Real Time Video Transmission in UMTS

Postgraduate Thesis

in

Information and Communication

Technology

By

Frode Ørbeck Hansen

Auckland, New Zealand May 2001

Supervisor

Lars Schei

Abstract

Universal Mobile Telecommunications System is denoted as a 3rd cellular system and has been designed with the objective to be a system with global coverage. With the major improvement of in bandwidth capability the Universal Mobile Telecommunications System will be able to support real time multimedia services.

This thesis starts with a presentation of the possible video services that can be exploited in third generation cellular systems. Video services are known to be bandwidth consuming and have rigid requirements to the real time requirements. Though the major increase in available bandwidth in the Universal Mobile Telecommunications System it is not sufficient to support transmission of uncompressed video over wireless networks.

MPEG-4 and H.263 (+) are video codecs designed to provide high compression efficiency and error resilient transmission in heterogeneous error prone environments. These two codecs contains post-processing mechanisms for to conceal errors due to bit errors, burst errors, packet loss etc. In addition, the codecs use scalable coding to confront the constantly varying bit-rates available in a wireless network. Scalable coding means that the resolution of a video can be lessened or the quality of the video may be sacrificed to meet the available bandwidth in the network.

Preface

This thesis is performed to complete the Master of Science programme in Information and Communication Technology (ICT) at Agder University College, Faculty of Engineering and Science in Grimstad Norway.

In this context I would like to thank first of all Lars Line and the International office at Agder University College who made it possible for me to complete my degree as a "study abroad" programme in New Zealand. Further, I would like to thank Ericsson AS in Grimstad for their contribution of a scholarship and Ericsson Communications Ltd in Auckland New Zealand who gracefully supplied me with the necessary equipment and office space to perform my thesis.

I would also like to direct a "big thank you" to Lars Schei at Ericsson AS in Grimstad who formulated my thesis initiative and gave me superb guidance during the entire project period. Furthermore thank you, Stein Bergsmark at Ericsson AS in Grimstad, for giving me helpful advice and for your positive attitude.

Last, but not least I would like to thank my girlfriend, Janicke S. Gjestad, for being a loving and caring person regardless of my mood swings and long working hours during the completion of this thesis.

Frode Ørbeck Hansen

Auckland, New Zealand May 2001

Table of Contents

List of Figures

List of Tables

1 Introduction

Mobile telecommunication equipment is able to transfer voice, audio, data and video faster than ever done before. UMTS (Universal Mobile Telecommunications System) enabled mobile devices make it possible for users to be connected to the Internet at any time. In this manner mobile telecommunication and Internet "melts" together and become one universal network consisting of both the wireless and fixed telecommunications network.

Market forecasts for mobile services are enormous. In Norway today, 70 percent of the population is carrying a GSM telephone and it is forecasted that in about 10 years the same amount of users will find interest in UMTS (Telenor, 2001 and Netcom, 2000). The UMTS Forum, which represents the global mobile industry, estimates that by 2010 there will be around 2 billion mobile users Worldwide.

With UMTS, a bit-rate of up to 2 Mbit/s will be available to indoor, outdoor and vehicle-based users. With this major increase in bandwidth from GSM's capacity of 9.6 kbit/s, opens an entire New World of service possibilities. Increasingly exploited area of services on Internet is the utilisation of real time multimedia services such as video on demand and video conferencing (video chatting). The imminent arrival of UMTS will enable users to utilise these services and many believe that these services are going to be extensively exploited in UMTS.

The challenges become substantial introducing services with considerable demands to the quality of service. This yields specially for services like video conferencing, which requires that the communication session is performed in real time. In order to make the video services financially attractive for the mobile users, the amount of bandwidth required for the service must be reduced to a minimum without sacrificing the quality of the video. Transmission of data that requires high quality is a major issue in wireless networks and this type of network tends to be error prone due to certain atmospherically disturbances (e.g. attenuation, reflection or diffraction), which affect the transmitted

signal. As a result of this, state of the art video compression techniques like MPEG-4 (Motion Picture Experts Group) and H.263 (+), have been developed to support very low bit-rate video compression and error robustness for video transmission over error prone networks such as UMTS.

1.1 Thesis scope

The scope of this thesis is to conduct a study on video services and coders for use in mobile applications. First, the task is to give an overview of the various video services that is predicted to be widely used in UMTS.

Even though UMTS makes available a bit-rate that will enable the transfer of video content, compression of the video have to be performed due to the fact that the bit-rate a uncompressed video requires is enormous (about 216 Mbit/s) compared to what UMTS offers (maximum 2Mbit/s). Therefore a study of the commonly used video compression techniques and a comparison of the most common ones will be performed.

Furthermore, a study to determine methods to evaluate loss of information due to encoding, decoding and noise will be carried out. The methods to evaluate loss shall be used to evaluate the performance of the video, with relatively high real time requirements. A simulation environment is developed to study how the respond towards different kinds of “mobile” noise (with a view to UMTS).

1.2 Objective

The objective in this thesis is to give the reader an overview over the video services that are going to be widely used in UMTS. Moreover, a detailed description of various video codecs is performed to get an understanding for the importance of video compression and consolidation of the video quality in UMTS. Finally, after a simulation process, the objective is to give a presentation on how the various kinds of noise can effect the quality of the video.

1.3 Known limitations

Continuously changing bit-rate will not be considered in the simulation due to added costs in testing equipment.

1.4 Methodology

Most of the work is based on examination of research reports and technical reports from standardisation organisations. A lot of testing has been performed on the error resilience of various video codecs. Therefore, a study of several test reports found on WWW was performed in order to be able to develop the simulation environment.

1.5 Outline

The thesis is structured in eight chapters, which will be described here briefly. Chapter 2 gives first an introduction to UMTS where the essential areas together with the architecture of UMTS are discussed. Based on this a short overview of the possible services is presented in the same chapter.

In chapter 3 gives an extended overview of the video services with emphasis on the technology and the real time characteristics for each service.

The next chapter (chapter 4) is the most comprehensive one and examines various video codecs. The video codec H.263 and MPEG-4 are discussed in detail due to its importance for video transmission in error prone networks.

Chapter 5 provides an overview of the various factors that has an effect on the quality of a transmission in error prone networks. In addition, methods for evaluation quality of the video after transmission are discussed.

Chapter 6 contains the simulation results and performance evaluation. The simulation results are discussed further in chapter 7.

The final chapter 8 contains the concluding remarks and suggested further studies.

2 UMTS Services

The following section presents a brief introduction to UMTS. This section will highlight the process of UMTS and finally the services that can be of interest for users of UMTS terminals.

2.1 A brief introduction to UMTS

Universal Mobile Telecommunications System (UMTS) is denoted as a 3rd cellular system and is developed within the framework that has been defined by the International Telecommunications Union (ITU) (known as the European version of IMT-2000). The process of standardising UMTS is carried out by the European Telecommunications Standards Institute (ETSI) in co-operation with other regional and national standardisation bodies around the globe to produce a standard that meets the needs of a growing market (UMTS Forum Report No. 2, 2000).

2.1.1 Mobility and coverage

UMTS has been designed with the objective to be a system with global coverage. This is done through introducing both terrestrial and global satellite components. The UMTS terminals will also be backward compatible with the 2nd generation systems, which makes it possible for a terminal to roam between the various networks (Inter-Network Roaming) (UMTS Forum Report No. 2, 2000).

ITU has set some indicative minimum requirements concerning data rates UMTS must support. The users of mobile services today moves between indoor, outdoor, urban and rural environments where the mobile terminals moves in a speed ranging from practically stationary (e.g. office environments) up to very high speeds (e.g. a car). The requirements ITU has laid down are defined according to the mobility of the UMTS terminal. In UMTS different cells of hierarchical structure is defined where the data rates available in each cell is dependent on the level of mobility (speed) of the UMTS terminal and

population density. Figure 1 (UMTS Forum Report No. 2, 2000) depicts hierarchical assembly of the cell classification that is typical for UMTS.

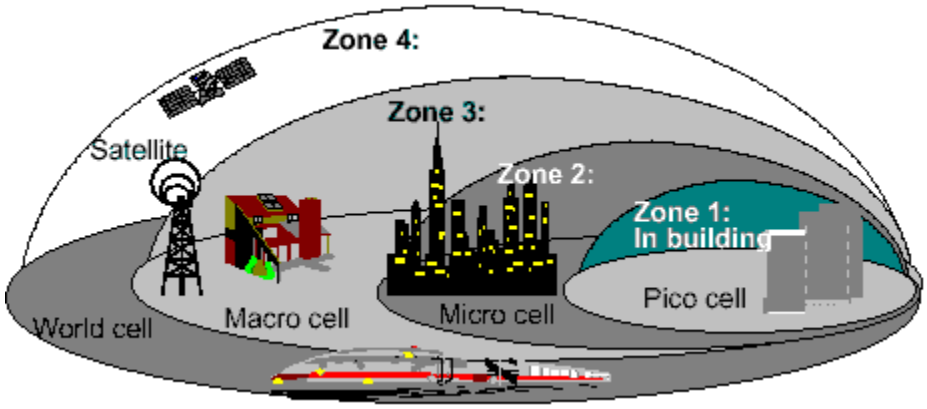

Figure 1 UMTS coverage

The cell types in UMTS is divided into:

- **Pico cell** – These are cells of limited radius that provides coverage in areas such as train stations, airports and office landscape.
  - Bit-rate: <=2048 kbit/s
  - Terminal speed (Mobility): < 10 km/h
  - Cell radius: 10 – 50 m

- **Micro cell** – These cells are directed to provide coverage in dense areas (e.g. a pedestrian located between buildings).
  - Bit-rate: 384 – 2048 kbit/s
  - Terminal speed (Mobility): < 120 km/h
  - Cell radius: 500 m

- **Macro cell** – Provides coverage in larger areas such as suburbs or more rural areas.
  - Bit-rate: 144 – 384 kbit/s
  - Terminal speed (Mobility): 120 – 500 km/h
  - Cell radius: 2 - 6 km

- **World (Global) cell –** Makes UMTS services available globally through satellite technology (remote areas such as mountains or oceans).
    - Bit-rate: 144 – 384 kbit/s
    - Terminal speed (Mobility): < 1000 km/h

2.1.2 UMTS architecture

The UMTS architecture is an evolution from GSM phase 2 + due to the fact, the UMTS reuse the core network in Global System for Mobile Communication (GSM) and General Packet Radio System (GPRS). An overview over the UMTS architecture is depicted in figure 2.

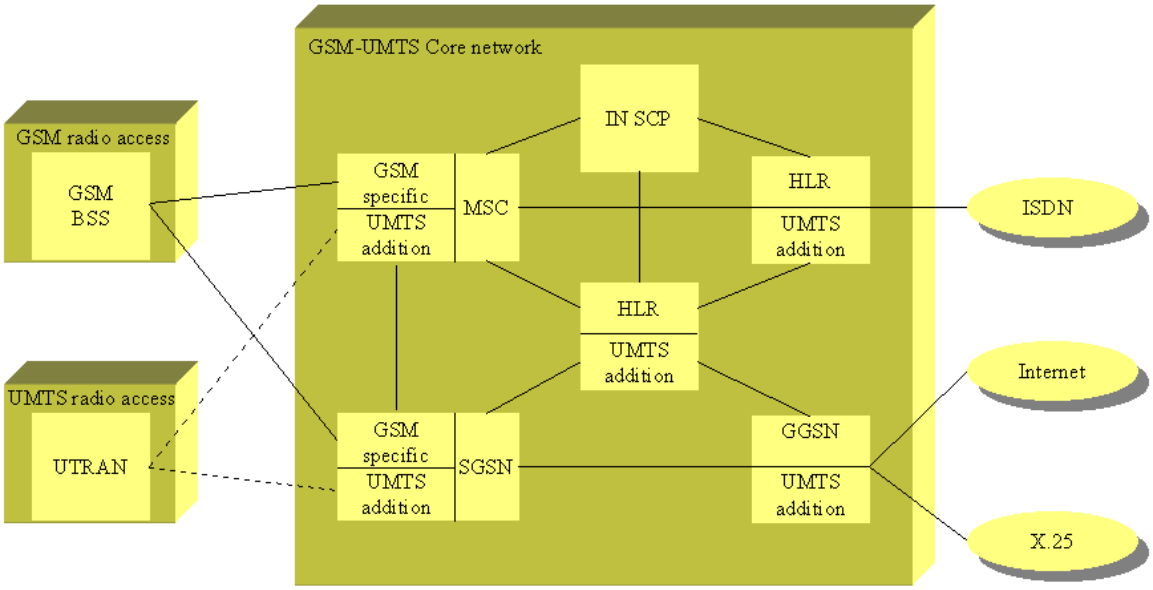

Figure 2 UMTS architecture

The UMTS architecture utilises a module principle where the access- and core network is clearly separated and developed more or less independently. To accomplish the high bit-rates a radio access system called UTRA (UMTS Terrestrial Radio Access) was developed. The UTRA has a maximum transfer capability of 2 Mbit/s in the frequency spectrum of 1880 to 2170 MHz (Vinck, 1998).

Because of the major increase in available bandwidth, UMTS is able to support real time services including multimedia as well as packet data services. There are two transport mode candidates for the core network, ATM (Asynchronous Transfer Mode) and IP (Internet Protocol). The intention was that UMTS was going to be based exclusively on IP, but many believes that ATM will be selected as transport mechanism for UMTS in the core network, while IP will be used for routing of packet data in the core network (Neubauer, 2000).

2.2 Services

The data rates that will be offered by the UMTS standard will make it possible to introduce a great deal of new applications. In the 2nd system, only speech and transfer of small amount of data were supported. Whereas in the 3rd generation system users will be able to use services such as games-on-demand independently of location and time, or watching yesterday's nine o'clock news on a train (UMTS Forum Report No. 11, 2000).

The services that UMTS is offering are not only for leisure. Services for business such as video conferencing makes it possible for e.g. a manager to attend to a meeting without sitting in a specific environment. In addition, the hospitals can make use of the telemedecine service by sending detailed situation reports from for example a car crash. This can help the medical team at a hospital to get a better understanding over the situation and perform the necessary preparation prior to the patient(s) arrival.

Another possible market where UMTS can be benefited from is the automobile industry. Already, car manufactures have made field trials to install GSM chipsets into vehicles, which monitors the car's performance. This telematics service can send warnings to the manufacturer indication the problem. Further, telematics is planned to be used to install a navigation system in the vehicles that would include real time mapping and street guidance service (UTMS Forum Report No. 11, 2000).

Many services will be launched when the UMTS is available on the commercial market. UMTS Forum has made a list of possible services that may be offered to the consumers. The various services are listed in table 1 below (UTMS Forum Report No. 11, 2000).

Table 1 List of various UMTS services

Info.	Education	Entertainment	Community	Business And Financial	Communication	Telematics and special
Browsing	Virtual school	Audio on demand	Emergency services	Virtual banking	Video telephony	Telemedicine
Interactive shopping	On-line library	Games on demand	Government procedures	On-line billing	Video conferencing	Security monitoring
	Training	Video Streaming		Universal SIM-card and Creditcard	Voice response and recognition	Instant help line

3 Video Services in UMTS

Over the last decade the multimedia services on the Internet has flourished. News stations broadcasts streaming news on the Internet and users are able to download music video's to be played back on their computer with help of e.g. Windows Media Player. Furthermore, have consumers been able to download a presentation of e.g. "How to brew your own beer" with PowerPoint slides with additional speech from a presentation lecturer. Because of the new capabilities that UMTS brings, these scenarios may also become a reality within the telecommunication business.

Multimedia services are telecommunication services that handle two or more types of media (e.g. video and speech) simultaneously. The multimedia service can involve multiple parties, connections and with the possibility to add or delete resources or users within a single communication session (ITU-T F.700, 2000).

Several different media components can be used in a multimedia service call. The components that are regarded as multimedia components are audio (e.g. AMR MP3 and WAV), still images (e.g. JPEG and GIF), graphics and lastly video (e.g. H.263, MPEG-4 and QuickTime) (3GPP TS 23.140, 2000). In table 2 it is shown the improvement in the download time for the different platforms (UMTS Forum Report No. 11, 2000).

Table 2 Platforms for Multimedia Applications

Services	**2G**	**PSTN**	**ISDN**	**2G+**	**UMTS**
E-mail (10kbyte)	8 s	3 s	1 s	0.7 s	**0.04 s**
Web-page (9kbyte)	9 s	3 s	1 s	0.8 s	**0.04 s**
Text file (40kbyte)	33 s	11 s	5 s	3 s	**0.2 s**
Large Report (2Mbyte)	28 min	9 min	4 min	2 min	**7 s**
Video clip (4Mbyte)	48 min	18 min	8 min	4 min	**14 s**
Film with TV Quality	1100 hr	350 hr	104 hr	52 hr	**>5 hr**

The focus of this thesis is on the video services that UMTS can provide. Market researchers have different meanings of which service is going to be the “killer” service (the most profitable). In the 2nd-generation system, the SMS service was significantly underestimated by the telecommunication branch and became far more lucrative service than they first thought. In the light of this, perhaps the Multimedia Messaging Service is going to be the most cost-effective service in UMTS. However, NTT DoCoMo regards the video streaming service to be the “killer” application (NTT DoCoMo, 2000), where the consumers can watch the daily news or view a video independent of location or time.

Regardless of the discussions about which of the UMTS services that are going to become the favourite among the consumers, video services is going to be popular and widely used. This chapter discusses the different kinds of video services, envisaged to be widely used in UMTS, in detail.

3.1 Video conferencing

3.1.1 General

A video conferencing service is defined to be an audio visual conversational conference service providing two-way real-time transfer of voice and video between groups of users in two or more separate locations. Albeit the audio and video data are the most fundamental parts of the service, other types of data, such as still pictures, text or graphics may also be exchanged (ITU-T F.702, 1996).

The video conferencing service is a service that enables a user to attend to meetings without being at the meeting physically. Today, video conferencing is done with help of the fixed telephone network. This requires that e.g. a sales employee have to go to a particular physical location. A third generation network makes it possible for the sales employee to be on the move while attending to a meeting. The sales person can be present at the meeting while sitting on a bus, or sitting in a car while stuck in traffic.

3.1.2 Video conferencing services defined

The video conferencing service has high requirements regarding real-time characteristics. During a videoconference, the latency ought to be kept to a minimum. This means that the round trip delay between the parties involved in the conversation is as low as possible (<400ms) in order to get an efficient transfer of the audio and video information. Usually one doesn't want to have buffering (low buffering) of data in a videoconference because it would cause an extra delay component in the conversation. In spite of the great enhancements in network capacity, network providers are interested in offer a Constant Bit Rate (CBR) to the participants of the videoconference. Thus, it is easy for the network provider to assign system resources to the conference. However, when the network becomes congested, it is desirable to offer Variable Bit Rate (VBR) which produce traffic that varies between providing CBR and being idle. Transmission errors caused between the terminals are not corrected due to this would add delay to the transfer. This means that if an error occurs, the quality of the information may need to be compromised to meet the latency requirements (3GPP2 S.R0022, 2000).

3.1.3 The technology

The video conferencing service consists of several components. These components are responsible for encoding, transmission, interpretation and decoding of the information exchange between the participants. There is not set any standards for what kind of video codec that will be used by the UMTS terminals, but 3GPP2 have suggested that both codec's H.263 and MPEG-4 is supported (3GPP2 S.R0022, 2000). These two standards are designed to support transmission at low bit-rates. The report from 3GPP2 states that the video conferencing service shall operate with a throughput of 32kbit/s and higher for both packet-switched (PS) and circuit-switched (CS) mode. For the packet switched services will an international standard packetisation scheme such as ITU-T H.225 be used. For the circuit switched services, will an international standard circuit switched such as ITU-T H.223 be used. The H.223 consists of two layers that control the errors for the various streams, provides multiplexing/demultiplexing for the assorted streams and adaptation to the underlying physical layer (Ericsson, 2000).

Depending of the type of service the audio information and video information is packetised/multiplexed before it is sent to the participants of the videoconference. At the recipients, the information is depacketised/demultiplexed before it is decoded with the corresponding video and audio codecs. The scope of the video conferencing service is presented in figure 3 beneath (Ericsson, 2000).

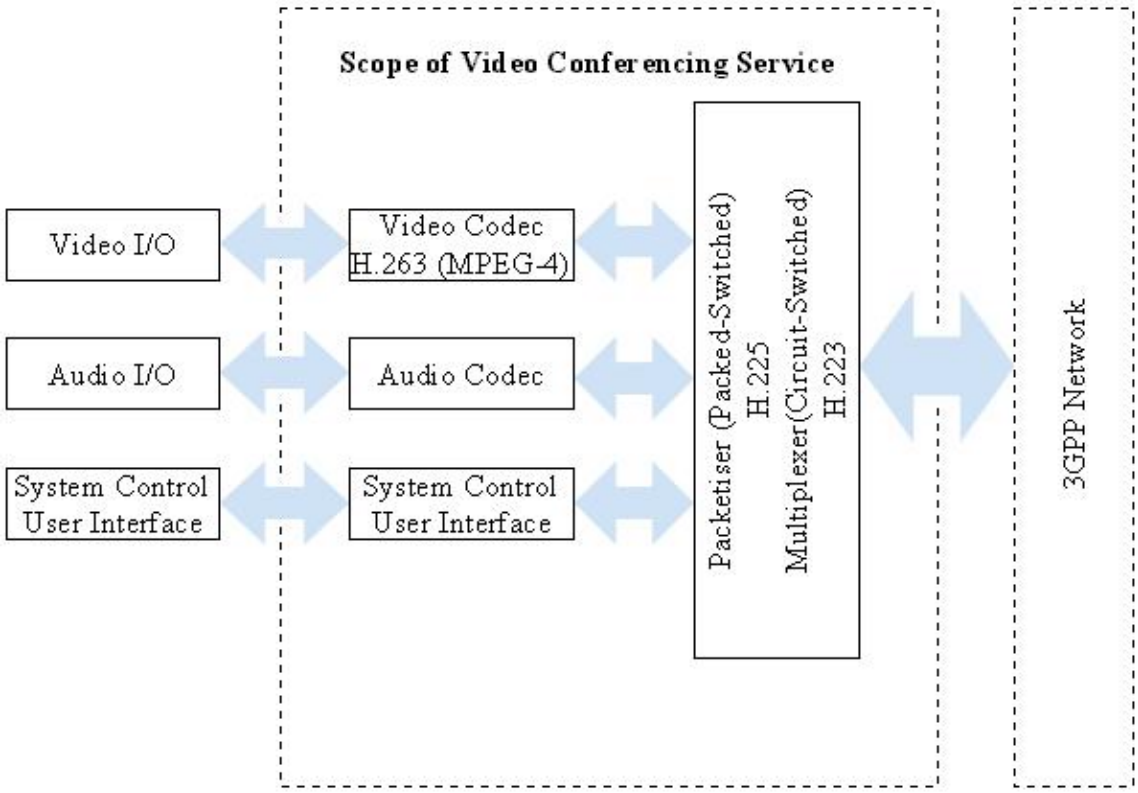

Figure 3 Scope of Video Conferencing Service.

