that the quality achieved by the "M-JPEG encoder without prediction" at 1500kbps (table 5.23) is almost the same with the quality achieved by the "M-JPEG encoder with prediction" at 1300kbps (table 5.22). The results are reproduced in the next table:

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	19709432	1:3.22	16.59	-	7.88	10.7	174.1
No Pred.	29077128	1:2.85	16.25	-	7.5	10.3	0

Table 5.24: Encoding results at same quality

From table 5.24 we reach the conclusion that the "M-JPEG encoder with prediction" can achieve approximately the same quality as the "M-JPEG encoder without prediction" with 32.2% less bits.

TwoEnterShop2front.mpeg

This is a test sequence with low activity. The average number of MCUs classified as ROIs is 12.0 MCUs per frame. The total number of frames in the sequence is 375 and the size of the uncompressed video sequence is 62208000 bits.

The encoding results of both encoders at a bit-rate of 1100kbps are shown in table 5.25.

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	16630312	1:3.74	14.89	17.56	6.87	9.6	118
No Pred.	16105336	1:3.86	13.60	16.0	6.1	8.8	0

Table 5.25: Encoding results at 1100kbps

From the above table we notice that the "M-JPEG encoder with prediction" has higher average quality, in terms of SNR. The SNR of the Y component of the frame is higher by 9.5% and the SNR of the Y component of the MCUs classified as ROIs by 9.7%. The quality improvement for the other two components (Cb, Cr) is 12.6% and 9.1% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with predictions" is 17.9% higher than the quality of the whole frame, while for "M-JPEG encoder without prediction" 17.6%.

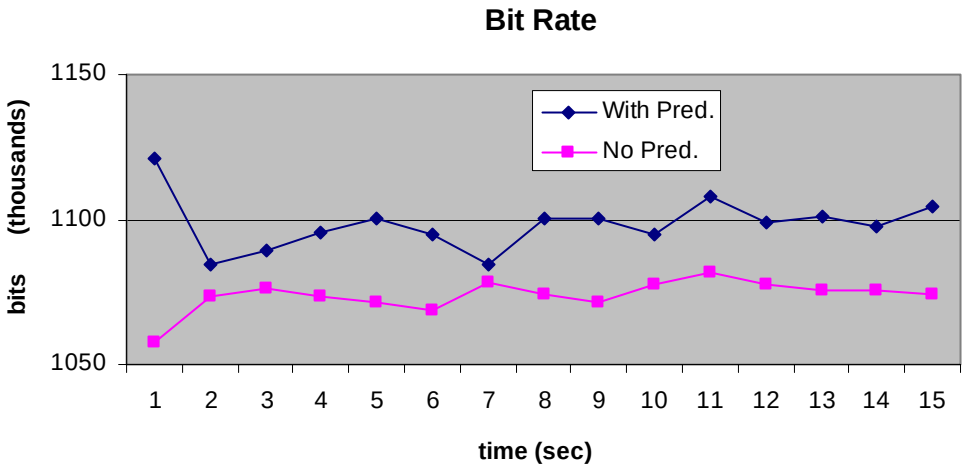


Figure 5.55: Bitrate at 1100kbps

From figure 5.55, we can see that the bit-rate of the "M-JPEG encoder without prediction" exceeds 1100kbps many times. The bit rate of the "M-JPEG with prediction" presents much better behavior.

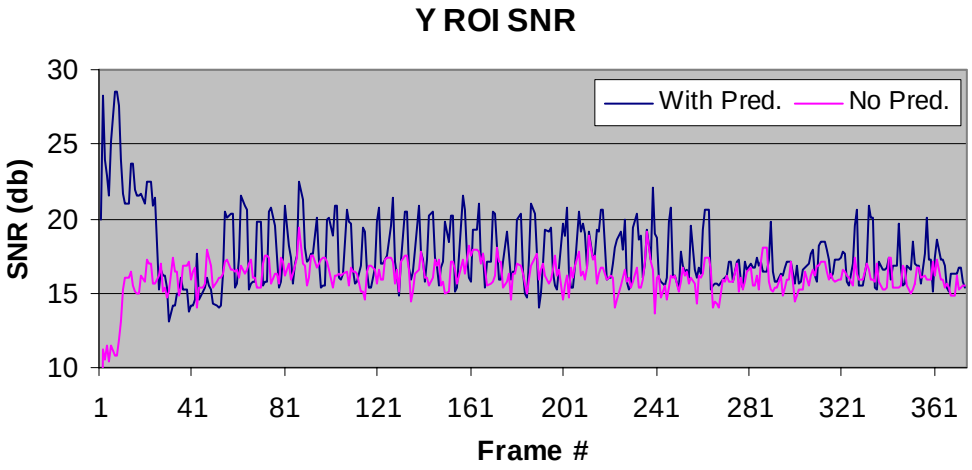


Figure 5.56: Signal to Noise ratio of Y component of ROIs at 1100kbps

As seen in figure 5.56 the quality of ROIs of the "M-JPEG encoder without prediction" presents smaller variability than than "M-JPEG encoder with prediction".

The "M-JPEG encoder with prediction" is more unstable. The high quality of the first 25 frames is due to the large number of predicted MCUs (over 400) compared to the rest frames (approximately 320) and the time required for the rate control to stabilize.

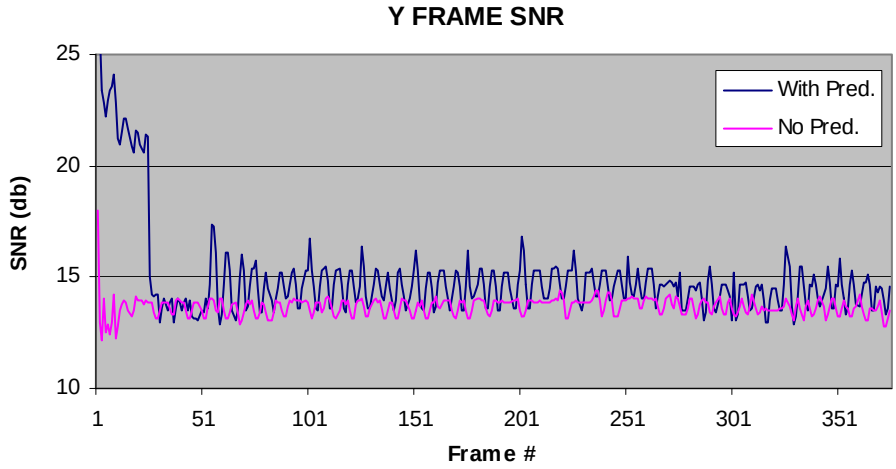


Figure 5.57: Signal to Noise ratio of Y component at 1100kbps

The encoding results at a bitrate of 1300kbps are presented in table 5.26

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Pred.	19115360	1:3.25	15.69	17.93	7.31	10.0	126.5
No Pred.	19020119	1:3.27	14.38	16.62	5.07	6.8	0

Table 5.26: Encoding results at 1300kbps

The quality of the Y component of the whole frame is higher by 9.1%, while for the Y component of ROIs by 7.9%. The quality of the Cb, Cr components is higher by 6.31% and 9% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with predictions" is 14.2% higher than the quality of the whole frame, while for "M-JPEG encoder without predictions" 15.5%.

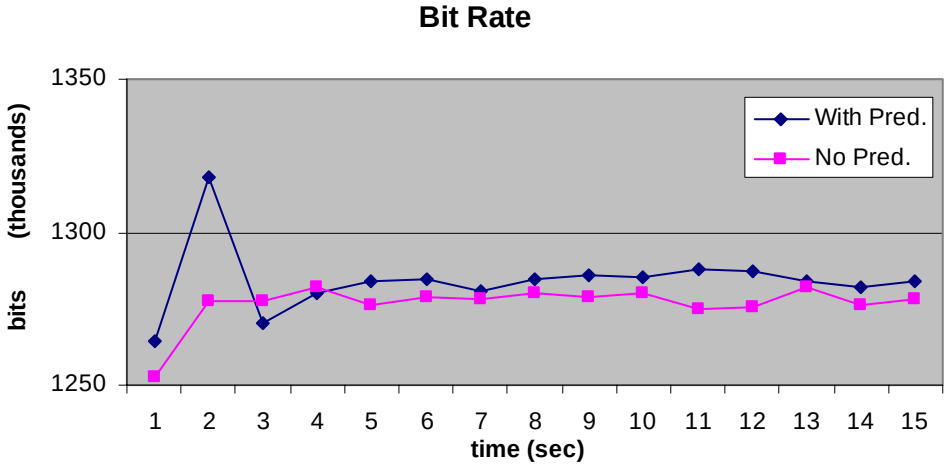


Figure 5.58: Bitrateat 1300kbps

Both the "M-JPEG encoder with prediction" and without prediction present a very stable behavior.

