TECHNISCHE UNIVERSITÄT WIEN
ESCOLA POLITÈCNICA SUPERIOR DE
CASTELLDEFELS, UPC

Institut für Nachrichtentechnik und Hochfrequenztechnik

Bachelor Thesis

TEST BED DESIGN FOR INTERACTIVE VIDEO CONFERENCE SERVICES

Professor: Markus RUPP
Supervisor: Michal RIES

Author: Elena RECAS DE BUEN

Contents

Abstract

In the last decade, telecommunication industry has been widely developed in area of the multimedia interactive services. Actually, there are still emerging new video-conference and instant messaging applications.

For the provisioning of multimedia interactive services it is essential to provide a required level of customer satisfaction, given by the end-user quality. The video and audio compression improvement of the newest video coding standards H.264/AVC and AAC allows for providing video and audio streaming for low bit and frame rates while preserving the perceptual quality. This is especially suitable for interactive multimedia applications in 3G wireless networks.

The aim of this thesis was to design the "State of the Art" video conferencing environment supporting H.264/AVC and AAC codecs. Moreover, this environment provides opportunity to analyze end user quality at all layer of OSI model.

As a result of this design, we have an open source application, that offers a good quality of image and sound at very low rates (90kbps, 9fps for video) at the same time that reduces the reception delay that now exists in commercial applications.

Resum

En la ultima decada les telecomunicacions han evolucionat molt en el camp de la multimedia i els serveis interactius. A dia d'avui disposem d'una gran varietat d'aplicacions de videoconferencia i missategria instantania.

Durant tots aquests anys la qualitat que s'oferia als usuaris sempre ha sigut objecte d'estudi a fi d'aconseguir donar un bon servei. Amb els avenços fets en els nous standards de video i d'àudio, H.264 i AAC respectivament, s'aconseguit oferir a taxes molt baixes una qualitat subjectiva acceptable per l'usuari. Degut a això, aquests dos codecs són especialment aptes per aplicacions multimedia interactives en xarxes wireless 3G.

L'objectiu d'aquest projecte es, tenint en compte aquesta qualitat oferida en els serveis actuals, disenyar un sistema capaç d'emular un servei de videoconferencia en Temps Real que faci servir H.264 i AAC.

Com a resultat d'aquest disseny, s'ha obtingut una aplicacio, basada en codi obert, que ofereix una gran qualitat d'imatge i de so a taxes molt baixes (90kbps, 9fps per video) alhora que redueix el retard de recepcio que actualment existeix en les applicacions que s'han comercialitzat.

List of Figures

List of Tables

Chapter 1

Introduction

Since first video sequence was received until now, we have been measuring the quality of the video signal in order to improve it and offer a good service. Nowadays, video applications are part of our lives, and that is why VideoCall applications have been object of investigation since years ago.

But current encoding video systems introduce different artifacts that may be avoided. Furthermore coding degradation is introduced by compression artifacts and network degradation is dependant of the network quality (delay and packet loss). Thereby we can see how important is the need of measuring perceived video quality. But when talking about the perceived video quality it is not only important the "quality of the image", so also are the temporal characteristics, and the synchronization with the video sound. We should found the way to measure all these parameters.

The most traditional ways of evaluating quality of digital video are calculation of Mean Square Error (MSE) and peak signal-to-noise ratio (PSNR) between the original video signal and the error prone transmission. However the PSNR alone as a static parameter does not reflect the subjective perceptual quality evaluation as is explained in [1]. For measuring the audio quality we usually use MOS scale, defined in [2]. The aim of this project is to design an interactive videoconference test bed. For further study the impact on interactivity of the different settings applied.

1.1 Interactive Services

In the last two decades, there has been an increasing demand in the use of multimedia applications, which have evolutioned quickly responding to the user's requirements. They are known also as interactive applications. But, what do we understand by *interactive application*? To describe it, we first need to define the term *interactivity*

Nowadays this word is used with two different meanings. In one hand, as a synonym of participation in communicating relations established between people. On the other hand as the relation established between human beings and machines, that is, the method the user uses to communicate with the computer. The essence of interactivity relies on the bidirectional conversation.

In a one-way communication, only the sender sends the message but doesn't receive any answer. In a two-way communication, messages go in both ways and that requires that next messages answer considering the last messages. Then, the conversation reaches a completely interactive level. That involves that sender, and receiver roles are absolutely commutable.

$$Interactivity \equiv \tau \tag{1.1}$$

$$\tau = f(AudioQuality,\ VideoQuality) \tag{1.2}$$

Thus, we could say that an interactive application is that one which can interact, answer, entertain, or illustrate the user, and it can exchange its role of sender with the user. These services are able to hold user's attention. For that it is very important the interaction time between user and application. Skype is one example of this kind of new services; you can hold calls over the network with a minimal waiting time, and a very little delay. Therefore, we can conclude that time is an important parameter to consider when measuring the interactivity of an application.

1.2 Conversational models

To measure the conversational interactivity we should put into consideration the time spent on it. The delay of the messages is very important, so also is the time the people are speaking.

A good way to relate the time with the conversation behaviour and its interactivity is comparing it with a Markov continuous time chain [3]. Four conversational events are defined in [4]: talk spurt, mutual silence, double talk, and interruptions. Considering these four cases we can model a conversation with a Markov chain of four states. The model is represented like this [3]:

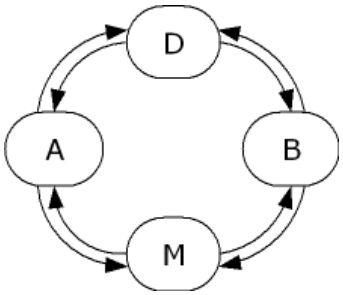

Figure 1.1: Conversational Model.

Figure 1.1 shows a model of four states, which consists in "A" (only A is speaking), "B" (only B is speaking), "D" (both are speaking), and "M" (both are silent). Nevertheless, there are little differences with conversational events explained before [4]. It has been decided that, to simplify our model, "interruptions" event has to be eliminated. Those will be considered in the case that when one is speaking and the other starts to talk, will be like passing through states, for example A-D-A.

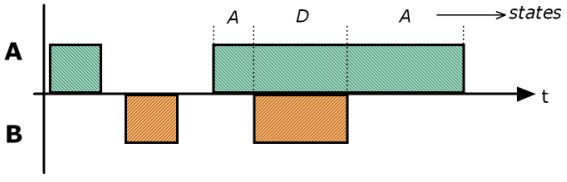

Figure 1.2: Interruption case 1.

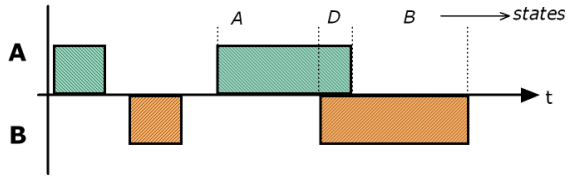

Figure 1.3: Interruption case 2.

Figure 1.2 and Figure 1.3 show two different cases of interruptions, that can be modelled, as said before, as A-D-A, or A-D-B.

Figure 1.1 is the model for a conversation of two people, and we can not use it for a conversational of three, four or more people. For that case it will be easier to omit the states "M" and "D", and phases of mutual silence and/or double speaking will be assigned to one of the speakers [3].

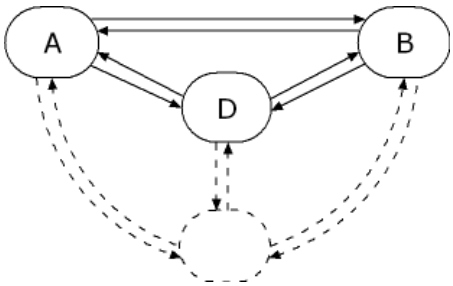

Figure 1.4: Conversational Model (n≥2 States).

We can also model a conversation between two people classificating the states according the "semantic functionality". We will have new states representing for example a pause when somebody is speaking, or a transition between A and B.

Interactivity then, is closed related with the time spent in each state. As explained in [3] interactivity is a parameter in function of time:

$$\tau(t) \tag{1.3}$$

Where t is the sojourn time spent in a state, and τ is the interactivity. Thus, interactivity depends on the time, but both are indirectly proportional. Obviously if someone is speaking during a long time, or nobody speaks, then interactivity will decrease (1.4).

$$lim_{t\to\infty}\tau(t) = 0 \tag{1.4}$$

And in the other way round, if everybody is speaking more or less the same time, and everybody participates in the conversation, then interactivity will increase (1.5).

$$lim_{t\to 0}\tau(t) = \infty \tag{1.5}$$

1.3 Audio and Video Quality

In the last years, several metrics for video quality measurements [5],[6],[7] and for audio quality measurements [8],[9] were investigated.

To select optimal codec parameters for audio and video [10], it is important to consider corresponding quality requirements based on human perception [7]. But the great majority of the publications assume only a single continuous medium, either audio, or video.

In one hand we have that the quality of an audio signal which has been subjected to any kind of processing is determined by the subjective perspective of a person when listening to the processed signal. An objective and technical measure can give only a very poor estimate of the audio perceived quality as long as it does not take into account characteristics of the human perception. The only reliable method to assess the basic audio quality of perceptually coded signals have been subjective listening tests of the encoded sound comparing to the original sound, and using MOS scale [2].

On the other hand, perceived video quality also plays an important role in a lot of applications of image processing. The subjective quality measurement MOS has been used for many years, but it is too slow for practical usage. The aim of the objective metrics is to predict perceived video Quality automatically.