3.1.4 Quality of Service

As for most of the multimedia services, the video conferencing service has high the Quality of Service (QoS) requirements. First, it is important that the quality of the video is of acceptable standard. In UMTS, there are great deals of factors that intervene with upholding the quality of the video service. The conferencing services shall, in accordance to 3GPP2 support bi-directional transmission of video in wireless packet- and circuit-switched transmission networks.

In table 3 the QoS features for the video conferencing service listed (3GPP2 S.R0022, 2000).

Table 3 QoS features for the video conferencing service.

Feature	**Description**
System	Shall be compatible to international standardised video conferencing systems (e.g. ITU-T H.324/H.323, IETF RFC 243 SIP). Shall be able to exchange controlling information with peers (e.g. ITU-T H.245, H225)
Synchronisation	The service should maintain synchronisation between the video and audio components. Inter-media skew < 20 ms
Video codec	The video codec should be standard compliant (e.g. MPEG-4 and H.263). Optional addition of coding efficiency and error robustness should be available
Speech codec	Appropriate speech codec
Throughput	> 32 kbit/s, incl. 384 kbit/s and 128 kbit/s for packet- and circuit-switched modes
Multiplexing	International standard multiplexing scheme (e.g. ITU-T H.223) for speech and video over circuit-switched network services
Packetisation	International standard packetisation scheme (e.g. ITU-T H.225) for speech and video over packet-switched network services
Playout delay	< 400 ms end-to-end delay
Delay jitter	The system should be operational under transmission delay jitter of up to 200 ms.
Error rate	End-to-end BER for circuit switched: 10^{-3} FER for packet switched: 10^{-2}
Picture size	CIF (352x288 pixels) and QCIF (176x144 pixels)

3.1.5 Products

A product that could be used for video conferencing in UMTS is Microsoft's NetMeeting[1]. This video conferencing tool is today targeted for conferencing over Internet, however the low system requirements will allow, with the right adjustment, utilisation in UMTS. NetMeeting should then meet the minimum requirements for QoS, suggested by 3GPP2. Beneath a picture of an Ericsson terminal, running Microsoft's NetMeeting.

Figure 4 Microsoft's NetMeeting as a conferencing application

3.2 Video Telephony

3.2.1 General

The 1st and 2nd generation mobile systems were only suitable for speech and small data transfers due to the limited bandwidth. When the third generation system is going to be released, consumers will be able to use video telephony that allows the users to see each other while talking (implied that the mobile terminals are equipped with a camera). The video telephony service is defined by the ITU-T to be an audio-visual conversational

[1] See also section 3.2.5. Packet Video can also be used in video conferencing.

service that provides a two-way real-time transmission of video and audio between two different locations (Person-to-Person) (ITU-T F.720, 1992)[2].

Implementing video telephony into the 3rd generation mobile terminals may help speech- and hearing-impaired persons. This group of impaired persons has had little use of 1st and 2nd generation mobile terminals. With an 3G-enabled terminal, they can use the camera on the device to transfer sign language. It may require some additional requirements, such as support for increased resolution, because it will be little use for speech- and hearing-impaired persons to use a UMTS terminal of QCIF resolution.

3.2.2 Video telephony services defined

The video telephony service is similar to the conferencing service. The difference is that in a conferencing service you are allowed to have multi-point connections. The telephony service could involve several possible applications. In the ITU-T recommendation F.720 it is listed some of them:

- Face to face dialogue, which involves at least head and shoulder images
- Dialogue that includes interactive viewing of documents (sketches, diagrams or charts and objects that can be shown on the screen)
- Remote video surveillance

The architecture of the service is the same as with the conferencing service (section 3.3 and figure 3).

[2] Speech supported by still picture transmission is not regarded as a part of video telephony.

3.2.3 Quality of Service

The QoS requirements for video telephony can be derived from the requirements set for the conferencing service (see table 3 for details). Nevertheless, for video telephony the requirement of having a multiple conversational setup does not apply. As for a typical video conferencing session, the most important object in a scene is the face, thus a video standard which obtain advantage of object-based encoding (e.g. MPEG-4), is preferably used.

3.2.4 Products

A great number of companies are doing research and develops 3G terminals. Beneath it is shown two 3G terminals from Ericsson.

Figure 5 UMTS terminals from Ericsson

3.3 Video Streaming

3.3.1 General

It is obvious that with the 3rd generation mobile system will make it possible for multimedia applications. Many try to predict the service (or application) that is going to be the most profitable one (thus most wanted by the consumers). One of the services that several believe to be the one is the video streaming service[3]. Video streaming is a service that contains continuos video and audio data delivered to an end user (3GPP2 S.R0021, 2000). The video streaming service enables the user to view video content without downloading the whole file before the playback starts, which is a huge advantage compared with video services where the complete video file needs to be downloaded before the user can start viewing the content.

Using this service enables a user to view the 9 o'clock news in Norway from a hotel room in Auckland, New Zealand or maybe hook up to a family web cam to view the birthday of your wife that you couldn't make because you had to work overtime.

3.3.2 Video streaming services defined

The video streaming service is not only a "one way video" conferencing service. Video streaming allows higher delays (5-10 sec), which allows additional buffering. This makes it possible to use error correction mechanisms (or retransmission) which reduces for example bit errors and packet loss and consequently improves the quality of the transfer in comparison with to the video conferencing service.

3.3.3 The technology

The streaming service operates more or less in the same way as the video conferencing service. One of the differences are that the streaming service can be considered as an asymmetrical service because most of the information flows in one direction; from the

[3] NTT DoCoMo regards streaming more important than video conferencing (Hartung, 2000).

server where the information is stored, to one or more clients. The requests from the client can be e.g. a request of retransmission due to an error in transmission.

As with the video conferencing service, no standard for the video codec have been determined at the moment, but the recommendation is that the codec should be standard compliant (3GPP2 S.R0021, 2000). On the market today various standards are suitable for use in the streaming service. The standards that are very efficient (good compression ratio) are also feebler to transmission errors compared with an algorithm that is not so efficient. 3GPP2 suggest that the streaming applications have options that the user can choose from depending on the quality the user desires.

Figure 6 shows the principal of the video streaming service. The video (and audio) information is stored on a server in which the consumer can access and request for a video clip (e.g. a pre recorded news update).

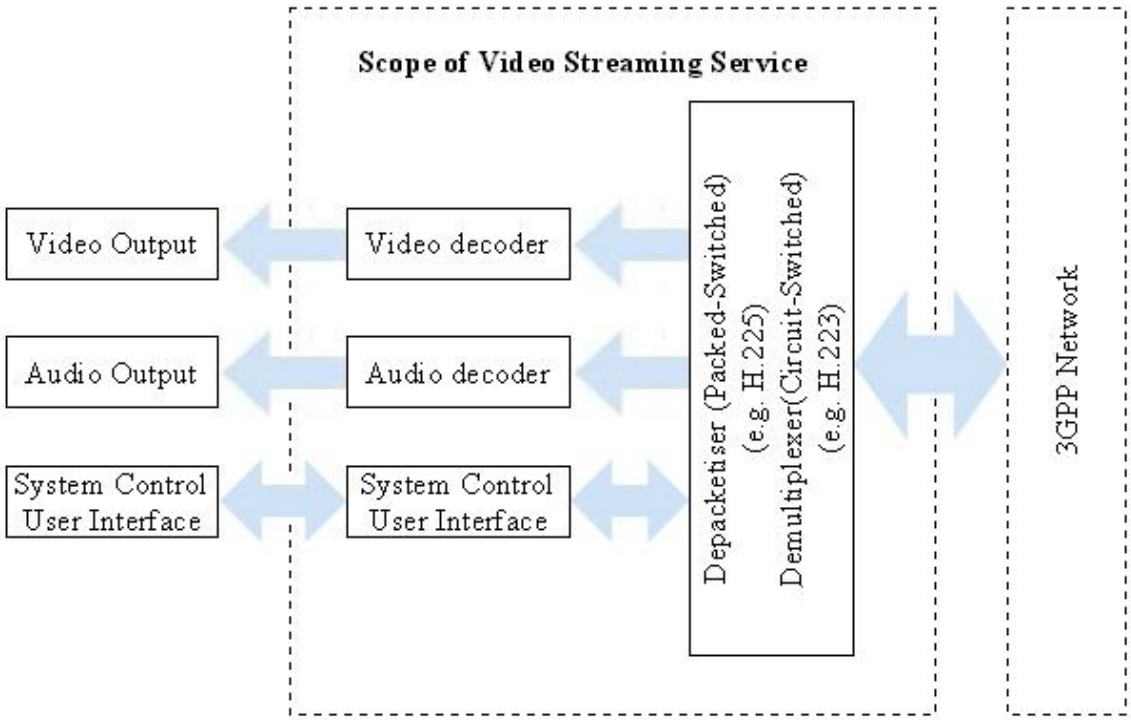

Figure 6 Scope of Video Streaming Service

The bit-rate in wireless system tends to fluctuate a great deal compared with a fixed network that offers more or less the same bandwidth during a transmission, and the bandwidth offered in fixed networks is much higher. To deal with the varying bandwidth that the wireless networks offers, the use of video codecs that supports scalable video compression is wanted (Horn, Scheider, Hartung and Niebert, 2000). H.263 and MPEG-4 video coding algorithms supports scalable video compression[4], which means that the video codec supports compression at different bit-rates. In short, the encoder adds a tag to each encoded frame containing an indication of the bit-rate to the data frame that enables a streaming proxy to change the bit-rate to a wanted transmission rate. Figure 7 illustrates the wireless-streaming platform developed by Ericsson.

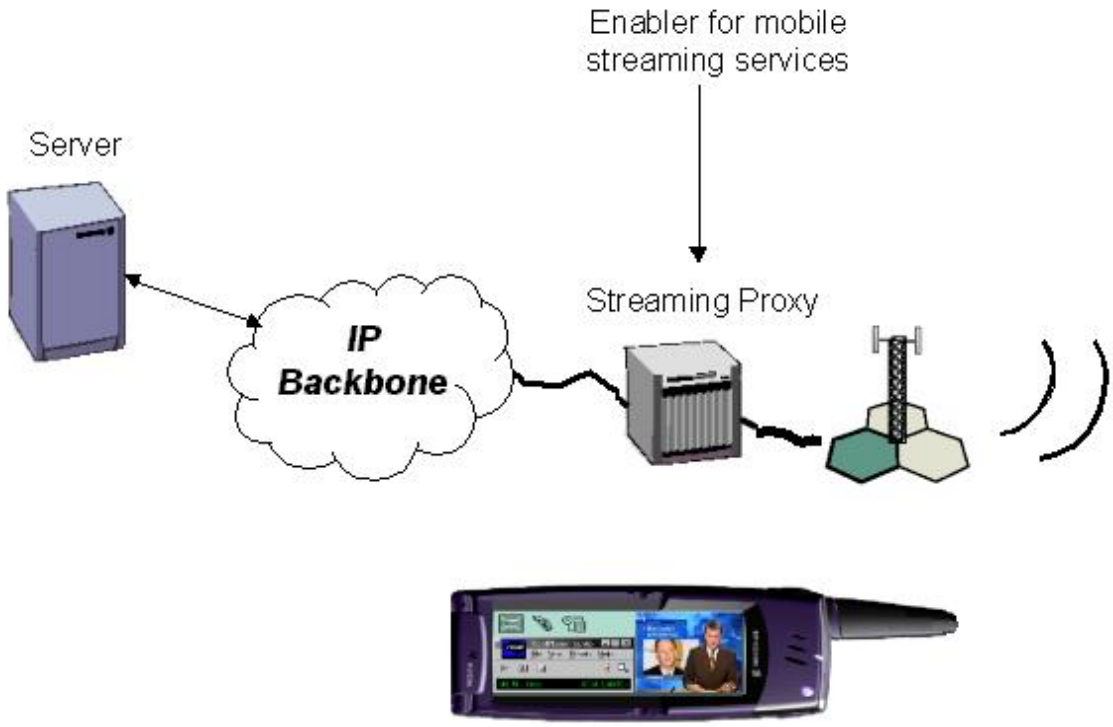

Figure 7 *Ericsson wireless streaming platform*

[4] Ericsson wireless streaming platform uses a scalable H.263+ video codec and a streaming proxy for scalable transmission.

3.3.4 Quality of Service

The video streaming service has relatively high Quality of Service (QoS) requirements, however, since video streaming service is not a bi-directional conversational service the requirements is not that rigid as it is with video conferencing services. In compliance with 3GPP2 shall the video streaming service support one-way transmission of video/audio streams in wireless packet-switched and circuit-switched transmission networks.

In table 4 the QoS features for the video conferencing service is listed (3GPP2 S.R0021, 2000).

Table 4 QoS features for the video streaming service.

Feature	Description
System	Shall be compatible to international video streaming formats (e.g. ASF, SMIL)
Synchronisation	The service should maintain synchronisation between the video and audio components of the stream. Inter-media skew < 20 ms
Video codec	The video codec should be standard compliant (e.g. MPEG-4 and H.263). Optional addition of coding efficiency and error robustness should be available
Speech codec	The audio codec should be standard compliant. Can operate at 8, 64 and 128 kbit/s
Minimum bandwidth	>28 kbit/s for video and audio. > 9.6 kbit/s for streams with audio only
Playout delay	< 30 seconds
Delay jitter	Three times the RLP retransmission time in the network with retransmission activated
Error rate	End-to-end BER for circuit switched: 10^{-3} FER for packet switched: 10^{-2}
Picture size	CIF (352x288 pixels) and QCIF (176x144 pixels)

3.3.5 Products

A product that could be used in UMTS for video streaming services is PacketVideo Player (PVPlayer™). PVPlayer is delivered by Packet Video and is a wireless stand-alone streaming application for video enabled mobile devices with limited procession power. The technology enables one-way video (streaming) and two-way video (conferencing).

Figure 8 Mobile device with PVPlayer software (Packet Video, 2000)

PVPlayer is based on the video compression standard MPEG-4 and supports delivery of data ranging from 9.6 kbit/s to 384 kbit/s (scalability). The PVPlayer supports delivery of data over constant bit-rate channels (circuit switched), and variable rate channels (packet switched). Table 5 taken from the Packet Video Technology Overview (Packet Video, 2000) shows the various levels of MPEG-4 and corresponding parameters.

Table 5 Requirements of MPEG-4 simple profile

Compliance Level	Max Frame Size	Max Frame Rate	Max Bit Rate	Max Video Packet Length	Max Buffer Size
1	QCIF	15 fps	64 kbit/s	256 bytes	10 kbyte
2	CIF	15 fps	128 kbit/s	512 bytes	40 kbyte
3	CIF	30 fps	384 kbit/s	1024 bytes	40 kbyte

4 Video compression algorithms

The need for mobile communications is ever increasing due to the sense of timeliness and flexibility it offers. The increasing diversity of mobile services is bandwidth-demanding communications, not only in the form of speech and data, but also with synthetic and natural images and video and is referred to as mobile multimedia. Nevertheless, multimedia is expensive in the sense of its bandwidth requirement, with video being highly bandwidth intensive (Puri, Eleftheriadis, 1997).

Multimedia services are in widespread use on the Internet today and a large variety of video compression techniques is in use to lower the video's requirement to the bandwidth. Popular video streaming formats used on low-bandwidth Internet's video services at present time is QuickTime™, RealVideo and Windows Media. Though, these video-streaming solutions are widely used, the quality and the error resilience they offer are rather limited.

The focus of this thesis is to study particular contemplations for "real time video transmission in UMTS". As an ephemeral orientation, ISO MPEG-1 video standard was developed to encode videos of VHS quality onto video compact discs. The second standard MPEG-2 focused on better quality video without the constraint of bandwidth and is today used for both DVD-Video discs and digital television. Furthermore, transmission of video using the MPEG-1 standard assumes a relatively error free channel and MPEG-2 standard considers very basic error-resilience techniques. Besides these two ISO standards, ITU-T has also developed video coding standards. The first standard H.261 was developed to support for use of video telephony over ISDN.

In this chapter, an overview of these three standards will be performed due to its widespread use in today's video applications. However, two other standards, MPEG-4 and H.263 (+) have been developed to support video transmission in error prone networks

and these will be extensively explored due to its importance for video transmission in UMTS.

4.1 Background

The need for video compression is significant in wireless telecommunication. The bandwidth of an original video sequence is too large to fit the bandwidth available in UMTS. Take into consideration the video streaming service, which can be used to view a movie, would uncompressed full-screen/full-motion video require a data rate of 216 Mbit/s or more. The maximum bandwidth UMTS offer in pico cells is on 2 Mbit/s. However, use of mobile UMTS terminals in pico cells implies that the terminal is not moving (<10km/h). In environments where the terminal is allowed to move in speeds beyond 10 km/h (micro-, macro- and global-cells) the available bandwidth is ranging from 144 kbit/s to 384 kbit/s. Thus the need of compression of the video information is present.

Compressing the video information will have an effect on the quality of the motion video. The quality of a video sequence is represented by three main factors (Adobe Systems Inc, 2000):

- **Frame Rate** is the number of pictures being shown per second to give the viewer the impression of motion. The higher frame rate, the smoother playback. The frame rate for full motion movies shown in theatres is 24 frames per second (fps).

- The second factor is the **Colour Depth**, which represent the number of bits per pixel for denoting the colour information. If representation of the three Red, Green and Blue (RGB) were on each 8 bits, would the total bits of representing the colour information be 24 bits. 24 bits can denote 16.7 million (2^{24}) colours.

- The last factor that has a major impact on the quality of the motion video is the **Resolution** of the frame. The number of individual picture elements (pixels), horizontal and vertical (e.g. 680x480) usually represents the resolution.

Reducing the frame rate to a full motion movie by half (12fps), will also reduce the size of the video information by half as well. However, a reduction in the frame rate will also influence the smoothness of the playback. A frame rate below 10fps will cause jerkiness in the playback. Furthermore, the eye is more sensitive to the luminance (Y-component) of an image than the changes in the colour (U (Cb) and V (Cr)). This means that one can reduce the number of bits in representing the colour depth without the human eye noticing it. This will additionally reduce the size of the video information. Lastly, diminish the resolution of each frame will compress the video further. For example, scaling down a full screen PC display (640x480) to a resolution of 160x120 will reduce the size of the video information to one-eight of its original size. Beneath it is illustrated the data rate by various frame rates, resolutions and colour representations.

Table 6 Different compression ratios

Compression	Frame Rate	Resolution	Colour Depth	**Data Rate**
None	30 fps	640x480 pixels	24 bit	**216 Mbit/s**
1/12	15 fps	320x240 pixels	16 bit	**18 Mbit/s**
1/48	15 fps	160x120 pixels	8 bit	**4.5Mbit/s**

It is easy to conclude that a need for further compression must be in place to deal with the low rates if you compare the data rates required for the different compression ratios and the bit-rates that is offered in UMTS (> 32 kbit/s for video conferencing in UMTS).

Further compression of the video information is achieved by use of a video codec. Codec is an acronym and symbolises coding/decoding (compression/decompression). A codec is a software component, which translates video between its uncompressed form and the compressed form in which it is stored by use of several algorithms (CodecCentral, 2000).

These video codecs takes the advantage of the fact that the information of each frame in a video sequence varies little compared with the frames around it. Thus, instead of storing the entire frame, it stores only the changes between the current frame and the previous one. This compression method is called “inter-frame” compression (temporal compression) and is also used together with “intra-frame” compressing (spatial

compression) techniques such as reduction in the number of bits used to represent the colours in a frame.

In the following sections of the chapter it is explained in more detail the technology behind various video codecs with emphasis on the codecs that are of importance for use in UMTS.

4.2 H.261

The H.261 video codec for audiovisual services at p x 64 kbit/s (p ranges from 0-30) was created by the ITU in 1990 for global video telephone and video conferencing applications over ISDN and is specified by ITU-T H.320. The characteristics for the H.261 video compression codec are viewed in table 7 beneath (ITU-T H.261 1993).

Table 7 Characteristics for the H.261 video compression codec

Resolution	Frame Rate	Bit Rate
QCIF – 176x144 CIF – 352x288	7.5, 10, 15, 30 fps	40 kbit/s – 2 Mbit/s

H.261 make use of both intraframe and interframe encoding. H.261 defines two different picture formats, CIF (Common Intermediate Format) and QCIF (Quarter CIF). In which the QCIF operation is compulsory, while CIF operation is optional. QCIF is usually used for low bit-rates, such as p<3 (192 Kbit/s). The video frames are composed of three colour components, luminance and two colour differences. Where the two colour difference components contain half the amount of information compared with the luminance component.

The intraframe-encoding scheme used in H.261 operates more or less in the same manner as with JPEG (Joint Picture Expert Group). When interframe encoding is used, a prediction for limited areas in the current video frame is made based on the previous video frame. Further, quantising steps are used to determine the amount of information

that is sent. The standard is not further developed within the ITU due to the fact that H.261 can be seen as a subset of H.263 v.1 and v.2 (Örbrink, 1999).

4.3 H.263 version 1

The H.263 is ITU-T's recommendation for low bit-rate communication. This codec is an extension of the ITU-T recommendation H.261 and targets video encoding below 64 kbit/s. It provides better picture quality at low bit-rates (Côte, Erol, Gallant and Kossentini, 1998) and better error recovery performance (Cherriman, 1999). The H.263 version 2 (H.263+) is an enhancement of H.263 that provides better performance over packet-switched networks and permits scalable bit streams (Alnuweiri, Kossentini and Erol, 2000).

4.3.1 Video frame structure

While H.261 only supports two standardised picture formats (CIF and QCIF), the H.263/H.263+ adopts three additional standardised picture formats (sub-QCIF, 4CIF and 16CIF). Further, it is also possible to negotiate a custom picture format (ITU-T H.263, 1998).

Each picture are coded as luminance and two colour difference components (Y, Cb and Cr). The luminance (Y) represents brightness information in picture and the chrominance (Cb and Cr) represents colour difference components. For the different picture formats the luminance samplings structure is *dx* per line, *dy* lines per picture in an orthogonal arrangement. Sampling of each the two colour difference components is at dx/2 pixels per line, dy/2 lines per picture, orthogonal (ITU-T H.263, 1998).