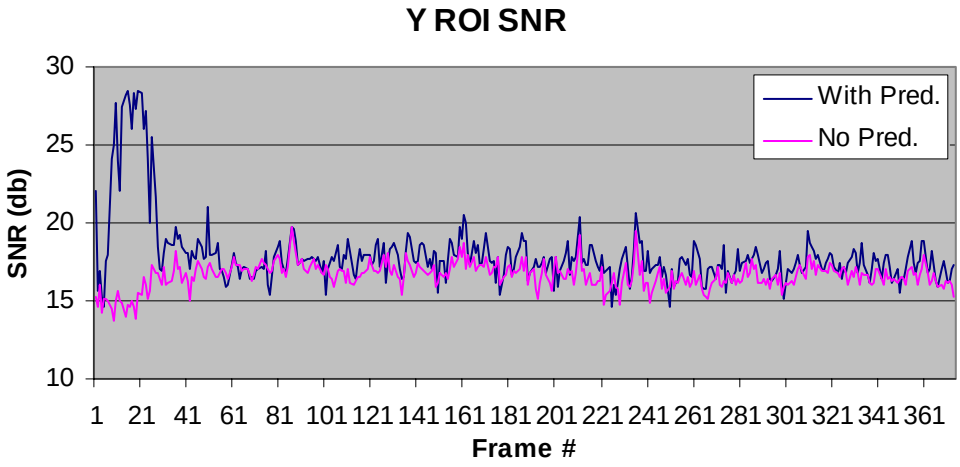


Figure 5.59: SNR of Y component of ROIs at 1300kbps.

In figures 5.59 and 5.60 we notice that the "M-JPEG encoder with predictions" presents only a slight improvement of the quality, even though 126.5 MCUs, in average, are predicted in every frame.

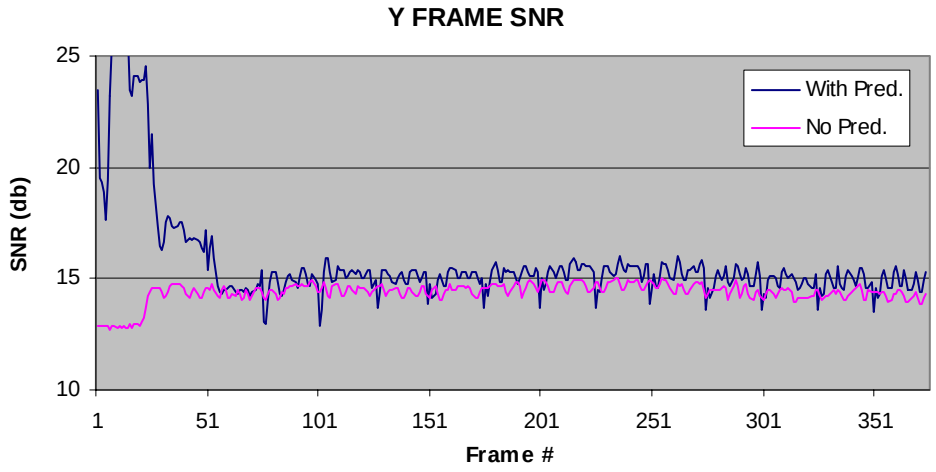


Figure 5.60: SNR of Y component at 1300kbps.

The encoding results at a bit rate of 1500kbps are presemted in table 5.27

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With pre.	22060387	1:2.82	24.25	27.53	10.7	14.6	395.8
No pred.	21989576	1:2.83	16.10	18.72	7.46	10.3	0

Table 5.27: Encoding results at 1500kbps

The quality of the Y component of the whole frame is higher by 50%, while for the Y component of ROIs by 47%. The quality of the Cb, Cr components is higher by 43.3% and 41.7% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with predictions" is 13.5% higher than the quality of the whole frame, while for "M-JPEG encoder without predictions" 16.2%.

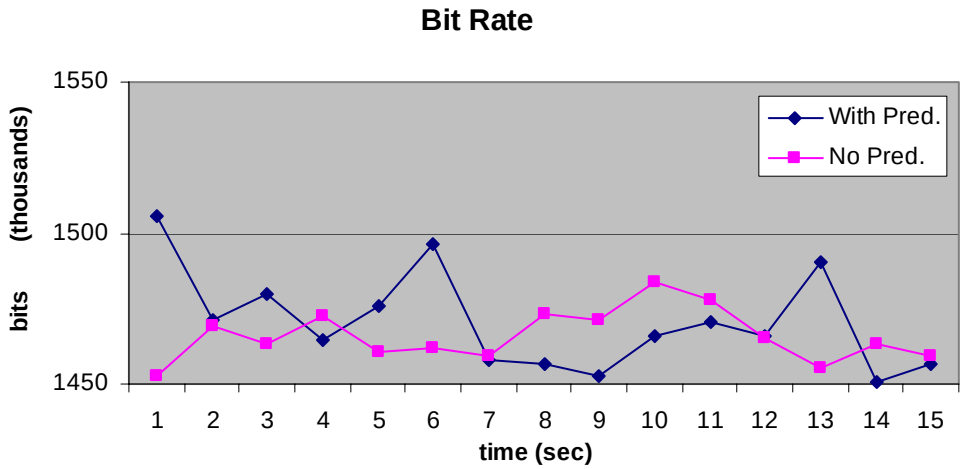


Figure 5.61: Bitrate at 1500kbps

The bit rate of the "M-JPEG encoder without prediction" remains good- it has small variability and it never exceeds the desired bit rate (fig 5.61).

On the other hand, the "M-JPEG encoder with prediction" presents high bit rate variability and exceeds the desired bit rate at the beginning of the sequence.

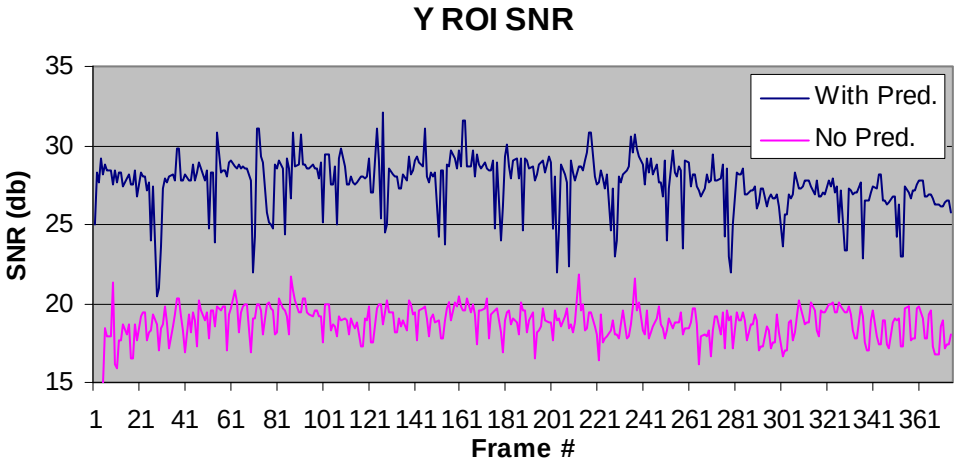


Figure 5.62: SNR of Y component of ROIs at 1600kbps.