Nevertheless, nowadays multimedia systems are becoming more and more important. In multimedia systems, video and audio modes not only interact, but

there is even a synergy of component media (audio and video) [11]. One of the most important disadvantages is the desynchronization of image and sound. Following [12] we can say that when audio and video are presented in perfect synchrony, intelligibility increases and consequently we can say that interactivity will increase also. By the contrary, if both signals desynchronize, then intelligibility will decrease, and so will interactivity.

Chapter 2

Test Bed

The aim of this work was the design of a test bed capable to emulate a Real Time videoconference service based on H.264 codec. In order to emulate the service it was necessary to find an apropiate application that fulfilled the requirements we wanted to reach, as for example the usage of specific audio and video codecs. For this emulation would be also desirable to have good equipments that satisfy the conditions for running the chosen application.

As an approach of the general idea of the structure of the system, in Figure 2.1 we have which will be the parts of our design. The main idea is to control the stream taken from a webcam by an application capable to send audio and video streams over the network and receive this stream in the remote computer. The most important features that our streaming application had to achieve are:

- Support H.264 and AAC and all their profiles and settings.
- Record the sent and the received stream.
- Send both streams independently.
- Work in asynchronous mode.

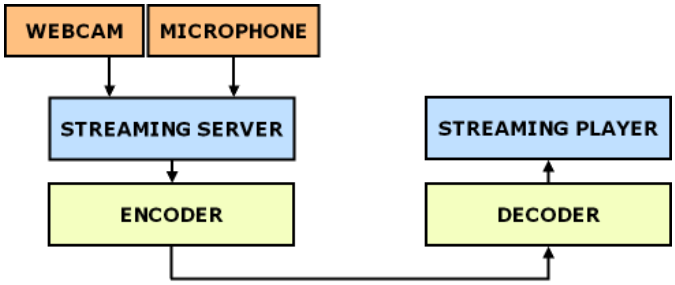

Figure 2.1: General concept of the test bed.

The flow of the streams would be starting in the webcam, and microphone, and passing through our application. Afterwards, both streams would be encoded by the selected codecs, and with the apropiated and chosen settings. When the stream is received in the remote computer, it will be decoded and played in the player.

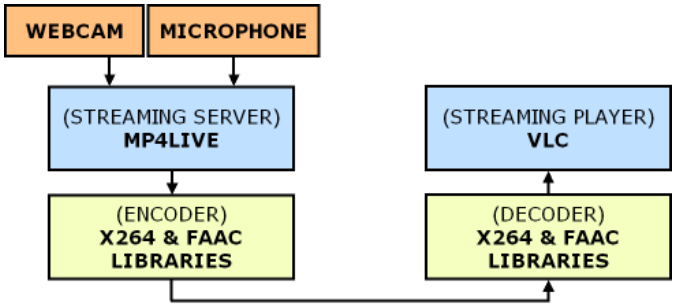

Figure 2.2: Test bed design.

The best application we have found is *MP4Live*, which is an application based on MPEG4IP open source application for audio and video streaming (Figure 2.2). It streams over IP/UDP/RTP using x264 and FAAC libraries. But it is not suitable for playing the stream received, so it was necessary to find also an appropriate player that supported H.264 and AAC. In this case we use VLC, because although it is not made on purpose for Real Time application, it performs really good and supports H.264.

In our case Logitech Quickcam 5000 with UVC video driver is used as video input, although a recorded video can be also used as video input. Depending on

the configuration, one of the supported sound systems (ALSA or OSS) can be used as audio input. All the settings are sent over SSH and SCP protocol.

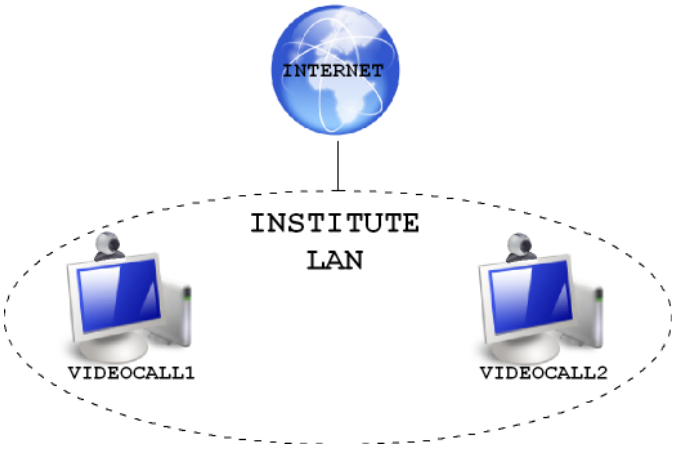

Figure 2.3: Emulation Scenario.

In order to estimate the delay between both computers we set them next to each other and put them in the same LAN. After delay tests, in order to achieve the suitable condition for the videoconference test we divided both computers in different rooms.

2.1 Streaming Server

MP4LIVE is a Linux audio/video capture utility that can capture and encode audio and video in real-time. The results can be written to either an .mp4 file or transmitted onto the network. The audio is encoded with MP3 or AAC, and the video with MPEG-4 Simple Profile.

2.1.1 Audio and Video Streaming

Default start-up of *MP4Live* is to be server, thus, when START button is clicked three actions will be performed:

- sending all audio and video profiles using SSH

- sending .sdp file describing local stream using SCP
- streaming using selected local codecs profiles on local computer and on remote computer

When starting in server mode, the application establishes an SSH connection with the remote computer. For this connection the local and the remote computer need to have a public and a private Key, in order to make the Diffie-Hellman's exchange of keys.

After that it tries to launch streaming in client mode on remote computer and playback the received stream from local computer on remote computer. You should see three windows on each computer:

- application main window
- preview window
- vlc player window with the remote stream playing

Depending on the working mode we are using, the application behaviour on the client will be different:

Working in `--no-remote` mode

To run this mode `--no-remote` switch has to be used when starting. This mode is the original one; application is started and waits for user action. On the remote computer where we want to receive the stream captured by the other computer, we will have to open the VLC player, which is our stream receiver.

Working in `--remote` mode

When -automatic switch is provided the application is started and immediately starts streaming using last selected audio and video profile. So in this case, *MP4Live* application will be opened also in the remote computer and will execute the same steps as the local computer. On the other hand, both computer

will open the VLC player where they will receive the stream captured by the other computer.

In Figure 2.4 all the performed actions in `remote` mode are shown. As a summary we have:

1. Open *MP4Live* in local computer. This action will automatically activate the camera.
2. If PREVIEW checkbox is checked then we will see what the camera is capturing in a little window in our screen.
3. When START button is pressed, an SSH connection will be established, and all the codec settings and sdp files will be sent.
4. On the remote computer, first, VLC will pop up playing the sdp file, received from the other computer.
5. After that, MP4Live will pop up on the remote computer
6. And the preview of the remote camera will appear in a little window
7. Then, all the codec settings of the remote computer will be sent via scp, and VLC will pop up in the local computer.

2.1.2 Interface

The aim of this project is to analyze this application with all the different settings of the different available codecs. In order to do that there are several options that can be set, enable or disable.

First of all, the main interface has to be introduced. As shown in Figure 2.5 it is separated in different parts, and the most important ones are INPUTS and OUTPUTS.

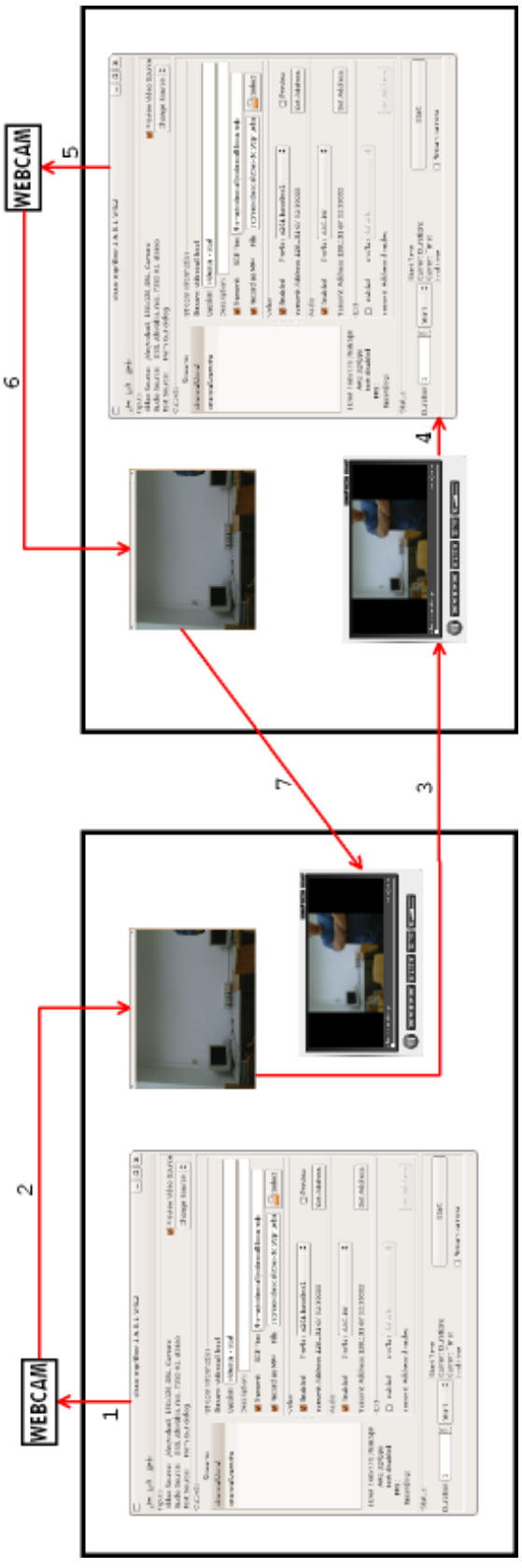

Figure 2.4: Data Flow Diagram.