In table 8 the various standardised H.263 picture-formats are listed.

Table 8 *Number of pixels per line and number of lines for each of the picture formats*

Picture format	No. of pixels for luminance (dx)	No. of lines for luminance (dy)	No. of pixels for chrominance (dx/2)	No. of lines for chrominance (dy/2)
Sub-QCIF	128	96	64	48
QCIF	176	144	88	72
CIF	352	288	176	144
4CIF	704	576	352	288
16CIF	1408	1152	704	576

Each picture in the video sequence is divided into groups of blocks (GOB). For a picture of QCIF resolution, the number of GOB is 9. Each GOB is further divided into macroblocks. Each macroblock consists of four luminance blocks of 8 pixels x 8 lines and the spatially corresponding 8 pixels by 8 lines of Cb and Cr (ITU-T H.263, 1998). In figure 9, an illustration of the picture, structure at QCIF resolution is given (Côte et. al., 1998).

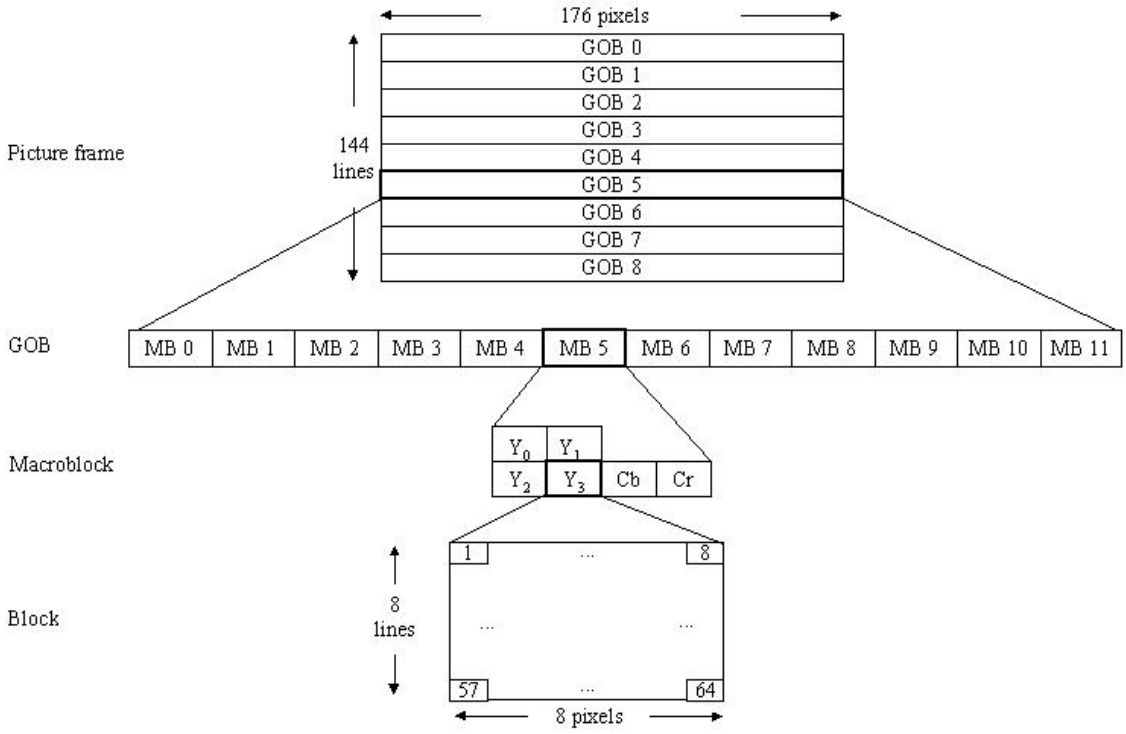

Figure 9 *H.263 picture structure at QCIF resolution*

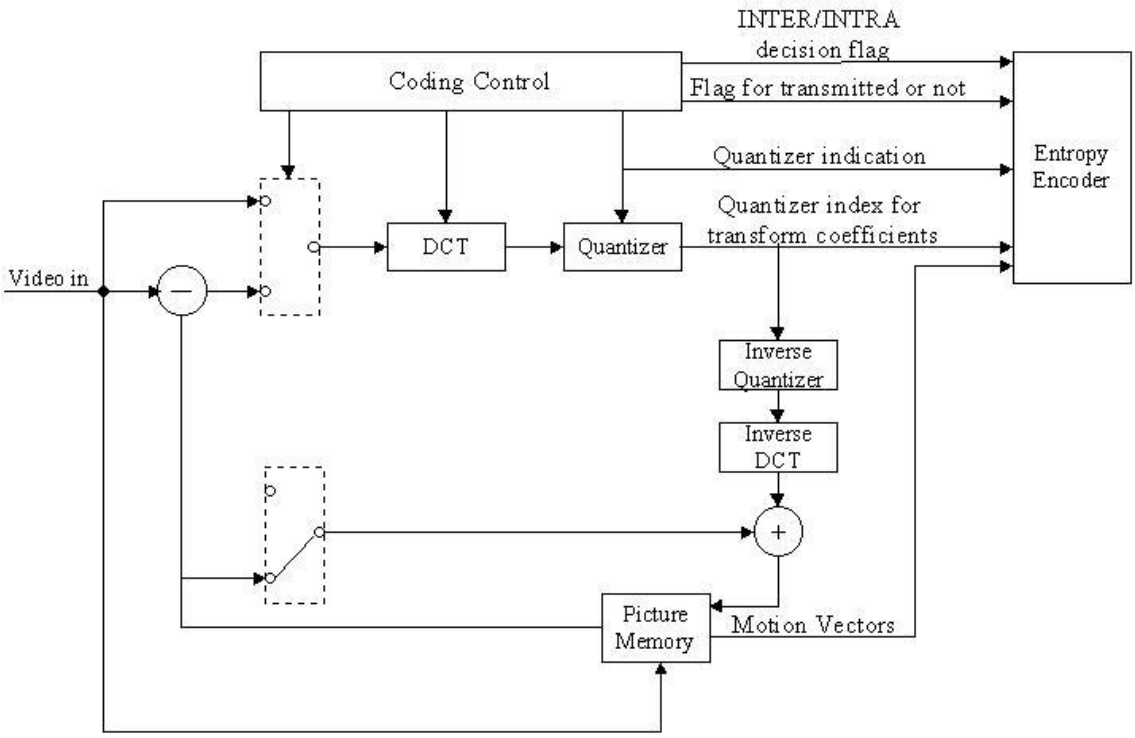

Figure 10 H.263 encoder

4.3.2 Motion estimation and compensation

H.263 supports inter-picture prediction, which is based on motion estimation and compensation. The coding mode in which temporal prediction is applied is called INTER-mode (P-pictures) – where only the difference between original pictures and motion-compensated predicted pictures needs to be encoded. Where no temporal prediction is applied (no reference to any other picture), the coding mode is called INTRA-mode (I-pictures) coding. In the case of the PB-mode is B-pictures always-coded in INTER-mode. B-pictures are bi-directionally predicted, i.e. predicted both from the previous decoded P-picture and the P-picture currently being decoded (ITU-T H.263, 1998). Figure 11, illustrates how the various pictures are predicted.

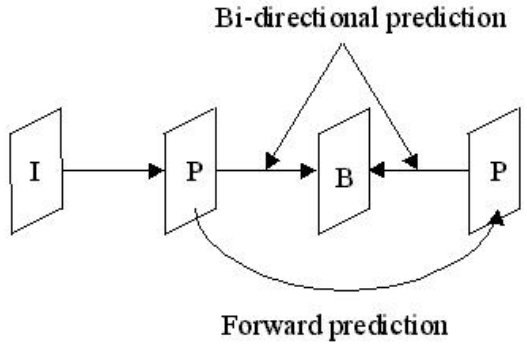

Figure 11 Picture prediction

In short, areas in the picture, which do not change (e.g. background), will not be encoded. Further reduction of the size of the picture information is achieved by attempting to estimate where areas of the previous picture have moved to in the current picture and compensate for this movement. Two-dimensional motion vectors (MV) is used to denote this motion information. A motion estimation (ME) module compares each macroblock in the current picture with its surrounding area in the previous picture to find matching areas and moves these areas into the current macroblock by use of the motion compensator module. Further, the motion compensated macroblock is subtracted from the candidate macroblock. The most widely used method to resolving the best matching blocks in motion estimating is the Sum of Absolute Difference (SAD) due to this method demands low computational complexity and performance (Côte et. al., 1998). The SAD operation is defined (Alnuweiri et. al., 2000):

$$SAD = \frac{1}{M \cdot N} \sum_{m=1}^{M} \sum_{n=1}^{N} \left| X_{m,n} - X_{m+i,n+j} \right| \quad \text{, where}$$

N, M – dimensions of the block

$X_{m, n}$ – value of the block element located in row m and column n

$X_{m+i, n+j}$ – value of the block located in row m+i and column n+j in reference to the preceding picture

4.3.3 Discrete Cosine Transform (DCT)

The Discrete Cosine Transform (DCT) is the most computationally intensive part of the H.263 baseline encoding (Alnuweiri et. al., 2000). DCT have the purpose to decorrelate the 8 x 8 blocks of original pixels and to compact their energy into a set of spatial frequency components. Though the 8 x 8 DCT is computationally intensive it is rather simple and efficient. The 8 x 8 DCT is defined by (ISO/IEC 13818, 2000):

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

$$C(u) = C(v) = \sqrt{\frac{1}{8}}, \text{ for } u, v = 0$$

$$C(u) = C(v) = \sqrt{\frac{1}{4}}, \text{ otherwise}$$

u, v, x, y = 0, 1, 2 …7

x, y – are spatial coordinates in the sample domain

u, v – are coordinates in the transform domain

and F(u, v) denotes the coefficients of the 8 x 8 DCT transformed block and f(x, y) denotes x, y^{th} pixel of the 8 x 8 original block.

4.3.4 Quantisation

The human eye is more sensitive to reconstruction errors that are associated with low spatial frequencies than the associated high spatial frequencies (Jayant, Johnston and Safranek, 1993). The changes that are important for the eye are the gradual linear changes in the intensity or colour (low frequencies). The frequencies that change rapidly is not so important due to the fact that they may not be noticed, thus these frequencies may be thrust aside. Dividing every coefficient in the DCT, output matrix by an integer scale factor and truncating the result does this. The quantisation value is defined by (Côte et. al., 1998):

$C^{q}_{m,n} = \frac{C_{m,n}}{Q_{m,n}}$, where

$C_{m,n}$ – is the (m, n)th DCT coefficient and,
$Q_{m,n}$ – is the (m, n)th quantisation integer scale factor

In more layman terms, for a normal block of pixels, most of the coefficients produced by the DCT are close to zero. The quantisation reduces the accuracy of every coefficient so that the coefficients that are near zero are set to zero and only a few significant non-zero coefficients are left.

In H.263, quantisation is achieved using the same step size within a macro block. The quantising step estimates the amount of information that is sent. If the step size value is increased it results in more information sent and better video quality is achieved. Moreover, of course, if the quantising step size decreases, less information about the video frame is sent resulting in a poorer quality.

4.3.5 Entropy encoder

The entropy encoder is a type of encoder that uses a code table to replace frequently occurring values by short length codes and less frequent occurring values by longer codes (VLC's). This leads to that the quantised DC coefficients are compressed and resulting in a sequence of VLC's. Further, will these codes be combined with synchronisation and control information (e.g. the motion vectors) to form the H.263 bitstream.

4.4 H.263 version 2

The H.263 version 1 commenced a better compression efficiency and improved picture quality with a small amount of additional complexity, although the coding structure was based on H.261. The development of H.263 version 2 (H.263+) introduced a number of optional feature enhancements that are supplemented to the core syntax and semantics of H.263. These enhancements were developed to make H.263 more resilient to errors in error prone networks (Alnuweiri et. al., 2000).

To accomplish a more error resilient video coding method, H.263+ introduce the concept of scalable video coding. Scalable video coding makes it possible to separate the video bit stream into multiple logical channels, with the intention that some data can be discarded without harming the video representation. Leaving out some video information can be benefited from if the bandwidth available in a network decreases considerably. As mentioned earlier this way of transmitting video information is taken advantage of in video streaming servers[5].

The types of picture scalability H.263+ employ are temporal-, spatial- and SNR scalability.

4.4.1 Temporal scalability

Temporal scalability supplies the means for enhancing perceptual quality by increasing the video picture rate. Temporal scalability is achieved using bidirectionally-predicted pictures (B-pictures). The B-pictures is inserted between anchor picture pairs (between I-pictures and P-pictures) as illustrated in figure 12. Accordingly, for equal quantising level, this property generally results in improved compression efficiency as compared to that of P-pictures (ITU-T H.263, 1998), which is only forward predicted.

[5] See section 3.3 for more information

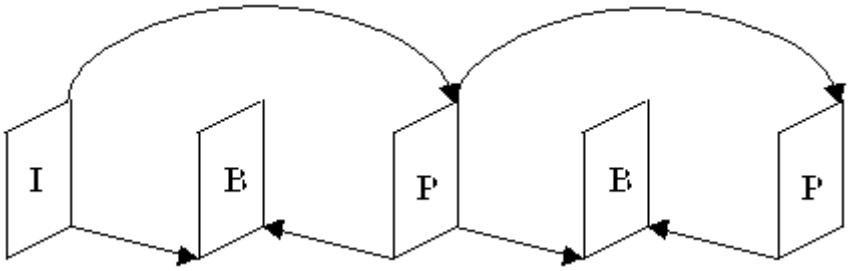

Figure 12 Temporal scalability

B-pictures are not used as reference pictures for the prediction of any other pictures. This property allows for B-pictures to be discarded if necessary, without impacting the visual picture quality of the future pictures.

Though use of temporal scalability may increase the picture quality, it adds structural complexity, larger memory is required and additional delay is introduced.

4.4.2 SNR Scalability

The other basic method to achieve scalability is through spatial/SNR enhancement. Spatial and SNR scalability are closely related. The distinction is that spatial scalability provides increased spatial resolution. The next figure (figure 13) depicts an example of SNR scalability. SNR scalability involves the creation of multiple bit streams. This allows recovery of coding errors, or the difference, between an original picture and a reconstructed one. This is achieved through use of a finer quantiser for encoding in the enhancement layer to encode the difference. The extra data serves to increase the signal-to-noise ratio of the video picture, and hence, the term SNR scalability (ITU-T H.263, 1998).

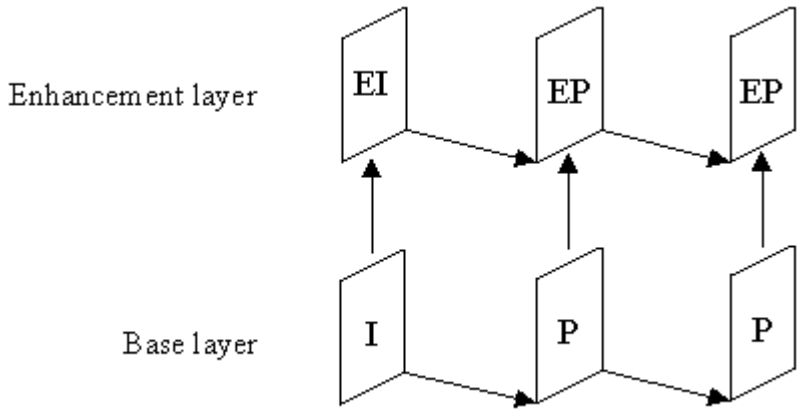

Figure 13 SNR scalability

The vertical arrows from the base layer illustrate that the picture in the enhancement layer is predicted from a reconstructed approximation of that picture in the reference (base) layer. If prediction is only formed from the base layer, then the enhancement layer picture is referred to as an EI-picture. It is possible, however, to create a modified bi-directionally-predicted picture using both a prior enhancement layer picture and a temporally simultaneous lower layer reference picture. This type of picture is referred to as an EP-picture or "Enhancement" P-picture (ITU-T H.263, 1998).

4.4.3 Spatial scalability

Use of spatial scalability makes it possible to encode bit streams of more than one resolution to meet varying display requirements/constraints for a wide range of users. Considering a videoconference were the participants use display tools of different resolution capability (e.g. a UMTS terminal of QCIF resolution and a computer screen with CIF resolution capability), a streaming server can send video of different resolution to the participants. The next figure (figure 14) illustrates the concept behind spatial scalability.

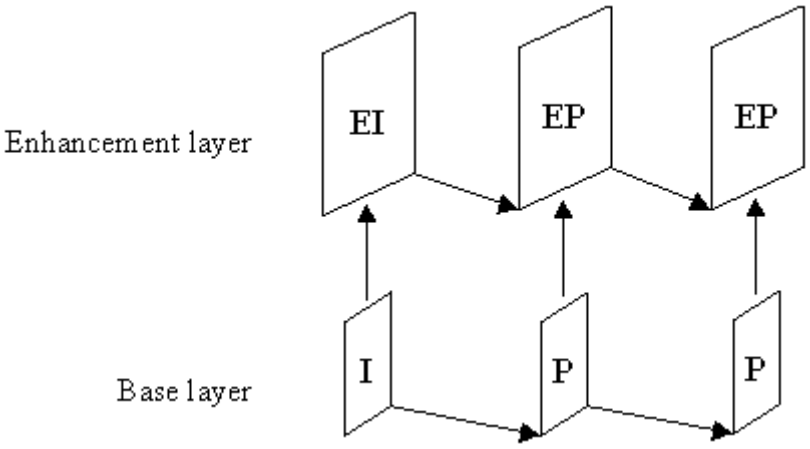

Figure 14 Spatial scalability

As mentioned earlier is Spatial and SNR scalability essentially the same. The difference is that here a spatial enhancement layer tries to recover the coding loss between an upsampled version of the reconstructed base layer picture and a higher resolution version of the original picture (Côte et. al., 1998). For example, the base layer can be of QCIF resolution and the enhancement layer may be of CIF resolution. The base layer must be scaled in a way that the picture in the enhancement layer can correctly be predicted from it. In H.263+ the resolution can be increased by the factor of two both directions (horizontally and vertically) separately or concurrently.

Since there is very little syntactical distinction between pictures using SNR scalability and pictures using spatial scalability, the pictures used for either purpose are called EI- and EP-pictures (ITU-T H.263, 1998).

4.4.4 Other enhancements

Though the introduction of scalable video coding was the key enhancement in H.263 (+), other enhancements were also introduced. The enhancements, which have an influence on the error resilience of the codec, is listed and explained below (ITU-T H.263, 1998).

- Slice Structured (SS): The purposes of this mode are to provide enhanced error resilience capability, to make the bitstream more amenable for use with an underlying packet transport delivery, and to minimise video delay

- Reference Picture Selection (RPS): Improves the performance of real-time video communication over error-prone channel by allowing temporal prediction from pictures other than the most recently-sent reference picture. In error-prone channel environments, this mode allows the encoder to optimise its video encoding for the conditions of the channel.

- Independent Segment Decoding (ISD): Allows a picture to be constructed without any data dependencies which cross GOB or multi-GOB video picture segments or slice boundaries. This mode provides error robustness by preventing the propagation of erroneous data across the boundaries of the video picture segment areas.

4.4.5 Error resilience and robustness in error prone environments

In environments where the available bandwidth is limited, and the transmission suffers from bit errors, the need for post-processing mechanism is present. These post-processing mechanisms are also known as Forward Error Correction (FEC) tools, because they provides the means for error correction at the encoder for use in the decoder. This section contains post-processing mechanisms that are taken advantage of in H.263+.

4.4.5.1 Error detection and resynchronisation

Before any attempts to conceal the errors at the decoder, the errors must be detected. Remember from chapter 3 that the ITU-T H.223 can control errors for various streams. A media packet in H.223 is called an Adaptation Layer Protocol Data Unit (AL-PDU). This AL-PDU consists of an optional control field, media payload (which in this case is video information) and a Cyclic Redundancy Check (CRC) checksum. The H.223 is capable of detecting errors or loss of media packets (through block numbering in the video bit stream) using CRC checksum information and pass this information on to the decoder.

The decoder can then choose whether to drop the complete packet if errors are detected or indicated (Côte, Kossentini and Shirani, 2000). When detection of errors fails in the multiplexing layer, media packets containing errors are passed on to the decoder. The decoder can still detect bit errors using syntactic or semantic violations of the bit stream. These include (Côte, Kossentini and Shirani, 2000):

- Motion vector outside allowable range.
- Invalid VLC table entry.
- DCT coefficient out of range.
- Number of DCT coefficients in a block is exceeded.

Techniques to accomplish resynchronisation due to errors are introduced in H.263+. The use of resynchronisation enables the decoder to decode valid video information after an error has occurred. Usually the location of decoded bits within the picture cannot be determined because when an error occur, the number of missing symbols is not known. To get rid of this dilemma, unique resynchronisation code words in regular intervals are inserted (see figure 15). In H.263 (+) these code words are placed in the GOB header to provide an absolute location in the picture. This introduces concerns due to the fact that the size of the GOB's is varying [6]. This results in that the resynchronisation markers within a bit stream is of changing gaps.

[6] MPEG-4 has solved this problem by inserting resynchronisation markers every N bits (see section 4.6.2.5). Experiments shows that this method is a more efficient way of recovering from transmission errors.

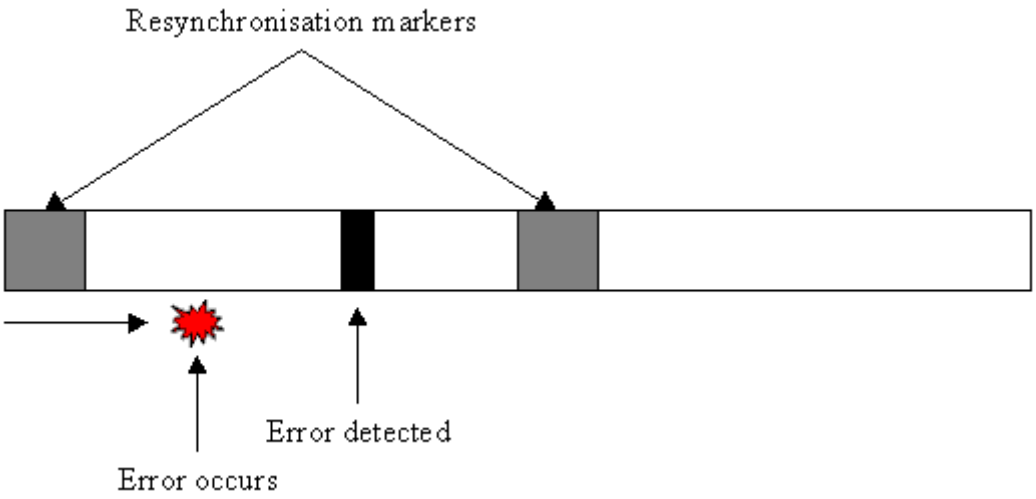

Figure 15 Resynchronisation markers in and H.263+ bit stream

4.4.5.2 Error concealment

After detecting the error, error concealment is performed. Error concealment is used to hide visual distortion because of residual errors. There exists two basic methods to perform error concealment.