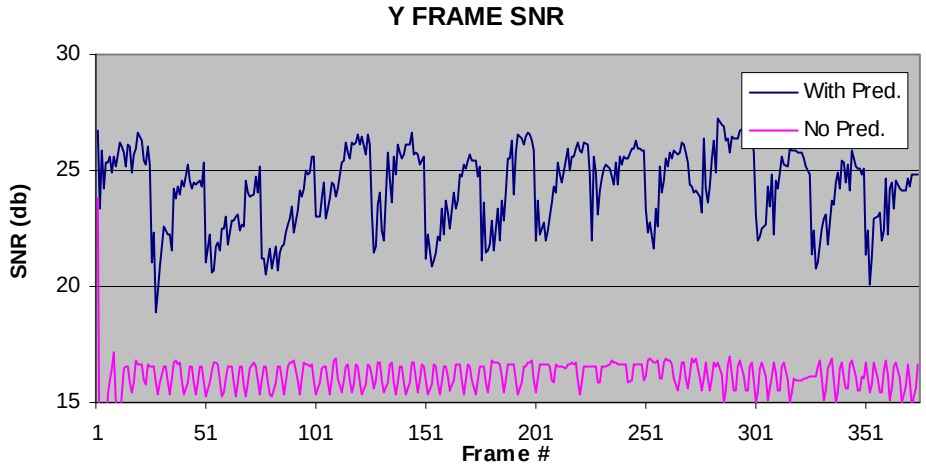


Figure 5.63: SNR of Y component at 1500kbps.

Notice that the quality achieved by the "M-JPEG encoder without prediction" at 1300kbps (table 5.27) is almost the same with the quality achieved by the "M-JPEG encoder with prediction" at 1100kbps (table 5.26). The results are reproduced in the next table:

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	19115360	1:3.25	15.69	17.93	7.31	10.0	126.5
No Pred.	21989576	1:2.83	16.10	18.72	7.46	10.3	0

Table 5.28: Encoding results at same quality

From table 5.28 we reach the conclusion that the "M-JPEG encoder with prediction" can achieve approximately the same quality as the "M-JPEG encoder without prediction" with 13.07% less bits.

TwoEnterShop1cor.mpeg

This is a test sequence with high activity. The average number of MCUs classified as ROIs is 25.0 MCUs per frame. The total number of frames in the sequence is 625 and the size of the video sequence, uncompressed, is 103680000 bits.

The encoding results of both encoders at a bit-rate of 1100kbps are shown in table 5.29.

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	26961232	1:3.84	16.01	18.62	5.23	6.78	188
No Pred.	26385296	1:3.92	14.02	16.62	4.59	5.75	0

Table 5.29: Encoding results at 1100kbps

From table 5.29 we notice that the "M-JPEG encoder with prediction" has higher average quality, in terms of SNR, than the "M-JPEG encoder without prediction". The SNR of the Y component of the frame is higher by 14.1% and the SNR of the Y component of the MCUs classified as ROIs by 12%. The quality improvement for the other two components (Cb, Cr) is 13.9% and 17.9% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with prediction" is 16.3% higher than the quality of the whole frame, while for "M-JPEG encoder without prediction" 18.5%.

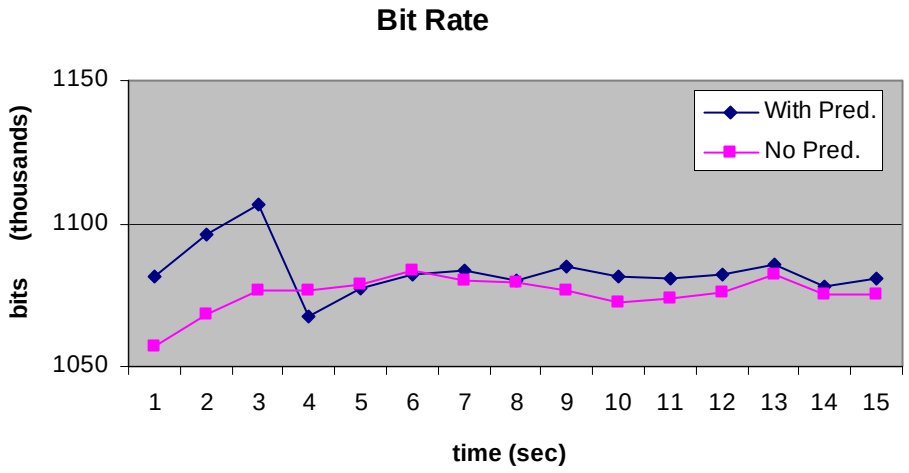


Figure 5.64: Bitrate at 1100kbps

From figure 5.64, we can see that the bit-rate of the "M-JPEG encoder without predictions" has small bit-rate variability. The bit rate of the "M-JPEG with prediction" presents a very good behavior, too, but exceeds the desired bit rate (1100kbps).

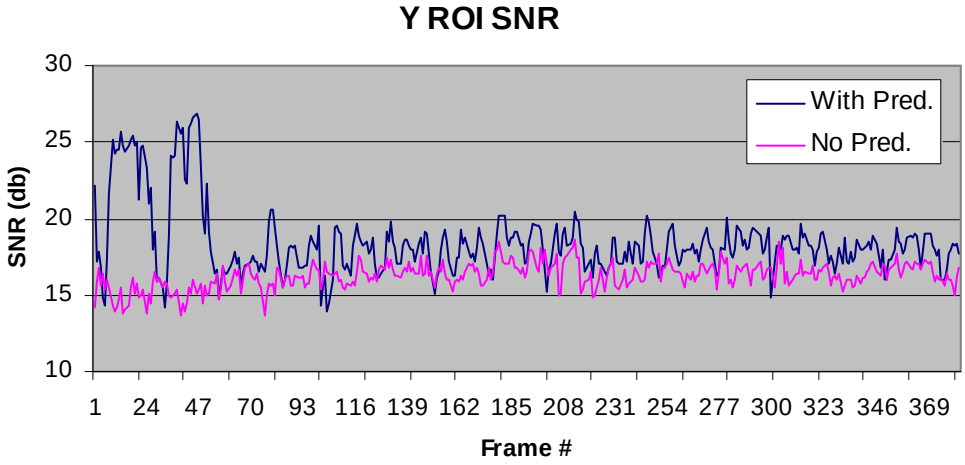


Figure 5.65: Signal to Noise ratio of Y component of ROIs at 1100kbps

As seen in figure 5.65 the quality of ROIs of the "M-JPEG encoder without prediction" presents smaller variability than the "M-JPEG encoder with prediction".

The "M-JPEG encoder with prediction" is more unstable. The spikes high quality of the first 25 frames is due to the large number of predicted MCUs (over 400) compared to the rest frames (approximately 180) and the time required for the rate control to stabilize.

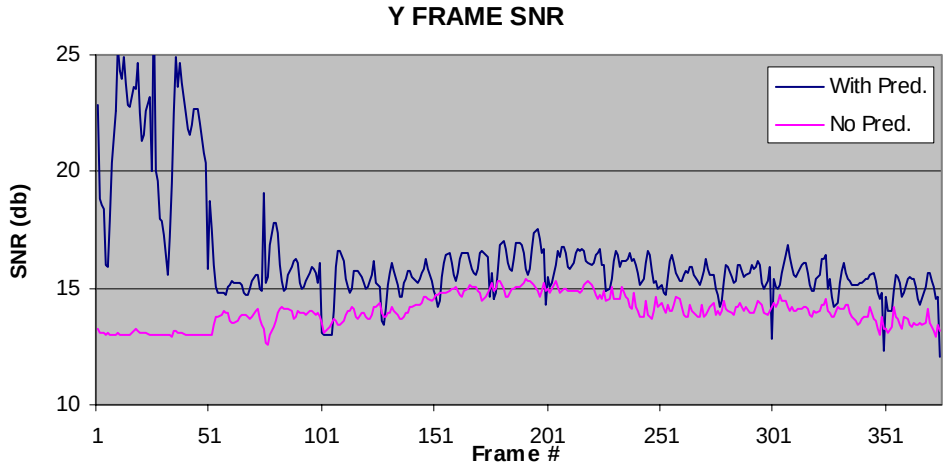


Figure 5.66: Signal to Noise ratio of Y component at 1100kbps

The encoding results at a bitrate of 1300kbps are presented in table 5.30

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Pred.	31910184	1:3.25	20.46	22.92	6.75	9.65	319
No Pred.	31711048	1:3.26	15.51	18.33	5.09	6.5	0

Table 5.30: Encoding results at 1300kbps

The quality of the Y component of the whole frame of the "M-JPEG encoder with prediction" is higher by 31.9% than the quality achieved by the "M-JPEG encoder without predictions", while for the Y component of ROIs by 25%. The quality of the Cb, Cr chrominance components is higher by 32.6% and 48% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with prediction" is 12.2% higher than the quality of the whole frame, while for "M-JPEG encoder without predictions" 21.3%.