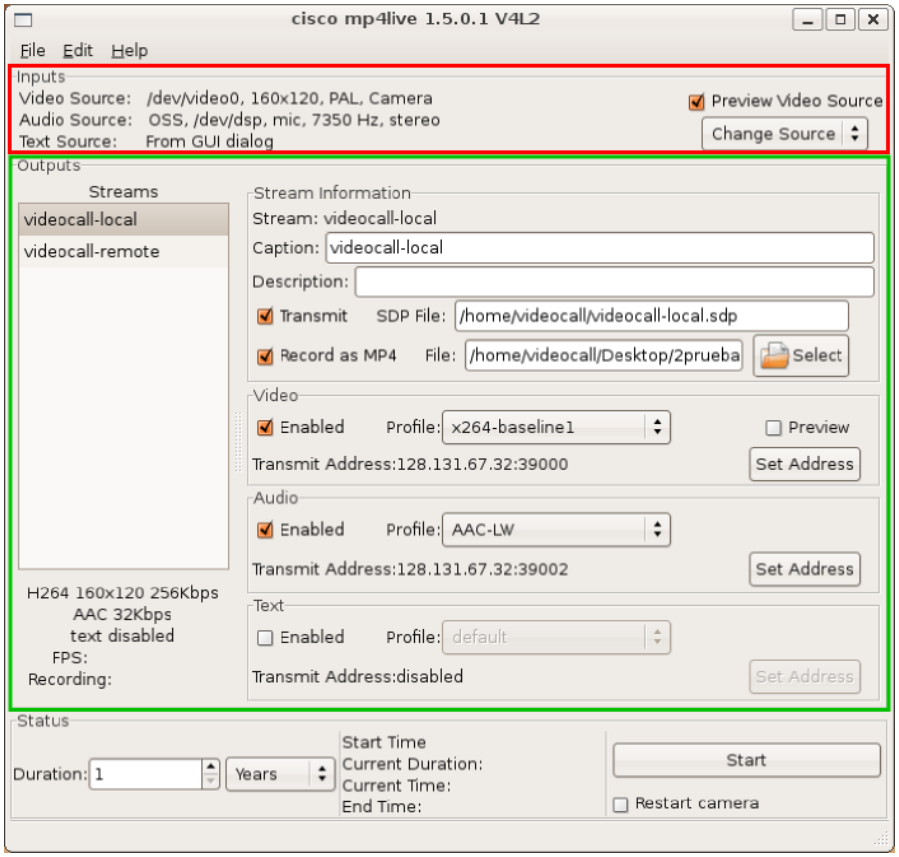

Figure 2.5: Main interface.

- **INPUTS:** Audio Source, Video Source, Text Source.

  In this field you can find the information of the video and audio input. You can also change the source of the video, for example, instead of the camera, you can use a recorded video.

- **OUTPUTS:** This field is the most complex, because is the one that has all the different options for the codecs. Some information of the stream send is shown here, and also which the used profiles.

In the main interface you can also find a little summary of information about the stream you are sending, like the video profile, bit rate, frame rate, audio profile and audio bit rate.

2.2 Settings and Profiles

To start the application you just have to click on the icon and run the application. Before starting the streaming, you can select the different video or audio profiles you want to use.

It is possible to store different audio and video profiles. A profile defines a set of coding tools or algorithms that can be used in generating a conforming bitstream, whereas a level places constraints on certain key parameters of the bitstream.

Different profiles can be selected before starting the stream. To select one just click on profile combo box in the middle of application's window. Different encoders have different settings. To edit profile settings click on Change profile settings menu item in combo box. To add a new one click on "Add profile".

Audio

For audio profile you can set encoder, number of channels (mono or stereo), sample rate and bit rate (Figure 2.6). In advanced settings window you can generally set only audio compatibility mode. Precompiled mode for AAC audio encoding is low delay (LD) for streaming purposes.

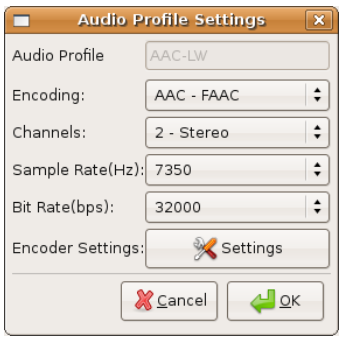

Figure 2.6: Audio Profile.

ENCODING	AAC - FAAC, G.711, MP3
CHANNELS	1 - MONO, 2 - STEREO
SAMPLE RATE (Hz)	7350 - 96000
BIT RATE (bps)	8000 - 320000

Table 2.1: Audio options

Video

For video profile you can set encoder, dimension (picture size), aspect ratio, video filter (post processing the picture, e. g. blend), frame rate and bit rate in video profile basic settings(see Figure 2.7 left box). To edit advanced settings click on Settings button(see Figure 2.7 right box).

ENCODER	MPEG4 - XVID, H264 - x264, H261
RESOLUTION	SQCIF, 14 SIF, QCIF, SIF, CIF, 4SIF, NTSC CCIR601, 4CIF, PAL SQ Pixel
CROP TO ASPECT RATIO	Standard 4:3, Letterbox 2.35, Letterbox 1.85, HDTV 16:9
VIDEO FILTER	none, deinterlace - blend
FRAME RATE (fps)	6 - 25
BIT RATE (kbps)	25 - 4000

Table 2.2: Video options

Further details of the codec used and the reasons of its usage will be explained in Chapter 4. Thus, in Figure 2.7 we have all the different parameters of the codec that we will be able to change. We can change the number of B frames, the size of the macroblock, we can choose also to use Cabac, CBR, or VBV settings. But as by default we use BASELINE profile and VBV settings, it's not possible to use CABAC, because this profile does not support this feature.

2.3 Player

A media player is an application that has the right codecs to support different media formats. Some media players support only audio or video, but for our case we will need a very flexible one, which supports audio and video.

MP4Live is an application that only works for capturing the video from a video source and send the stream over the network with different settings, therefore we will need a receiver application which plays the received stream.

Different players have been tested, like for example Quicktime, which supports also AAC and H.264, but has a high delay and it is not open source. Finally, and after comparing different players, the chosen one has been VLC.

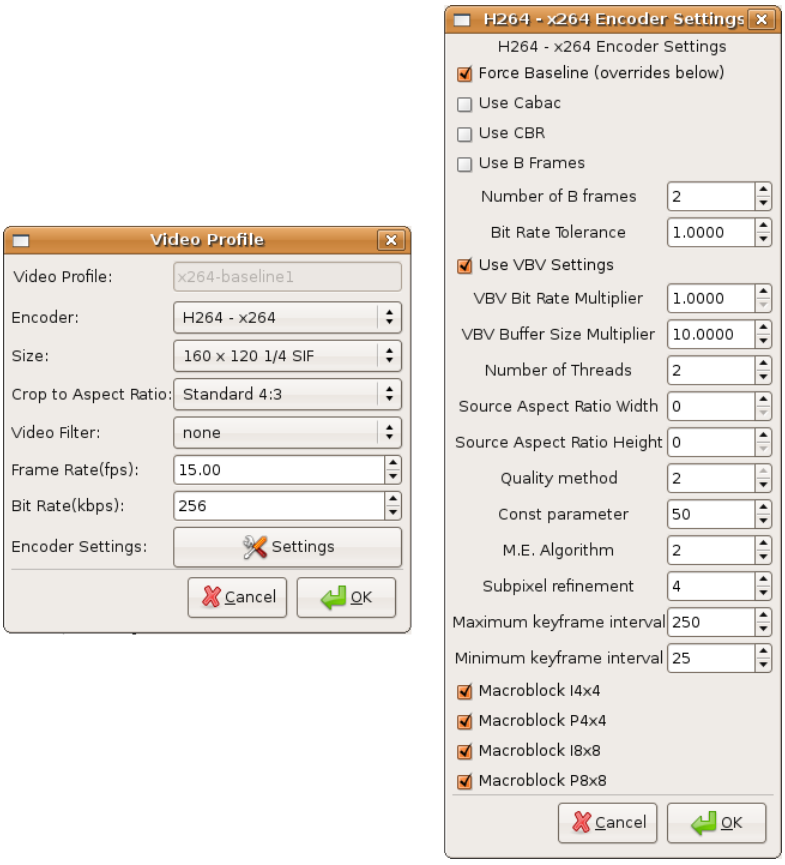

Figure 2.7: Video settings.

2.3.1 VLC

There are several reasons for choosing VLC as our player. First of all, VLC media player is a highly portable multimedia player that works on many different platforms, as Linux, Windows, MAC OS X, BeOS, BSD, Solaris, Familiar Linux, Yopi/linupy, and QNX. It is also able to play many different formats, like:

- MPEG-1, MPEG-2 and MPEG-4 / DivX files from a hard disk, a CD-ROM drive, ...
- DVDs, VCDs, and Audio CDs
- from satellite card (DVB-S),
- Several types of network stream: UDP Unicast, UDP Multicast (MPEG-TS), HTTP, RTP/RTSP, MMS, etc.
- From acquisition or encoding cards (on GNU/Linux and Windows only)

It can also be used as a streaming server or client because it supports different internet protocols for receiving streaming media content, such as HTTP, RTSP or MMS, therefore, that's why it's being used a the receiver of the video streaming of *MP4Live*, because it is very flexible and of course it is open source code.

But the main reason why we are using it, is because it supports H.264, AAC and is the one that gives minimum delay.