- Spatial error concealment
- Motion compensated temporal error concealment

One or both of these approaches can be used depending on the condition. When spatial error concealment is used, the missing pixel values are reconstructed using neighbouring GOBs spatial information. In temporal error concealment the lost data is reconstructed from the same GOB in the previous frame (Côte, Kossentini and Shirani, 2000).

4.5 MPEG-1 and MPEG-2

4.5.1 MPEG-1

The MPEG (Motion Pictures Expert Group) committee was formed in 1988 by the hands of Leonardo Chairigloione and Hiroshi Yasuda with the objective of standardising video and audio for compact discs.

In 1992, a meeting with ISO (International Standards Organisation) and IEC (International Electrotechnical Commission) resulted in the first MPEG standard for audio and video coding (MPEG-1). This standard targeted to the storage and retrieval of video and audio on compact disc (Chiariglione, 1998), however it also could be used to delivery of video over telecommunications networks (fixed networks) (Sikora, 1997a). The MPEG-1 standard is a generic standard, i.e. a standard that is independent of a particular application and of the delivery media; however, it could not ignore the requirements of the applications (Martins and Teixeira, 1996). MPEG-1 is progressively scanned video (non-interlaced). And the performance is optimised for SIF (Source Input Format) resolution images at 352 x 288 (in Europe at 50 Hz) at 30 fps at a bit-rate ranging from 1.2 Mbit/s to 1.5 Mbit/s (Timsari and Shahabi, year unknown).

The MPEG-1 algorithm was developed with respect to the JPEG and H.261 activities. It was developed to maintain a similarity to the H.261 so that implementations supporting both standards were possible (Sikora, 1997a). However, MPEG-1 was primarily targeted for storage and retrieval of video and audio on compact disc.

4.5.2 MPEG-2

MPEG-2 became a standard in 1994 with the object to support a wider collection of applications and can be seen as a superset of MPEG-1 and is backward compatible with MPEG-1 (Lee, 1997). In MPEG-2 new coding characteristics was added and made it possible to improve the functionality and quality at higher bit-rates (up to 40 Mbit/s).

Thus prediction modes were developed to support efficient coding of interlaced video; i.e. two interleaved fields are used to scan out one video frame. In addition introduced MPEG-2 scalable video coding. This allows division of the continuos bitstream into two or more coded bit streams that represents the video at different resolutions, picture quality or picture rates (Martins and Teixeira, 1996).

The coded video information consists of an ordered set of bit streams, called layers. If there is only one layer, the coded video stream is identified as a non-scalable video bit stream. For two layers or more, the coded video information is called a scalable video bit stream (ISO/IEC 13818-2, 2000).

4.5.2.1 Non-scalable syntax

The non-scalable syntax in MPEG-2 is different from the one in MPEG-1, due to the extra compression tools for interlaced video signals. The algorithm is similar to the compression algorithm used in MPEG-1. First of all the algorithm uses block-based motion compensation to reduce the temporal redundancy. Motion compensation is used both for casual prediction of the current picture from a previous picture, and for non-casual, interpolative prediction from past and future pictures. The prediction error is further compressed using Discrete Cosine Transform to remove spatial correlation before it is quantised. Finally, the entropy encoder combines the motion vectors with the quantised DCT information and encoded using variable length codes (ISO/IEC 13818-2, 2000).

4.5.2.2 Scalable syntax

The scalable syntax is designed to accomplish support for applications beyond those supported with the non-scalable syntax (single layer coding). The objective of this coding method is to offer interoperability among various services and provide additional support for receivers with diverse display capabilities (Lee, 1997). Applications such as in the area of video telecommunication, video on Asynchronous Transfer Mode (ATM) networks, internetworking of video standards and video service hierarchies with multiple

spatial, temporal and quality resolutions are some the noteworthy application areas (ISO/IEC 13818-2, 2000).

Some receivers are not capable to reconstruct the full resolution video. With the scalable coding syntax, these coders have the possibility to decode subsets of the bitstream to playback the video at lower or temporal resolution or with lower quality[7].

4.5.3 MPEG-1 and MPEG-2 compared

The major differences between MPEG-1 and MPEG-2 are outlined below, and the typical coding parameters is stated in the following table 9 (Sikora, 1997a).

Differences between MPEG-1 and MPEG-2:

- MPEG-2 – allows progressive sequences and interlaced ones, while MPEG-1 only allows for progressive picture sequence

- MPEG-2 – allows higher bitrates than MPEG-1

- MPEG-2 – allows surround sound, and alternate language channels

- MPEG-2 – has extra spatial scalability information so that different decoders can get different quality outputs

- MPEG-2 – allows scalability so that one stream can be displayed at different frame rates

[7] More information on temporal, SNR and spatial scalability can be found in section 4.4 (H.263 (+)).

Table 9 Typical parameters for the two standards.

	MPEG-1	**MPEG-2**
Standardised	1992	1994
Main Application	Digital video on CD-ROM	Digital TV (and HDTV)
Spatial Resolution	CIF Format (1/4 TV)	TV (4 x TV) 720 x 576 pixels (1440 x 1152 pixels)
Temporal Resolution	25 - 30 frames/s	50-60 fields/s (100-120 fields/s)
Bit Rate	1.5 Mbit/s	4 Mbit/s (20 Mbit/s)
Quality	comparable to VHS	Comparable to NTSC/PAL
Compression Ratio	20 - 30	30-40 (30-40)

4.6 MPEG-4

MPEG-4 (together with H.263) has been selected as a multimedia standard for 3rd generation wireless network by Third Generation Partner Project (3GPP). MPEG-4 is an ISO/IEC standard developed by MPEG in 1998. This standard was made with the purpose to be the next standard in the world of multimedia, which supports very low bit-rates. Dissimilar with MPEG-1 and MPEG-2, which focused on better compression efficiency, MPEG-4 emphasises on new functionalities.

This standard supplies the users with the opportunity to achieve a variety of forms of interactivity with audio and video information from a scene and to merge synthetic and natural audio and video content (Chiariglione, 1998). MPEG-4 is designed to provide high degree of flexibility and extensibility in order to take the advantage of the rapidly evolving technologies such as in wireless communication systems (Martins and Teixeira, 1996). Moreover MPEG-4 video supports a wide range of bitrates, and is targeted to be 5-64 kbit/s for mobile applications and up to 4 Mbit/s for TV applications.

4.6.1 The video functionalities

The standard introduces new video functionalities which have been described in the MPEG-4 Proposal Package Description (MPEG AOE Group, 1995) can be grouped in to three classes (content-based interactivity, coding efficiency and universal access) and are defined below:

4.6.1.1 Content-based interactivity

- Content-based manipulation and bit stream editing
- Hybrid natural and synthetic data coding
- Improved temporal random access

In the two previous MPEG standards (MPEG-1 and MPEG-2), manipulation of the video sequence was only possible in the original domain. This means that the compressed video sequence has to be decoded to the original format before encoded in to desired compressed bitstream. MPEG-4 makes it possible to access and manipulate visual-objects in a compressed domain. This results in a reduction in complexity, reduced bandwidth requirements, low delay, no quality loss and object-based manipulation (Zhang, 1997).

Further, MPEG-4 provides the means for harmonious interaction of video objects of various origins, which can be either, natural or synthetic (ISO/IEC 14496-2, 2000). This means that it is possible to encode a video sequence in a way that allows separate decoding and reconstruction of objects in a scene and allow manipulation of the original scene by simple operations on the bit stream. This is possible because the bit stream is "object layered" and the shape and transparency of each object are described in the stream of each object layer (Sikora, 1997). Therefore, a receiver of a bit stream have the opportunity to choose if the bit stream shall be reconstructed fully or in a manipulated form.

In figure 16 it is illustrated how an MPEG-4 bit stream can be manipulated.

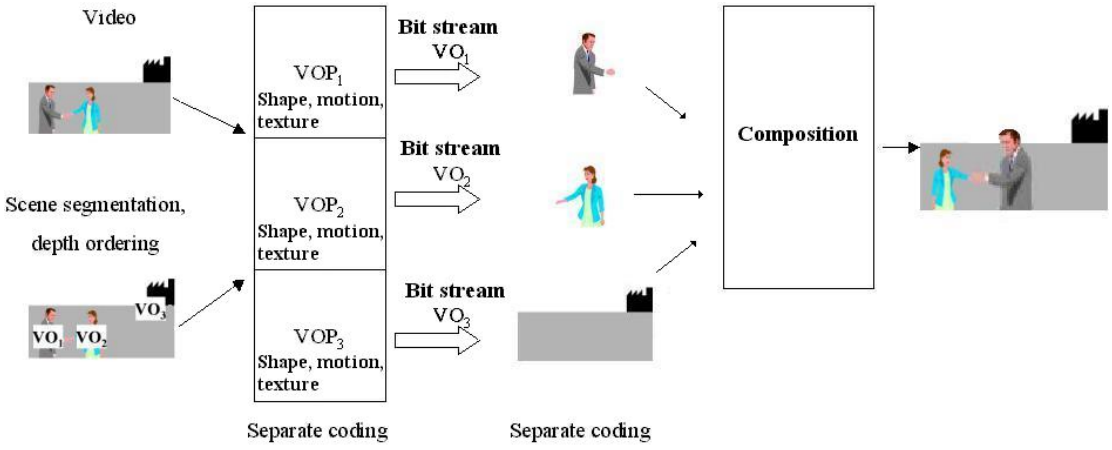

Figure 16 *Manipulated bit stream*

As illustrated in the figure, the video object VO_1 is manipulated and made larger than it was originally. In addition, VO_1 and VO_2 have changed position. Due to the fact that the bit stream of the sequence is organised in an "object layered" form, the manipulation is

achieved on bit stream level without the need of transcoding (i.e. decode the bitstream, manipulate it and encode into compressed bitstream).

Lastly, in the terms of content-based interactivity, MPEG-4 provides efficient methods to randomly access, within a limited time and with fine resolution, parts (e.g. frames or objects) from a video sequence (Zhang, 1997).

4.6.1.2 Coding efficiency

- Improved coding efficiency
- Coding of multiple concurrent data streams

In networks where the available bandwidth is rather limited (wireless networks), it is important that improved coding efficiency is provided. MPEG-4 offers tools and algorithms that offers better compression without affecting the quality of the video. This makes it possible for use of applications such as video-conferencing and –streaming in mobile networks. In, addition MPEG-4 provides methods to code multiple views of a scene efficiently.

4.6.1.3 Universal access

- Robustness in error-prone environments
- Content-based scalability

The problem with many compression algorithms is that when the compression ratio increases the risk for bit error in the transmitted stream increases, especially in mobile systems. Typical for mobile networks is that the transmission channels are subject to noise, fading, shadowing and interference, which present a significant challenge to provide a reliable transmission of the coded video sequence (Zhang, 1997). MPEG-4 provides error robustness capabilities for storage- (e.g. video-on-demand applications)

and communication-applications (e.g. video conferencing applications) in heterogeneous error-prone environments that may suffer from severe error conditions (e.g. long error bursts) (Sikora, 1997).

The key functionality of MPEG-4 is content-based scalability, which provides the ability to attain scalability with fine granularity in content, quality and complexity. In transmission scenarios where the available bandwidth suddenly changes, scalability provides the ability to reduce the quality of the video frames transmitted and consequently reduces the bandwidth requirements to the application (described further in section 4.6.2.4).

4.6.2 Structure of the video coding algorithm

The video coding algorithm will support all the functionalities that is provided in MPEG-1 and MPEG-2. This includes the provision to efficiently compress standard rectangular sized image sequences at varying levels of input formats, frame rates, bit-rates and different levels of spatial-, temporal- and quality scalability.

Figure below illustrates the basic classification of the bit-rates and functionalities that is provided by the MPEG-4 visual standard for natural images and video, with the attempt to cluster bit-rate levels vs. sets of functionalities.

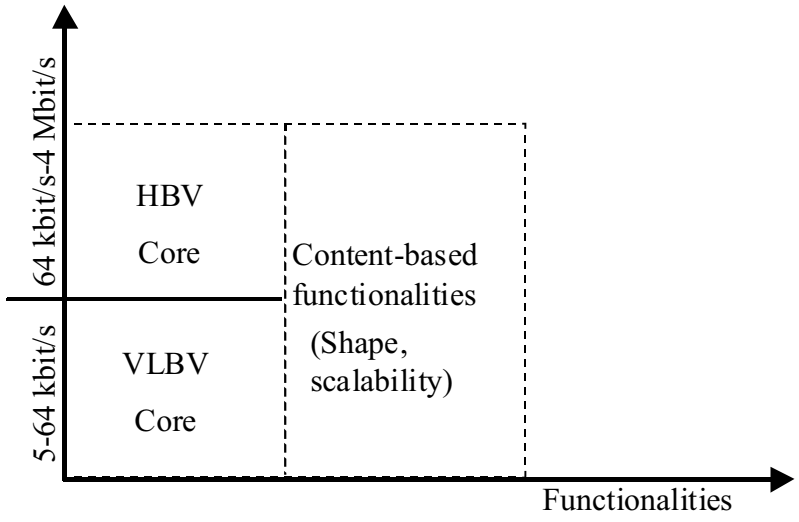

Figure 17 Structure of the MPEG-4 video-coding standard

The VLBV (Very Low Bit rate Video) Core provides algorithms and tools for applications, which works at bit-rates in the range of 5-64 kbit/s. It espouses image sequences with relatively low spatial resolution, such as CIF (176x144), and lower frame rates (< 15 fps). The VLBV Core supports various applications specific functionalities that consist of coding of rectangular-size images with high coding efficiency, high error robustness and low complexity and latency for real-time applications (e.g. real time video transmission in wireless systems).

Furthermore, HBV Core provides the same basic functionalities as for VLBV. However the bit-rates provided to the applications is ranging from 64 kbit/s up to 4 Mbit/s. The applications include broadcast or interactive retrieval of signals with qualities comparable to digital TV (Chiariglione, 1997).

An MPEG-4 Verification Model (VM) was defined in order to evolve video-coding techniques in a combined effort. The MPEG-4 verification model specifies the input and output formats for the uncoded information and the format of the bit stream containing the coded information. Further, it demonstrates a defined core video coding algorithm platform for the standard in progress. The MPEG-4 video verification model supports the main features summarised underneath (Sikora, 1997):

- Standard Y:U:V luminance and chrominance intensity representation of regularity sampled pixels in 4:2:0 format. The intensity of each Y, U or V pixel is quantised into 8 bits. The size and shape of the image depends on the application.

- Coding of multiple “Video Object Planes” (VOP’s) as images of arbitrary shape to support many of the content based functionalities.

- Coding of shape and transparency information of each VOP by coding binary or grey scale alpha plane image sequences using a particularly optimised Modified Reed Code method.

- Support for intra (I) coded VOP's as well as temporally predicted (P) and bi-directionally (B) predicted VOP's. Standard MPEG and H.263 I, P, B frames are supported as special case.

- Support of fixed and variable frame rates of the input VOP sequences of arbitrary or rectangular shape.

- 8x8 pixel block-based and 16x16 pixel Macroblock-based motion estimation and compensation of the pixel values within VOP's, including provisions for block overlapping motion compensation.

- Texture coding in I, P and B VOP's using an 8x8 DCT or alternatively a shape adaptive DCT (SADCT) adopted to regions for arbitrary shape, followed by MPEG-1 and MPEG-2 or H.261 and H.263 like quantisation and run-length coding.

- Efficient prediction of DC and AC coefficients of the DCT in intra coded VOP's.

- Support for efficient static as well as dynamic SPRITE prediction of global motion from a VOP panoramic memory using 8 global motion parameters.

- Temporal and spatial scalability for arbitrary shaped VOP's.

- Adaptive macroblock slices as well as improved bit stuffing and motion maskers for resynchronisation in error prone environments.

- Almost backward compatibility with the standards H.261, H.263 and MPEG-1 coding algorithms if the input image sequences are coded in a single layer using a single rectangular VOP structure.

4.6.2.1 Video Object, Video Object Layer (VOL) and Video Object Plane (VOP)

In contrast to the frame-based video coding algorithms such as MPEG-1, MPEG-2, H.261 and H.263, MPEG-4 is an object-based algorithm. The MPEG-4 video-coding algorithm provides an efficient representation of visual objects of arbitrary shape[8], with the objective to support "content-based" functionalities. The content-based functionalities make it possible to separate the encoding and decoding of physical objects in a scene (i.e. each physical object is encoded/decoded individually[9]). Within the context MPEG-4, the ability to identify and selective decide and reconstruct video content of interest is referred to as "content-based scalability" (Sikora, 1997b).

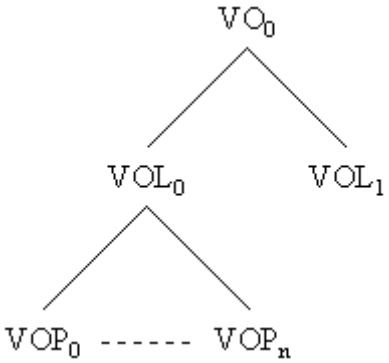

Figure 18 Structure of VO

The representation architecture envisioned in the verification model is based on the concept of Video Object Planes (VOP's). VOP's is a number of arbitrary shaped image regions where each region can cover particular video content, i.e. describing physical objects or content within scenes. Thus the video input content is no longer considered to be a rectangular region as with MPEG-1 and MPEG-2 video standards. The video input can be a VOP image region of arbitrary shape where the shape and the location vary over time.

[8] In addition, MPEG-4 supports conventional rectangular video, similar to the MPEG-1 and MPEG-2 coding.

[9] If scene segmentation is not available or not useful, e.g. in a very simple wireless video communication, the standard defines the entire scene as one physical object (Fitzek and Reisslein, 2000).

VOP's that belongs to, or describes the same physical object in a scene is, commonly known as "Video Objects", VO's. Each VO may have two or more scalability layers, where the lowest level is a base layer and the other layers function as enhancement layers, referred to as Video Object Layers (VOL's). The VOL's contains the shape, motion and texture information of VOP's that belongs to the same VO. Further, the VOL's contains relevant information that is necessary to identify each of the VOL's and how the different VOL's are organised at the receiver to recreate the entire original sequence. This makes it possible to exploit separate decoding of the VOP's and allows content-based manipulation without the need for transcoding.

4.6.2.2 Coding of shape, motion and texture information for each VOP

In order to support separate decoding of VO's the data associated to the shape, motion and texture for the VOP's (belonging to the same VO) are coded into separate VOL-layers (depicted in figure 19). The Video VM exploits an identical algorithm for coding of shape, motion and texture information in each of the VOL's. The coding scheme MPEG-4 provides if the input video sequence only consists of rectangular sized images have the same structure comparable to MPEG-1 and MPEG-2 video coding algorithms. In this case, the shape information for the VOP's is not transmitted to the decoder.

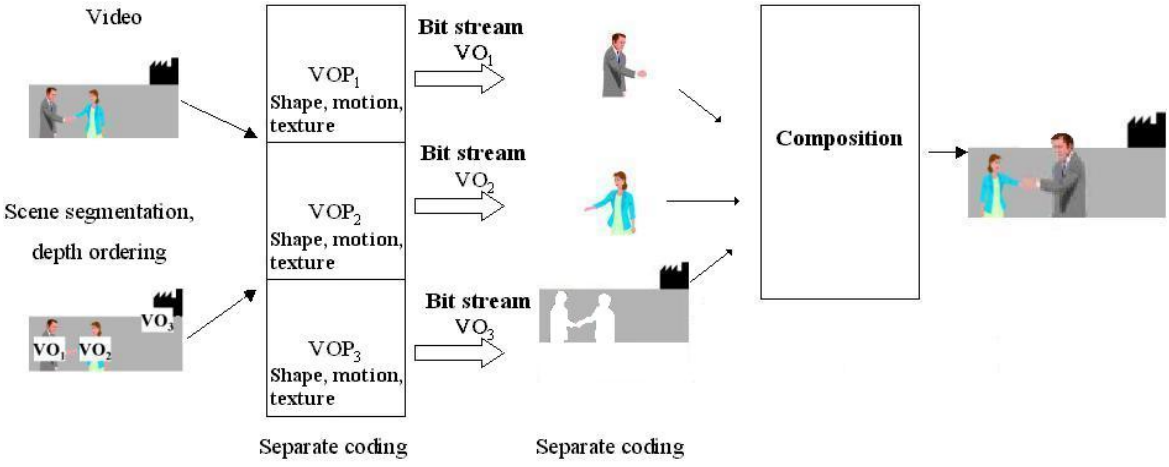

Figure 19 Coding of video sequence using MPEG-4

Figure 20 illustrates the basic approach of the MPEG-4 video algorithms to encode input video sequences (Koenen, 2000). The MPEG-4 compression algorithm used to encode rectangular as well as arbitrary shaped input VOP image sequences is based on the coding technique that is used in the other MPEG coding standards.