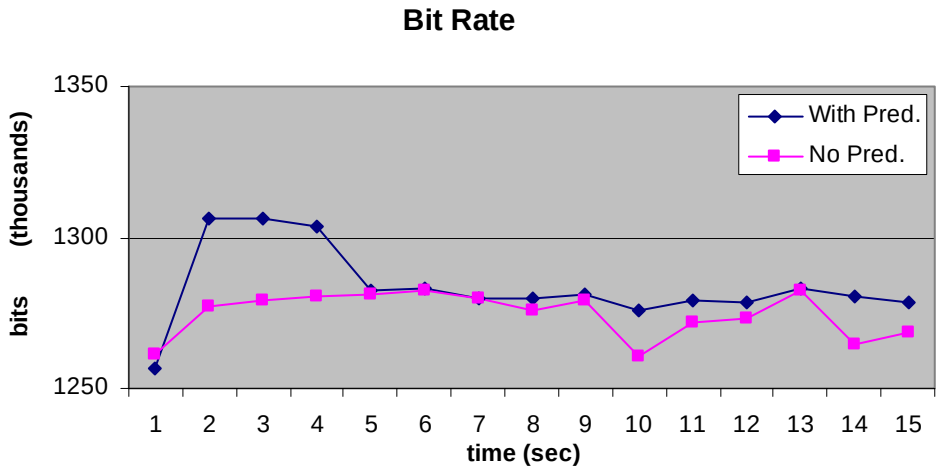


Figure 5.67: Bitrate at 1300kbps

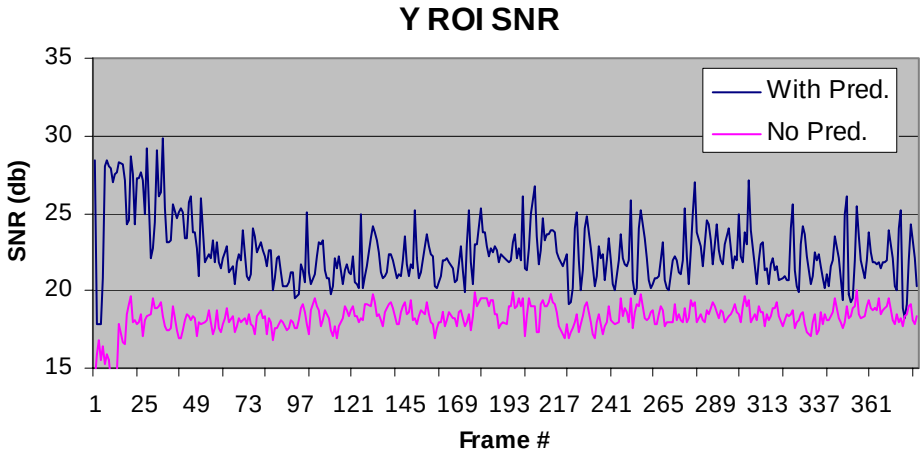


Figure 5.68: SNR of Y component of ROIs at 1300kbps.

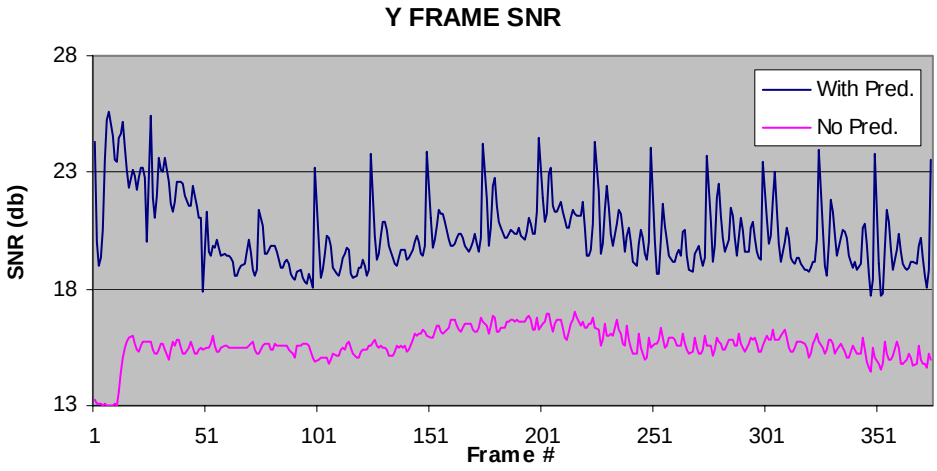


Figure 5.69: SNR of Y component at 1300kbps.

The encoding results at a bit rate of 1500kbps are presented in table 5.31

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With pred.	36892568	1:2.81	22.77	26.1	7.3	10.7	342
No pred.	36560680	1:2.83	17.6	20.69	5.6	7.78	0

Table 5.31: Encoding results at 1500kbps

The quality of the Y component of the whole frame of the "M-JPEG encoder with prediction" is higher by 29.3%, while for the Y component of ROIs by 26.1%. The quality of the Cb, Cr components is higher by 30.3% and 37.5% respectively.

Moreover, the quality of ROIs of the "M-JPEG encoder with predictions" is 14.6% higher than the quality of the whole frame, while for "M-JPEG encoder without predictions" 17.5%.

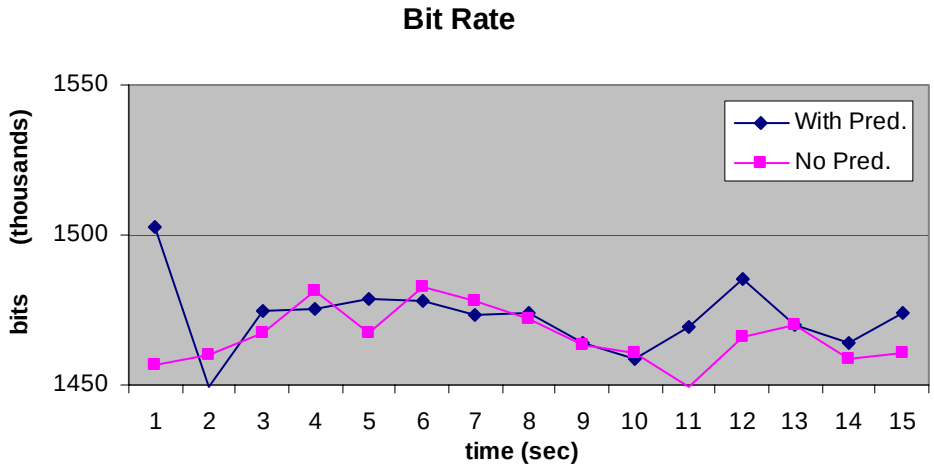


Figure 5.70: Bitrate at 1500kbps

The bit rate of the "M-JPEG encoder without prediction" remains good- it has small variability and it never exceeds the desired bit rate (fig. 5.70).

On the other hand, the "M-JPEG encoder with prediction" presents high bit rate variability and exceeds the desired bit rate.