We can set the application to be used in remote mode, or in not-remote mode. In `remote` mode, VLC just pops up when the ssh handshake is finished, and then after *MP4Live* in remote computer also pops up.

In `no-remote` mode it is different because you have to force VLC to listen on a port, in order to receive the stream. The process is also very simple.

1. Open VLC
2. File / Open File
3. Browse: in order to receive the stream from the local computer videocall-remote.sdp has to be chosen.

2.4 Used Hardware

The most important factor to consider, and therefore to control, is the delay. Due to it, we should, apart from control the code of the application, also have good computers and hardware which permit minimize it to the maximum. Those computers also have to follow the requirements of the future used application, like being a dual-2GHz Pentium IV machine. These are the main reasons because we have chosen those computers. Their process rate and memory Table 2.3 give us excellent results with the application we are going to use.

As it is a test bed for measuring the quality of the videoconference application, audio and video quality are very important, so we have chosen cameras and microphones with good features (Table 2.4 and Table 2.5). The microphones are omnidirectional and have very high sensitivity. And Webcams have very high resolution.

The characteristics tables of the webcams, microphones and computers are shown bellow:

PC type	Intel(R) Core(TM) 2
CPU Frequency	2'4GHz
Memory	1019224KBytes
Operating System	UBUNTU GNU/Linux
Version OS	feisty fawn (7.04)

Table 2.3: HP Computers Characteristics

High quality VGA sensor with RightLight 2 Technology
Still image capture: up to 1'3Mpixels
Built-in mic with RightSound Technology

Table 2.4: Logitech QuickCam Pro 5000

Microphone model	Philips SBC ME570
Frequency range	50-18 000 Hz
Sensitivity	-45dB
Impedance	600Ohms

Table 2.5: Microphones Characteristics

Chapter 3

Protocols

In order to bring the regular running of our application we have been using different transport and application protocols. Those will be described in this chapter in order to show how they work, and why we are using them.

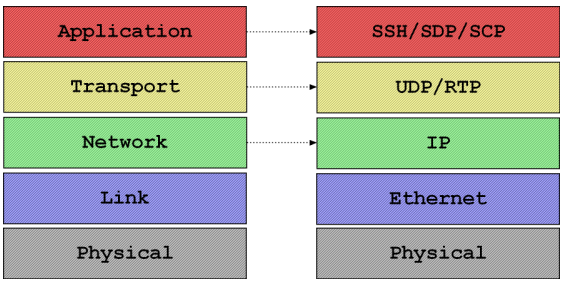

Figure 3.1: Used protocols.

3.1 Transport Layer

The transport layer is the fourth layer of the TCP/IP model and its responsibilities include end-to-end message transfer capabilities independent of the underlying network, along with error control, fragmentation and flow control. End to end message transmission or connecting applications at the transport layer can be categorized as follows:

- **connection-oriented** e.g. TCP

- **connectionless** e.g UDP

The transport layer can be thought of literally as a transport mechanism that connects applications through ports. Since IP provides only a best effort delivery, the transport layer is the first layer to address reliability.

In our case we are using UDP (User Datagram Protocol). It's a connectionless and best effort datagram protocol. It's typically used for applications such as streaming media where on-time arrival is more important than reliability, or for simple query/response applications, where the overhead of setting up a reliable connection is disproportionately large.

3.1.1 RTP - Real-Time Transport Protocol

Nowadays Internet is growing exponentially and it has become the platform of most networking activities. But as a shared datagram network, Internet is not suitable for real-time traffic data. It does not guarantee enough bandwidth, and jitter and delay cannot be controlled, therefore, it is necessary to work with an application-layer protocol, such in this case, RTP, that helps us to deal with these problems.

As said in [13], RTP provides end-to-end delivery services for data with real-time characteristics, such as *MP4Live*. RTP services include *payload type identification*, *sequence numbering*, *timestamping* and *monitoring of the delivered data*. To run real-time applications, RTP normally works over UDP, in order to use its multiplexing and checksum services. However, RTP, may be used with other underlying network or transport protocols.

It has to be taken into consideration that RTP does not guarantee resource reservation neither Quality of Service for RealTime Services. But in parallel there is the RTCP protocol that provides minimal control and identification. Both have been designed to be independent of the underlying transport and network layers.

RTP packet format

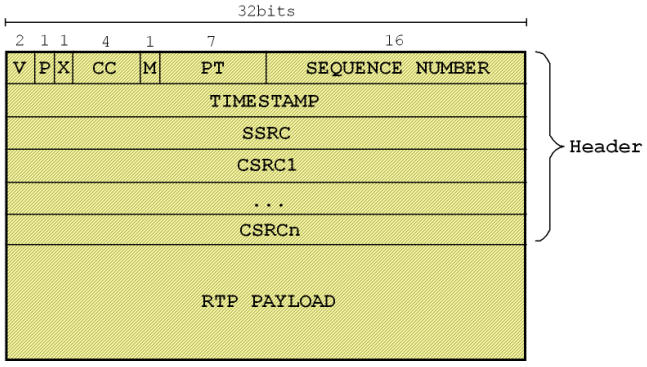

Figure 3.2: RTP packet format.

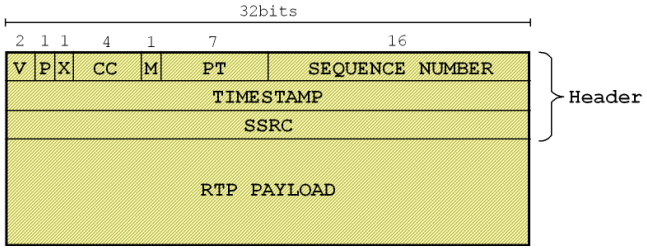

Figure 3.3: Used RTP packet format.

An RTP packet is in that case the UDP payload, and consists in a fixed RTP header, a possibly empty list of contributing sources, and the payload data.

- **Version (V):** This field is the version of RTP that depends of the specification you are following. Nowadays this value is 10, referred to RTP v2.
- **Padding (P):** If this bit is set, then the packet contains one or more additional padding octets at the end of the payload. This may be needed by some encryption algorithms which use fixed block sizes or to fix an RTP packet in a lower data unit.
- **Extension (X):** If this bit is set, then the fixed header is followed by exactly one header extension, with a specific format defined in [13].

- **CSRC Count (CC):** The CSRC count contains the number of CSRC identifiers that follow the fixed header. In our case, as we only have one source, CC will be zero.

- **Marker (M):** The interpretation of this bit is defined by a profile. In video case, it means the end of a frame, because one frame doesn't fit in one single RTP packet and, in audio case, it's always set to 1.

- **Payload Type (PT):** This field identifies the format of the RTP payload and determines its interpretation by the application. A profile specifies a default static mapping of payload type codes to payload formats. In our case, video would be (97) and audio would be (96).

- **Sequence number:** The sequence number increments by one for each RTP data packet sent, and may be used by the receiver to detect packet loss and to restore packet sequence.

- **Timestamp:** This value reflects the sampling instant of the first octet of data. The sampling instant must be derived from a clock that increments monotonically and linearly in time to allow synchronization and jitter calculations. The initial value of the timestamp is random. Several consecutive RTP packets may have equal timestamps if they are generated at once, for example all packets containing parts of the same frame.

- **SSRC:** The SSRC field identifies the synchronization source and it's chosen randomly

- **CSRC list:** (0 to 15 items) The CSRC list identifies the contributing sources for the payload contained in this packet. But, in this case, this list will be empty as shown in Figure 3.3, because we only have one source.

- **RTP payload:** The data transported by RTP in a packet. In our case, audio and video streams, defined in terms of encoding by SDP protocol.

3.2 Application Layer

The application layer is the fifth level of the TCP/IP model. Data is given to the transport layer in an application-specific format and then encapsulated into a transport protocol packet.

Since the TCP/IP stack has no session and presentation layers, the application layer must include the protocols that would act in those layers.

3.2.1 SSH - Secure SHell

SSH is an application layer protocol [14] for secure remote login and exchanging data between a couple of computers connected to an insecure network (Internet). It's typically used to log into a remote machine and execute commands, but also supports tunnelling, and can transfer files using associated protocols such as SFTP or SCP. The major features and guarantees of SSH protocol are:

- Privacy of your data (strong encryption)
- Integrity of communications
- Authentication
- Authorization
- Forwarding or tunnelling to encrypt other TCP-sessions

SSH version 2, that is the one used nowadays, has been separated into modules that consist in three protocols working together:

- **The Transport Layer Protocol [SSH-TRANS]:** provides server authentication, key exchange, encryption, confidentiality, integrity protection, and optionally it provides compression. It also derives a unique session ID that may be used by higher-level protocols.

- **The User Authentication Protocol [SSH-USERAUTH]:** authenticates the client-side user to the server using a suite of different mechanisms. These mechanisms use the session id provided by SSH-TRANS protocol.

- **The Connection Protocol [SSH-CONNECT]:** specifies a mechanism to multiplex multiple streams (channels) of data over the confidential and authenticated transport. It also specifies channels for accessing an interactive shell, for proxy-forwarding various external protocols over the secure transport, and for accessing secure subsystems on the server host.

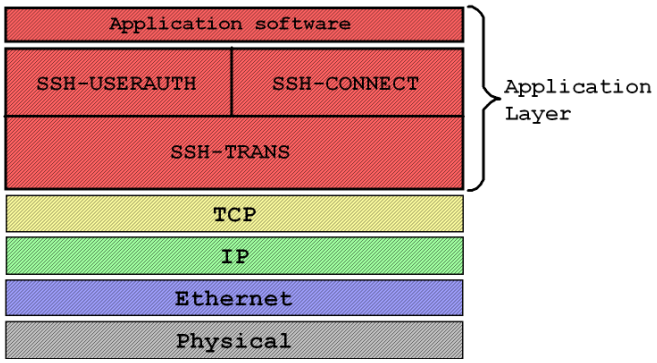

Figure 3.4: SSH Encapsulation.