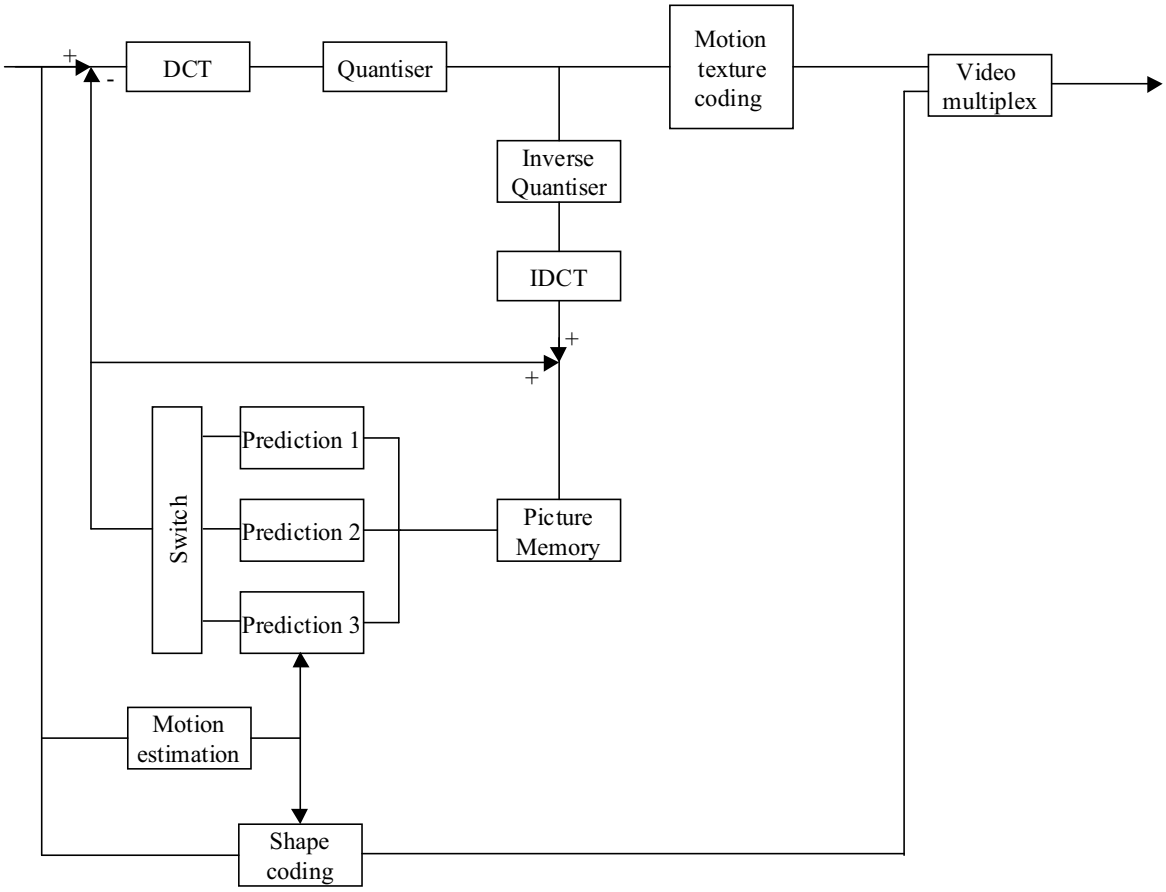

Figure 20 Structure of Video Object Plane (VOP) based encoder

As depicted in figure 21 MPEG-4 coding algorithm encodes the first VOP in Intra frame VOP coding mode. Next frame is coded using VOP inter frame prediction where only nearest previously decoded VOP frame is used for motion prediction. For advanced quality of the images, MPEG-4 supports also use of bi-directionally-coded VOP's[10].

[10] See chapter on motion compensation for H.263 for further information

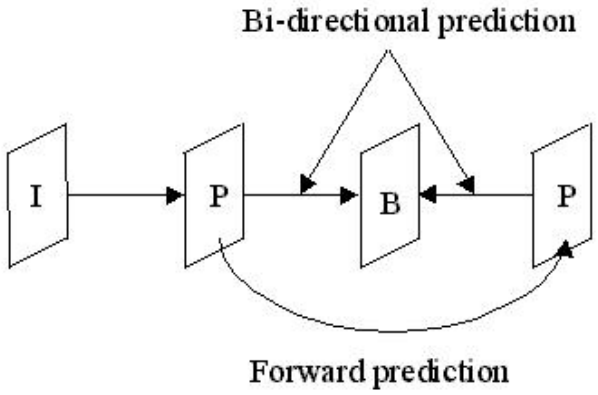

Figure 21 VOP prediction

MPEG-4 video algorithm uses a block-based approach to process images of a VOP sequence (similar to the other MPEG algorithms). The VOP images colour input is separated into macroblocks that doesn't overlap each other. Each of the macroblocks contains 4 luminance- and 2 chrominance-blocks with a size of 8x8 or 16x16 pixels. When a VOP picture is sent, it will be stored in a VOP picture store, indicating that this is the previous VOP picture (N-1) sent, in both encoder and decoder. This VOP picture is used for motion compensation. Motion compensation is achieved on a block or macroblock basis and only one motion vector is estimated between VOP picture (N) (current VOP picture) and VOP picture (N-1) for a block and a macroblock to be encoded (Sikora, 1997b). The motion compensation prediction error is calculated by subtracting each pixel in a block or macroblock that belongs to the current VOP picture (N) with its surrounding area in the previous VOP picture (N-1). Further is an 8x8 or 16x16 DCT (or alternatively shape adaptive DCT[11]) added to the 8x8 or 16x16 blocks contained in the block or macroblock followed by quantisation of the DCT coefficients to divest the high frequencies. The size of the quantising steps for the DCT can be adjusted for each macroblock depending on the targeted bit-rate. Lastly subsequent run-length coding and entropy coding is added (see H.263 chapter).

[11] The shape adaptive DCT (SADCT) is applied to the boundary 8x8 blocks of the VOP, and the concept is to perform vertical DCTs of the active pixels first and then perform horizontal DCTs of the vertical DCT coefficients with the same frequency index (Li, 1997).

Input images in the VOP layers that is going to be encoded are of arbitrary shape and location that varies with time if it is compared to a reference window. All the VOLs that is going to be encoded for a given input video sequence are defined with reference to the reference window. Figure 22 illustrates a VOP image inside a reference window and a shape adaptive macroblock grid for the VOP image.

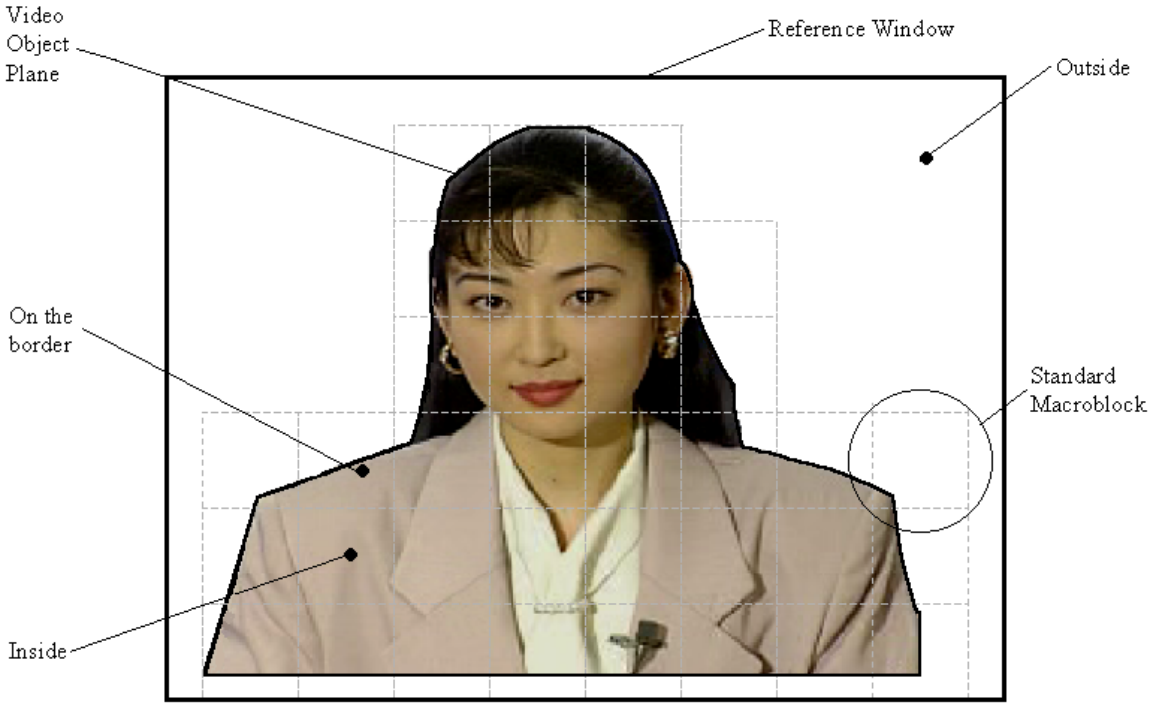

Figure 22 An MPEG-4 macroblock grid for a VOP image

The macroblock grid is used for the VOP's shape information, motion-estimation and -compensation and for block and DCT-based texture coding. The VOP window is built up in multiples of 16 pixels in both horizontal and vertical direction of the VOP image, by 16x16 pixel macro blocks.

Shape coding

In MPEG-4, it is assumed that VO's is provided with its corresponding shape information. This shape information is presented in two formats (Brailean, 1997):

- Binary format – is made up of a pixel map, where each pixel has value information whether the pixel is within the VO or not.

- Grey scale format – is similar to binary format, however the pixel value can have a value ranging from 0-255. Value 0 indicates transparency and 255 indicate that the pixel is opaque. Values between 0 and 255 indicate to an intermediate intensity of transparency.

The shape information is encoded by bounding the VOP with a rectangular box (reference window) and then dividing the bounding box into shape blocks. Each shape block is classified as lying (see figure 22) inside the object, on the border of the object or out side the object (inside the reference window).

Motion estimation and compensation

As for MPEG-1, MPEG-2, H.261 and H.263, MPEG-4 gain benefit of block-based motion estimation and compensation techniques to explore temporal redundancies of the video information in the separate VOP layers. The principle for motion estimation and compensation is depicted in figure 23, where the intention is to find the best-matched block in the previous VOP frame for every block in the current one. A motion vector is estimated for each current block and indicates the displacement of the previous block. The motion compensated prediction error is estimated by subtracting the pixel values of the predicted block from the pixel values of the current block (Li, 1997).

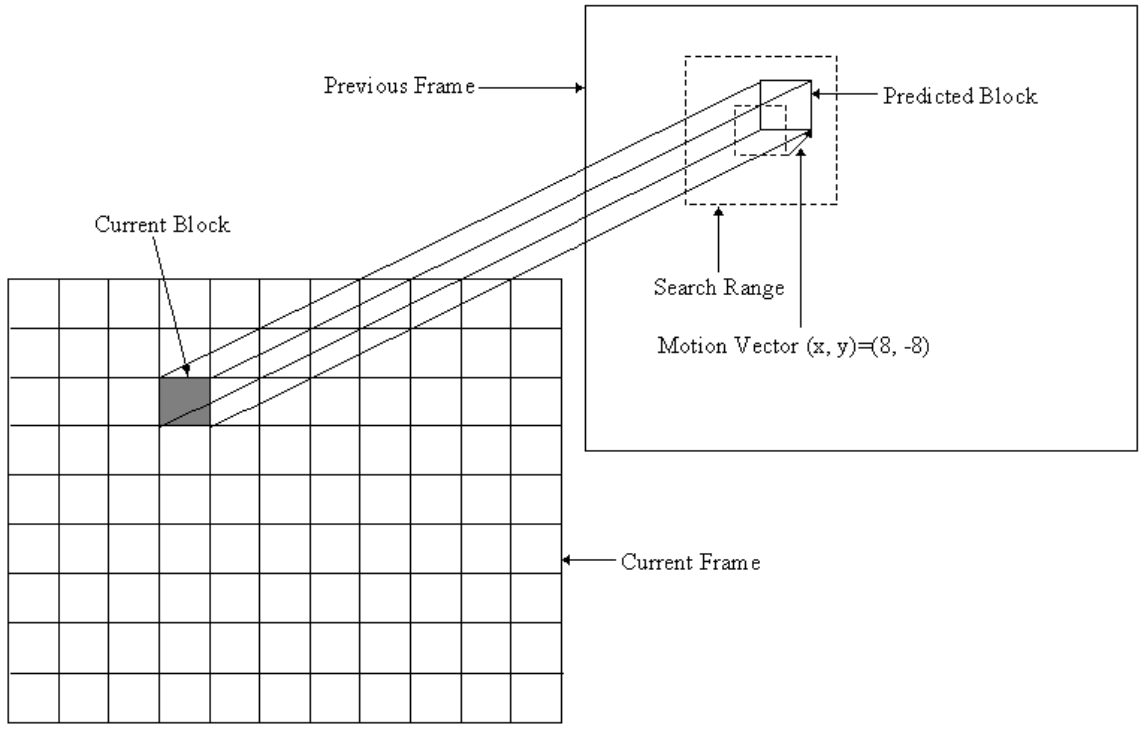

Figure 23 Motion Estimation and Compensation

Motion estimation and compensation in MPEG-4 introduce furthermore a "padding technique" that excludes the pixels not belonging to the active VOP area. The padding procedure can be regard as a prediction of the pixels outside of the VOP based on pixels inside it (result of a padding technique is illustrated in figure 24). First padding of the previous reference VOP frame is done. After this operation a "polygon" matching method is utilised to define the part of the macro block which is located on the border of the VOP that belongs to the active area inside of the current VOP frame to be encoded.

Figure 24 Image padding technique illustrated

Texture coding

Remember that the motion compensated prediction error is estimated by subtracting the pixel values of the predicted block from the pixel values of the current block. After this, texture coding is performed on the prediction error block. The coded motion vector and the coded texture information is then sent to the decoder. The decoder can then make use of to reconstruct the current block by adding the quantised error prediction block to the predicted block according to the motion vector.

4.6.2.3 Coding efficiency

MPEG-4 coding algorithm provides a very high coding efficiency over a very extensive range of bit-rates. Due to the fact that MPEG-4 makes it possible to code only a single object-layer the VOP input image sequence format and the MPEG-4 video coding algorithm can be made nearly compatible with H.263 and MPEG-1.

4.6.2.4 Scalable coding of video objects

Remember from the section about the MPEG-4 video functionalities (section 4.6.1) stated that one of the key functionalities in MPEG-4 is that the video-coding algorithm supports content-based scalability. This allows end-users to select from various bandwidths, display capabilities or display requests to allow video database browsing and multi-resolution playback of the video content-based on the bandwidth resources available in the particular mobile network. Further, another important concept that MPEG-4 video provides support for, is in the area of object-based functionalities. Object-based functionalities allow coding of VO's of arbitrary shapes. This implies that users that is either not capable of or not willing to reconstruct the full resolution arbitrarily shaped VOP's is able to decode subsets of the layered bit stream to display the arbitrarily shaped VOP's content or objects at lower resolution or quality.

The scalability framework is referred to as generalised scalability and includes the spatial and temporal scalabilities. These two scalability types are discussed in the remainder of this section.

Spatial scalability

Figure 25 illustrates the objective behind spatial scalability scheme. In this illustration, two layers are provided where the "mid processor" performs spatial up or down sampling of the input. Each of the layers supports a VOP at different spatial resolution scales. The downscaled version of the VOP is encoded into the base layer of the bit stream and is encoded with a non-scalable coding technique. The VOP in the enhancement layer is either encoded as a P-VOP (inter predicted-VOP) or a B-VOP (bi-directionally predicted VOP).

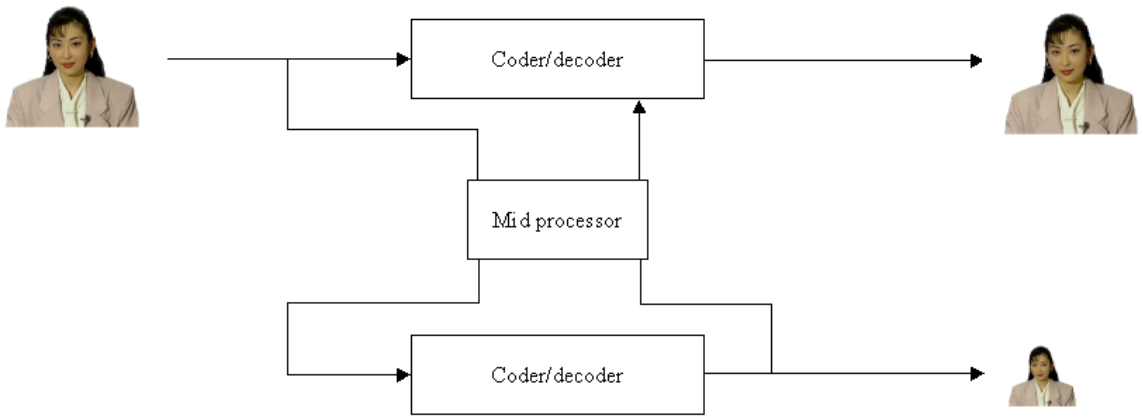

Figure 25 Spatial scalability scheme

This makes it possible for a end-user to choose to display VOP signals with full resolution or VOP signals that is reconstructed by decoding the base layer bit streams (Sikora, 1997b).

Temporal scalability

The temporal scalability offers scalability of the temporal resolution and figure 26 illustrates the objective behind the temporal scalability scheme. In this case the "mid processor does not perform and spatial resolution conversion (ISO/IEC 14496-2, 2000). Layering is achieved through temporal prediction for the enhancement layer based on coded video from the lower layers. Using MPEG-4 temporal scalability makes it possible to supply different display rates on different VOL's within the same video sequence (Sikora, 1997b).

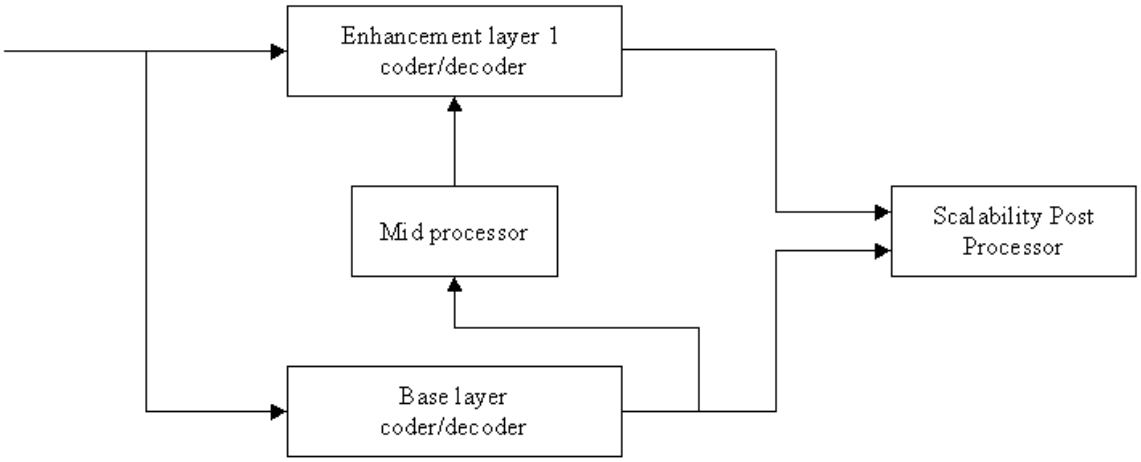

Figure 26 Temporal scalability scheme

4.6.2.5 Error resilience and robustness in error prone environments

As mentioned earlier MPEG-4 provides the means of error robustness and resilience to allow accessing video information over wireless networks. The error resilience tools (FEC-tools) provided in MPEG-4 is divided into three different areas: resynchronisation, data recovery and error concealment. The tools contained in these categories are not unique for MPEG-4, but have been developed to provide error resilience for video transmission (Koenen, 2000). In this section, a description of the error resilience and robustness tools is provided.

Resynchronisation

The tools provided in resynchronisation enables resynchronisation after errors have been detected. The resynchronisation approach implemented in MPEG-4 is a packet approach and is based on providing periodic resynchronisation markers at every N bits (N is an arbitrary integer value) throughout the bit stream. Use of resynchronisation markers allows the decoder to move forward to the next resynchronisation marker when it detects an uncorrectable error, thus lessening the amount of data that must be discarded when a resynchronisation loss occurs. Figure 27 gives an example on loss of data in a video frame. As the figure depicts, almost all the data in the video frame is lost when from where the error occurs.

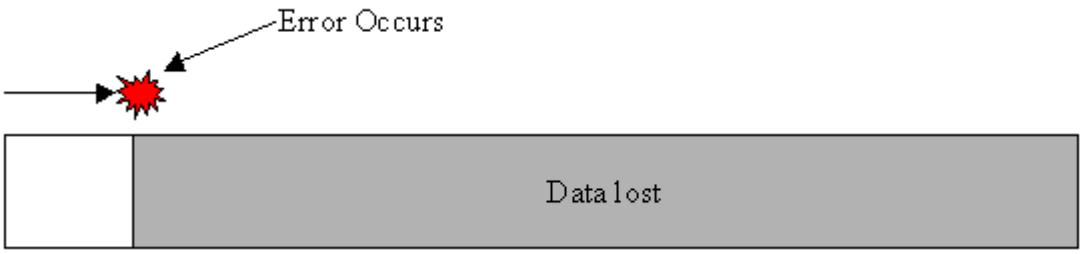

Figure 27 Normal video frame. A lot of data lost in the frame

When the video frame is divided into video packets with resynchronisation markers in-between, less data will be lost due to the fact that the video packets can be decoded independently[12] (Chiariglione, 1997) (depicted in figure 28).

[12] The marker header contains all the necessary information to restart the decoding process

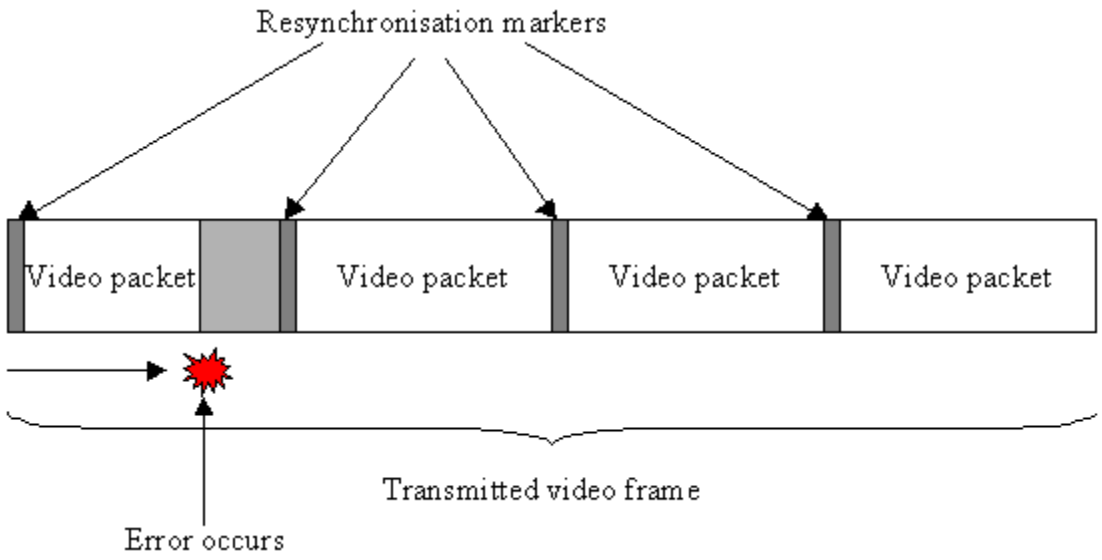

Figure 28 Resynchronisation markers within a video frame divided into video packets

Data recovery

When the synchronisation has been re-established, tool for data recovery attempts to recover data that in general would be lost. These tools are not tools that provide error correction, but techniques that encode the data in an error resilient behaviour (Brailean, 1997). A tool which MPEG-4 provides is Reversible Variable Length Coding (RVLC) where the variable length code-words are designed so it is possible to read them in both forward and in a reverse manner.