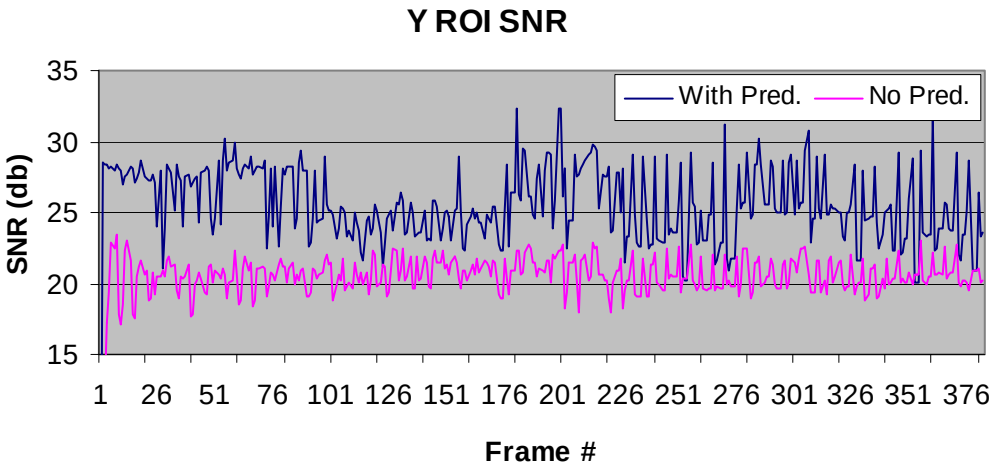


Figure 5.71: SNR of Y component of ROIs at 1600kbps.

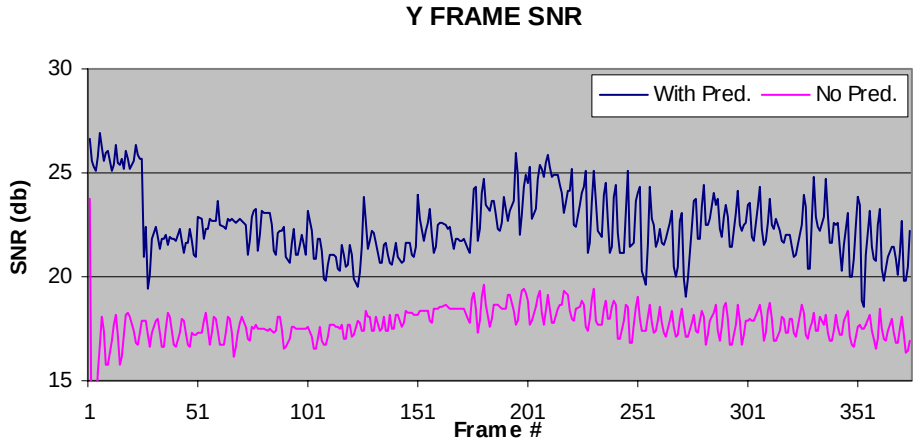


Figure 5.72: SNR of Y component at 1500kbps.

Notice that the average quality achieved by the "M-JPEG encoder without prediction" at 1300kbps (table 5.29) is almost the same with the average quality achieved by the "M-JPEG encoder with prediction at 1100kbps (table 5.30). The results are reproduced in the next table:

MJPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
With Ped	26961232	1:3.84	16.01	18.62	5.23	6.78	188
No Pred.	31711048	1:3.26	15.51	18.33	5.09	6.5	0

Table 5.32: Encoding results at same quality

From table 5.32 we reach the conclusion that the "M-JPEG encoder with prediction" can achieve approximately the same quality as the "M-JPEG encoder without prediction" with 14.9% less bits.

In the next table we present the encoding results of **the "reference" M-JPEG** encoder without adaptivity (and of course without prediction) at 1100kbps, 1300kbps and 1500kbps.

M-JPEG	Total Bits	Comp. ratio	Average SNR (db)				Average Predicted MCUs
			Y	Y- ROI	Cb	Cr	
1500kbps	38438544	1:2.69	18.69	17.06	6.0	8.41	0
1300kbps	31390400	1:3.3	16.94	15.32	5.3	7.3	0
1100kbps	26589632	1:3.89	15.44	13.64	4.86	6.32	0

Table 5.33: Encoding results of Reference M-JPEG encoder

From the above table we can notice that the M-JPEG encoder without adaptivity compresses ROIs with lower quality than the whole frame at all bit rates.
In addition, at the same bit rate it achieves higher overall frame quality than the "M-JPEG encoder without prediction" by 9.7% in average, but much lower ROI quality (approximately 19.4% lower).
The "M-JPEG encoder with prediction" achieves better quality of both ROIs and frame at all bit rates. However, at low bit rates (1100kbps), the improvement in the quality of the frame is very small (approximately 3.6%). At higher bit rates (1300kbps & 1500kbps) the improvement is approximately 19.4%. Finally, the "M-JPEG encoder with prediction" achieves at the same bit rate with the reference M-JPEG encoder approximately 46.5% higher quality of ROIs.

5.2.5.2 Encoding Speed

For the tests of the encoding speed we used the same parameters as described at the beginning of section 5.25 (table 5.16). The frame rate of all sequences is 25fps and the results stem from encoding at 1500kbps.

We encoded the same test sequences with both encoder several times and kept the fastest time. We calculate the actual time required to encode the test sequences; the time to read the various parameters required for the encoding process is not included. The time to read the input video from the disk and to store the compressed sequence back to the hard disk is included in the measurements.

The tests where performed on a general purpose desktop PC with an Intel Penitum4 at 2.53GHz processor. The results are presented in the next table.

	#frames	No Predictions	With Predictios
EnterExitCrossingPaths1cor.mpeg	383	9463 msec	16413 msec
TwoEnterShop2front.mpeg	375	8902 msec	16113 msec
TwoEnterShop1cor.mpeg	625	15041 msec	27870 msec

Table 5.34: Encoding delay results

In order to encode a sequence with frame rate 25 frame/sec in real time, an encoder should be able to encode a frame in maximum 40msec. The "M-JPEG encoder without prediction" requires, in average, 24.3msec/frame. The "M-JPEG encoder with prediction" requires in average 43.47msec/frame.

The "M-JPEG encoder without prediction" is 44% faster than the "M-JPEG encoder with prediction" and can compress a video sequence of resolution 384x288 and frame rate 25fps in real time.

5.2.5.3 Conclusions

From the above tests we conclude that both adaptive M-JPEG encoders manage to encode ROIs with better quality than non-ROIs. The "M-JPEG encoder with prediction" prooved more efficient than the "M-JPEG encoder without prediction" in terms of quality (especially in high bit-rates) but slower in terms of encoding delay and with higher quality variability and bit rate variability.

The variability of the "M-JPEG encoder with prediction" is mainly a result of the prediction decision scheme we use. The **TAD** of the original frame from the previous reconstructed frame **is not an efficient way to discriminate between MCUs that should be predicted** or not. From frame to frame the number of MCUs that are predicted, by using this measure, changes significantly and affects negatively the behaviour of the rate control algorithm. In addition, at low bit rates, where more predicted MCUs would improve the whole encoding process, less MCUs are predicted.

Finally, in low bit-rates, where the number of MCUs that are predicted is not very high (100-200 MCUs per frame in average, where 432 are all the MCUs in a frame), **the predictions don't seem to improve the quality** over the M-JPEG encoder that doesn't use predictions. The reason is that the quantized DC coefficients are coded separately from the quantized AC terms (see section 2.3.5). The DC coefficient is a measure of the average value of the 64 image samples. Because there is usually strong correlation between the DC coefficients of adjacent 8x8 blocks, the quantized DC coefficient is encoded as the difference from the DC term of the previous block in encoding order, a process called differential coding. By mixing predicted MCUs with non- predicted MCUs the strong spatial correlation ceases to exist and the frame is not coded efficiently. Thus, when the number of MCUs predicted in a frame is small, there is no gain in the encoding efficiency. Actually, the coding efficiency might become worse. In MPEG-2, in P- and B-frames, where predicted MBs are allowed, the DC terms of the quantized coefficients are treated just like the AC coefficients. One solution to this problem might be to predict all MCUs in a frame.

Regarding the encoding delay, the "M-JPEG encoder without prediction" manages to compress a sequence of 25fps frame rate and 384x288 resolution in real time, while the "M-JPEG encoder with prediction" not.