As it has been said before, SSH offers a number of ways to transfer files between machines. The one used in our application is SCP defined in the next section.

SCP - Secure CoPy

The SCP protocol is the file transfer mechanism used originally by SSH. The protocol itself does not provide authentication and security; it relies on the already defined protocol, SSH, to provide these features.

SCP can interactively request any passwords or passphrases required to make a connection to a remote host, unlike rcp.

The SCP protocol only implements file transfers. It does so by connecting to the host using SSH and there executes an SCP server, that is typically the same program as the SCP client.

For uploads, the client feeds the server with files to be uploaded, optionally including their basic attributes (permissions, timestamps). This is an advantage over the common FTP protocol, which does not have provision for uploads to include the original date/timestamp attribute.

For downloads, the client sends a request for files or directories to be downloaded. When downloading a directory, the server feeds the client with its subdirectories and files. Thus the download is server-driven, which imposes a security risk when connected to a malicious server.

Over SCP we sent the SDP files that will describe the streaming session.

3.2.2 SDP - Session Description Protocol

The Session Description Protocol (SDP), is a format for describing streaming media initialization parameters and it's widely explained in [15].

When initiating a video conference, or other multimedia sessions, some media and encoding details have to be discussed. This protocol provides a standard representation for such information, independently of the transport protocol. Its purpose is to convey information about media streams in multimedia sessions to allow the recipients of a session description to participate in the session.

SDP is a format for session description, describing streaming parameters. It can be used with different transport protocols. It is intended to be general purpose so that it can be used in a wide range of network environments and applications.

SDP started off as a component of the Session Announcement Protocol (SAP), but found other uses in conjunction with RTP, and SIP, and that's why it is being used in this application.

Examples of SDP usage:

- Session Initiation
- Streaming Media
- Email and the World Wide Web

- Multicast Session Announcement

Media and Transport Information

An SDP file includes the following media information:

- The type of media (video, audio, etc.)
- The transport protocol (RTP/UDP/IP, H.320, etc.)
- The format of the media (H.261 video, MPEG video, etc.)

In addition to media format and transport protocol, SDP conveys address and port details. If the session is multicast, then we will have this:

- The multicast group address for media
- The transport port for media

This address and port are the destination address and destination port of the multicast stream, whether being sent, received, or both. For unicast IP sessions, the following are conveyed:

- The remote address for media
- The remote transport port for media

```
v=0
o=- 1183049142279914 1183049142279914 IN IP4 127.0.1.1
s=videocall-remote
i=
e=NONE
b=RR:0
t=0 0
m=video 40000 RTP/AVP 96
c=IN IP4 128.131.67.33
b=AS:500
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=00000d;
        sprop-parameter-sets=Z0LADZp0FidCAAADAAQAAAMAfR4oVUA=,
        aM48gA==; packetization-mode=1
m=audio 40002 RTP/AVP 97
c=IN IP4 128.131.67.33
b=AS:64
a=rtpmap:97 MP4A-LATM/8000
a=fmtp:97 profile-level-id=15;object=2;
        cpresent=0; config=40002b103fc0
```

Figure 3.5: Example of an SDP file.

In Figure 3.5 we can see all the different parameters sent during the initiation of the session in *MP4Live*

Chapter 4

Codecs

For our test bed we had some codec requirements. Chosen codecs should be standardized, have good performance over the network and good "audio and video quality". For audio AAC seemed to be the most apropiated one because is the one that nowadays is being used on Internet by multimedia applications and it is demonstrated that has better performance than other codecs. For video the newest video coding standard H.264 will be the right one because it allows providing video streaming for low bit and frame rates in acceptable quality.

4.1 Audio Codec

AAC

AAC is a combination of the best work from the world's leading audio coding laboratories. Fraunhofer, Dolby, Sony and AT&T were primary collaborators that offered components for AAC. The result was the great quality at 64kbps per mono channel. That differs from the original with a great quality, no trespassing the threshold below the "perceptible but not annoying" item in controlled listening tests.

AAC designers chose to use a new modular approach for the project, with components being plugged-in to a general framework in order to match specific

application requirements and the always present performance/complexity trade-offs.

The basic layout of AAC encoder is depicted in Figure 4.1

This modular thinking has the advantage that there is the possibility to combine different components from different developers taking the best pieces from each one. AAC was built on a similar structure to Layer 3, and thus retains most of its features. When compared side-by-side, AAC proves itself worthy of replacing MP3 as the new audio standard. AAC benefits from some important new additions to the coding toolkit:

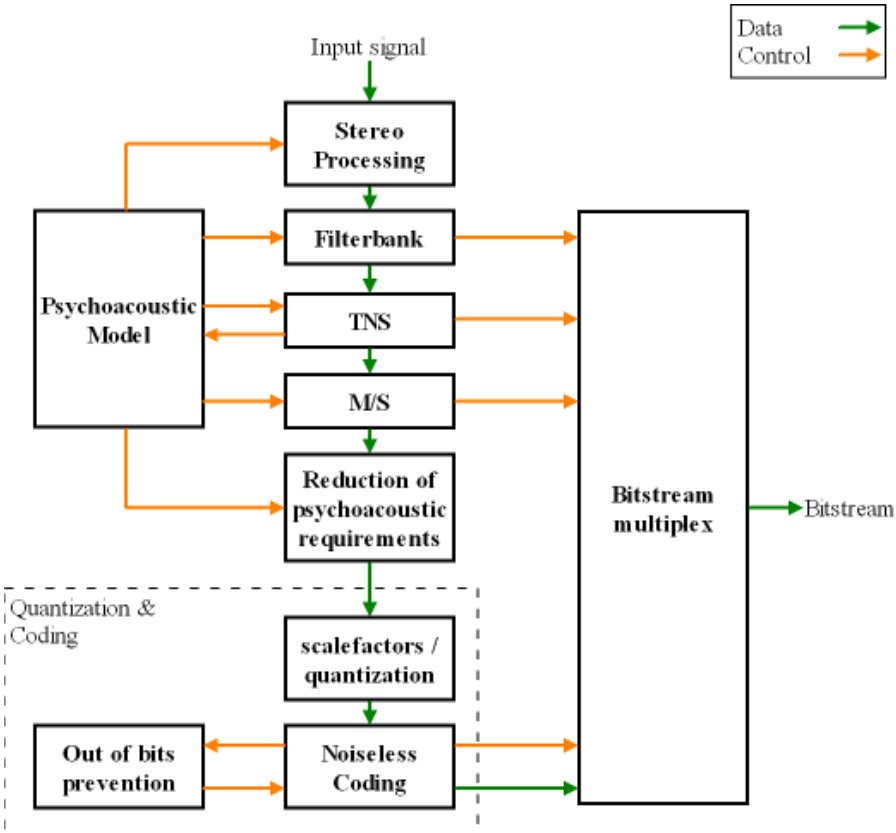

Figure 4.1: AAC Encoder Block Diagram.

- An improved filter bank with a frequency resolution of 2048 spectral components, nearly four times than for Layer 3.
- Temporal Noise Shaping, a new and powerful element that minimizes the

effect of temporal spread. This benefits voice signals, in particular.

- A Prediction module guides the quantizer to very effective coding when there is a noticeable signal pattern, like high tonality.
- Perceptual Noise Shaping allows a finer control of quantization resolution, so bits can be used more efficiently.
- Improved compression provides higher-quality results with smaller file sizes
- Support for multichannel audio, providing up to 48 full frequency channels
- Higher resolution audio, yielding sampling rates up to 96 kHz
- Improved decoding efficiency, requiring less processing power for decode

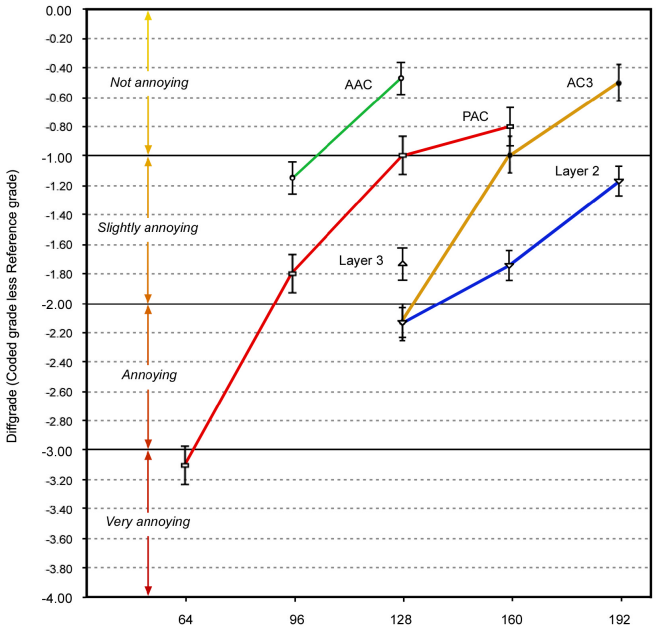

Figure 4.2: Comparison of overall quality between different codecs [16].

After different studies done along the years, it's been concluded that AAC achieves the ITU "indistinguishable quality" goal. AAC at 128 kbps/stereo measured higher than any of the codecs tested. Looking at the graphic from [16] depicted in Figure 4.2 we can see that AAC performs perfectly, in the range of

96kbps and 128kbps, in the layer of "not annoying". Anyway, it's known that it also performs excellent in lower bit rates.