In transmissions where an error has corrupted a section of the video packet, all the data between two resynchronisation markers will normally be lost without use of RVLC. Figure 28 illustrates that by use of RVLC some of the data can be recovered. Though this will improve the error resilience significantly, the compression efficiency will be abridged.

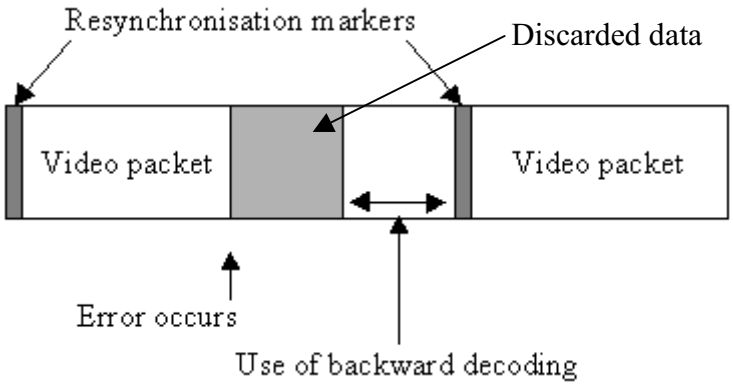

Figure 29 Example on use of RVLC

Error concealment

An extremely vital element of any error robust video codec is error concealment. The error concealment tool is dependent on the performance of the resynchronisation scheme and if the resynchronisation method is able to effectively identify the error then the error concealment problem becomes a great deal more tractable (ISO/IEC 14496-2, 2000).

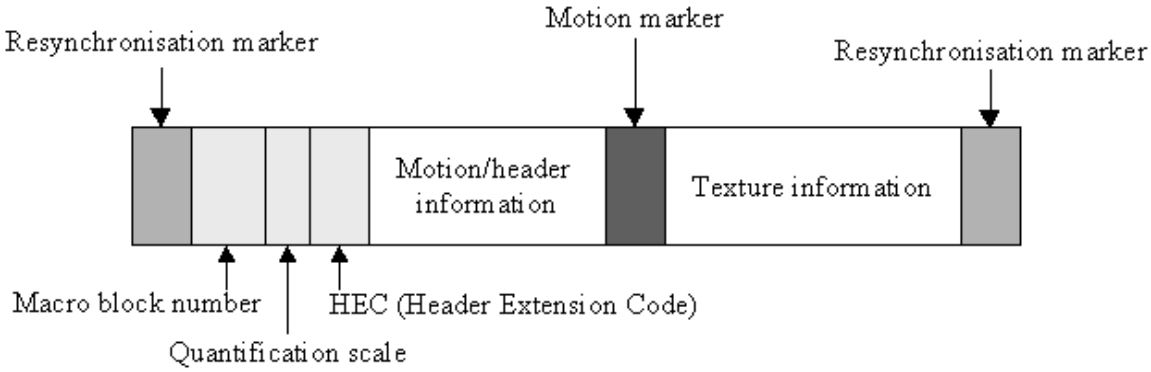

Figure 30 Data partitioning

To further enhance the error concealment capabilities and advance the ability of the decoder to identify errors an approach called "data-partitioning" is exploited[13]. Data

[13] The bitstream is partitioned between the channels so the more critical parts of the bitstream are transmitted in the channel with the best error performance, and less critical data is transported in the channel with poorer performance.

partitioning separates motion and header data from texture data within each of the video packets by inserting a second resynchronisation marker. Data partitioning is signalled to the decoder in the VOL. If an error occurs and the texture information is lost, data partitioning makes use of the motion data to conceal the error by using the motion information to motion compensate the previously decoded VOP (ISO/IEC 14496-2, 2000).

5 Simulation environment

5.1 Introduction

Transmission of interactive video communication services over wireless networks introduces a great deal of technical challenges. It is important for the users of a video service that the quality of the received video meets the expectations. Transmitting real time video over wireless networks is characterised as a very demanding service in terms of bit-rate error sensitivity and delay tolerance. Furthermore, transmissions in wireless networks suffer occasionally from high bit-error rates and varying bit-rate. Connecting the wireless network to the Internet introduces also the error characteristics of transmission over the IP-network, which can be in terms of e.g. packet loss and transmission delay.

Because the bit-rate offered in wireless networks is limited, in comparison with what a transmission of interactive video requires from the network[14], video compression is needed in order to decrease the bandwidth demand of video applications. The higher the video compression rate is makes it exceedingly vulnerable to transmission errors. In addition, the wireless transmission media is highly unreliable due to that the video signal may be attenuated during the transmission in the radio channel. Following phenomena encountered in a mobile radio channel normally causes this unreliability:

- Multipath fading (e.g. Rayleigh- and frequency selective fading)
- shadowing (i.e. obstructions between the transmitter and the receiver, e.g. buildings or hills)
- path loss/delay
- doppler shift
- added noise

[14] 3rd generation networks opens for interactive video communication, however the video needs to be compressed.

These characteristics may have multiple effects on the transmitted video signal and triggers the receiver to deal with a bitstream that may consist of randomly occurring bit-errors or burst-errors (e.g. gaussian white noise). On top of this the receiver video bit stream may also be influenced by the characteristics present when sending video over IP (e.g. packet loss).

In order to comprehend the effect of the various noise factors consequence on the transmitted video sequence, a simulation environment has been developed. The remainder of this chapter presents the construction of the simulation environment along with a portrayal of the evaluation methods used to assess the quality of the video sequence after the distortion factors are simulated.

5.2 Video performance experiments

Experiments were performed to test the quality of the MPEG-4 video codec under different conditions of mobile noise. To evaluate the quality of a degraded video sequence with the original one, two evaluation methods were singled out to measure the quality.

Distortion Measure

Generally, the Root Mean Square Error (RMSE) is used as a measure to assess the video degradation due to noise that is present in the radio channel. To compute the value of RMSE, the video sequence has to be divided into grey scale (8 bit) frames of size NxM pixels (for QCIF 176 x 144). Then the original pixel frame is compared with the degraded one (see formula beneath).

$$RMSE := \sqrt{\frac{\sum_{x=0}^{N}\sum_{y=0}^{M}\left(f(x,y)_1 - f(x,y)_2\right)^2}{N \cdot M}}$$

Where,

$f(x, y)_1$ – is the original frame, and

$f(x, y)_2$ – is the degraded frame

The RMSE value will become zero when the original- and degraded frame is identical. The final RMSE value is calculated as an average value calculated from all the transmitted video frames. The actual average value that is calculated has no meaning in it self, however a comparison of two values for different reconstructed video sequences presents a measure for quality (Stuhlmüller, Färber, Link and Girod, 2000).

The more common approach to measure quality of video is to calculate the Peak Signal to Noise Ratio (PSNR). PSNR is derived using RMSE to refer to a video frame of degraded quality from the original in dB (see formula) (Stuhlmüller et. al., 2000).

$$PSNR = 20 \log_{10} \frac{255}{RMSE}, \text{ where}$$

the value 255 is derived from assuming that it is an eight-bit video frame (2^8). The PSNR values were calculated for each frame and then averaged over the total number of frames in the video sequence.

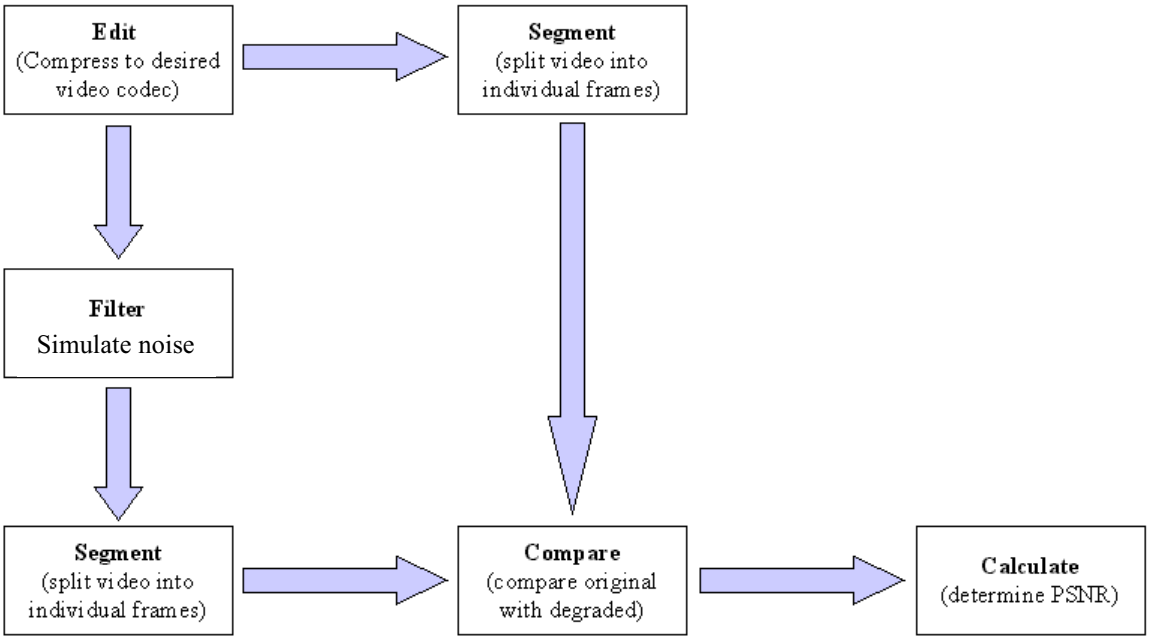

Figure 31 Objective evaluation set up

The approach of calculating the RMSE and PSNR is shown in figure 31. First of all the original video sequence is compressed into a desired format of video (in this case MPEG-4 of QCIF resolution). With help a video manipulation application (virtualdub.com), the original compressed format (prior to noise insertion) is segmented in to individual frames of grey scale format of 8 bit[15]. Furthermore, the original compressed video is simulated various distortion factors by help of reading the video sequence into a C-program

(developed within the framework of this thesis) where various filter functions have been coded to manipulate the content of the video sequence (Appendix B, "simulate noise.c"). This distorted video sequence is then fragmented as described earlier. Finally, the impaired video frame is compared (Appendix B "PSNR calculation.c") with the original compressed frame to estimate the Peak Signal to Noise Ratio.

Subjective evaluation

Another important aspect in evaluating the quality of the video is to perform a subjective assessment on the resulting degraded video sequence. This is because that the calculated PSNR values do not enlighten without a person's view of the difference between the original and the degraded video sequence. The subjective judgement was valid for all of the different error rates and was rated from 1 to 5 where 1 is low quality and 5 is implies that the video is of good quality.

The approach to go through with the subjective assessment of the video quality is comparable with the way selected in the objective evaluation (see figure 32, next page). The difference is that the video sequence does not have to be segmented in the same way as for the objective evaluation. Here is the video sequence loaded directly into the filter functions which simulates the noise that may affect the signal in a 3rd generation mobile network.

[15] It is normal to measure the PSNR of only the luminance (brightness) in the video frame

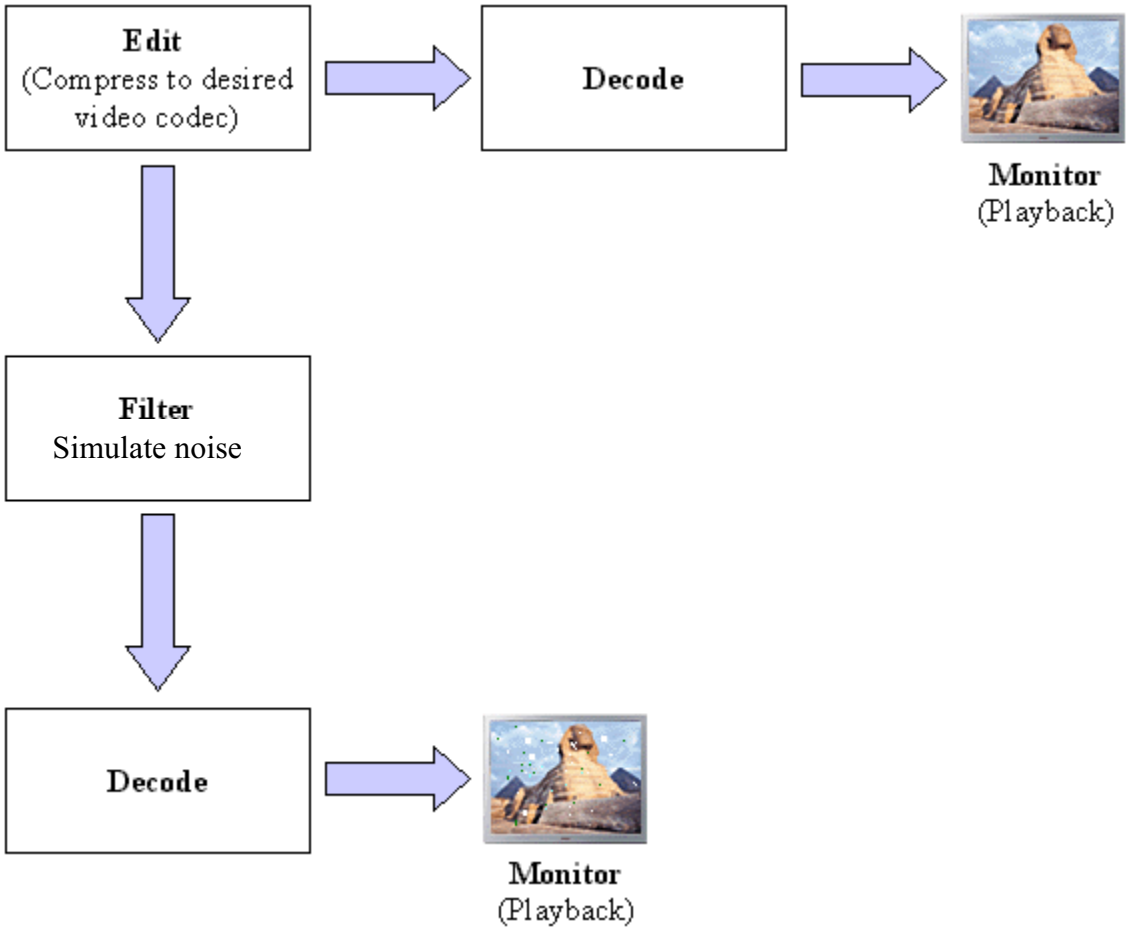

Figure 32 Subjective evaluation set up

6 Simulation results

The video sequence used in the experiments was the Akiyo video sequence. This sequence is regarded as a low-motion sequence (typical for videoconferencing) and is selected from a standard set of test sequences the have been used throughout the development of both the H.263 and MPEG-4 standard. The video sequence contained no additional audio. The table beneath shows the simulation parameters to the video sequence.

Table 10 Simulation parameters

Video codec	MPEG-4			
Frame size and rate	QCIF (176x144 pixels) at 30 fps			
Video sequence	Akiyo (300 frames, 10 s)			
Error Rates	Bit Error Rate	Burst Error Rate	Packet loss	Gaussian Error Rate
	10^{-3}, 10^{-4}, 10^{-5} and 10^{-6}	10^{-3}, 10^{-4}, 10^{-5} and 10^{-6}	1 %, 5 %, 10 % and 20 %	10^{-3}, 10^{-4}, 10^{-5} and 10^{-6}

The simulation environment had the capability of generating bit errors, burst errors, packet loss and Gaussian white noise (random bit errors). The video sequence was then tested on various impairment degrees to get an understanding of the effects that the various forms of noise could have on the video. The effects and evaluation of the results is shown and commented through the remainder of this chapter.

6.1 Effects due to bit error

To simulate what effect bit errors have on the quality of the video; bit error rates (BER) ranging from 10^{-6} to 10^{-3} were inserted into the bit stream.

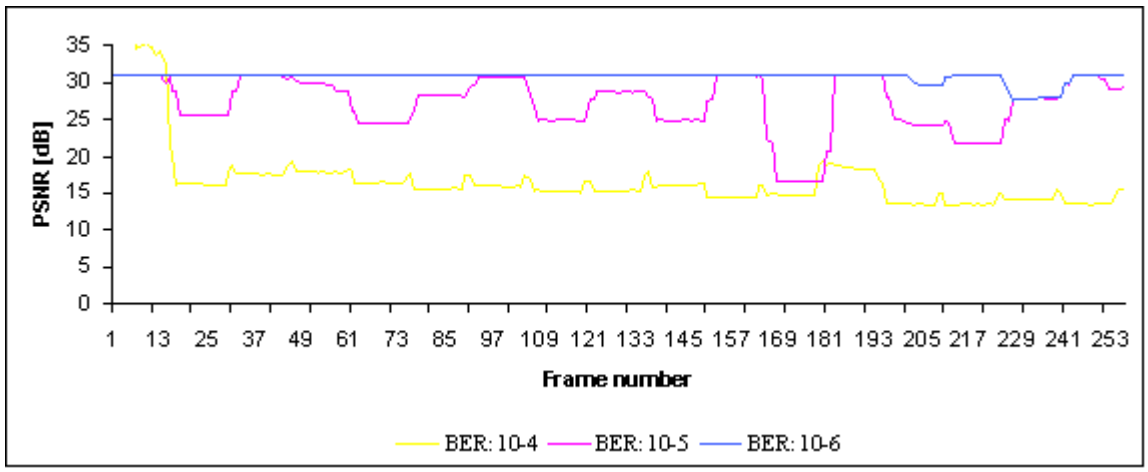

Figure 33 PSNR values for bit error noise simulation

The figure above shows the calculated PSNR values for each of the frames in the video sequence. As we can se from the figure the quality severely degrades when the BER becomes larger than 10^{-5}. The same discovery can be made by studying the figures beneath were snapshots of the video quality is shown at selected bit error rates.

(10^{-4}) (10^{-5}) (10^{-6})

Figure 34 Video quality for various bit error rates

The snapshot were the BER is 10^{-4} clearly shows that the image is degraded in form of blockiness, where as for the snapshots were the BER is 10^{-5} and 10^{-6} it is difficult to see any degradation at all. Though it is tricky to see the degradation caused by bit errors in

the two latter cases, it is possible to compare the results by comparing the calculated average PSNR for the two video sequences. The average PSNR values together with the subjective judgement is summarised in the table beneath.

Table 11 Effects of bit error on the video quality for the Akiyo video sequence

Video Sequence	Bit error rate	Subjective judgement	Average PSNR
Akiyo (400 kbit/s) (MPEG-4)	10^{-3}	1 (very poor)	N/A
	10^{-4}	2 (very poor)	16.66 dB
	10^{-5}	3 (fair)	27.52 dB
	10^{-6}	4 –5 (good)	30.88 dB
	0	5 (good)	N/A

6.2 Effects due to burst error

The second experiment conducted was to see how the video behaved when simulated burst errors were added to the bit stream. Due to the limitations in the software PSNR values or subjective evaluation could not be performed for BER on 10^{-4}. Meanwhile, simulation on the effects burst error had on the video could be performed for BER 10^{-5} and 10^{-6}, where the length of the burst errors was set to last 1 millisecond.

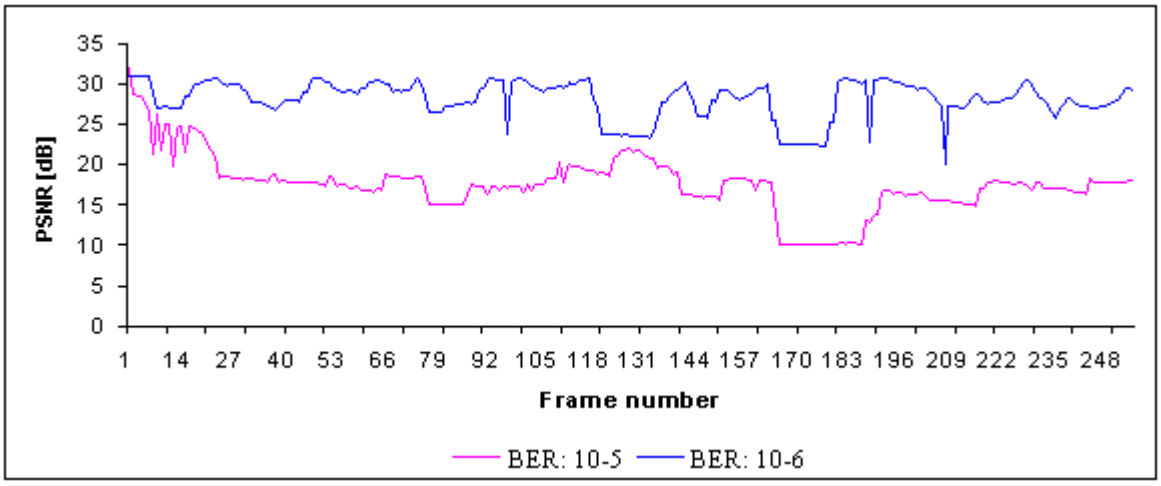

Figure 35 PSNR values for burst error noise simulation

Again, figure 35 shows the PSNR values of the simulation for two different error rates. Comparing it with the case where only single bit errors occurred, one can see that the PSNR values have decreased when numerous burst errors were added to the bit stream. Furthermore, the errors more visible at the BER 10^{-5} and 10^{-6} due to that the error bursts lasts for about 1 millisecond.

(10^{-5}) (10^{-6})

Figure 36 Video quality for various burst rates

A recap of the quality to the video sequence is shown in the table beneath. The average PSNR for BER 10^{-4} could not be achieved because the quality was too bad.

Table 12 Effects of burst error on the video quality for the Akiyo video sequence (1 ms)

Video Sequence	Burst error rate	Subjective judgement	Average PSNR
Akiyo (400 kbit/s) (MPEG-4)	10^{-3}	1 (very poor)	N/A
	10^{-4}	1 (very poor)	N/A
	10^{-5}	2-3 (poor)	19.00 dB
	10^{-6}	4 (good)	28.10 dB
	0	5 (good)	N/A

6.3 Effects due to packet loss

To simulate loss of packets in the simulation environment was simply done by replacing the original content in the video file with series of 0's where each 0 was represented by one byte. The length of the packets was fixed through the whole "transmission". The PSNR values for different percentage of packet loss rate is depicted in figure 36.