In summary, the conclusions we reach from the test results are:

- The "M-JPEG encoder with prediction" can at the same compression ratio with the "M-JPEG encoder without prediction" achieve in average **25.16% better frame quality** in terms of SNR and **19.8% better quality of regions of interest**.

- The “M-JPEG encoder with prediction” can achieve the same quality with the “M-JPEG encoder without prediction” with approximately **19.82% less bits**.
- The “M-JPEG encoder with prediction” achieves **14.57% higher quality of ROIs** than the whole frame, contrary to the reference M-JPEG encoder that encodes ROIs with lower quality.
- The “M-JPEG encoder without prediction” achieves **18.81% higher quality of ROIs** than the whole frame, contrary to the reference M-JPEG encoder that encodes ROIs with lower quality.
- The **bitrate variability** of the “M-JPEG encoder with prediction” is high due to the variable number of MCUs predicted in each frame.
- Due to the way the JPEG library works (it receives and compresses one MCU row in each call) it is hard to control accurately the bit rate.
- “M-JPEG encoder without prediction” is 44% faster than the “M-JPEG encoder with prediction” and achieves real time compression at 25fps and 384x288 resolution.
- “M-JPEG encoder with prediction” fails to compress in real time.

6. Final Conclusions

Our encoders managed to comress regions of interest adaptively. In average the improvement in the quality of regions of interest over non regions of interest was 16% (in all cases) in terms of Signal to Noise Ratio measured in db.

The modified MPEG-2 encoder showed a very stable behaviour of bit rate, and slightly higher quality variability than the original MPEG-2 encoder. In addition it achieved the highest compression ratios.

The "M-JPEG encoder without prediction" showed the most stable behaviour in quality but achieved much smaller compression ratios than MPEG-2. At compression ratio 1:2.83 it can achieve comparable quality with the modified MPEG-2 encoder at compression ratio 1:23.

Finally, the "M-JPEG encoder with prediction" didn't have very stable quality behaviour. The smaller the bit rate, the less MCUs where predicted per frame and the less the improvement in coding efficiency over the "M-JPEG encoder without prediction". In higher bit rates the improvement is more visible. However, the compression ratios achieved, were higher than those of the "M-JPEG encoder without prediction", especially in high bit-rates. Because of the unstable behaviour it is difficult to compare it with MPEG-2 but we could say that at compression rates at 1:3.25 it can achieve approximately the same quality as the modified MPEG-2 encoder at compression ratio 1:13.8

Our other goal was to estimate the encoding delay of the encoders and to reach a conclusion on whether it is possible to compress video in real time without the use of special hardware. Real time video compression is a very demanding task that is difficult to achieve without special hardware on a common PC. From the test results we show that only the "M-JPEG encoder without prediction" is capable of compressing a sequence of 25fps at resolutions of 384x288 and higher without problems in real time. In addition, we should point that the input of all the encoders was already downsampled data in YCbCr format. In real applications the additional time to perform color transformation and downsampling of the input sequence should be taken into account.

In summary, the modified MPEG-2 encoder achieves the better compression ratios with a stable behaviour both in terms of quality and bitrate but cannot compress a video sequence in real time without special hardware.

"M-JPEG without prediction" showed the best behaviour in terms of quality and bitrate variability. However, it achieves much smaller compression ratios than the modified MPEG-2 encoder. On the other hand it makes real time encoding on a common desktop PC feasible.

Finally, the "M-JPEG encoder with prediction" showed the most unstable behaviour. The quality improvement over the "M-JPEG encoder without prediction" heavily depended on the bitrate and the quality variability was petty high. The prediction of only a part of the MCUs in a frame we introduced was not a very good idea. Because of the differential encoding of the DC coefficients the compression ratio achieved for a certain quality depends on the number of predicted MCUs, or better, on the number of successive changes in not predicted MCUs and predicted MCUs. A better solution might be to

consider all the MCUs in a frame as predicted MCUs. In addition, it fails to compress a sequence of 25fps and 384x288 in real time. However, it is much faster than the Modified MPEG-2 encoder, and with some minor alterations or a faster CPU, real time compression might become possible.

7. Future Work

The most important improvement that applies to all implementations is to find a more efficient way to discriminate between ROIs and non ROIs. The TAD measure showed high variability in the regions (MBs or MCUs) it detected as ROIs from frame to frame. In addition TAD detects only moving objects. In applications that involve surveillance cameras a person just standing would be region of interest that the TAD measure fails to detect. However, a new scheme to detect ROIs shouldn't be very complex or else the encoding delay would increase.We should notice that the high variability in ROIs from frame to frame affected the rate control algorithms and their performance.

Another important point would be the improvement of the "M-JPEG encoder with prediction", and more particurlary the way predicted MCUs are detected. The TAD of a MCU from the previous reconstructed MCU at the same position (the scheme we used) or even the TAD from the previous original MCU at the same position are not efficient ways to take a decision. Maybe the best solution is to predict all MCUs. This would make the implementation faster and improve the coding efficiency and quality variability.

Finally, it might be better to make a precoding pass to compute the parameters α_i, α_p of the rate control algorithm used in the adaptive M-JPEG encoders. The "M-JPEG without prediction" would require only one precoding pass to estimate α_i for the first frame, while the "M-JPEG with prediction" two precoding passes, one for each type of frame.

Appendix A: Rate-Distortion theory

Rate distortion theory is the branch of information theory addressing the problem of determining the minimal amount of entropy (or information) R that should be communicated over a channel such that the source (input signal) can be reconstructed at the receiver (output signal) with given distortion D. As such, rate distortion theory gives theoretical bounds for how much compression can be achieved using lossy data compression methods. In the most simple case (which is actually used in most cases), the distortion is defined as the variance of the difference between input and output signal (i.e., the mean-squared error of the difference)

The functions that relate the rate and distortion are found as the solution of the following minimization problem:

$$\min_{Q_{Y|X}(y|x)} I_Q(Y;X) \text{ subject to } D_Q \leq D^*.$$

Here $Q_{Y\mid X}(y \mid x)$ is the conditional probability density function (PDF) of the communication channel output (compressed signal) Y for a given input (original signal) X, and $I_Q(Y ; X)$ is the **mutual information** between **Y** and X defined as

$I(Y;X) = H(Y) - H(Y \mid X)$

where $H(Y)$ and $H(Y \mid X)$ are the entropy of the output signal Y and the conditional entropy of the output signal given the input signal, respectively:

$$H(Y) = \int_{-\infty}^{\infty} P_Y(y) \log_2(P_Y(y))\, dy$$
$$H(Y|X) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} Q_{Y|X}(y|x) P_X(x) \log_2(Q_{Y|X}(y|x))\, dx\, dy.$$

The mutual information can be understood as a measure for *prior* uncertainty the receiver has about the sender's signal ($H(Y)$), diminished by the uncertainty that is left after receiving information about the sender's signal ($H(Y \mid X)$). Of course the decrease in uncertainty is due to the communicated amount of information, which is $I(Y; X)$.

As an example, in case there is *no* communication at all, then $H(Y \mid X) = H(Y)$ and $I(Y; X) = 0$. Alternatively, if the communication channel is perfect and the received signal Y is identical to the signal X at the sender, then $H(Y \mid X) = 0$ and $I(Y; X) = H(Y) = H(X)$.

In the definition of the rate-distortion function, D_Q and D^* are the distortion between X and Y for a given $Q_{Y\mid X}(y \mid x)$ and the prescribed maximum distortion, respectively. When we use the mean squared error as distortion measure, we have (for amplitude-continuous signals):

$$D_Q = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P_{X,Y}(x,y)(x-y)^2\,dx\,dy = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} Q_{Y|X}(y|x)P_X(x)(x-y)^2\,dx\,dy$$

As the above equations show, calculating a rate-distortion function require the stochastic description of the input X in terms of the PDF $P_X(x)$, and then aims at finding the conditional PDF $Q_{Y|X}(y \mid x)$ that minimize rate for a given distortion D^*.

Unfortunately, solving this minimization problem can be done only for few cases. However, although exact solutions are only available in a few cases, measured rate-distortion functions in real life tend to have very similar forms.