The result of all this is that the researchers succeeded: AAC provides performance superior to any known codec at bitrates greater than 64 kbps and excellent performance relative to the alternatives at bitrates reaching as low as 16 kbps.

Because of its exceptional performance and quality, Advanced Audio Coding (AAC) is at the core of the MPEG-4, 3GPP and 3GPP2 specifications and is the audio codec of choice for Internet, wireless and digital broadcast arenas. It provides audio encoding that compresses much more efficiently than older formats; therefore, that means that sounds better, downloads faster and takes less storage space or network bandwidth. Nowadays is part of the 3GPP standard [3GPP TS 26.403 Release 7 version 7.0.0 SP-32][17], document that describes the AAC encoder part of the Enhanced aacPlus general audio codec.

4.2 Video Codec

H.264

For this application and the consequent tests we are going to use H.264 baseline profile [3GPP TS 26.234 Release 6 version 6.1.0 SP-25]. This codec delivers stunning quality at remarkably low data rates.

H.264 uses the latest innovations in video compression technology to provide incredible video quality from the smallest amount of video data. This means you see crisp, clear video in much smaller files, saving you bandwidth and storage costs over previous generations of video codecs. H.264 delivers the same quality as MPEG-2 at a third to half the data rate and up to four times the frame size of MPEG-4 Part 2 at the same data rate [18].

H.264 achieves the best-ever compression efficiency for a broad range of applications, such as broadcast, DVD, video conferencing, video-on-demand, streaming and multimedia messaging. And true to its advanced design, H.264 delivers ex-

cellent quality across a wide operating range, from 3G to HD and everything in between.

The most important problem when finding an apropiate codec for this application, was the delay. We need one codec that had the minimum delay comparing with the others. Right now, the state-of-the-art of H.264, tells that it is suitable to decrease it. This was then a good reason to choose it, and therefore, it has been already ratified as part of the MPEG-4 standard - MPEG-4 Part 10 - and the ITU-T's latest video-conferencing standard. H.264 [20]is now mandatory for the HD-DVD and Blu-ray specifications (the two formats for high-definition DVDs) and ratified in the latest versions of the DVB (Digital Video Broadcasters) and 3GPP (3rd Generation Partnership Project) standards. Numerous broadcast, cable, videoconferencing and consumer electronics companies consider H.264 the video codec of choice for their new products and services.

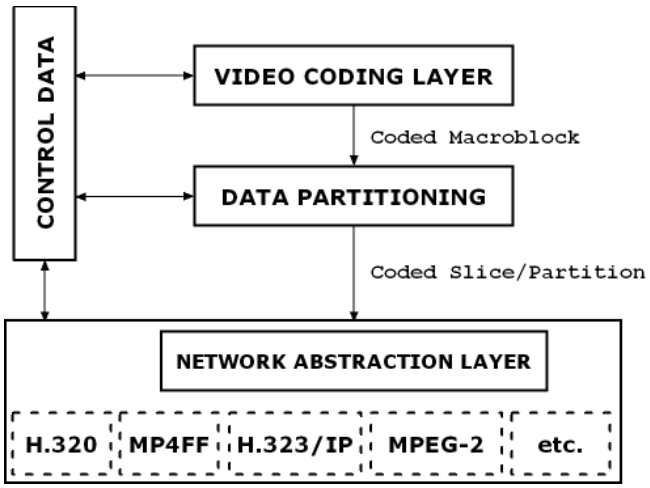

Figure 4.3: Structure of H.264/AVC video encoder.

Features

- **Multipicture inter picture prediction:** using previously encoded frames as references in a much more flexible way than in past standards. This feature gives improvements in bit rate and quality.

- **Variable block-size motion compensation:** it can use instead of the

normal size 16x16, a 4x4pixels block and thus, complicated regions can be coded in a better way.

- **Entropy coding**
- **Flexible Macroblock Ordering:** which is a technique for reconstructing the ordering of the representation of the macroblocks in frames.
- **Redundant Slices:** they are used in order to make the codec robust in front of losses or errors.
- **Switching slices:** feature that allow an encoder to direct a decoder to jump into an ongoing video stream for such purposes as video streaming bit rate.

Profiles

Profiles and levels specify conformance points, which are designed to facilitate interoperability between various applications of the standard with similar functional requirements. A profile defines a set of coding tools or algorithms used in generating a conforming bitstream [18].

All decoders conforming to a specific profile must support all features in that profile. By the contrary, encoders are not required to make use of any particular set of features supported in a profile but have to provide conforming bitstreams. In H.264/AVC, three profiles are defined, which are the Baseline, Main and Extended Profile.

- **Baseline Profile.** Intended for low-complexity applications such as video conferencing and mobile multimedia with low delay. It has a 1'5 times better estimated improved efficiency over MPEG-2
- **Main Profile.** Intended for the majority of general uses, such as Internet, mobile multimedia, and stored content.
- **Extended Profile.** Intended for streaming applications, where stream switching technologies can be beneficial.

The Baseline Profile, which is the one that is going to be used because it is suitable for video conferencing and mobile multimedia applications, supports all features in H.264/AVC except the following two feature sets:

- **SET 1:** slices, weighted prediction, CABAC, field coding, and picture or macroblock adaptive switching between frame and field coding.
- **SET 2:** SP (Switching I)/SI (Switching P) slices, and slice data partitioning.

Set 1 is supported by the Main profile. However, the Main profile does not support the FMO, ASO, and redundant pictures features which are supported by the Baseline Profile. The Extended Profile supports all features of the Baseline profile, and both sets of features, except for CABAC.

In H.264/AVC there are 15 levels defined, specifying upper limits for the picture size, decoder-processing rate, size of the multipicture buffers, video bit rate, and video buffer size. Main characteristics of some of these 15 profiles are:

- **High Profile.** The primary profile for broadcast and disc storage.
- **High 10 Profile.** This profile builds on top of the High Profile adding support for up to 10 bits per sample.
- **High 4:2:2 Profile.** This profile builds on top of the High 10 Profile adding support for the 4:2:2 chroma sampling format.
- **High 4:4:4 Predictive Profile.** This profile builds on top of the High 4:2:2 Profile supporting up to 4:4:4 chroma sampling.

Benefits

- **4x4 integer transform.** H.264 is designed to operate on much smaller blocks of pixels than other common codecs. Thanks to this feature H.264 is able to mitigate some video artifacts, as for example blocking, smearing, and ringing artifacts. So H.264 video is crystal clear even in areas of fine detail.

- **Increased precision in motion estimation.** With this, H.264 simplifies redundant data across a series of frames. By expressing information to 1/4-pixel resolution, H.264 represents fast and slow moving scenes more precisely. Therefore objects in motion during decoding are more precisely reconstructed, providing a better representation of the original video.

- **Flexible block sizes in motion estimation.** Traditional codecs commonly process frames at the macroblock level, but H.264 can process on segments within a macroblock, ranging in size from 16x16 to as small as 4x4, which helps to code complex motion in areas of high detail.

- **Intraframe prediction.** H.264 is able to gain much of its efficiency by simplifying redundant data not only across a series of frames, but also within a single frame, a technique called intraframe prediction. The H.264 encoder uses intraframe prediction with more ways to reference neighbouring pixels, so it compresses details and gradients better than previous codecs. Intraframe prediction is especially beneficial in highmotion areas, which are traditionally difficult to encode.

x264 Software encoder

The result of the effort done by ITU-T's Video Coding Experts Group and ISO/IEC's Moving Pictures Experts Group reached the standardization of H.264/MPEG-4 AVC in 2003. Like previous standards, H.264/AVC specifies only a decoder, which allows to improve the encoder design. Since its standardization, there has been range of H.264 encoders developed by individuals and organizations.

Another H.264 open source encoder is the x264. Its development started in 2004 by Videolan Developers, and since then it has been widely used in different software like ffdshow, ffmpeg and MEncoder.

The main reason for what we are using x264 is because it is a free software library for encoding H.264/MPEG-4 AVC video streams and it is released under the terms of GNU General Public License [21].

In [19] different codecs are compared for different scenarios. There is an important section in this report, analyzing the behaviour for videoconference encoding. Chosen sequences have relatively simple motion and small resolution. The following codecs were considered: DivX 6.2.1, MainConcept, Intel H.264, VSS, x264.