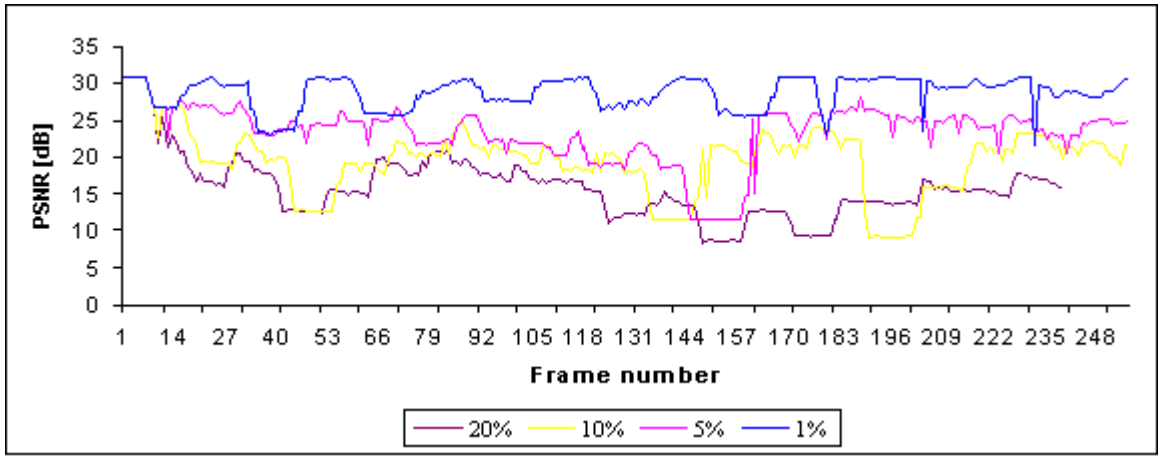

Figure 37 PSNR values for packet loss simulation

The figures beneath show the quality of the video during playback of the video for different percentage packet loss. Not surprisingly, decreases the quality of the video as the loss of packet's increases.

(20%)

(10%)

(5%)

(1%)

Figure 38 Video quality for various packet loss rates

Looking at the visual result of the damage the packet loss have added to the video it was evaluated and inserted into a table together with the calculated average PSNR to compare it with results of the degrees of packet loss.

Table 13 Effects of packet loss on the video quality for Akiyo video sequence

Video Sequence	Packet loss	Subjective judgement	Average PSNR
Akiyo (400 kbit/s) (MPEG-4)	20 %	1 (very poor)	15.90 dB
	10 %	2 – 3 (poor to fair)	19.67 dB
	5 %	3 (fair-poor)	23.44 dB
	1 %	4 (good)	28.50 dB

6.4 Effects due to gaussian noise

The last simulation performed was to investigate the video quality when the bit stream was exposed to random bit error rates ranging from 10^{-6} to 10^{-3}. This was accomplished through using the "rand ()" function in C.

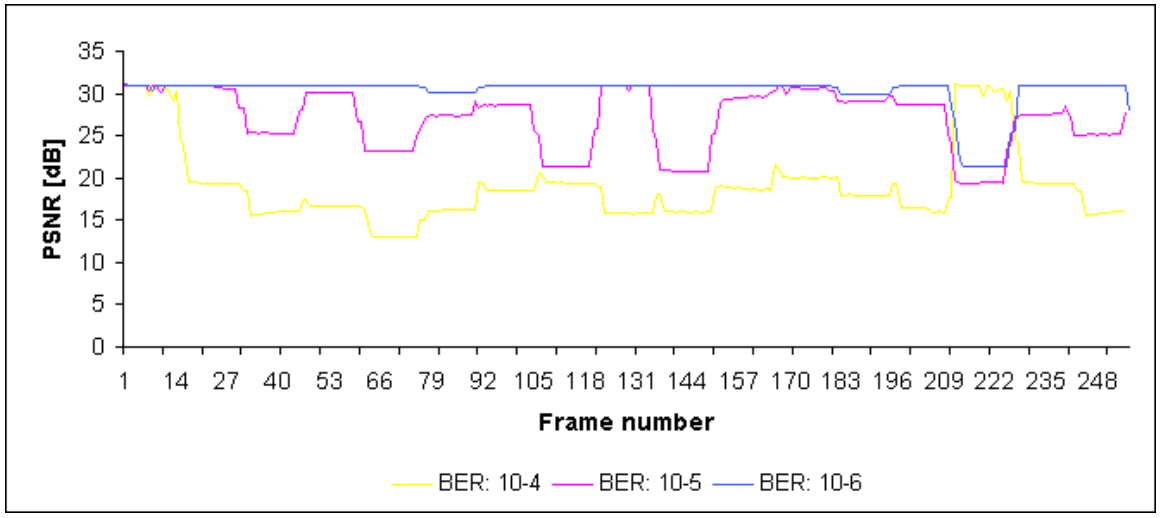

Figure 39 PSNR values for Gaussian noise simulation

Inserting bit error randomly may cause that all the bit error is located very close to each other. From the figure above, one can notice that this has happened for BER = 10^{-6} in the frame interval from frame number 211 – 229. Though in some cases randomly occurring bit errors may visually appear worse than bit errors spread over the entire video sequence, the average PSNR will be pretty much the same.

(10^{-4}) (10^{-5}) (10^{-6})

Figure 40 Video quality for various gaussian error rates

If you compare table 11 and 14, you can see that the calculated PSNR values are similar. However, the visual quality was slightly worse when BER was 10^{-5}.

Table 14 Effects of Gaussian noise on the video quality for Akiyo video sequence

Video Sequence	Bit error rate	Subjective judgement	Average PSNR
Akiyo (400 kbit/s) (MPEG-4)	10^{-3}	1 (very poor)	N/A
	10^{-4}	2 (very poor)	18.05 dB
	10^{-5}	3 (fair-poor)	27.16 dB
	10^{-6}	4 – 5 (good)	30. 50 dB
	0	5 (good)	N/A

7 Discussion

When I first sat down and wrote the scope of this thesis, the intention was to use the video codecs MPEG-2 and DV in the simulation experiment. The intention was to use DV as a reference format and encode it to MPEG-2, simulate noise and evaluate the results. However, since MPEG-2 is far from being a standard used in video transmission in heterogeneous wireless networks it came to my mind that this was quite “lame”. After reading papers on video transmission over wireless networks, I learned that there were two codecs that discerned themselves from the rest of the codecs. These two codecs H.263 (+) and MPEG-4 are optimised for transmission at very low bit-rates (10 kbit/s to 24 kbit/s for H.263 and 5 kbit/s to 4 Mbit/s for MPEG-4) and allows transmission in heterogeneous error prone environments. However, due to that MPEG-4 and H.263 (+) is strongly based on the preceding standards MPEG-1/2 and H.261 I felt that an overview of these standards had to be given due to the importance of them in the past (VideoCD, DVD, video conferencing in ISDN-network).

Selecting these two video codecs for use in the simulation, I learned that very few commercial software programs were developed. Moreover, none of these commercial products was made to support error resilience. Encoders and decoders have been developed in research environments and at Universities, but are difficult to get reach of. Therefore I used a combination of commercial software were I was able to encode video sequences into MPEG-4 format and a self developed simulation environment based on the programming language C were simulation of noise and evaluation of the quality after noise simulation could be performed. The only constraint that remained was that error resilience tools could not be added to the encoded sequence due to limitations in software and time.

With this in mind, simulations were done and results were produced. The following sections discuss what my foundation of for evaluation criteria was together with the results from the simulation and, finally, the validity of the simulation process.

7.1 Evaluation criteria

7.1.1 Evaluation methods

During simulation of noise, two evaluation methods were used to estimate the quality of the video. The decision of objective and subjective evaluation had to be in place to get the right judgement of the video quality.

Selecting only one of these methods could not give a complete picture of the video's quality. For example, an objective evaluation would give you a product of a calculation, which can be used to compare video sequences with various degree of error. However, it would not display anything about the visual quality. In addition, for example simulation of a uniformly distributed BER of 10^{-3} would give more or less the same result (PSNR) as a BER of 10^{-3} of Gaussian distribution. This can be concluded because the number of bit errors in the video stream would be the same, but with Gaussian distribution the bit errors would be placed randomly and might result in that the bit errors is situated in only sections of the video stream. This result in that some portions of the bit stream, visually, would be of very deficient quality and other portions might not visually show any degradation at all.

7.1.2 Video Sequence

Various video sequences have been used by the industry to evaluate the performance of the video codecs. One of these video sequences was also used in the simulation environment described in chapter 5. The advantage of selecting a video sequence that is used in tests is that the results can be compared to see if they have meaning. A disadvantage could be that you are "blinded" by the results in such way that your evaluation is not only based on your own simulation results.

To get a trustworthy evaluation of the video sequences' response to errors in the bit stream, the sequence could not be to short. The objective evaluation was based on an average calculation of the PSNR. A video sequence that was to short would not provide an average PSNR that was applicable describing the video's quality.

The quality of the video sequences degrades when the encoding bit-rate lessens. Selecting a video sequence encoded at a low bit-rate would also make the sequence more vulnerable to distortions in the transmission cycle. This means that a video sequence encoded at a bit-rate of e.g. 120 kbit/s may be extra vulnerable towards error propagation in the bit stream, compared with a sequence encoded at 400 kbit/s that was used in this simulation. As an effect, bit streams containing errors can be difficult to decode because most of the commercial decoders on the market today are not optimised for handling streams containing bit errors.

Further, the frame rate would have an effect on the transition between each frame in the sequence. Deciding on a frame rate that is low (e.g. 5 fps) might caused a jerkiness effect in the playback of the video, however this would cause that the amount of information transmitted is very little compared with selecting a relatively high frame rate (e.g. 25 fps). In this simulation, the video sequence kept the original frame rate of 30 fps because then it was easier to evaluate the visual quality after error simulation. Selecting a video sequence encoded at 15 fps would in it self degraded the quality of the video and made it more difficult to evaluate the consequences due to error propagation.

7.2 Results

The video codecs of current interest (MPEG-4 and H.263) for use in UMTS provides means for upholding the quality of the video sent over the wireless network. The error resilience parameters are not necessarily turned on as standard. As for the decoders, commercial encoders are not optimised for providing the techniques to preserve the error resilience of a video codec. Special configuration files must be in place to enable the FEC (Forward Error Correction) tools (e.g. error concealment) that is taken advantage of in these two codecs.

Using a commercial encoder and decoder implied that none of the FEC tools was implemented in the simulation environment so that evaluation was done solely on the effects of errors occurrence in the video sequence.

7.2.1 Bit error

The presentation of the simulation results for insertion single bit errors are shown in figures 33 and 34 and are summarised in table 11. A BER of 10^{-7} was also simulated to see if there was difference in quality compared with a BER of 10^{-6}. As depicted in figure 33, the PSNR values for each video frame simulated with BER of 10^{-6} are more or less constant the whole interval. This implies that little or no error is found in a video sequence simulated with a BER of 10^{-7} and the average PSNR value for an error free video sequence will be close to the BER of 10^{-6} (30.88 dB). This corresponds with other simulations were no or little degradation of the video signal occurs at BER of 10^{-6}.

The objective results (PSNR values) correspond with the subjective observation in figure 34. Degradation can be seen at the centre bottom of the picture and may have been eliminated by use of FEC tools.

As anticipated, both the objective and subjective results became worse when the BER increased. Notice that the quality severely degrades when moving from BER 10^{-5} to BER 10^{-4}. If you observe for BER $10^{-5,}$ the signal returns to the "optimal" value of periodically. The reasons for this is due to the INTRA coded pictures, which contain all the information about a picture. For the BER of 10^{-4} the density of the bit error occurs is so immense that bit error occurs in all of the I-pictures, as for BER 10^{-5} only a few I-pictures are corrupted.

7.2.2 Burst error

Not unexpectedly had simulation of burst error increased degradation effect on the video sequence compared with insertion of single bit errors. From the objective results, we can see that the PSNR values for a Burst Error Rate of 10^{-6} fluctuates a great deal. However, the subjective result was not especially poor. Of course was the degradation of the signal additionally visible, but the overall view was good.

Conversely, when the Burst Error Rate became 10^{-5} the poverty in the signal increased dramatically. In comparison with simulation of single bit errors, were PSNR dropped from 30.88 dB (10^{-6}) to 27.52 dB (10^{-5}), the PSNR results fell from 28.10 dB (10^{-6}) to 19.00 dB (10^{-5}).

The reason for this is quite obvious, because single bit errors appears in "chunks" (bursts) that is present in 1 ms and visually cause nasty appearances in the video sequence (see figure 36 and 10^{-5}).

7.2.3 Packet loss

In this case, the commercial software was more tolerant to the errors that occurred in the video sequence. A reason for this could be due to insertion of zeros in contradiction to insert a mix of zeros and ones that is done in the other simulations.

The appearance of errors for a packet loss of 1% was quite unnoticeable in the subjective evaluation. On the other hand, increasing the packet loss to 5% the appearance of the video becomes less appreciated. The drop in quality of the video sequence for packet loss rates for 1% to 5% and from 5% to 10% is quite noticeable from a subjective perspective. Conversely, for packet loss rate from 10% to 20%, the drop in visual quality is not so obvious, despite the fact that the fall off for the average PSNR values between all of the packet loss rates is in the range of 4 dB and 5 dB.

7.2.4 Gaussian noise

In this simulation, it could be expected to get a result close to the average PSNR values calculated for simulation single bit errors. This can be predicted due to that the total bit errors are the same, but in the Gaussian distribution of the bit errors implies that the errors are randomly situated in the bit stream.

The results shows that the average PSNR values is close to the values found for simulation of single bit errors. In addition, if we study the PSNR values for each video

frame in figure 39 we can see that the bit errors have quite similar pattern. This was quite unexpected, therefore the simulation was executed a couple of times to see if the random function in C managed to create a pattern where it clearly demonstrated what could happen with Gaussian distribution of the bit errors. The result was that it for one instance created a sequence that illustrated the case of what may be possible for bit errors of Gaussian distribution and is clearly displayed in the interval from frame number 157 to 248.

7.3 Validity of the simulation method

The entire simulation method was based on functions written in C-programming language. The structure of the code (shown in appendix B) was very simple and should not be hard to understand.

7.3.1 Noise simulation

The simulation environment for simulation various kinds of noise consists of four main functions.

- filter_bit_error
- filter_burst_error
- filter_packet_loss
- filter_gauss_error

Manipulating the video sequence was quite straightforward. All video files has a file header that describes the structure of the file (type of video, length, bit-rate, etc.). Thus, the only thing that had to be considered before the simulation of the various error conditions was to implement an error free period in the beginning of the video sequence. In “real life”, this error free period could be implemented by using the data portioning tool in MPEG-4. Use of this tool makes it possible to give this portion of the video sequence a higher priority so that this portion has higher error robustness when it is send over the mobile network.

The content of the video sequence was read in to the functions on byte by byte basis. Therefore, a KEY was used to simulate an occurrence of a bit error within a bit stream. The length of this KEY was one byte and had the ability to change the value of a bit in a byte by xor'ing these two byte values. This KEY had the same value for each byte that was read and manipulated. A variable KEY that had the ability to change different bits for each byte would have been interesting explore to see if it had an effect on the calculated PSNR results for the same video sequence and equal error conditions, but was not considered.

The same KEY-value was used in manipulation of the video sequence for simulating both burst and Gaussian error patterns. Maybe one can question the use of the random (rand () – for C programming language) function to generate the Gaussian noise samples. However using of this function gave an insinuation of how the video's quality was affected.

Simulating packet loss, the program simply overwrites the original content with series of "0"s. The length of the video packets that is sent over packet switched networks may vary. In IPv4, the length of the packet varies according to how much the data routers are capable of sending. In the upcoming IPv6, it is possible to negotiate a packet length that is valid for the complete transmission (end-to-end). Minimum legal packet length of an IP packet is 512 bytes. This includes IP-header and selectively RTP header (for videostreaming) and other header information necessary in the transmission. In UMTS, it will maybe be most profitable to negotiate to get a packet of minimum size to avoid too high loss of information when the conditions become bad.

7.3.2 Objective noise evaluation

The choice of calculating the PSNR values for the luminance component only might not give the full view of the quality of the degraded video sequence. However, the human eye is more sensitive to luminance of an image than the changes in colour (explained in

section 4.1). Therefore calculating a PSNR value for only the luminance component provides sufficient amount of information about the video quality.

8 Conclusion

In the framework of this thesis, video services and video compression standards for supporting real time video transmission over UMTS have been discussed. Of all the services available for the 3rd generation multimedia systems, transmission of real time video is the most demanding one in terms of bandwidth requirements and transmission delay. In addition to the rigid requirements of bandwidth requirements and transmission delay, The transmission of video must maintain certain quality of service requirements.

The standardisation of higher bandwidth networks like UMTS, together with video compression techniques makes it possible to transmit video over mobile telecommunications networks. The error propagation in such networks, however, effects the visual representation of the transmitted video. In addition, the highly compressed video data is extremely vulnerable towards transmission error. Since low bit-rate video coding schemes rely on interframe coding to achieve high coding efficiency. Consequently, the loss of one transmitted video frame has major impact on the quality of the following video frames.

Error correction methods in form of retransmission may be used to contend the error propagation in the mobile network, but in video services like videoconferencing, the requirements to delay are strict so that any form of retransmission is not wanted. The video codecs like MPEG-4 and H.263 (+) provide forward error correction methods to comprehend some of the corrupted information. However, because transmission errors in heterogeneous mobile channels can range from single bit errors, burst errors, packet loss or maybe completely loss of signal, the forward error correction methods may not be efficient.

The use of the scalability in MPEG-4 and H.263 (+) can be useful to contend the changing bit-rates present in the mobile network. Video in these two standards may be encoded in several layers, one base layer and one or more enhancement layers, providing different bit-rates at a certain layer. If the available bandwidth allows video transmission

of good quality the base layer and the enhancement layers may be decoded. A congested network may not be able to assign the necessary bandwidth for the transmission of all the video layers. If the network can provide this information to for example a streaming server, the server can then just leave out the video data of one layer. This provides an effective mechanism to make available different bit-rates fast and may prevent a congested network. This is on expense on the quality, but compared with a video without scaling possibilities where the entire signal may disappear, the solution is very much acceptable.

When it comes to what kind of video codec to select for a video service the choice depends on the users expectations of the QoS as well as on network conditions and constraints in the hardware of an UMTS terminal. For example, MPEG-4 provides the means for object based scalable coding. Within a videoconference or video telephony session the most important object to transfer is the person itself. This means that a base layer of a video stream may only consist of that object and the enhancement layers contain additional information about the background and so forth. However, MPEG-4 is a codec that has substantial requirements to processing power due to its complexity compared with H.263 (+). For a handheld UMTS terminal with limited memory and computer power H.263 (+) may be the right video codec to choose. 3GPP2 suggests also that this may be up to the user of the service to select the error robustness within a communication session.

The simulation analysis within this thesis gives a basis for comprehending what effect error propagation in a mobile channel can have on an MPEG-4 encoded video stream without error resilience. The videoconferencing and video streaming service shall, according to 3GPP2, work under conditions of BER 10^{-3} and a packet loss of 20%. Bearing this in mind, the simulation results show that the need for error robustness is enormous for transmissions over mobile networks as well as fixed IP networks.

8.1 Further studies

In this thesis a simulation of the MPEG-4 video codec was performed. In addition, the video codec was simulated without any form of error resilience tools added. Further studies can be done were various error resilience additions are experimented with. Moreover, a study on the performance of the two video codecs could be carried out to see which video service they are suited for and which of the two codecs performs better under various network impairment conditions when error resilience tools are applied.

9 Bibliography

- 3GPP TSG Terminals TS 23.140 v 3.0.1 (2000). *Multimedia Messaging Service (MMS) Functional description Stage 2* [Online] Available: http://www.3gpp.org

- 3GPP2 Specification S.R0022 (2000). *Video Conferencing Services, Stage 1* [Online] Available: http://208.45.131.70/docs/newsd/3GPP2_Specs_Doc_New.html

- 3GPP2 Specification S.R0021 (2000). *Video Streaming Services, Stage 1* [Online] Available: http://208.45.131.70/docs/newsd/3GPP2_Specs_Doc_New.html

- Adobe Systems Inc. Dynamic Media Group (2000). *A Digital Video Primer* [Online] Available: http://www.adobe.com

- Alnuweiri, H., Kossentini, F. and Erol, B. (2000). IEEE transactions on Circuits and Systems for Video Technology, No. 6, pp. 843-856 September 2000. *Efficient Coding and Mapping Algorithms for Software-only Real-Time Video Coding at Low Bit Rates*

- Brailean, J. (1997). *ISCAS Tutorial on MPEG-4. Chapter 3.3* [Online] Available: http://bs.hhi.de/mpeg-video/contrib/iscas97/

- Cherriman, Dr. P. (1999). *H.263 Video Coding* [Online] Available: http://www-mobile.ecs.soton.ac.uk/peter/h263/h263.html

- Chiariglione, L. (1997). *MPEG and multimedia communications* [Online] Available: http://www.cselt.it/leonardo/paper/lcpaper.htm

- Chiariglione, L. (1998). *MPEG-4, why use it?* [Online] Available: http://www.cselt.it/leonardo/paper/lcpaper.htm

- CodecCentral (2000). *Glossary*. [Online]
  Available: http://www.icanstream.tv/info/glossary.html

- Côte, G., Erol, B., Gallant, M. and Kossentini, F. (1998). IEEE transactions on Circuits and Systems for Video Technology, pp. 849-866, November 1998. *H.263+: Video Coding at Low Bit Rates*

- Côte, G., Kossentini, F. and Shirani, S. (2000). IEEE Journal on Selected Areas in Communications, vol. 18, no. 6, pp. 952-965, June 2000. *Optimal mode selection and synchronization for robust video communications over error prone networks* .