Memoryless (independent) Gaussian source

If we assume that $P_X(x)$ is Gaussian with variance σ^2, and if we assume that successive samples of the signal X are stochastically independent (the source is *memoryless*, or the signal is *uncorrelated*), we find the following analytical expression for the rate-distortion function:

$$R(D) = \begin{cases} \frac{1}{2}\log_2(\sigma_x^2/D), & \text{if } D \le \sigma_x^2 \\ 0, & \text{if } D > \sigma_x^2 \end{cases}$$

The following figure shows what this function looks like

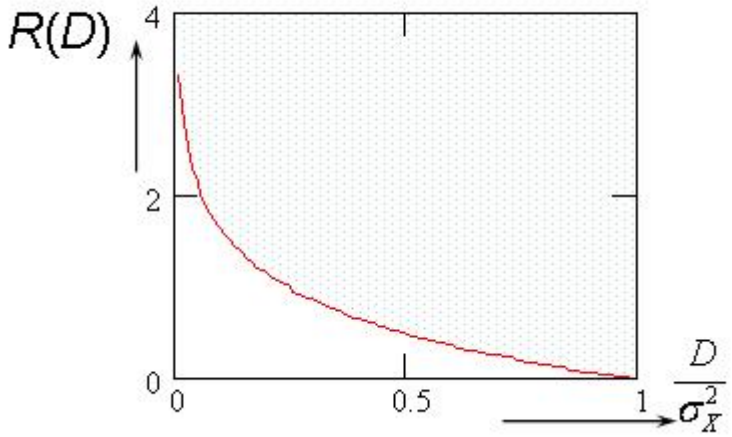


Fig. A.1: R-D curve

Rate distortion theory tell us that *no compression system exists that performs outside the green dotted area*. The closer a practical compression system is to the red (lower) bound, the better it performs. It should be emphasized that this rate-distortion function holds only for Gaussian memoryless sources. The performance of a practical compression system working on -- say -- images, may well below the *R(D)* lower bound shown.

Appendix B: HVS and JPEG

In this section we describe some papers that take advantage of the tools provided by the JPEG specification and more specifically variable quantization.

B.1 Perceptual Adaptive JPEG Coding

In [6] a method for perceptual optimization of the set of multipliers in adaptive JPEG based on their imporatance in the HVS is proposed.

B.1.1 Perceptual Error Metric

For adaptive quantization, a measure of the local perceptual error is needed. The perceptual metric developed in [7] is adopted. In that metric, the quantization errors for each coefficient in each block are scaled by the corresponding visual sensitivities to each DCT basis function in each block. These sensitivities are determined by three factors: contrast sensitivity, luminance masking and contrast masking.

The contrast sensitivity function: The CSF is a function of the (bi-dimensional) spatial frequency f that gives the value of the sensitivity of the HVS v_{ij} for a visual stimulus of given frequency f when it is displayed on a uniform background (usually the luminance value of the background is chosen as the middle-range value, i.e. 128 for 8 bit/pixel images).

The v_{ij} for each DCT basis function has been measured by Ahamuda [7] who also provided a formula for approximating this sensitivity under a variety of viewing conditions. According to [7] the contrast sensitivity is given by the following function:
$V_{ij}=1/CT$, where CT is the visibility threshold, (see B.2).

Luminance masking function: This function describes how the visual threshold varies when the luminance of the background on which the visual stimula are displayed is different from the standard middle-range value used for determining the CSF. Experimental results show that the visibility thresholds increase on both low luminance backgrounds and high luminance backgrounds; middle range luminance values yield the lower visibility thresholds. The luminance masking effect is modelled by means of a function that gives an adaption factor for each value of background luminance: the corrected visibility thresholds are therefore obtained by multiplying the initial values (computed with the CSF) by the adaptation factor. While the CSF models a frequency-related property of the HVS, thus giving a fixed visibility threshold for each subband, the luminance masking effect (as well as the contrast masking effect described below) is a spatial domain characteristic: as a result, the luminance masking function allows to spatially adjust the visibility thresholds values, thus leading to locally adaptive quantization schemes that allocate less bits for coding coefficients corresponding to spatial locations with high visibility thresholds.

Watson suggested that one may approximate this luminance masking by computing block contrast sensitivities:

$v_{ijk} = v_{ij}(DC_k/\hat{c})^{-aT}$, i,j = 1,2,..8 (position in a block), k: number of block

Where DC_k is the DC coefficient of block k, $\hat{c}$ is the mean luminance of the display and aT determines the degree of masking (a value of 0.65 is recommended).

Contrast masking function: This function accounts for the reduced visibility of a visual stimulus (referred to as the target signal) when it is displayed on a non-uniform background. Precisely, the actual visibility of the target signal depends on the contrast value and the frequency content of the background signal. Experimental results obtained from subjective tests on human observers show that the higher is the value of the background signal contrast, the higher becomes the visibility threshold for the target signal; in addition, this effect is more evident when the frequency contents and the spatial orientations of the target and background signals are similar. By taking into account the contrast masking effects, spatially adaptive quantization schemes are obtained.

Following Watson and model this contrast masking as

$s_{ijk} = v_{ijk} * \min[1, |c_{ijk}| * v_{ijk}]^{-d\_mask_{ij}}$, i,j = 1,2,..8 (position in a block), k: number of block

Where s_{ijk} is the block masked contrast sensitivity and d_mask_{ij} determines the degree of contrast masking ($d\_mask_{00}=0$ and $d\_mask_{ij}=0.7$ is recommended).

Perceptual error in each frequency of each block is then computed by multiplying the quantization error e_{ijk} by the block masked contrast sensitivity:

$dp_{ijk}=e_{ijk}*s_{ijk}$

To get the total perceptual error the errors over both frequency and space are pooled. To simplify the optimization procedure:
$P=\max(dp_{ijk})$.

B.1.2 Optimization

The procedure for optimizing the multipliers for a given total perceptual error, P, is quite parallel to that used by Watson in [8] to find the optimal quantization matrix.

First, following the optimization procedure introduced by Watson the optimal quantization table for the specific image is found and stored in the bitstream.

Then, the quantization scale factors are optimized. Since P is equal to the maximum block perceptual error p_k, the minimum bitrate for a given P is obtained when all $p_k=P$. Therefore the optimization reduces to separately adjusting each scale factor until the block perceptual error equals to the desired total perceptual error.

B.2 The Luminance Detection Model [7]

The luminance detection model predicts the threshold of detection of the luminance error image generated by quantization of a single DCT coefficient in position i,j of a block. The subscript Y is used for luminance. This error image is assumed to be below the threshold of visibility if its zero-to-peak luminance is less than a threshold $CT_{i,j}$ given by:

$$\begin{aligned}\text{Log } CT_{ij} &= P(\, fi_j \,;\, b_Y,\, k_Y,\, f_Y) \\ &= \log b_Y && , \text{ if } f_{ij} <= f_Y \,, \\ &= \log b_Y + k_Y\, (\log f_{ij} - \log f_Y)^2 && , \text{ if } f_{ij} > f_Y \,.\end{aligned}$$

Where

$f_{ij} = (1/2N)((i/\text{PixSp}_x)^2+(j/\text{PixSp}_y)^2)^{0.5}$, is the spatial frequency, f_{ij}, associated with the i,jth basis function and PixSp_x and PixSp_y are the horizontal and vertical pixel spacings in degrees of visual angle.

$b_Y = s\ T_Y\ /\ \theta_{ij}$: The parameter s is a fraction, $0 < s <= 1$, to account for spatial summation of quantization errors over blocks. We set it to unity to model detection experiments with only one block. The summation results suggest that it should be equal to the inverse of the fourth root of the number of blocks contributing to detection. We suggest the value $s = 0.25$, corresponding to 16× 16 blocks. The factor T_Y gives the dependence of the threshold on the image average luminance Y_0.

$$\begin{aligned}T_Y &= Y_0^{a_T}\ Y_T^{1-a_T}\ /\ S_0 && ,\ Y_0 <= Y_T \\ T_Y &= Y_0\ /\ S_0 && ,\ Y_0 > Y_T \,,\end{aligned}$$

where suggested parameter values are $Y_T = 15$ cd/m^2, $S_0 = 40$, and $a_T = 0.65$.