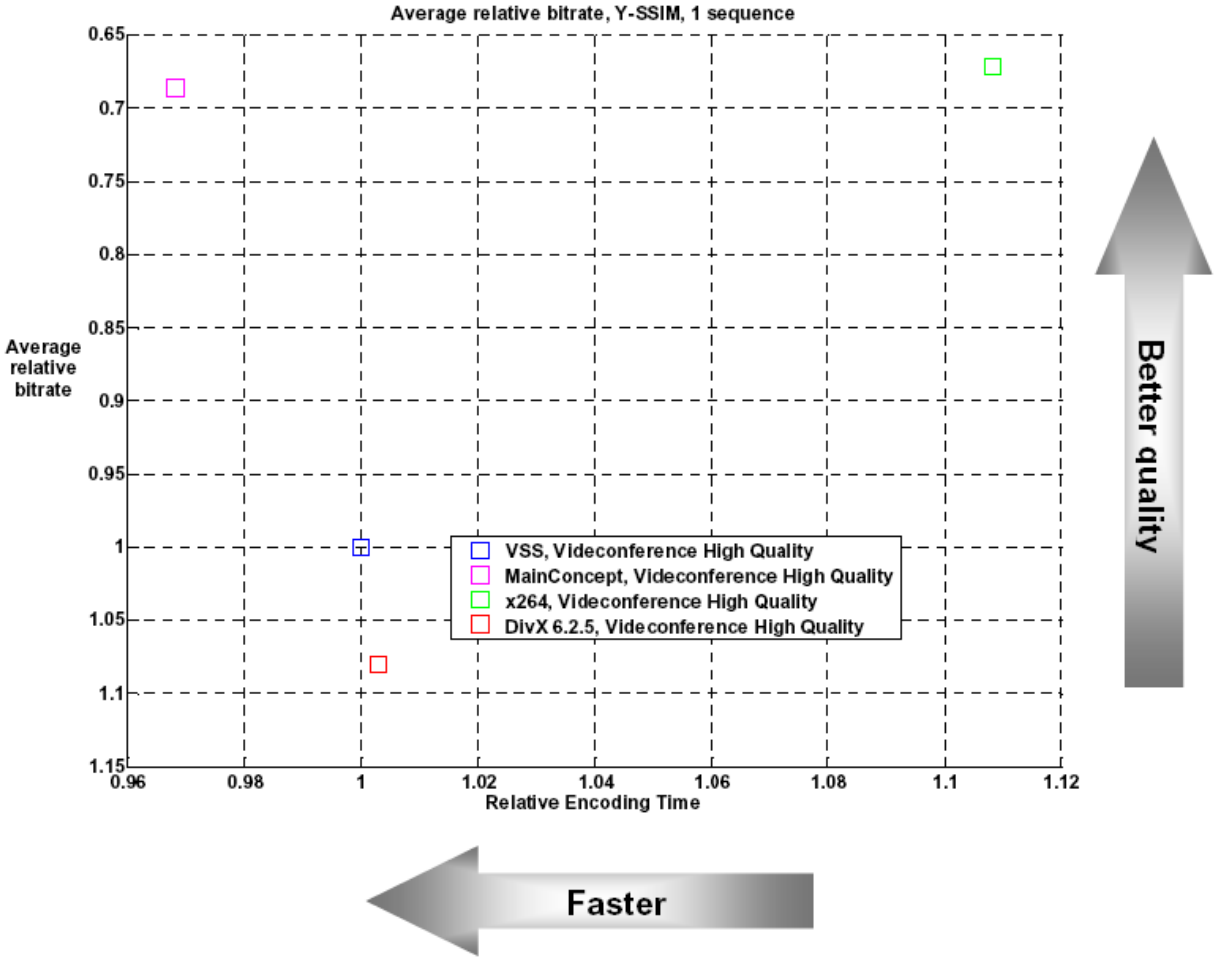

Figure 4.4: Relative bitrateRelative time. "High Quality" preset [19].

Figure 4.4 illustrates how x264 sacrifices the encoding time in order to improve the quality almost a 30%. Therefore, x264 codec shows better quality at the expense of 11% speed degradation comparing it with MainConcept codec. As our application has been design to study the interactivity, but also having into account the video quality, it was important for us to choose a codec which gave a good

quality image so does x264.

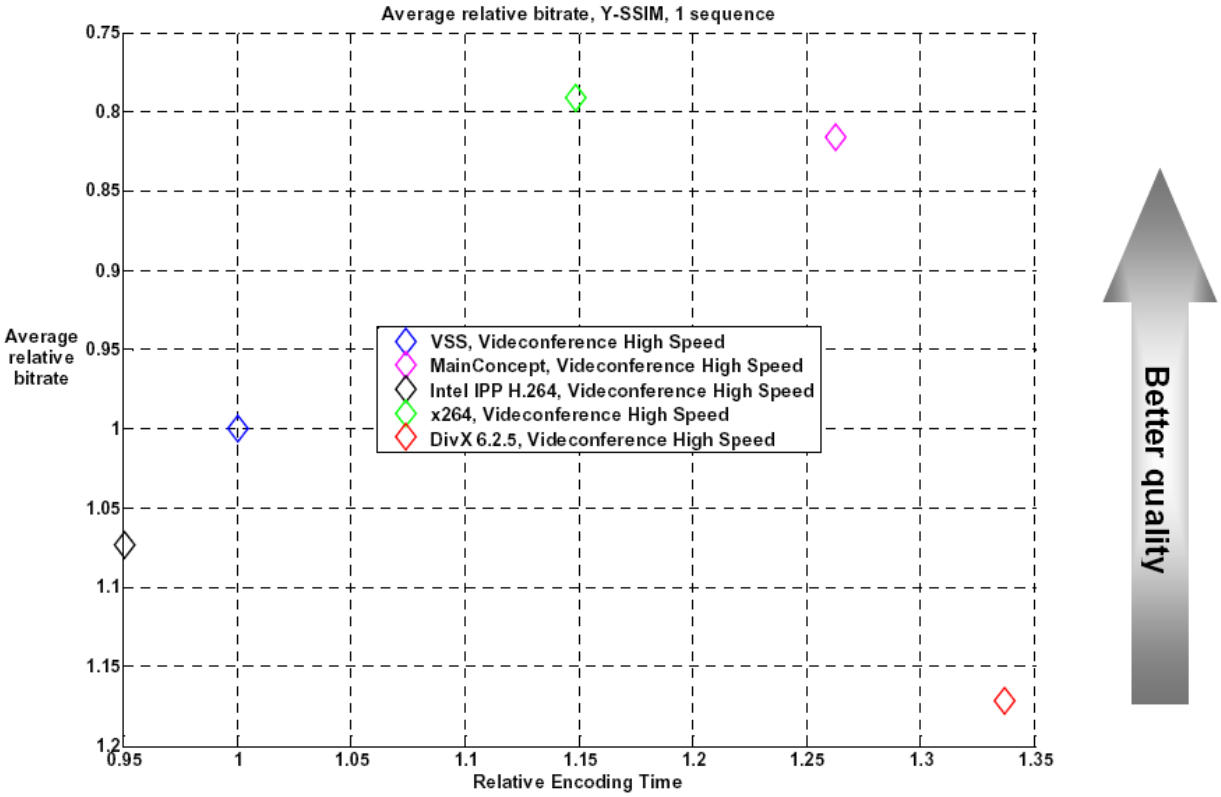

Figure 4.5: Relative bitrateRelative time. "High Speed" preset [19].

In Figure 4.5, "High speed" preset has been used. We can appreciate that in this case x264 codec provides higher quality, almost a 20% more than the MainConcept codec while maintaining higher compression speed.

Using "High Quality" preset and comparing with all the other codecs, x264 provides a good improvement in video quality (30%), which is one of the important features we are looking for in our application, and it is only an 11% under the MainConcept codec when speaking about encoding time, which is not a very high value. On the other hand, when using "High speed" preset, x264 is 20% better quality than the other codecs. Therefore, following [19] and looking at the results, we can conclude, that the most suitable codec for us would be x264 as we already had proposed.

Chapter 5

Performance Evaluation and Conclusions

The videoconference test bed introduces delay in the transmission of the video conference call. Apart from that there are other different factors which increase it. In Figure 5.1 we have a simple diagram of all the delays involved in our case.

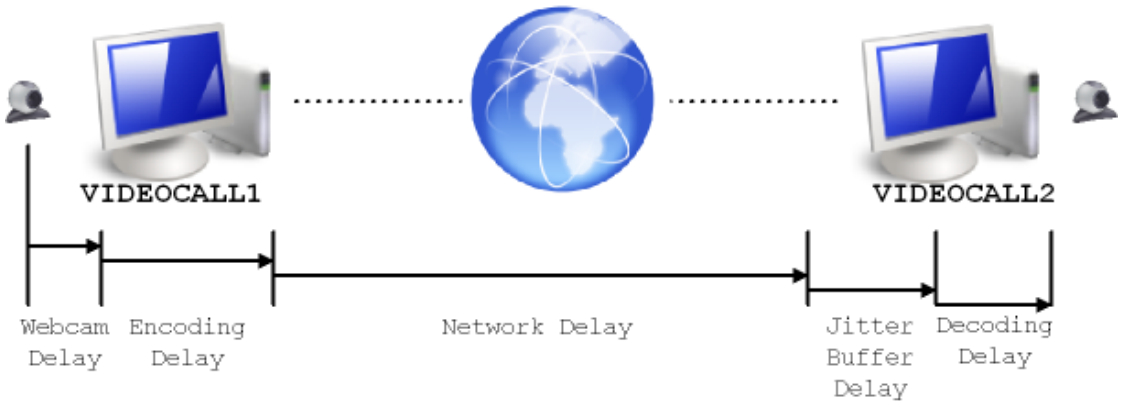

Figure 5.1: One-way-delay diagram.

As we can see there are four important delays to have into consideration at the time of measuring the one-way-delay. The first one will be the webcam delay, that compared with the other ones, is very small. After that we will have the delay of the encoder on the sender, and the delay of the decoder on the receiver.

In the receiver we will have a delay that unfortunately cannot be reduced by the moment, that is the jitter buffer delay. And finally we have the network delay, due to transmssion delay, switch delay, packet loss, etc.