- Ericsson (2000). Internal technical documents

- Fitzek, F. and Reisslein, M. (2000). TKN Group TU Berlin. Technical Report TKN-00-06. *MPEG-4 and H.263 Video Traces for Network Performance Evaluation.*[Online]
  Available: http://www-tkn.ee.tu-berlin.de/~fitzek/TRACE/pub.html

- Horn, U., Scheider, A., Hartung, F and Niebert, N. (2000). Proceedings World Telecommunications Congress (WTC/ISS2000), Birmingham, UK, May 2000. *Mobile Multimedia Streaming – The Real Challenge for UMTS*

- ISO/IEC 13818-2 (2000). *Information technology – Generic coding of moving pictures and associated audio information: Video*

- ISO/IEC 14496-2 (2000). *Information technology – Coding of audio-visual objects – Part 2: Visual*

- ITU-T Recommendation F.700 (2000). *Framework Recommendation for Multimedia Services*

- ITU-T Recommendation F.702 (1996). *Multimedia Conference Services*

- ITU-T Recommendation F.720 (1992). *Video Telephony Services – General*

- ITU-T Recommendation H.261 (1993). *Video Codec for Audiovisual Services at p x 64 kbit/s*

- ITU-T Recommendation H.263 (1998). *Video coding for low bit rate communication*

- Jayant, N., Johnston, J. and Safranek, R. (1993). Proc. IEEE, pp. 1385-1422, October 1993. *Signal Compression based on models of human perception*

- Koenen R. (Editor). ISO/IEC JTC1/SC29/WG11 N3747. (2000). *Overview of the MPEG-4 standard*

- Lee, H. (1997). *Standard Coding for MPEG-1, MPEG-2 and Advanced Coding for MPEG-4* [Online] Available: http://citeseer.nj.nec.com/lee97standard.html

- Li, W. (1997). *ISCAS Tutorial on MPEG-4. Chapter 3.2* [Online] Available: http://bs.hhi.de/mpeg-video/contrib/iscas97/

- Martins, M. and Teixeira, L. (1996). European Conference on Multimedia Applications, Services and Techniques, Proceedings Part II, pp. 615-634, Louvain-la-Neuve, Belgium, May 1996. *Video Compression: The MPEG Standards*

- MPEG AOE Group. (1995). *Proposal Package Description (PPD) – Revision 3 Tokyo meeting, document ISO/IEC/JTC1/SC29/WG11 N998 July 1995*

- Neubauer, T. (2000). *UMTS - Universal Mobile Telecommunications System*. [Online] Available: http://www.nt.tuwien.ac.at/mobile/projects/UMTS/

- Netcom (2000). *Resultat for 2. kvartal 2000*. [Online] Available: http://www.netcom.no/presse/mer.cfm?presse_id=130

- NTT DoCoMo (2000). [Online] Available: http://www.nttdocomo.com

- Packet Video (2000). [Online] Available: http://www.packetvideo.com

- Puri, A. and Eleftheriadis, A. (1997). ACM Mobile Networks and Applications Journal, Special Issue on Mobile Multimedia Communications, August 1997 (invited paper). *MPEG-4: An Object-based Multimedia Coding Standard supporting Mobile Applications.*

- Sikora, T. (1997a). *MPEG-1 and MPEG-2 Digital Video Coding Standards* [Online] Available: http://wwwam.hhi.de/mpeg-video/papers/sikora/mpeg1_2/mpeg1_2.htm

- Sikora, T. (1997b). IEEE transactions on Circuits and Systems for Video Technology, Vol. 7, No. 1, February 1997. *The MPEG-4 Video Standard Verification Model* [Online] Available: http://wwwam.hhi.de/mpeg-video/

- Stuhlmüller, K., Färber, N., Link, M. and Girod, B. (2000). IEEE Journal on Selected Areas in Communications, Special Issue on Error Resilient Image and Video Transmission, pp. 1012-1032, 18 (6), June 2000. *Analysis of Video Transmission over Lossy Channels*

- Telenor (2001). *2001 - Et godt mobile år i Norge*. [Online] Available: http://telenormobil.no/omtelenormobil/presse/nyhetsarkiv2001/010216.jsp

- UMTS Forum Report No. 2 (2000). *The Path towards UMTS –Technologies for the Information Society*. [Online] Available: http://www.umts-forum.org/reports.html

- UMTS Forum Report No. 11 (2000). *Enabling UMTS/Third Generation Services and Applications* [Online] Available: http://www.umts-forum.org/reports.html

- Vinck, B. (Editor) (1998). *Overview of UMTS architecture*. [Online]
  Available: http://www.useca.freeserve.co.uk/

- Zhang, Y. (1997). *ISCAS Tutorial on MPEG-4. Chapter 3.4* [Online]
  Available: http://bs.hhi.de/mpeg-video/contrib/iscas97/

- Örbrink, T. (1999). *How to meet the user requirements of room-based video conferencing* [Online]
  Available: http://www.stacken.kth.se/~tobias/Research/papers.html

Acronyms

3G	Third Generation
3GPP	Third Generation Partner Project
3GPP2	Third Generation Partner Project 2
AC	Alternating Current
AL-PDU	Adoption Layer Protocol Data Unit
AMR	Alternating Motion Rate
ASF	Advanced Streaming Format
ATM	Asynchronous Transfer Mode
BER	Bit Error Rate
BSS	Base Station System
CBR	Constant Bit Rate
CIF	Common Intermediate Format
CRC	Cyclic Redundancy Check
CS	Circuit Switched
DC	Direct Current
DCT	Discrete Cosine Transform
EI	Enhancement I-picture
EP	Enhancement P-picture
ETSI	European Telecommunications Standards Institute
FEC	Forward Error Correction
FER	Frame Error Rate
FPS	Frames Per Second
GGSN	Gateway GPRS Support Node
GIF	Graphics Interchange Format
GOB	Group Of Blocks
GPRS	General Packet Radio Service
GSM	Global System for Mobile communications
HBV	High Bit rate Video

HDTV	High Definition TV
HLR	Home Location Register
IDCT	Inverse Discrete Cosine Transform
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISD	Independent Segment Decoding
ISDN	Integrated Services Digital Network
ISO	International Organisation for Standardisation
ITU	International Telecommunication Union
JPEG	Joint Photographic Experts Group
MB	Macro Block
ME	Motion Estimation
MPEG	Motion Pictures Expert Group
MP3	MPEG-1 audio layer 3
MSC	Mobile service Switching Centre
MV	Motion Vector
NTSC	National Television Standards Committee
PAL	Phase Alternation Line
PS	Packet Switched
PSNR	Peak Signal to Noise Ratio
QCIF	Quarter Common Intermediate Format
QoS	Quality of Service
RFC	Request For Comments
RLP	Radio Link Protocol
RMSE	Root Mean Square Error
RPS	Reference Picture Selection
RTP	Real Time Protocol
RTSP	Real Time Streaming Protocol
RVLC	Reversible Variable Length Code
SAD	Sum of Absolute Difference

SGSN	Serving GPRS Support Node
SIF	Source Input Format
SMIL	Synchronised Multimedia Integration Language
SMS	Short Message Service
SNR	Signal to Noise Ratio
SS	Slice Structured
UMTS	Universal Mobile Telecommunications System
UTRA(N)	UMTS Radio Access (Network)
VBR	Variable Bit Rate
VHS	Video Home System
VLBV	Very Low Bit rate Video
VLC	Variable Length Code
VM	Verification Model
VO	Virtual Object
VOL	Virtual Object Layer
VOP	Virtual Object Plane
WAV	Wave

Appendix A Initial thesis definition

The scope of this thesis is to conduct a study on video services and coders for use in mobile applications. First of all, the task is to give an overview over the commonly used video coders (on web and in various conference and broadcasting systems), and compare the most common ones (coding principal, picture quality, BW requirements and real time characteristics).

Second, a study to determine methods to evaluate loss due to encoding, decoding and noise will be performed. Some of these methods will be chosen to test the various video encoders (formats DV and MPEG2 are selected), with relatively high real time requirements, in a simulated environment to study how the respond towards different kinds of “mobile” noise (with a view to UMTS). The noise can be white noise or noise with characteristics from mobile telephone network. Noise from the fixed network (ISDN) will not be an issue because ISDN provides no noise, constant delay and fixed BW. The packet loss caused in the IP network will be considered as a noise factor.

Continuously changing bit-rate will not be evaluated due to added costs in testing equipment. Furthermore, a study on in what degree of error correction can be performed without harming the real time characteristics.

Appendix B Application source code

Simulate noise.c

```
#include <stdio.h>
#include <stdlib.h>
#include <memory.h>
#include <string.h>
#include <malloc.h>
#include <math.h>
#include <time.h>

#define BUFFSIZE 50000000                // maximum size on the buffer
#define BUFF_NOISE 10000                 // maximum size on the buffer
#define KEY 0x0010                       // character ^P

/* ---------------------------------------- initialisations ----------------------------------------*/

unsigned char * buffer;                  // the buffer associated with video streams
unsigned char noise_buffer [BUFF_NOISE];// elements in buffer is rep. by one byte
unsigned char burst_noise [BUFF_NOISE];

char * input_file = "input_file";        // original video sequence
char * output_file;
char * noise_file;

FILE * input, * output, * noise_sim;     // streams associated with the video files
int items_read, items_read_noise;        // no of elements read from video file

int video_header = 38000;                // default for MPEG-4

void sim_bit_error ( char * input_file, char * output_file );
void sim_burst_error ( char * input_file, char * output_file );
void sim_packet_loss ( char * input_file, char * output_file, char * noise_file );
void sim_gauss_noise( char * input_file, char * output_file );
void open_file (char * file, char * mode);
void filter_bit_error ( float BER);
void filter_burst_error ( float BER );
void filter_packet_loss ( float PER, int packet_size );
void filter_gauss_error ( float BER );
```

```
/* ---------------------------------------------- main ----------------------------------------------*/

void main(){

        int sim_select;

        // allocation of memory for the input sequence

        buffer = ( unsigned char * ) calloc ( BUFFSIZE, sizeof ( unsigned char ) );

        // listing of menu selection

        printf ("From the options below, select the kind of\n"
                    "noise to be simulated (Type in the number):\n\n");

        printf ( " 1. Bit Error\n"
                 " 2. Burst Error\n"
                 " 3. Packet loss\n"
                 " 4. Gaussian White Noise\n"
                 "\nSelection: ");

        scanf ("%d", &sim_select);


        switch ( sim_select ){

                case 1:
                        printf ("\nYour selection was simulation of bit error!\n\n");

                        // video output
                        output_file = "output file";

                        // call function for  bit error
                        sim_bit_error ( input_file, output_file );
                        break;

                case 2:
                        printf ("\nYour selection was simulation of burst error!\n\n");

                        // video output
                        output_file = "output file";

                        // call function for burst error
                        sim_burst_error ( input_file, output_file );
                        break;
```

```
                case 3:
                        printf ("\nYour selection was simulation of packet loss!\n\n");

                        // video output
                        output_file = "output file";

                        noise_file = "sim_packet_loss.dat";   // file with series of 0's

                        // call function for packet loss
                        sim_packet_loss( input_file, output_file, noise_file );
                        break;

                case 4:
                        printf ("\nYour selection was simulation of Gaussian Noise!\n\n");

                        // video output
                        output_file = "output file";

                        // call function for  Gaussian Noise
                        sim_gauss_noise ( input_file, output_file );
                        break;

                default:
                        printf ("\nYour selection was not in the list. Try again!\n\n");
                        break;
  } // end switch
}// main

/* ---------------------------------------- sim_bit_error ----------------------------------------*/

void sim_bit_error ( char * input_file, char * output_file ){

        float BER;

        open_file ( input_file, "rb");          // open the input file for binary read
        fclose (input);                         // close input

        printf ("\nSelect BER ( Bit Error Rate)!\n");
        printf ("\nBER: ");
        scanf ("%f", &BER);

        filter_bit_error ( BER );               // call filter
        open_file ( output_file, "wb");         // open the output file for binary write
        fclose ( output );
} // sim_bit_error
```

```
/* ---------------------------------------- sim_burst_error ----------------------------------------*/

void sim_burst_error ( char * input_file, char * output_file ){

        float BER;

        open_file ( input_file, "rb");          // open the input file for binary read
        fclose (input);                         // close input

        printf ("\nSelect BER ( Burst Error Rate)!\n");
        printf ("\nBER: ");
        scanf ("%f", &BER);

        filter_burst_error ( BER );             // call filter
        open_file ( output_file, "wb");         // open the output file for binary write
        fclose ( output );
} // sim_burst_error

/* ---------------------------------------- sim_packet_loss ----------------------------------------*/

void sim_packet_loss ( char * input_file, char * output_file, char * noise_file ){

        float PER;
        int packet_size;

        open_file ( input_file, "rb");          // open the input file for binary read
        fclose (input);                         // close input

        open_file ( noise_file, "rb");          // open the noise file for binary read
        fclose ( noise_sim );                   // close noise_sim

        printf ("\nSelect PER ( Packet Error Rate) and size of the packets! \n");
        printf ("\nPER: ");
        scanf ("%f", &PER);
        printf ("\nPacket size: ");
        scanf ("%d", &packet_size);

        filter_packet_loss ( PER, packet_size ); // call function packet loss

        open_file ( output_file, "wb");         // open the output file for binary write
        fclose ( output );                      // close output

} // end sim_packet_loss
```

```
/* ---------------------------------------- sim_gauss_noise ----------------------------------------*/

void sim_gauss_noise ( char * input_file, char * output_file ){

        float BER;

        open_file ( input_file, "rb");          // open the input file for binary read
        fclose (input);                         // close input

        printf ("\nSelect BER ( Bit Error Rate)!\n");
        printf ("\nBER: ");
        scanf ("%f", &BER);

        filter_gauss_error ( BER );             // call function filter
        open_file ( output_file, "wb");         // open the output file for binary write
        fclose ( output );

} // end sim_gauss_noise

/* ---------------------------------------- open_file ----------------------------------------*/

void open_file (char * file, char * mode){

        printf ("\n\nINSIDE FUNCTION open_file\n\n");

        if ( file == input_file){

                input = fopen (file, mode);     // open input file for binary read

                if (input == NULL){
                        printf ("Error! File '%s' could not be opened!\n", input_file );
                } // end if

                else printf ("File '%s' opened successfully!\n", input_file);

                items_read = fread (buffer,                 // array of elements
                                    sizeof(unsigned char),  // size of each element
                                    BUFFSIZE,               // max no of elements to read
                                    input);                 // input stream ass with video

        } // end if



        else if ( file == output_file){
```

```
            output = fopen (file, mode);    //open output file for binary write

            if (output == NULL){
                printf ( "Error! File '%s' could not be opened!\n", output_file );
            } // end if

            else printf ("File '%s' opened successfully!\n", output_file);

            fwrite (buffer,                 // array of elements
                sizeof(unsigned char),      // size of each element
                items_read,                 // no of elements to write
                output);                    // output stream ass. with video

        } // end else if

        else if ( file == noise_file){

            noise_sim = fopen (file, mode);     // open noise file for binary read

            if (noise_sim == NULL){
                printf ("Error! File '%s' could not be opened!\n", noise_file );
            } // end if

            else printf ("File '%s' opened successfully!\n", noise_file);

            items_read_noise = fread (noise_buffer,     // array of elements
                                sizeof(unsigned char),// size of each element
                                BUFFSIZE_NOISE,// max no of elements to read
                                noise_sim);         // input stream ass with video

        } // end else if

} // end open_file

/* ---------------------------------------- filter functions ---------------------------------------- */

/* ---------------------------------------- filter_bit_error ---------------------------------------- */
/* The function beneath first estimates the number of bit errors that is going to be       */
/* inserted into the video file, with help of the BER and the size of the file. To         */
/* evenly distribute the bit errors, a position is calculated from the BER and the         */
/* last position of a bit error insertion. The byte that that is placed in the position is */
/* "xor'ed" with a KEY so that only one bit in the byte changes. Then the error is         */
/* inserted into the buffer.                                                               */
/* ------------------------------------------------------------------------------------------ */
```

```
void filter_bit_error ( float BER ){

        float no_of_bit_errors;
        int count = 0;
        long int position;

        no_of_bit_errors = items_read * BER;

        while ( count <= no_of_bit_errors){

                position = video_header + count/BER;

                * noise_buffer = buffer [position]^KEY;

                memcpy ( buffer + position, noise_buffer, 1);

                count ++;

        } // end while

} // end filter_bit_error

/* ---------------------------------------- filter_burst_error ------------------------------------ */
/* The function beneath reads one byte from the buffer out of time and xor it with KEY */
/* then the new value is stored in a temporary buffer, burst_noise. Then the burst_noise */
/* buffer is copied into the buffer, buffer. The length of bursts is determined by      */
/* burst_length.                                                                        */
/* ------------------------------------------------------------------------------------------------ */

void filter_burst_error ( float BER ){

        int no_of_burst_errors;
        int count = 0;
        int position =0;
        int i = 0;
        int burst_length = 1370;        // length of the burst in bytes

        no_of_burst_errors = items_read * BER;

        while ( count <= no_of_burst_errors){

                position = video_header + count * items_read/no_of_burst_errors;

                while ( position > items_read){
                        position = ( video_header
                                        + count
```

```
                                *items_read/no_of_burst_errors)
                                - 40101;
                }

                i = 0;

                while ( i<= ( burst_length -1 )){

                        burst_noise [i] = buffer [position + i]^KEY;
                        i ++;
                }

                memcpy ( buffer + position, burst_noise, burst_length);

                count ++;

        } // end while

} // end filter_burst_error

/* ---------------------------------------- filter_packet_loss ------------------------------------ */
/* The function beneath determines first how many packets lost in the transmission by   */
/* use of PER ( Packet Error Rate ). The simulation of packet loss is achieved by "0"     */
/* insertion (series of "0"s) from a file. The "0"s are copied into the buffer indicated by */
/* position.                                                                               */
/* --------------------------------------------------------------------------------------------- */

void filter_packet_loss ( float PER, int packet_size ){

        float lost_bytes, lost_packets;
        int position;
        int       packets_sent;
        int count = 0;

        lost_bytes = items_read * PER;
        lost_packets = lost_bytes/packet_size;
        packets_sent = items_read/packet_size;


        while ( count <= lost_packets ){

                position = video_header + count * packet_size/PER;

                memcpy ( buffer + position, noise_buffer, packet_size );

                count ++;
```

```
        } // end while

} // end filter_packet_loss

/* ---------------------------------------- filter_gauss_error ---------------------------------- */
/* The function beneath is very equal to the filter function simulating bit error.              */
/* However, in this function the bit error are randomly distributed over the video              */
/* sequence. This is done by using the function rand (), which returns a randomly               */
/* value of an integer.                                                                         */
/* -------------------------------------------------------------------------------------------- */

void filter_gauss_error ( float BER ){

        float no_of_gauss_bit_errors;
        int count = 0;
        long int position;
        int gauss_var = 0;

        no_of_gauss_bit_errors = items_read * BER;


        while ( count <= no_of_gauss_bit_errors){


                gauss_var = rand ( ) * 50;
                position = gauss_var + video_header;

                while ( gauss_var >= ( items_read - video_header )){

                        gauss_var = rand( ) /count;
                }



                * noise_buffer = buffer [position]^KEY;

                memcpy ( buffer + position, noise_buffer, 1);

                count ++;

        } // end while
} // filter_gauss_error
```

PSNR calculation.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <stdio.h>
#include <string.h>

#define BUFFSIZE 5000000
#define INF 1.0e+100

/* ---------------------------------------- initialisations ----------------------------------------*/
unsigned char buffer[BUFFSIZE];
unsigned char m_buffer[BUFFSIZE];

FILE * input, * psnr, * values;          // streams associated with the video frames
int items_read;                          // no of elements read from video file


void open_file (int file, int j);
void write_psnr_to_file (int no_of_frames, double PSNR, int j);

/* ---------------------------------------------- main ----------------------------------------------*/
void main (){

        double total_error = 0.0;
        double RMS, PSNR;
        double average_PSNR = 0.0;
        int file_header = 1078;          // size of file header for bitmap pictures
        int count, no_of_frames, total_iterations, iterations, j, k;
        no_of_frames = 299;
        total_iterations = iterations = 1;
        k=0;

        while (iterations > 0){

                j = 0;
                while ( j <= no_of_frames ){

                        open_file ( 1, j );      // open original frame
                        open_file ( 2, j );      // open degraded frame

                        count = items_read;
```

```
                    for(int i = 0; i <= count; i++){
                        total_error = total_error
                                    + ((buffer[i]
                                    – m_buffer[i])
                                    * (buffer[i]
                                    - m_buffer[i]));
                    }// end for

                    RMS = sqrt (total_error/(double) (items_read-file_header));

                    total_error = 0.0;

                    PSNR = 20.0 * log (255.0/RMS)/ log (10.0);

                    write_psnr_to_file (no_of_frames, PSNR, j);

                    if (PSNR > INF){
                        PSNR = 0;
                        k ++;
                    }// end if

                    average_PSNR += PSNR;

                    j ++;
                }// end while
                iterations --;
        } end while
        printf ("\nPSNR average: %f\n",
                 average_PSNR/(total_iterations*no_of_frames-k));

} // end main

/* ----------------------------------------- open file -----------------------------------------*/

void open_file (int file, int j){

        char file_name [60] = "";
        char file_number [4] ="";
        char file_type [5] = ".bmp";                    // 8 bit grey scale bitmap picture

        if ( file == 1){

                strcat ( file_name, "input_frame");     // original frame
                if (j < 10){
                        strcat (file_name, "0");
```

```
            }// end if

            itoa( j, file_number, 10 );
            strcat( file_name, file_number);
            strcat( file_name, file_type);

            input = fopen (file_name, "rb");              // open frame for binary read

            if (input == NULL){
                  printf ("Error! File '%s' could not be opened!\n", file_name );
            }// end if

            items_read = fread (buffer,                   // array of elements
                              sizeof(unsigned char),      // size of each element
                              BUFFSIZE,                   // max no of elements to read
                              input);                     // input stream ass with video
      fclose (input);
      }// end if

      if ( file == 2){

            strcat (file_name, "degraded_frame");         // degraded frame

            if (j < 10){
                  strcat(file_name, "0");
            } // end if

            itoa ( j, file_number, 10 );
            strcat ( file_name, file_number);
            strcat ( file_name, file_type);

            psnr = fopen (file_name, "rb");               // open frame for binary read

            if (psnr == NULL){
                  printf ("Error! File '%s' could not be opened!\n", file_name );
            } // end if

            items_read = fread (m_buffer,                 // array of elements
                              sizeof (unsigned char),     // size of each element
                              BUFFSIZE,                   // max no of elements to read
                              psnr);                      // input stream ass with video
            fclose (psnr);
      } // end if

} // end open_file
```

```
/* ------------------------------------ write_psnr_to_file ----------------------------------------*/

void write_psnr_to_file (int no_of_frames, double PSNR, int j){
        double psnr_values [300];
        psnr_values [j] = PSNR;

        values = fopen ("psnr_values.txt", "a");        // open input video file for appending
        if (values == NULL){
                printf ("Error! File '%s' could not be opened!\n", "psnr_values" );
        } // end if

        fprintf (values, "%f ", psnr_values[j]);

        fclose (values);

} // end write_psnr_to_file
```