The product of a cosine in the x with a cosine in the y direction can be expressed as the sum of two cosines of the same radial spatial frequency but differing in orientation. The factor

$$\theta_{ij} = r + (\, 1 - r \,)\, (1 - [\, 2\, f_{i0}\, f_{0j}\ /\ f_{ij}^2 \,]^2 \,)$$

accounts for the imperfect summation of two such frequency components as a function of the angle between them. Based on the fourth power summation rule for the two components when they are orthogonal, r is set to 0.6. An additional oblique effect can be included by decreasing the value of r.

The parameters f_Y and k_Y determine the shape of P and depend on the average luminance Y_0.

$$f_Y = f_0 \; Y_0^{a_f} \; Y_f^{-a_f} \quad , \; Y_0 <= Y_f$$
$$= f_0 \quad , \; Y_0 > Y_f$$

and

$$k_Y = k_0 \; Y_0^{a_k} \; Y_k^{-a_k} \quad , \; Y_0 <= Y_k$$
$$= k_0 \quad , \; Y_0 > Y_k$$

where $f_0 = 6.8$ cycles/deg, $a_f = 0.182$, $Y_f = 300$ cd/m^2, $k_0 = 2$, $a_k = 0.0706$, and $Y_k = 300$ cd/m^2.

Notation

W	Quantization matrix.
C_{ij}	DCT coefficient value at block position ij.
QC_{ij}	Quantized value of DCT coefficient at block position ij.
QS	Quantization scale factor.
QP	Quantization parameter-quantization scale code (1-31).
Q_{REF}	Reference quantization parameter.
R	Rate; number of bits.
pt	Picture type (I, P or B).
S_{pt}	Number of bits generated by encoding the picture of type pt.
Q_{pt}	Average quantization parameter used for encoding the picture of type pt.
X_{pt}	Global complexity measure for the pictures of type pt.
bit_rate	Target bit rate of sequence.
T_{pt}	Target bits for the picture of type pt.
K_p, K_b	Constants dependent on the quantization matrices.
Rem	Remaining number of bits assigned to the GOP.
GOP	Group of pictures.
picture_rate	Rate of pictures.
N_p	Number of P-pictures remaining in the current GOP in the encoding order.
N_b	Number of B-pictures remaining in the current GOP in the encoding order.
B_j	Number of bits generated by encoding all macroblocks in the picture up to and including j.
MB_cnt	Number of macroblocks in the picture.
$b\_f_j^{pt}$	Fullnesses of virtual buffers at macroblock j- one for each picture type.
r	Reaction parameter.
act_j	Activity of macroblock j.
N_actj	Normalized activity of macroblock j.
σ^2	Variance.
d_i	Power of the distortion induced by quantization.
D	Distortion.
MD_p	Quantization noise in a macroblock with perceptual index p.
β	Target quantization noise.
ac	AC DCT coefficients.
QAC	Quantized AC DCT coefficient.
qt	Quantization type (intra-i or non intra-n).
REC	Reconstructed value of quantized AC coefficients.
Sbit(QP)	Number of signal bits to be generated from the DCT coefficients of a frame at a given QP level.
NZC	Non zero coefficients.
DC_e	Sum of the distortion from the intra-DC components.
E(QP)	Total number of bits to be generated at a given QP level.

$X_{1,2}$	First and second order model coefficients.
CSF	Contrast sensitivity function.
HVS	Human Visual System.
v_{ij}	Sensitivity of the HVS in each frequency.
CT	Visibility threshold.
ĉ	Mean luminance of the display.
Obit	Overhead bits.
s_{ijk}	Block masked contrast sensitivity in each frequency of each block k.
d_mask_{ij}	Constant that determines the degree of contrast masking.
e_{ijk}	Quantization error in each frequency of each block k.
Dp_{ijk}	Perceptual error in each frequency of each block k.
aT	Constant that determines the degree of luminance masking.
UQP(i,j)	QP level that makes the absolute value of the (i,j)th components of the quantized DCT coefficients into 1.
TR_f	Bits transmitted during a frame period.
k_b	The frame distance (in coding order) from the previous anchor frame to the current frame.
MAD	Mean Absolute Difference.
TAD	Total Absolute Difference.
TSD	Total Squared difference.

References

[1] *"Information Technology-Digital compression and coding of continuous-tone still images-requirements and guidelines"*, the international telegraph and telephone consultative committee, ITU 1992.

[2] *"The JPEG still Compression Standard"*, G. K. Wallace, IEEE transaction on Consumer electronics, 1991.

[3] "*Adaptive model-driven bit allocation for MPEG video coding"*, Bo Tao, Dickinson, B.W.; Peterson, H.A.; IEEE Transactions on Circuits and Systems for Video Technology Volume 10, Issue 1, Feb. 2000 Page(s):147 – 157.

[4] TM5 rate control algorithm.

[5] *"Rate control of MPEG video for consistent picture quality"*, Sung-Hoon Hong; Sang-Jo Yoo; Si-Woong Lee; Hyun-Soo Kang; Sung Yong Hong; IEEE Transactions on Broadcasting Volume 49, Issue 1, March 2003 Page(s):1 – 13.

[6] *"Perceptual adaptive JPEG coding"*, Rosenholtz, R.; Watson, A.B.; Proceedings, IEEE International Conference on Image Processing Volume 1, 16-19 Sept. 1996 Page(s):901 - 904.

[7] *"A visual detection model for DCT quantization"*, A. J. Ahumada Jr., A. B. Watson, & H. A. Peterson, "AIAA Computing in Aerospace, pp. 314-318, San Diego, CA, 1993 A. B. Watson, "DCT quantization matrices visually optimized for individual images," Proc. SPIE, 1913:202-16, 1993.

[8] "*DCT quantization matrices visually optimized for individual images"*, A. B. Watson, Proc. SPIE, 1913:202-16, 1993.

[9] "*Video Adaptation: Concepts, Technologies and Open Issues"*, Shih-Fu Chang and Antony Vetro, Proc. of IEEE, vol. 93, no.1, Jan 2005.

[10] "*Improved estimation for just-noticeable visual distortion"*, X.H. Zhang, W.S. Lin, P. Xue, Elsevier Signal Processing 85, 2005.

[11] *"Object –Based Coding for Long term archive of surveillance video"*, Anthony Vetro, Tetsuji Haga, Kazuhiko Sumi, Huifang SU, IEEE, 2004.

[12] The official site of MPEG, http://www.mpeg.org.

[13] The official site of the IJG (independent JPEG group), http://www.ijg.org.

[14] *"Semantic Video Adaptation based on Automatic Annotation of Sport Videos"*, M. Bertini, R. Cucchiara, W. Nunziati, A. Prati, International Multimedia Conference, Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval, 2004.

[15] "*Improved Single-Video-Object rate control for MPEG-4"*, King N. Ngan, Zhenzhong Chen, IEEE transactions on circuits and systems for video technology, VOl. 13, No.5, May 2003.

[16] "Visual Data Compression for Multimedia Applications", T. Ebrahimi, M. Kunt, Proceedings of the IEEE, Vol. 86, No. 6, June 1998.

[17] *"MPEG-4 Rate Control for Multiple Video Objects"*, Anthony Vetro, Huifang Sun, Yao Wang, IEEE transaction on circuits and systems for video technology, Vol. 9, No.1, February 1999.

[18] *"Rate control and Bit allocation for MPEG-4"*, Jose Ronda, Fernando Jaureguizar, Narsico Garcia, IEEE transactions on circuits and systems for video technology, VOl. 9, No. 8, December 1999.

[19] "*MPEG Video Compression Standard"*, J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, D. J. LeGall, Chapman & Hall editions, 1997.

[20] "*Efficient Algorithms for MPEG Video Compression"*, Dzung Tien Hoang, Jeffrey Scott Vitter Wiley editions, 2002.