Following the ITU-T Recommendation [22], one-way delay or latency should be around 400ms, and that is what we want to reach with this application. In order to check it, we have done different tests with all the normally used frame rates, bit rates and resolutions and we have obtained the results showed in the tables bellow

- **FRAME RATES:** 6fps, 12fps, 24fps
- **BIT RATES:** 128kbps, 192kbps, 256kbps
- **RESSOLUTIONS:** QCIF, CIF, SIF, 4SIF

Results:

- QCIF

	6fps	**12fps**	**24fps**
128kbps	1,118s	0,477s	0,429s
192kbps	2,197s	0,529s	0,442s
256kbps	1,418s	0,487s	0,682s

Table 5.1: QCIF Results

- CIF

	6fps	**12fps**	**24fps**
128kbps	1,456s	0,548s	0,475s
192kbps	1,105s	0,491s	0,448s
256kbps	0,788s	0,498s	0,463s

Table 5.2: CIF Results

- SIF

	6fps	**12fps**	**24fps**
128kbps	3,382s	0,558s	0,438s
192kbps	1,895s	0,509s	0,507s
256kbps	3,496s	0,563s	0,427s

Table 5.3: SIF Results

- 4SIF

	6fps	**12fps**	**24fps**
128kbps	1,177s	0,406s	0,464s
192kbps	0,806s	0,576s	0,482s
256kbps	1,222s	0,514s	0,486s

Table 5.4: 4SIF Results

It is noticeable that in all resollutions, at 6fps the delay increases considerably. This is because, as the frame rate is lower, the player needs more buffering for synchronizing the video stream, with the audio stream, and thus, play it properly. By the contrary, delays at 12fps and 24 fps are more or less the same, around the 500ms.

Settings

After different tests we have chosen minimal, and maximal settings, speaking about quality.

	1 (minimum)	**2**	**3 (maximum)**
VIDEO	90kbps and 8fps	128kbps and 12fps	256kbps and 24fps
AUDIO	8kbps	16kbps	32kbps

Table 5.5: Test Settings

Conclusions

We have developed a videconference system for the study of interactivity which fulfill our criteria and requirements, such as the usage of specific codecs, low transmision delay, and high video quality, which involved the usage of specific software and consequently specific hardware.

Used codecs for audio and video were AAC and H.264 baseline profile respectively. Both codecs follow the latest recommendations and 3GPP standards which was one of the most important requirements.

After several tests and code changes, we have achieved a mean one-way-delay value of 500 miliseconds with frame rates higher than 8fps, and comparing with other videoconference systems, we can conclude that our test bed, gives better results in terms of delay and quality.

Nevertheless, according to the results of the tests made with different resolutions, frame rates and bit rates it's important to say that is not possible to minimize the delay due to the impossibility to decrease the synchronization buffer of the player used.

APPENDIX A: VLC

VideoLAN is a complete software solution for video streaming and playback, developed by students of the Ecole Centrale Paris and developers from all over the world, under the GNU General Public License (http://www.gnu.org/copyleft/gpl.html) (GPL). VideoLAN is designed to stream MPEG videos on high bandwidth networks. VideoLAN was originally designed for network streaming but VideoLAN's main software, VLC media player has evolved to become a full-featured cross-platform media player. More details about the project can be found on the VideoLAN Web site (http://www.videolan.org/).

VLC Linux Features

- **Input**
  - **Input Media:** *UDP/RTP Unicast*, UDP/RTP Multicast, HTTP/FTP, MMS, File, DVD, VCD, SVCD (Partially), Audio CD, DVB, MPEG encoder, Video acquisition V4L
  - **Input Formats:** MPEG, ID3 tags, AVI, ASF/WMV/WMA, MP4/MOV/3GP, OGG/OGM/Annodex, Matroska (MKV), WAV (including DTS), Raw Audio (DTS, *AAC*, AC3/A52), Raw DV, Flac, FLV (Flash)
- **Video**
  - **Decoders:** MPEG 1-2, DIVX (1/2/3), MPEG_4, DivX 5, XviD, 3ivX

D4, *H.264*, Sorenson 1/3 (QT), DV Cinepak, Theora (alpha 3), H.263/H.263i, MJPEG (A/B), WMV 1/2, WMV 3 / WMV-9 / VC-1

- **Audio**
  - **Decoders:** MPEG Layer 1/2, MP3, AC3 - A/52, DTS, LPCM, *AAC*, Vorbis, WMA 1/2, WMA 3, ADPCM, DV Audio, FLAC, QDM2/QDMC (QuickTime), MACE, Speex

Bibliography

[1] Olivia Nemethova, Michal Ries and Markus Rupp. *"Quality assessment for H.264 coded low-rate and low-resolution video sequences"* Published in the proceedings of CIIT,St. Thomas, US Virgin Islands, November 22-24, 2004.

[2] ITU-T Recommendation P.800.1. *"Mean Opinion Score (MOS) terminology"*, July 2006.

[3] Peter Reichl, Gernot Kubin and Florian Hammer. *"A general temperature metric framework for conversational interactivity"*, Telecommunications Research Center Vienna (ftw.), Signal Processing and Speech Communication Laboratory (TU Graz).

[4] Brady, P.T. *"A statistical analysis of on-off patterns in 16 conversations"*.

[5] ANSI T1.801.03, *"American National Standard for Telecommunications - Digital transport of one-way video signals. Parameters for objective performance assessment"*, American National Standards Institute, 2003.

[6] S. Winkler, Ch. Faller. *"Maximizing audiovisual quality at low bitrates"*, in Proc. Workshop on Video Processing and Quality Metrics for Consumer Electronics, Scottsdale, AZ, 2005.

[7] A.A. Webster, C.T. Jones, M.H. Pinson, S.D. Voran, S. Wolf. *"An objective video quality assessment system based on human perception"*, in Proc. SPIE Human Vision, Processing and digital display, vol. 1913, pp. 15-26, San Jose, CA, 1993.

[8] ITU-T Recommendation P.862. *"Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow band telephone networks and speech codecs"*, 2001.

[9] S. Voran. *"Objective Estimation of Perceived Speech Quality. Part II: Evaluation of the Measuring Normalizing Block Technique"*, in Proc. IEEE Transactions on speech and audio, vol. 7, no. 4, pp. 385-390, 1999.

[10] Michal Ries, Rachele Puglia, Tommaso Tebaldi, Olivia Nemethova, Markus Rupp. *Audiovisual quality estimation for mobile streaming services*, published in the Proceedings of the 2nd International Symposium on Wireless Communication Systems 2005, 5-7 September 2005, Siena, Italy.

[11] S. Tasaka, Y. Ishibashi. *"Mutually Compensatory Property of Multimedia QoS"* IEEE Transactions, Nagoya Institute of Technology, Nagoya, Japan, 2002.

[12] Ken W. Grant and Steven Greenberg. *"Speech intelligibility derived from asynchronous processing of auditory-visual information."*, Submitted to AVSP-2001 (Auditory-Visual Speech Processing) Workshop, 2001.

[13] Audio-Video Transport Working Group, H. Schulzrinne, GMD Fokus, S.Casner, Precept Spftware Inc., R. Frederick, Xerox Palo Alto Research Center, V. Jacobson, Lawrence Berkeley National Laboratory. *"RFC 1889 - RTP: A Transport Protocol for Real-Time Applications"* , January 1996.

[14] T. Ylonen, SSH Communications Security Corp, C. Lonvick Ed., Cysco Systems, Inc. *"RFC 4251 - The Secure Shell (SSH) Protocol Architecture"*, January 2006.

[15] M. Handley, UCL, V. Jacobson, Packet Design, C.Perkins, University of Glasgow. *"RFC 4566 - SDP: Session Description Protocol"*, July 2006.

[16] Steve Church. *"On beer and audio coding"*, Telos Systems.

[17] 3GPP TS 26.403 v7.0.0 *"General audio codec audio processing functions; Enhanced aacPlus general audio codec; Encoder Specification; Advanced Audio Coding (AAC) part 7 (Release 7)"*, June 2006.

[18] Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. *"Overview of the H.264/AVC Video Coding Standard"*, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, N.7, July 2003.

[19] Dmitriy Vatolin. *"MPEG-4 AVC/H.264 Video Codecs Comparison. Short version of Report"*, CS MSU Graphics&Media Lab Video Group, November 2006

[20] ITU-T Recommendation H.264. *"Advanced video coding for generic audiovisual services"*, March 2005.

[21] Loren Merritt. *"x264: A high performance H.264/AVC encoder"*.

[22] ITU-T Recommendation E.800. *"One way transmission time"*, May 2000.

[23] Florian Hammer and Peter Reichl. *"How to measure interactivity in telecommunications"*, Telecommunications Research Center Vienna (ftw.).

[24] ITU-T Recommendation G.109. *"Definition of categories of speech transmission quality"*, September 1999.

[25] ITU-T Recommendation P.800. *"Methods for subjective determination of transmission quality"*, August 1996.

[26] ITU-T Recommendation P10/G.100. *"Vocabulary for performance and quality of service"*, July 2006.

[27] ITU-T Recommendation E.800. *"Terms and definitions related to quality of service and network performance including dependability"*, August 1994.

[28] Florian Hammer. *"Quality aspects of packet-based interactive speech communication"* Wien, June 2006.

[29] Florian Hammer, Peter Reichl and Alexander Raake. *"Elements of interactivity in telephone conversations"*, Telecommunications Research Center Vienna (ftw.) (Austria) and Institute of Communication Acoustics (IKA), Ruhr-University Bochum (Germany).

[30] Zhou Wang, Alan C. Bovik and Ligang Lu. *"Why is image quality assessment so difficult?"*, Lab for Image and Video Engineering, Department of ECE University of Texas at Austin (Austin), and IBM T. J. Watson Research Center Yorktown Heights.

[31] Michal Ries, Catalina Crespi, Olivia Nemethova and Markus Rupp. *"Content based video quality estimation for H.264/AVC video streaming"*, Institute of Communications and Radio-Frequency Engineering Vienna University of Technology.

[32] Audio-Video Transport Group, H.Schulzrinne, GMD Fokus. *"RFC 1890 - RTP Profile for Audio and Video Conferences with Minimal Control"*, January 1996.